NXP Semiconductors
Reference Manual

Document identifier: IMXRT1180RM
Rev. 6, 09/2024

i.MX RT1180 Reference Manual

Contents

Chapter 1
About This Manual

1.1 Audience

The reference manual is intended for the board-level product designers and product software developers. This manual assumes that the reader has a background in computer engineering and/or software engineering and understands the concepts of the digital system design, microprocessor architecture, input/output (I/O) devices, industry standard communication, and device interface protocols.

1.2 Organization

This manual has two main sets of chapters.

- Chapters in the first set contain information that applies to all components on the chip.
- Chapters in the second set are organized into functional groupings that detail particular areas of functionality.
  - Examples of these groupings are clocking, timers, and communication interfaces.
  - Each grouping includes chapters that provide a technical description of individual modules.

1.2.1 Attachments

This manual includes key information in the files attached to it. For example, memory map and I/O details. Use the content in these attachments in conjunction with this manual's content.

NOTE

Select the paperclip icon on the left side of the PDF window to see the list of attachments.

1.3 General Information

The following documentation provides useful background information about the ARM Cortex processor.

For information about the ARM Cortex processor see:

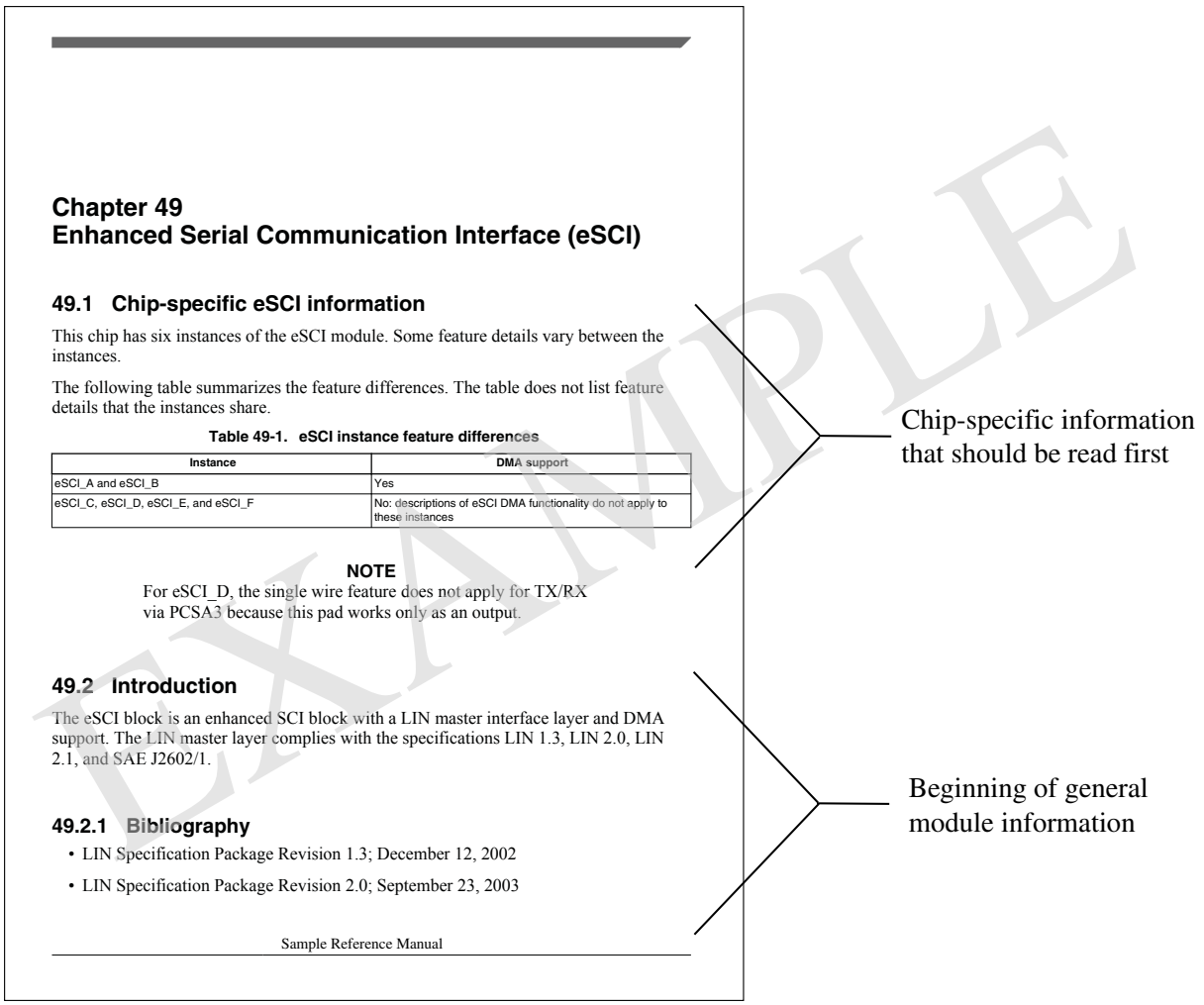
- http://infocenter.arm.com

1.4 Module descriptions

Each module chapter has two main parts:

- The first section, *chip-specific [module name]* information, provides details such as the number of module instances on the chip and connections between that module and the other ones. Read this section *first* because its content is crucial for understanding the information in the other sections of the chapter.
- The subsequent sections provide general information about the module, including its signals, registers, and functional description.

The following figure shows you an example of this demarcation.



Chapter 49
Enhanced Serial Communication Interface (eSCI)

49.1 Chip-specific eSCI information

This chip has six instances of the eSCI module. Some feature details vary between the instances.

The following table summarizes the feature differences. The table does not list feature details that the instances share.

Table 49-1. eSCI instance feature differences

Instance	DMA support
eSCI_A and eSCI_B	Yes
eSCI_C, eSCI_D, eSCI_E, and eSCI_F	No: descriptions of eSCI DMA functionality do not apply to these instances

NOTE
For eSCI_D, the single wire feature does not apply for TX/RX via PCSA3 because this pad works only as an output.

49.2 Introduction

The eSCI block is an enhanced SCI block with a LIN master interface layer and DMA support. The LIN master layer complies with the specifications LIN 1.3, LIN 2.0, LIN 2.1, and SAE J2602/1.

49.2.1 Bibliography

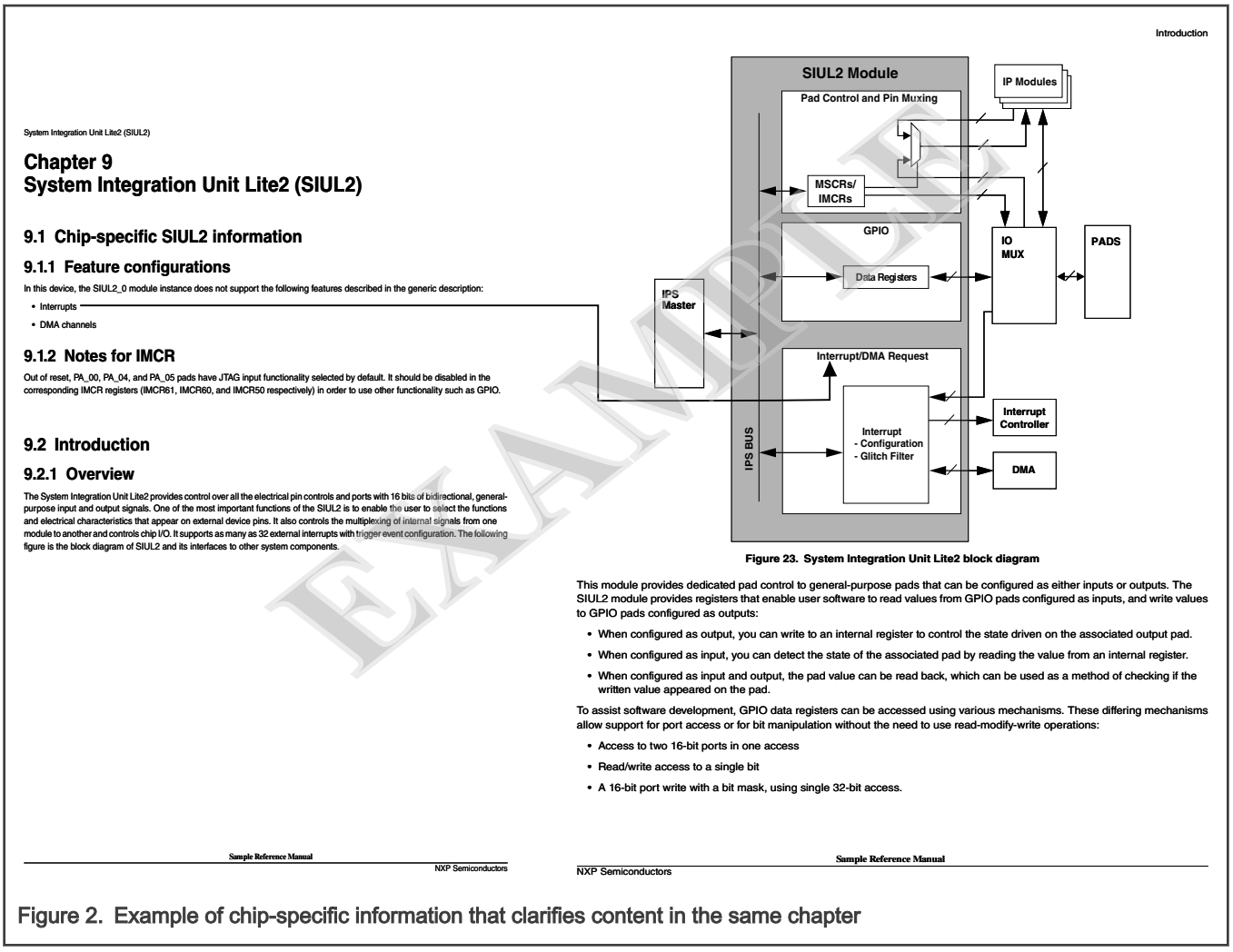
- LIN Specification Package Revision 1.3; December 12, 2002
- LIN Specification Package Revision 2.0; September 23, 2003

Sample Reference Manual

Figure 1. Example of chapter chip-specific information and general module information

1.4.1 Chip-specific information that clarifies content in the same chapter

The following figure shows an example of chip-specific information that clarifies general module information presented later in the chapter. In this case, the chip-specific register reset values supersede the reset values that appear in the register diagram.



Introduction

System Integration Unit Lite2 (SIUL2)

Chapter 9
System Integration Unit Lite2 (SIUL2)

9.1 Chip-specific SIUL2 information

9.1.1 Feature configurations

In this device, the SIUL2_0 module instance does not support the following features described in the generic description:

- Interrupts
- DMA channels

9.1.2 Notes for IMCR

Out of reset, PA_00, PA_04, and PA_05 pads have JTAG input functionality selected by default. It should be disabled in the corresponding IMCR registers (IMCR61, IMCR60, and IMCR50 respectively) in order to use other functionality such as GPIO.

9.2 Introduction

9.2.1 Overview

The System Integration Unit Lite2 provides control over all the electrical pin controls and ports with 16 bits of bidirectional, general-purpose input and output signals. One of the most important functions of the SIUL2 is to enable the user to select the functions and electrical characteristics that appear on external device pins. It also controls the multiplexing of internal signals from one module to another and controls chip I/O. It supports as many as 32 external interrupts with trigger event configuration. The following figure is the block diagram of SIUL2 and its interfaces to other system components.

Figure 23. System Integration Unit Lite2 block diagram

This module provides dedicated pad control to general-purpose pads that can be configured as either inputs or outputs. The SIUL2 module provides registers that enable user software to read values from GPIO pads configured as inputs, and write values to GPIO pads configured as outputs:

- When configured as output, you can write to an internal register to control the state driven on the associated output pad.
- When configured as input, you can detect the state of the associated pad by reading the value from an internal register.
- When configured as input and output, the pad value can be read back, which can be used as a method of checking if the written value appeared on the pad.

To assist software development, GPIO data registers can be accessed using various mechanisms. These differing mechanisms allow support for port access or for bit manipulation without the need to use read-modify-write operations:

- Access to two 16-bit ports in one access
- Read/write access to a single bit
- A 16-bit port write with a bit mask, using single 32-bit access.

Sample Reference Manual

NXP Semiconductors

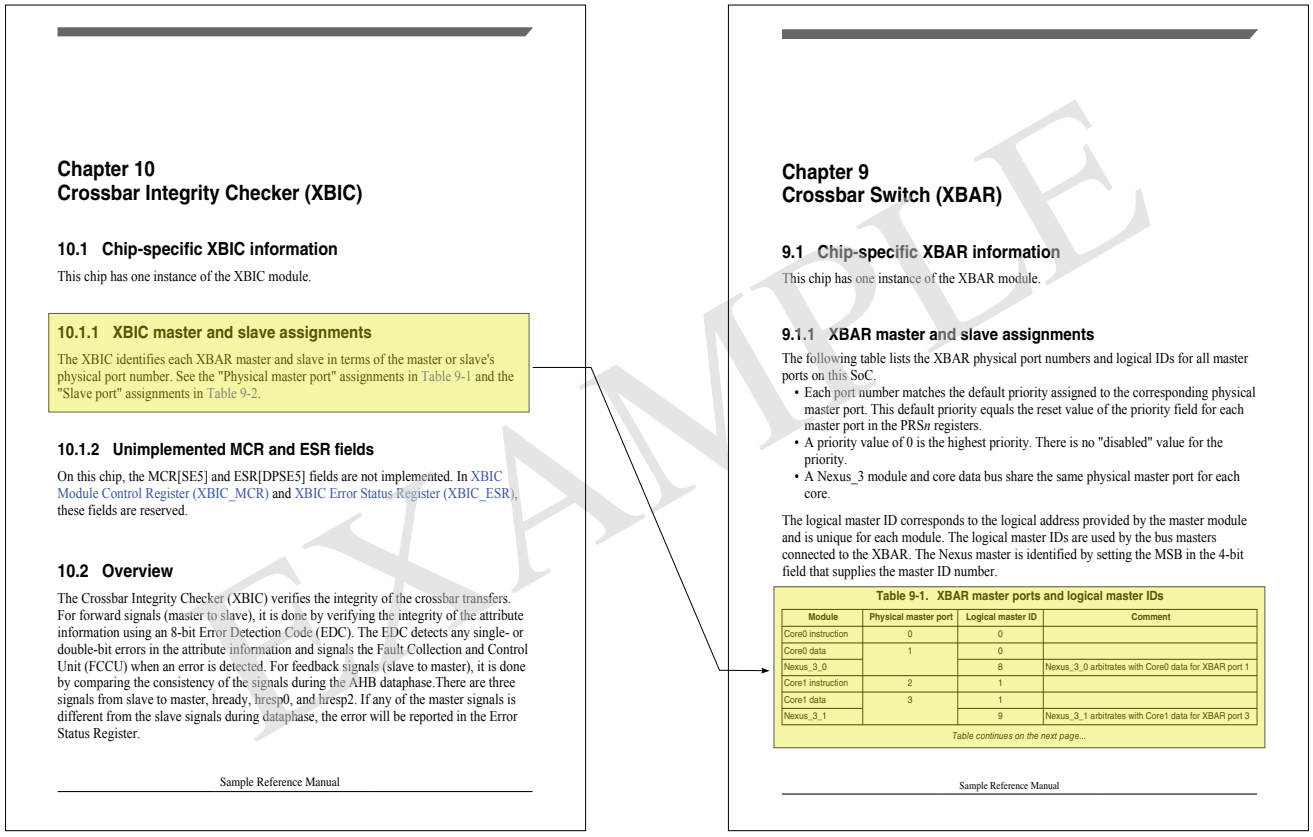
Sample Reference Manual

NXP Semiconductors

Figure 2. Example of chip-specific information that clarifies content in the same chapter

1.4.2 Chip-specific information that refers to a different chapter

Related chip-specific information may be provided in different chapters of the manual. The following figure shows an example of two such connected pieces of information. In this case, read both before you proceed.



Chapter 10
Crossbar Integrity Checker (XBIC)

10.1 Chip-specific XBIC information

This chip has one instance of the XBIC module.

10.1.1 XBIC master and slave assignments

The XBIC identifies each XBAR master and slave in terms of the master or slave's physical port number. See the "Physical master port" assignments in Table 9-1 and the "Slave port" assignments in Table 9-2.

10.1.2 Unimplemented MCR and ESR fields

On this chip, the MCR[SE5] and ESR[DPSE5] fields are not implemented. In XBIC Module Control Register (XBIC_MCR) and XBIC Error Status Register (XBIC_ESR), these fields are reserved.

10.2 Overview

The Crossbar Integrity Checker (XBIC) verifies the integrity of the crossbar transfers. For forward signals (master to slave), it is done by verifying the integrity of the attribute information using an 8-bit Error Detection Code (EDC). The EDC detects any single- or double-bit errors in the attribute information and signals the Fault Collection and Control Unit (FCCU) when an error is detected. For feedback signals (slave to master), it is done by comparing the consistency of the signals during the AHB dataphase.There are three signals from slave to master, hready, hresp0, and hresp2. If any of the master signals is different from the slave signals during dataphase, the error will be reported in the Error Status Register.

Sample Reference Manual

Chapter 9
Crossbar Switch (XBAR)

9.1 Chip-specific XBAR information

This chip has one instance of the XBAR module.

9.1.1 XBAR master and slave assignments

The following table lists the XBAR physical port numbers and logical IDs for all master ports on this SoC.

- Each port number matches the default priority assigned to the corresponding physical master port. This default priority equals the reset value of the priority field for each master port in the PRS*n* registers.
- A priority value of 0 is the highest priority. There is no "disabled" value for the priority.
- A Nexus_3 module and core data bus share the same physical master port for each core.

The logical master ID corresponds to the logical address provided by the master module and is unique for each module. The logical master IDs are used by the bus masters connected to the XBAR. The Nexus master is identified by setting the MSB in the 4-bit field that supplies the master ID number.

Table 9-1. XBAR master ports and logical master IDs

Module	Physical master port	Logical master ID	Comment
Core0 instruction	0	0	
Core0 data	1	0	
Nexus_3_0		8	Nexus_3_0 arbitrates with Core0 data for XBAR port 1
Core1 instruction	2	1	
Core1 data	3	1	
Nexus_3_1		9	Nexus_3_1 arbitrates with Core1 data for XBAR port 3

Table continues on the next page...

Sample Reference Manual

Figure 3. Example of chip-specific information that refers to a different chapter

1.5 Register descriptions

Module chapters present register information in the following:

- Memory maps, which contain:
  - An offset from the module's base address
  - The mnemonic and name of each register
  - The width of each register (in bits)
  - The reset value of each register
- Register figures
- Field-description tables
- Associated text

The following figure shows register figure conventions used throughout the manual.

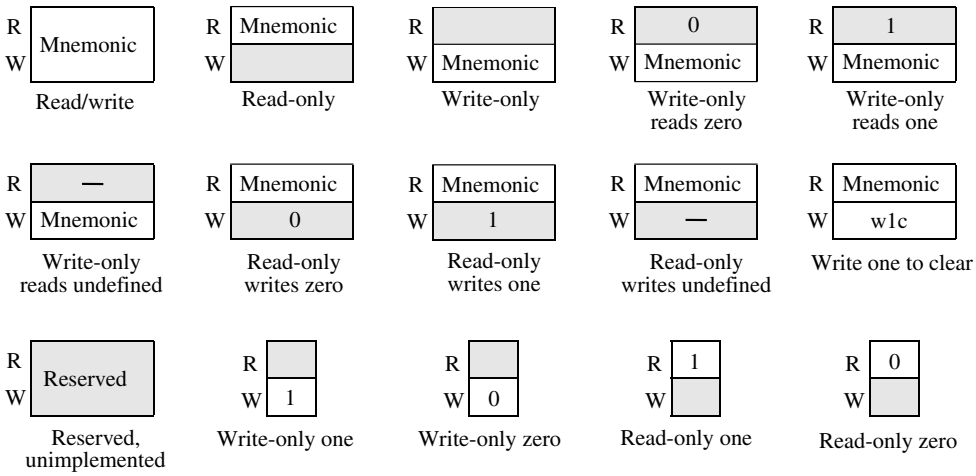


Figure 4. Register figure conventions

NOTE

Reset values of reserved locations documented in this manual are subject to change and must not be used for diagnostic purposes.

1.6 Conventions

1.6.1 Notes and cautions

Specific information is provided as part of notes and cautions throughout this manual.

NOTE

Emphasizes information that deserves extra attention.

CAUTION

Informs you of situations that could lead to highly undesirable outcomes—such as damage to the chip or irreversible malfunction.

1.6.2 Numbering systems

The following suffixes identify different numbering systems:

Table 1. Numbering systems

This suffix	Identifies a
b	Binary number. For example, the binary equivalent of the number 5 is mentioned as 101b. In some cases, *0b* is prefixed to binary numbers.
d	Decimal number. Decimal numbers are followed by this suffix only when there is a possibility of confusion. In general, decimal numbers are used without a suffix.
h	Hexadecimal number. For example, the hexadecimal equivalent of the number 60 is mentioned as 3Ch. In some cases, *0x* is prefixed to hexadecimal numbers.

1.6.3 Typographic notation

The following typographic notations are used throughout this document:

Table 2. Typographic notation

Example	Description
x and other italicized text	The italicized, lowercase *x* is used as a placeholder for replaceable numbers. In general, italicized text is used for titles of publications and for emphasis. Additionally, italics could be used for metasymbols in syntax descriptions. Plain lowercase letters are used as placeholders for single letters and numbers.
`code font`	Fixed-width font (such as Courier) used for code. It is used for a letter, word, or phrase that you want the user to type. For example, "Type `Read` and press Enter." This type of font is also used for instruction mnemonics, directives, symbols, subcommands, parameters, operators, computer-language elements, code listings, commands that appear in running text, and for sample code. Instruction mnemonics and directives in text and tables are mentioned in all caps; for example, BSR.
SR[SCM]	A mnemonic in square brackets represents the name of a register field. This example refers to the Scaling Mode (SCM) field in the Status Register (SR).
REVNO[6:4], XAD[7:0]	Numbers in brackets that are separated by a colon represent either: • A subset of a register's named field For example, REVNO[6:4] refers to bits 6-4 that are part of the COREREV field occupying bits 6-0 of the REVNO register. • A continuous range of individual signals of a bus For example, XAD[7:0] refers to signals 7-0 of the XAD bus.
MOD.REG	A period separates the elements of a hierarchy: subsystem.module.register. For example: • SWT.TO means that the TO register is located in the SWT module. • SMU.XRDC.CR means that the CR register is located in the XRDC module within the SMU subsystem.

1.6.4 Special terms

The following terms have special meanings.

Table 3. Special terms

Term	Meaning
Asserted	Refers to the state of a signal as follows: • An active-high signal is asserted when high (1). • An active-low signal is asserted when low (0).
Deasserted	Refers to the state of a signal as follows: • An active-high signal is deasserted when low (0). • An active-low signal is deasserted when high (1). In some cases, deasserted signals are described as *negated*.
Reserved	Refers to memory space, register, field, or programming setting. Writes to a reserved location can result in unpredictable functionality or behavior. You must:

Table continues on the next page...

Table 3. Special terms (continued)

Term	Meaning
	• Not modify the default value of a reserved programming setting, such as the reset value of a reserved register field. • Consider undefined locations in memory to be reserved. • Not use the reset values of reserved locations documented in this manual for diagnostic purposes. They are subject to change.
Write 1 to clear (w1c)	Refers to the access type of a register field that is used to clear the field by writing the value 1 to it.
Undefined (u)	Refers to undefined reset values

1.7 Editorial changes

Each new release of this document includes editorial improvements such as:

- Spelling
- Grammar
- Punctuation
- Voice
- Tense
- Capitalization
- Formatting
- Presentation
- Navigation

1.8 Acronyms and Abbreviations

The table below contains acronyms and abbreviations used in this document.

Acronyms and Abbreviated Terms

Term	Meaning
ACMP	Analog Comparator
ADC	Analog-to-Digital Converter
AHB	Advanced High-performance Bus
AIPS	Arm IP Bus
ALU	Arithmetic Logic Unit
AMBA	Advanced Microcontroller Bus Architecture
AON	Always On domain

Table continues on the next page...

Table continued from the previous page...

Term	Meaning
APB	Advanced Peripheral Bus
ASRC	Asynchronous Sample Rate Converter
AXI	Advanced eXtensible Interface
BBNSM	Battery Backed Non-Secure Module
BBSM	Battery Backed Secure Module
CAN	Controller Area Network
CCM	Clock Controller Module
CM7	Arm Cortex M7 Core
CPU	Central Processing Unit
CSI	CMOS Sensor Interface
CTI	Cross Trigger Interface
D-cache	Data cache
DAP	Debug Access Port
DDR	Double data rate
DMA	Direct memory access
DPLL	Digital phase-locked loop
DRAM	Dynamic random access memory
ECC	Error correcting codes
LPSPI	Low-power SPI
EDMA	Enhanced Direct Memory Access
EIM	Error Injection Module
ENET	Ethernet
EPROM	Erasable Programmable Read-Only Memory
ETF	Embedded Trace FIFO
ETM	Embedded Trace Macrocell
FIFO	First-In-First-Out
GIC	General Interrupt Controller
GPC	General Power Controller
GPIO	General-Purpose I/O

Table continues on the next page...

Table continued from the previous page...

Term	Meaning
GPR	General-Purpose Register
GPS	Global Positioning System
GPT	General-Purpose Timer
GPV	Global Programmers View
HAB	High-Assurance Boot
I-cache	Instruction cache
I2C or I^2C	Inter-Integrated Circuit
IC	Integrated Circuit
IEEE	Institute of Electrical and Electronics Engineers
IOMUX	Input-Output Multiplexer
IP	Intellectual Property
IrDA	Infrared Data Association
JTAG	Joint Test Action Group (a serial bus protocol usually used for test purposes)
LDO	Low-Dropout
LSB	Least-Significant Byte
LUT	Look-Up Table
LVDS	Low Voltage Differential Signaling
MAC	Medium Access Control
MCM	Miscellaneous Control Module
MMC	Multimedia Card
MSB	Most-Significant Byte
OCRAM	On-Chip Random-Access Memory
OCOTP	On-Chip One-Time Programmable Controller
PCI	Peripheral Component Interconnect
PCIe	PCI express
PIC	Programmable Interrupt Controller

Table continues on the next page...

Table continued from the previous page...

Term	Meaning
PMU	Power Management Unit
POR	Power-On Reset
PSRAM	Pseudo-Static Random Access Memory
PWM	Pulse Width Modulation
QoS	Quality of Service
RISC	Reduced Instruction Set Computing
ROM	Read-Only Memory
RTOS	Real-Time Operating System
Rx	Receive
SAI	Synchronous Audio Interface
SD	Secure Digital
SDIO	Secure Digital Input/Output
SDMA	Smart DMA
SIM	Subscriber Identification Module
SoC	System-on-Chip
SPDIF	Sony Philips Digital Interface
SPI	Serial Peripheral Interface
SRAM	Static Random-Access Memory
SRC	System Reset Controller
TPIU	Trace Port Interface Unit
Tx	Transmit
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
USDHC	Ultra Secured Digital Host Controller
WDOG	Watchdog
WLAN	Wireless Local Area Network

Chapter 2
Introduction

2.1 Overview

The i.MX RT1180 is a high-performance networking crossover MCU, which features a main computer Arm® Cortex®-M7 at 800 MHz, and a power efficient Arm Cortex-M33 at 240 MHz. The i.MX RT1180 includes an integrated Gbps Time Sensitive Networking (TSN) switch.

This device also has an EtherCAT Device Controller to support multiple Industrial networking protocols, bridging the communications between real-time Ethernet and Industry 4.0 systems.

The i.MX RT1180 integrates advanced power management that reduces complexity of an external power supply and simplifies power sequencing. The i.MX RT1180 also provides various memory interfaces and a wide range of peripherals.

2.1.1 Block Diagram

The functional block diagram is shown in the figure below. This diagram provides a view of the chip's major functional components and core complexes.

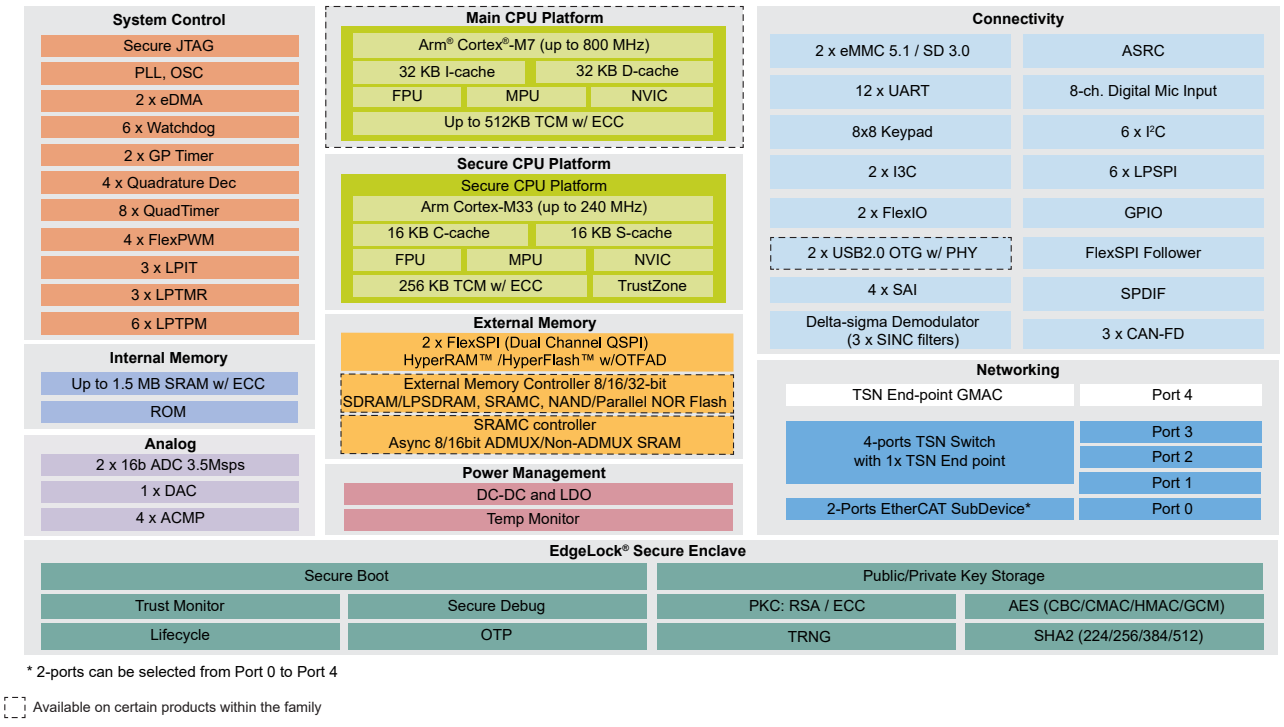


Figure 5. Simplified Block Diagram

2.1.2 System Bus Diagram

The system bus diagram is shown below.

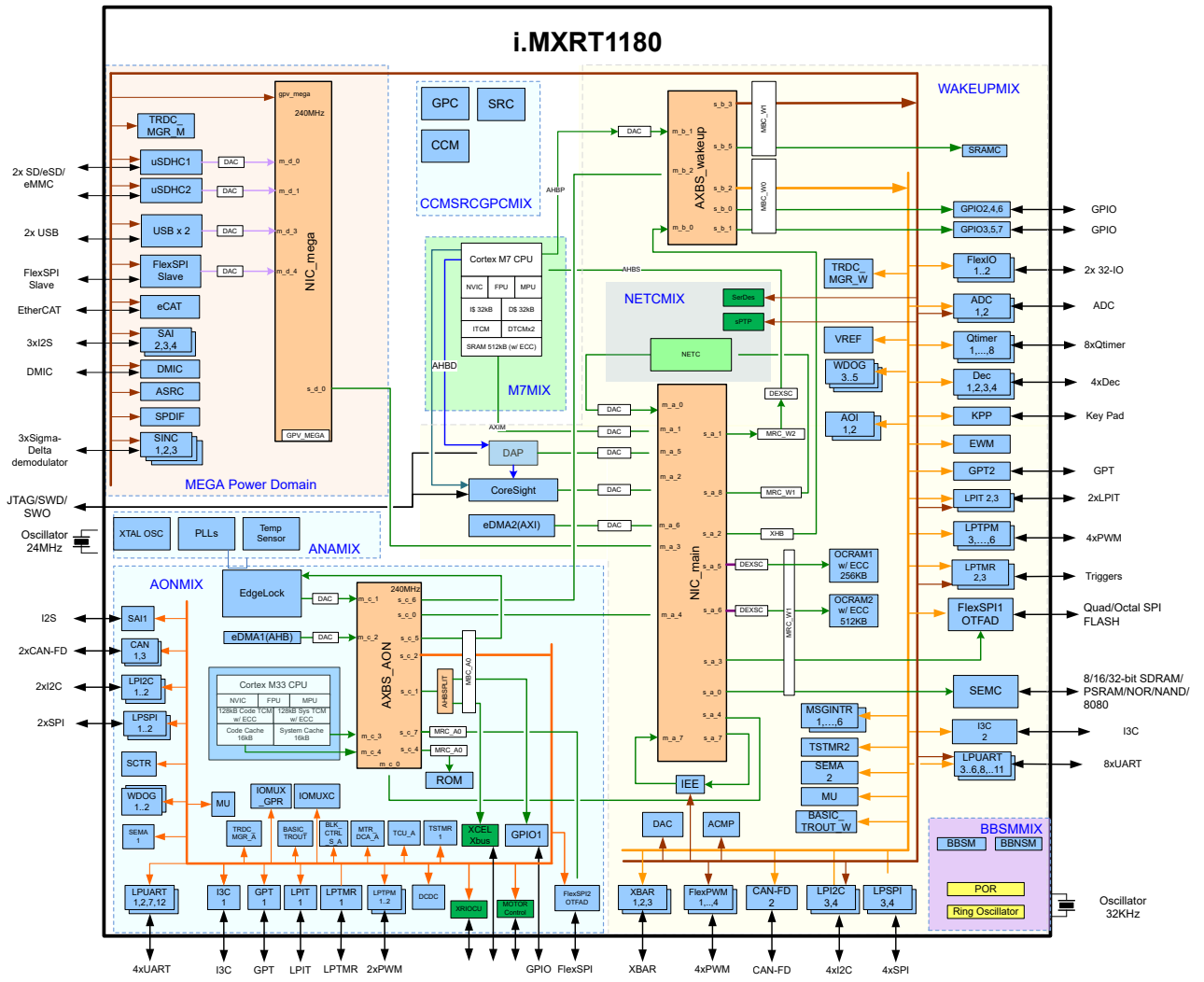


Figure 6. System Bus Diagram

2.1.3 Features

The i.MX RT1180 processor is based on Arm Cortex-M7 Platform and Arm Cortex-M33 which have the following features:

Arm Cortex-M7 Platform:

- Arm Cortex-M7 Processor:
  - 32 KB L1 Instruction Cache
  - 32 KB L1 Data Cache
  - Full featured Floating Point Unit (FPU) with support of the VFPv5 architecture
  - Support the ARMv7-M Thumb instruction set, defined in the ARM v7-M architecture
- Integrated Memory Protection Unit (MPU), up to 16 individual protection regions
- Up to 512 KB I-TCM and D-TCM in total
- Target frequency of 800 MHz at 1.1 V (over voltage) with Forward Body Biasing (FBB)
- ECC support for cache and TCM

Arm Cortex®-M33 Platform:

- Microcontroller available both for boot and for customer application
- Arm Cortex®-M33 Processor:
  - 16KB L1 Instruction Cache
  - 16KB L1 System Cache
  - 256 KB TCM, also accessible as SRAM by the rest of the system
- Target frequency of 240MHz at 1.0V without body biasing
- ECC support for both cache and TCM

Security:

- TRDC – Trusted Resource Domain Controller
  - Supports up to 16 resource domains
- Arm TrustZone® (TZ) architecture
- Secure and trusted access control
- Battery Backed Security Module (BBSM)
  - Monotonic counter - Secure real-time clock (RTC) - Zeroized Master Key
- Inline Encryption Engine (IEE)
  - External memory encryption/decryption
  - I/O direct encrypted storage and retrieval (Stream Support)

System Debug:

- Arm CoreSight® debug and trace architecture
- Trace Port Interface Unit (TPIU) to support off-chip real-time trace
- Support for 5-pin (JTAG) and SWD debug interfaces

On Chip Memory:

- Boot ROM (160KB)
- 512KB Tightly Coupled RAM (TCM) for CM7, with ECC
- 256KB Tightly Coupled RAM (TCM) for CM33, with ECC
- Dedicated 768KB OCRAM, with ECC

External Memory Interfaces:

- Smart External Memory Controller (SEMC)
  - 8/16/32-bit SDRAM interface, 4 Chip Select (CS) and each CS up to 512Mb
  - 8/16-bit NAND FLASH interface, with hardware ECC
  - 8/16-bit NOR flash interface
  - SRAM Interface
- FlexSPI with support for XIP and support for either one Octal SPI, or parallel read mode of two identical Quad SPI FLASH devices
  - OTFAD is supported for decryption with 0 cycle delay
- SRAMC Interface (FPGA I/F)

Audio:

- SPDIF Input and Output
- 4x Synchronous Audio Interface (SAI) modules supporting I2S, AC97, TDM, and Codec/DSP interfaces
- Digital microphone input, 8-channel PDM
- 1x Asynchronous Sample Rate Converter (ASRC)

Connectivity:

- 2x USB 2.0 OTG controller with integrated PHY interfaces
- 2x Ultra Secure Digital Host Controller (uSDHC) interfaces
  - uSDHC1 with boot support for eMMC 5.1 compliance with HS400 DDR signaling to support up to 400 MB/sec
  - uSDHC2 with boot support for SD/SDIO 3.0 compliance with 200 MHZ SDR signaling to support up to 100 MB/sec
  - Support for SDXC (extended capacity)
- EtherCAT®
- 12x Low-power Universal asynchronous receiver/transmitter (LPUARTs) modules
- 6x LPSPI modules
- 6x I2C modules
- 2x I3C modules
- 3x CAN-FD modules
- 2x FlexIO modules
- Advanced and flexible Ethernet:
  - Port virtualization support on ENETC1 MAC for dual Access (CM33/CM7)
  - Extended support of TSN standards over flexible architecture
  - 1x independent 1Gbps TSN MAC end point
  - 5-Ports (4 external + 1 internal) TSN Switch with 1Gbps TSN MAC
  - OPC UA Frame Summation HW acceleration integrated in switch
  - Maximum 5 RGMII/MII/RMII external ports
- 1x FlexSPI follower, 4/8 bit, 133MHz

Timers and PWMs:

- 2x General Programmable Timer (GPT)
  - 4-channel generic 32-bit resolution timer each
  - Each support standard capture and compare operation
- 3x Low Power Periodical Interrupt Timer (LPIT)
  - 4 channel
  - 4 external trigger source
  - Generic 32-bit resolution timer
  - Periodical interrupt generation
- 6x Timer/PWM module (TPM)
  - Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
  - 32-bit counter, support free-running counter or modulo counter mode, counting up or down

  - Includes 4 channels that can be configured for input capture, output compare, edge-aligned PWM mode, or center-aligned PWM mode
- 8x Quad Timer (TMR)
  - 4-channel generic 16-bit resolution timer each
  - Each support standard capture and compare operation
  - Quadrature decoder integrated
- 4x FlexPWM
  - Maximum 8 individual PWM channels per each
  - 16-bit resolution PWM suitable for Motor Control applications
- 4x Quadrature Decoder

GPIO and Pin Multiplexing:

- General-purpose input/output (GPIO) modules with interrupt capability
- Input/output multiplexing controller (IOMUXC) to provide centralized pad control

Analog:

- 2x 16-bit dual-channel SAR ADC, 3.5Msps, 30-channels
- 4x Analog Comparators (CMP)
- 1x 12-bit Digital-Analog-Converter (DAC)
- 3x SINC filter(4-ch) to interface to external sigma-delta ADC

Power Management:

- Full PMIC integration, including on-chip DCDC and LDO
- Temperature sensor with programmable trim points
- GPC hardware power management controller

2.2 Target Applications

The i.MX RT1180 can be used but not limited to applications such as:

- Industrial
- IoT

2.3 Endianness Support

The chip supports only the Little Endian mode.

Chapter 3
Memory Maps

3.1 Memory system overview

This section introduces the memory architecture of the chip.

NOTE

Accessing the reserved memory regions can result in unpredictable behavior.

NOTE

The first 16k OCRAM is reserved as a rom patch area and cannot be safely used by application images.

3.2 System memory map (CM7)

The table below shows the CM7 memory map. There are both Non-Secure (NS) and Secure (S) access possibilities.

Table 4. System memory map (CM7)

Start address	End address	Size (KB)	Description
E010_0000 (NS)	FFFF_FFFF (NS)	523264	Reserved
E004_4000 (NS)	E00F_FFFF (NS)	752	M7 Private Peripheral Bus
E000_0000 (NS)	E004_3FFF (NS)	272	M7 Private Peripheral Bus
D000_0000 (NS)	DFFF_FFFF (NS)	262144	SEMC (offset 1.25 GB)
C000_0000 (NS)	CFFF_FFFF (NS)	262144	SEMC (offset 1 GB)
B000_0000 (NS)	BFFF_FFFF (NS)	262144	SEMC (offset 768 MB)
A000_0000 (NS)	AFFF_FFFF (NS)	262144	SEMC (offset 512 MB)
9000_0000 (NS)	9FFF_FFFF (NS)	262144	SEMC (offset 256 MB)
8000_0000 (NS)	8FFF_FFFF (NS)	262144	SEMC0 (offset 0 MB)
6100_0000 (NS)	7FFF_FFFF (NS)	507904	Reserved
6000_0000 (NS)	60FF_FFFF (NS)	16384	NETC register
4DE1_1000 (NS) 5DE1_1000 (S)	4FFF_FFFF (NS) 5FFF_FFFF (S)	34748	Reserved
4DE1_0000 (NS) 5DE1_0000 (S)	4DE1_0FFF (NS) 5DE1_0FFF (S)	4	FlexSPI2 TX FIFO
4DE0_1000 (NS)	4DE0_FFFF (NS)	60	Reserved

Table continues on the next page...

Table 4. System memory map (CM7) (continued)

Start address	End address	Size (KB)	Description
5DE0_1000 (S)	5DE0_FFFF (S)		
4DE0_0000 (NS) 5DE0_0000 (S)	4DE0_0FFF (NS) 5DE0_0FFF (S)	4	FlexSPI2 RX FIFO
4B88_0000 (NS) 5B88_0000 (S)	4DDF_FFFF (NS) 5DDF_FFFF (S)	38400	Reserved
4B87_0000 (NS) 5B87_0000 (S)	4B87_FFFF (NS) 5B87_FFFF (S)	64	CM7 Domain Process Monitor
4B86_0000 (NS) 5B86_0000 (S)	4B86_FFFF (NS) 5B86_FFFF (S)	64	CM7 Domain EIM
4B85_0000 (NS) 5B85_0000 (S)	4B85_FFFF (NS) 5B85_FFFF (S)	64	CM7 Domain MCM
4758_0000 (NS) 5758_0000 (S)	4B84_FFFF (NS) 5B84_FFFF (S)	68416	Reserved
4750_0000 (NS) 5750_0000 (S)	4757_FFFF (NS) 5757_FFFF (S)	512	Edgelock AHB Slave
4743_1000 (NS) 5743_1000 (S)	474F_FFFF (NS) 574F_FFFF (S)	828	Reserved
4743_0000 (NS) 5743_0000 (S)	4743_0FFF (NS) 5743_0FFF (S)	4	FlexSPI1 TX FIFO
4742_1000 (NS) 5742_1000 (S)	4742_FFFF (NS) 5742_FFFF (S)	60	Reserved
4742_0000 (NS) 5742_0000 (S)	4742_0FFF (NS) 5742_0FFF (S)	4	FlexSPI1 RX FIFO
4741_0000 (NS) 5741_0000 (S)	4741_FFFF (NS) 5741_FFFF (S)	64	Reserved
4740_0000 (NS) 5740_0000 (S)	4740_FFFF (NS) 5740_FFFF (S)	64	GPIO1
4480_0000 (NS) 5480_0000 (S)	473F_FFFF (NS) 573F_FFFF (S)	45056	Reserved

Table continues on the next page...

Table 4. System memory map (CM7) (continued)

Start address	End address	Size (KB)	Description
4400_0000 (NS) 5400_0000 (S)	447F_FFFF (NS) 547F_FFFF (S)	8192	AIPS Peripheral Bridge
43A0_0000 (NS) 53A0_0000 (S)	43FF_FFFF (NS) 53FF_FFFF (S)	6144	Reserved
4390_0000 (NS) 5390_0000 (S)	439F_FFFF (NS) 539F_FFFF (S)	1024	NIC_MAIN GPV Registers
4388_0000 (NS) 5388_0000 (S)	438F_FFFF (NS) 538F_FFFF (S)	512	SRAMC
4386_0000 (NS) 5386_0000 (S)	4387_FFFF (NS) 5387_FFFF (S)	128	Reserved
4385_0000 (NS) 5385_0000 (S)	4385_FFFF (NS) 5385_FFFF (S)	64	GPIO6
4384_0000 (NS) 5384_0000 (S)	4384_FFFF (NS) 5384_FFFF (S)	64	GPIO5
4383_0000 (NS) 5383_0000 (S)	4383_FFFF (NS) 5383_FFFF (S)	64	GPIO4
4382_0000 (NS) 5382_0000 (S)	4382_FFFF (NS) 5382_FFFF (S)	64	GPIO3
4381_0000 (NS) 5381_0000 (S)	4381_FFFF (NS) 5381_FFFF (S)	64	GPIO2
4380_0000 (NS) 5380_0000 (S)	4380_FFFF (NS) 5380_FFFF (S)	64	Edgelock ISP_AP
4300_0000 (NS) 5300_0000 (S)	437F_FFFF (NS) 537F_FFFF (S)	8192	Reserved
4280_0000 (NS) 5280_0000 (S)	42FF_FFFF (NS) 52FF_FFFF (S)	8192	AIPS Peripheral Bridge
4200_0000 (NS) 5200_0000 (S)	427F_FFFF (NS) 527F_FFFF (S)	8192	AIPS Peripheral Bridge
4110_0000 (NS)	41FF_FFFF (NS)	15360	Reserved

Table continues on the next page...

Table 4. System memory map (CM7) (continued)

Start address	End address	Size (KB)	Description
5110_0000 (S)	51FF_FFFF (S)		
4000_0000 (NS) 5000_0000 (S)	410F_FFFF (NS) 510F_FFFF (S)	17408	Debug - DAP
2800_0000 (NS) 3800_0000 (S)	2FFF_FFFF (NS) 3FFF_FFFF (S)	131072	FlexSPI1
2054_0000 (NS) 3054_0000 (S)	21FF_FFFF (NS) 31FF_FFFF (S)	27392	Reserved
2050_0000 (NS) 3050_0000 (S)	2053_FFFF (NS) 3053_FFFF (S)	256	OCRAM2 (256 KB)
2048_0000 (NS) 3048_0000 (S)	204F_FFFF (NS) 304F_FFFF (S)	512	OCRAM1 (512 KB)
2024_0000 (NS) 3024_0000 (S)	2047_FFFF (NS) 3047_FFFF (S)	2304	Reserved
2022_0000 (NS) 3022_0000 (S)	2023_FFFF (NS) 3023_FFFF (S)	128	M33 System TCM (continued aliased if M33 SW config option Code TCM=0KB)
2020_0000 (NS) 3020_0000 (S)	2021_FFFF (NS) 3021_FFFF (S)	128	M33 System TCM (aliased)
201E_0000 (NS) 301E_0000 (S)	201F_FFFF (NS) 301F_FFFF (S)	128	M33 Code TCM (aliased)
201C_0000 (NS) 301C_0000 (S)	201D_FFFF (NS) 301D_FFFF (S)	128	M33 Code TCM (continued aliased if M33 SW config option System TCM=0KB)
2008_0000 (NS) 3008_0000 (S)	201B_FFFF (NS) 301B_FFFF (S)	1280	Reserved
2004_0000 (NS)	2007_FFFF (NS)	256	M7 DTCM (256KB ECC if M7 SW config option ITCM=0KB)
2002_0000 (NS)	2003_FFFF (NS)	128	M7 DTCM (continued)
2000_0000 (NS)	2001_FFFF (NS)	128	M7 DTCM (256KB)
0800_0000 (NS) 1800_0000 (S)	0FFF_FFFF (NS) 1FFF_FFFF (S)	131072	Reserved

Table continues on the next page...

Table 4. System memory map (CM7) (continued)

Start address	End address	Size (KB)	Description
0400_0000 (NS) 1400_0000 (S)	07FF_FFFF (NS) 17FF_FFFF (S)	65536	FlexSPI2
0014_0000 (NS) 1014_0000 (S)	03FF_FFFF (NS) 13FF_FFFF (S)	64256	Reserved
0010_0000 (NS) 1010_0000 (S)	0013_FFFF (NS) 1013_FFFF (S)	256	M33 Boot ROM (aliased)
0008_0000 (NS) 1008_0000 (S)	000F_FFFF (NS) 100F_FFFF (S)	512	Reserved
0004_0000 (NS)	0007_FFFF (NS)	256	M7 ITCM (256KB if M7 SW config option DTCM=0KB)
0002_8000 (NS)	0003_FFFF (NS)	96	M7 ITCM (96KB)
0002_0000 (NS)	00027FFF (NS)	32	M7 ITCM (32KB)
0000_0000 (NS)	0001_FFFF (NS)	128	M7 ITCM (128KB)

3.3 System memory map (CM33)

The table below shows the CM33 memory map. There are both Non-Secure (NS) and Secure (S) access possibilities.

Table 5. System memory map (CM33)

Start address	End address	Size (KB)	Description
E010_0000 (NS) E010_0000 (S)	FFFF_FFFF (NS) FFFF_FFFF (S)	523264	Reserved
E0044000 (NS) E0044000 (S)	E00F_FFFF (NS) E00F_FFFF (S)	752	M33 Private Peripheral Bus - External
E000_0000 (NS) E000_0000 (S)	E004_3FFF (NS) E004_3FFF (S)	272	M33 Private Peripheral Bus - Internal
C000_0000 (NS) D000_0000 (S)	CFFF_FFFF (NS) DFFF_FFFF (S)	262144	SEMC (offset 1 GB)
A000_0000 (NS) B000_0000 (S)	AFFF_FFFF (NS) BFFF_FFFF (S)	262144	SEMC (offset 512 MB)
8000_0000 (NS) 9000_0000 (S)	8FFF_FFFF (NS) 9FFF_FFFF (S)	262144	SEMC0 (offset 0 MB)

Table continues on the next page...

Table 5. System memory map (CM33) (continued)

Start address	End address	Size (KB)	Description
6100_0000 (NS) 7100_0000 (S)	6FFF_FFFF (NS) 7FFF_FFFF (S)	245760	Reserved
6000_0000 (NS) 7000_0000 (S)	60FF_FFFF (NS) 70FF_FFFF (S)	16384	NETC Register
4DE11000 (NS) 5DE11000 (S)	4FFF_FFFF (NS) 5FFF_FFFF (S)	34748	Reserved
4DE1_0000 (NS) 5DE1_0000 (S)	4DE1_0FFF (NS) 5DE1_0FFF (S)	4	FlexSPI2 TX FIFO
4B88_0000 (NS) 5B88_0000 (S)	4DE0_FFFF (NS) 5DE0_FFFF (S)	38464	Reserved
4B87_0000 (NS) 5B87_0000 (S)	4B87_FFFF (NS) 5B87_FFFF (S)	64	CM7 Domain Process Monitor
4B86_0000 (NS) 5B86_0000 (S)	4B86_FFFF (NS) 5B86_FFFF (S)	64	CM7 Domain EIM
4B85_0000 (NS) 5B85_0000 (S)	4B85_FFFF (NS) 5B85_FFFF (S)	64	CM7 Domain MCM
4758_0000 (NS) 5758_0000 (S)	4B84_FFFF (NS) 5B84_FFFF (S)	68416	Reserved
4750_0000 (NS) 5750_0000 (S)	4757_FFFF (NS) 5757_FFFF (S)	512	Edgelock AHB Slave
47431000 (NS) 57431000 (S)	474F_FFFF (NS) 574F_FFFF (S)	828	Reserved
4743_0000 (NS) 5743_0000 (S)	4743_0FFF (NS) 5743_0FFF (S)	4	FlexSPI1 TX FIFO
47421000 (NS) 57421000 (S)	4742_FFFF (NS) 5742_FFFF (S)	60	Reserved
4742_0000 (NS) 5742_0000 (S)	4742_0FFF (NS) 5742_0FFF (S)	4	FlexSPI1 RX FIFO
4741_0000 (NS)	4741_FFFF (NS)	64	Reserved

Table continues on the next page...

Table 5. System memory map (CM33) (continued)

Start address	End address	Size (KB)	Description
5741_0000 (S)	5741_FFFF (S)		
4740_0000 (NS) 5740_0000 (S)	4740_FFFF (NS) 5740_FFFF (S)	64	GPIO1
4480_0000 (NS) 5480_0000 (S)	473F_FFFF (NS) 573F_FFFF (S)	45056	Reserved
4400_0000 (NS) 5400_0000 (S)	447F_FFFF (NS) 547F_FFFF (S)	8192	AIPS Peripheral Bridge
43A0_0000 (NS) 53A0_0000 (S)	43FF_FFFF (NS) 53FF_FFFF (S)	6144	Reserved
4390_0000 (NS) 5390_0000 (S)	439F_FFFF (NS) 539F_FFFF (S)	1024	NIC_MAIN GPV Registers
4386_0000 (NS) 5386_0000 (S)	438F_FFFF (NS) 538F_FFFF (S)	640	Reserved
4388_0000 (NS) 5388_0000 (S)	438F_FFFF (NS) 538F_FFFF (S)	512	Reserved
4385_0000 (NS) 5385_0000 (S)	4385_FFFF (NS) 5385_FFFF (S)	64	GPIO6
4384_0000 (NS) 5384_0000 (S)	4384_FFFF (NS) 5384_FFFF (S)	64	GPIO5
4383_0000 (NS) 5383_0000 (S)	4383_FFFF (NS) 5383_FFFF (S)	64	GPIO4
4382_0000 (NS) 5382_0000 (S)	4382_FFFF (NS) 5382_FFFF (S)	64	GPIO3
4381_0000 (NS) 5381_0000 (S)	4381_FFFF (NS) 5381_FFFF (S)	64	GPIO2
4380_0000 (NS) 5380_0000 (S)	4380_FFFF (NS) 5380_FFFF (S)	64	Edgelock ISP_AP
4300_0000 (NS) 5300_0000 (S)	437F_FFFF (NS) 537F_FFFF (S)	8192	Reserved

Table continues on the next page...

Table 5. System memory map (CM33) (continued)

Start address	End address	Size (KB)	Description
4280_0000 (NS) 5280_0000 (S)	42FF_FFFF (NS) 52FF_FFFF (S)	8192	AIPS Peripheral Bridge
4200_0000 (NS) 5200_0000 (S)	427F_FFFF (NS) 527F_FFFF (S)	8192	AIPS Peripheral Bridge
4110_0000 (NS) 5110_0000 (S)	41FF_FFFF (NS) 51FF_FFFF (S)	15360	Reserved
4000_0000 (NS) 5000_0000 (S)	410F_FFFF (NS) 510F_FFFF (S)	17408	Debug - DAP
2800_0000 (NS) 3800_0000 (S)	2FFF_FFFF (NS) 3FFF_FFFF (S)	131072	FlexSPI1
2400_0000 (NS) 3400_0000 (S)	27FF_FFFF (NS) 37FF_FFFF (S)	65536	Reserved
2200_0000 (NS) 3200_0000 (S)	23FF_FFFF (NS) 33FF_FFFF (S)	32768	Aliased first 32 MB of FlexSPI2
2054_0000 (NS) 3054_0000 (S)	21FF_FFFF (NS) 31FF_FFFF (S)	27392	Reserved
2050_0000 (NS) 3050_0000 (S)	2053_FFFF (NS) 3053_FFFF (S)	256	OCRAM2 (256 KB)
2048_0000 (NS) 3048_0000 (S)	204F_FFFF (NS) 304F_FFFF (S)	512	OCRAM1 (512 KB)
2044_0000 (NS) 3044_0000 (S)	2047_FFFF (NS) 3047_FFFF (S)	256	M7 DTCM (continued aliased if M7 SW config option ITCM=0KB)
2040_0000 (NS) 3040_0000 (S)	2043_FFFF (NS) 3043_FFFF (S)	256	M7 DTCM (aliased)
203C_0000 (NS) 303C_0000 (S)	203F_FFFF (NS) 303F_FFFF (S)	256	M7 ITCM (aliased)
2038_0000 (NS) 3038_0000 (S)	203B_FFFF (NS) 303B_FFFF (S)	256	M7 ITCM (continued aliased if M7 SW config option DTCM=0KB)
2004_0000 (NS)	2037_FFFF (NS)	3328	Reserved

Table continues on the next page...

Table 5. System memory map (CM33) (continued)

Start address	End address	Size (KB)	Description
3004_0000 (S)	3037_FFFF (S)		
2002_0000 (NS) 3002_0000 (S)	2003_FFFF (NS) 3003_FFFF (S)	128	M33 System TCM (128KB, if M33 SW config option Code TCM=0KB)
2000_0000 (NS) 3000_0000 (S)	2001_FFFF (NS) 3001_FFFF (S)	128	M33 System TCM (128KB)
0FFE_0000 (NS) 1FFE_0000 (S)	0FFF_FFFF (NS) 1FFF_FFFF (S)	128	M33 Code TCM (128KB)
0FFC_0000 (NS) 1FFC_0000 (S)	0FFD_FFFF (NS) 1FFD_FFFF (S)	128	M33 Code TCM (128KB, if M33 SW config option System TCM=0KB)
0E00_0000 (NS) 1E00_0000 (S)	0FFB_FFFF (NS) 1FFB_FFFF (S)	32512	Reserved
0C00_0000 (NS) 1C00_0000 (S)	0DFF_FFFF (NS) 1DFF_FFFF (S)	32768	SEMC Code Space (aliased First 32 MB of DRAM at 0x8000_0000)
0800_0000 (NS) 1800_0000 (S)	0BFF_FFFF (NS) 1BFF_FFFF (S)	65536	Reserved
0400_0000 (NS) 1400_0000 (S)	07FF_FFFF (NS) 17FF_FFFF (S)	65536	FlexSPI2
0200_0000 (NS) 1200_0000 (S)	03FF_FFFF (NS) 13FF_FFFF (S)	32768	Aliased first 32 MB of FlexSPI1
00028000 (NS) 10028000 (S)	01FF_FFFF (NS) 11FF_FFFF (S)	32608	Reserved
0002_0000 (NS) 1002_0000 (S)	0002_7FFF (NS) 1002_7FFF (S)	32	M33 Boot ROM (32 KB)
0000_0000 (NS) 1000_0000 (S)	0001_FFFF (NS) 1001_FFFF (S)	128	M33 Boot ROM (128 KB)

3.4 AIPS-1 Memory Map

The table below shows the AIPS-1 memory map. The peripherals have both Non-Secure (NS) and Secure (S) access possibilities.

The start address of AIPS1 Peripherals for non-secure access is 0x4400_0000

The start address of AIPS1 Peripherals for secure access is 0x5400_0000

Table 6. AIPS-1 memory map

Start Address	End Address	Size (KB)	NIC Port
446D_0000 (NS) 546D_0000 (S)	447F_FFFF (NS) 547F_FFFF (S)	1216	Reserved
446C_0000 (NS) 546C_0000 (S)	446C_FFFF (NS) 546C_FFFF (S)	64	GPT1
445F_0000 (NS) 545F_0000 (S)	446B_FFFF (NS) 546B_FFFF (S)	448	Reserved
445E_0000 (NS) 545E_0000 (S)	445E_FFFF (NS) 545E_FFFF (S)	64	FlexSPI_OTFAD2
445C_0000 (NS) 545C_0000 (S)	445D_FFFF (NS) 545D_FFFF (S)	128	Reserved
445B_0000 (NS) 545B_0000 (S)	445B_FFFF (NS) 545B_FFFF (S)	64	CAN-FD3
4459_0000 (NS) 5459_0000 (S)	445A_FFFF (NS) 545A_FFFF (S)	128	Reserved
4458_0000 (NS) 5458_0000 (S)	4458_FFFF (NS) 5458_FFFF (S)	64	LPUART12
4457_0000 (NS) 5457_0000 (S)	4457_FFFF (NS) 5457_FFFF (S)	64	LPUART7
4453_0000 (NS) 5453_0000 (S)	4456_FFFF (NS) 5456_FFFF (S)	256	Reserved
4452_0000 (NS) 5452_0000 (S)	4452_FFFF (NS) 5452_FFFF (S)	64	DCDC
4451_0000 (NS) 5451_0000 (S)	4451_FFFF (NS) 5451_FFFF (S)	64	AXBS
4450_0000 (NS) 5450_0000 (S)	4450_FFFF (NS) 5450_FFFF (S)	64	BASIC_TROUT1
444F_0000 (NS) 544F_0000 (S)	444F_FFFF (NS) 544F_FFFF (S)	64	BLK_CTRL_S_AON
444E_0000 (NS)	444E_FFFF (NS)	64	TCU4

Table continues on the next page...

Table 6. AIPS-1 memory map (continued)

Start Address	End Address	Size (KB)	NIC Port
544E_0000 (S)	544E_FFFF (S)		
444D_0000 (NS) 544D_0000 (S)	444D_FFFF (NS) 544D_FFFF (S)	64	TCU3
444C_0000 (NS) 544C_0000 (S)	444C_FFFF (NS) 544C_FFFF (S)	64	TCU2
444B_0000 (NS) 544B_0000 (S)	444B_FFFF (NS) 544B_FFFF (S)	64	TCU1
444A_0000 (NS) 544A_0000 (S)	444A_FFFF (NS) 544A_FFFF (S)	64	MTR_DCA1
4449_0000 (NS) 5449_0000 (S)	4449_FFFF (NS) 5449_FFFF (S)	64	MTR_MSTR1
4448_0000 (NS) 5448_0000 (S)	4448_FFFF (NS) 5448_FFFF (S)	64	ANA_TOP
4447_0000 (NS) 5447_0000 (S)	4447_FFFF (NS) 5447_FFFF (S)	64	GPC
4446_0000 (NS) 5446_0000 (S)	4446_FFFF (NS) 5446_FFFF (S)	64	SRC
4445_0000 (NS) 5445_0000 (S)	4445_FFFF (NS) 5445_FFFF (S)	64	CCM_CTRL
4444_0000 (NS) 5444_0000 (S)	4444_FFFF (NS) 5444_FFFF (S)	64	BBNSM
4443_0000 (NS) 5443_0000 (S)	4443_FFFF (NS) 5443_FFFF (S)	64	ROMCP1
4442_0000 (NS) 5442_0000 (S)	4442_FFFF (NS) 5442_FFFF (S)	64	M33_TCM_MECC1
4441_0000 (NS) 5441_0000 (S)	4441_FFFF (NS) 5441_FFFF (S)	64	BLK_CTRL_BBSM
4440_0000 (NS) 5440_0000 (S)	4440_FFFF (NS) 5440_FFFF (S)	64	M33_CACHE_CTRL1

Table continues on the next page...

Table 6. AIPS-1 memory map (continued)

Start Address	End Address	Size (KB)	NIC Port
443F_0000 (NS) 543F_0000 (S)	443F_FFFF (NS) 543F_FFFF (S)	64	M33_PSF1
443E_0000 (NS) 543E_0000 (S)	443E_FFFF (NS) 543E_FFFF (S)	64	M33_PCF1
443D_0000 (NS) 543D_0000 (S)	443D_FFFF (NS) 543D_FFFF (S)	64	IPC1
443C_0000 (NS) 543C_0000 (S)	443C_FFFF (NS) 543C_FFFF (S)	64	IOMUXC_AON
443B_0000 (NS) 543B_0000 (S)	443B_FFFF (NS) 543B_FFFF (S)	64	SAI1
443A_0000 (NS) 543A_0000 (S)	443A_FFFF (NS) 543A_FFFF (S)	64	CAN-FD1
4439_0000 (NS) 5439_0000 (S)	4439_FFFF (NS) 5439_FFFF (S)	64	LPUART2
4438_0000 (NS) 5438_0000 (S)	4438_FFFF (NS) 5438_FFFF (S)	64	LPUART1
4437_0000 (NS) 5437_0000 (S)	4437_FFFF (NS) 5437_FFFF (S)	64	LPSPI2
4436_0000 (NS) 5436_0000 (S)	4436_FFFF (NS) 5436_FFFF (S)	64	LPSPI1
4435_0000 (NS) 5435_0000 (S)	4435_FFFF (NS) 5435_FFFF (S)	64	LPI2C2
4434_0000 (NS) 5434_0000 (S)	4434_FFFF (NS) 5434_FFFF (S)	64	LPI2C1
4433_0000 (NS) 5433_0000 (S)	4433_FFFF (NS) 5433_FFFF (S)	64	I3C1
4432_0000 (NS) 5432_0000 (S)	4432_FFFF (NS) 5432_FFFF (S)	64	TPM2
4431_0000 (NS)	4431_FFFF (NS)	64	TPM1

Table continues on the next page...

Table 6. AIPS-1 memory map (continued)

Start Address	End Address	Size (KB)	NIC Port
5431_0000 (S)	5431_FFFF (S)		
4430_0000 (NS) 5430_0000 (S)	4430_FFFF (NS) 5430_FFFF (S)	64	LPTMR1
442F_0000 (NS) 542F_0000 (S)	442F_FFFF (NS) 542F_FFFF (S)	64	LPIT1
442E_0000 (NS) 542E_0000 (S)	442E_FFFF (NS) 542E_FFFF (S)	64	LPWDOG2
442D_0000 (NS) 542D_0000 (S)	442D_FFFF (NS) 542D_FFFF (S)	64	LPWDOG1
442C_0000 (NS) 542C_0000 (S)	442C_FFFF (NS) 542C_FFFF (S)	64	TSTMR1
442B_0000 (NS) 542B_0000 (S)	442B_FFFF (NS) 542B_FFFF (S)	64	SYS_CTR_RDBASE1
442A_0000 (NS) 542A_0000 (S)	442A_FFFF (NS) 542A_FFFF (S)	64	SYS_CTR_CMPBASE1
4429_0000 (NS) 5429_0000 (S)	4429_FFFF (NS) 5429_FFFF (S)	64	SYS_CTR_CTLBASE1
4428_0000 (NS) 5428_0000 (S)	4428_FFFF (NS) 5428_FFFF (S)	64	TRDC-MC1
4427_0000 (NS) 5427_0000 (S)	4427_FFFF (NS) 5427_FFFF (S)	64	TRDC1 (AON Domain)
4426_0000 (NS) 5426_0000 (S)	4426_FFFF (NS) 5426_FFFF (S)	64	SEMA42_1
4424_0000 (NS) 5424_0000 (S)	4425_FFFF (NS) 5425_FFFF (S)	128	Reserved
4423_0000 (NS) 5423_0000 (S)	4423_FFFF (NS) 5423_FFFF (S)	64	MU1_B
4422_0000 (NS) 5422_0000 (S)	4422_FFFF (NS) 5422_FFFF (S)	64	MU1_A

Table continues on the next page...

Table 6. AIPS-1 memory map (continued)

Start Address	End Address	Size (KB)	NIC Port
4421_0000 (NS) 5421_0000 (S)	4421_FFFF (NS) 5421_FFFF (S)	64	BLK_CTRL_NS_AON
4420_0000 (NS) 5420_0000 (S)	4420_FFFF (NS) 5420_FFFF (S)	64	EDMA3_1
441F_0000 (NS) 541F_0000 (S)	441F_FFFF (NS) 541F_FFFF (S)	64	EDMA3_1
441E_0000 (NS) 541E_0000 (S)	441E_FFFF (NS) 541E_FFFF (S)	64	EDMA3_1
441D_0000 (NS) 541D_0000 (S)	441D_FFFF (NS) 541D_FFFF (S)	64	EDMA3_1
441C_0000 (NS) 541C_0000 (S)	441C_FFFF (NS) 541C_FFFF (S)	64	EDMA3_1
441B_0000 (NS) 541B_0000 (S)	441B_FFFF (NS) 541B_FFFF (S)	64	EDMA3_1
441A_0000 (NS) 541A_0000 (S)	441A_FFFF (NS) 541A_FFFF (S)	64	EDMA3_1
4419_0000 (NS) 5419_0000 (S)	4419_FFFF (NS) 5419_FFFF (S)	64	EDMA3_1
4418_0000 (NS) 5418_0000 (S)	4418_FFFF (NS) 5418_FFFF (S)	64	EDMA3_1
4417_0000 (NS) 5417_0000 (S)	4417_FFFF (NS) 5417_FFFF (S)	64	EDMA3_1
4416_0000 (NS) 5416_0000 (S)	4416_FFFF (NS) 5416_FFFF (S)	64	EDMA3_1
4415_0000 (NS) 5415_0000 (S)	4415_FFFF (NS) 5415_FFFF (S)	64	EDMA3_1
4414_0000 (NS) 5414_0000 (S)	4414_FFFF (NS) 5414_FFFF (S)	64	EDMA3_1
4413_0000 (NS)	4413_FFFF (NS)	64	EDMA3_1

Table continues on the next page...

Table 6. AIPS-1 memory map (continued)

Start Address	End Address	Size (KB)	NIC Port
5413_0000 (S)	5413_FFFF (S)		
4412_0000 (NS) 5412_0000 (S)	4412_FFFF (NS) 5412_FFFF (S)	64	EDMA3_1
4411_0000 (NS) 5411_0000 (S)	4411_FFFF (NS) 5411_FFFF (S)	64	EDMA3_1
4410_0000 (NS) 5410_0000 (S)	4410_FFFF (NS) 5410_FFFF (S)	64	EDMA3_1
440F_0000 (NS) 540F_0000 (S)	440F_FFFF (NS) 540F_FFFF (S)	64	EDMA3_1
440E_0000 (NS) 540E_0000 (S)	440E_FFFF (NS) 540E_FFFF (S)	64	EDMA3_1
440D_0000 (NS) 540D_0000 (S)	440D_FFFF (NS) 540D_FFFF (S)	64	EDMA3_1
440C_0000 (NS) 540C_0000 (S)	440C_FFFF (NS) 540C_FFFF (S)	64	EDMA3_1
440B_0000 (NS) 540B_0000 (S)	440B_FFFF (NS) 540B_FFFF (S)	64	EDMA3_1
440A_0000 (NS) 540A_0000 (S)	440A_FFFF (NS) 540A_FFFF (S)	64	EDMA3_1
4409_0000 (NS) 5409_0000 (S)	4409_FFFF (NS) 5409_FFFF (S)	64	EDMA3_1
4408_0000 (NS) 5408_0000 (S)	4408_FFFF (NS) 5408_FFFF (S)	64	EDMA3_1
4407_0000 (NS) 5407_0000 (S)	4407_FFFF (NS) 5407_FFFF (S)	64	EDMA3_1
4406_0000 (NS) 5406_0000 (S)	4406_FFFF (NS) 5406_FFFF (S)	64	EDMA3_1
4405_0000 (NS) 5405_0000 (S)	4405_FFFF (NS) 5405_FFFF (S)	64	EDMA3_1

Table continues on the next page...

Table 6. AIPS-1 memory map (continued)

Start Address	End Address	Size (KB)	NIC Port
4404_0000 (NS) 5404_0000 (S)	4404_FFFF (NS) 5404_FFFF (S)	64	EDMA3_1
4403_0000 (NS) 5403_0000 (S)	4403_FFFF (NS) 5403_FFFF (S)	64	EDMA3_1
4402_0000 (NS) 5402_0000 (S)	4402_FFFF (NS) 5402_FFFF (S)	64	EDMA3_1
4401_0000 (NS) 5401_0000 (S)	4401_FFFF (NS) 5401_FFFF (S)	64	EDMA3_1
4400_0000 (NS) 5400_0000 (S)	4400_FFFF (NS) 5400_FFFF (S)	64	EDMA3_1

3.5 AIPS-2 Memory Map

The table below shows the AIPS-2 memory map. The peripherals have both Non-Secure (NS) and Secure (S) access possibilities.

The start address of AIPS2 Peripherals for non-secure access is 0x4200_0000

The start address of AIPS2 Peripherals for secure access is 0x5200_0000

Table 7. AIPS-2 memory map

Start Address	End Address	Size	NIC Port
427F_0000 (NS) 527F_0000 (S)	427F_FFFF (NS) 527F_FFFF (S)	64	AOI4
427E_0000 (NS) 527E_0000 (S)	427E_FFFF (NS) 527E_FFFF (S)	64	AOI3
427C_0000 (NS) 527C_0000 (S)	427D_FFFF (NS) 527D_FFFF (S)	128	Reserved
427B_0000 (NS) 527B_0000 (S)	427B_FFFF (NS) 527B_FFFF (S)	64	EWM
427A_0000 (NS) 527A_0000 (S)	427A_FFFF (NS) 527A_FFFF (S)	64	Reserved
4279_0000 (NS) 5279_0000 (S)	4279_FFFF (NS) 5279_FFFF (S)	64	AOI2
4278_0000 (NS)	4278_FFFF (NS)	64	AOI1

Table continues on the next page...

Table 7. AIPS-2 memory map (continued)

Start Address	End Address	Size	NIC Port
5278_0000 (S)	5278_FFFF (S)		
4277_0000 (NS) 5277_0000 (S)	4277_FFFF (NS) 5277_FFFF (S)	64	XBAR3
4276_0000 (NS) 5276_0000 (S)	4276_FFFF (NS) 5276_FFFF (S)	64	XBAR2
4275_0000 (NS) 5275_0000 (S)	4275_FFFF (NS) 5275_FFFF (S)	64	XBAR1
4274_0000 (NS) 5274_0000 (S)	4274_FFFF (NS) 5274_FFFF (S)	64	eQDC4
4273_0000 (NS) 5273_0000 (S)	4273_FFFF (NS) 5273_FFFF (S)	64	eQDC3
4272_0000 (NS) 5272_0000 (S)	4272_FFFF (NS) 5272_FFFF (S)	64	eQDC2
4271_0000 (NS) 5271_0000 (S)	4271_FFFF (NS) 5271_FFFF (S)	64	eQDC1
4270_0000 (NS) 5270_0000 (S)	4270_FFFF (NS) 5270_FFFF (S)	64	TMR8
426F_0000 (NS) 526F_0000 (S)	426F_FFFF (NS) 526F_FFFF (S)	64	TMR7
426E_0000 (NS) 526E_0000 (S)	426E_FFFF (NS) 526E_FFFF (S)	64	TMR6
426D_0000 (NS) 526D_0000 (S)	426D_FFFF (NS) 526D_FFFF (S)	64	TMR5
426C_0000 (NS) 526C_0000 (S)	426C_FFFF (NS) 526C_FFFF (S)	64	TMR4
426B_0000 (NS) 526B_0000 (S)	426B_FFFF (NS) 526B_FFFF (S)	64	TMR3
426A_0000 (NS) 526A_0000 (S)	426A_FFFF (NS) 526A_FFFF (S)	64	TMR2

Table continues on the next page...

Table 7. AIPS-2 memory map (continued)

Start Address	End Address	Size	NIC Port
4269_0000 (NS) 5269_0000 (S)	4269_FFFF (NS) 5269_FFFF (S)	64	TMR1
4268_0000 (NS) 5268_0000 (S)	4268_FFFF (NS) 5268_FFFF (S)	64	PWM4
4267_0000 (NS) 5267_0000 (S)	4267_FFFF (NS) 5267_FFFF (S)	64	PWM3
4266_0000 (NS) 5266_0000 (S)	4266_FFFF (NS) 5266_FFFF (S)	64	PWM2
4265_0000 (NS) 5265_0000 (S)	4265_FFFF (NS) 5265_FFFF (S)	64	PWM1
4264_0000 (NS) 5264_0000 (S)	4264_FFFF (NS) 5264_FFFF (S)	64	Reserved
4263_0000 (NS) 5263_0000 (S)	4263_FFFF (NS) 5263_FFFF (S)	64	BASIC_TROUT2
4262_0000 (NS) 5262_0000 (S)	4262_FFFF (NS) 5262_FFFF (S)	64	TCU5
4261_0000 (NS) 5261_0000 (S)	4261_FFFF (NS) 5261_FFFF (S)	64	MTR_DCA2
4260_0000 (NS) 5260_0000 (S)	4260_FFFF (NS) 5260_FFFF (S)	64	LPADC1
425F_0000 (NS) 525F_0000 (S)	425F_FFFF (NS) 525F_FFFF (S)	64	Reserved
425E_0000 (NS) 525E_0000 (S)	425E_FFFF (NS) 525E_FFFF (S)	64	FLEXSPI1
425D_0000 (NS) 525D_0000 (S)	425D_FFFF (NS) 525D_FFFF (S)	64	FLEXIO2
425C_0000 (NS) 525C_0000 (S)	425C_FFFF (NS) 525C_FFFF (S)	64	FLEXIO1
425B_0000 (NS)	425B_FFFF (NS)	64	CAN-FD2

Table continues on the next page...

Table 7. AIPS-2 memory map (continued)

Start Address	End Address	Size	NIC Port
525B_0000 (S)	525B_FFFF (S)		
425A_0000 (NS) 525A_0000 (S)	425A_FFFF (NS) 525A_FFFF (S)	64	LPUART6
4259_0000 (NS) 5259_0000 (S)	4259_FFFF (NS) 5259_FFFF (S)	64	LPUART5
4258_0000 (NS) 5258_0000 (S)	4258_FFFF (NS) 5258_FFFF (S)	64	LPUART4
4257_0000 (NS) 5257_0000 (S)	4257_FFFF (NS) 5257_FFFF (S)	64	LPUART3
4256_0000 (NS) 5256_0000 (S)	4256_FFFF (NS) 5256_FFFF (S)	64	LPSPI4
4255_0000 (NS) 5255_0000 (S)	4255_FFFF (NS) 5255_FFFF (S)	64	LPSPI3
4254_0000 (NS) 5254_0000 (S)	4254_FFFF (NS) 5254_FFFF (S)	64	LPI2C4
4253_0000 (NS) 5253_0000 (S)	4253_FFFF (NS) 5253_FFFF (S)	64	LPI2C3
4252_0000 (NS) 5252_0000 (S)	4252_FFFF (NS) 5252_FFFF (S)	64	MIPI_I3C2
4251_0000 (NS) 5251_0000 (S)	4251_FFFF (NS) 5251_FFFF (S)	64	TPM6
4250_0000 (NS) 5250_0000 (S)	4250_FFFF (NS) 5250_FFFF (S)	64	TPM5
424F_0000 (NS) 524F_0000 (S)	424F_FFFF (NS) 524F_FFFF (S)	64	TPM4
424E_0000 (NS) 524E_0000 (S)	424E_FFFF (NS) 524E_FFFF (S)	64	TPM3
424D_0000 (NS) 524D_0000 (S)	424D_FFFF (NS) 524D_FFFF (S)	64	LPTMR2

Table continues on the next page...

Table 7. AIPS-2 memory map (continued)

Start Address	End Address	Size	NIC Port
424C_0000 (NS) 524C_0000 (S)	424C_FFFF (NS) 524C_FFFF (S)	64	LPIT2
424B_0000 (NS) 524B_0000 (S)	424B_FFFF (NS) 524B_FFFF (S)	64	RTWDOG5
424A_0000 (NS) 524A_0000 (S)	424A_FFFF (NS) 524A_FFFF (S)	64	RTWDOG4
4249_0000 (NS) 5249_0000 (S)	4249_FFFF (NS) 5249_FFFF (S)	64	RTWDOG3
4248_0000 (NS) 5248_0000 (S)	4248_FFFF (NS) 5248_FFFF (S)	64	TSTMR2
4247_0000 (NS) 5247_0000 (S)	4247_FFFF (NS) 5247_FFFF (S)	64	TRDC-MC2
4246_0000 (NS) 5246_0000 (S)	4246_FFFF (NS) 5246_FFFF (S)	64	TRDC2 (WAKEUP Domain)
4245_0000 (NS) 5245_0000 (S)	4245_FFFF (NS) 5245_FFFF (S)	64	SEMA2
4244_0000 (NS) 5244_0000 (S)	4244_FFFF (NS) 5244_FFFF (S)	64	MU2_B
4243_0000 (NS) 5243_0000 (S)	4243_FFFF (NS) 5243_FFFF (S)	64	MU2_A
4242_0000 (NS) 5242_0000 (S)	4242_FFFF (NS) 5242_FFFF (S)	64	BLK_CTRL_WAKEUP
4221_0000 (NS) 5221_0000 (S)	4241_FFFF (NS) 5241_FFFF (S)	2112	Reserved
4220_0000 (NS) 5220_0000 (S)	4220_FFFF (NS) 5220_FFFF (S)	64	DMA4
421F_0000 (NS) 521F_0000 (S)	421F_FFFF (NS) 521F_FFFF (S)	64	DMA4
421E_0000 (NS)	421E_FFFF (NS)	64	DMA4

Table continues on the next page...

Table 7. AIPS-2 memory map (continued)

Start Address	End Address	Size	NIC Port
521E_0000 (S)	521E_FFFF (S)		
421D_0000 (NS) 521D_0000 (S)	421D_FFFF (NS) 521D_FFFF (S)	64	DMA4
421C_0000 (NS) 521C_0000 (S)	421C_FFFF (NS) 521C_FFFF (S)	64	DMA4
421B_0000 (NS) 521B_0000 (S)	421B_FFFF (NS) 521B_FFFF (S)	64	DMA4
421A_0000 (NS) 521A_0000 (S)	421A_FFFF (NS) 521A_FFFF (S)	64	DMA4
4219_0000 (NS) 5219_0000 (S)	4219_FFFF (NS) 5219_FFFF (S)	64	DMA4
4218_0000 (NS) 5218_0000 (S)	4218_FFFF (NS) 5218_FFFF (S)	64	DMA4
4217_0000 (NS) 5217_0000 (S)	4217_FFFF (NS) 5217_FFFF (S)	64	DMA4
4216_0000 (NS) 5216_0000 (S)	4216_FFFF (NS) 5216_FFFF (S)	64	DMA4
4215_0000 (NS) 5215_0000 (S)	4215_FFFF (NS) 5215_FFFF (S)	64	DMA4
4214_0000 (NS) 5214_0000 (S)	4214_FFFF (NS) 5214_FFFF (S)	64	DMA4
4213_0000 (NS) 5213_0000 (S)	4213_FFFF (NS) 5213_FFFF (S)	64	DMA4
4212_0000 (NS) 5212_0000 (S)	4212_FFFF (NS) 5212_FFFF (S)	64	DMA4
4211_0000 (NS) 5211_0000 (S)	4211_FFFF (NS) 5211_FFFF (S)	64	DMA4
4210_0000 (NS) 5210_0000 (S)	4210_FFFF (NS) 5210_FFFF (S)	64	DMA4

Table continues on the next page...

Table 7. AIPS-2 memory map (continued)

Start Address	End Address	Size	NIC Port
420F_0000 (NS) 520F_0000 (S)	420F_FFFF (NS) 520F_FFFF (S)	64	DMA4
420E_0000 (NS) 520E_0000 (S)	420E_FFFF (NS) 520E_FFFF (S)	64	DMA4
420D_0000 (NS) 520D_0000 (S)	420D_FFFF (NS) 520D_FFFF (S)	64	DMA4
420C_0000 (NS) 520C_0000 (S)	420C_FFFF (NS) 520C_FFFF (S)	64	DMA4
420B_0000 (NS) 520B_0000 (S)	420B_FFFF (NS) 520B_FFFF (S)	64	DMA4
420A_0000 (NS) 520A_0000 (S)	420A_FFFF (NS) 520A_FFFF (S)	64	DMA4
4209_0000 (NS) 5209_0000 (S)	4209_FFFF (NS) 5209_FFFF (S)	64	DMA4
4208_0000 (NS) 5208_0000 (S)	4208_FFFF (NS) 5208_FFFF (S)	64	DMA4
4207_0000 (NS) 5207_0000 (S)	4207_FFFF (NS) 5207_FFFF (S)	64	DMA4
4206_0000 (NS) 5206_0000 (S)	4206_FFFF (NS) 5206_FFFF (S)	64	DMA4
4205_0000 (NS) 5205_0000 (S)	4205_FFFF (NS) 5205_FFFF (S)	64	DMA4
4204_0000 (NS) 5204_0000 (S)	4204_FFFF (NS) 5204_FFFF (S)	64	DMA4
4203_0000 (NS) 5203_0000 (S)	4203_FFFF (NS) 5203_FFFF (S)	64	DMA4
4202_0000 (NS) 5202_0000 (S)	4202_FFFF (NS) 5202_FFFF (S)	64	DMA4
4201_0000 (NS)	4201_FFFF (NS)	64	DMA4

Table continues on the next page...

Table 7. AIPS-2 memory map (continued)

Start Address	End Address	Size	NIC Port
5201_0000 (S)	5201_FFFF (S)		
4200_0000 (NS) 5200_0000 (S)	4200_FFFF (NS) 5200_FFFF (S)	64	DMA4

3.6 AIPS-3 Memory Map

The table below shows the AIPS-3 memory map. The peripherals have both Non-Secure (NS) and Secure (S) access possibilities.

The start address of AIPS3 Peripherals for non-secure access is 0x4280_0000

The start address of AIPS3 Peripherals for secure access is 0x5280_0000

Table 8. AIPS-3 memory map

Start Address	End Address	Size	NIC Port
42FE_0000 (NS) 52FE_0000 (S)	42FF_FFFF (NS) 52FF_FFFF (S)	128	Reserved
42FD_0000 (NS) 52FD_0000 (S)	42FD_FFFF (NS) 52FD_FFFF (S)	64	PROCESS_MONITOR
42FC_0000 (NS) 52FC_0000 (S)	42FC_FFFF (NS) 52FC_FFFF (S)	64	SFA
42FB_0000 (NS) 52FB_0000 (S)	42FB_FFFF (NS) 52FB_FFFF (S)	64	TCU6
42FA_0000 (NS) 52FA_0000 (S)	42FA_FFFF (NS) 52FA_FFFF (S)	64	Reserved
42F8_0000 (NS) 52F8_0000 (S)	42F9_FFFF (NS) 52F9_FFFF (S)	128	Reserved
42ED_0000 (NS) 52ED_0000 (S)	42F7_FFFF (NS) 52F7_FFFF (S)	704	Reserved
42EC_0000 (NS) 52EC_0000 (S)	42EC_FFFF (NS) 52EC_FFFF (S)	64	GPT2
42E5_0000 (NS) 52E5_0000 (S)	42EB_FFFF (NS) 52EB_FFFF (S)	448	Reserved
42E4_0000 (NS) 52E4_0000 (S)	42E4_FFFF (NS) 52E4_FFFF (S)	64	IEE

Table continues on the next page...

Table 8. AIPS-3 memory map (continued)

Start Address	End Address	Size	NIC Port
42E3_0000 (NS) 52E3_0000 (S)	42E3_FFFF (NS) 52E3_FFFF (S)	64	VREF
42E2_0000 (NS) 52E2_0000 (S)	42E2_FFFF (NS) 52E2_FFFF (S)	64	DAC
42E1_0000 (NS) 52E1_0000 (S)	42E1_FFFF (NS) 52E1_FFFF (S)	64	Reserved
42E0_0000 (NS) 52E0_0000 (S)	42E0_FFFF (NS) 52E0_FFFF (S)	64	LPADC2
42DF_0000 (NS) 52DF_0000 (S)	42DF_FFFF (NS) 52DF_FFFF (S)	64	CMP4
42DE_0000 (NS) 52DE_0000 (S)	42DE_FFFF (NS) 52DE_FFFF (S)	64	CMP3
42DD_0000 (NS) 52DD_0000 (S)	42DD_FFFF (NS) 52DD_FFFF (S)	64	CMP2
42DC_0000 (NS) 52DC_0000 (S)	42DC_FFFF (NS) 52DC_FFFF (S)	64	CMP1
42DB_0000 (NS) 52DB_0000 (S)	42DB_FFFF (NS) 52DB_FFFF (S)	64	Reserved
42DA_0000 (NS) 52DA_0000 (S)	42DA_FFFF (NS) 52DA_FFFF (S)	64	LPUART8
42D9_0000 (NS) 52D9_0000 (S)	42D9_FFFF (NS) 52D9_FFFF (S)	64	LPUART11
42D8_0000 (NS) 52D8_0000 (S)	42D8_FFFF (NS) 52D8_FFFF (S)	64	LPUART10
42D7_0000 (NS) 52D7_0000 (S)	42D7_FFFF (NS) 52D7_FFFF (S)	64	LPUART9
42D6_0000 (NS) 52D6_0000 (S)	42D6_FFFF (NS) 52D6_FFFF (S)	64	LPSPI6
42D5_0000 (NS)	42D5_FFFF (NS)	64	LPSPI5

Table continues on the next page...

Table 8. AIPS-3 memory map (continued)

Start Address	End Address	Size	NIC Port
52D5_0000 (S)	52D5_FFFF (S)		
42D4_0000 (NS) 52D4_0000 (S)	42D4_FFFF (NS) 52D4_FFFF (S)	64	LPI2C6
42D3_0000 (NS) 52D3_0000 (S)	42D3_FFFF (NS) 52D3_FFFF (S)	64	LPI2C5
42CE_0000 (NS) 52CE_0000 (S)	42D2_FFFF (NS) 52D2_FFFF (S)	320	Reserved
42CD_0000 (NS) 52CD_0000 (S)	42CD_FFFF (NS) 52CD_FFFF (S)	64	LPTMR3
42CC_0000 (NS) 52CC_0000 (S)	42CC_FFFF (NS) 52CC_FFFF (S)	64	LPIT3
42CB_0000 (NS) 52CB_0000 (S)	42CB_FFFF (NS) 52CB_FFFF (S)	64	USB_PHY2
42CA_0000 (NS) 52CA_0000 (S)	42CA_FFFF (NS) 52CA_FFFF (S)	64	USBPHY
42C9_0000 (NS) 52C9_0000 (S)	42C9_FFFF (NS) 52C9_FFFF (S)	64	USB_OTG2
42C8_0000 (NS) 52C8_0000 (S)	42C8_FFFF (NS) 52C8_FFFF (S)	64	USB
42C7_0000 (NS) 52C7_0000 (S)	42C7_FFFF (NS) 52C7_FFFF (S)	64	USB_PL301
42C2_0000 (NS) 52C2_0000 (S)	42C6_FFFF (NS) 52C6_FFFF (S)	320	Reserved
42C1_0000 (NS) 52C1_0000 (S)	42C1_FFFF (NS) 52C1_FFFF (S)	64	SINC3
42C0_0000 (NS) 52C0_0000 (S)	42C0_FFFF (NS) 52C0_FFFF (S)	64	SINC2
42BF_0000 (NS) 52BF_0000 (S)	42BF_FFFF (NS) 52BF_FFFF (S)	64	SINC1

Table continues on the next page...

Table 8. AIPS-3 memory map (continued)

Start Address	End Address	Size	NIC Port
42BE_0000 (NS) 52BE_0000 (S)	42BE_FFFF (NS) 52BE_FFFF (S)	64	PDM
42BD_0000 (NS) 52BD_0000 (S)	42BD_FFFF (NS) 52BD_FFFF (S)	64	SAI4
42BC_0000 (NS) 52BC_0000 (S)	42BC_FFFF (NS) 52BC_FFFF (S)	64	SAI3
42BB_0000 (NS) 52BB_0000 (S)	42BB_FFFF (NS) 52BB_FFFF (S)	64	SAI2
42BA_0000 (NS) 52BA_0000 (S)	42BA_FFFF (NS) 52BA_FFFF (S)	64	SPDIF
42A9_0000 (NS) 52A9_0000 (S)	42B9_FFFF (NS) 52B9_FFFF (S)	1088	Reserved
42A8_0000 (NS) 52A8_0000 (S)	42A8_FFFF (NS) 52A8_FFFF (S)	64	ECAT
42A2_0000 (NS) 52A2_0000 (S)	42A7_FFFF (NS) 52A7_FFFF (S)	384	Reserved
42A1_0000 (NS) 52A1_0000 (S)	42A1_FFFF (NS) 52A1_FFFF (S)	64	IOMUXC
42A0_0000 (NS) 52A0_0000 (S)	42A0_FFFF (NS) 52A0_FFFF (S)	64	KPP
429B_0000 (NS) 529B_0000 (S)	429F_FFFF (NS) 529F_FFFF (S)	320	Reserved
429A_0000 (NS) 529A_0000 (S)	429A_FFFF (NS) 529A_FFFF (S)	64	ASRC
4294_0000 (NS) 5294_0000 (S)	4299_FFFF (NS) 5299_FFFF (S)	384	Reserved
4293_0000 (NS) 5293_0000 (S)	4293_FFFF (NS) 5293_FFFF (S)	64	MECC2
4292_0000 (NS)	4292_FFFF (NS)	64	MECC1

Table continues on the next page...

Table 8. AIPS-3 memory map (continued)

Start Address	End Address	Size	NIC Port
5292_0000 (S)	5292_FFFF (S)		
4291_0000 (NS) 5291_0000 (S)	4291_FFFF (NS) 5291_FFFF (S)	64	SEMC
4290_0000 (NS) 5290_0000 (S)	4290_FFFF (NS) 5290_FFFF (S)	64	FLEXSPI_FLR
428F_0000 (NS) 528F_0000 (S)	428F_FFFF (NS) 528F_FFFF (S)	64	MSGINTR6
428E_0000 (NS) 528E_0000 (S)	428E_FFFF (NS) 528E_FFFF (S)	64	MSGINTR5
428D_0000 (NS) 528D_0000 (S)	428D_FFFF (NS) 528D_FFFF (S)	64	MSGINTR4
428C_0000 (NS) 528C_0000 (S)	428C_FFFF (NS) 528C_FFFF (S)	64	MSGINTR3
428B_0000 (NS) 528B_0000 (S)	428B_FFFF (NS) 528B_FFFF (S)	64	MSGINTR2
428A_0000 (NS) 528A_0000 (S)	428A_FFFF (NS) 528A_FFFF (S)	64	MSGINTR1
4289_0000 (NS) 5289_0000 (S)	4289_FFFF (NS) 5289_FFFF (S)	64	MTR_DCA4
4288_0000 (NS) 5288_0000 (S)	4288_FFFF (NS) 5288_FFFF (S)	64	BASIC_TROUT3
4287_0000 (NS) 5287_0000 (S)	4287_FFFF (NS) 5287_FFFF (S)	64	MTR_DCA3
4286_0000 (NS) 5286_0000 (S)	4286_FFFF (NS) 5286_FFFF (S)	64	USDHC2
4285_0000 (NS) 5285_0000 (S)	4285_FFFF (NS) 5285_FFFF (S)	64	USDHC1
4284_0000 (NS) 5284_0000 (S)	4284_FFFF (NS) 5284_FFFF (S)	64	GPV_MEGA2

Table continues on the next page...

Table 8. AIPS-3 memory map (continued)

Start Address	End Address	Size	NIC Port
4283_0000 (NS) 5283_0000 (S)	4283_FFFF (NS) 5283_FFFF (S)	64	GPV_MEGA1
4282_0000 (NS) 5282_0000 (S)	4282_FFFF (NS) 5282_FFFF (S)	64	TRDC-MC3
4281_0000 (NS) 5281_0000 (S)	4281_FFFF (NS) 5281_FFFF (S)	64	TRDC3 (MEGA Domain)
4280_0000 (NS) 5280_0000 (S)	4280_FFFF (NS) 5280_FFFF (S)	64	BLK_CTRL_MEGA

Table 9. GPV address space compression

Address to GPV	Compressed SoC Address Map Base = 4283_0000 (Non-Secure) Base = 5283_0000 (Secure)	Features
0x00000 – 0x11FFF	0x00000 - 0x11FFF	Address control Peripheral ID Master interface 0-15
0x12000 – 0x41FFF	N/A	Not supported: Master interface 16-63
0x42000 – 0x4FFFF	0x12000 – 0x1FFFF	Slave interface 0-13
0x50000 – 0xFFFFF	N/A	Not supported: Slave interface 16-127 Internal interface 0-61

3.7 AIPS-4 Memory Map

The AIPS-4 memory map is not for customer use.

3.8 PPB Memory Map

The table below shows the detailed PPB memory map. There are both Non-Secure (NS) and Secure (S) access possibilities. The addressing is the same for both M7 and M33.

Table 10. PPB memory map

Start Address	End Address	Region	Size (KB)	Allocation
4000_0000 (NS) 5000_0000 (S)	40FF_FFFF (NS) 50FF_FFFF (S)	Debug Access Port	16384	Debug Access Port ROM (DAPROM)

Table continues on the next page...

Table 10. PPB memory map (continued)

Start Address	End Address	Region	Size (KB)	Allocation
4102_0000 (NS) 5102_0000 (S)	4102_FFFF (NS) 5102_FFFF (S)	CoreSight	64	CoreSight Serial Wire Output (CS_SWO)
4103_0000 (NS) 5103_0000 (S)	4103_FFFF (NS) 5103_FFFF (S)		64	CoreSight Embedded Trace FIFO (CS_ETF)
4104_0000 (NS) 5104_0000 (S)	4104_FFFF (NS) 5104_FFFF (S)		64	CoreSight Embedded Trace Router (CS_ETR)
4105_0000 (NS) 5105_0000 (S)	4105_FFFF (NS) 5105_FFFF (S)		64	CoreSight Cross-Trigger Interface 0 (CS_CT0)
4106_0000 (NS) 5106_0000 (S)	4106_FFFF (NS) 5106_FFFF (S)		64	CoreSight Cross-Trigger Interface 1 (CS_CT1)
4108_0000 (NS) 5108_0000 (S)	4107_FFFF (NS) 5107_FFFF (S)	-	128	Reserved

Chapter 4
Interrupt, DMA Events, and XBAR Assignments

4.1 Overview

This section provides information on the assignments of the interrupts, DMA, and XBARs.

4.2 System interrupts

The Nested Vectored Interrupt Controller (NVIC) collects interrupt request sources and provides an interface to the Cortex-M cores. The IRQ Mask is shown below.

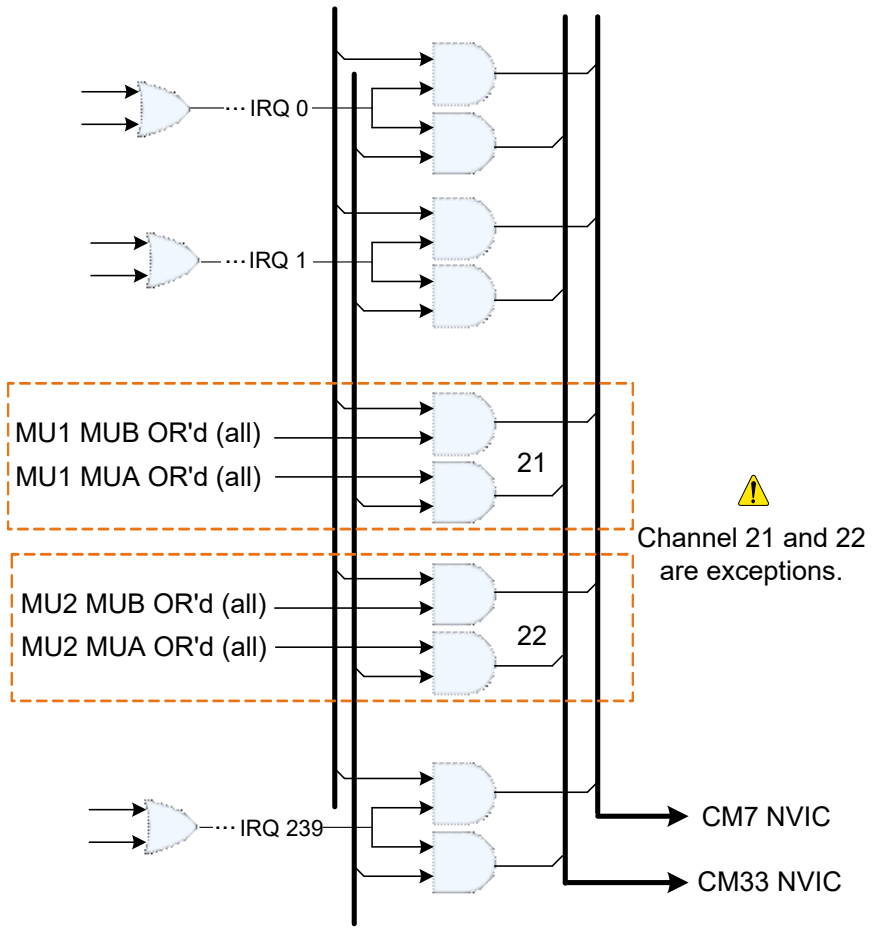


Figure 7. IRQ Mask

The table below describes the interrupt sources:

Table 11. System interrupt summary

IRQ	Module	LOGIC	Description
0	TMR1	OR	TMR Channel 0 interrupt
0		OR	TMR Channel 1 interrupt
0		OR	TMR Channel 2 interrupt

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
0		OR	TMR Channel 3 interrupt
1	DBG_TRACE	OR	DAP Interrupt
2	CM7 Domain	OR	CTI trigger0 outputs from CM7 platform
2		OR	CTI trigger1 outputs from CM7 platform
3	CM33	OR	CTI trigger0 outputs from CM33 platform
3	CM33	OR	CTI trigger1 outputs from CM33 platform
4	TMR5	OR	TMR Channel 0 interrupt
4		OR	TMR Channel 1 interrupt
4		OR	TMR Channel 2 interrupt
4		OR	TMR Channel 3 interrupt
5	TMR6	OR	TMR Channel 0 interrupt
5		OR	TMR Channel 1 interrupt
5		OR	TMR Channel 2 interrupt
5		OR	TMR Channel 3 interrupt
6	TMR7	OR	TMR Channel 0 interrupt
6		OR	TMR Channel 1 interrupt
6		OR	TMR Channel 2 interrupt
6		OR	TMR Channel 3 interrupt
7	TMR8	OR	TMR Channel 0 interrupt
7		OR	TMR Channel 1 interrupt
7		OR	TMR Channel 2 interrupt
7		OR	TMR Channel 3 interrupt
8	CAN-FD1	OR	AON Domain CAN-FD Interrupt from bus-off
8		OR	AON Domain CAN-FD Interrupt from CAN line error
8		OR	AON Domain CAN-FD Ored interrupts from ipi_int_MB
8		OR	AON Domain CAN-FD Rx warning Interrupt
8		OR	AON Domain CAN-FD Tx warning Interrupt
8		OR	AON Domain CAN-FD Interrupt from wake up
8		OR	AON Domain CAN-FD Interrupt from match in PN
8		OR	AON Domain CAN-FD Interrupt from timeout in PN
8		OR	AON Domain CAN-FD bus-off done interrupt
8		OR	AON Domain CAN-FD FD error interrupt

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
8		OR	AON Domain CAN-FD FIFO underflow interrupt
8		OR	AON Domain CAN-FD FIFO overflow interrupt
8		OR	AON Domain CAN-FD FIFO watermarked interrupt
8		OR	AON Domain CAN-FD FIFO data available interrupt
9		OR	AON Domain CAN-FD Correctable error interrupt
9		OR	AON Domain CAN-FD Non correctable error int host
9		OR	AON Domain CAN-FD Non correctable error int internal
10	GPIO1	OR	Interrupt 0
10		OR	Async interrupt 0
11		OR	Interrupt 1
11		OR	Async interrupt 1
12	I3C1	OR	Slave: Wake on SA/DA match in SCL
12		OR	I3C interrupt
13	LPI2C1	OR	I2C interrupt in master mode
13		OR	Async I2C interrupt in master mode
13		OR	I2C interrupt in slave mode
13		OR	Async I2C interrupt in slave mode
14	LPI2C2	OR	I2C interrupt in master mode
14		OR	Async I2C interrupt in master mode
14		OR	I2C interrupt in slave mode
14		OR	Async I2C interrupt in slave mode
15	LPIT1	OR	LPIT global interrupt
16	LPSPI1	OR	Ored LPSPI interrupt
16		OR	Async Ored LPSPI interrupt
17	LPSPI2	OR	Ored LPSPI interrupt
17		OR	Async Ored LPSPI interrupt
18	LPTMR1	OR	LPTMR interrupt
19	LPUART1	OR	TX interrupt
19		OR	Async TX interrupt
19		OR	RX interrupt
19		OR	Async RX interrupt
20	LPUART2	OR	TX interrupt

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
20		OR	Async TX interrupt
20		OR	RX interrupt
20		OR	Async RX interrupt
21	MU1	-	MU Ored of all (tx,rx,gp,core,murip) interrupt to MUB (M7,M33 - to communicate with M7 core)
21		-	MU Ored of all (tx,rx,gp,core,murip) interrupt to MUA (M7,M33 - to communicate with M33 core)
22	MU2	-	MU Ored of all (tx,rx,gp,core,murip) interrupt to MUB (M7, M33 - to communicate with M7 core)
22		-	MU Ored of all (tx,rx,gp,core,murip) interrupt to MUA (M7, M33 - to communicate with M33 core)
23	FLEXPWM1	OR	Reload error interrupt
23		OR	Fault input interrupt
24		OR	Submodule 0 input capture interrupt
24		OR	Submodule 0 compare interrupt
24		OR	Submodule 0 reload interrupt
25		OR	Submodule 1 input capture interrupt
25		OR	Submodule 1 compare interrupt
25		OR	Submodule 1 reload interrupt
26		OR	Submodule 2 input capture interrupt
26		OR	Submodule 2 compare interrupt
26		OR	Submodule 2 reload interrupt
27		OR	Submodule 3 input capture interrupt
27		OR	Submodule 3 compare interrupt
27		OR	Submodule 3 reload interrupt
28	Edgelock Enclave	-	Edgelock Trust MUA RX full interrupt
29		-	Edgelock Trust MUA TX empty interrupt
30		-	Edgelock Apps Core MUA RX full interrupt
31		-	Edgelock Apps Core MUA TX empty interrupt
32		-	Edgelock Realtime Core MUA RX full interrupt
33		-	Edgelock Realtime Core MUA TX empty interrupt
34		-	Edgelock secure interrupt
35		-	Edgelock non-secure interrupt
36	TPM1	OR	TPM interrupt

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
36		OR	Async TPM interrupt
37	TPM2	OR	TPM interrupt
37		OR	Async TPM interrupt
38	WDOG1	OR	Time out interrupt
38		OR	Async time out interrupt
39	WDOG2	OR	Time out interrupt
39		OR	Async time out interrupt
40	TRDC_MGR_AON	-	transfer error interrupt
41	MIC	OR	Hardware Voice Activity Detector event interrupt
41		OR	Async Hardware Voice Activity Detector event interrupt
42		OR	Hardware Voice Activity Detector event interrupt
42		OR	Async Hardware Voice Activity Detector event interrupt
43		OR	FIFO interrupt
43		OR	Async FIFO interrupt
44		OR	FIFO error
44		OR	Async FIFO error
45	Reserved	-	-
46	CM33	OR	CM33 cache 1bit ECC error interrupt
46		OR	CM33 cache 2bit ECC error interrupt
47	CM33_ECC_MCM	-	TCM 1bit ECC error
48		-	TCM 2bit ECC error
49	CM7 Domain	-	CM7 TCM 1bit ECC error
50		-	CM7 TCM 2bit ECC error
51	CAN-FD2	OR	WAKEUP Domain CAN-FD Interrupt from bus-off
51		OR	WAKEUP Domain CAN-FD Interrupt from CAN line error
51		OR	WAKEUP Domain CAN-FD Ored interrupts from ipi_int_MB
51		OR	WAKEUP Domain CAN-FD Rx warning Interrupt
51		OR	WAKEUP Domain CAN-FD Tx warning Interrupt
51		OR	WAKEUP Domain CAN-FD Interrupt from wake up
51		OR	WAKEUP Domain CAN-FD Interrupt from match in PN

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
51		OR	WAKEUP Domain CAN-FD Interrupt from timeout in PN
51		OR	WAKEUP Domain CAN-FD bus-off done interrupt
51		OR	WAKEUP Domain CAN-FD FD error interrupt
51		OR	WAKEUP Domain CAN-FD FIFO underflow interrupt
51		OR	WAKEUP Domain CAN-FD FIFO overflow interrupt
51		OR	WAKEUP Domain CAN-FD FIFO watermarked interrupt
51		OR	WAKEUP Domain CAN-FD FIFO data available interrupt
52		OR	WAKEUP Domain CAN-FD Correctable error interrupt
52		OR	WAKEUP Domain CAN-FD Non correctable error int host
52		OR	WAKEUP Domain CAN-FD Non correctable error int internal
53	FLEXIO1	OR	SSF/SEF/TSF/PSF/ETSF flags
53		OR	SSF/SEF/TSF/PSF/ETSF flags in async
54	FLEXIO2	OR	SSF/SEF/TSF/PSF/ETSF flags
54		OR	SSF/SEF/TSF/PSF/ETSF flags in async
55	FLEXSPI1	-	FlexSPI1 (AON Domain) interrupt
56	FLEXSPI2	-	FlexSPI2 (WAKEUP Domain) interrupt
57	GPIO2	OR	Interrupt 0
57		OR	Async interrupt 0
58		OR	Interrupt 1
58		OR	Async interrupt 1
59	GPIO3	OR	Interrupt 0
59		OR	Async interrupt 0
60		OR	Interrupt 1
60		OR	Async interrupt 1
61	I3C2	OR	Slave: Wake on SA/DA match in SCL
61		OR	I3C interrupt
62	LPI2C3	OR	I2C interrupt in master mode
62		OR	Async I2C interrupt in master mode
62		OR	I2C interrupt in slave mode

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
62		OR	Async I2C interrupt in slave mode
64	Reserved	-	-
64	LPIT2	-	LPIT global interrupt
65	LPSPI3	OR	Ored LPSPI interrupt
65		OR	Async Ored LPSPI interrupt
66	LPSPI4	OR	Ored LPSPI interrupt
66		OR	Async Ored LPSPI interrupt
67	LPTMR2	-	LPTMR interrupt
68	LPUART3	OR	TX interrupt
68		OR	Async TX interrupt
68		OR	RX interrupt
68		OR	Async RX interrupt
69	LPUART4	OR	TX interrupt
69		OR	Async TX interrupt
69		OR	RX interrupt
69		OR	Async RX interrupt
70	LPUART5	OR	TX interrupt
70		OR	Async TX interrupt
70		OR	RX interrupt
70		OR	Async RX interrupt
71	LPUART6	OR	TX interrupt
71		OR	Async TX interrupt
71		OR	RX interrupt
71		OR	Async RX interrupt
73	BBNSM	OR	WAKEUP Alarm
73		OR	RTC Alarm
73		OR	Interrupt to request power on
73		OR	Interrupt to request power off
74	SYS_CTR1	OR	System Counter compare interrupt 0
74		OR	System counter compare interrupt 1
75	TPM3	OR	TPM interrupt
75		OR	Async TPM interrupt

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
76	TPM4	OR	TPM interrupt
76		OR	Async TPM interrupt
77	TPM5	OR	TPM interrupt
77		OR	Async TPM interrupt
78	TPM6	OR	TPM interrupt
78		OR	Async TPM interrupt
79	WDOG3	OR	Time out interrupt
79		OR	Async time out interrupt
80	WDOG4	OR	Time out interrupt
80		OR	Async time out interrupt
81	WDOG5	OR	Time out interrupt
81		OR	Async time out interrupt
82	TRDC_MGR_WKUP	-	transfer error interrupt
83	TMPSNS	-	temperature alarm
84	BBSM	-	WAKEUP Alarm
85	LDO	-	Brown out
86	USDHC1	-	uSDHC interrupt
87	USDHC2	-	uSDHC interrupt
88	TRDC_MGR_MEGA	-	transfer error interrupt
89	SFA	OR	Signal Frequency Analyzer interrupt
89	TMR2	OR	TMR Channel 1 interrupt
89		OR	TMR Channel 2 interrupt
89		OR	TMR Channel 3 interrupt
90	LDO	-	Brown out
91	MECC1	OR	Normal interrupt
91		OR	Fatal interrupt when 'multiple bits error' is detected for read
92	MECC2	OR	Normal interrupt
92		OR	Fatal interrupt when 'multiple bits error' is detected for read
93	ADC1	OR	Async interrupt for STOP mode wake up
93		OR	Trigger Completion or High Priority Trigger Exception
94	eDMA3	-	AON Domain eDMA error interrupt

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
95		-	AON Domain eDMA channel 0 interrupt
96		-	AON Domain eDMA channel 1 interrupt
97		-	AON Domain eDMA channel 2 interrupt
98		-	AON Domain eDMA channel 3 interrupt
99		-	AON Domain eDMA channel 4 interrupt
100		-	AON Domain eDMA channel 5 interrupt
101		-	AON Domain eDMA channel 6 interrupt
102		-	AON Domain eDMA channel 7 interrupt
103		-	AON Domain eDMA channel 8 interrupt
104		-	AON Domain eDMA channel 9 interrupt
105		-	AON Domain eDMA channel 10 interrupt
106		-	AON Domain eDMA channel 11 interrupt
107		-	AON Domain eDMA channel 12 interrupt
108		-	AON Domain eDMA channel 13 interrupt
109		-	AON Domain eDMA channel 14 interrupt
110		-	AON Domain eDMA channel 15 interrupt
111		-	AON Domain eDMA channel 16 interrupt
112		-	AON Domain eDMA channel 17 interrupt
113		-	AON Domain eDMA channel 18 interrupt
114		-	AON Domain eDMA channel 19 interrupt
115		-	AON Domain eDMA channel 20 interrupt
116		-	AON Domain eDMA channel 21 interrupt
117		-	AON Domain eDMA channel 22 interrupt
118		-	AON Domain eDMA channel 23 interrupt
119		-	AON Domain eDMA channel 24 interrupt
120		-	AON Domain eDMA channel 25 interrupt
121		-	AON Domain eDMA channel 26 interrupt
122		-	AON Domain eDMA channel 27 interrupt
123		OR	AON Domain eDMA channel 28 interrupt
124		OR	AON Domain eDMA channel 29 interrupt
125		OR	AON Domain eDMA channel 30 interrupt
126		OR	AON Domain eDMA channel 31 interrupt

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
127	eDMA4	-	WAKEUP Domain eDMA error interrupt
128		OR	WAKEUP Domain eDMA channel 0/1/32/33 interrupt
129		OR	WAKEUP Domain eDMA channel 2/3/34/35 interrupt
130		OR	WAKEUP Domain eDMA channel 4/5/36/37 interrupt
131		OR	WAKEUP Domain eDMA channel 6/7/38/39 interrupt
132		OR	WAKEUP Domain eDMA channel 8/9/40/41 interrupt
133		OR	WAKEUP Domain eDMA channel 10/11/42/43 interrupt
134		OR	WAKEUP Domain eDMA channel 12/13/44/45 interrupt
135		OR	WAKEUP Domain eDMA channel 14/15/46/47 interrupt
136		OR	WAKEUP Domain eDMA channel 16/17/48/49 interrupt
137		OR	WAKEUP Domain eDMA channel 18/19/50/51 interrupt
138		OR	WAKEUP Domain eDMA channel 20/21/52/53 interrupt
139		OR	WAKEUP Domain eDMA channel 22/23/54/55 interrupt
140		OR	WAKEUP Domain eDMA channel 24/25/56/57 interrupt
141		OR	WAKEUP Domain eDMA channel 26/27/58/59 interrupt
142		OR	WAKEUP Domain eDMA channel 28/29/60/61 interrupt
143		OR	WAKEUP Domain eDMA channel 30/31/62/63 interrupt
144	XBAR1	OR	XBAR1 channel 0 interrupt
144		OR	XBAR1 channel 1 interrupt
145		OR	XBAR1 channel 2 interrupt
145		OR	XBAR channel 3 interrupt
146	SINC3	OR	SINC Filter channel 0/1/2/3 interrupt
147	EWM	OR	An interrupt request to indicate the assertion of the EWM reset out signal
148	Reserved	-	-
149	LPIT	OR	LPIT global interrupt
150	LPTMR3	OR	LPTMR interrupt
151	TMR4	OR	TMR channel 0/1/2/3 interrupt
151		OR	TMR channel 1 interrupt
151		OR	TMR channel 2 interrupt
151		OR	TMR channel 3 interrupt
152	Reserved	-	-

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
153	LPI2C6	OR	I2C interrupt in master mode
153		OR	Async I2C interrupt in master mode
153		OR	I2C interrupt in slave mode
153		OR	Async I2C interrupt in slave mode
154	Reserved	-	-
155	Reserved	-	-
156-159	Reserved	-	-
160	INTG_BOOTROM_DEBUG_CTRL	-	CM33, CM7, DAP Access IRQ
161	EDGELOCK	-	Edgelock reset source if no Edgelock Enclave reference clock is detected. Output synchronized to 32khz clk.
162		-	Edgelock reset source Edgelock Enclave reference clock is not detected or too slow. Output synchronized to ref1_clk.
163		-	Edgelock reset source Edgelock Enclave reference clock is not detected or too slow. Output synchronized to ref2_clk.
164	TMR3	OR	TMR Channel 0 interrupt
164		OR	TMR Channel 1 interrupt
164		OR	TMR Channel 2 interrupt
164		OR	TMR Channel 3 interrupt
165	JTAGC	OR	JTAGC SRC reset source
166	CM33	-	CM33 SYSREQRST SRC reset source
167		-	CM33 LOCKUP SRC reset source
168	CM7 Domain	-	CM7 SYSREQRST SRC reset source
169		-	CM7 LOCKUP SRC reset source
170	FLEXPWM2	OR	Reload error interrupt
170		OR	Fault input interrupt
171		OR	Submodule 0 input capture interrupt
171		OR	Submodule 0 compare interrupt
171		OR	Submodule 0 reload interrupt
172		OR	Submodule 1 input capture interrupt
172		OR	Submodule 1 compare interrupt
172		OR	Submodule 1 reload interrupt

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
173		OR	Submodule 2 input capture interrupt
173		OR	Submodule 2 compare interrupt
173		OR	Submodule 2 reload interrupt
174		OR	Submodule 3 input capture interrupt
174		OR	Submodule 3 compare interrupt
174		OR	Submodule 3 reload interrupt
175	FLEXPWM3	OR	Reload error interrupt
175		OR	Fault input interrupt
176		OR	Submodule 0 input capture interrupt
176		OR	Submodule 0 compare interrupt
176		OR	Submodule 0 reload interrupt
177		OR	Submodule 1 input capture interrupt
177		OR	Submodule 1 compare interrupt
177		OR	Submodule 1 reload interrupt
178		OR	Submodule 2 input capture interrupt
178		OR	Submodule 2 compare interrupt
178		OR	Submodule 2 reload interrupt
179		OR	Submodule 3 input capture interrupt
179		OR	Submodule 3 compare interrupt
179		OR	Submodule 3 reload interrupt
180	FLEXPWM4	OR	Reload error interrupt
180		OR	Fault input interrupt
181		OR	Submodule 0 input capture interrupt
181		OR	Submodule 0 compare interrupt
181		OR	Submodule 0 reload interrupt
182		OR	Submodule 1 input capture interrupt
182		OR	Submodule 1 compare interrupt
182		OR	Submodule 1 reload interrupt
183		OR	Submodule 2 input capture interrupt
183		OR	Submodule 2 compare interrupt
183		OR	Submodule 2 reload interrupt
184		OR	Submodule 3 input capture interrupt

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
184		OR	Submodule 3 compare interrupt
184		OR	Submodule 3 reload interrupt
185	eQDC1 [1]	OR	INDEX/PRESET signal transition interrupt (ipi_int_index)
185		OR	HOME/ENABLE signal transition interrupt (ipi_int_home)
185		OR	Watchdog timeout interrupt (ipi_int_wdog)
185		OR	Reserved
185		OR	Simultaneous PHASEA and PHASEB change interrupt (ipi_int_sab)
185		OR	Count direction change interrupt (ipi_int_dir)
185		OR	Roll-over interrupt (ipi_int_ro)
185		OR	Roll-under interrupt (ipi_int_ru)
186	eQDC2[1]	OR	INDEX/PRESET signal transition interrupt (ipi_int_index)
186		OR	HOME/ENABLE signal transition interrupt (ipi_int_home)
186		OR	Watchdog timeout interrupt (ipi_int_wdog)
186		OR	Reserved
186		OR	Simultaneous PHASEA and PHASEB change interrupt (ipi_int_sab)
186		OR	Count direction change interrupt (ipi_int_dir)
186		OR	Roll-over interrupt (ipi_int_ro)
186		OR	Roll-under interrupt (ipi_int_ru)
187	eQDC3[1]	OR	INDEX/PRESET signal transition interrupt (ipi_int_index)
187		OR	HOME/ENABLE signal transition interrupt (ipi_int_home)
187		OR	Watchdog timeout interrupt (ipi_int_wdog)
187		OR	Reserved
187		OR	Simultaneous PHASEA and PHASEB change interrupt (ipi_int_sab)
187		OR	Count direction change interrupt (ipi_int_dir)
187		OR	Roll-over interrupt (ipi_int_ro)
187		OR	Roll-under interrupt (ipi_int_ru)

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
188	eQDC4[1]	OR	INDEX/PRESET signal transition interrupt (ipi_int_index)
188		OR	HOME/ENABLE signal transition interrupt (ipi_int_home)
188		OR	Watchdog timeout interrupt (ipi_int_wdog)
188		OR	Reserved
188		OR	Simultaneous PHASEA and PHASEB change interrupt (ipi_int_sab)
188		OR	Count direction change interrupt (ipi_int_dir)
188		OR	Roll-over interrupt (ipi_int_ro)
188		OR	Roll-under interrupt (ipi_int_ru)
189	ADC2	OR	Async interrupt for STOP mode wake up
189		OR	Trigger Completion or High Priority Trigger Exception
190	DCDC	OR	Brown out
191	CAN-FD3	OR	WAKEUP Domain CAN-FD Interrupt from bus-off
191		OR	WAKEUP Domain CAN-FD Interrupt from CAN line error
191		OR	WAKEUP Domain CAN-FD Ored interrupts from ipi_int_MB
191		OR	WAKEUP Domain CAN-FD Rx warning Interrupt
191		OR	WAKEUP Domain CAN-FD Tx warning Interrupt
191		OR	WAKEUP Domain CAN-FD Interrupt from wake up
191		OR	WAKEUP Domain CAN-FD Interrupt from match in PN
191		OR	WAKEUP Domain CAN-FD Interrupt from timeout in PN
191		OR	WAKEUP Domain CAN-FD bus-off done interrupt
191		OR	WAKEUP Domain CAN-FD FD error interrupt
191		OR	WAKEUP Domain CAN-FD FIFO underflow interrupt
191		OR	WAKEUP Domain CAN-FD FIFO overflow interrupt
191		OR	WAKEUP Domain CAN-FD FIFO watermarked interrupt
191		OR	WAKEUP Domain CAN-FD FIFO data available interrupt
192		OR	WAKEUP Domain CAN-FD Correctable error interrupt
192		OR	WAKEUP Domain CAN-FD Non correctable error int host

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
192		OR	WAKEUP Domain CAN-FD Non correctable error int internal
193	DAC	OR	DAC interrupt
193		OR	DAC async interrupt
194	LPSPI5	OR	Ored LPSPI interrupt
194		OR	Async Ored LPSPI interrupt
195	LPSPI6	OR	Ored LPSPI interrupt
195		OR	Async Ored LPSPI interrupt
196	LPUART7	OR	TX interrupt
196		OR	Async TX interrupt
196		OR	RX interrupt
196		OR	Async RX interrupt
197	LPUART8	OR	TX interrupt
197		OR	Async TX interrupt
197		OR	RX interrupt
197		OR	Async RX interrupt
198	-	OR	XCEL
199	Reserved	-	-
200	CMP1	OR	Comparator Interrupt
200		OR	Async Comparator Interrupt for STOP mode wake up
201	CMP2	OR	Comparator Interrupt
201		OR	Async Comparator Interrupt for STOP mode wake up
202	CMP3	OR	Comparator Interrupt
202		OR	Async Comparator Interrupt for STOP mode wake up
203	CMP4	OR	Comparator Interrupt
203		OR	Async Comparator Interrupt for STOP mode wake up
204	CM7 Domain	OR	CM7 Cache 1bit ECC error
204		OR	CM7 Cache 2bit ECC error
205		-	CM7 MCM interrupt
206	CM33	-	CM33 MCM interrupt
207	Reserved	-	Reserved
208	SAFETY_CLK_MON	-	Safety clock monitor interrupt
209	GPT1	-	Ored interrupt

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
210	GPT2	-	Ored interrupt
211	KPP	-	Keypad interrupt
212	USBPHY1	OR	Wakeup Interrupt
212		OR	USBPHY interrupt
213	USBPHY2	OR	Wakeup Interrupt
213		OR	USBPHY interrupt
214	USB	-	USB interrupt
215		-	USB interrupt
216	FlexSPI_FLR	-	FlexSPI follower interrupt
217	NETC	-	NETC PCI INTA
218	MSGINTR1	OR	MSIR0 interrupt
218		OR	MSIR1 interrupt
218		OR	MSIR2 interrupt
219	MSGINTR2	OR	MSIR0 interrupt
219		OR	MSIR1 interrupt
219		OR	MSIR2 interrupt
220	MSGINTR3	OR	MSIR0 interrupt
220		OR	MSIR1 interrupt
220		OR	MSIR2 interrupt
221	MSGINTR4	OR	MSIR0 interrupt
221		OR	MSIR1 interrupt
221		OR	MSIR2 interrupt
222	MSGINTR5	OR	MSIR0 interrupt
222		OR	MSIR1 interrupt
222		OR	MSIR2 interrupt
223	MSGINTR6	OR	MSIR0 interrupt
223		OR	MSIR1 interrupt
223		OR	MSIR2 interrupt
224	SINC1	-	Ored channel 0 interrupt
225		-	Ored channel 1 interrupt
226		-	Ored channel 2 interrupt
227		-	Ored channel 3 interrupt

Table continues on the next page...

Table 11. System interrupt summary (continued)

IRQ	Module	LOGIC	Description
228	SINC2	-	Ored channel 0 interrupt
229		-	Ored channel 1 interrupt
230		-	Ored channel 2 interrupt
231		-	Ored channel 3 interrupt
232	GPIO4	OR	Interrupt 0
232		OR	Async interrupt 0
232		OR	Interrupt 1
232		OR	Asyc interrupt 1
233	TMR2	OR	TMR channel 0 interrupt
233		OR	TMR channel 1 interrupt
233		OR	TMR channel 2 interrupt
233		OR	TMR channel 3 interrupt
234	GPIO5	OR	Interrupt 0
234		OR	Async interrupt 0
234		OR	Interrupt 1
234		OR	Async interrupt 1
235	Reserved		-
236	GPIO6	OR	Interrupt 0
236		OR	Async interrupt 0
236		OR	Interrupt 1
236		OR	Async interrupt 1
237	DBG_TRACE		JTAGSW DAP MDM-AP SRC reset source
238	Reserved	-	-
239	EdgeLock	-	EdgeLock interrupt

1. See the eQDC chapter, Interrupts section for more info.

4.3 DMA Mux

The tables below shows the DMA request signals for the peripherals in the chip.

4.3.1 eDMA3 Mux Mapping

Table 12. eDMA3 Mux Mapping - AON Domain

DMA	Module	Logic	Description
0	-	-	-

Table continues on the next page...

Table 12. eDMA3 Mux Mapping - AON Domain (continued)

DMA	Module	Logic	Description
1	CAN-FD1	-	CAN-FD1 DMA request and response
2	-	-	-
3	GPIO1	-	GPIO1 channel 0 DMA request and response
4	GPIO1	-	GPIO1 channel 1 DMA request and response
5	I3C1	-	I3C1 "To-bus" DMA request and response
6	I3C1	-	I3C1 "from-bus" DMA request and response
7	LPI2C1	OR	LPI2C1 Master TX DMA request
7	LPI2C1	OR	LPI2C1 Slave TX DMA request
8	LPI2C1	OR	LPI2C1 Master RX DMA request
8	LPI2C1	OR	LPI2C1 Slave RX DMA request
9	LPI2C2	OR	LPI2C2 Master TX DMA request
9	LPI2C2	OR	LPI2C2 Slave TX DMA request
10	LPI2C2	OR	LPI2C2 Master RX DMA request
10	LPI2C2	OR	LPI2C2 Slave RX DMA request
11	LPSPI1	-	LPSPI1 TX DMA request
12	LPSPI1	-	LPSPI1 RX DMA request
13	LPSPI2	-	LPSPI2 TX DMA request
14	LPSPI2	-	LPSPI2 RX DMA request
15	LPTMR1	-	LPTIMER1 DMA request and response
16	LPUART1	-	LPUART1 DMA TX request
17	LPUART1	-	LPUART1 DMA RX request and response
18	LPUART2	-	LPUART2 DMA TX request
19	LPUART2	-	LPUART2 DMA RX request and response
20	EDGELOCK	-	Edgelock enclave DMA request and response
21-22	Reserved	-	-
23	TPM1	OR	TPM1 DMA Channel 0 request and response
23	TPM1	OR	TPM1 DMA Channel 2 request and response
24	TPM1	OR	TPM1 DMA Channel 1 request and response
24	TPM1	OR	TPM1 DMA Channel 3 request and response
25	TPM1	-	TPM1 DMA Overflow request and response
26	TPM2	OR	TPM2 DMA Channel 0 request and response
26	TPM2	OR	TPM2 DMA Channel 2 request and response

Table continues on the next page...

Table 12. eDMA3 Mux Mapping - AON Domain (continued)

DMA	Module	Logic	Description
27	TPM2	OR	TPM2 DMA Channel 1 request and response
27	TPM2	OR	TPM2 DMA Channel 3 request and response
28	TPM2	-	TPM2 DMA Overflow request and response
29	LPUART7	-	LPUART7 DMA TX request
30	LPUART7	-	LPUART7 DMA RX request and response
31-32	Reserved	-	-
33	FLEXSPI2	-	FLEXSPI2 DMA TX request
34	FLEXSPI2	-	FLEXSPI2 DMA RX request
35-37	-	-	-
38	CAN3	-	CAN-FD 3 DMA request and response

4.3.2 eDMA4 Mux Mapping

Table 13. eDMA4 Mux Mapping - WAKEUP Domain

DMA	Module	Logic	Description
0	-	-	-
2	GPIO2	-	GPIO2 channel 0 DMA request and response
3	GPIO2	-	GPIO2 channel 1 DMA request and response
4	GPIO3	-	GPIO3 channel 0 DMA request and response
5	GPIO3	-	GPIO3 channel 1 DMA request and response
6	I3C2	-	I3C2 "To-bus" DMA request and response
7	I3C2	-	I3C2 "from-bus" DMA request and response
8	LPI2C3	OR	LPI2C3 Master TX DMA request
8	LPI2C3	OR	LPI2C3 Slave TX DMA request
9	LPI2C3	OR	LPI2C3 Master RX DMA request
9	LPI2C3	OR	LPI2C3 Slave RX DMA request
10	Reserved	OR	Reserved
10	Reserved	OR	Reserved
11	Reserved	OR	Reserved
11	Reserved	OR	Reserved
12	LPSPI3	-	LPSPI3 TX DMA request
13	LPSPI3	-	LPSPI3 RX DMA request
14	LPSPI4	-	LPSPI4 TX DMA request
15	LPSPI4	-	LPSPI4 RX DMA request

Table continues on the next page...

Table 13. eDMA4 Mux Mapping - WAKEUP Domain (continued)

DMA	Module	Logic	Description
16	LPTMR2	-	LPTIMER2 DMA request and response
17	LPUART3	-	LPUART3 DMA TX request
18	LPUART3	-	LPUART3 DMA RX request and response
19	LPUART4	-	LPUART4 DMA TX request
20	LPUART4	-	LPUART4 DMA RX request and response
21	LPUART5	-	LPUART5 DMA TX request
22	LPUART5	-	LPUART5 DMA RX request and response
23	LPUART6	-	LPUART6 DMA TX request
24	LPUART6	-	LPUART6 DMA RX request and response
25	TPM3	OR	TPM3 DMA Channel 0 request and response
25	TPM3	OR	TPM3 DMA Channel 2 request and response
26	TPM3	OR	TPM3 DMA Channel 1 request and response
26	TPM3	OR	TPM3 DMA Channel 3 request and response
27	TPM3	-	TPM3 DMA Overflow request and response
28	TPM4	OR	TPM4 DMA Channel 0 request and response
28	TPM4	OR	TPM4 DMA Channel 2 request and response
29	TPM4	OR	TPM4 DMA Channel 1 request and response
29	TPM4	OR	TPM4 DMA Channel 3 request and response
30	TPM4	-	TPM4 DMA Overflow request and response
31	TPM5	OR	TPM5 DMA Channel 0 request and response
31	TPM5	OR	TPM5 DMA Channel 2 request and response
32	TPM5	OR	TPM5 DMA Channel 1 request and response
32	TPM5	OR	TPM5 DMA Channel 3 request and response
33	TPM5	-	TPM5 DMA Overflow request and response
34	TPM6	OR	TPM6 DMA Channel 0 request and response
34	TPM6	OR	TPM6 DMA Channel 2 request and response
35	TPM6	OR	TPM6 DMA Channel 1 request and response
35	TPM6	OR	TPM6 DMA Channel 3 request and response
36	TPM6	-	TPM6 DMA Overflow request and response
37	FLEXIO1	OR	FLEXIO1 Shifter DMA Channel 0 request
37	FLEXIO1	OR	FLEXIO1 Timer DMA Channel 0 request
38	FLEXIO1	OR	FLEXIO1 Shifter DMA Channel 1 request

Table continues on the next page...

Table 13. eDMA4 Mux Mapping - WAKEUP Domain (continued)

DMA	Module	Logic	Description
38	FLEXIO1	OR	FLEXIO1 Timer DMA Channel 1 request
39	FLEXIO1	OR	FLEXIO1 Shifter DMA Channel 2 request
39	FLEXIO1	OR	FLEXIO1 Timer DMA Channel 2 request
40	FLEXIO1	OR	FLEXIO1 Shifter DMA Channel 3 request
40	FLEXIO1	OR	FLEXIO1 Timer DMA Channel 3 request
41	FLEXIO1	OR	FLEXIO1 Shifter DMA Channel 4 request
41	FLEXIO1	OR	FLEXIO1 Timer DMA Channel 4 request
42	FLEXIO1	OR	FLEXIO1 Shifter DMA Channel 5 request
42	FLEXIO1	OR	FLEXIO1 Timer DMA Channel 5 request
43	FLEXIO1	OR	FLEXIO1 Shifter DMA Channel 6 request
43	FLEXIO1	OR	FLEXIO1 Timer DMA Channel 6 request
44	FLEXIO1	OR	FLEXIO1 Shifter DMA Channel 7 request
44	FLEXIO1	OR	FLEXIO1 Timer DMA Channel 7 request
45	FLEXIO2	OR	FLEXIO2 Shifter DMA Channel 0 request
45	FLEXIO2	OR	FLEXIO2 Timer DMA Channel 0 request
46	FLEXIO2	OR	FLEXIO2 Shifter DMA Channel 1 request
46	FLEXIO2	OR	FLEXIO2 Timer DMA Channel 1 request
47	FLEXIO2	OR	FLEXIO2 Shifter DMA Channel 2 request
47	FLEXIO2	OR	FLEXIO2 Timer DMA Channel 2 request
48	FLEXIO2	OR	FLEXIO2 Shifter DMA Channel 3 request
48	FLEXIO2	OR	FLEXIO2 Timer DMA Channel 3 request
49	FLEXIO2	OR	FLEXIO2 Shifter DMA Channel 4 request
49	FLEXIO2	OR	FLEXIO2 Timer DMA Channel 4 request
50	FLEXIO2	OR	FLEXIO2 Shifter DMA Channel 5 request
50	FLEXIO2	OR	FLEXIO2 Timer DMA Channel 5 request
51	FLEXIO2	OR	FLEXIO2 Shifter DMA Channel 6 request
51	FLEXIO2	OR	FLEXIO2 Timer DMA Channel 6 request
52	FLEXIO2	OR	FLEXIO2 Shifter DMA Channel 7 request
52	FLEXIO2	OR	FLEXIO2 Timer DMA Channel 7 request
53	FLEXSPI1	-	FLEXSPI1 DMA TX request
54	FLEXSPI1	-	FLEXSPI1 DMA RX request
55	-	-	-

Table continues on the next page...

Table 13. eDMA4 Mux Mapping - WAKEUP Domain (continued)

DMA	Module	Logic	Description
56	-	-	-
57	ADC1	-	ADC FIFO0 DMA request
58	FLEXPWM1	-	FLEXPWM1 Read DMA request for capture regs of sub-module0
59	FLEXPWM1	-	FLEXPWM1 Read DMA request for capture regs of sub-module1
60	FLEXPWM1	-	FLEXPWM1 Read DMA request for capture regs of sub-module2
61	FLEXPWM1	-	FLEXPWM1 Read DMA request for capture regs of sub-module3
62	FLEXPWM1	-	FLEXPWM1 Write DMA request for value regs of sub-module0
63	FLEXPWM1	-	FLEXPWM1 Write DMA request for value regs of sub-module1
64	FLEXPWM1	-	FLEXPWM1 Write DMA request for value regs of sub-module2
65	FLEXPWM1	-	FLEXPWM1 Write DMA request for value regs of sub-module3
66	FLEXPWM2	-	FLEXPWM2 Read DMA request for capture regs of sub-module0
67	FLEXPWM2	-	FLEXPWM2 Read DMA request for capture regs of sub-module1
68	FLEXPWM2	-	FLEXPWM2 Read DMA request for capture regs of sub-module2
69	FLEXPWM2	-	FLEXPWM2 Read DMA request for capture regs of sub-module3
70	FLEXPWM2	-	FLEXPWM2 Write DMA request for value regs of sub-module0
71	FLEXPWM2	-	FLEXPWM2 Write DMA request for value regs of sub-module1
72	FLEXPWM2	-	FLEXPWM2 Write DMA request for value regs of sub-module2
73	FLEXPWM2	-	FLEXPWM2 Write DMA request for value regs of sub-module3
74	FLEXPWM3	-	FLEXPWM3 Read DMA request for capture regs of sub-module0
75	FLEXPWM3	-	FLEXPWM3 Read DMA request for capture regs of sub-module1

Table continues on the next page...

Table 13. eDMA4 Mux Mapping - WAKEUP Domain (continued)

DMA	Module	Logic	Description
76	FLEXPWM3	-	FLEXPWM3 Read DMA request for capture regs of sub-module2
77	FLEXPWM3	-	FLEXPWM3 Read DMA request for capture regs of sub-module3
78	FLEXPWM3	-	FLEXPWM3 Write DMA request for value regs of sub-module0
79	FLEXPWM3	-	FLEXPWM3 Write DMA request for value regs of sub-module1
80	FLEXPWM3	-	FLEXPWM3 Write DMA request for value regs of sub-module2
81	FLEXPWM3	-	FLEXPWM3 Write DMA request for value regs of sub-module3
82	FLEXPWM4	-	FLEXPWM4 Read DMA request for capture regs of sub-module0
83	FLEXPWM4	-	FLEXPWM4 Read DMA request for capture regs of sub-module1
84	FLEXPWM4	-	FLEXPWM4 Read DMA request for capture regs of sub-module2
85	FLEXPWM4	-	FLEXPWM4 Read DMA request for capture regs of sub-module3
86	FLEXPWM4	-	FLEXPWM4 Write DMA request for value regs of sub-module0
87	FLEXPWM4	-	FLEXPWM4 Write DMA request for value regs of sub-module1
88	FLEXPWM4	-	FLEXPWM4 Write DMA request for value regs of sub-module2
89	FLEXPWM4	-	FLEXPWM4 Write DMA request for value regs of sub-module3
90	TMR1	-	QUAD TIMER1 DMA read request for capt in timer #0
91	TMR1	-	QUAD TIMER1 DMA read request for capt in timer #1
92	TMR1	-	QUAD TIMER1 DMA read request for capt in timer #2
93	TMR1	-	QUAD TIMER1 DMA read request for capt in timer #3
94	TMR1	OR	QUAD TIMER1 DMA write request for cmpld1 in timer #0
94	TMR1	OR	QUAD TIMER1 DMA write request for cmpld2 in timer #1
95	TMR1	OR	QUAD TIMER1 DMA write request for cmpld1 in timer #1
95	TMR1	OR	QUAD TIMER1 DMA write request for cmpld2 in timer #0
96	TMR1	OR	QUAD TIMER1 DMA write request for cmpld1 in timer #2

Table continues on the next page...

Table 13. eDMA4 Mux Mapping - WAKEUP Domain (continued)

DMA	Module	Logic	Description
96	TMR1	OR	QUAD TIMER1 DMA write request for cmpld2 in timer #3
97	TMR1	OR	QUAD TIMER1 DMA write request for cmpld1 in timer #3
97	TMR1	OR	QUAD TIMER1 DMA write request for cmpld2 in timer #2
98	TMR2	-	QUAD TIMER2 DMA read request for capt in timer #0
99	TMR2	-	QUAD TIMER2 DMA read request for capt in timer #1
100	TMR2	-	QUAD TIMER2 DMA read request for capt in timer #2
101	TMR2	-	QUAD TIMER2 DMA read request for capt in timer #3
102	TMR2	OR	QUAD TIMER2 DMA write request for cmpld1 in timer #0
102	TMR2	OR	QUAD TIMER2 DMA write request for cmpld2 in timer #1
103	TMR2	OR	QUAD TIMER2 DMA write request for cmpld1 in timer #1
103	TMR2	OR	QUAD TIMER2 DMA write request for cmpld2 in timer #0
104	TMR2	OR	QUAD TIMER2 DMA write request for cmpld1 in timer #2
104	TMR2	OR	QUAD TIMER2 DMA write request for cmpld2 in timer #3
105	TMR2	OR	QUAD TIMER2 DMA write request for cmpld1 in timer #3
105	TMR2	OR	QUAD TIMER2 DMA write request for cmpld2 in timer #2
106	TMR3	-	QUAD TIMER3 DMA read request for capt in timer #0
107	TMR3	-	QUAD TIMER3 DMA read request for capt in timer #1
108	TMR3	-	QUAD TIMER3 DMA read request for capt in timer #2
109	TMR3	-	QUAD TIMER3 DMA read request for capt in timer #3
110	TMR3	OR	QUAD TIMER3 DMA write request for cmpld1 in timer #0
110	TMR3	OR	QUAD TIMER3 DMA write request for cmpld2 in timer #1
111	TMR3	OR	QUAD TIMER3 DMA write request for cmpld1 in timer #1
111	TMR3	OR	QUAD TIMER3 DMA write request for cmpld2 in timer #0
112	TMR3	OR	QUAD TIMER3 DMA write request for cmpld1 in timer #2
112	TMR3	OR	QUAD TIMER3 DMA write request for cmpld2 in timer #3
113	TMR3	OR	QUAD TIMER3 DMA write request for cmpld1 in timer #3
113	TMR3	OR	QUAD TIMER3 DMA write request for cmpld2 in timer #2
114	TMR4	-	QUAD TIMER4 DMA read request for capt in timer #0
115	TMR4	-	QUAD TIMER4 DMA read request for capt in timer #1
116	TMR4	-	QUAD TIMER4 DMA read request for capt in timer #2
117	TMR4	-	QUAD TIMER4 DMA read request for capt in timer #3
118	TMR4	OR	QUAD TIMER4 DMA write request for cmpld1 in timer #0

Table continues on the next page...

Table 13. eDMA4 Mux Mapping - WAKEUP Domain (continued)

DMA	Module	Logic	Description
118	TMR4	OR	QUAD TIMER4 DMA write request for cmpld2 in timer #1
119	TMR4	OR	QUAD TIMER4 DMA write request for cmpld1 in timer #1
119	TMR4	OR	QUAD TIMER4 DMA write request for cmpld2 in timer #0
120	TMR4	OR	QUAD TIMER4 DMA write request for cmpld1 in timer #2
120	TMR4	OR	QUAD TIMER4 DMA write request for cmpld2 in timer #3
121	TMR4	OR	QUAD TIMER4 DMA write request for cmpld1 in timer #3
121	TMR4	OR	QUAD TIMER4 DMA write request for cmpld2 in timer #2
122	TMR5	-	QUAD TIMER5 DMA read request for capt in timer #0
123	TMR5	-	QUAD TIMER5 DMA read request for capt in timer #1
124	TMR5	-	QUAD TIMER5 DMA read request for capt in timer #2
125	TMR5	-	QUAD TIMER5 DMA read request for capt in timer #3
126	TMR5	OR	QUAD TIMER5 DMA write request for cmpld1 in timer #0
126	TMR5	OR	QUAD TIMER5 DMA write request for cmpld2 in timer #1
127	TMR5	OR	QUAD TIMER5 DMA write request for cmpld1 in timer #1
127	TMR5	OR	QUAD TIMER5 DMA write request for cmpld2 in timer #0
128	TMR5	OR	QUAD TIMER5 DMA write request for cmpld1 in timer #2
128	TMR5	OR	QUAD TIMER5 DMA write request for cmpld2 in timer #3
129	TMR5	OR	QUAD TIMER5 DMA write request for cmpld1 in timer #3
129	TMR5	OR	QUAD TIMER5 DMA write request for cmpld2 in timer #2
130	TMR6	-	QUAD TIMER6 DMA read request for capt in timer #0
131	TMR6	-	QUAD TIMER6 DMA read request for capt in timer #1
132	TMR6	-	QUAD TIMER6 DMA read request for capt in timer #2
133	TMR6	-	QUAD TIMER6 DMA read request for capt in timer #3
134	TMR6	OR	QUAD TIMER6 DMA write request for cmpld1 in timer #0
134	TMR6	OR	QUAD TIMER6 DMA write request for cmpld2 in timer #1
135	TMR6	OR	QUAD TIMER6 DMA write request for cmpld1 in timer #1
135	TMR6	OR	QUAD TIMER6 DMA write request for cmpld2 in timer #0
136	TMR6	OR	QUAD TIMER6 DMA write request for cmpld1 in timer #2
136	TMR6	OR	QUAD TIMER6 DMA write request for cmpld2 in timer #3
137	TMR6	OR	QUAD TIMER6 DMA write request for cmpld1 in timer #3
137	TMR6	OR	QUAD TIMER6 DMA write request for cmpld2 in timer #2
138	TMR7	-	QUAD TIMER7 DMA read request for capt in timer #0

Table continues on the next page...

Table 13. eDMA4 Mux Mapping - WAKEUP Domain (continued)

DMA	Module	Logic	Description
139	TMR7	-	QUAD TIMER7 DMA read request for capt in timer #1
140	TMR7	-	QUAD TIMER7 DMA read request for capt in timer #2
141	TMR7	-	QUAD TIMER7 DMA read request for capt in timer #3
142	TMR7	OR	QUAD TIMER7 DMA write request for cmpld1 in timer #0
142	TMR7	OR	QUAD TIMER7 DMA write request for cmpld2 in timer #1
143	TMR7	OR	QUAD TIMER7 DMA write request for cmpld1 in timer #1
143	TMR7	OR	QUAD TIMER7 DMA write request for cmpld2 in timer #0
144	TMR7	OR	QUAD TIMER7 DMA write request for cmpld1 in timer #2
144	TMR7	OR	QUAD TIMER7 DMA write request for cmpld2 in timer #3
145	TMR7	OR	QUAD TIMER7 DMA write request for cmpld1 in timer #3
145	TMR7	OR	QUAD TIMER7 DMA write request for cmpld2 in timer #2
146	TMR8	-	QUAD TIMER8 DMA read request for capt in timer #0
147	TMR8	-	QUAD TIMER8 DMA read request for capt in timer #1
148	TMR8	-	QUAD TIMER8 DMA read request for capt in timer #2
149	TMR8	-	QUAD TIMER8 DMA read request for capt in timer #3
150	TMR8	OR	QUAD TIMER8 DMA write request for cmpld1 in timer #0
150	TMR8	OR	QUAD TIMER8 DMA write request for cmpld2 in timer #1
151	TMR8	OR	QUAD TIMER8 DMA write request for cmpld1 in timer #1
151	TMR8	OR	QUAD TIMER8 DMA write request for cmpld2 in timer #0
152	TMR8	OR	QUAD TIMER8 DMA write request for cmpld1 in timer #2
152	TMR8	OR	QUAD TIMER8 DMA write request for cmpld2 in timer #3
153	TMR8	OR	QUAD TIMER8 DMA write request for cmpld1 in timer #3
153	TMR8	OR	QUAD TIMER8 DMA write request for cmpld2 in timer #2
154	XBAR1	-	XBAR DMA request 0
155	XBAR1	-	XBAR DMA request 1
156	XBAR1	-	XBAR DMA request 2
157	XBAR1	-	XBAR DMA request 3
158	ADC2	-	ADC FIFO0 DMA request
159	eQDC1	-	eQDC1 DMA request
160	eQDC2	-	eQDC2 DMA request
161	eQDC3	-	eQDC3 DMA request
162	eQDC4	-	eQDC4 DMA request

Table continues on the next page...

Table 13. eDMA4 Mux Mapping - WAKEUP Domain (continued)

DMA	Module	Logic	Description
163	Reserved	OR	Reserved
163	Reserved	OR	Reserved
164	Reserved	OR	Reserved
164	Reserved	OR	Reserved
165	LPI2C6	OR	LPI2C6 Master TX DMA request
165	LPI2C6	OR	LPI2C6 Slave TX DMA request
166	LPI2C6	OR	LPI2C6 Master RX DMA request
166	LPI2C6	OR	LPI2C6 Slave RX DMA request
167	LPSPI5	-	LPSPI5 TX DMA request
168	LPSPI5	-	LPSPI5 RX DMA request
169	LPSPI6	-	LPSPI6 TX DMA request
170	LPSPI6	-	LPSPI6 RX DMA request
171	LPTMR3	-	LPTIMER3 DMA request and response
172-177	Reserved	-	-
178	LPUART8	-	LPUART8 DMA TX request
179	LPUART8	-	LPUART8 DMA RX request and response
180-185	Reserved	-	-
186	DAC	-	DAC DMA request and response
187	CMP1	-	CMP1 DMA request
188	CMP2	-	CMP2 DMA request
189	CMP3	-	CMP3 DMA request
190	CMP4	-	CMP4 DMA request
191-196	Reserved	-	-
197-198	Reserved	-	-
199	PDM	-	PDM DMA request
200	GPIO4	-	GPIO4 channel 0 DMA request and response
201	GPIO4	-	GPIO4 channel 1 DMA request and response
202	GPIO5	-	GPIO5 channel 0 DMA request and response
203	GPIO5	-	GPIO5 channel 1 DMA request and response
204	GPIO6	-	GPIO6 channel 0 DMA request and response
205	GPIO6	-	GPIO6 channel 1 DMA request and response
206	-	-	-

Table continues on the next page...

Table 13. eDMA4 Mux Mapping - WAKEUP Domain (continued)

DMA	Module	Logic	Description
207	-	-	-
208	SINC1	-	Channel 0 DMA request
209	SINC1	-	Channel 1 DMA request
210	SINC1	-	Channel 2 DMA request
211	SINC1	-	Channel 3 DMA request
212	SINC2	-	Channel 0 DMA request
213	SINC2	-	Channel 1 DMA request
214	SINC2	-	Channel 2 DMA request
215	SINC2	-	Channel 3 DMA request
216	SINC3	-	Channel 0 DMA request
217	SINC3	-	Channel 1 DMA request
218	SINC3	-	Channel 2 DMA request
219	SINC3	-	Channel 3 DMA request
220	ADC1	-	ADC FIFO1 DMA request
221	ADC2	-	ADC FIFO1 DMA request

4.4 XBAR Resource Assignments

The figures and tables below show the XBAR topology and resource assignments.

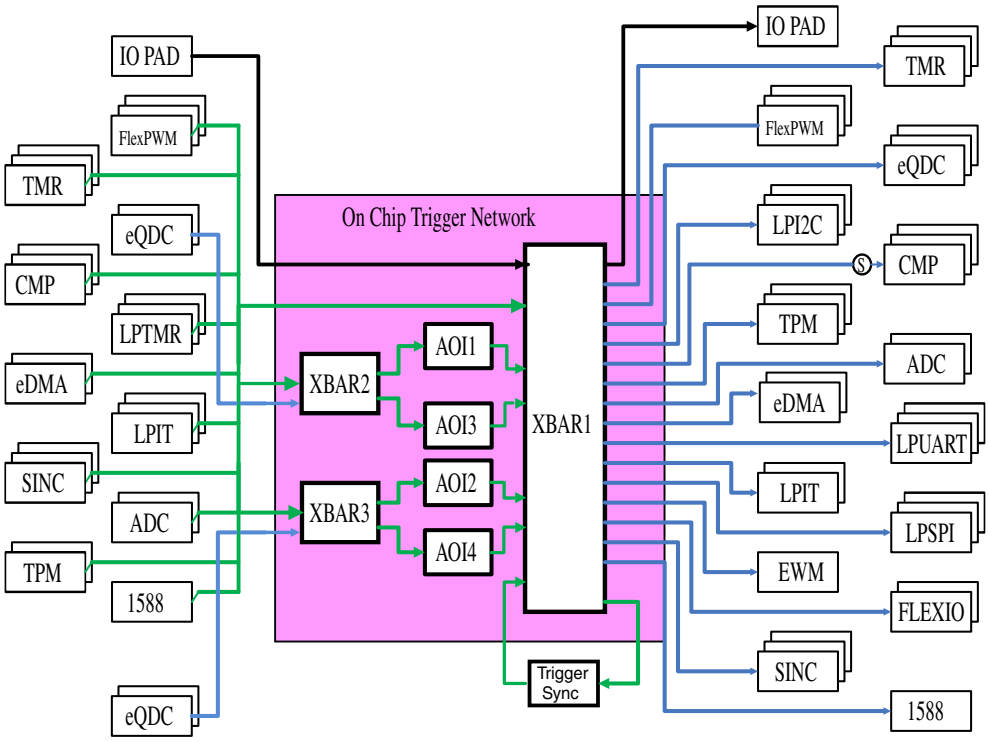


Figure 8. XBAR system diagram

NOTE

The Trigger Sync module can be configured via the blk_ctrl_wakeup XBAR_TRIG_SYNC_CTRL1 and XBAR_TRIG_SYNC_CTRL2 registers.

4.4.1 XBAR Input Assignments

Table 14. XBAR Input Assignments

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN0	-	GND	-
XBAR1	XBAR_IN1	-	1'B1	-
XBAR1	XBAR_IN2	-	GND	-
XBAR1	XBAR_IN3	-	1'B1	-
XBAR1	XBAR_IN4	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_00[MUX_MODE] = 0x1 (XBAR_INOUT04)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN5	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_01[MUX_MODE] = 0x1 (XBAR_INOUT05)	See IOMUX Controller (IOMUXC) for more information

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN6	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_02[MUX_MODE] = 0x1 (XBAR_INOUT06)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN7	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_03[MUX_MODE] = 0x1 (XBAR_INOUT07)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN8	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_04[MUX_MODE] = 0x1 (XBAR_INOUT08)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN9	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_05[MUX_MODE] = 0x1 (XBAR_INOUT09)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN10	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_26[MUX_MODE] = 0x2 (XBAR_INOUT10)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN11	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_27[MUX_MODE] = 0x2 (XBAR_INOUT11)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN12	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_28[MUX_MODE] = 0x2 (XBAR_INOUT12)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN13	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_29[MUX_MODE] = 0x2 (XBAR_INOUT13)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN14	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_16[MUX_MODE] = 0x6 (XBAR_INOUT14) SW_MUX_CTL_PAD_GPIO_EMC_B1_30[MUX_MODE] = 0x2 (XBAR_INOUT14)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN15	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_17[MUX_MODE] = 0x6 (XBAR_INOUT15) SW_MUX_CTL_PAD_GPIO_EMC_B1_39[MUX_MODE] = 0x2 (XBAR_INOUT15)	See IOMUX Controller (IOMUXC) for more information

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN16	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_18[MUX_MODE] = 0x6 (XBAR_INOUT16)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN17	-	SW_MUX_CTL_PAD_GPIO_AD_33[MUX_MODE] = 0x2 (XBAR_INOUT17) SW_MUX_CTL_PAD_GPIO_SD_B2_00[MUX_MODE] = 0x3 (XBAR_INOUT17)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN18	-	SW_MUX_CTL_PAD_GPIO_AD_12[MUX_MODE] = 0x6 (XBAR_INOUT18) SW_MUX_CTL_PAD_GPIO_AD_34[MUX_MODE] = 0x2 (XBAR_INOUT18)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN19	-	SW_MUX_CTL_PAD_GPIO_AD_19[MUX_MODE] = 0x2 (XBAR_INOUT19) SW_MUX_CTL_PAD_GPIO_AD_35[MUX_MODE] = 0x2 (XBAR_INOUT19)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN20	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_00[MUX_MODE] = 0x6 (XBAR_INOUT20) SW_MUX_CTL_PAD_GPIO_SD_B1_00[MUX_MODE] = 0x2 (XBAR_INOUT20)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN21	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_01[MUX_MODE] = 0x6 (XBAR_INOUT21) SW_MUX_CTL_PAD_GPIO_SD_B1_01[MUX_MODE] = 0x2 (XBAR_INOUT21)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN22	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_02[MUX_MODE] = 0x6 (XBAR_INOUT22) SW_MUX_CTL_PAD_GPIO_SD_B1_02[MUX_MODE] = 0x2 (XBAR_INOUT22)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN23	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_03[MUX_MODE] = 0x6 (XBAR_INOUT23) SW_MUX_CTL_PAD_GPIO_SD_B1_03[MUX_MODE] = 0x2 (XBAR_INOUT23)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN24	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_04[MUX_MODE] = 0x6 (XBAR_INOUT24)	See IOMUX Controller (IOMUXC) for more information

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
			SW_MUX_CTL_PAD_GPIO_AD_30[MUX_MODE] = 0x9 (XBAR_INOUT24)	
XBAR1	XBAR_IN25	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_05[MUX_MODE] = 0x6 (XBAR_INOUT25) SW_MUX_CTL_PAD_GPIO_AD_31[MUX_MODE] = 0x9 (XBAR_INOUT25)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN26	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_06[MUX_MODE] = 0x6 (XBAR_INOUT26) SW_MUX_CTL_PAD_GPIO_B1_00[MUX_MODE] = 0x4 (XBAR_INOUT26)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN27	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_07[MUX_MODE] = 0x6 (XBAR_INOUT27) SW_MUX_CTL_PAD_GPIO_B1_01[MUX_MODE] = 0x4 (XBAR_INOUT27)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN28	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_08[MUX_MODE] = 0x6 (XBAR_INOUT28) SW_MUX_CTL_PAD_GPIO_B1_02[MUX_MODE] = 0x4 (XBAR_INOUT28)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN29	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_09[MUX_MODE] = 0x6 (XBAR_INOUT29) SW_MUX_CTL_PAD_GPIO_B1_03[MUX_MODE] = 0x4 (XBAR_INOUT29)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN30	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_10[MUX_MODE] = 0x6 (XBAR_INOUT30) SW_MUX_CTL_PAD_GPIO_B1_04[MUX_MODE] = 0x4 (XBAR_INOUT30)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN31	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_11[MUX_MODE] = 0x6 (XBAR_INOUT31) SW_MUX_CTL_PAD_GPIO_B1_05[MUX_MODE] = 0x4 (XBAR_INOUT31)	See IOMUX Controller (IOMUXC) for more information

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN32	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_12[MUX_MODE] = 0x6 (XBAR_INOUT32) SW_MUX_CTL_PAD_GPIO_B1_06[MUX_MODE] = 0x4 (XBAR_INOUT32)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN33	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_13[MUX_MODE] = 0x6 (XBAR_INOUT33) SW_MUX_CTL_PAD_GPIO_B1_07[MUX_MODE] = 0x4 (XBAR_INOUT33)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN34	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_14[MUX_MODE] = 0x6 (XBAR_INOUT34) SW_MUX_CTL_PAD_GPIO_B1_10[MUX_MODE] = 0x4 (XBAR_INOUT34)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN35	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_15[MUX_MODE] = 0x6 (XBAR_INOUT35) SW_MUX_CTL_PAD_GPIO_B1_11[MUX_MODE] = 0x4 (XBAR_INOUT35)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN36	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_19[MUX_MODE] = 0x6 (XBAR_INOUT36) SW_MUX_CTL_PAD_GPIO_B1_08[MUX_MODE] = 0x4 (XBAR_INOUT36)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN37	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_20[MUX_MODE] = 0x6 (XBAR_INOUT37) SW_MUX_CTL_PAD_GPIO_B1_09[MUX_MODE] = 0x4 (XBAR_INOUT37)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_IN38	CMP1	COUT	-
XBAR1	XBAR_IN39	CMP2	COUT	-
XBAR1	XBAR_IN40	CMP3	COUT	-
XBAR1	XBAR_IN41	CMP4	COUT	-
XBAR1	XBAR_IN42	TMR1	TMR0_OUTPUT	-
XBAR1	XBAR_IN43	TMR1	TMR1_OUTPUT	-
XBAR1	XBAR_IN44	TMR1	TMR2_OUTPUT	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN45	TMR1	TMR3_OUTPUT	-
XBAR1	XBAR_IN46	TMR2	TMR0_OUTPUT	-
XBAR1	XBAR_IN47	TMR2	TMR1_OUTPUT	-
XBAR1	XBAR_IN48	TMR2	TMR2_OUTPUT	-
XBAR1	XBAR_IN49	TMR2	TMR3_OUTPUT	-
XBAR1	XBAR_IN50	TMR3	TMR0_OUTPUT	-
XBAR1	XBAR_IN51	TMR3	TMR1_OUTPUT	-
XBAR1	XBAR_IN52	TMR3	TMR2_OUTPUT	-
XBAR1	XBAR_IN53	TMR3	TMR3_OUTPUT	-
XBAR1	XBAR_IN54	TMR4	TMR0_OUTPUT	-
XBAR1	XBAR_IN55	TMR4	TMR1_OUTPUT	-
XBAR1	XBAR_IN56	TMR4	TMR2_OUTPUT	-
XBAR1	XBAR_IN57	TMR4	TMR3_OUTPUT	-
XBAR1	XBAR_IN58	TMR5	TMR0_OUTPUT	-
XBAR1	XBAR_IN59	TMR5	TMR1_OUTPUT	-
XBAR1	XBAR_IN60	TMR5	TMR2_OUTPUT	-
XBAR1	XBAR_IN61	TMR5	TMR3_OUTPUT	-
XBAR1	XBAR_IN62	TMR6	TMR0_OUTPUT	-
XBAR1	XBAR_IN63	TMR6	TMR1_OUTPUT	-
XBAR1	XBAR_IN64	TMR6	TMR2_OUTPUT	-
XBAR1	XBAR_IN65	TMR6	TMR3_OUTPUT	-
XBAR1	XBAR_IN66	TMR7	TMR0_OUTPUT	-
XBAR1	XBAR_IN67	TMR7	TMR1_OUTPUT	-
XBAR1	XBAR_IN68	TMR7	TMR2_OUTPUT	-
XBAR1	XBAR_IN69	TMR7	TMR3_OUTPUT	-
XBAR1	XBAR_IN70	TMR8	TMR0_OUTPUT	-
XBAR1	XBAR_IN71	TMR8	TMR1_OUTPUT	-
XBAR1	XBAR_IN72	TMR8	TMR2_OUTPUT	-
XBAR1	XBAR_IN73	TMR8	TMR3_OUTPUT	-
XBAR1	XBAR_IN74	FLEXPWM1	PWM0_MUX_TRIGGER0	-
XBAR1	XBAR_IN75	FLEXPWM1	PWM0_MUX_TRIGGER1	-
XBAR1	XBAR_IN76	FLEXPWM1	PWM1_MUX_TRIGGER0	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN77	FLEXPWM1	PWM1_MUX_TRIGGER1	-
XBAR1	XBAR_IN78	FLEXPWM1	PWM2_MUX_TRIGGER0	-
XBAR1	XBAR_IN79	FLEXPWM1	PWM2_MUX_TRIGGER1	-
XBAR1	XBAR_IN80	FLEXPWM1	PWM3_MUX_TRIGGER0	-
XBAR1	XBAR_IN81	FLEXPWM1	PWM3_MUX_TRIGGER1	-
XBAR1	XBAR_IN82	FLEXPWM2	PWM0_MUX_TRIGGER0	OR
XBAR1	XBAR_IN82	FLEXPWM2	PWM0_MUX_TRIGGER1	OR
XBAR1	XBAR_IN83	FLEXPWM2	PWM1_MUX_TRIGGER0	OR
XBAR1	XBAR_IN83	FLEXPWM2	PWM1_MUX_TRIGGER1	OR
XBAR1	XBAR_IN84	FLEXPWM2	PWM2_MUX_TRIGGER0	OR
XBAR1	XBAR_IN84	FLEXPWM2	PWM2_MUX_TRIGGER1	OR
XBAR1	XBAR_IN85	FLEXPWM2	PWM3_MUX_TRIGGER0	OR
XBAR1	XBAR_IN85	FLEXPWM2	PWM3_MUX_TRIGGER1	OR
XBAR1	XBAR_IN86	FLEXPWM3	PWM0_MUX_TRIGGER0	OR
XBAR1	XBAR_IN86	FLEXPWM3	PWM0_MUX_TRIGGER1	OR
XBAR1	XBAR_IN87	FLEXPWM3	PWM1_MUX_TRIGGER0	OR
XBAR1	XBAR_IN87	FLEXPWM3	PWM1_MUX_TRIGGER1	OR
XBAR1	XBAR_IN88	FLEXPWM3	PWM2_MUX_TRIGGER0	OR
XBAR1	XBAR_IN88	FLEXPWM3	PWM2_MUX_TRIGGER1	OR
XBAR1	XBAR_IN89	FLEXPWM3	PWM3_MUX_TRIGGER0	OR
XBAR1	XBAR_IN89	FLEXPWM3	PWM3_MUX_TRIGGER1	OR
XBAR1	XBAR_IN90	FLEXPWM4	PWM0_MUX_TRIGGER0	OR
XBAR1	XBAR_IN90	FLEXPWM4	PWM0_MUX_TRIGGER1	OR
XBAR1	XBAR_IN91	FLEXPWM4	PWM1_MUX_TRIGGER0	OR
XBAR1	XBAR_IN91	FLEXPWM4	PWM1_MUX_TRIGGER1	OR
XBAR1	XBAR_IN92	FLEXPWM4	PWM2_MUX_TRIGGER0	OR
XBAR1	XBAR_IN92	FLEXPWM4	PWM2_MUX_TRIGGER1	OR
XBAR1	XBAR_IN93	FLEXPWM4	PWM3_MUX_TRIGGER0	OR
XBAR1	XBAR_IN93	FLEXPWM4	PWM3_MUX_TRIGGER1	OR
XBAR1	XBAR_IN94	LPIT1	LPIT_TRIG_OUT0	-
XBAR1	XBAR_IN95	LPIT1	LPIT_TRIG_OUT1	-
XBAR1	XBAR_IN96	LPIT1	LPIT_TRIG_OUT2	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN97	LPIT1	LPIT_TRIG_OUT3	-
XBAR1	XBAR_IN98	LPIT2	LPIT_TRIG_OUT0	-
XBAR1	XBAR_IN99	LPIT2	LPIT_TRIG_OUT1	-
XBAR1	XBAR_IN100	LPIT2	LPIT_TRIG_OUT2	-
XBAR1	XBAR_IN101	LPIT2	LPIT_TRIG_OUT3	-
XBAR1	XBAR_IN102	LPIT3	LPIT_TRIG_OUT0	-
XBAR1	XBAR_IN103	LPIT3	LPIT_TRIG_OUT1	-
XBAR1	XBAR_IN104	LPIT3	LPIT_TRIG_OUT2	-
XBAR1	XBAR_IN105	LPIT3	LPIT_TRIG_OUT3	-
XBAR1	XBAR_IN106	TRIGGER_SYNC	TRIGGER_SYNC_OUT0	-
XBAR1	XBAR_IN107	TRIGGER_SYNC	TRIGGER_SYNC_OUT1	-
XBAR1	XBAR_IN108	TRIGGER_SYNC	TRIGGER_SYNC_OUT2	-
XBAR1	XBAR_IN109	TRIGGER_SYNC	TRIGGER_SYNC_OUT3	-
XBAR1	XBAR_IN110	eDMA4	DMA_TRIGGER_OUT0	-
XBAR1	XBAR_IN111	eDMA4	DMA_TRIGGER_OUT1	-
XBAR1	XBAR_IN112	eDMA4	DMA_TRIGGER_OUT2	-
XBAR1	XBAR_IN113	eDMA4	DMA_TRIGGER_OUT3	-
XBAR1	XBAR_IN114	eDMA4	DMA_TRIGGER_OUT4	-
XBAR1	XBAR_IN115	eDMA4	DMA_TRIGGER_OUT5	-
XBAR1	XBAR_IN116	eDMA4	DMA_TRIGGER_OUT6	-
XBAR1	XBAR_IN117	eDMA4	DMA_TRIGGER_OUT7	-
XBAR1	XBAR_IN118	eDMA3	DMA_TRIGGER_OUT0	-
XBAR1	XBAR_IN119	eDMA3	DMA_TRIGGER_OUT1	-
XBAR1	XBAR_IN120	eDMA3	DMA_TRIGGER_OUT2	-
XBAR1	XBAR_IN121	eDMA3	DMA_TRIGGER_OUT3	-
XBAR1	XBAR_IN122	eDMA3	DMA_TRIGGER_OUT4	-
XBAR1	XBAR_IN123	eDMA3	DMA_TRIGGER_OUT5	-
XBAR1	XBAR_IN124	eDMA3	DMA_TRIGGER_OUT6	-
XBAR1	XBAR_IN125	eDMA3	DMA_TRIGGER_OUT7	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN126	ADC1	ADC_TCOMP_PULSE0	-
XBAR1	XBAR_IN127	ADC1	ADC_TCOMP_PULSE1	-
XBAR1	XBAR_IN128	ADC1	ADC_TCOMP_PULSE2	-
XBAR1	XBAR_IN129	ADC1	ADC_TCOMP_PULSE3	-
XBAR1	XBAR_IN130	ADC1	ADC_TCOMP_PULSE4	-
XBAR1	XBAR_IN131	ADC1	ADC_TCOMP_PULSE5	-
XBAR1	XBAR_IN132	ADC1	ADC_TCOMP_PULSE6	-
XBAR1	XBAR_IN133	ADC1	ADC_TCOMP_PULSE7	-
XBAR1	XBAR_IN134	ADC2	ADC_TCOMP_PULSE0	-
XBAR1	XBAR_IN135	ADC2	ADC_TCOMP_PULSE1	-
XBAR1	XBAR_IN136	ADC2	ADC_TCOMP_PULSE2	-
XBAR1	XBAR_IN137	ADC2	ADC_TCOMP_PULSE3	-
XBAR1	XBAR_IN138	ADC2	ADC_TCOMP_PULSE4	-
XBAR1	XBAR_IN139	ADC2	ADC_TCOMP_PULSE5	-
XBAR1	XBAR_IN140	ADC2	ADC_TCOMP_PULSE6	-
XBAR1	XBAR_IN141	ADC2	ADC_TCOMP_PULSE7	-
XBAR1	XBAR_IN142	TPM1	TPM_CH_TRIGGER0	-
XBAR1	XBAR_IN143	TPM1	TPM_CH_TRIGGER1	-
XBAR1	XBAR_IN144	TPM1	TPM_CH_TRIGGER2	-
XBAR1	XBAR_IN145	TPM1	TPM_CH_TRIGGER3	-
XBAR1	XBAR_IN146	TPM1	TPM_TRIGGER	-
XBAR1	XBAR_IN147	TPM2	TPM_CH_TRIGGER0	-
XBAR1	XBAR_IN148	TPM2	TPM_CH_TRIGGER1	-
XBAR1	XBAR_IN149	TPM2	TPM_CH_TRIGGER2	-
XBAR1	XBAR_IN150	TPM2	TPM_CH_TRIGGER3	-
XBAR1	XBAR_IN151	TPM2	TPM_TRIGGER	-
XBAR1	XBAR_IN152	TPM3	TPM_CH_TRIGGER0	-
XBAR1	XBAR_IN153	TPM3	TPM_CH_TRIGGER1	-
XBAR1	XBAR_IN154	TPM3	TPM_CH_TRIGGER2	-
XBAR1	XBAR_IN155	TPM3	TPM_CH_TRIGGER3	-
XBAR1	XBAR_IN156	TPM3	TPM_TRIGGER	-
XBAR1	XBAR_IN157	TPM4	TPM_CH_TRIGGER0	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN158	TPM4	TPM_CH_TRIGGER1	-
XBAR1	XBAR_IN159	TPM4	TPM_CH_TRIGGER2	-
XBAR1	XBAR_IN160	TPM4	TPM_CH_TRIGGER3	-
XBAR1	XBAR_IN161	TPM4	TPM_TRIGGER	-
XBAR1	XBAR_IN162	TPM5	TPM_CH_TRIGGER0	-
XBAR1	XBAR_IN163	TPM5	TPM_CH_TRIGGER1	-
XBAR1	XBAR_IN164	TPM5	TPM_CH_TRIGGER2	-
XBAR1	XBAR_IN165	TPM5	TPM_CH_TRIGGER3	-
XBAR1	XBAR_IN166	TPM5	TPM_TRIGGER	-
XBAR1	XBAR_IN167	TPM6	TPM_CH_TRIGGER0	-
XBAR1	XBAR_IN168	TPM6	TPM_CH_TRIGGER1	-
XBAR1	XBAR_IN169	TPM6	TPM_CH_TRIGGER2	-
XBAR1	XBAR_IN170	TPM6	TPM_CH_TRIGGER3	-
XBAR1	XBAR_IN171	TPM6	TPM_TRIGGER	-
XBAR1	XBAR_IN172	LPTMR1	LPTIMER_TRIGGER_DELAY	-
XBAR1	XBAR_IN173	LPTMR2	LPTIMER_TRIGGER_DELAY	-
XBAR1	XBAR_IN174	LPTMR3	LPTIMER_TRIGGER_DELAY	-
XBAR1	XBAR_IN175	NETC	TMR_1588_PP1	-
XBAR1	XBAR_IN176	NETC	TMR_1588_PP2	-
XBAR1	XBAR_IN177	NETC	TMR_1588_PP3	-
XBAR1	XBAR_IN178	SINC1	Channel 0 break due to clock absence detected	OR
XBAR1	XBAR_IN178	SINC1	Channel 0 break due to high limit	OR
XBAR1	XBAR_IN178	SINC1	Channel 0 break due to low limit	OR
XBAR1	XBAR_IN178	SINC1	Channel 0 break due to short circuit detected	OR
XBAR1	XBAR_IN178	SINC1	Channel 0 break due to window limit	OR
XBAR1	XBAR_IN179	SINC1	Channel 1 break due to clock absence detected	OR
XBAR1	XBAR_IN179	SINC1	Channel 1 break due to high limit	OR
XBAR1	XBAR_IN179	SINC1	Channel 1 break due to low limit	OR
XBAR1	XBAR_IN179	SINC1	Channel 1 break due to short circuit detected	OR

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN179	SINC1	Channel 1 break due to window limit	OR
XBAR1	XBAR_IN180	SINC1	Channel 2 break due to clock absence detected	OR
XBAR1	XBAR_IN180	SINC1	Channel 2 break due to high limit	OR
XBAR1	XBAR_IN180	SINC1	Channel 2 break due to low limit	OR
XBAR1	XBAR_IN180	SINC1	Channel 2 break due to short circuit detected	OR
XBAR1	XBAR_IN180	SINC1	Channel 2 break due to window limit	OR
XBAR1	XBAR_IN181	SINC1	Channel 3 break due to clock absence detected	OR
XBAR1	XBAR_IN181	SINC1	Channel 3 break due to high limit	OR
XBAR1	XBAR_IN181	SINC1	Channel 3 break due to low limit	OR
XBAR1	XBAR_IN181	SINC1	Channel 3 break due to short circuit detected	OR
XBAR1	XBAR_IN181	SINC1	Channel 3 break due to window limit	OR
XBAR1	XBAR_IN182	SINC2	Channel 0 break due to clock absence detected	OR
XBAR1	XBAR_IN182	SINC2	Channel 0 break due to high limit	OR
XBAR1	XBAR_IN182	SINC2	Channel 0 break due to low limit	OR
XBAR1	XBAR_IN182	SINC2	Channel 0 break due to short circuit detected	OR
XBAR1	XBAR_IN182	SINC2	Channel 0 break due to window limit	OR
XBAR1	XBAR_IN183	SINC2	Channel 1 break due to clock absence detected	OR
XBAR1	XBAR_IN183	SINC2	Channel 1 break due to high limit	OR
XBAR1	XBAR_IN183	SINC2	Channel 1 break due to low limit	OR
XBAR1	XBAR_IN183	SINC2	Channel 1 break due to short circuit detected	OR
XBAR1	XBAR_IN183	SINC2	Channel 1 break due to window limit	OR
XBAR1	XBAR_IN184	SINC2	Channel 2 break due to clock absence detected	OR
XBAR1	XBAR_IN184	SINC2	Channel 2 break due to high limit	OR
XBAR1	XBAR_IN184	SINC2	Channel 2 break due to low limit	OR
XBAR1	XBAR_IN184	SINC2	Channel 2 break due to short circuit detected	OR

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN184	SINC2	Channel 2 break due to window limit	OR
XBAR1	XBAR_IN185	SINC2	Channel 3 break due to clock absence detected	OR
XBAR1	XBAR_IN185	SINC2	Channel 3 break due to high limit	OR
XBAR1	XBAR_IN185	SINC2	Channel 3 break due to low limit	OR
XBAR1	XBAR_IN185	SINC2	Channel 3 break due to short circuit detected	OR
XBAR1	XBAR_IN185	SINC2	Channel 3 break due to window limit	OR
XBAR1	XBAR_IN186	SINC3	Channel 0 break due to clock absence detected	OR
XBAR1	XBAR_IN186	SINC3	Channel 0 break due to high limit	OR
XBAR1	XBAR_IN186	SINC3	Channel 0 break due to low limit	OR
XBAR1	XBAR_IN186	SINC3	Channel 0 break due to short circuit detected	OR
XBAR1	XBAR_IN186	SINC3	Channel 0 break due to window limit	OR
XBAR1	XBAR_IN187	SINC3	Channel 1 break due to clock absence detected	OR
XBAR1	XBAR_IN187	SINC3	Channel 1 break due to high limit	OR
XBAR1	XBAR_IN187	SINC3	Channel 1 break due to low limit	OR
XBAR1	XBAR_IN187	SINC3	Channel 1 break due to short circuit detected	OR
XBAR1	XBAR_IN187	SINC3	Channel 1 break due to window limit	OR
XBAR1	XBAR_IN188	SINC3	Channel 2 break due to clock absence detected	OR
XBAR1	XBAR_IN188	SINC3	Channel 2 break due to high limit	OR
XBAR1	XBAR_IN188	SINC3	Channel 2 break due to low limit	OR
XBAR1	XBAR_IN188	SINC3	Channel 2 break due to short circuit detected	OR
XBAR1	XBAR_IN188	SINC3	Channel 2 break due to window limit	OR
XBAR1	XBAR_IN189	SINC3	Channel 3 break due to clock absence detected	OR
XBAR1	XBAR_IN189	SINC3	Channel 3 break due to high limit	OR
XBAR1	XBAR_IN189	SINC3	Channel 3 break due to low limit	OR
XBAR1	XBAR_IN189	SINC3	Channel 3 break due to short circuit detected	OR

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR1	XBAR_IN189	SINC3	Channel 3 break due to window limit	OR
XBAR1	XBAR_IN190	AOI1	AOI_OUT0	-
XBAR1	XBAR_IN191	AOI1	AOI_OUT1	-
XBAR1	XBAR_IN192	AOI1	AOI_OUT2	-
XBAR1	XBAR_IN193	AOI1	AOI_OUT3	-
XBAR1	XBAR_IN194	AOI2	AOI_OUT0	-
XBAR1	XBAR_IN195	AOI2	AOI_OUT1	-
XBAR1	XBAR_IN196	AOI2	AOI_OUT2	-
XBAR1	XBAR_IN197	AOI2	AOI_OUT3	-
XBAR1	XBAR_IN198	TRIGGER_SYNC	TRIGGER_SYNC_OUT4	-
XBAR1	XBAR_IN199	TRIGGER_SYNC	TRIGGER_SYNC_OUT5	-
XBAR1	XBAR_IN200	TRIGGER_SYNC	TRIGGER_SYNC_OUT6	-
XBAR1	XBAR_IN201	TRIGGER_SYNC	TRIGGER_SYNC_OUT7	-
XBAR1	XBAR_IN202	-	Reserved	-
XBAR1	XBAR_IN203	-	Reserved	-
XBAR1	XBAR_IN204	-	Reserved	-
XBAR1	XBAR_IN205	-	Reserved	-
XBAR1	XBAR_IN206	AOI3	AOI_OUT0	-
XBAR1	XBAR_IN207	AOI3	AOI_OUT1	-
XBAR1	XBAR_IN208	AOI3	AOI_OUT2	-
XBAR1	XBAR_IN209	AOI3	AOI_OUT3	-
XBAR1	XBAR_IN210	AOI4	AOI_OUT0	-
XBAR1	XBAR_IN211	AOI4	AOI_OUT1	-
XBAR1	XBAR_IN212	AOI4	AOI_OUT2	-
XBAR1	XBAR_IN213	AOI4	AOI_OUT3	-
XBAR2	XBAR_IN0	-	GND	-
XBAR2	XBAR_IN1	-	1'B1	-
XBAR2	XBAR_IN2	-	GND	-
XBAR2	XBAR_IN3	-	1'B1	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR2	XBAR_IN4	CMP1	COUT	-
XBAR2	XBAR_IN5	CMP2	COUT	-
XBAR2	XBAR_IN6	CMP3	COUT	-
XBAR2	XBAR_IN7	CMP4	COUT	-
XBAR2	XBAR_IN8	TMR1	TMR0_OUTPUT	-
XBAR2	XBAR_IN9	TMR1	TMR1_OUTPUT	-
XBAR2	XBAR_IN10	TMR1	TMR2_OUTPUT	-
XBAR2	XBAR_IN11	TMR1	TMR3_OUTPUT	-
XBAR2	XBAR_IN12	TMR2	TMR0_OUTPUT	-
XBAR2	XBAR_IN13	TMR2	TMR1_OUTPUT	-
XBAR2	XBAR_IN14	TMR2	TMR2_OUTPUT	-
XBAR2	XBAR_IN15	TMR2	TMR3_OUTPUT	-
XBAR2	XBAR_IN16	TMR3	TMR0_OUTPUT	-
XBAR2	XBAR_IN17	TMR3	TMR1_OUTPUT	-
XBAR2	XBAR_IN18	TMR3	TMR2_OUTPUT	-
XBAR2	XBAR_IN19	TMR3	TMR3_OUTPUT	-
XBAR2	XBAR_IN20	TMR4	TMR0_OUTPUT	-
XBAR2	XBAR_IN21	TMR4	TMR1_OUTPUT	-
XBAR2	XBAR_IN22	TMR4	TMR2_OUTPUT	-
XBAR2	XBAR_IN23	TMR4	TMR3_OUTPUT	-
XBAR2	XBAR_IN24	TMR5	TMR0_OUTPUT	-
XBAR2	XBAR_IN25	TMR5	TMR1_OUTPUT	-
XBAR2	XBAR_IN26	TMR5	TMR2_OUTPUT	-
XBAR2	XBAR_IN27	TMR5	TMR3_OUTPUT	-
XBAR2	XBAR_IN28	TMR6	TMR0_OUTPUT	-
XBAR2	XBAR_IN29	TMR6	TMR1_OUTPUT	-
XBAR2	XBAR_IN30	TMR6	TMR2_OUTPUT	-
XBAR2	XBAR_IN31	TMR6	TMR3_OUTPUT	-
XBAR2	XBAR_IN32	TMR7	TMR0_OUTPUT	-
XBAR2	XBAR_IN33	TMR7	TMR1_OUTPUT	-
XBAR2	XBAR_IN34	TMR7	TMR2_OUTPUT	-
XBAR2	XBAR_IN35	TMR7	TMR3_OUTPUT	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR2	XBAR_IN36	TMR8	TMR0_OUTPUT	-
XBAR2	XBAR_IN37	TMR8	TMR1_OUTPUT	-
XBAR2	XBAR_IN38	TMR8	TMR2_OUTPUT	-
XBAR2	XBAR_IN39	TMR8	TMR3_OUTPUT	-
XBAR2	XBAR_IN40	FLEXPWM1	PWM0_MUX_TRIGGER0	OR
XBAR2	XBAR_IN40	FLEXPWM1	PWM0_MUX_TRIGGER1	OR
XBAR2	XBAR_IN41	FLEXPWM1	PWM1_MUX_TRIGGER0	OR
XBAR2	XBAR_IN41	FLEXPWM1	PWM1_MUX_TRIGGER1	OR
XBAR2	XBAR_IN42	FLEXPWM1	PWM2_MUX_TRIGGER0	OR
XBAR2	XBAR_IN42	FLEXPWM1	PWM2_MUX_TRIGGER1	OR
XBAR2	XBAR_IN43	FLEXPWM1	PWM3_MUX_TRIGGER0	OR
XBAR2	XBAR_IN43	FLEXPWM1	PWM3_MUX_TRIGGER1	OR
XBAR2	XBAR_IN44	FLEXPWM2	PWM0_MUX_TRIGGER0	OR
XBAR2	XBAR_IN44	FLEXPWM2	PWM0_MUX_TRIGGER1	OR
XBAR2	XBAR_IN45	FLEXPWM2	PWM1_MUX_TRIGGER0	OR
XBAR2	XBAR_IN45	FLEXPWM2	PWM1_MUX_TRIGGER1	OR
XBAR2	XBAR_IN46	FLEXPWM2	PWM2_MUX_TRIGGER0	OR
XBAR2	XBAR_IN46	FLEXPWM2	PWM2_MUX_TRIGGER1	OR
XBAR2	XBAR_IN47	FLEXPWM2	PWM3_MUX_TRIGGER0	OR
XBAR2	XBAR_IN47	FLEXPWM2	PWM3_MUX_TRIGGER1	OR
XBAR2	XBAR_IN48	FLEXPWM3	PWM0_MUX_TRIGGER0	OR
XBAR2	XBAR_IN48	FLEXPWM3	PWM0_MUX_TRIGGER1	OR
XBAR2	XBAR_IN49	FLEXPWM3	PWM1_MUX_TRIGGER0	OR
XBAR2	XBAR_IN49	FLEXPWM3	PWM1_MUX_TRIGGER1	OR
XBAR2	XBAR_IN50	FLEXPWM3	PWM2_MUX_TRIGGER0	OR
XBAR2	XBAR_IN50	FLEXPWM3	PWM2_MUX_TRIGGER1	OR
XBAR2	XBAR_IN51	FLEXPWM3	PWM3_MUX_TRIGGER0	OR
XBAR2	XBAR_IN51	FLEXPWM3	PWM3_MUX_TRIGGER1	OR
XBAR2	XBAR_IN52	FLEXPWM4	PWM0_MUX_TRIGGER0	OR
XBAR2	XBAR_IN52	FLEXPWM4	PWM0_MUX_TRIGGER1	OR
XBAR2	XBAR_IN53	FLEXPWM4	PWM1_MUX_TRIGGER0	OR
XBAR2	XBAR_IN53	FLEXPWM4	PWM1_MUX_TRIGGER1	OR

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR2	XBAR_IN54	FLEXPWM4	PWM2_MUX_TRIGGER0	OR
XBAR2	XBAR_IN54	FLEXPWM4	PWM2_MUX_TRIGGER1	OR
XBAR2	XBAR_IN55	FLEXPWM4	PWM3_MUX_TRIGGER0	OR
XBAR2	XBAR_IN55	FLEXPWM4	PWM3_MUX_TRIGGER1	OR
XBAR2	XBAR_IN56	LPIT1	LPIT_TRIG_OUT0	-
XBAR2	XBAR_IN57	LPIT1	LPIT_TRIG_OUT1	-
XBAR2	XBAR_IN58	LPIT1	LPIT_TRIG_OUT2	-
XBAR2	XBAR_IN59	LPIT1	LPIT_TRIG_OUT3	-
XBAR2	XBAR_IN60	LPIT2	LPIT_TRIG_OUT0	-
XBAR2	XBAR_IN61	LPIT2	LPIT_TRIG_OUT1	-
XBAR2	XBAR_IN62	LPIT2	LPIT_TRIG_OUT2	-
XBAR2	XBAR_IN63	LPIT2	LPIT_TRIG_OUT3	-
XBAR2	XBAR_IN64	LPIT3	LPIT_TRIG_OUT0	-
XBAR2	XBAR_IN65	LPIT3	LPIT_TRIG_OUT1	-
XBAR2	XBAR_IN66	LPIT3	LPIT_TRIG_OUT2	-
XBAR2	XBAR_IN67	LPIT3	LPIT_TRIG_OUT3	-
XBAR2	XBAR_IN68	SINC1	PULSE_TRG0	-
XBAR2	XBAR_IN69	SINC1	PULSE_TRG1	-
XBAR2	XBAR_IN70	SINC1	PULSE_TRG2	-
XBAR2	XBAR_IN71	SINC1	PULSE_TRG3	-
XBAR2	XBAR_IN72	eDMA4	DMA_TRIGGER_OUT0	-
XBAR2	XBAR_IN73	eDMA4	DMA_TRIGGER_OUT1	-
XBAR2	XBAR_IN74	eDMA4	DMA_TRIGGER_OUT2	-
XBAR2	XBAR_IN75	eDMA4	DMA_TRIGGER_OUT3	-
XBAR2	XBAR_IN76	eDMA4	DMA_TRIGGER_OUT4	-
XBAR2	XBAR_IN77	eDMA4	DMA_TRIGGER_OUT5	-
XBAR2	XBAR_IN78	eDMA4	DMA_TRIGGER_OUT6	-
XBAR2	XBAR_IN79	eDMA4	DMA_TRIGGER_OUT7	-
XBAR2	XBAR_IN80	eDMA3	DMA_TRIGGER_OUT0	-
XBAR2	XBAR_IN81	eDMA3	DMA_TRIGGER_OUT1	-
XBAR2	XBAR_IN82	eDMA3	DMA_TRIGGER_OUT2	-
XBAR2	XBAR_IN83	eDMA3	DMA_TRIGGER_OUT3	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR2	XBAR_IN84	eDMA3	DMA_TRIGGER_OUT4	-
XBAR2	XBAR_IN85	eDMA3	DMA_TRIGGER_OUT5	-
XBAR2	XBAR_IN86	eDMA3	DMA_TRIGGER_OUT6	-
XBAR2	XBAR_IN87	eDMA3	DMA_TRIGGER_OUT7	-
XBAR2	XBAR_IN88	ADC1	ADC_TCOMP_PULSE0	-
XBAR2	XBAR_IN89	ADC1	ADC_TCOMP_PULSE1	-
XBAR2	XBAR_IN90	ADC1	ADC_TCOMP_PULSE2	-
XBAR2	XBAR_IN91	ADC1	ADC_TCOMP_PULSE3	-
XBAR2	XBAR_IN92	ADC1	ADC_TCOMP_PULSE4	-
XBAR2	XBAR_IN93	ADC1	ADC_TCOMP_PULSE5	-
XBAR2	XBAR_IN94	ADC1	ADC_TCOMP_PULSE6	-
XBAR2	XBAR_IN95	ADC1	ADC_TCOMP_PULSE7	-
XBAR2	XBAR_IN96	ADC2	ADC_TCOMP_PULSE0	-
XBAR2	XBAR_IN97	ADC2	ADC_TCOMP_PULSE1	-
XBAR2	XBAR_IN98	ADC2	ADC_TCOMP_PULSE2	-
XBAR2	XBAR_IN99	ADC2	ADC_TCOMP_PULSE3	-
XBAR2	XBAR_IN100	ADC2	ADC_TCOMP_PULSE4	-
XBAR2	XBAR_IN101	ADC2	ADC_TCOMP_PULSE5	-
XBAR2	XBAR_IN102	ADC2	ADC_TCOMP_PULSE6	-
XBAR2	XBAR_IN103	ADC2	ADC_TCOMP_PULSE7	-
XBAR2	XBAR_IN104	TPM1	TPM_CH_TRIGGER0	-
XBAR2	XBAR_IN105	TPM1	TPM_CH_TRIGGER1	-
XBAR2	XBAR_IN106	TPM1	TPM_CH_TRIGGER2	-
XBAR2	XBAR_IN107	TPM1	TPM_CH_TRIGGER3	-
XBAR2	XBAR_IN108	TPM1	TPM_TRIGGER	-
XBAR2	XBAR_IN109	TPM2	TPM_CH_TRIGGER0	-
XBAR2	XBAR_IN110	TPM2	TPM_CH_TRIGGER1	-
XBAR2	XBAR_IN111	TPM2	TPM_CH_TRIGGER2	-
XBAR2	XBAR_IN112	TPM2	TPM_CH_TRIGGER3	-
XBAR2	XBAR_IN113	TPM2	TPM_TRIGGER	-
XBAR2	XBAR_IN114	TPM3	TPM_CH_TRIGGER0	-
XBAR2	XBAR_IN115	TPM3	TPM_CH_TRIGGER1	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR2	XBAR_IN116	TPM3	TPM_CH_TRIGGER2	-
XBAR2	XBAR_IN117	TPM3	TPM_CH_TRIGGER3	-
XBAR2	XBAR_IN118	TPM3	TPM_TRIGGER	-
XBAR2	XBAR_IN119	TPM4	TPM_CH_TRIGGER0	-
XBAR2	XBAR_IN120	TPM4	TPM_CH_TRIGGER1	-
XBAR2	XBAR_IN121	TPM4	TPM_CH_TRIGGER2	-
XBAR2	XBAR_IN122	TPM4	TPM_CH_TRIGGER3	-
XBAR2	XBAR_IN123	TPM4	TPM_TRIGGER	-
XBAR2	XBAR_IN124	TPM5	TPM_CH_TRIGGER0	-
XBAR2	XBAR_IN125	TPM5	TPM_CH_TRIGGER1	-
XBAR2	XBAR_IN126	TPM5	TPM_CH_TRIGGER2	-
XBAR2	XBAR_IN127	TPM5	TPM_CH_TRIGGER3	-
XBAR2	XBAR_IN128	TPM5	TPM_TRIGGER	-
XBAR2	XBAR_IN129	TPM6	TPM_CH_TRIGGER0	-
XBAR2	XBAR_IN130	TPM6	TPM_CH_TRIGGER1	-
XBAR2	XBAR_IN131	TPM6	TPM_CH_TRIGGER2	-
XBAR2	XBAR_IN132	TPM6	TPM_CH_TRIGGER3	-
XBAR2	XBAR_IN133	TPM6	TPM_TRIGGER	-
XBAR2	XBAR_IN134	LPTMR1	LPTIMER_TRIGGER_DELAY	-
XBAR2	XBAR_IN135	LPTMR2	LPTIMER_TRIGGER_DELAY	-
XBAR2	XBAR_IN136	LPTMR3	LPTIMER_TRIGGER_DELAY	-
XBAR2	XBAR_IN137	NETC	TMR_1588_PP1	-
XBAR2	XBAR_IN138	NETC	TMR_1588_PP2	-
XBAR2	XBAR_IN139	NETC	TMR_1588_PP3	-
XBAR2	XBAR_IN140	SINC1	Channel 0 break due to clock absence detected	OR
XBAR2	XBAR_IN140	SINC1	Channel 0 break due to high limit	OR
XBAR2	XBAR_IN140	SINC1	Channel 0 break due to low limit	OR
XBAR2	XBAR_IN140	SINC1	Channel 0 break due to short circuit detected	OR
XBAR2	XBAR_IN140	SINC1	Channel 0 break due to window limit	OR
XBAR2	XBAR_IN141	SINC1	Channel 1 break due to clock absence detected	OR

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR2	XBAR_IN141	SINC1	Channel 1 break due to high limit	OR
XBAR2	XBAR_IN141	SINC1	Channel 1 break due to low limit	OR
XBAR2	XBAR_IN141	SINC1	Channel 1 break due to short circuit detected	OR
XBAR2	XBAR_IN141	SINC1	Channel 1 break due to window limit	OR
XBAR2	XBAR_IN142	SINC1	Channel 2 break due to clock absence detected	OR
XBAR2	XBAR_IN142	SINC1	Channel 2 break due to high limit	OR
XBAR2	XBAR_IN142	SINC1	Channel 2 break due to low limit	OR
XBAR2	XBAR_IN142	SINC1	Channel 2 break due to short circuit detected	OR
XBAR2	XBAR_IN142	SINC1	Channel 2 break due to window limit	OR
XBAR2	XBAR_IN143	SINC1	Channel 3 break due to clock absence detected	OR
XBAR2	XBAR_IN143	SINC1	Channel 3 break due to high limit	OR
XBAR2	XBAR_IN143	SINC1	Channel 3 break due to low limit	OR
XBAR2	XBAR_IN143	SINC1	Channel 3 break due to short circuit detected	OR
XBAR2	XBAR_IN143	SINC1	Channel 3 break due to window limit	OR
XBAR2	XBAR_IN144	SINC2	Channel 0 break due to clock absence detected	OR
XBAR2	XBAR_IN144	SINC2	Channel 0 break due to high limit	OR
XBAR2	XBAR_IN144	SINC2	Channel 0 break due to low limit	OR
XBAR2	XBAR_IN144	SINC2	Channel 0 break due to short circuit detected	OR
XBAR2	XBAR_IN144	SINC2	Channel 0 break due to window limit	OR
XBAR2	XBAR_IN145	SINC2	Channel 1 break due to clock absence detected	OR
XBAR2	XBAR_IN145	SINC2	Channel 1 break due to high limit	OR
XBAR2	XBAR_IN145	SINC2	Channel 1 break due to low limit	OR
XBAR2	XBAR_IN145	SINC2	Channel 1 break due to short circuit detected	OR
XBAR2	XBAR_IN145	SINC2	Channel 1 break due to window limit	OR
XBAR2	XBAR_IN146	SINC2	Channel 2 break due to clock absence detected	OR

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR2	XBAR_IN146	SINC2	Channel 2 break due to high limit	OR
XBAR2	XBAR_IN146	SINC2	Channel 2 break due to low limit	OR
XBAR2	XBAR_IN146	SINC2	Channel 2 break due to short circuit detected	OR
XBAR2	XBAR_IN146	SINC2	Channel 2 break due to window limit	OR
XBAR2	XBAR_IN147	SINC2	Channel 3 break due to clock absence detected	OR
XBAR2	XBAR_IN147	SINC2	Channel 3 break due to high limit	OR
XBAR2	XBAR_IN147	SINC2	Channel 3 break due to low limit	OR
XBAR2	XBAR_IN147	SINC2	Channel 3 break due to short circuit detected	OR
XBAR2	XBAR_IN147	SINC2	Channel 3 break due to window limit	OR
XBAR2	XBAR_IN148	SINC3	Channel 0 break due to clock absence detected	OR
XBAR2	XBAR_IN148	SINC3	Channel 0 break due to high limit	OR
XBAR2	XBAR_IN148	SINC3	Channel 0 break due to low limit	OR
XBAR2	XBAR_IN148	SINC3	Channel 0 break due to short circuit detected	OR
XBAR2	XBAR_IN148	SINC3	Channel 0 break due to window limit	OR
XBAR2	XBAR_IN149	SINC3	Channel 1 break due to clock absence detected	OR
XBAR2	XBAR_IN149	SINC3	Channel 1 break due to high limit	OR
XBAR2	XBAR_IN149	SINC3	Channel 1 break due to low limit	OR
XBAR2	XBAR_IN149	SINC3	Channel 1 break due to short circuit detected	OR
XBAR2	XBAR_IN149	SINC3	Channel 1 break due to window limit	OR
XBAR2	XBAR_IN150	SINC3	Channel 2 break due to clock absence detected	OR
XBAR2	XBAR_IN150	SINC3	Channel 2 break due to high limit	OR
XBAR2	XBAR_IN150	SINC3	Channel 2 break due to low limit	OR
XBAR2	XBAR_IN150	SINC3	Channel 2 break due to short circuit detected	OR
XBAR2	XBAR_IN150	SINC3	Channel 2 break due to window limit	OR
XBAR2	XBAR_IN151	SINC3	Channel 3 break due to clock absence detected	OR

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR2	XBAR_IN151	SINC3	Channel 3 break due to high limit	OR
XBAR2	XBAR_IN151	SINC3	Channel 3 break due to low limit	OR
XBAR2	XBAR_IN151	SINC3	Channel 3 break due to short circuit detected	OR
XBAR2	XBAR_IN151	SINC3	Channel 3 break due to window limit	OR
XBAR2	XBAR_IN152	-	Reserved	-
XBAR2	XBAR_IN153	-	Reserved	-
XBAR2	XBAR_IN154	-	Reserved	-
XBAR2	XBAR_IN155	-	Reserved	-
XBAR2	XBAR_IN156	eQDC1	POS_MATCH0	-
XBAR2	XBAR_IN157	eQDC1	POS_MATCH1	-
XBAR2	XBAR_IN158	eQDC1	POS_MATCH2	-
XBAR2	XBAR_IN159	eQDC1	POS_MATCH3	-
XBAR2	XBAR_IN160	eQDC1	COMP_FLG0	-
XBAR2	XBAR_IN161	eQDC1	COMP_FLG1	-
XBAR2	XBAR_IN162	eQDC1	COMP_FLG2	-
XBAR2	XBAR_IN163	eQDC1	COMP_FLG3	-
XBAR2	XBAR_IN164	eQDC1	CNT_DN	-
XBAR2	XBAR_IN165	eQDC1	CNT_UP	-
XBAR2	XBAR_IN166	eQDC1	DIR	-
XBAR2	XBAR_IN167	eQDC3	POS_MATCH0	-
XBAR2	XBAR_IN168	eQDC3	POS_MATCH1	-
XBAR2	XBAR_IN169	eQDC3	POS_MATCH2	-
XBAR2	XBAR_IN170	eQDC3	POS_MATCH3	-
XBAR2	XBAR_IN171	eQDC3	COMP_FLG0	-
XBAR2	XBAR_IN172	eQDC3	COMP_FLG1	-
XBAR2	XBAR_IN173	eQDC3	COMP_FLG2	-
XBAR2	XBAR_IN174	eQDC3	COMP_FLG3	-
XBAR2	XBAR_IN175	eQDC3	CNT_DN	-
XBAR2	XBAR_IN176	eQDC3	CNT_UP	-
XBAR2	XBAR_IN177	eQDC3	DIR	-
XBAR3	XBAR_IN0	-	GND	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR3	XBAR_IN1	-	1'B1	-
XBAR3	XBAR_IN2	-	GND	-
XBAR3	XBAR_IN3	-	1'B1	-
XBAR3	XBAR_IN4	CMP1	COUT	-
XBAR3	XBAR_IN5	CMP2	COUT	-
XBAR3	XBAR_IN6	CMP3	COUT	-
XBAR3	XBAR_IN7	CMP4	COUT	-
XBAR3	XBAR_IN8	TMR1	TMR0_OUTPUT	-
XBAR3	XBAR_IN9	TMR1	TMR1_OUTPUT	-
XBAR3	XBAR_IN10	TMR1	TMR2_OUTPUT	-
XBAR3	XBAR_IN11	TMR1	TMR3_OUTPUT	-
XBAR3	XBAR_IN12	TMR2	TMR0_OUTPUT	-
XBAR3	XBAR_IN13	TMR2	TMR1_OUTPUT	-
XBAR3	XBAR_IN14	TMR2	TMR2_OUTPUT	-
XBAR3	XBAR_IN15	TMR2	TMR3_OUTPUT	-
XBAR3	XBAR_IN16	TMR3	TMR0_OUTPUT	-
XBAR3	XBAR_IN17	TMR3	TMR1_OUTPUT	-
XBAR3	XBAR_IN18	TMR3	TMR2_OUTPUT	-
XBAR3	XBAR_IN19	TMR3	TMR3_OUTPUT	-
XBAR3	XBAR_IN20	TMR4	TMR0_OUTPUT	-
XBAR3	XBAR_IN21	TMR4	TMR1_OUTPUT	-
XBAR3	XBAR_IN22	TMR4	TMR2_OUTPUT	-
XBAR3	XBAR_IN23	TMR4	TMR3_OUTPUT	-
XBAR3	XBAR_IN24	TMR5	TMR0_OUTPUT	-
XBAR3	XBAR_IN25	TMR5	TMR1_OUTPUT	-
XBAR3	XBAR_IN26	TMR5	TMR2_OUTPUT	-
XBAR3	XBAR_IN27	TMR5	TMR3_OUTPUT	-
XBAR3	XBAR_IN28	TMR6	TMR0_OUTPUT	-
XBAR3	XBAR_IN29	TMR6	TMR1_OUTPUT	-
XBAR3	XBAR_IN30	TMR6	TMR2_OUTPUT	-
XBAR3	XBAR_IN31	TMR6	TMR3_OUTPUT	-
XBAR3	XBAR_IN32	TMR7	TMR0_OUTPUT	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR3	XBAR_IN33	TMR7	TMR1_OUTPUT	-
XBAR3	XBAR_IN34	TMR7	TMR2_OUTPUT	-
XBAR3	XBAR_IN35	TMR7	TMR3_OUTPUT	-
XBAR3	XBAR_IN36	TMR8	TMR0_OUTPUT	-
XBAR3	XBAR_IN37	TMR8	TMR1_OUTPUT	-
XBAR3	XBAR_IN38	TMR8	TMR2_OUTPUT	-
XBAR3	XBAR_IN39	TMR8	TMR3_OUTPUT	-
XBAR3	XBAR_IN40	FLEXPWM1	PWM0_MUX_TRIGGER0	OR
XBAR3	XBAR_IN40	FLEXPWM1	PWM0_MUX_TRIGGER1	OR
XBAR3	XBAR_IN41	FLEXPWM1	PWM1_MUX_TRIGGER0	OR
XBAR3	XBAR_IN41	FLEXPWM1	PWM1_MUX_TRIGGER1	OR
XBAR3	XBAR_IN42	FLEXPWM1	PWM2_MUX_TRIGGER0	OR
XBAR3	XBAR_IN42	FLEXPWM1	PWM2_MUX_TRIGGER1	OR
XBAR3	XBAR_IN43	FLEXPWM1	PWM3_MUX_TRIGGER0	OR
XBAR3	XBAR_IN43	FLEXPWM1	PWM3_MUX_TRIGGER1	OR
XBAR3	XBAR_IN44	FLEXPWM2	PWM0_MUX_TRIGGER0	OR
XBAR3	XBAR_IN44	FLEXPWM2	PWM0_MUX_TRIGGER1	OR
XBAR3	XBAR_IN45	FLEXPWM2	PWM1_MUX_TRIGGER0	OR
XBAR3	XBAR_IN45	FLEXPWM2	PWM1_MUX_TRIGGER1	OR
XBAR3	XBAR_IN46	FLEXPWM2	PWM2_MUX_TRIGGER0	OR
XBAR3	XBAR_IN46	FLEXPWM2	PWM2_MUX_TRIGGER1	OR
XBAR3	XBAR_IN47	FLEXPWM2	PWM3_MUX_TRIGGER0	OR
XBAR3	XBAR_IN47	FLEXPWM2	PWM3_MUX_TRIGGER1	OR
XBAR3	XBAR_IN48	FLEXPWM3	PWM0_MUX_TRIGGER0	OR
XBAR3	XBAR_IN48	FLEXPWM3	PWM0_MUX_TRIGGER1	OR
XBAR3	XBAR_IN49	FLEXPWM3	PWM1_MUX_TRIGGER0	OR
XBAR3	XBAR_IN49	FLEXPWM3	PWM1_MUX_TRIGGER1	OR
XBAR3	XBAR_IN50	FLEXPWM3	PWM2_MUX_TRIGGER0	OR
XBAR3	XBAR_IN50	FLEXPWM3	PWM2_MUX_TRIGGER1	OR
XBAR3	XBAR_IN51	FLEXPWM3	PWM3_MUX_TRIGGER0	OR
XBAR3	XBAR_IN51	FLEXPWM3	PWM3_MUX_TRIGGER1	OR
XBAR3	XBAR_IN52	FLEXPWM4	PWM0_MUX_TRIGGER0	OR

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR3	XBAR_IN52	FLEXPWM4	PWM0_MUX_TRIGGER1	OR
XBAR3	XBAR_IN53	FLEXPWM4	PWM1_MUX_TRIGGER0	OR
XBAR3	XBAR_IN53	FLEXPWM4	PWM1_MUX_TRIGGER1	OR
XBAR3	XBAR_IN54	FLEXPWM4	PWM2_MUX_TRIGGER0	OR
XBAR3	XBAR_IN54	FLEXPWM4	PWM2_MUX_TRIGGER1	OR
XBAR3	XBAR_IN55	FLEXPWM4	PWM3_MUX_TRIGGER0	OR
XBAR3	XBAR_IN55	FLEXPWM4	PWM3_MUX_TRIGGER1	OR
XBAR3	XBAR_IN56	LPIT1	LPIT_TRIG_OUT0	-
XBAR3	XBAR_IN57	LPIT1	LPIT_TRIG_OUT1	-
XBAR3	XBAR_IN58	LPIT1	LPIT_TRIG_OUT2	-
XBAR3	XBAR_IN59	LPIT1	LPIT_TRIG_OUT3	-
XBAR3	XBAR_IN60	LPIT2	LPIT_TRIG_OUT0	-
XBAR3	XBAR_IN61	LPIT2	LPIT_TRIG_OUT1	-
XBAR3	XBAR_IN62	LPIT2	LPIT_TRIG_OUT2	-
XBAR3	XBAR_IN63	LPIT2	LPIT_TRIG_OUT3	-
XBAR3	XBAR_IN64	LPIT3	LPIT_TRIG_OUT0	-
XBAR3	XBAR_IN65	LPIT3	LPIT_TRIG_OUT1	-
XBAR3	XBAR_IN66	LPIT3	LPIT_TRIG_OUT2	-
XBAR3	XBAR_IN67	LPIT3	LPIT_TRIG_OUT3	-
XBAR3	XBAR_IN68	SINC2	PULSE_TRG0	-
XBAR3	XBAR_IN69	SINC2	PULSE_TRG1	-
XBAR3	XBAR_IN70	SINC2	PULSE_TRG2	-
XBAR3	XBAR_IN71	SINC2	PULSE_TRG3	-
XBAR3	XBAR_IN72	eDMA4	DMA_TRIGGER_OUT0	-
XBAR3	XBAR_IN73	eDMA4	DMA_TRIGGER_OUT1	-
XBAR3	XBAR_IN74	eDMA4	DMA_TRIGGER_OUT2	-
XBAR3	XBAR_IN75	eDMA4	DMA_TRIGGER_OUT3	-
XBAR3	XBAR_IN76	eDMA4	DMA_TRIGGER_OUT4	-
XBAR3	XBAR_IN77	eDMA4	DMA_TRIGGER_OUT5	-
XBAR3	XBAR_IN78	eDMA4	DMA_TRIGGER_OUT6	-
XBAR3	XBAR_IN79	eDMA4	DMA_TRIGGER_OUT7	-
XBAR3	XBAR_IN80	eDMA3	DMA_TRIGGER_OUT0	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR3	XBAR_IN81	eDMA3	DMA_TRIGGER_OUT1	-
XBAR3	XBAR_IN82	eDMA3	DMA_TRIGGER_OUT2	-
XBAR3	XBAR_IN83	eDMA3	DMA_TRIGGER_OUT3	-
XBAR3	XBAR_IN84	eDMA3	DMA_TRIGGER_OUT4	-
XBAR3	XBAR_IN85	eDMA3	DMA_TRIGGER_OUT5	-
XBAR3	XBAR_IN86	eDMA3	DMA_TRIGGER_OUT6	-
XBAR3	XBAR_IN87	eDMA3	DMA_TRIGGER_OUT7	-
XBAR3	XBAR_IN88	ADC1	ADC_TCOMP_PULSE0	-
XBAR3	XBAR_IN89	ADC1	ADC_TCOMP_PULSE1	-
XBAR3	XBAR_IN90	ADC1	ADC_TCOMP_PULSE2	-
XBAR3	XBAR_IN91	ADC1	ADC_TCOMP_PULSE3	-
XBAR3	XBAR_IN92	ADC1	ADC_TCOMP_PULSE4	-
XBAR3	XBAR_IN93	ADC1	ADC_TCOMP_PULSE5	-
XBAR3	XBAR_IN94	ADC1	ADC_TCOMP_PULSE6	-
XBAR3	XBAR_IN95	ADC1	ADC_TCOMP_PULSE7	-
XBAR3	XBAR_IN96	ADC2	ADC_TCOMP_PULSE0	-
XBAR3	XBAR_IN97	ADC2	ADC_TCOMP_PULSE1	-
XBAR3	XBAR_IN98	ADC2	ADC_TCOMP_PULSE2	-
XBAR3	XBAR_IN99	ADC2	ADC_TCOMP_PULSE3	-
XBAR3	XBAR_IN100	ADC2	ADC_TCOMP_PULSE4	-
XBAR3	XBAR_IN101	ADC2	ADC_TCOMP_PULSE5	-
XBAR3	XBAR_IN102	ADC2	ADC_TCOMP_PULSE6	-
XBAR3	XBAR_IN103	ADC2	ADC_TCOMP_PULSE7	-
XBAR3	XBAR_IN104	TPM1	TPM_CH_TRIGGER0	-
XBAR3	XBAR_IN105	TPM1	TPM_CH_TRIGGER1	-
XBAR3	XBAR_IN106	TPM1	TPM_CH_TRIGGER2	-
XBAR3	XBAR_IN107	TPM1	TPM_CH_TRIGGER3	-
XBAR3	XBAR_IN108	TPM1	TPM_TRIGGER	-
XBAR3	XBAR_IN109	TPM2	TPM_CH_TRIGGER0	-
XBAR3	XBAR_IN110	TPM2	TPM_CH_TRIGGER1	-
XBAR3	XBAR_IN111	TPM2	TPM_CH_TRIGGER2	-
XBAR3	XBAR_IN112	TPM2	TPM_CH_TRIGGER3	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR3	XBAR_IN113	TPM2	TPM_TRIGGER	-
XBAR3	XBAR_IN114	TPM3	TPM_CH_TRIGGER0	-
XBAR3	XBAR_IN115	TPM3	TPM_CH_TRIGGER1	-
XBAR3	XBAR_IN116	TPM3	TPM_CH_TRIGGER2	-
XBAR3	XBAR_IN117	TPM3	TPM_CH_TRIGGER3	-
XBAR3	XBAR_IN118	TPM3	TPM_TRIGGER	-
XBAR3	XBAR_IN119	TPM4	TPM_CH_TRIGGER0	-
XBAR3	XBAR_IN120	TPM4	TPM_CH_TRIGGER1	-
XBAR3	XBAR_IN121	TPM4	TPM_CH_TRIGGER2	-
XBAR3	XBAR_IN122	TPM4	TPM_CH_TRIGGER3	-
XBAR3	XBAR_IN123	TPM4	TPM_TRIGGER	-
XBAR3	XBAR_IN124	TPM5	TPM_CH_TRIGGER0	-
XBAR3	XBAR_IN125	TPM5	TPM_CH_TRIGGER1	-
XBAR3	XBAR_IN126	TPM5	TPM_CH_TRIGGER2	-
XBAR3	XBAR_IN127	TPM5	TPM_CH_TRIGGER3	-
XBAR3	XBAR_IN128	TPM5	TPM_TRIGGER	-
XBAR3	XBAR_IN129	TPM6	TPM_CH_TRIGGER0	-
XBAR3	XBAR_IN130	TPM6	TPM_CH_TRIGGER1	-
XBAR3	XBAR_IN131	TPM6	TPM_CH_TRIGGER2	-
XBAR3	XBAR_IN132	TPM6	TPM_CH_TRIGGER3	-
XBAR3	XBAR_IN133	TPM6	TPM_TRIGGER	-
XBAR3	XBAR_IN134	LPTMR1	LPTIMER_TRIGGER_DELAY	-
XBAR3	XBAR_IN135	LPTMR2	LPTIMER_TRIGGER_DELAY	-
XBAR3	XBAR_IN136	LPTMR3	LPTIMER_TRIGGER_DELAY	-
XBAR3	XBAR_IN137	NETC	TMR_1588_PP1	-
XBAR3	XBAR_IN138	NETC	TMR_1588_PP2	-
XBAR3	XBAR_IN139	NETC	TMR_1588_PP3	-
XBAR3	XBAR_IN140	SINC1	Channel 0 break due to clock absence detected	OR
XBAR3	XBAR_IN140	SINC1	Channel 0 break due to high limit	OR
XBAR3	XBAR_IN140	SINC1	Channel 0 break due to low limit	OR

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR3	XBAR_IN140	SINC1	Channel 0 break due to short circuit detected	OR
XBAR3	XBAR_IN140	SINC1	Channel 0 break due to window limit	OR
XBAR3	XBAR_IN141	SINC1	Channel 1 break due to clock absence detected	OR
XBAR3	XBAR_IN141	SINC1	Channel 1 break due to high limit	OR
XBAR3	XBAR_IN141	SINC1	Channel 1 break due to low limit	OR
XBAR3	XBAR_IN141	SINC1	Channel 1 break due to short circuit detected	OR
XBAR3	XBAR_IN141	SINC1	Channel 1 break due to window limit	OR
XBAR3	XBAR_IN142	SINC1	Channel 2 break due to clock absence detected	OR
XBAR3	XBAR_IN142	SINC1	Channel 2 break due to high limit	OR
XBAR3	XBAR_IN142	SINC1	Channel 2 break due to low limit	OR
XBAR3	XBAR_IN142	SINC1	Channel 2 break due to short circuit detected	OR
XBAR3	XBAR_IN142	SINC1	Channel 2 break due to window limit	OR
XBAR3	XBAR_IN143	SINC1	Channel 3 break due to clock absence detected	OR
XBAR3	XBAR_IN143	SINC1	Channel 3 break due to high limit	OR
XBAR3	XBAR_IN143	SINC1	Channel 3 break due to low limit	OR
XBAR3	XBAR_IN143	SINC1	Channel 3 break due to short circuit detected	OR
XBAR3	XBAR_IN143	SINC1	Channel 3 break due to window limit	OR
XBAR3	XBAR_IN144	SINC2	Channel 0 break due to clock absence detected	OR
XBAR3	XBAR_IN144	SINC2	Channel 0 break due to high limit	OR
XBAR3	XBAR_IN144	SINC2	Channel 0 break due to low limit	OR
XBAR3	XBAR_IN144	SINC2	Channel 0 break due to short circuit detected	OR
XBAR3	XBAR_IN144	SINC2	Channel 0 break due to window limit	OR
XBAR3	XBAR_IN145	SINC2	Channel 1 break due to clock absence detected	OR
XBAR3	XBAR_IN145	SINC2	Channel 1 break due to high limit	OR
XBAR3	XBAR_IN145	SINC2	Channel 1 break due to low limit	OR

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR3	XBAR_IN145	SINC2	Channel 1 break due to short circuit detected	OR
XBAR3	XBAR_IN145	SINC2	Channel 1 break due to window limit	OR
XBAR3	XBAR_IN146	SINC2	Channel 2 break due to clock absence detected	OR
XBAR3	XBAR_IN146	SINC2	Channel 2 break due to high limit	OR
XBAR3	XBAR_IN146	SINC2	Channel 2 break due to low limit	OR
XBAR3	XBAR_IN146	SINC2	Channel 2 break due to short circuit detected	OR
XBAR3	XBAR_IN146	SINC2	Channel 2 break due to window limit	OR
XBAR3	XBAR_IN147	SINC2	Channel 3 break due to clock absence detected	OR
XBAR3	XBAR_IN147	SINC2	Channel 3 break due to high limit	OR
XBAR3	XBAR_IN147	SINC2	Channel 3 break due to low limit	OR
XBAR3	XBAR_IN147	SINC2	Channel 3 break due to short circuit detected	OR
XBAR3	XBAR_IN147	SINC2	Channel 3 break due to window limit	OR
XBAR3	XBAR_IN148	SINC3	Channel 0 break due to clock absence detected	OR
XBAR3	XBAR_IN148	SINC3	Channel 0 break due to high limit	OR
XBAR3	XBAR_IN148	SINC3	Channel 0 break due to low limit	OR
XBAR3	XBAR_IN148	SINC3	Channel 0 break due to short circuit detected	OR
XBAR3	XBAR_IN148	SINC3	Channel 0 break due to window limit	OR
XBAR3	XBAR_IN149	SINC3	Channel 1 break due to clock absence detected	OR
XBAR3	XBAR_IN149	SINC3	Channel 1 break due to high limit	OR
XBAR3	XBAR_IN149	SINC3	Channel 1 break due to low limit	OR
XBAR3	XBAR_IN149	SINC3	Channel 1 break due to short circuit detected	OR
XBAR3	XBAR_IN149	SINC3	Channel 1 break due to window limit	OR
XBAR3	XBAR_IN150	SINC3	Channel 2 break due to clock absence detected	OR
XBAR3	XBAR_IN150	SINC3	Channel 2 break due to high limit	OR
XBAR3	XBAR_IN150	SINC3	Channel 2 break due to low limit	OR

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR3	XBAR_IN150	SINC3	Channel 2 break due to short circuit detected	OR
XBAR3	XBAR_IN150	SINC3	Channel 2 break due to window limit	OR
XBAR3	XBAR_IN151	SINC3	Channel 3 break due to clock absence detected	OR
XBAR3	XBAR_IN151	SINC3	Channel 3 break due to high limit	OR
XBAR3	XBAR_IN151	SINC3	Channel 3 break due to low limit	OR
XBAR3	XBAR_IN151	SINC3	Channel 3 break due to short circuit detected	OR
XBAR3	XBAR_IN151	SINC3	Channel 3 break due to window limit	OR
XBAR3	XBAR_IN152	-	Reserved	-
XBAR3	XBAR_IN153	-	Reserved	-
XBAR3	XBAR_IN154	-	Reserved	-
XBAR3	XBAR_IN155	-	Reserved	-
XBAR3	XBAR_IN156	eQDC2	POS_MATCH0	-
XBAR3	XBAR_IN157	eQDC2	POS_MATCH1	-
XBAR3	XBAR_IN158	eQDC2	POS_MATCH2	-
XBAR3	XBAR_IN159	eQDC2	POS_MATCH3	-
XBAR3	XBAR_IN160	eQDC2	COMP_FLG0	-
XBAR3	XBAR_IN161	eQDC2	COMP_FLG1	-
XBAR3	XBAR_IN162	eQDC2	COMP_FLG2	-
XBAR3	XBAR_IN163	eQDC2	COMP_FLG3	-
XBAR3	XBAR_IN164	eQDC2	CNT_DN	-
XBAR3	XBAR_IN165	eQDC2	CNT_UP	-
XBAR3	XBAR_IN166	eQDC2	DIR	-
XBAR3	XBAR_IN167	eQDC4	POS_MATCH0	-
XBAR3	XBAR_IN168	eQDC4	POS_MATCH1	-
XBAR3	XBAR_IN169	eQDC4	POS_MATCH2	-
XBAR3	XBAR_IN170	eQDC4	POS_MATCH3	-
XBAR3	XBAR_IN171	eQDC4	COMP_FLG0	-
XBAR3	XBAR_IN172	eQDC4	COMP_FLG1	-
XBAR3	XBAR_IN173	eQDC4	COMP_FLG2	-
XBAR3	XBAR_IN174	eQDC4	COMP_FLG3	-

Table continues on the next page...

Table 14. XBAR Input Assignments (continued)

Destination Module	Destination Signal	Source Module	Source Signal	Logic
XBAR3	XBAR_IN175	eQDC4	CNT_DN	-
XBAR3	XBAR_IN176	eQDC4	CNT_UP	-
XBAR3	XBAR_IN177	eQDC4	DIR	-

4.4.2 XBAR Output Assignments

Table 15. XBAR Output Assignments

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT0	eDMA4	DMA_MUX_REQ154	-
XBAR1	XBAR_OUT1	eDMA4	DMA_MUX_REQ155	-
XBAR1	XBAR_OUT2	eDMA4	DMA_MUX_REQ156	-
XBAR1	XBAR_OUT3	eDMA4	DMA_MUX_REQ157	-
XBAR1	XBAR_OUT4	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_00[MUX_MODE] = 0x1 (XBAR_INOUT04)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT5	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_01[MUX_MODE] = 0x1 (XBAR_INOUT05)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT6	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_02[MUX_MODE] = 0x1 (XBAR_INOUT06)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT7	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_03[MUX_MODE] = 0x1 (XBAR_INOUT07)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT8	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_04[MUX_MODE] = 0x1 (XBAR_INOUT08)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT9	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_05[MUX_MODE] = 0x1 (XBAR_INOUT09)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT10	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_26[MUX_MODE] = 0x2 (XBAR_INOUT10)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT11	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_27[MUX_MODE] = 0x2 (XBAR_INOUT11)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT12	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_28[MUX_MODE] = 0x2 (XBAR_INOUT12)	See IOMUX Controller (IOMUXC) for more information

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT13	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_29[MUX_MODE] = 0x2 (XBAR_INOUT13)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT14	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_30[MUX_MODE] = 0x2 (XBAR_INOUT14) SW_MUX_CTL_PAD_GPIO_EMC_B2_16[MUX_MODE] = 0x6 (XBAR_INOUT14)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT15	-	SW_MUX_CTL_PAD_GPIO_EMC_B1_39[MUX_MODE] = 0x2 (XBAR_INOUT15) SW_MUX_CTL_PAD_GPIO_EMC_B2_17[MUX_MODE] = 0x6 (XBAR_INOUT15)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT16	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_18[MUX_MODE] = 0x6 (XBAR_INOUT16)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT17	-	SW_MUX_CTL_PAD_GPIO_AD_33[MUX_MODE] = 0x2 (XBAR_INOUT17) SW_MUX_CTL_PAD_GPIO_SD_B2_00[MUX_MODE] = 0x3 (XBAR_INOUT17)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT18	-	SW_MUX_CTL_PAD_GPIO_AD_12[MUX_MODE] = 0x6 (XBAR_INOUT18) SW_MUX_CTL_PAD_GPIO_AD_34[MUX_MODE] = 0x2 (XBAR_INOUT18)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT19	-	SW_MUX_CTL_PAD_GPIO_AD_19[MUX_MODE] = 0x2 (XBAR_INOUT19) SW_MUX_CTL_PAD_GPIO_AD_35[MUX_MODE] = 0x2 (XBAR_INOUT19)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT20	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_00[MUX_MODE] = 0x6 (XBAR_INOUT20) SW_MUX_CTL_PAD_GPIO_SD_B1_00[MUX_MODE] = 0x2 (XBAR_INOUT20)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT21	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_01[MUX_MODE] = 0x6 (XBAR_INOUT21)	See IOMUX Controller (IOMUXC) for more information

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
			SW_MUX_CTL_PAD_GPIO_SD_B1_01[MUX_MODE] = 0x2 (XBAR_INOUT21)	
XBAR1	XBAR_OUT22	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_02[MUX_MODE] = 0x6 (XBAR_INOUT22) SW_MUX_CTL_PAD_GPIO_SD_B1_02[MUX_MODE] = 0x2 (XBAR_INOUT22)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT23	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_03[MUX_MODE] = 0x6 (XBAR_INOUT23) SW_MUX_CTL_PAD_GPIO_SD_B1_03[MUX_MODE] = 0x2 (XBAR_INOUT23)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT24	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_04[MUX_MODE] = 0x6 (XBAR_INOUT24) SW_MUX_CTL_PAD_GPIO_AD_30[MUX_MODE] = 0x9 (XBAR_INOUT24)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT25	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_05[MUX_MODE] = 0x6 (XBAR_INOUT25) SW_MUX_CTL_PAD_GPIO_AD_31[MUX_MODE] = 0x9 (XBAR_INOUT25)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT26	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_06[MUX_MODE] = 0x6 (XBAR_INOUT26) SW_MUX_CTL_PAD_GPIO_B1_00[MUX_MODE] = 0x4 (XBAR_INOUT26)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT27	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_07[MUX_MODE] = 0x6 (XBAR_INOUT27) SW_MUX_CTL_PAD_GPIO_B1_01[MUX_MODE] = 0x4 (XBAR_INOUT27)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT28	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_08[MUX_MODE] = 0x6 (XBAR_INOUT28) SW_MUX_CTL_PAD_GPIO_B1_02[MUX_MODE] = 0x4 (XBAR_INOUT28)	See IOMUX Controller (IOMUXC) for more information

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT29	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_09[MUX_MODE] = 0x6 (XBAR_INOUT29) SW_MUX_CTL_PAD_GPIO_B1_03[MUX_MODE] = 0x4 (XBAR_INOUT29)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT30	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_10[MUX_MODE] = 0x6 (XBAR_INOUT30) SW_MUX_CTL_PAD_GPIO_B1_04[MUX_MODE] = 0x4 (XBAR_INOUT30)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT30	TRIGGER_SYNC	TRIGGER_SYNC_IN0	-
XBAR1	XBAR_OUT31	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_11[MUX_MODE] = 0x6 (XBAR_INOUT31) SW_MUX_CTL_PAD_GPIO_B1_05[MUX_MODE] = 0x4 (XBAR_INOUT31)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT31	TRIGGER_SYNC	TRIGGER_SYNC_IN1	-
XBAR1	XBAR_OUT32	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_12[MUX_MODE] = 0x6 (XBAR_INOUT32) SW_MUX_CTL_PAD_GPIO_B1_06[MUX_MODE] = 0x4 (XBAR_INOUT32)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT32	TRIGGER_SYNC	TRIGGER_SYNC_IN2	-
XBAR1	XBAR_OUT33	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_13[MUX_MODE] = 0x6 (XBAR_INOUT33) SW_MUX_CTL_PAD_GPIO_B1_07[MUX_MODE] = 0x4 (XBAR_INOUT33)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT33	TRIGGER_SYNC	TRIGGER_SYNC_IN3	-
XBAR1	XBAR_OUT34	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_14[MUX_MODE] = 0x6 (XBAR_INOUT34) SW_MUX_CTL_PAD_GPIO_B1_10[MUX_MODE] = 0x4 (XBAR_INOUT34)	See IOMUX Controller (IOMUXC) for more information

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT34	TRIGGER_SYNC	TRIGGER_SYNC_IN4	-
XBAR1	XBAR_OUT35	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_15[MUX_MODE] = 0x6 (XBAR_INOUT35) SW_MUX_CTL_PAD_GPIO_B1_11[MUX_MODE] = 0x4 (XBAR_INOUT35)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT35	TRIGGER_SYNC	TRIGGER_SYNC_IN5	-
XBAR1	XBAR_OUT36	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_19[MUX_MODE] = 0x6 (XBAR_INOUT36) SW_MUX_CTL_PAD_GPIO_B1_08[MUX_MODE] = 0x4 (XBAR_INOUT36)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT36	TRIGGER_SYNC	TRIGGER_SYNC_IN6	-
XBAR1	XBAR_OUT37	-	SW_MUX_CTL_PAD_GPIO_EMC_B2_20[MUX_MODE] = 0x6 (XBAR_INOUT37) SW_MUX_CTL_PAD_GPIO_B1_09[MUX_MODE] = 0x4 (XBAR_INOUT37)	See IOMUX Controller (IOMUXC) for more information
XBAR1	XBAR_OUT37	TRIGGER_SYNC	TRIGGER_SYNC_IN7	-
XBAR1	XBAR_OUT38	CMP1	SAMPLE	-
XBAR1	XBAR_OUT39	CMP2	SAMPLE	-
XBAR1	XBAR_OUT40	CMP3	SAMPLE	-
XBAR1	XBAR_OUT41	CMP4	SAMPLE	-
XBAR1	XBAR_OUT42	FLEXPWM1	EXTA0	-
XBAR1	XBAR_OUT43	FLEXPWM1	EXTA1	-
XBAR1	XBAR_OUT44	FLEXPWM1	EXTA2	-
XBAR1	XBAR_OUT45	FLEXPWM1	EXTA3	-
XBAR1	XBAR_OUT46	FLEXPWM1	EXT_SYNC0	-
XBAR1	XBAR_OUT47	FLEXPWM1	EXT_SYNC1	-
XBAR1	XBAR_OUT48	FLEXPWM1	EXT_SYNC2	-
XBAR1	XBAR_OUT49	FLEXPWM1	EXT_SYNC3	-
XBAR1	XBAR_OUT50	FLEXPWM1	EXT_CLK	-

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT51	FLEXPWM1	FAULT0	-
XBAR1	XBAR_OUT52	FLEXPWM1	FAULT1	-
XBAR1	XBAR_OUT53	FLEXPWM1	FAULT2	-
XBAR1	XBAR_OUT53	FLEXPWM2	FAULT2	-
XBAR1	XBAR_OUT53	FLEXPWM3	FAULT2	-
XBAR1	XBAR_OUT53	FLEXPWM4	FAULT2	-
XBAR1	XBAR_OUT54	FLEXPWM1	FAULT3	-
XBAR1	XBAR_OUT54	FLEXPWM2	FAULT3	-
XBAR1	XBAR_OUT54	FLEXPWM3	FAULT3	-
XBAR1	XBAR_OUT54	FLEXPWM4	FAULT3	-
XBAR1	XBAR_OUT55	FLEXPWM1	EXT_FORCE	-
XBAR1	XBAR_OUT56	FLEXPWM2	EXTA0	-
XBAR1	XBAR_OUT57	FLEXPWM2	EXTA1	-
XBAR1	XBAR_OUT58	FLEXPWM2	EXTA2	-
XBAR1	XBAR_OUT59	FLEXPWM2	EXTA3	-
XBAR1	XBAR_OUT60	FLEXPWM2	EXT_SYNC0	-
XBAR1	XBAR_OUT61	FLEXPWM2	EXT_SYNC1	-
XBAR1	XBAR_OUT62	FLEXPWM2	EXT_SYNC2	-
XBAR1	XBAR_OUT63	FLEXPWM2	EXT_SYNC3	-
XBAR1	XBAR_OUT64	FLEXPWM2	EXT_CLK	-
XBAR1	XBAR_OUT65	FLEXPWM2	FAULT0	-
XBAR1	XBAR_OUT66	FLEXPWM2	FAULT1	-
XBAR1	XBAR_OUT67	FLEXPWM2	EXT_FORCE	-
XBAR1	XBAR_OUT68	FLEXPWM3	EXTA0	-
XBAR1	XBAR_OUT68	FLEXPWM4	EXTA0	-
XBAR1	XBAR_OUT69	FLEXPWM3	EXTA1	-
XBAR1	XBAR_OUT69	FLEXPWM4	EXTA1	-
XBAR1	XBAR_OUT70	FLEXPWM3	EXTA2	-
XBAR1	XBAR_OUT70	FLEXPWM4	EXTA2	-
XBAR1	XBAR_OUT71	FLEXPWM3	EXTA3	-
XBAR1	XBAR_OUT71	FLEXPWM4	EXTA3	-
XBAR1	XBAR_OUT72	FLEXPWM3	EXT_CLK	-

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT72	FLEXPWM4	EXT_CLK	-
XBAR1	XBAR_OUT73	FLEXPWM3	EXT_SYNC0	-
XBAR1	XBAR_OUT74	FLEXPWM3	EXT_SYNC1	-
XBAR1	XBAR_OUT75	FLEXPWM3	EXT_SYNC2	-
XBAR1	XBAR_OUT76	FLEXPWM3	EXT_SYNC3	-
XBAR1	XBAR_OUT77	FLEXPWM3	FAULT0	-
XBAR1	XBAR_OUT78	FLEXPWM3	FAULT1	-
XBAR1	XBAR_OUT79	FLEXPWM3	EXT_FORCE	-
XBAR1	XBAR_OUT80	FLEXPWM4	EXT_SYNC0	-
XBAR1	XBAR_OUT81	FLEXPWM4	EXT_SYNC1	-
XBAR1	XBAR_OUT82	FLEXPWM4	EXT_SYNC2	-
XBAR1	XBAR_OUT83	FLEXPWM4	EXT_SYNC3	-
XBAR1	XBAR_OUT84	FLEXPWM4	FAULT0	-
XBAR1	XBAR_OUT85	FLEXPWM4	FAULT1	-
XBAR1	XBAR_OUT86	FLEXPWM4	EXT_FORCE	-
XBAR1	XBAR_OUT87	eQDC1	PHASE_A_INPUT	-
XBAR1	XBAR_OUT88	eQDC1	PHASE_B_INPUT	-
XBAR1	XBAR_OUT89	eQDC1	INDEX	-
XBAR1	XBAR_OUT90	eQDC1	HOME	-
XBAR1	XBAR_OUT91	eQDC1	TRIGGER	-
XBAR1	XBAR_OUT92	eQDC2	PHASE_A_INPUT	-
XBAR1	XBAR_OUT93	eQDC2	PHASE_B_INPUT	-
XBAR1	XBAR_OUT94	eQDC2	INDEX	-
XBAR1	XBAR_OUT95	eQDC2	HOME	-
XBAR1	XBAR_OUT96	eQDC2	TRIGGER	-
XBAR1	XBAR_OUT97	eQDC3	PHASE_A_INPUT	-
XBAR1	XBAR_OUT98	eQDC3	PHASE_B_INPUT	-
XBAR1	XBAR_OUT99	eQDC3	INDEX	-
XBAR1	XBAR_OUT100	eQDC3	HOME	-
XBAR1	XBAR_OUT101	eQDC3	TRIGGER	-
XBAR1	XBAR_OUT102	eQDC4	PHASE_A_INPUT	-
XBAR1	XBAR_OUT103	eQDC4	PHASE_B_INPUT	-

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT104	eQDC4	INDEX	-
XBAR1	XBAR_OUT105	eQDC4	HOME	-
XBAR1	XBAR_OUT106	eQDC4	TRIGGER	-
XBAR1	XBAR_OUT107	TMR1	TMR0_INPUT	When QTIMER_CTRL1[QTIMER1_TMR0_INPUT_SEL] = 1
XBAR1	XBAR_OUT108	TMR1	TMR1_INPUT	When QTIMER_CTRL1[QTIMER1_TMR1_INPUT_SEL] = 1
XBAR1	XBAR_OUT109	TMR1	TMR2_INPUT	When QTIMER_CTRL1[QTIMER1_TMR2_INPUT_SEL] = 1
XBAR1	XBAR_OUT110	TMR1	TMR3_INPUT	When QTIMER_CTRL1[QTIMER1_TMR3_INPUT_SEL] = 1
XBAR1	XBAR_OUT111	TMR2	TMR0_INPUT	When QTIMER_CTRL1[QTIMER2_TMR0_INPUT_SEL] = 1
XBAR1	XBAR_OUT112	TMR2	TMR1_INPUT	When QTIMER_CTRL1[QTIMER2_TMR1_INPUT_SEL] = 1
XBAR1	XBAR_OUT113	TMR2	TMR2_INPUT	When QTIMER_CTRL1[QTIMER2_TMR2_INPUT_SEL] = 1
XBAR1	XBAR_OUT114	TMR2	TMR3_INPUT	When QTIMER_CTRL1[QTIMER2_TMR3_INPUT_SEL] = 1
XBAR1	XBAR_OUT115	TMR3	TMR0_INPUT	When QTIMER_CTRL1[QTIMER3_TMR0_INPUT_SEL] = 1
XBAR1	XBAR_OUT116	TMR3	TMR1_INPUT	When QTIMER_CTRL1[QTIMER3_TMR1_INPUT_SEL] = 1
XBAR1	XBAR_OUT117	TMR3	TMR2_INPUT	When QTIMER_CTRL1[QTIMER3_TMR2_INPUT_SEL] = 1
XBAR1	XBAR_OUT118	TMR3	TMR3_INPUT	When QTIMER_CTRL1[QTIMER3_TMR3_INPUT_SEL] = 1

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT119	TMR4	TMR0_INPUT	When QTIMER_CTRL1[QTIMER4_TMR0_INPUT_SEL] = 1
XBAR1	XBAR_OUT120	TMR4	TMR1_INPUT	When QTIMER_CTRL1[QTIMER4_TMR1_INPUT_SEL] = 1
XBAR1	XBAR_OUT121	TMR4	TMR2_INPUT	When QTIMER_CTRL1[QTIMER4_TMR2_INPUT_SEL] = 1
XBAR1	XBAR_OUT122	TMR4	TMR3_INPUT	When QTIMER_CTRL1[QTIMER4_TMR3_INPUT_SEL] = 1
XBAR1	XBAR_OUT123	TMR5	TMR0_INPUT	When QTIMER_CTRL2[QTIMER5_TMR0_INPUT_SEL] = 1
XBAR1	XBAR_OUT124	TMR5	TMR1_INPUT	When QTIMER_CTRL2[QTIMER5_TMR1_INPUT_SEL] = 1
XBAR1	XBAR_OUT125	TMR5	TMR2_INPUT	When QTIMER_CTRL2[QTIMER5_TMR2_INPUT_SEL] = 1
XBAR1	XBAR_OUT126	TMR5	TMR3_INPUT	When QTIMER_CTRL2[QTIMER5_TMR3_INPUT_SEL] = 1
XBAR1	XBAR_OUT127	TMR6	TMR0_INPUT	When QTIMER_CTRL2[QTIMER6_TMR0_INPUT_SEL] = 1
XBAR1	XBAR_OUT128	TMR6	TMR1_INPUT	When QTIMER_CTRL2[QTIMER6_TMR1_INPUT_SEL] = 1
XBAR1	XBAR_OUT129	TMR6	TMR2_INPUT	When QTIMER_CTRL2[QTIMER6_TMR2_INPUT_SEL] = 1
XBAR1	XBAR_OUT130	TMR6	TMR3_INPUT	When QTIMER_CTRL2[QTIMER6_TMR3_INPUT_SEL] = 1
XBAR1	XBAR_OUT131	TMR7	TMR0_INPUT	When QTIMER_CTRL2[QTIMER7_TMR0_INPUT_SEL] = 1
XBAR1	XBAR_OUT132	TMR7	TMR1_INPUT	When QTIMER_CTRL2[QTIMER7_TMR1_INPUT_SEL] = 1

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT133	TMR7	TMR2_INPUT	When QTIMER_CTRL2[QTIMER7_TMR2_INPUT_SEL] = 1
XBAR1	XBAR_OUT134	TMR7	TMR3_INPUT	When QTIMER_CTRL2[QTIMER7_TMR3_INPUT_SEL] = 1
XBAR1	XBAR_OUT135	TMR8	TMR0_INPUT	When QTIMER_CTRL2[QTIMER8_TMR0_INPUT_SEL] = 1
XBAR1	XBAR_OUT136	TMR8	TMR1_INPUT	When QTIMER_CTRL2[QTIMER8_TMR1_INPUT_SEL] = 1
XBAR1	XBAR_OUT137	TMR8	TMR2_INPUT	When QTIMER_CTRL2[QTIMER8_TMR2_INPUT_SEL] = 1
XBAR1	XBAR_OUT138	TMR8	TMR3_INPUT	When QTIMER_CTRL2[QTIMER8_TMR3_INPUT_SEL] = 1
XBAR1	XBAR_OUT139	EWM	EWM_IN	-
XBAR1	XBAR_OUT140	ADC1	HW_TRG0	-
XBAR1	XBAR_OUT140	ADC2	HW_TRG0	-
XBAR1	XBAR_OUT141	ADC1	HW_TRG1	-
XBAR1	XBAR_OUT141	ADC2	HW_TRG1	-
XBAR1	XBAR_OUT142	ADC1	HW_TRG2	-
XBAR1	XBAR_OUT142	ADC2	HW_TRG2	-
XBAR1	XBAR_OUT143	ADC1	HW_TRG3	-
XBAR1	XBAR_OUT143	ADC2	HW_TRG3	-
XBAR1	XBAR_OUT144	ADC1	HW_TRG4	-
XBAR1	XBAR_OUT144	ADC2	HW_TRG4	-
XBAR1	XBAR_OUT145	ADC1	HW_TRG5	-
XBAR1	XBAR_OUT145	ADC2	HW_TRG5	-
XBAR1	XBAR_OUT146	ADC1	HW_TRG6	-
XBAR1	XBAR_OUT146	ADC2	HW_TRG6	-
XBAR1	XBAR_OUT147	ADC1	HW_TRG7	-
XBAR1	XBAR_OUT147	ADC2	HW_TRG7	-
XBAR1	XBAR_OUT148	SINC1	EXT_TRIGGER0	-

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT148	SINC2	EXT_TRIGGER0	-
XBAR1	XBAR_OUT148	SINC3	EXT_TRIGGER0	-
XBAR1	XBAR_OUT149	SINC1	EXT_TRIGGER1	-
XBAR1	XBAR_OUT149	SINC2	EXT_TRIGGER1	-
XBAR1	XBAR_OUT149	SINC3	EXT_TRIGGER1	-
XBAR1	XBAR_OUT150	SINC1	EXT_TRIGGER2	-
XBAR1	XBAR_OUT150	SINC2	EXT_TRIGGER2	-
XBAR1	XBAR_OUT150	SINC3	EXT_TRIGGER2	-
XBAR1	XBAR_OUT151	SINC1	EXT_TRIGGER3	-
XBAR1	XBAR_OUT151	SINC2	EXT_TRIGGER3	-
XBAR1	XBAR_OUT151	SINC3	EXT_TRIGGER3	-
XBAR1	XBAR_OUT152	FLEXIO1	FLEXIO_TRIGGER_IN0	-
XBAR1	XBAR_OUT153	FLEXIO1	FLEXIO_TRIGGER_IN1	-
XBAR1	XBAR_OUT154	FLEXIO2	FLEXIO_TRIGGER_IN0	-
XBAR1	XBAR_OUT155	FLEXIO2	FLEXIO_TRIGGER_IN1	-
XBAR1	XBAR_OUT156	LPI2C1	LPI2C_TRG_INPUT	-
XBAR1	XBAR_OUT157	LPI2C2	LPI2C_TRG_INPUT	-
XBAR1	XBAR_OUT158	LPI2C3	LPI2C_TRG_INPUT	-
XBAR1	XBAR_OUT159	Reserved	LPI2C_TRG_INPUT	-
XBAR1	XBAR_OUT160	Reserved	LPI2C_TRG_INPUT	-
XBAR1	XBAR_OUT161	LPI2C6	LPI2C_TRG_INPUT	-
XBAR1	XBAR_OUT162	LPSPI1	LPSPI_TRG_INPUT	-
XBAR1	XBAR_OUT163	LPSPI2	LPSPI_TRG_INPUT	-
XBAR1	XBAR_OUT164	LPSPI3	LPSPI_TRG_INPUT	-
XBAR1	XBAR_OUT165	LPSPI4	LPSPI_TRG_INPUT	-
XBAR1	XBAR_OUT166	LPSPI5	LPSPI_TRG_INPUT	-
XBAR1	XBAR_OUT167	LPSPI6	LPSPI_TRG_INPUT	-
XBAR1	XBAR_OUT168	LPUART1	LPUART_TRG_INPUT	-
XBAR1	XBAR_OUT169	LPUART2	LPUART_TRG_INPUT	-
XBAR1	XBAR_OUT170	LPUART3	LPUART_TRG_INPUT	-
XBAR1	XBAR_OUT171	LPUART4	LPUART_TRG_INPUT	-
XBAR1	XBAR_OUT172	LPUART5	LPUART_TRG_INPUT	-

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT173	LPUART6	LPUART_TRG_INPUT	-
XBAR1	XBAR_OUT174	LPUART7	LPUART_TRG_INPUT	-
XBAR1	XBAR_OUT175	LPUART8	LPUART_TRG_INPUT	-
XBAR1	XBAR_OUT176 - XBAR_OUT179	Reserved	-	-
XBAR1	XBAR_OUT180	LPIT1	LPIT_EXT_TRIG_IN0	-
XBAR1	XBAR_OUT180	LPIT2	LPIT_EXT_TRIG_IN0	-
XBAR1	XBAR_OUT180	LPIT3	LPIT_EXT_TRIG_IN0	-
XBAR1	XBAR_OUT181	LPIT1	LPIT_EXT_TRIG_IN1	-
XBAR1	XBAR_OUT181	LPIT2	LPIT_EXT_TRIG_IN1	-
XBAR1	XBAR_OUT181	LPIT3	LPIT_EXT_TRIG_IN1	-
XBAR1	XBAR_OUT182	LPIT1	LPIT_EXT_TRIG_IN2	-
XBAR1	XBAR_OUT182	LPIT2	LPIT_EXT_TRIG_IN2	-
XBAR1	XBAR_OUT182	LPIT3	LPIT_EXT_TRIG_IN2	-
XBAR1	XBAR_OUT183	LPIT1	LPIT_EXT_TRIG_IN3	-
XBAR1	XBAR_OUT183	LPIT2	LPIT_EXT_TRIG_IN3	-
XBAR1	XBAR_OUT183	LPIT3	LPIT_EXT_TRIG_IN3	-
XBAR1	XBAR_OUT184	TPM1	TPM_TRIGGER_IN0	-
XBAR1	XBAR_OUT184	TPM2	TPM_TRIGGER_IN0	-
XBAR1	XBAR_OUT184	TPM3	TPM_TRIGGER_IN0	-
XBAR1	XBAR_OUT185	TPM1	TPM_TRIGGER_IN1	-
XBAR1	XBAR_OUT186	TPM2	TPM_TRIGGER_IN1	-
XBAR1	XBAR_OUT187	TPM3	TPM_TRIGGER_IN1	-
XBAR1	XBAR_OUT188	TPM1	TPM_TRIGGER_IN2	-
XBAR1	XBAR_OUT188	TPM2	TPM_TRIGGER_IN2	-
XBAR1	XBAR_OUT188	TPM3	TPM_TRIGGER_IN2	-
XBAR1	XBAR_OUT189	TPM1	TPM_TRIGGER_IN3	-
XBAR1	XBAR_OUT190	TPM2	TPM_TRIGGER_IN3	-
XBAR1	XBAR_OUT191	TPM3	TPM_TRIGGER_IN3	-
XBAR1	XBAR_OUT192	TPM4	TPM_TRIGGER_IN0	-
XBAR1	XBAR_OUT192	TPM5	TPM_TRIGGER_IN0	-
XBAR1	XBAR_OUT192	TPM6	TPM_TRIGGER_IN0	-

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR1	XBAR_OUT193	TPM4	TPM_TRIGGER_IN1	-
XBAR1	XBAR_OUT194	TPM5	TPM_TRIGGER_IN1	-
XBAR1	XBAR_OUT195	TPM6	TPM_TRIGGER_IN1	-
XBAR1	XBAR_OUT196	TPM4	TPM_TRIGGER_IN2	-
XBAR1	XBAR_OUT196	TPM5	TPM_TRIGGER_IN2	-
XBAR1	XBAR_OUT196	TPM6	TPM_TRIGGER_IN2	-
XBAR1	XBAR_OUT197	TPM4	TPM_TRIGGER_IN3	-
XBAR1	XBAR_OUT198	TPM5	TPM_TRIGGER_IN3	-
XBAR1	XBAR_OUT199	TPM6	TPM_TRIGGER_IN3	-
XBAR1	XBAR_OUT200	NETC	TMR_1588_TRIG1	-
XBAR1	XBAR_OUT201	NETC	TMR_1588_TRIG2	-
XBAR1	XBAR_OUT202	-	Reserved	-
XBAR1	XBAR_OUT203	-	Reserved	-
XBAR1	XBAR_OUT204	-	Reserved	-
XBAR1	XBAR_OUT205	-	Reserved	-
XBAR1	XBAR_OUT206	eQDC1	ICAP1	-
XBAR1	XBAR_OUT207	eQDC1	ICAP2	-
XBAR1	XBAR_OUT208	eQDC1	ICAP3	-
XBAR1	XBAR_OUT209	eQDC2	ICAP1	-
XBAR1	XBAR_OUT210	eQDC2	ICAP2	-
XBAR1	XBAR_OUT211	eQDC2	ICAP3	-
XBAR1	XBAR_OUT212	eQDC3	ICAP1	-
XBAR1	XBAR_OUT213	eQDC3	ICAP2	-
XBAR1	XBAR_OUT214	eQDC3	ICAP3	-
XBAR1	XBAR_OUT215	eQDC4	ICAP1	-
XBAR1	XBAR_OUT216	eQDC4	ICAP2	-
XBAR1	XBAR_OUT217	eQDC4	ICAP3	-
XBAR1	XBAR_OUT218 - XBAR_OUT219	Reserved	1-	-
XBAR1	XBAR_OUT220	SAFETY_CLK_MON	DUT_CLK	-
XBAR2	XBAR2_XBAR_OUT0	AOI1	AOI_IN0	-

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR2	XBAR2_XBAR_OUT1	AOI1	AOI_IN1	-
XBAR2	XBAR2_XBAR_OUT2	AOI1	AOI_IN2	-
XBAR2	XBAR2_XBAR_OUT3	AOI1	AOI_IN3	-
XBAR2	XBAR2_XBAR_OUT4	AOI1	AOI_IN4	-
XBAR2	XBAR2_XBAR_OUT5	AOI1	AOI_IN5	-
XBAR2	XBAR2_XBAR_OUT6	AOI1	AOI_IN6	-
XBAR2	XBAR2_XBAR_OUT7	AOI1	AOI_IN7	-
XBAR2	XBAR2_XBAR_OUT8	AOI1	AOI_IN8	-
XBAR2	XBAR2_XBAR_OUT9	AOI1	AOI_IN9	-
XBAR2	XBAR2_XBAR_OUT10	AOI1	AOI_IN10	-
XBAR2	XBAR2_XBAR_OUT11	AOI1	AOI_IN11	-
XBAR2	XBAR2_XBAR_OUT12	AOI1	AOI_IN12	-
XBAR2	XBAR2_XBAR_OUT13	AOI1	AOI_IN13	-
XBAR2	XBAR2_XBAR_OUT14	AOI1	AOI_IN14	-
XBAR2	XBAR2_XBAR_OUT15	AOI1	AOI_IN15	-
XBAR2	XBAR2_XBAR_OUT16	AOI3	AOI_IN0	-
XBAR2	XBAR2_XBAR_OUT17	AOI3	AOI_IN1	-
XBAR2	XBAR2_XBAR_OUT18	AOI3	AOI_IN2	-
XBAR2	XBAR2_XBAR_OUT19	AOI3	AOI_IN3	-

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR2	XBAR2_XBAR_OUT20	AOI3	AOI_IN4	-
XBAR2	XBAR2_XBAR_OUT21	AOI3	AOI_IN5	-
XBAR2	XBAR2_XBAR_OUT22	AOI3	AOI_IN6	-
XBAR2	XBAR2_XBAR_OUT23	AOI3	AOI_IN7	-
XBAR2	XBAR2_XBAR_OUT24	AOI3	AOI_IN8	-
XBAR2	XBAR2_XBAR_OUT25	AOI3	AOI_IN9	-
XBAR2	XBAR2_XBAR_OUT26	AOI3	AOI_IN10	-
XBAR2	XBAR2_XBAR_OUT27	AOI3	AOI_IN11	-
XBAR2	XBAR2_XBAR_OUT28	AOI3	AOI_IN12	-
XBAR2	XBAR2_XBAR_OUT29	AOI3	AOI_IN13	-
XBAR2	XBAR2_XBAR_OUT30	AOI3	AOI_IN14	-
XBAR2	XBAR2_XBAR_OUT31	AOI3	AOI_IN15	-
XBAR3	XBAR3_XBAR_OUT0	AOI2	AOI_IN0	-
XBAR3	XBAR3_XBAR_OUT1	AOI2	AOI_IN1	-
XBAR3	XBAR3_XBAR_OUT2	AOI2	AOI_IN2	-
XBAR3	XBAR3_XBAR_OUT3	AOI2	AOI_IN3	-
XBAR3	XBAR3_XBAR_OUT4	AOI2	AOI_IN4	-
XBAR3	XBAR3_XBAR_OUT5	AOI2	AOI_IN5	-
XBAR3	XBAR3_XBAR_OUT6	AOI2	AOI_IN6	-

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR3	XBAR3_XBAR_OUT7	AOI2	AOI_IN7	-
XBAR3	XBAR3_XBAR_OUT8	AOI2	AOI_IN8	-
XBAR3	XBAR3_XBAR_OUT9	AOI2	AOI_IN9	-
XBAR3	XBAR3_XBAR_OUT10	AOI2	AOI_IN10	-
XBAR3	XBAR3_XBAR_OUT11	AOI2	AOI_IN11	-
XBAR3	XBAR3_XBAR_OUT12	AOI2	AOI_IN12	-
XBAR3	XBAR3_XBAR_OUT13	AOI2	AOI_IN13	-
XBAR3	XBAR3_XBAR_OUT14	AOI2	AOI_IN14	-
XBAR3	XBAR3_XBAR_OUT15	AOI2	AOI_IN15	-
XBAR3	XBAR3_XBAR_OUT16	AOI4	AOI_IN0	-
XBAR3	XBAR3_XBAR_OUT17	AOI4	AOI_IN1	-
XBAR3	XBAR3_XBAR_OUT18	AOI4	AOI_IN2	-
XBAR3	XBAR3_XBAR_OUT19	AOI4	AOI_IN3	-
XBAR3	XBAR3_XBAR_OUT20	AOI4	AOI_IN4	-
XBAR3	XBAR3_XBAR_OUT21	AOI4	AOI_IN5	-
XBAR3	XBAR3_XBAR_OUT22	AOI4	AOI_IN6	-
XBAR3	XBAR3_XBAR_OUT23	AOI4	AOI_IN7	-
XBAR3	XBAR3_XBAR_OUT24	AOI4	AOI_IN8	-
XBAR3	XBAR3_XBAR_OUT25	AOI4	AOI_IN9	-

Table continues on the next page...

Table 15. XBAR Output Assignments (continued)

Source Module	Source Signal	Destination Module	Destination Signal	Control
XBAR3	XBAR3_XBAR_OUT26	AOI4	AOI_IN10	-
XBAR3	XBAR3_XBAR_OUT27	AOI4	AOI_IN11	-
XBAR3	XBAR3_XBAR_OUT28	AOI4	AOI_IN12	-
XBAR3	XBAR3_XBAR_OUT29	AOI4	AOI_IN13	-
XBAR3	XBAR3_XBAR_OUT30	AOI4	AOI_IN14	-
XBAR3	XBAR3_XBAR_OUT31	AOI4	AOI_IN15	-

Chapter 5
Enhanced Direct Memory Access (eDMA3)

5.1 Chip-specific eDMA3 Information

Table 16. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

5.2 Overview

The enhanced direct memory access (eDMA) controller is capable of performing complex data transfers with minimal intervention from a host processor. The hardware microarchitecture includes:

- A DMA engine that performs:
  - Source address and destination address calculations
  - Data-movement operations
- Local memory containing transfer control descriptors for each of the 32 channels

5.2.1 Block diagram

Figure 9 illustrates the components of the eDMA system, including the eDMA module (engine).

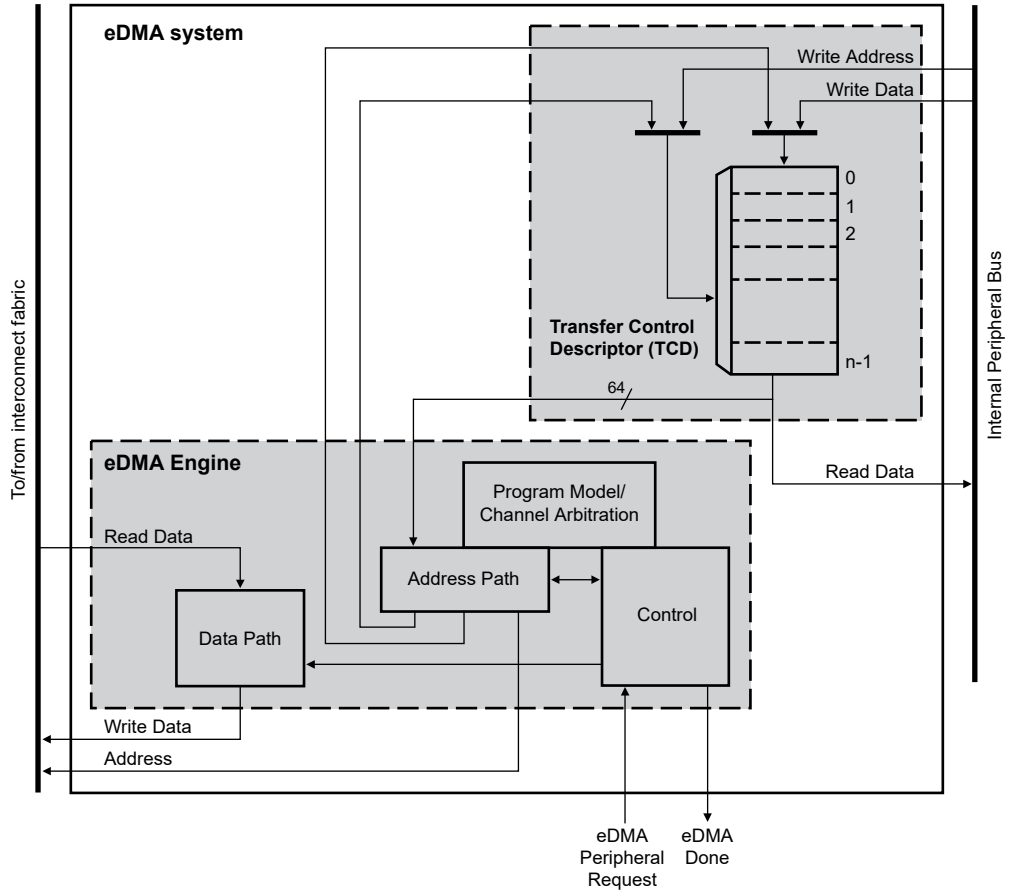


Figure 9. Block diagram

5.2.2 Block parts

The eDMA module is partitioned into two major modules: the eDMA engine and the transfer control descriptor local memory.

The eDMA engine is further partitioned into four submodules:

Table 17. eDMA engine submodules

Submodule	Function
Address path	This block: • Implements a primary channel and secondary (preempt) channel • Manages all master bus-address calculations All the channels provide the same functionality. This structure allows data transfers associated with one channel to be preempted after the completion of a read/write sequence if a higher priority channel activation is asserted while the primary channel is active. After a channel is activated, it runs until the minor loop is completed, unless preempted by a higher priority channel. This provides a mechanism (enabled by CH*n*_PRI[ECP]) where a large data transfer can be preempted to minimize the time another channel is blocked from execution.

Table continues on the next page...

Table 17. eDMA engine submodules (continued)

Submodule	Function
	When any channel is selected to execute, the contents of its TCD are read from local memory and loaded into the address path channel x registers for a normal start and into channel y registers for a preemption start. After the minor loop completes execution, the address path hardware writes the new values for the TCD*n*_{SADDR, DADDR, CITER} back to local memory. If the major iteration count is exhausted, additional processing is performed, including the final address pointer updates, reloading the TCD*n*_CITER field, and a possible fetch of a new TCD*n* from memory as part of a scatter/gather operation. See Dynamic scatter/gather for more details.
Data path	This block implements the bus master read/write data path. It includes a data buffer and the necessary multiplex logic to support any required data alignment. The internal read data bus is the primary input, and the internal write data bus is the primary output. The address and data path modules directly support the 2-stage pipelined internal bus. The address path module represents the first stage of the bus pipeline (address phase), and the data path module implements the second stage of the pipeline (data phase).
Program model/channel arbitration	This block implements the first section of the eDMA programming model as well as the channel arbitration logic. The programming model registers are connected to the internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs are also connected to this block (via control logic).
Control	This block provides all the control functions for the eDMA engine. For data transfers where the source and destination sizes are equal, the eDMA engine performs a series of source read/destination write operations until the number of bytes specified in the minor loop byte count has been moved from the source to the destination. For descriptors where the sizes are not equal, multiple accesses of the smaller size data are required for each reference of the larger size. As an example, if the source size references 16-bit data and the destination is 32-bit data, the eDMA performs two reads, then one 32-bit write.

The transfer control descriptor local memory is further partitioned into:

Table 18. Transfer control descriptor memory

Submodule	Description
Memory controller	This logic implements the required dual-ported controller, and manages accesses from the eDMA engine as well as references from the internal peripheral bus. In simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage for each channel's transfer profile.

5.2.3 Features

The eDMA is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known and is not defined within the transferred data itself. The eDMA module features:

- All data movement via dual-address transfers: read from source, write to destination
  - Programmable source and destination addresses and transfer size
  - Support for complex address calculations
- 32-channel implementation that performs complex data transfers with minimal intervention from a host processor
  - Internal data buffer, used as temporary storage for all transfers

  - Connections to the crossbar switch for bus mastering the data movement
- TCD organized to support two-deep, nested transfer operations
  - 32-byte TCD stored in local memory for each channel
  - An inner data transfer loop defined by a minor byte transfer count
  - An outer data transfer loop defined by a major iteration count
- Channel activation via one of three methods:
  - Explicit software initiation
  - Initiation via a channel-to-channel linking mechanism for continuous transfers
  - Peripheral-paced hardware requests, one per channel
- Fixed-priority and round-robin channel arbitration
- Channel completion reported via programmable interrupt requests
  - One interrupt per channel, which can be asserted at completion of major iteration count
  - Programmable error terminations per channel that are logically summed together to form one error interrupt to the interrupt controller
- Programmable support for scatter/gather DMA processing
- Support for complex data structures

In the discussion of this module, *n* is used to reference the channel number.

5.3 Functional description

The operation of eDMA is described in the following subsections.

5.3.1 Modes of operation

The eDMA operates in the following modes:

Table 19. Modes of operation

Mode	Description
Normal	In Normal mode, eDMA transfers data between a source and a destination. The source and destination can be a memory block or an I/O block capable of operation with eDMA. A service request initiates a transfer of a specific number of bytes (NBYTES) as specified in the TCD. The minor loop is the sequence of read-write operations that transfers these NBYTES per service request. Each service request executes one iteration of the major loop, which transfers NBYTES of data.
Debug	eDMA operation is configurable in Debug mode via the control register: • If CSR[EDBG] is cleared to 0, eDMA continues to operate. • If CSR[EDBG] is set to 1, eDMA stops transferring data. If Debug mode is entered when a channel is active, eDMA continues operation until the channel retires.

5.3.2 eDMA basic data flow

The basic flow of a data transfer can be partitioned into three segments.

As shown in the following diagram, the first segment involves the channel activation:

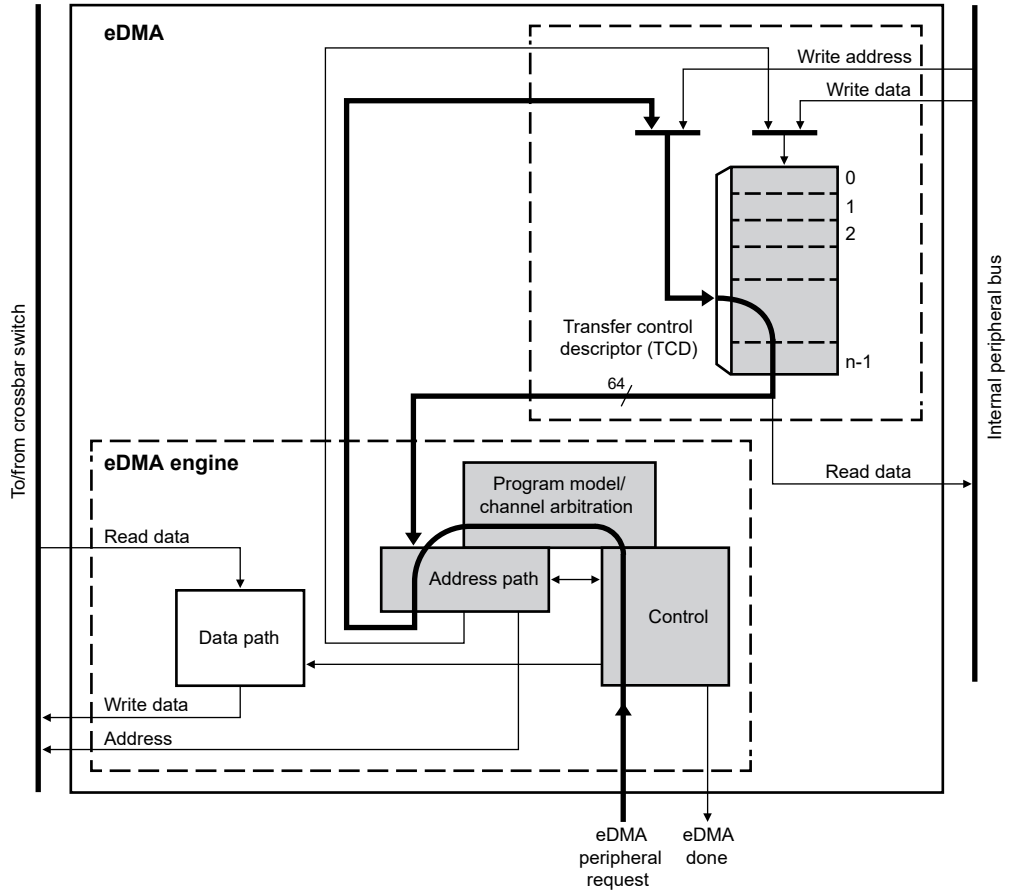


Figure 10. eDMA operation, part 1

This example uses the assertion of the eDMA peripheral request signal to request service for channel *n*. Channel activation via software and the TCD*n*_CSR[START] field follows the same basic flow as peripheral requests. The eDMA request input signal is registered internally and then routed through the eDMA engine: first through the control module, then into the program model and channel arbitration.

In the next cycle, the channel arbitration begins using fixed-priority plus the optional round-robin algorithm. After arbitration is complete, the activated channel number is sent through the address path and converted into the required address to access the local memory for TCD*n*. Next, the TCD memory is accessed and the required descriptor is read from the local memory and then loaded into the eDMA engine address path's primary or secondary channel execution registers. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the address path registers.

The following diagram illustrates the second part of the basic data flow:

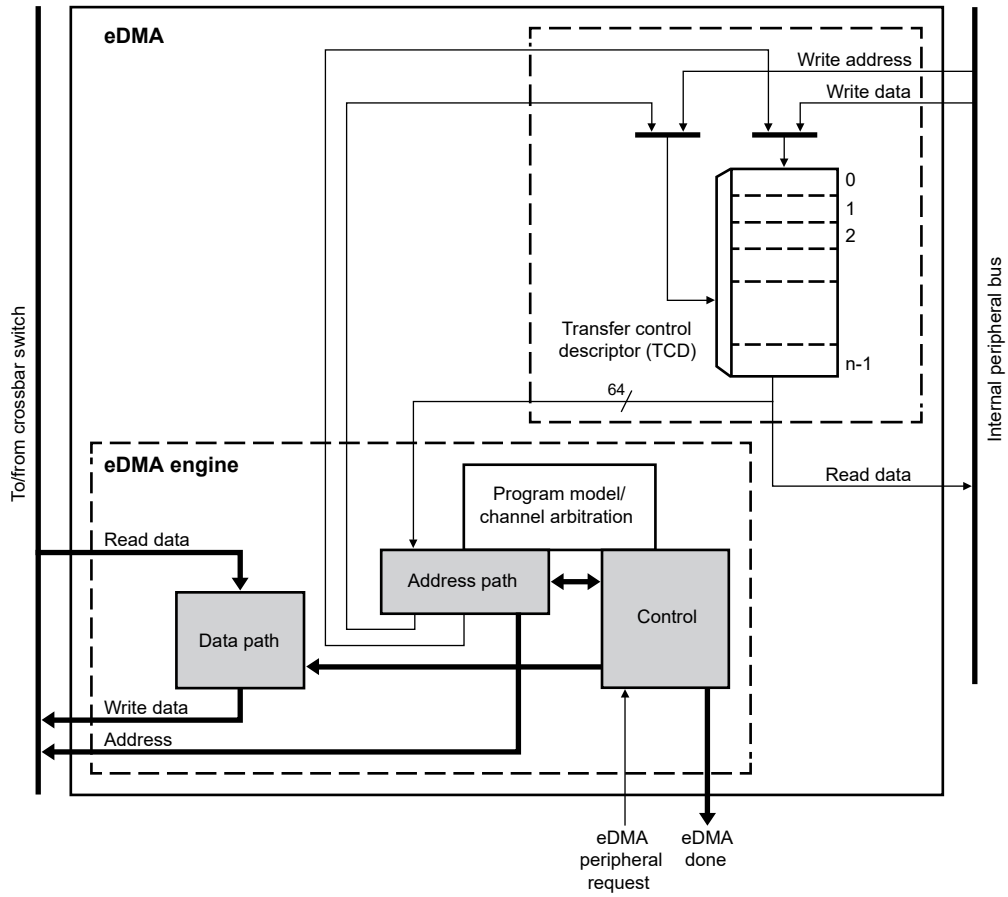


Figure 11. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) go through the required sequence of source reads and destination writes to perform the actual data movement. The source reads are initiated, and the fetched data is temporarily stored in the data path block until it is gated onto the internal bus during the destination write. This source read/destination write processing continues until the byte count, NBYTES, has been transferred.

After NBYTES of data has been moved, the final phase of the basic data flow is performed. In this segment, the address path logic performs the required updates to certain fields in the appropriate TCD (for example, SADDR, DADDR, CITER). If the major iteration count is exhausted, additional operations are performed. These include the final address adjustments and reloading of the BITER field into the CITER field. Assertion of an optional interrupt request also occurs at this time, as does a possible fetch of a new TCD from memory using the scatter/gather address pointer included in the descriptor (if scatter/gather is enabled). The updates to the TCD memory and the assertion of an interrupt request are shown in the following diagram.

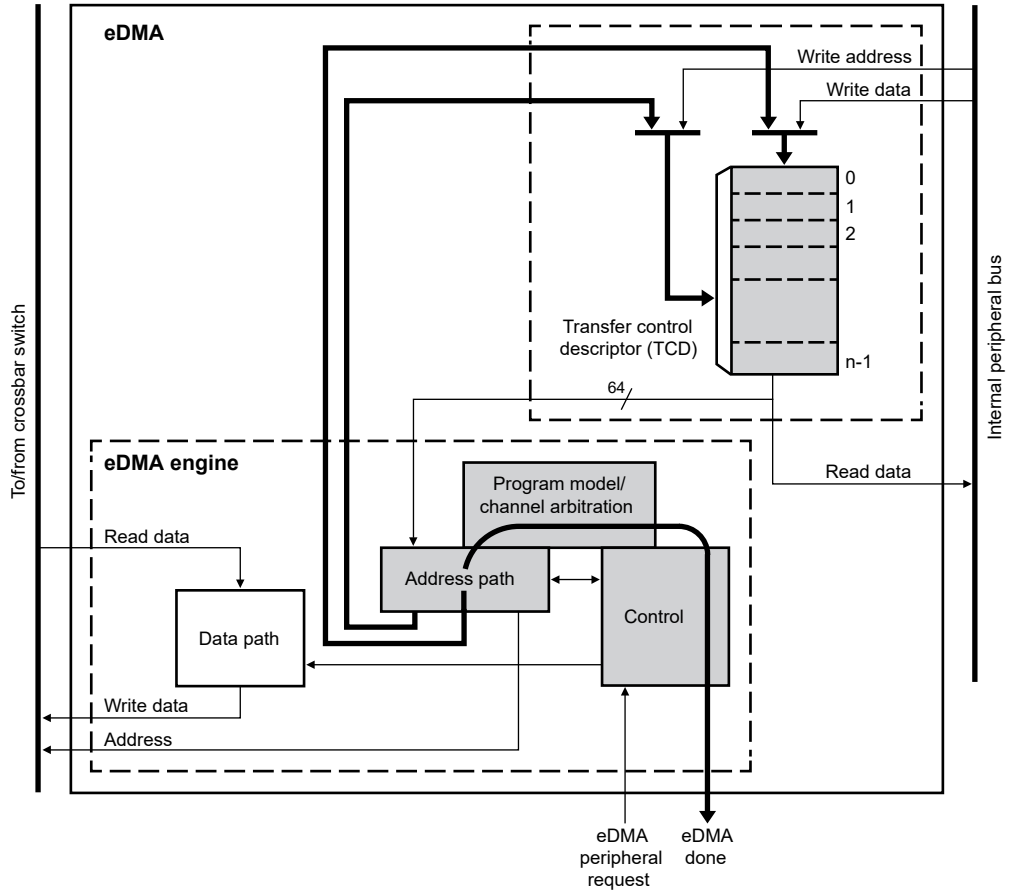


Figure 12. eDMA operation, part 3

5.3.3 Fault reporting and handling

Channel errors are reported in the Error Status register (CHn_ES) and can be caused by any of the following:

- A configuration error, which is an illegal setting in the transfer control descriptor
- An active channel canceled via a "cancel transfer with error" hardware or software request
- An error termination to a bus master read or write cycle

A configuration error is reported when an inconsistent state is represented by one of these factors:

- Starting source or destination address
- Source or destination offsets
- Minor loop byte count
- Transfer size

Each of these possible causes is detailed below:

- The addresses and offsets must be aligned on zero-modulo-transfer-sized boundaries.
- The minor loop byte count must be a multiple of the source and destination transfer sizes.
- All source reads and destination writes must be configured to the natural boundary of the programmed transfer size.

NOTE

To aid in debugging, set the Halt After Error field in the DMA's Control Status register, CSR[HAE]. Upon any error condition, the DMA is halted after the error is recorded. The DMA remains halted and does not process any channel service requests. After the error is fixed, the DMA may be enabled again by clearing the Halt field, CSR[HALT].

- If a scatter/gather operation is enabled upon channel completion, a configuration error is reported if the scatter/gather address (TCDn_DLAST_SGA) is not aligned on a 32-byte boundary.
- If minor loop channel linking is enabled upon channel completion, a configuration error is reported when the link is attempted if the TCDn_CITER[ELINK] field does not equal the TCDn_BITER[ELINK] field.

If enabled, all configuration error conditions, except the scatter/gather and minor-loop link errors, are reported as the channel activates and asserts an error interrupt request. A scatter/gather configuration error is reported when the scatter/gather operation begins at major loop completion if properly enabled. A minor loop channel link configuration error is reported when the link operation is serviced at minor loop completion.

If a system bus read or write is terminated with an error, the data transfer is stopped and the appropriate bus error flag set. In this case, the state of the channel's transfer control descriptor is updated by the eDMA engine with the current source address, destination address, and current iteration count at the point of the fault. When a system bus error occurs, the channel terminates after the next transfer. Due to pipeline effect, the next transfer is already in progress when the bus error is received by the eDMA. If a bus error occurs on the last read prior to beginning the write sequence, the write executes using the data captured during the bus error. If a bus error occurs on the last write prior to switching to the next read sequence, the read sequence executes before the channel terminates due to the destination bus error.

The occurrence of any error causes the eDMA engine to stop normal processing of the active channel immediately (it goes to its error processing states and the transaction to the system bus still has pipeline effect), and the appropriate channel field in the eDMA error register is set to 1. At the same time, the details of the error condition are loaded into the Error Status register (CHn_ES). The major loop complete indicators, setting the transfer control descriptor DONE flag, and the possible assertion of an interrupt request are not affected when an error is detected.

After the error status has been updated, the eDMA engine continues operating by servicing the next appropriate channel. A channel that experiences an error condition is not automatically disabled. If a channel is terminated by an error and then issues another service request before the error is fixed, that channel executes and terminates with the same error condition.

The error status fields are read-only. These error indicators are sticky and cannot be cleared. They show the last recorded error until the DMA is reset. CHn_ES[ERR] is used to determine if a new error condition exists. This field is the logical OR of each channel's error interrupt field (ERR).

After the software has resolved all errors and cleared all of the error interrupt fields, the MP_ES[VLD] is cleared to 0 but the cause of the last error is still indicated.

5.3.4 Channel preemption

The eDMA uses a priority vector value to determine the highest priority channel requesting service.

The priority vector is a combination of:

1. the channel's group priority, CHn_GRPRI
2. the channel's priority level, CHn_PRI[APL]
3. the channel number

Priority vector = ((CHn_GRPRI << 8) + (CHn_PRI[APL] << 5) + CHn_*)

A channel requesting service with the highest priority vector value will receive the next execution slot.

An execution slot is available:

1. immediately if the eDMA is idle
2. when an active channel retires
3. when valid preemption conditions exist

NOTE

Preemption is strictly priority based. Preemption is not bound by a specific group number as defined by CHn_GRPRI.

Channel preemption is enabled on a per-channel basis by setting the CHn_PRI[ECP] field. Channel preemption allows the executing channel's data transfers to temporarily suspend in favor of starting a higher-priority channel. After the preempting channel has completed all of its minor loop data transfers, the preempted channel is restored and resumes execution.

After the restored channel completes one read/write sequence, it is again eligible for preemption. If any higher priority channel is requesting service, the restored channel is suspended, and the higher-priority channel is serviced. Nested preemption, that is, attempting to preempt a preempting channel, is not supported. After a preempting channel begins execution, it cannot be preempted.

A channel's ability to preempt another channel can be disabled by setting CHn_PRI[DPA] to 1. When a channel's preempt ability is disabled, that channel cannot suspend a lower-priority channel's data transfer, regardless of the lower-priority channel's ECP setting. This allows for a pool of low-priority, large-data-moving channels to be defined.

You can configure these low-priority channels to not preempt each other, thus preventing a low-priority channel from consuming the preempt slot normally available to a true high-priority channel. When you enable round-robin channel arbitration mode (CSR[ERCA] is set to 1), any channel with a priority level equal to 0 (CHn_PRI[APL] = 0) has preemption disabled and cannot preempt another channel.

5.3.5 Clocking

This module has no clocking considerations.

5.3.6 Interrupts

Software can enable the interrupt for each channel for the following events:

1. The major loop is half complete (INTHALF)
2. The major loop is complete (INTMAJOR)
3. A configuration error occurs (EEI)

5.4 External signals

This module has no external signals.

5.5 Initialization

The following sections discuss initialization of the eDMA and programming considerations.

5.5.1 eDMA initialization

To initialize the eDMA:

1. Write to the MP_CSR if a configuration other than the default is wanted.
2. Write the channel priority levels to the CHn_PRI registers and group priority levels to the CHn_GRPRI registers if a configuration other than the default is wanted.
3. Enable error interrupts in the CHn_CSR[EEI] registers if they are wanted.
4. Write the 32-byte TCD for each channel that may request service.
5. Enable any hardware service requests via the CHn_CSR[ERQ] registers.
6. Request channel service via either:
   - Software: setting TCDn_CSR[START]
   - Hardware: slave device asserting its eDMA peripheral request signal

After any channel requests service, a channel is selected for execution based on the arbitration and priority levels written into the programmer's model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in Table 20, for the selected channel into its internal address path module.

As the TCD is read, the first transfer is initiated on the internal bus, unless a configuration error is detected. Transfers from the source, defined by TCDn_SADDR, to the destination, defined by TCD_DADDR, continue until the number of bytes specified by TCDn_NBYTES are transferred.

When the transfer is complete, the eDMA engine's local TCDn_SADDR, TCDn_DADDR, and TCDn_CITER are written back to the main TCD memory and any minor loop channel linking is performed, if enabled. If the major loop is exhausted, then eDMA executes further post-processing, such as interrupts, major loop channel linking, and scatter/gather operations, if enabled.

Table 20. TCD control and status (TCDn_CSR) fields

TCDn_CSR field name	Description
START	Control field to start the channel explicitly when using a software-initiated DMA service (automatically cleared by hardware)
EEOP	Control field to enable end-of-packet processing
ESDA	Control field to enable storing of the destination address to system memory after the major loop completes
DREQ	Control field to disable hardware-initiated DMA service requests after major loop completion
BWC	Control field for throttling the bandwidth control of a channel
ESG	Control field to enable the scatter-gather feature
INTHALF	Control field to enable interrupt when major loop is half-complete
INTMAJOR	Control field to enable interrupt when major loop completes

Table 21. Channel control and status (CHn_CSR) fields

CHn_CSR field name	Description
ACTIVE	Status field indicating the channel is currently in execution
DONE	Status field indicating major loop completion (cleared by software when a channel begins execution)
EEI	Control field to enable error interrupts
EARQ	Control field to enable external, asynchronous wakeup event in conjunction with the ERQ field
ERQ	Control field to enable hardware service requests

The following figure shows how each DMA request initiates one minor-loop transfer, or iteration, without CPU intervention. DMA arbitration can occur after each minor loop, and one level of minor loop DMA preemption is allowed. The number of minor loops in a major loop is specified by the beginning iteration count (BITER).

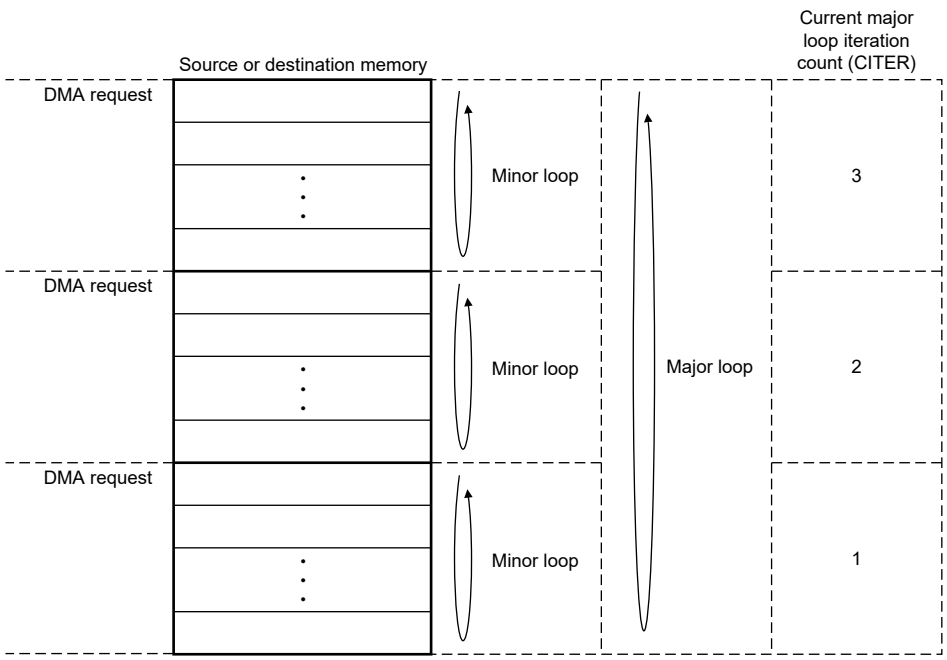


Figure 13. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings are related.

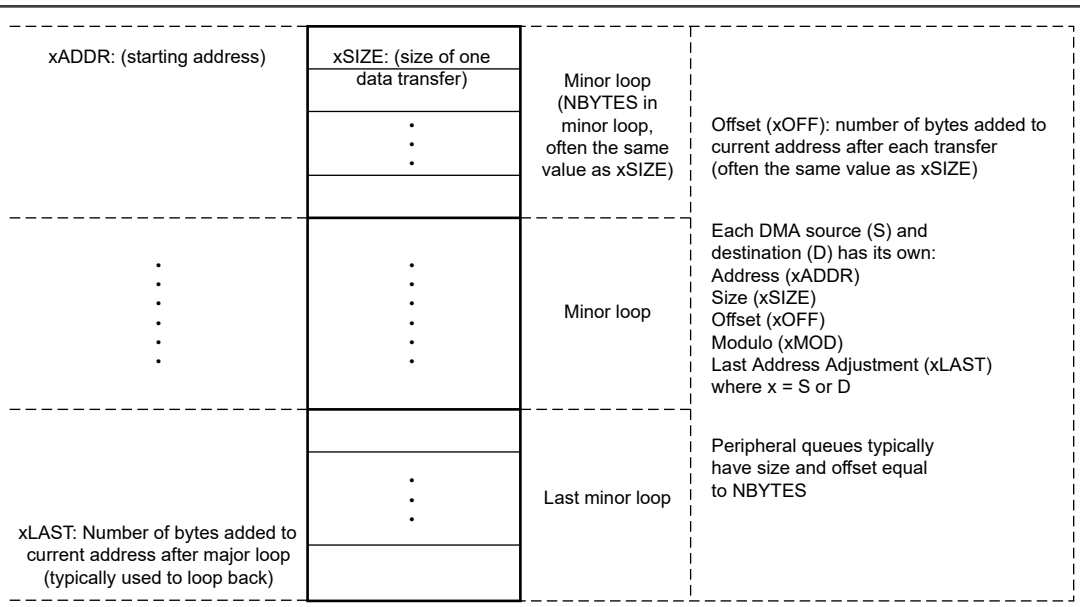


Figure 14. Memory array terms

5.5.2 eDMA arbitration

The eDMA uses a layered arbitration scheme composed of multiple priority levels. The eDMA uses a fixed-priority arbitration scheme with optional round-robin arbitration under specific conditions. The priorities are evaluated in the following order:

Table 22. eDMA arbitration priorities

Priority	Scheme	Description
1 (Highest)	Arbitration group priority	Each channel is assigned an arbitration group via the CHn_GRPRI registers. Priority is given to the highest value (31 being the highest possible value) down to the lowest value (zero, the default).
2	Channel priority	Each channel is assigned a channel priority level via the CHn_PRI registers. The channel priority is a relative priority level within an arbitration group. Priority is given to the highest value (seven being the highest possible value) down to the lowest value (zero, the default). Channel priorities within each arbitration group need not be unique. If multiple channels have the same channel priority level, the channel number will be used to determine priority as defined in row three.
3	Channel number	When two or more channels have the same arbitration group priority and channel priority, the channel number (CHn_NUM) is used to determine the highest priority. Priority is given to the highest channel number. Lowest priority is channel 0. The channel numbers are static and cannot be changed in the programmer's model.
4 (Lowest)	Round-robin	When round-robin is enabled, any channel configured for round-robin operation has lowest priority within an arbitration group. Round-robin is enabled by setting the MP_CSR[ERCA] field to 1. After being enabled, channels with a channel priority of zero (CHn_PRI=0) will use round-robin arbitration. Round-robin arbitration will rotate the channel selection among the channels requesting service with CHn_PRI=0 within the arbitration group. Any non-zero channel within the arbitration group will continue to use fixed-priority arbitration, and if requesting service will be selected over any round-robin channels.

For fixed arbitration, the overall priority can be considered a number composed of three concatenated priority levels: CHn_GRPRI:CHn_PRI:CH_NUM. The largest number has the highest priority and the lowest number has the lowest priority.

For round-robin arbitration, the priority number is CHn_GRPRI:0:X. The module rotates through the CHn_PRI=0 channels requesting service without regard to priority among these channels. Any channel within the arbitration group for which CHn_PRI is greater than 0 will be serviced before the round-robin channels.

5.5.3 Programming errors

The eDMA performs various tests on the transfer control descriptor to verify consistency in the descriptor data.

The channel number causing the error is recorded in the Error Status register (CHn_ES). If the error source is not removed before the next activation of the problematic channel, the error is detected and recorded again. Setting the halt after error field, CSR[HAE], will halt the DMA and prevent recurrence of the error.

5.5.4 Arbitration mode considerations

This section discusses arbitration considerations for eDMA.

5.5.4.1 Fixed group arbitration, fixed channel arbitration

In this mode, eDMA selects for execution the channel service request from the highest-priority channel in the highest-priority group. If eDMA is programmed so that the channels within a high-priority group have a high number of requests or large data transfers, that group may consume all the bandwidth of the eDMA controller. That is, no lower-priority groups are serviced if there is always at least one DMA request pending on a channel in the highest-priority group when the controller arbitrates the next DMA request. The advantage of this scenario is that latency can be small for channels that need to be serviced quickly.

5.5.4.2 Fixed group arbitration, round-robin channel arbitration

The highest-priority group with a request is serviced. Lower-priority groups are serviced if no pending requests exist in the higher-priority groups.

Within each group, channels are serviced starting with the highest non-zero channel priority. For all channels with a channel priority programmed to 0, selection begins with the highest channel number requesting service and then rotates through to the lowest channel number requesting service. The round-robin channel arbitration can provide a fairness mechanism to lower-priority channels.

This scenario could cause the same bandwidth consumption problem as indicated in Fixed group arbitration, fixed channel arbitration, but all the channels in the highest-priority group will be serviced. Service latency is short on the highest-priority group, but could potentially be very much longer as the group priority decreases.

5.5.5 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

5.5.5.1 Single request

To perform a simple transfer of n bytes of data with one activation, set the major loop to one (TCD*n*_CITER = TCD*n*_BITER = 1). The data transfer begins after the channel service request is acknowledged and the channel is selected to execute. After the transfer is complete, the CH*n*_CSR[DONE] field is set to 1 and an interrupt is generated if properly enabled.

For example, the following TCD entry is configured to transfer 16 bytes of data. The eDMA is programmed for one iteration of the major loop transferring 16 bytes per iteration. The source memory has a byte-wide memory port located at 0x1000. The destination memory has a 32-bit port located at 0x2000. The address offsets are programmed in increments to match the transfer size: one byte for the source, and four bytes for the destination. The final source and destination addresses are adjusted to return to their beginning values.

```
TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
```

```
TCDn_DLAST_SGA= -16
TCDn_CSR[INTMAJ] = 1
TCDn_CSR[START] = 1 (should be written last after all other fields have been initialized)
All other TCDn fields = 0
```

This generates the following event sequence:

1. User write to the TCD*n*_CSR[START] field requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes:
   - CH*n*_CSR[DONE] = 0
   - TCD*n*_CSR[START] = 0
   - CH*n*_CSR[ACTIVE] = 1
4. eDMA engine reads: channel TCD data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:
   a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
   b. Write 32 bits to location 0x2000 → first iteration of the minor loop.
   c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
   d. Write 32 bits to location 0x2004 → second iteration of the minor loop.
   e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
   f. Write 32 bits to location 0x2008 → third iteration of the minor loop.
   g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
   h. Write 32 bits to location 0x200C → last iteration of the minor loop → major loop complete.
6. The eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 1 (TCD*n*_BITER).
7. The eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
8. The channel retires and the eDMA goes idle or services the next channel.

5.5.5.2 Multiple requests

The following example transfers 32 bytes via two hardware requests, but is otherwise the same as the previous example. The only fields that change are the major loop iteration count and the final address offsets. The eDMA is programmed for two iterations of the major loop, transferring 16 bytes per iteration. After the channel's hardware requests are enabled via the CHn_CSR[ERQ] register field, the slave device initiates channel service requests.

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

This would generate the following sequence of events:

1. First hardware (eDMA peripheral) requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD*n* data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:

a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
b. Write 32 bits to location 0x2000 → first iteration of the minor loop.
c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
d. Write 32 bits to location 0x2004 → second iteration of the minor loop.
e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
f. Write 32 bits to location 0x2008 → third iteration of the minor loop.
g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
h. Write 32 bits to location 0x200C → last iteration of the minor loop.

6. eDMA engine writes: TCD*n*_SADDR = 0x1010, TCD*n*_DADDR = 0x2010, TCD*n*_CITER = 1.
7. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0.
8. The channel retires, which concludes one iteration of the major loop. The eDMA goes idle or services the next channel.
9. Second hardware (eDMA peripheral) requests channel service.
10. The channel is selected by arbitration for servicing.
11. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
12. eDMA engine reads: Channel TCD data from local memory to internal register file.
13. The source-to-destination transfers are executed as follows:
    a. Read byte from location 0x1010, read byte from location 0x1011, read byte from 0x1012, read byte from 0x1013.
    b. Write 32 bits to location 0x2010 → first iteration of the minor loop.
    c. Read byte from location 0x1014, read byte from location 0x1015, read byte from 0x1016, read byte from 0x1017.
    d. Write 32 bits to location 0x2014 → second iteration of the minor loop.
    e. Read byte from location 0x1018, read byte from location 0x1019, read byte from 0x101A, read byte from 0x101B.
    f. Write 32 bits to location 0x2018 → third iteration of the minor loop.
    g. Read byte from location 0x101C, read byte from location 0x101D, read byte from 0x101E, read byte from 0x101F.
    h. Write 32 bits to location 0x201C → last iteration of the minor loop → major loop complete.
14. eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 2 (TCD*n*_BITER).
15. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
16. The channel retires, which concludes with the major loop complete. The eDMA goes idle or services the next channel.

5.5.5.3 Using the modulo feature

The modulo feature of the eDMA allows implementation of a circular data queue in which the size of the queue is a power of 2. xMOD is a 5-bit field for the source and destination in the TCD, and it specifies which lower address bits increment from their original value after the address+offset calculation. All upper address bits remain the same as in the original value. A setting of 0 for this field disables the modulo feature. Modulo addressing applies to cases where the minor loop offset is enabled; that is, the upper address bits remain the same after the minor loop offset is added to the source or destination address.

The following table shows how the transfer addresses are specified based on the setting of the MOD field. Here a circular buffer is created where the address wraps to the original value but the 28 upper address bits (0x1234567*x*) retain their original value. In this example, the source address is set to 0x12345670, the offset is set to four bytes, and the MOD field is set to four, which allows for a 2^4 byte (16 byte) queue size.

Table 23. Modulo example

Transfer number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

5.5.6 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

5.5.6.1 Testing for minor loop completion

There are two methods to test for minor loop completion when using software-initiated service requests.

1. The first method is to read the TCDn_CITER field and test for a change.
2. The second method, extracted from the sequence shown below, is to test the TCDn_CSR[START] field and the CHn_CSR[ACTIVE] field. The minor-loop-complete condition is indicated by both fields reading 0 after TCDn_CSR[START] is set to 1. Polling the CHn_CSR[ACTIVE] field only may be inconclusive because the active status may be missed if the channel execution is short in duration.

The CHn_CSR and TCDn_CSR status fields execute the following sequence for a software-activated channel:

Stage	TCDn_CSR field	CHn_CSR fields		State
	START	ACTIVE	DONE	
1	1	0	0	Initiate channel service request via software.
2	0	1	0	Channel is executing.
3a	0	0	0	Channel has completed the minor loop and is idle.
3b	0	0	1	Channel has completed the major loop and is idle.

The best method to test for minor-loop completion when using hardware-initiated (that is, peripheral-initiated) service requests is to read the TCDn_CITER field and test for a change. The hardware request and acknowledge handshake signals are not visible in the programmer's model.

The TCD status fields execute the following sequence for a hardware-activated channel:

Stage	TCDn_CSR field	CHn_CSR fields		State
	START	ACTIVE	DONE	
1	0	0	0	Initiate channel service request via hardware (peripheral request asserted).
2	0	1	0	Channel is executing.
3a	0	0	0	Channel has completed the minor loop and is idle.
3b	0	0	1	Channel has completed the major loop and is idle.

For both activation types, the major-loop-complete status is explicitly indicated via the CHn_CSR[DONE] field.

The TCDn_CSR[START] field is cleared to 0 automatically when the channel begins execution, regardless of how the channel activates.

5.5.6.2 Reading the transfer descriptors of active channels

The eDMA reads back the true TCDn_SADDR, TCDn_DADDR, and TCDn_NBYTES values if they are read when a channel executes. The true values of SADDR, DADDR, and NBYTES are the values the eDMA engine currently uses in its internal register file, and not the values in the TCD local memory for that channel. The addresses, SADDR and DADDR, and NBYTES (which decrements to zero as the transfer progresses), can give an indication of the progress of the transfer. All other values are read back from the TCD local memory.

5.5.6.3 Checking channel preemption status

A preemptive situation is one in which a preempt-enabled channel is executing and a higher-priority request becomes active. When round-robin channel arbitration mode is enabled, all channels with their channel priority set to 0 lose their preempt ability. Channel priorities of 0 are treated as equal, that is, they are constantly rotating, when round-robin arbitration mode is enabled.

The CHn_CSR[ACTIVE] field for the preempted channel remains asserted throughout the preemption. The preempted channel is temporarily suspended when the preempting channel executes one major loop iteration. If two CHn_CSR[ACTIVE] fields are set simultaneously in the global TCD map, a higher-priority channel is actively preempting a lower-priority channel.

5.5.7 Channel linking

Channel linking (or chaining) is a mechanism in which one channel sets the TCDn_CSR[START] field of another channel (or itself), thus initiating a service request for that channel. When properly enabled, the eDMA engine automatically performs this operation at the major or minor loop completion.

The minor loop channel linking occurs at the completion of the minor loop (or one iteration of the major loop). The TCDn_CITER[ELINK] field determines whether a minor loop link is requested. When enabled, the channel link is made after each iteration of the major loop except for the last. When the major loop is exhausted, only the major loop channel link fields are used to determine if a channel link should be made. For example, using an initial field setting of:

```
TCDn_CITER[ELINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER] value = 0x4
TCDn_CSR[MAJORELINK] = 1
TCDn_CSR[MAJORLINKCH] = 0x7
```

executes as:

1. Minor loop done → set TCD12_CSR[START] field

2. Minor loop done → set TCD12_CSR[START] field
3. Minor loop done → set TCD12_CSR[START] field
4. Minor loop done, major loop done→ set TCD7_CSR[START] field

When minor loop linking is enabled (TCDn_CITER[ELINK] = 1), the TCDn_CITER[CITER] field uses a nine-bit vector to form the current iteration count. When minor loop linking is disabled (TCDn_CITER[ELINK] = 0), the TCDn_CITER[CITER] field uses a 15-bit vector to form the current iteration count. The bits associated with the TCDn_CITER[LINKCH] field are concatenated onto the CITER value to increase the range of the CITER.

NOTE

The TCDn_CITER[ELINK] field and the TCDn_BITER[ELINK] field must be equal — if they are not, a configuration error is reported. The CITER and BITER vector widths must be equal to calculate the major loop halfway done interrupt point.

The following table summarizes how a DMA channel can link to another DMA channel, that is, use another channel's TCD, at the end of a loop.

Table 24. Channel linking parameters

Wanted link behavior	TCD control field name	Description
Link at end of minor loop	TCDn_CITER[ELINK]	Enable channel-to-channel linking on minor loop completion (current iteration)
	TCDn_CITER[LINKCH]	Link channel number when linking at end of minor loop (current iteration)
Link at end of major loop	TCDn_CSR[MAJORELINK]	Enable channel-to-channel linking on major loop completion
	TCDn_CSR[MAJORLINKCH]	Link channel number when linking at end of major loop

5.5.8 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

5.5.8.1 Dynamically changing the channel priority

To change group or channel priority levels:

1. Halt the DMA by writing 1 to the CSR[HALT] field.
2. Change the group or channel priorities as wanted.
3. Enable normal DMA operations by writing 0 to the CSR[HALT] field.

5.5.8.2 Dynamic channel linking

Dynamic channel linking is the process of setting the TCDn_CSR[MAJORELINK] field during channel execution (see the diagram in TCD structure). This field is read from the TCD local memory at the end of channel execution, thus allowing you to enable the feature during channel execution.

Because you are allowed to change the configuration during execution, you need a coherency model. Consider the scenario where you attempt to execute a dynamic channel link by enabling the TCDn_CSR[MAJORELINK] field at the same time the eDMA engine is retiring the channel. TCDn_CSR[MAJORELINK] would be set in the programmer's model, but it would be unclear whether the actual link was made before the channel retired.

We recommend that you use the following coherency model when executing a dynamic channel link request.

1. Write 1 to the TCDn_CSR[MAJORELINK] field.

2. Read back the TCDn_CSR[MAJORELINK] field.
3. Test the TCDn_CSR[MAJORELINK] request status:
   - If TCDn_CSR[MAJORELINK] = 1, the dynamic link attempt was successful.
   - If TCDn_CSR[MAJORELINK] = 0, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces the TCDn_CSR[MAJORELINK] field to 0 on any writes to a channel's TCDn_CSR[7:0] after that channel's CHn_CSR[DONE] field is set to 1, indicating the major loop is complete.

NOTE

You must clear the CHn_CSR[DONE] field to 0 before writing to the TCDn_CSR[MAJORELINK] field. The CHn_CSR[DONE] field is cleared to 0 automatically by the eDMA engine after a channel begins execution.

5.5.8.3 Dynamic scatter/gather

Scatter/gather is the process of automatically loading a new TCD into a channel. It allows a DMA channel to use multiple TCDs; this enables a DMA channel to scatter the DMA data to multiple destinations or gather it from multiple sources. When scatter/gather is enabled and the channel has finished its major loop, a new TCD is fetched from system memory and loaded into that channel's descriptor location in the eDMA programmer's model, thus replacing the current descriptor.

Because you are allowed to change the configuration during execution, you need a coherency model. Consider the scenario where you attempt to execute a dynamic scatter/gather operation by enabling the TCDn_CSR[ESG] field at the same time the eDMA engine is retiring the channel. The TCDn_CSR[ESG] field would be set in the programmer's model, but it would be unclear whether the actual scatter/gather request was honored before the channel retired.

Two methods are recommended for executing a dynamic scatter/gather request. Whenever the TCDn_CSR is written, the TCD local memory controller forces the TCDn_CSR[ESG] field to 0 on any writes to a channel's TCDn_CSR[7:0] after that channel's CHn_CSR[DONE] field has been set to 1, indicating the major loop is complete. If attempting to set the ESG, ensure the DONE field is cleared to 0.

NOTE

You must clear the CHn_CSR[DONE] field to 0 before writing the TCDn_CSR[MAJORELINK] or TCDn_CSR[ESG] fields. The CHn_CSR[DONE] field is cleared to 0 automatically by the eDMA engine after a channel begins execution and is set to 1 upon major loop completion.

5.5.8.3.1 Method 1 (channel not using major loop channel linking)

For a channel not using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request.

When the TCDn_CSR[MAJORELINK] field is 0, the TCDn_CSR[MAJORLINKCH] field is not used by the eDMA. In this case, the TCDn_CSR[MAJORLINKCH] bits may be used for other purposes. This method uses the TCDn_CSR[MAJORLINKCH] field as a TCDn_CSR identification (ID).

When the descriptors are built, write a unique TCDn_CSR ID in the TCDn_CSR[MAJORLINKCH] field for each TCDn_CSR associated with a channel using dynamic scatter/gather.

1. Write a 1 to the TCDn_CSR[DREQ] field. Should a dynamic scatter/gather attempt fail, setting the TCDn_CSR[DREQ] field to 1 will prevent future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a DLAST final offset value.
2. Write the TCDn_DLAST_SGA field with the scatter/gather address.
3. Write a 1 to the TCDn_CSR[ESG] field.
4. Read back the 16-bit TCDn_CSR control/status field.
5. Test the TCDn_CSR[ESG] request status and TCDn_CSR[MAJORLINKCH] value:
   - If ESG = 1, the dynamic scatter/gather attempt was successful.

- If ESG = 0 and the MAJORLINKCH (ID) did not change, the dynamic scatter/gather attempt was not successful (the channel was already retiring).
- If ESG = 0 and the MAJORLINKCH (ID) changed, the dynamic scatter/gather attempt was successful (the new TCDn_CSR's ESG value cleared the ESG field to 0).

5.5.8.3.2 Method 2 (channel using major loop channel linking)

For a channel using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request. This method uses the TCDn_DLAST_SGA field as a TCD identification (ID).

1. Write a 1 to the TCDn_CSR[DREQ] field. Should a dynamic scatter/gather attempt fail, setting the DREQ field to 1 will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (DADDR) that was calculated using a scatter/gather address (written in the next step) instead of a DLAST final offset value.
2. Write the TCDn_DLAST_SGA field with the scatter/gather address.
3. Write a 1 to the TCDn_CSR[ESG] field.
4. Read back the TCDn_CSR[ESG] field.
5. Test the TCDn_CSR[ESG] request status:
   - If ESG = 1, the dynamic scatter/gather attempt was successful.
   - If ESG = 0, read the 32-bit TCDn_DLAST_SGA field.
   - If ESG = 0 and the TCDn_DLAST_SGA did not change, the dynamic scatter/gather attempt was not successful (the channel was already retiring).
   - If ESG = 0 and the TCDn_DLAST_SGA changed, the dynamic scatter/gather attempt was successful (the new TCDn_CSR's ESG value cleared the ESG field to 0).

5.5.9 Suspend/resume a DMA channel with active hardware service requests

The DMA allows you to move data from memory or peripheral registers to another location in memory or to peripheral registers without CPU interaction. After the DMA and peripherals are configured and active, it is rare but supported to suspend a peripheral's service request dynamically. In this scenario, there are certain restrictions to disabling a DMA hardware service request. For coherency, you must follow a specific procedure. This section provides guidance on how to coherently suspend and resume a Direct Memory Access (DMA) channel when the DMA is triggered by a slave module such as the Serial Peripheral Interface (SPI), Sigma Delta Analog to Digital Convertor (SDADC), or other module.

5.5.9.1 Suspend an active DMA channel

To suspend an active DMA channel:

1. Stop the DMA service request at the peripheral first. Confirm it has been disabled by reading back the appropriate register in the peripheral.
2. Check the DMA's Hardware Request Status (MP_HRS) to ensure there is no service request to the DMA channel being suspended. Then disable the hardware service request by clearing the ERQ field to 0 on the appropriate DMA channel.

For example, assume the SPI is set as a master for transmitting data via a DMA service request when the TXFIFO has an empty slot. The DMA will transfer the next command and data to the TXFIFO upon the request. If you need to suspend the DMA/SPI transfer loop, perform the following steps:

1. Disable the DMA service request at the source by writing 0 to DSPI_RSER[TFFF_RE]. Confirm that DSPI_RSER[TFFF_RE] is 0.
2. Ensure there is no DMA service request from the SPI by verifying that MP_HRS[HRS] is 0 for the appropriate channel. If no service request is present, disable the DMA channel by clearing the channel's ERQ field to 0. If a service request is present, wait until the request has been processed and the HRS field reads 0.

5.5.9.2 Resume a DMA channel

To resume a DMA channel:

1. Enable the DMA service request on the appropriate channel by setting its ERQ field to 1.
2. Enable the DMA service request at the peripheral.

5.6 Memory map/register definition

The eDMA programming model is partitioned into three parts:

1. The first part defines a number of registers providing overall control functions and is known as the management page.
2. The second part corresponds to the channel (CH) control, status, and configuration.
3. The third part corresponds to the local TCD memory.

TCD memory

Each channel requires a 32-byte transfer control descriptor for defining the data movement operation. Each TCDn definition is presented as 11 registers of 16 or 32 bits. See DMA TCD register descriptions for details.

TCD initialization

Prior to activating a channel, you must initialize its TCD with the appropriate transfer profile.

TCD structure

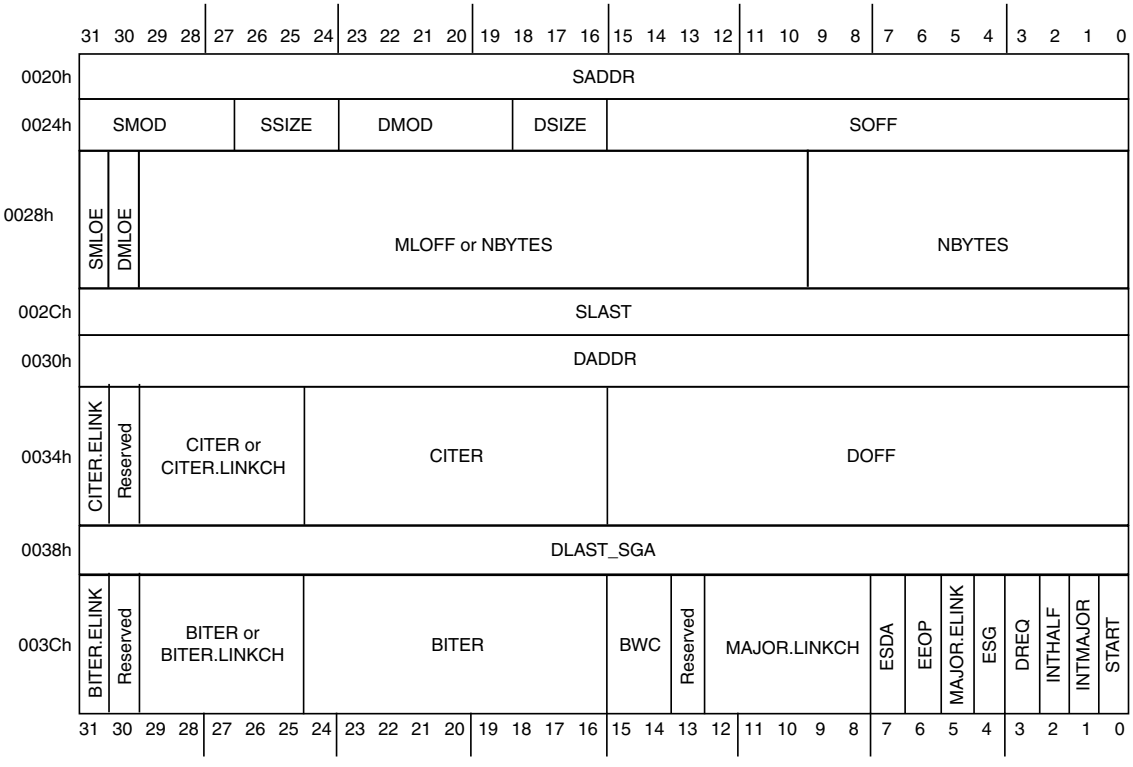


Figure 15. TCD structure

Accesses to reserved memory and fields

- Reading reserved fields in a register returns the value of zero.

- Writes to reserved fields in a register are ignored.
- Reading or writing a reserved memory location generates a bus error.

5.6.1 DMA MP register descriptions

5.6.1.1 MP memory map

DMA3.MP base address: 4400_0000h

NOTE

For registers in the following table with *Protection*, see the REG_PROT details for more information.

Offset	Register	Width (In bits)	Access	Reset value	Protection
0h	Management Page Control (MP_CSR)	32	RW	0031_0000h	Yes
4h	Management Page Error Status (MP_ES)	32	R	0000_0000h	No
8h	Management Page Interrupt Request Status (MP_INT)	32	R	0000_0000h	No
Ch	Management Page Hardware Request Status (MP_HRS)	32	R	0000_0000h	No
100h - 17Ch	Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI)	32	RW	0000_0000h	Yes

5.6.1.2 Management Page Control (MP_CSR)

Offset

Register	Offset
MP_CSR	0h

Function

The Management Page Control register defines the basic operating configuration of the DMA.

Arbitration uses a two-tier priority system; group and channel priority. The eDMA assigns each channel to a priority group. Group arbitration is fixed-priority and cannot be changed. Channel arbitration uses fixed priority and may be configured to use a selective round-robin scheme for specified channels within each priority group. For fixed-priority arbitration, eDMA selects for execution the highest priority channel requesting service in the highest priority arbitration group.

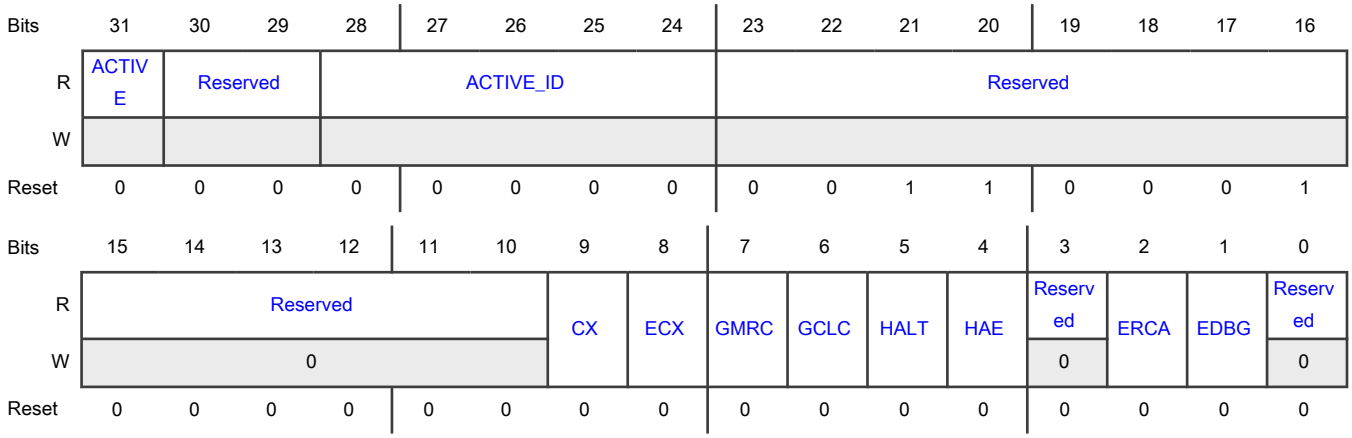
The channel priority registers assign the relative priorities within each arbitration group; see CHn_PRI. All channels with a non-zero CHn_PRI value use fixed-priority arbitration.

When you enable round-robin arbitration, all channels with channel priority set to zero do not have a priority and, of those channels requesting service, are cycled through (from high to low channel number) without regard to priority relative to each other within the same priority group. Any channel with a non-zero CHn_PRI value automatically has a higher priority over the round-robin channels. A channel's priority group is assigned in Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI).

NOTE

For correct operation, changes to the MP_CSR[ERCA, GCLC, GMRC] fields must be performed when the DMA channels are inactive; that is, when the MP_CSR[ACTIVE] field is 0.

Diagram



Fields

Field	Function
31 ACTIVE	DMA Active Status 0b - eDMA is idle 1b - eDMA is executing a channel
30-29 —	Reserved
28-24 ACTIVE_ID	Active Channel ID This field identifies the channel number that is executing when the ACTIVE bit is 1.
23-16 —	Reserved
15-10 —	Reserved
9 CX	Cancel Transfer When set to 1, this field cancels the remaining data transfer, stops the executing channel, and forces the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. CX clears itself to 0 after the cancel has been honored. This cancel retires the channel normally as if the minor loop had been completed. 0b - Normal operation 1b - Cancel the remaining data transfer
8 ECX	Cancel Transfer With Error Cancellation of the remaining data transfer is similar to that of the CX field. Execution of the channel is stopped and the minor loop is forced to finish. The cancellation takes effect after the last write of the current read/write sequence. The ECX field clears itself to 0 after the cancel is honored. In addition to

Table continues on the next page...

Table continued from the previous page...

Field	Function
	cancelling the transfer, ECX treats the cancel as an error condition, thus updating Management Page Error Status (MP_ES) and generating an optional error interrupt. 0b - Normal operation 1b - Cancel the remaining data transfer
7 GMRC	Global Master ID Replication Control **NOTE** If master ID replication is disabled, the nonsecure, privileged protection level for DMA transfers is used. 0b - Master ID replication disabled for all channels 1b - Master ID replication available and controlled by each channel's CHn_SBR[EMI] setting
6 GCLC	Global Channel Linking Control 0b - Channel linking disabled for all channels 1b - Channel linking available and controlled by each channel's link settings
5 HALT	Halt DMA Operations This field stalls the start of any new channels. Executing channels are allowed to complete. Channel execution resumes when this field is cleared to 0. 0b - Normal operation 1b - Stall the start of any new channels
4 HAE	Halt After Error When this field is set to 1, any error causes the HALT field to be set to 1. Then all service requests are ignored until the HALT field is cleared to 0. 0b - Normal operation 1b - Any error causes the HALT field to be set to 1
3 —	Reserved
2 ERCA	Enable Round Robin Channel Arbitration 0b - Round-robin channel arbitration disabled. Fixed priority arbitration used for channel selection within each group 1b - Round-robin channel arbitration enabled. Round-robin arbitration used for channel selection within each group
1 EDBG	Enable Debug When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. DMA resumes channel execution when the system exits debug mode or clears the EDBG field to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Debug mode disabled. When in debug mode, the DMA continues to operate 1b - Debug mode is enabled. When in debug mode, the DMA stalls the start of a new channel
0 —	Reserved

5.6.1.3 Management Page Error Status (MP_ES)

Offset

Register	Offset
MP_ES	4h

Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer control descriptor
- An error termination to a bus master read or write cycle
- An uncorrectable error that occurred when the device was accessing the TCD SRAM
- A "cancel transfer with error" request was made via the corresponding cancel transfer field or input signal

Upon any error condition, the software must initialize the TCD of the channel that contains the error, as it is in an incomplete state after an error. See Fault reporting and handling for more details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	Reserved			ERRCHN				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							ECX	SAE	SOE	DAE	DOE	NCE	SGE	SBE	DBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Valid

Table continues on the next page...

Table continued from the previous page...

Field	Function
VLD	Logical OR of all CHn_ES[ERR] status fields. 0b - No CHn_ES[ERR] fields are set to 1 1b - At least one CHn_ES[ERR] field is set to 1, indicating a valid error exists that software has not cleared
30-29 —	Reserved
28-24 ERRCHN	Error Channel Number or Canceled Channel Number The channel number of the last recorded error or last recorded error-canceled transfer.
23-9 —	Reserved
8 ECX	Transfer Canceled The ECX operation is a management page function. When employed, the targeted channel's CHn_ES register reports an unspecified error; that is, only the CHn_ES[ERR] field is set to 1. The management page has full view of the error condition. 0b - No canceled transfers 1b - Last recorded entry was a canceled transfer by the error cancel transfer input
7 SAE	Source Address Error When this field is 1, it indicates that TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SADDR field
6 SOE	Source Offset Error When this field is 1, it indicates that TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SOFF field
5 DAE	Destination Address Error When this field is 1, it indicates that TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DADDR field
4 DOE	Destination Offset Error When this field is 1, it indicates that TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DOFF field

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 NCE	NBYTES/CITER Configuration Error This error indicates that one of the following has occurred: • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE] • TCDn_CITER[CITER] is equal to zero • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK] 0b - No NBYTES/CITER configuration error 1b - The last recorded error was NBYTES equal to zero or a CITER not equal to BITER error. Last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields
2 SGE	Scatter/Gather Configuration Error When this field is 1, it indicates that TCDn_DLAST_SGA is not on a 32-byte boundary. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. 0b - No scatter/gather configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DLAST_SGA field
1 SBE	Source Bus Error 0b - No source bus error 1b - Last recorded error was a bus error on a source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - Last recorded error was a bus error on a destination write

5.6.1.4 Management Page Interrupt Request Status (MP_INT)

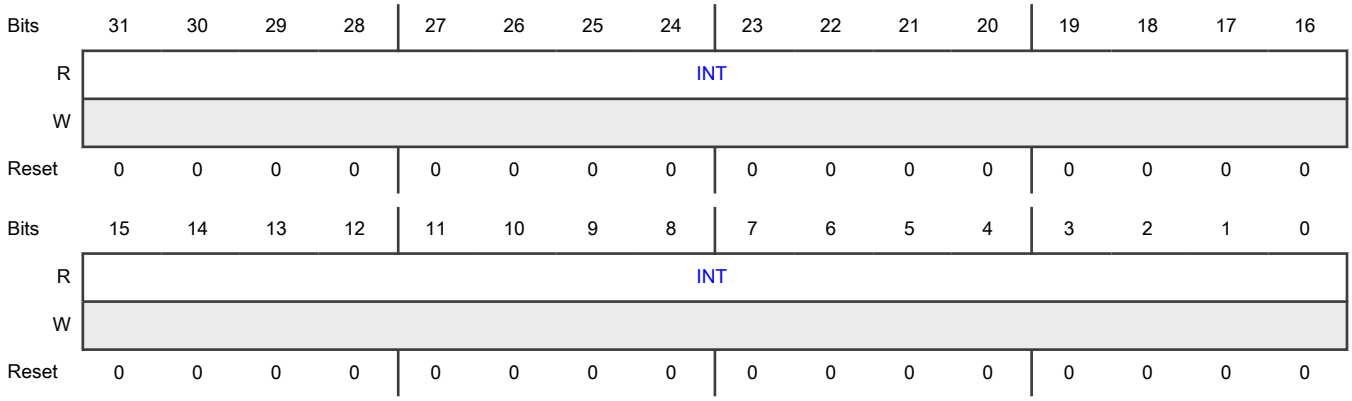
Offset

Register	Offset
MP_INT	8h

Function

This register shows the current state of the interrupt service requests for all eDMA channels.

Diagram



Fields

Field	Function
31-0 INT	Interrupt Request Status The INT register presents the interrupt request status for each eDMA channel. Depending on the appropriate field setting in the transfer control descriptors, the eDMA engine generates an interrupt on data transfer completion. The eDMA routes channel interrupt requests to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate field in the channel's interrupt request register, CHn_INT, thus negating the interrupt request. 0b - Interrupt request for corresponding channel not present 1b - Interrupt request for corresponding channel present

5.6.1.5 Management Page Hardware Request Status (MP_HRS)

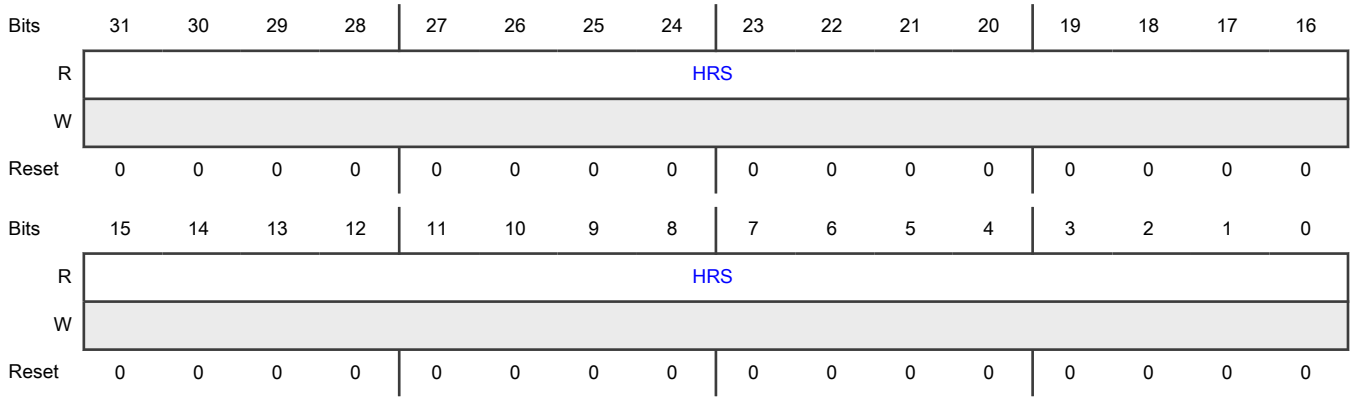
Offset

Register	Offset
MP_HRS	Ch

Function

The hardware request status register (HRS) shows the current state of the hardware service request signaling as seen by eDMA's arbitration logic.

Diagram



Fields

Field	Function
31-0 HRS	Hardware Request Status The HRS bit for its respective channel remains asserted for the period when a hardware request is present on the channel. 0b - Hardware service request for corresponding channel is not present 1b - Hardware service request for corresponding channel is present

5.6.1.6 Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI)

Offset

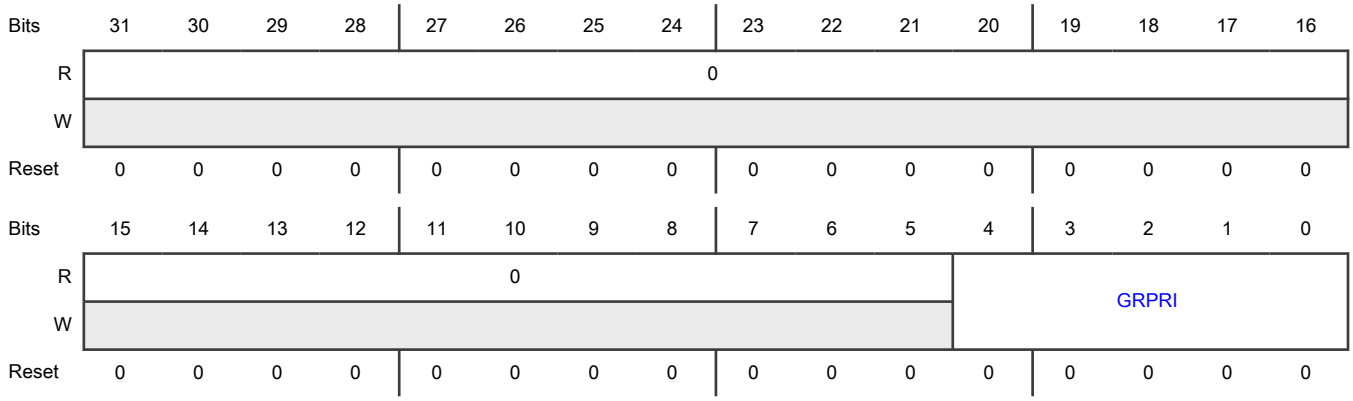
For n = 0 to 31:

Register	Offset
CHn_GRPRI	100h + (n × 4h)

Function

The contents of this register define the arbitration group associated with each channel. Using a fixed-priority group arbitration scheme, eDMA evaluates the arbitration group priorities by numeric value from highest group number to lowest; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, and so on. The range of the group priority values is limited to the values of 0 through 31. Within each arbitration group, the channel priority assignment CH*n*_PRI determines the highest-priority channel.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 GRPRI	Arbitration Group For Channel n Fixed-priority arbitration group number.

5.6.2 DMA TCD register descriptions

5.6.2.1 TCD memory map

DMA3.TCD base address: 4401_0000h

NOTE

For registers in the following table with *Protection*, see the REG_PROT details for more information.

Offset	Register	Width (In bits)	Access	Reset value	Protection
0h - 1F_0000h	Channel Control and Status (CH0_CSR - CH31_CSR)	32	RW	0000_0000h	Yes
4h - 1F_0004h	Channel Error Status (CH0_ES - CH31_ES)	32	RW	0000_0000h	No
8h - 1F_0008h	Channel Interrupt Status (CH0_INT - CH31_INT)	32	RW	0000_0000h	No
Ch - 1F_000Ch	Channel System Bus (CH0_SBR - CH31_SBR)	32	RW	0000_8007h	Yes
10h - 1F_0010h	Channel Priority (CH0_PRI - CH31_PRI)	32	RW	0000_0000h	Yes

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value	Protection
14h - 1F_0014h	Channel Multiplexor Configuration (CH0_MUX - CH31_MUX)	32	RW	0000_0000h	No
20h - 1F_0020h	TCD Source Address (TCD0_SADDR - TCD31_SADDR)	32	RW	See section	Yes
24h - 1F_0024h	TCD Signed Source Address Offset (TCD0_SOFF - TCD31_SOFF)	16	RW	See section	Yes
26h - 1F_0026h	TCD Transfer Attributes (TCD0_ATTR - TCD31_ATTR)	16	RW	See section	Yes
28h - 1F_0028h	TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO - TCD31_NBYTES_MLOFFNO)	32	RW	See section	Yes
28h - 1F_0028h	TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES - TCD31_NBYTES_MLOFFYES)	32	RW	See section	No
2Ch - 1F_002Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA - TCD31_SLAST_SDA)	32	RW	See section	Yes
30h - 1F_0030h	TCD Destination Address (TCD0_DADDR - TCD31_DADDR)	32	RW	See section	Yes
34h - 1F_0034h	TCD Signed Destination Address Offset (TCD0_DOFF - TCD31_DOFF)	16	RW	See section	Yes
36h - 1F_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD31_CITER_ELINKNO)	16	RW	See section	Yes
36h - 1F_0036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD31_CITER_ELINKYES)	16	RW	See section	No
38h - 1F_0038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA - TCD31_DLAST_SGA)	32	RW	See section	Yes
3Ch - 1F_003Ch	TCD Control and Status (TCD0_CSR - TCD31_CSR)	16	RW	See section	Yes
3Eh - 1F_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD31_BITER_ELINKNO)	16	RW	See section	Yes
3Eh - 1F_003Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD31_BITER_ELINKYES)	16	RW	See section	No

5.6.2.2 Channel Control and Status (CH0_CSR - CH31_CSR)

Offset

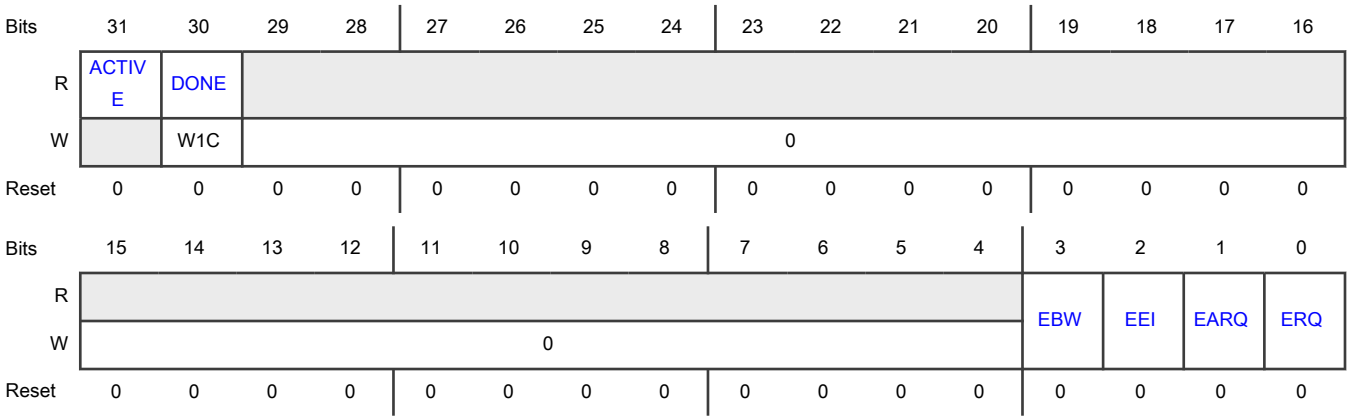
For n = 0 to 31:

Register	Offset
CHn_CSR	0h + (n × 1_0000h)

Function

This register contains several fields related to hardware and interrupt requests, configuration, and status for the given channel.

Diagram



Fields

Field	Function
31 ACTIVE	Channel Active The ACTIVE field indicates the channel was selected by arbitration and is executing the prescribed transfers. The eDMA sets it to 1 when channel service begins, and clears it to 0 as the minor loop completes or when any error condition is detected. Except for dynamic scatter/gather or dynamic channel linking, you must not modify the transfer control descriptor when a channel is active.
30 DONE	Channel Done The DONE field indicates the eDMA has completed the major loop. The eDMA engine sets this field as the CITER count reaches zero. If enabled, the eDMA generates an interrupt request corresponding to this completed channel. The software clears it, or the hardware clears it when the channel is activated. **NOTE** This field must be cleared to 0 before writing the MAJORELINK or ESG fields.
29-4 —	Reserved
3	Enable Buffered Writes

Table continues on the next page...

Table continued from the previous page...

Field	Function
EBW	When buffered writes are enabled, all writes except for the last write sequence of the minor loop are signaled by the eDMA as bufferable. 0b - Buffered writes on system bus disabled. Buffered writes on system bus disabled 1b - Buffered writes on system bus enabled. Bufferable write signal asserted on all system bus writes except during last write sequence
2 EEI	Enable Error Interrupt The EEI field enables the error interrupt signal for the channel. The DMA error indicator and the error interrupt enable flag must be asserted before an error interrupt request for a given channel is asserted to the interrupt controller. 0b - Error signal for corresponding channel does not generate error interrupt 1b - Assertion of error signal for corresponding channel generates error interrupt request
1 EARQ	Enable Asynchronous DMA Request The enable asynchronous DMA request field (EARQ) does not affect DMA operations. When set to 1, this field allows the hardware service request enable field (ERQ) to propagate out of the DMA to the power controller. When cleared to 0, this field masks the hardware service request enable field to the power controller. 0b - Disable asynchronous DMA request for the channel 1b - Enable asynchronous DMA request for the channel
0 ERQ	Enable DMA Request Disable a channel's hardware service request at the source before clearing the channel's ERQ field. The DMA hardware request input signal and the enable request field (ERQ) must be asserted before a channel's hardware service request is accepted. The state of the eDMA enable request field does not affect a channel service request made explicitly through software or channel linking. The state of the ERQ field does not affect the channel's START field. 0b - DMA hardware request signal for corresponding channel disabled 1b - DMA hardware request signal for corresponding channel enabled

5.6.2.3 Channel Error Status (CH0_ES - CH31_ES)

Offset

For n = 0 to 31:

Register	Offset
CHn_ES	4h + (n × 1_0000h)

Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer control descriptor

- An error termination to a bus master read or write cycle

The ERR field signals the presence of an error for the channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate field in this register. The outputs of this register are enabled by the contents of the CHn_CSR[EEI] field, then logically summed across all channels to form an error interrupt request, which may be routed to the interrupt controller.

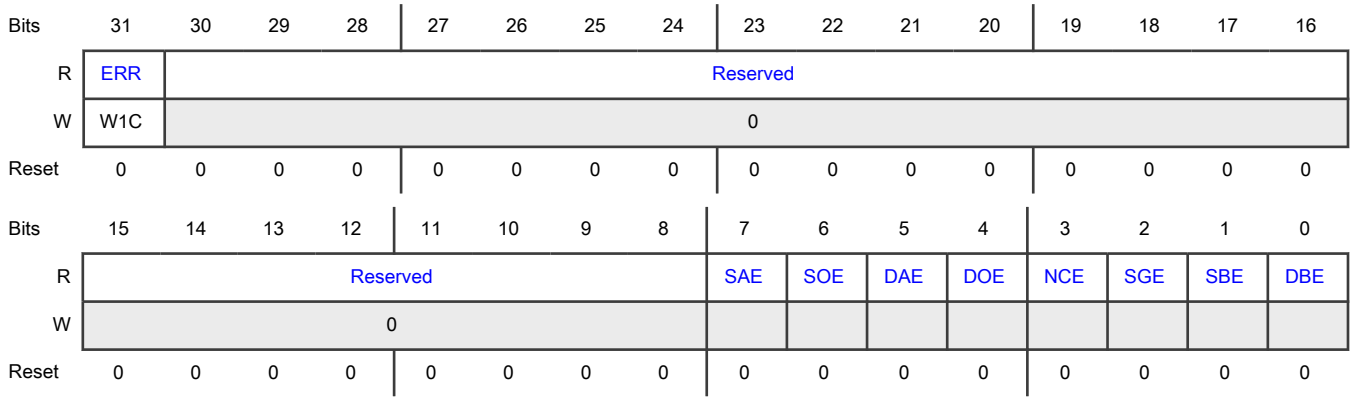
During the execution of the interrupt service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request) are not affected when eDMA detects an error. The contents of this ERR register field can also be polled because a non-zero value indicates the presence of a channel error, regardless of the state of the EEI mask.

The state of any given channel's error indicators is affected by writes to this register. Writing a 1 to the ERR field clears the channel's error status, and writing a 0 has no effect.

An unspecified error, where only the ERR field is set to 1, indicates that either a transfer was cancelled with an error. The Management Page Error Status register has full view of the error condition.

See Fault reporting and handling for more details.

Diagram



Fields

Field	Function
31 ERR	Error In Channel 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
30-8 —	Reserved
7 SAE	Source Address Error TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SADDR field
6	Source Offset Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
SOE	TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SOFF field
5 DAE	Destination Address Error TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DADDR field
4 DOE	Destination Offset Error TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DOFF field
3 NCE	NBYTES/CITER Configuration Error This error indicates that one of the following has occurred: • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE] • TCDn_CITER[CITER] is equal to zero • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK] 0b - No NBYTES/CITER configuration error 1b - Last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields
2 SGE	Scatter/Gather Configuration Error When this field is 1, it indicates that TCDn_DLAST_SGA is not on a 32-byte boundary. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG]is enabled. 0b - No scatter/gather configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DLAST_SGA field
1 SBE	Source Bus Error 0b - No source bus error 1b - Last recorded error was bus error on source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - Last recorded error was bus error on destination write

5.6.2.4 Channel Interrupt Status (CH0_INT - CH31_INT)

Offset

For n = 0 to 31:

Register	Offset
CHn_INT	8h + (n × 1_0000h)

Function

The INT field signals the presence of an interrupt request for the channel. Depending on the appropriate bit setting in the transfer control descriptors, the eDMA engine generates an interrupt on data transfer completion.

The outputs of this register are directly routed to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit, negating the interrupt request. On writes to INT, a 1 clears the channel's interrupt request. A zero has no effect on the channel's current interrupt status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															INT
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 INT	Interrupt Request 0b - Interrupt request for corresponding channel cleared 1b - Interrupt request for corresponding channel active

5.6.2.5 Channel System Bus (CH0_SBR - CH31_SBR)

Offset

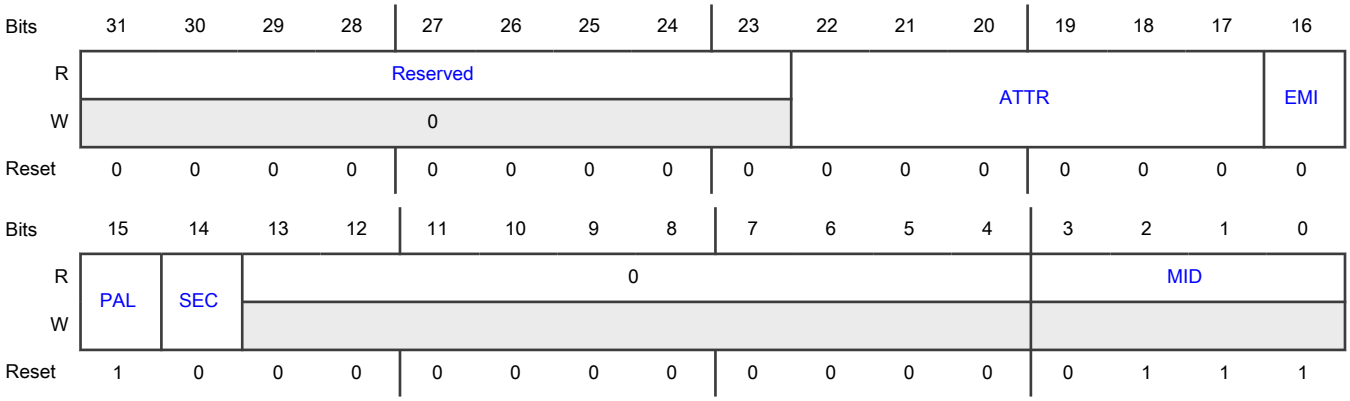
For n = 0 to 31:

Register	Offset
CHn_SBR	Ch + (n × 1_0000h)

Function

The Channel System Bus register places identification and attribute information on the system bus interface for the eDMA.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-17 ATTR	Attribute Output DMA's system bus attribute output value.
16 EMI	Enable Master ID Replication The eDMA master ID replication field allows the eDMA to use the same protection level and system bus ID of the master programming the eDMA's TCD. When enabled, the eDMA uses the master ID and protection level stored in the CHn_SBR registers, instead of the eDMA's default values. When a master (for example a core) programs a TCD, its master ID is captured when the TCD*n*_CSR control attributes are written. A scatter/gather operation does not affect the CHn_SBR registers. You can write the EMI only if MP_CSR[GMRC] = 1, which means Global Master ID Replication Control is enabled; otherwise, the EMI is forced to zero. **NOTE** If master ID replication is disabled, the nonsecure, privileged protection level for DMA transfers is used. 0b - Master ID replication is disabled 1b - Master ID replication is enabled
15	Privileged Access Level

Table continues on the next page...

Table continued from the previous page...

Field	Function
PAL	This field controls DMA's protection level on the system bus when the channel is active. **NOTE** The value written into this register cannot exceed the security and privilege level of the core or other master writing the channel's system bus register; CH*n*_SBR. The order of precedence is SecurePriv>SecureUser>NonsecurePriv>NonsecureUser 0b - User protection level for DMA transfers 1b - Privileged protection level for DMA transfers
14 SEC	Security Level DMA's security level on the system bus when the channel is active. **NOTE** The value written into this register cannot exceed the security and privilege level of the core or other master writing the channel's system bus register; CH*n*_SBR. The order of precedence is SecurePriv>SecureUser>NonsecurePriv>NonsecureUser 0b - Nonsecure protection level for DMA transfers 1b - Secure protection level for DMA transfers
13-4 —	Reserved
3-0 MID	Master ID This field controls the DMA's master ID on the system bus when the channel is active. **NOTE** The ID captured in this register reflects the master ID of the core or other master writing the channel's security attributes, TCD*n*_SBR[SEC].

5.6.2.6 Channel Priority (CH0_PRI - CH31_PRI)

Offset

For n = 0 to 31:

Register	Offset
CHn_PRI	10h + (n × 1_0000h)

Function

The contents of these registers define unique priorities associated with each channel within the same channel group. Channel grouping is programmed via Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI).

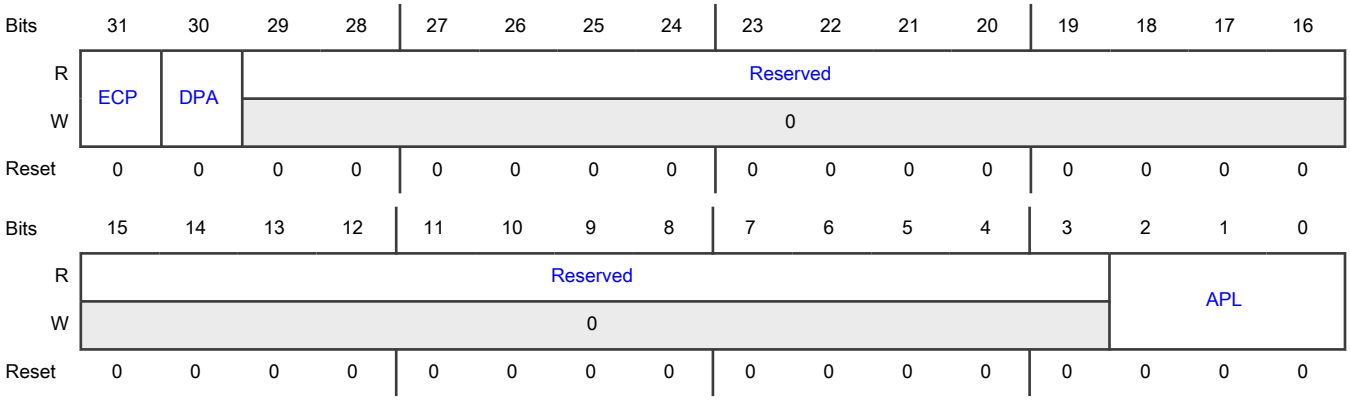
The channel priorities within a group are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, and so on. Software must program the channel priorities with unique values; otherwise, channel numbers with the same, non-zero value, will be selected based on channel number with the higher channel number having higher priority.

If more than one channel in a group has an arbitration priority level value of zero, then the arbitration mode field MP_CSR[ERCA] is used to determine the arbitration scheme for all channels with APL=0 within a group.

When you enable round-robin channel arbitration (MP_CSR[ERCA] = 1), all channels with APL=0 within a group will use a round-robin arbitration scheme, which rotates among these channels requesting service without regard to priority. Round-robin provides a fairness mechanism within an arbitration group.

When you enable fixed-priority channel arbitration (MP_CSR[ERCA] = 0), eDMA selects channels with APL=0 based on channel number, with the higher channel number having higher priority.

Diagram



Fields

Field	Function
31 ECP	Enable Channel Preemption 0b - Channel cannot be suspended by a higher-priority channel's service request 1b - Channel can be temporarily suspended by a higher-priority channel's service request
30 DPA	Disable Preempt Ability 0b - Channel can suspend a lower-priority channel 1b - Channel cannot suspend any other channel, regardless of channel priority
29-3 —	Reserved
2-0 APL	Arbitration Priority Level Channel priority level for arbitration within the assigned arbitration group.

5.6.2.7 Channel Multiplexor Configuration (CH0_MUX - CH31_MUX)

Offset

For n = 0 to 31:

Register	Offset
CHn_MUX	14h + (n × 1_0000h)

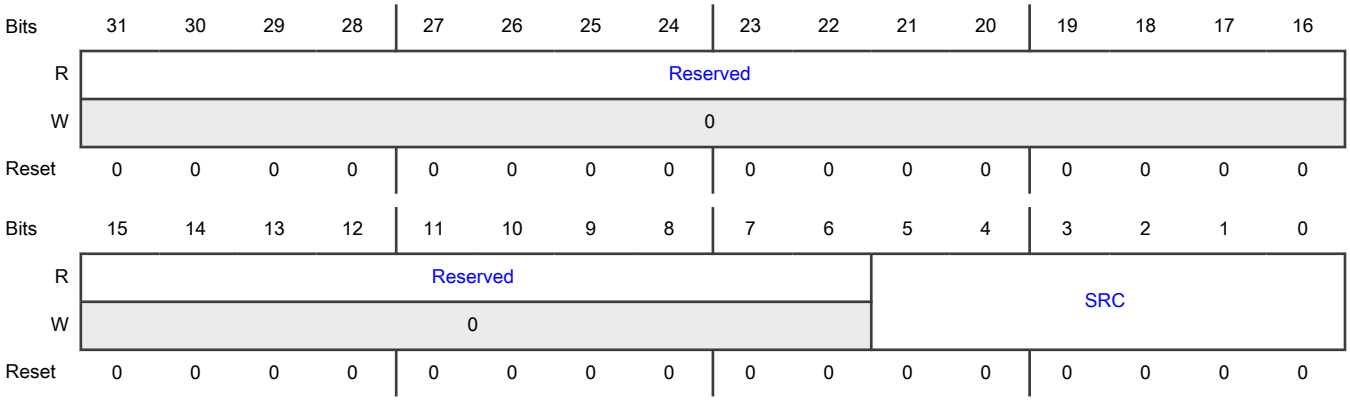
Function

Each of the DMA channels can be independently associated with various peripherals in the system. The Channel Multiplexor Configuration register selects the peripheral assigned to each channel. Service requests from the peripheral should be disabled when configuring a channel to a peripheral source.

Each channel must have a unique value when selecting a peripheral slot in the channel mux configuration. The only value that may overlap is source 0. If there is an attempt to write a mux configuration value that is already consumed by any channel, a mux configuration of 0 (SRC = 0) will be written.

All channels will default to source 0. When a particular peripheral is needed, the channel's mux configuration is set to that source number. When the peripheral is no longer needed, the mux configuration for that channel should be written to 0, thus releasing the resource.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 SRC	Service Request Source Hardware service request source for the channel. **NOTE** With the exception of 0, attempts to write a value already in use will be forced to 0.

5.6.2.8 TCD Source Address (TCD0_SADDR - TCD31_SADDR)

Offset

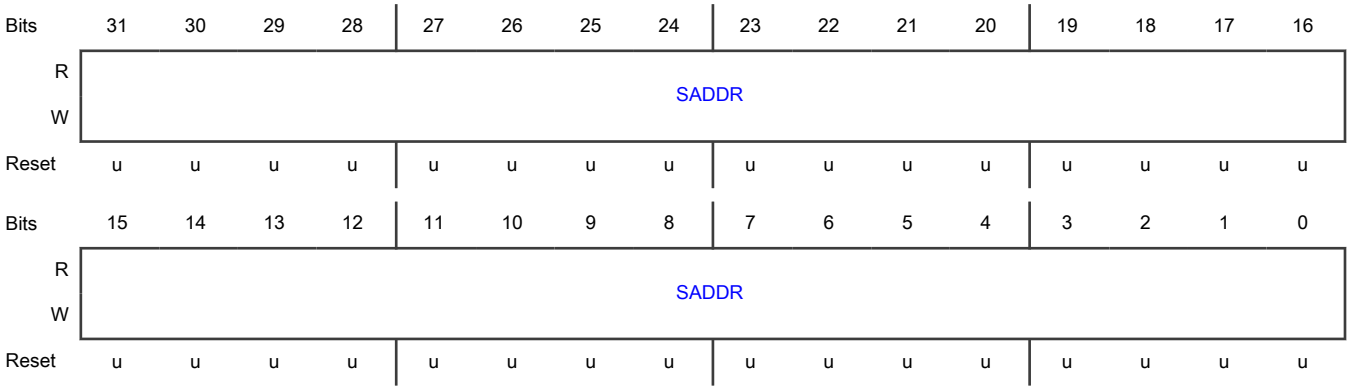
For n = 0 to 31:

Register	Offset
TCDn_SADDR	20h + (n × 1_0000h)

Function

This register contains the address for the read transactions.

Diagram



Fields

Field	Function
31-0 SADDR	Source Address Memory address pointing to the source data.

5.6.2.9 TCD Signed Source Address Offset (TCD0_SOFF - TCD31_SOFF)

Offset

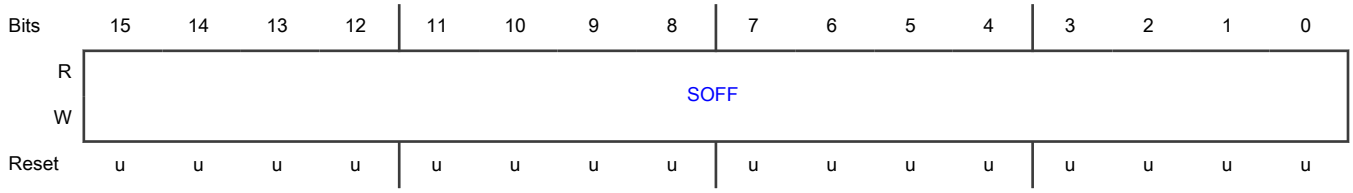
For n = 0 to 31:

Register	Offset
TCDn_SOFF	24h + (n × 1_0000h)

Function

This register contains the sign-extended value added to Source Address register after each read transaction.

Diagram



Fields

Field	Function
15-0 SOFF	Source Address Signed Offset Sign-extended offset applied to the current source address to form the next-state value as each source read is completed.

5.6.2.10 TCD Transfer Attributes (TCD0_ATTR - TCD31_ATTR)

Offset

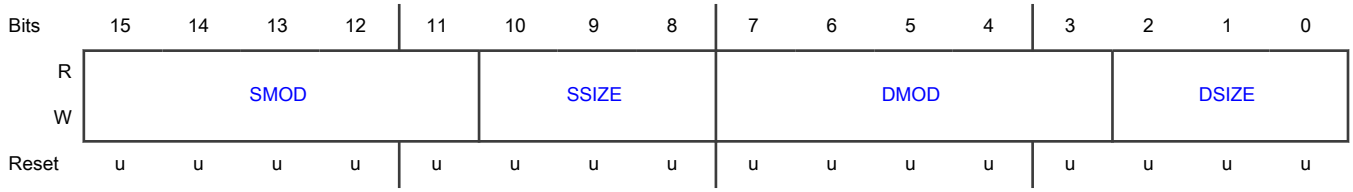
For n = 0 to 31:

Register	Offset
TCDn_ATTR	26h + (n × 1_0000h)

Function

This register contains size and option modulo addressing information for source and destination addresses.

Diagram



Fields

Field	Function
15-11 SMOD	Source Address Modulo This field defines a specific address range, which is the value after the SADDR + SOFF calculation is performed on the original register value. Setting this field makes it easy to implement a circular data queue. For data queues requiring power-of-2-sized bytes, the queue must start at a 0-modulo-size address and the SMOD field must be set to the appropriate value for the queue, freezing the required number of upper address bits.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The value programmed into this field specifies the number of lower address bits that are allowed to change. For a circular queue application, you typically set TCDn_SOFF[SOFF] to the transfer size to implement post-increment addressing, with the SMOD function constraining the addresses to a 0-modulo-size range. 0_0000b - Source address modulo feature disabled 0_0001b - Source address modulo feature enabled for any non-zero value [1-31]
10-8 SSIZE	Source Data Transfer Size 000b - 8-bit 001b - 16-bit 010b - 32-bit 011b - 64-bit 100b - 16-byte 101b - 32-byte 110b - 64-byte 111b - Reserved
7-3 DMOD	Destination Address Modulo See the SMOD definition.
2-0 DSIZE	Destination Data Transfer Size See the SSIZE definition.

5.6.2.11 TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO - TCD31_NBYTES_MLOFFNO)

Offset

For n = 0 to 31:

Register	Offset
TCDn_NBYTES_MLOFFNO	28h + (n × 1_0000h)

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offsets are address offset values added to the final source address (TCDn_SADDR), or destination address (TCDn_DADDR), upon minor loop completion. Minor loop completion is when the channel has finished the service request and has transferred NBYTES. When minor loop offsets are enabled, the minor loop offset value (TCDn_NBYTES_MLOFFYES[MLOFF]) is added to the final source address (TCDn_SADDR), to the final destination address (TCDn_DADDR), or to both, prior to the addresses being written back to the TCD. If the major loop is complete, the minor loop offset is ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When minor loop mapping is enabled (SMLOE or DMLOE is 1), TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is redefined. A portion of TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is used to specify multiple fields:

- A source enable bit (SMLOE) to specify the minor loop offset must be applied to the source address (TCDn_SADDR) upon minor loop completion
- A destination enable bit (DMLOE) to specify the minor loop offset must be applied to the destination address (TCDn_DADDR) upon minor loop completion
- The sign extended minor loop offset value (MLOFF)

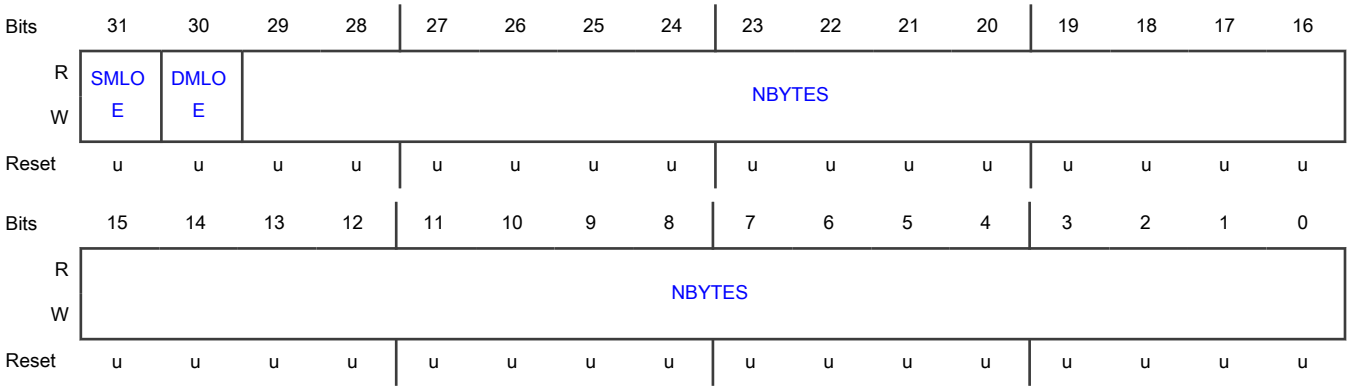
The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. If both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCDn_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is defined as follows:

- If SMLOE = 0 and DMLOE = 0, then see the TCDn_NBYTES_MLOFFNO register description.
- If either SMLOE or DMLOE is 1, then see the TCDn_NBYTES_MLOFFYES register description.

Diagram



Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - Minor loop offset not applied to SADDR 1b - Minor loop offset applied to SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - Minor loop offset not applied to DADDR 1b - Minor loop offset applied to DADDR
29-0	Number of Bytes To Transfer Per Service Request

Table continues on the next page...

Table continued from the previous page...

Field	Function
NBYTES	Number of bytes to be transferred for each service request of the channel. When a channel activates, the module loads the appropriate TCD contents into the eDMA engine and performs the appropriate reads and writes until the byte transfer count has been reached. This process is normally an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the byte count is exhausted, the SADDR and DADDR values are written back into the TCD memory, and the major loop iteration count (CITER) is decremented by one and written back to the TCD memory. If the major iteration count is complete, additional processing is performed.

5.6.2.12 TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES - TCD31_NBYTES_MLOFFYES)

Offset

For n = 0 to 31:

Register	Offset
TCDn_NBYTES_MLOFFYES	28h + (n × 1_0000h)

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offset is an address offset value added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) upon minor loop completion. Minor loop completion occurs when the channel has finished the service request and has transferred NBYTES. Minor loop offsets are enabled by setting either the source enable bit (SMLOE) or the destination enable bit (DMLOE).

The source enable bit (SMLOE) specifies the minor loop offset value (MLOFF) that is to be applied to the source address (TCDn_SADDR) upon minor loop completion. The destination enable bit (DMLOE) specifies the minor loop offset (MLOFF) that is to be applied to the destination address (TCDn_DADDR) upon minor loop completion.

If the major loop is complete, the minor loop offsets are ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

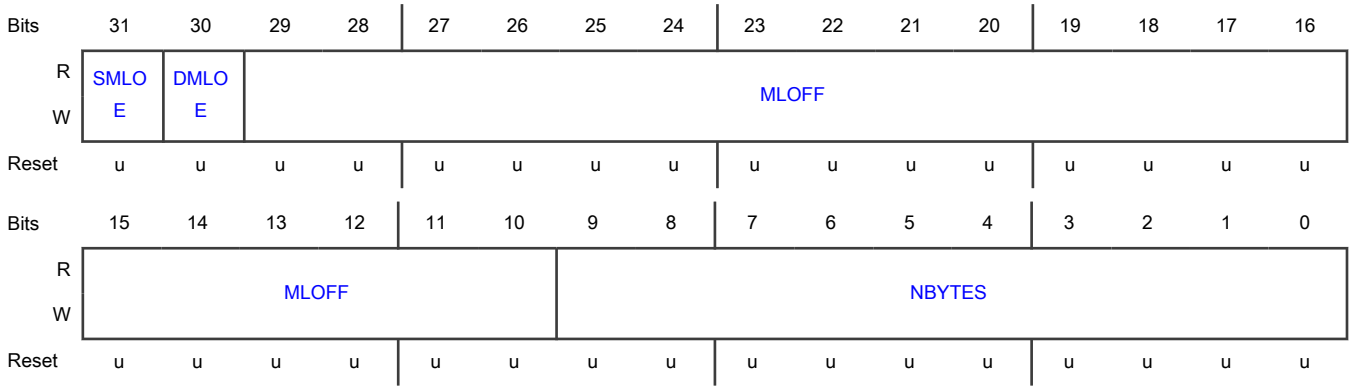
When you enable the minor loop offset overlay (either SMLOE or DMLOE is 1), eDMA redefines TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES. A portion of TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES specifies the sign-extended minor loop offset value (MLOFF). The same offset value (MLOFF) applies to both source and destination minor loop offsets. When the minor loop offset is enabled, you must align it to the transfer size of the source or destination it is associated with. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. If both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCDn_NBYTES_MLOFFNO) defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCDn_NBYTES_MLOFFYES is defined as follows:

- If either minor loop offset is enabled (SMLOE or DMLOE = 1), then see the TCDn_NBYTES_MLOFFYES register description.
- If SMLOE and DMLOE are both 0, then see the TCDn_NBYTES_MLOFFNO register description.

Diagram



Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - Minor loop offset not applied to SADDR 1b - Minor loop offset applied to SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - Minor loop offset not applied to DADDR 1b - Minor loop offset applied to DADDR
29-10 MLOFF	Minor Loop Offset If SMLOE or DMLOE is 1, this field represents a sign-extended offset applied to the source or destination address to form the next-state value after the minor loop completes.
9-0 NBYTES	Number of Bytes To Transfer Per Service Request The number of bytes to be transferred in each service request of the channel. As a channel activates, the module loads the appropriate TCD contents into the eDMA engine and performs the appropriate reads and writes until the minor byte transfer count has been reached. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, and the major iteration count is decremented and restored to the TCD memory. If the major iteration count is complete, additional processing is performed.

5.6.2.13 TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA - TCD31_SLAST_SDA)

Offset

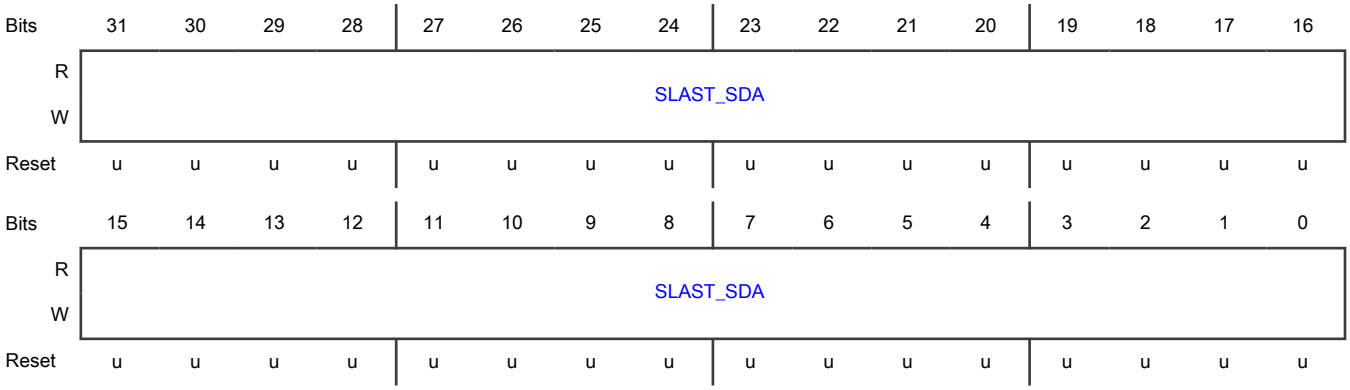
For n = 0 to 31:

Register	Offset
TCDn_SLAST_SDA	2Ch + (n × 1_0000h)

Function

This register contains the value added to the source address when the major loop is complete. When the store destination address option is enabled, this field provides a pointer to memory for storing the final destination address.

Diagram



Fields

Field	Function
31-0 SLAST_SDA	Last Source Address Adjustment / Store DADDR Address Source last address adjustment or the system memory address for destination address (DADDR) storage. If (TCDn_CSR[ESDA] = 0), then: • Adjustment value is added to the source address at the completion of the major iteration count. This value can be used to restore the source address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final source address adjustment. Otherwise: • This address points to the 32-bit-aligned memory location where the destination address (DADDR) is to be stored in system memory. By saving the final destination address in system memory via the ESDA feature, you are able to compute the size of a variable destination data buffer by simply subtracting the beginning DADDR from the final, saved DADDR. This feature is used together with the scatter/gather operation to prevent the loss of the final DADDR, which is overwritten during the scatter/gather operation. The "Store Destination Address" (SDA) value must be a 32-bit-aligned location because the eDMA forces the lower two address bits of the SLAST_SDA field to zero when ESDA is enabled. The module performs this write operation when the major loop is done; that is, when the major iteration count (CITER) decrements to zero.

5.6.2.14 TCD Destination Address (TCD0_DADDR - TCD31_DADDR)

Offset

For n = 0 to 31:

Register	Offset
TCDn_DADDR	30h + (n × 1_0000h)

Function

This register contains the address for the write transactions.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 DADDR	Destination Address Memory address pointing to the destination data.

5.6.2.15 TCD Signed Destination Address Offset (TCD0_DOFF - TCD31_DOFF)

Offset

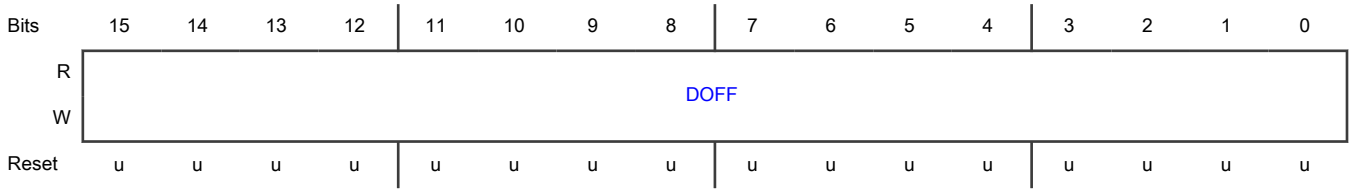
For n = 0 to 31:

Register	Offset
TCDn_DOFF	34h + (n × 1_0000h)

Function

This register contains the sign-extended value added to Destination Address register after each write transaction.

Diagram



Fields

Field	Function
15-0 DOFF	Destination Address Signed Offset Sign-extended offset that is applied to the current destination address to form the next-state value as each destination write is completed.

5.6.2.16 TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD31_CITER_ELINKNO)

Offset

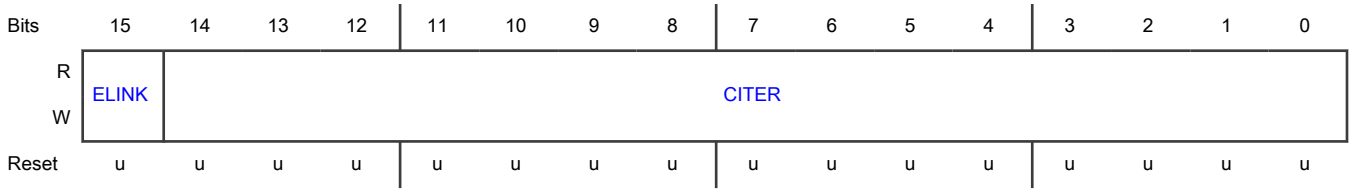
For n = 0 to 31:

Register	Offset
TCDn_CITER_ELINKNO	36h + (n × 1_0000h)

Function

If TCDn_CITER[ELINK] is 0, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by the relevant LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel to 1. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of MAJORELINK channel linking.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field must be equal to the BITER[ELINK] field; otherwise, a configuration error is reported. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and is written back to TCD memory. After the major iteration count is exhausted, the channel performs a number of operations — for example, final source and destination address calculations — and optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

5.6.2.17 TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD31_CITER_ELINKYES)

Offset

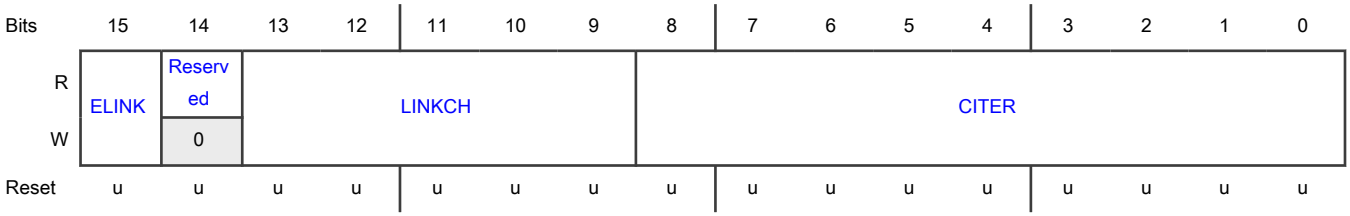
For n = 0 to 31:

Register	Offset
TCDn_CITER_ELINKYES	36h + (n × 1_0000h)

Function

If TCDn_CITER[ELINK] is 1, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by the relevant LINKCH field. When enabled, an internal mechanism sets the TCDn_CSR[START] field of the specified channel (LINKCH) upon minor loop completion. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of MAJORELINK channel linking. **NOTE** This field must be equal to the BITER[ELINK] field; otherwise, a configuration error is reported. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14 —	Reserved
13-9 LINKCH	Minor Loop Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted the eDMA engine initiates a channel service request to the channel defined by this field by writing that channel's TCDn_CSR[START] field to 1.
8-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and is written back to the TCD memory. After the major iteration count is exhausted, the channel performs a number of operations — for example, final source and destination address calculations — and optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

5.6.2.18 TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA - TCD31_DLAST_SGA)

Offset

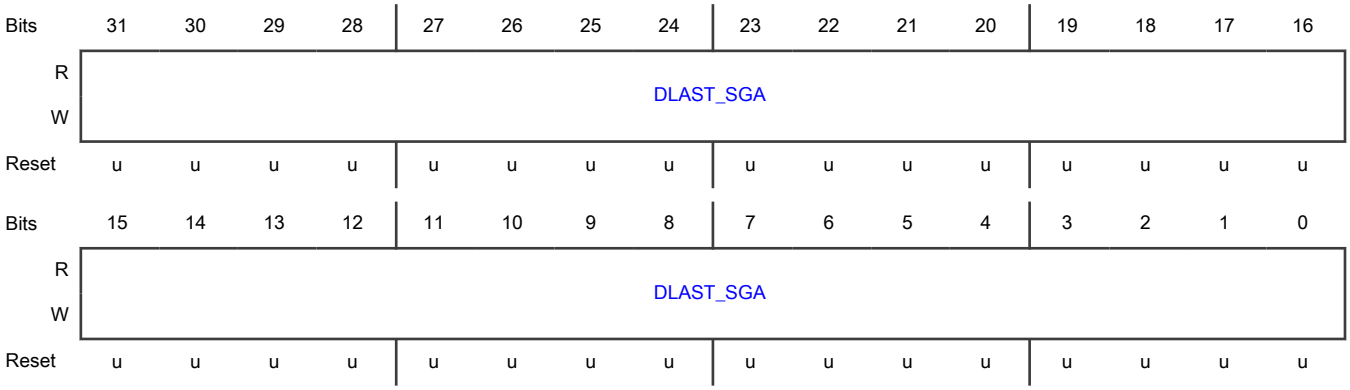
For n = 0 to 31:

Register	Offset
TCDn_DLAST_SGA	38h + (n × 1_0000h)

Function

This register contains the value added to the destination address when the major loop is complete. When the Scatter/Gather option is enabled, this field provides a pointer to memory for fetching a transfer control descriptor to reprogram the channel.

Diagram



Fields

Field	Function
31-0 DLAST_SGA	Last Destination Address Adjustment / Scatter Gather Address Adjustment of the last destination address or the memory address for the next transfer control descriptor to be loaded into this channel (scatter/gather). If (TCDn_CSR[ESG] = 0) then: • Adjustment value is added to the destination address at the completion of the major iteration count. This value can apply to restore the destination address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final destination address adjustment. Otherwise: • This address points to the beginning of a 0-modulo 32-byte region containing the next transfer control descriptor to be loaded into this channel. This channel reload is performed as the major iteration count completes. The scatter/gather address must be 0-modulo 32-byte, or else a configuration error is reported.

5.6.2.19 TCD Control and Status (TCD0_CSR - TCD31_CSR)

Offset

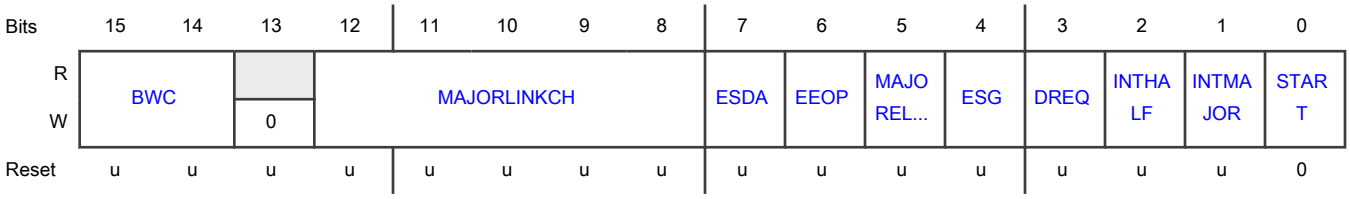
For n = 0 to 31:

Register	Offset
TCDn_CSR	3Ch + (n × 1_0000h)

Function

This register is used to enable optional features.

Diagram



Fields

Field	Function
15-14 BWC	Bandwidth Control Throttles the amount of bus bandwidth consumed by the eDMA. Generally, as the eDMA processes the minor loop, it continuously generates read/write sequences until the minor count is exhausted. This field forces eDMA to stall after the completion of each read/write access, to control the bus request bandwidth seen by the system bus interconnect. **NOTE** If the source and destination sizes are equal, this field is ignored between the first and second transfers and after the last write of each minor loop. This behavior is a side effect of reducing start-up latency. 00b - No eDMA engine stalls 01b - Reserved 10b - eDMA engine stalls for 4 cycles after each R/W 11b - eDMA engine stalls for 8 cycles after each R/W
13 —	Reserved
12-8 MAJORLINKCH	Major Loop Link Channel Number If (MAJORELINK = 0) then: • No channel-to-channel linking, or chaining, is performed after the major loop counter is exhausted. Otherwise:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] field to 1.
7 ESDA	Enable Store Destination Address As the channel completes the major loop by either the current iteration counter (CITER) decrementing to 0, or by receiving an enabled end-of-packet signal, this field enables writing the destination address (DADDR) to the address stored in the SLAST_SDA field. The value written to system memory is the last DADDR value prior to the DLAST_SGA offset being applied, or overwritten by an enabled scatter/gather operation. When the ESDA bit is 1, SLAST_SDA contains the write pointer instead of the final source address offset. Because this is a pointer and not a final offset, a last source address offset of zero is applied to SADDR instead of the SLAST_SGA value. 0b - Ability to store destination address to system memory disabled 1b - Ability to store destination address to system memory enabled
6 EEOP	Enable End-Of-Packet Processing When enabled by the EEOP field, an end-of-packet hardware input signal directs eDMA to discontinue executing the active channel, and to treat the shutdown as the major-loop-completed event. If the EEOP field is 1, the end-of-packet signal from supported peripherals is accepted. If the EEOP field is 0, the end-of-packet input is ignored. With an end-of-packet retirement, the current TCD destination address (or ESDA-saved destination address), minus the software-saved initial address (DADDR), reflects the total amount of data transferred. 0b - End-of-packet operation disabled 1b - End-of-packet hardware input signal enabled
5 MAJORELINK	Enable Link When Major Loop Complete As the channel completes the major loop, this flag enables linking to another channel defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. **NOTE** To support the dynamic linking coherency model, this field is forced to 0 if written when TCDn_CSR[DONE] is 1. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
4 ESG	Enable Scatter/Gather Processing As the channel completes the major loop, this flag enables scatter/gather processing in the current channel. If enabled, the eDMA engine uses TCDn_DLAST_SGA as a memory pointer to a 0-modulo 32-bit address containing a 32-byte data structure, which is loaded as the transfer control descriptor into local memory. **NOTE** To support the dynamic scatter/gather coherency model, this field is forced to 0 if written when TCDn_CSR[DONE] is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Current channel's TCD is normal format 1b - Current channel's TCD specifies scatter/gather format.
3 DREQ	Disable Request If this flag is 1, the eDMA hardware automatically clears the corresponding ERQ bit when the current major iteration count reaches 0. 0b - No operation. Channel's ERQ field not affected 1b - Clear the ERQ field to 0 upon major loop completion, thus disabling hardware service requests. Channel's ERQ field cleared to 0 when major loop complete
2 INTHALF	Enable Interrupt If Major Counter Half-complete If this flag is 1, the channel generates an interrupt request by setting the appropriate field in the INT register to 1 when the current major iteration count reaches the halfway point. Specifically, the comparison performed by the eDMA engine is (CITER = (BITER/2)). This halfway point interrupt request is provided to support double-buffered, also known as ping-pong, schemes, or other types of data movement where the processor needs an early indication of the transfer's progress. **NOTE** If BITER = 1, do not use INTHALF; use INTMAJOR instead. 0b - Halfway point interrupt disabled 1b - Halfway point interrupt enabled
1 INTMAJOR	Enable Interrupt If Major count complete If this flag is 1, the channel generates an interrupt request by setting the appropriate field in the INT register to 1 when the current major iteration count (CITER) reaches 0. 0b - End-of-major loop interrupt disabled 1b - End-of-major loop interrupt enabled
0 START	Channel Start If this flag is 1, the channel is requesting service. The eDMA hardware automatically clears this flag to 0 after the channel begins execution. 0b - Channel not explicitly started 1b - Channel explicitly started via a software-initiated service request

5.6.2.20 TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD31_BITER_ELINKNO)

Offset

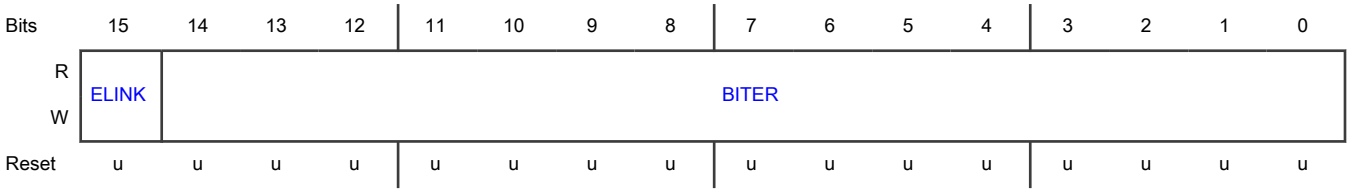
For n = 0 to 31:

Register	Offset
TCDn_BITER_ELINKNO	3Eh + (n × 1_0000h)

Function

If the TCDn_BITER[ELINK] field is 0, the TCDn_BITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enables Link As the channel completes the minor loop, this flag enables linking to another channel as defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. If channel linking is disabled, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-0 BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be set equal to the value in the CITER field. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER must be 0x0001.

5.6.2.21 TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD31_BITER_ELINKYES)

Offset

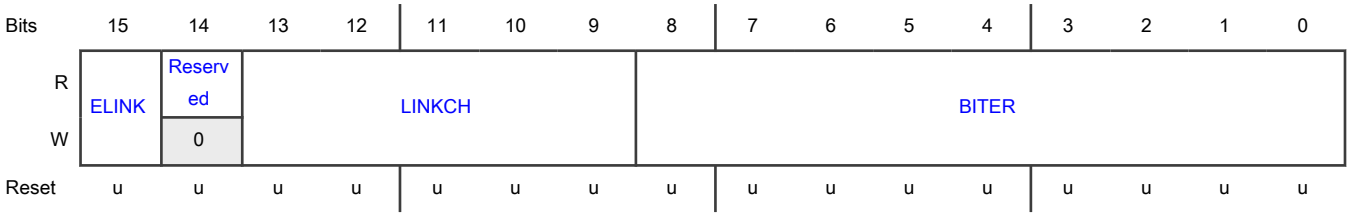
For n = 0 to 31:

Register	Offset
TCDn_BITER_ELINKYES	3Eh + (n × 1_0000h)

Function

If the TCDn_BITER[ELINK] field is set, the TCDn_BITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. If channel linking disables, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14 —	Reserved
13-9 LINKCH	Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] field. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8-0 BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be set equal to the value in the CITER field. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER must be 0x0001.

Chapter 6
Enhanced Direct Memory Access (eDMA4)

6.1 Chip-specific eDMA4 Information

Table 25. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

6.2 Overview

The enhanced direct memory access (eDMA) controller is capable of performing complex data transfers with minimal intervention from a host processor. The hardware microarchitecture includes:

- A DMA engine that performs:
  - Source address and destination address calculations
  - Data-movement operations
- Local memory containing transfer control descriptors for each of the 64 channels

6.2.1 Block diagram

Figure 16 illustrates the components of the eDMA system, including the eDMA module ("engine").

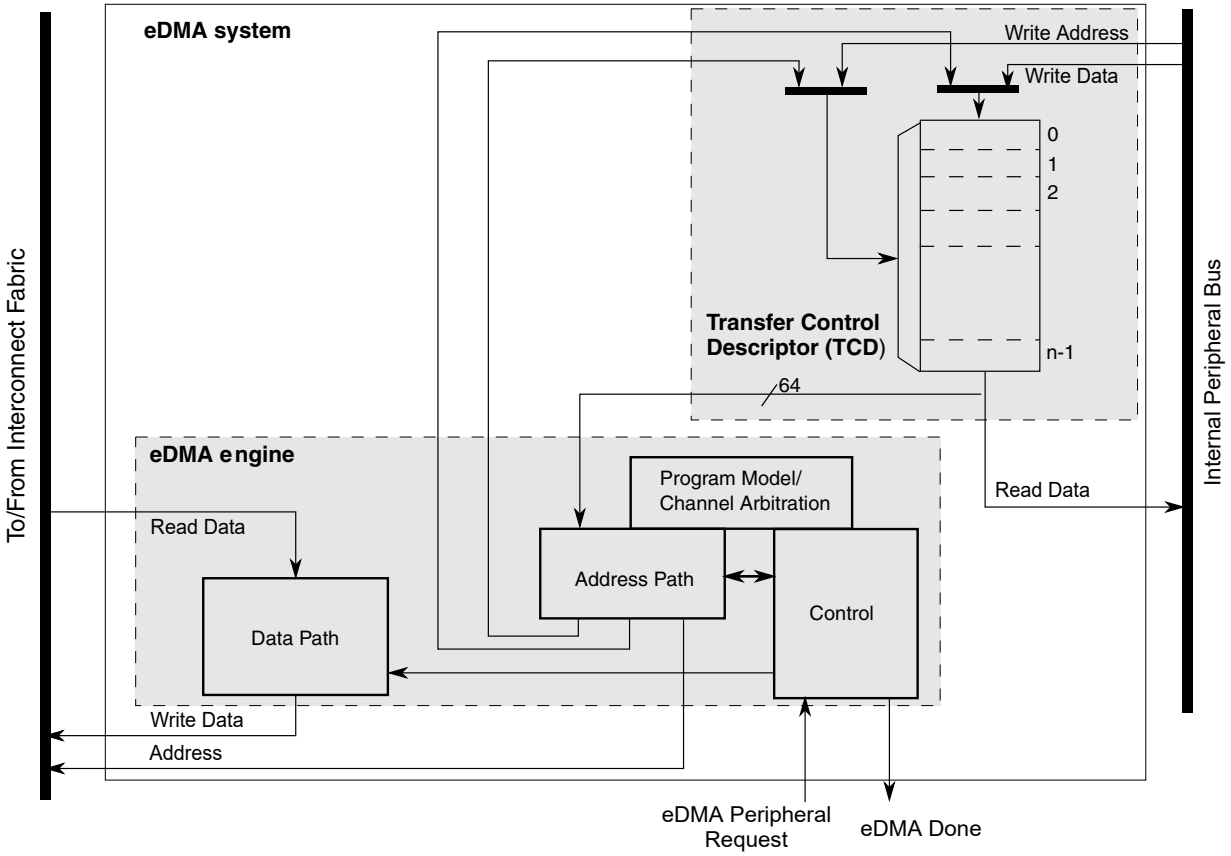


Figure 16. Block diagram

6.2.2 Block parts

The eDMA module is partitioned into two major modules: the eDMA engine and the transfer-control descriptor local memory.

The eDMA engine is further partitioned into four submodules:

Table 26. eDMA engine submodules

Submodule	Function
Address path	This block implements registered versions of two channel transfer control descriptors, a primary channel and secondary (preempt) channel, and manages all master bus-address calculations. All the channels provide the same functionality. This structure allows data transfers associated with one channel to be preempted after the completion of a read/write sequence if a higher priority channel activation is asserted while the first channel is active. After a channel is activated, it runs until the minor loop is completed, unless preempted by a higher priority channel. This provides a mechanism (enabled by CH*n*_PRI[ECP]) where a large data move operation can be preempted to minimize the time another channel is blocked from execution. When any channel is selected to execute, the contents of its TCD are read from local memory and loaded into the address path channel x registers for a normal start and into channel y registers for a preemption start. After the minor loop completes execution, the address path hardware writes the new values for the TCD*n*_{SADDR, DADDR, CITER} back to local memory. If the major iteration count is exhausted, additional processing is performed, including the final address pointer updates,

Table continues on the next page...

Table 26. eDMA engine submodules (continued)

Submodule	Function
	reloading the TCD*n*_CITER field, and a possible fetch of the next TCD*n* from memory as part of a scatter/gather operation.
Data path	This block implements the bus master read/write datapath. It includes a data buffer and the necessary multiplex logic to support any required data alignment. The internal read data bus is the primary input, and the internal write data bus is the primary output. The address and data path modules directly support the 2-stage pipelined internal bus. The address path module represents the 1st stage of the bus pipeline (address phase), while the data path module implements the 2nd stage of the pipeline (data phase).
Program model/channel arbitration	This block implements the first section of the eDMA programming model as well as the channel arbitration logic. The programming model registers are connected to the internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs are also connected to this block (via control logic).
Control	This block provides all the control functions for the eDMA engine. For data transfers where the source and destination sizes are equal, the eDMA engine performs a series of source read/destination write operations until the number of bytes specified in the minor loop byte count has moved. For descriptors where the sizes are not equal, multiple accesses of the smaller size data are required for each reference of the larger size. As an example, if the source size references 16-bit data and the destination is 32-bit data, two reads are performed, then one 32-bit write.

The transfer-control descriptor local memory is further partitioned into:

Table 27. Transfer control descriptor memory

Submodule	Description
Memory controller	This logic implements the required dual-ported controller, managing accesses from the eDMA engine as well as references from the internal peripheral bus. As noted earlier, in the event of simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage for each channel's transfer profile.

6.2.3 Features

The eDMA is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known and not defined within the transferred data itself. The eDMA module features:

- All data movement via dual-address transfers: read from source, write to destination plus a write-only initialization mode
  - Programmable source and destination addresses and transfer size
  - Support for complex address calculations
- 64-channel implementation that performs complex data transfers with minimal intervention from a host processor
  - Internal data buffer, used as temporary storage to support large multibyte transfers
  - Connections to the interconnect fabric for bus mastering the data movement
- Transfer control descriptor (TCD) organized to support two-deep, nested transfer operations
  - 32-byte TCD stored in local memory for each channel
  - An inner data transfer loop defined by a minor byte transfer count

  - An outer data transfer loop defined by a major iteration count
- Channel activation via one of three methods:
  - Explicit software initiation
  - Initiation via a channel-to-channel linking mechanism for continuous transfers
  - Peripheral-paced hardware requests, one per channel
- Fixed-priority and round-robin channel arbitration
- Channel completion reported via programmable interrupt requests
  - One interrupt per channel, which can be asserted at completion of major iteration count
  - Programmable error terminations per channel and logically summed together to form one error interrupt to the interrupt controller
- Programmable support for scatter/gather DMA processing
- Support for complex data structures

In the discussion of this module, *n* is used to reference the channel number.

6.3 Functional description

The operation of the eDMA is described in the following subsections.

6.3.1 Modes of operation

The eDMA operates in the following modes:

Table 28. Modes of operation

Mode	Description
Normal	In Normal mode, the eDMA transfers data between a source and a destination. The source and destination can be a memory block or an I/O block capable of operation with the eDMA. A service request initiates a transfer of a specific number of bytes (NBYTES) as specified in the transfer control descriptor (TCD). The minor loop is the sequence of read-write operations that transfers these NBYTES per service request. Each service request executes one iteration of the major loop, which transfers NBYTES of data.
Debug	eDMA operation is configurable in Debug mode via the control register: • If CSR[EDBG] is cleared, the eDMA continues to operate. • If CSR[EDBG] is set, the eDMA stops transferring data. If Debug mode is entered while a channel is active, the eDMA continues operation until the channel retires.

6.3.2 eDMA basic data flow

The basic flow of a data transfer can be partitioned into three segments.

As shown in the following diagram, the first segment involves the channel activation:

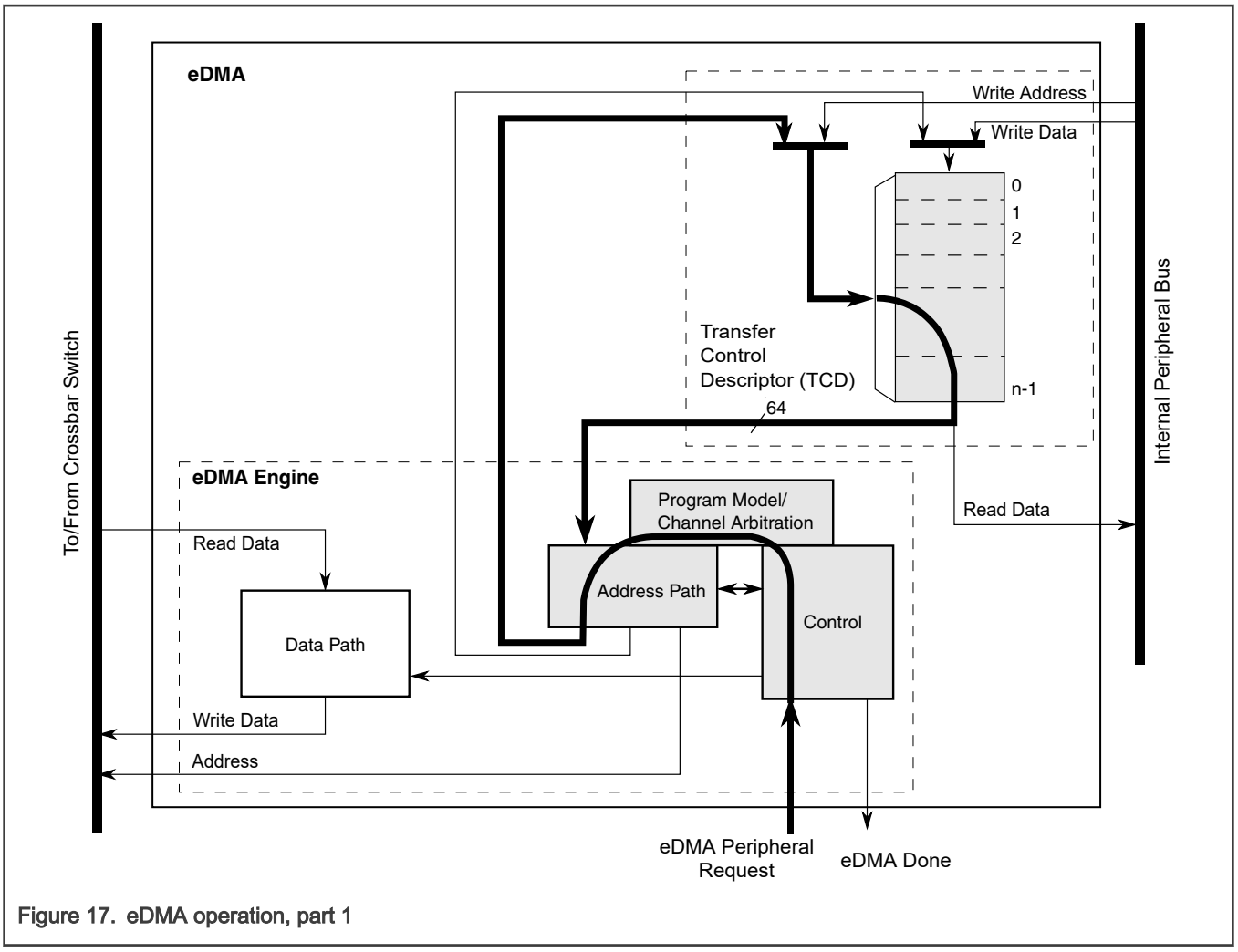


Figure 17. eDMA operation, part 1

This example uses the assertion of the eDMA peripheral request signal to request service for channel *n*. Channel activation via software and the TCD*n*_CSR[START] bit follows the same basic flow as peripheral requests. The eDMA request input signal is registered internally and then routed through the eDMA engine: first through the control module, then into the program model and channel arbitration. In the next cycle, the channel arbitration begins using fixed-priority plus the optional round-robin algorithm. After arbitration is complete, the activated channel number is sent through the address path and converted into the required address to access the local memory for TCD*n*. Next, the TCD memory is accessed and the required descriptor read from the local memory and loaded into the eDMA engine address path's primary or secondary channel execution registers. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the address path registers.

The following diagram illustrates the second part of the basic data flow:

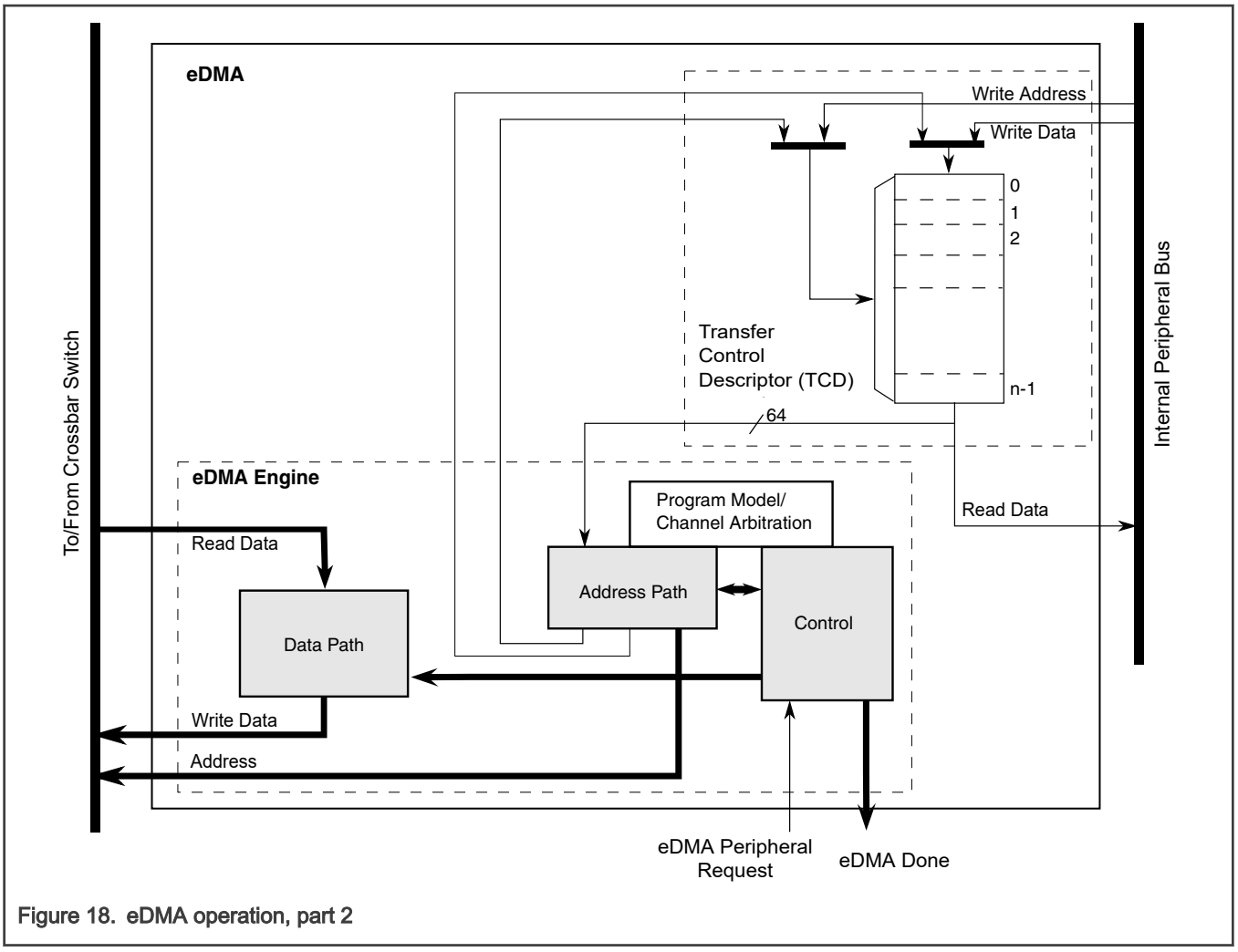


Figure 18. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) sequence through the required source reads and destination writes to perform the actual data movement. The source reads are initiated and the fetched data is temporarily stored in the data path block until it is gated onto the internal bus during the destination write. This source read/destination write processing continues until the byte count, NBYTES, has transferred.

After NBYTES of data has been moved, the final phase of the basic data flow is performed. In this segment, the address path logic performs the required updates to certain fields in the appropriate TCD, for example, SADDR, DADDR, CITER. If the major iteration count is exhausted, additional operations are performed. These include the final address adjustments and reloading of the BITER field into the CITER. Assertion of an optional interrupt request also occurs at this time, as does a possible fetch of a new TCD from memory using the scatter/gather address pointer included in the descriptor (if scatter/gather is enabled). The updates to the TCD memory and the assertion of an interrupt request are shown in the following diagram.

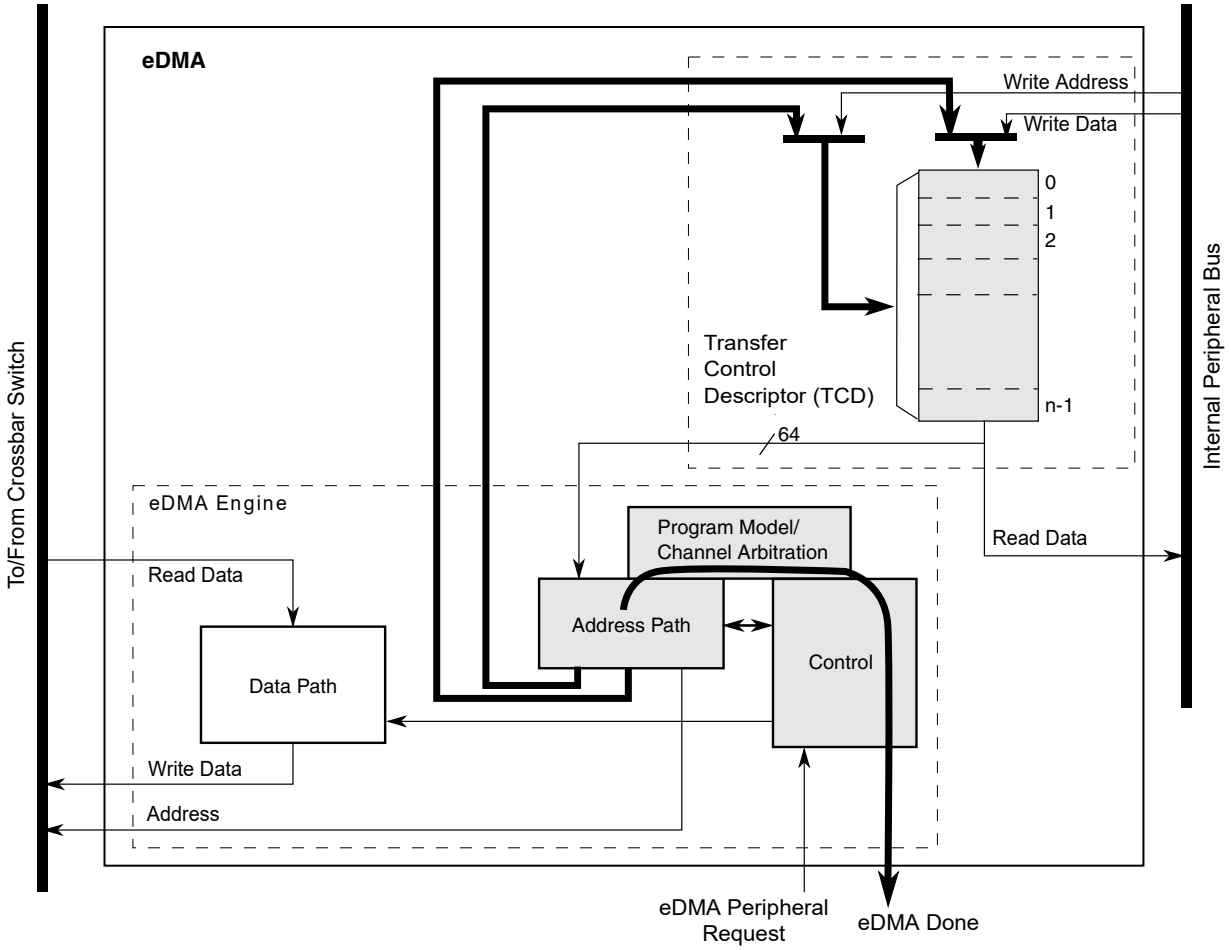


Figure 19. eDMA operation, part 3

6.3.3 Fault reporting and handling

Channel errors are reported in the Error Status register (ES) and can be caused by:

- A configuration error, which is an illegal setting in the transfer-control descriptor, or
- An active channel canceled via a "cancel transfer with error" software request, or
- An error termination to a bus master read or write cycle

A configuration error is reported when the starting source or destination address, source or destination offsets, minor loop byte count, or the transfer size represent an inconsistent state. Each of these possible causes are detailed below:

- The addresses and offsets must be aligned on 0-modulo-transfer-size boundaries.
- The minor loop byte count must be a multiple of the source and destination transfer sizes.
- All source reads and destination writes must be configured to the natural boundary of the programmed transfer size respectively.

NOTE

To aid in debug, set the Halt After Error bit in the DMA's Control Status Register, CSR[HAE]. Upon any error condition, the DMA will be halted after the error is recorded. The DMA will remain halted and will not process any channel service requests. Once the error is fixed, the DMA may be enabled again by clearing the Halt bit, CSR[HALT].

- If a scatter/gather operation is enabled upon channel completion, a configuration error is reported if the scatter/gather address (DLAST_SGA) is not aligned on a 32-byte boundary.
- If minor loop channel linking is enabled upon channel completion, a configuration error is reported when the link is attempted if the TCDn_CITER[ELINK] bit does not equal the TCDn_BITER[ELINK] bit.

If enabled, all configuration error conditions, except the scatter/gather and minor-loop link errors, report as the channel activates and asserts an error interrupt request if enabled. A scatter/gather configuration error is reported when the scatter/gather operation begins at major loop completion when properly enabled. A minor loop channel link configuration error is reported when the link operation is serviced at minor loop completion.

If a system bus read or write is terminated with an error, the data transfer is stopped and the appropriate bus error flag set. In this case, the state of the channel's transfer control descriptor is updated by the eDMA engine with the current source address, destination address, and current iteration count at the point of the fault. When a system bus error occurs, the channel terminates processing. Due to pipeline effect, transfers already in progress when the bus error is received by the eDMA are not aborted. Bus errors are imprecise. The source and destination addresses stored in the TCD after a bus error may not be the address that caused the error.

A transfer may be cancelled by software with the CSR[CX] bit. When a cancel transfer request is recognized, the DMA engine stops processing the channel. Transactions in flight are allowed to finish. If the cancel occurs on the last sequence of a major or minor loop, the cancel request is discarded and the channel retires normally.

The error cancel transfer is the same as a cancel transfer except the Error Status register (ES) is updated with the cancelled channel number and ECX is set. The TCD of a cancelled channel contains the source and destination addresses of the last transfer saved in the TCD. If the channel needs to be restarted, you must re-initialize the TCD because the aforementioned fields no longer represent the original parameters. When a transfer is cancelled by the error cancel transfer mechanism, the channel number is loaded into the Management Page ES[ERRCHN], and ECX and VLD are set. In addition, an error interrupt may be generated if enabled.

NOTE

The cancel transfer request allows the user to stop a large data transfer in the event the full data transfer is no longer needed. The cancel transfer bit does not abort the channel. It simply stops the transferring of data and then retires the channel through its normal shutdown sequence. The cancel transfer is an imprecise operation. The application software must handle the context of the cancel. If an interrupt is desired (or not), then the interrupt should be enabled (or disabled) before the cancel request. The application software must clean up the transfer control descriptor since the full transfer did not occur.

The occurrence of any error causes the eDMA engine to stop normal processing of the active channel immediately (it goes to its error processing states and the transaction to the system bus still has pipeline effect), and the appropriate channel bit in the eDMA error register is asserted. At the same time, the details of the error condition are loaded into the Error Status register (ES). The major loop complete indicators, setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request, are not affected when an error is detected. After the error status has been updated, the eDMA engine continues operating by servicing the next appropriate channel. A channel that experiences an error condition is automatically disabled from hardware service requests. If a channel is terminated by an error, its CHn_CSR[ERQ] enable hardware requests bit is cleared thus preventing continuous errors from a peripheral's service request.

The error status bits are read-only. These error indicators are sticky and cannot be cleared. They will show the last recorded error until the DMA is reset. The valid bit (VLD) can be used to determine is a new error condition exists. This bit is the logical OR of each channel's error interrupt bit (ERR). Once the software has resolved any errors and cleared all of the error interrupt bits, the valid bit will be cleared, but the cause of the last error will still be indicated.

6.3.4 Channel preemption

The eDMA uses a priority vector value to determine the highest priority channel requesting service.

The priority vector is a combination of:

1. the channel's group priority, CHn_GRPRI
2. the channel's priority level, CHn_PRI[APL]

3. the channel number

Priority vector = ((CHn_GRPRI << 8) + (CHn_PRI[APL] << 5) + CHn_*)

A channel requesting service with the highest priority vector value will receive the next execution slot.

An execution slot is available:

1. immediately if the eDMA is idle
2. when an active channel retires
3. when valid preemption conditions exist

NOTE

Preemption is strictly priority based. Preemption is not bound by a specific group number as defined by CHn_GRPRI.

Channel preemption is enabled on a per-channel basis by setting the CHn_PRI[ECP] bit. Channel preemption allows the executing channel's data transfers to temporarily suspend in favor of starting a higher priority channel. After the preempting channel has completed all its minor loop data transfers, the preempted channel is restored and resumes execution. After the restored channel completes one read/write sequence, it is again eligible for preemption. If any higher priority channel is requesting service, the restored channel is suspended and the higher priority channel is serviced. Nested preemption, that is, attempting to preempt a preempting channel, is not supported. After a preempting channel begins execution, it cannot be preempted.

A channel's ability to preempt another channel can be disabled by setting CHn_PRI[DPA]. When a channel's preempt ability is disabled, that channel cannot suspend a lower priority channel's data transfer, regardless of the lower priority channel's ECP setting. This allows for a pool of low priority, large data-moving channels to be defined. These low priority channels can be configured to not preempt each other, thus preventing a low priority channel from consuming the preempt slot normally available to a true, high priority channel. When Round-Robin channel arbitration mode is enabled, CSR[ERCA] = 1, any channel with a priority level equal to zero, CHn_PRI[APL] = 0, has preemption disabled and will not preempt another channel.

6.3.5 Clocking

This module has no clocking considerations.

6.3.6 Interrupts

This module has no interrupts.

6.4 External signals

This module has no external signals.

6.5 Initialization

The following sections discuss initialization of the eDMA and programming considerations.

6.5.1 eDMA initialization

To initialize the eDMA:

1. Write to the CSR if a configuration other than the default is desired.
2. Write the channel priority levels to the CH_PRI registers and group priority levels to the MP_GPRI registers if a configuration other than the default is desired.
3. Enable error interrupts in the CH_CSR[EEI] registers if so desired.
4. Write the 32-byte TCD for each channel that may request service.
5. Enable any hardware service requests via the CH_CSR[ERQ] registers.

6. Request channel service via either:
   - Software: setting the TCD_CSR[START]
   - Hardware: slave device asserting its eDMA peripheral request signal

After any channel requests service, a channel is selected for execution based on the arbitration and priority levels written into the programmer's model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in the following table, for the selected channel into its internal address path module.

As the TCD is read, the first transfer is initiated on the internal bus, unless a configuration error is detected. Transfers from the source, as defined by TCD*n*_SADDR, to the destination, as defined by TCD*n*_DADDR, continue until the number of bytes specified by TCD*n*_NBYTES are transferred.

When the transfer is complete, the eDMA engine's local TCD*n*_SADDR, TCD*n*_DADDR, and TCD*n*_CITER are written back to the main TCD memory and any minor loop channel linking is performed, if enabled. If the major loop is exhausted, further post processing executes, such as interrupts, major loop channel linking, and scatter/gather operations, if enabled.

Table 29. TCD Control and Status (TCD_CSR) fields

TCD*n*_CSR field name	Description
START	Control bit to start channel explicitly when using a software initiated DMA service (Automatically cleared by hardware)
EEOP	Control bit to enable End-Of-Packet processing
ESDA	Control bit to enable the storing of the destination address to system memory after the major loop completes
DREQ	Control bit to disable hardware initiated DMA service requests after major loop completion
BWC	Control bits for throttling bandwidth control of a channel
ESG	Control bit to enable scatter-gather feature
INTHALF	Control bit to enable interrupt when major loop is half complete
INTMAJ	Control bit to enable interrupt when major loop completes

Table 30. Channel Control and Status (CH_CSR) fields

TCD*n*_CSR field name	Description
ACTIVE	Status bit indicating the channel is currently in execution
DONE	Status bit indicating major loop completion (cleared by software when a channel begins execution)
EEI	Control bit to enable error interrupts
EARQ	Control bit to enable external, asynchronous wake-up event in conjunction with the ERQ bit
ERQ	Control bit to enable hardware service requests

The following figure shows how each DMA request initiates one minor-loop transfer, or iteration, without CPU intervention. DMA arbitration can occur after each minor loop, and one level of minor loop DMA preemption is allowed. The number of minor loops in a major loop is specified by the beginning iteration count (BITER).

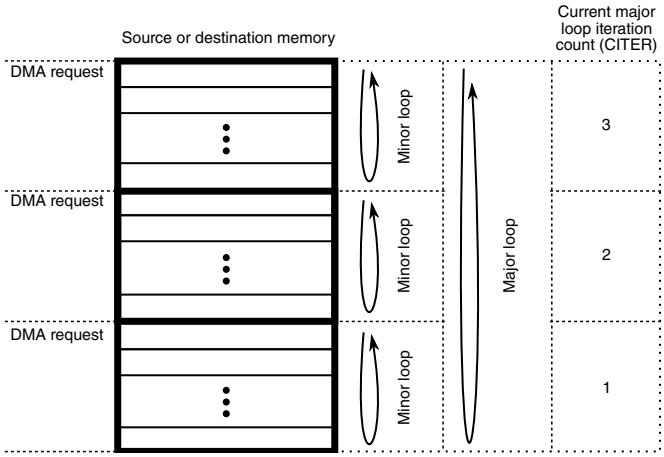


Figure 20. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings interrelate.

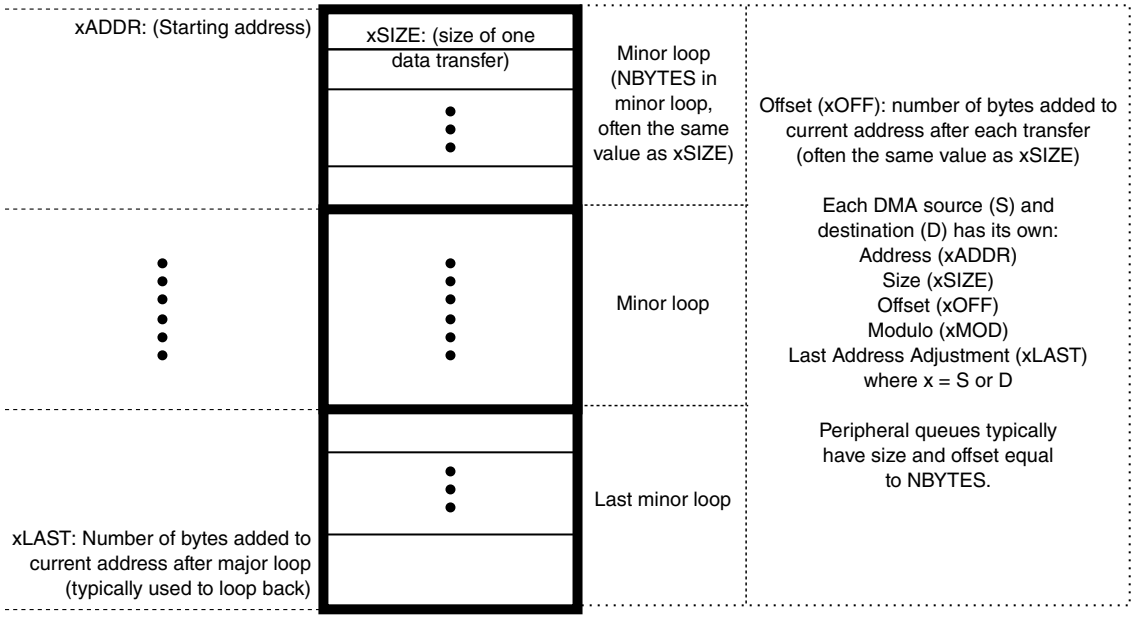


Figure 21. Memory array terms

6.5.2 eDMA arbitration

The eDMA uses a layered arbitration scheme composed of multiple priority levels. The eDMA uses a fixed priority arbitration scheme with an optional, round-robin arbitration under specific conditions. The priorities are evaluated in the following order:

1. Highest

   Arbitration Group Priority. Each channel is assigned an arbitration group via the MP_GRPI registers. Priority is given the to highest value (31 being the highest possible value) down to the lowest value (zero, the default).

2. Channel Priority. Each channel is assigned a channel priority level via the CH_PRI registers. The channel priority is a relative priority level within an arbitration group. Priority is given to the highest value (7 being the highest possible value) down to the lowest value (zero, the default). Channel priorities within each arbitration group do not have to be unique. If multiple channels have the same channel priority level, the channel number will be used to determine priority as defined in item 3.

3. Channel Number. When two or more channels have the same arbitration group priority and channel priority, the channel number (CH_NUM) is used to determine the highest priority. Priority is giver to the highest channel number. Lowest priority is channel 0. The channel numbers are static and cannot be changed in the programmer's model.
4. Lowest

   When round robin is enabled, any channel configured for round robin operation has lowest priority within an arbitration group. Round robin is enabled be setting the CSR[ERCA] bit. Once enabled, channels with a channel priority of zero (CH_PRI=0) will use round robin arbitration. Round robin arbitration will rotate the channel selection amongst the channel's requesting service with CH_PRI=0 within the arbitration group. Any nonzero channel within the arbitration group will continue to use fixed priority arbitration and if requesting service, will be selected over any round robin channels.

For fixed arbitration, the overall priority can be consider a number composed of three concatenated priority levels- MP_GRPI:CH_PRI:CH_NUM. The largest number will have the highest priority and the lowest number will have the lowest priority.

For round robin arbitration, the priority number is- MP_GRPI:0:X. The CH_PRI=0 channels requesting service are rotated through without regard to priority amongst these channels. Any channel within the arbitration group with a CH_PRI > 0 will be serviced before the round robin channels.

6.5.3 Programming errors

The eDMA performs various tests on the transfer control descriptor to verify consistency in the descriptor data.

The channel number causing the error is recorded in the Error Status register (ES). If the error source is not removed before the next activation of the problem channel, the error is detected and recorded again. Setting the halt after error bit, CSR[HAE], will halt the DMA and prevent the recurrence of the error.

6.5.4 Arbitration mode considerations

This section discusses arbitration considerations for the eDMA.

6.5.4.1 Fixed group arbitration, Fixed channel arbitration

In this mode, the channel service request from the highest priority channel in the highest priority group is selected to execute. If the eDMA is programmed so that the channels within a high priority group have a high number of request or large data transfers, that group may consume all the bandwidth of the eDMA controller. That is, no lower priority groups will be serviced if there is always at least one DMA request pending on a channel in the highest priority group when the controller arbitrates the next DMA request. The advantage of this scenario is that latency can be small for channels that need to be serviced quickly.

6.5.4.2 Fixed group arbitration, Round-robin channel arbitration

The highest priority group with a request will be serviced. Lower priority groups will be serviced if no pending requests exist in the higher priority groups.

Within each group, channels are serviced starting with the highest non-zero channel priority. For all channels with a channel priority programmed to zero, the highest channel number requesting service is selected and then rotating through to the lowest channel number requesting service. The round-robin channel arbitration may provide a fairness mechanism to lower priority channels.

This scenario could cause the same bandwidth consumption problem as indicated in Fixed group arbitration, Fixed channel arbitration, but all the channels in the highest priority group will be serviced. Service latency will be short on the highest priority group, but could potentially be very much longer as the group priority decreases.

6.5.5 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

6.5.5.1 Single request

To perform a simple transfer of n bytes of data with one activation, set the major loop to one (TCD*n*_CITER = TCD*n*_BITER = 1). The data transfer begins after the channel service request is acknowledged and the channel is selected to execute. After the transfer is complete, the CH*n*_CSR[DONE] bit is set and an interrupt generates if properly enabled.

For example, the following TCD entry is configured to transfer 16 bytes of data. The eDMA is programmed for one iteration of the major loop transferring 16 bytes per iteration. The source memory has a byte wide memory port located at 0x1000. The destination memory has a 32-bit port located at 0x2000. The address offsets are programmed in increments to match the transfer size: one byte for the source and four bytes for the destination. The final source and destination addresses are adjusted to return to their beginning values.

```
TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
TCDn_DLAST_SGA= -16
TCDn_CSR[INTMAJ] = 1
TCDn_CSR[START] = 1 (Should be written last after all other fields have been initialized)
All other TCDn fields = 0
```

This generates the following event sequence:

1. User write to the TCD*n*_CSR[START] bit requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:
    a. Continuously read data from the source addresses by issuing read transactions and adjusting the source address via the source address offset value after each transaction.
    b. Continuously write data to the destination addresses after sufficient data has been buffered. Adjust the destination address via the destination address offset after each transaction.
6. The eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 1 (TCD*n*_BITER).
7. The eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
8. The channel retires and the eDMA goes idle or services the next channel.

6.5.5.2 Multiple requests

The following example transfers 32 bytes via two hardware requests, but is otherwise the same as the previous example. The only fields that change are the major loop iteration count and the final address offsets. The eDMA is programmed for two iterations of the major loop transferring 16 bytes per iteration. After the channel's hardware requests are enabled via the CH_CSR[ERQ] register bit, the slave device initiates channel service requests.

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

This would generate the following sequence of events:

1. First hardware, that is, eDMA peripheral, request for channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD*n* data from local memory to internal register file.
5. The source to destination transfers are executed as follows:
    a. Continuously read data from the source addresses by issuing read transactions and adjusting the source address via the source address offset value after each transaction.
    b. Continuously write data to the destination addresses after sufficient data has been buffered. Adjust the destination address via the destination address offset after each transaction.
6. eDMA engine writes: TCD*n*_SADDR = 0x1010, TCD*n*_DADDR = 0x2010, TCD*n*_CITER = 1.
7. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0.
8. The channel retires → one iteration of the major loop. The eDMA goes idle or services the next channel.
9. Second hardware, that is, eDMA peripheral, requests channel service.
10. The channel is selected by arbitration for servicing.
11. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
12. eDMA engine reads: channel TCD data from local memory to internal register file.
13. The source to destination transfers are executed as follows:
    a. Continuously read data from the source addresses by issuing read transactions and adjusting the source address via the source address offset value after each transaction.
    b. Continuously write data to the destination addresses after sufficient data has been buffered. Adjust the destination address via the destination address offset after each transaction.
14. eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 2 (TCD*n*_BITER).
15. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
16. The channel retires → major loop complete. The eDMA goes idle or services the next channel.

6.5.5.3 Using the modulo feature

The modulo feature of the eDMA provides the ability to implement a circular data queue in which the size of the queue is a power of 2. MOD is a 5-bit field for the source and destination in the TCD, and it specifies which lower address bits increment from their original value after the address+offset calculation. All upper address bits remain the same as in the original value. A setting of 0 for this field disables the modulo feature. Modulo addressing applies to cases where the minor loop offset is enabled; that is, the upper address bits remain the same after the minor loop offset is added to the source or destination address.

The following table shows how the transfer addresses are specified based on the setting of the MOD field. Here a circular buffer is created where the address wraps to the original value while the 28 upper address bits (0x1234567*x*) retain their original value. In this example the source address is set to 0x12345670, the offset is set to 4 bytes and the MOD field is set to 4, allowing for a 2^4 byte (16-byte) size queue.

Table 31. Modulo example

Transfer number	Address
1	0x12345670

Table continues on the next page...

Table 31. Modulo example (continued)

Transfer number	Address
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

6.5.6 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

6.5.6.1 Testing for minor loop completion

There are two methods to test for minor loop completion when using software initiated service requests. The first is to read the TCD*n*_CITER field and test for a change. Another method may be extracted from the sequence shown below. The second method is to test the TCD*n*_CSR[START] bit and the CH*n*_CSR[ACTIVE] bit. The minor-loop-complete condition is indicated by both bits reading zero after the TCD*n*_CSR[START] was set. Polling the CH*n*_CSR[ACTIVE] bit only may be inconclusive because the active status may be missed if the channel execution is short in duration.

The CH_CSR and TCD_CSR status bits execute the following sequence for a software activated channel:

Stage	TCD*n*_CSR bits			State
	START	ACTIVE	DONE	
1	1	0	0	Channel service request via software
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

The best method to test for minor-loop completion when using hardware, that is, peripheral, initiated service requests is to read the TCD*n*_CITER field and test for a change. The hardware request and acknowledge handshake signals are not visible in the programmer's model.

The TCD status bits execute the following sequence for a hardware-activated channel:

Stage	TCD*n*_CSR bits			State
	START	ACTIVE	DONE	
1	0	0	0	Channel service request via hardware (peripheral request asserted)

Table continues on the next page...

Table continued from the previous page...

Stage	TCD*n*_CSR bits			State
	START	ACTIVE	DONE	
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

For both activation types, the major-loop-complete status is explicitly indicated via the CH*n*_CSR[DONE] bit.

The TCD*n*_CSR[START] bit is cleared automatically when the channel begins execution regardless of how the channel activates.

6.5.6.2 Checking channel preemption status

A preemptive situation is one in which a preempt-enabled channel is executing and a higher priority request becomes active. When Round-Robin channel arbitration mode is enabled, all channels with their channel priority set to zero lose their preempt ability. Channel priorities of zero are treated as equal, that is, constantly rotating, when Round-Robin arbitration mode is enabled.

The CH*n*_CSR[ACTIVE] bit for the preempted channel remains asserted throughout the preemption. The preempted channel is temporarily suspended while the preempting channel executes one major loop iteration. If two CH*n*_CSR[ACTIVE] bits are set simultaneously in the global TCD map, a higher priority channel is actively preempting a lower priority channel.

6.5.7 Channel linking

Channel linking (or chaining) is a mechanism where one channel sets the TCD*n*_CSR[START] bit of another channel (or itself), therefore initiating a service request for that channel. When properly enabled, the eDMA engine automatically performs this operation at the major or minor loop completion.

The minor loop channel linking occurs at the completion of the minor loop (or one iteration of the major loop). The TCD*n*_CITER[ELINK] field determines whether a minor loop link is requested. When enabled, the channel link is made after each iteration of the major loop except for the last. When the major loop is exhausted, only the major loop channel link fields are used to determine if a channel link should be made. For example, the initial fields of:

```
TCDn_CITER[ELINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER] value = 0x4
TCDn_CSR[MAJORELINK] = 1
TCDn_CSR[MAJORLINKCH] = 0x7
```

executes as:

1. Minor loop done → set TCD12_CSR[START] bit
2. Minor loop done → set TCD12_CSR[START] bit
3. Minor loop done → set TCD12_CSR[START] bit
4. Minor loop done, major loop done→ set TCD7_CSR[START] bit

When minor loop linking is enabled (TCD*n*_CITER[ELINK] = 1), the TCD*n*_CITER[CITER] field uses a nine bit vector to form the current iteration count. When minor loop linking is disabled (TCD*n*_CITER[ELINK] = 0), the TCD*n*_CITER[CITER] field uses a 15-bit vector to form the current iteration count. The bits associated with the TCD*n*_CITER[LINKCH] field are concatenated onto the CITER value to increase the range of the CITER.

NOTE

The TCD*n*_CITER[ELINK] bit and the TCD*n*_BITER[ELINK] bit must equal or a configuration error is reported. The CITER and BITER vector widths must be equal to calculate the major loop, half-way done interrupt point.

The following table summarizes how a DMA channel can link to another DMA channel, i.e, use another channel's TCD, at the end of a loop.

Table 32. Channel Linking Parameters

Desired Link Behavior	TCD Control Field Name	Description
Link at end of Minor Loop	TCD_CITER[ELINK]	Enable channel-to-channel linking on minor loop completion (current iteration)
	TCD_CITER[LINKCH]	Link channel number when linking at end of minor loop (current iteration)
Link at end of Major Loop	TCD_CSR[MAJORELINK]	Enable channel-to-channel linking on major loop completion
	TCD_CSR[MAJORLINKCH]	Link channel number when linking at end of major loop

6.5.8 Dynamic channel linking

Dynamic channel linking is the process of setting the TCDn_CSR[MAJORELINK] and TCDn_CSR[MAJORLINKCH] fields during channel execution. This bits are read from the TCD profile at the end of channel execution, thus allowing the user to enable the feature during channel execution. When attempting a dynamic channel link, the user must write the TCDn_CSR[MAJORELINK] and TCDn_CSR[MAJORLINKCH] in a single write transaction; that being a write size of at least 16 bits to the TCDn_CSR register.

Because the user is allowed to change the configuration during execution, a coherency model is needed. Consider the scenario where the user attempts to execute a dynamic channel link by enabling the TCD_CSR[MAJORELINK] bit at the same time the eDMA engine is retiring the channel. The TCD_CSR[MAJORELINK] would be set in the programmer’s model, but it would be unclear whether the actual link was made before the channel retired.

The following coherency model is recommended when executing a dynamic channel link request.

1. Write 1 to the TCDn_CSR[MAJORELINK] bit.
2. Read back the TCDn_CSR[MAJORELINK] bit.
3. Test the TCDn_CSR[MAJORELINK] request status:
   - If TCDn_CSR[MAJORELINK] = 1, the dynamic link attempt was successful.
   - If TCDn_CSR[MAJORELINK] = 0, the attempted dynamic link did not succeed (the channel was already retiring).

The TCD local memory controller forces the TCDn_CSR[MAJORELINK] bit to zero on any writes to a channel’s TCDn_CSR after that channel’s CHn_CSR[DONE] bit is set, indicating the major loop is complete.

NOTE

The user must clear the CHn_CSR[DONE] bit before writing the TCDn_CSR[MAJORELINK] bit. The CHn_CSR[DONE] bit is cleared automatically by the eDMA engine after a channel begins execution.

6.5.9 Suspend/resume a DMA channel with active hardware service requests

The DMA allows the user to move data from memory or peripheral registers to another location in memory or peripheral registers without CPU interaction. Once the DMA and peripherals have been configured and are active, it is rare to suspend a peripheral's service request dynamically. In this scenario, there are certain restrictions to disabling a DMA hardware service request. For coherency, a specific procedure must be followed. This section provides guidance on how to coherently suspend and resume a Direct Memory Access (DMA) channel when the DMA is triggered by a slave module.

6.5.9.1 Suspend an active DMA channel

To suspend an active DMA channel:

1. Stop the DMA service request at the peripheral first. Confirm it has been disabled by reading back the appropriate register in the peripheral.
2. Check the DMA's Hardware Request Status Register (DMA_HRSn) to ensure there is no service request to the DMA channel being suspended. Then disable the hardware service request by clearing the ERQ bit on appropriate DMA channel.

For example, assume the SPI is set as a master for transmitting data via a DMA service request when the DSPI_TXFIFO has an empty slot. The DMA will transfer the next command and data to the TXFIFO upon the request. If the user needs to suspend the DMA/SPI transfer loop, perform the following steps:

1. Disable the DMA service request at the source by writing 0 to DSPI_RSER[TFFF_RE]. Confirm that DSPI_RSER[TFFF_RE] is 0.
2. Ensure there is no DMA service request from the SPI by verifying that DMA_HRS[HRS*n*] is 0 for the appropriate channel. If no service request is present, disable the DMA channel by clearing the channel's ERQ bit. If a service request is present, wait until the request has been processed and the HRS bit reads zero.

6.5.9.2 Resume a DMA channel

To resume a DMA channel:

1. Enable the DMA service request on the appropriate channel by setting the its ERQ bit.
2. Enable the DMA service request at the peripheral.

6.6 Memory map/register definition

The eDMA's programming model is partitioned into three parts:

- The first part defines a number of registers providing overall control functions: the management page
- The second part corresponds to the channel (CH) control, status, and configuration
- The third part corresponds to the local transfer control descriptor (TCD) memory

TCD memory

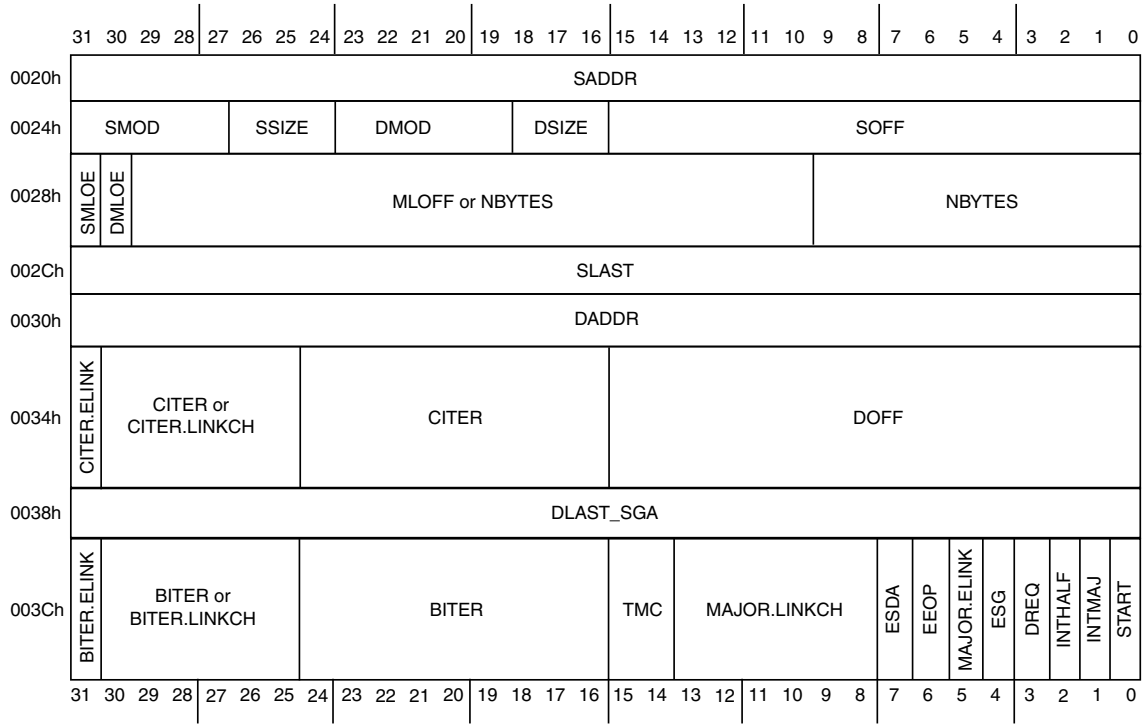
Each channel requires a 32-byte transfer control descriptor for defining the desired data movement operation. The channel descriptors are stored in the local memory in sequential order: channel 0, channel 1, ... channel 63.

This table provides a 32-bit view of the eDMA's memory map.

TCD initialization

The TCD memory is in an unknown state after reset. Only the TCD START bit is initialized to 0. Prior to activating a channel, you must initialize its TCD with the appropriate transfer profile.

TCD structure



Reserved memory and bit fields

- Reading reserved bits in a register returns the value of zero.
- Writes to reserved bits in a register are ignored.
- Reading or writing a reserved memory location generates a bus error.

6.6.1 DMA MP register descriptions

6.6.1.1 MP memory map

DMA4.MP base address: 4200_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Management Page Control Register (MP_CSR)	32	RW	0040_0000h
4h	Management Page Error Status Register (MP_ES)	32	R	0000_0000h
8h	Management Page Interrupt Request Status Register - Low (MP_INT_LOW)	32	R	0000_0000h
Ch	Management Page Interrupt Request Status Register- High (MP_INT_HIGH)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10h	Management Page Hardware Request Status Register - Low (MP_HRS_LOW)	32	R	0000_0000h
14h	Management Page Hardware Request Status Register - High (MP_HRS_HIGH)	32	R	0000_0000h
100h - 1FCh	Channel Arbitration Group Register (CH0_GRPRI - CH63_GRPRI)	32	RW	0000_0000h

6.6.1.2 Management Page Control Register (MP_CSR)

Offset

Register	Offset
MP_CSR	0h

Function

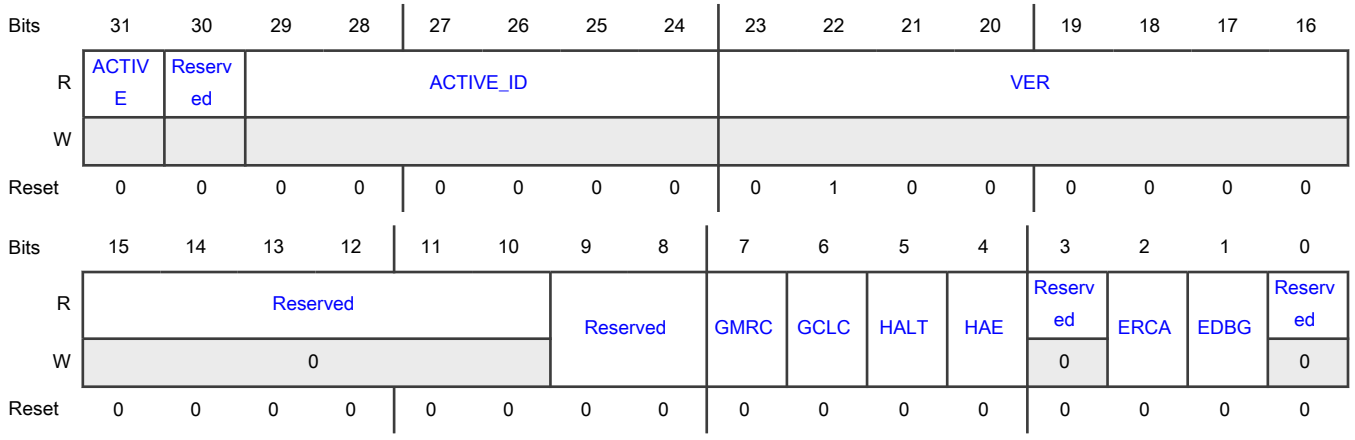
The Management Page Control Register defines the basic operating configuration of the DMA.

Arbitration uses a two-tier priority system; group and channel priority. Each channel is assigned a priority group. Group arbitration is fixed-priority and cannot be changed. Channel arbitration uses fixed priority and may be configured to use a selective round-robin scheme for specified channels within each priority group. For fixed-priority arbitration, the highest priority channel requesting service in the highest priority arbitration group is selected to execute. The channel priority registers assign the relative priorities within each arbitration group; see the CH *n* _PRI registers. All channels with a non-zero CH *n* _PRI value use fixed-priority arbitration. When round-robin arbitration is enabled, any channel with its channel priority set to zero does not have a priority and, of these channels requesting service, are cycled through (from high to low channel number) without regard to priority relative to each other within the same priority group. Any channel with a non-zero CH *n* _PRI value automatically has a higher priority over the round-robin channels. A channel's priority group is assigned in the management page group priority registers (MP_GRPRI).

NOTE

For correct operation, changes to the CSR[ERCA, GCLC, GMRC] bits must be performed when the DMA channels are inactive; that is, when CSR[ACTIVE] bit is cleared.

Diagram



Fields

Field	Function
31 ACTIVE	DMA Active Status 0b - eDMA is idle. 1b - eDMA is executing a channel.
30 —	Reserved
29-24 ACTIVE_ID	Active channel ID
23-16 VER	eDMA version
15-10 —	Reserved
9-8 —	Do not change the value of this field.
7 GMRC	Global Master ID Replication Control **NOTE** If master ID replication is disabled, the nonsecure, privileged protection level for DMA transfers is used. 0b - Master ID replication is disabled for all channels. 1b - Master ID replication is available and is controlled by each channel's CHn_SBR[EMI] setting.
6 GCLC	Global Channel Linking Control 0b - Channel linking is disabled for all channels. 1b - Channel linking is available and controlled by each channel's link settings.
5 HALT	Halt DMA Operations 0b - Normal operation 1b - Stall the start of any new channels. Executing channels are allowed to complete. Channel execution resumes when this bit is cleared.
4 HAE	Halt After Error 0b - Normal operation 1b - Any error causes the HALT bit to set. Subsequently, all service requests are ignored until the HALT bit is cleared.
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 ERCA	Enable Round Robin Channel Arbitration 0b - Round robin channel arbitration is disabled. Fixed priority arbitration is used for channel selection within each group . 1b - Round robin channel arbitration is enabled. Round robin arbitration is used for channel selection within each group .
1 EDBG	Enable Debug 0b - Debug mode is disabled. When in debug mode, the DMA continues to operate. 1b - Debug mode is enabled. When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. Channel execution resumes when the system exits debug mode or the EDBG bit is cleared.
0 —	Reserved

6.6.1.3 Management Page Error Status Register (MP_ES)

Offset

Register	Offset
MP_ES	4h

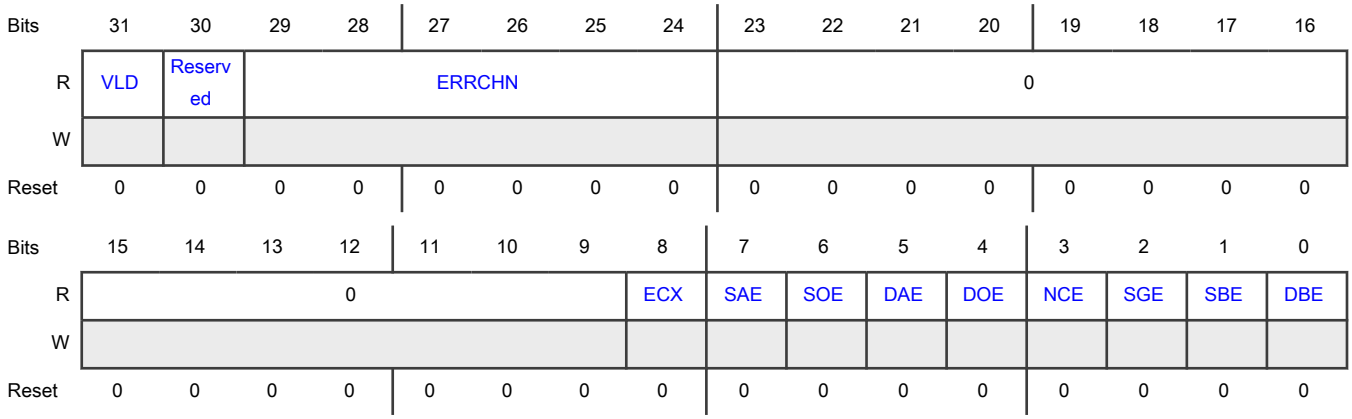
Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer-control descriptor
- An error termination to a bus master read or write cycle
- An uncorrectable error occurred while the device was accessing the TCD SRAM
- A cancel transfer with error request was made via the corresponding cancel transfer bit or input signal

Upon any error condition, the software must initialize the errant channel's TCD as it is in an incomplete state after an error. See Fault reporting and handling for more details.

Diagram



Fields

Field	Function
31 VLD	Valid Logical OR of all ERR status bits. 0b - No ERR bits are set. 1b - At least one ERR bit is set indicating a valid error exists that has not been cleared.
30 —	Reserved
29-24 ERRCHN	Error Channel Number or Canceled Channel Number The channel number of the last recorded error or last recorded error canceled transfer.
23-9 —	Reserved
8 ECX	Transfer Canceled The ECX operation is a Management Page function. When employed, the targeted channel's CH_ES register will report an unspecified error; only the ERR bit is set. The Management Page has full view of the error condition. 0b - No canceled transfers 1b - The last recorded entry was a canceled transfer by the error cancel transfer input.
7 SAE	Source Address Error 0b - No source address configuration error. 1b - The last recorded error was a configuration error detected in the TCDn_SADDR field. TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE].
6 SOE	Source Offset Error 0b - No source offset configuration error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - The last recorded error was a configuration error detected in the TCDn_SOFF field. TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE].
5 DAE	Destination Address Error 0b - No destination address configuration error 1b - The last recorded error was a configuration error detected in the TCDn_DADDR field. TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE].
4 DOE	Destination Offset Error 0b - No destination offset configuration error 1b - The last recorded error was a configuration error detected in the TCDn_DOFF field. TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE].
3 NCE	NBYTES/CITER Configuration Error 0b - No NBYTES/CITER configuration error 1b - The last recorded error was NBYTES equal to zero or a CITER not equal to BITER error. The last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields. TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE], or TCDn_CITER[CITER] is equal to zero, or TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK]
2 SGE	Scatter/Gather Configuration Error 0b - No scatter/gather configuration error 1b - The last recorded error was a configuration error detected in the TCDn_DLASTSGA field. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. TCDn_DLASTSGA is not on a 32 byte boundary.
1 SBE	Source Bus Error 0b - No source bus error 1b - The last recorded error was a bus error on a source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - The last recorded error was a bus error on a destination write

6.6.1.4 Management Page Interrupt Request Status Register - Low (MP_INT_LOW)

Offset

Register	Offset
MP_INT_LOW	8h

Function

This register shows the current state of the interrupt service request for all eDMA channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	INT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 INT	Interrupt Request Status for channels 31 - 0 The INT register presents the interrupt request status for each eDMA channel. Depending on the appropriate bit setting in the transfer-control descriptors, the eDMA engine generates an interrupt on data transfer completion . The eDMA channel interrupt requests are routed to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit in the channel’s interrupt request register, CHn_INT, thus negating the interrupt request. • 0 - An interrupt request for the corresponding channel is not present. • 1 - An interrupt request for the corresponding channel is present.

6.6.1.5 Management Page Interrupt Request Status Register- High (MP_INT_HIGH)

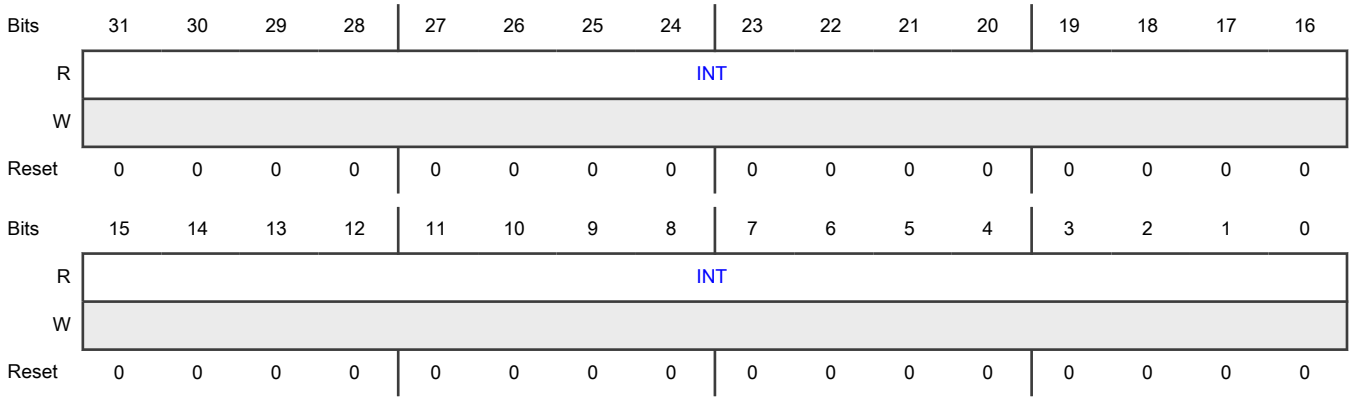
Offset

Register	Offset
MP_INT_HIGH	Ch

Function

This register shows the current state of the interrupt service request for all eDMA channels.

Diagram



Fields

Field	Function
31-0 INT	Interrupt Request Status for channels 63-32 The INT register presents the interrupt request status for each eDMA channel. Depending on the appropriate bit setting in the transfer-control descriptors, the eDMA engine generates an interrupt on data transfer completion . The eDMA channel interrupt requests are routed to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit in the channel’s interrupt request register, CHn_INT, thus negating the interrupt request. • 0 - An interrupt request for the corresponding channel is not present. • 1 - An interrupt request for the corresponding channel is present.

6.6.1.6 Management Page Hardware Request Status Register - Low (MP_HRS_LOW)

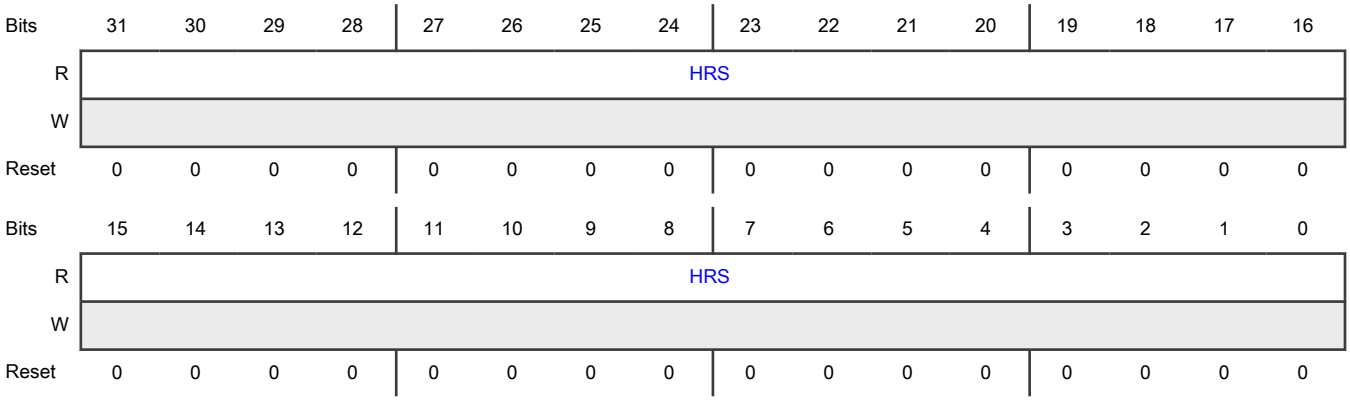
Offset

Register	Offset
MP_HRS_LOW	10h

Function

The hardware request status register (HRS) shows the current state of the hardware service request signaling as seen by the eDMA's arbitration logic.

Diagram



Fields

Field	Function
31-0 HRS	Hardware Request Status for channels 31 - 0 The HRS bit for its respective channel remains asserted for the period when a hardware request is present on the channel. After the request is completed and channel is free, the HRS bit is automatically cleared by hardware. 0000_0000_0000_0000_0000_0000_0000_0000b - A hardware service request for the channel is not present 0000_0000_0000_0000_0000_0000_0000_0001b - A hardware service request for channel 0 is present

6.6.1.7 Management Page Hardware Request Status Register - High (MP_HRS_HIGH)

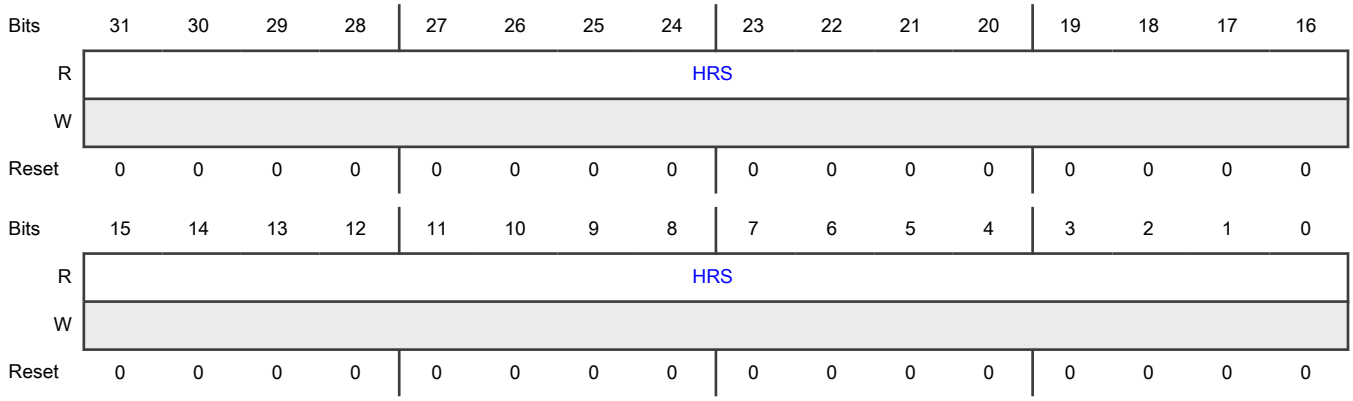
Offset

Register	Offset
MP_HRS_HIGH	14h

Function

The hardware request status register (HRS) shows the current state of the hardware service request signaling as seen by the eDMA's arbitration logic.

Diagram



Fields

Field	Function
31-0 HRS	Hardware Request Status for channels 63-32 The HRS bit for its respective channel remains asserted for the period when a hardware request is present on the channel. After the request is completed and channel is free, the HRS bit is automatically cleared by hardware. 0000_0000_0000_0000_0000_0000_0000_0000b - A hardware service request for the channel is not present 0000_0000_0000_0000_0000_0000_0000_0001b - A hardware service request for channel 0 is present

6.6.1.8 Channel Arbitration Group Register (CH0_GRPRI - CH63_GRPRI)

Offset

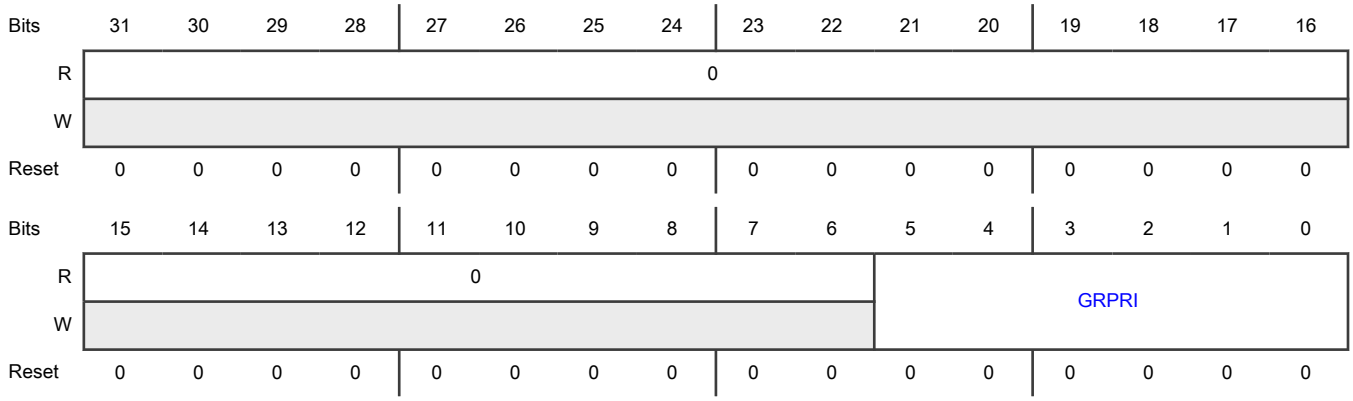
For n = 0 to 63:

Register	Offset
CHn_GRPRI	100h + (n × 4h)

Function

The contents of this register defines the arbitration group associated with each channel. Using fixed-priority group arbitration scheme, the arbitration group priorities are evaluated by numeric value from highest group number to lowest; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, etc. The range of the group priority values is limited to the values of 0 through 31. Within each arbitration group, the channel priority, CH *n* _PRI, assignment determines the highest priority channel.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 GRPRI	Arbitration group per channel. Fixed-priority arbitration group number.

6.6.2 DMA TCD register descriptions

6.6.2.1 TCD memory map

DMA4.TCD base address: 4201_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 1F_8000h	Channel Control and Status Register (CH0_CSR - CH63_CSR)	32	RW	0000_0000h
4h - 1F_8004h	Channel Error Status Register (CH0_ES - CH63_ES)	32	RW	0000_0000h
8h - 1F_8008h	Channel Interrupt Status Register (CH0_INT - CH63_INT)	32	RW	0000_0000h
Ch - 1F_800Ch	Channel System Bus Register (CH0_SBR - CH63_SBR)	32	RW	0000_8007h
10h - 1F_8010h	Channel Priority Register (CH0_PRI - CH63_PRI)	32	RW	0000_0000h
14h - 1F_8014h	Channel Multiplexor Configuration (CH0_MUX - CH63_MUX)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18h - 1F_8018h	Memory Attributes Register (CH0_MATTR - CH63_MATTR)	16	RW	0000h
20h - 1F_8020h	TCD Source Address Register (TCD0_SADDR - TCD63_SADDR)	32	RW	See section
24h - 1F_8024h	TCD Signed Source Address Offset Register (TCD0_SOFF - TCD63_SOFF)	16	RW	See section
26h - 1F_8026h	TCD Transfer Attributes Register (TCD0_ATTR - TCD63_ATTR)	16	RW	See section
28h - 1F_8028h	TCD Transfer Size without Minor Loop Offsets Register (TCD0_NBYTES_MLOFFNO - TCD63_NBYTES_MLOFFNO)	32	RW	See section
28h - 1F_8028h	TCD Transfer Size with Minor Loop Offsets Register (TCD0_NBYTES_MLOFFYES - TCD63_NBYTES_MLOFFYES)	32	RW	See section
2Ch - 1F_802Ch	TCD Last Source Address Adjustment / Store DADDR Address Register (TCD0_SLAST_SDA - TCD63_SLAST_SDA)	32	RW	See section
30h - 1F_8030h	TCD Destination Address Register (TCD0_DADDR - TCD63_DADDR)	32	RW	See section
34h - 1F_8034h	TCD Signed Destination Address Offset Register (TCD0_DOFF - TCD63_DOFF)	16	RW	See section
36h - 1F_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) Register (TCD0_CITER_ELINKNO - TCD63_CITER_ELINKNO)	16	RW	See section
36h - 1F_8036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) Register (TCD0_CITER_ELINKYES - TCD63_CITER_ELINKYES)	16	RW	See section
38h - 1F_8038h	TCD Last Destination Address Adjustment / Scatter Gather Address Register (TCD0_DLAST_SGA - TCD63_DLAST_SGA)	32	RW	See section
3Ch - 1F_803Ch	TCD Control and Status Register (TCD0_CSR - TCD63_CSR)	16	RW	See section
3Eh - 1F_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) Register (TCD0_BITER_ELINKNO - TCD63_BITER_ELINKNO)	16	RW	See section
3Eh - 1F_803Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) Register (TCD0_BITER_ELINKYES - TCD63_BITER_ELINKYES)	16	RW	See section

6.6.2.2 Channel Control and Status Register (CH0_CSR - CH63_CSR)

Offset

For n = 0 to 63:

Register	Offset
CHn_CSR	0h + (n × 8000h)

Function

The DMA hardware request input signal and the enable request bit (ERQ) must be asserted before a channel's hardware service request is accepted. The state of the eDMA enable request bit does not affect a channel service request made explicitly through software or channel linking. The state of the ERQ bit does not affect the channel's START bit.

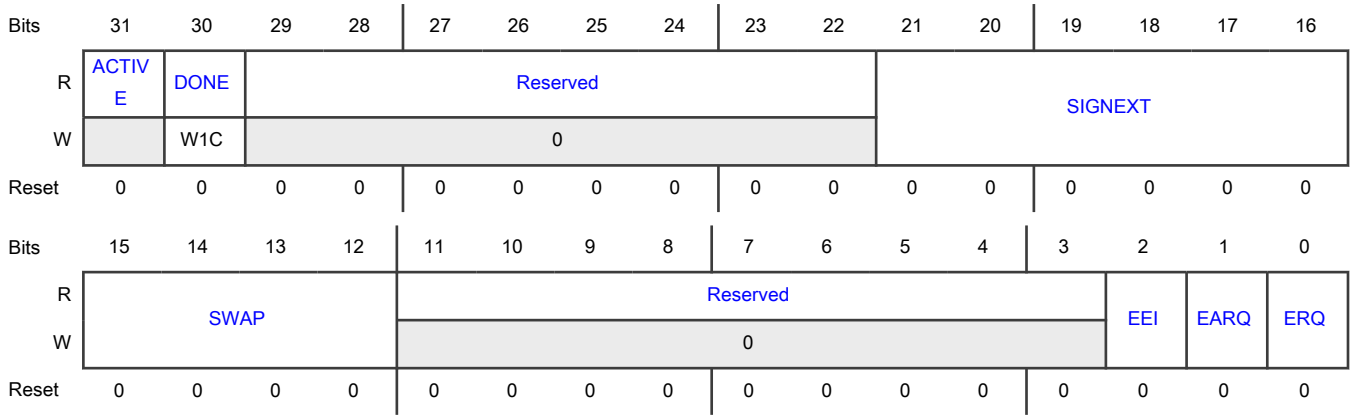
The enable asynchronous DMA request bit (EARQ) does not affect DMA operations. When set, this bit allows the hardware service request enable bit (ERQ) to propagate out of the DMA to the power controller. When cleared, this bit masks the hardware service request enable bit to the power controller.

The EEI register bit enables the error interrupt signal for the channel. The DMA error indicator and the error interrupt enable flag must be asserted before an error interrupt request for a given channel is asserted to the interrupt controller.

The DONE register bit indicates the channel has completed the major loop. If enabled, the eDMA will generate an interrupt request corresponding to this completed channel.

The ACTIVE register bit indicates the channel was selected by arbitration and is executing the prescribed transfers. Except for dynamic channel linking, the Transfer Control Descriptor should not be modified while a channel is active .

Diagram



Fields

Field	Function
31 ACTIVE	Channel Active This flag indicates the channel is currently in execution. It is set when channel service begins, and is cleared by the eDMA as the minor loop completes or when any error condition is detected.
30 DONE	Channel Done This flag indicates the eDMA has completed the major loop. The eDMA engine sets this bit as the CITER count reaches zero. The software clears it, or the hardware when the channel is activated. **NOTE** This bit must be cleared to write a 1 to the MAJORELINK bit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-22 —	Reserved
21-16 SIGNEXT	Sign Extension The data written on each transaction will be sign extended at the bit specified in this field. If 0, sign extension is disabled. 00_0000b - disabled 00_0001b - A non-zero value specifying the sign extend bit position
15-12 SWAP	Swap size Swap data elements within the specified transfer. Swap occurs with respect to the specified transfer size (SSIZE or DSIZE). 0000b - disabled 0001b - read with 8-bit swap 0010b - read with 16-bit swap 0011b - read with 32-bit swap 0100b-1000b - reserved 1001b - write with 8-bit swap 1010b - write with 16-bit swap 1011b - write with 32-bit swap 1100b-1111b - reserved
11-3 —	Reserved
2 EEI	Enable Error Interrupt 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
1 EARQ	Enable Asynchronous DMA Request 0b - Disable asynchronous DMA request for the channel. 1b - Enable asynchronous DMA request for the channel.
0 ERQ	Enable DMA Request Disable a channel's hardware service request at the source before clearing the channel's ERQ bit. 0b - The DMA hardware request signal for the corresponding channel is disabled. 1b - The DMA hardware request signal for the corresponding channel is enabled.

6.6.2.3 Channel Error Status Register (CH0_ES - CH63_ES)

Offset

For n = 0 to 63:

Register	Offset
CHn_ES	4h + (n × 8000h)

Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer-control descriptor,
- An error termination to a bus master read or write cycle

The ERR bit signals the presence of an error for the channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate bit in this register. The outputs of this register are enabled by the contents of the EEI, then logically summed across all channels to form an error interrupt request, which may be routed to the interrupt controller. During the execution of the interrupt-service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request) are not affected when an error is detected. The contents of this ERR register bit can also be polled because a non-zero value indicates the presence of a channel error regardless of the state of the EEI mask. The state of any given channel's error indicators is affected by writes to this register. On writes to the ERR bit, a one clears the channel's error status. A zero has no affect on the channel's current error status.

An unspecified error, where only the ERR bit is set, indicates a transfer was cancelled with an error . The Management Page Error Status register has full view of the error condition.

See Fault reporting and handling for more details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ERR	Reserved														
W	W1C	0														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								SAE	SOE	DAE	DOE	NCE	SGE	SBE	DBE
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ERR	Error In Channel 0b - An error in this channel has not occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - An error in this channel has occurred
30-8 —	Reserved
7 SAE	Source Address Error 0b - No source address configuration error. 1b - The last recorded error was a configuration error detected in the TCDn_SADDR field. TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE].
6 SOE	Source Offset Error 0b - No source offset configuration error 1b - The last recorded error was a configuration error detected in the TCDn_SOFF field. TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE].
5 DAE	Destination Address Error 0b - No destination address configuration error 1b - The last recorded error was a configuration error detected in the TCDn_DADDR field. TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE].
4 DOE	Destination Offset Error 0b - No destination offset configuration error 1b - The last recorded error was a configuration error detected in the TCDn_DOFF field. TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE].
3 NCE	NBYTES/CITER Configuration Error 0b - No NBYTES/CITER configuration error 1b - The last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields. TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE], or TCDn_CITER[CITER] is equal to zero, or TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK]
2 SGE	Scatter/Gather Configuration Error 0b - No scatter/gather configuration error 1b - The last recorded error was a configuration error detected in the TCDn_DLASTSGA field. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. TCDn_DLASTSGA is not on a 32 byte boundary.
1 SBE	Source Bus Error 0b - No source bus error 1b - The last recorded error was a bus error on a source read
0	Destination Bus Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
DBE	0b - No destination bus error 1b - The last recorded error was a bus error on a destination write

6.6.2.4 Channel Interrupt Status Register (CH0_INT - CH63_INT)

Offset

For n = 0 to 63:

Register	Offset
CHn_INT	8h + (n × 8000h)

Function

The INT register bit signals the presence of an interrupt request for the channel. Depending on the appropriate bit setting in the transfer-control descriptors, the eDMA engine generates an interrupt on data transfer completion . The outputs of this register are directly routed to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit, negating the interrupt request. On writes to INT, a 1 clears the channel's interrupt request. A zero has no affect on the channel's current interrupt status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															INT
W	0															W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 INT	Interrupt Request 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active

6.6.2.5 Channel System Bus Register (CH0_SBR - CH63_SBR)

Offset

For n = 0 to 63:

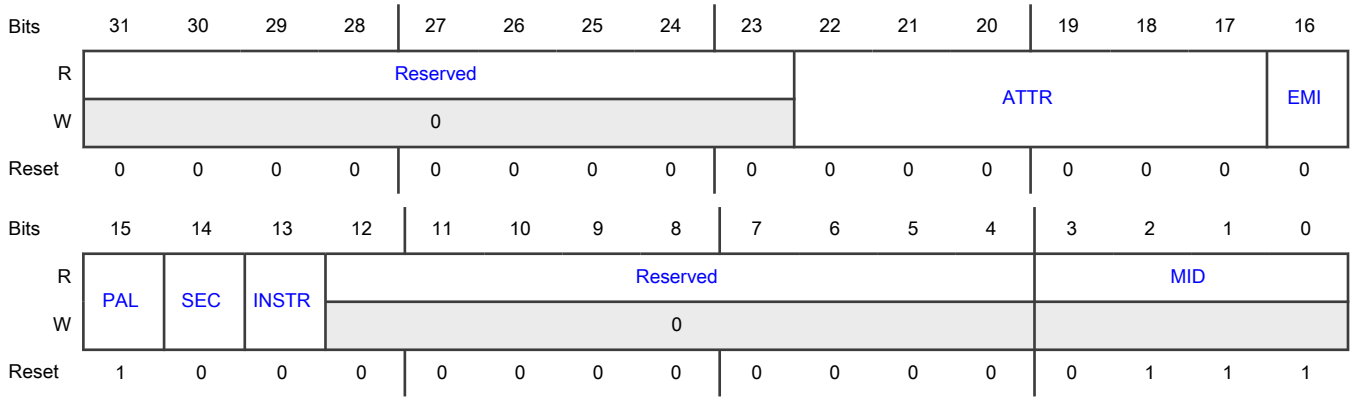
Register	Offset
CHn_SBR	Ch + (n × 8000h)

Function

The System Bus Register places identification and protection information on the system bus interface for the eDMA.

The ATTR register outputs the registers values onto the system bus interface for further decode by the security system.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-17 ATTR	Attribute Output DMA's system bus attribute output value.
16 EMI	Enable Master ID replication The eDMA master ID replication register allows the eDMA to use the same protection level and system bus ID of the master programming the eDMA's TCD. When enabled, the eDMA uses the master ID and protection level stored in the SBR registers instead of the eDMA's default values. When a master, for example a core, programs a TCD, its master ID is captured when the TCD *n* _CSR control is written. The protection and security levels are captured when the CH *n* _SBR security attributes are written. A scatter/gather operation does not affect the CH *n* _SBR registers. The EMI bit may be written only if Management Page CSR[GMRC] = 1, Global Master ID Replication Control is enabled; otherwise, the EMI will be forced to zero.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If master ID replication is disabled, the nonsecure, privileged protection level for DMA transfers is used. 0b - Master ID replication is disabled 1b - Master ID replication is enabled
15 PAL	Privileged Access Level DMA's protection level on the system bus when the channel is active. **NOTE** The value written into this register cannot exceed the security and privilege level of the core or other master writing the channel's system bus register; CH *n* _SBR. The order of precedence is SecurePriv>SecureUser>NonsecurePriv>NonsecureUser 0b - User protection level for DMA transfers 1b - Privileged protection level for DMA transfers
14 SEC	Security Level DMA's security level on the system bus when the channel is active. **NOTE** The value written into this register cannot exceed the security and privilege level of the core or other master writing the channel's system bus register; CH *n* _SBR. The order of precedence is SecurePriv>SecureUser>NonsecurePriv>NonsecureUser 0b - Nonsecure protection level for DMA transfers 1b - Secure protection level for DMA transfers
13 INSTR	Instruction/Data Access DMA's transactions type on the system bus when the channel is active. 0b - Data access for DMA transfers 1b - Instruction access for DMA transfers
12-4 —	Reserved
3-0 MID	Master ID DMA's master ID on the system bus when the channel is active. **NOTE** When master ID replication is enabled, the ID captured in this register reflects the master ID of the core or other master writing the channel's control attributes, lower byte of TCD *n* _CSR.

6.6.2.6 Channel Priority Register (CH0_PRI - CH63_PRI)

Offset

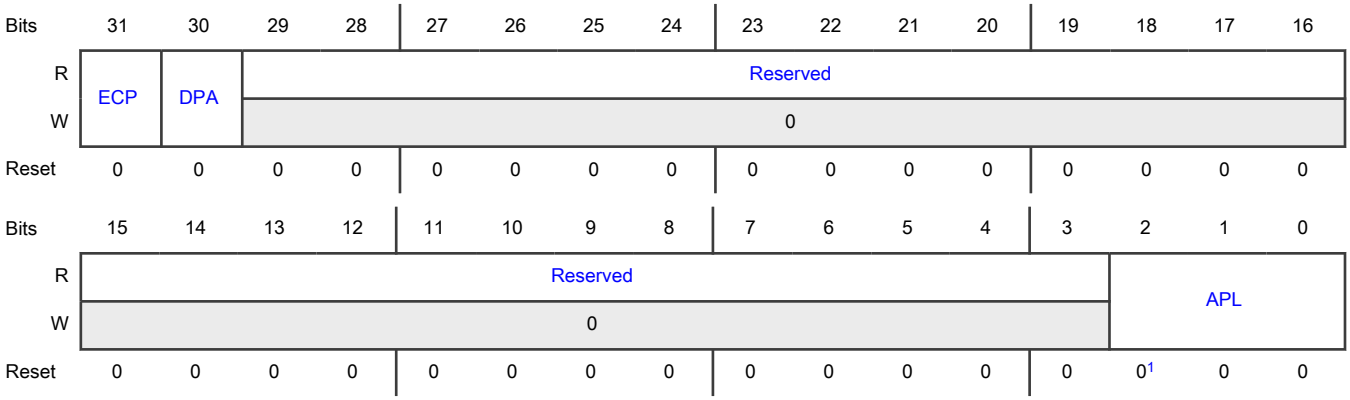
For n = 0 to 63:

Register	Offset
CHn_PRI	10h + (n × 8000h)

Function

The contents of these registers define unique priorities associated with each channel within the same channel group. Channel grouping is programmed via the group priority registers MP_GRPRI. The channel priorities within a group are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, etc. Software must program the channel priorities with unique values; otherwise, channel numbers with the same, non-zero value, will be selected based on channel number with the higher channel number having higher priority. If more than one channel in a group has an Arbitration Priority Level value of zero, then the arbitration mode bit MP_CSR[ERCA] is used to determine the arbitration scheme for all channels with APL=0 within a group. When round-robin channel arbitration is enabled (MP_CSR[ERCA] = 1), all channels with APL=0 within a group will use a round-robin arbitration scheme which will rotate among these channels requesting service without regard to priority. Round-robin provides a fairness mechanism within an arbitration group. When fixed-priority channel arbitration is enabled (MP_CSR[ERCA] = 0), channels with APL=0 will be selected based on channel number with the higher channel number having higher priority.

Diagram



1. See bit field description.

Fields

Field	Function
31 ECP	Enable Channel Preemption. 0b - The channel cannot be suspended by a higher priority channel's service request. 1b - The channel can be temporarily suspended by the service request of a higher priority channel.
30 DPA	Disable Preempt Ability.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - The channel can suspend a lower priority channel. 1b - The channel cannot suspend any other channel, regardless of channel priority.
29-3 —	Reserved
2-0 APL	Arbitration Priority Level Channel priority level for arbitration within the assigned arbitration group

6.6.2.7 Channel Multiplexor Configuration (CH0_MUX - CH63_MUX)

Offset

For n = 0 to 63:

Register	Offset
CHn_MUX	14h + (n × 8000h)

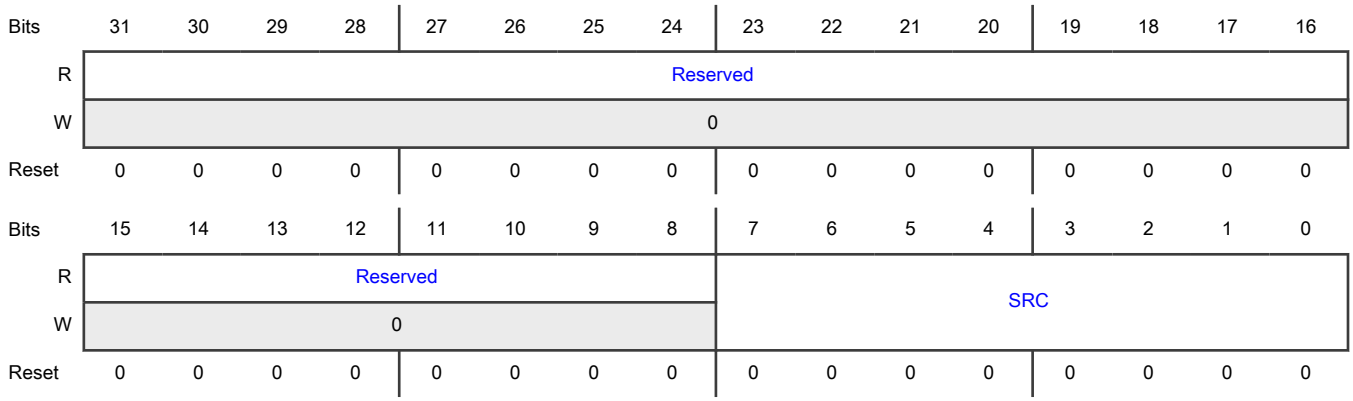
Function

Each of the DMA channels can be independently associated with various peripherals in the system. The Channel Multiplexor Configuration register selects the peripheral assigned to each channel. Service requests from the peripheral should be disabled when configuring a channel to a peripheral source.

Each channel must have a unique value when selecting a peripheral slot in the channel mux configuration. The only value that may overlap is source 0. If there is an attempt to write a mux configuration value that is already consumed by another channel, a mux configuration of 0 (SRC = 0) will be written.

All channels will default to source 0. When a particular peripheral is needed, the channel's mux configuration is set to that source number. When the peripheral is no longer needed, the mux configuration for that channel should be written to 0, thus releasing the resource.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SRC	Service Request Source Hardware service request source for the channel. **NOTE** With the exception of 0, attempts to write a value already in use will be forced to 0.

6.6.2.8 Memory Attributes Register (CH0_MATTR - CH63_MATTR)

Offset

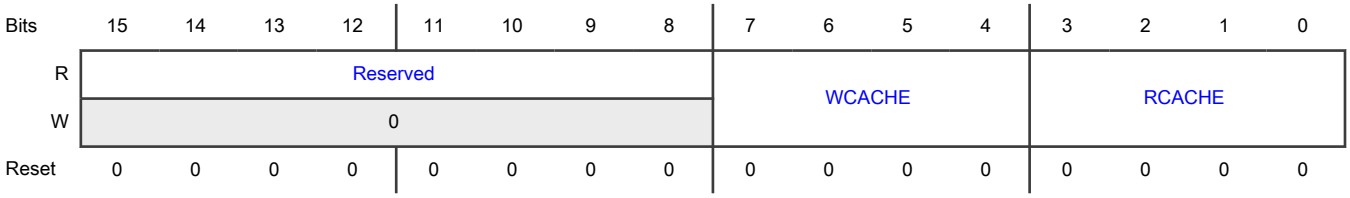
For n = 0 to 63:

Register	Offset
CHn_MATTR	18h + (n × 8000h)

Function

The Memory Attributes Register defines a set of transaction attributes for memory and peripherals. The cache attributes specify how a transaction progresses through the system and how they are handled.

Diagram



Fields

Field	Function
15-8 —	Reserved
7-4 WCACHE	Write Cache Attributes This field defines the attributes associated with a write transaction. • [7] 0b0 no write allocate, 0b1 write-allocate • [6] 0b0 no read allocate, 0b1 read-allocate

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• [5] 0b0 non-cacheable, 0b1 cacheable • [4] 0b0 non-bufferable, 0b1 bufferable
3-0 RCACHE	Read Cache Attributes This field defines the attributes associated with a read transaction. • [3] 0b0 no write allocate, 0b1 write-allocate • [2] 0b0 no read allocate, 0b1 read-allocate • [1] 0b0 non-cacheable, 0b1 cacheable • [0] 0b0 non-bufferable, 0b1 bufferable

6.6.2.9 TCD Source Address Register (TCD0_SADDR - TCD63_SADDR)

Offset

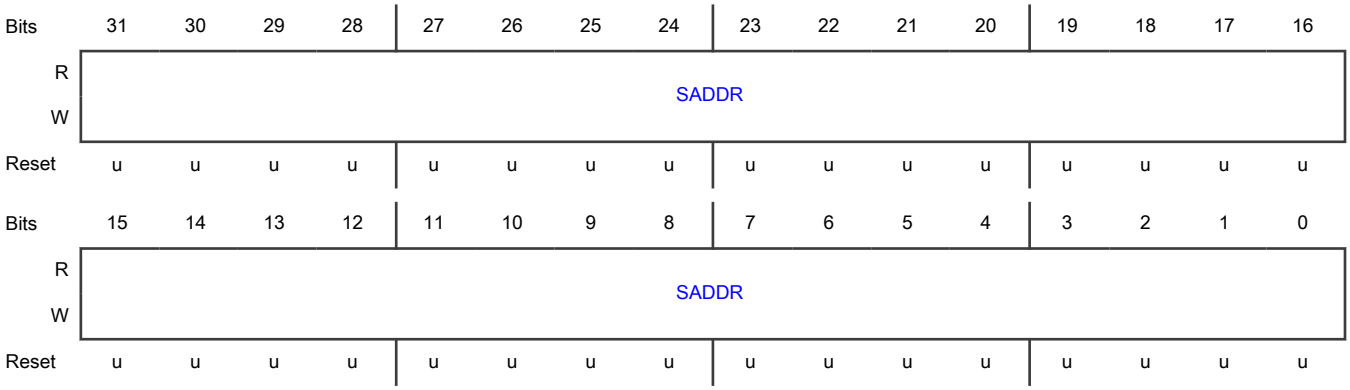
For n = 0 to 63:

Register	Offset
TCDn_SADDR	20h + (n × 8000h)

Function

This register contains the address for the read transactions.

Diagram



Fields

Field	Function
31-0 SADDR	Source Address Memory address pointing to the source data.

6.6.2.10 TCD Signed Source Address Offset Register (TCD0_SOFF - TCD63_SOFF)

Offset

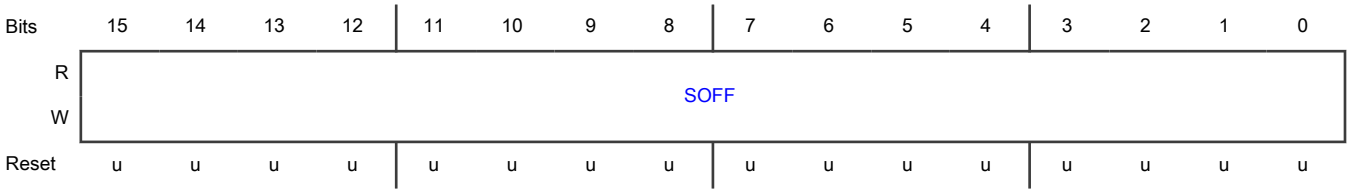
For n = 0 to 63:

Register	Offset
TCDn_SOFF	24h + (n × 8000h)

Function

This register contains the sign-extended value added to Source Address register after each read transaction.

Diagram



Fields

Field	Function
15-0 SOFF	Source address signed offset Sign-extended offset applied to the current source address to form the next-state value as each source read is completed.

6.6.2.11 TCD Transfer Attributes Register (TCD0_ATTR - TCD63_ATTR)

Offset

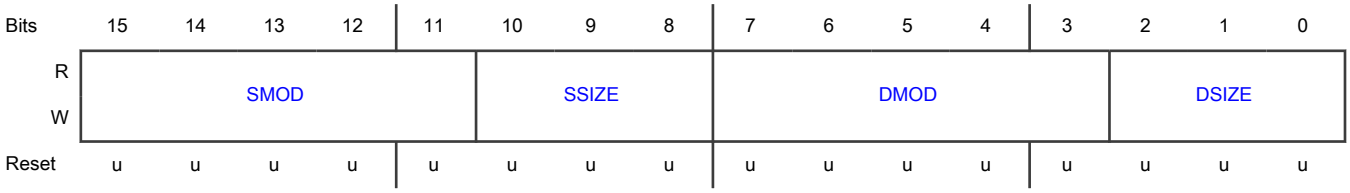
For n = 0 to 63:

Register	Offset
TCDn_ATTR	26h + (n × 8000h)

Function

This register contains size and option modulo addressing information for source and destination addresses.

Diagram



Fields

Field	Function
15-11 SMOD	Source address modulo This field defines a specific address range specified to be the value after SADDR + SOFF calculation is performed on the original register value. Setting this field provides the ability to implement a circular data queue easily. For data queues requiring power-of-2 size bytes, the queue should start at a 0-modulo-size address and the SMOD field should be set to the appropriate value for the queue, freezing the desired number of upper address bits. The value programmed into this field specifies the number of lower address bits allowed to change. For a circular queue application, the SOFF is typically set to the transfer size to implement post-increment addressing with the SMOD function constraining the addresses to a 0-modulo-size range. 0_0000b - Source address modulo feature is disabled 0_0001b - Source address modulo feature is enabled for any non-zero value [1-31]
10-8 SSIZE	Source data transfer size This field specifies the size of each read transaction on the system bus. **NOTE** NBYTES must be an integer multiple of the source transfer size. 000b - 8-bit 001b - 16-bit 010b - 32-bit 011b - 64-bit 100b - 16-byte 101b - 32-byte 110b - 64-byte 111b - 128-byte
7-3 DMOD	Destination address modulo See the SMOD definition
2-0 DSIZE	Destination data transfer size This field specifies the size of each write transaction on the system bus. **NOTE** NBYTES must be an integer multiple of the destination transfer size.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - 8-bit 001b - 16-bit 010b - 32-bit 011b - 64-bit 100b - 16-byte 101b - 32-byte 110b - 64-byte 111b - 128-byte

6.6.2.12 TCD Transfer Size without Minor Loop Offsets Register (TCD0_NBYTES_MLOFFNO - TCD63_NBYTES_MLOFFNO)

Offset

For n = 0 to 63:

Register	Offset
TCDn_NBYTES_MLOFFNO	28h + (n × 8000h)

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offsets are address offset values added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) upon minor loop completion. Minor loop completion is when the channel has finished the service request and has transferred NBYTES. When minor loop offsets are enabled, the minor loop offset value (MLOFF) is added to the final source address (TCDn_SADDR), to the final destination address (TCDn_DADDR), or to both prior to the addresses being written back to the TCD. If the major loop is complete, the minor loop offset is ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When minor loop mapping is enabled (SMLOE or DMLOE is 1), TCDa word2 is redefined. A portion of TCDn word2 is used to specify multiple fields: a source enable bit (SMLOE) to specify the minor loop offset should be applied to the source address (TCDn_SADDR) upon minor loop completion, a destination enable bit (DMLOE) to specify the minor loop offset should be applied to the destination address (TCDn_DADDR) upon minor loop completion, and the sign extended minor loop offset value (MLOFF). The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. When both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCD_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCD word 2 is defined as follows if:

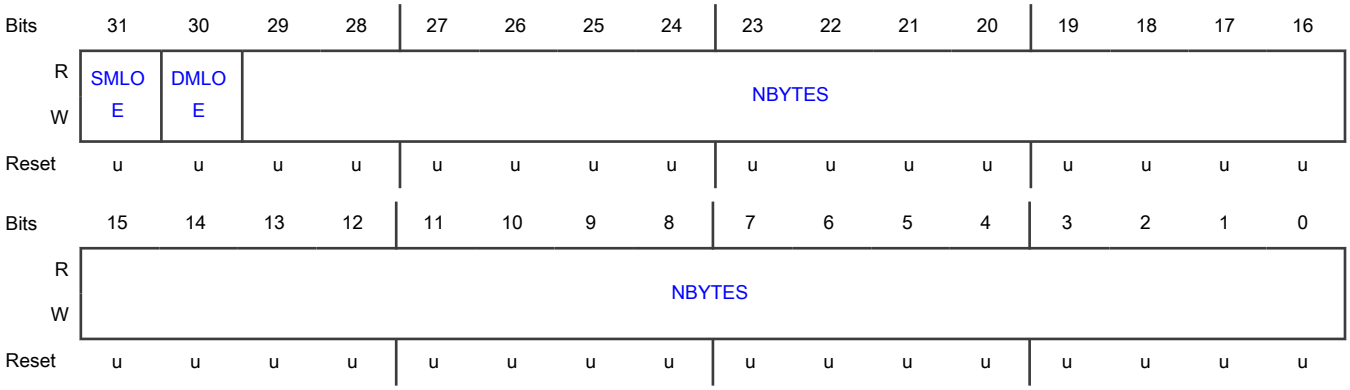
- SMLOE = 0 and DMLOE = 0

If either SMLOE or DMLOE is set, then refer to the TCD_NBYTES_MLOFFYES register description.

NOTE

NBYTES must be an integer multiple of the source transfer size and an integer multiple of the destination transfer size.

Diagram



Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - The minor loop offset is not applied to the SADDR 1b - The minor loop offset is applied to the SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - The minor loop offset is not applied to the DADDR 1b - The minor loop offset is applied to the DADDR
29-0 NBYTES	Number of Bytes to transfer per service request Number of bytes to be transferred for each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the byte transfer count has transferred. This is normally an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the byte count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major loop iteration count (CITER) is decremented by one and written back to the TCD memory. If the major iteration count is completed, additional processing is performed.

6.6.2.13 TCD Transfer Size with Minor Loop Offsets Register (TCD0_NBYTES_MLOFFYES - TCD63_NBYTES_MLOFFYES)

Offset

For n = 0 to 63:

Register	Offset
TCDn_NBYTES_MLOFFYES	28h + (n × 8000h)

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offsets is an address offset value added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) upon minor loop completion. Minor loop completion is when the channel has finished the service request and has transferred NBYTES. Minor loop offsets are enabled by setting either the source enable bit (SMLOE) or the destination enable bit (DMLOE). The source enable bit (SMLOE) specifies the minor loop offset value (MLOFF) should be applied to the source address (TCDn_SADDR) upon minor loop completion. The destination enable bit (DMLOE) specifies the minor loop offset (MLOFF) should be applied to the destination address (TCDn_DADDR) upon minor loop completion. If the major loop is complete, the minor loop offsets are ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When the minor loop offset overlay is enabled (either SMLOE or DMLOE is 1), TCDn_NBYTES_* is redefined. A portion of TCDn_NBYTES_* is used to specify the sign extended minor loop offset value (MLOFF). The same offset value (MLOFF) is used for both source and destination minor loop offsets. When enabled, the minor loop offset must be aligned to the transfer size of the source or destination it is associated with. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. When both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCD_NBYTES_MLOFFNO), defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCDn_NBYTES_ is defined as follows if:

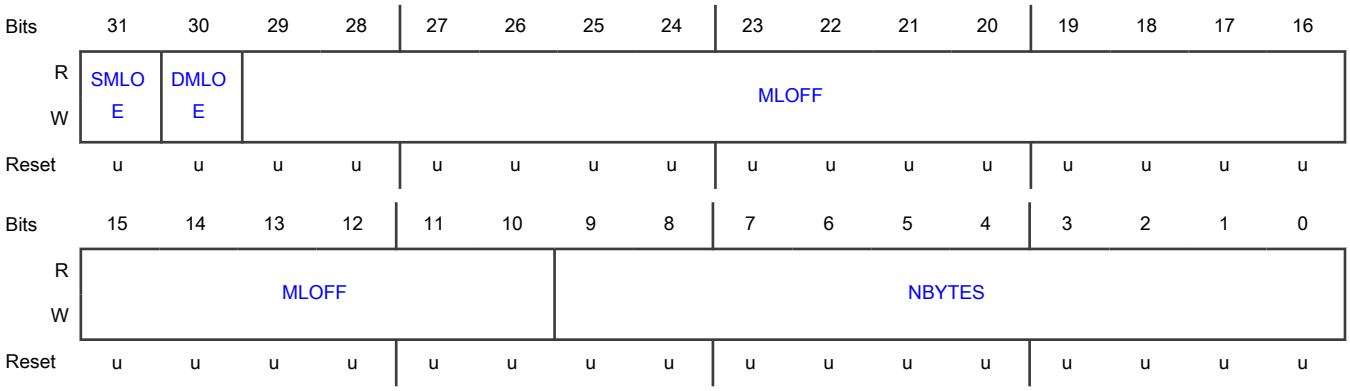
- Either minor loop offset is enabled (SMLOE or DMLOE = 1), then refer to the TCD_NBYTES_MLOFFYES register description.

If SMLOE and DMLOE are cleared, then refer to the TCD_NBYTES_MLOFFNO register description.

NOTE

NBYTES must be an integer multiple of the source transfer size and an integer multiple of the destination transfer size.

Diagram



Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - The minor loop offset is not applied to the SADDR 1b - The minor loop offset is applied to the SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - The minor loop offset is not applied to the DADDR 1b - The minor loop offset is applied to the DADDR
29-10 MLOFF	Minor Loop Offset If SMLOE or DMLOE is set, this field represents a sign-extended offset applied to the source or destination address to form the next-state value after the minor loop completes.
9-0 NBYTES	Number of Bytes to transfer per service request Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

6.6.2.14 TCD Last Source Address Adjustment / Store DADDR Address Register (TCD0_SLAST_SDA - TCD63_SLAST_SDA)

Offset

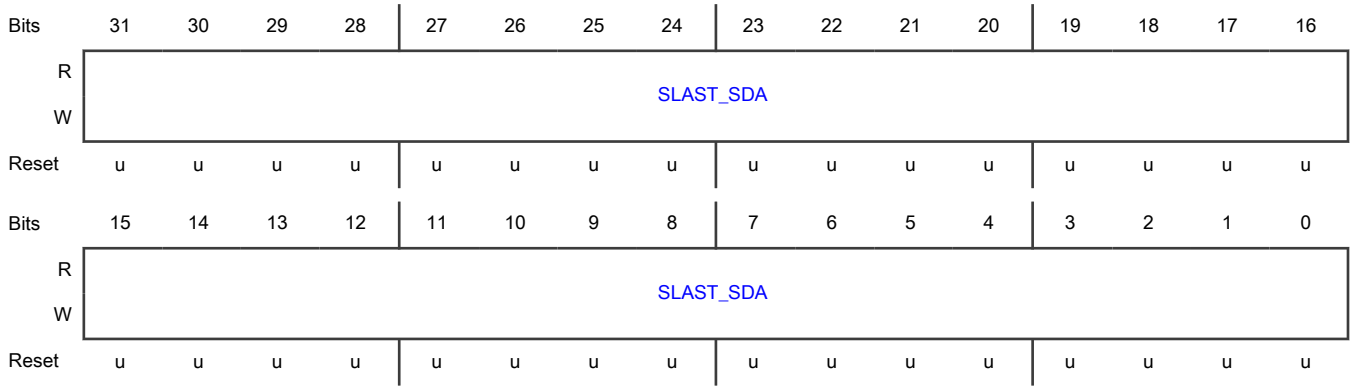
For n = 0 to 63:

Register	Offset
TCDn_SLAST_SDA	2Ch + (n × 8000h)

Function

This register contains the value added to the source address when the major loop is complete. When the store destination address option is enabled, this field provides a pointer to memory for storing the final destination address.

Diagram



Fields

Field	Function
31-0 SLAST_SDA	Last Source Address Adjustment / Store DADDR Address Source last address adjustment or the system memory address for destination address (DADDR) storage. If (TCDn_CSR[ESDA] = 0), then: • Adjustment value is added to the source address at the completion of the major iteration count. This value can be used to restore the source address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final source address adjustment. Otherwise: • This address points to the 32-bit-aligned memory location where the destination address (DADDR) is to be stored in system memory. By saving the final destination address in system memory via the ESDA feature, you are able to compute the size of a variable destination data buffer by simply subtracting the beginning DADDR from the final, saved DADDR. This feature is used together with the scatter/gather operation to prevent the loss of the final DADDR, which is overwritten during the scatter/gather operation. The "Store Destination Address" (SDA) value must be a 32-bit-aligned location because the eDMA forces the lower two address bits of the SLAST_SDA field to zero when ESDA is enabled. The module performs this write operation when the major loop is done; that is, when the major iteration count (CITER) decrements to zero.

6.6.2.15 TCD Destination Address Register (TCD0_DADDR - TCD63_DADDR)

Offset

For n = 0 to 63:

Register	Offset
TCDn_DADDR	30h + (n × 8000h)

Function

This register contains the address for the write transactions.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DADDR															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DADDR															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 DADDR	Destination Address Memory address pointing to the destination data.

6.6.2.16 TCD Signed Destination Address Offset Register (TCD0_DOFF - TCD63_DOFF)

Offset

For n = 0 to 63:

Register	Offset
TCDn_DOFF	34h + (n × 8000h)

Function

This register contains the sign-extended value added to Destination Address register after each write transaction.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DOFF															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
15-0 DOFF	Destination Address Signed Offset Sign-extended offset applied to the current destination address to form the next-state value as each destination write is completed.

6.6.2.17 TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) Register (TCD0_CITER_ELINKNO - TCD63_CITER_ELINKNO)

Offset

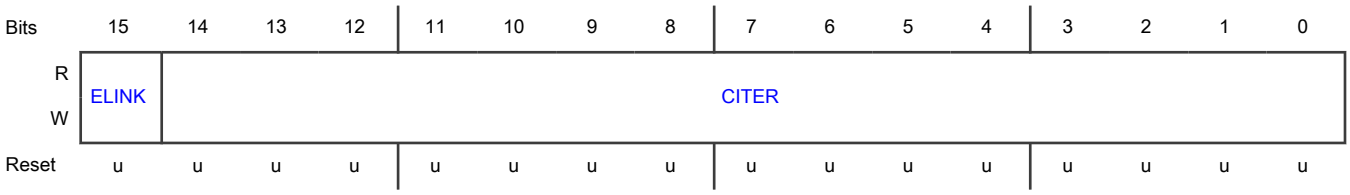
For n = 0 to 63:

Register	Offset
TCDn_CITER_ELINKNO	36h + (n × 8000h)

Function

If TCDn_CITER[ELINK] is cleared, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported. 0b - The channel-to-channel linking is disabled 1b - The channel-to-channel linking is enabled
14-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and is written back to the TCD memory. After the major iteration count is exhausted, the channel performs a number of operations, for example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field for the proper operation of the optional TCDn_CSR[INTHALF] interrupt. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

6.6.2.18 TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) Register (TCD0_CITER_ELINKYES - TCD63_CITER_ELINKYES)

Offset

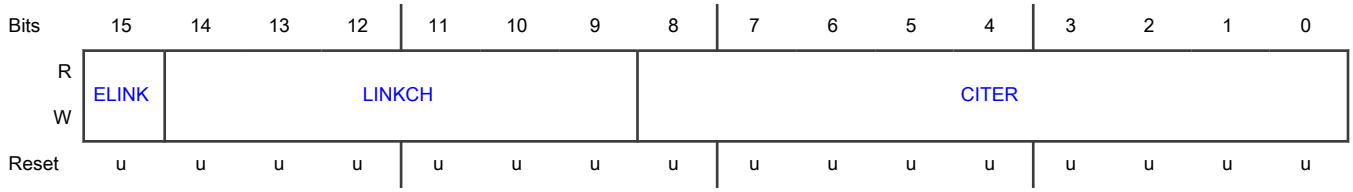
For n = 0 to 63:

Register	Offset
TCDn_CITER_ELINKYES	36h + (n × 8000h)

Function

If TCDn_CITER[ELINK] is set, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. When enabled, an internal mechanism sets the TCDn_CSR[START] bit of the specified channel (LINKCH) upon minor loop completion. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported. 0b - The channel-to-channel linking is disabled 1b - The channel-to-channel linking is enabled
14-9 LINKCH	Minor Loop Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request to the channel defined by this field by setting that channel's TCDn_CSR[START] bit.
8-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and written back to the TCD memory. After the major iteration count is exhausted, the channel performs a number of operations, for example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field for the proper operation of the optional TCDn_CSR[INTHALF] interrupt. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

6.6.2.19 TCD Last Destination Address Adjustment / Scatter Gather Address Register (TCD0_DLAST_SGA - TCD63_DLAST_SGA)

Offset

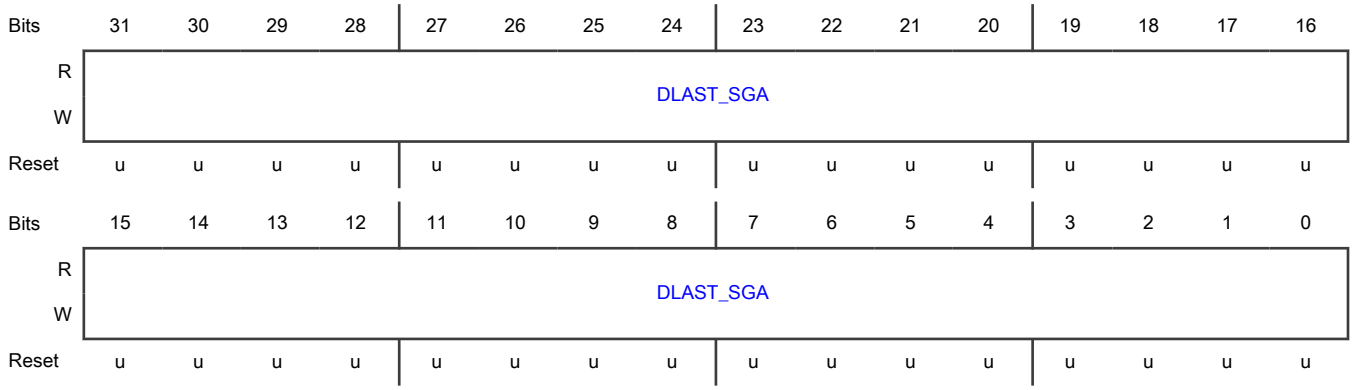
For n = 0 to 63:

Register	Offset
TCDn_DLAST_SGA	38h + (n × 8000h)

Function

This register contains the value added to the destination address when the major loop is complete. When the Scatter/Gather option is enabled, this field provides a pointer to memory for fetching a transfer control descriptor to reprogram the channel.

Diagram



Fields

Field	Function
31-0 DLAST_SGA	Final Destination Address Adjustment / Scatter Gather Address Destination final address adjustment or the memory address for the next transfer control descriptor to be loaded into this channel (Scatter/Gather). If (TCDn_CSR[ESG] = 0), then: • Adjustment value is added to the destination address at the completion of the major iteration count. This value can be used to restore the destination address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final destination address adjustment. Otherwise: • This address points to the beginning of a 0-modulo-32-byte region containing the next transfer control descriptor to be loaded into this channel. This channel reload is performed as the major iteration count completes. The scatter/gather address must be 0-modulo-32-byte, otherwise a configuration error is reported.

6.6.2.20 TCD Control and Status Register (TCD0_CSR - TCD63_CSR)

Offset

For n = 0 to 63:

Register	Offset
TCDn_CSR	3Ch + (n × 8000h)

Function

This register is used to enable optional features.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TMC		MAJORLINKCH						ESDA	EEOP	MAJO REL...	ESG	DREQ	INTHA LF	INTMA JOR	STAR T
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	0

Fields

Field	Function
15-14 TMC	Transfer Mode Control Transfer Mode Control specifies the direction the DMA will transfer data. For normal data movement operation, Read/Write mode reads from the source address and writes to the destination address. For Read Only mode, data is read from the source address and discarded. Read Only mode may be useful for calculating CRC values on a block of memory. For Write Only mode, the source address is read once and all subsequent writes to the destination uses the data from the first and only read. Write only mode may be used to initialize a block of memory to a specific value. 00b - Read/Write 01b - Read Only 10b - Write Only 11b - Reserved
13-8 MAJORLINKCH	Major loop link channel number If (MAJORELINK = 0) then: • No channel-to-channel linking, or chaining, is performed after the major loop counter is exhausted. Otherwise: • After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit.
7 ESDA	Enable store destination address As the channel completes the major loop by either the current iteration counter (CITER) decrementing to zero or by receiving an enabled end-of-packet signal,, this bit enables writing the destination address (DADDR) to the address pointed to by the SLAST_SDA field. When the SDA bit is set, the SLAST_SDA contains the write pointer instead of the final source address offset. The value written to system memory pointed by SLAST_SDA is different for two cases: • For a normal successful transfer: the value is the last DADDR value prior to the DLAST_SGA offset being applied or overwritten by an enabled scatter/gather operation. **NOTE** Software can also compute this value using the TCD information. • For a transfer that was ended early by EOP: the value written to the memory is the last destination address that was written with the last good read data before EOP was issued. 0b - The store destination address to system memory operation is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - The store destination address to system memory operation is enabled.
6 EEOP	Enable end-of-packet processing When enabled by the EEOP bit, a end-of-packet hardware input signal directs the eDMA to discontinue executing the active channel and to treat the shutdown as major loop completed. If the EEOP bit is set, the end-of-packet signal from supported peripherals will be accepted. If the EEOP bit is cleared, the end-of-packet input is ignored. With an end-of-packet retirement, the current TCD destination address (or ESDA saved destination address) minus the software-saved initial (DADDR) reflects the total amount of data transferred. For the aforementioned calculation, the destination offset (DOFF) must match the natural boundary of the destination transfer size DSIZE. For example, if DSIZE = 32-bit, DOFF = 4. 0b - The end-of-packet operation is disabled. 1b - The end-of-packet hardware input signal is enabled.
5 MAJORELINK	Enable channel-to-channel linking on major loop complete As the channel completes the major loop, this flag enables the linking to another channel, defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. **NOTE** To support the dynamic linking coherency model, this field is forced to zero when written to while the TCDn_CSR[DONE] bit is set. This field and the MAJORLINKCH must be written at the same time when attempting a dynamic channel link. 0b - The channel-to-channel linking is disabled. 1b - The channel-to-channel linking is enabled.
4 ESG	Enable Scatter/Gather processing As the channel completes the major loop, this flag enables scatter/gather processing in the current channel. If enabled, the eDMA engine uses DLASTSGA as a memory pointer to a 0-modulo-32 address containing a 32-byte data structure loaded as the transfer control descriptor into the local memory. **NOTE** Any changes made to this field while the channel is executing will not be recognized until the next time the channel is activated. 0b - The current channel's TCD is normal format. 1b - The current channel's TCD specifies a scatter gather format. The DLASTSGA field provides a memory pointer to the next TCD to be loaded into this channel after the major loop completes its execution.
3 DREQ	Disable request If this flag is set, the eDMA hardware automatically clears the corresponding ERQ bit when the current major iteration count reaches zero. 0b - No operation. The channel's ERQ {H,L} bit is not affected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clear the ERQ bit upon major loop completion, thus disabling hardware service requests. The channel's ERQ {H,L} bit is cleared when the major loop is complete.
2 INTHALF	Enable an interrupt when major counter is half complete. If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT register when the current major iteration count reaches the halfway point. Specifically, the comparison performed by the eDMA engine is (CITER == (BITER/2)). This halfway point interrupt request is provided to support double-buffered, also known as ping-pong, schemes or other types of data movement where the processor needs an early indication of the transfer's progress. **NOTE** If BITER = 1, do not use INTHALF. Use INTMAJOR instead. 0b - The half-point interrupt is disabled. 1b - The half-point interrupt is enabled.
1 INTMAJOR	Enable an interrupt when major iteration count completes. If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT when the current major iteration count (CITER) reaches zero. 0b - The end-of-major loop interrupt is disabled. 1b - The end-of-major loop interrupt is enabled.
0 START	Channel Start If this flag is set, the channel is requesting service. The eDMA hardware automatically clears this flag after the channel begins execution. 0b - The channel is not explicitly started. 1b - The channel is explicitly started via a software initiated service request.

6.6.2.21 TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) Register (TCD0_BITER_ELINKNO - TCD63_BITER_ELINKNO)

Offset

For n = 0 to 63:

Register	Offset
TCDn_BITER_ELINKNO	3Eh + (n × 8000h)

Function

If the TCDn_BITER[ELINK] bit is cleared, the TCDn_BITER register is defined as follows.

Diagram

Fields

Field	Function
15 ELINK	Enables channel-to-channel linking on minor loop complete As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. 0b - The channel-to-channel linking is disabled 1b - The channel-to-channel linking is enabled
14-0 BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field should be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field for the proper operation of the optional TCDn_CSR[INTHALF] interrupt. As the major iteration count is exhausted, the contents of this field is reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

6.6.2.22 TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) Register (TCD0_BITER_ELINKYES - TCD63_BITER_ELINKYES)

Offset

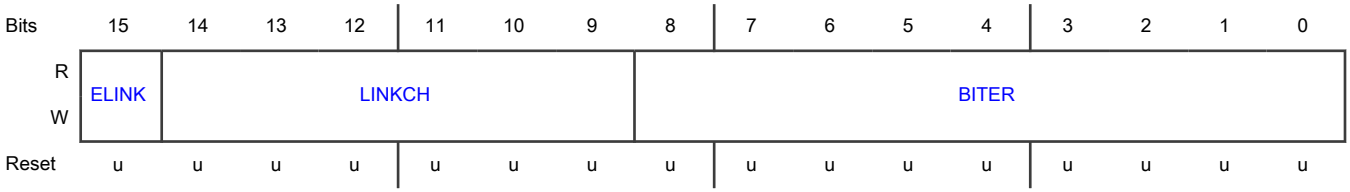
For n = 0 to 63:

Register	Offset
TCDn_BITER_ELINKYES	3Eh + (n × 8000h)

Function

If the TCDn_BITER[ELINK] bit is set, the TCDn_BITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enables channel-to-channel linking on minor loop complete As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking disables, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. 0b - The channel-to-channel linking is disabled 1b - The channel-to-channel linking is enabled
14-9 LINKCH	Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit. **NOTE** As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.
8-0 BITER	Starting major iteration count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field for the proper operation of the optional TCDn_CSR[INTHALF] interrupt. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

Chapter 7
Security Overview

7.1 Security features

Protecting customers' assets is at the core of the chip's security architecture. Valuable assets comprise intellectual property, confidential data, device integrity, service availability, as well as company's reputation. By controlling access to the device as well as providing trusted computing environment, the chip offers various security features and tools that allow customers to protect their assets. Some of the basic features include:

- **Secure boot** ensuring that only authentic entity is allowed to access the device or execute code on it.
- **Secure debug** allowing customers to securely develop and debug software while the device is at manufacturing or deployed in the field.
- **Secure life cycle management** guaranteeing that security mechanisms are enforced, and device reaches its highest security level after it has been deployed or returned to NXP for failure analysis.
- **Hardware Security Module (HSM)** and cryptographic services ensuring that cryptographic keys remain hidden from the application in all circumstances.

This chapter provides an overview of security-relevant SoC components, explaining the main purpose and features of each of them.

- **EdgeLock secure enclave (ELE)** acts as a Root of Trust for platform security. It manages the SoC life cycle, enforces secure boot, controls debug access, and provides full key management and key isolation from the rest of SoC.
- **Trusted Resource Domain Controller (TRDC)** is essential in enabling isolation between various processes running within the SoC. It is used to define and enforce separation between security critical and other processes by introducing the notion of domain IDs, as well as enforcing the trusted execution environment defined by ARM TrustZone.
- **On The Fly AES Decryption (OTFAD)** protects the confidentiality of code stored in external memory using AES in CTR mode. The key used for decryption is securely handled by ELE and delivered to OTFAD using a dedicated private bus, ensuring that the key is not observable by any application running on the SoC. Two instances of OTFAD are available, interfacing FlexSPI1 and FlexSPI2. OTFAD supports up to four independent memory contexts, each having a unique 128-bit decryption key.
- **Inline Encryption Engine (IEE)** uses AES-128 in CTR mode for decryption only and AES-128 XTS mode for encryption and decryption of code and data stored in external memory. Decryption only is supported through FlexSPI interface and is meant to be used for code stored in FLASH (similar to OTFAD), while the content stored in SDRAM can be both encrypted and decrypted using IEE in AES-128 XTS mode. IEE supports up to eight independent memory contexts, each having a unique 128-bit key.
- **Battery Backed Secure Memory (BBSM)** is used as a non-volatile storage of security parameters including general purpose registers, run-time counter with a wake-up alarm, monotonic counters, as well as tamper detection inputs.

Building on security-relevant SoC components a number of security features emerge which, altogether, serve to protect the valuable assets.

7.1.1 Edgelock Enclave (ELE)

The Edgelock Enclave (ELE) is asynchronous to the rest of the AON Domain, so that the ELE can be in complete control of its clock root, regardless of what clock is used for other IP in the AON Domain.

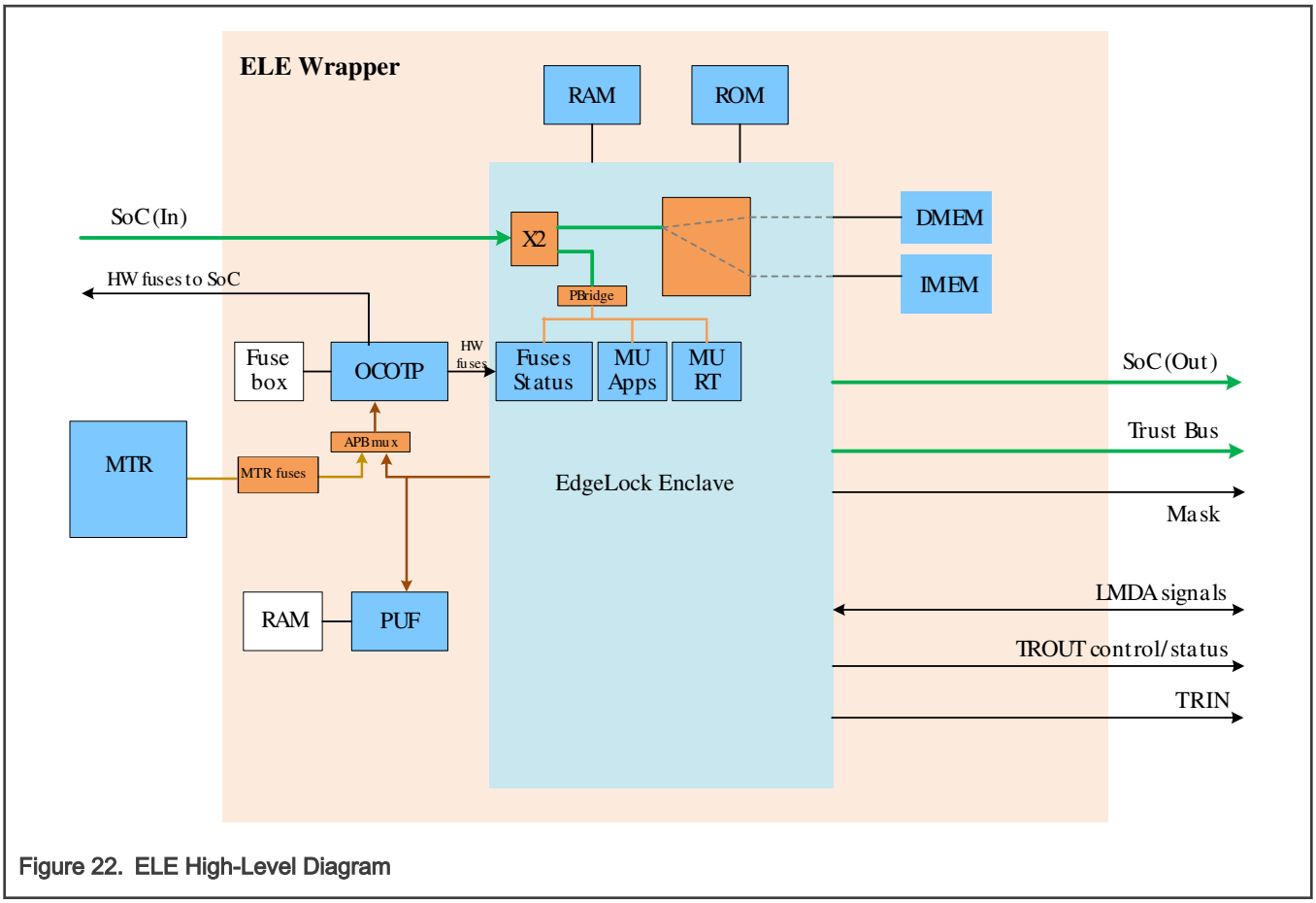


Figure 22. ELE High-Level Diagram

Name	Description
SoC Out	External Memory Accesses
SoC In	Communicate to MUs, and fast boot port to load FW
Trust Bus	Dedicated SoC security paths, security configurations and load keys
TRIN/TROUT	SoC security state machine interface
LMDA	System control
HW Fuses	Hardware only fuses
MTR	Memory Test Repair

ELE has a 32-bit bus with access to the SoC. The ELE has multiple MU instances, and private buses (Trust Bus and TROUT) to connect to security components outside ELE, such as BBSM and FlexSPI's OTFAD.

ELE has its own internal WDOG timer to ensure operation and frequency detect logic to ensure that the CM33 clock root frequency is no faster than the ELE clock root frequency.

NOTE

The ELE is not customer programmable. To use the ELE provided services, the ELE and SoC domains requesting (requestor) ELE services communicate using one of the available Messaging Units (S3MU). Refer to the EdgeLock Enclave (ELE) API Reference Guide for the complete list of available ELE services and details on the messaging protocol. For details on the S3MU, refer to the S3MU chapter in this manual.

7.1.1.1 ELE active timer

Every day (24 hours), a SoC application needs to kick the ELE Active Timer by exercising the PING request. If the SoC application does not handle this, ELE will reset the chip.

See the *EdgeLock Secure Enclave (ELE) ROM API Guide* for more information on the PING command.

7.1.2 On-Chip OTP Controller (OCOTP_CTRL)

The OCOTP_CTRL block is within the Edgelock subsystem and is used to control the OTP fuses. Access to fuses by the rest of the SoC is handled through Edgelock.

OCOTP supports:

- Load fuse into shadow registers
  - Hardware load: fuse will load into shadow registers automatically after power-on reset (POR)
  - Software load: fuse will reload into shadow registers after SW register configuration
- Register interface to allow SW to read, override or program the fuse.
- Flexible permission control to the fuse, including read-protect, override-protect, program-protect, lock, etc.
- Programming sequence to allow single bit to be programmed individually, and also allow the non-programmed bit to be programmed later.

7.1.3 Battery Backed Secure / Non-Secure Module (BBSM/BBNSM)

The BBSM and BBNSM is in the low power section VDD_BBSM_IN domain, which can be battery backed. This enables it to keep this data valid and continue to increment the time counter when the power goes down in the rest of the device. The always-powered up part of the module is isolated from the rest of the logic to ensure that it does not get corrupted when the device is powered down.

The BBSM and BBNSM are designed to comply with Digital Rights Management (DRM) and other security application rules and requirements. The BBSM incorporates a security monitor that checks for various security conditions. If a security violation is detected, it zeroizes the secret data and issues an interrupt request. The BBSM can also be configured to ignore specific violation inputs, so it can be used by systems that do not require security.

7.1.3.1 Monotonic Counter

- The counter can only increment.
- The counter does not roll over.
- The counter value is invalidated if there is loss of power to the VDD_BBSM_IN domain.
- The counter maintains state as long as the VDD_BBSM_IN is powered, even when SoC power is off.

7.1.3.2 Secure Real Time Clock

- Can be driven by an independent clock source (other than the clock that the SoC uses).
- Clock counter does not roll over. The RTC Alarm Register in BBSM represents the number of seconds, and in theory supports time alarm as high as ~140 years.
- Has a programmable timed interrupt. The RTC Alarm (from BBSM/BBNSM) are mapped as interrupt to NVIC and (where present) GIC. As per other interrupts in the i.MX RT1180, the RTC alarm is also a wake-up source from any low power modes.
- The Secure Real Time Clock continues counting when SoC power is off.

7.1.3.3 Tamper Detection

The chip supports up to 10 tamper pins: 5 of which are configurable to be either input or output, and 5 of which are statically input-only. It is SoC-specific how many tamper pins are connected in a given SoC.

7.1.4 TrustZone

TrustZone-M (TZ-M) is a feature on the Cortex M33, which provides segmentation of memory arrays and peripherals into either Secure (S) or Non-Secure (NS). This feature allows the M33 core to execute in either S or NS mode, and provides a secure separation between various application software.

The TrustZone® security architecture is supported. TrustZone-M is always enabled, as this is an integral feature of the Cortex-M33 BootROM and system security. For on-chip RAM, all of OCRAM/OCRAM_S have TrustZone® access control support. One region with configurable address range of the OCRAM/OCRAM_S can be set to TZ access only. The TrustZone memory region protection of DRAM is supported through TRDC.

The majority of system initiators in the SoC do not natively support TrustZone (and more specifically, do not support a concept of AxPROT[0] indicating controllable privileged vs normal access or AxPROT[1] indicating non-secure vs TrustZone secure access). Therefore, the TRDC DAC provides controls to override IP initiator's AxPROT[1:0] inputs to the subsystem NICs. However, initiators which do have concept of these AxPROT[1:0] signals, do not have their signals overridden by the TRDC DAC.

7.1.5 Trusted Resource Domain Control (TRDC)

The chip uses a domain based resource control architecture for the memory/peripheral resource sharing and isolation between the Cortex®-M7 core, Cortex®-M33 core, and other bus masters. A key feature of this architecture is that it must be possible for either Cortex-M7 platform or the Cortex-M33 platform to program the resource domain control access lists to implement a customer-defined policy. Another key feature is that any peripheral can be assigned to any domain. The TRDC supports multiple regions with flexible access permission control for each memory space.

7.1.6 Inline Encryption Engine (IEE)

The IEE provides for encryption and decryption in different contexts including multiple key sizes, encryption modes and algorithms. It can be used for code or data, volatile or non-volatile storage. It supports multiple regions where each region may have its own context. The context for each region is configured in the IEE and then each region is associated to a context in an upstream Memory Region Checker (MRC) on each master bus.

It provides the following features:

- SDRAM encryption/decryption
- FlexSPI decryption only
- Secure scan
- Secure on-chip key loading
- I/O direct encrypted storage and retrieval (Stream Support)

7.1.7 On-the-Fly AES Decryption (OTFAD)

The OTFAD provides decryption on the fly through the FLEXSPI interface, to decrypt the code and data fetched from the external flash memory.

It provides the following features:

- AES-128 Counter Mode On-the-Fly Decryption
  - 128-bit key and 128-bit data sizes
  - Receives 64-bit encrypted data from FlexSPI
- Hardware support for unwrapping key blobs
- Hardware support for 4 independent decryption segments, known as memory contexts

7.1.8 Block Control Secure (BLK_CTRL_S_AON)

The chip includes a block called BLK_CTRL_S_AON which contains assorted configuration controls that can potentially have security implications. The BLK_CTRL_S_AON registers are reset only by POR (when Edgelock is reset). These registers can only be accessed by TrustZone Secure World.

BLK_CTRL_S_AON has the following functions:

- AXBS_AON priority and arbitration controls.
- Cortex-M NVIC Interrupt Masking.
- WDOG Timer output masking for combined watchdog timer SoC output pin.

Chapter 8
System Debug

8.1 Overview

The chip debug architecture is based on Arm CoreSight technology, with support for the Cortex-M7, Cortex-M33, and Edgelock cores.

8.1.1 Block Diagram

The block diagram for the system level debug architecture is provided below.

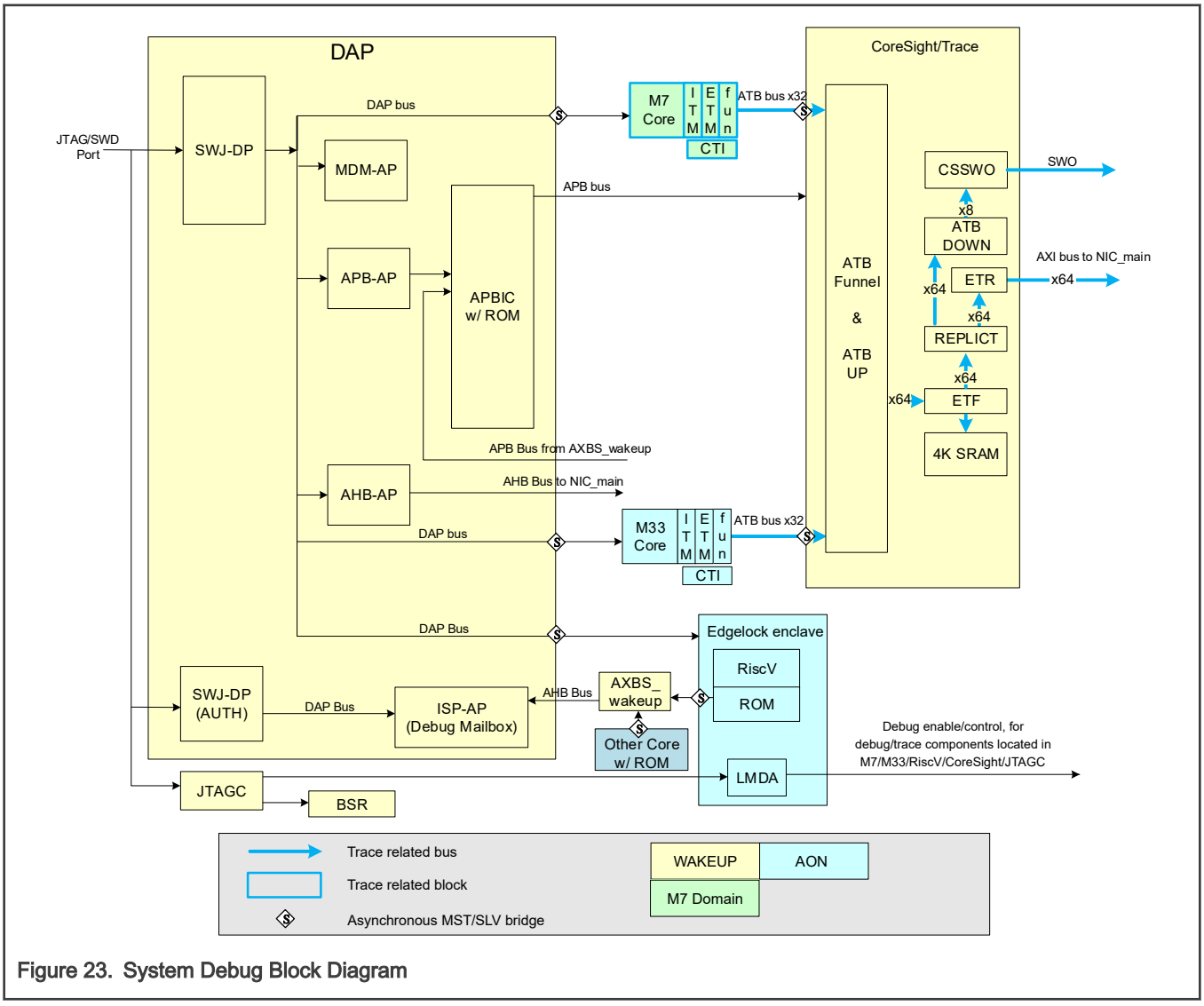


Figure 23. System Debug Block Diagram

The debug/trace components are listed below:

- Debug Access Port (DAP) - Has Serial Wire port (SWJ-DP) which interprets the data coming in and routes to appropriate Access Port (AP).
- Serial Wire port (SWJ-DP) - Consists of a wrapper around the JTAG-DP and SW-DP. It selects JTAG or SWD as the connection mechanism and enables either JTAG-DP or SW-DP as the interface to the various on chip debug/trace components. It is enabled only after debug is authenticated.

- SWJ-DP (AUTH) is another SWJ-DP instance dedicated for ISP-AP. It is enabled only when debug authentication is required, means it will be disabled if debug is authenticated.
- JTAGC
  - JTAGC and DAP/SWJ-DP share the same JTAG interface from PAD.
  - Achieve boundary scan testing, with an external BSR connected.
  - Supports Challenge-response authentication, with LMDA in Edgelock.
  - Supports IEEE 1149.6/AC JTAG.
- Debug Access port for Cortex-M7 platform.
- Debug Access port for CoreSight.
- Cortex-M33 DAP-AP - Debug Access port for M33 core.
- RISC-V DAP-AP - Debug Access port for RISC-V core in Edgelock.
- ISP-AP - Debug Access port for Debug Mailbox, to allow the debugger to communicate with onboard ROM code.
- MDM-AP registers for debugger to control multi-core halt/resume cores.
- CoreSight components.
- Cross-Trigger Interface (CTI) and Cross-Trigger Matrix (CTM).

8.1.2 Features

The key features of the debug system include:

- Supports 1149.1/ARM SWD interface. Supports both 5-pins (JTAG) and 2-pins (SWD) interfaces, to make efficient use of package pins, serial wire shares the JTAG pins use an auto-detect mechanism.

NOTE

1149.7 Compact JTAG is NOT supported.

- Supports both non-intrusive and halt-mode trace/debug options
- Supports trace for various cores, including a trace Memory Controller (TMC) to enable capturing trace
  - 4KB ETF in SOC trace block
  - ETR (4G memory range, 64-bit wide) to allow routing trace data to system memory
  - CoreSight Serial Wire Output
- Supports cross trigger between the Cortex-M7 and Cortex-M33
- MDM-AP registers for debugger to control multi-core halt/resume cores
- Supports narrow timestamp for trace components
- Supports ISP-AP (Debug Mailbox), allowing the debugger to communicate with onboard ROM code, providing debug authentication based on messages
- Supports challenge-response authentication

NOTE

Arm ELA-500 and CoreSight SoC-600 are not supported.

8.2 Functional description

8.2.1 Debug Access Port (DAP) TAP

DAP is a standard Arm component, comprising of several components. These components are used to access the DAP from an external debugger and Access Ports to access on-chip debug system resources. The DAP supports 1149.1/Arm SW-DP interface, which means that the JTAG interface can be operate in standard 5-pin JTAG-DP interface or in 2-pin SW-DP interface. The following figure shows the connectivity between the DAP and the pads.

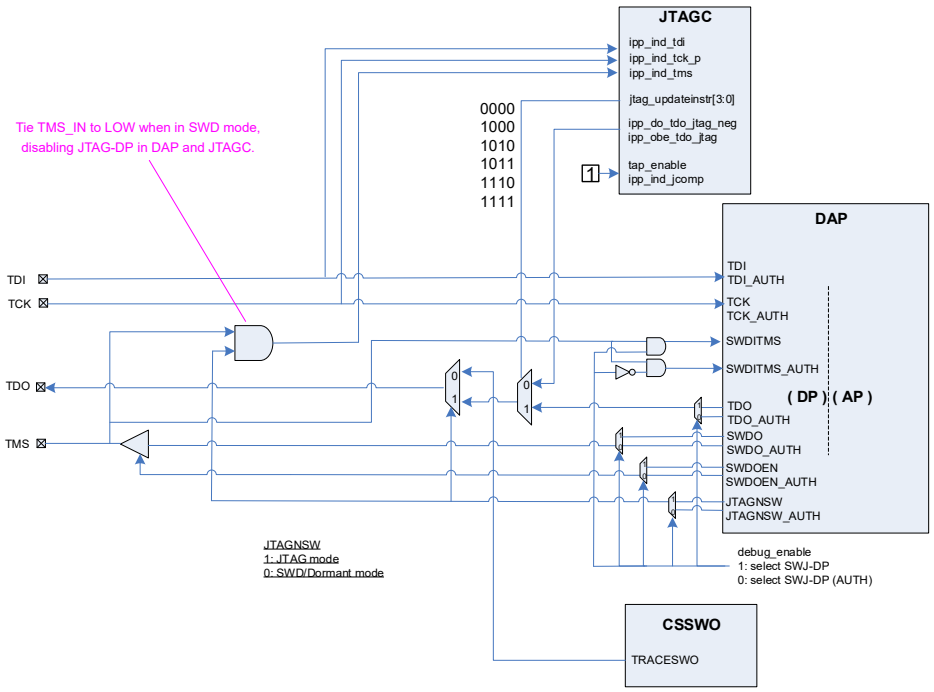


Figure 24. JTAG Pad Connectivity

For PAD descriptions, see External Signals.

8.2.2 JTAGC and JTAG mux

The JTAGC and SWJ-DP/SWJ-DP (AUTH) share the same 5-pin JTAG interface from PAD. SWJ-DP and SWJ-DP (AUTH) will not be enabled at the same time, controlled by the authentication status from LMDA.If debug is authenticated, select SWJ-DP, otherwise select SWJ-DP (AUTH).

SW-DP pins (TCK,TMS) are connected directly from and to SoC pins.

- When SW-DP is enabled, both JTAGC and JTAG-DP will be disabled.
- When SW-DP is disabled, the output to TDO from JTAGC and JTAG-DP will be selected by a mux and that mux is controlled by decode of the IR inside the JTAGC (JTAGC output port jtag_updateinstr[4:0]), and then connected to SoC pins for debug and/or test purposes.

Selection is based on the following table of decodes.

Table 33. Instruction Register

Instruction	Value	Module
EXTEST (Arm)	4'b0000	Arm JTAG-DP
IDCODE (JTAGC)	4'b0001	JTAGC
SAMPLE/PRELOAD	4'b0010	JTAGC

Table continues on the next page...

Table 33. Instruction Register (continued)

Instruction	Value	Module
SAMPLE	4'b0011	JTAGC
EXTEST (JTAGC)	4'b0100	JTAGC
TEST_LEAKAGE	4'b0101	JTAGC
ENABLE_TEST_CTRL	4'b0110	JTAGC
ABORT (Arm)	4'b1000	Arm JTAG-DP
HIGHZ	4'b1001	JTAGC
DPACC (Arm)	4'b1010	Arm JTAG-DP
APACC (Arm)	4'b1011	Arm JTAG-DP
CLAMP	4'b1100	JTAGC
IDCODE (Arm)	4'b1110	Arm JTAG-DP
BYPASS (Arm)	4'b1111	Arm JTAG-DP

Partial of TARGETID[31:0] input for SWJ-DP should also be connected to JTAGC, for coherence, as below:

Table 34. JTAG TARGETID

JTAGC	SWJ-DP	description
jtagc_pin_plug[9:0]	TARGETID[21:12]	Part Identification Number
jtagc_prn_plug[3:0]	TARGETID[31:28]	Part Revision Number
jtagc_jcomp_plug[0:J-1]	1'b1 (J=1)	JTAG JCOMP value

8.2.3 AP Select Decoding

The recommendation for AP selects decoding is provided in the following table.

NOTE

SWJ-DP (AUTH) has only one DAP slot (the first DAP slot) targeting ISP-AP, so it has no DAPBUSIC decoding logic.

The DAP BUS interconnect (DAPBUSIC) in the table below is only for SWJ-DP rather than SWJ-DP (AUTH).

Table 35. Address Decoding for DAPBUSIC

Destination	DAPBUSIC_AP_SEL[31:24]
AHB-AP to System Bus	8'h00 (M0)
APB-AP0 to CoreSight System	8'h01 (M1) See Address decoding for APBIC
DAP-AP to M7	8'h02 (M2)
DAP-AP to M33	8'h03 (M3)
DAP-AP to Edgelock enclave RISC-V core	8'h04 (M4)

Table continues on the next page...

Table 35. Address Decoding for DAPBUSIC (continued)

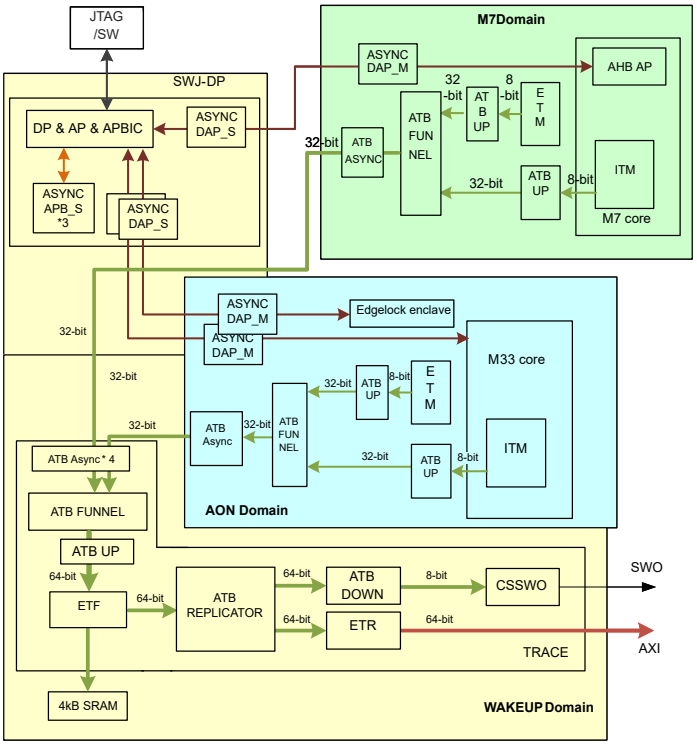
Destination	DAPBUSIC_AP_SEL[31:24]
Debug Mailbox ISP-AP to Edgelock enclave	Reserved
MDM-AP	8'h06 (M6)
Reserved (Default AP response)	8'h07 -8'hFF

Table 36. Address Decoding for APBIC

Destination	APBIC_TA_REG[24:16], 64KB
DAPROM - Debug Access Port ROM	9'b000000000
Reserved	9'b000000001 - 9'b001111111
Reserved	9'b010000000 - 9'b011111111
Reserved	9'b100000000
Reserved	9'b100000001
CoreSight Serial Wire Output	9'b100000010
CoreSight Embedded Trace FIFO	9'b100000011
CoreSight Embedded Trace Router	9'b100000100
CoreSight Cross Trigger Interface	9'b100000101
CoreSight Cross Trigger Interface	9'b100000110
Reserved	9'b100000111 - 9'b111111111

8.2.4 Trace

The debug and trace implementation for the Cortex-M7 and Cortex-M33 is shown in the following figure.



Note: APB/DAP/ATB ASYNC means couples of MST/SLV bridges located in corresponding domains.

Figure 25. Trace Diagram

- Trace sources include ETM in Cortex-M7 and ITM/ETM in Cortex-M33.
- One Embedded Trace FIFO (ETF, can operate as ETB, programmable), 4KB size as 128x256 (width x depth) with single port of SRAM.
- ATB funnel can capture trace data from Cortex-M7, Cortex-M33 ETM and Cortex-M33 ITM.
- ETR - Routing trace to system enabling to reuse SoC resources. (4G memory range, 64-bit wide).

NOTE

SRC_SCR_BT_RELEASE_M7 bit should be set for dual core parts when CM33 trace activity is proceeded.

8.2.5 Embedded Cross Trigger (ECT)

The ECT for CoreSight consists of a few CTIs and CTM. The diagram below shows the structure of the ECT. The ECT enables subsystems to interact, that is cross trigger with each other. It is used for multiple core run control and trace cross trigger.

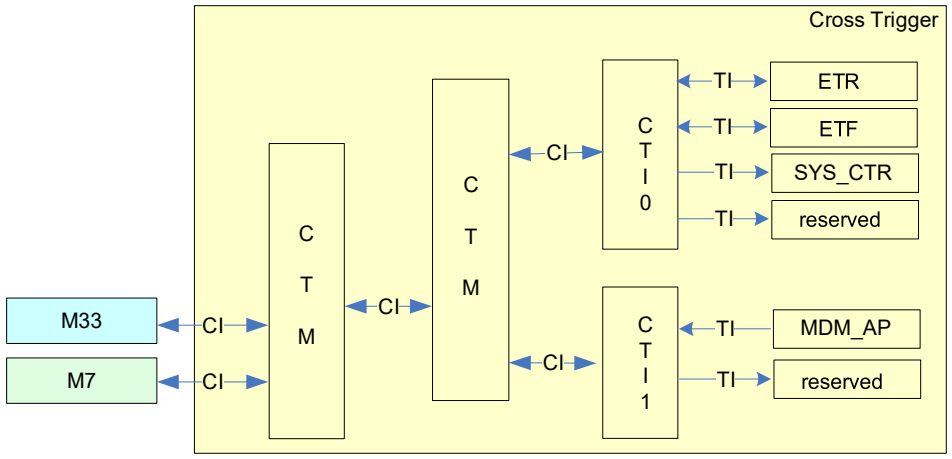


Figure 26. Embedded Cross Trigger Diagram

8.2.6 Timestamp Generation

There is no Arm CoreSight Timestamp Generator in the system, and instead the System Counter is the source of the time used by all of the following components:

- System Counter internal counters and comparators.
- Cortex-M7 timestamp value (for trace).
- Cortex-M33 timestamp value (for trace).
- TSTMR1 and TSTMR2 read-only timestamp snapshot registers.

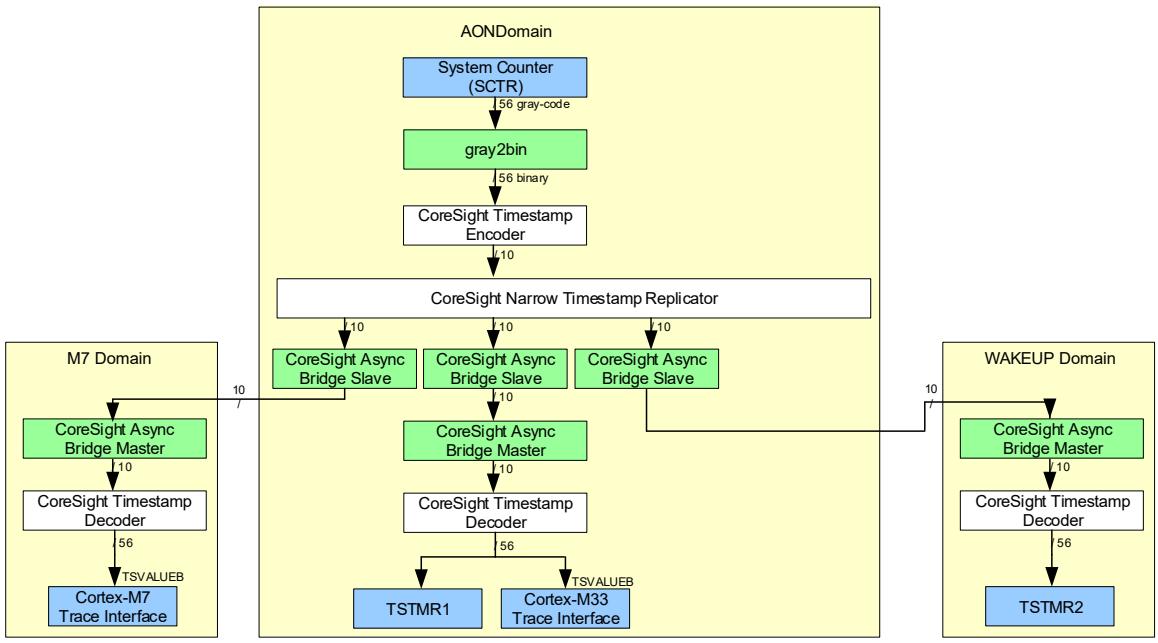


Figure 27. Timestamp Distribution

8.2.7 JTAG IDCODE

The JTAG_IDCODE (JTAG ID) is a 32-bit identifier that identifies the chip silicon for the purpose of both debug and boundary scan access.

Table 37. JTAG_IDCODE

Device	JTAG_IDCODE
i.MX RT1180	0x1892_001D

8.3 External Signals

The JTAG signals are outlined in the following table.

Table 38. External signals

Name	I/O	JTAG	SWD	Pullup (PU) / Pulldown (PD)
TMS/SWDIO	I/IO	Mode selection	Data In/Out	PU
TCLK/SWCLK	I	Clock	Clock	PD
TDI	I	Data input	-	PU
TDO	O	Data output	-	NC

Chapter 9
JTAG Controller (JTAGC)

9.1 Chip-specific JTAGC information

Table 39. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

This device has one instance of the JTAGC module.

The initialization sequence is needed before running the boundary scan test. Please send the JTAG instruction "ENABLE_TEST_CTRL" and then configure the corresponding JTAG data register 5'b00001. Please make sure no JTAG reset is triggered after the initialization sequence.

9.2 JTAC instructions

See Table 33 of the System Debug chapter for the instruction set for this device.

9.3 Overview

JTAGC allows you to test chip functionality and connectivity while remaining transparent to system logic when not in test mode. Testing is performed via a boundary scan technique, as defined in the IEEE 1149.1-2001 standard. All data input to and output from JTAGC is communicated in serial format.

9.3.1 Block diagram

The following is a simplified block diagram of the JTAG Controller (JTAGC) block. See the chip-specific configuration information within this chapter as well as Register description for more information about the JTAGC registers.

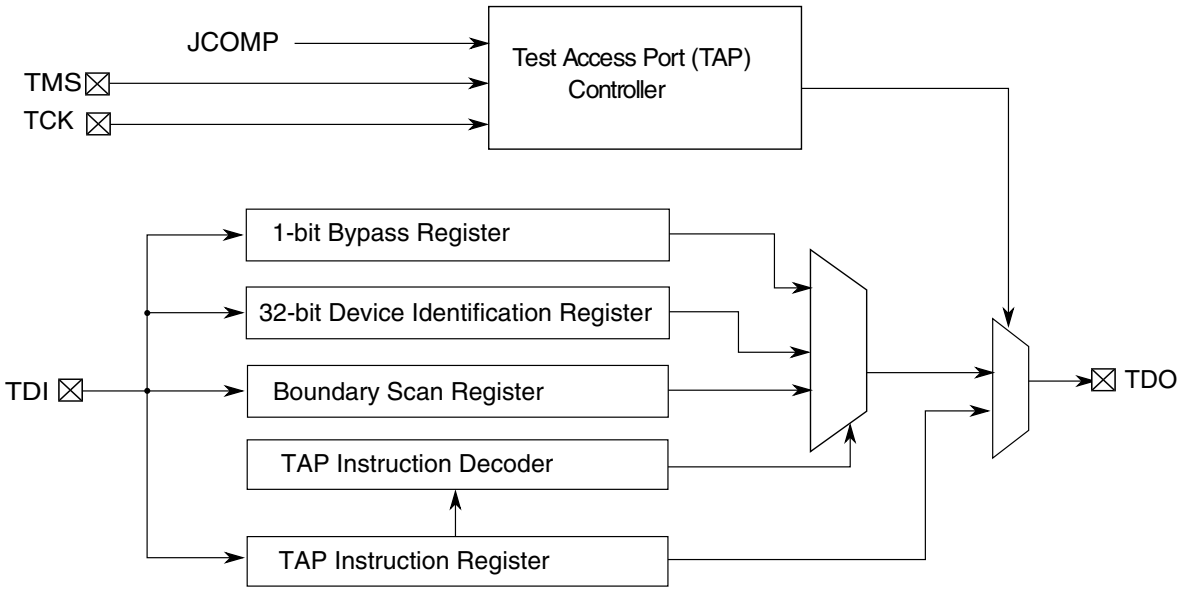


Figure 28. JTAG (IEEE 1149.1) block diagram

9.3.2 Features

The JTAGC block is compliant with the IEEE 1149.1-2001 standard, and supports the following features:

- IEEE 1149.1-2001 TAP interface
  - Four pins (TDI, TMS, TCK, and TDO)
- JCOMP input that provides reset control
- Instruction register that supports several IEEE 1149.1–2001 defined instructions as well as several public and private device-specific instructions (see JTAGC block instructions for a list of supported instructions).
- Bypass register, boundary scan register, and device identification register
- TAP controller state machine that controls the operation of the data registers, instruction register, and associated circuitry

9.4 Functional description

9.4.1 Modes of operation

The JTAGC block uses JCOMP and a power-on reset indication as its primary reset signals. Several IEEE 1149.1–2001 defined test modes are supported, as well as a bypass mode.

9.4.1.1 IEEE 1149.1–2001 defined test modes

The JTAGC block supports several IEEE 1149.1–2001 defined test modes. A test mode is selected by loading the appropriate instruction into the instruction register when the JTAGC is enabled. Supported test instructions include EXTEST, HIGHZ, CLAMP, PRELOAD.

Each instruction defines the set of data register(s) that may operate and interact with the on-chip system logic when the instruction is current. Only one test data register path is enabled to shift data between TDI and TDO for each instruction.

The boundary scan register is enabled for serial access between TDI and TDO when the EXTEST, PRELOAD instructions are active. The single-bit bypass register shift stage is enabled for serial access between TDI and TDO when the following instructions are active:

- BYPASS

- HIGHZ
- CLAMP

The functionality of each test mode is explained in more detail in JTAGC block instructions.

9.4.1.2 Bypass mode

When no test operation is required, the BYPASS instruction can be loaded to place the JTAGC block into bypass mode. When in bypass mode, the single-bit bypass shift register is used to provide a minimum-length serial path to shift data between TDI and TDO.

9.4.2 IEEE 1149.1-2001 (JTAG) TAP

The JTAGC block uses the IEEE 1149.1-2001 TAP for accessing registers. This port can be shared with other TAP controllers on the MCU. Ownership of the port is determined by the value of the currently loaded instruction.

Data is shifted between TDI and TDO through the selected register starting with the least significant bit, as illustrated in the following figure. This applies for the instruction register, test data registers, and the bypass register.

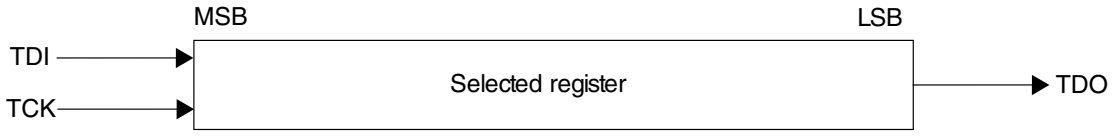
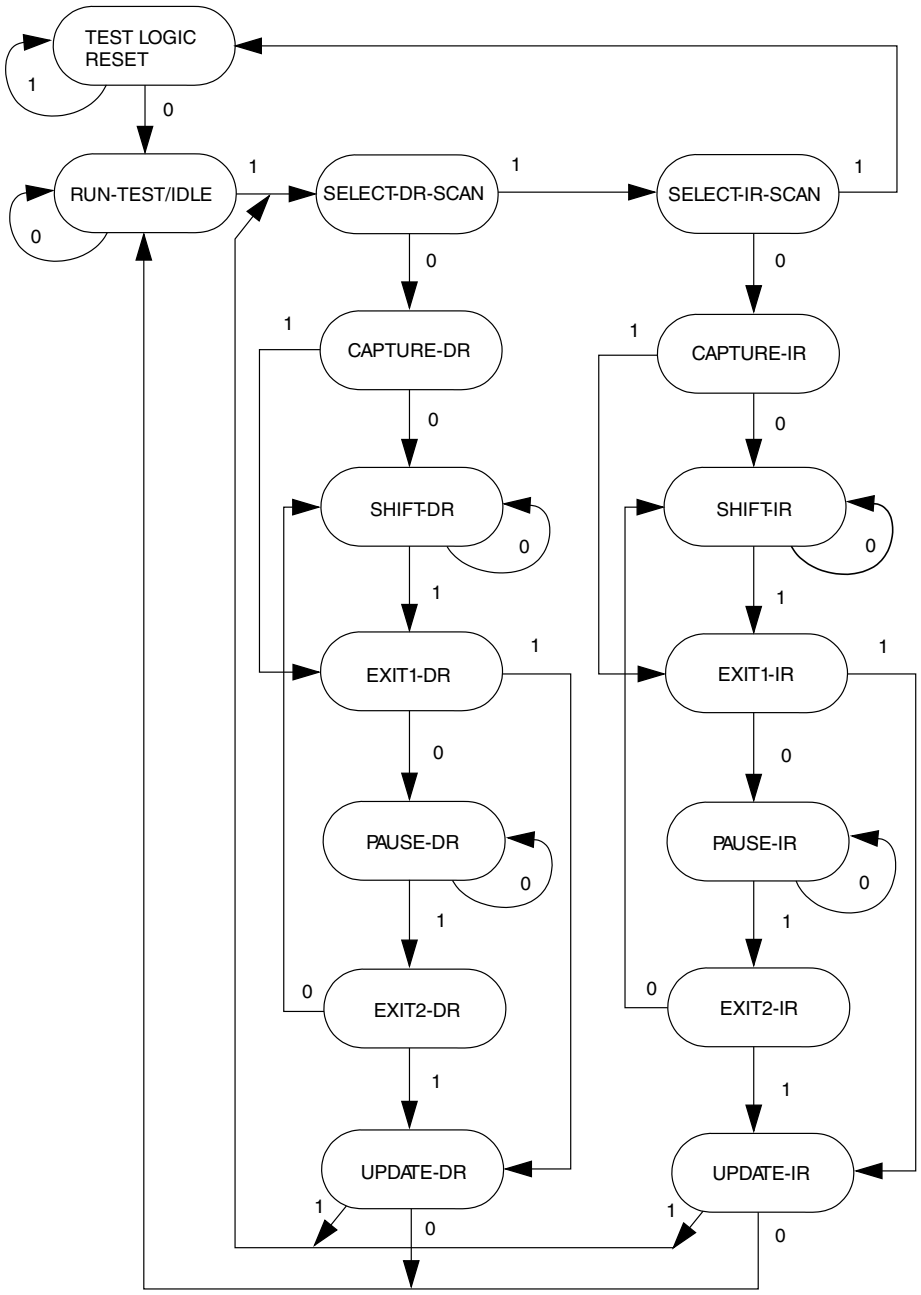


Figure 29. Shifting data through a register

9.4.3 TAP controller state machine

The TAP controller is a synchronous state machine that interprets the sequence of logical values on the TMS pin.

The following figure shows the machine's states. The value shown next to each state is the value of the TMS signal sampled on the rising edge of the TCK signal. As the figure shows, holding TMS at logic 1 when clocking TCK through a sufficient number of rising edges also causes the state machine to enter the Test-Logic-Reset state.



The value shown adjacent to each state transition in this figure represents the value of TMS at the time of a rising edge of TCK.

Figure 30. IEEE 1149.1-2001 TAP controller finite state machine

9.4.3.1 Enabling the TAP controller

The JTAGC TAP controller is enabled by setting JCOMP to a logic 1 value.

9.4.3.2 Selecting an IEEE 1149.1-2001 register

Access to the JTAGC data registers is achieved by loading the instruction register with any of the JTAGC block instructions when the JTAGC is enabled. Instructions are shifted in via the Select-IR-Scan path and loaded in the Update-IR state. At this point, all data register access is performed via the Select-DR-Scan path.

The Select-DR-Scan path is used to read or write the register data by shifting in the data (LSB first) during the Shift-DR state. When reading a register, the register value is loaded into the IEEE 1149.1-2001 shifter during the Capture-DR state. When writing a register, the value is loaded from the IEEE 1149.1-2001 shifter to the register during the Update-DR state. When reading a register, there is no requirement to shift out the entire register contents. Shifting may be terminated after the required number of bits have been acquired.

9.4.4 JTAGC block instructions

The JTAGC block implements the IEEE 1149.1-2001 defined instructions. See the IEEE 1149.1-2001 standard for more details. All undefined opcodes are reserved.

NOTE

See the chip specific information JTAGC block instructions for applicable to your device.

9.4.4.1 IDCODE instruction

IDCODE selects the 32-bit device identification register as the shift path between TDI and TDO. This instruction allows interrogation of the MCU to determine its version number and other part identification data. IDCODE is the instruction placed into the instruction register when the JTAGC block is reset.

9.4.4.2 PRELOAD instruction

The PRELOAD instruction has two functions:

- The PRELOAD portion of the instruction initializes the boundary scan register cells before selecting the EXTEST or CLAMP instructions to perform boundary scan tests. This is achieved by shifting in initialization data to the boundary scan register during the Shift-DR state. The initialization data is transferred to the parallel outputs of the boundary scan register cells on the falling edge of TCK in the Update-DR state. The data is applied to the external output pins by the EXTEST or CLAMP instruction. System operation is not affected.

9.4.4.3 EXTEST external test instruction

EXTEST selects the boundary scan register as the shift path between TDI and TDO. It allows testing of off-chip circuitry and board-level interconnections by driving preloaded data contained in the boundary scan register onto the system output pins. Typically, the preloaded data is loaded into the boundary scan register using the PRELOAD instruction before the selection of EXTEST. EXTEST asserts the internal system reset for the MCU to force a predictable internal state when performing external boundary scan operations.

9.4.4.4 HIGHZ instruction

HIGHZ selects the bypass register as the shift path between TDI and TDO. When HIGHZ is active all output drivers are placed in an inactive drive state (for example, high impedance). HIGHZ also asserts the internal system reset for the MCU to force a predictable internal state.

9.4.4.5 CLAMP instruction

CLAMP allows the state of signals driven from MCU pins to be determined from the boundary scan register when the bypass register is selected as the serial path between TDI and TDO. CLAMP enhances test efficiency by reducing the overall shift path to a single bit (the bypass register) when conducting an EXTEST type of instruction through the boundary scan register. CLAMP also asserts the internal system reset for the MCU to force a predictable internal state.

9.4.4.6 BYPASS instruction

BYPASS selects the bypass register, creating a single-bit shift register path between TDI and TDO. BYPASS enhances test efficiency by reducing the overall shift path when no test operation of the MCU is required. This allows more rapid movement of test data to and from other components on a board that are required to perform test functions. When the BYPASS instruction is active the system logic operates normally.

9.4.5 Boundary scan

The boundary scan technique allows signals at component boundaries to be controlled and observed through the shift-register stage associated with each pad. Each stage is part of a larger boundary scan register cell, and cells for each pad are interconnected serially to form a shift-register chain around the border of the design. The boundary scan register consists of this shift-register chain, and is connected between TDI and TDO when the EXTEST, PRELOAD instructions are loaded. The shift-register chain contains a serial input and serial output, as well as clock and control signals.

9.4.6 Clocking

There is one clock for this module. For more information See External signals.

9.4.7 Interrupts

This module has no interrupts.

9.4.8 Reset

The JTAGC block is placed in reset when:

- Power-on reset is asserted
- JCOMP is negated
- TMS input is held high for enough consecutive rising edges of TCK to sequence the TAP controller state machine into the Test-Logic-Reset state

Holding TMS high for five consecutive rising edges of TCK guarantees entry into the Test-Logic-Reset state regardless of the current TAP controller state. Asserting power-on reset or setting JCOMP to a value other than the value required to enable the JTAGC block results in asynchronous entry into the reset state.

When in reset, the following actions occur:

- The TAP controller is forced into the Test-Logic-Reset state, thereby disabling the test logic and allowing normal operation of the on-chip system logic to continue unhindered.
- The instruction register is loaded with the IDCODE instruction.

9.4.8.1 JTAGC reset configuration

When in reset, the TAP controller is forced into the Test-Logic-Reset state, thus disabling the test logic and allowing normal operation of the on-chip system logic. In addition, the instruction register is loaded with the IDCODE instruction.

9.5 External signals

JTAGC includes a set of signals that connect to off-chip development tools and allow access to test support functions. The JTAGC signals are outlined in the following table and described in the following sections.

Table 40. JTAG signal properties

Name	I/O	Function	Reset state
TCK	Input	Test clock	Weak pulldown
TDI	Input	Test data in	Weak pullup
TDO	Output	Test data out	High Z[1]
TMS	Input	Test mode select	Weak pullup
JCOMP	Input	JTAG compliance	Weak pulldown

1. TDO output buffer enable is negated when the JTAGC is not in the Shift-IR or Shift-DR states. A weak pull may be implemented at the TDO pad for use when JTAGC is inactive.

9.5.1 Test clock input (TCK)

Test Clock Input (TCK) is an input pin used to synchronize the test logic and control register access through the TAP.

9.5.2 Test data input (TDI)

Test Data Input (TDI) is an input pin that receives serial test instructions and data. TDI is sampled on the rising edge of TCK.

9.5.3 Test data output (TDO)

Test Data Output (TDO) is an output pin that transmits serial output for test instructions and data. TDO is tristateable and is actively driven only in the Shift-IR and Shift-DR states of the TAP controller state machine, which is described in TAP controller state machine.

9.5.4 Test mode select (TMS)

Test Mode Select (TMS) is an input pin used to sequence the IEEE 1149.1-2001 test control state machine. TMS is sampled on the rising edge of TCK.

9.5.5 JCOMP JTAG compliance

The JCOMP signal provides IEEE 1149.1-2001 compatibility and provides the ability to share the TAP. The JTAGC TAP controller is enabled when JCOMP is set to the JTAGC enable encoding—otherwise the JTAGC TAP controller remains in reset.

9.6 Initialization

To initialize the JTAGC block and enable access to registers, the following sequence is required:

1. Set the JCOMP signal to the JTAGC enable value, thereby enabling the JTAGC TAP controller.
2. Load the appropriate instruction for the test or action to be performed.

9.7 Application information

The test logic is a static logic design, and TCK can be stopped in either a high or low state without loss of data. However, the system clock is not synchronized to TCK internally. Any mixed operation using both the test logic and the system functional logic requires external synchronization.

9.8 Register description

This section provides a detailed description of the JTAGC block registers accessible through the TAP interface, including data registers and the instruction register. Individual bit-level descriptions and reset states of each register are included. These registers are not memory-mapped and can only be accessed through the TAP.

9.8.1 Instruction register

The JTAGC block uses a 4-bit instruction register as shown in the following figure.

The instruction register allows instructions to be loaded into the block to select the test to be performed, or the test data register to be accessed, or both. Instructions are shifted in through TDI when the TAP controller is in the Shift-IR state, and latched on the falling edge of TCK in the Update-IR state. The latched instruction value can only be changed in the Update-IR and Test-Logic-Reset TAP controller states.

Synchronous entry into the Test-Logic-Reset state results in the IDCODE instruction being loaded on the falling edge of TCK. Asynchronous entry into the Test-Logic-Reset state results in asynchronous loading of the IDCODE instruction. During the

Capture-IR TAP controller state, the instruction shift register is loaded with the value 0b1, making this value the register's read value when the TAP controller is sequenced into the Shift-IR state.

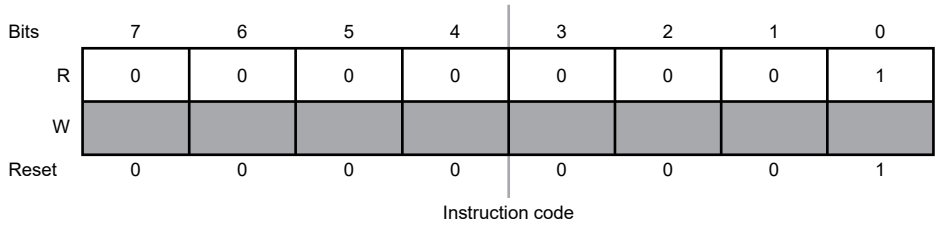


Figure 31. Instruction register

9.8.2 Bypass register

The bypass register is a single-bit shift register path selected for serial data transfer between TDI and TDO when the following instructions are active:

- BYPASS
- HIGHZ
- CLAMP

After entry into the Capture-DR state, the single-bit shift register is set to a logic 0. Therefore, the first bit shifted out after selecting the bypass register is always a logic 0.

9.8.3 Device identification register

The device identification (JTAG ID) register, shown in the following figure, allows the revision number, part number, manufacturer, and design center responsible for the design of the part to be determined through the TAP.

The device identification register is selected for serial data transfer between TDI and TDO when the IDCODE instruction is active. Entry into the Capture-DR state when the device identification register is selected loads the IDCODE into the shift register to be shifted out on TDO in the Shift-DR state. No action occurs in the Update-DR state.

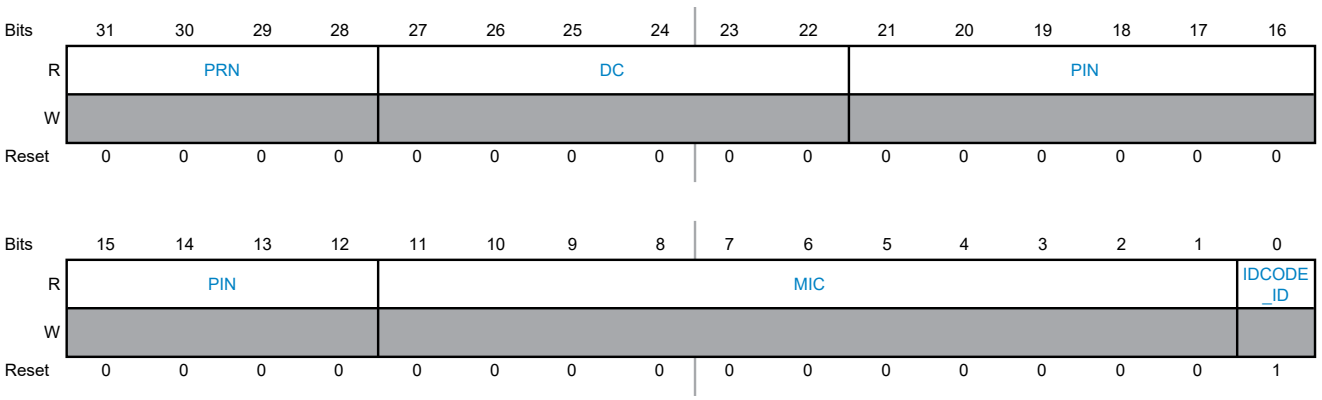


Figure 32. Device identification register

The following table describes the device identification register functions. The device identification register values are described in the chip-specific JTAGC information.

Table 41. Device identification register field descriptions

Field	Function
PRN	Part revision number Contains the revision number of the part.
DC	Design center Indicates the design center.
PIN	Part identification number Contains the part number of the device.
MIC	Manufacturer identity code Contains the reduced Joint Electron Device Engineering Council (JEDEC) ID .
IDCODE ID	IDCODE register ID Identifies this register as the device identification register and not the bypass register. Always set to 1.

9.8.4 Boundary scan register

The boundary scan register is connected between TDI and TDO when the EXTEST, PRELOAD instructions are active. It is used to:

- Capture input pin data
- Force fixed values on output pins
- Select a logic value and direction for bidirectional pins

Each bit of the boundary scan register represents a separate boundary scan register cell, as described in the IEEE 1149.1-2001 standard and discussed in Boundary scan. The size of the boundary scan register and bit ordering is device-dependent and can be found in the device BSDL file.

Chapter 10
System Counter (SYS_CTR)

10.1 Chip-specific SYS_CTR Information

Table 42. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

10.2 Overview

SYS_CTR is a programmable system counter that provides you with a shared time base to multiple processors. It is intended for applications in which the counter is always powered on and supports multiple unrelated clocks.

10.2.1 Block diagram

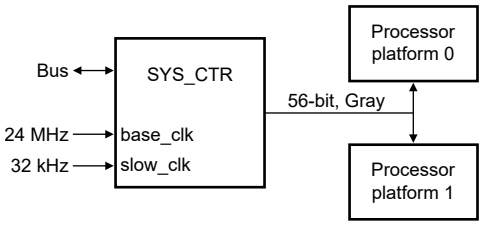


Figure 33. Block diagram

10.2.2 Features

- Two counter clock frequencies:
  - Base clock for normal operation
  - Alternate slow clock for low-power operation

- 56-bit counter width
- Gray-coded counter output for distribution to the processor timers
- Two compare frames, with each frame:
  - Containing a 64-bit compare value
  - Supporting a programmable interrupt generation

10.3 Functional description

SYS_CTR inputs two counter clock sources and outputs a gray-coded counter value and interrupt signals (one per compare frame) to the platform's interrupt controller.

10.3.1 Modes of operation

Table 43. Modes of operation

Mode	Function
Normal	The counter increments on the rising edge of the selected clock when enabled in Normal mode.
Debug	You can configure the SYS_CTR debug operation via Counter Control (CNTCR): • If CNTCR[HDBG] = 0, SYS_CTR continues to operate. • If CNTCR[HDBG] = 1, SYS_CTR halts when its debug input signal is asserted.

10.3.2 Operation

After reset, SYS_CTR is disabled with the count value reset to 0 and base frequency selected. After the counter is enabled, it increments the appropriate value on each rising edge of the selected clock. Because SYS_CTR handles a 56-bit count value across multiple clock domains, synchronization is necessary. SYS_CTR provides synchronization mechanisms between the various clock domains.

After the system switches the counter's clock source, there is a short pause while the clock multiplexer is handling the clock transition. To maintain an accurate count value, the clock control logic employs two offset counters: one for base-to-slow transition and the other one for slow-to-base transition. These offset counters only operate during the clock transition time to compensate for the idled source clock. Both counters run off the base clock. The transition offset values are added to the system count value at an appropriate time after the counter's clock is restored.

NOTE

Both the base clock and alternate clock must be running when changing frequencies.

10.3.3 Clocks

Table 44. Clocks

Clock	Description
base_clk	Base clock This clock is used during normal operation.
slow_clk	Slow clock This clock is used during low-power mode. It is internally divided by 64 before use.

10.3.4 Interrupts

Each SYS_CTR compare frame (SYS_CTR_COMPARE) is capable of generating one maskable interrupt.

10.4 External signals

This module has no external signals.

10.5 Initialization

To initialize SYS_CTR:

1. Write an initial value to Counter Count Value Low (CNTCV0) and Counter Count Value High (CNTCV1), if a different value is desired.
2. Write 1 to CNTCR[EN] to enable the counter.

10.6 Application information

To use a compare frame:

1. Write the desired count value into Compare Count Value Low (CMPCVL0 - CMPCVL1) and Compare Count Value High (CMPCVH0 - CMPCVH1).
2. Enable interrupts using CMPCR0[IMASK], if required.
3. Write 1 to CMPCR0[EN] to enable the compare function.
4. Read CMPCR0[ISTAT] to check whether the count value is reached.

10.7 Programming model

SYS_CTR contains three programming model sections. Each section has an independent module enable to allow individual secure access control. See the following table for details.

Table 45. Programming model

Programming model	Description
Control	Controls the counter frequency, enable, debug, and count value. This model also contains the frequency-modes table that supplies control and status for the frequency of SYS_CTR.
Read	Allows you to read the count value.
Compare	Controls the compare value frequency, enable, interrupt mask, and interrupt status for each compare frame.

10.8 Memory map and register definition

10.8.1 SYS_CTR_CONTROL register descriptions

NOTE

Write or read access to reserved locations does not generate a transfer error.

NOTE

All 32-bit registers allow only 32-bit wide write operations, and are aligned to 32 bits.

10.8.1.1 SYS_CTR_CONTROL memory map

SYS_CTR_CONTROL base address: 4429_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Counter Control (CNTCR)	32	RW	0000_0000h
4h	Counter Status (CNTSR)	32	R	0000_0100h
8h	Counter Count Value Low (CNTCV0)	32	RW	0000_0000h
Ch	Counter Count Value High (CNTCV1)	32	RW	0000_0000h
20h	Frequency-Modes Table 0 (CNTFID0)	32	R	016E_3600h
24h	Frequency-Modes Table 1 (CNTFID1)	32	R	0000_0200h
28h	Frequency-Modes Table 2 (CNTFID2)	32	R	0000_0000h
C0h	Counter Control 2 (CNTCR2)	32	RW	0000_0000h
FD0h	Counter ID (CNTID0)	32	R	0000_0000h

10.8.1.2 Counter Control (CNTCR)

Offset

Register	Offset
CNTCR	0h

Function

Defines the basic operating configuration of SYS_CTR, which operates using a fixed base frequency. However, the counter can increment at a lower alternate frequency than the base frequency, using a correspondingly larger increment. For example, the counter can increment by 46,875 and run at a frequency of 1/46875 of the base frequency, which is the normal operating frequency. The alternate frequency is a significantly lower operating frequency used to reduce power consumption. The frequency-modes table identifies the two available frequencies. These two frequencies are the base frequency (table entry 0) and the lower, alternate frequency (table entry 1). Writing 1 to CNTCR[FCR1] selects the alternate frequency, and writing 1 to CNTCR[FCR0] selects the base frequency. Writing 1 or writing 0 to both FCR0 and FCR1 has no effect and the frequency does not change.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						FCR1	FCR0	Reserved						HDBG	EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 FCR1	Frequency Change Request, ID 1 Selects frequency-modes table entry 1. 0b - No change 1b - Base frequency
8 FCR0	Frequency Change Request, ID 0 Selects frequency-modes table entry 0. 0b - No change 1b - Base frequency
7-2 —	Reserved
1 HDBG	Enable Debug Halt Specifies whether the assertion of the debug input is ignored. 0b - Ignored 1b - Causes SYS_CTR to halt
0 EN	Enable Counting Enables counting. 0b - Disable 1b - Enable

10.8.1.3 Counter Status (CNTSR)

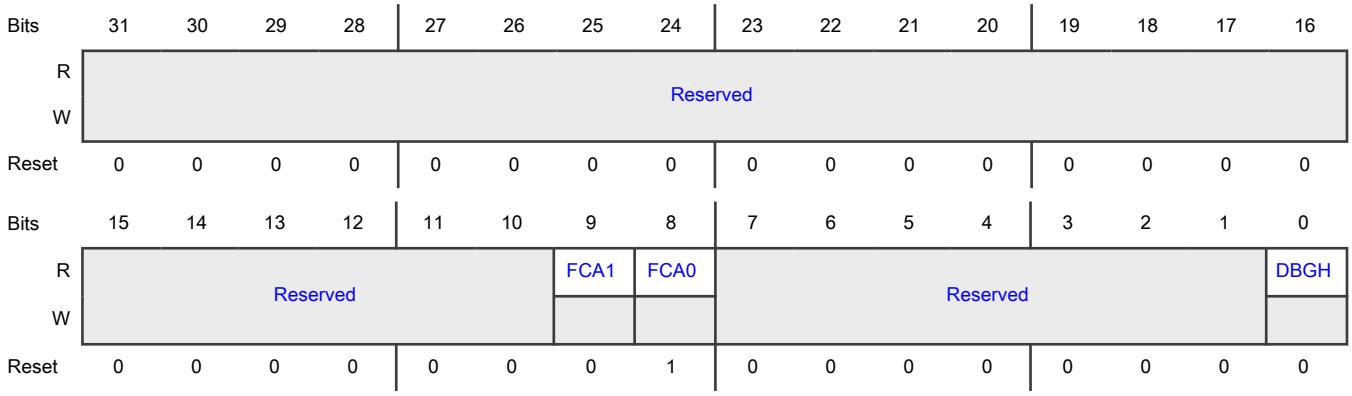
Offset

Register	Offset
CNTSR	4h

Function

Provides information concerning the clock frequency and debug state of SYS_CTR.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 FCA1	Frequency Change Acknowledge, ID 1 Indicates whether the base frequency, entry 1, is selected. 0b - Not selected 1b - Selected
8 FCA0	Frequency Change Acknowledge, ID 0 Indicates whether the base frequency, entry 0, is selected. 0b - Not selected 1b - Selected
7-1 —	Reserved
0 DBGH	Debug Halt Indicates whether debug halted the counter. 0b - Did not halt 1b - Halted

10.8.1.4 Counter Count Value Low (CNTCV0)

Offset

Register	Offset
CNTCV0	8h

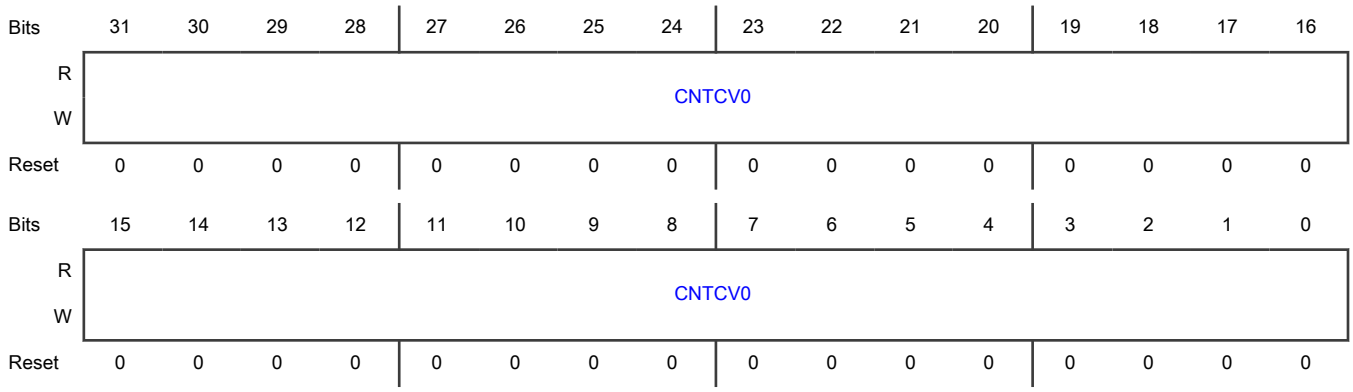
Function

Specifies the current count value bits (31–0).

NOTE

You must write to the CNTCV registers when operating on the base frequency only. Writes to these registers when running on the alternate frequency may have unpredictable results.

Diagram



Fields

Field	Function
31-0 CNTCV0	Counter Count Value Bits [31:0] Specifies the lower 32 bits of the 56-bit system count value.

10.8.1.5 Counter Count Value High (CNTCV1)

Offset

Register	Offset
CNTCV1	Ch

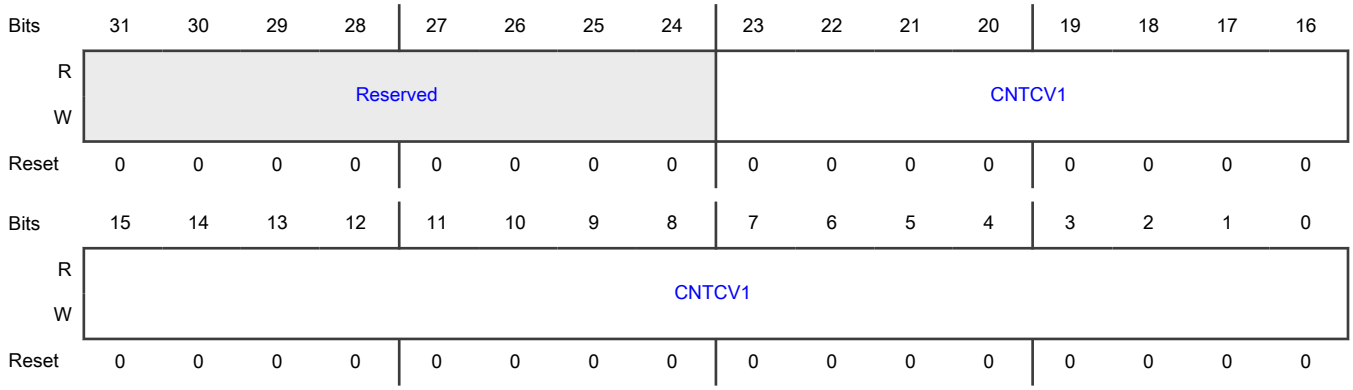
Function

Specifies the current count value bits (63–32).

NOTE

You must write to the CNTCV registers when operating on the base frequency only. Writes to these registers when running on the alternate frequency may have unpredictable results.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 CNTCV1	Counter Count Value Bits [55:32] Specifies the upper 24 bits of the 56-bit system count value.

10.8.1.6 Frequency-Modes Table 0 (CNTFID0)

Offset

Register	Offset
CNTFID0	20h

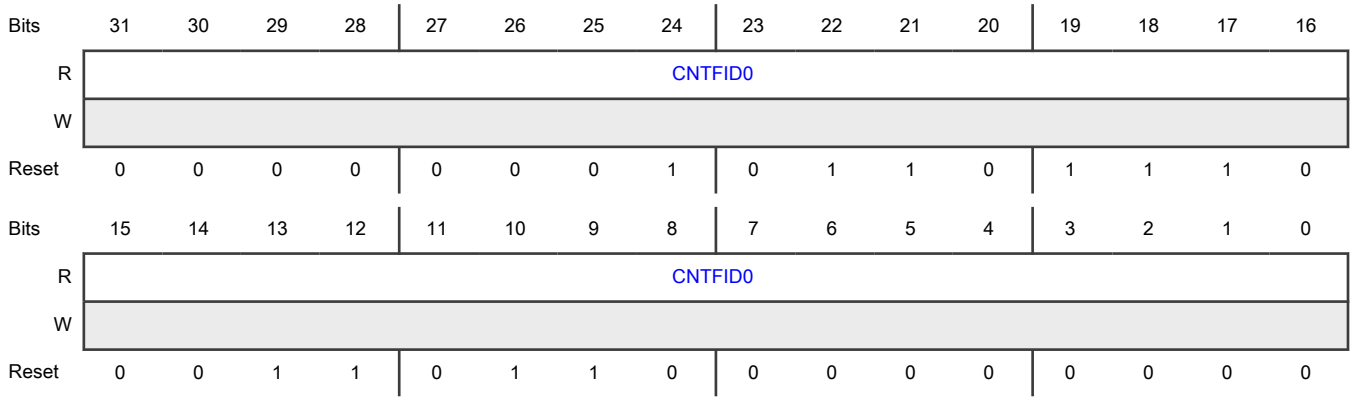
Function

Indicates the frequency-modes table starting at offset 20h.

Table entries contain 32 bits, and each entry specifies the SYS_CTR update frequency in Hz. The first entry in the table specifies the base frequency of SYS_CTR. To ensure that the overall counter accuracy is maintained, any subsequent entries in the table are exact divisors of the base frequency.

When SYS_CTR is operating at a lower frequency than the base frequency, the increment applied at each counter update is base_frequency ÷ selected_frequency; a 32-bit word of 0 value marks the end of the table. That is, the word of memory immediately after the last entry in the table is 0.

Diagram



Fields

Field	Function
31-0 CNTFID0	Counter Frequency ID 0 Indicates the base frequency, 24 MHz.

10.8.1.7 Frequency-Modes Table 1 (CNTFID1)

Offset

Register	Offset
CNTFID1	24h

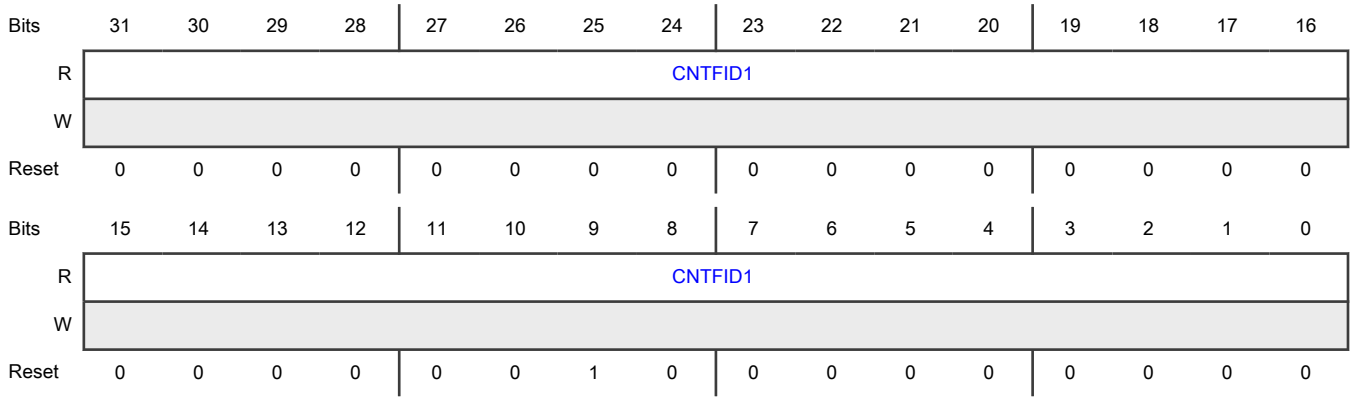
Function

Indicates the frequency-modes table starting at offset 20h.

Table entries contain 32 bits, and each entry specifies the SYS_CTR update frequency in Hz. The first entry in the table specifies the base frequency of SYS_CTR. To ensure that the overall counter accuracy is maintained, any subsequent entries in the table are exact divisors of the base frequency.

When SYS_CTR is operating at a lower frequency than the base frequency, the increment applied at each counter update is base frequency ÷ selected frequency; a 32-bit word of 0 value marks the end of the table. That is, the word of memory immediately after the last entry in the table is 0.

Diagram



Fields

Field	Function
31-0 CNTFID1	Counter Frequency ID 1 Indicates the alternate frequency, 512 Hz.

10.8.1.8 Frequency-Modes Table 2 (CNTFID2)

Offset

Register	Offset
CNTFID2	28h

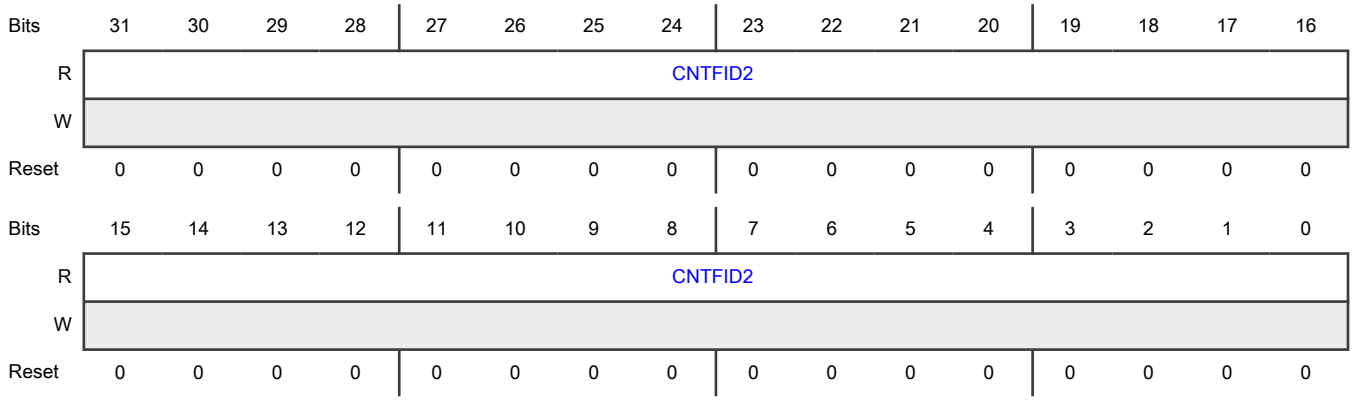
Function

Indicates the frequency-modes table starting at offset 20h.

Table entries are 32 bits, and each entry specifies the SYS_CTR update frequency in Hz. The first entry in the table specifies the base frequency of SYS_CTR. To ensure that the overall counter accuracy is maintained, any subsequent entries in the table are exact divisors of the base frequency.

When SYS_CTR is operating at a lower frequency than the base frequency, the increment applied at each counter update is base frequency ÷ selected frequency; a 32-bit word of 0 value marks the end of the table. That is, the word of memory immediately after the last entry in the table is 0.

Diagram



Fields

Field	Function
31-0 CNTFID2	Counter Frequency ID 2 Indicates the end of the frequency-modes table.

10.8.1.9 Counter Control 2 (CNTCR2)

Offset

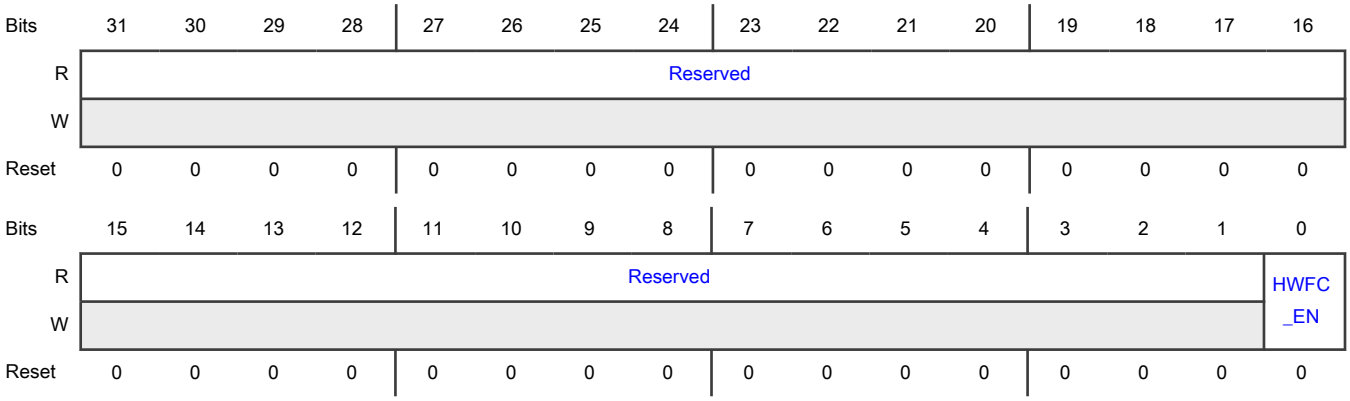
Register	Offset
CNTCR2	C0h

Function

Enables automatic configuration of SYS_CTR via hardware.

SYS_CTR operates using a fixed base frequency. However, the counter can increment at a lower alternate frequency than the base frequency, using a correspondingly larger increment. For example, the counter can increment by 46,875 and run at a frequency of 1/46875 of the base frequency, which is the normal operating frequency. The alternate frequency is a significantly lower operating frequency used to reduce power consumption. The frequency-modes table identifies the two available frequencies. These two frequencies are the base frequency (table entry 0), and the lower and alternate frequency (table entry 1). Writing 1 to CNTCR2[HWFC_EN] enables hardware to automatically switch to the lower, alternate frequency clock after the chip enters low-power mode. When exiting this mode, hardware automatically switches back to the base frequency. If CNTCR2[HWFC_EN] = 0, frequency changes are fully under software control via Counter Control (CNTCR).

Diagram



Fields

Field	Function
31-1 —	Reserved
0 HWFC_EN	Hardware Frequency Change Enable Enables automatic configuration of SYS_CTR via hardware. If this field is 1, it performs the same frequency change request that you can do using CNTCR[FCR0] and CNTCR[FCR1]. 0b - No effect 1b - Same as performed via software

10.8.1.10 Counter ID (CNTID0)

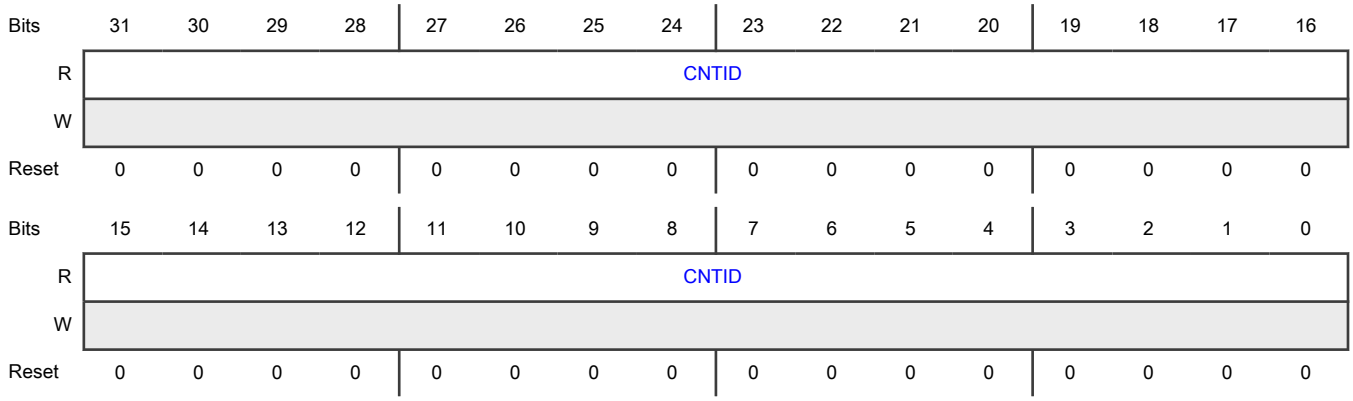
Offset

Register	Offset
CNTID0	FD0h

Function

Indicates the architectural version 0 of SYS_CTR.

Diagram



Fields

Field	Function
31-0 CNTID	Counter Identification Specifies counter ID 0, the architectural version of this system counter.

10.8.2 SYS_CTR_READ register descriptions

These read frame registers read the same values as the control frame registers for the count value and counter ID. They are processed via a separate mechanism from the control frame to allow Nonsecure and User mode accesses.

NOTE

Write or read access to reserved locations does not generate a transfer error.

NOTE

All 32-bit registers allow only 32-bit wide write operations, and are aligned to 32 bits.

10.8.2.1 SYS_CTR_READ memory map

SYS_CTR_READ base address: 442B_0000h

Offset	Register	Width (In bits)	Access	Reset value
8h	Counter Count Value Low (CNTCV0)	32	R	0000_0000h
Ch	Counter Count Value High (CNTCV1)	32	R	0000_0000h
FD0h	Counter ID (CNTID0)	32	R	0000_0000h

10.8.2.2 Counter Count Value Low (CNTCV0)

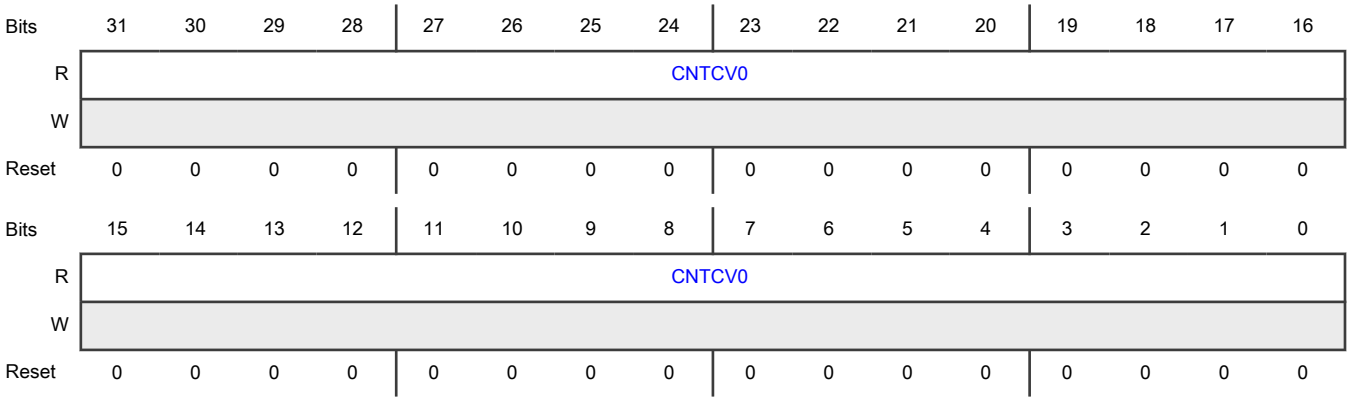
Offset

Register	Offset
CNTCV0	8h

Function

Indicates the current count value bits, 31–0.

Diagram



Fields

Field	Function
31-0 CNTCV0	Counter Count Value Bits [31:0] Indicates the lower 32 bits of the 56-bit system count value.

10.8.2.3 Counter Count Value High (CNTCV1)

Offset

Register	Offset
CNTCV1	Ch

Function

Indicates the current count value bits 63–32.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								CNTCV1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CNTCV1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 CNTCV1	Counter Count Value Bits [55:32] Indicates the upper 24 bits of the 56-bit system count value. Bits 63–56 are always zero.

10.8.2.4 Counter ID (CNTID0)

Offset

Register	Offset
CNTID0	FD0h

Function

Indicates the architectural version 0 of SYS_CTR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CNTID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CNTID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CNTID	Counter Identification Indicates counter ID 0, the architectural version of this system counter.

10.8.3 SYS_CTR_COMPARE register descriptions

Each compare frame:

- Consists of a 256-byte region.
- Has its own compare value and control register.
- Is capable of generating a maskable interrupt.

NOTE

Write or read access to reserved locations does not generate a transfer error.

NOTE

All 32-bit registers allow only 32-bit wide write operations, and are aligned to 32 bits.

10.8.3.1 SYS_CTR_COMPARE memory map

SYS_CTR_COMPARE base address: 442A_0000h

Offset	Register	Width (In bits)	Access	Reset value
20h	Compare Count Value Low (CMPCVL0)	32	RW	0000_0000h
24h	Compare Count Value High (CMPCVH0)	32	RW	0000_0000h
2Ch	Compare Control (CMPCR0)	32	RW	0000_0000h
120h	Compare Count Value Low (CMPCVL1)	32	RW	0000_0000h
124h	Compare Count Value High (CMPCVH1)	32	RW	0000_0000h
12Ch	Compare Control (CMPCR1)	32	RW	0000_0000h
FD0h	Counter ID (CNTID0)	32	R	0000_0000h

10.8.3.2 Compare Count Value Low (CMPCVL0 - CMPCVL1)

Offset

Register	Offset
CMPCVL0	20h
CMPCVL1	120h

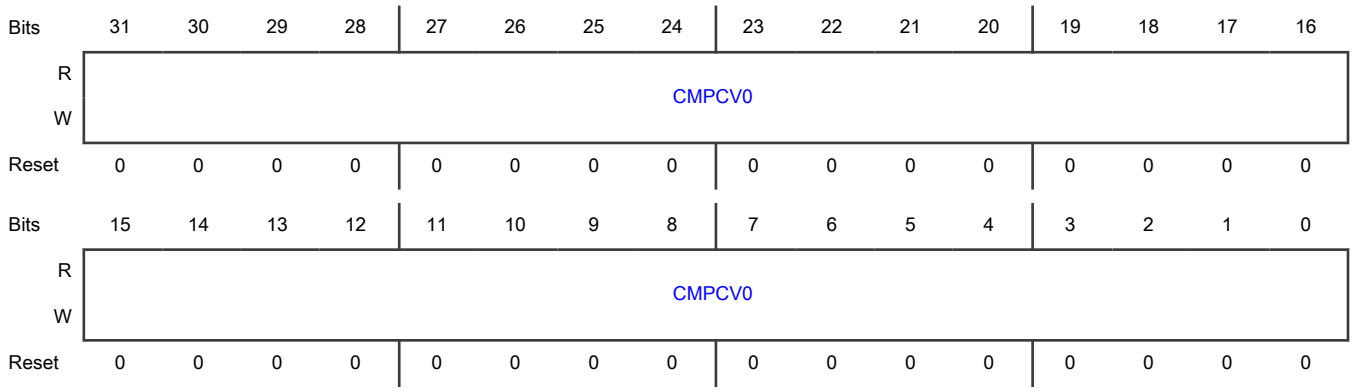
Function

Indicates the current count value bits 31–0.

NOTE

You must write to this set of registers when operating on the base frequency only. Writes to these registers when running on the alternate frequency may have unpredictable results.

Diagram



Fields

Field	Function
31-0 CMPCV0	Compare Count Value Bits [31:0] Specifies the lower 32 bits of the 56-bit compare value.

10.8.3.3 Compare Count Value High (CMPCVH0 - CMPCVH1)

Offset

Register	Offset
CMPCVH0	24h
CMPCVH1	124h

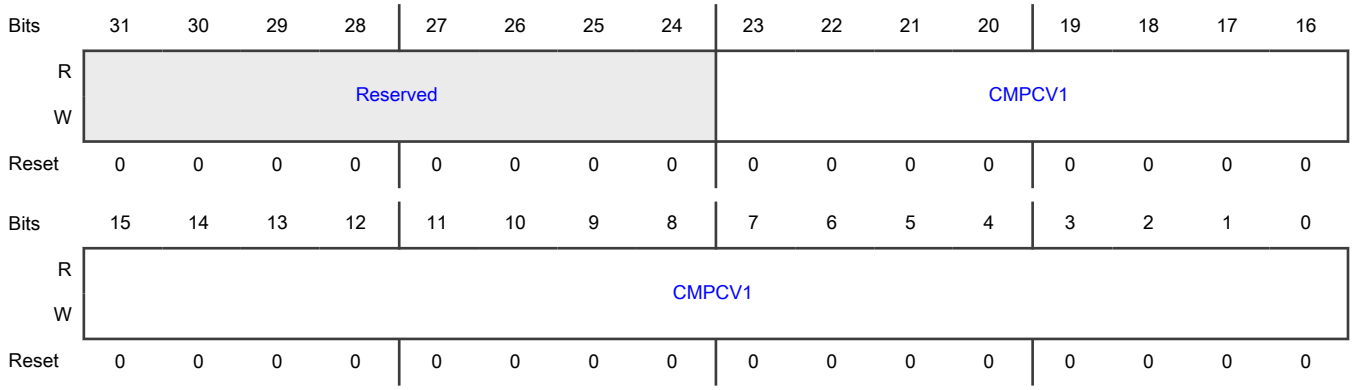
Function

Indicates the current count value bits 63–32.

NOTE

You must write to this set of registers when operating on the base frequency only. Writes to these registers when running on the alternate frequency may have unpredictable results.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 CMPCV1	Compare Count Value Bits [55:32] Specifies the upper 24 bits of the 56-bit compare value. Bits[63:56] are always zero.

10.8.3.4 Compare Control (CMPCR0 - CMPCR1)

Offset

Register	Offset
CMPCR0	2Ch
CMPCR1	12Ch

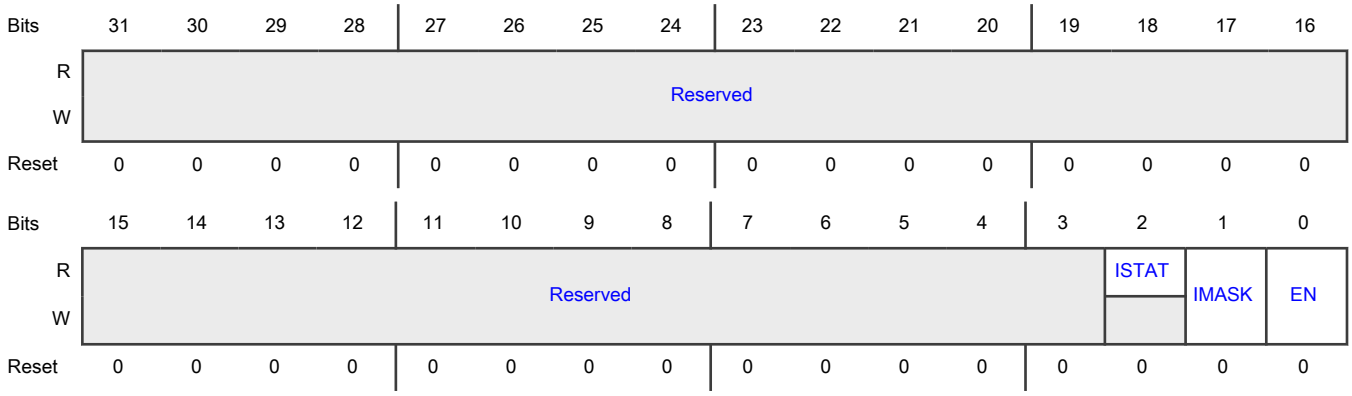
Function

Provides control and status of the compare function. When enabled, CMPCR0[ISTAT] indicates whether the counter value is greater than or equal to the value in CMPCV*n*. The ISTAT equation is:

Value of CMPCR0[ISTAT] = CNTCV*n* ≥ CMPCV*n*

CMPCR0[ISTAT] takes no account of the value of CMPCR0[IMASK]. If CMPCR0[ISTAT] = 1 and CMPCR0[IMASK] = 0, the interrupt request is asserted. Writing 0 to CMPCR0[EN] makes CMPCR0[ISTAT] = 0 and negates the interrupt output signal.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 ISTAT	Compare Interrupt Status Indicates whether the counter value is less than, greater than, or equal to the compare value. 0b - Either less than the compare value or compare is disabled 1b - Greater than or equal to the compare value and compare is enabled
1 IMASK	Interrupt Request Mask Specifies whether the interrupt output signal is masked. 0b - Not masked 1b - Masked
0 EN	Compare Enable Enables the compare function. 0b - Disable 1b - Enable

10.8.3.5 Counter ID (CNTID0)

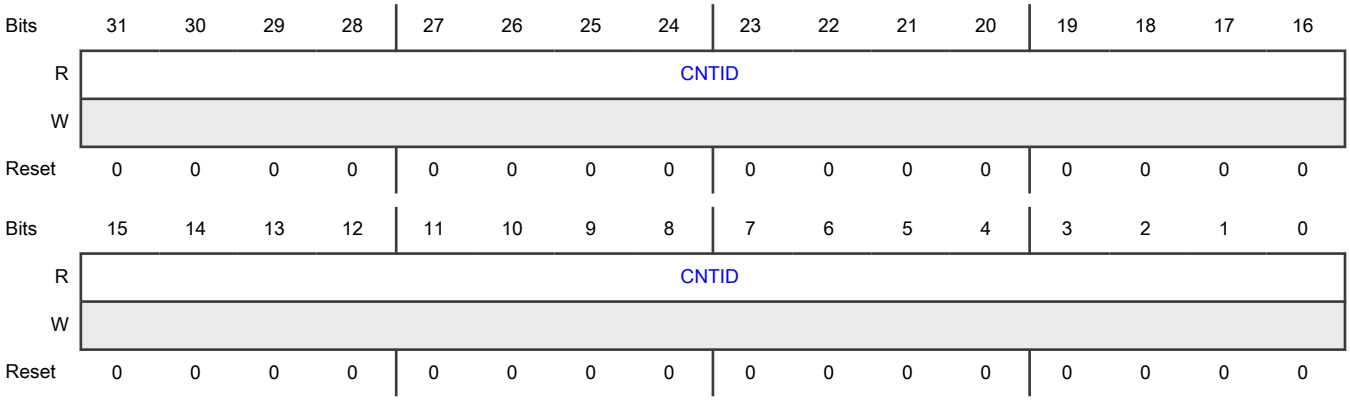
Offset

Register	Offset
CNTID0	FD0h

Function

Indicates the architectural version 0 of SYS_CTR.

Diagram



Fields

Field	Function
31-0 CNTID	Counter Identification Indicates counter ID 0, the architectural version of this system counter.

Chapter 11
Time Stamp Timer (TSTMR)

11.1 Chip-specific TSTMR Information

Table 46. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

11.2 Overview

TSTMR is a free-running, 56-bit incrementing clock cycle counter that requires two 32-bit reads to read the full timestamp value.

TSTMR:

- Resets after every system reset.
- Is accessible through the LOW and HIGH registers.
- Stops only when the clock to TSTMR is disabled.

11.2.1 Block diagram

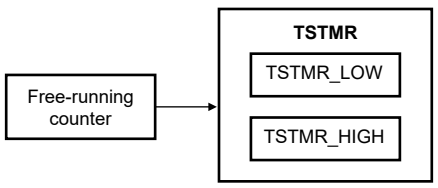


Figure 34. Block diagram

11.2.2 Features

TSTMR has the following features:

- Provides a free-running timer
- Counts the number of clock cycles since the last chip reset

11.3 Functional description

TSTMR starts running after system reset deassertion. You can read the counter at any time to determine software ticks. However, you must follow the read sequence as mentioned in the TSTMR register descriptions for correctly reading TSTMR values.

See the chip-specific TSTMR information for implementation details of this module's instances.

11.3.1 Dual mode

Dual Timer mode is used to implement a second virtual TSTMR for CPU2. The original TSTMR is assumed to be present in the CPU1 domain as a free running timer that resets only when the CPU1 domain power cycles. The CPU2 domain can power cycle independently of the CPU1 domain but it must do that while the CPU1 domain is power cycling. The second virtual TSTMR always runs in sync with the original TSTMR but starts from 0 when the CPU2 domain power cycles.

11.3.2 Clocking

TSTMR uses the 1 MHz chip clock.

11.3.3 Interrupts

TSTMR has no interrupts.

11.4 External signals

TSTMR has no external signals.

11.5 Initialization

You can initialize TSTMR after the chip is powered up. No additional steps or considerations are necessary.

11.6 TSTMR_A memory map and register definition

NOTE

You can read TSTMR registers with 32-bit accesses only.

11.6.1 Accessing registers

You must always perform the following procedure in full to receive consistent timer values:

- Use a 32-bit access to read Timestamp Timer Low (LOW) register.
- Use a 32-bit access to read Timestamp Timer High (HIGH) register.

This procedure is necessary because, after you read LOW, TSTMR freezes the value of HIGH until you read HIGH. If you neglect to read HIGH and attempt to read LOW again, the timestamp value remains unchanged.

11.6.2 TSTMR register descriptions

This section contains detailed register descriptions for the TSTMR registers.

11.6.2.1 TSTMR memory map

TSTMR1.TSTMRA base address: 442C_0000h

TSTMR2.TSTMRA base address: 4248_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Timestamp Timer Low (LOW)	32	R	0000_0000h
4h	Timestamp Timer High (HIGH)	32	R	0000_0000h

11.6.2.2 Timestamp Timer Low (LOW)

Offset

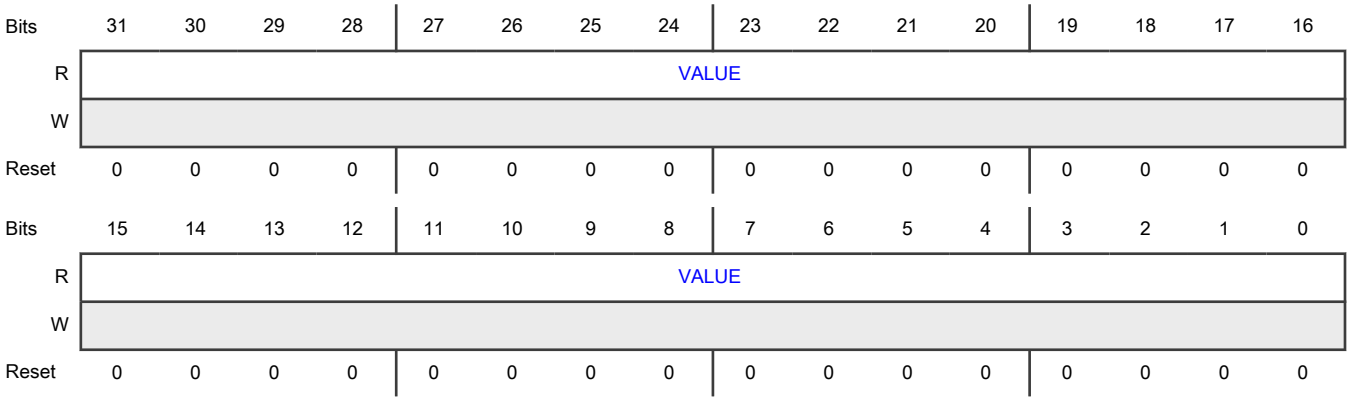
Register	Offset
LOW	0h

Function

Indicates the lower 32 bits of the full 56-bit timestamp value.

See Accessing registers for information on how to use this register.

Diagram



Fields

Field	Function
31-0 VALUE	Timestamp Timer Low Indicates the lower 32 bits of the 56-bit timestamp value. You may not be able to detect the situation of VALUE = 0, because it increases before you can read it.

11.6.2.3 Timestamp Timer High (HIGH)

Offset

Register	Offset
HIGH	4h

Function

Indicates the higher 32 bits of the full 56-bit timestamp value.

See Accessing registers for information on how to use this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								VALUE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 VALUE	Timestamp Timer High Indicates the higher 24 bits of the 56-bit timestamp value. You may not be able to detect the situation of VALUE = 0, because it increases before you can read it.

Chapter 12
System Boot

12.1 Overview

The boot process begins at any Reset where the hardware reset logic forces the boot core (Cortex-M33 core) to begin execution starting from the on-chip boot ROM.

The boot ROM uses the state of the internal register bitfield SBMR2[BOOT_MODE] (which reflects the status of Boot Mode Pins), as well as the state of various fuse settings to determine the boot flow behavior of the device.

The boot ROM code also allows the downloading of programs to be run on the device. An example is a provisioning program that can make further use of the serial connection to provide a boot device with a new image. Typically, the provisioning program is downloaded to the internal RAM and allows to program the boot devices, such as the FlexSPI NOR flash. The boot ROM serial downloader uses LPSPI, LPUART, or a high-speed USB in a non-stream mode connection.

The External Memory Configuration Data (XMCD) allows the boot ROM code to configure the SDRAM connected to the SEMC controller, or the HyperRAM/APMemory PSRAM via the FlexSPI controller from an external program image residing on the boot device. The XMCD aims to simplify the external RAM enablement, and it is a replacement for the legacy Device Configuration Data (DCD).

A key feature of the boot ROM is the ability to perform a secure boot. The boot ROM is also called Advanced High-Assurance Boot (AHAB) which protects the system from executing unauthorized program images. Before the AHAB allows the user image to execute, the image must be signed. The signing process is done during the image build process by the private key holder and the signatures are then included as a part of the final program image. If configured to do so, the AHAB verifies the signatures using the public keys included in the program image. In addition to supporting the digital signature verification to authenticate the program images, encrypted boot is also supported. The encrypted boot can be used to prevent the cloning of the program image directly off the boot device. A secure boot with AHAB can be performed on all boot devices supported on the chip in addition to the serial downloader. The out-of-fab setting for the LIFE_CYCLE is the open configuration, in which the ROM/AHAB performs the image authentication, but all authentication errors are ignored and the image is still allowed to execute.

12.1.1 Features

The main features of the boot ROM include:

- Supports booting from Cortex-M33
- Supports booting from various boot devices
- Supports booting from recovery boot devices once failed booting from the primary boot device
- Supports serial downloader
- Supports external RAM expansion using eXternal Memory Configuration Data (XMCD)
- Supports encrypted boot via AES-CBC
- Supports digital signature and hash function based Advanced High-Assurance Boot (AHAB)
- Supports encrypted execute-in-place (XIP) booting on Serial NOR via FlexSPI interface powered by:
  - On-the-Fly AES Decryption (OTFAD)
  - Inline Encryption Engine (IEE)

  Supports loading NXP-supplied security firmware to expand the security features.

The boot ROM supports the following primary boot devices:

- Serial NOR Flash via FlexSPI
  - Supports booting from either FlexSPI instances (FlexSPI1 or FlexSPI2)
  - Supports booting from either pin groups of the FlexSPI instance

- Supports booting from either PORTA or PORTB.
- Supports Flash Auto-probing based on JESD216
- Supports Encrypted XIP booting via OTFAD or IEE

- Serial NAND Flash via FlexSPI
  - Supports booting from either FlexSPI instances (FlexSPI1 or FlexSPI2)
  - Supports booting from either pin groups of the FlexSPI instance
  - Supports booting from either PORTA or PORTB
- SLC RAWNAND Flash via SEMC
  - Supports FLASH build-in ECC or the SEMC BCH4/BCH8 ECC engine
- SD/eMMC via uSDHC
  - Supports booting from either uSDHC instance
  - Supports booting from SD, SDHC, SDXC card
  - Supports 1/4bit width for SD booting
  - Supports SDR12, SDR25, SDR50 and SDR104 for SD booting
  - Supports booting from eMMC4.5 and higher
  - Supports 4bit SDR, 8bit SDR, 4bit DDR, and 8bit DDR for eMMC booting
  - Supports booting from eMMC boot operation mode (Fast boot)

The boot ROM supports the following recovery boot device:

- SPI NOR/EEPROM via LPSPI
  - Supports booting from LPSPI1, 2, 4 or 5
  - Supports Dual and Quad mode for SPI NOR FLASH booting

The boot ROM supports the following serial downloader peripherals:

- USB-HID via USB1
- LPUART1
- LPSPI1

The boot ROM supports external RAM expansion:

- HyperRAM via FlexSPI
- APMemory via FlexSPI
- SDRAM via SEMC

The boot ROM supports the following digital signature features:

- Supports ECDSA: prime256, secp384, secp521
- Supports RSA PSS: 2048, 3072, 4096
- Supports hashes: SHA-256, SHA-384, SHA-512
- The maximum number of certificate supported: 4
- Supports revoking certificates

12.2 Chip-specific Boot Information

For booting consideration, the boot ROM needs to configure some modules. The settings are documented in this section.

12.2.1 Boot ROM Pinmux

This device has various peripherals supported by the boot ROM.

Table 47. Serial Downloader Peripherals PinMux

Peripheral	Instance	Port (IO function)	PAD	Mode	Note
LPUART	1	LPUART1_TX	GPIO_AON_08	ALT0	This peripherals can be used as serial downloader. Refer to Serial Boot (Serial Downloader) for more information. All serial downloader peripherals can be disabled by the DIS_SERIAL_DWNLD fuse. Each type of serial downloader peripheral can be disabled separately by setting DIS_USB_HID_DWNLD, DIS_SPI_WDNLD and DIS_UART_DWNLD.
		LPUART1_RX	GPIO_AON_09	ALT0	
LPSPI	1	LPSPI1_SCK	GPIO_AON_04	ALT0	
		LPSPI1_PCS0	GPIO_AON_05	ALT0	
		LPSPI1_SDO	GPIO_AON_06	ALT0	
		LPSPI1_SDI	GPIO_AON_07	ALT0	
USB	1	USB1_DN	USB1_DN	-	
		USB1_DP	USB1_DP	-	

Table 48. Boot ROM Peripheral PinMux

Peripheral	Instance	Port (IO function)	PAD	Mode	Note
LPSPI	1	LPSPI1_SCK	GPIO_AON_04	ALT0	Serial NOR/EEPROM connected to one of the LPSPI ports can be used as a recovery device. Refer to Serial NOR/EEPROM Recovery Boot via LPSPI for more information. Recovery device boot is disabled by default. The RECOVERY_BOOT_EN fuse must be blown in order to enable this option. The SPI instance 1 is selected by default, which can be overwritten by the LPSPI_PORT_SEL fuse. PCS2/DATA2 and PCS3/DATA3 are only used in Dual/Quad Mode, which can be selected by the LPSPI_SPEED fuse.
		LPSPI1_PCS0	GPIO_AON_05	ALT0	
		LPSPI1_SDO/DATA0	GPIO_AON_06	ALT0	
		LPSPI1_SDI/DATA1	GPIO_AON_07	ALT0	
		LPSPI1_PCS2/DATA2	GPIO_AON_09	ALT8	
		LPSPI1_PCS3/DATA3	GPIO_AON_03	ALT4	
	2	LPSPI2_SCK	GPIO_AON_22	ALT6	
		LPSPI2_PCS0	GPIO_AON_25	ALT6	
		LPSPI2_SDO/DATA0	GPIO_AON_23	ALT6	
		LPSPI2_SDI/DATA1	GPIO_AON_24	ALT6	
		LPSPI2_PCS2/DATA2	GPIO_AON_26	ALT1	
		LPSPI2_PCS3/DATA3	GPIO_AON_27	ALT1	
	4	LPSPI4_SCK	GPIO_SD_B2_08	ALT4	
		LPSPI4_PCS0	GPIO_SD_B2_09	ALT4	

Table continues on the next page...

Table 48. Boot ROM Peripheral PinMux (continued)

Peripheral	Instance	Port (IO function)	PAD	Mode	Note
		LPSPI4_SDO/DATA0	GPIO_SD_B2_10	ALT4	
		LPSPI4_SDI/DATA1	GPIO_SD_B2_11	ALT4	
		LPSPI4_PCS2/DATA2	GPIO_SD_B2_06	ALT4	
		LPSPI4_PCS3/DATA3	GPIO_SD_B2_05	ALT4	
	5	LPSPI5_SCK	GPIO_AD_28	ALT0	
		LPSPI5_PCS0	GPIO_AD_29	ALT0	
		LPSPI5_SDO/DATA0	GPIO_AD_30	ALT0	
		LPSPI5_SDI/DATA1	GPIO_AD_31	ALT0	
		LPSPI5_PCS2/DATA2	GPIO_AD_26	ALT2	
		LPSPI5_PCS3/DATA3	GPIO_AD_25	ALT2	
SEMC NAND/SDRAM	N/A	**SEMC_DATA00**	**GPIO_EMC_B1_00**	ALT0	Parallel NAND flash connected to the SEMC is a primary boot option. Refer to Parallel NAND Flash Boot via SEMC for more information. SDRAM connected to the SEMC is a memory expansion option. Refer to SDRAM support for more information. By default ROM reads from the 8-bit NAND device via the **bold** pins. The "I/O Port Size" fuse must be blown to enable 16-bit NAND device support. The PCS_Selection fuse must be blown to select other PCS configurations.
		SEMC_DATA01	**GPIO_EMC_B1_01**	ALT0	
		SEMC_DATA02	**GPIO_EMC_B1_02**	ALT0	
		SEMC_DATA03	**GPIO_EMC_B1_03**	ALT0	
		SEMC_DATA04	**GPIO_EMC_B1_04**	ALT0	
		SEMC_DATA05	**GPIO_EMC_B1_05**	ALT0	
		SEMC_DATA06	**GPIO_EMC_B1_06**	ALT0	
		SEMC_DATA07	**GPIO_EMC_B1_07**	ALT0	
		SEMC_DATA08	GPIO_EMC_B1_30	ALT0	
		SEMC_DATA09	GPIO_EMC_B1_31	ALT0	
		SEMC_DATA10	GPIO_EMC_B1_32	ALT0	
		SEMC_DATA11	GPIO_EMC_B1_33	ALT0	
		SEMC_DATA12	GPIO_EMC_B1_34	ALT0	
		SEMC_DATA13	GPIO_EMC_B1_35	ALT0	
		SEMC_DATA14	GPIO_EMC_B1_36	ALT0	

Table continues on the next page...

Table 48. Boot ROM Peripheral PinMux (continued)

Peripheral	Instance	Port (IO function)	PAD	Mode	Note
		SEMC_DATA15	GPIO_EMC_B1_37	ALT0	
		SEMC_ADDR09	**GPIO_EMC_B1_18**	ALT0	
		SEMC_ADDR11	**GPIO_EMC_B1_19**	ALT0	
		SEMC_ADDR12	**GPIO_EMC_B1_20**	ALT0	
		SEMC_BA1	**GPIO_EMC_B1_22**	ALT0	
		SEMC_CSX0	**GPIO_EMC_B1_41**	ALT0	
		SEMC_CSX1	GPIO_B1_00	ALT2	
		SEMC_CSX2	GPIO_B1_01	ALT2	
		SEMC_CSX3	GPIO_B2_00	ALT1	
uSDHC	1	USDHC1_CD_B	GPIO_AD_32	ALT4	eMMC or SD connected to one of the USDHC ports is a primary boot option. Refer to eMMC/SD Boot via uSDHC for more information on USDHC boot. The device type, instance, boot width are fixed when BOOT MODE is selected by BOOT MODE Pins. Boot speed still can be selected by the SD_SPEED or MMC_SPEED fuse. The device type, instance, boot width and boot speed are determined by fuse when BOOT MODE is selected by the BOOT_MODE_FROM_FUSE fuse.
		USDHC1_VSELECT	GPIO_AD_34	ALT4	
		USDHC1_RESET_B	GPIO_AD_35	ALT4	
		USDHC1_CMD	GPIO_SD_B1_00	ALT0	
		USDHC1_CLK	GPIO_SD_B1_01	ALT0	
		USDHC1_DATA0	GPIO_SD_B1_02	ALT0	
		USDHC1_DATA1	GPIO_SD_B1_03	ALT0	
		USDHC1_DATA2	GPIO_SD_B1_04	ALT0	
		USDHC1_DATA3	GPIO_SD_B1_05	ALT0	
	2	USDHC2_VSELECT	GPIO_AD_29	ALT9	
		USDHC2_RESET_B	GPIO_SD_B2_06	ALT0	
		USDHC2_CMD	GPIO_SD_B2_05	ALT0	
		USDHC2_CLK	GPIO_SD_B2_04	ALT0	
		USDHC2_DATA0	GPIO_SD_B2_03	ALT0	
		USDHC2_DATA1	GPIO_SD_B2_02	ALT0	
		USDHC2_DATA2	GPIO_SD_B2_01	ALT0	

Table continues on the next page...

Table 48. Boot ROM Peripheral PinMux (continued)

Peripheral	Instance	Port (IO function)	PAD	Mode	Note
		USDHC2_DATA3	GPIO_SD_B2_00	ALT0	
		USDHC2_DATA4	GPIO_SD_B2_08	ALT0	
		USDHC2_DATA5	GPIO_SD_B2_09	ALT0	
		USDHC2_DATA6	GPIO_SD_B2_10	ALT0	
		USDHC2_DATA7	GPIO_SD_B2_11	ALT0	
FlexSPI1 NOR/ NAND/RAM (Primary Pin Group)	1	FLEXSPI1_A_DQS (Primary DQS pin)	GPIO_B2_07	ALT7	If QSPI, HyperFlash, or Octal memory attached to FlexSPI is a primary boot option. Refer to Serial NOR Flash Boot via FlexSPI for more information. The ROM will read the 512-byte FlexSPI NOR configuration parameters described in FlexSPI Serial NOR Flash Boot Operation. If Serial NAND memory attached to FlexSPI is a primary boot option. Refer to Serial NAND Flash Boot via FlexSPI for more information. If HyperRAM or APMemory attached to FlexSPI is a memory expansion option. Refer to FlexSPI RAM Configuration Block for more information. This pin group is used for FlexSPI NOR/NAND boot if: • The FLEXSPI_INSTANCE fuse/ boot pin is 0. • The FLEXSPI_PIN_GROUP_SEL fuse bit is 0. The fuse FLEXSPI_DQS_PIN_SEL determines the DQS pin option. By default, ROM accesses the QSPI NOR/NAND via the **bold** pins. This default behavior can be overridden by setting the FLASH_CONNECTION_SEL fuse
		FLEXSPI1_A_DQS (Secondary DQS Pin)	GPIO_SD_B2_12_DUMMY	ALT0	
		FLEXSPI1_A_SS0_ B	**GPIO_B2_09**	ALT7	
		FLEXSPI1_A_SS1_B	GPIO_B2_01	ALT6	
		FLEXSPI1_A_SCLK	**GPIO_B2_08**	ALT7	
		FLEXSPI1_A_SCLK_B	**GPIO_B2_02**	ALT6	
		FLEXSPI1_A_DATA 0	**GPIO_B2_10**	ALT7	
		FLEXSPI1_A_DATA 1	**GPIO_B2_11**	ALT7	
		FLEXSPI1_A_DATA 2	GPIO_B2_12	ALT7	
		FLEXSPI1_A_DATA 3	GPIO_B2_13	ALT7	
		FLEXSPI1_A_DATA4	GPIO_B2_03	ALT7	
		FLEXSPI1_A_DATA5	GPIO_B2_04	ALT7	
		FLEXSPI1_A_DATA 6	GPIO_B2_05	ALT7	
		FLEXSPI1_A_DATA 7	GPIO_B2_06	ALT7	
		FLEXSPI1_B_DQS (Primary DQS pin)	GPIO_SD_B2_05	ALT1	
		FLEXSPI1_B_DQS (Secondary DQS Pin)	GPIO_SD_B2_12_DUMMY	ALT1	
		FLEXSPI1_B_SS0_B	GPIO_SD_B2_06	ALT1	

Table continues on the next page...

Table 48. Boot ROM Peripheral PinMux (continued)

Peripheral	Instance	Port (IO function)	PAD	Mode	Note
		FLEXSPI1_B_SS1_B	GPIO_SD_B2_04	ALT1	**NOTE** Primary pin group is not available on all packages. For example, BGA144 does not include the GPIO_B1/ GPIO_B2 pin group.
		FLEXSPI1_B_SCLK	GPIO_SD_B2_07	ALT1	
		FLEXSPI1_B_DATA 0	GPIO_SD_B2_08	ALT1	
		FLEXSPI1_B_DATA1	GPIO_SD_B2_09	ALT1	
		FLEXSPI1_B_DATA 2	GPIO_SD_B2_10	ALT1	
		FLEXSPI1_B_DATA 3	GPIO_SD_B2_11	ALT1	
		FLEXSPI1_B_DATA 4	GPIO_SD_B2_00	ALT1	
		FLEXSPI1_B_DATA 5	GPIO_SD_B2_01	ALT1	
		FLEXSPI1_B_DATA 6	GPIO_SD_B2_02	ALT1	
		FLEXSPI1_B_DATA 7	GPIO_SD_B2_03	ALT1	
FlexSPI1 NOR/ NAND/RAM (Secondary Pin Group)	1	FLEXSPI1_B_DQS (Primary DQS pin)	GPIO_B1_03	ALT7	If QSPI, Hyperflash, or Octal memory attached to FlexSPI is a primary boot option. Refer to Serial NOR Flash Boot via FlexSPI for more information. The ROM will read the 512-byte FlexSPI NOR configuration parameters described in FlexSPI Serial NOR Flash Boot Operation. If Serial NAND memory attached to FlexSPI is a primary boot option. Refer to Serial NAND Flash Boot via FlexSPI for more information. If HyperRAM or APMemory attached to FlexSPI is a memory expansion option. Refer to FlexSPI RAM Configuration Block for more information. This pin group is used for FlexSPI NOR/NAND boot if: • FLEXSPI_INSTANCE fuse/ boot pin is 0. • FLEXSPI_PIN_GROUP_SEL fuse bit is 1. Only PORTB is supported via the 2nd Pin group.
		FLEXSPI1_B_DQS (Secondary DQS Pin)	GPIO_SD_B2_12_DU MMY	ALT1	
		FLEXSPI1_B_SS0_ B	**GPIO_B1_04**	ALT7	
		FLEXSPI1_B_SS1_ B	GPIO_B1_02	ALT7	
		FLEXSPI1_B_SCLK	**GPIO_B1_05**	ALT7	
		FLEXSPI1_B_DATA 0	**GPIO_B1_13**	ALT7	
		FLEXSPI1_B_DATA 1	**GPIO_B1_12**	ALT7	
		FLEXSPI1_B_DATA 2	GPIO_B1_11	ALT7	
		FLEXSPI1_B_DATA 3	GPIO_B1_10	ALT7	
		FLEXSPI1_B_DATA4	GPIO_B1_09	ALT7	
		FLEXSPI1_B_DATA5	GPIO_B1_08	ALT7	
		FLEXSPI1_B_DATA 6	GPIO_B1_07	ALT7	
		FLEXSPI1_B_DATA 7	GPIO_B1_06	ALT7	

Table continues on the next page...

Table 48. Boot ROM Peripheral PinMux (continued)

Peripheral	Instance	Port (IO function)	PAD	Mode	Note
					The Maximum Flash frequency supported on this port is 100MHz. By default, ROM accesses the QSPI NOR/NAND via the **bold** pins. This default behavior can be overridden by setting the FLASH_CONNECTION_SEL fuse.
FlexSPI2 NOR/ NAND/RAM (Primary Pin Group)	2	FLEXSPI2_A_DQS (Primary DQS pin)	GPIO_AON_21	ALT8	If QSPI, Hyperflash, or Octal memory attached to FlexSPI is a primary boot option. Refer to Serial NOR Flash Boot via FlexSPI for more information. The ROM will read the 512-byte FlexSPI NOR configuration parameters described in FlexSPI Serial NOR Flash Boot Operation. If Serial NAND memory attached to FlexSPI is a primary boot option. Refer to Serial NAND Flash Boot via FlexSPI for more information. If HyperRAM or APMemory attached to FlexSPI is a memory expansion option. Refer to FlexSPI RAM Configuration Block for more information. This pin group is used for FlexSPI NOR/NAND boot if: • FLEXSPI_INSTANCE fuse/ boot pin is 1. • FLEXSPI_PIN_GROUP_SEL fuse bit is 0. By default, ROM accesses the QSPI NOR/NAND via the **bold** pins. This default behavior can be overridden by setting the FLASH_CONNECTION_SEL fuse.
		FLEXSPI2_A_DQS (Secondary DQS Pin)	GPIO_AON_28_DUMMY	ALT0	
		FLEXSPI2_A_SS0_ B	**GPIO_AON_22**	ALT0	
		FLEXSPI2_A_SS1_B	GPIO_AON_20	ALT1	
		FLEXSPI2_A_SCLK	**GPIO_AON_23**	ALT0	
		FLEXSPI2_A_DATA 0	**GPIO_AON_24**	ALT0	
		FLEXSPI2_A_DATA 1	**GPIO_AON_25**	ALT0	
		FLEXSPI2_A_DATA 2	GPIO_AON_26	ALT0	
		FLEXSPI2_A_DATA 3	GPIO_AON_27	ALT0	
		FLEXSPI2_B_DQS (Primary DQS pin)	GPIO_AON_20	ALT0	
		FLEXSPI2_B_DQS (Secondary DQS Pin)	GPIO_AON_28_DUMMY	ALT1	
		FLEXSPI2_B_SS0_B	GPIO_AON_21	ALT0	
		FLEXSPI2_B_SCLK	GPIO_AON_19	ALT0	
		FLEXSPI2_B_DATA 0	GPIO_AON_18	ALT0	
		FLEXSPI2_B_DATA 1	GPIO_AON_17	ALT0	
		FLEXSPI2_B_DATA 2	GPIO_AON_16	ALT0	
		FLEXSPI2_B_DATA 3	GPIO_AON_15	ALT0	

Table continues on the next page...

Table 48. Boot ROM Peripheral PinMux (continued)

Peripheral	Instance	Port (IO function)	PAD	Mode	Note
					NOTE The Maximum Flash frequency supported on this port (GPIO_AON) is 100MHz.
FlexSPI2 NOR/ NAND/RAM (Secondary Pin Group)	2	FLEXSPI2_A_DQS (Primary DQS pin)	GPIO_EMC_B1_40	ALT3	If QSPI memory attached to FlexSPI is a primary boot option. Refer to Serial NOR Flash Boot via FlexSPI for more information. The ROM will read the 512-byte FlexSPI NOR configuration parameters described in FlexSPI Serial NOR Flash Boot Operation. If Serial NAND memory attached to FlexSPI is a primary boot option. Refer to Serial NAND Flash Boot via FlexSPI for more information. If HyperRAM or APMemory attached to FlexSPI is a memory expansion option. Refer to FlexSPI RAM Configuration Block for more information. This pin group is used for FlexSPI NOR/NAND boot if: • FLEXSPI_INSTANCE fuse/ boot pin is 1. • FLEXSPI_PIN_GROUP_SEL fuse bit is 1. By default, ROM accesses the QSPI NOR/NAND via the **bold** pins. This default behavior can be overridden by setting the FLASH_CONNECTION_SEL fuse. **NOTE** If the GPIO_AON_28_DUMMY pin is used for DQS, the clock cannot exceed 100MHz.
		FLEXSPI2_A_DQS (Secondary DQS Pin)	GPIO_AON_28_DUMMY	ALT0	
		FLEXSPI2_A_SS0_ B	**GPIO_EMC_B1_39**	ALT3	
		FLEXSPI2_A_SS1_ B	GPIO_EMC_B1_26	ALT3	
		FLEXSPI2_A_SCLK	**GPIO_EMC_B1_41**	ALT3	
		FLEXSPI2_A_DATA 0	**GPIO_EMC_B1_35**	ALT3	
		FLEXSPI2_A_DATA 1	**GPIO_EMC_B1_36**	ALT3	
		FLEXSPI2_A_DATA 2	GPIO_EMC_B1_37	ALT3	
		FLEXSPI2_A_DATA 3	GPIO_EMC_B1_38	ALT3	

Table continues on the next page...

Table 48. Boot ROM Peripheral PinMux (continued)

Peripheral	Instance	Port (IO function)	PAD	Mode	Note
FLEXSPI2		FLEXSPI2_B_DQS (Primary DQS pin)	GPIO_EMC_B1_29	ALT3	
		FLEXSPI2_B_DQS (Secondary DQS pin)	GPIO_EMC_B1_32	ALT1	
		FLEXSPI2_B_SS0_B	GPIO_EMC_B1_28	ALT3	
		FLEXSPI2_B_SS1_B	GPIO_EMC_B1_27	ALT3	
		FLEXSPI2_B_SCLK	GPIO_EMC_B1_34	ALT3	
		FLEXSPI2_B_DATA 0	GPIO_EMC_B1_33	ALT3	
		FLEXSPI2_B_DATA 1	GPIO_EMC_B1_32	ALT3	
		FLEXSPI2__B_DATA 2	GPIO_EMC_B1_31	ALT3	
		FLEXSPI2__B_DATA 3	GPIO_EMC_B1_30	ALT3	
FlexSPI RESET	-	GPIO5_IO04 (Primary RESET Pin)	GPIO_SD_B1_00	ALT5	The FlexSPI RESET pin is used as the dedicated RESET pin to restore the Serial NOR/ NAND device connected to the FlexSPI interface. The FlexSPI RESET pin is enabled if the RESET_PIN_EN fuse bit is blown. The pin option is selected by RESET_PIN_SEL fuse bit.
		GPIO3_IO08 (Secondary RESET Pin)	GPIO_EMC_B1_40	ALT5	

NOTE

1. The boot ROM always tries to access Serial NOR/NAND flash using SS0 as the chip selection signal, as well as PORTA by default. It can access the Serial NOR/NAND device via CS0 and PORTB, if the xSPI_NOR_CONNECTION_SEL/xSPI_NAND_CONNECTION_SEL fuse is blown with a value of 1.
2. For Octal NOR/NAND, connected to FlexSPI2, the upper 4-bit pads must connect to the PORTB_DATA pads.
3. PORTB_SCLK serves as PORTA_SCLK_B for the 1.8V HyperFLASH/HyperRAM.
4. A_SS1/B_SS1 is not supported by default.
5. The FlexSPI RESET pin is not used by default. This feature can be enabled by the xSPI_NOR_RESET_TYPE/xSPI_NAND_RESET_TYPE and xSPI_RESET_PIN_SEL fuses.

12.2.2 Boot Fuse List

This following table is a comprehensive list of the configuration parameters that the ROM uses. For additional details, please refer to the related sections.

Table 49. Boot Fuses

Word Number	Fuse	Owner	Field	Definition	Shipped value
24	BOOT_CFG0	OEM	[15-0]	Boot flow configuration Refer to Boot Flow Fuse Configuration for more information.	0
			[23-16]	Recovery boot configuration Refer to Serial NOR FUSE Configuration for more information.	0
			[31-24]	MISC configuration0 Refer to MISC Fuse Configuration for more information.	0
25	BOOT_CFG1	OEM	[31-0]	FlexSPI NOR Flash boot configuration Refer to Serial NOR FUSE Configuration for more information.	0
26	BOOT_CFG2	OEM	[31-0]	FlexSPI common configuration ReferSerial NOR FUSE Configuration and Serial NAND FUSE Configuration for more information.	0
27	BOOT_CFG3	OEM	[31-0]	FlexSPI NAND Flash boot configuration Refer to Serial NAND FUSE Configuration for more information.	0
28	BOOT_CFG4	OEM	[31-0]	eMMC boot configuration Refer to eMMC/SD Device FUSE Configuration for more information.	0
29	BOOT_CFG5	OEM	[31-0]	SEMC SLC RAW NAND boot configuration Refer to Parallel NAND eFuse Configuration for more information.	0
30	BOOT_CFG6	OEM	[31-0]	SD boot configuration Refer to eMMC/SD Device FUSE Configuration for more information.	0
31	BOOT_CFG7	OEM	[7-0]	FlexSPI NOR Flash boot configuration2	0
			[23-8]	MISC configuration1 Refer to MISC Fuse Configuration for more information.	0
32	BOOT_CFG8	OEM	[31-0]	MISC configuration2	0

Table continues on the next page...

Table 49. Boot Fuses (continued)

Word Number	Fuse	Owner	Field	Definition	Shipped value
				Refer to MISC Fuse Configuration for more information.	
33	BOOT_CFG9	OEM	[31-0]	MISC configuration3 (Reserved)	0
34	BOOT_CFG10	OEM	[31-0]	MISC configuration4 (Reserved)	0
35	BOOT_CFG11	OEM	[31-0]	MISC configuration5 (Reserved)	0
312	COM_DEVICE_ID_CFG0	OEM	[15:0]	USB_VID[15:0]	0
			[31:16]	USB1_PID[15:0]	
327	OEM_SW_CFG	OEM	[3:2]	CM33_TCM_CFG[1:0]	0
			[6:4]	CM7_TCM_CFG[2:0]	

12.2.3 Boot Block Activation

The boot ROM affects a number of different hardware blocks which are activated and play a vital role in the boot flow.

The ROM configures and uses the blocks listed in Table 50 during the boot process.

NOTE

The blocks actually used depend on the boot mode and the boot device selections.

Table 50. Boot Block List

Domain	Block(s)	Description
M7 Domain	Cortex-M7	Cortex-M7 Core and SCS
CCMSRCGPC Domain	SRC	System Reset Controller
AON Domain	Cortex-M33	Cortex-M33 Core and SCS
	WDOG1, WDOG2	Watch Dog
	IOMUXC	I/O Multiplexer Control
	LPUART1	Low Power UART
	MU	Message Unit
	LPSPI1, LPSPI2	Low Power SPI
	LPIT1, LPIT2	Periodic Interrupt Timer
	PMU	Power Management Unit
	XTALOSC	Crystal Oscillator
	CCM	Clock Control Module
	FlexSPI2 OTFAD	Flexible SPI Interface with On-the-Fly AES Decryption

Table continues on the next page...

Table 50. Boot Block List (continued)

Domain	Block(s)	Description
WAKEUP Domain	EDMA4	Enhanced DMA
	LPSPI4, LPSPI5	Low Power SPI
	GPIO4, GPIO5, GPIO6	General Purpose I/O
	WDOG3, WDOG4, WDOG5	Watch Dog
WAKEUP Domain	SEMC	Smart External Memory Controller
WAKEUP Domain	FlexSPI1 OTFAD	Flexible SPI Interface with On-the-Fly AES Decryption
MEGA Domain	uSDHC1, uSDHC2	Ultra-Secure Digital Host Controller
	USB1	USB

12.3 Boot Flow

This section describes the entire boot flow, and the related configurations to control the flows.

12.3.1 High-level Boot Sequence

Once the boot core starts from the on-chip boot ROM, the boot ROM performs the hardware initialization, loads the program image from the chosen boot device, performs the image validation using the AHAB library, and then jumps to an address derived from the program image. If an error occurs during the boot, the boot ROM jumps to the next boot alternative. A secure boot using the AHAB is possible in all the boot alternatives. After all boot alternatives have failed, the boot ROM jumps to the Serial Downloader.

Figure 35 shows the high-level boot ROM code flow.

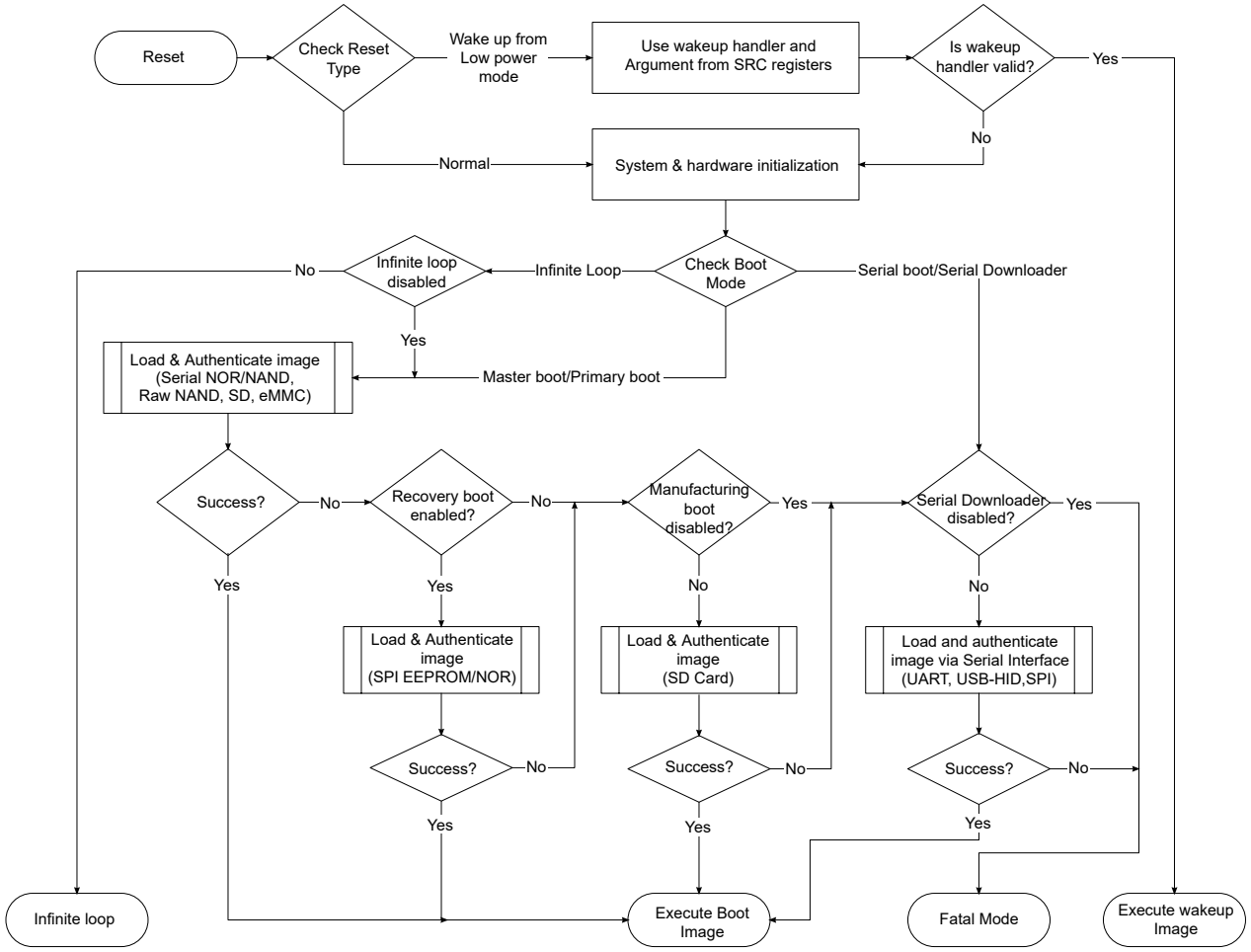


Figure 35. High-level Boot Flow

12.3.2 Boot Flow Fuse Configuration

The following table is a list of the configuration parameters that the ROM uses for boot flow control.

Table 51. Boot Flow Fuse Configuration

Fuse	Definitions	Settings	Shipped Value
BOOT_CFG0[2:0]	BOOT_MODE_FROM_FUSE	0 – FlexSPI NOR Flash 1 – FlexSPI NAND Flash 2 – eMMC 3 – SD 4 – SEMC SLC RAW NAND Flash Refer to Boot Mode Fuse Settings for more information	0

Table continues on the next page...

Table 51. Boot Flow Fuse Configuration (continued)

Fuse	Definitions	Settings	Shipped Value
BOOT_CFG0[3]	Reserved	Must be 0	0
BOOT_CFG0[4]	RECOVERY_BOOT_EN	0 – Recovery boot is disabled 1 – Recovery boot is enabled	0
BOOT_CFG0[5]	DIS_SDMMC_MFG_BOOT	0 – Manufacturing boot is enabled 1 – Manufacturing boot is disabled	0
BOOT_CFG0[6]	BT_FUSE_SEL	0 – Fuse configuration is invalid for ROM booting 1 – Fuse configuration is valid for ROM booting	0
BOOT_CFG0[7]	FORCE_BT_FROM_FUSE	0 – BOOT MODE is determined by BOOT MODE pins 1 – BOOT MODE is determined by BOOT_MODE_FROM_FUSE.	0
BOOT_CFG0[9-8]	Reserved	Must be 0	0
BOOT_CFG0[10]	DIS_UART_DWNLD	0 – UART serial downloader is enabled 1 – UART serial downloader is disabled	0
BOOT_CFG0[11]	DIS_SPI_DWNLD	0 – SPI serial downloader is enabled 1 – SPI serial downloader is disabled	0
BOOT_CFG0[12]	DIS_USB_HID_DWNLD	0 – USB-HID serial downloader is enabled 1 – USB-HID serial downloader is disabled	0
BOOT_CFG0[13]	DIS_INFINITE_LOOP	0 – Infinite loop is enabled 1 – Infinite loop is disabled	0
BOOT_CFG0[14]	DIS_BT_MODE_API	0 – BOOT MODE from ROM API call is allowed 1 – BOOT MODE from ROM API call is ignored Refer to Enter Bootloader API for more information.	0
BOOT_CFG0[15]	DIS_SERIAL_DWNLD	0 – All serial downloaders are enabled 1 – All serial downloaders are disabled.	0

12.3.3 Boot Modes

The boot ROM determines the boot mode via the combination of the following settings:

- ROM API calls
- Fuse settings: DIS_BT_MODE_API, FORCE_BT_FROM_FUSE, BT_FUSE_SEL, BOOT_MODE_FROM_FUSE, DIS_INFINTE_LOOP.
- BOOT_MODE Pins

Figure 36 depicts the boot mode determination logic.

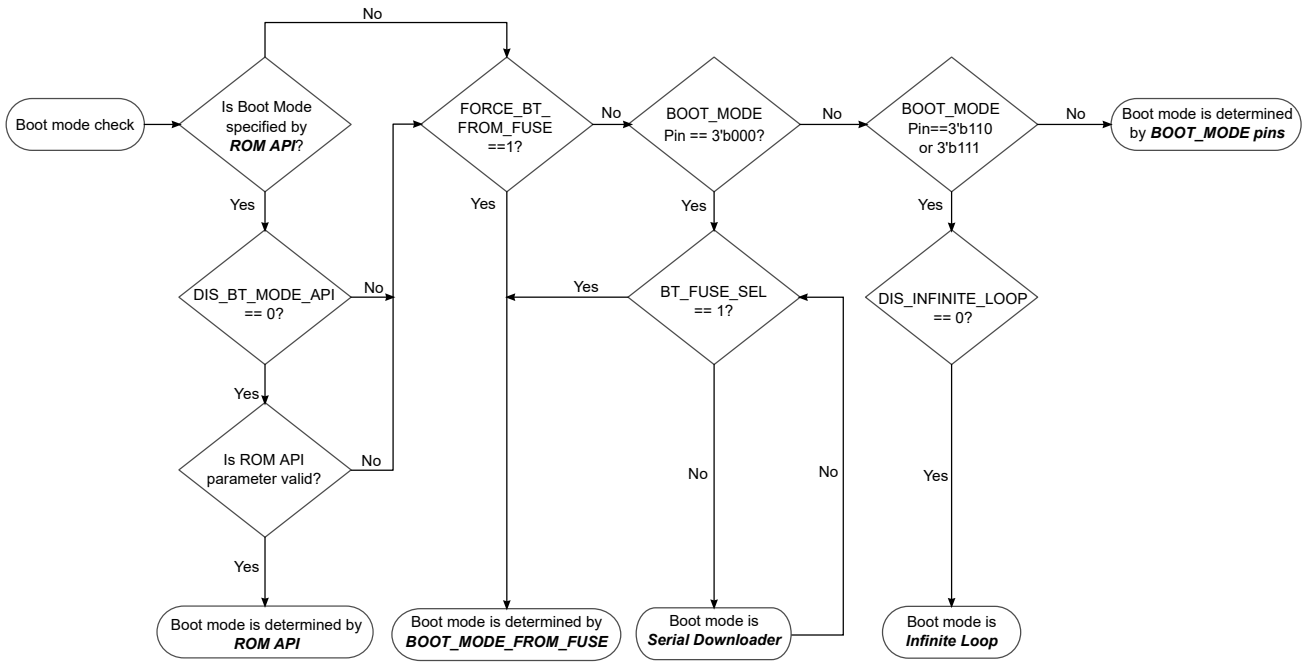


Figure 36. Boot Mode Determination

12.3.3.1 Boot Mode Pin Settings

The device has 3 boot mode pins. The boot mode is selected based on the binary value stored in the internal register field SBMR2[BOOT_MODE], which is initialized by sampling the BOOT_MODE[2:0] pin inputs on the rising edge of the POR_B. After these inputs are sampled, their subsequent state does not affect the contents of SBMR2[BOOT_MODE]. Table 52 shows the boot mode pin definitions.

Table 52. Boot Mode Pin Settings

BOOT_MODE[2:0]	Boot Type
000	Boot From Internal Fuses
001	Serial Downloader
010	eMMC 8bit via uSDHC2[1]
011	SD 4bit via uSDHC1[1]
100	Serial NOR Flash with JESD216 via FlexSPI1, Primary group, PortA[2]

Table continues on the next page...

Table 52. Boot Mode Pin Settings (continued)

BOOT_MODE[2:0]	Boot Type
101	Serial NAND Flash with 2K Page via FlexSPI1, Primary group, PortA[2]
110	Infinite Loop Modes
111	Test Mode[3]

1. Boot device, instance and bus width are fixed. The other options can be set by the values in the BOOT_CFG fuse words.
2. Boot device is fixed. The other options can be overwritten by the values in the BOOT_CFG fuse words, including instance, pin group, port, etc.
3. Test Mode is reserved for NXP usage only.

12.3.3.2 Boot Mode Fuse Settings

When booting from internal fuses, additional boot modes and options are supported by the boot ROM. The boot mode is selected via the BOOT_MODE_FROM_FUSE fuse.Table 53 shows the boot mode fuse definitions.

Table 53. Boot Mode Fuse Settings

BOOT_MODE_FROM_FUSE[2:0]	Boot Type
000	Serial NOR Flash via FlexSPI
001	Serial NAND Flash via FlexSPI
010	eMMC via uSDHC
011	SD via uSDHC
100	SLC RAW NAND via SEMC
101	Serial downloader
110	Invalid
111	Invalid

Additional boot device options can be set via the BOOT_CFG fuse words. Refer to the fuse configuration sections of each boot device for more information.

12.3.3.3 Boot From Internal Fuse

A value of 000b in the SBMR[BOOT_MODE] selects Boot From Internal Fuses.

In this mode, the boot ROM uses the BOOT_CFG fuse settings.

If set to Boot From Internal Fuses, the boot flow is controlled by the BT_FUSE_SEL fuse value. If BT_FUSE_SEL = 0, indicating that the boot device (for example, Flash, fuse) was not programmed yet, the boot flow jumps directly to the Serial Downloader. If BT_FUSE_SEL = 1, the normal boot flow is followed, where the ROM attempts to boot from the selected boot device.

Setting the BT_FUSE_SEL=0 forces the ROM code to jump directly to the Serial Downloader. This allows a bootloader to be downloaded which can then provision the boot device with a program image and blow the BT_FUSE_SEL and the other boot configuration fuses. After the reset, the boot ROM code determines that the BT_FUSE_SEL is blown (BT_FUSE_SEL = 1) and the ROM code performs an internal boot according to the new fuse settings. This allows the user to set BOOT_MODE[2:0]=000b

on a production device and burn the fuses on the same device (by forcing the entry to the Serial Downloader), without changing the value of the BOOT_MODE[2:0] or the pullups/pulldowns on the BOOT_MODE pins.

The first time a board is used, the default fuses may be configured incorrectly for the hardware on the platform. In such case, the Boot ROM code may try to boot from a device that does not exist. This may cause an electrical/logic violation on some pads.

12.3.3.4 Serial Downloader

The Serial Downloader provides a means to download a Program Image to the on-chip RAM over USB, UART, and SPI connection. In this mode, typically a host PC can communicate to the ROM bootloader using the MCU-BOOT protocol. Serial downloader and the protocol are discussed in Serial Boot (Serial Downloader).

12.3.3.5 Infinite Loop

ROM will run into an infinite loop instead booting from the selected boot devices or running into serial downloader. This gives the user the benefit of debugging or developing their applications, during which booting is not required. The infinite loop mode can be disabled by setting the DIS_INFINITE_LOOP fuse.

12.3.4 Error Recovery Fast Boot

When the Cortex-M33 is warm reset without resetting the rest of the SoC, the Cortex-M33 Boot ROM jumps straight to the resume entry pointed by the INITSVTOR or INITNSVTOR register in BLK_CTRL_S_AONMIX (See the chip's Security Reference Manual for details), after validating the resume entry, as well as Stack Pointer and Program Counter pointed by the resume entry are valid.

This flow can be used to recover image execution quickly when an issue occurs and only the Cortex-M33 reset is needed. It can only be used after a standard boot flow, when image authentication has already been done.

To utilize recovery fast boot flow, the BLK_CTRL_S_AON INITSVTOR or INITNSVTOR register has to be filled in with a valid resume entry which should be an address in OCRAM or Cortex-M33 TCM, before the flow is executed. Similarly, Stack Pointer (SP) and Program Counter (PC) pointed by this resume entry should also be valid, either in OCRAM or Cortex-M33 TCM. Meanwhile, the SP should be in a readable and writable region and the PC should be in an executable region. This flow is executed right after a Cortex-M33 only reset.

12.4 Device Initialization

This section describes the details of the ROM and provides the initialization details. This includes details on:

- The ROM memory map
- The RAM memory map
- On-chip blocks that the ROM must use or change the POR register default values
- Clock initialization
- Exception handling and interrupt handling

12.4.1 Internal ROM/RAM Memory Map

The following figure shows the internal ROM and RAM memory map.

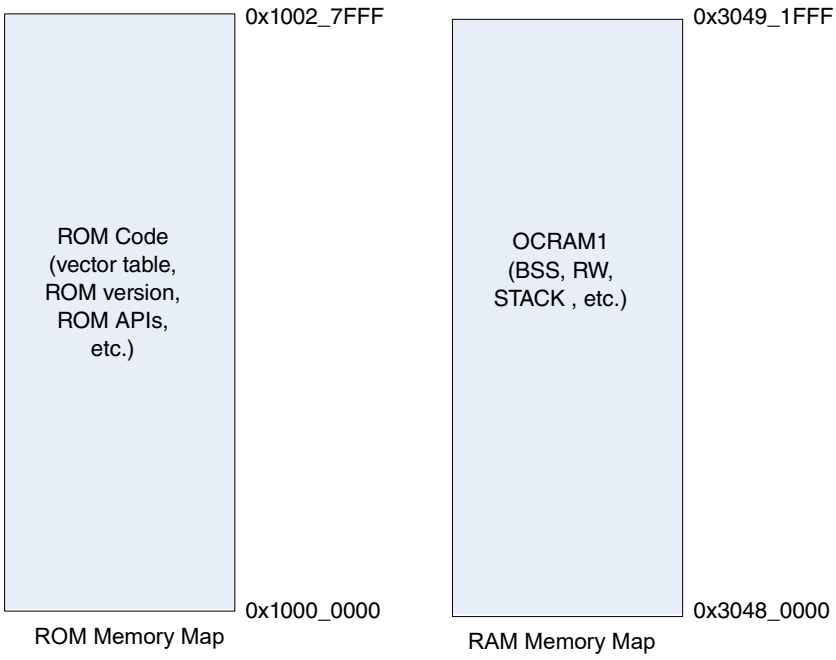


Figure 37. Internal ROM and RAM memory map

NOTE

The RAM space occupied by ROM cannot be used as part of the boot image. The entire OCRAM1 region can be used freely after the boot.

12.4.2 MISC Fuse Configuration

The following table is a list of the parameters that the ROM uses for misc configuration.

Table 54. MISC Fuse definitions

Fuse	Definitions	Settings	Shipped Value
BOOT_CFG0[27:24]	BOOT_FAIL_IND_PIN	Select the pin used as boot failure indicator pin 0x0 ~ 0xB – GPIO_AD_00 ~ GPIO_AD_12 0xC – GPIO_B1_00 0xD – GPIO_B1_01 0xE – GPIO_B2_00 0xF – GPIO_AD_33 Refer to Boot Failure Indicator for more information.	0
BOOT_CFG0[30:28]	Reserved	Must be 0	0

Table continues on the next page...

Table 54. MISC Fuse definitions (continued)

Fuse	Definitions	Settings	Shipped Value
BOOT_CFG0[31]	BOOT_FAIL_IND_EN	0 – Boot failure indicator pin is disabled 1 – Boot failure indicator pin is enabled	0
BOOT_CFG7[8]	WDOG_EN	0 – All watchdogs are disabled during boot 1 – WDOG1 is enabled during boot, the others are disabled. Refer to Boot Watchdog for more information.	0
BOOT_CFG7[11:9]	WDOG_TIMEOUT_SEL	WDOG1 timeout 0 – 128s 1 – 64s 2 – 32s 3 – 16s 4 – 8s 5 - 4s 6 - 2s 7 - 1s	0
BOOT_CFG7[15:12]	Reserved	Must be 0	0
BOOT_CFG7[16]	DIS_POR_PRELOAD_M33_TCM_ECC	0 – Cortex-M33 TCMs are preloaded with 0s by ROM 1 – Cortex-M33 TCMs are not preloaded with 0s by ROM Refer to TCM ECC Configuration for more information.	0
BOOT_CFG7[17]	POR_PRELOAD_M7_TCM_ECC_EN	0 – Cortex-M7 TCMs are not preloaded with 0s by ROM 1 – Cortex-M7 TCMs are preloaded with 0s by ROM (requires RELEASE_M7_RST_STAT to be set) Refer to TCM ECC Configuration for more information.	0
BOOT_CFG7[18]	RELEASE_M7_RST_STAT	Determine whether ROM will release Cortext-M7 Core reset and let it run. 0 – Cortex-M7 Reset is kept as hold 1 – Cortex-M7 Reset is released by ROM Refer to TCM ECC Configuration for more information.	0
BOOT_CFG7[19]	BOOT_FREQ	Select the boot core and bus frequency source. 0 – Derived from RCOSC200M 1 – Derived from PLL3	0

Table continues on the next page...

Table 54. MISC Fuse definitions (continued)

Fuse	Definitions	Settings	Shipped Value
		Refer to Clock Configuration for Boot ROM for more information.	
BOOT_CFG7[20]	XMC_CRC32_CHECK_EN	0 – XMCD CRC32 check is disabled 1 – XMCD CRC32 check is enabled Refer to External Memory Configuration Data (XMCD) for more information.	0
BOOT_CFG7[23:16]	Reserved	Must be 0	0
BOOT_CFG8[31:0]	XMC_CRC32_CHECK_SUM	XMCD CRC32 sum value Refer to External Memory Configuration Data (XMCD) for more information.	0
BOOT_CFG9[31:0]	Reserved	Must be 0	0
BOOT_CFG10[31:0]	Reserved	Must be 0	0
BOOT_CFG11[31:2]	Reserved	Must be 0	0

12.4.3 TCM ECC Configuration

The TCM ECC is enabled by default on this device. Any access to the TCM memories will cause an ECC error thrown out after a POR reset due to the TCM Memories are filled with invalid random bytes.

ROM offers a TCM preloading features to clear the TCM memories to all 0s after a POR reset. ROM will always perform a preloading for the entire Cortex-M33 Code TCM and System TCM. This operation can be canceled by setting DIS_POR_PRELOAD_M33_TCM_ECC fuse. ROM will perform a preloading for the entire Cortex-M7 ITCM and DTCM when POR_PRELOAD_M7_TCM_ECC_EN and RELEASE_M7_RST_STAT are set. Once RELEASE_M7_RST_STAT is set, ROM will release the Cortex-M7 core reset. Once the Cortex-M7 Core is released from reset, the default initial vector address (0x0) will be loaded into Cortex-M7 core, which means that the Cortex-M7 core can only start to run from address 0x0.

ROM also offers a user-configurable TCM preloading for the other types of reset. GPR9 low half-word is used for Cortex-M33 TCM preloading while high half-word is used for Cortex-M7 TCM preloading. Write 0x4646(ASICII code for “FF”, the abbreviation of “Forced Filling”) to the bitfield before issuing reset to activate the ROM execution. The writing is one-shot, ROM will clear the relative bitfield and then perform the preloading. User-configuration only takes effect when the relative ROM TCM preloading feature is enabled by setting DIS_POR_PRELOAD_M33_TCM_ECC and POR_PRELOAD_M7_TCM_ECC_EN correctly.

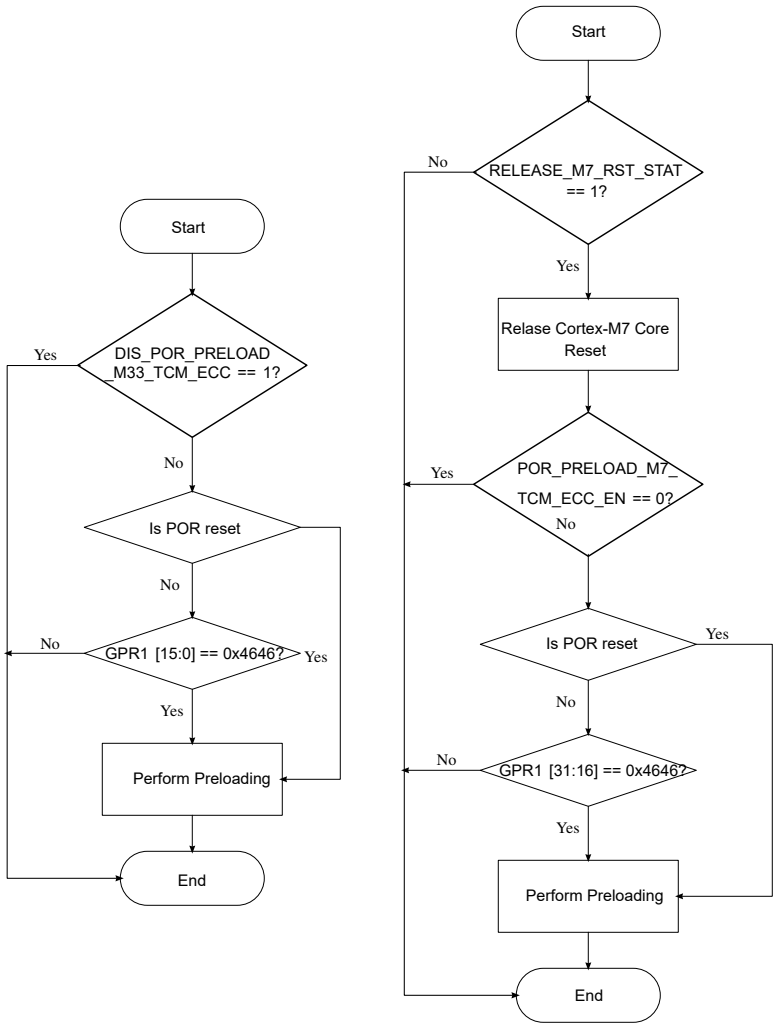


Figure 38. TCM ECC Configuration

NOTE

TCM preloading will increase the boot time significantly. If a user is concerned about the boot time, related fuses must be blown to optimize the boot time.

12.4.4 Clock Configuration for Boot ROM

The boot ROM configures the clocks of the boot core, bus, and other required controllers to a high frequency of clock roots during boot for high-performance consideration. When the user's application needs to re-configure the clock roots, PLLs or the PFDs, the SoC may crash if the clocks are not handled properly. To avoid such issues the following suggestions must be applied:

- Switch the root clock of the boot core, bus, and other affected controllers to a fixed clock root. before configuring the clocks.
- Configure the clock of each peripheral explicitly instead of relying on the default settings listed in the Reference Manual.

The following table provide the clock frequency configurations.

Table 55. Clock Configurations for Boot ROM

Clocks	BOOT_FREQ = 0		BOOT_FREQ=1	
	Clock Root	Frequency	Clock Root	Frequency
M33 Clock	RCOSC200M	200MHz	PLL3_CLK	240MHz
Edgelock Clock	RCOSC200M	133MHz	PLL2_PFD1_CLK	200MHz
Bus Aon Clock	RCOSC200M	100MHz	PLL2_CLK	132MHz
Bus Wakeup Clock	RCOSC200M	100MHz	PLL2_CLK	132MHz
Wakeup AXI Clock	RCOSC200M	133MHz	PLL3_CLK	160MHz
FlexSPI1 Clock	PLL3_PFD0_CLK	The value varies at different frequencies	The same values as for BOOT_FREQ = 0	
FlexSPI2 Clock	PLL3_PFD2_CLK	The value varies at different frequencies		
LPUART1 Clock	PLL3_DIV2_CLK	80MHz		
LPSPI1,2,3,4,5,6 Clock	PLL2_CLK	The value varies at different frequencies		
USDHC1 Clock	PLL2_PFD2_CLK	198MHz	The same values as for BOOT_FREQ = 0	
USDHC1 Clock	PLL2_PFD2_CLK	396MHz		
SEMC Clock	PLL2_PFD0_CLK	The value varies at different frequencies	The same values as for BOOT_FREQ = 0	
USB Clock	PLL3_CLK	480MHz		

12.4.5 ROM Settings and Considerations for User Applications

The registers in the modules listed below may be updated if they are in use for specific boot modes. ROM does not restore the registers in these modules after they are updated. The user application should not rely on the default values of the registers in these modules.

- FlexSPI1 - if the device boots from memory connected to FlexSPI1.
- FlexSPI2 - if the device boots from memory connected to FlexSPI2.
- uSDHC1 - if the device boots from memory connected to uSDHC1.
- uSDHC2 - if the device boots from memory connected to uSDHC2.
- SEMC - if the device boots from memory connected to SEMC.
- FlexSPI1, FlexSPI2, uSDHC1, uSDHC2, SEMC, LPSPI1, LPSPI2, LPSPI4, and LPSPI5 - if the device boots from memory connected to relative modules.
- USB1, USB-PHY1, LPUART1, and LPSPI1 - if serial downloader is used.
- ANADIG_OSC - The OSC related configurations are not restored.
- ANADIG_PMU - The ANADIG PMU related settings are not restored.

- ANADIG_PLL - The ANADIG_PLL related settings are not restored (e.g. PLL_480_CTRL, PLL_480_PFD, PLL_528_CTRL, PLL_528_PFD).
- CCM - The clock settings for the modules used during boot are not restored.
- IOMUXC - cannot fully be restored because some settings are still in use before leaving ROM.
- WDOG1-5 - disabled by ROM.
- SRC GPR - ROM occupies SRC GPR0-GPR4 and GPR9.

12.4.6 Error Handling

12.4.6.1 Exception Handling

All types of Faults cause SYSRESETREQ, including the following:

- HardFault, BusFault, MemMangeFault, UsageFault
- ROM boot flow mismatch (treated as a side-channel attack)
- Illegal ROM configuration (treated as a side-channel attack)

12.4.6.2 Boot Failure Indicator

ROM indicates the following failure(s) if the boot failure indicator feature is enabled by the BOOT_FAIL_IND_EN and BOOT_FAIL_IND_PIN fuse fields.

- The primary/recovery/manufacturing boot failed before entering serial downloader

12.4.6.3 Boot Watchdog

ROM provides the WDOG feature if the WDOG_EN fuse is blown. The WDOG_TIMEOUT_SEL fuse field provides the pre-defined timeout value.

WDOG1 is enabled and serviced by ROM during booting if the WDOG feature is enabled.

ROM will not disable the WDOG after booting, and application needs to keep servicing the WDOG or disable it.

12.4.7 Persistent Bits

Some modes of the boot ROM require the registers that keep their values after a warm reset. The SRC General-Purpose registers are used for this purpose. During ROM execution, neither registers can be changed, otherwise, the ROM behavior is unexpectable.

The following table provides the list of persistent bits and their descriptions.

Table 56. Persistent bits

Bit location	Bit name	Description
GPR0[31:0]	SRSR_BACKUP	32-bit value of SRC[SRSR] backed up at the beginning of error recovery fast boot flow for user application. See Error Recovery Fast Boot for more information.
GPR1[15:0]	FORCE_M33_TCM_LOADING	Force to preload zeros to M33 TCMs for user application, no matter it is a POR reset or a warm reset. See TCM ECC Configuration for more information.

Table continues on the next page...

Table 56. Persistent bits (continued)

Bit location	Bit name	Description
GPR1[31:16]	FORCE_M7_TCM_LOADING	Force to preload zeros to M7 TCMs for user application, no matter it is a POR reset or a warm reset. Refer to TCM ECC Configuration for more information.
GPR2[31:0]	FLASH_STATUS_CTX	Flash status context. This field logs the flash state during boot, so that the flash driver can retrieve the state of the flash and operate the flash devices properly.
GPR3[31:0]	ROM_API_CTX	ROM API context. See Enter Bootloader API for more information.
GPR9[15:0]	BOOT_STAGE	ROM current boot stage.
GPR9[26]	SECONDARY_BOOT	This bit identifies which image must be used — primary and secondary. Used only for eMMC/SD/FlexSPI NOR boot.
GPR9[27:26]	REDUNDANT_BOOT	This field identifies which image must be used - 0/1/2/3. Used for both SPI NAND and SLC raw NAND devices.

12.5 Boot Devices

The chip supports the following boot flash devices:

- Serial NOR Flash via FlexSPI Interface
- Serial NAND Flash via FlexSPI Interface
- NAND Flash with SEMC interface, located on CS0, 8-bit/16-bit bus width.
- SD/eMMC via uSDHC interface, supporting high capacity cards.
- Serial NOR/EEPROM via LPSPI

The selection of the external boot device type is controlled by the BOOT MODE. Refer to Boot Modes for more information.

12.5.1 Serial NOR Flash Boot via FlexSPI

12.5.1.1 Serial NOR FUSE Configuration

Table 57. Fuse definition for Serial NOR over FlexSPI

Fuse	Definition	Settings	Shipped Value
BOOT_CFG1[1:0]	XSPI_NOR_PROBE_TYPE	0 - JESD216 1 - MXIC Octal 2 - Micron Octal 3 - Adesto Octal	0
BOOT_CFG1[3:2]	XSPI_NOR_HOLD_TIME	0 - 500us 1 - 1ms 2 - 3ms	0

Table continues on the next page...

Table 57. Fuse definition for Serial NOR over FlexSPI (continued)

Fuse	Definition	Settings	Shipped Value
		3 - 10ms	
BOOT_CFG1[6:4]	XSPI_NOR_FREQ	0 - 100MHz 1 - 120MHz 2 - 133MHz 3 - 166MHz 4 - 200MHz 5 - 80MHz 6 - 60MHz 7 - 50MHz	0
BOOT_CFG1[7]	XSPI_NOR_CONNECTION_SEL	0 - PORTA CS0 1 - PORTB CS0	0
BOOT_CFG1[10:8]	XSPI_NOR_TYPE	0 - Boot with default 0x03 Read 1 - Reserved 2 - HyperFLASH 1V8 3 - HyperFLASH 3V0 4 - MXIC Octal Read 5 - Micron Octal DDR Read Others - Reserved	0
BOOT_CFG1[11]	RST_REQUIRED	0 - No system reset after NOR boot failure 1 - System reset after NOR boot failure	0
BOOT_CFG1[12]	XSPI_NOR_RESET_TYPE	0 - Reset sequence is specified in FLASH_STATE_CTX Register 1 - Reset FLASH by Reset Pin before access 2 - Reset FLASH by JEDEC Hardware Reset flow 3 - Typical Reset Sequence (0x66, 0x99)	0
BOOT_CFG1[16]	XSPI_NOR_ENCRYPT_XIP_EN	0 - Encrypted XIP is disabled 1 - Encrypted XIP is enabled	0
BOOT_CFG1[17]	XSPI_NOR_ENCRYPT_ENG	0 - OTFAD 1 - IEE	0
BOOT_CFG1[19:18]	XSPI_NOR_FCB_OFFSET	0 - 0x400 1 - 0x600 2 - 0xA00	0

Table continues on the next page...

Table 57. Fuse definition for Serial NOR over FlexSPI (continued)

Fuse	Definition	Settings	Shipped Value
		3 - 0xC00	
BOOT_CFG1[20]	DIS_XSPI_NOR_FCB	0 - Enable FLASH CFG Block 1 - Disable FLASH CFG Block	0
BOOT_CFG2[0]	XSPI_PAD_SLEW_OVERRIDE	0 - Fast 1 - Slow	0
BOOT_CFG2[1]	XSPI_PAD_DRIVE_OVERRIDE	0 - Normal Driver 1 - High Driver	0
BOOT_CFG2[2]	XSPI_PAD_PULL_OVERRIDE	0 - Float 1 - Internal Pulled	0
BOOT_CFG2[3]	XSPI_PAD_OVERRIDE_EN	0 - Use Default Pad Settings 1 - Use Pad Settings from BOOT_CFG2[2:0]	0
BOOT_CFG2[4]	XSPI_RESET_PIN_SEL	0 - Reset Pin Option 0 (GPIO_SD_B1_00) 1 - Reset Pin Option 1 (GPIO_EMC_B1_40)	0
BOOT_CFG2[5]	XSPI_LPB_DQS_PIN_SEL	0 - Dummy DQS Pin Used for Loopbacked DQS 1 - Standard DQS Pin Used for Loopbacked DQS	0
BOOT_CFG2[6]	XSPI_PIN_GROUP_SEL	0 - Primary Pin Group Selected 1 - Secondary Pin Group Selected	0
BOOT_CFG2[7]	XSPI_INSTANCE	0 - FlexSPI1 1 - FlexSPI2	0
BOOT_CFG2[12:8]	XSPI_DELAY_CELL_NUM	0 - Use the Delay cell value calculated by ROM Others - Use the Delay cell value in terms of 100ps	0
BOOT_CFG7[5:0]	XSPI_2ND_IMG_OFFSET	0 - No 2nd physical image presend n - 2nd physical image located at XSPI_2ND_IMG_SIZE_UNIT * n	0
BOOT_CFG7[21]	XMC_DISABLE	0 - Enabled 1 - Disabled	0
BOOT_CFG7[7:6]	XSPI_2ND_IMG_SIZE_UNIT	0 - 256KB, 1 - 1MB 2 - 4MB 3 - 16MB	0

NOTE

If the xSPI FLASH Auto Probe feature is enabled, the following is the logic of how this feature works with other fuse combinations:

- Flash Type - If Flash type is 0, the "xSPI FLASH Auto Probe Type" takes effect for the Flash type selection. If Flash Type is greater than 1, the "Flash Type" fuse is used for Flash type selection, ROM will issue specific command to probe the presence of Serial NOR FLASH.
- xSPI FLASH Frequency - This field is used for specifying the Flash working frequency.

12.5.1.2 FlexSPI Serial NOR Flash Boot Operation

The Boot ROM attempts to boot from Serial NOR flash if boot pins/BOOT_CFG0 fuse bits are configured correspondingly, then the ROM will initialize the FlexSPI interface. FlexSPI interface initialization is a two-step process.

1. The ROM expects the 512-byte FlexSPI NOR configuration parameters (refer to Serial NOR Flash Configuration Block (512 bytes)) to be present at offset 0x400 (ROM supports four FCBs, user can decide which one to use via Fuse XSPI_NOR_FCB_OFFSET if DIS_XSPI_NOR_FCB is not set) in Serial NOR flash attached to FLEXSPI_A_SS0_B. The ROM can probe the presence of Serial NOR Flash and generate these configuration parameters accordingly via the combination of Fuses in Table 57, or reads these configuration parameters using the read command specified with XSPI_NOR_TYPE with serial clock operating at 30 MHz.
2. ROM configures FlexSPI interface with the parameters provided in configuration block read from Serial NOR flash and starts the boot procedure. Refer to Table 65 for details regarding FlexSPI configuration parameters and to the FlexSPI NOR boot flow chart for the detailed boot flow of FlexSPI NOR.

Booting both XIP and non-XIP images is supported from the Serial NOR Flash. For XIP boot, the image has to be built for the FlexSPI address space. For non-XIP boot, the image can be built to execute from internal RAM.

NOTE

A non-XIP image's execute address space should NOT overlap with the reserved OCRAM region described in the section "Internal ROM/RAM memory map". It is also highly recommended to avoid using the address space near 0x0000_0000, typically up to 0x0000_0003.

12.5.1.3 FlexSPI NOR Boot Flow Chart

The FlexSPI NOR boot flow is provided below.

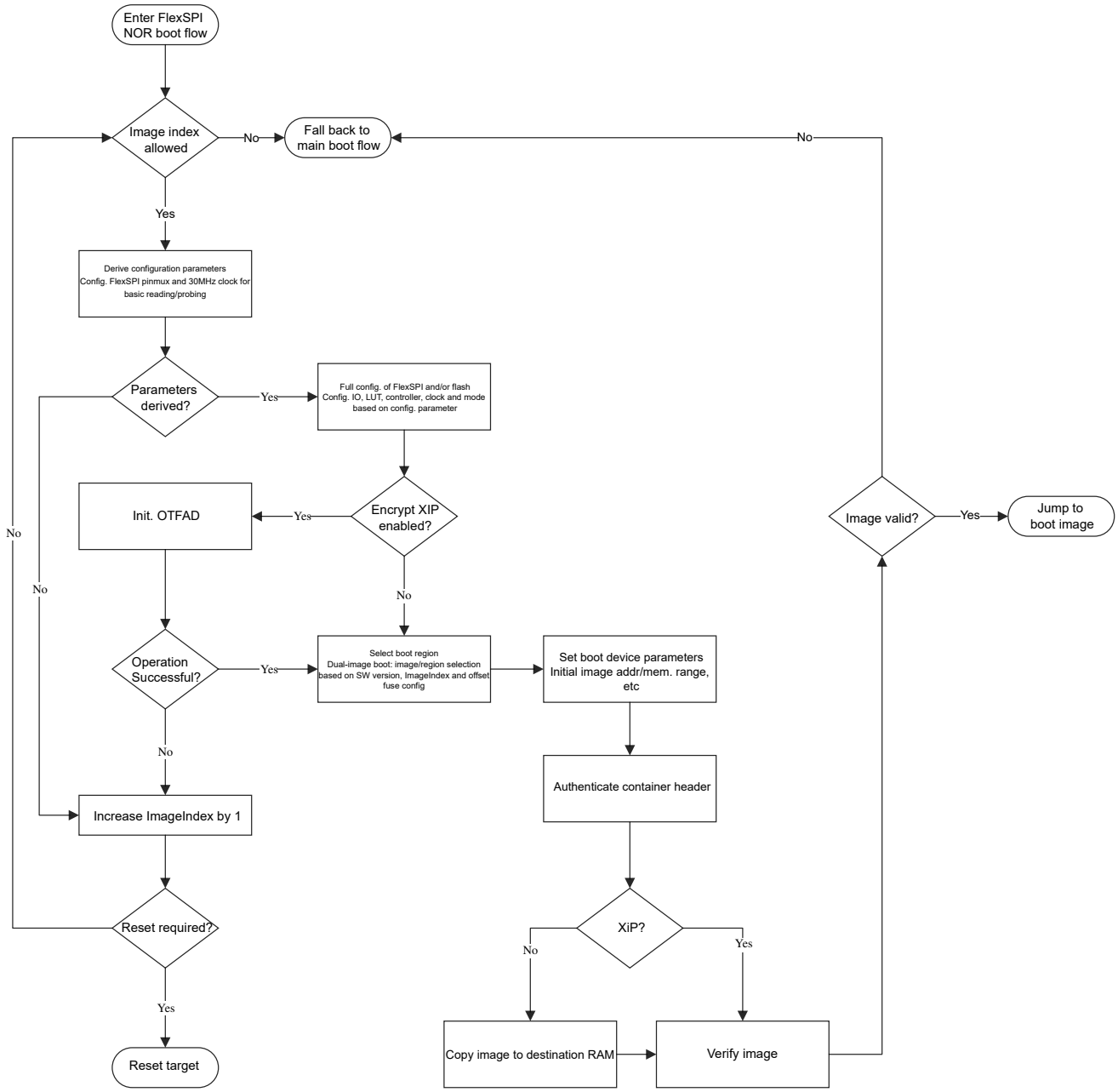


Figure 39. FlexSPI NOR boot flow

12.5.2 Serial NAND Flash Boot via FlexSPI

The boot ROM supports a number of Serial NAND Flash devices from different vendors. NAND devices' built-in ECC function is required.

NOTE

OEM image container can not cross NAND page.

12.5.2.1 Serial NAND FUSE Configuration

The boot ROM determines the configuration of external Serial NAND flash by parameters, either provided by fuses, or sampled on GPIO pins during boot. See the following table for parameters details.

Table 58. Fuse definition for Serial NAND over FlexSPI

Fuse	Definition	Settings	Shipped Value
BOOT_CFG2[4]	XSPI_RESET_PIN_SEL	0 - Reset Pin Option 0 (GPIO_SD_B1_00) 1 - Reset Pin Option 1 (GPIO_EMC_B1_40)	0
BOOT_CFG2[6]	XSPI_PIN_GROUP_SEL	0 - Primary Pin Group Selected 1 - Secondary Pin Group Selected	0
BOOT_CFG2[7]	XSPI_INSTANCE	0 - FlexSPI1 1 - FlexSPI2	0
BOOT_CFG3[0]	XSPI_NAND_SEARCH_CNT	0 - Search at most 1 block (for BCB) 1 - Search at most 2 blocks	0
BOOT_CFG3[1]	XSPI_NAND_COL_ADDR_WIDTH	0 - 12 bits (for 2KB-page devices) 1 - 13 bits (for 4KB-page devices)	0
BOOT_CFG3[3:2]	XSPI_NAND_CS_INTERVAL	CS de-asserted interval between two commands 0 - 100ns 1 - 200ns 2 - 400ns 3 - 50ns	0
BOOT_CFG3[5:4]	XSPI_NAND_SEARCH_STRIDE	BCB search step/stride, aligned with block size 0 - 64 pages 1 - 128 pages 2 - 256 pages 3 - 32 pages	0
BOOT_CFG3[7:6]	XSPI_NAND_HOLD_TIME	Hold time before interaction with NAND 0 - Wait until NAND is idle (by polling NAND's status register) 1 - 500us 2 - 1ms 3 - 3ms	0
BOOT_CFG3[11:10]	XSPI_NAND_RST_TYPE	0 - No reset before interaction with NAND 1 - Reset the NAND with 0xFF 2 - Reset the NAND with 0x66, 0x99 3 - Reset the NAND with dedicated Reset Pin	0

Table continues on the next page...

Table 58. Fuse definition for Serial NAND over FlexSPI (continued)

Fuse	Definition	Settings	Shipped Value
BOOT_CFG3[12]	XSPI_NAND_CONN_SEL	0 - Connected via PORTA CS0 1 - Connected via PORB CS0	0
BOOT_CFG3[18:16]	XSPI_NAND_FREQ	0 - 80MHz 6 - 60MHz 7 - 30MHz Otherwise - Reserved	

NOTE

All the fuse fields in BOOT_CFG2 are there for FCB derivation stage. After successful FCB derivation, FlexSPI is re-configured as per FCB.

12.5.2.2 FlexSPI NAND Flash Boot Flow and Boot Control Blocks (BCB)

There are two BCB data structures:

- FCB
- DBBT

As part of the NAND media initialization, the ROM driver uses safe NAND timings to search for a Firmware Configuration Block (FCB) that contains the optimum NAND timings, page address of Discovered Bad Block Table (DBBT) Search Area and start page address of primary and secondary firmware.

The built-in HW ECC in Serial NAND device is used during data read, the FCB data structure is protected using CRC checksum. Driver reads raw 2048 bytes of first sector and runs through CRC check that determines whether FCB data is valid or not.

If the FCB is found, the optimum NAND timings are loaded for further reads. If the ECC fails, or the fingerprints do not match, the Block Search state machine increments page number to Search Stride number of pages to read for the next BCB until SearchCount pages have been read.

If search fails to find a valid FCB, the NAND driver responds with an error and the boot ROM falls back to the main boot flow (e.g., turn to recovery boot).

The FCB contains the page address of DBBT Search Area, and the page address for primary and secondary boot images. ROM searches DBBT Search Area for DBBT, resembling the search for FCB. After the FCB is derived, the DBBT is loaded, and the primary or secondary boot image is loaded using starting page address from FCB.

Figure 40 shows the state diagram of FCB search.

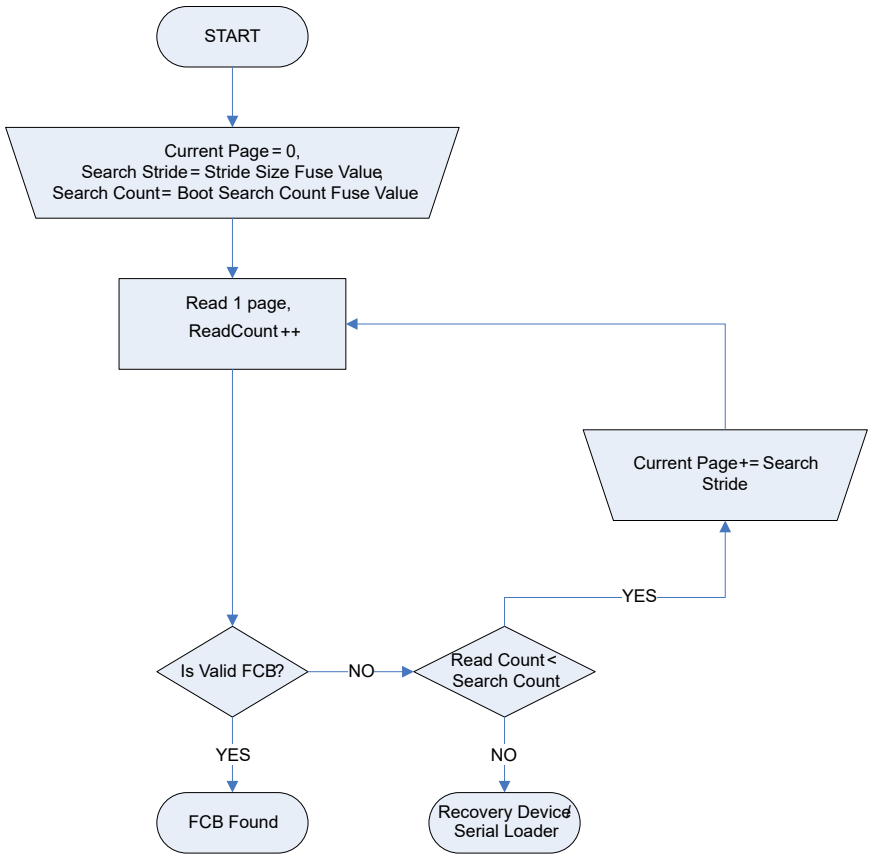


Figure 40. FCB Search Flow

After FCB is found, the boot ROM searches for the Discovered Bad Blocks Table (DBBT). If DBBTSearchStartPage is 0 in the FCB, then ROM assumes that there are no bad blocks in the NAND device boot area. See Figure 41 for the DBBT search flow.

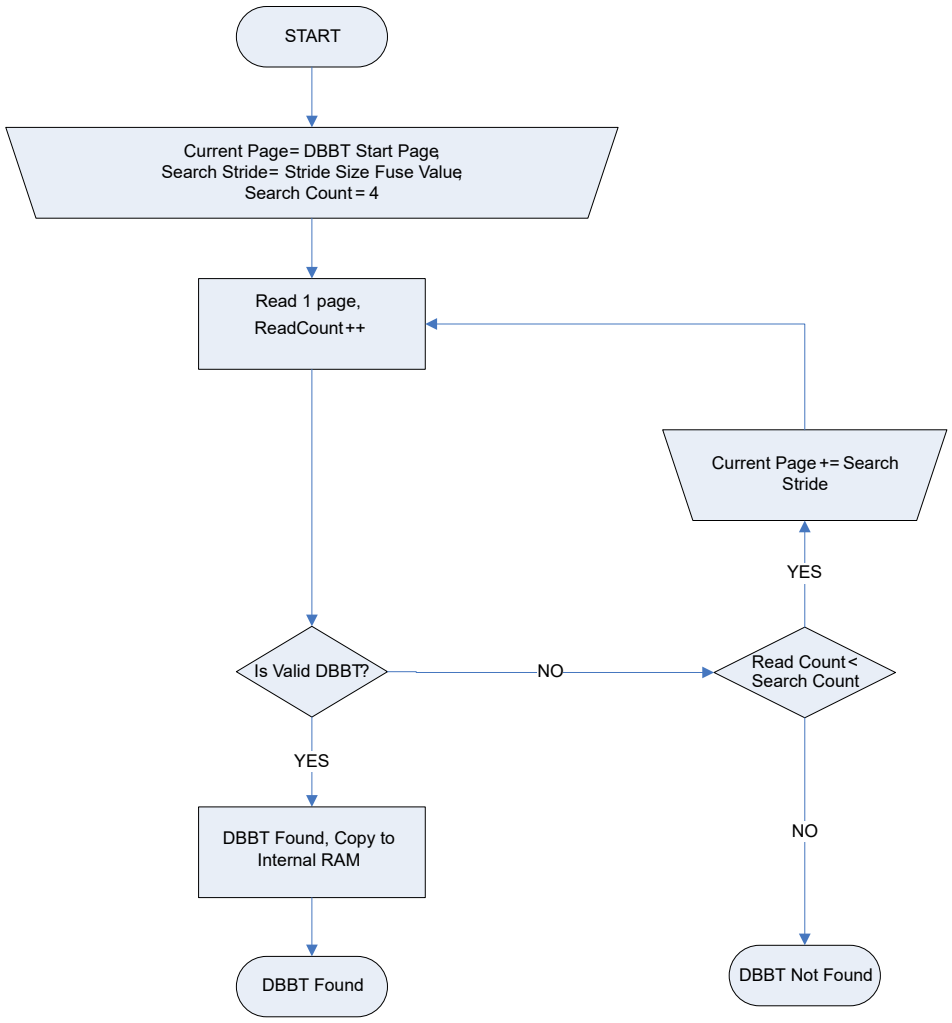


Figure 41. DBBT Search Flow

If during primary image read there is a page with a number of errors higher than ECC can correct, the boot ROM will turn on PERSIST_REDUNDANT_BOOT bit and try a secondary image.

If during secondary image read there is a page with number of errors higher than ECC can correct, the boot ROM goes back to main boot flow.

12.5.2.3 Firmware Configuration Block (FCB)

The Firmware Configuration Block (FCB) is at the first page in the first good block. The FCB should be present at each search stride of the search area.

The search area contains copies of the FCB at each stride distance, in case the first Serial NAND block becomes corrupted, the ROM will find its copy in the next Serial NAND block. The search area should span over at least two Serial NAND blocks. The location information for DBBT search area and images are all specified in the FCB. The following table shows the Flash Control Block Structure.

Table 59. Flash Control Block Structure

Name	Offset	Size Bytes	Description
crcChecksum	0x000	4	Checksum
fingerprint	0x004	4	0x4E46_4342 ASCII: “NFCB”
version	0x008	4	0x0000_0001
DBBTSearchStartPage	0x00C	4	Start Page address for bad block table search area
searchStride	0x010	2	Search stride for DBBT and FCB search. Not used by ROM Max value is 8.
searchCount	0x012	2	Copies of DBBT and FCB. Not used by ROM, max value is 8.
firmwareCopies	0x014	4	Firmware copies Valid range 1-4.
Reserved	0x018	40	Reserved for future use Must be set to 0.
firmwareInfoTable	0x40	64	This table consists of (up to 8 entries): **Field** / **Size** / **Description** StartPage / 4 / Start page of this firmware page Count / 4 / Pages in this firmware **NOTE** The StartPage must be the first page of a NAND block.
Reserved	0x080	128	Reserved Must be set to 0
spiNandConfigBlock	0x100	512	Serial NAND Flash configuration block over FlexSPI. Refer to Serial NAND Flash Configuration Block (512 bytes) for details.
Reserved	0x300	256	Must be set to 0

NOTE

1. The “crcChecksum” is calculated with an MPEG2 variant of CRC-32. See Table 60 for more details.
2. The “crcChecksum” calculation starts from fingerprint to the end of FCB, 1020 bytes in total.
3. The “spiNandConfigBlock” is FlexSPI NAND configuration block which consists of common FlexSPI memory configuration block and Serial NAND specified configuration parameters.

Table 60. CRC-32 variant algorithm

Property	Description
Width	32 bits
Polynomial	0x04C11DB7
Init Value	0xFFFFFFFF
Reflect in	False
Reflect Out	False
XOR Out	0x00000000

12.5.2.4 Discovered Bad Blocks Table (DBBT)

Table 61. DBBT Structure

Name	Offset	Size in Bytes	Description
crcChecksum	0x000	4	Checksum
Fingerprint	0x004	4	32-bit word with a value of 0x4442_4254, in ASCII “DBBT”
Version	0x008	4	32-bit version number, this version of DBBT is 0x00000001
-	0x00C	4	Reserved
badBlockNumber	0x010	4	Number of bad blocks
Reserved	0x014	12	Must be filled with 0x00s
Bad Block entries	0x020	1024 (256*4)	Each bad block entry is a 32- bit value specifying the number of a found bad block. The number of valid bad block entries is specified by the badBlockNumber field, where valid bad block entries are stored sequentially starting at the beginning of the bad block entries segment. Unused bad block entries (those beyond the badBlockNumber) should be filled with 0xFs.

NOTE

1. Maximum badBlockNumber is 256.
2. The "crcChecksum" is calculated with the same algorithm as the one in FCB, from Fingerprint to the end of DBBT, 1052 bytes in total.

12.5.2.5 Bad Block Handling in ROM

During the firmware boot, at the block boundary, the Bad Block table is searched for a match to the next block.

If no match is found, the next block can be loaded. If a match is found, the block must be skipped and the next block checked.

12.5.3 Serial NOR and NAND Flash Configuration based on FlexSPI Interface

The ROM SW supports Serial NOR Flash and Serial NAND Flash based on FlexSPI module, using a 448-bytes common FlexSPI configuration block and several specified parameters for Serial NOR Flash and Serial NAND Flash respectively. See the following sections for more details.

12.5.3.1 FlexSPI Configuration Block

FlexSPI Configuration block consists of parameters regarding specific Flash devices including read command sequence, quad mode enablement sequence (optional), etc.

Table 62. FlexSPI Configuration block

Name	Offset	Size(bytes)	Description
Tag	0x000	4	0x42464346, ascii: 'FCFB'
Version	0x004	4	[07:00] bugfix = 0 [15:08] minor [23:16] major = 1 [31:24] ascii ‘V’
-	0x008	4	Reserved
readSampleClkSrc	0x00C	1	0 – internal loopback 1 – loopback from DQS pad 3 – Flash provided DQS
csHoldTime	0x00D	1	Serial Flash CS Hold Time Recommend default value is 0x03
csSetupTime	0x00E	1	Serial Flash CS setup time Recommended default value is 0x03
columnAdressWidth	0x00F	1	3 – For HyperFlash 12/13 – For Serial NAND, see the NAND FLASH datasheet to find the correct value 0 – Other devices
deviceModeCfgEnable	0x010	1	Device Mode Configuration Enable feature 0 – Disabled 1 – Enabled
deviceModeType	0x011	1	0 - Generic 1 - Quad Enable

Table continues on the next page...

Table 62. FlexSPI Configuration block (continued)

Name	Offset	Size(bytes)	Description
			2 - SPI-to-xSPI mode 3 - xSPI-to-SPI mode 4 - SPI-to-NoCmd mode
waitTimeCfgCommands	0x012	2	Wait time for all configuration commands, unit 100us. Available for device that support v1.1.0 FlexSPI configuration block. If it is greater than 0, ROM will wait waitTimeCfgCommands * 100us for all device memory configuration commands instead of using read status to wait until these commands complete.
deviceModeSeq	0x014	4	Sequence parameter for device mode configuration Bit[7:0] - number of LUT sequences for Device mode configuration command Bit[15:8] - starting LUT index of Device mode configuration command Bit[31:16] - must be 0
deviceModeArg	0x018	4	Device Mode argument, effective only when deviceModeCfgEnable = 1
configCmdEnable	0x01C	1	Config Command Enable feature 0 – Disabled 1 – Enabled
-	0x01D	3	Reserved
configCmdSeqs	0x020	12	Sequences for Config Command, allow 3 separate configuration command sequences.
-	0x02C	4	Reserved
cfgCmdArgs	0x030	12	Arguments for each separate configuration command sequence.
-	0x03C	4	Reserved
controllerMiscOption	0x040	4	Bit0 – differential clock enable Bit1 – CK2 enable, must set to 0 in this silicon Bit2 – ParallelModeEnable Bit3 – wordAddressableEnable Bit4 – Safe Configuration Frequency enable set to 1 for the devices that support DDR Read instructions Bit5 – Pad Setting Override Enable

Table continues on the next page...

Table 62. FlexSPI Configuration block (continued)

Name	Offset	Size(bytes)	Description
			Bit6 – DDR Mode Enable, set to 1 for device supports DDR read command Bit 7 - Pad Setting Override Enable Bit 8 - Second Pin group Bit 9 - Second DQS pin group Bit 10 - Write Mask Enable Bit 11 - Write Opt1 Clear
deviceType	0x044	1	1 – Serial NOR 2 – Serial NAND 3 - Serial RAM (HyperRAM/APMemory)
sflashPadType	0x045	1	1 – Single pad 2 – Dual pads 4 – Quad pads 8 – Octal pads
serialClkFreq	0x046	1	Chip specific value, for this silicon 1 – 30 MHz 2 – 50 MHz 3 – 60 MHz 4 – 80 MHz 5 – 100 MHz 6 – 120 MHz 7 – 133 MHz 8 – 166 MHz Other value: 30 MHz
lutCustomSeqEnable	0x047	1	0 – Use pre-defined LUT sequence index and number 1 - Use LUT sequence parameters provided in this block
-	0x048	8	Reserved
sflashA1Size	0x050	4	For SPI NOR, need to fill with actual size For SPI NAND, need to fill with actual size * 2
sflashA2Size	0x054	4	The same as above
sflashB1Size	0x058	4	The same as above
sflashB2Size	0x05C	4	The same as above

Table continues on the next page...

Table 62. FlexSPI Configuration block (continued)

Name	Offset	Size(bytes)	Description
csPadSettingOverrideEn	0x060	1	Set to 0 if it is not supported
csPadSettingOverride	0x061	1	Overriding pad setting of CS
sclkPadSettingOverrideEn	0x064	1	Set to 0 if it is not supported
sclkPadSettingOverride	0x065	1	Overriding pad setting of SCLK
dataPadSettingOverrideEn	0x068	1	Set to 0 if it is not supported
dataPadSettingOverride	0x069	1	Overriding pad setting of data signals
dqsPadSettingOverrideEn	0x06C	1	Set to 0 if it is not supported
dqsPadSettingOverride	0x06D	1	Overriding pad setting of DQS
timeoutInMs	0x070	4	Maximum wait time during read busy status 0 – Disabled timeout checking feature Other value – Timeout if the wait time exceeds this value
commandInterval	0x074	4	Unit: ns Currently, it is used for SPI NAND only at high frequency
dataValidTime	0x078	4	Time from clock edge to data valid edge, unit ns. This field is used when the FlexSPI Root clock is less than 100 MHz and the read sample clock source is device provided DQS signal without CK2 support. [31:16] data valid time for DLLB in terms of 0.1 ns [15:0] data valid time for DLLA in terms of 0.1 ns
busyOffset	0x07C	2	busy bit offset, valid range :0-31
busyBitPolarity	0x07E	2	1. 0 – busy bit is 1 if device is busy 2. 1 – busy bit is 0 if device is busy
lookupTable	0x080	256	Lookup table
lutCustomSeq	0x180	48	Customized LUT sequence, see below table for details.
	0x1B0	16	Reserved for future use

Note:

1. To customize the LUT sequence for some specific device, users need to enable “lutCustomSeqEnable” and fill in corresponding “lutCustomSeq” field specified by command index below.
2. For Serial (SPI) NOR, the pre-defined LUT index is as follows:

Table 63. LUT sequence definition for Serial NOR

Command Index	Name	Index in lookup table	Description
0	Read	0	Read command Sequence
1	ReadStatus	1	Read Status command
2	WriteEnable	3	Write Enable command sequence
3	EraseSector	5	Erase Sector Command
4	PageProgram	9	Page Program Command
5	ChipErase	11	Full Chip Erase
6	Dummy	15	Dummy Command as needed
7-12	Reserved	2,4,6,7,8,10,12,13,14	All reserved indexes can be freely used for other purpose
13	NOR_CMD_LUT_SEQ_IDX_READ_SFDP	13	Read SFDP sequence in lookupTable id stored in config block
14	NOR_CMD_LUT_SEQ_IDX_RESTORE_NOCMD	14	Restore 0-4-4/0-8-8 mode sequence id in lookupTable stored in config block

3. For Serial (SPI) NAND, the pre-defined LUT index is as follows:

Table 64. LUT sequence definition for Serial NAND

Command Index	Name	Index in lookup table	Description
0	ReadFromCache	0	Read from cache
1	ReadStatus	1	Read Status
2	WriteEnable	3	Write Enable
3	BlockErase	5	Erase block
4	ProgramLoad	9	Program Load
5	ReadPage	11	Read page to cache
6	ReadEccStatus	13	Read ECC Status
7	ProgramExecute	14	Program Execute
8	ReadFromCacheOdd	4	Read from Cache while page in odd plane
9	ProgramLoadOdd	10	Program Load for pages within odd blocks
	Reserved	2,6,7,8,12,15	All reserved indexes can be freely used for other purposes

NOTE

1. All the pre-defined LUT indexes are only applicable to boot stage. User application can use the whole 16 LUT entries freely based on their requirement.
2. The FlexSPI NOR ROM APIs occupy LUT index 0-5. User application should **NOT** use these LUT indexes if the ROM API is called in their application frequently.

12.5.3.2 Serial NOR Flash Configuration Block (512 bytes)

Table 65. Serial NOR Flash configuration block

Name	Offset	Size (Bytes)	Description
memCfg	0	448	The common memory configuration block, see FlexSPI configuration block for more details
pageSize	0x1C0	4	Page size in terms of bytes, not used by ROM
sectorSize	0x1C4	4	Sector size in terms of bytes, not used by ROM
ipCmdSerialClkFreq	0x1C8	1	Chip specific value, not used by ROM 0 – No change, keep current serial clock unchanged 1 – 30 MHz 2 – 50 MHz 3 – 60 MHz 4 – 80 MHz 5 – 100 MHz 6 – 120 MHz – 133 MHz
isUniformBlockSize	0x1C9	1	Device has uniform block size 0 - No, block size != sector size 1 - Yes, block size = sector size
isDataOrderSwapped	0x1CA	1	The data order is swapped in OPI DDR mode 0 - Not swapped 1 - Swapped (this is applicable to Macronix MX25UM512/256/128 series only)

Table continues on the next page...

Table 65. Serial NOR Flash configuration block (continued)

Name	Offset	Size (Bytes)	Description
Reserved	0x1CB	5	Reserved
blockSize	0x1D0	4	blockSize
flashStateCtx	0x1D4	4	Flash State Context after being configured [7:0] Flash POR Mode: 0x00 - Extended SPI mode 0x42 - QPI DDR mode 0x82 - OPI DDR mode -- [15:8] Flash Current Mode: 0x00 - Extended SPI mode 0x42 - QPI DDR mode 0x82 - OPI DDR mode -- [23:16] Reserved -- [31:24] Flash Restoring Sequence: 0x6 - Send 0x66_0x99 (Micron Octal Flash) 0x7 - Send 0x6699_0x9966 (Macronix Octal Flash) 0x8 - Send 0x06 0xFF (Adesto Octal Flash)
Reserved	0x1D8	24	Reserved

12.5.3.3 Serial NAND Flash Configuration Block (512 bytes)

Table 66. Serial NAND Flash configuration block

Name	Offset	Size (Bytes)	Description
memCfg	0	448	The common memory configuration block, see FlexSPI configuration block for more details
pageDataSize	0x1C0	4	Page size in terms of bytes, usually, it is 2048 or 4096
pageTotalSize	0x1C4	4	It equals to 2 ^ width of column adddress

Table continues on the next page...

Table 66. Serial NAND Flash configuration block (continued)

Name	Offset	Size (Bytes)	Description
pagesPerBlock	0x1C8	4	Pages in one block
bypassReadStatus	0x1CC	1	0 – Read Status Register 1 – Bypass Read status register
bypassEccRead	0x1CD	1	0 – Perform ECC read 1 – Bypass ECC read
hasMultiPlanes	0x1CE	1	0 – Only 1 plane 1 – Has two planes
skippOddBlocks	0x1CF	1	0 – Read Odd blocks 1 – Skip Odd blocks
eccCheckCustomEnable	0x1D0	1	0 – Use the common ECC check command and ECC related masks 1 - Use ECC check related masks provided in this configuration block
ipCmdSerialClkFreq	0x1D1	1	Chip specific value, not used by ROM 0 – No change, keep current serial clock unchanged 1 – 30 MHz 2 – 50 MHz 3 – 60 MHz 4 – 75 MHz 5 – 80 MHz 6 – 100 MHz 7 – 133 MHz 8 – 166 MHz
readPageTimeUs	0x1D2	2	Wait time during page read, this field will take effect on if the bypassReadStatus is set to 1. **NOTE** Only applicable to ROM.

Table continues on the next page...

Table 66. Serial NAND Flash configuration block (continued)

Name	Offset	Size (Bytes)	Description
eccStatusMask	0x1D4	4	ECC Status Mask
eccFailureMask	0x1D8	4	ECC Check Failure mask
blocksPerDevice	0x1DC	4	Blocks in a Serial NAND
Reserved	0x1ED	32	Reserved for future use

Below is an example of Serial NAND configuration block for Winbond W25N01GVZEIG:

```
const flexspi_nand_config_t kSerialNandCfgBlk =
{
    .memConfig =
    {
        .tag = FLEXSPI_CFG_BLK_TAG,
        .version = FLEXSPI_CFG_BLK_VERSION,
        .readSampleClkSrc = kFlexSPIReadSampleClk_LoopbackInternally,
        .dataHoldTime = 3,
        .dataSetupTime = 3,
        .columnAddressWidth = 12,
        .deviceModeCfgEnable = 1,
        .deviceModeSeq = { 1, 2 },
        .deviceType = kFlexSpiDeviceType_SerialNAND,
        .sflashPadType = kSerialFlash_4Pads,
        .serialClkFreq = kFlexSpiSerialClk_50MHz,
        .lutCustomSeqEnable = 0,
        .sflashA1Size = 128 * 1024 * 1024U * 2, // Flash size = 2 * actual data size (exclude spare
space)
        .lookupTable =
        {
            // Read cache 4 I/0
            [4 * NOR_CMD_LUT_SEQ_IDX_READ] = FLEXSPI_LUT_SEQ(CMD_SDR, FLEXSPI_1PAD, 0xEB, CADDR_SDR,
FLEXSPI_4PAD, 0x10),
            [4 * NOR_CMD_LUT_SEQ_IDX_READ + 1] = FLEXSPI_LUT_SEQ(DUMMY_SDR, FLEXSPI_4PAD, 0x04,
READ_SDR, FLEXSPI_4PAD, 0x80),
            // Clear Status1 flag
            [4 * 2] = FLEXSPI_LUT_SEQ(CMD_SDR, FLEXSPI_1PAD, 0x1F, CMD_SDR, FLEXSPI_1PAD, 0xA0),
            [4 * 2 + 1] = FLEXSPI_LUT_SEQ(CMD_SDR, FLEXSPI_1PAD, 0x00, STOP, FLEXSPI_1PAD, 0x00),
            // Read Page
            [4 * NAND_CMD_LUT_SEQ_IDX_READPAGE] = FLEXSPI_LUT_SEQ(CMD_SDR, FLEXSPI_1PAD, 0x13,
RADDR_SDR, FLEXSPI_1PAD, 0x18),
            // Read Status
            [4 * NAND_CMD_LUT_SEQ_IDX_READSTATUS] = FLEXSPI_LUT_SEQ(CMD_SDR, FLEXSPI_1PAD, 0x0F,
CMD_SDR, FLEXSPI_1PAD, 0xC0),
            [4 * NAND_CMD_LUT_SEQ_IDX_READSTATUS + 1] = FLEXSPI_LUT_SEQ(READ_SDR, FLEXSPI_1PAD, 0x01,
STOP, FLEXSPI_1PAD, 0),

            // Write Enable
            [4 * NAND_CMD_LUT_SEQ_IDX_WRITEENABLE] = FLEXSPI_LUT_SEQ(CMD_SDR, FLEXSPI_1PAD, 0x06,
STOP, FLEXSPI_1PAD, 0),

            // Page Program Load 4x
            [4 * NAND_CMD_LUT_SEQ_IDX_PROGRAMLOAD] = FLEXSPI_LUT_SEQ(CMD_SDR, FLEXSPI_1PAD, 0x32,
CADDR_SDR, FLEXSPI_1PAD, 0x10),
            [4 * NAND_CMD_LUT_SEQ_IDX_PROGRAMLOAD + 1] = FLEXSPI_LUT_SEQ(WRITE_SDR, FLEXSPI_4PAD,
0x40, STOP, FLEXSPI_1PAD, 0),
            // Page Program Execute
```

```
            [4 * NAND_CMD_LUT_SEQ_IDX_PROGRAMEXECUTE] = FLEXSPI_LUT_SEQ(CMD_SDR, FLEXSPI_1PAD, 0x10,
RADDR_SDR, FLEXSPI_1PAD, 0x18),
            // Erase Sector
            [4 * NAND_CMD_LUT_SEQ_IDX_ERASEBLOCK] = FLEXSPI_LUT_SEQ(CMD_SDR, FLEXSPI_1PAD, 0xD8,
RADDR_SDR, FLEXSPI_1PAD, 0x18),
            // Read ECC status
            [4 * NAND_CMD_LUT_SEQ_IDX_READECCSTAT] = FLEXSPI_LUT_SEQ(CMD_SDR, FLEXSPI_1PAD, 0x0F,
CMD_SDR, FLEXSPI_1PAD, 0xC0),
            [4 * NAND_CMD_LUT_SEQ_IDX_READECCSTAT + 1] = FLEXSPI_LUT_SEQ(READ_SDR, FLEXSPI_1PAD,
0x01, STOP, FLEXSPI_1PAD, 0),
        },
    },
    .pageDataSize = 2048,
    .pageTotalSize = 4096,
    .pagesPerBlock = 64,
};
```

12.5.4 Parallel NAND Flash Boot via SEMC

The boot ROM supports a number of Parallel SLC NAND flash devices from different vendors. Both the Error Correction and Control (ECC) module in NAND device and software ECC algorithm (SECDED) in boot ROM can be used to detect the errors, based on the fuse settings.

12.5.4.1 Parallel NAND eFuse Configuration

The boot ROM determines the configuration of external Parallel NAND flash by parameters, either provided by eFuse, or sampled on GPIO pins, during boot. See below table for parameters details:

Table 67. Fuse definition for Parallel NAND over SEMC

Fuse	Definitions	Settings	Shipped Value
BOOT_CFG5[0]	SEARCH_COUNT	Boot Search count of FCB and DBBT 0 – 1 1 – 2	0
BOOT_CFG5[1]	PORT_WIDTH	I/O port size 0 – 8bit 1 – 16bit	
BOOT_CFG5[2]	ECC_SEL	ECC algorithm selection 0 - NAND Flash built-in ECC 1 – SEMC built-in ECC **NOTE** For “ECC selection” option, it can only be set as NAND Flash built-in ECC When the device has built-in ECC module and the ECC module is enabled by default.	

Table continues on the next page...

Table 67. Fuse definition for Parallel NAND over SEMC (continued)

Fuse	Definitions	Settings	Shipped Value
BOOT_CFG5[3]	ECC_MODE	SEMC BCH ECC mode 0 – BCH4 1 – BCH8	
BOOT_CFG5[4]	EDO_MODE	SEMC EDO mode 0 – EDO mode 1 – Non-EDO mode	
BOOT_CFG5[5]	ACCESS_CMD	SEMC access command 0 – IPG 1 – AXI	
BOOT_CFG5[6]	RDY_POLARITY	RDY pin active polarity 0 – High active 1 – Low active	
BOOT_CFG5[7]	RDY_CHECK_TYPE	RDY pin check type 0 – Via status register 1 – Via R/B# pin	
BOOT_CFG5[10:8]	TIMING_MODE	ONFI timing mode 000 - Mode0 -10MHz 001 - Mode1 -20MHz 010 - Mode2 -28MHz 011 - Mode3 -33MHz 100 - Mode4 -40MHz 101 - Mode5 -50MHz 11x - Fastest Mode	
BOOT_CFG5[11]	Reserved	Must be 0	
BOOT_CFG5[14:12]	PCS_SEL	PCS selection 000 - CS0 001 - CS1 010 - CS2 011 - CS3 100 – A8 Others – CSX0	

Table continues on the next page...

Table 67. Fuse definition for Parallel NAND over SEMC (continued)

Fuse	Definitions	Settings	Shipped Value
BOOT_CFG5[15]	Reserved	Must be 0	
BOOT_CFG5[19:16]	BOOT_SEARCH_STRIDE	Search stride for FCB and DBBT in terms of page 0000 – 64 pages Others – 2 ^ BOOT_SEARCH_STRIDE pages	
BOOT_CFG5[22:20]	ROW_COL_ADDR_MODE	Row and column address mode, applicable only for non-ONFI devices 00x – 5 bytes (CA2+RA3) 010 – 4 bytes (CA2+RA2) 011 – 3 bytes (CA2+RA1) 10x – 4 bytes (CA1+RA3) 110 – 3 bytes (CA1+RA2) 111 – 2 bytes (CA1+RA1)	
BOOT_CFG5[23]	STATUS_CMD_TYPE	Status command type, applicable only for non-ONFI device 0 – Common(0x70) 1 – Enhanced(0x78)	
BOOT_CFG5[26:24]	COL_ADDRESS_WIDTH	Column address width, applicable only for non-ONFI device 000 – 12 bits 001 – 09 bits 010 – 10 bits 011 – 11 bits 100 – 13 bits 101 – 14 bits 110 – 15 bits 111 – 16 bits	
BOOT_CFG5[27]	DEV_ECC_INIT_STATUS	Device ECC initial status，applicable only for non-ONFI device 0 – Enabled 1 – Disabled	
BOOT_CFG5[30:29]	PAGES_IN_BLOCK	Pages in block, applicable only for non-ONFI device 000 – 128 pages 001 – 8 pages	

Table continues on the next page...

Table 67. Fuse definition for Parallel NAND over SEMC (continued)

Fuse	Definitions	Settings	Shipped Value
		010 – 16 pages 011 – 32 pages 100 – 64 pages 101 – 256 pages 110 – 512 pages 111 – 1024 pages	
BOOT_CFG5[31]	ONFI_COMPLIANCE_SEL	NAND ONFI compliant 0 – ONFI 1.0 1 – Non-ONFI	

12.5.4.2 Parallel NAND Flash Boot Control Blocks (BCB)

There are two BCB data structures:

- Firmware Configuration Block (FCB)
- Discovered Bad Blocks Table (DBBT)

As part of the Parallel NAND media initialization, the ROM driver uses proper Parallel NAND parameters specified by FUSE to search for an FCB that contains the complete Parallel NAND parameters, page address of DBBT Search Area and Image info, including image copies, start page address and image size in terms of pages for each image.

FCB data structure is protected using Embedded ECC module in Parallel NAND devices or software ECC in ROM. The ROM driver reads 2048 bytes of first sector and checks the ECC check status to determine whether FCB data is valid or not.

If the FCB is found, the complete NAND parameters (Parallel NAND configuration block) are loaded for further reads, if the ECC fails, or the fingerprint does not match, or the CRC checksum does not match, the Block Search state machine increments page number to Search Stride number of pages to read for the next FCB until Search Count pages have been read.

If search fails to find a valid FCB, the Parallel NAND driver responds with an error and the boot ROM enters into Recovery boot mode (Secondary boot, if it is enabled, or Serial download mode).

The FCB contains the page address of DBBT Search Area, and the info for images. DBBT is searched in DBBT Search area just like how FCB is searched. After the FCB is read, the DBBT is loaded, then the boot image is loaded using starting page address from FCB.

12.5.4.3 Firmware Configuration Block (FCB)

The FCB is at the first page in the first good block. The FCB should be present at each search stride of the search area.

The search area contains copies of the FCB at each stride distance, in case the first Serial NAND block becomes corrupted, the ROM will find its copy in the next Parallel NAND block. The search area should span over at least two Parallel NAND blocks. The location information for DBBT search area and images are all specified in the FCB. The following table shows the FCB Structure.

Table 68. FCB Structure

Name	offset	Size (Bytes)	Description
bcbHeader	0x000	12	See Table 69 for details

Table continues on the next page...

Table 68. FCB Structure (continued)

DBBTSerachAreaStartPage	0x00c	4	Start Page address for bad block table search area
searchStride	0x010	2	Search stride for DBBT and FCB search. Not used by ROM Max value is 8.
searchCount	0x012	2	Copies of DBBT and FCB. Not used by ROM, max value is 8.
firmwareCopies	0x014	4	Firmware copies Valid range 1-4.
-	0x018	40	Reserved
firmwareTable	0x040	64	For details see Table 70
-	0x080	128	Reserved
nandConfig	0x100	256	Parallel NAND configuration block over SEMC
-	0x200	512	Reserved

Table 69. Header Description

Field	Size	Description
crcChecksum	4	Checksum
fingerprint	4	0x4E46_4342 ASCII: “NFCB”
version	4	0x0000_0001

Table 70. Table Descriptions

Field	Size	Description
startPage	4	Start page of this firmware
pagesInFirmware	4	Pages in this firmware

NOTE

- The “crcChecksum” is calculated with an MPEG2 variant of CRC-32.
- The “crcChecksum” calculation starts from fingerprint to the end of FCB, 1020 bytes in total.
- The “nandConfig” is SEMC NAND configuration block which consists of common SEMC memory configuration block and Parallel NAND specified configuration parameters. See Parallel NAND eFuse Configuration for more details.

12.5.4.4 Discovered Bad Blocks Table (DBBT)

Table 71. DBBT Structure

Name	offset	Size (Bytes)	Description
bcbHeader	0x000	12	See Table 72 for details
-	0x00C	4	Reserved
badBlockNumber	0x010	4	Number of bad blocks
-	0x014	12	Reserved
badBlockTable	0x020	1024 (256*4)	Each bad block entry is a 32-bit value specifying the number of a found bad block. The number of valid bad block entries is specified by the badBlockNumber field, where valid bad block entries are stored sequentially starting at the beginning of the bad block entries segment. Unused bad block entries (those beyond the badBlockNumber) should be filled with 0xFs.

Table 72. Header Description

Field	Size	Description
crcChecksum	4	Checksum
fingerprint	4	0x4442_4254 ASCII: “DBBT”
version	4	0x0000_0001

NOTE

- Maximum bad block number is 256.
- The "crcChecksum" is calculated with the same algorithm as the one in FCB, from Fingerprint to the end of DBBT, 1052 bytes in total.

12.5.4.5 Bad Block Handling in ROM

During the firmware boot, at the block boundary, the Bad Block table is searched for a match to the next block.

If no match is found, the next block can be loaded. If a match is found, the block must be skipped and the next block checked.

12.5.5 Parallel NAND Configuration based on SEMC Interface

The ROM SW supports Parallel NAND based on SEMC module, using an 80-bytes common SEMC configuration block and several specified parameters for Parallel NAND. See the following sections for more details.

12.5.5.1 SEMC Configuration Block

SEMC Configuration block consists of all parameters related to specific Flash devices.

Table 73. SEMC control block structure

Name	offset	Size (Bytes)	Description
tag	0x000	4	0x434D4553, ascii:"SEMC"
version	0x004	4	0x00010000 [07:00] bugfix = 0 [15:08] minor = 0 [31:16] major = 1
deviceMemType	0x008	1	0 – NOR Flash 1 – NAND Flash
accessCommandType	0x009	1	0 – IPG bus command 1 – AXI32 command
-	0x00A	2	Reserved
asyncClkFreq	0x00C	1	0 – 33MHz 1 – 40MHz 2 – 50MHz 3 – 66MHz 4 – 108MHz 5 – 133MHz 6 – 166MHz
busTimeoutCycles	0x00D	1	0 – 255 * 1024 cycles n – n * 1024 cycles
commandExecutionTimeoutCycles	0x00E	1	0 – 256 * 1024 cycles n – n * 1024 cycles
readStrobeMode	0x00F	1	0 – Dummy read strobe loopbacked internally 1 – Dummy read strobe loopbacked from DQS pad
nandMemConfig	0x050	64	See detail in Table 74

Table 74. SEMC NAND control block structure

Name	offset	Size (Bytes)	Description
axiMemBaseAddress	0x00	4	SoC level Base Address for NAND AXI command
axiMemSizeInByte	0x04	4	SoC level Memory size for NAND AXI command

Table continues on the next page...

Table 74. SEMC NAND control block structure (continued)

Name	offset	Size (Bytes)	Description
ipgMemBaseAddress	0x08	4	SoC level Base Address for NAND IPG command
ipgMemSizeInByte	0x0c	4	SoC level Memory size for NAND IPG command
edoMode	0x10	1	0 - EDO mode disabled 1 - EDO mode enabled
ioPortWidth	0x11	1	IO Port bit number
arrayAddressOption	0x12	1	0 – 5 bytes (CA2+RA3) 1 – 4 bytes (CA1+RA3) 2 – 4 bytes (CA2+RA2) 3 – 3 bytes (CA1+RA2) 4 – 3 bytes (CA2+RA1) 7 – 2 bytes (CA1+RA1)
columnAddressWidth	0x13	1	Column address bit number
burstLengthInBytes	0x14	1	Burst Length
columnAddressOption	0x15	1	0 - PageAreaAccess 1 - SpareAreaAccess
-	0x16	10	Reserved
cePortOutputSelection	0x20	1	0 – CSX0 1 – CSX1 2 – CSX2 3 – CSX3 4 – A8
rdyPortPolarity	0x21	1	0 – Low active 1 – High active
-	0x22	14	Reserved
ceSetupTime	0x30	1	value[3:0] + 1 cycles
ceMinHoldTime	0x31	1	value[3:0] + 1 cycles
ceMinIntervalTime	0x32	1	value[3:0] + 1 cycles
weLowTime	0x33	1	value[3:0] + 1 cycles
weHighTime	0x34	1	value[3:0] + 1 cycles
reLowTime	0x35	1	value[3:0] + 1 cycles
reHighTime	0x36	1	value[3:0] + 1 cycles

Table continues on the next page...

Table 74. SEMC NAND control block structure (continued)

Name	offset	Size (Bytes)	Description
weHighToReLowTime	0x37	1	value[5:0] + 1 cycles
reHighToWeLowTime	0x38	1	value[5:0] + 1 cycles
aleToDataStartTime	0x39	1	value[5:0] + 1 cycles
readyToReLowTime	0x3a	1	value[5:0] + 1 cycles
weHighToBusyTime	0x3b	1	value[5:0] + 1 cycles
asyncTurnaroundTime	0x3c	1	value[3:0] + 1 cycles
-	0x3d	3	Reserved

12.5.5.2 Parallel NAND Configuration Block (256 bytes)

Table 75. Parallel NAND control block structure

Name	offset	Size (Bytes)	Description
memConfig	0x000	80	See SEMC control block structure for more details
vendorType	0x050	1	0 – Micron 1 – Spansion 2 – Samsung 3 – Winbond 4 – Hynix 5 – Toshiba 6 – Macronix
cellTechnology	0x051	1	0 – SLC 1 – MLC
onfiVersion	0x052	1	0 – Non-ONFI 1 – ONFI 1.0 2 – ONFI 2.0 3 – ONFI 3.0 4 – ONFI 4.0
acTimingTableIndex	0x053	1	0 – User Defined 1 – ONFI 1.0 Mode0 10MHz 2 – ONFI 1.0 Mode1 20MHz 3 – ONFI 1.0 Mode2 28MHz 4 – ONFI 1.0 Mode3 33MHz 5 – ONFI 1.0 Mode4 40MHz

Table continues on the next page...

Table 75. Parallel NAND control block structure (continued)

Name	offset	Size (Bytes)	Description
			6 – ONFI 1.0 Mode5 50MHz 7 – Auto Detection
enableEccCheck	0x054	1	0 – Enabled 1 – Disabled
eccCheckType	0x055	1	0 – Device ECC 1 – SemcBCHECC
deviceEccStatus	0x056	1	0 – Enabled 1 – Disabled
-	0x057	1	Reserved
swEccBlockBytes	0x058	4	Software ECC block bytes (256, 512)
readyCheckOption	0x05c	1	0 – Via Status Register 1 – Via R/B# signal
statusCommandType	0x05d	1	0 – Common (0x70) 1 – Enhanced (0x78)
readyCheckTimeoutInMs	0x05e	2	Ready Check timeout
readyCheckIntervalInUs	0x060	2	Ready Check interval
userOnfiAcTimingModeCode	0x080	1	userOnfiAcTimingModeCode
-	0x081	31	Reserved
bytesInPageDataArea	0x0a0	4	Page Main data size
bytesInPageSpareArea	0x0a4	4	Page Spare data size
pagesInBlock	0x0a8	4	Page number in one block
blocksInPlane	0x0ac	4	Block number in one plane
planesInDevice	0x0b0	4	Plane number in Device
-	0x0b4	44	Reserved
enableReadbackVerify	0x0e0	1	0 - Enabled 1 - Disabled
-	0x0e1	3	Reserved
readbackPageBufferAddress	0x0e4	4	Read back page buffer address
-	0x0e8	24	Reserved

12.5.6 eMMC/SD Boot via uSDHC

The boot ROM supports booting from the eMMC and SD compliant devices via either uSDHC ports. The program image will be copied by boot ROM from devices to RAM where the code execution occurs.

The eMMC and SD devices shall match the following standards:

- eMMC which matches JESD84(Embedded Multi-Media Card Electrical Standard) Version 4.2/4.3/4.4/4.41/4.5/5.0
- SDSC, SDHC and SDXC card, which match SD Specifications Version 3.0.

eMMC fast boot mode and SD UHSI mode are supported on all uSDHC ports. eMMC fast boot mode requires JESD84 Version 4.3 or higher. SD UHSI mode is not applied to SDSC card, and can be applied to SDHC and SDXC card.

eMMC multiple partition is supported by fast boot and normal boot, which requires JESD84 Version 4.4 or higher.

12.5.6.1 eMMC/SD Device FUSE Configuration

The ROM determines the configuration of eMMC and SD by the parameters provided by fuses.

Table 76 shows the details of the configurations for eMMC boot.

Table 76. uSDHC Fuse Descriptions for eMMC

Fuse	Definition	Settings	Shipped value
BOOT_CFG4[0]	USDHC_PORT	0– uSDHC1 is used for eMMC boot 1 – uSDHC2 is used for eMMC boot	0
BOOT_CFG4[1]	SION_DISABLE	0 – SION is enabled for uSDHC.CMD line 1 – SION is disabled for uSDHC.CMD line	0
BOOT_CFG4[2]	USDHC1_RST_POLARITY	0 – uSDHC1.RST is active low 1 – uSDHC1.RST is active high	0
BOOT_CFG4[3]	USDHC2_RST_POLARITY	0 – uSDHC2.RST is active low 1 – uSDHC2.RST is active high	0
BOOT_CFG4[5:4]	PWR_CYCLE	uSDHC2.RST active time 0 – 20ms 1 – 10ms 2 – 5ms 3 – 2.5ms	0
BOOT_CFG4[6]	PWR_CYCLE_EN	0 – No power cycle is performed 1 – Power cycle is issued via uSDHCx.RST	0
BOOT_CFG4[7]	PWR_STABLE_CYCLE	Stable time after uSDHC2.RST is turned from active to inactive 0 – 5ms 1 – 2.5ms	0
BOOT_CFG4[8]	MMC_SPEED	eMMC bus speed mode	0

Table continues on the next page...

Table 76. uSDHC Fuse Descriptions for eMMC (continued)

Fuse	Definition	Settings	Shipped value
		0 – Backwards Compatibility with legacy MMC card, Max to 26MHz 1 – High Speed, 52MHz	
BOOT_CFG4[9]	NO_PREIDLE_STATE	0 – Issue GO_PRE_IDLE_STATE command(CMD0 with argument of 0xF0F0F0F0) before starting fastboot 1 – Do not issue GO_PRE_DILE_STATE command	0
BOOT_CFG4[10]	USDHC1_VSEL	Configure the uSDHC1.VSELECT pin output 0 – uSDHC1.VSELECT = 0, 3.3V is supplied to eMMC VCCQ 1 – uSDHC1.VSELECT = 1, 1.8V is supplied to eMMC VCCQ	0
BOOT_CFG4[11]	USDHC2_VSEL	Configure the uSDHC2.VSELECT pin output 0 – uSDHC2.VSELECT = 0, 3.3V is supplied to eMMC VCCQ 1 – uSDHC2.VSELECT = 1, 1.8V is supplied to eMMC VCCQ	0
BOOT_CFG4[13:12]	MMC_BUS_WIDTH	eMMC bus width mode 0 – 4bit 1 – 8bit 2 – 4bit DDR mode 3 – 8bit DDR mode	0
BOOT_CFG4[14]	MMC_FAST_BOOT_EN	eMMC boot mode selection 0 – Select eMMC normal boot 1 – Select eMMC fast boot	0
BOOT_CFG4[15]	MMC_FAST_BOOT_ACK_EN	0 – eMMC fast boot acknowledge is disabled	0
BOOT_CFG4[31:16]	Reserved	Must be 0	0

Table 77 shows the details of the configurations for SD boot.

Table 77. uSDHC Fuse Descriptions for SD

Fuse	Definition	Settings	Shipped value
BOOT_CFG6[0]	USDHC_PORT	0 – uSDHC1 is used for SD boot 1 – uSDHC2 is used for SD boot	0

Table continues on the next page...

Table 77. uSDHC Fuse Descriptions for SD (continued)

Fuse	Definition	Settings	Shipped value
BOOT_CFG6[1]	SION_DISABLE	0 – SION is enabled for uSDHC.CMD line 1 – SION is disabled for uSDHC.CMD line	0
BOOT_CFG6[2]	USDHC1_RST_POLARITY	0 – uSDHC1.RST is active low 1 – uSDHC1.RST is active high	0
BOOT_CFG6[3]	USDHC2_RST_POLARITY	0 – uSDHC2.RST is active low 1 – uSDHC2.RST is active high	0
BOOT_CFG6[5:4]	PWR_CYCLE	uSDHC2.RST active time 0 – 20ms 1 – 10ms 2 – 5ms 3 – 2.5ms	0
BOOT_CFG6[6]	PWR_CYCLE_EN	0 – No power cycle is performed 1 – Power cycle is issued via uSDHCx.RST	0
BOOT_CFG6[7]	PWR_STABLE_CYCLE	Stable time after uSDHC2.RST is turned from active to inactive 0 – 5ms 1 – 2.5ms	0
BOOT_CFG6[9:8]	SD_SPEED	SD speed mode. 0 – Default speed(SDR12), up to 25MHz, 3.3V signaling 1 – High speed(SDR25), up to 50MHz, 3.3V signaling 2 – SDR50, up to 100MHz, 1.8V signaling 3 – SDR104, up to 208MHz, 1.8V signaling	0
BOOT_CFG6[10]	SD_BUS_WIDTH	SD but width 0 – 1bit 1 – 4bit	0
BOOT_CFG6[21:11]	Reserved	Must be 0	0
BOOT_CFG6[23:22]	DLL_DELAY_CALIB_STEP	The increasing value for DLL delay calibration during tuning procedure 00b – 1 01b – 2 10b – 4	0

Table continues on the next page...

Table 77. uSDHC Fuse Descriptions for SD (continued)

Fuse	Definition	Settings	Shipped value
		11b – 8	
BOOT_CFG6[30:24]	DLL_DELAY_CALIB_START	The start value for DLL delay calibration during tuning procedure	0
BOOT_CFG6[31]	DLL_DELAY_OVERRIDE	Control if the values in DLL_DELAY_CALIB_STEP and DLL_DELAY_CALIB_START will overwrite the default ones (DLL_DELAY_CALIB_STEP=2, DLL_DELAY_CALIB_START=10) in ROM	0

12.5.6.2 eMMC Boot

The boot ROM supports booting from eMMC normal mode and boot mode.

12.5.6.2.1 eMMC Normal Boot

At the beginning of the initialization, the eMMC frequency is set to 300KHz. ROM puts the eMMC card into the device identification mode by performing a power cycle and sending CMD0 with the argument of 0xF0F0F0F0. Power cycle is controlled by the fuse bits PWR_CYCLE, PWR_CYCLE_EN and PWR_STABLE_CYCLE.

When the eMMC card enters the device identification mode, the boot ROM validates the operation voltage range and the access mode, identifies the device information, and finally assign a Relative Device Address(RCA) to the eMMC card.

The eMMC card enters the data transfer mode once a RCA is assigned to it. The boot ROM switches to the max frequency supported by normal mode(26MHz) to perform the later initializations.

After the initialization phase is complete, the boot ROM switches to the eMMC card clock to the final frequency selected by MMC_SPEED.

For the eMMC card that supports partition management(eMMC version 4.4 or higher), the boot ROM supports booting the images from the selected partitions. The boot ROM reads the BOOT_PARTITION_ENABLE field in EXT_CSD[179] to get the boot partition to be set. If there is no boot partition mentioned in the BOOT_PARTITION_ENABLE field or the user partition was mentioned, the ROM boots from the user partition.

12.5.6.2.2 eMMC Fast Boot

For the eMMC card that supports the boot mode (eMMC version 4.4 or higher), the boot ROM supports fast boot to skip the eMMC card initialization progress to reduce the boot time.

ROM puts the eMMC card into the boot mode by holding the CMD line for 74 clock cycles and more after performing a power cycle and sending CMD0 with the argument of 0xF0F0F0F0. Then the eMMC card starts to send the first boot data to the master on the DATA lines according to the non-volatile configurations in EXT_CSD.

The partition that is used for data transferring is selected by the BOOT_PARTITION_ENABLE field in EXT_CSD[179].

The data size that can be transferred in the boot mode is defined by the BOOT_SIZE_MULT field in EXT_CSD[226].

The speed and bus width are determined by the BOOT_MODE and BOOT_BUS_WIDTH field in EXT_CSD[177].

The boot acknowledge can be enabled by the BOOT_ACK field in EXT_CSD[179].If the boot acknowledge is enabled, the eMMC device has to send the acknowledge pattern [3b010] via the DATA lines within 50ms after the CMD line goes LOW. And the ROM waits 50ms to read the acknowledge patter to identify if the eMMC device enters the boot mode. If boot acknowledge is disabled, the ROM waits 1 second for data. If the BOOT ACK or data was received, the eMMC device is booted in the "boot mode", otherwise the ROM treats it as a boot failure.

The fast boot mode can be selected by the MMC_FAST_BOOT_EN fuse. The boot acknowledge is selected by the MMC_FAST_BOOT_ACK_EN fuse. The fuses, MMC_FAST_BOOT_ACK_EN, MMC_SPEED and MMC_BUS_WIDTH must match the relative configurations in BOOT_ACK, BOOT_MODE and BOOT_BUS_WIDTH. The BOOT_PARTITION_ENABLE must be specified to enable partition used in boot mode.

Despite the setting of BOOT_SIZE_MULT, the maximum image size supported in the eMMC fast boot mode is 32 MB. This is due to a limited number of uSDHC ADMA Buffer Descriptors allocated by the ROM.

12.5.6.3 SD Boot

After the normal boot mode initialization begins, the frequency is set to 300 kHz. During the identification phase, the boot ROM validates the operation voltage range, performs the voltage switch sequence if UHSI mode is selected.

The capacity of the card is also checked. The boot code supports the high-capacity and low-capacity SDSC/SDHC/SDXC cards after the voltage validation card initialization is done.

For the UHSI cards, the clock speed fuses can be set to SDR50 or SDR104 on USDHC1, USDHC2 ports. This enables the voltage switch process to set the signaling voltage to 1.8 V during the voltage validation. The bus width is fixed at a 4-bit width and a sampling point tuning process is needed to calibrate the number of the delay cells.

Comparing SDR12 and SDR25 mode, the boot time of SD UHSI mode might be increased when booting a small image, due the voltage switch process and tuning process might cost more time than the reduce of a higher speed. The calibration start value (DLL_DELAY_CALIB_START) and the step value DLL_DELAY_CALIB_STEP can be set to optimize the sample point tuning process. But it is suggested to use SD UHSI mode for large images(over 10MB) booting.

If the PWR_CYCLE_EN fuse is set, the ROM sets the uSDHC.RST low, waits for several milliseconds which determined by PWR_CYCLE fuse, and then sets the uSDHC.RST high and wait for several milliseconds which determined by PWR_STABLE_CYCLE fuse. If the uSDHC.RST is connected to the SD power supply enable logic on board, it enables the power cycle of the SD card. This may be crucial in case the SD logic is in the 1.8 V states and must be reset to the 3.3 V states.

12.5.6.4 eMMC/SD Boot Flow

The eMMC/SD boot flow is detailed in the following figures.

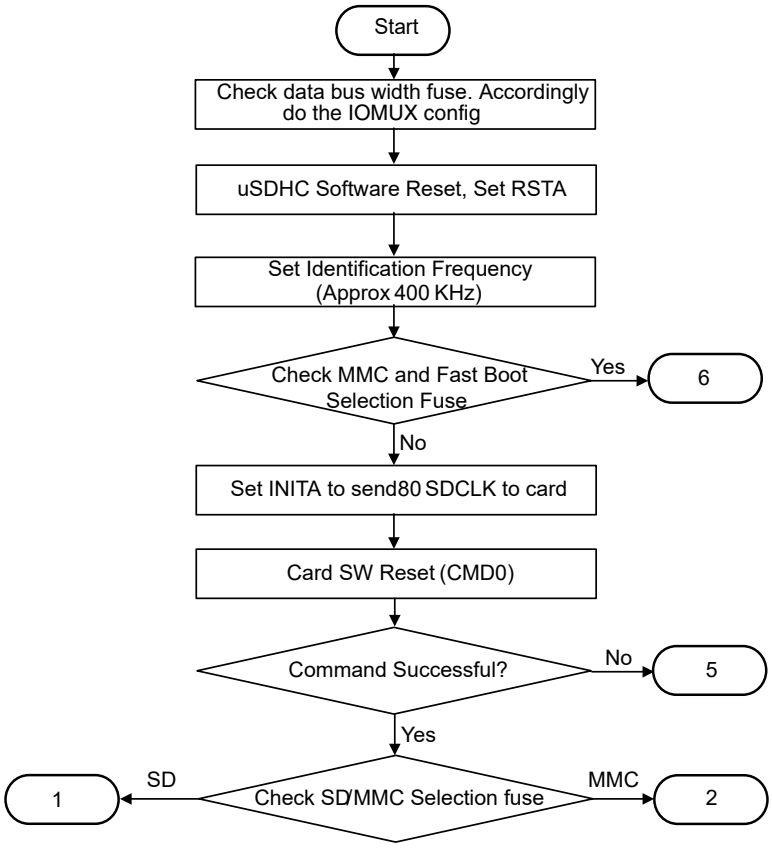


Figure 42. Expansion device boot flow (1 of 6)

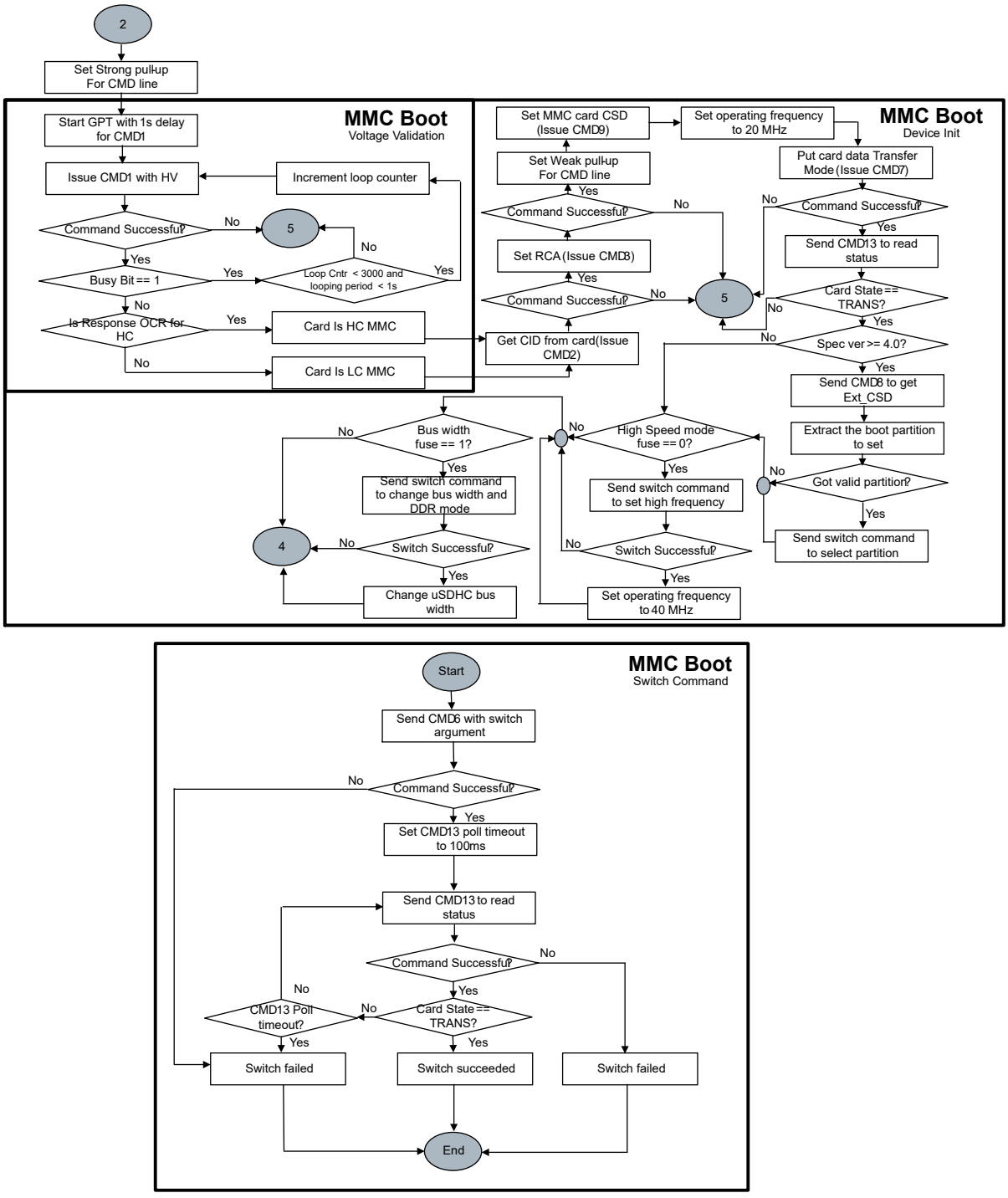


Figure 43. Expansion device (MMC) boot flow (2 of 6)

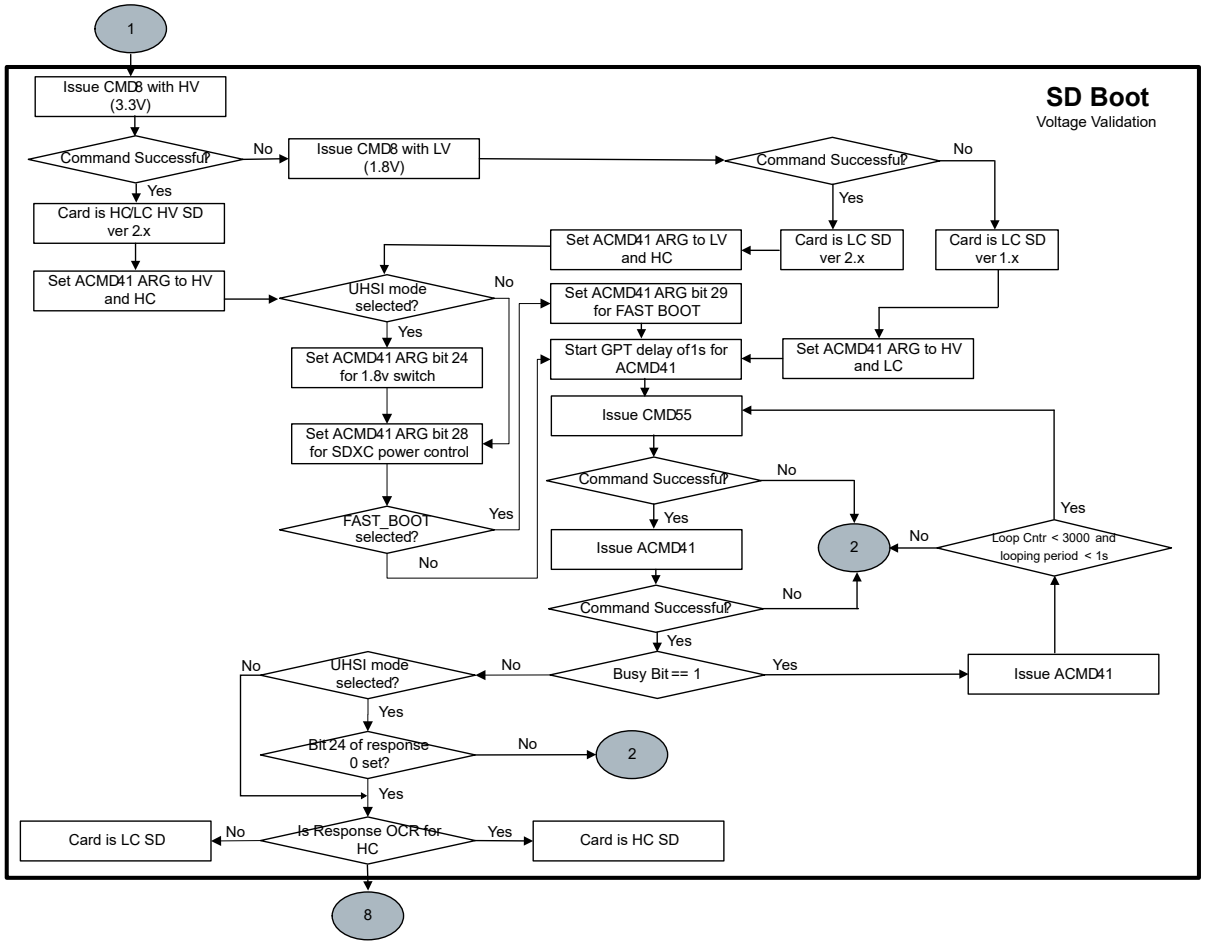


Figure 44. Expansion device (SD/eSD/SDXC) boot flow (3 of 6) part 1

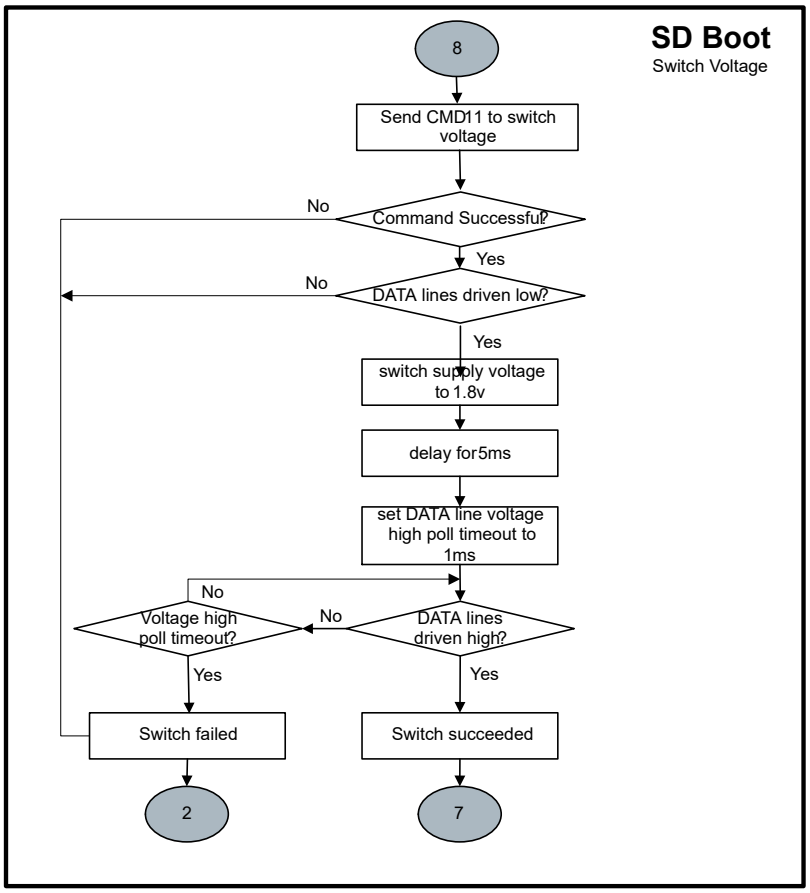


Figure 45. Expansion device (SD/eSD/SDXC) boot flow (3 of 6) part 2

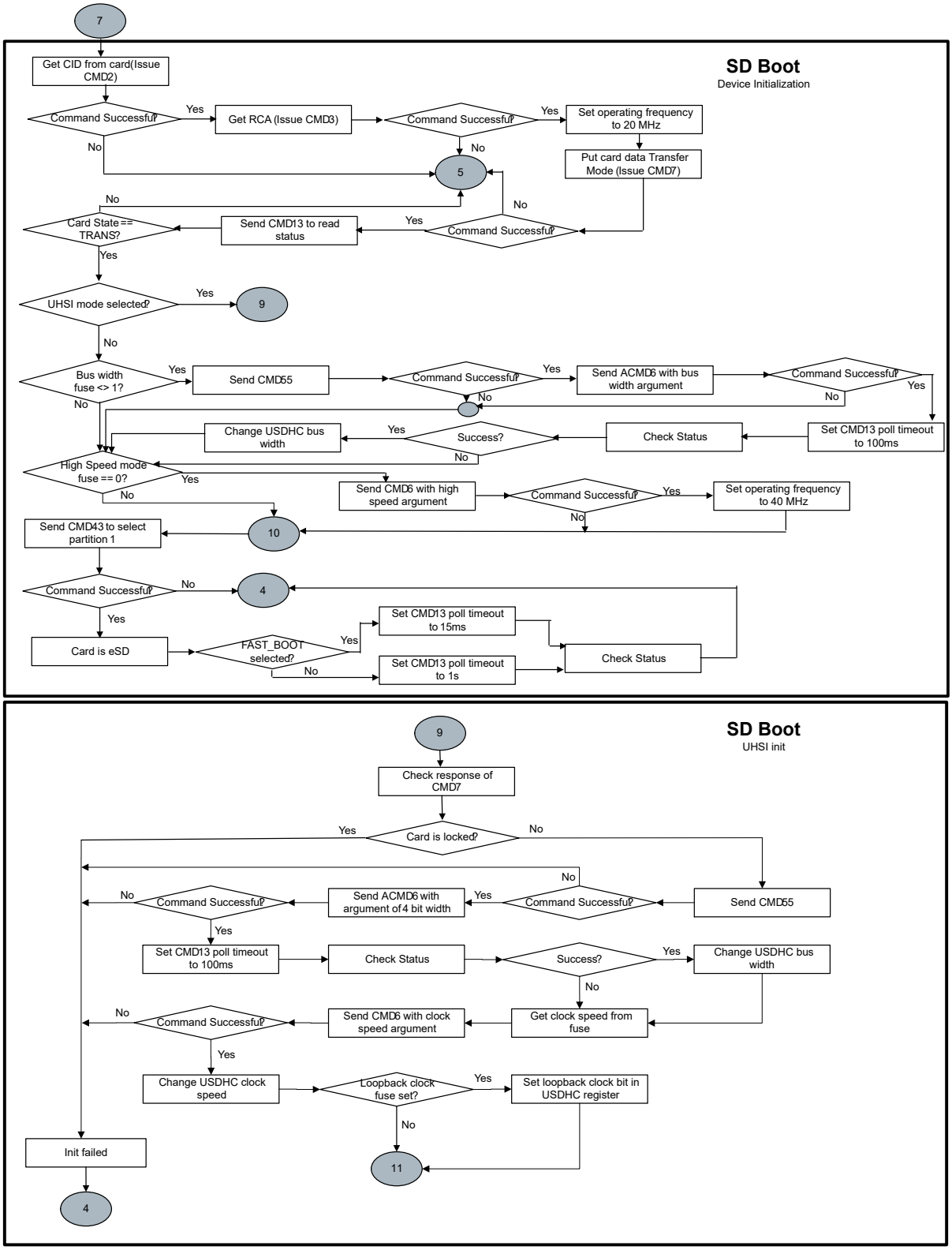


Figure 46. Expansion device (MMCSD/eSD/SDXC) boot flow (4 of 6)

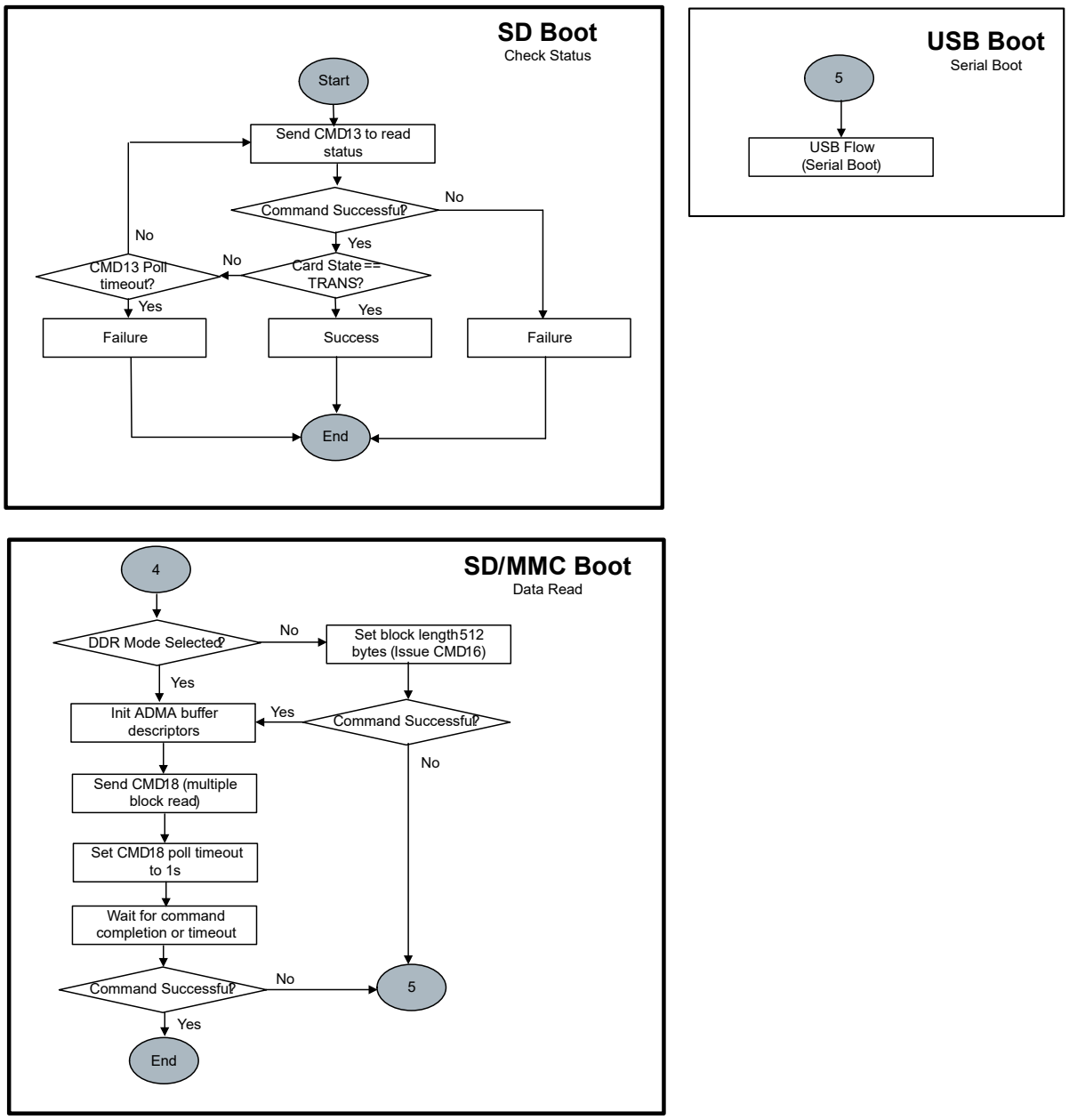
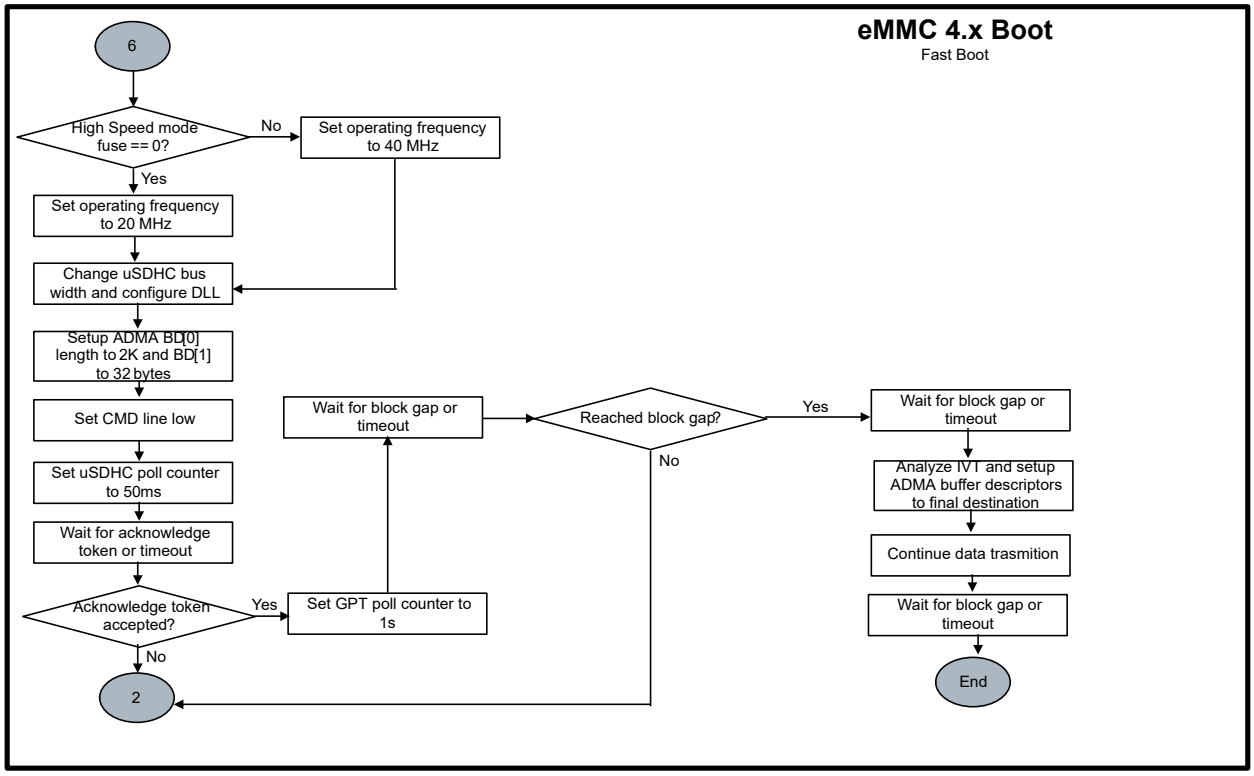


Figure 47. Expansion device (SD/eSD) boot flow (5 of 6)



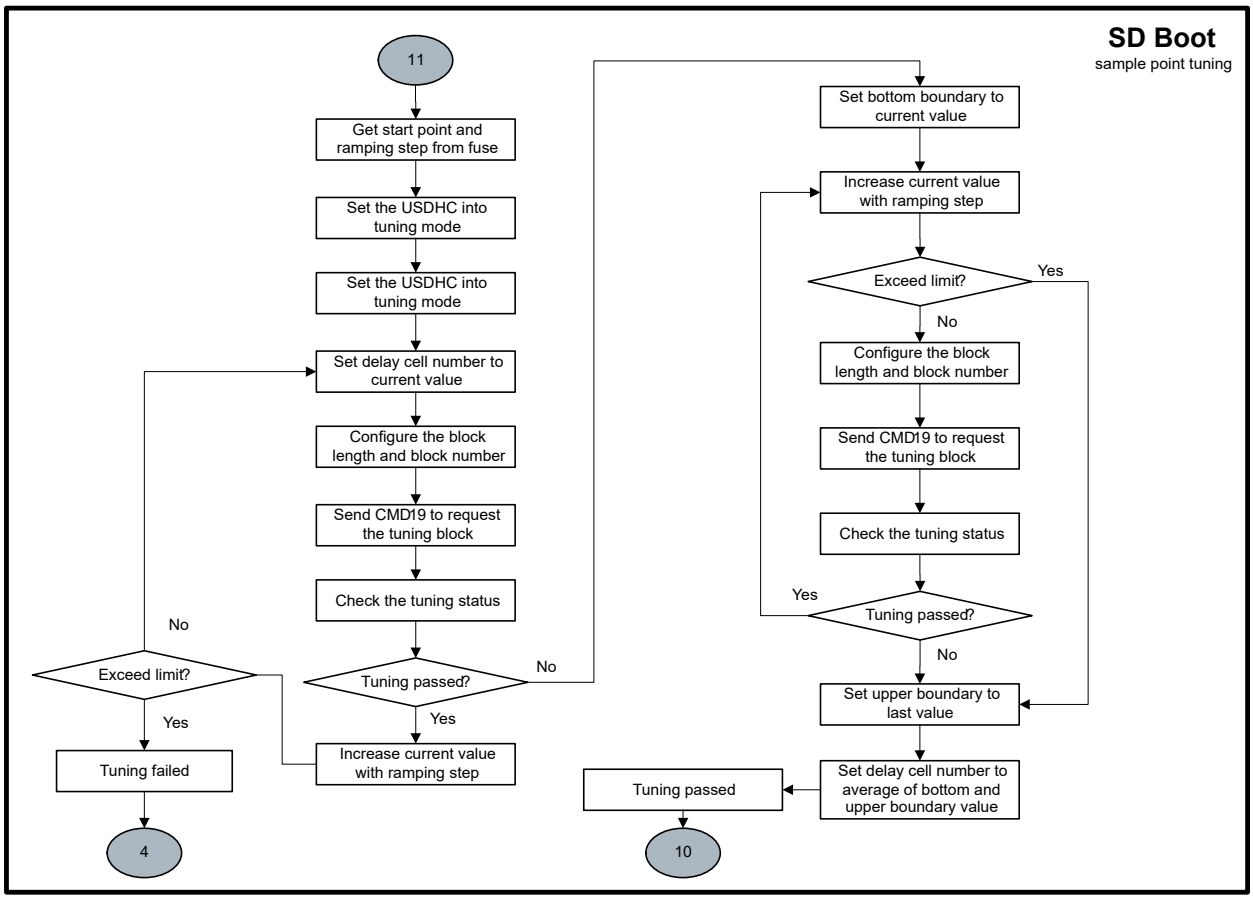


Figure 48. Expansion device boot flow (6 of 6)

12.5.6.5 Redundant Boot Support for eMMC/SD

The ROM supports the redundant boot for an eMMC/SD device. The primary or secondary image is selected, depending on the PERSIST_SECONDARY_BOOT setting. See Table 56.

If the PERSIST_SECONDARY_BOOT is 0, the boot ROM uses address 0x0 for the primary image.

If the PERSIST_SECONDARY_BOOT is 1, the boot ROM reads the secondary image table from address 0x200 on the boot media and uses the address specified in the table.

Table 78. Secondary image table format

Reserved (chipNum)
Reserved (driveType)
tag
firstSectorNumber
Reserved (sectorCount)

Where:

- The tag is used as an indication of the valid secondary image table. It must be 0x00112233.
- The firstSectorNumber is the first 512-byte sector number of the secondary image.

For the secondary image support, the primary image must reserve the space for the secondary image table. See the following figure for the typical structures layout on an expansion device.

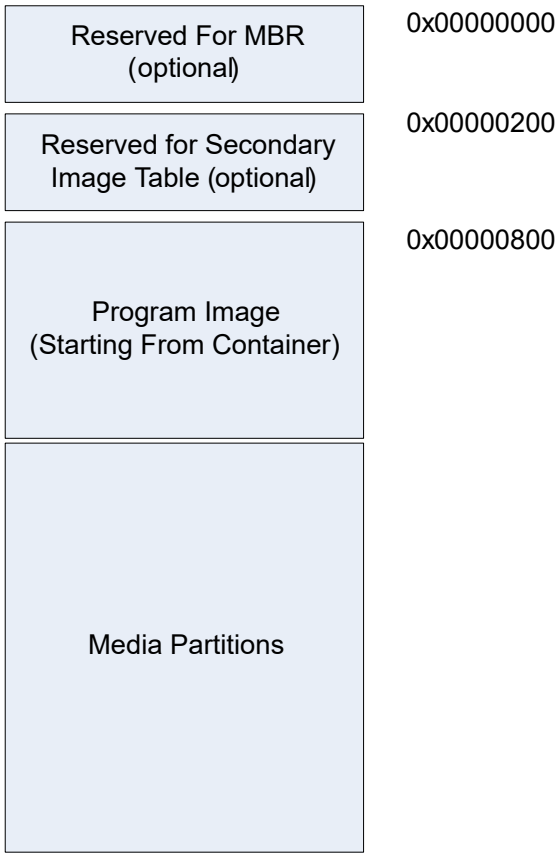


Figure 49. eMMC/SD structures layout

For the Closed mode, if there are failures during primary image authentication, the boot ROM turns on the PERSIST_SECONDARY_BOOT bit (see Table 56) and performs the software reset. After the software reset, the secondary image is used.

12.5.7 Serial NOR/EEPROM Recovery Boot via LPSPI

The chip supports recovery devices, such as serial EEPROM and NOR flash, using the LPSPI. If the primary boot device fails, the boot ROM tries to boot from the recovery device using one of the LPSPI ports.

NOTE

Only 16-bit image offset is supported. If multi-images are used and the image offset is over 16 bits, then combine multi-images into one CM33 image and handle the other images by CM33 application.

12.5.7.1 Serial NOR/EEPROM FUSE Configuration

The boot ROM code determines the type of device using the following parameters, provided by the fuse settings during boot.

Table 79. Serial NOR/EEPROM boot FUSE descriptions

Fuse	Definition	Settings	Shipped Value
BOOT_CFG0[4]	RECOVERY_BOOT_EN	0 - Recovery boot is forbidden 1 - Recovery boot is allowed	0

Table continues on the next page...

Table 79. Serial NOR/EEPROM boot FUSE descriptions (continued)

Fuse	Definition	Settings	Shipped Value
BOOT_CFG0[17:16]	LPSPI_PORT_SEL	0 - LPSPI1 1 - LPSPI2 2 - LPSPI4 3 - LPSPI5	0
BOOT_CFG0[19:18]	LPSPI_SPEED	0 - 30MHz 1 - 10MHz 2 - 30MHz Dual Read (1-2-2, 0xBB) 3 - 30MHz Quad Read (1-4-4, 0xEB)	0
BOOT_CFG0[22:20]	LPSPI_DUMMY_BYTES	0 - Default (4 cycles for dual/6 cycles for quad) *n* (non-zero) - 4*n* cycles for dual/2*n* cycles for quad	0
BOOT_CFG0[23]	LPSPI_INTERNAL_PULL	0 - No internal pull-up 1 - Internal pull-up enabled	0

The LPSPI*n* block can be used as a boot device using the LPSPI interface for the serial ROM boot. The SPI interface is configured to operate at the speed specified by the LPSPI_SPEED fuse field.

The boot ROM tries to copy the boot image container header from serial NOR/EEPROM's offset 0x400, to the MCU's internal RAM. If it is a valid container, ROM then tries to boot with it. External Memory Configuration_Data (XMCD) is supported for recovery boot. For more information on XMCD, refer to the External Memory Configuration Data (XMCD) section.

12.5.8 SD/MMC Manufacturing Boot via uSDHC

When the master/primary boot and recovery boot (if enabled) fail, the boot goes to the SD/MMC manufacture mode before the serial download mode. In the manufacture mode, uSDHC1, 1-bit bus width and SDR12 is used despite of the fuse settings in the BOOT_CFG.

uSDHC.CD pin is configured to detect if SD/MMC is inserted or not. Low means the card is inserted, otherwise, manufacturing boot will be treated as failure, and ROM goes to serial downloader mode.

12.6 Program Image

This section describes the boot image format that is used by ROM. The format names container-based image format.

The top level view of the boot image layout is shown in the figure below.

Table 80. Top-level view of the boot image

Memory Configuration Block	
Container 1	Container Header
	Image Array Entry
	Signature Block
	Padding to 1KB (0x400) Alignment

Table continues on the next page...

Table 80. Top-level view of the boot image (continued)

Container 2(Optional)	Container Header
	Image Array Entry
	Signature Block
	Padding to 1KB (0x400) Alignment
Padding to 8KB(0x2000) Alignment	
NXP Images(Optional)	NXP Firmware
User/OEM Images	Application Data (Optional)
	CM7 Image (Optional)
	CM33 Image

12.6.1 Memory Configuration Block

The memory configuration block is a data region from the beginning of the boot device memory, to the start of the container-based image. The data region contains words used by ROM to initialize the boot devices. It includes BCB, Serial NOR FLASH configuration Block, and XMCD.

The memory configuration block has different size and layout for different boot devices. The different layouts and sizes are provided in the following figures.

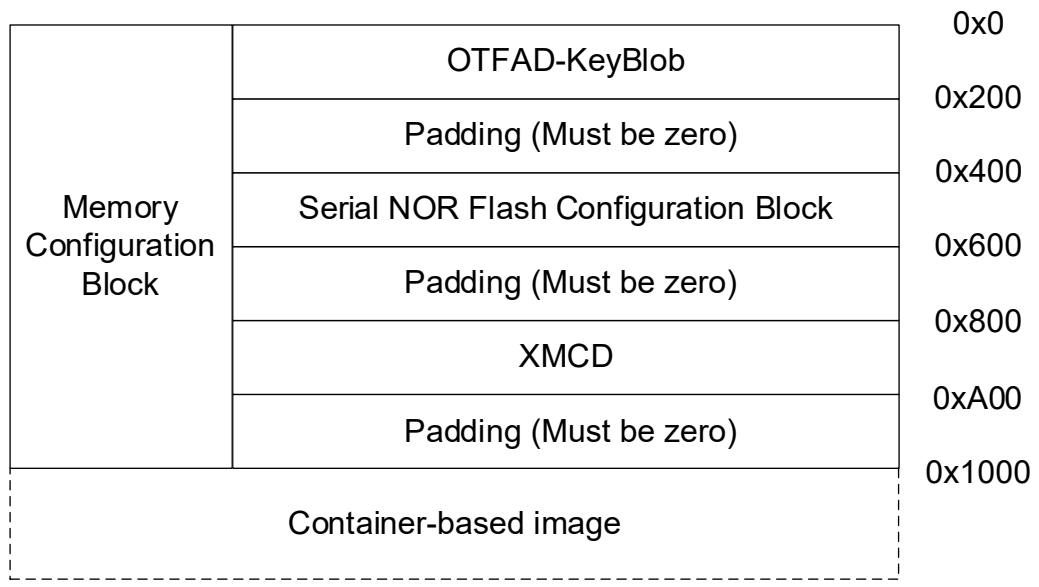


Figure 50. Layout for Serial NOR FLASH boot

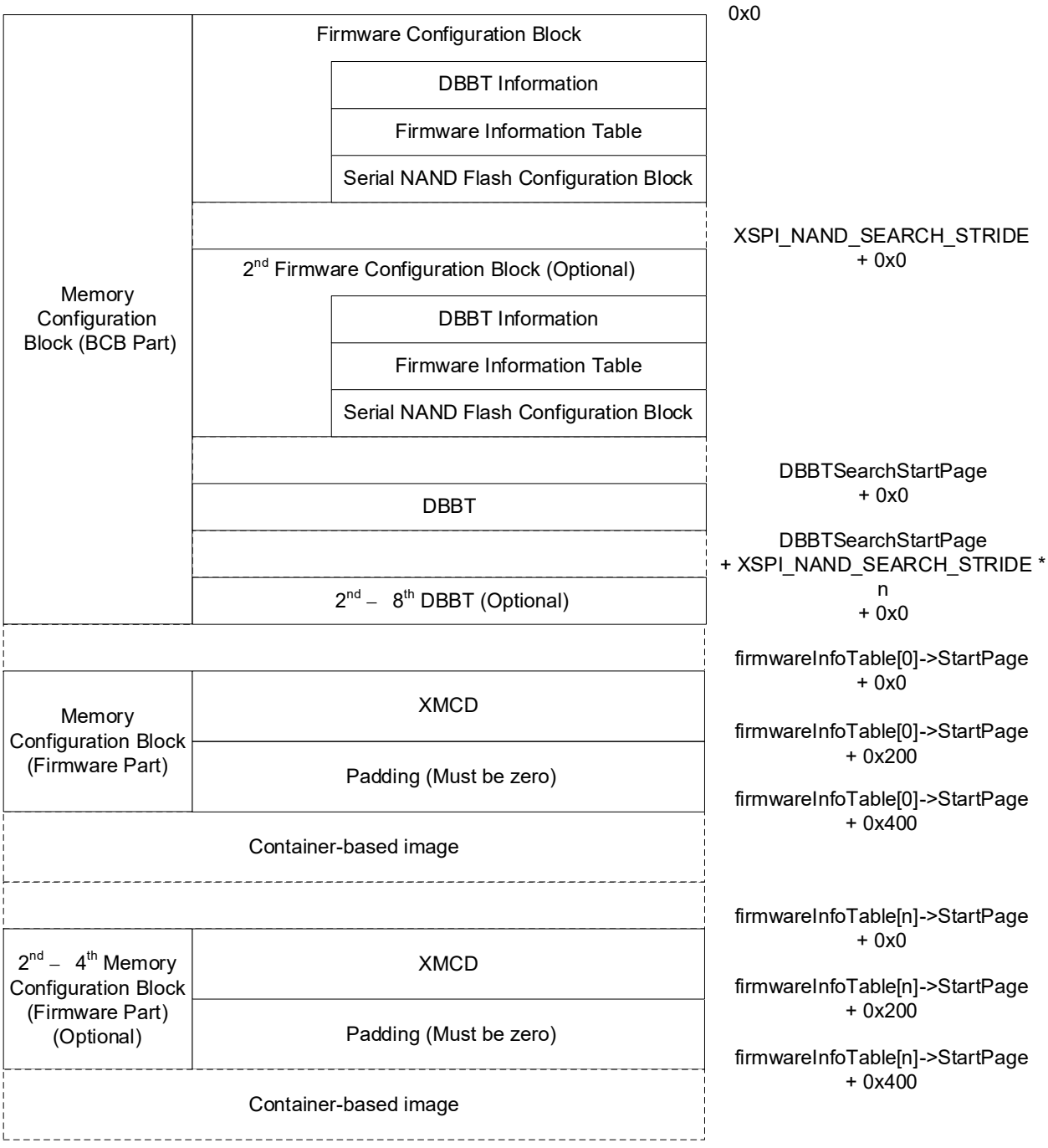


Figure 51. Layout for Serial/Parallel NAND Flash

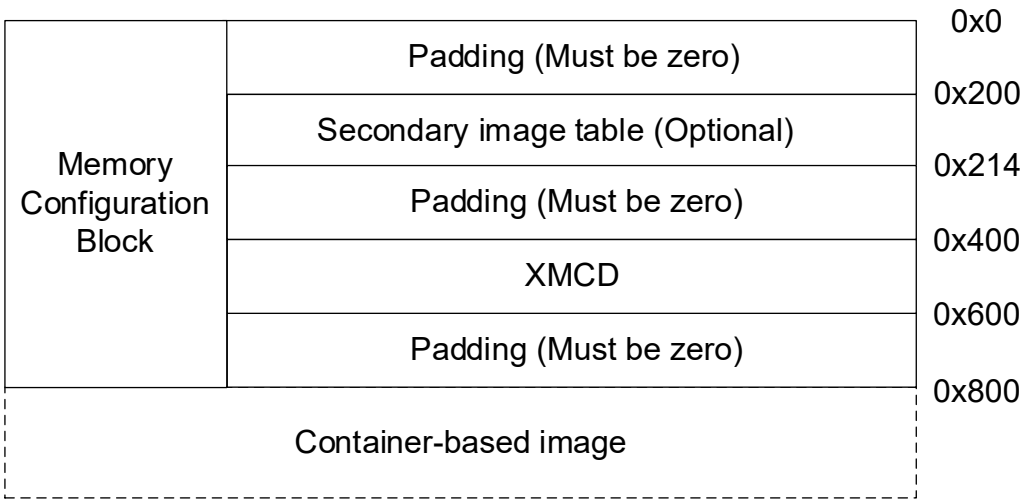


Figure 52. Layout for SD/eMMC boot

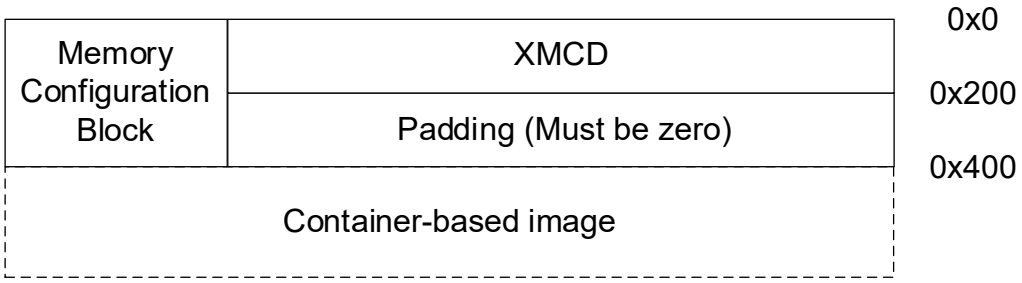


Figure 53. Layout for Serial NOR Flash/E2PROM (Recovery) boot and serial downloader boot

12.6.2 Container Format

The image container format consists of the following parts.

- Container header
- Image array entry
- Signature block
- User program images and data

The high-level view of the container format is shown in the following figure.

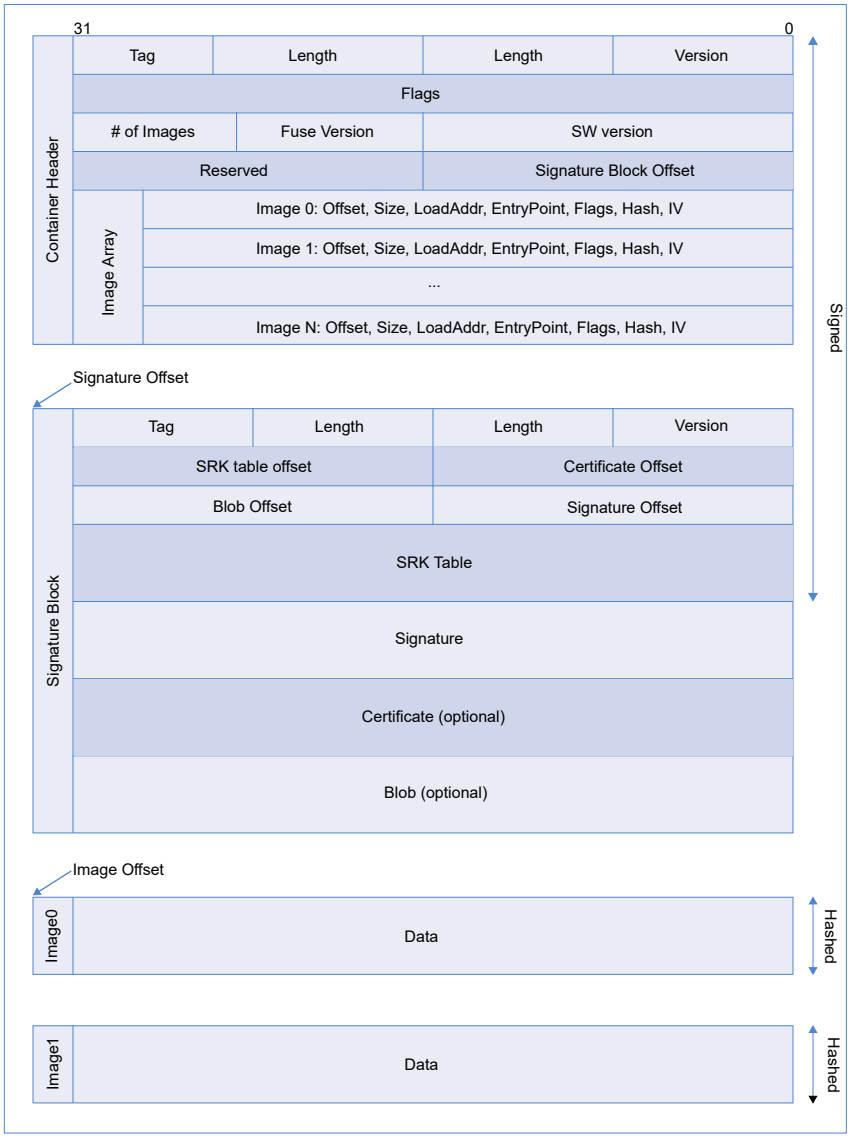


Figure 54. High-level view of the container format

NOTE

Certificate and Blob are not currently supported and will be supported by a future version of the Edgelock FW.

NOTE

If there are multiple images in the container, ELE verifies the hash for each image. If there is a failure on hash check, the device falls into infinite reset loop. Limitation: If multiple images are used, the hash for each image must be correct.

The container is located at a fixed address which is determined by the selected boot device.

Table 81. Image container start offset

Boot device	Container start offset
Serial NOR Flash via FlexSPI	4 KB = 0x1000 byte
Serial NAND Flash via FlexSPI	1 KB = 0x400 byte after FCB/DBBT blocks
Parallel NAND Flash via SEMC	1 KB = 0x400 byte after FCB/DBBT blocks
eMMC/SD via uSDHC(including manufacturing boot)	2 KB = 0x800 byte
SPI NOR Flash/EEPROM via LPSPI	1 KB = 0x400 byte
Serial downloader	1 KB = 0x400 byte

The maximum number of the containers is restricted to 2, one NXP container for NXP firmware, one OEM container for user/OEM images. If NXP firmware is required, the NXP container must be located at the first container position, while the OEM container must be located after the NXP container. If NXP firmware is not used, first container can be the OEM container, and the second container shall be padded with zeros.

The maximum number of the images in the NXP container is restricted to 1. The OEM container can have 8 images at the maximum. The Cortex-M33 core (booting core) images must be located after the other ones.

Following are the general notes for the container format.

- All multi-byte fields are stored in little-endian format
- The first word of each structure matches the format as used in HAB and other legacy (i.MX 6/7) used data structures
- All padding bytes are filled with zeros
- Only container header, image array entry and most of signature block are signed. Image data is not used to calculate the signature, but is only used indirectly via hash values
- Header can be signed by SRK or by image key that was signed by SRK. If an image key is used, it must be the same algorithm and key size as the SRK. In both cases, the referenced SRK must not have been revoked.
- If SRK is used to sign header, there is only one signature verification per container. If an image key is used, then there are 2 verifications.
- If an SRK revoke bit is set, Edgelock will set the fuse for that, only after successful authentication of the header that contains the SRK revocation command. A revoked SRK cannot be used to authenticate an image.
- Images may be encrypted individually, but all use the same decryption key, which is stored in the decryption blob.
- Encrypted images are encrypted before assembling the container. In other words, the hash is the hash of the encrypted image.
- The IV field is always present in the image array descriptor, whether the image is encrypted or not, and it is the SHA256 hash over the plaintext. While only the lower 128 bits will be used as IV in the encryption process, the IV field size is always 256 bits since it will serve for integrity checking the decrypted image as well.
- The image array immediately follows the signature block. Each image must start on a 64-bit boundary.
- The signature is done only over the part of the header containing data blocks information (including the corresponding hashes), not over the entire file.
- All data in the container header, including most of the signature block (everything except the signature itself, optional certificate and optional blob) is signed.

12.6.2.1 Container Header

The container header contains essential information about the entire container image which is used by the ROM. Table 82 shows the container header format.

Table 82. Container header format

Offset	Bits 31-24	Bits 23-16	Bits 15-8	Bits 7-0
0x00	Tag	Length (MSB)	Length (LSB)	Version
0x04	Flags			
0x08	# of Images	Fuse Version	SW Version	
0x0c	Reserved		Signature Block Offset	

The container header format fields are defined below.

- **TAG**: Must be 0x87
- **Length**: Size of the container in bytes, from header, up through and including signature block.
- **Version**: Must be 0x00
- **Flags**: Table 83 provides the description of each bit field in the Flags.

Table 83. Flags bitfield description

Bit Field	Description
0-1	SRK Set 0x0 – Container not authenticated 0x1 – NXP SRK 0x2 – OEM SRK 0x3 – Reserved
2-3	Reserved (must be 0)
4-5	SRK Selection 0x0 – SRK1 is used 0x1 – SRK2 is used 0x2 – SRK3 is used 0x3 – SDR4 is used
6-7	Reserved (must be 0)
8-11	SRK Revoke Mask bitmask used to indicate which SRKs to revoke – zero bits set means no SRKs are to be revoked, bit0 – SRK1 is to be revoked, bit1 – SRK2 is to be revoked,

Table continues on the next page...

Table 83. Flags bitfield description (continued)

Bit Field	Description
	bit2 – SRK3 is to be revoked, bit3 – SRK4 is to be revoked,
12-15	Reserved (must be 0)
16-31	Reserved (must be 0)

- **# of images**: Number of images in the image array
- **Fuse Version**: This value must be equal to or greater than the version stored in the fuses to allow loading this container
- **SW Version**: Used to select between multiple images with same Fuse version field.
- **Signature Block Offset**: Offset in bytes from beginning of header to signature block (keys, signature, blob)

12.6.2.2 Image Array Entry

The boot ROM loads images in the order that the images appear in the array.

Table 84 shows the Image Array Entry format

Table 84. Image Array Entry format

Offset	Bits 31-24	Bits 23-16	Bits 15-8	Bits 7-0
0x00	Image Offset			
0x04	Image Size			
0x08	Load Address (64 bits)			
0x0c				
0x10	Entry point (64 bits)			
0x14				
0x18	Flags			
0x1c	Image meta data			
0x20	Hash (512 bits)			
…				
0x60	IV (256 bits)			

The image array entry format fields are defined below.

- **Image Offset**: Offset in bytes from start of current container header to beginning of the image

NOTE

The image offset should be page-size aligned if the NAND boot was used. The image offset should be within 16bit and 256 byte aligned if the recovery boot was used.

- **Image Size**: Size of the image in bytes
- **Load Address**: Address where the image is copied to in memory (absolute address in system memory map) by the ROM
- **Entry Point**: Entry point of the image (absolute address). Only valid for executable image types.
- **Flags**: Table 85 provides the description of each bit field in the Flags.

Table 85. Flags bitfield description

Bit Field	Description
0-3	Type of image 0x3 Executable 0x4 Data 0x7 Provisioning image 0x9 Provisioning data All other values reserved (These values extend HAB IVT V1)
4-7	Core ID 0x1 Cortex-M33 0x2 Cortext-M7 All other values reserved
8-10	Hash type (all images in the container have the same HASH size) 0x0 – SHA256 0x1 – SHA384 0x2 – SHA512
11	Encrypted Indicate if this image is encrypted or not
12-31	Reserved

- **Image meta data**: For NXP FW private usage.
- **Hash**: SHA hash of image. Fixed size at 512 bits. Left aligned and padded with zero for hash sizes below 512 bits.
- **IV**: Used only for encrypted images (zero otherwise): SHA256 of the plain text image. Fixed size at 256 bits. The lower 128-bit part of the SHA256 value will be retained as IV in the encryption/decryption process.

12.6.2.3 Signature Block

Table 86. Signature block format

Offset	Bits 31-24	Bits 23-16	Bits 15-8	Bits 7-0
0x00	Tag	Length (MSB)	Length (LSB)	Version
0x04	SRK Table Offset		Certificate Offset	
0x08	Blob Offset		Signature Offset	
0x0A	Reserved (for 64bit alignment purposes) 32 bit			
0x10	SRK Table			
…	Signature			
…	Certificate			
…	Blob			

The signature block format fields are defined below.

- **TAG**: 0x90
- **Version**: 0x00
- **SRK Table Offset**: Offset in bytes from beginning of signature block to SRK table. If 0x0000, then SRK table is not present. This should be the case only if the container is not authenticated, and the SRK key set field in the Container header should be 0.
- **Certificate Offset**: Offset in bytes from beginning of signature block to signing key certificate. If 0x0000, then only a SRK is used in signature verification
- **Signature Offset**: Offset in bytes from beginning of signature block to signature
- **Blob Offset**: Offset in bytes from beginning of signature block to DEK Blob. If 0x0000, then no images are encrypted

NOTE

1. SRK Table, Signature, Certificate and Blob must be in the order shown
2. Certificate and Blob are optional
3. SRK Table, Certificate, Blob, and Signature must start on 64 bit boundaries. Any padding needed to make this happen must be zeros.
4. The container signature is calculated of all data from beginning of container header to the last byte before the signature starts. In other words, most of the signature block itself is included in the signature calculations.

12.6.2.3.1 SRK Table

The SRK table determines the signature type and length, and consists of 1 to 4 SRK records which contains the public keys. The format of the SRK table is inherited from HAB v4.x

Table 87. SRK table format

Offset	Bits 31-24	Bits 23-16	Bits 15-8	Bits 7-0
0x00	Version	Length of SRK Table		Tag
0x04	SRK Record 1			
-	SRK Record 2			
-	SRK Record 3			
-	SRK Record 4			

The SRK table fields are defined below.

- **TAG**: 0xD7
- **Length**: The size of the SRK table in bytes
- **Version**: 0x42

12.6.2.3.1.1 SRK Record

Table 88. SRK record format

Offset	Bits 31-24	Bits 23-16		Bits 15-8	Bits 7-0
0x00	Signing Algorithm	Length of SRK			Tag
0x04	SRK Flags	Reserved	Key Size/Curve		Hash Algorithm
0x08	RSA: Exponent Length (bytes)	RSA: Modulus Length (bytes)			
	ECDSA: Y Length (bytes)	ECDSA: X Length (bytes)			
0x0C	RSA: Modulus (Big endian)				
	ECDSA: X (Big endian)				
-	RSA: Exponent (Big endian)				
	ECDSA: Y (Big endian)				

The SRK record fields are defined below.

- **TAG**: 0xE1
- **Length**: The size of the SRK record
- **Signing Algorithm**:
  - RSA-PSS: 0x22
  - ECDSA: 0x27
- **Hash Algorithm**:
  - SHA-256: 0x00
  - SHA-384: 0x01
  - SHA-512: 0x02

- **Key Size/Curve**:
  - PRIME256V1: 0x1
  - SEC384R1: 0x2
  - SEC521R1: 0x3
  - RSA2048: 0x5
  - RSA3072: 0x6
  - RSA4096: 0x7
- **SRK Flags**:
  - None: 0x00
  - CA Flag: 0x80

NOTE

The keys in the SRK table must match the signing key in algorithm and key length

12.6.2.3.2 Signature

Table 89. Signature format

Offset	>Bits 31-24	Bits 23-16	Bits 15-8	Bits 7-0
0x00	Tag	Length (MSB)	Length (LSB)	Version
0x04	Reserved			
0x08	Signature Data			

The signature format fields are defined below.

- **TAG**: 0xD8
- **Version**: 0x00
- **Signature Data**:Signature.
  - RSA-PSS encoding
  - ECDSA: The r and s components are padded with leading zeros to make them each 8 bit aligned, then concatenated – r|s

12.7 External Memory Configuration Data (XMCD)

For comprehensive or feature-rich applications, the on-chip RAM may not be enough for the application use. Usually, such an application requires a large capacity of RAM and needs to run on the external RAM for high execution performance. However, enabling the external RAM on the SoC is relatively complicated, and the settings are SoC specific. The Boot ROM on the SoC simplifies the external RAM configuration by introducing the XMCD.

The XMCD data structure resides out of the program image, and the offset varies depending on the boot device. The XMCD offset for each device type is provided in the following table.

Table 90. XMCD Offset

Boot device	XMCD Offset
Serial NOR Flash via FlexSPI	0x800

Table continues on the next page...

Table 90. XMCD Offset (continued)

Boot device	XMCD Offset
Serial NAND Flash via FlexSPI	0x0 after FCB/DBBT blocks
Parallel NAND Flash via SEMC	0x0 after FCB/DBBT blocks
eMMC/SD via uSDHC(including manufacturing boot)	0x400
Serial NOR Flash/EEPROM via LPSPI	0x0
Serial downloader	0x0

XMCD consists of a XMCD header and a memory specific configuration block. On this SoC, the following types of external RAM devices are supported. Each type of RAM device has a specific configuration block.

- HyperRAM/APMemory via FlexSPI – FlexSPI RAM configuration block
- SDRAM via SEMC – SEMC SDRAM Configuration block

12.7.1 XMCD Header

The header definition is as follows.

Table 91. XMCD Header Definition

[31:28] Tag	[27:24] Version	[23:20] Memory Interface	[19:16] Instance	[15:12] Configuration Block Type	[11:0] Configuration block size
Valid value = 0xC	Fixed value of '0'	0 - FlexSPI 1 - SEMC	SoC defined instances	0 - Simplified Configuration Option Block 1 - Full Configuration Block	Configuration block size (including the XMCD Header itself).

12.7.2 XMCD CRC

There is an optional integrity check of the XMCD by programming the CRC checksum to the fuse. The ROM can perform the integrity check before adopting the XMCD block settings. It calculates the CRC check based on the value of the "Configuration block size" field. ROM determines whether to enable the XMCD integrity check by checking the XMC_CRC32_CHECK_EN fuse field and the XMC_CRC32_CHECK_SUM fuse. Details of the CRC algorithm used for the integrity check are provided in the following table.

Table 92. CRC Algorithm Details

Description	Value
Width	32
Polynomial	0x04C11BD7
Init Value	0xFFFFFFFF
Reflect In	FALSE
Reflect Out	FALSE
XOR Out	0x00000000

The following procedure shows the steps in the CRC calculation:

1. CRC initialization
   - Set the initial CRC as 0xFFFFFFFF, which clears the CRC byte count to 0.
2. CRC calculation
   - Start the calculation from the first byte of the XMCD header. Total calculation bytes are "Configuration block size" bytes.
3. CRC finalization
   - Check if the CRC byte count is 4-bytes aligned. If it is not 4-bytes aligned, then pad it with the necessary zeroes to finalize the CRC. Otherwise, return to the current computed CRC.

12.7.3 FlexSPI RAM Configuration Block

The XMCD offers a simplified FlexSPI RAM configuration option block which can meet the typical usage of the HyperRAM or APMemory devices.

However, the user application may need to enable the advanced features of the external RAM which cannot be configured by the simplified configuration option. In this case, the XMCD also offers the complete 512-bytes FlexSPI RAM configuration block which supports flexible configuration.

12.7.3.1 Simplified FlexSPI RAM Configuration Option Structure

The simplified FlexSPI RAM configuration option structure is shown in the following tables.

Table 93. Simplified FlexSPI RAM Configuration Option 0

[31:28] Tag	[27:24] Option Size	[23:20] device type	[19:16] Reserved	[15:12] Misc.	[11:8] Maximum Frequency	[7:0] Size in MB
Fixed to 0x0C	0 - Option words = 1 1 - Option words = 2 Others - Reserved	0 - HyperRAM 1 - APMemory	Reserved	For HyperRAM 0 - Differential clock driven 1 - Single-ended clock driven	0 - Invalid 1 - 30MHz 2 - 50MHz 3 - 60MHz 4 - 80MHz 5 - 100MHz 6 - 120MHz 7 - 133MHz 8 - 166MHz 9 - 200MHz Others - Reserved	0 - Auto detection Others - Size in MB

Table 94. Simplified FlexSPI RAM Configuration Option 1

[31:28] RAM_CONNECTION	[27:24] Reserved	[23:20] DQS_PINMUX_GROUP	[19:16] PINMUX ROUP	[15:8] Reserved	[7:4] write dummy cycles	[3:0] read dummy cycles
0 - PORTA 1 - PORTB	Reserved	0 - Default Group 1 - Secondary group	0 - Primary group 1 - Secondary group	Reserved	0 - Auto detection Others - Specified dummy cycles	0 - Auto detection Others - Specified dummy cycles

12.7.3.2 Full FlexSPI RAM Configuration Block Structure

The following table lists the data structure of the full 512-bytes FlexSPI RAM Configuration Block.

Table 95. Full HyperRAM/APMemory Configuration Block Structure

Name	Offset	Size(Bytes)	Description
Tag	0x000	4	0x42464346, ascii: "FCFB"
Version	0x004	4	[07:00] bugfix [15:08] minor [23:16] major = 1 [31:24] ascii 'V'
Reserved	0x008	4	Reserved
readSampleClkSrc	0x00c	1	0 – Internal loopback 1 – loopback from DQS pad 2 - Reserved 3 – Flash provided DQS
csHoldTime	0x00d	1	Serial Flash CS Hold Time Recommend default value is 0x03
csSetupTime	0x00e	1	Serial Flash CS setup time Recommended default value is 0x03
columnAdressWidth	0x00f	1	3 – For HyperFlash/HyperRAM 12/13 – For Serial NAND. See datasheet to find correct value. 0 – Other devices
deviceModeCfgEnable	0x010	1	Device Mode Configuration Enable feature 0 – Disabled 1 – Enabled
deviceModeType	0x11	1	Device Mode Type

Table continues on the next page...

Table 95. Full HyperRAM/APMemory Configuration Block Structure (continued)

Name	Offset	Size(Bytes)	Description
			0 – Generic 1 – Quad Enable 2 – SPI-to-xSPI Mode 3 – xSPI-to-SPI mode
waitTimeCfgCommands	0x12	2	Time in terms of 100us This field is defined for the mode switch from SPI to xSPI mode or vice versa.
deviceModeSeq	0x014	4	Sequence parameter for device mode configuration [7:0] - Number of sequences [15:8] - Sequence Index [31:16] - Reserved, fixed to 0
deviceModeArg	0x018	4	Device Mode argument, effective only when deviceModeCfgEnable = 1
configCmdEnable	0x01c	1	Config Command Enable feature 0 – Disabled 1 – Enabled
configModeType	0x01d	3	This field has the same definitions as "deviceModeType" byte 0 - configModeType for configCmdSeq[0] byte 1 - configModeType for configCmdSeq[1] byte 2 - configModeType for configCmdSeq[2]
configCmdSeqs	0x020	12	Sequences for Config Command allows 3 separate configuration command sequences. For each configCmdSeq, the definition of the word is: [7:0] - Number of sequences [15:8] - Sequence Index [31:16] - Reserved, fixed to 0
Reserved	0x02c	4	Reserved
cfgCmdArgs	0x030	12	Arguments for each separate configuration command sequence.
Reserved	0x03c	4	Reserved
controllerMiscOption	0x040	4	Bit0 – differential clock enable Bit1 – CK2 enable Bit2 – ParallelModeEnable

Table continues on the next page...

Table 95. Full HyperRAM/APMemory Configuration Block Structure (continued)

Name	Offset	Size(Bytes)	Description
			Bit3 – wordAddressableEnable Bit4 – Half-Speed access enable Bit5 – Pad Setting Override Enable Bit6 – DDR Mode Enable Bit7 - Pad Setting Override Enable Bit 8 - Second Pinmux group Bit 9 - Second DQS pin mux group Bit 10 - Write Mask Enable Bit 11 - Write Opt1 Clear
deviceType	0x044	1	1 – Serial NOR 2 – Serial NAND 3 - Serial RAM
sflashPadType	0x045	1	1 – Single pad 2 – Dual pads 4 – Quad pads 8 – Octal pads
serialClkFreq	0x046	1	Chip specific value
lutCustomSeqEnable	0x047	1	0 – Use pre-defined LUT sequence index and number 1 - Use LUT sequence parameters provided in this block
Reserved	0x048	8	Reserved
sflashA1Size	0x050	4	For SPI NOR, need to fill with actual size For SPI NAND, need to fill with actual size * 2
sflashA2Size	0x054	4	For SPI NOR, need to fill with actual size For SPI NAND, need to fill with actual size * 2
sflashB1Size	0x058	4	For SPI NOR, need to fill with actual size For SPI NAND, need to fill with actual size * 2
sflashB2Size	0x05c	4	For SPI NOR, need to fill with actual size For SPI NAND, need to fill with actual size * 2
csPadSettingOverride	0x060	4	Set to 0 if it is not supported
sclkPadSettingOverride	0x064	4	Set to 0 if it is not supported

Table continues on the next page...

Table 95. Full HyperRAM/APMemory Configuration Block Structure (continued)

Name	Offset	Size(Bytes)	Description
dataPadSettingOverrid e	0x068	4	Set to 0 if it is not supported
dqsPadSettingOverride	0x06c	4	Set to 0 if it is not supported
timeoutInMs	0x070	4	Maximum wait time during read/write
commandInterval	0x074	4	Unit: ns. Currently, it is for SPI NAND at the high working frequency. For the serial NAND and serial RAM device, this field is 0.
dataValidTime	0x078	4	Time from clock edge to data valid edge. Unit: ns. This field takes effect when the FlexSPI Root clock is less than 100MHz, and the read sample clock source is a device provided DQS signal without CK2 support. [31:16] - data valid time for DLLB in terms of 0.1ns [15:0] - data valid time for DLLA in terms of 0.1ns
busyOffset	0x07c	2	Busy bit offset. Valid range :0-31.
busyBitPolarity	0x07e	2	0 – 1 represents busy 1 – 0 represents busy
lookupTable	0x080	16*4*4	Lookup table
lutCustomSeq	0x180	4*12	Customized LUT sequence. See the note at the end of the table for details.
Reserved	0x1b0	5*16	Reserved for future use

NOTE

- LUT for read must be placed at LUT entry 0.
- LUT for write must be placed at LUT entry 9.

12.7.3.3 Example of XMCD for HyperRAM Support

The following table shows an example of XMCD for HyperRAM support.

Table 96. Example XMCD block for HyperRAM

Offset	Field	Description
0	0xC000_0008	Tag = 0x0C Version = 0 Memory Interface: FLEXSPI Instance: 1 - First Instance Configuration block type: Simplified

Table continues on the next page...

Table 96. Example XMCD block for HyperRAM (continued)

Offset	Field	Description
		Configuration block size: 8 (4-byte header + 4-byte option block)
1	0xC000_0700	Tag = 0x0C Option_Size = 0 DeviceType: HyperRAM Reserved: 0 Misc: HyperRAM 1V8 Max Freq: 7 Memory Size: Auto-detection

NOTE

- The XMCD feature can be disabled by Fuse XMC_DISABLE.
- Dual-Die-Package (DDP) HyperRAM is not supported.

12.7.4 SEMC SDRAM Configuration Block

The XMCD also supports the SDRAM configuration via the SEMC SDRAM Configuration Block, which offers both the simplified configuration option and the fully customizable configuration block.

12.7.4.1 SEMC SDRAM Configuration Block Structure

The SEMC SDRAM configuration block structure is shown in the following table.

Table 97. SDRAM Configuration Block structure

Offset	Width (bytes)	Field	Description
0	1	magic_number	Must be 0xA1
1	1	version	Set to 1 for this implementation
2	1	config_option	0x00 - Simplified configuration • Select SDRAM CS0 as default and can only config clk_MHz, sdram0_size_kB and port_size • All unconfigured fields must be 0-filled 0xFF - Full configuration • Must config all fields.
3	1	clk_MHz	Set the working frequency in the unit of MHz
4	4	sdram0_size_kB	Set the memory size of SDRAM CS0 in the unit of kilobytes Range: 4~4*1024*1024

Table continues on the next page...

Table 97. SDRAM Configuration Block structure (continued)

Offset	Width (bytes)	Field	Description
8	1	port_size	Port size of SDRAM 0 - 8bit 1 - 16bit 2 - 32bit Others - Invalid value
9	1	pin_config_pull	Pull config of the SDRAM GPIO pin 0 – forbidden 1 – pull up 2 – pulldown 3 – no pull Others - Invalid value
10	1	pin_config_drive_strength	Driver config of SDRAM GPIO pin 0 – high driver 1 – normal driver Others - Invalid value
11	1	mux_rdy	SDRAM CSn device selection 1 - SDRAM CS1 2 - SDRAM CS2 3 - SDRAM CS3 Others – Invalid for SDRAM, select other external devices **NOTE** To select CS1/CS2/CS3, you can only use one of the five items: mux_rdy ,mux_csx0, mux_csx1,mux_csx2 ,mux_csx3). If using one item, other items cannot be used, set them to 0.
12[3:0]	1	mux_csx0	SDRAM CSn device selection 1 - SDRAM CS1 2 - SDRAM CS2 3 - SDRAM CS3 Others – Invalid for SDRAM, select other external devices

Table continues on the next page...

Table 97. SDRAM Configuration Block structure (continued)

Offset	Width (bytes)	Field	Description
12[7:4]	1	mux_csx1	SDRAM CSn device selection 1 - SDRAM CS1 2 - SDRAM CS2 3 - SDRAM CS3 Others – Invalid for SDRAM, select other external devices
13[3:0]	1	mux_csx2	SDRAM CSn device selection 1 - SDRAM CS1 2 - SDRAM CS2 3 - SDRAM CS3 Others – Invalid for SDRAM, select other external devices
13[7:4]	1	mux_csx3	SDRAM CSn device selection 1 - SDRAM CS1 2 - SDRAM CS2 3 - SDRAM CS3 Others – Invalid for SDRAM, select other external devices
14[0]		dccr_sdramen	DCCR SDRAM enable selection 0 – default 1 - DCCR enable
14[5:1]		dccr_sdramval	DCCR SDRAM value Valid when the dccr_sdramen set 1
14[6]		reserved	-
14[7]		dccr_en	DCCR SDRAM reconfigurable selection 0 – dccr value configure by default 1 – dccr configure by user
15[0]		sdramcr0_en	SDRAMCR0 configuration selection 0 – set by default 1 – Set by user
15[1]		Sdramcr1_en	SDRAMCR1 configuration selection 0 – set by default 1 – Set by user

Table continues on the next page...

Table 97. SDRAM Configuration Block structure (continued)

Offset	Width (bytes)	Field	Description
15[2]		Sdramcr2_en	SDRAMCR2 configuration selection 0 – set by default 1 - Set by user
15[3]		Sdramcr3_en	SDRAMCR3 configuration selection 0 – set by default 1 - Set by user
15[7:4]		reserved	-
16	1	bank	Bank numbers of SDRAM device 0 – 4 banks 1 – 2 banks Others - Invalid value
17	1	burst_len	Burst length 0 – 1 1 – 2 2 – 4 3 – 8 Others - Invalid value
18	1	column_addr_bit_num	Column address bit number 0 – 12bit 1 – 11bit 2 – 10bit 3 – 9bit 4 – 8bit Others - Invalid value
19	1	cas_latency	CAS Latency 1 – 1 2 – 2 3 – 3 Others - Invalid value
20	1	write_recovery_ns	Write recovery time in unit of nanosecond This could help to meet tWR timing requirement by SDRAM device.

Table continues on the next page...

Table 97. SDRAM Configuration Block structure (continued)

Offset	Width (bytes)	Field	Description
21	1	refresh_recovery_ns	Refresh recovery time in unit of nanosecond This could help to meet tRFC timing requirement by SDRAM device.
22	1	act2readwrite_ns	Act to read/write wait time in unit of nanosecond This could help to meet tRCD timing requirement by SDRAM device.
23	1	precharge2act_ns	Precharge to active wait time in unit of nanosecond This could help to meet tRP timing requirement by SDRAM device.
24	1	act2act_banks_ns	Active to active wait time between two different banks in unit of nanosecond This could help to meet tRRD timing requirement by SDRAM device.
25	1	refresh2refresh_ns	Auto refresh to auto refresh wait time in unit of nanosecond This could help to meet tRFC timing requirement by SDRAM device.
26	1	selfref_recovery_ns	Self refresh recovery time in unit of nanosecond This could help to meet tXSR timing requirement by SDRAM device.
27	1	act2prechage_min_ns	ACT to Precharge minimum time in unit of nanosecond This could help to meet tRAS(min) timing requirement by SDRAM device.
28	4	act2prechage_max_ns	ACT to Precharge maximum time in unit of nanosecond This could help to meet tRAS(max) timing requirement by SDRAM device.
32	4	refreshperiod_perrow_ns	Refresh timer period in unit of nanosecond Set to (tREF(ms) * 1000000/rows) value.
36	4	mode_register	Define the specific mode of operation of SDRAM Set to the value required by SDRAM device. **NOTE** The low bits of the input value must corresponds to the low bits of the mode register.

Table continues on the next page...

Table 97. SDRAM Configuration Block structure (continued)

Offset	Width (bytes)	Field	Description
40	4	sdram0_base	Base address of SDRAM CS0 Range: 0x80000000~0xDFFFFFFF **NOTE** SDRAM CS0~CS3 addresses cannot overlap and when CSn is not used, set the address to 0.
44	4	sdram1_base	Base address of SDRAM CS1 Range: 0x80000000~0xDFFFFFFF
48	4	sdram2_base	Base address of SDRAM CS2 Range: 0x80000000~0xDFFFFFFF
52	4	sdram3_base	Base address of SDRAM CS3 Range: 0x80000000~0xDFFFFFFF
56	4	sdram1_size_kB	Set the memory size of SDRAM CS1 in unit of kbytes Range: 4~4*1024*1024
60	4	sdram2_size_kB	Set the memory size of SDRAM CS2 in unit of kbytes Range: 4~4*1024*1024
64	4	sdram3_size_kB	Set the memory size of SDRAM CS3 in unit of kbytes Range: 4~4*1024*1024

12.7.4.2 Example for SDRAM Support (32-bit)

The following table shows an example for the SDRAM support (32-bit) on this device.

Table 98. Example SDRAM Configuration Block

Offset	Field	Description
0	0xC010_000D	Tag = 0xC Version = 0 Memory Interface: SEMC Instance: 0 - ignored Configuration block type: 0 - Ignored (Handled inside the SDRAM configuration structure) Configuration block size: 13 (4-byte header + 9-byte option block)
1	0xA600_01A1	Magic_number = 0xA1

Table continues on the next page...

Table 98. Example SDRAM Configuration Block (continued)

Offset	Field	Description
		Version = 1 Config_option: Simplified SDRAM clock: 166MHz
2	0x0001_0000	SDRAM CS0 size: 64MBytes
3	0x02	Port_size: 32-bit

12.8 Serial Boot (Serial Downloader)

The Serial boot provides a means to download a boot image to the chip and execute the image over the following serial peripherals:

- LPUART
- USB-HID
- LPSPI

The default enabled peripherals for active peripheral detection is determined by the combination of fuses and Boot Mode pins.

The boot ROM can detect the active peripherals and shut down all inactive peripherals before any communications between the host and the device.

The high-level serial boot flow is provided below. Details of communication protocols, sequences, and flows can be found in the following sections.

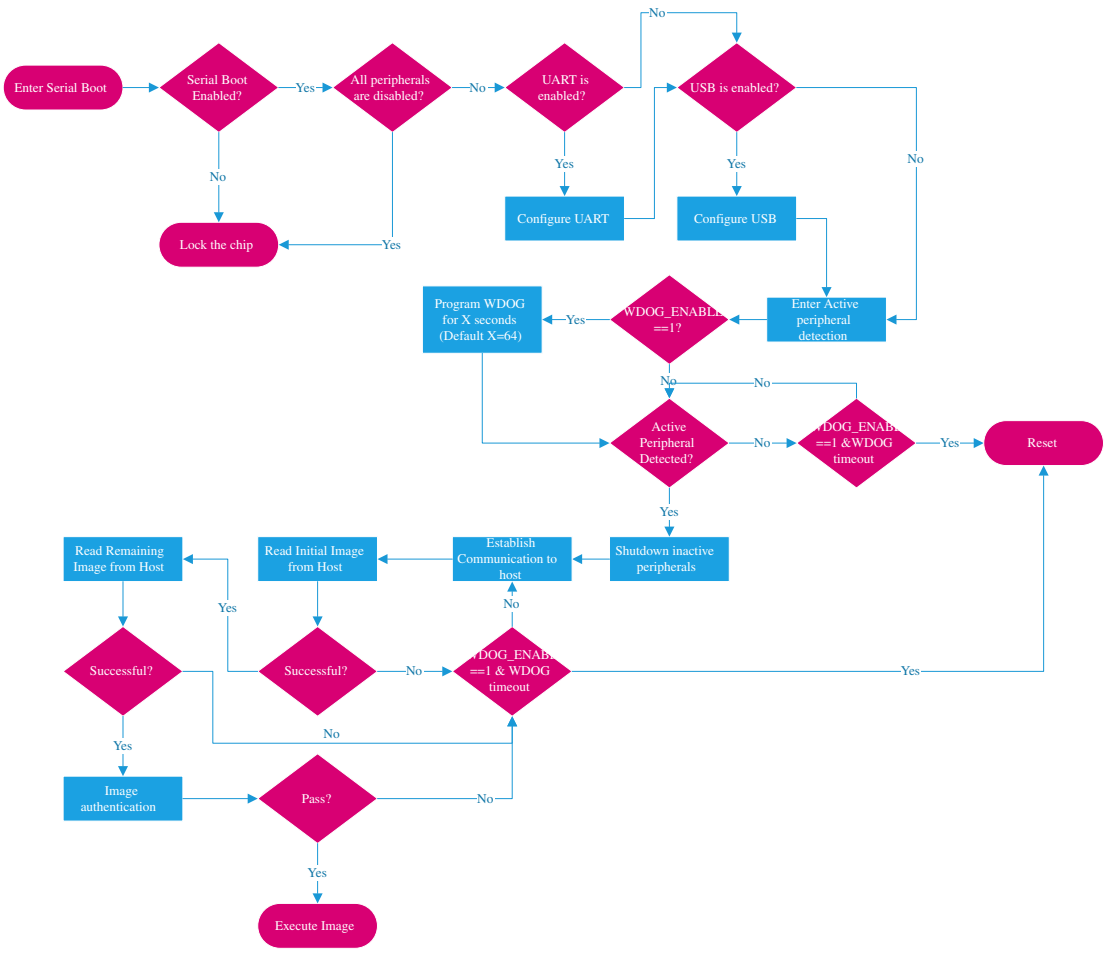


Figure 55. Serial Boot Flow

The communication protocol for serial boot mode is compatible with the widely-used MCUBOOT protocol in the NXP MCU products. Using simplified protocol, only two types of phase are supported:

1. **Command-only phase**. “get-property” command is supported, but the supported properties are limited to:
   - Bootloader version
   - Target version
   - Maximum payload size in the packet
   - Last Status

   For all other properties queried by the host, the boot ROM responds with unknown property error code.
2. **Data-only phase**. Boot ROM treats the data from the host as data streaming; no command phase is needed in this phase.

12.8.1 MCU-BOOT Protocol

The section demonstrates the general protocol for the packet transfers between the host and the boot ROM. The description includes the transfer of packets for different transactions, such as command with no data phase and data only phase. The next section describes various packet types used in a transaction.

- Each command sent from the host is replied to with a response command.

- Each data sent from the host is replied with an ACK at the packet level.
- In all protocols (described in the following subsections), the Ack sent in response to a Command or Data packet can arrive at any time before, during, or after the Command/Data packet has processed.

Command-only phase

The protocol for command-only phase content:

- Command packet (from the host)
- Response command packet (to host)

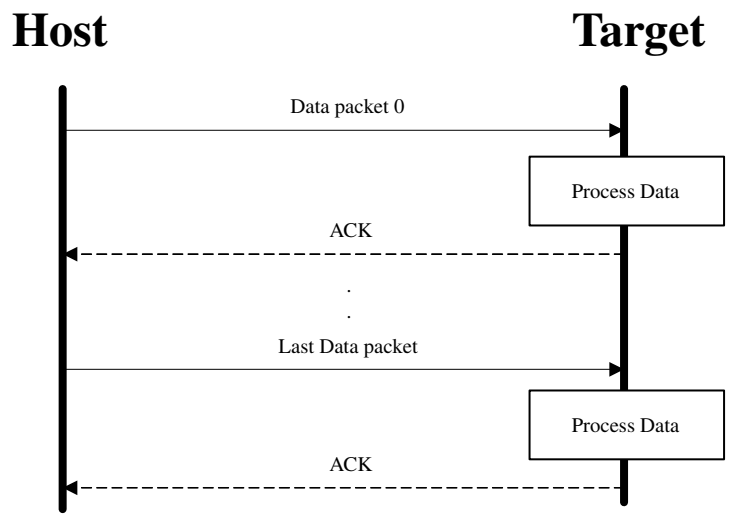


Figure 56. Protocol for a command-only phase

Data-only phase

The protocol for data-only phase content:

- A data packet (from the host)

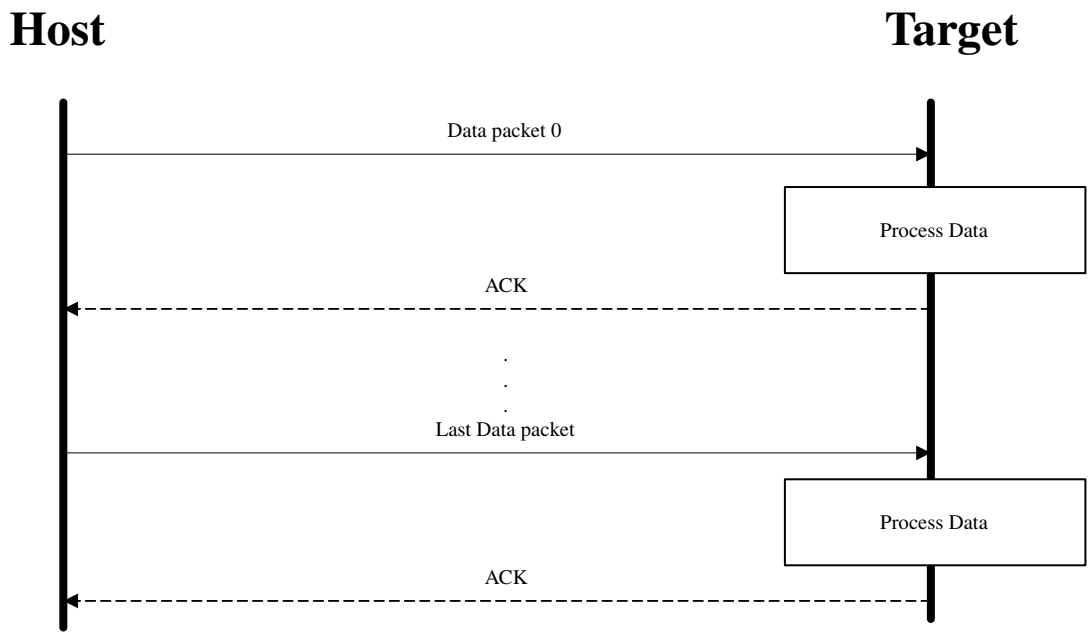


Figure 57. Protocol for a data-only phase

There are 2 types of packet protocol defined for the peripherals with or without built-in flow control.

- **Serial packet protocol**. It is defined for UART. A packet level flow control mechanism and error-detection mechanism is designed in this protocol.
- **HID packet protocol**. It is defined for USB-HID, the USB built-in flow control and error-detection is used behind this protocol.

In each protocol, several packet types are defined to let the boot ROM:

- Establish the communication to a host
- Recognize different phases (command-only/ data-only)

12.8.1.1 Packet Types for Serial Packet Protocol

There are 5 types of packets supported by the Serial Packet Protocol:

- Ping packet
- Ping Response packet
- Framing packet
- Command packet
- Data packet

12.8.1.1.1 Ping Packet

The ping packet is the first packet sent from a host to target, in order to establish a connection on a UART peripheral for auto-baudrate detection. For other peripherals, it is optional.

Table 99. Ping packet format

Byte #	Value	Name
0	0x5A	Start Byte
1	0xA6	Packet Type: Ping packet

12.8.1.1.2 Ping Response Packet

In response to a Ping packet, the target sends a Ping Response packet.

Table 100. Ping response packet format

Byte #	Value	Parameter
0	0x5A	Start Byte
1	0xA7	Packet Type: Ping Response packet
2		Protocol version: bug fix
3		Protocol version: minor
4		Protocol version: major
5		Protocol name: 'P' (0x50)
6		Reserved: set to 0
7		Reserved: set to 0
8		crc16_low
9		crc16_high

NOTE

1. Please refer to CRC Algorithm for details regarding the CRC algorithm.
2. Byte 0 - 7 are including in the CRC calculation.

Example ping packet:

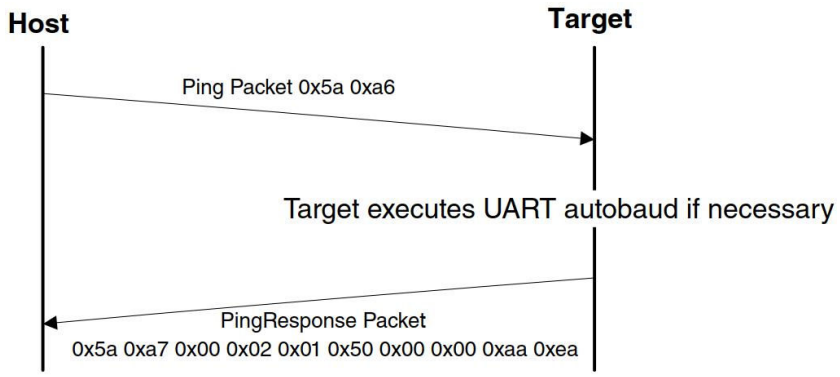


Figure 58. Ping Command Protocol

12.8.1.1.3 Framing Packet

The framing packet is for flow control and error detection, for the communications links that do not have such features built-in. The framing packet structure sits between the link layer and the command layer. It wraps command and data packets as well.

NOTE

Command packet is optional in ROM

Every framing packet containing data sent in one direction results in a synchronizing response framing packet in the opposite direction.

The framing packet described in the section is for serial peripherals, including UART

The framing packet format for serial packet is provided below:

Table 101. Framing packet format

Byte #	Data type	Value	Field	Notes
0	Framing header	0x5a	Start Byte	Must be 0x5A
1		0xA5/0xA4	Packet Type	0xA5 for Data packet 0xA4 for command packet
2		2	Payload Size	16-bit little-endian, max size is 512 bytes
4		2	CRC-16 checksum	CRC checksum covers the whole packet including start byte, packet type, payload size, and payload data, but does not contain the CRC checksum bytes
6…n	Payload	payload data	The payload of the data	Max size is 512

A particular framing packet that contains only a start byte and a packet type is employed for synchronization between the host and target. The format details are provided below:

Table 102. Special Framing Packet Format

Byte #	Value	Parameter
0	0x5A	Start Byte
1	0xAn	Packet Type: 0xA1 - Ack: The previous packet was received successfully; the sending of more data is allowed 0xA2 - NAK: The previous packet was corrupted, and needs to be re-sent. 0xA3 - Abort: Data transfer is aborted.

12.8.1.1.4 CRC Algorithm

This section provides details about the CRC-16 algorithm.

The CRC is computed over each byte in the framing packet header, excluding the crc16 field itself, plus all the payload bytes. The CRC algorithm is the XMODEM variant of CRC-16.

The Characteristics of the XMODEM variants are:

Table 103. Characteristics of the XMODEM variant

Name	Value
width	16
polynomial	0x1021
init value	0x0000
reflect in	false
reflect out	false

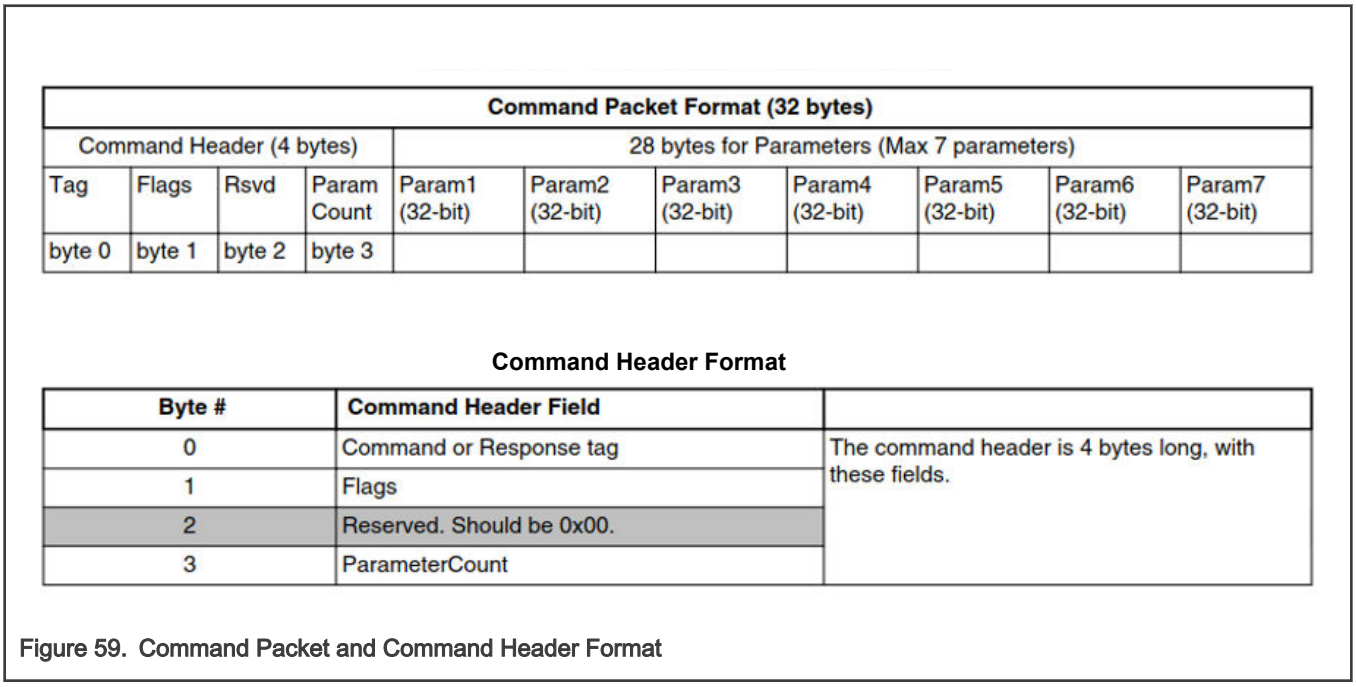
Table continues on the next page...

Table 103. Characteristics of the XMODEM variant (continued)

Name	Value
xor out	0x0000
check result	0x31C3

12.8.1.1.5 Command Packet

The command packet carries a 32-bit command header and a list of 32-bit parameters.



Command Packet Format (32 bytes)										
Command Header (4 bytes)				28 bytes for Parameters (Max 7 parameters)						
Tag	Flags	Rsvd	Param Count	Param1 (32-bit)	Param2 (32-bit)	Param3 (32-bit)	Param4 (32-bit)	Param5 (32-bit)	Param6 (32-bit)	Param7 (32-bit)
byte 0	byte 1	byte 2	byte 3							

Command Header Format

Byte #	Command Header Field	
0	Command or Response tag	The command header is 4 bytes long, with these fields.
1	Flags	
2	Reserved. Should be 0x00.	
3	ParameterCount	

Figure 59. Command Packet and Command Header Format

The header is followed by 32-bit parameters up to the value of the ParameterCount field specified in the header. Because a command packet is 32 bytes long, only 7 parameters can fit into the command packet.

Command packets are also used by the target to send responses back to the host. Command packets and data packets are embedded into framing packets for all the transfers.

The supported command and response list is provided in the following tables:

Table 104. Supported command list

Command	Name
0x07	GetProperty

Table 105. Supported response list

Response	Name
0xA0	Generic Response
0xA7	GetProperty Response - used for sending a response to GetProperty command only

12.8.1.1.6 Data Packet

The data packet carries just the data that the host sends to target. The data packet is also wrapped within a framing packet to ensure the correct packet data is received.

The contents of a data packet are simply the data itself. There are no other fields so that the most data per packet can be transferred. Framing packets are responsible for ensuring that the correct packet data is received.

The host must send the data packet with the maximum payload size until the last packet, the packet with smaller payload size is treated as the last packet during the transfer.

12.8.1.1.7 Response Packet

The responses are carried using the same command packet format wrapped with framing packet data. Types of responses supported in the boot ROM include:

- GenericResponse
- GetPropertyResponse

GenericResponse: After the boot ROM has processed a command, the boot ROM sends a generic response with status and command tag information to the host. The generic response is the last packet in the command protocol sequence. The generic response packet contains the framing packet data and the command packet data (with generic response tag = 0xA0) and a list of parameters. The parameter count field in the header is always set to 2, for status code and command tag parameters.

The GenericResponse parameters are provided below:

Table 106. GenericResponse parameters

Byte #	Parameter	Description
0 - 3	Status code	The status codes are errors encountered during the execution of a command by the target. If a command succeeds, then a success code is returned.
4 - 7	Command tag	The command tag parameter identifies the response to the command sent by the host.

NOTE

GenericResponse in ROM is only used as a response for all unsupported commands, in which:

- Status = 10000 (Unknown command)
- Command tag = command tag in the Command header.

GetPropertyResponse: The target sends the GetPropertyResponse packet in response to the host query that uses the GetProperty command. The GetPropertyResponse packet contains the framing packet data and the command packet data, with the command/response tag set to a GetPropertyResponse tag value (0xA7).

The parameter count field in the header is set to greater than 1, to always include the status code and one or many property values.

The GetPropertyResponse parameters are provided below:

Table 107. GetPropertyResponse parameters

Byte #	Value	Parameter
0 - 3		Status code
4 - 7		Property code
. . .		. . .
		Can be up to maximum 6 property values. Limited to the size of the 32-bit command packet and property type.

The possible status are:

- 0 - kStatus_Success
- 10300 - Unknown Property

12.8.1.2 Packet Types for USB-HID Protocol

There are 3 types of packets supported in USB-HID protocol:

- Framing packet
- Command packet
- Data packet

12.8.1.2.1 HID Reports

There are 4 HID reports defined and used by the boot ROM USB HID protocol. The report ID determines the direction and type of packet sent in the report. Otherwise, the contents of all reports are the same.

The HID reports for the USB-HID protocol are provided below:

Table 108. HID reports for the USB-HID protocol

Report ID	Packet Type	Direction
1	Command	OUT (Host -> Device)
2	Data	OUT (Host -> Device)
3	Command	IN (Device -> Host)

12.8.1.2.2 Framing Packet

The framing packet has a 4-byte framing header, in which the report ID and the packet length is used for the packet type definition and the real payload size.

Table 109. USB-HID Framing Packet

Byte #	Data Type	Value	Note
0	Framing Header	Report ID	
1		Padding byte	Must be 0
2		Packet Length LSB	Max size is limited to 1016 bytes
3		Packet Length MSB	
4	Payload	Packet [0]	
5		Packet [1]	
		…	
N+4-1		Packet [N-1]	

12.8.1.2.3 Command Packet

The Report ID needs to be '1' for Command, and it must be '3' for Response. The payload definition is the same with Serial Packet Protocol.

12.8.1.2.4 Data Packet

The Report ID must be '2' in data packet. The payload definition is the same with Serial Protocol.

12.8.2 Serial Boot via UART and LPSPI

12.8.2.1 UART Configuration Details

Provided below are the details of UART configuration in boot ROM:

- **8-N-1** (8-bit data, No parity bit, 1 stop bit)
- Auto-baud detection is supported during Serial Boot.
- The host tool needs to send out the data packet with maximum payload size (512 bytes) set in framing packet by packet, except for the last packet. ROM will end the transfer if the payload size is less than the maximum payload size.
- The first packet on UART must be Ping to perform auto-baud detection.

12.8.2.2 LPSPI Configuration Details

Provided below are the details of LPUART configuration in boot ROM:

- 4-wire interface: CS, SCK, MOSI(SIN) and MISO(SOUT)
- After the host tool sent out a request, ROM may need some time to process the request before sending out actual response. If SPI host is expecting a response packet, the host tool should poll for start byte(0x5A). A valid start byte, instead of an empty byte(0x00), indicates ROM has done processing and response packet is ready.
- Recommended polling interval is ≥120μs.
- The host tool needs to send out the data packet with maximum payload size (512 bytes) set in framing packet by packet, except for the last packet. ROM will end the transfer if the payload size is less than the maximum payload size.

12.8.2.3 Example Communication Sequences

1. **Ping** (for auto-baud detection)
   Host: Ping packet [5a **a6**]
   Device: Ping response < 5a **a7** 00 02 01 50 00 00 aa ea>
2. **Command packet** (Get-Property 1)
   Host: Command [5a **a4** 0c 00 4b 33 07 00 00 02 01 00 00 00 00 00 00 00]
   - 5a - Start Byte
   - **a4** - Packet Type: Command packet
   - 0c 00 - Payload size(0x000c)
   - 4b 33 - CRC checksum(0x334b)
   - 07 00 00 02 01 00 00 00 00 00 00 00 - Payload (Get-property 1)

   Device: Ack < 5a **a1**>
   - 5a - Start Byte
   - **a1 -** ACK

   Device: GetPropertyResponse < 5a **a4** 0c 00 64 18 a7 00 00 02 00 00 00 00 00 01 02 4b>
   - 5a - Start Byte
   - **a4** - Packet Type: Command packet
   - 0c 00 - Payload size (0x000c)
   - 64 18 - CRC Checksum (0x1864)
   - a7 00 00 02 00 00 00 00 00 01 02 4b - Payload (GetPropertyResponse, status = 0, property = 0x4b020100)

   Host: Ack [5a **a1**]
   - 5a - Start Byte
   - **a1 -** ACK
3. **Load Image**
   1. First Data packet:
      Host: Data packet [5a **a5** 00 02 55 03 d1 00 20 41 …]
      - 5a - Start Byte
      - **a5** - Packet Type: Data packet
      - 00 02 - Payload size(0x0200)
      - 55 03 - CRC Checksum(0x0355)
      - d1 00 20 41 … - Payload (Data stream)

      Device: Ack < 5a **a1**>
      - 5a - Start Byte
      - **a1 -** ACK

   2. Report the remaining data transfer following the same flow as the first data packet.
   3. Last Data packet:
      Host: Data packet [5a **a5** 84 00 37 39 07 46 38 …]
      - 5a - Start Byte
      - **a5** - Packet Type: Data packet
      - 84 00 - Payload size(0x0084)
      - 37 39 - CRC Checksum(0x3937)
      - 07 46 38 … - Payload (Data stream)

      Device: Ack < 5a **a1**>

Figure 60. Example Communication sequences

12.8.3 Serial Boot via USB-HID

12.8.3.1 USB configuration Details

The USB OTG function device driver supports a high speed (HS for UTMI) non-stream mode with a maximal packet size of 512 B and a low-level USB OTG function.

The VID/PID and strings for the USB device driver are listed in the following table.

Table 110. VID/PID and strings for USB device driver

Descriptor	Value
VID	0x1FC9 (NXP vendor ID)
PID[1]	0x014C
String Descriptor1 (manufacturer)	NXP SEMICONDUCTORS
String Descriptor2 (product)	#LC Blank RT Family **NOTE** #LC is the 2 characters of the abbreviation for lifecycles: OEM OPEN: OO OEM SECURE: OS OEM CLOSED: OC OEM FIELD RETURN: FO OEM LOCKED: OL
String Descriptor3	MCU HID GENERIC DEVICE

1. Allocation based on the BPN (Before Part Number)

12.8.3.2 Endpoints

The HID peripheral uses 3 endpoints:

- Control (0)
- Interrupt IN (1)
- Interrupt OUT (2)

The interrupt OUT endpoint is optional for HID class devices, but the boot ROM uses it as a pipe, where the firmware can NAK send requests from the USB host.

12.8.3.3 Example Communication Sequences

1. **Command packet** (Get-Property 1)
   Host: Command [**01** 00 0c 00 07 00 00 02 01 00 00 00 00 00 00 00]
   - **01** - Command report, Direction: OUT, Host to device
   - 00 - Padding byte
   - 0c 00 - Payload size(0x000c)
   - 07 00 00 02 01 00 00 00 00 00 00 00 - payload

   Device: Ack (USB Hardware Ack)
   Device: GetPropertyResponse < **03** 00 0c 00 a7 00 00 02 00 00 00 00 00 01 03 4b 00 00 00 00 00 … 00>
   - **03 -** Command report, Direction: IN, Device to host
   - 00 - Padding byte
   - 0c 00 - Payload size(0x000c)
   - a7 00 00 02 00 00 00 00 00 01 02 4b - Payload (GetPropertyResponse, status=0, property=0x4b030100)
   - 00 00 00 00 00 … 00 - Padding bytes

   Host: Ack (USB Hardware Ack)
2. **Load Image**
   1. First Data packet:
      Host: Data packet [**02** 00 f4 03 d1 00 20 41 01 a4 00 00 00 00 00 00 00 …]
      02 - Data report, Direction: OUT, Host to device
      00 - Padding byte
      f4 03 - Payload size(0x03f4)
      d1 00 20 41 01 a4 00 00 00 00 00 00 00 … - Payload
      Device: Ack (USB Hardware Ack)
   2. Report the remaining data transfer following the same flow as the first data packet.
   3. Last Data packet:
      Host: Data packet [**02** 00 d4 00 20 c5 28 d9 00 f0 0b f8 00 …]
      02 - Data report, Direction: OUT, Host to device
      00 - Padding byte
      d4 00 - Payload size(0x00d4)
      20 c5 28 d9 00 f0 0b f8 00 … - Payload
      Device: Ack (USB Hardware Ack)

Figure 61. Example Communication sequences

12.9 ROM APIs

12.9.1 Introduction

The ROM bootloader provides a set of ROM APIs to simplify the In-Application Programming (IAP).

The ROM bootloader supports the following APIs:

- runBootloader API
- FlexSPI NOR FLASH Driver API

The ROM API root pointer for Cortex-M33 core is located at address 0x1000_001C. There is no ROM API root pointer for Cortex-M7 due to the vector table is only for Cortex_M33 core. The ROM API root for Cortex-M7 is located at 0x0011_F000. Please see the following figure for details of the ROM API layout.

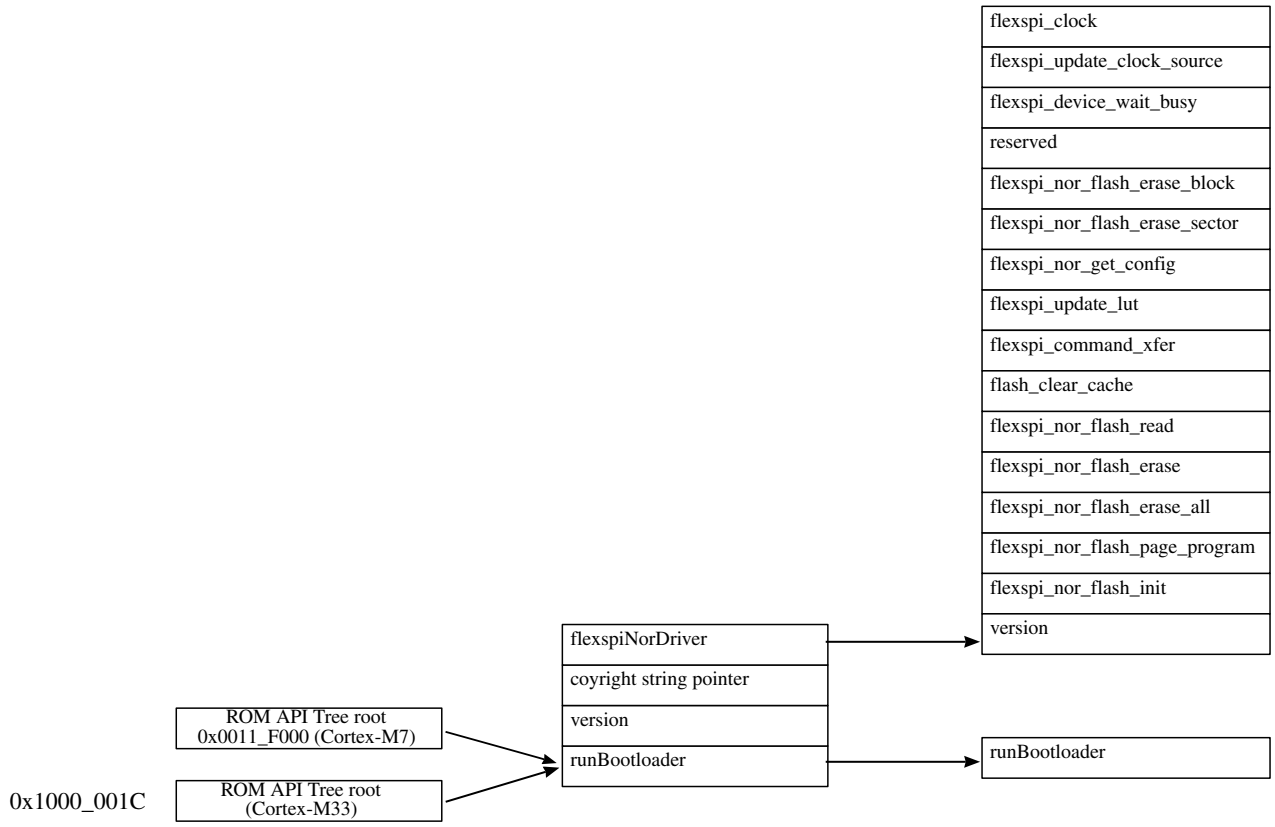


Figure 62. ROM API Structure

The ROM API structure definitions are as below.

- Bootloader API Entry Structure

```
                    typedef struct BootloaderTree
                    {
                        void (*runBootloader)(void *arg);                   //Function to start
the bootloader executing.
                        standard_version_t version;                         //Bootloader
version number.
                        const char *copyright;                              //Copyright string.
                        const flexspi_nor_flash_driver_t *flexspiNorDriver; //FlexSPI NOR FLASH
Driver API.
                    } bootloader_tree_t;
```

- FlexSPI NOR Driver API structure

```
                typedef struct
                {
                    uint32_t version;
                    status_t (*init)(uint32_t instance, flexspi_nor_config_t *config);

                    status_t (*page_program)(uint32_t instance, flexspi_nor_config_t *config,
uint32_t dstAddr, const uint32_t *src);
```

```
                    status_t (*erase_all)(uint32_t instance, flexspi_nor_config_t *config);
                    status_t (*erase)(uint32_t instance, flexspi_nor_config_t *config, uint32_t
start, uint32_t length);

                    status_t (*read)(uint32_t instance, flexspi_nor_config_t *config, uint32_t
*dst, uint32_t start, uint32_t bytes);
                    void (*clear_cache)(unit32_t instance);
                    status_t (*xfer)(uint32_t instance, flexspi_xfer_t *xfer);

                    status_t (*update_lut)(uint32_t instance, uint32_t seqIndex, const uint32_t
*lutBase, uint32_t numberOfSeq);
                    status_t (*get_config)(uint32_t instance, flexspi_nor_config_t *config,
serial_nor_config_option_t *option);
                    status_t (*erase_sector)(uint32_t instance, flexspi_nor_config_t *config,
uint32_t address);
                    status_t (*erase_block)(uint32_t instance, flexspi_nor_config_t *config,
uint32_t address);
                    uint32_t reserved;
                    status_t (*wait_busy)(uint32_t instance, flexspi_nor_config_t *config, bool
isParallelMode, uint32_t address);
                    status_t (*set_clock_source)(uint32_t instance, uint32_t clockSrc);
                    void (*config_clock)(uint32_t instance, uint32_t freqOption,
uint32_t sampleClkMode);

                } flexspi_nor_flash_driver_t;
```

12.9.2 FlexSPI NOR APIs

The ROM bootloader provides a set of Serial NOR FLASH APIs to simplify the external FLASH enablement on the chip. The version of the FlexSPI NOR API in the chip's ROM bootloader is 1.8.3.

12.9.2.1 Flexspi_nor_get_config

This API is used for getting the flash_config context.

Prototype

status_t flexspi_nor_get_config(uint32_t instance, flexspi_nor_config_t *config, serial_nor_config_option_t *option)

The parameters are defined below:

Parameter	Description
instance	FlexSPI instance
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
option	Serial NOR flash configuration option

The flexspi_nor_config_t is defined as following:

```
          typedef struct _flexspi_nor_config
          {
              flexspi_mem_config_t memConfig; // Common memory configuration info via FlexSPI
              uint32_t pageSize;              // Page size of Serial NOR
              uint32_t sectorSize;            // Sector size of Serial NOR
              uint8_t ipcmdSerialClkFreq;     // Clock frequency for IP command
              uint8_t isUniformBlockSize;     // Sector/Block size is the same
```

```
                uint8_t isDataOrderSwapped;     // Data order (D0, D1, D2, D3) is swapped (D1,D0, D3,
D2)
                uint8_t reserved0[1];           // Reserved for future use
                uint8_t serialNorType;          // Serial NOR Flash type: 0/1/2/3
                uint8_t needExitNoCmdMode;      // Need to exit NoCmd mode before other IP command
                uint8_t halfClkForNonReadCmd;   // Half the Serial Clock for non-read command: true/
false
                uint8_t needRestoreNoCmdMode;   // Need to Restore NoCmd mode after IP commmand
execution
                uint32_t blockSize;             // Block size
                uint32_t flashStateCtx;         // Flash State Context
                uint32_t reserve2[10];          // Reserved for future use
            } flexspi_nor_config_t;
```

The flexspi_mem_config_t is defined as following:

```


            typedef struct _FlexSPIConfig
            {
                uint32_t tag;               // [0x000-0x003] Tag, fixed value 0x42464346UL
                uint32_t version;           // [0x004-0x007] Version,[31:24] -'V', [23:16] - Major,
[15:8] - Minor, [7:0] - bugfix
                uint32_t reserved0;         // [0x008-0x00b] Reserved for future use
                uint8_t readSampleClkSrc;   // [0x00c-0x00c] Read Sample Clock Source, valid value:
0/1/3
                uint8_t csHoldTime;         // [0x00d-0x00d] CS hold time, default value: 3
                uint8_t csSetupTime;        // [0x00e-0x00e] CS setup time, default value: 3
                uint8_t columnAddressWidth; // [0x00f-0x00f] Column Address with, for HyperBus
protocol, it is fixed to 3, For
                //! Serial NAND, need to refer to datasheet
                uint8_t deviceModeCfgEnable; // [0x010-0x010] Device Mode Configure enable flag, 1 -
Enable, 0 - Disable
                uint8_t deviceModeType; // [0x011-0x011] Specify the configuration command type:Quad
Enable, DPI/QPI/OPI switch,
                //! Generic configuration, etc.
                uint16_t waitTimeCfgCommands; // [0x012-0x013] Wait time for all configuration
commands, unit: 100us, Used for
                //! DPI/QPI/OPI switch or reset command
                flexspi_lut_seq_t deviceModeSeq; // [0x014-0x017] Device mode sequence info, [7:0] -
LUT sequence id, [15:8] - LUt
                //! sequence number, [31:16] Reserved
                uint32_t deviceModeArg;    // [0x018-0x01b] Argument/Parameter for device
configuration
                uint8_t configCmdEnable;   // [0x01c-0x01c] Configure command Enable Flag, 1 -
Enable, 0 - Disable
                uint8_t configModeType[3]; // [0x01d-0x01f] Configure Mode Type, similar as
deviceModeTpe
                flexspi_lut_seq_t
                configCmdSeqs[3]; // [0x020-0x02b] Sequence info for Device Configuration command,
similar as deviceModeSeq
                uint32_t reserved1;   // [0x02c-0x02f] Reserved for future use
                uint32_t configCmdArgs[3];     // [0x030-0x03b] Arguments/Parameters for device
Configuration commands
                uint32_t reserved2;            // [0x03c-0x03f] Reserved for future use
                uint32_t controllerMiscOption; // [0x040-0x043] Controller Misc Options, see Misc
feature bit definitions for more
                //! details
                uint8_t deviceType;    // [0x044-0x044] Device Type:  See Flash Type Definition for
more details
```

```
                uint8_t sflashPadType; // [0x045-0x045] Serial Flash Pad Type: 1 - Single, 2 - Dual,
4 - Quad, 8 - Octal
                uint8_t serialClkFreq; // [0x046-0x046] Serial Flash Frequencey, device specific
definitions, See System Boot
                //! Chapter for more details
                uint8_t lutCustomSeqEnable; // [0x047-0x047] LUT customization Enable, it is required
if the program/erase cannot
                //! be done using 1 LUT sequence, currently, only applicable to HyperFLASH
                uint32_t reserved3[2]; // [0x048-0x04f] Reserved for future use
                uint32_t sflashA1Size; // [0x050-0x053] Size of Flash connected to A1
                uint32_t sflashA2Size; // [0x054-0x057] Size of Flash connected to A2
                uint32_t sflashB1Size; // [0x058-0x05b] Size of Flash connected to B1
                uint32_t sflashB2Size; // [0x05c-0x05f] Size of Flash connected to B2
                uint16_t reserved4;    // [0x060-0x063] CS pad setting override value
                uint8_t csPadSettingOverrideEn;
                uint8_t csPadSettingOverride;
                uint16_t reserved5; // [0x064-0x067] SCK pad setting override value
                uint8_t sclkPadSettingOverrideEn;
                uint8_t sclkPadSettingOverride;
                uint16_t reserved6; // [0x068-0x06b] data pad setting override value
                uint8_t dataPadSettingOverrideEn;
                uint8_t dataPadSettingOverride;
                uint16_t reserved7; // [0x06c-0x06f] DQS pad setting override value
                uint8_t dqsPadSettingOverrideEn;
                uint8_t dqsPadSettingOverride;
                uint32_t timeoutInMs;                // [0x070-0x073] Timeout threshold for read
status command
                uint32_t commandInterval;            // [0x074-0x077] CS deselect interval between
two commands
                flexspi_dll_time_t dataValidTime[2]; // [0x078-0x07b] CLK edge to data valid time for
PORT A and PORT B
                uint16_t busyOffset;                 // [0x07c-0x07d] Busy offset, valid value: 0-31
                uint16_t busyBitPolarity; // [0x07e-0x07f] Busy flag polarity, 0 - busy flag is 1
when flash device is busy, 1 -
                //! busy flag is 0 when flash device is busy
                uint32_t lookupTable[64];           // [0x080-0x17f] Lookup table holds Flash command
sequences
                flexspi_lut_seq_t lutCustomSeq[12]; // [0x180-0x1af] Customizable LUT Sequences
                uint32_t reserved8[4];              // [0x1b0-0x1bf] Reserved for future use
            } flexspi_mem_config_t;
```

The serial_nor_config_option_t is defined as following:

```


            typedef struct _serial_nor_config_option
            {
            union
            {
            struct
            {
            uint32_t max_freq : 4;                  // Maximum supported Frequency
            uint32_t misc_mode : 4;                 // miscellaneous mode
            uint32_t quad_mode_setting : 4;     // Quad mode setting
            uint32_t cmd_pads : 4;                  // Command pads
            uint32_t query_pads : 4;                // SFDP read pads
            uint32_t device_type : 4;               // Device type
            uint32_t option_size : 4;               // Option size, in terms of uint32_t, size =
(option_size + 1) * 4
            uint32_t tag : 4;                           // Tag, must be 0x0E
```

```
            } B;
            uint32_t U;
            } option0;

            union
            {
            struct
            {
            uint32_t dummy_cycles : 8;             // Dummy cycles before read
            uint32_t status_override : 8;          // Override status register value during device
mode configuration
            uint32_t pinmux_group : 4;            // The pinmux group selection
            uint32_t dqs_pinmux_group : 4;     // The DQS Pinmux Group Selection
            uint32_t pad_setting : 3;              // The Drive Strength of FlexSPI Pads
            uint32_t pad_setting_en : 1;        // The Drive Strength enable of FlexSPI Pads
            uint32_t flash_connection : 4;         // Flash connection option: 0 - Single Flash
connected to port A, -
            //! 1- Parallel mode, 2 - Single Flash connected to Port B
            } B;
            uint32_t U;
            } option1;

            } serial_nor_config_option_t;
```

Table 111. serial_nor_config_option_t definition

Offset	Field	Description
0	Option0	See Table 112 for more details.
4	Option1	Optional, valid only if the Option Size field in Option0 is non-zero. See Table 113 for more details.

Table 112. Option0 definition

Field	Bits	Description
tag	31:28	The tag of the config option, fixed to 0x0C
option_size	27:24	Size in bytes = (Option Size + 1) × 4 It is 0 if only option0 is required
device_type	23:20	Device Detection Type 0 - Read SFDP for SDRcommands 1 - Read SFDP for DDR Read commands 2 - HyperFLASH 1V8 3 - HyperFLASH 3V 4 - Macronix Octal DDR 6 - Micron Octal DDR 8 - Adesto EcoXiP DDR
query_pad	19:16	Data pads during Query command (read SFDP or read MID)

Table continues on the next page...

Table 112. Option0 definition (continued)

Field	Bits	Description
		0 - 1 2 - 4 3 – 8
cmd_pad	15:12	Data pads during Flash access command 0 - 1 2 - 4 3 – 8
quad_mode_sett ing	11:8	Quad Mode Enable Setting 0 - Not configured 1 - Set bit 6 in Status Register1 2 - Set bit 1 in Status Register2 3 - Set bit 7 in Status Register2 4 - Set bit 1 in Status Register 2 vis 0x31command **NOTE** This field is valid only if a device is compliant with JESD216 only (9 longword SDFP table).
misc_mode	7:4	Set to 0.
max_freq	3:0	Max Flash Operation speed 0 - Do not change FlexSPI clock setting **NOTE** The field has a restriction that the FlexSPI clock source must be PLL480_PFD0. Keep it as 0 and manually configure the FLEXSPI clock if another clock source is selected in user application.

Table 113. Option1 definition

Field	Bits	Description
flash_connection	31:28	Flash connection selection 0 - Single FLASH connected to Port A 1 - Parallel mode 2 - Single FLASH connected to Port B
reserved	27:20	Reserved

Table continues on the next page...

Table 113. Option1 definition (continued)

Field	Bits	Description
pin_group	19:16	PinMux group 0 - Primary pin group 1 - Secondary pingroup
reserved	15:8	Reserved
dummy_cycles	7:0	Dummy cycles for the read command 0 - Use detected dummy cycle Others - dummy cycles provided in flash datasheet

NOTE

- These APIs only support FLASH devices connected to SS0.
- The APIs always use 30MHz clock for the programming option. Users will need to change the "ipcmdSerialClkFreq" field in flexspi_nor_config_t field after flexspi_nor_get_config option, if a higher programming speed is expected.
- User application needs to set "max_freq" to 0 and manually configure the FLEXSPI clock before calling the flexspi_nor_get_config API, if the expected FLEXSPI clock source is not the default clock source configured by the ROM bootloader.
- The pad drive strength is configured to full-driver mode (see IOMUXC chapter for more details). Users can change the "dataPadOverride" and "sclkPadOverride" setting in flexspi_nor_config_t structure after calling flexspi_nor_get_config, if necessary.

12.9.2.2 flexspi_nor_flash_init

This API is used for initializing the flash controller and the flash_config context.

Prototype

status_t flexspi_nor_flash_init(uint32_t instance, flexspi_nor_config_t *config)

The parameters are defined below:

Parameter	Description
instance	FlexSPI instance
config	Pointer to flash_config_t data structure in memory to store driver runtime state.

12.9.2.3 flexspi_nor_flash_page_program

This API is used for programming data to Serial NOR via FlexSPI.

Prototype

status_t flexspi_nor_flash_page_program(uint32_t instance, flexspi_nor_config_t *config, uint32_t dstAddr, const uint32_t *src)

The parameters are defined below:

Parameter	Description
instance	FlexSPI instance
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
dstAddr	The destination address
src	The source address

12.9.2.4 flexspi_nor_flash_erase_all

This API is used for erasing all the Serial NOR devices connected on FlexSPI.

Prototype

status_t flexspi_nor_flash_erase_all(uint32_t instance, flexspi_nor_config_t *config)

The parameters are defined below:

Parameter	Description
instance	FlexSPI instance
config	Pointer to flash_config_t data structure in memory to store driver runtime state.

12.9.2.5 flexspi_nor_flash_erase

This API is used to erase Flash region specified by address and length.

Prototype

status_t flexspi_nor_flash_erase(uint32_t instance, flexspi_nor_config_t *config, uint32_t start, uint32_t length)

The parameters are defined below:

Parameter	Description
instance	FlexSPI instance
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
start	The start address to be erased.
length	The length of bytes to be erased.

The start address will be aligned down to the sector size. Erase size is sector aligned.

12.9.2.6 flexspi_nor_flash_read

This API is used for reading the FLASH via FLEXSPI using IP read command.

Prototype

status_t flexspi_nor_flash_read(uint32_t instance, flexspi_nor_config_t *config, uint32_t *dst, uint32_t start, uint32_t bytes)

The parameters are defined below:

Parameter	Description
instance	FlexSPI instance
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
dst	The rx buffer to be read to.
start	The start address to be read from.
bytes	Number of bytes to be read.

12.9.2.7 flexspi_update_lut

This API is used to update the specified LUT entries.

Prototype

status_t flexspi_update_lut(uint32_t instance, uint32_t seqIndex, const uint32_t *lutBase, uint32_t numberOfSeq)

The parameters are defined below:

Parameter	Description
instance	FlexSPI instance
seqIndex	The sequence index.
lutBase	Pointer to lookup table base.
numberOfSeq	Number of sequence.

12.9.2.8 flexspi_command_xfer

This API is used to perform FlexSPI command.

Prototype

status_t flexspi_command_xfer(uint32_t instance, flexspi_xfer_t *xfer)

The parameters are defined below:

Parameter	Description
instance	FlexSPI instance
xfer	Pointer to flexspi_xfer_t

flexspi_xfer_t definition is as below:

```
typedef struct _FlexSpiXfer
{
    flexspi_operation_t operation;        // FlexSPI operation
    uint32_t baseAddress;                     // FlexSPI operation base address
    uint32_t seqId;                       // Sequence Id
    uint32_t seqNum;                       // Sequence Number
    bool isParallelModeEnable;         // Is a parallel transfer
    uint32_t *txBuffer;                    // Tx buffer
    uint32_t txSize;                       // Tx size in bytes
    uint32_t *rxBuffer;                    // Rx buffer
```

```
    uint32_t rxSize;                    // Rx size in bytes
} flexspi_xfer_t;
```

flexspi_operation_t definition is as below:

```
typedef enum _FlexSPIOperationType
{
kFlexSpiOperation_Command,
kFlexSpiOperation_Config,
kFlexSpiOperation_Write,
kFlexSpiOperation_Read,
kFlexSpiOperation_End = kFlexSpiOperation_Read,
} flexspi_operation_t;
```

12.9.2.9 flexspi_clear_cache

This API is used to clear the AHB buffer in FlexSPI.

Prototype

void flexspi_clear_cache(uint32_t instance)

The parameters are defined below:

Parameter	Description
instance	FlexSPI instance

12.9.2.10 flexspi_update_clock_source

This API is used to configure the FLEXSPI NOR clock to specified frequency.

Prototype

status_t flexspi_update_clock_source(uint32_t instance, uint32_t source)

The parameters are defined below:

Parameter	Description
instance	FlexSPI instance
source	Clock sourcet. See the following table for details.

The source values are as defined below:

Value of source	FlexSPI1	FlexSPI2
0	Clk sourced from RC24M	
1	Clk sourced from RC400M	
2	Clk sourced from PFD0 of PLL480	Clk sourced from PFD2 of PLL480
3	Clk sourced from PFD0 of PLL528	Clk sourced from PFD1 of PLL528

12.9.2.11 flexspi_clock_config

This API is used to Configure the FLEXSPI clock to specified frequency.

Prototype

void flexspi_clock_config(uint32_t instance, uint32_t freq, uint32_t sampleClkMode)

The parameters are defined below:

Parameter	Description
instance	FlexSPI instance
freq	Enum of flexSpi serial clock.
sampleClkMode	Enum of clock mode.

flexSpi serial clock is defined as following.

```
enum
{
kFlexSpiSerialClk_30MHz = 1,
kFlexSpiSerialClk_50MHz = 2,
kFlexSpiSerialClk_60MHz = 3,
kFlexSpiSerialClk_80MHz = 4,
kFlexSpiSerialClk_100MHz = 5,
kFlexSpiSerialClk_120MHz = 6,
kFlexSpiSerialClk_133MHz = 7,
kFlexSpiSerialClk_166MHz = 8,

};
```

Clock mode is defined as following.

```
enum
{
kFlexSpiClk_SDR, // Clock configure for SDR mode
kFlexSpiClk_DDR, // Clock configurat for DDR mode
};
```

12.9.2.12 Status Codes for the FlexSPI NOR API

The following table lists all the error and status codes for the FlexSPI NOR API.

Table 114. Status and error codes for the FlexSPI NOR API

Status	Code	Description
kStatus_Success	0	Operation succeeded without error
kStatus_Fail	1	The operation failed with a generic error

Table continues on the next page...

Table 114. Status and error codes for the FlexSPI NOR API (continued)

Status	Code	Description
kStatus_InvalidArgument	4	The requested argument is invalid
kStatus_Timeout	5	A timeout occurred
kStatus_FLEXSPI_SequenceExecutionTimeout	7000	The command timed out
kStatus_FLEXSPI_InvalidSequence	7001	Invalid LUT sequence
kStatus_FLEXSPI_DeviceTimeout	7002	The busy time exceeded the specified timeout value
kStatus_FLEXSPINOR_ProgramFail	20100	Program Command failed
kStatus_FLEXSPINOR_EraseSectorFail	20101	Erase sector command failed
kStatus_FLEXSPINOR_EraseAllFail	20102	Erase All command failed
kStatus_FLEXSPINOR_WaitTimeout	20103	The wait time exceeded the specified timeout value
kStatus_FlexSPINOR_NotSupported	20104	The operation is not supported
kStatus_FlexSPINOR_WriteAlignmentEr ror	20105	Write address is unaligned to page size
kStatus_FlexSPINOR_CommandFailure	20106	Command failed
kStatus_FlexSPINOR_SFDP_NotFound	20107	SFDP table was not found, used for flexspi_nor_flash_get_config API
kStatus_FLEXSPINOR_Flash_NotFound	20109	Cannot detect a FLASH device
kStatus_FLEXSPINOR_DTRRead_Dum myProbeFailed	20110	The dummy cycle for DDR/DTR read cannot be probed.

12.9.2.12.1 Typical Options for the Serial NOR Devices in the Market

The following list provides typical options for the serial NOR flash devices in the market.

- QuadSPI NOR - Quad SDR Read: option0 = 0xc0000007 (133MHz)
- QuadSPI NOR - Quad DDR Read: option0 = 0xc0100003 (60MHz)
- HyperFLASH 1V8: option0 = 0xc0233008 (166MHz)
- HyperFLASH 3V0: option0 = 0xc0333005 (100MHz)
- MXIC OPI DDR (OPI DDR enabled by default): option=0xc0433007(133MHz)
- Micron Octal DDR: option0=0xc0600005 (100MHz)
- Micron OPI DDR: option0=0xc0603007 (133MHz), SPI->OPI DDR
- Micron OPI DDR (DDR read enabled by default): option0 = 0xc0633007 (133MHz)
- Adesto OPI DDR: option0=0xc0803007(133MHz)

12.9.2.12.2 FLASH API Example

The following is a typical use case of the FlexSPI NOR API.

```
            #ifdef CPU_IS_ARM_CORTEX_M7
            #define G_BOOTLOADER_TREE_ADDR         (0x0011f000u)
            #else
            #define G_BOOTLOADER_TREE_ADDR         ((uint32_t) (*(uint32_t *)(0x1000001cu)))
            #endif
            #define g_bootloaderTree              ((bootloader_tree_t *)G_BOOTLOADER_TREE_ADDR)

            flexspi_nor_config_t config;
            serial_nor_config_option_t option;
            status_t status;
            uint32_t address = 0x40000; // 256KB
            uint32_t sector_size = 0x1000; // 4KB
            uint32_t page_buffer[256/ sizeof(uint32_t)];
            uint32_t instance = 1;
            option.option0.U = 0xC0000007; // QuadSPI NOR, Frequency: 133MHz
            status = g_bootloaderTree->flexspiNorDriver->get_config(instance, &config, &option);
            if (status != kStatus_Success)
            {
            return status;
            }
            status = g_bootloaderTree->flexspiNorDriver->init(instance, &config);
            if (status != kStatus_Success)
            {
            return status;
            }
            status = g_bootloaderTree->flexspiNorDriver->erase(instance, &config, address ,
sector_size); // Erase 1 sector
            if (status != kStatus_Success)
            {
            return status;
            }

            // Fill data into the page_buffer;
            for (uint32_t i=0; i<sizeof(page_buffer)/sizeof(page_buffer[0]); i++)
                {
                page_buffer[i] = (i << 24) | (i << 16) | (i << 8) | i;
                }
                // Program data to destination
                status = g_bootloaderTree->flexspiNorDriver->page_program(instance, &config, address,
page_buffer); //Program 1 page
                if (status != kStatus_Success)
                {
                return status;
                }

                // Do cache maintenance here if the D-Cache is enabled
                // Use memory mapped access to verify whether data are programmed into Flash
correctly uint32_t mem_address = 0x3800_0000 + address;

                if (0 == memcmp((void*)mem_address, page_buffer, sizeof(page_buffer))
                {
                // Success
                }
                else
                {
```

```
                // Report error
                }
```

12.9.3 Enter Bootloader API

The ROM bootloader provides an API for the user application to safely boot from a newly updated application after In-Application Programming (IAP) or re-enter serial downloader mode for an image update.

Prototype

void run_bootloader(void *arg)

The parameters are defined below.

Parameter	Description
arg	See following table.

The arg definition is provided below.

Table 115. ARG definition

Field	Offset	Description
Tag	[31:24]	Fixed value: 0xEB (Enter Boot)
boot mode	[23:20]	0– Default boot mode based on bootloader. 1 – Force serial downloader
Serial downloader media	[19:16]	0 - Auto detection 1 - USB 2 - UART
Reserved	[15:04]	Reserved
Boot image selection	[03: 00]	0 - Image0 1 - Image1 2 - Image2 3 - Image3 **NOTE** It takes effect only if the boot mode is 0. For FlexSPI NOR boot, the maximum Image index is 1. The address and length of the backup image is specified in the fuse. Image 0 is always the latest image. Image 1 always implies the backup image. If the dual image boot feature is not implemented, the backup image means the 2nd image, otherwise, the backup image means the "old" image. For the NAND FLASH devices, the maximum image index is 3. The Image address is specified in FCB.

12.9.3.1 Enter Bootloader Example

Typical use cases:

1. Enter Serial downloader mode and select USB as the communication peripheral:

```
uint32_t arg = 0xeb110000;
runBootloader (&arg);
```

2. Select Image1 as the boot image after reliable update in user application:

```
uint32_t arg = 0xeb000001;
runBootloader (&arg);
```

12.10 Cortex-M7 kick-off procedure

The kick-off sequence for Cortex-M7 is as follows:

1. On POR, by default, Cortex-M7 is held in reset state and Cortex-M7 CPUWAIT is high (See M7_CFG[WAIT] in the chapter "Block Control Secure AONMIX").
2. Configure VTOR (See M7_CFG[INITVTOR] in the chapter "Block Control Secure AONMIX").
3. If you want to use non-default Cortex-M7 TCM size, configure M7_CFG[TCM_SIZE] (see chapter "Block Control Secure AONMIX"). This will be overridden by fuse CM7_TCM_CFG.
4. Release Cortex-M7 reset (See SCR[BT_RELEASE_M7] in the chapter "System Reset Controller (SRC)").
5. M33 enables DMA4 to initialize Cortex-M7 TCM to all zero by 64-bit write operation.

   NOTE

   If POR_PRELOAD_CM7_TCM_ECC fuse bit is programmed, skip the steps 4 and 5. But the VTOR is fixed to 0 and step 2 has no effect.

6. Load image of Cortex-M7 to target memory.
7. Authenticate the image using ELE OEM Container Authenticate followed by Verify Image and Release Container commands.

   NOTE

   If Cortex-M7 image is put as an individual image in container, ROM can authenticate image and get ELE to release it, may skip this step.

8. Check TRDC configuration, and ensure execution access allowable in target memory.
9. Send "Enable APC request" command to ELE to release Cortex-M7.
10. Clear Cortex-M7 CPUWAIT (See M7_CFG[WAIT] in the chapter "Block Control Secure AONMIX") to kick-off Cortex-M7.

    NOTE

    When the two applications of Cortex-M33 and Cortex-M7 run in the same device (for example, Flash, SDRAM), Cortex-M33 application needs to release TRDC for Cortex-M7 application before its kick-off.

Chapter 13
External Signals and Pin Multiplexing

13.1 Overview

The chip contains a limited number of pins, most of which have multiple signal options. These signal-to-pin and pin-to-signal options are selected by the input-output multiplexer called IOMUX. The IOMUX is also used to configure other pin characteristics, such as voltage level, drive strength. The hysteresis for pin input is supported in default, and can not be configured.

The muxing options table lists the external signals grouped by the module instance, the muxing options for each signal, and the registers used to route the signal to the chosen pad.

The IOMUX allows to share one pin/pad for several modules. This sharing is achieved by multiplexing pin/pad's input and output signals. Each pin/pad has more than one function, called alternate functions, and as result, each pin/pad can have many alternate logical pin names. The function selected depends on the pin setting in the appropriate IOMUX registers; each pin/pad has a set of dedicated registers.

13.1.1 Muxing Options

Table 116. Muxing Options

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
ACMP1	ACMP1_OUT	GPIO_AD_17	ALT1
	ACMP1_IN1	GPIO_AD_00	No muxing
	ACMP1_IN2	GPIO_AD_01	No muxing
	ACMP1_IN3	GPIO_AD_02	No muxing
	ACMP1_IN4	GPIO_AD_03	No muxing
ACMP2	ACMP2_OUT	GPIO_AD_18	ALT1
	ACMP2_IN1	GPIO_AD_04	No muxing
	ACMP2_IN2	GPIO_AD_05	No muxing
	ACMP2_IN3	GPIO_AD_26	No muxing
	ACMP2_IN4	GPIO_AD_27	No muxing
ACMP3	ACMP3_OUT	GPIO_AD_19	ALT1
	ACMP3_IN1	GPIO_AD_28	No muxing
	ACMP3_IN2	GPIO_AD_29	No muxing
	ACMP3_IN3	GPIO_AD_30	No muxing
	ACMP3_IN4	GPIO_AD_31	No muxing
ACMP4	ACMP4_OUT	GPIO_AD_20	ALT1
	ACMP4_IN1	GPIO_AD_32	No muxing
	ACMP4_IN2	GPIO_AD_33	No muxing
	ACMP4_IN3	GPIO_AD_34	No muxing
	ACMP4_IN4	GPIO_AD_35	No muxing

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
ADC1	ADC1_CH0A	GPIO_AD_16	No muxing
	ADC1_CH0B	GPIO_AD_17	No muxing
	ADC1_CH1A	GPIO_AD_14	No muxing
	ADC1_CH1B	GPIO_AD_15	No muxing
	ADC1_CH2A	GPIO_AD_12	No muxing
	ADC1_CH2B	GPIO_AD_13	No muxing
	ADC1_CH3A	GPIO_AD_10	No muxing
	ADC1_CH3B	GPIO_AD_11	No muxing
	ADC1_CH4A	GPIO_AD_08	No muxing
	ADC1_CH4B	GPIO_AD_09	No muxing
	ADC1_CH5A	GPIO_AD_06	No muxing
	ADC1_CH5B	GPIO_AD_07	No muxing
	ADC1_CH6A	GPIO_AD_04	No muxing
	ADC1_CH6B	GPIO_AD_05	No muxing
	ADC1_CH7A	GPIO_AD_02	No muxing
	ADC1_CH7B	GPIO_AD_03	No muxing
ADC2	ADC2_CH0A	GPIO_AD_18	No muxing
	ADC2_CH0B	GPIO_AD_19	No muxing
	ADC2_CH1A	GPIO_AD_20	No muxing
	ADC2_CH1B	GPIO_AD_21	No muxing
	ADC2_CH2A	GPIO_AD_22	No muxing
	ADC2_CH2B	GPIO_AD_23	No muxing
	ADC2_CH3A	GPIO_AD_24	No muxing
	ADC2_CH3B	GPIO_AD_25	No muxing
	ADC2_CH4A	GPIO_AD_26	No muxing
	ADC2_CH4B	GPIO_AD_27	No muxing
	ADC2_CH5A	GPIO_AD_28	No muxing
	ADC2_CH5B	GPIO_AD_29	No muxing
	ADC2_CH6A	GPIO_AD_30	No muxing
	ADC2_CH6B	GPIO_AD_31	No muxing
CCM	CCM_CLKO1	GPIO_EMC_B2_02	ALT9
	CCM_CLKO1	GPIO_SD_B1_00	ALT12

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	CCM_CLKO2	GPIO_EMC_B2_08	ALT10
	CCM_CLKO2	GPIO_SD_B1_01	ALT12
	CCM_ENET_REF_CLK_25M	GPIO_EMC_B2_00	ALT1
	CCM_ENET_REF_CLK_25M	GPIO_AD_05	ALT7
	CCM_ENET_REF_CLK_25M	GPIO_B1_13	ALT3
	CCM_ENET_REF_CLK_25M	GPIO_B2_02	ALT9
ECAT	ECAT_LED_STATE_RUN	GPIO_AON_03	ALT12
	ECAT_LED_RUN	GPIO_AON_04	ALT12
	ECAT_LED_ERR	GPIO_AON_05	ALT12
	ECAT_SDA	GPIO_AON_06	ALT12
	ECAT_SCL	GPIO_AON_07	ALT12
	ECAT_LINK_ACT[0]	GPIO_AON_08	ALT12
	ECAT_LINK_ACT[1]	GPIO_AON_09	ALT12
	ECAT_PT0_TXD[3]	GPIO_EMC_B1_00	ALT10
	ECAT_PT0_TXD[2]	GPIO_EMC_B1_01	ALT10
	ECAT_PT0_RX_CLK	GPIO_EMC_B1_02	ALT10
	ECAT_PT0_RXD[3]	GPIO_EMC_B1_03	ALT10
	ECAT_PT0_RXD[2]	GPIO_EMC_B1_04	ALT10
	ECAT_PT0_TXD[0]	GPIO_EMC_B1_05	ALT10
	ECAT_PT0_TXD[1]	GPIO_EMC_B1_06	ALT10
	ECAT_PT0_TX_EN	GPIO_EMC_B1_07	ALT10
	ECAT_PT0_TX_CLK	GPIO_EMC_B1_08	ALT10
	ECAT_PT0_RXD[0]	GPIO_EMC_B1_09	ALT10
	ECAT_PT0_RXD[1]	GPIO_EMC_B1_10	ALT10
	ECAT_PT0_RX_DV	GPIO_EMC_B1_11	ALT10
	ECAT_PT0_RX_ER	GPIO_EMC_B1_12	ALT10
	ECAT_PT1_TXD[1]	GPIO_EMC_B1_26	ALT6
	ECAT_PT1_TXD[0]	GPIO_EMC_B1_27	ALT6
	ECAT_PT1_TX_EN	GPIO_EMC_B1_28	ALT6
	ECAT_PT1_TX_CLK	GPIO_EMC_B1_29	ALT6
	ECAT_PT1_RXD[0]	GPIO_EMC_B1_30	ALT6
	ECAT_PT1_RXD[1]	GPIO_EMC_B1_31	ALT6

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	ECAT_PT1_RX_DV	GPIO_EMC_B1_32	ALT6
	ECAT_PT1_RX_ER	GPIO_EMC_B1_33	ALT6
	ECAT_PT1_RXD[2]	GPIO_EMC_B1_34	ALT6
	ECAT_PT1_RXD[3]	GPIO_EMC_B1_35	ALT6
	ECAT_PT1_TXD[3]	GPIO_EMC_B1_36	ALT6
	ECAT_PT1_TXD[2]	GPIO_EMC_B1_37	ALT6
	ECAT_PT1_RX_CLK	GPIO_EMC_B1_38	ALT6
	ECAT_PT0_RX_CLK	GPIO_EMC_B2_00	ALT12
	ECAT_PT0_RXD[2]	GPIO_EMC_B2_01	ALT12
	ECAT_PT0_RXD[3]	GPIO_EMC_B2_02	ALT12
	ECAT_PT0_TXD[2]	GPIO_EMC_B2_03	ALT12
	ECAT_PT0_TXD[3]	GPIO_EMC_B2_04	ALT12
	ECAT_PT0_TXD[0]	GPIO_EMC_B2_05	ALT12
	ECAT_PT0_TXD[1]	GPIO_EMC_B2_06	ALT12
	ECAT_PT0_TX_EN	GPIO_EMC_B2_07	ALT12
	ECAT_PT0_TX_CLK	GPIO_EMC_B2_08	ALT12
	ECAT_PT0_RXD[0]	GPIO_EMC_B2_09	ALT12
	ECAT_PT0_RXD[1]	GPIO_EMC_B2_10	ALT12
	ECAT_PT0_RX_DV	GPIO_EMC_B2_11	ALT12
	ECAT_PT0_RX_ER	GPIO_EMC_B2_12	ALT12
	ECAT_PT1_TXD[0]	GPIO_EMC_B2_13	ALT12
	ECAT_PT1_TXD[1]	GPIO_EMC_B2_14	ALT12
	ECAT_PT1_TX_EN	GPIO_EMC_B2_15	ALT12
	ECAT_PT1_TX_CLK	GPIO_EMC_B2_16	ALT12
	ECAT_PT1_RXD[0]	GPIO_EMC_B2_17	ALT12
	ECAT_PT1_RXD[1]	GPIO_EMC_B2_18	ALT12
	ECAT_PT1_RX_DV	GPIO_EMC_B2_19	ALT12
	ECAT_PT1_RX_ER	GPIO_EMC_B2_20	ALT12
	ECAT_LINK_MII_0	GPIO_AD_16	ALT12
	ECAT_LINK_MII_1	GPIO_AD_17	ALT12
	ECAT_SCL	GPIO_AD_18	ALT12
	ECAT_SDA	GPIO_AD_19	ALT12

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	ECAT_LED_RUN	GPIO_AD_21	ALT12
	ECAT_LED_ERR	GPIO_AD_22	ALT12
	ECAT_LED_STATE_RUN	GPIO_AD_23	ALT12
	ECAT_LINK_ACT[0]	GPIO_AD_24	ALT12
	ECAT_LINK_ACT[1]	GPIO_AD_25	ALT12
	ECAT_MDC	GPIO_AD_30	ALT12
	ECAT_MDIO	GPIO_AD_31	ALT12
	ECAT_RESET_OUT	GPIO_SD_B1_02	ALT12
	ECAT_MDIO	GPIO_SD_B2_10	ALT12
	ECAT_MDC	GPIO_SD_B2_11	ALT12
	ECAT_PT1_RX_ER	GPIO_B2_01	ALT12
	ECAT_PT1_RXD[2]	GPIO_B2_02	ALT12
	ECAT_PT1_RXD[3]	GPIO_B2_03	ALT12
	ECAT_PT1_TXD[2]	GPIO_B2_04	ALT12
	ECAT_PT1_TXD[3]	GPIO_B2_05	ALT12
	ECAT_PT1_TXD[0]	GPIO_B2_06	ALT12
	ECAT_PT1_TXD[1]	GPIO_B2_07	ALT12
	ECAT_PT1_TX_EN	GPIO_B2_08	ALT12
	ECAT_PT1_TX_CLK	GPIO_B2_09	ALT12
	ECAT_PT1_RXD[0]	GPIO_B2_10	ALT12
	ECAT_PT1_RXD[1]	GPIO_B2_11	ALT12
	ECAT_PT1_RX_DV	GPIO_B2_12	ALT12
	ECAT_PT1_RX_CLK	GPIO_B2_13	ALT12
ETH0	ETH0_TX_DATA0	GPIO_EMC_B1_34	ALT9
	ETH0_TX_DATA1	GPIO_EMC_B1_35	ALT9
	ETH0_TX_EN	GPIO_EMC_B1_36	ALT9
	ETH0_TX_CLK	GPIO_EMC_B1_37	ALT9
	ETH0_RX_DATA0	GPIO_EMC_B1_38	ALT9
	ETH0_RX_DATA1	GPIO_EMC_B1_39	ALT9
	ETH0_RX_DV	GPIO_EMC_B1_40	ALT9
	ETH0_RX_ER	GPIO_EMC_B1_41	ALT9
	ETH0_RX_CLK	GPIO_EMC_B2_00	ALT4

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	ETH0_RX_DATA2	GPIO_EMC_B2_01	ALT4
	ETH0_RX_DATA3	GPIO_EMC_B2_02	ALT4
	ETH0_TX_DATA2	GPIO_EMC_B2_03	ALT4
	ETH0_TX_DATA3	GPIO_EMC_B2_04	ALT4
	ETH0_TX_DATA0	GPIO_EMC_B2_05	ALT3
	ETH0_TX_DATA1	GPIO_EMC_B2_06	ALT3
	ETH0_TX_EN	GPIO_EMC_B2_07	ALT3
	ETH0_TX_CLK	GPIO_EMC_B2_08	ALT3
	ETH0_RX_DATA0	GPIO_EMC_B2_09	ALT3
	ETH0_RX_DATA1	GPIO_EMC_B2_10	ALT3
	ETH0_RX_DV	GPIO_EMC_B2_11	ALT3
	ETH0_RX_ER	GPIO_EMC_B2_12	ALT3
	ETH0_TX_DATA3	GPIO_EMC_B2_13	ALT3
	ETH0_TX_DATA2	GPIO_EMC_B2_14	ALT3
	ETH0_RX_CLK	GPIO_EMC_B2_15	ALT3
	ETH0_RX_DATA2	GPIO_EMC_B2_16	ALT3
	ETH0_RX_DATA3	GPIO_EMC_B2_17	ALT3
	ETH0_TX_ER	GPIO_EMC_B2_18	ALT3
	ETH0_CRS	GPIO_EMC_B2_19	ALT3
	ETH0_COL	GPIO_EMC_B2_20	ALT3
ETH1	ETH1_TX_DATA0	GPIO_B1_00	ALT0
	ETH1_TX_DATA1	GPIO_B1_01	ALT0
	ETH1_TX_EN	GPIO_B1_02	ALT0
	ETH1_TX_CLK	GPIO_B1_03	ALT0
	ETH1_RX_DATA0	GPIO_B1_04	ALT0
	ETH1_RX_DATA1	GPIO_B1_05	ALT0
	ETH1_RX_DV	GPIO_B1_06	ALT0
	ETH1_TX_DATA2	GPIO_B1_07	ALT0
	ETH1_TX_DATA3	GPIO_B1_08	ALT0
	ETH1_RX_DATA2	GPIO_B1_09	ALT0
	ETH1_RX_DATA3	GPIO_B1_10	ALT0
	ETH1_RX_CLK	GPIO_B1_11	ALT0

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	ETH1_RX_ER	GPIO_B1_12	ALT0
	ETH1_TX_ER	GPIO_B1_13	ALT0
	ETH1_CRS	GPIO_B2_00	ALT0
	ETH1_COL	GPIO_B2_01	ALT0
ETH2	ETH2_RX_DV	GPIO_EMC_B1_13	ALT3
	ETH2_TX_EN	GPIO_EMC_B1_14	ALT3
	ETH2_TX_CLK	GPIO_EMC_B1_15	ALT3
	ETH2_RX_DATA0	GPIO_EMC_B1_16	ALT3
	ETH2_MDC	GPIO_EMC_B1_16	ALT9
	ETH2_RX_DATA1	GPIO_EMC_B1_17	ALT3
	ETH2_MDIO	GPIO_EMC_B1_17	ALT9
	ETH2_CRS	GPIO_EMC_B1_18	ALT4
	ETH2_COL	GPIO_EMC_B1_19	ALT4
	ETH2_TX_ER	GPIO_EMC_B1_20	ALT4
	ETH2_RX_CLK	GPIO_EMC_B1_21	ALT4
	ETH2_RX_DATA2	GPIO_EMC_B1_22	ALT4
	ETH2_RX_DATA3	GPIO_EMC_B1_23	ALT4
	ETH2_TX_DATA3	GPIO_EMC_B1_24	ALT4
	ETH2_TX_DATA2	GPIO_EMC_B1_25	ALT4
	ETH2_TX_DATA1	GPIO_EMC_B1_26	ALT4
	ETH2_TX_DATA0	GPIO_EMC_B1_27	ALT4
	ETH2_TX_EN	GPIO_EMC_B1_28	ALT4
	ETH2_TX_CLK	GPIO_EMC_B1_29	ALT4
	ETH2_RX_DATA0	GPIO_EMC_B1_30	ALT4
	ETH2_RX_DATA1	GPIO_EMC_B1_31	ALT4
	ETH2_RX_DV	GPIO_EMC_B1_32	ALT4
	ETH2_RX_ER	GPIO_EMC_B1_33	ALT4
	ETH2_RX_CLK	GPIO_EMC_B1_33	ALT10
	ETH2_RX_DATA2	GPIO_EMC_B1_34	ALT4
	ETH2_RX_DATA3	GPIO_EMC_B1_35	ALT4
	ETH2_TX_DATA3	GPIO_EMC_B1_36	ALT4
	ETH2_TX_DATA2	GPIO_EMC_B1_37	ALT4

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	ETH2_RX_CLK	GPIO_EMC_B1_38	ALT4
	ETH2_TX_ER	GPIO_EMC_B1_39	ALT4
	ETH2_CRS	GPIO_EMC_B1_40	ALT4
	ETH2_MDC	GPIO_EMC_B1_40	ALT2
	ETH2_COL	GPIO_EMC_B1_41	ALT4
	ETH2_MDIO	GPIO_EMC_B1_41	ALT2
	ETH2_MDIO	GPIO_B2_00	ALT10
	ETH2_MDC	GPIO_B2_01	ALT10
	ETH2_RX_ER	GPIO_B2_01	ALT11
	ETH2_RX_DATA2	GPIO_B2_02	ALT11
	ETH2_RX_DATA3	GPIO_B2_03	ALT11
	ETH2_TX_DATA2	GPIO_B2_04	ALT11
	ETH2_TX_DATA3	GPIO_B2_05	ALT11
	ETH2_TX_DATA0	GPIO_B2_06	ALT11
	ETH2_TX_DATA1	GPIO_B2_07	ALT11
	ETH2_TX_EN	GPIO_B2_08	ALT11
	ETH2_TX_CLK	GPIO_B2_09	ALT11
	ETH2_RX_DATA0	GPIO_B2_10	ALT11
	ETH2_RX_DATA1	GPIO_B2_11	ALT11
	ETH2_RX_DV	GPIO_B2_12	ALT11
	ETH2_RX_CLK	GPIO_B2_13	ALT11
ETH3	ETH3_TX_DATA3	GPIO_EMC_B1_00	ALT4
	ETH3_TX_DATA2	GPIO_EMC_B1_01	ALT4
	ETH3_RX_CLK	GPIO_EMC_B1_02	ALT4
	ETH3_RX_DATA3	GPIO_EMC_B1_03	ALT4
	ETH3_RX_DATA2	GPIO_EMC_B1_04	ALT4
	ETH3_TX_DATA0	GPIO_EMC_B1_05	ALT4
	ETH3_TX_DATA1	GPIO_EMC_B1_06	ALT4
	ETH3_TX_EN	GPIO_EMC_B1_07	ALT4
	ETH3_TX_CLK	GPIO_EMC_B1_08	ALT4
	ETH3_RX_DATA0	GPIO_EMC_B1_09	ALT4
	ETH3_RX_DATA1	GPIO_EMC_B1_10	ALT4

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	ETH3_RX_DV	GPIO_EMC_B1_11	ALT4
	ETH3_RX_ER	GPIO_EMC_B1_12	ALT4
	ETH3_TX_ER	GPIO_EMC_B1_13	ALT4
	ETH3_CRS	GPIO_EMC_B1_14	ALT4
	ETH3_COL	GPIO_EMC_B1_15	ALT4
	ETH3_MDC	GPIO_EMC_B1_16	ALT4
	ETH3_MDIO	GPIO_EMC_B1_17	ALT4
	ETH3_MDC	GPIO_EMC_B1_24	ALT9
	ETH3_MDIO	GPIO_EMC_B1_25	ALT9
	ETH3_CRS	GPIO_EMC_B2_03	ALT9
	ETH3_COL	GPIO_EMC_B2_04	ALT9
	ETH3_TX_DATA0	GPIO_EMC_B2_05	ALT9
	ETH3_TX_DATA1	GPIO_EMC_B2_06	ALT9
	ETH3_TX_EN	GPIO_EMC_B2_07	ALT9
	ETH3_TX_CLK	GPIO_EMC_B2_08	ALT9
	ETH3_RX_DATA0	GPIO_EMC_B2_09	ALT9
	ETH3_RX_DATA1	GPIO_EMC_B2_10	ALT9
	ETH3_RX_DV	GPIO_EMC_B2_11	ALT9
	ETH3_RX_ER	GPIO_EMC_B2_12	ALT9
	ETH3_TX_DATA3	GPIO_EMC_B2_13	ALT9
	ETH3_TX_DATA2	GPIO_EMC_B2_14	ALT9
	ETH3_RX_CLK	GPIO_EMC_B2_15	ALT9
	ETH3_RX_DATA2	GPIO_EMC_B2_16	ALT9
	ETH3_RX_DATA3	GPIO_EMC_B2_17	ALT9
	ETH3_TX_ER	GPIO_EMC_B2_18	ALT9
	ETH3_MDC	GPIO_EMC_B2_19	ALT9
	ETH3_MDIO	GPIO_EMC_B2_20	ALT9
ETH4	ETH4_TX_DATA3	GPIO_EMC_B1_00	ALT9
	ETH4_TX_DATA2	GPIO_EMC_B1_01	ALT9
	ETH4_RX_CLK	GPIO_EMC_B1_02	ALT9
	ETH4_RX_DATA3	GPIO_EMC_B1_03	ALT9
	ETH4_RX_DATA2	GPIO_EMC_B1_04	ALT9

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	ETH4_TX_DATA0	GPIO_EMC_B1_05	ALT9
	ETH4_TX_DATA1	GPIO_EMC_B1_06	ALT9
	ETH4_TX_EN	GPIO_EMC_B1_07	ALT9
	ETH4_TX_CLK	GPIO_EMC_B1_08	ALT9
	ETH4_RX_DATA0	GPIO_EMC_B1_09	ALT9
	ETH4_RX_DATA1	GPIO_EMC_B1_10	ALT9
	ETH4_RX_DV	GPIO_EMC_B1_11	ALT9
	ETH4_RX_ER	GPIO_EMC_B1_12	ALT9
	ETH4_TX_ER	GPIO_EMC_B1_13	ALT9
	ETH4_CRS	GPIO_EMC_B1_14	ALT9
	ETH4_COL	GPIO_EMC_B1_15	ALT9
	ETH4_MDC	GPIO_EMC_B1_16	ALT6
	ETH4_MDIO	GPIO_EMC_B1_17	ALT6
	ETH4_MDC	GPIO_EMC_B2_05	ALT1
	ETH4_CRS	GPIO_EMC_B2_05	ALT4
	ETH4_MDIO	GPIO_EMC_B2_06	ALT1
	ETH4_COL	GPIO_EMC_B2_06	ALT4
	ETH4_TX_ER	GPIO_EMC_B2_07	ALT1
	ETH4_RX_CLK	GPIO_EMC_B2_08	ALT1
	ETH4_RX_DATA3	GPIO_EMC_B2_09	ALT1
	ETH4_RX_DATA2	GPIO_EMC_B2_10	ALT1
	ETH4_TX_DATA3	GPIO_EMC_B2_11	ALT1
	ETH4_TX_DATA2	GPIO_EMC_B2_12	ALT1
	ETH4_TX_DATA0	GPIO_EMC_B2_13	ALT1
	ETH4_TX_DATA1	GPIO_EMC_B2_14	ALT1
	ETH4_TX_EN	GPIO_EMC_B2_15	ALT1
	ETH4_TX_CLK	GPIO_EMC_B2_16	ALT1
	ETH4_RX_DATA0	GPIO_EMC_B2_17	ALT1
	ETH4_RX_DATA1	GPIO_EMC_B2_18	ALT1
	ETH4_RX_DV	GPIO_EMC_B2_19	ALT1
	ETH4_RX_ER	GPIO_EMC_B2_20	ALT1
	ETH4_TX_DATA0	GPIO_B1_00	ALT8

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	ETH4_TX_DATA1	GPIO_B1_01	ALT8
	ETH4_TX_EN	GPIO_B1_02	ALT8
	ETH4_TX_CLK	GPIO_B1_03	ALT8
	ETH4_RX_DATA0	GPIO_B1_04	ALT8
	ETH4_RX_DATA1	GPIO_B1_05	ALT8
	ETH4_RX_DV	GPIO_B1_06	ALT8
	ETH4_RX_ER	GPIO_B1_07	ALT8
	ETH4_TX_ER	GPIO_B1_08	ALT8
	ETH4_RX_DATA2	GPIO_B1_09	ALT8
	ETH4_RX_DATA3	GPIO_B1_10	ALT8
	ETH4_RX_CLK	GPIO_B1_11	ALT8
	ETH4_TX_DATA2	GPIO_B1_12	ALT8
	ETH4_TX_DATA3	GPIO_B1_13	ALT8
	ETH4_CRS	GPIO_B2_00	ALT8
	ETH4_COL	GPIO_B2_01	ALT8
FLEXCAN1	FLEXCAN1_TX	GPIO_AON_00	ALT1
	FLEXCAN1_RX	GPIO_AON_01	ALT1
	FLEXCAN1_TX	GPIO_AON_06	ALT6
	FLEXCAN1_RX	GPIO_AON_07	ALT6
	FLEXCAN1_TX	GPIO_AON_16	ALT6
	FLEXCAN1_RX	GPIO_AON_17	ALT6
	FLEXCAN1_TX	GPIO_AD_15	ALT9
	FLEXCAN1_RX	GPIO_AD_16	ALT9
FLEXCAN3	FLEXCAN3_TX	GPIO_AON_02	ALT1
	FLEXCAN3_RX	GPIO_AON_03	ALT1
	FLEXCAN3_TX	GPIO_AON_18	ALT6
	FLEXCAN3_RX	GPIO_AON_19	ALT6
	FLEXCAN3_TX	GPIO_AD_06	ALT1
	FLEXCAN3_RX	GPIO_AD_07	ALT1
	FLEXCAN3_TX	GPIO_B2_10	ALT2
	FLEXCAN3_RX	GPIO_B2_11	ALT2
	FLEXCAN3_TX	GPIO_B2_12	ALT6

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	FLEXCAN3_RX	GPIO_B2_13	ALT6
FLEXIO1	FLEXIO1_D00	GPIO_EMC_B1_00	ALT8
	FLEXIO1_D01	GPIO_EMC_B1_01	ALT8
	FLEXIO1_D02	GPIO_EMC_B1_02	ALT8
	FLEXIO1_D03	GPIO_EMC_B1_03	ALT8
	FLEXIO1_D04	GPIO_EMC_B1_04	ALT8
	FLEXIO1_D05	GPIO_EMC_B1_05	ALT8
	FLEXIO1_D06	GPIO_EMC_B1_06	ALT8
	FLEXIO1_D07	GPIO_EMC_B1_07	ALT8
	FLEXIO1_D08	GPIO_EMC_B1_08	ALT8
	FLEXIO1_D09	GPIO_EMC_B1_09	ALT8
	FLEXIO1_D10	GPIO_EMC_B1_10	ALT8
	FLEXIO1_D11	GPIO_EMC_B1_11	ALT8
	FLEXIO1_D12	GPIO_EMC_B1_12	ALT8
	FLEXIO1_D13	GPIO_EMC_B1_13	ALT8
	FLEXIO1_D14	GPIO_EMC_B1_14	ALT8
	FLEXIO1_D15	GPIO_EMC_B1_15	ALT8
	FLEXIO1_D16	GPIO_EMC_B1_16	ALT8
	FLEXIO1_D17	GPIO_EMC_B1_17	ALT8
	FLEXIO1_D18	GPIO_EMC_B1_18	ALT8
	FLEXIO1_D19	GPIO_EMC_B1_19	ALT8
	FLEXIO1_D20	GPIO_EMC_B1_20	ALT8
	FLEXIO1_D21	GPIO_EMC_B1_21	ALT8
	FLEXIO1_D22	GPIO_EMC_B1_22	ALT8
	FLEXIO1_D23	GPIO_EMC_B1_23	ALT8
	FLEXIO1_D24	GPIO_EMC_B1_24	ALT8
	FLEXIO1_D25	GPIO_EMC_B1_25	ALT8
FLEXIO2	FLEXIO2_D00	GPIO_AD_00	ALT8
	FLEXIO2_D01	GPIO_AD_01	ALT8
	FLEXIO2_D02	GPIO_AD_02	ALT8
	FLEXIO2_D03	GPIO_AD_03	ALT8
	FLEXIO2_D04	GPIO_AD_04	ALT8

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	FLEXIO2_D05	GPIO_AD_05	ALT8
	FLEXIO2_D06	GPIO_AD_06	ALT8
	FLEXIO2_D07	GPIO_AD_07	ALT8
	FLEXIO2_D08	GPIO_AD_08	ALT8
	FLEXIO2_D09	GPIO_AD_09	ALT8
	FLEXIO2_D10	GPIO_AD_10	ALT8
	FLEXIO2_D11	GPIO_AD_11	ALT8
	FLEXIO2_D12	GPIO_AD_12	ALT8
	FLEXIO2_D13	GPIO_AD_13	ALT8
	FLEXIO2_D14	GPIO_AD_14	ALT8
	FLEXIO2_D15	GPIO_AD_15	ALT8
	FLEXIO2_D16	GPIO_AD_16	ALT8
	FLEXIO2_D17	GPIO_AD_17	ALT8
	FLEXIO2_D18	GPIO_AD_18	ALT8
	FLEXIO2_D19	GPIO_AD_19	ALT8
	FLEXIO2_D20	GPIO_AD_20	ALT8
	FLEXIO2_D21	GPIO_AD_21	ALT8
	FLEXIO2_D22	GPIO_AD_22	ALT8
	FLEXIO2_D23	GPIO_AD_23	ALT8
	FLEXIO2_D24	GPIO_AD_24	ALT8
	FLEXIO2_D25	GPIO_AD_25	ALT8
	FLEXIO2_D26	GPIO_AD_26	ALT8
	FLEXIO2_D27	GPIO_AD_27	ALT8
	FLEXIO2_D28	GPIO_AD_28	ALT8
	FLEXIO2_D29	GPIO_AD_29	ALT8
	FLEXIO2_D30	GPIO_AD_30	ALT8
	FLEXIO2_D31	GPIO_AD_31	ALT8
FLEXPWM1	FLEXPWM1_PWM0_A	GPIO_EMC_B1_24	ALT1
	FLEXPWM1_PWM0_B	GPIO_EMC_B1_25	ALT1
	FLEXPWM1_PWM1_A	GPIO_EMC_B1_26	ALT1
	FLEXPWM1_PWM1_B	GPIO_EMC_B1_27	ALT1
	FLEXPWM1_PWM2_B	GPIO_EMC_B1_28	ALT1

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	FLEXPWM1_PWM2_A	GPIO_EMC_B1_29	ALT1
	FLEXPWM1_PWM0_A	GPIO_EMC_B1_36	ALT1
	FLEXPWM1_PWM0_B	GPIO_EMC_B1_37	ALT1
	FLEXPWM1_PWM3_B	GPIO_EMC_B1_38	ALT1
	FLEXPWM1_PWM3_A	GPIO_EMC_B1_39	ALT1
	FLEXPWM1_PWM0_A	GPIO_AD_00	ALT4
	FLEXPWM1_PWM0_B	GPIO_AD_01	ALT4
	FLEXPWM1_PWM1_A	GPIO_AD_02	ALT4
	FLEXPWM1_PWM1_B	GPIO_AD_03	ALT4
	FLEXPWM1_PWM2_B	GPIO_AD_04	ALT4
	FLEXPWM1_PWM2_A	GPIO_AD_05	ALT4
	FLEXPWM1_PWM0_X	GPIO_AD_06	ALT4
	FLEXPWM1_PWM1_X	GPIO_AD_07	ALT4
	FLEXPWM1_PWM2_X	GPIO_AD_08	ALT4
	FLEXPWM1_PWM3_X	GPIO_AD_09	ALT4
FLEXPWM2	FLEXPWM2_PWM3_B	GPIO_EMC_B1_08	ALT1
	FLEXPWM2_PWM3_A	GPIO_EMC_B1_09	ALT1
	FLEXPWM2_PWM0_A	GPIO_EMC_B1_18	ALT1
	FLEXPWM2_PWM0_B	GPIO_EMC_B1_19	ALT1
	FLEXPWM2_PWM1_A	GPIO_EMC_B1_20	ALT1
	FLEXPWM2_PWM1_B	GPIO_EMC_B1_21	ALT1
	FLEXPWM2_PWM2_B	GPIO_EMC_B1_22	ALT1
	FLEXPWM2_PWM2_A	GPIO_EMC_B1_23	ALT1
	FLEXPWM2_PWM0_X	GPIO_AD_10	ALT4
	FLEXPWM2_PWM1_X	GPIO_AD_11	ALT4
	FLEXPWM2_PWM2_X	GPIO_AD_12	ALT4
	FLEXPWM2_PWM3_X	GPIO_AD_13	ALT4
	FLEXPWM2_PWM0_A	GPIO_AD_24	ALT4
	FLEXPWM2_PWM0_B	GPIO_AD_25	ALT4
	FLEXPWM2_PWM1_A	GPIO_AD_26	ALT4
	FLEXPWM2_PWM1_B	GPIO_AD_27	ALT4
	FLEXPWM2_PWM2_B	GPIO_AD_28	ALT4

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	FLEXPWM2_PWM2_A	GPIO_AD_29	ALT4
FLEXPWM3	FLEXPWM3_PWM3_B	GPIO_EMC_B1_10	ALT1
	FLEXPWM3_PWM3_A	GPIO_EMC_B1_11	ALT1
	FLEXPWM3_PWM0_A	GPIO_EMC_B1_30	ALT1
	FLEXPWM3_PWM0_B	GPIO_EMC_B1_31	ALT1
	FLEXPWM3_PWM1_A	GPIO_EMC_B1_32	ALT1
	FLEXPWM3_PWM1_B	GPIO_EMC_B1_33	ALT1
	FLEXPWM3_PWM2_B	GPIO_EMC_B1_34	ALT1
	FLEXPWM3_PWM2_A	GPIO_EMC_B1_35	ALT1
	FLEXPWM3_PWM0_A	GPIO_EMC_B2_00	ALT10
	FLEXPWM3_PWM0_B	GPIO_EMC_B2_01	ALT10
	FLEXPWM3_PWM1_A	GPIO_EMC_B2_02	ALT10
	FLEXPWM3_PWM1_B	GPIO_EMC_B2_03	ALT10
	FLEXPWM3_PWM2_B	GPIO_EMC_B2_04	ALT10
	FLEXPWM3_PWM2_A	GPIO_EMC_B2_05	ALT10
	FLEXPWM3_PWM3_B	GPIO_EMC_B2_06	ALT10
	FLEXPWM3_PWM3_A	GPIO_EMC_B2_07	ALT10
	FLEXPWM3_PWM0_X	GPIO_AD_14	ALT4
	FLEXPWM3_PWM1_X	GPIO_AD_15	ALT4
	FLEXPWM3_PWM2_X	GPIO_AD_16	ALT4
	FLEXPWM3_PWM3_X	GPIO_AD_17	ALT4
FLEXPWM4	FLEXPWM4_PWM3_B	GPIO_EMC_B1_06	ALT1
	FLEXPWM4_PWM3_A	GPIO_EMC_B1_07	ALT1
	FLEXPWM4_PWM0_A	GPIO_EMC_B1_12	ALT1
	FLEXPWM4_PWM0_B	GPIO_EMC_B1_13	ALT1
	FLEXPWM4_PWM1_A	GPIO_EMC_B1_14	ALT1
	FLEXPWM4_PWM1_B	GPIO_EMC_B1_15	ALT1
	FLEXPWM4_PWM2_B	GPIO_EMC_B1_16	ALT1
	FLEXPWM4_PWM2_A	GPIO_EMC_B1_17	ALT1
	FLEXPWM4_PWM0_X	GPIO_AD_18	ALT4
	FLEXPWM4_PWM1_X	GPIO_AD_19	ALT4
	FLEXPWM4_PWM2_X	GPIO_AD_20	ALT4

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	FLEXPWM4_PWM3_X	GPIO_AD_21	ALT4
FLEXSPI1	FLEXSPI1_A_SS1_B	GPIO_SD_B1_02	ALT9
	FLEXSPI1_B_SS1_B	GPIO_SD_B1_03	ALT9
	FLEXSPI1_A_SS1_B	GPIO_SD_B1_04	ALT9
	FLEXSPI1_A_SS1_B	GPIO_SD_B1_04	ALT8
	FLEXSPI1_B_SS0_B	GPIO_SD_B1_05	ALT9
	FLEXSPI1_B_DATA4	GPIO_SD_B2_00	ALT1
	FLEXSPI1_B_DATA5	GPIO_SD_B2_01	ALT1
	FLEXSPI1_B_DATA6	GPIO_SD_B2_02	ALT1
	FLEXSPI1_B_DATA7	GPIO_SD_B2_03	ALT1
	FLEXSPI1_B_SS1_B	GPIO_SD_B2_04	ALT1
	FLEXSPI1_B_DQS	GPIO_SD_B2_05	ALT1
	FLEXSPI1_B_SS0_B	GPIO_SD_B2_06	ALT1
	FLEXSPI1_B_SCLK	GPIO_SD_B2_07	ALT1
	FLEXSPI1_B_DATA0	GPIO_SD_B2_08	ALT1
	FLEXSPI1_B_DATA1	GPIO_SD_B2_09	ALT1
	FLEXSPI1_B_DATA2	GPIO_SD_B2_10	ALT1
	FLEXSPI1_B_DATA3	GPIO_SD_B2_11	ALT1
	FLEXSPI1_B_DQS	GPIO_SD_B2_12_DUMMY	ALT1
	FLEXSPI1_A_DQS	GPIO_SD_B2_12_DUMMY	ALT0
	FLEXSPI1_B_SS1_B	GPIO_B1_02	ALT7
	FLEXSPI1_B_DQS	GPIO_B1_03	ALT7
	FLEXSPI1_B_SS0_B	GPIO_B1_04	ALT7
	FLEXSPI1_B_SCLK	GPIO_B1_05	ALT7
	FLEXSPI1_B_DATA7	GPIO_B1_06	ALT7
	FLEXSPI1_B_DATA6	GPIO_B1_07	ALT7
	FLEXSPI1_B_DATA5	GPIO_B1_08	ALT7
	FLEXSPI1_B_DATA4	GPIO_B1_09	ALT7
	FLEXSPI1_B_DATA3	GPIO_B1_10	ALT7
	FLEXSPI1_B_DATA2	GPIO_B1_11	ALT7
	FLEXSPI1_B_DATA1	GPIO_B1_12	ALT7
	FLEXSPI1_B_DATA0	GPIO_B1_13	ALT7

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	FLEXSPI1_A_SS1_B	GPIO_B2_01	ALT6
	FLEXSPI1_B_SCLK	GPIO_B2_02	ALT6
	FLEXSPI1_A_DATA4	GPIO_B2_03	ALT7
	FLEXSPI1_A_DATA5	GPIO_B2_04	ALT7
	FLEXSPI1_A_DATA6	GPIO_B2_05	ALT7
	FLEXSPI1_A_DATA7	GPIO_B2_06	ALT7
	FLEXSPI1_A_DQS	GPIO_B2_07	ALT7
	FLEXSPI1_A_SCLK	GPIO_B2_08	ALT7
	FLEXSPI1_A_SS0_B	GPIO_B2_09	ALT7
	FLEXSPI1_A_DATA0	GPIO_B2_10	ALT7
	FLEXSPI1_A_DATA1	GPIO_B2_11	ALT7
	FLEXSPI1_A_DATA2	GPIO_B2_12	ALT7
	FLEXSPI1_A_DATA3	GPIO_B2_13	ALT7
FLEXSPI2	FLEXSPI2_B_DATA3	GPIO_AON_15	ALT0
	FLEXSPI2_B_DATA2	GPIO_AON_16	ALT0
	FLEXSPI2_B_DATA1	GPIO_AON_17	ALT0
	FLEXSPI2_B_DATA0	GPIO_AON_18	ALT0
	FLEXSPI2_B_SCLK	GPIO_AON_19	ALT0
	FLEXSPI2_B_DQS	GPIO_AON_20	ALT0
	FLEXSPI2_A_SS1_B	GPIO_AON_20	ALT1
	FLEXSPI2_B_SS0_B	GPIO_AON_21	ALT0
	FLEXSPI2_A_DQS	GPIO_AON_21	ALT8
	FLEXSPI2_A_SS0_B	GPIO_AON_22	ALT0
	FLEXSPI2_A_SCLK	GPIO_AON_23	ALT0
	FLEXSPI2_A_DATA0	GPIO_AON_24	ALT0
	FLEXSPI2_A_DATA1	GPIO_AON_25	ALT0
	FLEXSPI2_A_DATA2	GPIO_AON_26	ALT0
	FLEXSPI2_A_DATA3	GPIO_AON_27	ALT0
	FLEXSPI2_A_DQS	GPIO_AON_28_DUMMY	ALT0
	FLEXSPI2_B_DQS	GPIO_AON_28_DUMMY	ALT1
	FLEXSPI2_B_DQS	GPIO_EMC_B1_21	ALT10
	FLEXSPI2_B_DATA3	GPIO_EMC_B1_22	ALT10

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	FLEXSPI2_B_DATA2	GPIO_EMC_B1_23	ALT10
	FLEXSPI2_B_DATA1	GPIO_EMC_B1_24	ALT10
	FLEXSPI2_B_DATA0	GPIO_EMC_B1_25	ALT10
	FLEXSPI2_A_SS1_B	GPIO_EMC_B1_26	ALT3
	FLEXSPI2_B_SS1_B	GPIO_EMC_B1_27	ALT3
	FLEXSPI2_B_SS0_B	GPIO_EMC_B1_28	ALT3
	FLEXSPI2_B_DQS	GPIO_EMC_B1_29	ALT3
	FLEXSPI2_B_DATA3	GPIO_EMC_B1_30	ALT3
	FLEXSPI2_B_DATA2	GPIO_EMC_B1_31	ALT3
	FLEXSPI2_B_DATA1	GPIO_EMC_B1_32	ALT3
	FLEXSPI2_B_DATA0	GPIO_EMC_B1_33	ALT3
	FLEXSPI2_B_SCLK	GPIO_EMC_B1_34	ALT3
	FLEXSPI2_A_DATA0	GPIO_EMC_B1_35	ALT3
	FLEXSPI2_A_DATA1	GPIO_EMC_B1_36	ALT3
	FLEXSPI2_A_DATA2	GPIO_EMC_B1_37	ALT3
	FLEXSPI2_A_DATA3	GPIO_EMC_B1_38	ALT3
	FLEXSPI2_A_SS0_B	GPIO_EMC_B1_39	ALT3
	FLEXSPI2_A_DQS	GPIO_EMC_B1_40	ALT3
	FLEXSPI2_A_SCLK	GPIO_EMC_B1_41	ALT3
GPIO1	GPIO1_IO00	GPIO_AON_00	ALT5
	GPIO1_IO01	GPIO_AON_01	ALT5
	GPIO1_IO02	GPIO_AON_02	ALT5
	GPIO1_IO03	GPIO_AON_03	ALT5
	GPIO1_IO04	GPIO_AON_04	ALT5
	GPIO1_IO05	GPIO_AON_05	ALT5
	GPIO1_IO06	GPIO_AON_06	ALT5
	GPIO1_IO07	GPIO_AON_07	ALT5
	GPIO1_IO08	GPIO_AON_08	ALT5
	GPIO1_IO09	GPIO_AON_09	ALT5
	GPIO1_IO10	GPIO_AON_10	ALT5
	GPIO1_IO11	GPIO_AON_11	ALT5
	GPIO1_IO12	GPIO_AON_12	ALT5

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	GPIO1_IO13	GPIO_AON_13	ALT5
	GPIO1_IO14	GPIO_AON_14	ALT5
	GPIO1_IO15	GPIO_AON_15	ALT5
	GPIO1_IO16	GPIO_AON_16	ALT5
	GPIO1_IO17	GPIO_AON_17	ALT5
	GPIO1_IO18	GPIO_AON_18	ALT5
	GPIO1_IO19	GPIO_AON_19	ALT5
	GPIO1_IO20	GPIO_AON_20	ALT5
	GPIO1_IO21	GPIO_AON_21	ALT5
	GPIO1_IO22	GPIO_AON_22	ALT5
	GPIO1_IO23	GPIO_AON_23	ALT5
	GPIO1_IO24	GPIO_AON_24	ALT5
	GPIO1_IO25	GPIO_AON_25	ALT5
	GPIO1_IO26	GPIO_AON_26	ALT5
	GPIO1_IO27	GPIO_AON_27	ALT5
	GPIO1_IO28	GPIO_AON_28_DUMMY	ALT5
GPIO2	GPIO2_IO00	GPIO_EMC_B1_00	ALT5
	GPIO2_IO01	GPIO_EMC_B1_01	ALT5
	GPIO2_IO02	GPIO_EMC_B1_02	ALT5
	GPIO2_IO03	GPIO_EMC_B1_03	ALT5
	GPIO2_IO04	GPIO_EMC_B1_04	ALT5
	GPIO2_IO05	GPIO_EMC_B1_05	ALT5
	GPIO2_IO06	GPIO_EMC_B1_06	ALT5
	GPIO2_IO07	GPIO_EMC_B1_07	ALT5
	GPIO2_IO08	GPIO_EMC_B1_08	ALT5
	GPIO2_IO09	GPIO_EMC_B1_09	ALT5
	GPIO2_IO10	GPIO_EMC_B1_10	ALT5
	GPIO2_IO11	GPIO_EMC_B1_11	ALT5
	GPIO2_IO12	GPIO_EMC_B1_12	ALT5
	GPIO2_IO13	GPIO_EMC_B1_13	ALT5
	GPIO2_IO14	GPIO_EMC_B1_14	ALT5
	GPIO2_IO15	GPIO_EMC_B1_15	ALT5

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	GPIO2_IO16	GPIO_EMC_B1_16	ALT5
	GPIO2_IO17	GPIO_EMC_B1_17	ALT5
	GPIO2_IO18	GPIO_EMC_B1_18	ALT5
	GPIO2_IO19	GPIO_EMC_B1_19	ALT5
	GPIO2_IO20	GPIO_EMC_B1_20	ALT5
	GPIO2_IO21	GPIO_EMC_B1_21	ALT5
	GPIO2_IO22	GPIO_EMC_B1_22	ALT5
	GPIO2_IO23	GPIO_EMC_B1_23	ALT5
	GPIO2_IO24	GPIO_EMC_B1_24	ALT5
	GPIO2_IO25	GPIO_EMC_B1_25	ALT5
	GPIO2_IO26	GPIO_EMC_B1_26	ALT5
	GPIO2_IO27	GPIO_EMC_B1_27	ALT5
	GPIO2_IO28	GPIO_EMC_B1_28	ALT5
	GPIO2_IO29	GPIO_EMC_B1_29	ALT5
	GPIO2_IO30	GPIO_EMC_B1_30	ALT5
	GPIO2_IO31	GPIO_EMC_B1_31	ALT5
GPIO3	GPIO3_IO00	GPIO_EMC_B1_32	ALT5
	GPIO3_IO01	GPIO_EMC_B1_33	ALT5
	GPIO3_IO02	GPIO_EMC_B1_34	ALT5
	GPIO3_IO03	GPIO_EMC_B1_35	ALT5
	GPIO3_IO04	GPIO_EMC_B1_36	ALT5
	GPIO3_IO05	GPIO_EMC_B1_37	ALT5
	GPIO3_IO06	GPIO_EMC_B1_38	ALT5
	GPIO3_IO07	GPIO_EMC_B1_39	ALT5
	GPIO3_IO08	GPIO_EMC_B1_40	ALT5
	GPIO3_IO09	GPIO_EMC_B1_41	ALT5
	GPIO3_IO10	GPIO_EMC_B2_00	ALT5
	GPIO3_IO11	GPIO_EMC_B2_01	ALT5
	GPIO3_IO12	GPIO_EMC_B2_02	ALT5
	GPIO3_IO13	GPIO_EMC_B2_03	ALT5
	GPIO3_IO14	GPIO_EMC_B2_04	ALT5
	GPIO3_IO15	GPIO_EMC_B2_05	ALT5

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	GPIO3_IO16	GPIO_EMC_B2_06	ALT5
	GPIO3_IO17	GPIO_EMC_B2_07	ALT5
	GPIO3_IO18	GPIO_EMC_B2_08	ALT5
	GPIO3_IO19	GPIO_EMC_B2_09	ALT5
	GPIO3_IO20	GPIO_EMC_B2_10	ALT5
	GPIO3_IO21	GPIO_EMC_B2_11	ALT5
	GPIO3_IO22	GPIO_EMC_B2_12	ALT5
	GPIO3_IO23	GPIO_EMC_B2_13	ALT5
	GPIO3_IO24	GPIO_EMC_B2_14	ALT5
	GPIO3_IO25	GPIO_EMC_B2_15	ALT5
	GPIO3_IO26	GPIO_EMC_B2_16	ALT5
	GPIO3_IO27	GPIO_EMC_B2_17	ALT5
	GPIO3_IO28	GPIO_EMC_B2_18	ALT5
	GPIO3_IO29	GPIO_EMC_B2_19	ALT5
	GPIO3_IO30	GPIO_EMC_B2_20	ALT5
GPIO4	GPIO4_IO00	GPIO_AD_00	ALT5
	GPIO4_IO01	GPIO_AD_01	ALT5
	GPIO4_IO02	GPIO_AD_02	ALT5
	GPIO4_IO03	GPIO_AD_03	ALT5
	GPIO4_IO04	GPIO_AD_04	ALT5
	GPIO4_IO05	GPIO_AD_05	ALT5
	GPIO4_IO06	GPIO_AD_06	ALT5
	GPIO4_IO07	GPIO_AD_07	ALT5
	GPIO4_IO08	GPIO_AD_08	ALT5
	GPIO4_IO09	GPIO_AD_09	ALT5
	GPIO4_IO10	GPIO_AD_10	ALT5
	GPIO4_IO11	GPIO_AD_11	ALT5
	GPIO4_IO12	GPIO_AD_12	ALT5
	GPIO4_IO13	GPIO_AD_13	ALT5
	GPIO4_IO14	GPIO_AD_14	ALT5
	GPIO4_IO15	GPIO_AD_15	ALT5
	GPIO4_IO16	GPIO_AD_16	ALT5

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	GPIO4_IO17	GPIO_AD_17	ALT5
	GPIO4_IO18	GPIO_AD_18	ALT5
	GPIO4_IO19	GPIO_AD_19	ALT5
	GPIO4_IO20	GPIO_AD_20	ALT5
	GPIO4_IO21	GPIO_AD_21	ALT5
	GPIO4_IO22	GPIO_AD_22	ALT5
	GPIO4_IO23	GPIO_AD_23	ALT5
	GPIO4_IO24	GPIO_AD_24	ALT5
	GPIO4_IO25	GPIO_AD_25	ALT5
	GPIO4_IO26	GPIO_AD_26	ALT5
	GPIO4_IO27	GPIO_AD_27	ALT5
	GPIO4_IO28	GPIO_AD_28	ALT5
	GPIO4_IO29	GPIO_AD_29	ALT5
	GPIO4_IO30	GPIO_AD_30	ALT5
	GPIO4_IO31	GPIO_AD_31	ALT5
GPIO5	GPIO5_IO00	GPIO_AD_32	ALT5
	GPIO5_IO01	GPIO_AD_33	ALT5
	GPIO5_IO02	GPIO_AD_34	ALT5
	GPIO5_IO03	GPIO_AD_35	ALT5
	GPIO5_IO04	GPIO_SD_B1_00	ALT5
	GPIO5_IO05	GPIO_SD_B1_01	ALT5
	GPIO5_IO06	GPIO_SD_B1_02	ALT5
	GPIO5_IO07	GPIO_SD_B1_03	ALT5
	GPIO5_IO08	GPIO_SD_B1_04	ALT5
	GPIO5_IO09	GPIO_SD_B1_05	ALT5
	GPIO5_IO10	GPIO_SD_B2_00	ALT5
	GPIO5_IO11	GPIO_SD_B2_01	ALT5
	GPIO5_IO12	GPIO_SD_B2_02	ALT5
	GPIO5_IO13	GPIO_SD_B2_03	ALT5
	GPIO5_IO14	GPIO_SD_B2_04	ALT5
	GPIO5_IO15	GPIO_SD_B2_05	ALT5
	GPIO5_IO16	GPIO_SD_B2_06	ALT5

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	GPIO5_IO17	GPIO_SD_B2_07	ALT5
	GPIO5_IO18	GPIO_SD_B2_08	ALT5
	GPIO5_IO19	GPIO_SD_B2_09	ALT5
	GPIO5_IO20	GPIO_SD_B2_10	ALT5
	GPIO5_IO21	GPIO_SD_B2_11	ALT5
	GPIO5_IO22	GPIO_SD_B2_12_DUMMY	ALT5
GPIO6	GPIO6_IO00	GPIO_B1_00	ALT5
	GPIO6_IO01	GPIO_B1_01	ALT5
	GPIO6_IO02	GPIO_B1_02	ALT5
	GPIO6_IO03	GPIO_B1_03	ALT5
	GPIO6_IO04	GPIO_B1_04	ALT5
	GPIO6_IO05	GPIO_B1_05	ALT5
	GPIO6_IO06	GPIO_B1_06	ALT5
	GPIO6_IO07	GPIO_B1_07	ALT5
	GPIO6_IO08	GPIO_B1_08	ALT5
	GPIO6_IO09	GPIO_B1_09	ALT5
	GPIO6_IO10	GPIO_B1_10	ALT5
	GPIO6_IO11	GPIO_B1_11	ALT5
	GPIO6_IO12	GPIO_B1_12	ALT5
	GPIO6_IO13	GPIO_B1_13	ALT5
	GPIO6_IO14	GPIO_B2_00	ALT5
	GPIO6_IO15	GPIO_B2_01	ALT5
	GPIO6_IO16	GPIO_B2_02	ALT5
	GPIO6_IO17	GPIO_B2_03	ALT5
	GPIO6_IO18	GPIO_B2_04	ALT5
	GPIO6_IO19	GPIO_B2_05	ALT5
	GPIO6_IO20	GPIO_B2_06	ALT5
	GPIO6_IO21	GPIO_B2_07	ALT5
	GPIO6_IO22	GPIO_B2_08	ALT5
	GPIO6_IO23	GPIO_B2_09	ALT5
	GPIO6_IO24	GPIO_B2_10	ALT5
	GPIO6_IO25	GPIO_B2_11	ALT5

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	GPIO6_IO26	GPIO_B2_12	ALT5
	GPIO6_IO27	GPIO_B2_13	ALT5
GPT1	GPT1_CAPTURE1	GPIO_AD_12	ALT2
	GPT1_CAPTURE2	GPIO_AD_13	ALT2
	GPT1_COMPARE1	GPIO_AD_14	ALT2
	GPT1_COMPARE2	GPIO_AD_15	ALT2
	GPT1_COMPARE3	GPIO_AD_16	ALT2
	GPT1_CLK	GPIO_AD_17	ALT2
GPT2	GPT2_CAPTURE1	GPIO_AD_00	ALT3
	GPT2_CAPTURE2	GPIO_AD_01	ALT3
	GPT2_COMPARE1	GPIO_AD_02	ALT3
	GPT2_COMPARE2	GPIO_AD_03	ALT3
	GPT2_COMPARE3	GPIO_AD_04	ALT3
	GPT2_CLK	GPIO_AD_05	ALT3
I3C1	I3C1_PUR	GPIO_AON_06	ALT1
	I3C1_SDA	GPIO_AON_15	ALT9
	I3C1_SCL	GPIO_AON_16	ALT9
	I3C1_SDA	GPIO_AON_20	ALT3
	I3C1_SCL	GPIO_AON_21	ALT3
	I3C1_PUR	GPIO_AD_28	ALT2
I3C2	I3C2_PUR	GPIO_AD_17	ALT6
	I3C2_SCL	GPIO_AD_18	ALT6
	I3C2_SDA	GPIO_AD_19	ALT6
	I3C2_SCL	GPIO_AD_34	ALT0
	I3C2_SDA	GPIO_AD_35	ALT0
JTAG	JTAG_MUX_TRSTB	GPIO_AON_10	ALT0
	JTAG_MUX_TDO	GPIO_AON_11	ALT0
	JTAG_MUX_TDI	GPIO_AON_12	ALT0
	JTAG_MUX_TCK	GPIO_AON_13	ALT0
	JTAG_MUX_TMS	GPIO_AON_14	ALT0
KPP	KPP_ROW3	GPIO_EMC_B1_00	ALT6
	KPP_COL3	GPIO_EMC_B1_01	ALT6

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	KPP_ROW2	GPIO_EMC_B1_02	ALT6
	KPP_COL2	GPIO_EMC_B1_03	ALT6
	KPP_ROW1	GPIO_EMC_B1_04	ALT6
	KPP_ROW7	GPIO_EMC_B1_05	ALT6
	KPP_COL7	GPIO_EMC_B1_06	ALT6
	KPP_ROW6	GPIO_EMC_B1_07	ALT6
	KPP_COL6	GPIO_EMC_B1_08	ALT6
	KPP_ROW5	GPIO_EMC_B1_09	ALT6
	KPP_COL5	GPIO_EMC_B1_10	ALT6
	KPP_ROW4	GPIO_EMC_B1_11	ALT6
	KPP_COL4	GPIO_EMC_B1_12	ALT6
	KPP_COL1	GPIO_EMC_B1_13	ALT6
	KPP_ROW0	GPIO_EMC_B1_14	ALT6
	KPP_COL0	GPIO_EMC_B1_15	ALT6
	KPP_ROW7	GPIO_AD_12	ALT3
	KPP_COL7	GPIO_AD_13	ALT3
	KPP_ROW6	GPIO_AD_14	ALT3
	KPP_COL6	GPIO_AD_15	ALT3
	KPP_ROW5	GPIO_AD_16	ALT3
	KPP_COL5	GPIO_AD_17	ALT3
	KPP_ROW4	GPIO_AD_18	ALT3
	KPP_COL4	GPIO_AD_19	ALT3
	KPP_ROW0	GPIO_AD_26	ALT6
	KPP_COL0	GPIO_AD_27	ALT6
	KPP_ROW3	GPIO_AD_28	ALT6
	KPP_COL3	GPIO_AD_29	ALT6
	KPP_ROW2	GPIO_AD_30	ALT6
	KPP_COL2	GPIO_AD_31	ALT6
	KPP_ROW1	GPIO_AD_32	ALT6
	KPP_COL1	GPIO_AD_33	ALT6
	KPP_ROW7	GPIO_SD_B1_00	ALT8
	KPP_COL7	GPIO_SD_B1_01	ALT8

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	KPP_ROW6	GPIO_SD_B1_02	ALT8
	KPP_COL6	GPIO_SD_B1_03	ALT8
	KPP_ROW1	GPIO_SD_B2_00	ALT4
	KPP_COL1	GPIO_SD_B2_01	ALT4
	KPP_ROW0	GPIO_SD_B2_02	ALT4
	KPP_COL0	GPIO_SD_B2_03	ALT4
	KPP_ROW3	GPIO_SD_B2_04	ALT4
LPI2C1	LPI2C1_SDA	GPIO_AON_06	ALT3
	LPI2C1_SCL	GPIO_AON_07	ALT3
	LPI2C1_SDA	GPIO_AON_08	ALT6
	LPI2C1_SCL	GPIO_AON_09	ALT6
	LPI2C1_SCLS	GPIO_AON_10	ALT6
	LPI2C1_SDAS	GPIO_AON_11	ALT6
	LPI2C1_HREQ	GPIO_AON_12	ALT6
	LPI2C1_SDA	GPIO_AON_20	ALT2
	LPI2C1_SCL	GPIO_AON_21	ALT2
	LPI2C1_SDA	GPIO_AON_24	ALT1
	LPI2C1_SCL	GPIO_AON_25	ALT1
LPI2C2	LPI2C2_SDA	GPIO_AON_15	ALT4
	LPI2C2_SCL	GPIO_AON_16	ALT4
	LPI2C2_SDA	GPIO_AON_17	ALT3
	LPI2C2_SCL	GPIO_AON_18	ALT3
	LPI2C2_SDA	GPIO_AON_22	ALT1
	LPI2C2_SCL	GPIO_AON_23	ALT1
LPI2C3	LPI2C3_SCL	GPIO_EMC_B2_00	ALT9
	LPI2C3_SDA	GPIO_EMC_B2_01	ALT9
	LPI2C3_SCL	GPIO_EMC_B2_19	ALT8
	LPI2C3_SDA	GPIO_EMC_B2_20	ALT8
	LPI2C3_HREQ	GPIO_AD_17	ALT9
	LPI2C3_SCL	GPIO_AD_18	ALT9
	LPI2C3_SDA	GPIO_AD_19	ALT9
LPI2C4	LPI2C4_SCL	GPIO_AD_24	ALT1

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	LPI2C4_SDA	GPIO_AD_25	ALT1
	LPI2C4_SCL	GPIO_B2_10	ALT6
	LPI2C4_SDA	GPIO_B2_11	ALT6
LPI2C5	LPI2C5_SCL	GPIO_AD_08	ALT1
	LPI2C5_SDA	GPIO_AD_09	ALT1
	LPI2C5_SCLS	GPIO_AD_12	ALT1
	LPI2C5_SDAS	GPIO_AD_13	ALT1
	LPI2C5_HREQ	GPIO_AD_14	ALT1
	LPI2C5_SCL	GPIO_AD_32	ALT0
	LPI2C5_SDA	GPIO_AD_33	ALT0
LPI2C6	LPI2C6_SCL	GPIO_B1_02	ALT2
	LPI2C6_SDA	GPIO_B1_03	ALT2
	LPI2C6_SCL	GPIO_B2_08	ALT6
	LPI2C6_SDA	GPIO_B2_09	ALT6
LPIT1	LPIT2_TRIGGER0	GPIO_AD_20	ALT2
	LPIT2_TRIGGER1	GPIO_AD_21	ALT2
	LPIT2_TRIGGER2	GPIO_AD_22	ALT2
	LPIT2_TRIGGER3	GPIO_AD_23	ALT2
LPIT2	LPIT2_TRIGGER0	GPIO_AD_20	ALT2
	LPIT2_TRIGGER1	GPIO_AD_21	ALT2
	LPIT2_TRIGGER2	GPIO_AD_22	ALT2
	LPIT2_TRIGGER3	GPIO_AD_23	ALT2
LPIT3	LPIT3_TRIGGER0	GPIO_B2_00	ALT2
	LPIT3_TRIGGER1	GPIO_B2_01	ALT2
	LPIT3_TRIGGER2	GPIO_B2_02	ALT2
	LPIT3_TRIGGER3	GPIO_B2_03	ALT2
LPSPI1	LPSPI1_SCK	GPIO_AON_04	ALT0
	LPSPI1_SCK	GPIO_AON_12	ALT8
	LPSPI1_PCS0	GPIO_AON_05	ALT0
	LPSPI1_PCS0	GPIO_AON_13	ALT8
	LPSPI1_PCS1	GPIO_AON_03	ALT2
	LPSPI1_PCS1	GPIO_AON_08	ALT8

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	LPSPI1_PCS2	GPIO_AON_09	ALT8
	LPSPI1_PCS3	GPIO_AON_03	ALT4
	LPSPI1_SOUT	GPIO_AON_06	ALT0
	LPSPI1_SOUT	GPIO_AON_14	ALT8
	LPSPI1_SIN	GPIO_AON_07	ALT0
	LPSPI1_SIN	GPIO_AON_15	ALT8
LPSPI2	LPSPI2_SCK	GPIO_AON_19	ALT1
	LPSPI2_SCK	GPIO_AON_22	ALT6
	LPSPI2_PCS0	GPIO_AON_10	ALT1
	LPSPI2_PCS0	GPIO_AON_16	ALT1
	LPSPI2_PCS0	GPIO_AON_25	ALT6
	LPSPI2_PCS1	GPIO_AON_15	ALT1
	LPSPI2_PCS1	GPIO_AON_21	ALT1
	LPSPI2_PCS2	GPIO_AON_26	ALT1
	LPSPI2_PCS3	GPIO_AON_02	ALT2
	LPSPI2_PCS3	GPIO_AON_27	ALT1
	LPSPI2_SOUT	GPIO_AON_18	ALT1
	LPSPI2_SOUT	GPIO_AON_02	ALT3
	LPSPI2_SOUT	GPIO_AON_23	ALT6
	LPSPI2_SIN	GPIO_AON_03	ALT3
	LPSPI2_SIN	GPIO_AON_17	ALT1
	LPSPI2_SIN	GPIO_AON_24	ALT6
LPSPI3	LPSPI3_SCK	GPIO_EMC_B2_04	ALT8
	LPSPI3_SCK	GPIO_AD_16	ALT7
	LPSPI3_SCK	GPIO_SD_B1_01	ALT6
	LPSPI3_PCS0	GPIO_EMC_B2_05	ALT8
	LPSPI3_PCS0	GPIO_AD_17	ALT7
	LPSPI3_PCS0	GPIO_SD_B1_00	ALT6
	LPSPI3_PCS1	GPIO_EMC_B2_10	ALT8
	LPSPI3_PCS1	GPIO_AD_15	ALT7
	LPSPI3_PCS1	GPIO_SD_B1_04	ALT6
	LPSPI3_PCS2	GPIO_EMC_B2_09	ALT8

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	LPSPI3_PCS2	GPIO_SD_B1_05	ALT6
	LPSPI3_PCS3	GPIO_EMC_B2_08	ALT8
	LPSPI3_PCS3	GPIO_SD_B2_00	ALT6
	LPSPI3_SOUT	GPIO_EMC_B2_06	ALT8
	LPSPI3_SOUT	GPIO_AD_18	ALT7
	LPSPI3_SOUT	GPIO_SD_B1_02	ALT6
	LPSPI3_SIN	GPIO_EMC_B2_07	ALT8
	LPSPI3_SIN	GPIO_AD_19	ALT7
	LPSPI3_SIN	GPIO_SD_B1_03	ALT6
LPSPI4	LPSPI4_SCK	GPIO_EMC_B1_22	ALT3
	LPSPI4_SCK	GPIO_SD_B2_08	ALT4
	LPSPI4_SCK	GPIO_B2_10	ALT9
	LPSPI4_PCS0	GPIO_EMC_B1_25	ALT3
	LPSPI4_PCS0	GPIO_SD_B2_09	ALT4
	LPSPI4_PCS0	GPIO_B2_13	ALT9
	LPSPI4_PCS1	GPIO_SD_B2_07	ALT4
	LPSPI4_PCS2	GPIO_SD_B2_06	ALT4
	LPSPI4_PCS3	GPIO_SD_B2_05	ALT4
	LPSPI4_SOUT	GPIO_EMC_B1_24	ALT3
	LPSPI4_SOUT	GPIO_SD_B2_10	ALT4
	LPSPI4_SOUT	GPIO_B2_12	ALT9
	LPSPI4_SIN	GPIO_EMC_B1_23	ALT3
	LPSPI4_SIN	GPIO_SD_B2_11	ALT4
	LPSPI4_SIN	GPIO_B2_11	ALT9
LPSPI5	LPSPI5_SCK	GPIO_EMC_B1_31	ALT9
	LPSPI5_SCK	GPIO_EMC_B2_00	ALT8
	LPSPI5_SCK	GPIO_AD_28	ALT0
	LPSPI5_PCS0	GPIO_AD_29	ALT0
	LPSPI5_PCS0	GPIO_EMC_B2_01	ALT8
	LPSPI5_PCS0	GPIO_EMC_B1_34	ALT10
	LPSPI5_PCS1	GPIO_EMC_B1_35	ALT10
	LPSPI5_PCS1	GPIO_EMC_B2_13	ALT4

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	LPSPI5_PCS1	GPIO_AD_27	ALT2
	LPSPI5_PCS2	GPIO_EMC_B2_09	ALT8
	LPSPI5_PCS2	GPIO_EMC_B2_12	ALT4
	LPSPI5_PCS2	GPIO_AD_26	ALT2
	LPSPI5_PCS3	GPIO_EMC_B2_11	ALT4
	LPSPI5_PCS3	GPIO_AD_25	ALT2
	LPSPI5_SOUT	GPIO_EMC_B1_32	ALT9
	LPSPI5_SOUT	GPIO_EMC_B2_02	ALT8
	LPSPI5_SOUT	GPIO_AD_30	ALT0
	LPSPI5_SIN	GPIO_EMC_B1_33	ALT9
	LPSPI5_SIN	GPIO_EMC_B2_03	ALT8
	LPSPI5_SIN	GPIO_AD_31	ALT0
LPSPI6	LPSPI6_SCK	GPIO_EMC_B1_18	ALT3
	LPSPI6_SCK	GPIO_EMC_B1_26	ALT10
	LPSPI6_SCK	GPIO_B1_13	ALT9
	LPSPI6_PCS0	GPIO_EMC_B1_21	ALT3
	LPSPI6_PCS0	GPIO_EMC_B1_29	ALT10
	LPSPI6_PCS0	GPIO_B1_12	ALT9
	LPSPI6_PCS1	GPIO_EMC_B1_17	ALT10
	LPSPI6_PCS1	GPIO_EMC_B1_30	ALT10
	LPSPI6_PCS1	GPIO_B1_09	ALT9
	LPSPI6_PCS2	GPIO_EMC_B1_16	ALT10
	LPSPI6_PCS2	GPIO_EMC_B1_31	ALT10
	LPSPI6_PCS2	GPIO_B1_10	ALT9
	LPSPI6_PCS3	GPIO_EMC_B1_32	ALT10
	LPSPI6_PCS3	GPIO_B1_11	ALT9
	LPSPI6_SOUT	GPIO_EMC_B1_20	ALT3
	LPSPI6_SOUT	GPIO_EMC_B1_28	ALT10
	LPSPI6_SOUT	GPIO_B1_08	ALT9
	LPSPI6_SOUT	GPIO_B2_01	ALT9
	LPSPI6_SIN	GPIO_EMC_B1_19	ALT3
	LPSPI6_SIN	GPIO_EMC_B1_27	ALT10

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	LPSPI6_SIN	GPIO_B1_07	ALT9
	LPSPI6_SIN	GPIO_B2_00	ALT9
LPTMR1	LPTMR1_ALT1	GPIO_AON_00	ALT4
	LPTMR1_ALT2	GPIO_AON_01	ALT4
	LPTMR1_ALT3	GPIO_AON_02	ALT4
	LPTMR1_ALT1	GPIO_AON_13	ALT6
	LPTMR1_ALT2	GPIO_AON_14	ALT6
	LPTMR1_ALT3	GPIO_AON_15	ALT6
LPTMR2	LPTMR2_ALT1	GPIO_SD_B1_00	ALT3
	LPTMR2_ALT2	GPIO_SD_B1_01	ALT3
	LPTMR2_ALT3	GPIO_SD_B1_02	ALT3
LPTMR3	LPTMR3_ALT1	GPIO_SD_B1_03	ALT3
	LPTMR3_ALT2	GPIO_SD_B1_04	ALT3
	LPTMR3_ALT3	GPIO_SD_B1_05	ALT3
LPUART1	LPUART1_TXD	GPIO_AON_08	ALT0
	LPUART1_RXD	GPIO_AON_09	ALT0
	LPUART1_CTS_B	GPIO_AON_11	ALT2
	LPUART1_CTS_B	GPIO_AON_19	ALT4
	LPUART1_RTS_B	GPIO_AON_12	ALT2
	LPUART1_RTS_B	GPIO_AON_20	ALT4
	LPUART1_DCD_B	GPIO_AON_14	ALT3
	LPUART1_DCD_B	GPIO_AON_18	ALT4
	LPUART1_DSR_B	GPIO_AON_13	ALT3
	LPUART1_DSR_B	GPIO_AON_17	ALT4
	LPUART1_DTR_B	GPIO_AON_16	ALT3
	LPUART1_RI_B	GPIO_AON_15	ALT3
LPUART2	LPUART2_TXD	GPIO_AON_00	ALT6
	LPUART2_TXD	GPIO_AON_26	ALT2
	LPUART2_RXD	GPIO_AON_01	ALT6
	LPUART2_TXD	GPIO_AON_27	ALT2
	LPUART2_CTS_B	GPIO_AON_03	ALT6
	LPUART2_CTS_B	GPIO_AON_25	ALT2

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	LPUART2_RTS_B	GPIO_AON_02	ALT4
	LPUART2_RTS_B	GPIO_AON_24	ALT2
LPUART3	LPUART3_TXD	GPIO_EMC_B1_03	ALT3
	LPUART3_TXD	GPIO_AD_13	ALT6
	LPUART3_RXD	GPIO_EMC_B1_02	ALT3
	LPUART3_RXD	GPIO_AD_14	ALT6
	LPUART3_CTS_B	GPIO_EMC_B1_00	ALT3
	LPUART3_CTS_B	GPIO_AD_15	ALT6
	LPUART3_RTS_B	GPIO_EMC_B1_01	ALT3
	LPUART3_RTS_B	GPIO_AD_16	ALT6
	LPUART3_DCD_B	GPIO_EMC_B1_05	ALT3
	LPUART3_DSR_B	GPIO_EMC_B1_04	ALT3
	LPUART3_DTR_B	GPIO_EMC_B1_07	ALT3
	LPUART3_RI_B	GPIO_EMC_B1_06	ALT3
LPUART4	LPUART4_TXD	GPIO_EMC_B1_12	ALT2
	LPUART4_RXD	GPIO_EMC_B1_13	ALT2
	LPUART4_CTS_B	GPIO_EMC_B1_14	ALT10
	LPUART4_CTS_B	GPIO_EMC_B1_21	ALT9
	LPUART4_RTS_B	GPIO_EMC_B1_15	ALT10
	LPUART4_RTS_B	GPIO_EMC_B1_22	ALT9
	LPUART4_DCD_B	GPIO_EMC_B1_09	ALT3
	LPUART4_DSR_B	GPIO_EMC_B1_08	ALT3
	LPUART4_DTR_B	GPIO_EMC_B1_11	ALT3
	LPUART4_RI_B	GPIO_EMC_B1_10	ALT3
LPUART5	LPUART5_TXD	GPIO_EMC_B1_14	ALT2
	LPUART5_TXD	GPIO_EMC_B2_35	ALT2
	LPUART5_TXD	GPIO_EMC_B2_17	ALT2
	LPUART5_TXD	GPIO_AD_26	ALT1
	LPUART5_TXD	GPIO_EMC_B2_08	ALT6
	LPUART5_RXD	GPIO_EMC_B1_15	ALT2
	LPUART5_RXD	GPIO_EMC_B1_36	ALT2
	LPUART5_RXD	GPIO_EMC_B2_18	ALT2

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	LPUART5_RXD	GPIO_AD_27	ALT1
	LPUART5_RXD	GPIO_EMC_B2_09	ALT6
	LPUART5_CTS_B	GPIO_EMC_B1_37	ALT2
	LPUART5_CTS_B	GPIO_EMC_B2_19	ALT2
	LPUART5_CTS_B	GPIO_SD_B2_06	ALT6
	LPUART5_RTS_B	GPIO_EMC_B1_38	ALT2
	LPUART5_RTS_B	GPIO_EMC_B2_20	ALT2
	LPUART5_RTS_B	GPIO_SD_B2_07	ALT6
	LPUART5_DCD_B	GPIO_EMC_B2_15	ALT4
	LPUART5_DCD_B	GPIO_SD_B2_10	ALT6
	LPUART5_DSR_B	GPIO_EMC_B2_14	ALT4
	LPUART5_DSR_B	GPIO_SD_B2_11	ALT6
	LPUART5_DTR_B	GPIO_EMC_B2_16	ALT4
	LPUART5_DTR_B	GPIO_SD_B2_05	ALT6
	LPUART5_RI_B	GPIO_AD_18	ALT2
	LPUART5_RI_B	GPIO_SD_B2_04	ALT6
LPUART6	LPUART6_TXD	GPIO_EMC_B1_31	ALT2
	LPUART6_TXD	GPIO_AD_24	ALT0
	LPUART6_TXD	GPIO_B2_10	ALT4
	LPUART6_RXD	GPIO_EMC_B1_32	ALT2
	LPUART6_RXD	GPIO_AD_25	ALT0
	LPUART6_RXD	GPIO_B2_11	ALT4
	LPUART6_CTS_B	GPIO_EMC_B1_33	ALT2
	LPUART6_CTS_B	GPIO_AD_26	ALT0
	LPUART6_CTS_B	GPIO_B2_12	ALT4
	LPUART6_RTS_B	GPIO_EMC_B1_34	ALT2
	LPUART6_RTS_B	GPIO_AD_27	ALT0
	LPUART6_RTS_B	GPIO_B2_13	ALT4
	LPUART6_DCD_B	GPIO_EMC_B1_29	ALT9
	LPUART6_DCD_B	GPIO_B2_07	ALT3
	LPUART6_DSR_B	GPIO_EMC_B1_30	ALT9
	LPUART6_DSR_B	GPIO_B2_06	ALT3

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	LPUART6_DTR_B	GPIO_EMC_B1_28	ALT9
	LPUART6_DTR_B	GPIO_B2_09	ALT4
	LPUART6_RI_B	GPIO_EMC_B1_27	ALT9
	LPUART6_RI_B	GPIO_B2_08	ALT4
LPUART7	LPUART7_TXD	GPIO_AON_17	ALT2
	LPUART7_TXD	GPIO_AON_22	ALT2
	LPUART7_RXD	GPIO_AON_18	ALT2
	LPUART7_RXD	GPIO_AON_23	ALT2
	LPUART7_CTS_B	GPIO_AON_04	ALT6
	LPUART7_CTS_B	GPIO_AON_16	ALT8
	LPUART7_CTS_B	GPIO_AON_24	ALT3
	LPUART7_RTS_B	GPIO_AON_05	ALT6
	LPUART7_RTS_B	GPIO_AON_19	ALT8
	LPUART7_RTS_B	GPIO_AON_25	ALT3
LPUART8	LPUART8_TXD	GPIO_AD_30	ALT4
	LPUART8_TXD	GPIO_SD_B2_00	ALT9
	LPUART8_TXD	GPIO_B2_12	ALT2
	LPUART8_RXD	GPIO_AD_31	ALT4
	LPUART8_RXD	GPIO_SD_B2_01	ALT9
	LPUART8_RXD	GPIO_B2_13	ALT2
	LPUART8_CTS_B	GPIO_SD_B2_02	ALT9
	LPUART8_CTS_B	GPIO_B2_10	ALT3
	LPUART8_RTS_B	GPIO_SD_B2_03	ALT9
	LPUART8_RTS_B	GPIO_B2_10	ALT3
LPUART9	LPUART9_TXD	GPIO_EMC_B1_16	ALT2
	LPUART9_TXD	GPIO_B1_06	ALT2
	LPUART9_RXD	GPIO_EMC_B1_17	ALT2
	LPUART9_RXD	GPIO_B1_04	ALT2
	LPUART9_CTS_B	GPIO_B1_05	ALT2
	LPUART9_RTS_B	GPIO_B1_07	ALT2
LPUART10	LPUART10_TXD	GPIO_AD_15	ALT1
	LPUART10_TXD	GPIO_AD_32	ALT8

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	LPUART10_RXD	GPIO_AD_16	ALT1
	LPUART10_RXD	GPIO_AD_33	ALT8
	LPUART10_CTS_B	GPIO_AD_34	ALT8
	LPUART10_RTS_B	GPIO_AD_35	ALT8
LPUART11	LPUART11_TXD	GPIO_EMC_B2_13	ALT2
	LPUART11_TXD	GPIO_B1_02	ALT9
	LPUART11_TXD	GPIO_B2_06	ALT9
	LPUART11_RXD	GPIO_EMC_B2_14	ALT2
	LPUART11_RXD	GPIO_B1_03	ALT9
	LPUART11_RXD	GPIO_B2_07	ALT9
	LPUART11_CTS_B	GPIO_EMC_B2_15	ALT2
	LPUART11_RTS_B	GPIO_EMC_B2_16	ALT2
LPUART12	LPUART12_TXD	GPIO_AON_15	ALT2
	LPUART12_TXD	GPIO_AON_19	ALT9
	LPUART12_RXD	GPIO_AON_16	ALT2
	LPUART12_RXD	GPIO_AON_20	ALT9
	LPUART12_CTS_B	GPIO_AON_13	ALT2
	LPUART12_CTS_B	GPIO_AON_22	ALT3
	LPUART12_RTS_B	GPIO_AON_14	ALT2
	LPUART12_RTS_B	GPIO_AON_23	ALT3
MIC	MIC_CLK	GPIO_AD_00	ALT2
	MIC_BITSTREAM0	GPIO_AD_01	ALT2
	MIC_BITSTREAM1	GPIO_AD_02	ALT2
	MIC_BITSTREAM2	GPIO_AD_03	ALT2
	MIC_BITSTREAM3	GPIO_AD_04	ALT2
	MIC_BITSTREAM2	GPIO_AD_26	ALT12
	MIC_CLK	GPIO_AD_27	ALT12
	MIC_BITSTREAM0	GPIO_AD_28	ALT12
	MIC_BITSTREAM1	GPIO_AD_29	ALT12
	MIC_BITSTREAM3	GPIO_AD_32	ALT12
	MIC_BITSTREAM0	GPIO_SD_B2_00	ALT12
	MIC_BITSTREAM1	GPIO_SD_B2_01	ALT12

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	MIC_BITSTREAM2	GPIO_SD_B2_02	ALT12
	MIC_BITSTREAM3	GPIO_SD_B2_03	ALT12
	MIC_CLK	GPIO_SD_B2_04	ALT12
	MIC_CLK	GPIO_B2_05	ALT6
	MIC_BITSTREAM0	GPIO_B2_10	ALT0
	MIC_BITSTREAM1	GPIO_B2_11	ALT0
	MIC_BITSTREAM2	GPIO_B2_12	ALT0
	MIC_BITSTREAM3	GPIO_B2_13	ALT0
NETC	NETC_MDC	GPIO_EMC_B1_18	ALT10
	NETC_MDIO_DATA	GPIO_EMC_B1_19	ALT10
	NETC_MDC	GPIO_EMC_B1_40	ALT1
	NETC_MDIO_DATA	GPIO_EMC_B1_41	ALT1
	NETC_MDC	GPIO_EMC_B2_00	ALT3
	NETC_MDIO_DATA	GPIO_EMC_B2_01	ALT3
	NETC_MDC	GPIO_EMC_B2_19	ALT4
	NETC_MDIO_DATA	GPIO_EMC_B2_20	ALT4
	NETC_MDC	GPIO_AD_30	ALT7
	NETC_MDIO_DATA	GPIO_AD_31	ALT7
	NETC_MDIO_DATA	GPIO_SD_B2_10	ALT10
	NETC_MDC	GPIO_SD_B2_11	ALT10
	NETC_MDIO_DATA	GPIO_B1_12	ALT1
	NETC_MDC	GPIO_B1_13	ALT1
	NETC_MDIO_DATA	GPIO_B2_02	ALT3
	NETC_MDC	GPIO_B2_03	ALT3
	NETC_TMR_TRIG1	GPIO_AD_20	ALT6
		GPIO_AD_24	ALT7
		GPIO_AD_32	ALT7
		GPIO_SD_B2_11	ALT8
	NETC_TMR_TRIG2	GPIO_AD_21	ALT6
		GPIO_AD_25	ALT7
		GPIO_AD_33	ALT7
		GPIO_SD_B2_10	ALT8

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	NETC_1588_CLK	GPIO_AD_20	ALT7
		GPIO_SD_B2_00	ALT8
	NETC_TMR_GCLK	GPIO_AD_21	ALT7
		GPIO_SD_B2_01	ALT8
	NETC_TMR_PP1	GPIO_AD_26	ALT7
		GPIO_SD_B2_04	ALT8
		GPIO_SD_B2_09	ALT9
	NETC_TMR_PP2	GPIO_AD_27	ALT7
		GPIO_SD_B2_05	ALT8
		GPIO_SD_B2_08	ALT9
	NETC_TMR_PP3	GPIO_AD_28	ALT7
		GPIO_SD_B2_06	ALT8
		GPIO_SD_B2_10	ALT9
	NETC_TMR_ALARM1	GPIO_AD_22	ALT7
		GPIO_AD_34	ALT7
		GPIO_SD_B2_02	ALT8
		GPIO_SD_B2_07	ALT8
	NETC_TMR_ALARM2	GPIO_AD_23	ALT7
		GPIO_AD_35	ALT7
		GPIO_SD_B2_03	ALT8
		GPIO_SD_B2_08	ALT8
PMIC	PMIC_ON_REQ	PMIC_ON_REQ	ALT0
	PMIC_STBY_REQ	PMIC_STBY_REQ	ALT0
QTIMER1	QTIMER1_TIMER1	GPIO_EMC_B1_13	ALT10
	QTIMER1_TIMER0	GPIO_EMC_B1_18	ALT2
	QTIMER1_TIMER0	GPIO_EMC_B2_09	ALT10
	QTIMER1_TIMER1	GPIO_EMC_B2_10	ALT10
	QTIMER1_TIMER2	GPIO_EMC_B2_11	ALT10
	QTIMER1_TIMER3	GPIO_EMC_B2_12	ALT10
	QTIMER1_TIMER0	GPIO_B1_00	ALT3
	QTIMER1_TIMER1	GPIO_B1_01	ALT3
	QTIMER1_TIMER2	GPIO_B1_02	ALT3

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
QTIMER2	QTIMER2_TIMER0	GPIO_EMC_B1_19	ALT2
	QTIMER2_TIMER1	GPIO_EMC_B1_39	ALT6
	QTIMER2_TIMER0	GPIO_EMC_B2_13	ALT10
	QTIMER2_TIMER1	GPIO_EMC_B2_14	ALT10
	QTIMER2_TIMER2	GPIO_EMC_B2_15	ALT10
	QTIMER2_TIMER3	GPIO_EMC_B2_16	ALT10
	QTIMER2_TIMER0	GPIO_B1_03	ALT3
	QTIMER2_TIMER1	GPIO_B1_04	ALT3
	QTIMER2_TIMER2	GPIO_B1_05	ALT3
QTIMER3	QTIMER3_TIMER0	GPIO_EMC_B1_20	ALT2
	QTIMER3_TIMER1	GPIO_EMC_B1_40	ALT6
	QTIMER3_TIMER0	GPIO_EMC_B2_17	ALT10
	QTIMER3_TIMER1	GPIO_EMC_B2_18	ALT10
	QTIMER3_TIMER2	GPIO_EMC_B2_19	ALT10
	QTIMER3_TIMER3	GPIO_EMC_B2_20	ALT10
	QTIMER3_TIMER0	GPIO_B1_06	ALT3
	QTIMER3_TIMER1	GPIO_B1_07	ALT3
	QTIMER3_TIMER2	GPIO_B1_08	ALT3
QTIMER4	QTIMER4_TIMER0	GPIO_EMC_B1_21	ALT2
	QTIMER4_TIMER1	GPIO_EMC_B1_41	ALT6
	QTIMER4_TIMER0	GPIO_AD_00	ALT9
	QTIMER4_TIMER1	GPIO_AD_01	ALT9
	QTIMER4_TIMER2	GPIO_AD_02	ALT9
	QTIMER4_TIMER3	GPIO_AD_03	ALT9
	QTIMER4_TIMER0	GPIO_B1_09	ALT3
	QTIMER4_TIMER1	GPIO_B1_10	ALT3
	QTIMER4_TIMER2	GPIO_B1_11	ALT3
QTIMER5	QTIMER5_TIMER0	GPIO_EMC_B1_22	ALT2
	QTIMER5_TIMER1	GPIO_EMC_B2_00	ALT2
	QTIMER5_TIMER0	GPIO_AD_04	ALT9
	QTIMER5_TIMER1	GPIO_AD_05	ALT9
	QTIMER5_TIMER2	GPIO_AD_06	ALT9

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	QTIMER5_TIMER3	GPIO_AD_07	ALT9
	QTIMER5_TIMER0	GPIO_B2_07	ALT0
	QTIMER5_TIMER1	GPIO_B2_08	ALT0
	QTIMER5_TIMER2	GPIO_B2_09	ALT0
QTIMER6	QTIMER6_TIMER0	GPIO_EMC_B1_23	ALT2
	QTIMER6_TIMER1	GPIO_EMC_B2_01	ALT2
	QTIMER6_TIMER0	GPIO_AD_08	ALT9
	QTIMER6_TIMER1	GPIO_AD_09	ALT9
	QTIMER6_TIMER2	GPIO_AD_10	ALT9
	QTIMER6_TIMER3	GPIO_AD_11	ALT9
	QTIMER6_TIMER0	GPIO_SD_B2_01	ALT3
	QTIMER6_TIMER1	GPIO_SD_B2_02	ALT3
	QTIMER6_TIMER2	GPIO_SD_B2_03	ALT3
QTIMER7	QTIMER7_TIMER0	GPIO_EMC_B1_24	ALT2
	QTIMER7_TIMER1	GPIO_EMC_B2_02	ALT2
	QTIMER7_TIMER0	GPIO_SD_B2_04	ALT3
	QTIMER7_TIMER1	GPIO_SD_B2_05	ALT3
	QTIMER7_TIMER2	GPIO_SD_B2_06	ALT3
	QTIMER7_TIMER3	GPIO_SD_B2_07	ALT3
QTIMER8	QTIMER8_TIMER0	GPIO_EMC_B1_25	ALT2
	QTIMER8_TIMER1	GPIO_EMC_B2_03	ALT2
	QTIMER8_TIMER0	GPIO_SD_B2_08	ALT3
	QTIMER8_TIMER1	GPIO_SD_B2_09	ALT3
	QTIMER8_TIMER2	GPIO_SD_B2_10	ALT3
	QTIMER8_TIMER3	GPIO_SD_B2_11	ALT3
SAI1	SAI1_TX_DATA0	GPIO_AON_04	ALT2
	SAI1_RX_DATA1	GPIO_AON_04	ALT3
	SAI1_TX_SYNC	GPIO_AON_05	ALT2
	SAI1_TX_BCLK	GPIO_AON_06	ALT2
	SAI1_MCLK	GPIO_AON_07	ALT2
	SAI1_RX_DATA0	GPIO_AON_08	ALT2
	SAI1_TX_DATA1	GPIO_AON_08	ALT3

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	SAI1_RX_BCLK	GPIO_AON_09	ALT2
	SAI1_RX_SYNC	GPIO_AON_10	ALT2
	SAI1_TX_DATA0	GPIO_AON_21	ALT4
	SAI1_RX_DATA1	GPIO_AON_21	ALT9
	SAI1_TX_SYNC	GPIO_AON_22	ALT4
	SAI1_TX_BCLK	GPIO_AON_23	ALT4
	SAI1_MCLK	GPIO_AON_24	ALT4
	SAI1_RX_DATA0	GPIO_AON_25	ALT4
	SAI1_TX_DATA1	GPIO_AON_25	ALT7
	SAI1_RX_BCLK	GPIO_AON_26	ALT4
	SAI1_RX_SYNC	GPIO_AON_27	ALT4
SAI2	SAI2_MCLK	GPIO_EMC_B2_04	ALT2
	SAI2_RX_SYNC	GPIO_EMC_B2_05	ALT2
	SAI2_RX_BCLK	GPIO_EMC_B2_06	ALT2
	SAI2_RX_DATA	GPIO_EMC_B2_07	ALT2
	SAI2_TX_DATA	GPIO_EMC_B2_08	ALT2
	SAI2_TX_BCLK	GPIO_EMC_B2_09	ALT2
	SAI2_TX_SYNC	GPIO_EMC_B2_10	ALT2
SAI3	SAI3_RX_SYNC	GPIO_EMC_B2_11	ALT8
	SAI3_RX_BCLK	GPIO_EMC_B2_12	ALT8
	SAI3_RX_DATA	GPIO_EMC_B2_13	ALT8
	SAI3_TX_DATA	GPIO_EMC_B2_14	ALT8
	SAI3_TX_BCLK	GPIO_EMC_B2_15	ALT8
	SAI3_TX_SYNC	GPIO_EMC_B2_16	ALT8
	SAI3_MCLK	GPIO_EMC_B2_17	ALT8
SAI4	SAI4_MCLK	GPIO_AD_17	ALT0
	SAI4_RX_SYNC	GPIO_AD_18	ALT0
	SAI4_RX_BCLK	GPIO_AD_19	ALT0
	SAI4_RX_DATA0	GPIO_AD_20	ALT0
	SAI4_TX_DATA0	GPIO_AD_21	ALT0
	SAI4_TX_BCLK	GPIO_AD_22	ALT0
	SAI4_TX_SYNC	GPIO_AD_23	ALT0

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	SAI4_RX_DATA0	GPIO_B1_01	ALT12
	SAI4_RX_DATA1	GPIO_B1_02	ALT12
	SAI4_RX_DATA2	GPIO_B1_03	ALT12
	SAI4_RX_DATA3	GPIO_B1_04	ALT12
	SAI4_MCLK	GPIO_B1_05	ALT12
	SAI4_RX_BCLK	GPIO_B1_06	ALT12
	SAI4_RX_SYNC	GPIO_B1_07	ALT12
	SAI4_TX_BCLK	GPIO_B1_08	ALT12
	SAI4_TX_SYNC	GPIO_B1_09	ALT12
	SAI4_TX_DATA0	GPIO_B1_10	ALT12
	SAI4_TX_DATA1	GPIO_B1_11	ALT12
	SAI4_TX_DATA2	GPIO_B1_12	ALT12
	SAI4_TX_DATA3	GPIO_B1_13	ALT12
	SAI4_MCLK	GPIO_B2_00	ALT4
	SAI4_TX_BCLK	GPIO_B2_01	ALT4
	SAI4_TX_SYNC	GPIO_B2_02	ALT4
	SAI4_TX_DATA0	GPIO_B2_03	ALT4
	SAI4_RX_SYNC	GPIO_B2_04	ALT4
	SAI4_RX_BCLK	GPIO_B2_05	ALT4
	SAI4_RX_DATA0	GPIO_B2_06	ALT4
	SAI4_TX_DATA1	GPIO_B2_07	ALT4
	SAI4_RX_DATA1	GPIO_B2_07	ALT6
SECO	SECO_TX	GPIO_AON_08	ALT1
	SECO_RX	GPIO_AON_09	ALT1
SINC1	SINC1_MOD_CLK0	GPIO_AD_00	ALT6
	SINC1_MOD_CLK1	GPIO_AD_01	ALT6
	SINC1_MOD_CLK2	GPIO_AD_02	ALT6
	SINC1_EMCLK0	GPIO_AD_03	ALT6
	SINC1_EMBIT0	GPIO_AD_04	ALT6
	SINC1_EMCLK1	GPIO_AD_05	ALT6
	SINC1_EMBIT1	GPIO_AD_06	ALT6
	SINC1_EMCLK2	GPIO_AD_07	ALT6

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	SINC1_EMBIT2	GPIO_AD_08	ALT6
	SINC1_EMCLK3	GPIO_AD_09	ALT6
	SINC1_EMBIT3	GPIO_AD_10	ALT6
	SINC1_BREAK	GPIO_AD_11	ALT6
	SINC1_EMCLK0	GPIO_AD_20	ALT3
	SINC1_EMBIT0	GPIO_AD_21	ALT3
	SINC1_EMCLK1	GPIO_AD_22	ALT3
	SINC1_EMBIT1	GPIO_AD_23	ALT3
	SINC1_EMCLK2	GPIO_SD_B1_00	ALT1
	SINC1_EMBIT2	GPIO_SD_B1_01	ALT1
	SINC1_EMCLK3	GPIO_SD_B1_02	ALT1
	SINC1_EMBIT3	GPIO_SD_B1_03	ALT1
	SINC1_BREAK	GPIO_SD_B1_04	ALT1
	SINC1_MOD_CLK0	GPIO_B2_04	ALT0
	SINC1_MOD_CLK1	GPIO_B2_05	ALT0
	SINC1_MOD_CLK2	GPIO_B2_06	ALT0
SINC2	SINC2_MOD_CLK1	GPIO_AD_24	ALT3
	SINC2_MOD_CLK2	GPIO_AD_25	ALT3
	SINC2_EMCLK0	GPIO_AD_26	ALT3
	SINC2_EMBIT0	GPIO_AD_27	ALT3
	SINC2_EMCLK1	GPIO_AD_28	ALT3
	SINC2_EMBIT1	GPIO_AD_29	ALT3
	SINC2_EMCLK2	GPIO_AD_30	ALT3
	SINC2_EMBIT2	GPIO_AD_31	ALT3
	SINC2_EMCLK3	GPIO_AD_32	ALT3
	SINC2_EMBIT3	GPIO_AD_33	ALT3
	SINC2_BREAK	GPIO_AD_34	ALT3
	SINC2_MOD_CLK0	GPIO_AD_35	ALT3
	SINC2_EMCLK2	GPIO_SD_B1_04	ALT2
	SINC2_EMBIT2	GPIO_SD_B1_05	ALT2
	SINC2_MOD_CLK0	GPIO_B2_04	ALT1
	SINC2_MOD_CLK1	GPIO_B2_05	ALT1

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	SINC2_MOD_CLK2	GPIO_B2_06	ALT1
	SINC2_EMCLK2	GPIO_B2_08	ALT1
	SINC2_EMBIT2	GPIO_B2_09	ALT1
	SINC2_EMCLK3	GPIO_B2_10	ALT1
	SINC2_EMBIT3	GPIO_B2_11	ALT1
	SINC2_BREAK	GPIO_B2_12	ALT1
	SINC2_EMCLK0	GPIO_B2_13	ALT1
SINC3	SINC3_MOD_CLK0	GPIO_EMC_B1_00	ALT2
	SINC3_MOD_CLK1	GPIO_EMC_B1_01	ALT2
	SINC3_MOD_CLK2	GPIO_EMC_B1_02	ALT2
	SINC3_EMCLK0	GPIO_EMC_B1_03	ALT2
	SINC3_EMBIT0	GPIO_EMC_B1_04	ALT2
	SINC3_EMCLK1	GPIO_EMC_B1_05	ALT2
	SINC3_EMBIT1	GPIO_EMC_B1_06	ALT2
	SINC3_EMCLK2	GPIO_EMC_B1_07	ALT2
	SINC3_EMBIT2	GPIO_EMC_B1_08	ALT2
	SINC3_EMCLK3	GPIO_EMC_B1_09	ALT2
	SINC3_EMBIT3	GPIO_EMC_B1_10	ALT2
	SINC3_BREAK	GPIO_EMC_B1_11	ALT2
	SINC3_MOD_CLK0	GPIO_B2_04	ALT2
	SINC3_MOD_CLK1	GPIO_B2_05	ALT2
	SINC3_MOD_CLK2	GPIO_B2_06	ALT2
SPDIF	SPDIF_OUT	GPIO_EMC_B2_11	ALT2
	SPDIF_IN	GPIO_EMC_B2_12	ALT2
	SPDIF_LOCK	GPIO_AD_12	ALT0
	SPDIF_SR_CLK	GPIO_AD_13	ALT0
	SPDIF_EXT_CLK	GPIO_AD_14	ALT0
	SPDIF_IN	GPIO_AD_15	ALT0
	SPDIF_OUT	GPIO_AD_16	ALT0
	SPDIF_IN	GPIO_B2_08	ALT9
	SPDIF_OUT	GPIO_B2_09	ALT9
SRAMC	SRAMC_DATA_00	GPIO_EMC_B1_00	ALT12

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	SRAMC_DATA_01	GPIO_EMC_B1_01	ALT12
	SRAMC_DATA_02	GPIO_EMC_B1_02	ALT12
	SRAMC_DATA_03	GPIO_EMC_B1_03	ALT12
	SRAMC_DATA_04	GPIO_EMC_B1_04	ALT12
	SRAMC_DATA_05	GPIO_EMC_B1_05	ALT12
	SRAMC_DATA_06	GPIO_EMC_B1_06	ALT12
	SRAMC_DATA_07	GPIO_EMC_B1_07	ALT12
	SRAMC_LBB	GPIO_EMC_B1_08	ALT12
	SRAMC_ADDR_00	GPIO_EMC_B1_09	ALT12
	SRAMC_ADDR_01	GPIO_EMC_B1_10	ALT12
	SRAMC_ADDR_02	GPIO_EMC_B1_11	ALT12
	SRAMC_ADDR_03	GPIO_EMC_B1_12	ALT12
	SRAMC_ADDR_04	GPIO_EMC_B1_13	ALT12
	SRAMC_ADDR_05	GPIO_EMC_B1_14	ALT12
	SRAMC_ADDR_06	GPIO_EMC_B1_15	ALT12
	SRAMC_ADDR_07	GPIO_EMC_B1_16	ALT12
	SRAMC_ADDR_08	GPIO_EMC_B1_17	ALT12
	SRAMC_ADDR_09	GPIO_EMC_B1_18	ALT12
	SRAMC_ADDR_11	GPIO_EMC_B1_19	ALT12
	SRAMC_ADDR_12	GPIO_EMC_B1_20	ALT12
	SRAMC_ADDR_13	GPIO_EMC_B1_21	ALT12
	SRAMC_ADDR_14	GPIO_EMC_B1_22	ALT12
	SRAMC_ADDR_10	GPIO_EMC_B1_23	ALT12
	SRAMC_ADDR_15	GPIO_EMC_B1_24	ALT12
	SRAMC_ADDR_16	GPIO_EMC_B1_25	ALT12
	SRAMC_WE	GPIO_EMC_B1_26	ALT12
	SRAMC_OEB	GPIO_EMC_B1_27	ALT12
	SRAMC_ADV	GPIO_EMC_B1_28	ALT12
	SRAMC_CS0	GPIO_EMC_B1_29	ALT12
	SRAMC_DATA_08	GPIO_EMC_B1_30	ALT12
	SRAMC_DATA_09	GPIO_EMC_B1_31	ALT12
	SRAMC_DATA_10	GPIO_EMC_B1_32	ALT12

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	SRAMC_DATA_11	GPIO_EMC_B1_33	ALT12
	SRAMC_DATA_12	GPIO_EMC_B1_34	ALT12
	SRAMC_DATA_13	GPIO_EMC_B1_35	ALT12
	SRAMC_DATA_14	GPIO_EMC_B1_36	ALT12
	SRAMC_DATA_15	GPIO_EMC_B1_37	ALT12
	SRAMC_UBB	GPIO_EMC_B1_38	ALT12
	SRAMC_CS1	GPIO_EMC_B1_39	ALT12
	SRAMC_CS2	GPIO_EMC_B1_40	ALT12
	SRAMC_CS3	GPIO_EMC_B1_41	ALT12
SRC	SRC_BOOT_MODE0	GPIO_AON_00	ALT0
	SRC_BOOT_MODE1	GPIO_AON_01	ALT0
	SRC_BOOT_MODE2	GPIO_AON_02	ALT0
TAMPER	TAMPER0	GPIO_BBSM_00	ALT0
	TAMPER1	GPIO_BBSM_01	ALT0
	TAMPER2	GPIO_BBSM_02	ALT0
	TAMPER3	GPIO_BBSM_03	ALT0
	TAMPER4	GPIO_BBSM_04	ALT0
	TAMPER5	GPIO_BBSM_05	ALT0
	TAMPER6	GPIO_BBSM_06	ALT0
	TAMPER7	GPIO_BBSM_07	ALT0
	TAMPER8	GPIO_BBSM_08	ALT0
	TAMPER9	GPIO_BBSM_09	ALT0
TPM1	TPM1_EXTCLK	GPIO_AON_00	ALT8
	TPM1_CH0	GPIO_AON_01	ALT8
	TPM1_CH1	GPIO_AON_02	ALT8
	TPM1_CH2	GPIO_AON_03	ALT8
	TPM1_CH3	GPIO_AON_04	ALT8
TPM2	TPM2_EXTCLK	GPIO_AON_10	ALT4
	TPM2_CH0	GPIO_AON_11	ALT4
	TPM2_CH1	GPIO_AON_12	ALT4
	TPM2_CH2	GPIO_AON_13	ALT4
	TPM2_CH3	GPIO_AON_14	ALT4

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
TPM3	TPM3_EXTCLK	GPIO_B2_04	ALT8
	TPM3_CH0	GPIO_B2_05	ALT8
	TPM3_CH1	GPIO_B2_06	ALT8
	TPM3_CH2	GPIO_B2_07	ALT8
	TPM3_CH3	GPIO_B2_08	ALT8
TPM4	TPM4_EXTCLK	GPIO_B2_09	ALT8
	TPM4_CH0	GPIO_B2_10	ALT8
	TPM4_CH1	GPIO_B2_11	ALT8
	TPM4_CH2	GPIO_B2_12	ALT8
	TPM4_CH3	GPIO_B2_13	ALT8
TPM5	TPM5_CH0	GPIO_B1_00	ALT6
	TPM5_CH1	GPIO_B1_01	ALT6
	TPM5_CH2	GPIO_B1_02	ALT6
	TPM5_CH3	GPIO_B1_03	ALT6
	TPM5_EXTCLK	GPIO_B1_04	ALT6
TPM6	TPM6_EXTCLK	GPIO_B1_05	ALT6
	TPM6_CH0	GPIO_B1_06	ALT6
	TPM6_CH1	GPIO_B1_07	ALT6
	TPM6_CH2	GPIO_B1_08	ALT6
	TPM6_CH3	GPIO_B1_09	ALT6
USB_OTG1	USB_OTG1_ID	GPIO_AD_09	ALT0
	USB_OTG1_PWR	GPIO_AD_10	ALT0
	USB_OTG1_OC	GPIO_AD_11	ALT0
	USB_OTG1_ID	GPIO_AD_33	ALT1
	USB_OTG1_PWR	GPIO_AD_34	ALT1
	USB_OTG1_OC	GPIO_AD_35	ALT1
USB_OTG2	USB_OTG2_OC	GPIO_AD_06	ALT0
	USB_OTG2_PWR	GPIO_AD_07	ALT0
	USB_OTG2_ID	GPIO_AD_08	ALT0
	USB_OTG2_OC	GPIO_AD_30	ALT1
	USB_OTG2_PWR	GPIO_AD_31	ALT1
	USB_OTG2_ID	GPIO_AD_32	ALT1

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
USDHC1	USDHC1_CD_B	GPIO_AD_32	ALT4
	USDHC1_WP	GPIO_AD_33	ALT4
	USDHC1_VSELECT	GPIO_AD_34	ALT4
	USDHC1_RESET_B	GPIO_AD_35	ALT4
	USDHC1_CMD	GPIO_SD_B1_00	ALT0
	USDHC1_CLK	GPIO_SD_B1_01	ALT0
	USDHC1_DATA0	GPIO_SD_B1_02	ALT0
	USDHC1_DATA1	GPIO_SD_B1_03	ALT0
	USDHC1_DATA2	GPIO_SD_B1_04	ALT0
	USDHC1_DATA3	GPIO_SD_B1_05	ALT0
	USDHC1_CD_B	GPIO_B1_08	ALT2
	USDHC1_WP	GPIO_B1_09	ALT2
	USDHC1_RESET_B	GPIO_B1_10	ALT2
	USDHC1_VSELECT	GPIO_B1_13	ALT2
USDHC2	USDHC2_CD_B	GPIO_EMC_B2_01	ALT1
	USDHC2_WP	GPIO_EMC_B2_02	ALT1
	USDHC2_VSELECT	GPIO_EMC_B2_03	ALT1
	USDHC2_RESET_B	GPIO_EMC_B2_04	ALT1
	USDHC2_CD_B	GPIO_AD_13	ALT7
	USDHC2_WP	GPIO_AD_14	ALT7
	USDHC2_CD_B	GPIO_AD_26	ALT9
	USDHC2_WP	GPIO_AD_27	ALT9
	USDHC2_RESET_B	GPIO_AD_28	ALT9
	USDHC2_VSELECT	GPIO_AD_29	ALT9
	USDHC2_CD_B	GPIO_AD_29	ALT2
	USDHC2_DATA3	GPIO_SD_B2_00	ALT0
	USDHC2_DATA2	GPIO_SD_B2_01	ALT0
	USDHC2_DATA1	GPIO_SD_B2_02	ALT0
	USDHC2_DATA0	GPIO_SD_B2_03	ALT0
	USDHC2_CLK	GPIO_SD_B2_04	ALT0
	USDHC2_CMD	GPIO_SD_B2_05	ALT0
	USDHC2_RESET_B	GPIO_SD_B2_06	ALT0

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	USDHC2_STROBE	GPIO_SD_B2_07	ALT0
	USDHC2_DATA4	GPIO_SD_B2_08	ALT0
	USDHC2_DATA5	GPIO_SD_B2_09	ALT0
	USDHC2_DATA6	GPIO_SD_B2_10	ALT0
	USDHC2_DATA7	GPIO_SD_B2_11	ALT0
XBAR	XBAR_INOUT04	GPIO_EMC_B1_00	ALT1
	XBAR_INOUT05	GPIO_EMC_B1_01	ALT1
	XBAR_INOUT06	GPIO_EMC_B1_02	ALT1
	XBAR_INOUT07	GPIO_EMC_B1_03	ALT1
	XBAR_INOUT08	GPIO_EMC_B1_04	ALT1
	XBAR_INOUT09	GPIO_EMC_B1_05	ALT1
	XBAR_INOUT10	GPIO_EMC_B1_26	ALT2
	XBAR_INOUT11	GPIO_EMC_B1_27	ALT2
	XBAR_INOUT12	GPIO_EMC_B1_28	ALT2
	XBAR_INOUT13	GPIO_EMC_B1_29	ALT2
	XBAR_INOUT14	GPIO_EMC_B1_30	ALT2
	XBAR_INOUT15	GPIO_EMC_B1_39	ALT2
	XBAR_INOUT20	GPIO_EMC_B2_00	ALT6
	XBAR_INOUT21	GPIO_EMC_B2_01	ALT6
	XBAR_INOUT22	GPIO_EMC_B2_02	ALT6
	XBAR_INOUT23	GPIO_EMC_B2_03	ALT6
	XBAR_INOUT24	GPIO_EMC_B2_04	ALT6
	XBAR_INOUT25	GPIO_EMC_B2_05	ALT6
	XBAR_INOUT26	GPIO_EMC_B2_06	ALT6
	XBAR_INOUT27	GPIO_EMC_B2_07	ALT6
	XBAR_INOUT28	GPIO_EMC_B2_08	ALT6
	XBAR_INOUT29	GPIO_EMC_B2_09	ALT6
	XBAR_INOUT30	GPIO_EMC_B2_10	ALT6
	XBAR_INOUT31	GPIO_EMC_B2_11	ALT6
	XBAR_INOUT32	GPIO_EMC_B2_12	ALT6
	XBAR_INOUT33	GPIO_EMC_B2_13	ALT6
	XBAR_INOUT34	GPIO_EMC_B2_14	ALT6

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	XBAR_INOUT35	GPIO_EMC_B2_15	ALT6
	XBAR_INOUT14	GPIO_EMC_B2_16	ALT6
	XBAR_INOUT15	GPIO_EMC_B2_17	ALT6
	XBAR_INOUT16	GPIO_EMC_B2_18	ALT6
	XBAR_INOUT36	GPIO_EMC_B2_19	ALT6
	XBAR_INOUT37	GPIO_EMC_B2_20	ALT6
	XBAR_INOUT18	GPIO_AD_12	ALT6
	XBAR_INOUT19	GPIO_AD_19	ALT2
	XBAR_INOUT24	GPIO_AD_30	ALT9
	XBAR_INOUT25	GPIO_AD_31	ALT9
	XBAR_INOUT17	GPIO_AD_33	ALT2
	XBAR_INOUT18	GPIO_AD_34	ALT2
	XBAR_INOUT19	GPIO_AD_35	ALT2
	XBAR_INOUT20	GPIO_SD_B1_00	ALT2
	XBAR_INOUT21	GPIO_SD_B1_01	ALT2
	XBAR_INOUT22	GPIO_SD_B1_02	ALT2
	XBAR_INOUT23	GPIO_SD_B1_03	ALT2
	XBAR_INOUT17	GPIO_SD_B2_00	ALT3
	XBAR_INOUT26	GPIO_B1_00	ALT4
	XBAR_INOUT27	GPIO_B1_01	ALT4
	XBAR_INOUT28	GPIO_B1_02	ALT4
	XBAR_INOUT29	GPIO_B1_03	ALT4
	XBAR_INOUT30	GPIO_B1_04	ALT4
	XBAR_INOUT31	GPIO_B1_05	ALT4
	XBAR_INOUT32	GPIO_B1_06	ALT4
	XBAR_INOUT33	GPIO_B1_07	ALT4
	XBAR_INOUT36	GPIO_B1_08	ALT4
	XBAR_INOUT37	GPIO_B1_09	ALT4
	XBAR_INOUT34	GPIO_B1_10	ALT4
	XBAR_INOUT35	GPIO_B1_11	ALT4
XSPI	XSPI_SLV_CS	GPIO_SD_B1_00	ALT4
	XSPI_SLV_CLK	GPIO_SD_B1_01	ALT4

Table continues on the next page...

Table 116. Muxing Options (continued)

Instance	Port(logical pin names)	Pad	Mode (alternate functions)
	XSPI_SLV_DATA4	GPIO_SD_B1_02	ALT4
	XSPI_SLV_DATA5	GPIO_SD_B1_03	ALT4
	XSPI_SLV_DATA6	GPIO_SD_B1_04	ALT4
	XSPI_SLV_DATA7	GPIO_SD_B1_05	ALT4
	XSPI_SLV_DATA4	GPIO_SD_B2_00	ALT2
	XSPI_SLV_DATA5	GPIO_SD_B2_01	ALT2
	XSPI_SLV_DATA6	GPIO_SD_B2_02	ALT2
	XSPI_SLV_DATA7	GPIO_SD_B2_03	ALT2
	XSPI_SLV_DQS	GPIO_SD_B2_05	ALT2
	XSPI_SLV_CS	GPIO_SD_B2_06	ALT2
	XSPI_SLV_CLK	GPIO_SD_B2_07	ALT2
	XSPI_SLV_DATA0	GPIO_SD_B2_08	ALT2
	XSPI_SLV_DATA1	GPIO_SD_B2_09	ALT2
	XSPI_SLV_DATA2	GPIO_SD_B2_10	ALT2
	XSPI_SLV_DATA3	GPIO_SD_B2_11	ALT2
	XSPI_SLV_DATA4	GPIO_B2_03	ALT10
	XSPI_SLV_DATA5	GPIO_B2_04	ALT10
	XSPI_SLV_DATA6	GPIO_B2_05	ALT10
	XSPI_SLV_DATA7	GPIO_B2_06	ALT10
	XSPI_SLV_DQS	GPIO_B2_07	ALT10
	XSPI_SLV_CLK	GPIO_B2_08	ALT10
	XSPI_SLV_CS	GPIO_B2_09	ALT10
	XSPI_SLV_DATA0	GPIO_B2_10	ALT10
	XSPI_SLV_DATA1	GPIO_B2_11	ALT10
	XSPI_SLV_DATA2	GPIO_B2_12	ALT10
	XSPI_SLV_DATA3	GPIO_B2_13	ALT10

13.1.2 Pin Assignments

NOTE

Unless otherwise noted, the pad settings are configured through SW_MUX_CTL_PAD_* IONMUXC/ IONMUXC_AON registers.

Table 117. Pin Assignments

Pin Name	Pin Assignments	Pad Settings
POR_B	ALT0 - POR_B	PUE PUS (Configurable through BBNSM PAD_CTRL)
ONOFF	ALT0 - ONOFF	PUE PUS (Configurable through BBNSM PAD_CTRL)
WAKEUP	ALT0 - WAKEUP	PUE PUS (Configurable through BBNSM PAD_CTRL)
PMIC_ON_REQ	ALT0 - PMIC_ON_REQ	No Pull (not configurable)
PMIC_STBY_REQ	ALT0 - PMIC_STBY_REQ	No Pull (not configurable)
GPIO_BBSM_00	ALT0 - TAMPER0	PUE PUS (Configurable through BBSM PAD_CTRL)
GPIO_BBSM_01	ALT0 - TAMPER1	PUE PUS (Configurable through BBSM PAD_CTRL)
GPIO_BBSM_02	ALT0 - TAMPER2	PUE PUS (Configurable through BBSM PAD_CTRL)
GPIO_BBSM_03	ALT0 - TAMPER3	PUE PUS (Configurable through BBSM PAD_CTRL)
GPIO_BBSM_04	ALT0 - TAMPER4	PUE PUS (Configurable through BBSM PAD_CTRL)

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
GPIO_BBSM_05	ALT0 - TAMPER5	PUE PUS (Configurable through BBSM PAD_CTRL)
GPIO_BBSM_06	ALT0 - TAMPER6	PUE PUS (Configurable through BBSM PAD_CTRL)
GPIO_BBSM_07	ALT0 - TAMPER7	PUE PUS (Configurable through BBSM PAD_CTRL)
GPIO_BBSM_08	ALT0 - TAMPER8	PUE PUS (Configurable through BBSM PAD_CTRL)
GPIO_BBSM_09	ALT0 - TAMPER9	PUE PUS (Configurable through BBSM PAD_CTRL)
GPIO_AON_00	ALT0 - SRC_BOOT_MODE0 ALT1 - FLEXCAN1_TX ALT4 - LPTMR1_ALT1 ALT5 - GPIO1_IO00 ALT6 - LPUART2_TXD ALT8 - TPM1_EXTCLK	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_01	ALT0 - SRC_BOOT_MODE1 ALT1 - FLEXCAN1_RX ALT4 - LPTMR1_ALT2 ALT5 - GPIO1_IO01 ALT6 - LPUART2_RXD ALT8 - TPM1_CH0	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_02	ALT0 - SRC_BOOT_MODE2	ODE - Disabled

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT1 - FLEXCAN3_TX ALT2 - LPSPI2_PCS3 ALT3 - LPSPI2_SOUT ALT4 - LPTMR1_ALT3 ALT5 - GPIO1_IO02 ALT6 - LPUART2_RTS_B ALT8 - TPM1_CH1	PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_03	ALT1 - FLEXCAN3_RX ALT2 - LPSPI1_PCS1 ALT3 - LPSPI2_SIN ALT4 - LPSPI1_PCS3 ALT5 - GPIO1_IO03 ALT6 - LPUART2_CTS_B ALT8 - TPM1_CH2	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_04	ALT0 - LPSPI1_SCK ALT5 - GPIO1_IO04 ALT6 - LPUART7_CTS_B ALT8 - TPM1_CH3	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_05	ALT0 - LPSPI1_PCS0 ALT5 - GPIO1_IO05 ALT6 - LPUART7_RTS_B ALT7 - WAKEUP	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_06	ALT0 - LPSPI1_SOUT ALT1 - I3C1_PUR ALT3 - LPI2C1_SDA ALT5 - GPIO1_IO06 ALT6 - FLEXCAN1_TX	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_07	ALT0 - LPSPI1_SIN ALT3 - LPI2C1_SCL ALT5 - GPIO1_IO07	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT6 - FLEXCAN1_RX	DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_08	ALT0 - LPUART1_TXD ALT5 - GPIO1_IO08 ALT6 - LPI2C1_SDA ALT8 - LPSPI1_PCS1	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_09	ALT0 - LPUART1_RXD ALT3 - LPIT1_TRIGGER0 ALT5 - GPIO1_IO09 ALT6 - LPI2C1_SCL ALT8 - LPSPI1_PCS2	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_10	ALT0 - JTAG_MUX_TRSTB ALT1 - LPSPI2_PCS0 ALT3 - LPIT1_TRIGGER1 ALT4 - TPM2_EXTCLK ALT5 - GPIO1_IO10 ALT6 - LPI2C1_SCLS	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_11	ALT0 - JTAG_MUX_TDO ALT2 - LPUART1_CTS_B ALT3 - LPIT1_TRIGGER2 ALT4 - TPM2_CH0 ALT5 - GPIO1_IO11 ALT6 - LPI2C1_SDAS	ODE - Disabled PUS - Weak Pull Up PUE - Pull Disable (High Z) DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_12	ALT0 - JTAG_MUX_TDI ALT2 - LPUART1_RTS_B ALT3 - LPIT1_TRIGGER3 ALT4 - TPM2_CH1 ALT5 - GPIO1_IO12 ALT6 - LPI2C1_HREQ ALT8 - LPSPI1_SCK	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_13	ALT0 - JTAG_MUX_TCK ALT3 - LPUART1_DSR_B	ODE - Disabled PUS - Weak Pull Down

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT4 - TPM2_CH2 ALT5 - GPIO1_IO13 ALT6 - LPTMR1_ALT1 ALT8 - LPSPI1_PCS0	PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_14	ALT0 - JTAG_MUX_TMS ALT3 - LPUART1_DCD_B ALT4 - TPM2_CH3 ALT5 - GPIO1_IO14 ALT6 - LPTMR1_ALT2 ALT8 - LPSPI1_SOUT	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_15	ALT0 - FLEXSPI2_B_DATA3 ALT1 - LPSPI2_PCS1 ALT3 - LPUART1_RI_B ALT4 - LPI2C2_SDA ALT5 - GPIO1_IO15 ALT6 - LPTMR1_ALT3 ALT8 - LPSPI1_SIN ALT9 - I3C1_SDA	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_16	ALT0 - FLEXSPI2_B_DATA2 ALT1 - LPSPI2_PCS0 ALT3 - LPUART1_DTR_B ALT4 - LPI2C2_SCL ALT5 - GPIO1_IO16 ALT6 - FLEXCAN1_TX ALT8 - LPUART7_CTS_B ALT9 - I3C1_SCL	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_17	ALT0 - FLEXSPI2_B_DATA1 ALT1 - LPSPI2_SIN ALT2 - LPUART7_TXD ALT3 - LPI2C2_SDA ALT4 - LPUART1_DSR_B ALT5 - GPIO1_IO17 ALT6 - FLEXCAN1_RX	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
GPIO_AON_18	ALT0 - FLEXSPI2_B_DATA0 ALT1 - LPSPI2_SOUT ALT2 - LPUART7_RXD ALT3 - LPI2C2_SCL ALT4 - LPUART1_DCD_B ALT5 - GPIO1_IO18 ALT6 - FLEXCAN3_TX	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_19	ALT0 - FLEXSPI2_B_SCLK ALT1 - LPSPI2_SCK ALT3 - FLEXSPI2_A_SS1_B ALT4 - LPUART1_CTS_B ALT5 - GPIO1_IO19 ALT6 - FLEXCAN3_RX ALT8 - LPUART7_RTS_B	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_20	ALT0 - FLEXSPI2_B_DQS ALT1 - FLEXSPI2_A_SS1_B ALT2 - LPI2C1_SDA ALT3 - I3C1_SDA ALT4 - LPUART1_RTS_B ALT5 - GPIO1_IO20	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_21	ALT0 - FLEXSPI2_B_SS0_B ALT1 - LPSPI2_PCS1 ALT2 - LPI2C1_SCL ALT3 - I3C1_SCL ALT5 - GPIO1_IO21 ALT8 - FLEXSPI2_A_DQS	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_22	ALT0 - FLEXSPI2_A_SS0_B ALT1 - LPI2C2_SDA ALT2 - LPUART7_TXD ALT5 - GPIO1_IO22 ALT6 - LPSPI2_SCK	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_23	ALT0 - FLEXSPI2_A_SCLK	ODE - Disabled

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT1 - LPI2C2_SCL ALT2 - LPUART7_RXD ALT5 - GPIO1_IO23 ALT6 - LPSPI2_SOUT	PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_24	ALT0 - FLEXSPI2_A_DATA0 ALT1 - LPI2C1_SDA ALT2 - LPUART2_RTS_B ALT3 - LPUART7_CTS_B ALT5 - GPIO1_IO24 ALT6 - LPSPI2_SIN	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_25	ALT0 - FLEXSPI2_A_DATA1 ALT1 - LPI2C1_SCL ALT2 - LPUART2_CTS_B ALT3 - LPUART7_RTS_B ALT5 - GPIO1_IO25 ALT6 - LPSPI2_PCS0	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_26	ALT0 - FLEXSPI2_A_DATA2 ALT1 - LPSPI2_PCS2 ALT2 - LPUART2_TXD ALT5 - GPIO1_IO26	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_27	ALT0 - FLEXSPI2_A_DATA3 ALT1 - LPSPI2_PCS3 ALT2 - LPUART2_RXD ALT5 - GPIO1_IO27 ALT7 - EWM_OUT_B	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AON_28_DUMMY	ALT0 - FLEXSPI2_A_DQS ALT1 - FLEXSPI2_B_DQS ALT5 - GPIO1_IO28	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_EMC_B1_00	ALT0 - SEMC_DATA00	ODE - Disabled

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT1 - XBAR_INOUT04 ALT2 - SINC3_MOD_CLK0 ALT3 - LPUART3_CTS_B ALT4 - ETH3_TX_DATA3 ALT5 - GPIO2_IO00 ALT6 - KPP_ROW3 ALT8 - FLEXIO1_D00 ALT9 - ETH4_TX_DATA3	PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_01	ALT0 - SEMC_DATA01 ALT1 - XBAR_INOUT05 ALT2 - SINC3_MOD_CLK1 ALT3 - LPUART3_RTS_B ALT4 - ETH3_TX_DATA2 ALT5 - GPIO2_IO01 ALT6 - KPP_COL3 ALT8 - FLEXIO1_D01 ALT9 - ETH4_TX_DATA2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_02	ALT0 - SEMC_DATA02 ALT1 - XBAR_INOUT06 ALT2 - SINC3_MOD_CLK2 ALT3 - LPUART3_RXD ALT4 - ETH3_RX_CLK ALT5 - GPIO2_IO02 ALT6 - KPP_ROW2 ALT8 - FLEXIO1_D02 ALT9 - ETH4_RX_CLK	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_03	ALT0 - SEMC_DATA03 ALT1 - XBAR_INOUT07 ALT2 - SINC3_EMCLK0 ALT3 - LPUART3_TXD ALT4 - ETH3_RX_DATA3 ALT5 - GPIO2_IO03 ALT6 - KPP_COL2	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT8 - FLEXIO1_D03 ALT9 - ETH4_RX_DATA3	
GPIO_EMC_B1_04	ALT0 - SEMC_DATA04 ALT1 - XBAR_INOUT08 ALT2 - SINC3_EMBIT0 ALT3 - LPUART3_DSR_B ALT4 - ETH3_RX_DATA2 ALT5 - GPIO2_IO04 ALT6 - KPP_ROW1 ALT8 - FLEXIO1_D04 ALT9 - ETH4_RX_DATA2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_05	ALT0 - SEMC_DATA05 ALT1 - XBAR_INOUT09 ALT2 - SINC3_EMCLK1 ALT3 - LPUART3_DCD_B ALT4 - ETH3_TX_DATA0 ALT5 - GPIO2_IO05 ALT6 - KPP_ROW7 ALT8 - FLEXIO1_D05 ALT9 - ETH4_TX_DATA0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_06	ALT0 - SEMC_DATA06 ALT1 - FLEXPWM4_PWM3_B ALT2 - SINC3_EMBIT1 ALT3 - LPUART3_RI_B ALT4 - ETH3_TX_DATA1 ALT5 - GPIO2_IO06 ALT6 - KPP_COL7 ALT8 - FLEXIO1_D06 ALT9 - ETH4_TX_DATA1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_07	ALT0 - SEMC_DATA07 ALT1 - FLEXPWM4_PWM3_A ALT2 - SINC3_EMCLK2 ALT3 - LPUART3_DTR_B	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT4 - ETH3_TX_EN ALT5 - GPIO2_IO07 ALT6 - KPP_ROW6 ALT8 - FLEXIO1_D07 ALT9 - ETH4_TX_EN	
GPIO_EMC_B1_08	ALT0 - SEMC_DM0 ALT1 - FLEXPWM2_PWM3_B ALT2 - SINC3_EMBIT2 ALT3 - LPUART4_DSR_B ALT4 - ETH3_TX_CLK ALT5 - GPIO2_IO08 ALT6 - KPP_COL6 ALT8 - FLEXIO1_D08 ALT9 - ETH4_TX_CLK	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_09	ALT0 - SEMC_ADDR00 ALT1 - FLEXPWM2_PWM3_A ALT2 - SINC3_EMCLK3 ALT3 - LPUART4_DCD_B ALT4 - ETH3_RX_DATA0 ALT5 - GPIO2_IO09 ALT6 - KPP_ROW5 ALT8 - FLEXIO1_D09 ALT9 - ETH4_RX_DATA0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_10	ALT0 - SEMC_ADDR01 ALT1 - FLEXPWM3_PWM3_B ALT2 - SINC3_EMBIT3 ALT3 - LPUART4_RI_B ALT4 - ETH3_RX_DATA1 ALT5 - GPIO2_IO10 ALT6 - KPP_COL5 ALT8 - FLEXIO1_D10 ALT9 - ETH4_RX_DATA1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_11	ALT0 - SEMC_ADDR02	ODE - Disabled

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT1 - FLEXPWM3_PWM3_A ALT2 - SINC3_BREAK ALT3 - LPUART4_DTR_B ALT4 - ETH3_RX_DV ALT5 - GPIO2_IO11 ALT6 - KPP_ROW4 ALT8 - FLEXIO1_D11 ALT9 - ETH4_RX_DV	PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_12	ALT0 - SEMC_ADDR03 ALT1 - FLEXPWM4_PWM0_A ALT2 - LPUART4_TXD ALT4 - ETH3_RX_ER ALT5 - GPIO2_IO12 ALT6 - KPP_COL4 ALT8 - FLEXIO1_D12 ALT9 - ETH4_RX_ER	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_13	ALT0 - SEMC_ADDR04 ALT1 - FLEXPWM4_PWM0_B ALT2 - LPUART4_RXD ALT3 - ETH2_RX_DV ALT4 - ETH3_TX_ER ALT5 - GPIO2_IO13 ALT6 - KPP_COL1 ALT8 - FLEXIO1_D13 ALT9 - ETH4_TX_ER ALT10 - QTIMER1_TIMER1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_14	ALT0 - SEMC_ADDR05 ALT1 - FLEXPWM4_PWM1_A ALT2 - LPUART5_TXD ALT3 - ETH2_TX_EN ALT4 - ETH3_CRS ALT5 - GPIO2_IO14 ALT6 - KPP_ROW0	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT8 - FLEXIO1_D14 ALT9 - ETH4_CRS ALT10 - LPUART4_CTS_B	
GPIO_EMC_B1_15	ALT0 - SEMC_ADDR06 ALT1 - FLEXPWM4_PWM1_B ALT2 - LPUART5_RXD ALT3 - ETH2_TX_CLK ALT4 - ETH3_COL ALT5 - GPIO2_IO15 ALT6 - KPP_COL0 ALT8 - FLEXIO1_D15 ALT9 - ETH4_COL ALT10 - LPUART4_RTS_B	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_16	ALT0 - SEMC_ADDR07 ALT1 - FLEXPWM4_PWM2_B ALT3 - ETH2_RX_DATA0 ALT4 - ETH3_MDC ALT5 - GPIO2_IO16 ALT6 - ETH4_MDC ALT8 - FLEXIO1_D16 ALT9 - ETH2_MDC ALT10 - LPSPI6_PCS2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_17	ALT0 - SEMC_ADDR08 ALT1 - FLEXPWM4_PWM2_A ALT3 - ETH2_RX_DATA1 ALT4 - ETH3_MDIO ALT5 - GPIO2_IO17 ALT6 - ETH4_MDIO ALT8 - FLEXIO1_D17 ALT9 - ETH2_MDIO ALT10 - LPSPI6_PCS1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_18	ALT0 - SEMC_ADDR09 ALT1 - FLEXPWM2_PWM0_A	ODE - Disabled PULL - Pull Down

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT2 - QTIMER1_TIMER0 ALT3 - LPSPI6_SCK ALT4 - ETH2_CRS ALT5 - GPIO2_IO18 ALT8 - FLEXIO1_D18 ALT10 - NETC_EMDC	PDRV - High Driver
GPIO_EMC_B1_19	ALT0 - SEMC_ADDR11 ALT1 - FLEXPWM2_PWM0_B ALT2 - QTIMER2_TIMER0 ALT3 - LPSPI6_SIN ALT4 - ETH2_COL ALT5 - GPIO2_IO19 ALT8 - FLEXIO1_D19 ALT10 - NETC_EMDIO	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_20	ALT0 - SEMC_ADDR12 ALT1 - FLEXPWM2_PWM1_A ALT2 - QTIMER3_TIMER0 ALT3 - LPSPI6_SOUT ALT4 - ETH2_TX_ER ALT5 - GPIO2_IO20 ALT8 - FLEXIO1_D20	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_21	ALT0 - SEMC_BA0 ALT1 - FLEXPWM2_PWM1_B ALT2 - QTIMER4_TIMER0 ALT3 - LPSPI6_PCS0 ALT4 - ETH2_RX_CLK ALT5 - GPIO2_IO21 ALT8 - FLEXIO1_D21 ALT9 - LPUART4_CTS_B ALT10 - FLEXSPI2_B_DQS	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_22	ALT0 - SEMC_BA1 ALT1 - FLEXPWM2_PWM2_B ALT3 - LPSPI4_SCK	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT4 - ETH2_RX_DATA2 ALT5 - GPIO2_IO22 ALT8 - FLEXIO1_D22 ALT9 - LPUART4_RTS_B ALT10 - FLEXSPI2_B_DATA3	
GPIO_EMC_B1_23	ALT0 - SEMC_ADDR10 ALT1 - FLEXPWM2_PWM2_A ALT3 - LPSPI4_SIN ALT4 - ETH2_RX_DATA3 ALT5 - GPIO2_IO23 ALT8 - FLEXIO1_D23 ALT10 - FLEXSPI2_B_DATA2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_24	ALT0 - SEMC_CAS ALT1 - FLEXPWM1_PWM0_A ALT3 - LPSPI4_SOUT ALT4 - ETH2_TX_DATA3 ALT5 - GPIO2_IO24 ALT8 - FLEXIO1_D24 ALT9 - ETH3_MDC ALT10 - FLEXSPI2_B_DATA1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_25	ALT0 - SEMC_RAS ALT1 - FLEXPWM1_PWM0_B ALT3 - LPSPI4_PCS0 ALT4 - ETH2_TX_DATA2 ALT5 - GPIO2_IO25 ALT8 - FLEXIO1_D25 ALT9 - ETH3_MDIO ALT10 - FLEXSPI2_B_DATA0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_26	ALT0 - SEMC_CLK ALT1 - FLEXPWM1_PWM1_A ALT2 - XBAR_INOUT10 ALT3 - FLEXSPI2_A_SS1_B ALT4 - ETH2_TX_DATA1	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT5 - GPIO2_IO26 ALT10 - LPSPI6_SCK	
GPIO_EMC_B1_27	ALT0 - SEMC_CKE ALT1 - FLEXPWM1_PWM1_B ALT2 - XBAR_INOUT11 ALT3 - FLEXSPI2_B_SS1_B ALT4 - ETH2_TX_DATA0 ALT5 - GPIO2_IO27 ALT9 - LPUART6_RI_B ALT10 - LPSPI6_SIN	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_28	ALT0 - SEMC_WE ALT1 - FLEXPWM1_PWM2_B ALT2 - XBAR_INOUT12 ALT3 - FLEXSPI2_B_SS0_B ALT4 - ETH2_TX_EN ALT5 - GPIO2_IO28 ALT9 - LPUART6_DTR_B ALT10 - LPSPI6_SOUT	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_29	ALT0 - SEMC_CS0 ALT1 - FLEXPWM1_PWM2_A ALT2 - XBAR_INOUT13 ALT3 - FLEXSPI2_B_DQS ALT4 - ETH2_TX_CLK ALT5 - GPIO2_IO29 ALT9 - LPUART6_DCD_B ALT10 - LPSPI6_PCS0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_30	ALT0 - SEMC_DATA08 ALT1 - FLEXPWM3_PWM0_A ALT2 - XBAR_INOUT14 ALT3 - FLEXSPI2_B_DATA3 ALT4 - ETH2_RX_DATA0 ALT5 - GPIO2_IO30 ALT9 - LPUART6_DSR_B	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT10 - LPSPI6_PCS1	
GPIO_EMC_B1_31	ALT0 - SEMC_DATA09 ALT1 - FLEXPWM3_PWM0_B ALT2 - LPUART6_TXD ALT3 - FLEXSPI2_B_DATA2 ALT4 - ETH2_RX_DATA1 ALT5 - GPIO2_IO31 ALT9 - LPSPI5_SCK ALT10 - LPSPI6_PCS2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_32	ALT0 - SEMC_DATA10 ALT1 - FLEXPWM3_PWM1_A ALT2 - LPUART6_RXD ALT3 - FLEXSPI2_B_DATA1 ALT4 - ETH2_RX_DV ALT5 - GPIO3_IO00 ALT9 - LPSPI5_SOUT ALT10 - LPSPI6_PCS3	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_33	ALT0 - SEMC_DATA11 ALT1 - FLEXPWM3_PWM1_B ALT2 - LPUART6_CTS_B ALT3 - FLEXSPI2_B_DATA0 ALT4 - ETH2_RX_ER ALT5 - GPIO3_IO01 ALT8 - LVDS2_RXD ALT9 - LPSPI5_SIN ALT10 - ETH2_RX_CLK	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_34	ALT0 - SEMC_DATA12 ALT1 - FLEXPWM3_PWM2_B ALT2 - LPUART6_RTS_B ALT3 - FLEXSPI2_B_SCLK ALT4 - ETH2_RX_DATA2 ALT5 - GPIO3_IO02 ALT8 - LVDS2_TXD	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT9 - ETH0_TX_DATA0 ALT10 - LPSPI5_PCS0	
GPIO_EMC_B1_35	ALT0 - SEMC_DATA13 ALT1 - FLEXPWM3_PWM2_A ALT2 - LPUART5_TXD ALT3 - FLEXSPI2_A_DATA0 ALT4 - ETH2_RX_DATA3 ALT5 - GPIO3_IO03 ALT8 - LVDS2_TX_EN ALT9 - ETH0_TX_DATA1 ALT10 - LPSPI5_PCS1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_36	ALT0 - SEMC_DATA14 ALT1 - FLEXPWM1_PWM0_A ALT2 - LPUART5_RXD ALT3 - FLEXSPI2_A_DATA1 ALT4 - ETH2_TX_DATA3 ALT5 - GPIO3_IO04 ALT8 - LVDS1_RXD ALT9 - ETH0_TX_EN	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_37	ALT0 - SEMC_DATA15 ALT1 - FLEXPWM1_PWM0_B ALT2 - LPUART5_CTS_B ALT3 - FLEXSPI2_A_DATA2 ALT4 - ETH2_TX_DATA2 ALT5 - GPIO3_IO05 ALT8 - LVDS1_TXD ALT9 - ETH0_TX_CLK	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_38	ALT0 - SEMC_DM1 ALT1 - FLEXPWM1_PWM3_B ALT2 - LPUART5_RTS_B ALT3 - FLEXSPI2_A_DATA3 ALT4 - ETH2_RX_CLK ALT5 - GPIO3_IO06	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT8 - LVDS1_TX_EN ALT9 - ETH0_RX_DATA0	
GPIO_EMC_B1_39	ALT0 - SEMC_DQS ALT1 - FLEXPWM1_PWM3_A ALT2 - XBAR_INOUT15 ALT3 - FLEXSPI2_A_SS0_B ALT4 - ETH2_TX_ER ALT5 - GPIO3_IO07 ALT6 - QTIMER2_TIMER1 ALT8 - LVDS0_RXD ALT9 - ETH0_RX_DATA1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_40	ALT0 - SEMC_RDY ALT1 - NETC_EMDC ALT2 - ETH2_MDC ALT3 - FLEXSPI2_A_DQS ALT4 - ETH2_CRS ALT5 - GPIO3_IO08 ALT6 - QTIMER3_TIMER1 ALT8 - LVDS0_TXD ALT9 - ETH0_RX_DV	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B1_41	ALT0 - SEMC_CSX0 ALT1 - NETC_EMDIO ALT2 - ETH2_MDIO ALT3 - FLEXSPI2_A_SCLK ALT4 - ETH2_COL ALT5 - GPIO3_IO09 ALT6 - QTIMER4_TIMER1 ALT8 - LVDS0_TX_EN ALT9 - ETH0_RX_ER	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_00	ALT0 - SEMC_DATA16 ALT1 - CCM_ENET_REF_CLK_25M ALT3 - NETC_EMDC ALT4 - ETH0_RX_CLK	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT5 - GPIO3_IO10 ALT6 - XBAR_INOUT20 ALT8 - LPSPI5_SCK ALT9 - LPI2C3_SCL ALT10 - FLEXPWM3_PWM0_A	
GPIO_EMC_B2_01	ALT0 - SEMC_DATA17 ALT1 - USDHC2_CD_B ALT3 - NETC_EMDIO ALT4 - ETH0_RX_DATA2 ALT5 - GPIO3_IO11 ALT6 - XBAR_INOUT21 ALT8 - LPSPI5_PCS0 ALT9 - LPI2C3_SDA ALT10 - FLEXPWM3_PWM0_B	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_02	ALT0 - SEMC_DATA18 ALT1 - USDHC2_WP ALT4 - ETH0_RX_DATA3 ALT5 - GPIO3_IO12 ALT6 - XBAR_INOUT22 ALT8 - LPSPI5_SOUT ALT9 - CCM_CLKO1 ALT10 - FLEXPWM3_PWM1_A	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_03	ALT0 - SEMC_DATA19 ALT1 - USDHC2_VSELECT ALT4 - ETH0_TX_DATA2 ALT5 - GPIO3_IO13 ALT6 - XBAR_INOUT23 ALT8 - LPSPI5_SIN ALT9 - ETH3_CRS ALT10 - FLEXPWM3_PWM1_B	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_04	ALT0 - SEMC_DATA20 ALT1 - USDHC2_RESET_B ALT4 - ETH0_TX_DATA3	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT5 - GPIO3_IO14 ALT6 - XBAR_INOUT24 ALT8 - LPSPI3_SCK ALT9 - ETH3_COL ALT10 - FLEXPWM3_PWM2_B	
GPIO_EMC_B2_05	ALT0 - SEMC_DATA21 ALT1 - ETH4_MDC ALT3 - ETH0_TX_DATA0 ALT4 - ETH4_CRS ALT5 - GPIO3_IO15 ALT6 - XBAR_INOUT25 ALT8 - LPSPI3_PCS0 ALT9 - ETH3_TX_DATA0 ALT10 - FLEXPWM3_PWM2_A	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_06	ALT0 - SEMC_DATA22 ALT1 - ETH4_MDIO ALT3 - ETH0_TX_DATA1 ALT4 - ETH4_COL ALT5 - GPIO3_IO16 ALT6 - XBAR_INOUT26 ALT8 - LPSPI3_SOUT ALT9 - ETH3_TX_DATA1 ALT10 - FLEXPWM3_PWM3_B	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_07	ALT0 - SEMC_DATA23 ALT1 - ETH4_TX_ER ALT3 - ETH0_TX_EN ALT5 - GPIO3_IO17 ALT6 - XBAR_INOUT27 ALT8 - LPSPI3_SIN ALT9 - ETH3_TX_EN ALT10 - FLEXPWM3_PWM3_A	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_08	ALT0 - SEMC_DM2 ALT1 - ETH4_RX_CLK	ODE - Disabled PULL - Pull Down

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT3 - ETH0_TX_CLK ALT5 - GPIO3_IO18 ALT6 - XBAR_INOUT28 ALT8 - LPSPI3_PCS3 ALT9 - ETH3_TX_CLK ALT10 - CCM_CLKO2	PDRV - High Driver
GPIO_EMC_B2_09	ALT0 - SEMC_DATA24 ALT1 - ETH4_RX_DATA3 ALT3 - ETH0_RX_DATA0 ALT5 - GPIO3_IO19 ALT6 - XBAR_INOUT29 ALT8 - LPSPI3_PCS2 ALT9 - ETH3_RX_DATA0 ALT10 - QTIMER1_TIMER0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_10	ALT0 - SEMC_DATA25 ALT1 - ETH4_RX_DATA2 ALT3 - ETH0_RX_DATA1 ALT5 - GPIO3_IO20 ALT6 - XBAR_INOUT30 ALT8 - LPSPI3_PCS1 ALT9 - ETH3_RX_DATA1 ALT10 - QTIMER1_TIMER1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_11	ALT0 - SEMC_DATA26 ALT1 - ETH4_TX_DATA3 ALT3 - ETH0_RX_DV ALT4 - LPSPI5_PCS3 ALT5 - GPIO3_IO21 ALT6 - XBAR_INOUT31 ALT9 - ETH3_RX_DV ALT10 - QTIMER1_TIMER2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_12	ALT0 - SEMC_DATA27 ALT1 - ETH4_TX_DATA2 ALT3 - ETH0_RX_ER	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT4 - LPSPI5_PCS2 ALT5 - GPIO3_IO22 ALT6 - XBAR_INOUT32 ALT9 - ETH3_RX_ER ALT10 - QTIMER1_TIMER3	
GPIO_EMC_B2_13	ALT0 - SEMC_DATA28 ALT1 - ETH4_TX_DATA0 ALT3 - ETH0_TX_DATA3 ALT4 - LPSPI5_PCS1 ALT5 - GPIO3_IO23 ALT6 - XBAR_INOUT33 ALT9 - ETH3_TX_DATA3 ALT10 - QTIMER2_TIMER0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_14	ALT0 - SEMC_DATA29 ALT1 - ETH4_TX_DATA1 ALT3 - ETH0_TX_DATA2 ALT4 - LPUART5_DSR_B ALT5 - GPIO3_IO24 ALT6 - XBAR_INOUT34 ALT9 - ETH3_TX_DATA2 ALT10 - QTIMER2_TIMER1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_15	ALT0 - SEMC_DATA30 ALT1 - ETH4_TX_EN ALT3 - ETH0_RX_CLK ALT4 - LPUART5_DCD_B ALT5 - GPIO3_IO25 ALT6 - XBAR_INOUT35 ALT9 - ETH3_RX_CLK ALT10 - QTIMER2_TIMER2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_16	ALT0 - SEMC_DATA31 ALT1 - ETH4_TX_CLK ALT3 - ETH0_RX_DATA2 ALT4 - LPUART5_DTR_B	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT5 - GPIO3_IO26 ALT6 - XBAR_INOUT14 ALT9 - ETH3_RX_DATA2 ALT10 - QTIMER2_TIMER3	
GPIO_EMC_B2_17	ALT0 - SEMC_DM3 ALT1 - ETH4_RX_DATA0 ALT2 - LPUART5_TXD ALT3 - ETH0_RX_DATA3 ALT5 - GPIO3_IO27 ALT6 - XBAR_INOUT15 ALT9 - ETH3_RX_DATA3 ALT10 - QTIMER3_TIMER0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_18	ALT0 - SEMC_DQS4 ALT1 - ETH4_RX_DATA1 ALT2 - LPUART5_RXD ALT3 - ETH0_TX_ER ALT5 - GPIO3_IO28 ALT6 - XBAR_INOUT16 ALT8 - EWM_OUT_B ALT9 - ETH3_TX_ER ALT10 - QTIMER3_TIMER1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_19	ALT0 - SEMC_CLKX0 ALT1 - ETH4_RX_DV ALT2 - LPUART5_CTS_B ALT3 - ETH0_CRS ALT4 - NETC_EMDC ALT5 - GPIO3_IO29 ALT6 - XBAR_INOUT36 ALT8 - LPI2C3_SCL ALT9 - ETH3_MDC ALT10 - QTIMER3_TIMER2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_EMC_B2_20	ALT0 - SEMC_CLKX1 ALT1 - ETH4_RX_ER	ODE - Disabled PULL - Pull Down

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT2 - LPUART5_RTS_B ALT3 - ETH0_COL ALT4 - NETC_EMDIO ALT5 - GPIO3_IO30 ALT6 - XBAR_INOUT37 ALT8 - LPI2C3_SDA ALT9 - ETH3_MDIO ALT10 - QTIMER3_TIMER3	PDRV - High Driver
GPIO_AD_00	ALT2 - MIC_CLK ALT3 - GPT2_CAPTURE1 ALT4 - FLEXPWM1_PWM0_A ALT5 - GPIO4_IO00 ALT6 - SINC1_MOD_CLK0 ALT8 - FLEXIO2_D00 ALT9 - QTIMER4_TIMER0	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_01	ALT2 - MIC_BITSTREAM0 ALT3 - GPT2_CAPTURE2 ALT4 - FLEXPWM1_PWM0_B ALT5 - GPIO4_IO01 ALT6 - SINC1_MOD_CLK1 ALT8 - FLEXIO2_D01 ALT9 - QTIMER4_TIMER1	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_02	ALT2 - MIC_BITSTREAM1 ALT3 - GPT2_COMPARE1 ALT4 - FLEXPWM1_PWM1_A ALT5 - GPIO4_IO02 ALT6 - SINC1_MOD_CLK2 ALT8 - FLEXIO2_D02 ALT9 - QTIMER4_TIMER2	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_03	ALT2 - MIC_BITSTREAM2 ALT3 - GPT2_COMPARE2 ALT4 - FLEXPWM1_PWM1_B ALT5 - GPIO4_IO03	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT6 - SINC1_EMCLK0 ALT8 - FLEXIO2_D03 ALT9 - QTIMER4_TIMER3	SRE - Slow Slew Rate
GPIO_AD_04	ALT2 - MIC_BITSTREAM3 ALT3 - GPT2_COMPARE3 ALT4 - FLEXPWM1_PWM2_B ALT5 - GPIO4_IO04 ALT6 - SINC1_EMBIT0 ALT8 - FLEXIO2_D04	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_05	ALT3 - GPT2_CLK ALT4 - FLEXPWM1_PWM2_A ALT5 - GPIO4_IO05 ALT6 - SINC1_EMCLK1 ALT7 - CCM_ENET_REF_CLK_25M ALT8 - FLEXIO2_D05	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_06	ALT0 - USB_OTG2_OC ALT1 - FLEXCAN3_TX ALT4 - FLEXPWM1_PWM0_X ALT5 - GPIO4_IO06 ALT6 - SINC1_EMBIT1 ALT8 - FLEXIO2_D06	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_07	ALT0 - USB_OTG2_PWR ALT1 - FLEXCAN3_RX ALT4 - FLEXPWM1_PWM1_X ALT5 - GPIO4_IO07 ALT6 - SINC1_EMCLK2 ALT8 - FLEXIO2_D07	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_08	ALT0 - USB_OTG2_ID ALT4 - FLEXPWM1_PWM2_X ALT5 - GPIO4_IO08 ALT6 - SINC1_EMBIT2 ALT8 - FLEXIO2_D08	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
GPIO_AD_09	ALT0 - USB_OTG1_ID ALT4 - FLEXPWM1_PWM3_X ALT5 - GPIO4_IO09 ALT6 - SINC1_EMCLK3 ALT8 - FLEXIO2_D09	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_10	ALT0 - USB_OTG1_PWR ALT4 - FLEXPWM2_PWM0_X ALT5 - GPIO4_IO10 ALT6 - SINC1_EMBIT3 ALT8 - FLEXIO2_D10	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_11	ALT0 - USB_OTG1_OC ALT4 - FLEXPWM2_PWM1_X ALT5 - GPIO4_IO11 ALT6 - SINC1_BREAK ALT8 - FLEXIO2_D11	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_12	ALT2 - GPT1_CAPTURE1 ALT3 - KPP_ROW7 ALT4 - FLEXPWM2_PWM2_X ALT5 - GPIO4_IO12 ALT6 - XBAR_INOUT18 ALT7 - EWM_OUT_B ALT8 - FLEXIO2_D12	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_13	ALT2 - GPT1_CAPTURE2 ALT3 - KPP_COL7 ALT4 - FLEXPWM2_PWM3_X ALT5 - GPIO4_IO13 ALT6 - LPUART3_TXD ALT7 - USDHC2_CD_B ALT8 - FLEXIO2_D13	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_14	ALT2 - GPT1_COMPARE1 ALT3 - KPP_ROW6 ALT4 - FLEXPWM3_PWM0_X	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT5 - GPIO4_IO14 ALT6 - LPUART3_RXD ALT7 - USDHC2_WP ALT8 - FLEXIO2_D14	DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_15	ALT2 - GPT1_COMPARE2 ALT3 - KPP_COL6 ALT4 - FLEXPWM3_PWM1_X ALT5 - GPIO4_IO15 ALT6 - LPUART3_CTS_B ALT7 - LPSPI3_PCS1 ALT8 - FLEXIO2_D15 ALT9 - FLEXCAN1_TX	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_16	ALT2 - GPT1_COMPARE3 ALT3 - KPP_ROW5 ALT4 - FLEXPWM3_PWM2_X ALT5 - GPIO4_IO16 ALT6 - LPUART3_RTS_B ALT7 - LPSPI3_SCK ALT8 - FLEXIO2_D16 ALT9 - FLEXCAN1_RX	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_17	ALT1 - ACMP1_OUT ALT2 - GPT1_CLK ALT3 - KPP_COL5 ALT4 - FLEXPWM3_PWM3_X ALT5 - GPIO4_IO17 ALT6 - I3C2_PUR ALT7 - LPSPI3_PCS0 ALT8 - FLEXIO2_D17 ALT9 - LPI2C3_HREQ	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_18	ALT1 - ACMP2_OUT ALT2 - LPUART5_RI_B ALT3 - KPP_ROW4 ALT4 - FLEXPWM4_PWM0_X	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT5 - GPIO4_IO18 ALT6 - I3C2_SCL ALT7 - LPSPI3_SOUT ALT8 - FLEXIO2_D18 ALT9 - LPI2C3_SCL	SRE - Slow Slew Rate
GPIO_AD_19	ALT1 - ACMP3_OUT ALT2 - XBAR_INOUT19 ALT3 - KPP_COL4 ALT4 - FLEXPWM4_PWM1_X ALT5 - GPIO4_IO19 ALT6 - I3C2_SDA ALT7 - LPSPI3_SIN ALT8 - FLEXIO2_D19 ALT9 - LPI2C3_SDA	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_20	ALT1 - ACMP4_OUT ALT2 - LPIT2_TRIGGER0 ALT3 - SINC1_EMCLK0 ALT4 - FLEXPWM4_PWM2_X ALT5 - GPIO4_IO20 ALT6 - NETC_TMR_TRIG1 ALT7 - NETC_1588_CLK ALT8 - FLEXIO2_D20	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_21	ALT2 - LPIT2_TRIGGER1 ALT3 - SINC1_EMBIT0 ALT4 - FLEXPWM4_PWM3_X ALT5 - GPIO4_IO21 ALT6 - NETC_TMR_TRIG2 ALT7 - NETC_TMR_GCLK ALT8 - FLEXIO2_D21	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_22	ALT2 - LPIT2_TRIGGER2 ALT3 - SINC1_EMCLK1 ALT5 - GPIO4_IO22 ALT7 - NETC_TMR_ALARM1	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT8 - FLEXIO2_D22	SRE - Slow Slew Rate
GPIO_AD_23	ALT2 - LPIT2_TRIGGER3 ALT3 - SINC1_EMBIT1 ALT5 - GPIO4_IO23 ALT7 - NETC_TMR_ALARM2 ALT8 - FLEXIO2_D23	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_24	ALT0 - LPUART6_TXD ALT3 - SINC2_MOD_CLK1 ALT4 - FLEXPWM2_PWM0_A ALT5 - GPIO4_IO24 ALT7 - NETC_TMR_TRIG1 ALT8 - FLEXIO2_D24	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_25	ALT0 - LPUART6_RXD ALT2 - LPSPI5_PCS3 ALT3 - SINC2_MOD_CLK2 ALT4 - FLEXPWM2_PWM0_B ALT5 - GPIO4_IO25 ALT7 - NETC_TMR_TRIG2 ALT8 - FLEXIO2_D25	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_26	ALT0 - LPUART6_CTS_B ALT1 - LPUART5_TXD ALT2 - LPSPI5_PCS2 ALT3 - SINC2_EMCLK0 ALT4 - FLEXPWM2_PWM1_A ALT5 - GPIO4_IO26 ALT6 - KPP_ROW0 ALT7 - NETC_TMR_PP1 ALT8 - FLEXIO2_D26 ALT9 - USDHC2_CD_B ALT12 - MIC_BITSTREAM2	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_27	ALT0 - LPUART6_RTS_B ALT1 - LPUART5_RXD	ODE - Disabled PUS - Weak Pull Down

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT2 - LPSPI5_PCS1 ALT3 - SINC2_EMBIT0 ALT4 - FLEXPWM2_PWM1_B ALT5 - GPIO4_IO27 ALT6 - KPP_COL0 ALT7 - NETC_TMR_PP2 ALT8 - FLEXIO2_D27 ALT9 - USDHC2_WP ALT12 - MIC_CLK	PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_28	ALT0 - LPSPI5_SCK ALT2 - I3C1_PUR ALT3 - SINC2_EMCLK1 ALT4 - FLEXPWM2_PWM2_B ALT5 - GPIO4_IO28 ALT6 - KPP_ROW3 ALT7 - NETC_TMR_PP3 ALT8 - FLEXIO2_D28 ALT9 - USDHC2_RESET_B ALT12 - MIC_BITSTREAM0	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_29	ALT0 - LPSPI5_PCS0 ALT2 - USDHC2_CD_B ALT3 - SINC2_EMBIT1 ALT4 - FLEXPWM2_PWM2_A ALT5 - GPIO4_IO29 ALT6 - KPP_COL3 ALT7 - EWM_OUT_B ALT8 - FLEXIO2_D29 ALT9 - USDHC2_VSELECT ALT12 - MIC_BITSTREAM1	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_30	ALT0 - LPSPI5_SOUT ALT1 - USB_OTG2_OC ALT3 - SINC2_EMCLK2 ALT4 - LPUART8_TXD	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT5 - GPIO4_IO30 ALT6 - KPP_ROW2 ALT7 - NETC_EMDC ALT8 - FLEXIO2_D30 ALT9 - XBAR_INOUT24	SRE - Slow Slew Rate
GPIO_AD_31	ALT0 - LPSPI5_SIN ALT1 - USB_OTG2_PWR ALT3 - SINC2_EMBIT2 ALT4 - LPUART8_RXD ALT5 - GPIO4_IO31 ALT6 - KPP_COL2 ALT7 - NETC_EMDIO ALT8 - FLEXIO2_D31 ALT9 - XBAR_INOUT25	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_32	ALT1 - USB_OTG2_ID ALT2 - PMIC_READY ALT3 - SINC2_EMCLK3 ALT4 - USDHC1_CD_B ALT5 - GPIO5_IO00 ALT6 - KPP_ROW1 ALT7 - NETC_TMR_TRIG1 ALT12 - MIC_BITSTREAM3	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_33	ALT1 - USB_OTG1_ID ALT2 - XBAR_INOUT17 ALT3 - SINC2_EMBIT3 ALT4 - USDHC1_WP ALT5 - GPIO5_IO01 ALT6 - KPP_COL1 ALT7 - NETC_TMR_TRIG2	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_AD_34	ALT0 - I3C2_SCL ALT1 - USB_OTG1_PWR ALT2 - XBAR_INOUT18 ALT3 - SINC2_BREAK	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT4 - USDHC1_VSELECT ALT5 - GPIO5_IO02 ALT7 - NETC_TMR_ALARM1	SRE - Slow Slew Rate
GPIO_AD_35	ALT0 - I3C2_SDA ALT1 - USB_OTG1_OC ALT2 - XBAR_INOUT19 ALT3 - SINC2_MOD_CLK0 ALT4 - USDHC1_RESET_B ALT5 - GPIO5_IO03 ALT7 - NETC_TMR_ALARM2	ODE - Disabled PUS - Weak Pull Down PUE - Pull Enable DSE - High Driver SRE - Slow Slew Rate
GPIO_SD_B1_00	ALT0 - USDHC1_CMD ALT1 - SINC1_EMCLK2 ALT2 - XBAR_INOUT20 ALT3 - LPTMR2_ALT1 ALT4 - XSPI_SLV_CS ALT5 - GPIO5_IO04 ALT6 - LPSPI3_PCS0 ALT8 - KPP_ROW7 ALT10 - LVDS1_RXD ALT12 - CCM_CLKO1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B1_01	ALT0 - USDHC1_CLK ALT1 - SINC1_EMBIT2 ALT2 - XBAR_INOUT21 ALT3 - LPTMR2_ALT2 ALT4 - XSPI_SLV_CLK ALT5 - GPIO5_IO05 ALT6 - LPSPI3_SCK ALT8 - KPP_COL7 ALT10 - LVDS1_TXD ALT12 - CCM_CLKO2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B1_02	ALT0 - USDHC1_DATA0 ALT1 - SINC1_EMCLK3 ALT2 - XBAR_INOUT22	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT3 - LPTMR2_ALT3 ALT4 - XSPI_SLV_DATA4 ALT5 - GPIO5_IO06 ALT6 - LPSPI3_SOUT ALT8 - KPP_ROW6 ALT9 - FLEXSPI1_A_SS1_B ALT10 - LVDS1_TX_EN ALT12 - 0	
GPIO_SD_B1_03	ALT0 - USDHC1_DATA1 ALT1 - SINC1_EMBIT3 ALT2 - XBAR_INOUT23 ALT3 - LPTMR3_ALT1 ALT4 - XSPI_SLV_DATA5 ALT5 - GPIO5_IO07 ALT6 - LPSPI3_SIN ALT8 - KPP_COL6 ALT9 - FLEXSPI1_B_SS1_B ALT10 - LVDS0_RXD	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B1_04	ALT0 - USDHC1_DATA2 ALT1 - SINC1_BREAK ALT2 - SINC2_EMCLK2 ALT3 - LPTMR3_ALT2 ALT4 - XSPI_SLV_DATA6 ALT5 - GPIO5_IO08 ALT6 - LPSPI3_PCS1 ALT8 - FLEXSPI1_B_SS0_B ALT9 - FLEXSPI1_A_SS1_B ALT10 - LVDS0_TXD	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B1_05	ALT0 - USDHC1_DATA3 ALT2 - SINC2_EMBIT2 ALT3 - LPTMR3_ALT3 ALT4 - XSPI_SLV_DATA7 ALT5 - GPIO5_IO09	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT6 - LPSPI3_PCS2 ALT9 - FLEXSPI1_B_SS0_B ALT10 - LVDS0_TX_EN	
GPIO_SD_B2_00	ALT0 - USDHC2_DATA3 ALT1 - FLEXSPI1_B_DATA4 ALT2 - XSPI_SLV_DATA4 ALT3 - XBAR_INOUT17 ALT4 - KPP_ROW1 ALT5 - GPIO5_IO10 ALT6 - LPSPI3_PCS3 ALT8 - NETC_1588_CLK ALT9 - LPUART8_TXD ALT12 - MIC_BITSTREAM0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B2_01	ALT0 - USDHC2_DATA2 ALT1 - FLEXSPI1_B_DATA5 ALT2 - XSPI_SLV_DATA5 ALT4 - KPP_COL1 ALT5 - GPIO5_IO11 ALT8 - NETC_TMR_GCLK ALT9 - LPUART8_RXD ALT12 - MIC_BITSTREAM1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B2_02	ALT0 - USDHC2_DATA1 ALT1 - FLEXSPI1_B_DATA6 ALT2 - XSPI_SLV_DATA6 ALT4 - KPP_ROW0 ALT5 - GPIO5_IO12 ALT8 - NETC_TMR_ALARM2 ALT9 - LPUART8_CTS_B ALT12 - MIC_BITSTREAM2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B2_03	ALT0 - USDHC2_DATA0 ALT1 - FLEXSPI1_B_DATA7 ALT2 - XSPI_SLV_DATA7 ALT4 - KPP_COL0	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT5 - GPIO5_IO13 ALT8 - NETC_TMR_ALARM2 ALT9 - LPUART8_RTS_B ALT12 - MIC_BITSTREAM3	
GPIO_SD_B2_04	ALT0 - USDHC2_CLK ALT1 - FLEXSPI1_B_SS1_B ALT4 - KPP_ROW3 ALT5 - GPIO5_IO14 ALT6 - LPUART5_RI_B ALT8 - NETC_TMR_PP1 ALT12 - MIC_CLK	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B2_05	ALT0 - USDHC2_CMD ALT1 - FLEXSPI1_B_DQS ALT2 - XSPI_SLV_DQS ALT4 - LPSPI4_PCS3 ALT5 - GPIO5_IO15 ALT6 - LPUART5_DTR_B ALT8 - NETC_TMR_PP2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B2_06	ALT0 - USDHC2_RESET_B ALT1 - FLEXSPI1_B_SS0_B ALT2 - XSPI_SLV_CS ALT4 - LPSPI4_PCS2 ALT5 - GPIO5_IO16 ALT6 - LPUART5_CTS_B ALT8 - NETC_TMR_PP3	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B2_07	ALT0 - USDHC2_STROBE ALT1 - FLEXSPI1_B_SCLK ALT2 - XSPI_SLV_CLK ALT4 - LPSPI4_PCS1 ALT5 - GPIO5_IO17 ALT6 - LPUART5_RTS_B ALT8 - NETC_TMR_ALARM1 ALT10 - LVDS0_RXD	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
GPIO_SD_B2_08	ALT0 - USDHC2_DATA4 ALT1 - FLEXSPI1_B_DATA0 ALT2 - XSPI_SLV_DATA0 ALT4 - LPSPI4_SCK ALT5 - GPIO5_IO18 ALT6 - LPUART5_TXD ALT8 - NETC_TMR_ALARM2 ALT9 - NETC_TMR_PP2 ALT10 - LVDS0_TXD	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B2_09	ALT0 - USDHC2_DATA5 ALT1 - FLEXSPI1_B_DATA1 ALT2 - XSPI_SLV_DATA1 ALT4 - LPSPI4_PCS0 ALT5 - GPIO5_IO19 ALT6 - LPUART5_RXD ALT9 - NETC_TMR_PP1 ALT10 - LVDS0_TX_EN	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B2_10	ALT0 - USDHC2_DATA6 ALT1 - FLEXSPI1_B_DATA2 ALT2 - XSPI_SLV_DATA2 ALT4 - LPSPI4_SOUT ALT5 - GPIO5_IO20 ALT6 - LPUART5_DCD_B ALT8 - NETC_TMR_TRIG2 ALT9 - NETC_TMR_PP3 ALT10 - NETC_EMDIO	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_SD_B2_11	ALT0 - USDHC2_DATA7 ALT1 - FLEXSPI1_B_DATA3 ALT2 - XSPI_SLV_DATA3 ALT4 - LPSPI4_SIN ALT5 - GPIO5_IO21 ALT6 - LPUART5_DSR_B ALT7 - 0	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT8 - NETC_TMR_TRIG1 ALT10 - NETC_EMDC	
GPIO_SD_B2_12_DUMMY	ALT0 - FLEXSPI1_A_DQS ALT1 - FLEXSPI1_B_DQS ALT5 - GPIO5_IO22	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_00	ALT0 - ETH1_TX_DATA0 ALT2 - SEMC_CSX1 ALT3 - QTIMER1_TIMER0 ALT4 - XBAR_INOUT26 ALT5 - GPIO6_IO00 ALT8 - ETH4_TX_DATA0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_01	ALT0 - ETH1_TX_DATA1 ALT3 - QTIMER1_TIMER1 ALT4 - XBAR_INOUT27 ALT5 - GPIO6_IO01 ALT8 - ETH4_TX_DATA1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_02	ALT0 - ETH1_TX_EN ALT2 - LPI2C6_SCL ALT3 - QTIMER1_TIMER2 ALT4 - XBAR_INOUT28 ALT5 - GPIO6_IO02 ALT7 - FLEXSPI1_B_SS1_B ALT8 - ETH4_TX_EN	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_03	ALT0 - ETH1_TX_CLK ALT2 - LPI2C6_SDA ALT3 - QTIMER2_TIMER0 ALT4 - XBAR_INOUT29 ALT5 - GPIO6_IO03 ALT7 - FLEXSPI1_B_DQS ALT8 - ETH4_TX_CLK	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_04	ALT0 - ETH1_RX_DATA0 ALT3 - QTIMER2_TIMER1	ODE - Disabled PULL - Pull Down

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT4 - XBAR_INOUT30 ALT5 - GPIO6_IO04 ALT7 - FLEXSPI1_B_SS0_B ALT8 - ETH4_RX_DATA0	PDRV - High Driver
GPIO_B1_05	ALT0 - ETH1_RX_DATA1 ALT3 - QTIMER2_TIMER2 ALT4 - XBAR_INOUT31 ALT5 - GPIO6_IO05 ALT7 - FLEXSPI1_B_SCLK ALT8 - ETH4_RX_DATA1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_06	ALT0 - ETH1_RX_DV ALT3 - QTIMER3_TIMER0 ALT4 - XBAR_INOUT32 ALT5 - GPIO6_IO06 ALT7 - FLEXSPI1_B_DATA7 ALT8 - ETH4_RX_DV	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_07	ALT0 - ETH1_TX_DATA2 ALT3 - QTIMER3_TIMER1 ALT4 - XBAR_INOUT33 ALT5 - GPIO6_IO07 ALT7 - FLEXSPI1_B_DATA6 ALT8 - ETH4_RX_ER ALT9 - LPSPI6_SIN	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_08	ALT0 - ETH1_TX_DATA3 ALT2 - USDHC1_CD_B ALT3 - QTIMER3_TIMER2 ALT4 - XBAR_INOUT36 ALT5 - GPIO6_IO08 ALT7 - FLEXSPI1_B_DATA5 ALT8 - ETH4_TX_ER ALT9 - LPSPI6_SOUT	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_09	ALT0 - ETH1_RX_DATA2	ODE - Disabled

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT2 - USDHC1_WP ALT3 - QTIMER4_TIMER0 ALT4 - XBAR_INOUT37 ALT5 - GPIO6_IO09 ALT7 - FLEXSPI1_B_DATA4 ALT8 - ETH4_RX_DATA2 ALT9 - LPSPI6_PCS1	PULL - Pull Down PDRV - High Driver
GPIO_B1_10	ALT0 - ETH1_RX_DATA3 ALT2 - USDHC1_RESET_B ALT3 - QTIMER4_TIMER1 ALT4 - XBAR_INOUT34 ALT5 - GPIO6_IO10 ALT7 - FLEXSPI1_B_DATA3 ALT8 - ETH4_RX_DATA3 ALT9 - LPSPI6_PCS2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_11	ALT0 - ETH1_RX_CLK ALT3 - QTIMER4_TIMER2 ALT4 - XBAR_INOUT35 ALT5 - GPIO6_IO11 ALT7 - FLEXSPI1_B_DATA2 ALT8 - ETH4_RX_CLK ALT9 - LPSPI6_PCS3	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_12	ALT0 - ETH1_RX_ER ALT1 - NETC_EMDIO ALT5 - GPIO6_IO12 ALT7 - FLEXSPI1_B_DATA1 ALT8 - ETH4_TX_DATA2 ALT9 - LPSPI6_PCS0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B1_13	ALT0 - ETH1_TX_ER ALT1 - NETC_EMDC ALT2 - USDHC1_VSELECT ALT3 - CCM_ENET_REF_CLK_25M ALT5 - GPIO6_IO13	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT7 - FLEXSPI1_B_DATA0 ALT8 - ETH4_TX_DATA3 ALT9 - LPSPI6_SCK	
GPIO_B2_00	ALT0 - ETH1_CRS ALT2 - LPIT3_TRIGGER0 ALT5 - GPIO6_IO14 ALT8 - ETH4_CRS ALT9 - LPSPI6_SIN ALT10 - ETH2_MDIO	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B2_01	ALT0 - ETH1_COL ALT2 - LPIT3_TRIGGER1 ALT5 - GPIO6_IO15 ALT6 - FLEXSPI1_A_SS1_B ALT8 - ETH4_COL ALT9 - LPSPI6_SOUT ALT10 - ETH2_MDC ALT11 - ETH2_RX_ER	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B2_02	ALT2 - LPIT3_TRIGGER2 ALT3 - NETC_EMDIO ALT5 - GPIO6_IO16 ALT6 - FLEXSPI1_B_SCLK ALT9 - CCM_ENET_REF_CLK_25M ALT10 - EWM_OUT_B ALT11 - ETH2_RX_DATA2	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B2_03	ALT2 - LPIT3_TRIGGER3 ALT3 - NETC_EMDC ALT5 - GPIO6_IO17 ALT7 - FLEXSPI1_A_DATA4 ALT10 - XSPI_SLV_DATA4 ALT11 - ETH2_RX_DATA3	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B2_04	ALT0 - SINC1_MOD_CLK0 ALT1 - SINC2_MOD_CLK0	ODE - Disabled PULL - Pull Down

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT2 - SINC3_MOD_CLK0 ALT5 - GPIO6_IO18 ALT7 - FLEXSPI1_A_DATA5 ALT8 - TPM3_EXTCLK ALT10 - XSPI_SLV_DATA5 ALT11 - ETH2_TX_DATA2	PDRV - High Driver
GPIO_B2_05	ALT0 - SINC1_MOD_CLK1 ALT1 - SINC2_MOD_CLK1 ALT2 - SINC3_MOD_CLK1 ALT5 - GPIO6_IO19 ALT6 - MIC_CLK ALT7 - FLEXSPI1_A_DATA6 ALT8 - TPM3_CH0 ALT10 - XSPI_SLV_DATA6 ALT11 - ETH2_TX_DATA3	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B2_06	ALT0 - SINC1_MOD_CLK2 ALT1 - SINC2_MOD_CLK2 ALT2 - SINC3_MOD_CLK2 ALT3 - LPUART6_DSR_B ALT5 - GPIO6_IO20 ALT7 - FLEXSPI1_A_DATA7 ALT8 - TPM3_CH1 ALT10 - XSPI_SLV_DATA7 ALT11 - ETH2_TX_DATA0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B2_07	ALT3 - LPUART6_DCD_B ALT5 - GPIO6_IO21 ALT7 - FLEXSPI1_A_DQS ALT8 - TPM3_CH2 ALT10 - XSPI_SLV_DQS ALT11 - ETH2_TX_DATA1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B2_08	ALT1 - SINC2_EMCLK2 ALT4 - LPUART6_RI_B ALT5 - GPIO6_IO22	ODE - Disabled PULL - Pull Down PDRV - High Driver

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT6 - LPI2C6_SCL ALT7 - FLEXSPI1_A_SCLK ALT8 - TPM3_CH3 ALT10 - XSPI_SLV_CLK ALT11 - ETH2_TX_EN	
GPIO_B2_09	ALT1 - SINC2_EMBIT2 ALT4 - LPUART6_DTR_B ALT5 - GPIO6_IO23 ALT6 - LPI2C6_SDA ALT7 - FLEXSPI1_A_SS0_B ALT10 - XSPI_SLV_CS ALT11 - ETH2_TX_CLK	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B2_10	ALT0 - MIC_BITSTREAM0 ALT1 - SINC2_EMCLK3 ALT2 - FLEXCAN3_TX ALT3 - LPUART8_CTS_B ALT4 - LPUART6_TXD ALT5 - GPIO6_IO24 ALT7 - FLEXSPI1_A_DATA0 ALT9 - LPSPI4_SCK ALT10 - XSPI_SLV_DATA0 ALT11 - ETH2_RX_DATA0	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B2_11	ALT0 - MIC_BITSTREAM1 ALT1 - SINC2_EMBIT3 ALT2 - FLEXCAN3_RX ALT3 - LPUART8_RTS_B ALT4 - LPUART6_RXD ALT5 - GPIO6_IO25 ALT7 - FLEXSPI1_A_DATA1 ALT9 - LPSPI4_SIN ALT10 - XSPI_SLV_DATA1 ALT11 - ETH2_RX_DATA1	ODE - Disabled PULL - Pull Down PDRV - High Driver
GPIO_B2_12	ALT0 - MIC_BITSTREAM2	ODE - Disabled

Table continues on the next page...

Table 117. Pin Assignments (continued)

Pin Name	Pin Assignments	Pad Settings
	ALT1 - SINC2_BREAK ALT2 - LPUART8_TXD ALT4 - LPUART6_CTS_B ALT5 - GPIO6_IO26 ALT6 - FLEXCAN3_TX ALT7 - FLEXSPI1_A_DATA2 ALT9 - LPSPI4_SOUT ALT10 - XSPI_SLV_DATA2 ALT11 - ETH2_RX_DV	PULL - Pull Down PDRV - High Driver
GPIO_B2_13	ALT0 - MIC_BITSTREAM3 ALT1 - SINC2_EMCLK0 ALT2 - LPUART8_RXD ALT4 - LPUART6_RTS_B ALT5 - GPIO6_IO27 ALT6 - FLEXCAN3_RX ALT7 - FLEXSPI1_A_DATA3 ALT9 - LPSPI4_PCS0 ALT10 - XSPI_SLV_DATA3 ALT11 - ETH2_RX_CLK	ODE - Disabled PULL - Pull Down PDRV - High Driver

Chapter 14
Block Control - Always-on Domain, Non-secure (BLK_CTRL_NS_AON)

14.1 Chip-specific BLK_CTRL_AON Information

Table 118. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

14.2 Overview

The Always-On Domain, Non-secure (AON NS) Block Control (BLK_CTRL_NS_AON) consists of a group of registers to provide miscellaneous control and status for peripherals within the AON domain. These registers provide chip-level control outside the peripherals.

14.2.1 Block diagram

A block diagram of the AON NS BLK CTRL implementation is shown below.

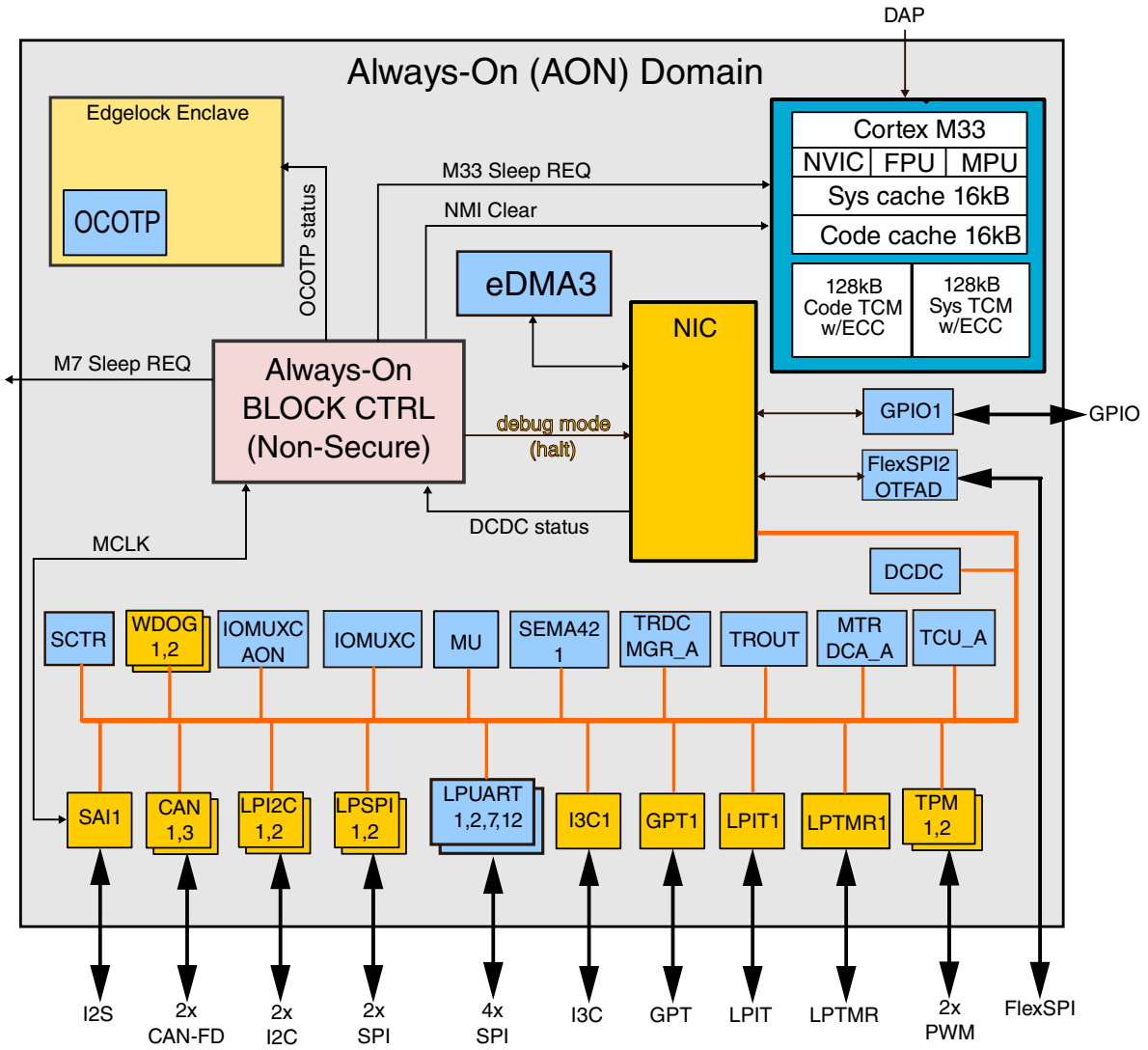


Figure 63. AON NS BLK CTRL Block Diagram

14.2.2 Features

Block Control Non-Secure AON Domain has the following features:

- Sleeping or deep sleeping selection for low power mode entry
- Debug halt mask bits for peripherals in AON Domain
- AON Domain SSI master block low power bits
- SAI1 MCLK direction control
- DCDC status bit and exception bits read and clear
- OCOTP miscellaneous status bits
- CM33 NMI interrupt clear bit
- I3C1 asynchronous interrupt clear bit
- I3C1 IO on chip strong pull enable bit

- GPIO_AON bank voltage range control

14.3 Clocks

The table found here describes the clock sources for BLK_CTRL_NS_AON. Please see the Clock Control chapter for clock setting, configuration, and gating information.

Table 119. Clocks

Clock Name	Description
bus_aon_clk	AON bus clock for register access.

14.4 Reset

The AON NS BLK_CTRL resets with the AON Domain.

14.5 Initialization

There are no initialization steps required for BLK_CTRL_NS_AON.

14.6 Memory Map and register definition

This section includes the BLK_CTRL_NS_AON memory map and detailed register descriptions.

14.6.1 Block Control Non-Secure AON Domain register descriptions

14.6.1.1 AON Domain Non_Secure Block Control memory map

BLK_CTRL_NS_AONMIX base address: 4421_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	GPC CORE SLEEP Request Select (GPC_CFG)	32	RW	0000_0000h
8h	IPG Debug mask (IPG_DEBUG)	32	RW	0000_0000h
1Ch	(SSI)	32	RW	0000_0002h
20h	SAI1 MCLK control register (SAI1_MCLK_CTRL)	32	RW	0000_0000h
24h	DCDC status register (DCDC_STATUS)	32	RW	0000_0000h
28h	Fuse access disable register (FUSE_ACC_DIS)	32	R	0000_0000h
2Ch	M33 NMI interrupt clear register (M33_NMI_CLR)	32	RW	0000_0000h
30h	I3C1 async wakeup control register (I3C1_ASYNC_WAKEUP_CTRL)	32	RW	0000_0000h
34h	Miscellaneous control register of IO (MISC_IO_CTRL)	32	RW	0000_0000h

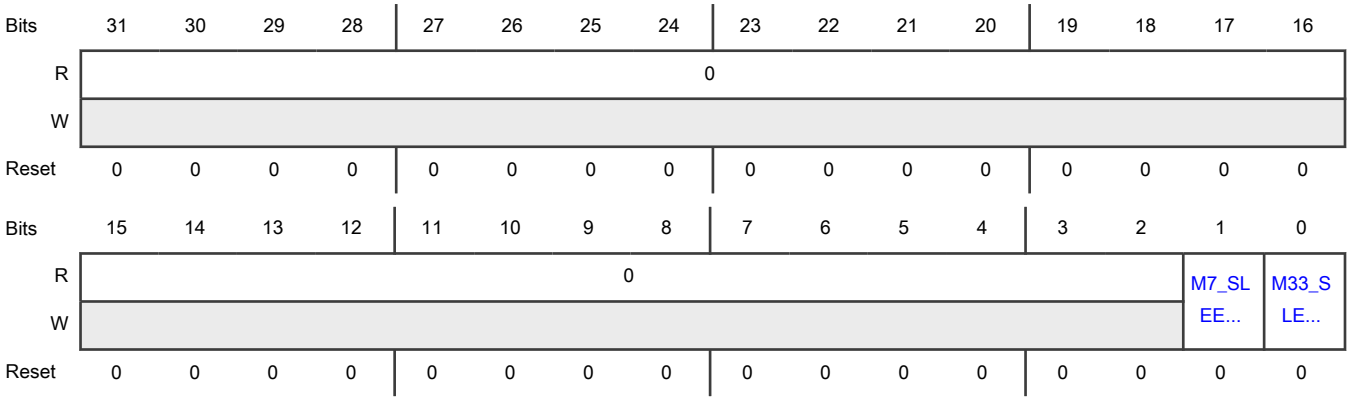
14.6.1.2 GPC CORE SLEEP Request Select (GPC_CFG)

Offset

Register	Offset
GPC_CFG	0h

Function
This register selects source for M33 and M7 core sleep request.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 M7_SLEEP_SEL	M7 SLEEP Request Select 0b - Select SLEEPING as request source 1b - Select SLEEPDEEP as request source
0 M33_SLEEP_SEL	M33 SLEEP Request Select 0b - Select SLEEPING as request source 1b - Select SLEEPDEEP as request source

14.6.1.3 IPG Debug mask (IPG_DEBUG)

Offset

Register	Offset
IPG_DEBUG	8h

Function

This register includes mask bits for HALTED from CM33/CM7 to peripheral ipg_debug

0: mask to 0

1: unmask

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M7_I3C1	M7_CAN3	M7_GPT1	M7_WDOG2	M7_WDOG1	M7_TPM2	M7_TPM1	M7_SAI1	M7_LPTM...	M7_LPSP...	M7_LPSP...	M7_LPIT1	M7_LPI2...	M7_LPI2...	M7_EDMA3	M7_CAN1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M33_I3C1	M33_CAN3	M33_GPT1	M33_WDO...	M33_WDO...	M33_TPM2	M33_TPM1	M33_SAI1	M33_LPT...	M33_LPS...	M33_LPS...	M33_LPI...	M33_LPI...	M33_LPI...	M33_EDM...	M33_CAN1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 M7_I3C1	Mask bit for I3C1 debug halted mode with M7 core 0b - I3C1 does not enter debug halted mode with CM7 1b - I3C1 enters debug halted mode when CM7 is debug halted
30 M7_CAN3	Mask bit for CAN3 debug halted mode with M7 core 0b - CAN3 does not enter debug halted mode with CM7 1b - CAN3 enters debug halted mode when CM7 is debug halted
29 M7_GPT1	Mask bit for GPT1 debug halted mode with M7 core 0b - GPT1 does not enter debug halted mode with CM7 1b - GPT1 enters debug halted mode when CM7 is debug halted
28 M7_WDOG2	Mask bit for WDOG2 debug halted mode with M7 core 0b - WDOG2 does not enter debug halted mode with CM7 1b - WDOG2 enters debug halted mode when CM7 is debug halted
27 M7_WDOG1	Mask bit for WDOG1 debug halted mode with M7 core 0b - WDOG1 does not enter debug halted mode with CM7 1b - WDOG1 enters debug halted mode when CM7 is debug halted
26 M7_TPM2	Mask bit for TPM2 debug halted mode with M7 core 0b - TPM2 does not enter debug halted mode with CM7 1b - TPM2 enters debug halted mode when CM7 is debug halted

Table continues on the next page...

Table continued from the previous page...

Field	Function
25 M7_TPM1	Mask bit for TPM1 debug halted mode with M7 core 0b - TPM1 does not enter debug halted mode with CM7 1b - TPM1 enters debug halted mode when CM7 is debug halted
24 M7_SAI1	Mask bit for SAI1 debug halted mode with M7 core 0b - SAI1 does not enter debug halted mode with CM7 1b - SAI1 enters debug halted mode when CM7 is debug halted
23 M7_LPTMR1	Mask bit for LPTMR1 debug halted mode with M7 core 0b - LPTMR1 does not enter debug halted mode with CM7 1b - LPTMR1 enters debug halted mode when CM7 is debug halted
22 M7_LPSPI2	Mask bit for LPSPI2 debug halted mode with M7 core 0b - LPSPI2 does not enter debug halted mode with CM7 1b - LPSPI2 enters debug halted mode when CM7 is debug halted
21 M7_LPSPI1	Mask bit for LPSPI1 debug halted mode with M7 core 0b - LPSPI1 does not enter debug halted mode with CM7 1b - LPSPI1 enters debug halted mode when CM7 is debug halted
20 M7_LPIT1	Mask bit for LPIT1 debug halted mode with M7 core 0b - LPIT1 does not enter debug halted mode with CM7 1b - LPIT1 enters debug halted mode when CM7 is debug halted
19 M7_LPI2C2	Mask bit for LPI2C2 debug halted mode with M7 core 0b - LPI2C2 does not enter debug halted mode with CM7 1b - LPI2C2 enters debug halted mode when CM7 is debug halted
18 M7_LPI2C1	Mask bit for LPI2C1 debug halted mode with M7 core 0b - LPI2C1 does not enter debug halted mode with CM7 1b - LPI2C1 enters debug halted mode when CM7 is debug halted
17 M7_EDMA3	Mask bit for EDMA3 debug halted mode with M7 core 0b - EDMA3 does not enter debug halted mode with CM7 1b - EDMA3 enters debug halted mode when CM7 is debug halted
16 M7_CAN1	Mask bit for CAN1 debug halted mode with M7 core 0b - CAN1 does not enter debug halted mode with CM7 1b - CAN1 enters debug halted mode when CM7 is debug halted
15 M33_I3C1	Mask bit for I3C1 debug halted mode with M33 core

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - I3C1 does not enter debug halted mode with CM33 1b - I3C1 enters debug halted mode when CM33 is debug halted
14 M33_CAN3	Mask bit for CAN3 debug halted mode with M33 core 0b - CAN3 does not enter debug halted mode with CM33 1b - CAN3 enters debug halted mode when CM33 is debug halted
13 M33_GPT1	Mask bit for GPT1 debug halted mode with M33 core 0b - GPT1 does not enter debug halted mode with CM33 1b - GPT1 enters debug halted mode when CM33 is debug halted
12 M33_WDOG2	Mask bit for WDOG2 debug halted mode with M33 core 0b - WDOG2 does not enter debug halted mode with CM33 1b - WDOG2 enters debug halted mode when CM33 is debug halted
11 M33_WDOG1	Mask bit for WDOG1 debug halted mode with M33 core 0b - WDOG1 does not enter debug halted mode with CM33 1b - WDOG1 enters debug halted mode when CM33 is debug halted
10 M33_TPM2	Mask bit for TPM2 debug halted mode with M33 core 0b - TPM2 does not enter debug halted mode with CM33 1b - TPM2 enters debug halted mode when CM33 is debug halted
9 M33_TPM1	Mask bit for TPM1 debug halted mode with M33 core 0b - TPM1 does not enter debug halted mode with CM33 1b - TPM1 enters debug halted mode when CM33 is debug halted
8 M33_SAI1	Mask bit for SAI1 debug halted mode with M33 core 0b - SAI1 does not enter debug halted mode with CM33 1b - SAI1 enters debug halted mode when CM33 is debug halted
7 M33_LPTMR1	Mask bit for LPTMR1 debug halted mode with M33 core 0b - LPTMR1 does not enter debug halted mode with CM33 1b - LPTMR1 enters debug halted mode when CM33 is debug halted
6 M33_LPSPI2	Mask bit for LPSPI2 debug halted mode with M33 core 0b - LPSPI2 does not enter debug halted mode with CM33 1b - LPSPI2 enters debug halted mode when CM33 is debug halted
5 M33_LPSPI1	Mask bit for LPSPI1 debug halted mode with M33 core 0b - LPSPI1 does not enter debug halted mode with CM33 1b - LPSPI1 enters debug halted mode when CM33 is debug halted

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 M33_LPIT1	Mask bit for LPIT1 debug halted mode with M33 core 0b - LPIT1 does not enter debug halted mode with CM33 1b - LPIT1 enters debug halted mode when CM33 is debug halted
3 M33_LPI2C2	Mask bit for LPI2C2 debug halted mode with M33 core 0b - LPI2C2 does not enter debug halted mode with CM33 1b - LPI2C2 enters debug halted mode when CM33 is debug halted
2 M33_LPI2C1	Mask bit for LPI2C1 debug halted mode with M33 core 0b - LPI2C1 does not enter debug halted mode with CM33 1b - LPI2C1 enters debug halted mode when CM33 is debug halted
1 M33_EDMA3	Mask bit for EDMA3 debug halted mode with M33 core 0b - EDMA3 does not enter debug halted mode with CM33 1b - EDMA3 enters debug halted mode when CM33 is debug halted
0 M33_CAN1	Mask bit for CAN1 debug halted mode with M33 core 0b - CAN1 does not enter debug halted mode with CM33 1b - CAN1 enters debug halted mode when CM33 is debug halted

14.6.1.4 (SSI)

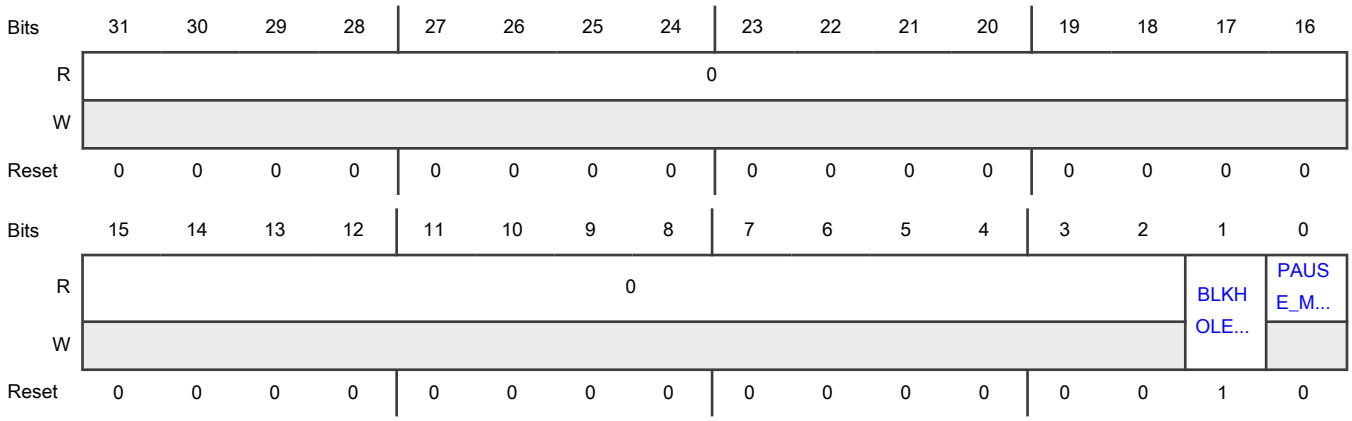
Offset

Register	Offset
SSI	1Ch

Function

SSI Master: AXI Async Bridge from AXIM to NIC400. Low power mode control.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 BLKHOLE_MODE_B	AON Domain SSI master blackhole mode 0: AON Domain SSI master will enter into blackhole mode 1: AON Domain SSI master will exit from blackhole mode 0b - AON Domain SSI master will enter into blackhole mode 1b - AON Domain SSI master will exit from blackhole mode
0 PAUSE_MODE	AON Domain SSI master pause mode 0: AON Domain SSI master is not in pause mode 1: AON Domain SSI master is in pause mode 0b - AON Domain SSI master is not in pause mode 1b - AON Domain SSI master is in pause mode

14.6.1.5 SAI1 MCLK control register (SAI1_MCLK_CTRL)

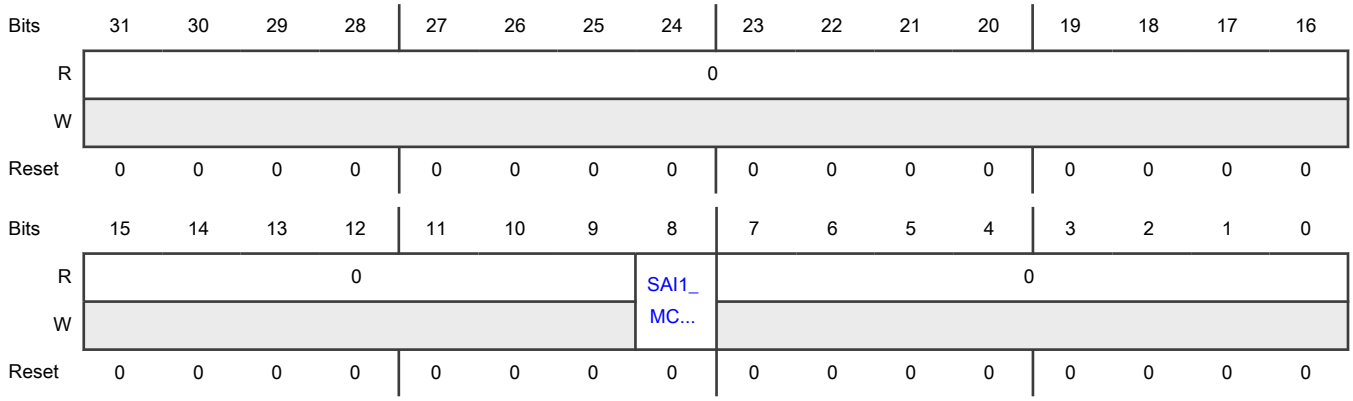
Offset

Register	Offset
SAI1_MCLK_CTRL	20h

Function

This register provides control of SAI1 MCLK direction

Diagram



Fields

Field	Function
31-9 —	Reserved
8 SAI1_MCLK_DIR	SAI1_MCLK IO direction control. IOMUX need select SAI1 MCLK function 0b - SAI1_MCLK is input signal 1b - SAI1_MCLK is output signal
7-0 —	Reserved

14.6.1.6 DCDC status register (DCDC_STATUS)

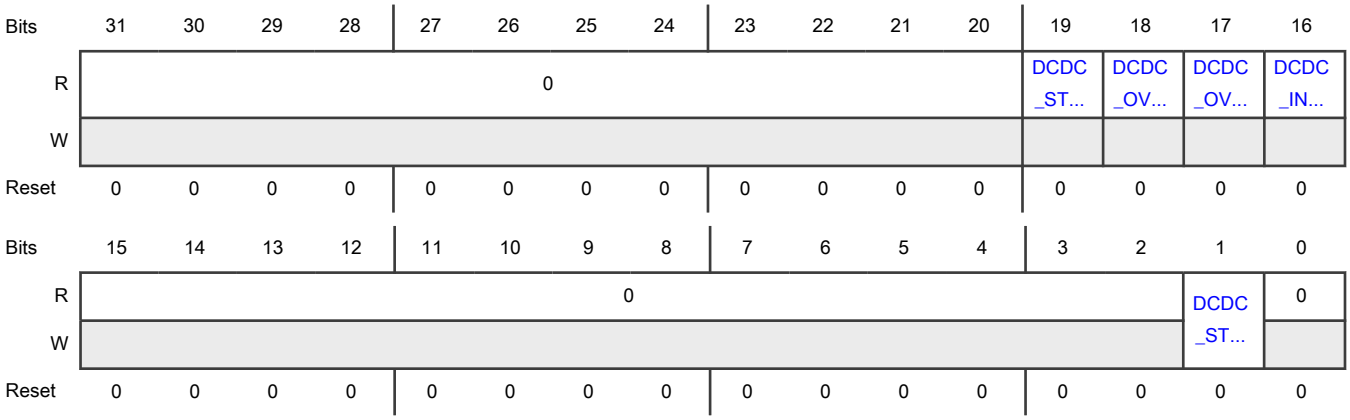
Offset

Register	Offset
DCDC_STATUS	24h

Function

This register records DCDC status. Failure statuses are stored in battery domain and can be read out via this register in next POR

Diagram



Fields

Field	Function
31-20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 DCDC_STS_DC_OK	DCDC status OK Status register to indicate DCDC status. 0b - DCDC is settling 1b - DCDC already settled
18 DCDC_OVER_VOL	DCDC output over voltage alert 0b - No Overvoltage on DCDC VDDLP0 or VDDLP8 output 1b - Overvoltage on DCDC VDDLP0 or VDDLP8 output
17 DCDC_OVER_CUR	DCDC output over current alert 0b - No Overcurrent on DCDC output 1b - Overcurrent on DCDC output
16 DCDC_IN_LOW_VOL	DCDC_IN low voltage detect 0b - Voltage on DCDC_IN is higher than 2.6V 1b - Voltage on DCDC_IN is lower than 2.6V
15-2 —	Reserved
1 DCDC_STATUS_CAPT_CLR	DCDC captured status clear Write logic 1 to clear the 3 bits of DCDC captured status: DCDC_OVER_VOL, DCDC_OVER_CUR, and DCDC_IN_LOW_VOL. 0b - No change 1b - Clear the 3 bits of DCDC captured status: DCDC_OVER_VOL, DCDC_OVER_CUR, and DCDC_IN_LOW_VOL
0 —	Reserved

14.6.1.7 Fuse access disable register (FUSE_ACC_DIS)

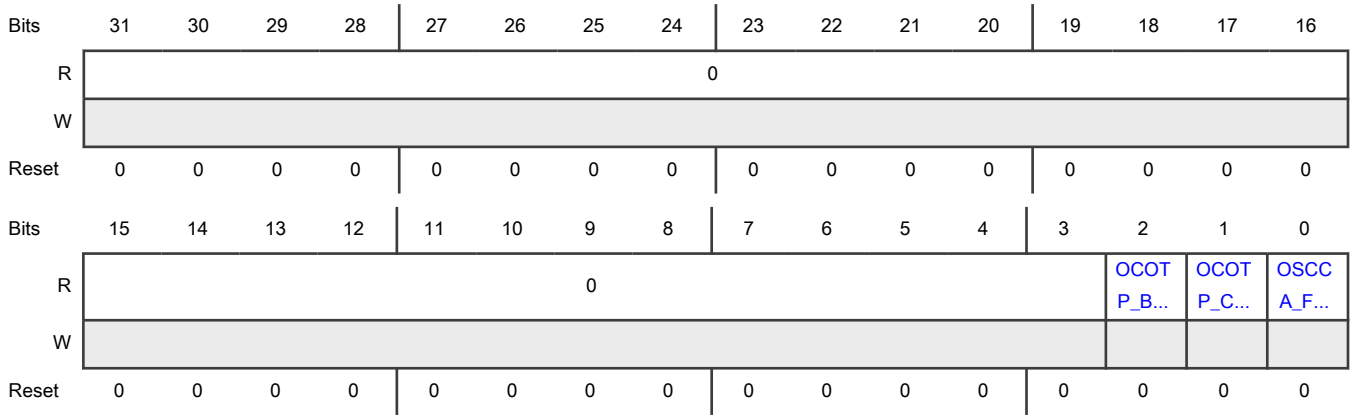
Offset

Register	Offset
FUSE_ACC_DIS	28h

Function

This register provides status bit to detect that SoC read access is not permitted of the OCOTP's shadow SRAM

Diagram



Fields

Field	Function
31-3 —	Reserved
2 OCOTP_BUSY	OCOTP busy flag This bit indicates whether OCOTP is busy. When it's busy fuse value is not readable. 0b - OCOTP is not busy 1b - OCOTP is busy
1 OCOTP_CALIBRATED	Fuse calibrate flag This bit indicates whether OCOTP calibration is done. 0b - OCOTP is not calibrated 1b - OCOTP is calibrated
0 OSCCA_FUSE_READ_DIS	Fuse read disable flag This bit indicates whether SoC is allowed to read OCOTP's shadow RAM. 0b - Read is allowed 1b - Read is not allowed

14.6.1.8 M33 NMI interrupt clear register (M33_NMI_CLR)

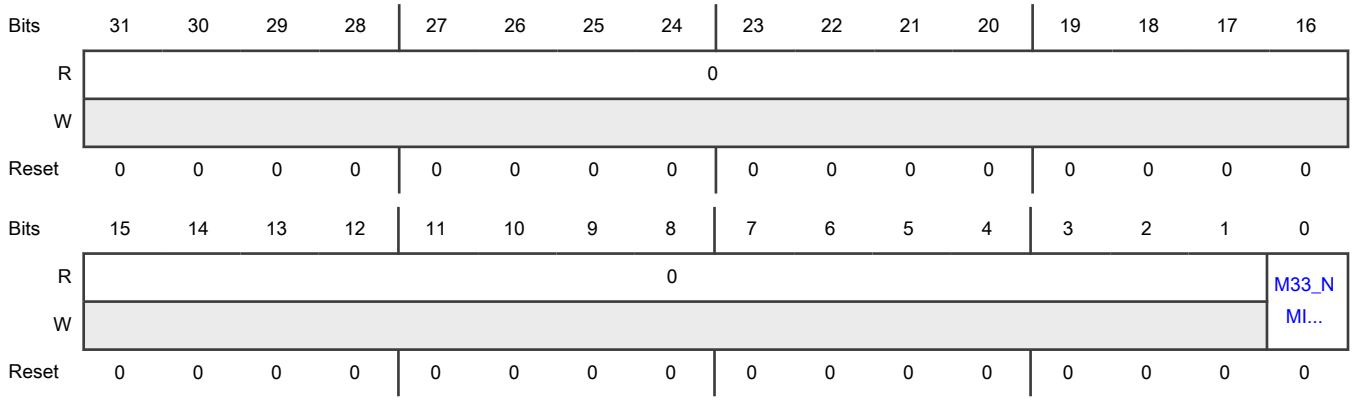
Offset

Register	Offset
M33_NMI_CLR	2Ch

Function

This register contains the M33_NMI_CLEAR bit used to clear the interrupt from the NMI input.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 M33_NMI_CLEAR	Clear CM33 NMI holding register The NMI input from IO will be held internally until this bit is set to 1. Note that this bit needs software to clear. Set it to 1 to clear the interrupt and following with writing it to 0 to complete the operation.

14.6.1.9 I3C1 async wakeup control register (I3C1_ASYNC_WAKEUP_CTRL)

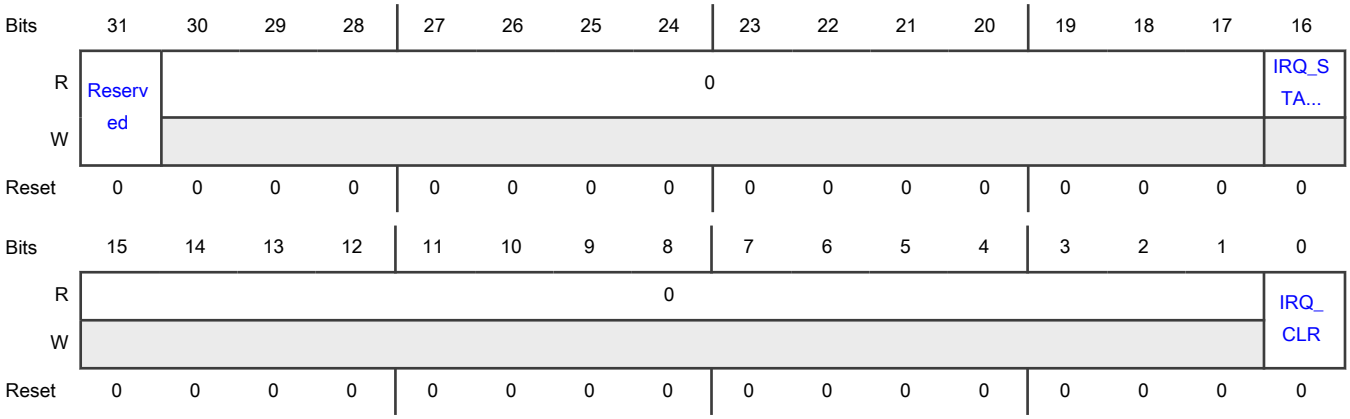
Offset

Register	Offset
I3C1_ASYNC_WAKEUP_CTRL	30h

Function

This register contains status and control for I3C1 async wakeup interrupt.

Diagram



Fields

Field	Function
31 —	Reserved
30-17 —	Reserved
16 IRQ_STATUS	Async wakeup interrupt status This bit indicates the status of I3C1 async wakeup interrupt for slave mode. The interrupt is enabled by bit 9 of I3C1 interrupt enable register of the I3C chapter. I3C master mode doesn't support async wakeup. 0b - Interrupt not asserted 1b - Interrupt asserted
15-1 —	Reserved
0 IRQ_CLR	Async wakeup interrupt clear This bit is to clear I3C1 async wakeup interrupt. The interrupt will be held until this bit is set to 1. Note that this bit needs software to clear. Set it to 1 to clear the interrupt and following with writing it to 0 to complete the operation.

14.6.1.10 Miscellaneous control register of IO (MISC_IO_CTRL)

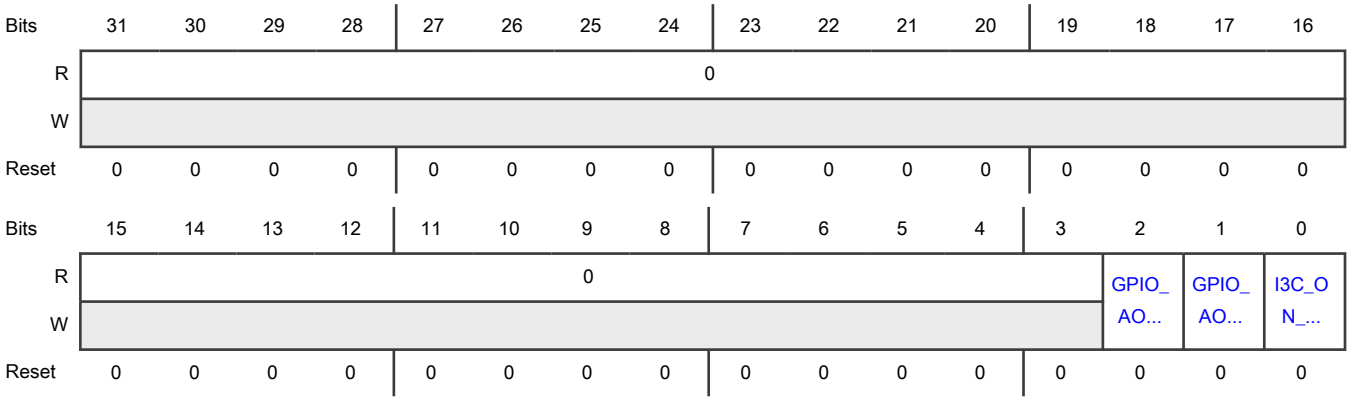
Offset

Register	Offset
MISC_IO_CTRL	34h

Function

This register provides controls to IO supply voltage detector and IO voltage range of particular IO banks.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 GPIO_AON_LOW_RANGE	GPIO_AON IO bank supply voltage range selection

GPIO_AON_HIGH_RANGE	GPIO_AON_LOW_RANGE	Mode
0	0	GPIO_AON_xx IO will work in continuous range mode with supply voltage in 1.71v-3.6v
0	1	GPIO_AON_xx IO will work in low range mode with supply voltage in 1.71v-1.98v
1	0	GPIO_AON_xx IO will work in high range mode with supply voltage in 3v-3.6v
1	1	Not allowed

Field	Function
1 GPIO_AON_HIGH_RANGE	GPIO_AON IO bank supply voltage range selection

GPIO_AON_HIGH_RANGE	GPIO_AON_LOW_RANGE	Mode
0	0	GPIO_AON_xx IO will work in continuous range mode with supply voltage in 1.71v-3.6v
0	1	GPIO_AON_xx IO will work in low range mode with supply voltage in 1.71v-1.98v

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>GPIO_AON_HIGH_RANGE</th><th>GPIO_AON_LOW_RANGE</th><th>Mode</th></tr>
<tr><td>1</td><td>0</td><td>GPIO_AON_xx IO will work in high range mode with supply voltage in 3v-3.6v</td></tr>
<tr><td>1</td><td>1</td><td>Not allowed</td></tr>
</table>
</td></tr>
<tr><td>0
I3C_ON_CHIP_STRONG_PULL_DIS</td><td>Disable I3C on-chip strong pull for I3C1
0b - On-chip strong pull is enabled
1b - On-chip strong pull is disabled</td></tr>
</table>

Chapter 15
Block Control - Always-on Domain, Secure (BLK_CTRL_S_AON)

15.1 Overview

Block control consists of a group of registers that provide control or status bits for other peripherals on the chip. These registers are to provide chip level control outside the peripherals. Block control covers peripherals of the corresponding mix. BLK_CTRL_NS_AONMIX and BLK_CTRL_S_AONMIX are in AONMIX.

Block Control Secure AONMIX has configuration controls that can potentially have security implications. Block Control Secure AONMIX memory slot is protected by TRDC configuration. The registers can only be accessed by TrustZone Secure World.

15.2 Features

Block Control Secure AONMIX has the following features:

- AXBS_AON priority and arbitration controls
- CM33 and CM7 NVIC interrupt masking
- EdgeLock interrupt and reset masking
- CM33 INITVTOR
- CM7 INITVTOR
- EdgeLock low power handshake enable
- IOMUXC domain control

15.3 Functional description

This block is a register container and doesn't have any function itself. The regisers it contained cover:

- AXBS_AON priority and arbitration controls
- CM33 and CM7 NVIC interrupt masking
- EdgeLock interrupt and reset masking
- CM33 INITVTOR
- CM7 INITVTOR
- EdgeLock low power handshake enable
- IOMUXC domain control

15.3.1 Clocks

Block Control Secure AONMIX uses BUS_AON_CLK for register read and write.

15.3.2 Reset

Block Control Secure AONMIX resets together with the AONMIX.

15.4 Memory Map and register definition

This section includes the Block Control Secure AONMIX module memory map and detailed descriptions of registers.

15.4.1 Block Control Secure AONMIX register descriptions

15.4.1.1 AONMIX Block Control for Secure World memory map

BLK_CTRL_S_AONMIX base address: 444F_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	CM33_IRQ_MASK0 (CM33_IRQ_MASK0)	32	RW	FFFF_FFFFh
4h	CM33 IRQ MASK1 (CM33_IRQ_MASK1)	32	RW	FFFF_FFFFh
8h	CM33_IRQ_MASK2 (CM33_IRQ_MASK2)	32	RW	FFFF_FFFFh
Ch	CM33_IRQ_MASK3 (CM33_IRQ_MASK3)	32	RW	FFFF_FFFFh
10h	CM33_IRQ_MASK4 (CM33_IRQ_MASK4)	32	RW	FFFF_FFFFh
14h	CM33_IRQ_MASK5 (CM33_IRQ_MASK5)	32	RW	FFFF_FFFFh
18h	CM33_IRQ_MASK6 (CM33_IRQ_MASK6)	32	RW	FFFF_FFFFh
1Ch	CM33_IRQ_MASK7 (CM33_IRQ_MASK7)	32	RW	FFFF_FFFFh
20h	CM7_IRQ_MASK0 (CM7_IRQ_MASK0)	32	RW	FFFF_FFFFh
24h	CM7_IRQ_MASK1 (CM7_IRQ_MASK1)	32	RW	FFFF_FFFFh
28h	CM7_IRQ_MASK2 (CM7_IRQ_MASK2)	32	RW	FFFF_FFFFh
2Ch	CM7_IRQ_MASK3 (CM7_IRQ_MASK3)	32	RW	FFFF_FFFFh
30h	CM7_IRQ_MASK4 (CM7_IRQ_MASK4)	32	RW	FFFF_FFFFh
34h	CM7_IRQ_MASK5 (CM7_IRQ_MASK5)	32	RW	FFFF_FFFFh
38h	CM7_IRQ_MASK6 (CM7_IRQ_MASK6)	32	RW	FFFF_FFFFh
3Ch	CM7_IRQ_MASK7 (CM7_IRQ_MASK7)	32	RW	FFFF_FFFFh
58h	EdgeLock reset request mask (EDGELOCK_RESET_REQ_MASK)	32	RW	0000_0192h
5Ch	EdgeLock IRQ request mask (EDGELOCK_IRQ_MASK)	32	RW	0000_01F7h
60h	M33 Configuration (M33_CFG)	32	RW	0000_0004h
64h	M33 INITSVTOR (M33_INITSVTOR)	32	RW	0000_0000h
68h	M33 INITNSVTOR (M33_INITNSVTOR)	32	RW	0000_0000h
80h	M7 Configuration (M7_CFG)	32	RW	0000_0010h
90h	AXBS_AON_CTRL (AXBS_AON_CTRL)	32	RW	0000_0001h
100h	DAP Access Sticky Bit (DAP_ACCESS_STKYBIT)	32	RW	0000_0000h
110h	Low power handshake enable (LP_HANDSHAKE)	32	RW	0000_0000h
114h	EdgeLock halt status (EDGELOCK_HALT_ST)	32	RW	0000_0000h
120h	ECC memory hardware initialization (ECC_MEM_INIT)	32	R	0000_0000h
148h	IOMUXC domain configure (IOMUXC_DOMAIN_CFG)	32	RW	0000_0420h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
14Ch	IOMUXC_AON domain configure (IOMUXC_AON_DOMAIN_CFG)	32	RW	0000_0420h
154h	NMI control (NMI_CTRL)	32	RW	0000_0000h
158h	s401_ipi_noclk_ref1 clear control (S401_NOCLK_CLEAR_CTRL)	32	RW	0000_0000h

15.4.1.2 CM33_IRQ_MASK0 (CM33_IRQ_MASK0)

Offset

Register	Offset
CM33_IRQ_MASK0	0h

Function

CM33 IRQ MASK0, please refer Interrupt Assignments for the usage of each bit

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 M	CM33 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.3 CM33 IRQ MASK1 (CM33_IRQ_MASK1)

Offset

Register	Offset
CM33_IRQ_MASK1	4h

Function

Please refer to Interrupts Assignments spreadsheet usage of each bit.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 M	CM33 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.4 CM33_IRQ_MASK2 (CM33_IRQ_MASK2)

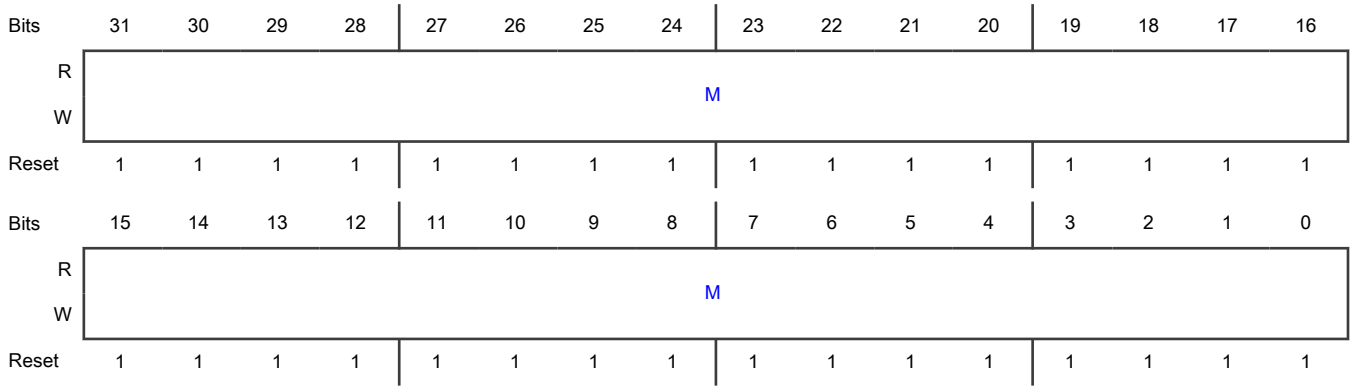
Offset

Register	Offset
CM33_IRQ_MASK2	8h

Function

CM33 IRQ MASK2, please refer Interrupt Assignments for the usage of each bit

Diagram



Fields

Field	Function
31-0 M	CM33 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.5 CM33_IRQ_MASK3 (CM33_IRQ_MASK3)

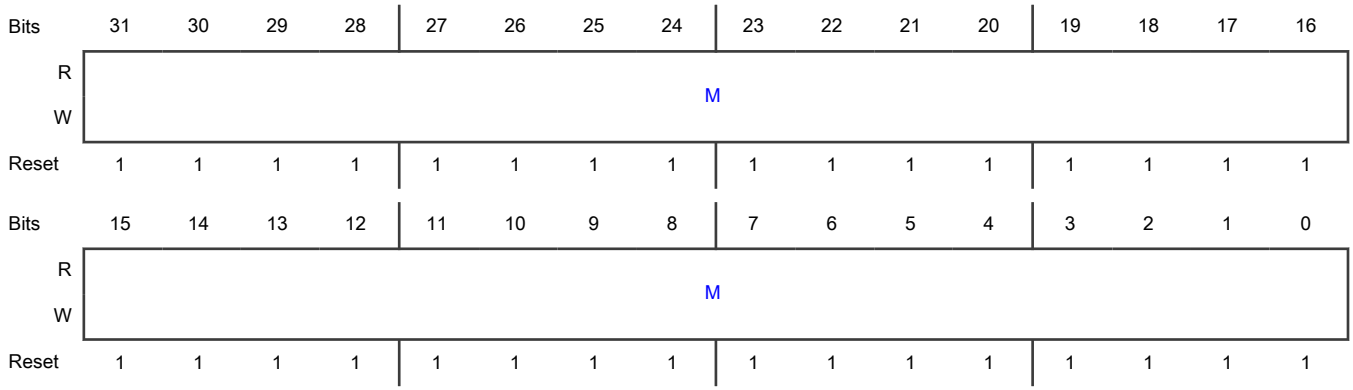
Offset

Register	Offset
CM33_IRQ_MASK3	Ch

Function

CM33 IRQ MASK3, please refer Interrupt Assignments for the usage of each bit

Diagram



Fields

Field	Function
31-0 M	CM33 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.6 CM33_IRQ_MASK4 (CM33_IRQ_MASK4)

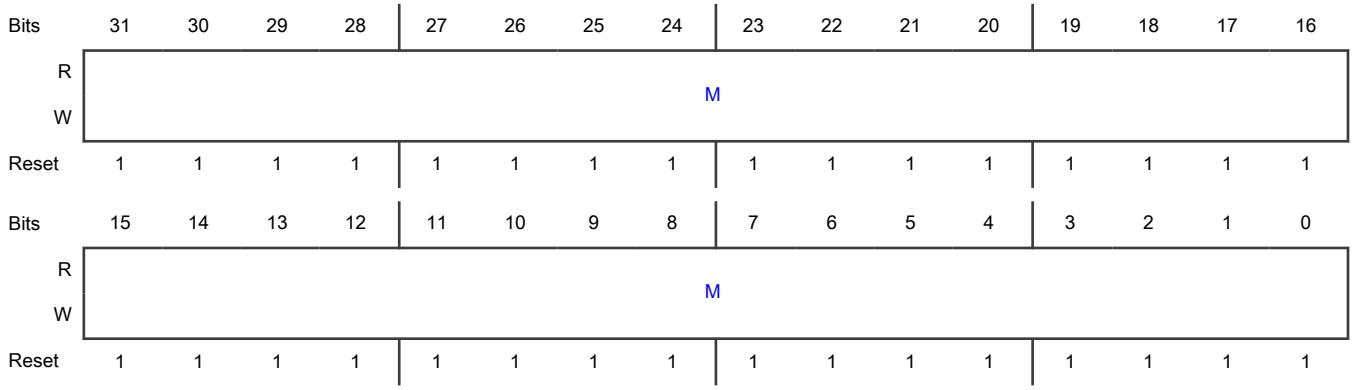
Offset

Register	Offset
CM33_IRQ_MASK4	10h

Function

CM33 IRQ MASK4, please refer Interrupt Assignments for the usage of each bit

Diagram



Fields

Field	Function
31-0 M	CM33 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.7 CM33_IRQ_MASK5 (CM33_IRQ_MASK5)

Offset

Register	Offset
CM33_IRQ_MASK5	14h

Function

CM33 IRQ MASK5, please refer Interrupt Assignments for the usage of each bit

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 M	CM33 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.8 CM33_IRQ_MASK6 (CM33_IRQ_MASK6)

Offset

Register	Offset
CM33_IRQ_MASK6	18h

Function

CM33 IRQ MASK6, please refer Interrupt Assignments for the usage of each bit

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 M	CM33 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.9 CM33_IRQ_MASK7 (CM33_IRQ_MASK7)

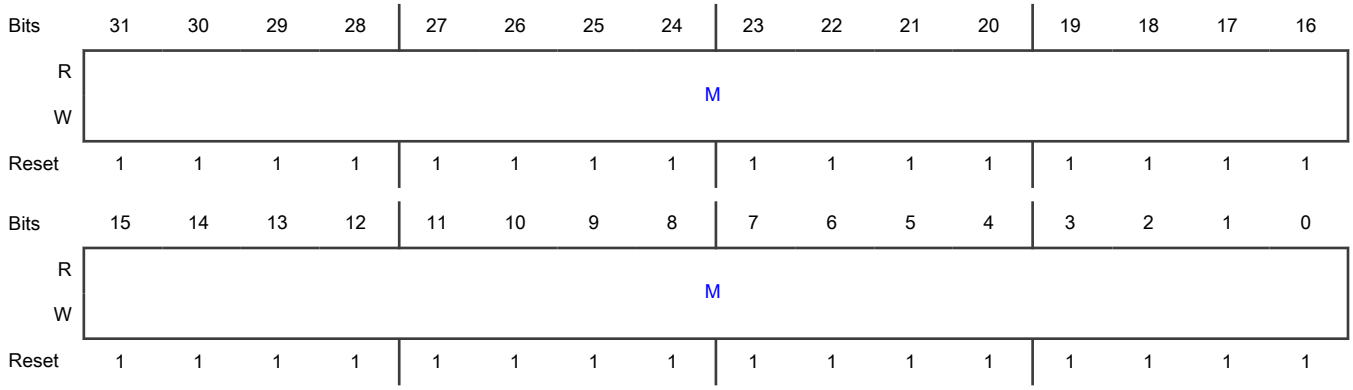
Offset

Register	Offset
CM33_IRQ_MASK7	1Ch

Function

CM33 IRQ MASK7, please refer Interrupt Assignments for the usage of each bit

Diagram



Fields

Field	Function
31-0 M	CM33 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.10 CM7_IRQ_MASK0 (CM7_IRQ_MASK0)

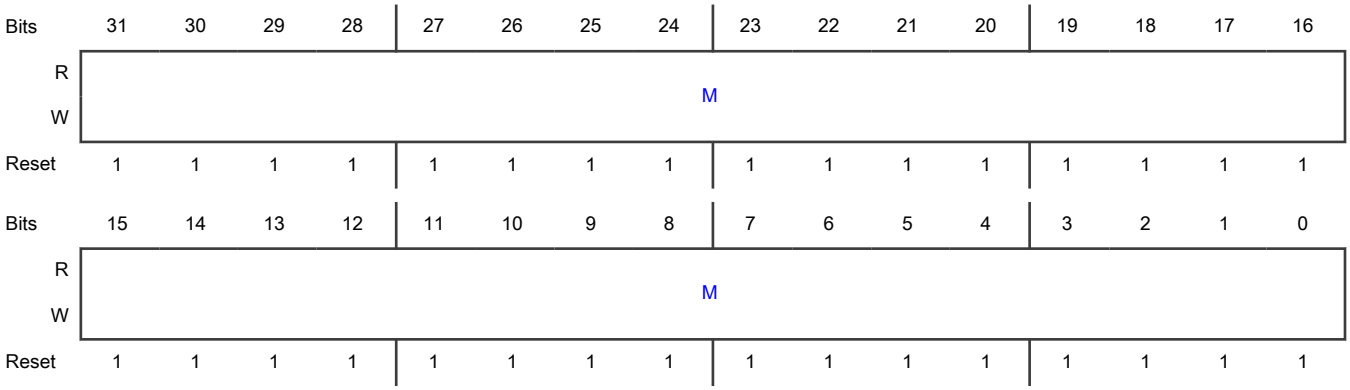
Offset

Register	Offset
CM7_IRQ_MASK0	20h

Function

CM7 IRQ MASK0, please refer Interrupt Assignments for the usage of each bit

Diagram



Fields

Field	Function
31-0 M	CM7 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.11 CM7_IRQ_MASK1 (CM7_IRQ_MASK1)

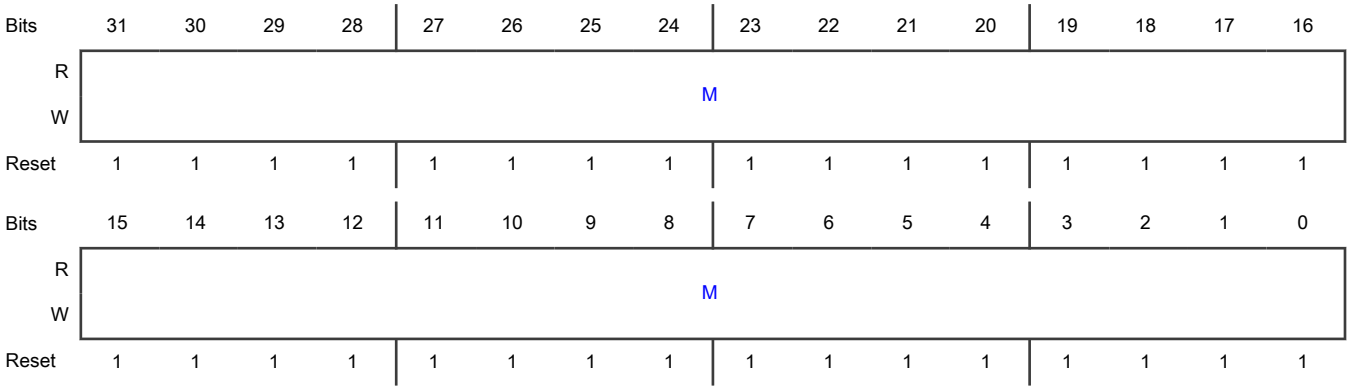
Offset

Register	Offset
CM7_IRQ_MASK1	24h

Function

CM7 IRQ MASK1, please refer Interrupt Assignments for the usage of each bit

Diagram



Fields

Field	Function
31-0 M	CM7 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.12 CM7_IRQ_MASK2 (CM7_IRQ_MASK2)

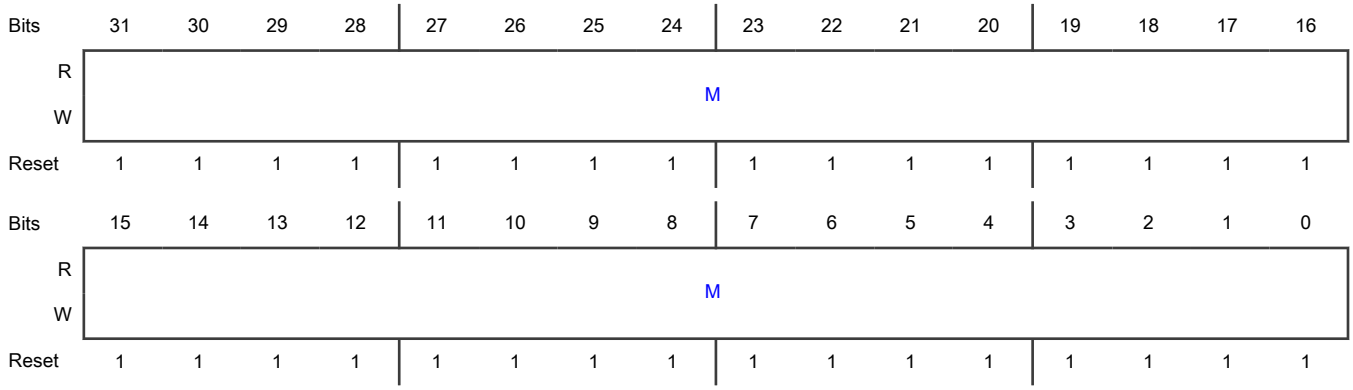
Offset

Register	Offset
CM7_IRQ_MASK2	28h

Function

CM7 IRQ MASK2, please refer Interrupt Assignments for the usage of each bit

Diagram



Fields

Field	Function
31-0 M	CM7 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.13 CM7_IRQ_MASK3 (CM7_IRQ_MASK3)

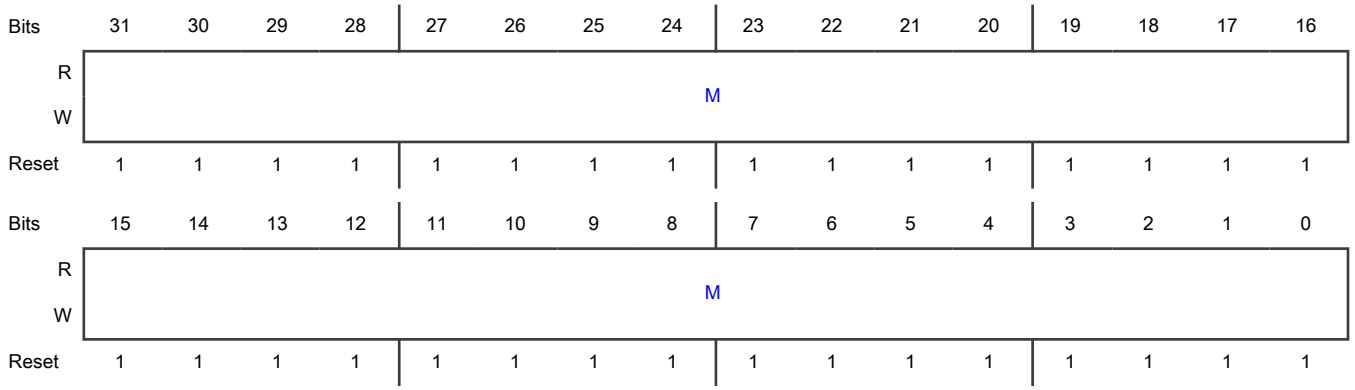
Offset

Register	Offset
CM7_IRQ_MASK3	2Ch

Function

CM7 IRQ MASK3, please refer Interrupt Assignments for the usage of each bit

Diagram



Fields

Field	Function
31-0 M	CM7 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.14 CM7_IRQ_MASK4 (CM7_IRQ_MASK4)

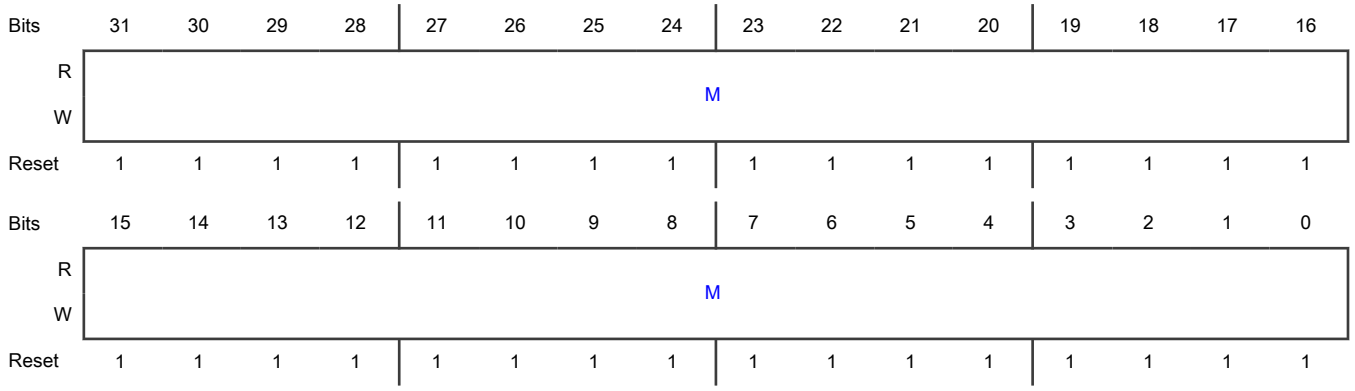
Offset

Register	Offset
CM7_IRQ_MASK4	30h

Function

CM7 IRQ MASK4, please refer Interrupt Assignments for the usage of each bit

Diagram



Fields

Field	Function
31-0 M	CM7 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.15 CM7_IRQ_MASK5 (CM7_IRQ_MASK5)

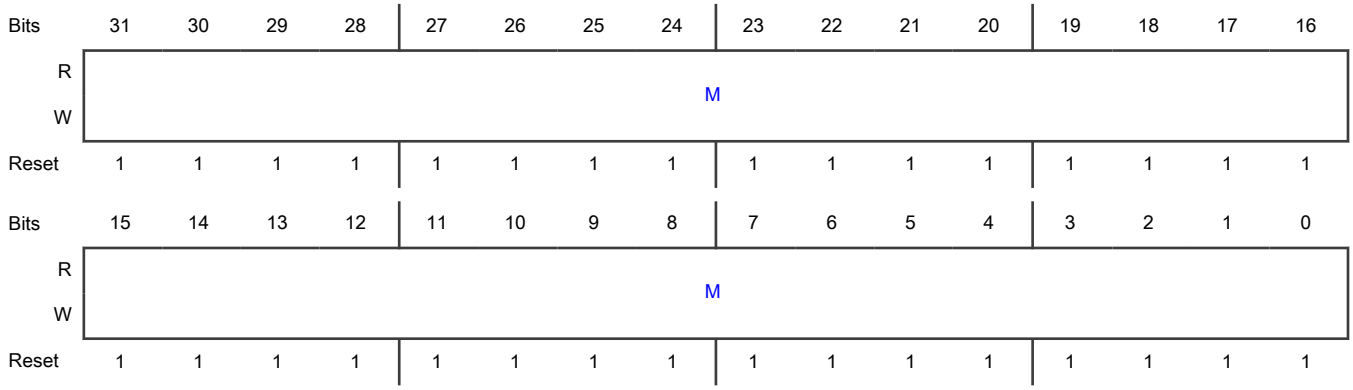
Offset

Register	Offset
CM7_IRQ_MASK5	34h

Function

CM7 IRQ MASK5, please refer Interrupt Assignments for the usage of each bit

Diagram



Fields

Field	Function
31-0 M	CM7 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.16 CM7_IRQ_MASK6 (CM7_IRQ_MASK6)

Offset

Register	Offset
CM7_IRQ_MASK6	38h

Function

CM7 IRQ MASK6, please refer Interrupt Assignments for the usage of each bit

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 M	CM7 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.17 CM7_IRQ_MASK7 (CM7_IRQ_MASK7)

Offset

Register	Offset
CM7_IRQ_MASK7	3Ch

Function

CM7 IRQ MASK7, please refer Interrupt Assignments for the usage of each bit

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 M	CM7 IRQ MASK 0000_0000_0000_0000_0000_0000_0000_0000b - Mask IRQ 0000_0000_0000_0000_0000_0000_0000_0001b - No Mask IRQ

15.4.1.18 EdgeLock reset request mask (EDGELOCK_RESET_REQ_MASK)

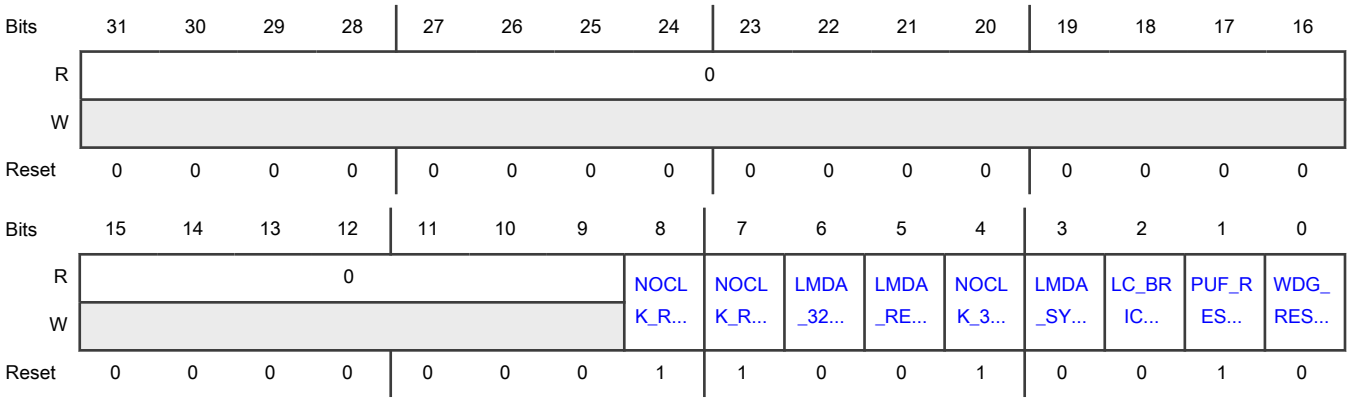
Offset

Register	Offset
EDGELOCK_RESET_REQ_MASK	58h

Function

This register provides mask bits for the reset requests from EdgeLock to SRC. When the mask bit is set 1 the corresponding reset will not be sent to SRC.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 NOCLK_REF2	EdgeLock OSC 24Mhz clock loss reset mask 0b - Unmask reset 1b - Mask reset
7	EdgeLock CM33 root clock loss reset mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
NOCLK_REF1	0b - Unmask reset 1b - Mask reset
6 LMDA_32K_RESET_REQ	EdgeLock LMDA reset request from 32k clock domain mask 0b - Unmask reset 1b - Mask reset
5 LMDA_RESET_REQ	EdgeLock LMDA reset request mask 0b - Unmask reset 1b - Mask reset
4 NOCLK_32K	EdgeLock 32k clock loss reset mask 0b - Unmask reset 1b - Mask reset
3 LMDA_SYS_FAIL	EdgeLock system failure reset mask 0b - Unmask reset 1b - Mask reset
2 LC_BRICKED	EdgeLock LMDA life cycle bricked reset mask 0b - Unmask reset 1b - Mask reset
1 PUF_RESET	EdgeLock PUF reset mask 0b - Unmask reset 1b - Mask reset
0 WDG_RESET	EdgeLock Wdog reset mask 0b - Unmask reset 1b - Mask reset

15.4.1.19 EdgeLock IRQ request mask (EDGELOCK_IRQ_MASK)

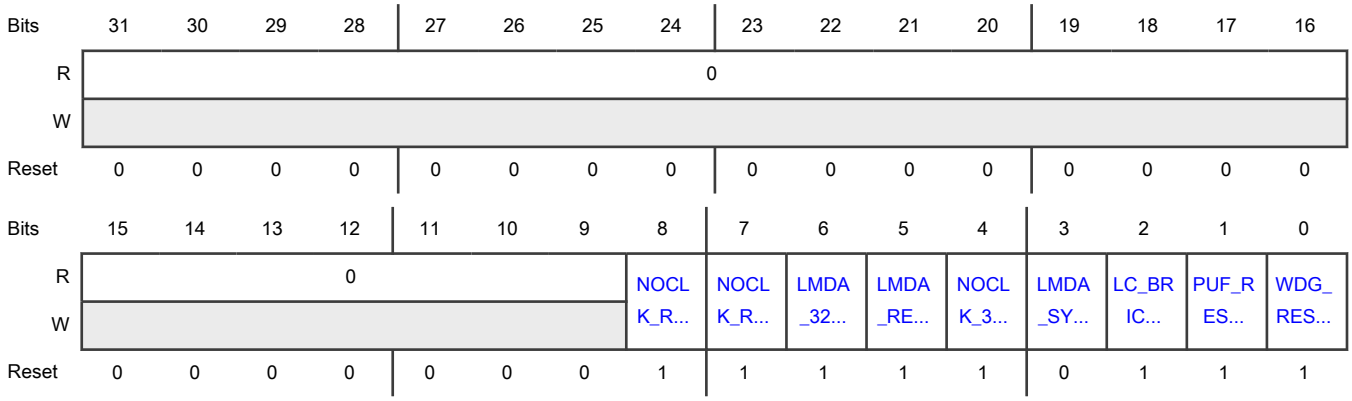
Offset

Register	Offset
EDGELOCK_IRQ_MASK	5Ch

Function

This register provides mask bits for the interrupt requests from EdgeLock to ARM core. When the mask bit is set 1 the corresponding interrupt will not be sent to ARM core.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 NOCLK_REF2	EdgeLock OSC 24Mhz clock loss interrupt mask 0b - Unmask interrupt 1b - Mask interrupt
7 NOCLK_REF1	EdgeLock cm33 root clock loss interrupt mask 0b - Unmask interrupt 1b - Mask interrupt
6 LMDA_32K_RESET_REQ	EdgeLock LMDA reset request from 32k clock domain interrupt mask 0b - Unmask interrupt 1b - Mask interrupt
5 LMDA_RESET_REQ	EdgeLock LMDA reset request interrupt mask 0b - Unmask interrupt 1b - Mask interrupt
4 NOCLK_32K	EdgeLock 32k clock loss interrupt mask 0b - Unmask interrupt 1b - Mask interrupt
3 LMDA_SYS_FAIL	EdgeLock system failure interrupt mask 0b - Unmask interrupt 1b - Mask interrupt
2 LC_BRICKED	EdgeLock LMDA life cycle bricked interrupt mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Unmask interrupt 1b - Mask interrupt
1 PUF_RESET	EdgeLock PUF reset interrupt mask 0b - Unmask interrupt 1b - Mask interrupt
0 WDG_RESET	EdgeLock Wdog reset interrupt mask 0b - Unmask interrupt 1b - Mask interrupt

15.4.1.20 M33 Configuration (M33_CFG)

Offset

Register	Offset
M33_CFG	60h

Function

This register is used for M33 core configuration.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										CORE CLK...	TCM_SIZE		WAIT	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

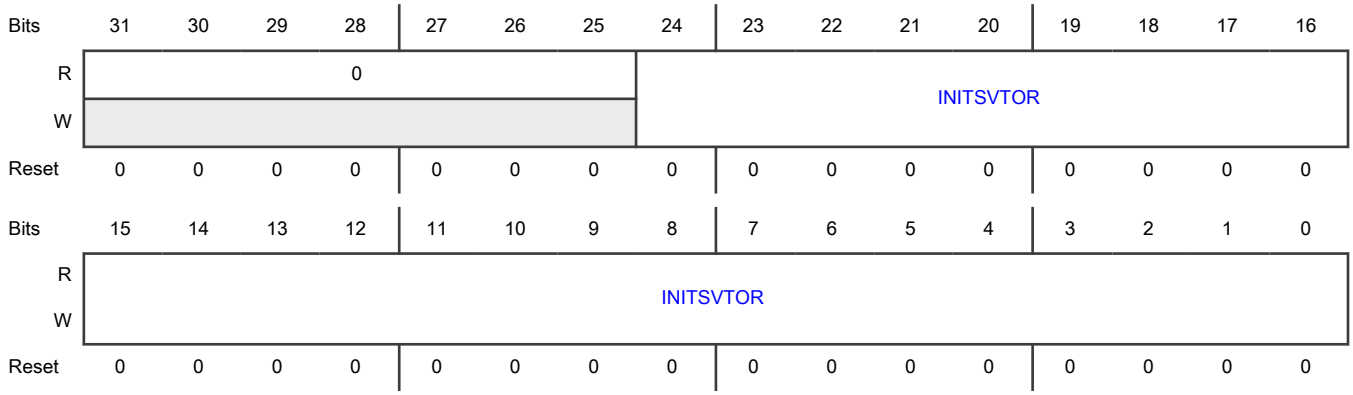
Field	Function
5 CORECLK_FORCE_ON	Force CM33 core clock on in WAIT mode In some debug case it may be desired to keep CPU clock running in WAIT mode after WFI instruction, this bit can force the CM33 core clock on in such case. 0b - CM33 core clock is off in WAIT mode 1b - CM33 core clock is on in WAIT mode
4-3 TCM_SIZE	M33 TCM SIZE When M33 TCM SIZE efuse bits are not blown, this field can be used to configure M33 TCM partition. Write to this field should only be performed when the Cortex-M33 core is in reset. 00b - Regular TCM, 128KB Code TCM and 128KB Sys TCM 01b - Double Code TCM, 256KB Code TCM 10b - Double Sys TCM, 256KB Sys TCM 11b - Reserved
2 WAIT	M33 CPU WAIT When HIGH out of reset, forces the core into a quiescent state. The core boot-up sequence and instruction execution is delayed until this signal is driven LOW. During this time, the processor does not perform any memory accesses. Debugger accesses continue when this signal is HIGH. CPUWAIT has no effect if driven HIGH when the processor is running
1-0 —	Reserved

15.4.1.21 M33 INITSVTOR (M33_INITSVTOR)

Offset

Register	Offset
M33_INITSVTOR	64h

Diagram



Fields

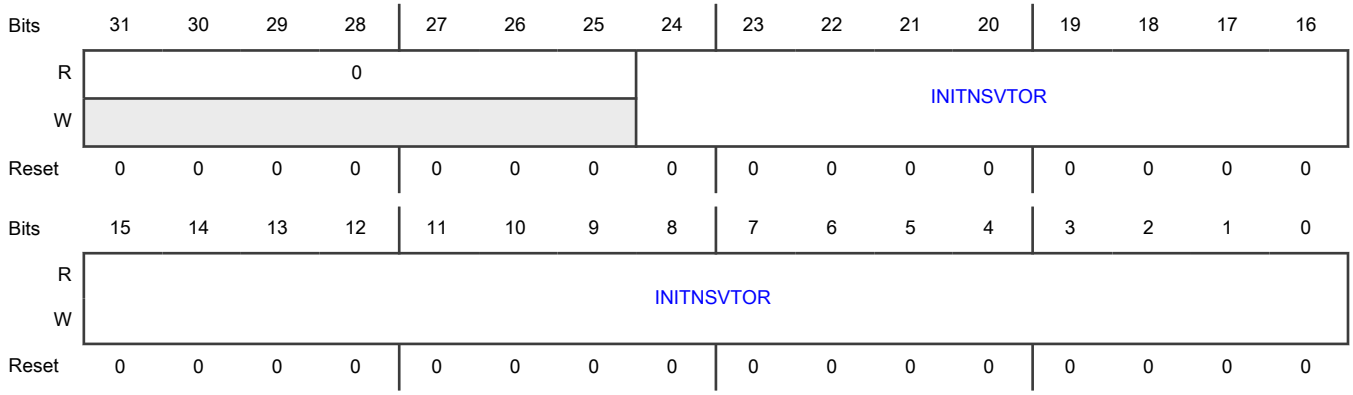
Field	Function
31-25 —	Reserved
24-0 INITSVTOR	M33 INITSVTOR M33 INITSVTOR[31:7]. This register stores value for the vector base address bit field of the CM33 Secure Vector Table Offset Register, VTOR_S.TBLOFF[31:7], for Boot ROM usage.

15.4.1.22 M33 INITNSVTOR (M33_INITNSVTOR)

Offset

Register	Offset
M33_INITNSVTOR	68h

Diagram



Fields

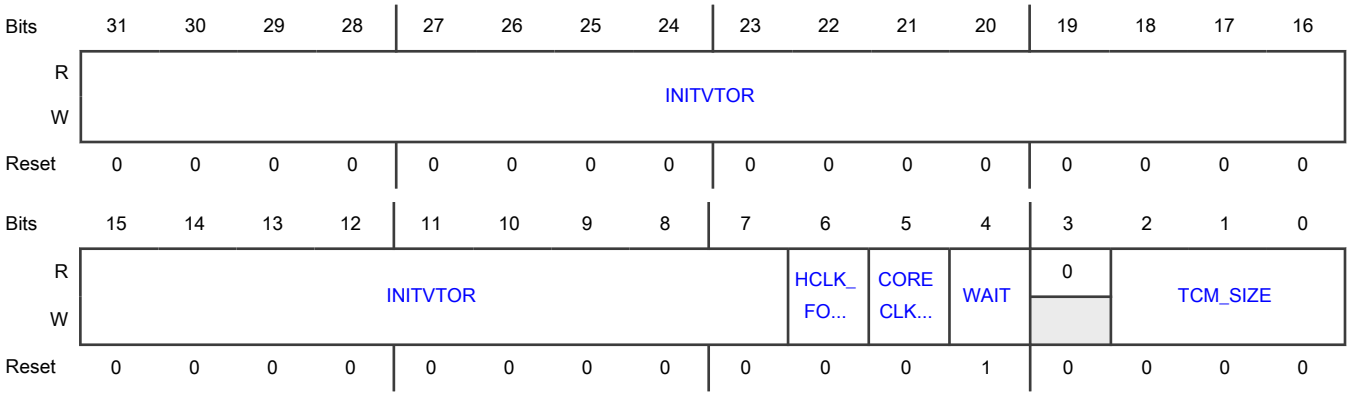
Field	Function
31-25 —	Reserved
24-0 INITNSVTOR	M33 INITNSVTOR M33 INITNSVTOR[31:7]. This register stores value for the vector base address bit field of the CM33 Non-secure Vector Table Offset Register, VTOR_NS.TBLOFF[31:7], for Boot ROM usage.

15.4.1.23 M7 Configuration (M7_CFG)

Offset

Register	Offset
M7_CFG	80h

Diagram



Fields

Field	Function
31-7 INITVTOR	M7 INITVTOR[31:7]. Vector table offset register out of reset. See the ARM v7-M Architecture Reference Manual for more information about the vector table offset register (VTOR).
6 HCLK_FORCE_ON	CM7 platform AHB clock enable This bitfield determines whether or not the AHB clock is running when CM7 is sleeping. If the AHB clock is not enabled with this bit, the TCM is not accessible. 0: AHB clock is not running (gated) when CM7 is sleeping and TCM is not accessible 1 AHB clock is running (enabled) when CM7 is sleeping and TCM is accessible
5	Force CM7 core clock on in WAIT mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
CORECLK_FORCE_ON	In some debug case it may be desired to keep CPU clock running in WAIT mode after WFI instruction, this bit can force the CM7 core clock on in such case. 0b - CM7 core clock is off in WAIT mode 1b - CM7 core clock is on in WAIT mode
4 WAIT	M7 CPUWAIT When HIGH out of reset, forces the core into a quiescent state. The core boot-up sequence and instruction execution is delayed until this signal is driven LOW. During this time, the processor does not perform any memory accesses. Debugger accesses continue when this signal is HIGH. CPUWAIT has no effect if driven HIGH when the processor is running
3 —	Reserved
2-0 TCM_SIZE	M7 TCM SIZE When M7 TCM SIZE efuse bits are not blown, this field can be used to configure M7 TCM partition. Write to this field should only be performed when the Cortex-M7 core is in reset. 000b - Regular TCM, 256KB ITCM and 256KB DTCM 001b - Double ITCM, 512KB ITCM 010b - Double DTCM, 512KB DTCM 100b - HALF ITCM, 128KB ITCM and 384KB DTCM 101b - HALF DTCM, 384KB ITCM and 128KB DTCM 110b - Reserved 111b - Reserved

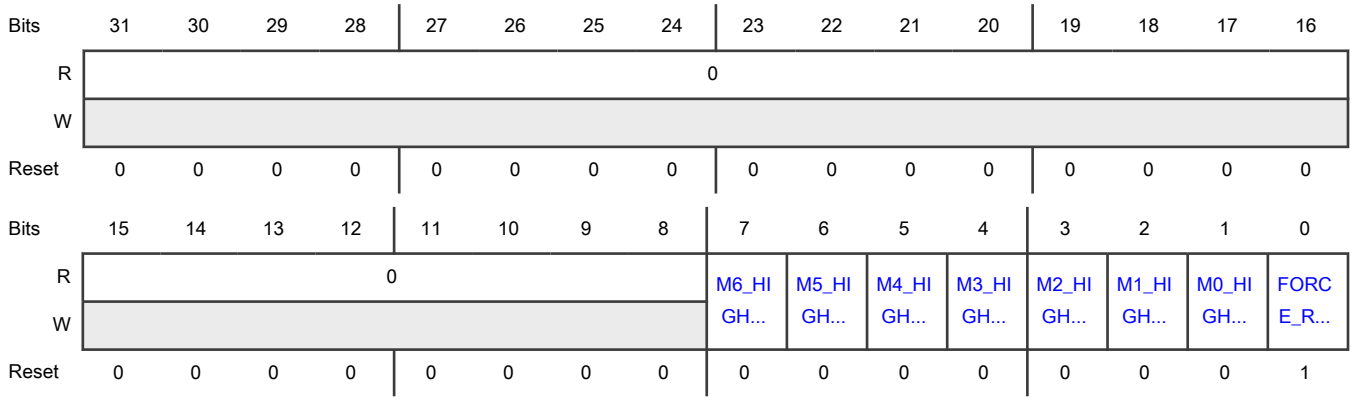
15.4.1.24 AXBS_AON_CTRL (AXBS_AON_CTRL)

Offset

Register	Offset
AXBS_AON_CTRL	90h

Function
AXBS AON Priority Control Register

Diagram



Fields

Field	Function
31-8 —	Reserved
7 M6_HIGH_PRIORITY	M6 High Priority Control Bit M6-MTR MASTER FBX BUS 0b - Default Priority 1b - High Priority
6 M5_HIGH_PRIORITY	M5 High Priority Control Bit M5-MTR MASTER MAIN BUS 0b - Default Priority 1b - High Priority
5 M4_HIGH_PRIORITY	M4 High Priority Control Bit M4-CM33 CODE BUS 0b - Default Priority 1b - High Priority
4 M3_HIGH_PRIORITY	M3 High Priority Control Bit M3-CM33 SYSTEM BUS 0b - Default Priority 1b - High Priority
3 M2_HIGH_PRIORITY	M2 High Priority Control Bit M2-EDMA1(AHB) 0b - Default Priority 1b - High Priority

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 M1_HIGH_PRIORITY	M1 High Priority Control Bit M1-SENEINEL SOC OUT 0b - Default Priority 1b - High Priority
1 M0_HIGH_PRIORITY	M0 High Priority Control Bit M0-NIC_MAIN 0b - Default Priority 1b - High Priority
0 FORCE_ROUND_ROBIN	AXBS_AON Force Round Robin 0b - Enable force round robin(default) 1b - Disable force round robin

15.4.1.25 DAP Access Sticky Bit (DAP_ACCESS_STKYBIT)

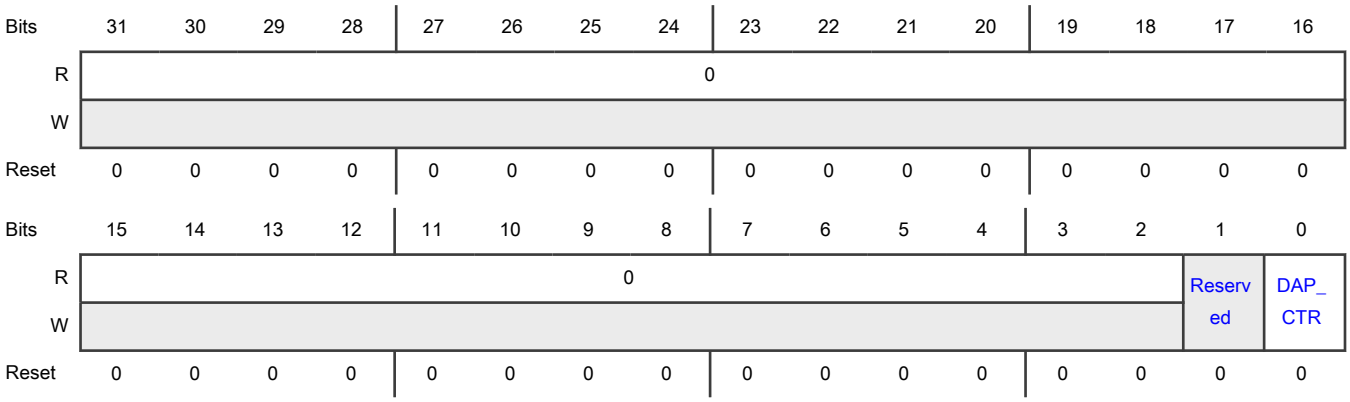
Offset

Register	Offset
DAP_ACCESS_STKYBIT	100h

Function

This register allows ROM code to receive a request whenever a debug request is received, gate the debug request until the ROM code enters a safe state to allow the debugger to continue. This register only works when fuse bit DAP_ACCESS_IRQ_STKYBIT_DISABLE[1:0] is 2'b10, otherwise there is no effect.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 —	Reserved
0 DAP_CTR	DAP access grant bit controlled by Cortex-M33 ROM, once set "1" will kept "1" unless there is a reset. 0b - DAP access is not granted by ROM 1b - DAP access is granted by ROM

15.4.1.26 Low power handshake enable (LP_HANDSHAKE)

Offset

Register	Offset
LP_HANDSHAKE	110h

Function

This register enables the handshake between EdgeLock and CCM/SRC/GPC to perform low power actions including turn on/off clock or reset of security related block.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved											OTFAD2_...	OTFAD1_...	IEE_CLK...	TRDC_CL...	CM7_CLK...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CM33_CL...	EDGELOC...	OTFAD2_...	OTFAD1_...	IEE_CLK...	TRDC_CL...	CM7_CLK...	CM33_CL...	EDGELOC...	MEGAMIX...	MEGAMIX...	WAKEUPM...	AONMIX_...	CM7_SUS...	CM7_RES...	CM33_RE...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 OTFAD2_CLK_ON_HS_EN	OTFAD2 clock on handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
19 OTFAD1_CLK_ON_HS_EN	OTFAD1 clock on handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
18 IEE_CLK_ON_HS_EN	IEE clock on handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
17 TRDC_CLK_ON_HS_EN	TRDC clock on handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
16 CM7_CLK_ON_HS_EN	CM7 clock on handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
15 CM33_CLK_ON_HS_EN	CM33 clock on handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
14 EDGELOCK_CLK_ON_HS_EN	EDGELOCK clock on handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
13 OTFAD2_CLK_OFF_HS_EN	OTFAD2 clock off handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
12 OTFAD1_CLK_OFF_HS_EN	OTFAD1 clock off handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
11 IEE_CLK_OFF_HS_EN	IEE clock off handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
10	TRDC clock off handshake enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRDC_CLK_OFF_HS_EN	0b - Handshake is not enabled 1b - Handshake is enabled
9 CM7_CLK_OFF_HS_EN	CM7 clock off handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
8 CM33_CLK_OFF_HS_EN	CM33 clock off handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
7 EDGELOCK_CLK_OFF_HS_EN	EDGELOCK clock off handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
6 MEGAMIX_LPM_HS_EN	Megamix low power mode exit reset handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
5 MEGAMIX_RESET_HS_EN	Megamix reset handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
4 WAKEUPMIX_RESET_HS_EN	Wakeupmix reset handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
3 AONMIX_RESET_HS_EN	AONMIX reset handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
2 CM7_SUSPEND_HS_EN	CM7 suspend exit reset handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
1 CM7_RESET_HS_EN	CM7 reset handshake enable 0b - Handshake is not enabled 1b - Handshake is enabled
0	CM33 reset handshake enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
CM33_RESET_HS_EN	0b - Handshake is not enabled 1b - Handshake is enabled

15.4.1.27 EdgeLock halt status (EDGELOCK_HALT_ST)

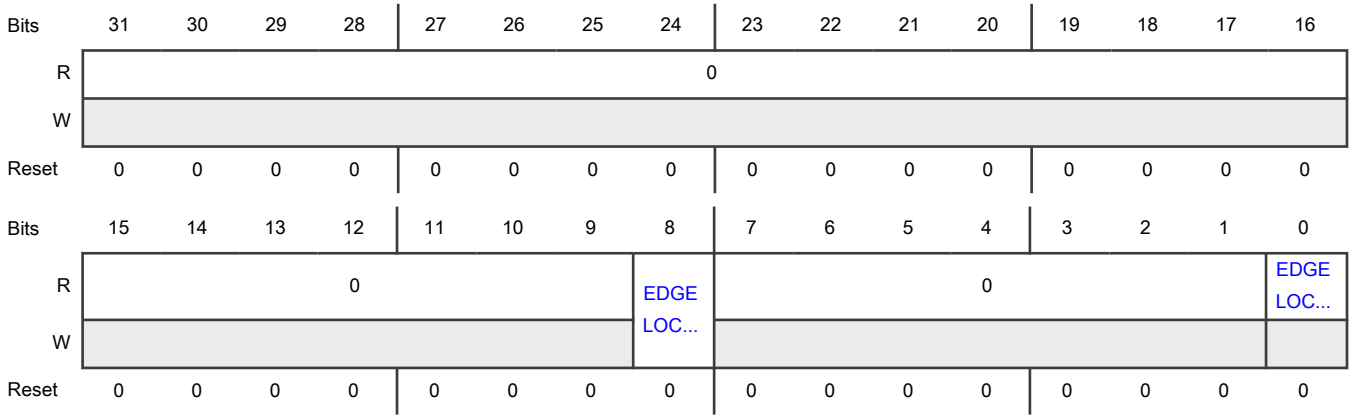
Offset

Register	Offset
EDGELOCK_HALT_ST	114h

Function

To stop clock for EdgeLock, EDGELOCK_HALK_ACK bit needs to be checked high. When EdgeLock need system to recover its clock, it will assert halt exit interrupt. EDGELOCK_HALT_EXIT_IRQ_CLR is used to clear this interrupt.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 EDGELOCK_HALT_EXIT_IRQ_CLR	EdgeLock halt exit interrupt clear 0b - Remove the clear signal. This bit is not self-clearing and need SW to clear. 1b - Clear EdgeLock halt exit interrupt
7-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 EDGELOCK_HALT_ACK	EdgeLock halt and clock status 0b - EdgeLock is not fully halted and its clocks must be enabled 1b - EdgeLock is fully halted indicating clocks may be removed

15.4.1.28 ECC memory hardware initialization (ECC_MEM_INIT)

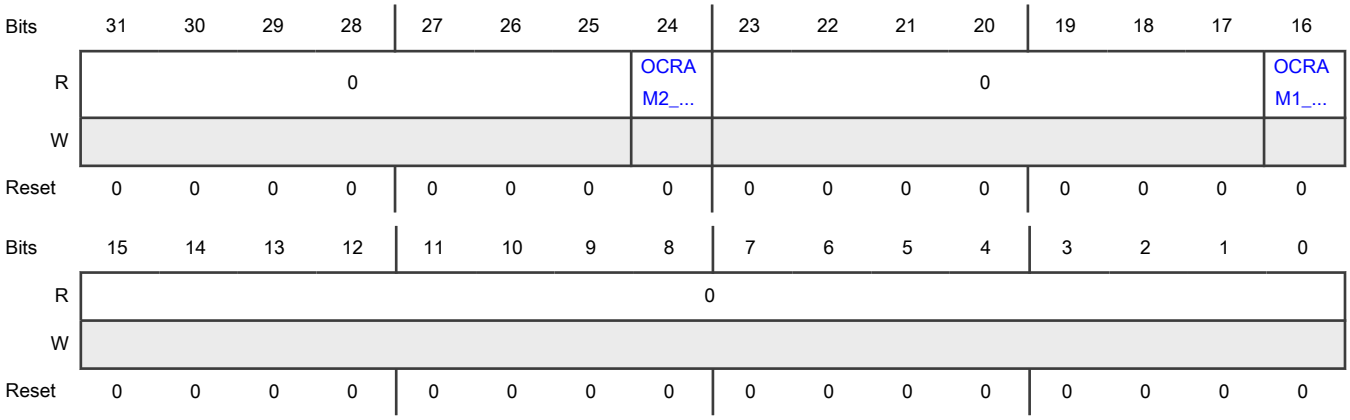
Offset

Register	Offset
ECC_MEM_INIT	120h

Function

ECC memory hardware initialization complete flags. The initialization happens at system reset.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 OCRAM2_INIT_DONE	OCRAM2 initialization status 0b - OCRAM2 memory is under initialization 1b - OCRAM2 memory initialization is complete
23-17 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 OCRAM1_INIT_DONE	OCRAM1 initialization status 0b - OCRAM1 memory is under initialization 1b - OCRAM1 memory initialization is complete
15-0 —	Reserved

15.4.1.29 IOMUXC domain configure (IOMUXC_DOMAIN_CFG)

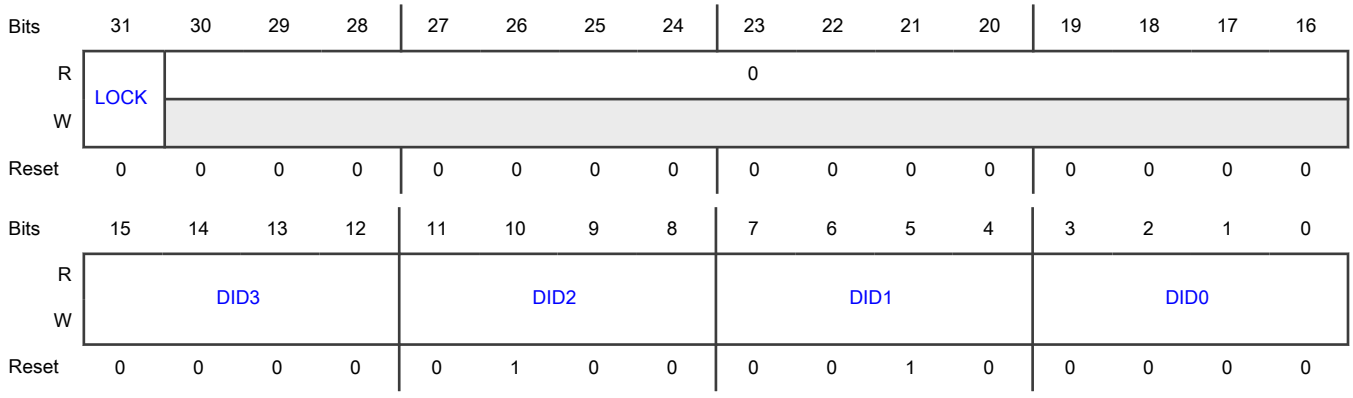
Offset

Register	Offset
IOMUXC_DOMAIN_CFG	148h

Function

This register provides the domain ID allowed to write IOMUXC registers. Only the write access from processor/bus master whose domain ID matches any of the ID in this register will be passed to IOMUXC, otherwise the write will be ignored. Read access to IOMUXC is always allowed regardless its domain ID. It provides write protection with smaller granularity in addition to the 64KB AIPS slot based protection of TRDC. Default domain ID of the masters can be found in TRDC chip specific information section.

Diagram



Fields

Field	Function
31 LOCK	Lock bit Once set to 1 this register cannot be changed any more.
30-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-12 DID3	Domain ID 3 Four bit domain ID value that allowed to write IOMUXC. The write from this domain will be checked against the corresponding IOMUXC register's DWP (write permission) bit3. If DWP bit3 is zero, the write will success. If DWP bit3 is set 1, the write will fail.
11-8 DID2	Domain ID 2 Four bit domain ID value that allowed to write IOMUXC. The write from this domain will be checked against the corresponding IOMUXC register's DWP (write permission) bit2. If DWP bit2 is zero, the write will success. If DWP bit2 is set 1, the write will fail.
7-4 DID1	Domain ID 1 Four bit domain ID value that allowed to write IOMUXC. The write from this domain will be checked against the corresponding IOMUXC register's DWP (write permission) bit1. If DWP bit1 is zero, the write will success. If DWP bit1 is set 1, the write will fail.
3-0 DID0	Domain ID 0 Four bit domain ID value that allowed to write IOMUXC. The write from this domain will be checked against the corresponding IOMUXC register's DWP (write permission) bit0. If DWP bit0 is zero, the write will success. If DWP bit0 is set 1, the write will fail.

15.4.1.30 IOMUXC_AON domain configure (IOMUXC_AON_DOMAIN_CFG)

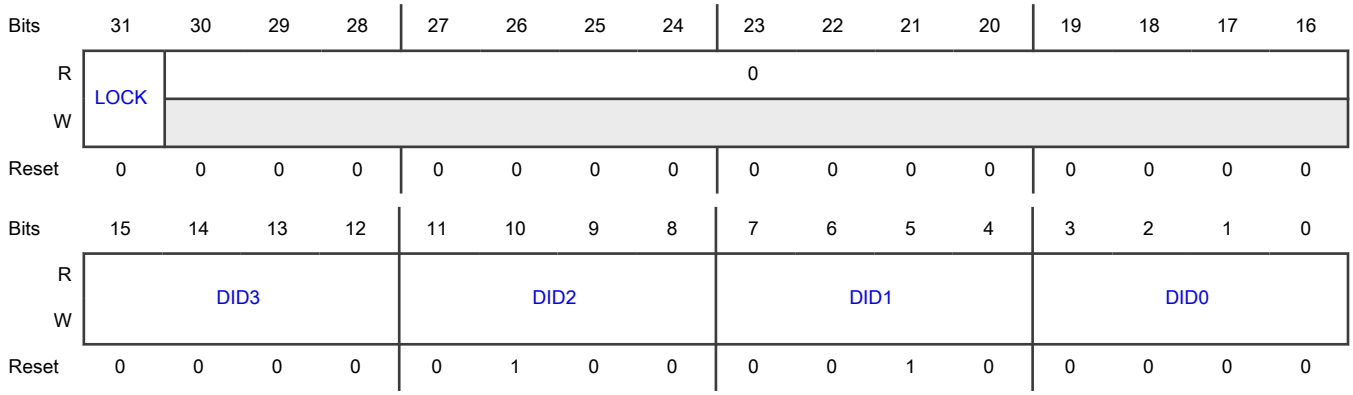
Offset

Register	Offset
IOMUXC_AON_DOMAIN_CFG	14Ch

Function

This register provides the domain ID allowed to write IOMUXC_AON registers. Only the write access from processor/bus master whose domain ID matches any of the ID in this register is passed to IOMUXC_AON, otherwise the write will be ignored. Read access to IOMUXC_AON is always allowed regardless its domain ID. It provides write protection with smaller granularity in addition to the 64KB AIPS slot based protection of TRDC. Default domain ID of the masters can be found in TRDC chip specific information section.

Diagram



Fields

Field	Function
31 LOCK	Lock bit Once set to 1 this register cannot be changed any more.
30-16 —	Reserved
15-12 DID3	Domain ID 3 Four bit domain ID value that allowed to write IOMUXC_AON. The write from this domain will be checked against the corresponding IOMUXC_AON register's DWP (write permission) bit3. If DWP bit3 is zero, the write will success. If DWP bit3 is set 1, the write will fail.
11-8 DID2	Domain ID 2 Four bit domain ID value that allowed to write IOMUXC_AON. The write from this domain will be checked against the corresponding IOMUXC_AON register's DWP (write permission) bit2. If DWP bit2 is zero, the write will success. If DWP bit2 is set 1, the write will fail.
7-4 DID1	Domain ID 1 Four bit domain ID value that allowed to write IOMUXC_AON. The write from this domain will be checked against the corresponding IOMUXC_AON register's DWP (write permission) bit1. If DWP bit1 is zero, the write will success. If DWP bit1 is set 1, the write will fail.
3-0 DID0	Domain ID 0 Four bit domain ID value that allows to write to IOMUXC_AON. The write from this domain is checked against the corresponding IOMUXC_AON register's DWP (write permission) bit 0. If DWP bit 0 is zero, the write will succeed. If DWP bit 0 is set to 1, the write will fail.

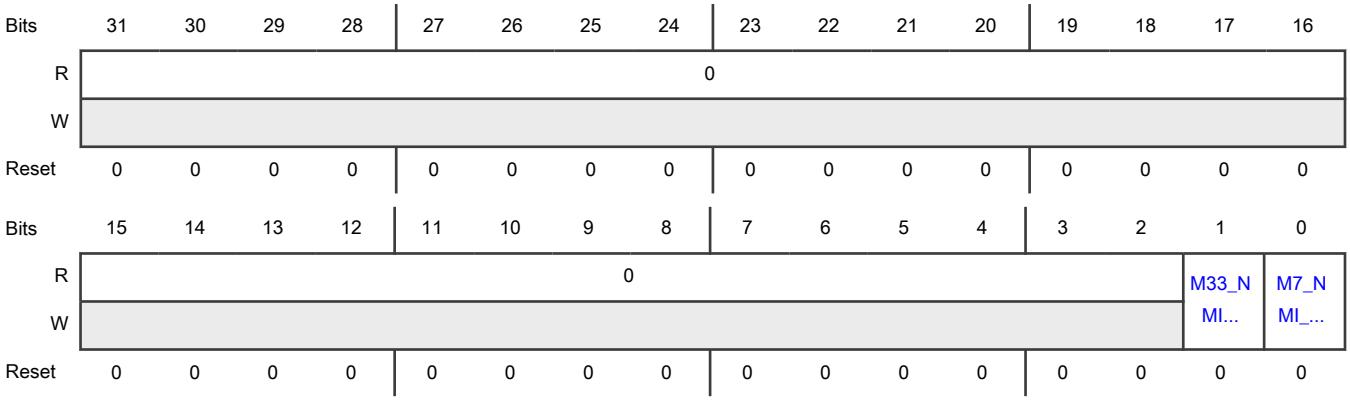
15.4.1.31 NMI control (NMI_CTRL)

Offset

Register	Offset
NMI_CTRL	154h

Function
This register provides control of NMI interrupt from IO.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 M33_NMI_MASK	Mask CM33 NMI pin input 0b - NMI input from IO to CM33 is not blocked 1b - NMI input from IO to CM33 is blocked
0 M7_NMI_MASK	Mask CM7 NMI pin input 0b - NMI input from IO to CM7 is not blocked 1b - NMI input from IO to CM7 is blocked

15.4.1.32 s401_ipi_noclk_ref1 clear control (S401_NOCLK_CLEAR_CTRL)

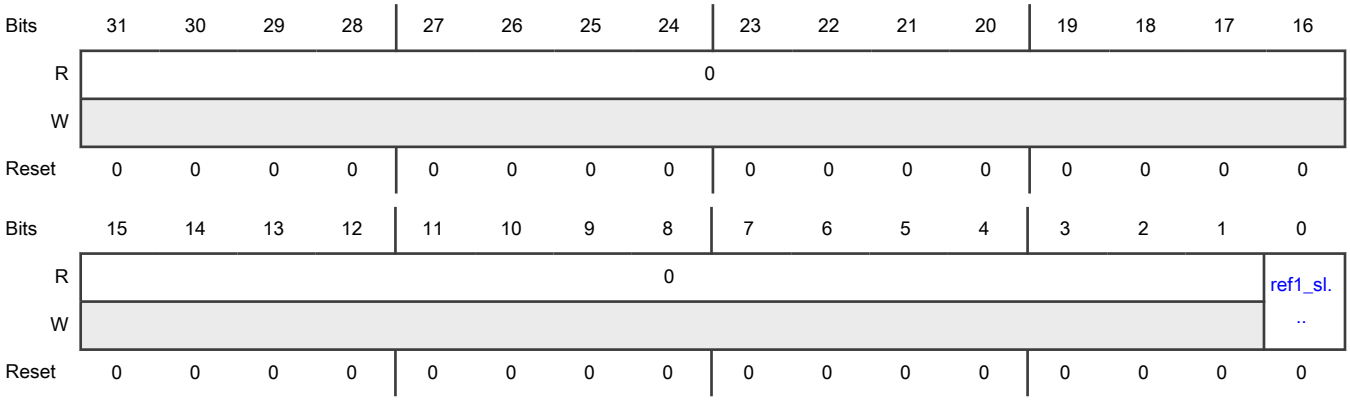
Offset

Register	Offset
S401_NOCLK_CLEAR_CTRL	158h

Function

This register provides clear control of FDET

Diagram



Fields

Field	Function
31-1 —	Reserved
0 ref1_slow_clear	clear the interrupt or reset source 0: no clear 1: clear

Chapter 16
Block Control - Wake-up Domain (BLK_CTRL_WAKEUP)

16.1 Chip-specific BLK_CTRL_WAKEUP Information

Table 120. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

16.2 Overview

The WAKEUP Domain Block Control (BLK_CTRL_WAKEUP) module consists of a group of registers to provide miscellaneous control and status for peripherals within the WAKEUP Domain. These registers provide chip-level control outside the peripherals.

16.2.1 Block diagram

A block diagram of the Wakeup BLK_CTRL implementation is shown below.

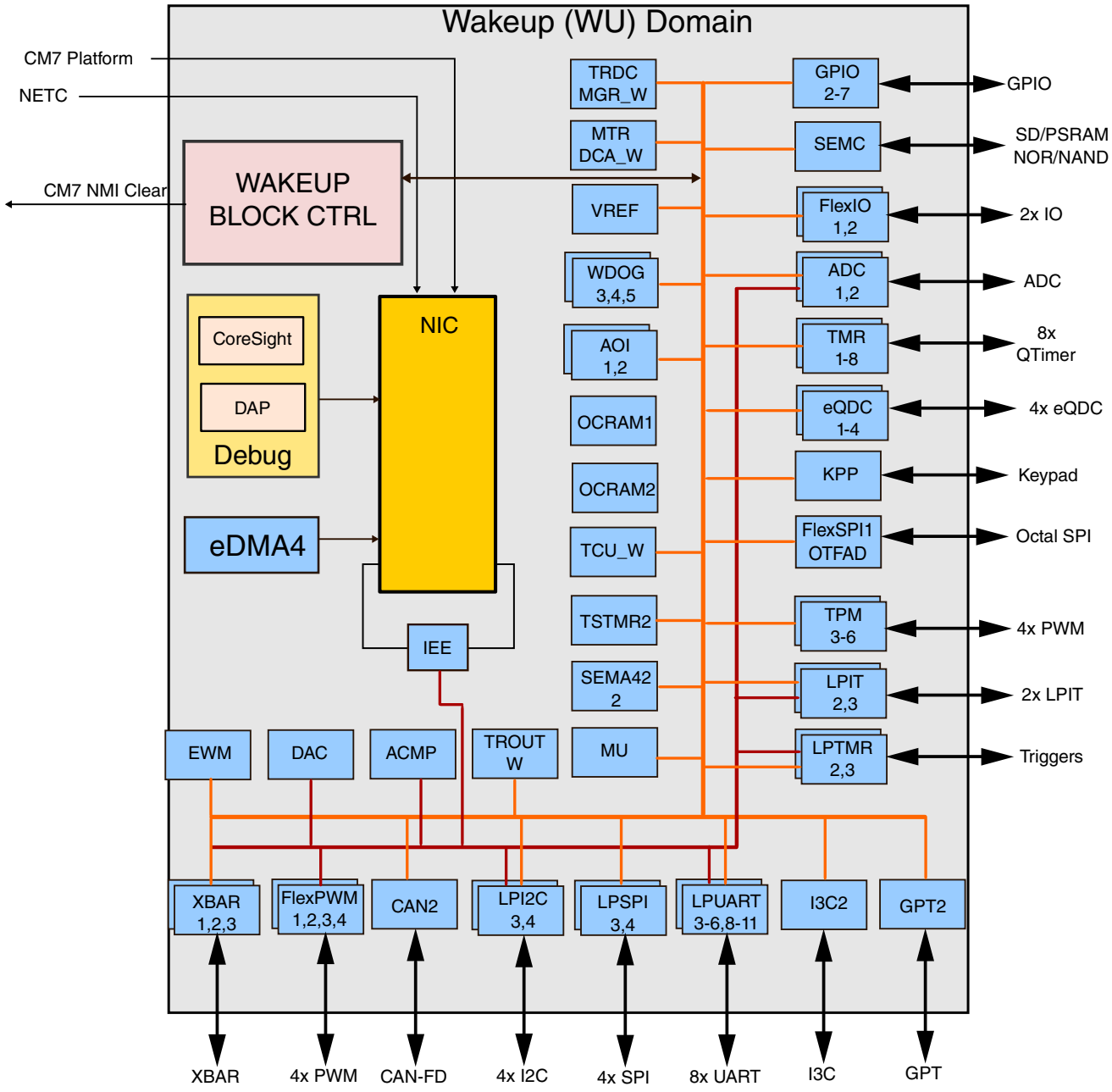


Figure 64. Wakeup BLK_CTRL Block Diagram

16.2.2 Features

The WAKEUP BLK_CTRL has the following features:

- Debug halt mask bits for peripherals in WAKEUP Domain
- WAKEUP SSI master block low power bits
- SAI2/3/4 MCLK direction control and MCLK source selection
- CM7 NMI interrupt clear bit
- USBPHY asynchronous interrupt clear bit

- I3C2 asynchronous interrupt clear bit
- I3C2 IO on-chip strong pull enable bit
- GPIO_AD bank voltage range control
- GPIO compensation cell registers
- Safety clock monitor registers
- XBAR and AOI write protection register
- XBAR pulse configuration registers
- XBAR IO direction control
- CM7 tight coupled SRAM controller registers
- EtherCAT SoC-level configuration register
- NETC SoC-level configuration registers
- USB and uSDHC on-chip AXI bus attribute configuration
- TMR (Quad Timer) and LPIT input trigger selection
- Exclusive access response configuration
- ADC stop mode configuration

16.3 Clocks

The table found here describes the clock sources for Wakeup BLK_CTRL. Please see the Clock Control chapter for clock setting, configuration, and gating information.

Table 121. Clocks

Clock Name	Description
bus_wakeup_clk	Wakeup bus clock for register access.

16.4 Reset

The Wakeup BLK_CTRL resets with the Wakeup Domain.

16.5 Initialization

There are no initialization steps required for Wakeup BLK_CTRL.

16.6 Memory Map and register definition

This section includes the BLK_CTRL_WAKEUP module memory map and detailed descriptions of all registers.

16.6.1 Block Control WAKEUP Domain register descriptions

16.6.1.1 WAKEUP Domain Block Control memory map

BLK_CTRL_WAKEUPMIX base address: 4242_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	IPG DEBUG mask bit (IPG_DEBUG1)	32	RW	0000_0000h
8h	IPG DEBUG mask bit (IPG_DEBUG2)	32	RW	0000_0000h
Ch	IPG DEBUG mask bit (IPG_DEBUG3)	32	RW	0000_0000h
14h	SSI master low power mode control (SSI)	32	RW	0000_0002h
18h	EtherCAT miscellaneous configuration (ECAT_MISC_CFG)	32	RW	0000_0008h
1Ch	DEXSC error response configuration (DEXSC_ERR)	32	RW	0000_0000h
20h	USBPHY miscellaneous control (USBPHY_MISC_CTRL)	32	RW	0000_0101h
24h	NETC Port miscellaneous configuration (NETC_PORT_MISC_CFG)	32	RW	0000_0000h
28h	M7 NMI interrupt clear register (M7_NMI_CLR)	32	RW	0000_0000h
30h	Qtimer miscellaneous control register 1 (QTIMER_CTRL1)	32	RW	0000_0000h
34h	Qtimer miscellaneous control register 2 (QTIMER_CTRL2)	32	RW	0000_0000h
38h	SAI2 MCLK control register (SAI2_MCLK_CTRL)	32	RW	0000_0000h
3Ch	SAI3 MCLK control register (SAI3_MCLK_CTRL)	32	RW	0000_0000h
40h	SAI4 MCLK control register (SAI4_MCLK_CTRL)	32	RW	0000_0000h
44h	XBAR IO direction control register (XBAR_DIR_CTRL1)	32	RW	0000_0000h
48h	XBAR IO direction control register (XBAR_DIR_CTRL2)	32	RW	0000_0000h
4Ch	LPIT trigger input select register (LPIT_TRIG_SEL)	32	RW	0000_0000h
50h	AXI bus attribute configuration register (AXI_ATTR_CFG)	32	RW	0000_0007h
54h	SRAM Control Register 0 (SRAMCR0)	32	RW	0000_0000h
58h	SRAM Control Register 1 (SRAMCR1)	32	RW	0000_0000h
60h	Slave stop mode configure register (SLAVE_STOP_MODE_CFG)	32	RW	0000_0000h
74h	I3C2 async wakeup control register (I3C2_ASYNC_WAKEUP_CTRL)	32	RW	0000_0000h
78h	XBAR and AOI write protect register (XBAR_AOI_WE)	32	RW	0000_0001h
7Ch	XBAR trigger synchronizer control register1 (XBAR_TRIG_SYNC_CTRL1)	32	RW	0000_0000h
80h	XBAR trigger synchronizer control register2 (XBAR_TRIG_SYNC_CTRL2)	32	RW	0000_0000h
100h	NETC link configuration for port0 (NETC_LINK_CFG0)	32	RW	0000_0000h
104h	NETC link configuration for port1 (NETC_LINK_CFG1)	32	RW	0000_0000h
108h	NETC link configuration for port2 (NETC_LINK_CFG2)	32	RW	0000_0000h
10Ch	NETC link configuration for port3 (NETC_LINK_CFG3)	32	RW	0000_0000h
110h	NETC link configuration for port4 (NETC_LINK_CFG4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
114h	NETC RevMII RGMII delay line configuration for port0 (NETC_REVMII_DLL0)	32	RW	0000_000Ch
118h	NETC RevMII RGMII delay line configuration for port1 (NETC_REVMII_DLL1)	32	RW	0000_000Ch
11Ch	NETC RevMII RGMII delay line configuration for port2 (NETC_REVMII_DLL2)	32	RW	0000_000Ch
120h	NETC RevMII RGMII delay line configuration for port3 (NETC_REVMII_DLL3)	32	RW	0000_000Ch
124h	NETC RevMII RGMII delay line configuration for port4 (NETC_REVMII_DLL4)	32	RW	0000_000Ch
130h	Safety clock monitor control and status register (SAFETY_CLK_MON_CS)	32	RW	0000_0000h
134h	Safety clock monitor threshold register (SAFETY_CLK_MON_TH)	32	RW	0000_0000h
140h	GPIO_EMC_B1 bank IO control (EMC_B1_IO_CTRL)	32	RW	0150_4000h
144h	GPIO_EMC_B2 bank IO control (EMC_B2_IO_CTRL)	32	RW	0150_4000h
148h	GPIO_SD_B1 bank IO control (SD_B1_IO_CTRL)	32	RW	0150_4000h
14Ch	GPIO_SD_B2 bank IO control (SD_B2_IO_CTRL)	32	RW	0150_4000h
150h	GPIO_B1 bank IO control (GPIO_B1_IO_CTRL)	32	RW	0150_4000h
154h	GPIO_B2 bank IO control (GPIO_B2_IO_CTRL)	32	RW	0150_4000h
158h	Miscellaneous control register of IO (MISC_IO_CTRL)	32	RW	0000_0000h

16.6.1.2 IPG DEBUG mask bit (IPG_DEBUG1)

Offset

Register	Offset
IPG_DEBUG1	4h

Function

Mask bit for HALTED from CM33/CM7 to peripheral

ipg_debug 0: mask to 0

1: unmask

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M7_WDOG4	M7_WDOG3	M7_TPM6	M7_TPM5	M7_TPM4	M7_TPM3	M7_LPTM...	M7_LPSP...	M7_LPSP...	M7_LPIT2	M7_LPI2...	M7_LPI2...	M7_FLEX...	M7_FLEX...	M7_EDMA4	M7_CAN2
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M33_WDO...	M33_WDO...	M33_TPM6	M33_TPM5	M33_TPM4	M33_TPM3	M33_LPT...	M33_LPS...	M33_LPS...	M33_LPI...	M33_LPI...	M33_LPI...	M33_FLE...	M33_FLE...	M33_EDM...	M33_CAN2
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 M7_WDOG4	WDOG4 debug halted mode with M7
30 M7_WDOG3	WDOG3 debug halted mode with M7
29 M7_TPM6	TPM6 debug halted mode with M7 0b - TPM5 does not enter debug halted mode with CM7 1b - TPM5 enters debug halted mode when CM7 is debug halted
28 M7_TPM5	TPM5 debug halted mode with M7 0b - TPM5 does not enter debug halted mode with CM7 1b - TPM5 enters debug halted mode when CM7 is debug halted
27 M7_TPM4	TPM4 debug halted mode with M7 0b - TPM4 does not enter debug halted mode with CM7 1b - TPM4 enters debug halted mode when CM7 is debug halted
26 M7_TPM3	TPM3 debug halted mode with M7 0b - TPM3 does not enter debug halted mode with CM7 1b - TPM3 enters debug halted mode when CM7 is debug halted
25 M7_LPTMR2	LPTMR2 debug halted mode with M7 0b - LPTMR2 does not enter debug halted mode with CM7 1b - LPTMR2 enters debug halted mode when CM7 is debug halted
24 M7_LPSPI4	LPSPI4 debug halted mode with M7 0b - LPSPI4 does not enter debug halted mode with CM7 1b - LPSPI4 enters debug halted mode when CM7 is debug halted

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 M7_LPSPI3	WDOG3 debug halted mode with M7 0b - WDOD3 does not enter debug halted mode with CM7 1b - WDOG3 enters debug halted mode when CM7 is debug halted
22 M7_LPIT2	LPIT2 debug halted mode with M7 0b - LPIT2 does not enter debug halted mode with CM7 1b - LPIT2 enters debug halted mode when CM7 is debug halted
21 M7_LPI2C4	LPI2C4 debug halted mode with M7 0b - LPI2C4 does not enter debug halted mode with CM7 1b - LPI2C4 enters debug halted mode when CM7 is debug halted
20 M7_LPI2C3	LPI2C3 debug halted mode with M7 0b - LPI2C3 does not enter debug halted mode with CM7 1b - LPI2C3 enters debug halted mode when CM7 is debug halted
19 M7_FLEXIO2	FLEXIO2 debug halted mode with M7 0b - FLEXIO2 does not enter debug halted mode with CM7 1b - FLEXIO2 enters debug halted mode when CM7 is debug halted
18 M7_FLEXIO1	FLEXIO1 debug halted mode with M7 0b - FLEXIO1 does not enter debug halted mode with CM7 1b - FLEXIO1 enters debug halted mode when CM7 is debug halted
17 M7_EDMA4	EDMA4 debug halted mode with M7 0b - EDMA4 does not enter debug halted mode with CM7 1b - EDMA4 enters debug halted mode when CM7 is debug halted
16 M7_CAN2	CAN2 debug halted mode with M7 0b - CAN2 does not enter debug halted mode with CM7 1b - CAN2 enters debug halted mode when CM7 is debug halted
15 M33_WDOG4	WDOG4 debug halted mode with M33 0b - WDOG4 does not enter debug halted mode with CM33 1b - WDOG4 enters debug halted mode when CM33 is debug halted
14 M33_WDOG3	WDOG3 debug halted mode with M33 0b - WDOG3 does not enter debug halted mode with CM33 1b - WDOG3 enters debug halted mode when CM33 is debug halted
13 M33_TPM6	TPM6 debug halted mode with M33

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - TPM6 does not enter debug halted mode with CM33 1b - TPM6 enters debug halted mode when CM33 is debug halted
12 M33_TPM5	TPM5 debug halted mode with M33 0b - TPM5 does not enter debug halted mode with CM33 1b - TPM5 enters debug halted mode when CM33 is debug halted
11 M33_TPM4	TPM3 debug halted mode with M33 0b - does not enter debug halted mode with CM33 1b - enters debug halted mode when CM33 is debug halted
10 M33_TPM3	debug halted mode with M33 0b - TPM3 does not enter debug halted mode with CM33 1b - TPM3 enters debug halted mode when CM33 is debug halted
9 M33_LPTMR2	LPTMR2 debug halted mode with M33 0b - LPTMR2 does not enter debug halted mode with CM33 1b - LPTMR2 enters debug halted mode when CM33 is debug halted
8 M33_LPSPI4	LPSPI4 debug halted mode with M33 0b - LPSPI4 does not enter debug halted mode with CM33 1b - LPSPI4 enters debug halted mode when CM33 is debug halted
7 M33_LPSPI3	LPSPI3 debug halted mode with M33 0b - LPSPI3 does not enter debug halted mode with CM33 1b - LPSPI3 enters debug halted mode when CM33 is debug halted
6 M33_LPIT2	LPIT2 debug halted mode with M33 0b - LPIT2 does not enter debug halted mode with CM33 1b - LPIT2 enters debug halted mode when CM33 is debug halted
5 M33_LPI2C4	LPI2C4 debug halted mode with M33 0b - LPI2C4 does not enter debug halted mode with CM33 1b - LPI2C4 enters debug halted mode when CM33 is debug halted
4 M33_LPI2C3	LPI2C3 debug halted mode with M33 0b - LPI2C3 does not enter debug halted mode with CM33 1b - enters debug halted mode when CM33 is debug halted
3 M33_FLEXIO2	FLEXIO2 debug halted mode with M33 0b - FLEXIO2 does not enter debug halted mode with CM33 1b - FLEXIO2 enters debug halted mode when CM33 is debug halted

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 M33_FLEXIO1	FLEXIO1 debug halted mode with M33 0b - FLEXIO1 does not enter debug halted mode with CM33 1b - FLEXIO1 enters debug halted mode when CM33 is debug halted
1 M33_EDMA4	EDMA4 debug halted mode with M33 0b - EDMA4 does not enter debug halted mode with CM33 1b - EDMA4 enters debug halted mode when CM33 is debug halted
0 M33_CAN2	CAN2 debug halted mode with M7 0b - CAN2 does not enter debug halted mode with CM33 1b - CAN2 enters debug halted mode when CM33 is debug halted

16.6.1.3 IPG DEBUG mask bit (IPG_DEBUG2)

Offset

Register	Offset
IPG_DEBUG2	8h

Function

Mask bit for HALTED from CM33/CM7 to peripheral ipg_debug 0: mask to 0 1: unmask

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	M7_SAI4	M7_SAI3	M7_SAI2	M7_MIC	M7_FLEX...	M7_FLEX...	M7_FLEX...	M7_FLEX...	M7_GPT2	M7_LPI2...	M7_LPI2...	M7_LPIT3	M7_LPSP...	M7_LPSP...	M7_LPTM...	M7_WDOG5
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	M33_SAI4	M33_SAI3	M33_SAI2	M33_MIC	M33_FLE...	M33_FLE...	M33_FLE...	M33_FLE...	M33_GPT2	M33_LPI...	M33_LPI...	M33_LPI...	M33_LPS...	M33_LPS...	M33_LPT...	M33_WDO...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	SAI4 debug halted mode with M7

Table continues on the next page...

Table continued from the previous page...

Field	Function
M7_SAI4	0b - SAI4 does not enter debug halted mode with CM7 1b - SAI4 enters debug halted mode when CM7 is debug halted
30 M7_SAI3	SAI3 debug halted mode with M7 0b - SAI3 does not enter debug halted mode with CM7 1b - SAI3 enters debug halted mode when CM7 is debug halted
29 M7_SAI2	SAI2 debug halted mode with M7 0b - SAI2 does not enter debug halted mode with CM7 1b - SAI2 enters debug halted mode when CM7 is debug halted
28 M7_MIC	MIC debug halted mode with M7 0b - MIC does not enter debug halted mode with CM7 1b - MIC enters debug halted mode when CM7 is debug halted
27 M7_FLEXPWM4	FLEXPWM4 debug halted mode with M7 0b - FLEXPWM4 does not enter debug halted mode with CM7 1b - FLEXPWM4 enters debug halted mode when CM7 is debug halted
26 M7_FLEXPWM3	FLEXPWM3 debug halted mode with M7 0b - FLEXPWM3 does not enter debug halted mode with CM7 1b - FLEXPWM3 enters debug halted mode when CM7 is debug halted
25 M7_FLEXPWM2	FLEXPWM2 debug halted mode with M7 0b - FLEXPWM2 does not enter debug halted mode with CM7 1b - FLEXPWM2 enters debug halted mode when CM7 is debug halted
24 M7_FLEXPWM1	FLEXPWM1 debug halted mode with M7 0b - FLEXPWM1 does not enter debug halted mode with CM7 1b - FLEXPWM1 enters debug halted mode when CM7 is debug halted
23 M7_GPT2	GPT2 debug halted mode with M7 0b - GPT2 does not enter debug halted mode with CM7 1b - GPT2 enters debug halted mode when CM7 is debug halted
22 M7_LPI2C6	LPI2C6" debug halted mode with M7 0b - LPI2C6" does not enter debug halted mode with CM7 1b - LPI2C6" enters debug halted mode when CM7 is debug halted
21 M7_LPI2C5	LPI2C5 debug halted mode with M7 0b - LPI2C5 does not enter debug halted mode with CM7 1b - LPI2C5 enters debug halted mode when CM7 is debug halted

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 M7_LPIT3	LPIT3 debug halted mode with M7 0b - LPIT3 does not enter debug halted mode with CM7 1b - LPIT3 enters debug halted mode when CM7 is debug halted
19 M7_LPSPI6	LPSPI6 debug halted mode with M7 0b - LPSPI6 does not enter debug halted mode with CM7 1b - LPSPI6 enters debug halted mode when CM7 is debug halted
18 M7_LPSPI5	LPTMR3 debug halted mode with M7 0b - LPTMR3 does not enter debug halted mode with CM7 1b - enters debug halted mode when CM7 is debug halted
17 M7_LPTMR3	LPTMR3 debug halted mode with M7 0b - LPTMR3 does not enter debug halted mode with CM7 1b - LPTMR3 enters debug halted mode when CM7 is debug halted
16 M7_WDOG5	WDOG5 debug halted mode with M7 0b - WDOG5 does not enter debug halted mode with CM7 1b - WDOG5 enters debug halted mode when CM7 is debug halted
15 M33_SAI4	SAI4 debug halted mode with M33 0b - SAI4 does not enter debug halted mode with CM33 1b - SAI4 enters debug halted mode when CM33 is debug halted
14 M33_SAI3	SAI3 debug halted mode with M33 0b - SAI3 does not enter debug halted mode with CM33 1b - SAI3 enters debug halted mode when CM33 is debug halted
13 M33_SAI2	SAI2 debug halted mode with M33 0b - SAI2 does not enter debug halted mode with CM33 1b - SAI2 enters debug halted mode when CM33 is debug halted
12 M33_MIC	MIC debug halted mode with M33 0b - MIC does not enter debug halted mode with CM33 1b - MIC enters debug halted mode when CM33 is debug halted
11 M33_FLEXPWM4	FLEXPWM4 debug halted mode with M33 0b - FLEXPWM4 does not enter debug halted mode with CM33 1b - FLEXPWM4 enters debug halted mode when CM33 is debug halted
10	FLEXPWM3 debug halted mode with M33

Table continues on the next page...

Table continued from the previous page...

Field	Function
M33_FLEXPWM3	0b - FLEXPWM3 does not enter debug halted mode with CM33 1b - FLEXPWM3 enters debug halted mode when CM33 is debug halted
9 M33_FLEXPWM2	FLEXPWM2 debug halted mode with M33 0b - FLEXPWM2 does not enter debug halted mode with CM33 1b - FLEXPWM2 enters debug halted mode when CM33 is debug halted
8 M33_FLEXPWM1	FLEXPWM1 debug halted mode with M33 0b - FLEXPWM1 does not enter debug halted mode with CM33 1b - FLEXPWM1 enters debug halted mode when CM33 is debug halted
7 M33_GPT2	GPT2 debug halted mode with M33 0b - GPT2 does not enter debug halted mode with CM33 1b - GPT2 enters debug halted mode when CM33 is debug halted
6 M33_LPI2C6	LPI2C6 debug halted mode with M33 0b - LPI2C6 does not enter debug halted mode with CM33 1b - LPI2C6 enters debug halted mode when CM33 is debug halted
5 M33_LPI2C5	LPI2C5 debug halted mode with M33 0b - LPI2C5 does not enter debug halted mode with CM33 1b - LPI2C5 enters debug halted mode when CM33 is debug halted
4 M33_LPIT3	LPIT3 debug halted mode with M33 0b - LPIT3 does not enter debug halted mode with CM33 1b - LPIT3 enters debug halted mode when CM33 is debug halted
3 M33_LPSPI6	LPSPI6 debug halted mode with M33 0b - LPSPI6 does not enter debug halted mode with CM33 1b - LPSPI6 enters debug halted mode when CM33 is debug halted
2 M33_LPSPI5	LPSPI5 debug halted mode with M33 0b - LPSPI5 does not enter debug halted mode with CM33 1b - LPSPI5 enters debug halted mode when CM33 is debug halted
1 M33_LPTMR3	LPTMR3 debug halted mode with M33 0b - LPTMR3 does not enter debug halted mode with CM33 1b - LPTMR3 enters debug halted mode when CM33 is debug halted
0 M33_WDOG5	WDOG5 debug halted mode with M7 0b - WDOG5 does not enter debug halted mode with CM33 1b - WDOG5 enters debug halted mode when CM33 is debug halted

Table continues on the next page...

Field	Function

16.6.1.4 IPG DEBUG mask bit (IPG_DEBUG3)

Offset

Register	Offset
IPG_DEBUG3	Ch

Function

Mask bit for HALTED from CM33/CM7 to peripheral ipg_debug 0: mask to 0 1: unmask

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				M7_I3C2	M7_QTIM...	M7_QTIM...	M7_QTIM...	M7_QTIM...	M7_QTIM...	M7_QTIM...	M7_QTIM...	M7_QTIM...	M7_SINC3	M7_SINC2	M7_SINC1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				M33_I3C2	M33_QTI...	M33_QTI...	M33_QTI...	M33_QTI...	M33_QTI...	M33_QTI...	M33_QTI...	M33_QTI...	M33_SIN...	M33_SIN...	M33_SIN...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 M7_I3C2	debug halted mode with M7 0b - I3C2 does not enter debug halted mode with CM7 1b - I3C2 enters debug halted mode when CM7 is debug halted
26 M7_QTIMER8	QTIMER8 debug halted mode with M7 0b - QTIMER8 does not enter debug halted mode with CM7 1b - QTIMER8 enters debug halted mode when CM7 is debug halted
25 M7_QTIMER7	QTIMER7 debug halted mode with M7 0b - QTIMER7 does not enter debug halted mode with CM7 1b - QTIMER7 enters debug halted mode when CM7 is debug halted
24	debug halted mode with M7

Table continues on the next page...

Table continued from the previous page...

Field	Function
M7_QTIMER6	0b - does not enter debug halted mode with CM7 1b - enters debug halted mode when CM7 is debug halted
23 M7_QTIMER5	QTIMER5 debug halted mode with M7 0b - QTIMER5 does not enter debug halted mode with CM7 1b - QTIMER5 enters debug halted mode when CM7 is debug halted
22 M7_QTIMER4	QTIMER4 debug halted mode with M7 0b - QTIMER4 does not enter debug halted mode with CM7 1b - QTIMER4 enters debug halted mode when CM7 is debug halted
21 M7_QTIMER3	QTIMER3 debug halted mode with M7 0b - QTIMER3 does not enter debug halted mode with CM7 1b - QTIMER3 enters debug halted mode when CM7 is debug halted
20 M7_QTIMER2	QTIMER2 debug halted mode with M7 0b - QTIMER2 does not enter debug halted mode with CM7 1b - QTIMER2 enters debug halted mode when CM7 is debug halted
19 M7_QTIMER1	QTIMER1 debug halted mode with M7 0b - QTIMER1 does not enter debug halted mode with CM7 1b - QTIMER1 enters debug halted mode when CM7 is debug halted
18 M7_SINC3	SINC3 debug halted mode with M7 0b - SINC3 does not enter debug halted mode with CM7 1b - SINC3 enters debug halted mode when CM7 is debug halted
17 M7_SINC2	SINC2 debug halted mode with M7 0b - SINC2 does not enter debug halted mode with CM7 1b - SINC2 enters debug halted mode when CM7 is debug halted
16 M7_SINC1	SINC1 debug halted mode with M7 0b - SINC1 does not enter debug halted mode with CM7 1b - SINC1 enters debug halted mode when CM7 is debug halted
15-12 —	Reserved
11 M33_I3C2	I3C2 debug halted mode with M33 0b - I3C2 does not enter debug halted mode with CM33 1b - I3C2 enters debug halted mode when CM33 is debug halted

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 M33_QTIMER8	QTIMER8 debug halted mode with M33 0b - QTIMER8 does not enter debug halted mode with CM33 1b - QTIMER8 enters debug halted mode when CM33 is debug halted
9 M33_QTIMER7	QTIMER7 debug halted mode with M33 0b - QTIMER7 does not enter debug halted mode with CM33 1b - QTIMER7 enters debug halted mode when CM33 is debug halted
8 M33_QTIMER6	QTIMER6 debug halted mode with M33 0b - QTIMER6 does not enter debug halted mode with CM33 1b - QTIMER6 enters debug halted mode when CM33 is debug halted
7 M33_QTIMER5	QTIMER5 debug halted mode with M33 0b - QTIMER5 does not enter debug halted mode with CM33 1b - QTIMER5 enters debug halted mode when CM33 is debug halted
6 M33_QTIMER4	QTIMER4 debug halted mode with M33 0b - QTIMER4 does not enter debug halted mode with CM33 1b - QTIMER4 enters debug halted mode when CM33 is debug halted
5 M33_QTIMER3	QTIMER3 debug halted mode with M33 0b - QTIMER3 does not enter debug halted mode with CM33 1b - QTIMER3 enters debug halted mode when CM33 is debug halted
4 M33_QTIMER2	QTIMER2 debug halted mode with M33 0b - QTIMER2 does not enter debug halted mode with CM33 1b - QTIMER2 enters debug halted mode when CM33 is debug halted
3 M33_QTIMER1	QTIMER1 debug halted mode with M33 0b - QTIMER1 does not enter debug halted mode with CM33 1b - QTIMER1 enters debug halted mode when CM33 is debug halted
2 M33_SINC3	SINC3 debug halted mode with M33 0b - SINC3 does not enter debug halted mode with CM33 1b - SINC3 enters debug halted mode when CM33 is debug halted
1 M33_SINC2	SINC2 debug halted mode with M33 0b - SINC2 does not enter debug halted mode with CM33 1b - SINC2 enters debug halted mode when CM33 is debug halted
0	I3C2 debug halted mode with M33

Table continues on the next page...

Table continued from the previous page...

Field	Function
M33_SINC1	0b - I3C2 does not enter debug halted mode with CM33 1b - I3C2 enters debug halted mode when CM33 is debug halted

16.6.1.5 SSI master low power mode control (SSI)

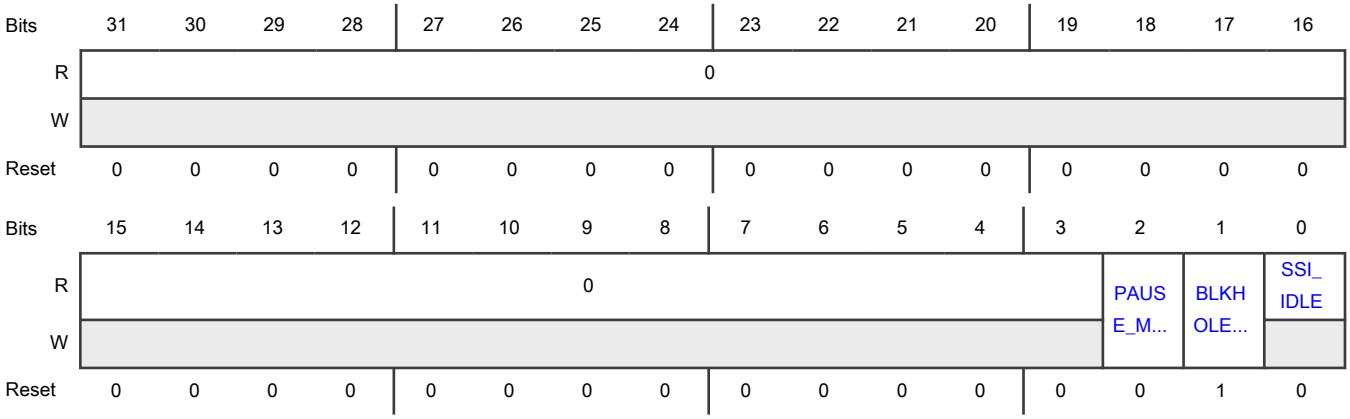
Offset

Register	Offset
SSI	14h

Function

This register enables WAKEUP Domain to M7 SSI Master to enter and exit pause mode.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 PAUSE_MODE	WAKEUP Domain to M7 SSI master pause mode When entering pause mode, new transactions are stopped by de-asserting ready. Any previously started transactions proceed as normal. 0b - WAKEUP Domain to M7 SSI master will enter pause mode 1b - WAKEUP Domain to M7 SSI master will exit pause mode
1	WAKEUP Domain to M7 SSI master blackhole mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLKHOLE_MODE_B	In blackhole mode any new transaction will be accepted and will generate slave errors. It is a requirement that the SSI first be moved to Pause mode to let the system drain and after SSI_IDLE is asserted then move to black hole mode. 0b - WAKEUP Domain to M7 SSI master will exit blackhole mode 1b - WAKEUP Domain to M7 SSI master will enter blackhole mode
0 SSI_IDLE	WAKEUP Domain to M7 SSI master idle 0b - WAKEUP Domain to M7 SSI master is not idle 1b - WAKEUP Domain to M7 SSI master is idle

16.6.1.6 EtherCAT miscellaneous configuration (ECAT_MISC_CFG)

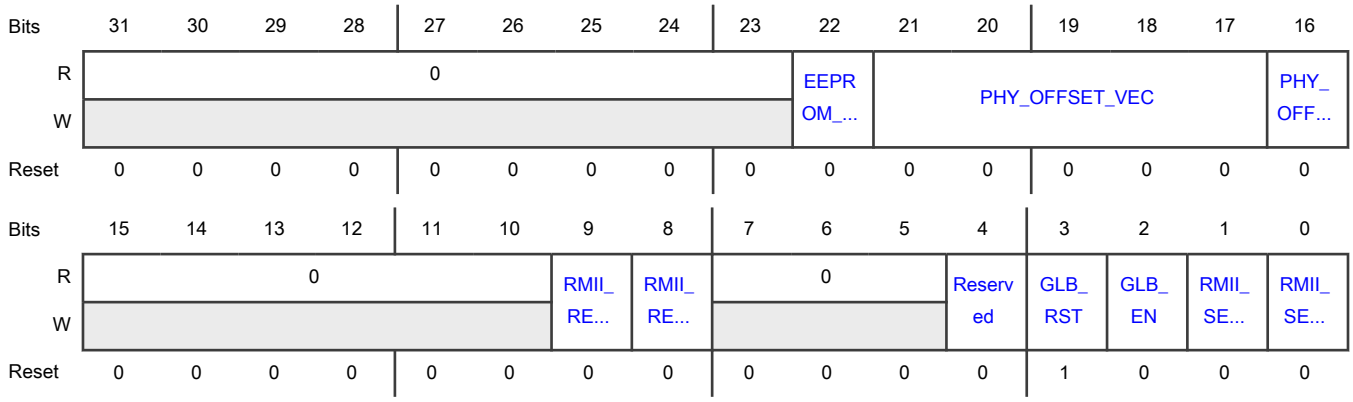
Offset

Register	Offset
ECAT_MISC_CFG	18h

Function

This register provides miscellaneous configuration of EtherCAT.

Diagram



Fields

Field	Function
31-23 —	Reserved
22	EtherCAT EEPROM SIZE OPTION

Table continues on the next page...

Table continued from the previous page...

Field	Function
EEPROM_SIZE_OPTION	This field configures EtherCAT EPROM size, 0: little EPROM Size, 8 bit I2C memory address range 1: larger EEPROM size 16 bit I2C memory address range.
21-17 PHY_OFFSET_VEC	EtherCAT PHY_OFFSET_VEC This field configures EtherCAT PHY_OFFSET_VEC[4:0] input value.
16 PHY_OFFSET	EtherCAT PHY_OFFSET This bit configures EtherCAT PHY_OFFSET input value.
15-10 —	Reserved
9 RMII_REF_CLK_DIR1	RMII Port1 REF_CLK direction control This bitfield controls the direction of RMII REF_CLK on IO for EtherCAT port 1. RMII REF_CLK is the 50MHz RMII clock. It can be input from PT1_TX_CLK IO when this bit is 0. It can be driven by ECAT_CLK_ROOT/2 when this bit is 1. This bit should be set together with the corresponding IOMUXC SW_MUX_CTL_PAD_xx SION bit so that ECAT_CLK_ROOT can provide 50MHz RMII clock to both EtherCAT and board. 0b - RMII REF_CLK is input 1b - RMII REF_CLK is output driven by ECAT_CLK_ROOT/2
8 RMII_REF_CLK_DIR0	RMII Port0 REF_CLK direction control This bitfield controls the direction of RMII REF_CLK on IO for EtherCAT port 0. RMII REF_CLK is the 50MHz RMII clock. It can be input from PT0_TX_CLK IO when this bit is 0. It can be driven by ECAT_CLK_ROOT/2 when this bit is 1. This bit should be set together with the corresponding IOMUXC SW_MUX_CTL_PAD_xx SION bit so that ECAT_CLK_ROOT can provide 50MHz RMII clock to both EtherCAT and board. 0b - RMII REF_CLK is input 1b - RMII REF_CLK is output driven by ECAT_CLK_ROOT/2
7-5 —	Reserved
4 —	Reserved
3 GLB_RST	Global reset of EtherCAT Clearing this bit gets EtherCAT out of reset. 0b - EtherCAT is out of reset 1b - EtherCAT is held in reset
2	Global enable of EtherCAT

Table continues on the next page...

Table continued from the previous page...

Field	Function
GLB_EN	Set this bit to enable EtherCAT. 0b - EtherCAT is off 1b - EtherCAT is on
1 RMII_SEL1	RMII mode selection for EtherCAT port 1 0b - EtherCAT port1 is in MII mode 1b - EtherCAT port1 is in RMII mode
0 RMII_SEL0	RMII mode selection for EtherCAT port 0 0b - EtherCAT port0 is in MII mode 1b - EtherCAT port0 is in RMII mode

16.6.1.7 DEXSC error response configuration (DEXSC_ERR)

Offset

Register	Offset
DEXSC_ERR	1Ch

Function

This register contains exclusive error response configuration bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														LOCK_EX...	EXC_ERR...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 LOCK_EXC_ERR_RESP_EN	Lock bit of EXC_ERR_RESP_EN Once is it set to 1, EXC_ERR_RESP_EN is locked
0 EXC_ERR_RESP_EN	Exclusive error response enable This bit sets the bus response value when CM7 or CM33 exclusive access to CM7 TCM, OCRAM1 or OCRAM2 fails. OKAY response: Normal access okay indicates if a normal access has been successful. Can also indicate an exclusive access failure. SLVError response: Slave error is used when the access has reached the slave successfully, but the slave wishes to return an error condition to the originating master. 0b - OKAY response 1b - SLVError response

16.6.1.8 USBPHY miscellaneous control (USBPHY_MISC_CTRL)

Offset

Register	Offset
USBPHY_MISC_CTRL	20h

Function

This register contains miscellaneous configuration bits for USBPHY1 and USBPHY2.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							USBPHY2...	0							USBPHY1...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							USBPHY2...	0							USBPHY1...
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Fields

Field	Function
31-25 —	Reserved
24 USBPHY2_WAKEUP_IRQ_CLEAR	Clear USBPHY2 wakeup interrupt holding register The wakeup interrupt from USBPHY2 will be held internally until this bit is set to 1. Not that this bit need software to clear. Set it to 1 to clear the interrupt and following with writing it to 0 to complete the operation.
23-17 —	Reserved
16 USBPHY1_WAKEUP_IRQ_CLEAR	Clear USBPHY1 wakeup interrupt holding register The wakeup interrupt from USBPHY1 will be held internally until this bit is set to 1. Not that this bit need software to clear. Set it to 1 to clear the interrupt and following with writing it to 0 to complete the operation.
15-9 —	Reserved
8 USBPHY2_IPG_CLK_ACTIVE	USBPHY2 register access clock enable Clearing this bit to 0 will stop USBPHY2 register access clock to save power.
7-1 —	Reserved
0 USBPHY1_IPG_CLK_ACTIVE	USBPHY1 register access clock enable Clearing this bit to 0 will stop USBPHY1 register access clock to save power.

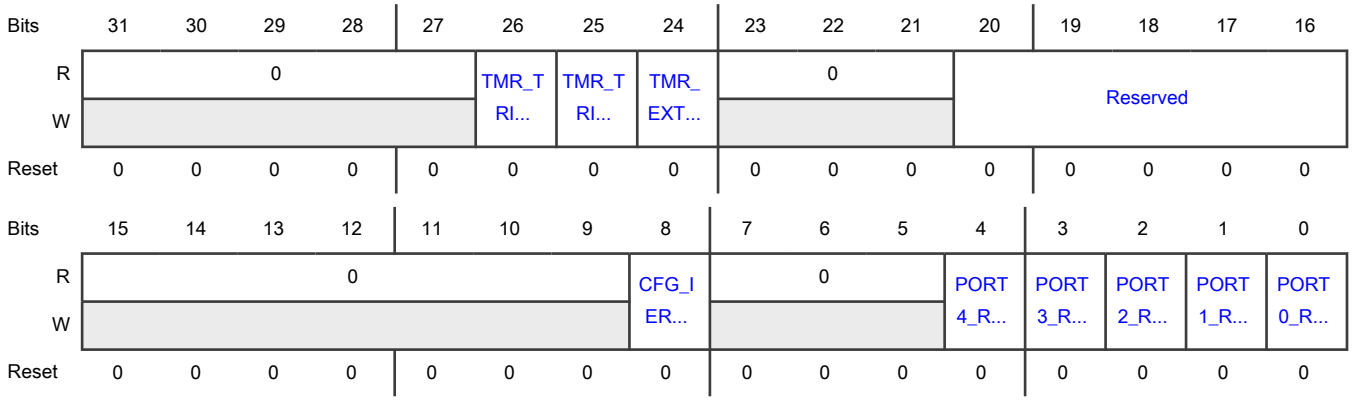
16.6.1.9 NETC Port miscellaneous configuration (NETC_PORT_MISC_CFG)

Offset

Register	Offset
NETC_PORT_MISC_CFG	24h

Function
This register contains miscellaneous configuration bits for NETC Port0-Port4.

Diagram



Fields

Field	Function
31-27 —	Reserved
26 TMR_TRIG2_SEL	1588 timer trigger2 input selection 0b - Input from IOMUX 1b - Input from XBAR
25 TMR_TRIG1_SEL	1588 timer trigger1 input selection 0b - Input from IOMUX 1b - Input from XBAR
24 TMR_EXT_CLK_SEL	1588 timer external clock selection When NETC is configured to select external clock input as 1588 timer reference clock, this bit can further configure the external clock input to be from CCM clock root or from chip pin. IOMUX need select the timer 1588 clock function to use the chip pin. 0b - CCM tmr_1588_clk_root is selected 1b - External pin is selected
23-21 —	Reserved
20-16 —	Reserved
15-9 —	Reserved
8	Default value for IERB NETCRR[LOCK] bit. Determines write accessibility of IERB registers after power-on-reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
CFG_IERB_LOCK	0b - Unlocked after power-on-reset. Normal read/write access to all IERB registers 1b - Locked after power-on-reset. Write access inhibited to all IERB registers, except NETCRR
7-5 —	Reserved
4 PORT4_RMII_REF_CLK_DIR	Port4 RMII Reference clock direction control 0b - Port4 RMII Reference clock is input 1b - Port4 RMII Reference clock is output
3 PORT3_RMII_REF_CLK_DIR	Port3 RMII Reference clock direction control 0b - Port3 RMII Reference clock is input 1b - Port3 RMII Reference clock is output
2 PORT2_RMII_REF_CLK_DIR	Port2 RMII Reference clock direction control 0b - Port2 RMII Reference clock is input 1b - Port2 RMII Reference clock is output
1 PORT1_RMII_REF_CLK_DIR	Port1 RMII Reference clock direction control 0b - Port1 RMII Reference clock is input 1b - Port1 RMII Reference clock is output
0 PORT0_RMII_REF_CLK_DIR	Port0 RMII Reference clock direction control When NETC Port0 is configured to RMII mode, this bitfield controls the direction of the 50MHz RMII clock. It should be set together with the corresponding IOMUXC SW_MUX_CTL_PAD_xx SION bit so that CCM can provide 50MHz RMII clock to both NETC Port0 and board. 0: Port0 RMII Reference clock is input 1: Port0 RMII Reference clock is output

16.6.1.10 M7 NMI interrupt clear register (M7_NMI_CLR)

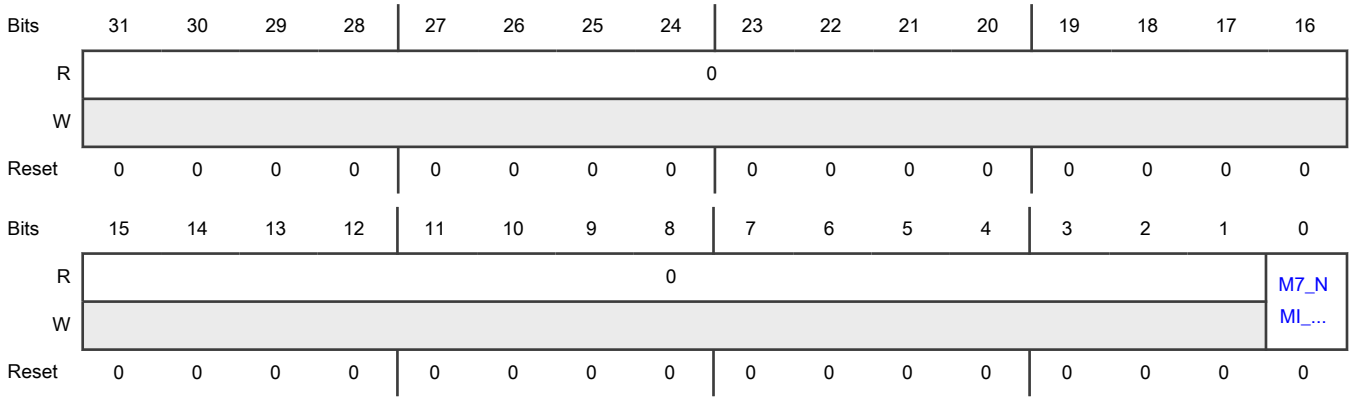
Offset

Register	Offset
M7_NMI_CLR	28h

Function

This register contains the M7_NMI_CLEAR bit used to clear the interrupt from the NMI input.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 M7_NMI_CLEAR	Clear CM7 NMI holding register The NMI input from IO will be held internally until this bit is set to 1. **NOTE** This bit need software to clear. Set it to 1 to clear the interrupt and following with writing it to 0 to complete the operation.

16.6.1.11 Qtimer miscellaneous control register 1 (QTIMER_CTRL1)

Offset

Register	Offset
QTIMER_CTRL1	30h

Function

This register provides input trigger select of QTIMER1-4

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			QTIMER4...	QTIMER4...	QTIMER4...	QTIMER4...	QTIMER4...	0			QTIMER3...	QTIMER3...	QTIMER3...	QTIMER3...	QTIMER3...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			QTIMER2...	QTIMER2...	QTIMER2...	QTIMER2...	QTIMER2...	0			QTIMER1...	QTIMER1...	QTIMER1...	QTIMER1...	QTIMER1...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 QTIMER4_TMR3_INPUT_SEL	QTIMER4 TMR3 input select 0b - Input from IOMUX 1b - Input from XBAR
27 QTIMER4_TMR2_INPUT_SEL	QTIMER4 TMR2 input select 0b - Input from IOMUX 1b - Input from XBAR
26 QTIMER4_TMR1_INPUT_SEL	QTIMER4 TMR1 input select 0b - Input from IOMUX 1b - Input from XBAR
25 QTIMER4_TMR0_INPUT_SEL	QTIMER4 TMR0 input select 0b - Input from IOMUX 1b - Input from XBAR
24 QTIMER4_TMR_CNTS_FREEZE	QTIMER4 timer counter freeze Setting this bit resets counters and output pins of QTIMER1. It is not self-clearing. 0b - Timer counter works normally 1b - Reset counter and output flags
23-21 —	Reserved
20	QTIMER3 TMR3 input select

Table continues on the next page...

Table continued from the previous page...

Field	Function
QTIMER3_TMR3_INPUT_SEL	0b - Input from IOMUX 1b - Input from XBAR
19 QTIMER3_TMR2_INPUT_SEL	QTIMER3 TMR2 input select 0b - Input from IOMUX 1b - Input from XBAR
18 QTIMER3_TMR1_INPUT_SEL	QTIMER3 TMR1 input select 0b - Input from IOMUX 1b - Input from XBAR
17 QTIMER3_TMR0_INPUT_SEL	QTIMER3 TMR0 input select 0b - Input from IOMUX 1b - Input from XBAR
16 QTIMER3_TMR_CNTS_FREEZE	QTIMER3 timer counter freeze Setting this bit resets counters and output pins of QTIMER3. It is not self-clearing. 0b - Timer counter works normally 1b - Reset counter and ouput flags
15-13 —	Reserved
12 QTIMER2_TMR3_INPUT_SEL	QTIMER2 TMR3 input select 0b - Input from IOMUX 1b - Input from XBAR
11 QTIMER2_TMR2_INPUT_SEL	QTIMER2 TMR2 input select 0b - Input from IOMUX 1b - Input from XBAR
10 QTIMER2_TMR1_INPUT_SEL	QTIMER2 TMR1 input select 0b - Input from IOMUX 1b - Input from XBAR
9 QTIMER2_TMR0_INPUT_SEL	QTIMER2 TMR0 input select 0b - Input from IOMUX 1b - Input from XBAR
8	QTIMER2 timer counter freeze Setting this bit resets counters and output pins of QTIMER2. It is not self-clearing.

Table continues on the next page...

Table continued from the previous page...

Field	Function
QTIMER2_TMR_CNTS_FREEZE	0b - Timer counter works normally 1b - Reset counter and output flags
7-5 —	Reserved
4 QTIMER1_TMR3_INPUT_SEL	QTIMER1 TMR3 input select 0b - Input from IOMUX 1b - Input from XBAR
3 QTIMER1_TMR2_INPUT_SEL	QTIMER1 TMR2 input select 0b - Input from IOMUX 1b - Input from XBAR
2 QTIMER1_TMR1_INPUT_SEL	QTIMER1 TMR1 input select 0b - Input from IOMUX 1b - Input from XBAR
1 QTIMER1_TMR0_INPUT_SEL	QTIMER1 TMR0 input select 0b - Input from IOMUX 1b - Input from XBAR
0 QTIMER1_TMR_CNTS_FREEZE	QTIMER1 timer counter freeze Setting this bit resets counters and output pins of QTIMER1. It is not self-clearing. 0b - Timer counter works normally 1b - Reset counter and output flags

16.6.1.12 Qtimer miscellaneous control register 2 (QTIMER_CTRL2)

Offset

Register	Offset
QTIMER_CTRL2	34h

Function
This register provides input trigger select of QTIMER5-8

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			QTIMER8...	QTIMER8...	QTIMER8...	QTIMER8...	QTIMER8...	0			QTIMER7...	QTIMER7...	QTIMER7...	QTIMER7...	QTIMER7...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			QTIMER6...	QTIMER6...	QTIMER6...	QTIMER6...	QTIMER6...	0			QTIMER5...	QTIMER5...	QTIMER5...	QTIMER5...	QTIMER5...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 QTIMER8_TMR3_INPUT_SEL	QTIMER8 TMR3 input select 0b - Input from IOMUX 1b - Input from XBAR
27 QTIMER8_TMR2_INPUT_SEL	QTIMER8 TMR2 input select 0b - Input from IOMUX 1b - Input from XBAR
26 QTIMER8_TMR1_INPUT_SEL	QTIMER8 TMR1 input select 0b - Input from IOMUX 1b - Input from XBAR
25 QTIMER8_TMR0_INPUT_SEL	QTIMER8 TMR0 input select 0b - Input from IOMUX 1b - Input from XBAR
24 QTIMER8_TMR_CNTS_FREEZE	QTIMER8 timer counter freeze Setting this bit resets counters and output pins of QTIMER1. It is not self-clearing. 0b - Timer counter works normally 1b - Reset counter and output flags
23-21 —	Reserved
20	QTIMER7 TMR3 input select

Table continues on the next page...

Table continued from the previous page...

Field	Function
QTIMER7_TMR3_INPUT_SEL	0b - Input from IOMUX 1b - Input from XBAR
19 QTIMER7_TMR2_INPUT_SEL	QTIMER7 TMR2 input select 0b - Input from IOMUX 1b - Input from XBAR
18 QTIMER7_TMR1_INPUT_SEL	QTIMER7 TMR1 input select 0b - Input from IOMUX 1b - Input from XBAR
17 QTIMER7_TMR0_INPUT_SEL	QTIMER7 TMR0 input select 0b - Input from IOMUX 1b - Input from XBAR
16 QTIMER7_TMR_CNTS_FREEZE	QTIMER7 timer counter freeze Setting this bit resets counters and output pins of QTIMER7. It is not self-clearing. 0b - Timer counter works normally 1b - Reset counter and output flags
15-13 —	Reserved
12 QTIMER6_TMR3_INPUT_SEL	QTIMER6 TMR3 input select 0b - Input from IOMUX 1b - Input from XBAR
11 QTIMER6_TMR2_INPUT_SEL	QTIMER6 TMR2 input select 0b - Input from IOMUX 1b - Input from XBAR
10 QTIMER6_TMR1_INPUT_SEL	QTIMER6 TMR1 input select 0b - Input from IOMUX 1b - Input from XBAR
9 QTIMER6_TMR0_INPUT_SEL	QTIMER6 TMR0 input select 0b - Input from IOMUX 1b - Input from XBAR
8	QTIMER6 timer counter freeze Setting this bit resets counters and output pins of QTIMER6. It is not self-clearing.

Table continues on the next page...

Table continued from the previous page...

Field	Function
QTIMER6_TMR_CNTS_FREEZE	0: Timer counter works normally 1 Reset counter and output flags
7-5 —	Reserved
4 QTIMER5_TMR3_INPUT_SEL	QTIMER5 TMR3 input select 0b - Input from IOMUX 1b - Input from XBAR
3 QTIMER5_TMR2_INPUT_SEL	QTIMER5 TMR2 input select 0b - Input from IOMUX 1b - Input from XBAR
2 QTIMER5_TMR1_INPUT_SEL	QTIMER5 TMR1 input select 0b - Input from IOMUX 1b - Input from XBAR
1 QTIMER5_TMR0_INPUT_SEL	QTIMER5 TMR0 input select 0b - Input from IOMUX 1b - Input from XBAR
0 QTIMER5_TMR_CNTS_FREEZE	QTIMER5 timer counter freeze Setting this bit resets counters and output pins of QTIMER5. It is not self-clearing. 0b - Timer counter works normally 1b - Reset counter and output flags

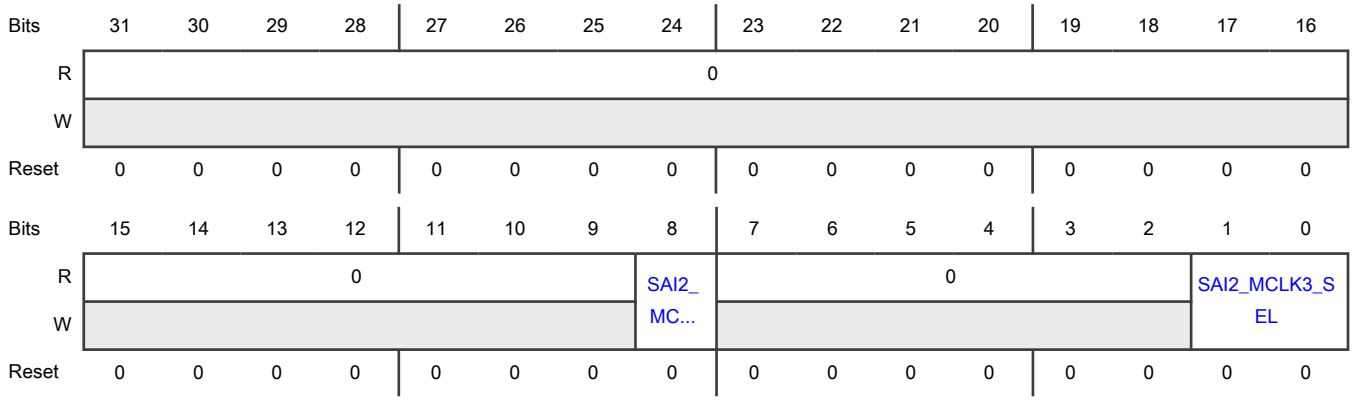
16.6.1.13 SAI2 MCLK control register (SAI2_MCLK_CTRL)

Offset

Register	Offset
SAI2_MCLK_CTRL	38h

Function
This register provides control of SAI2 MCLK direction and selection

Diagram



Fields

Field	Function
31-9 —	Reserved
8 SAI2_MCLK_DIR	SAI2_MCLK IO direction control. IOMUX need select SAI2 MCLK function. 0b - SAI2_MCLK is input signal 1b - SAI2_MCLK is output signal
7-2 —	Reserved
1-0 SAI2_MCLK3_SEL	SAI2 MCLK3 source select See the Audio subsystem clocking diagram in the Audio Overview Chapter for more information. 00b - SPDIF_CLK_ROOT 01b - spdif_tx_clk2 10b - spdif_srclk 11b - spdif_outclock

16.6.1.14 SAI3 MCLK control register (SAI3_MCLK_CTRL)

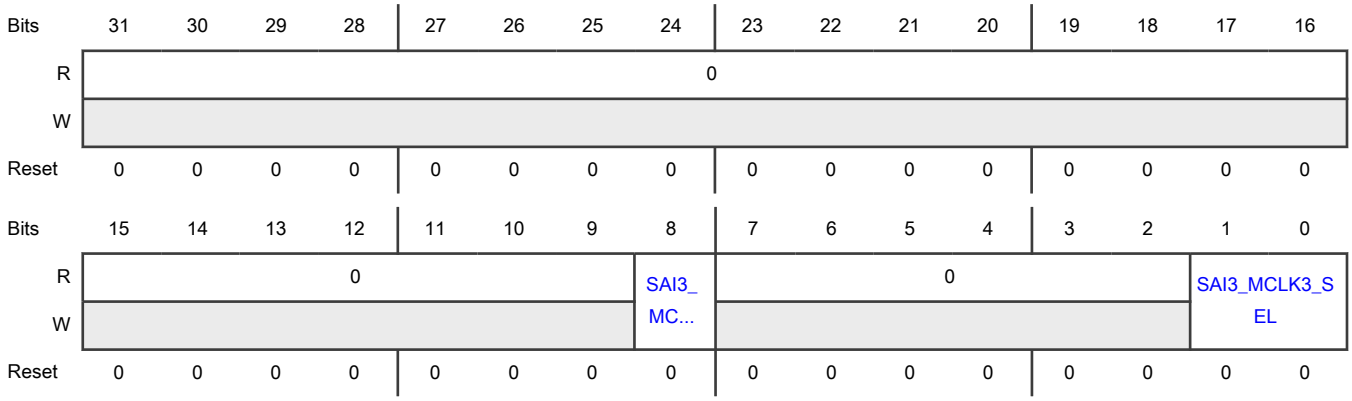
Offset

Register	Offset
SAI3_MCLK_CTRL	3Ch

Function

This register provides control of SAI3 MCLK direction and selection

Diagram



Fields

Field	Function
31-9 —	Reserved
8 SAI3_MCLK_DIR	SAI3_MCLK IO direction control. IOMUX need select SAI3 MCLK function. 0b - SAI3_MCLK is input signal 1b - SAI3_MCLK is output signal
7-2 —	Reserved
1-0 SAI3_MCLK3_SEL	SAI3 MCLK3 source select See the Audio subsystem clocking diagram in the Audio Overview Chapter for more information. 00b - SPDIF_CLK_ROOT 01b - spdif_tx_clk2 10b - spdif_srclk 11b - spdif_outclock

16.6.1.15 SAI4 MCLK control register (SAI4_MCLK_CTRL)

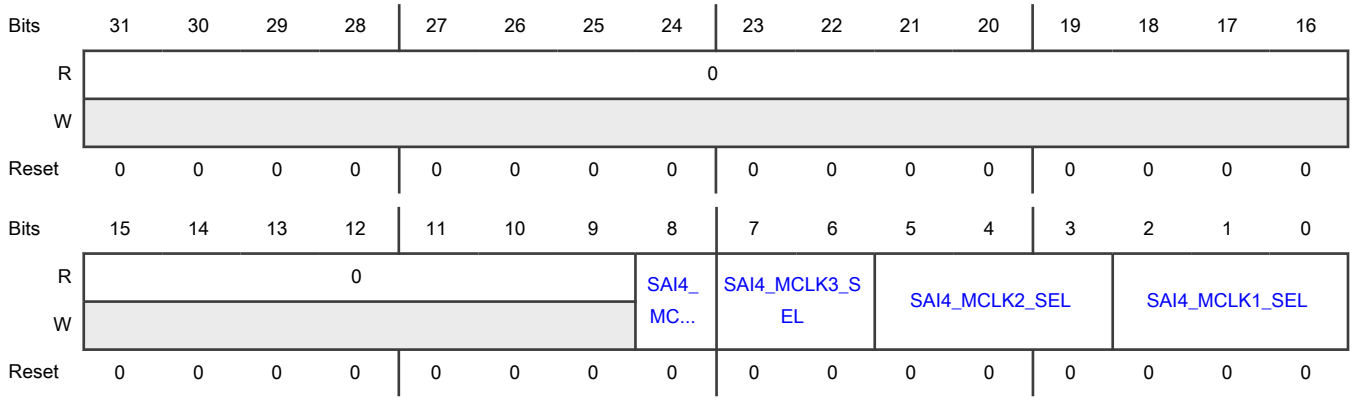
Offset

Register	Offset
SAI4_MCLK_CTRL	40h

Function

This register provides control of SAI4 MCLK direction and selection

Diagram



Fields

Field	Function
31-9 —	Reserved
8 SAI4_MCLK_DIR	SAI4_MCLK IO direction control. IOMUX need select SAI4 MCLK function. 0: SAI4_MCLK is input signal 1: SAI4_MCLK is output signal
7-6 SAI4_MCLK3_SEL	SAI4 MCLK3 source select See the Audio subsystem clocking diagram in the Audio Overview Chapter for more information. 00b - SPDIF_CLK_ROOT 01b - spdif_tx_clk2 10b - spdif_srclk 11b - spdif_outclock
5-3 SAI4_MCLK2_SEL	SAI4 MCLK2 source select See the Audio subsystem clocking diagram in the Audio Overview Chapter for more information. 000b - SAI4_CLK_ROOT 001b - SAI2_CLK_ROOT 011b - SAI3_CLK_ROOT 100b - SAI4 MCLK IO pin 101b - SAI2 MCLK IO pin 110b - SAI3 MCLK IO pin 111b - Reserved
2-0	SAI4 MCLK1 source select

Table continues on the next page...

Table continued from the previous page...

Field	Function
SAI4_MCLK1_SEL	000 SAI4_CLK_ROOT See the Audio subsystem clocking diagram in the Audio Overview Chapter for more information. 000b - SAI4_CLK_ROOT 001b - SAI2_CLK_ROOT 011b - SAI3_CLK_ROOT 100b - SAI4 MCLK IO pin 101b - SAI2 MCLK IO pin 110b - SAI3 MCLK IO pin 111b - Reserved

16.6.1.16 XBAR IO direction control register (XBAR_DIR_CTRL1)

Offset

Register	Offset
XBAR_DIR_CTRL1	44h

Function

This register controls direction of the IOs that select XBAR function

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	IOMUXC_...	0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	IOMUXC XBAR_INOUT31 function direction select 0b - XBAR_INOUT as input

Table continues on the next page...

Table continued from the previous page...

Field	Function
IOMUXC_XBAR_DIR_SEL_31	1b - XBAR_INOUT as output
30 IOMUXC_XBAR_DIR_SEL_30	IOMUXC XBAR_INOUT30 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
29 IOMUXC_XBAR_DIR_SEL_29	IOMUXC XBAR_INOUT29 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
28 IOMUXC_XBAR_DIR_SEL_28	IOMUXC XBAR_INOUT28 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
27 IOMUXC_XBAR_DIR_SEL_27	IOMUXC XBAR_INOUT27 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
26 IOMUXC_XBAR_DIR_SEL_26	IOMUXC XBAR_INOUT26 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
25 IOMUXC_XBAR_DIR_SEL_25	IOMUXC XBAR_INOUT25 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
24 IOMUXC_XBAR_DIR_SEL_24	IOMUXC XBAR_INOUT24 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
23 IOMUXC_XBAR_DIR_SEL_23	IOMUXC XBAR_INOUT23 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
22 IOMUXC_XBAR_DIR_SEL_22	IOMUXC XBAR_INOUT22 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
21 IOMUXC_XBAR_DIR_SEL_21	IOMUXC XBAR_INOUT21 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 IOMUXC_XBAR_DIR_SEL_20	IOMUXC XBAR_INOUT20 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
19 IOMUXC_XBAR_DIR_SEL_19	IOMUXC XBAR_INOUT19 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
18 IOMUXC_XBAR_DIR_SEL_18	IOMUXC XBAR_INOUT18 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
17 IOMUXC_XBAR_DIR_SEL_17	IOMUXC XBAR_INOUT17 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
16 IOMUXC_XBAR_DIR_SEL_16	IOMUXC XBAR_INOUT16 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
15 IOMUXC_XBAR_DIR_SEL_15	IOMUXC XBAR_INOUT15 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
14 IOMUXC_XBAR_DIR_SEL_14	IOMUXC XBAR_INOUT14 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
13 IOMUXC_XBAR_DIR_SEL_13	IOMUXC XBAR_INOUT13 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
12 IOMUXC_XBAR_DIR_SEL_12	IOMUXC XBAR_INOUT12 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
11 IOMUXC_XBAR_DIR_SEL_11	IOMUXC XBAR_INOUT11 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
10	IOMUXC XBAR_INOUT10 function direction select

Table continues on the next page...

Table continued from the previous page...

Field	Function
IOMUXC_XBAR_DIR_SEL_10	0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
9 IOMUXC_XBAR_DIR_SEL_9	IOMUXC XBAR_INOUT9 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
8 IOMUXC_XBAR_DIR_SEL_8	IOMUXC XBAR_INOUT8 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
7 IOMUXC_XBAR_DIR_SEL_7	IOMUXC XBAR_INOUT7 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
6 IOMUXC_XBAR_DIR_SEL_6	IOMUXC XBAR_INOUT6 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
5 IOMUXC_XBAR_DIR_SEL_5	IOMUXC XBAR_INOUT5 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
4 IOMUXC_XBAR_DIR_SEL_4	IOMUXC XBAR_INOUT4 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
3-0 —	Reserved

16.6.1.17 XBAR IO direction control register (XBAR_DIR_CTRL2)

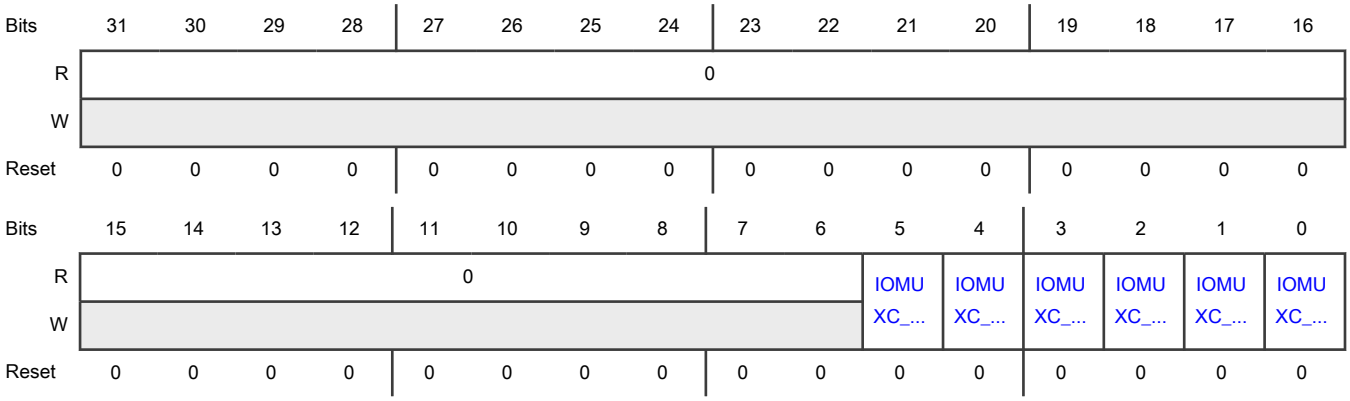
Offset

Register	Offset
XBAR_DIR_CTRL2	48h

Function

This register controls direction of the IOs that select XBAR function

Diagram



Fields

Field	Function
31-6 —	Reserved
5 IOMUXC_XBAR_DIR_SEL_37	IOMUXC XBAR_INOUT37 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
4 IOMUXC_XBAR_DIR_SEL_36	IOMUXC XBAR_INOUT36 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
3 IOMUXC_XBAR_DIR_SEL_35	IOMUXC XBAR_INOUT35 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
2 IOMUXC_XBAR_DIR_SEL_34	IOMUXC XBAR_INOUT34 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
1 IOMUXC_XBAR_DIR_SEL_33	IOMUXC XBAR_INOUT33 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output
0 IOMUXC_XBAR_DIR_SEL_32	IOMUXC XBAR_INOUT32 function direction select 0b - XBAR_INOUT as input 1b - XBAR_INOUT as output

16.6.1.18 LPIT trigger input select register (LPIT_TRIG_SEL)

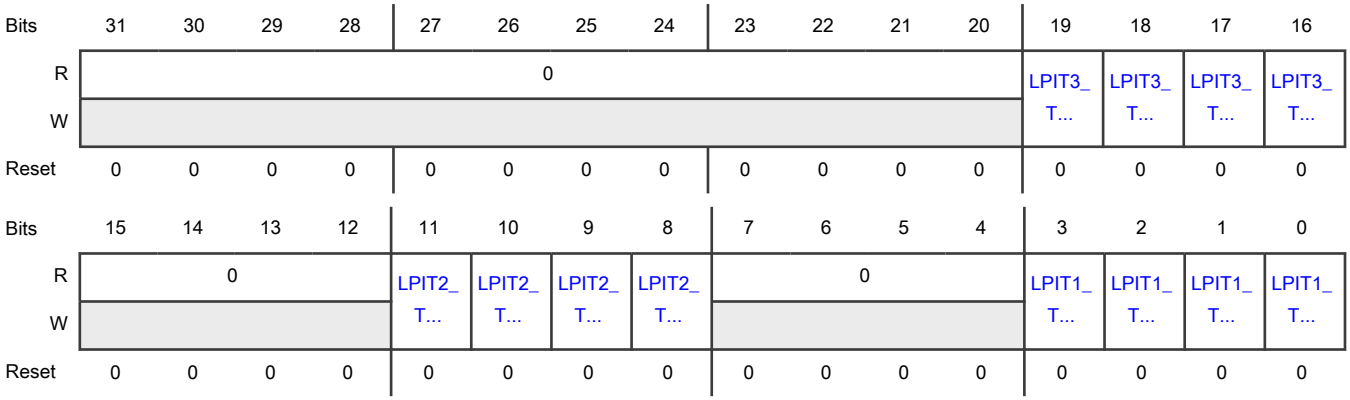
Offset

Register	Offset
LPIT_TRIG_SEL	4Ch

Function

This register provides input trigger select of LPIT1-3

Diagram



Fields

Field	Function
31-20 —	Reserved
19 LPIT3_TRIG3_INPUT_SEL	LPIT3 TRIG3 input select 0b - Input from IOMUX 1b - Input from XBAR
18 LPIT3_TRIG2_INPUT_SEL	LPIT3 TRIG2 input select 0b - Input from IOMUX 1b - Input from XBAR
17 LPIT3_TRIG1_INPUT_SEL	LPIT3 TRIG1 input select 0b - Input from IOMUX 1b - Input from XBAR
16 LPIT3_TRIG0_INPUT_SEL	LPIT3 TRIG0 input select 0b - Input from IOMUX 1b - Input from XBAR

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11 LPIT2_TRIG3_INPUT_SEL	LPIT2 TRIG3 input select 0b - Input from IOMUX 1b - Input from XBAR
10 LPIT2_TRIG2_INPUT_SEL	LPIT2 TRIG2 input select 0b - Input from IOMUX 1b - Input from XBAR
9 LPIT2_TRIG1_INPUT_SEL	LPIT2 TRIG1 input select 0b - Input from IOMUX 1b - Input from XBAR
8 LPIT2_TRIG0_INPUT_SEL	LPIT2 TRIG0 input select 0b - Input from IOMUX 1b - Input from XBAR
7-4 —	Reserved
3 LPIT1_TRIG3_INPUT_SEL	LPIT1 TRIG3 input select 0b - Input from IOMUX 1b - Input from XBAR
2 LPIT1_TRIG2_INPUT_SEL	LPIT1 TRIG2 input select 0b - Input from IOMUX 1b - Input from XBAR
1 LPIT1_TRIG1_INPUT_SEL	LPIT1 TRIG1 input select 0b - Input from IOMUX 1b - Input from XBAR
0 LPIT1_TRIG0_INPUT_SEL	LPIT1 TRIG0 input select 0b - Input from IOMUX 1b - Input from XBAR

16.6.1.19 AXI bus attribute configuration register (AXI_ATTR_CFG)

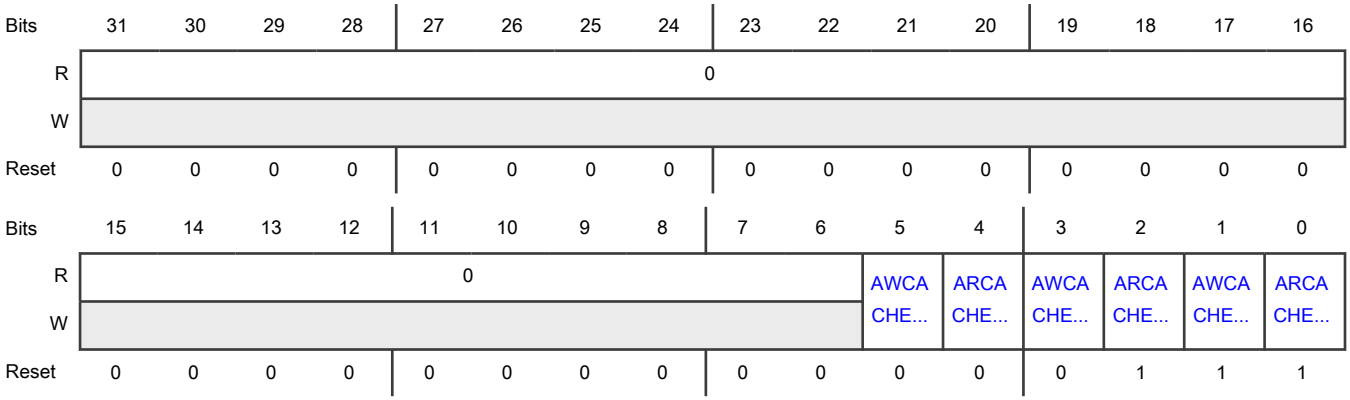
Offset

Register	Offset
AXI_ATTR_CFG	50h

Function

Write this register to configure AXI bus attribute for masters including USB and uSDHC.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 AWCACHE_USB	USB block cacheable attribute value of AXI write transactions If the current transaction is cacheable, the bus infrastructure (NIC) will reorganize the transaction to optimize bus performance. The NIC will combine two 32-bit USB transactions into one 64-bit transaction. 0b - Cacheable attribute is off for write transactions 1b - Cacheable attribute is on for write transactions
4 ARCACHE_USB	USB block cacheable attribute value of AXI read transactions If the current transaction is cacheable, the bus infrastructure (NIC) will reorganize the transaction to optimize bus performance. The NIC will combine two 32-bit USB transactions into one 64-bit transaction. 0b - Cacheable attribute is off for read transactions 1b - Cacheable attribute is on for read transactions
3 AWCACHE_USDHC2	uSDHC2 block cacheable attribute value of AXI write transactions

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If the current transaction is cacheable, the bus infrastructure (NIC) will reorganize the transaction to optimize bus performance. The NIC will combine two 32-bit uSDHC transactions into one 64-bit transaction. 0b - Cacheable attribute is off for write transactions 1b - Cacheable attribute is on for write transactions
2 ARCACHE_USDHC2	uSDHC2 block cacheable attribute value of AXI read transactions If the current transaction is cacheable, the bus infrastructure (NIC) will reorganize the transaction to optimize bus performance. The NIC will combine two 32-bit uSDHC transactions into one 64-bit transaction. 0b - Cacheable attribute is off for read transactions 1b - Cacheable attribute is on for read transactions
1 AWCACHE_USDHC1	uSDHC1 block cacheable attribute value of AXI write transactions If the current transaction is cacheable, the bus infrastructure (NIC) will reorganize the transaction to optimize bus performance. The NIC will combine two 32-bit uSDHC transactions into one 64-bit transaction. 0b - Cacheable attribute is off for write transactions 1b - Cacheable attribute is on for write transactions
0 ARCACHE_USDHC1	uSDHC1 block cacheable attribute value of AXI read transactions If the current transaction is cacheable, the bus infrastructure (NIC) will reorganize the transaction to optimize bus performance. The NIC will combine two 32-bit uSDHC transactions into one 64-bit transaction. 0b - Cacheable attribute is off for read transactions 1b - Cacheable attribute is on for read transactions

16.6.1.20 SRAM Control Register 0 (SRAMCR0)

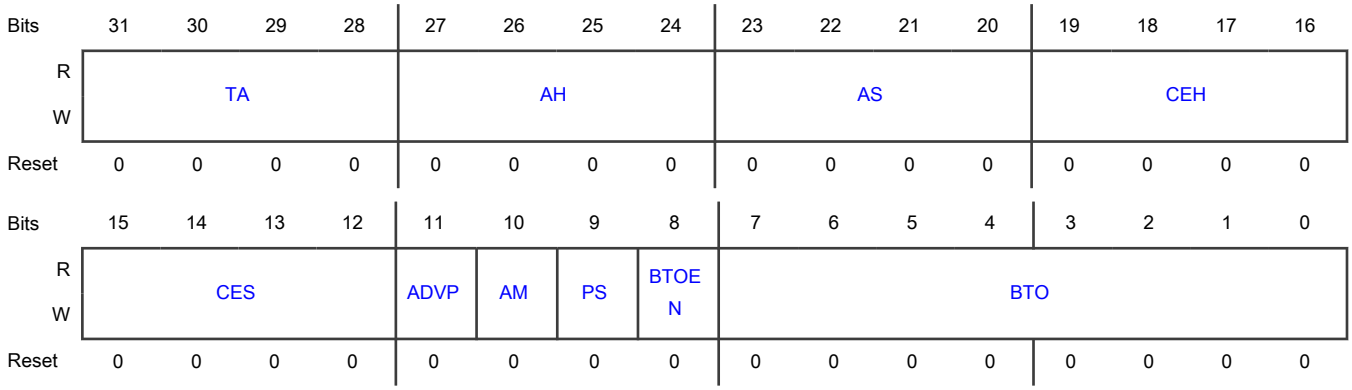
Offset

Register	Offset
SRAMCR0	54h

Function

This register configures SRAM device

Diagram



Fields

Field	Function
31-28 TA	Turnaround time Turnaround time is TA+1 time granularity. The time granularity is configured by SRAMCR1[PRE] field. Both the SRAMC and Device do not drive Data Lines to avoid bus confliction.
27-24 AH	Address hold time Address hold time is AH+1 time granularity. The time granularity is configured by SRAMCR1[PRE] field. The address hold time follows address setup time only.
23-20 AS	Address setup time Address setup time is AS time granularity. The time granularity is configured by SRAMCR1[PRE] field. Address setup time has 1 clock cycle when AS is 0 in ADMUX mode.
19-16 CEH	CE hold time CE Hold time is CEH time granularity. The time granularity is configured by SRAMCR1[PRE] field.
15-12 CES	CE setup time CE setup time is CES time granularity. The time granularity is configured by SRAMCR1[PRE] field.
11 ADVP	ADV# polarity 0b - ADV# is active low. 1b - ADV# is active high.
10 AM	Address Mode It defines the address mode. 0b - Address/Data MUX mode (ADMUX) 1b - Address/Data non-MUX mode (Non-ADMUX)
9 PS	Port Size The port size must be aligned with SRAM data width.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - 8bit 1b - 16bit
8 BTOEN	AHB Bus Timeout Enable AHB bus timeout enable. 0b - AHB bus timeout counter is not enabled. 1b - AHB bus timeout counter is enabled.
7-0 BTO	AHB Bus Timeout Wait Cycle If AHB Command is not granted by arbitrator, it will timeout after BTO * 1024 AHB Clock cycles. When an AHB command grant timeout occurs, there will be an AHB bus error response.

16.6.1.21 SRAM Control Register 1 (SRAMCR1)

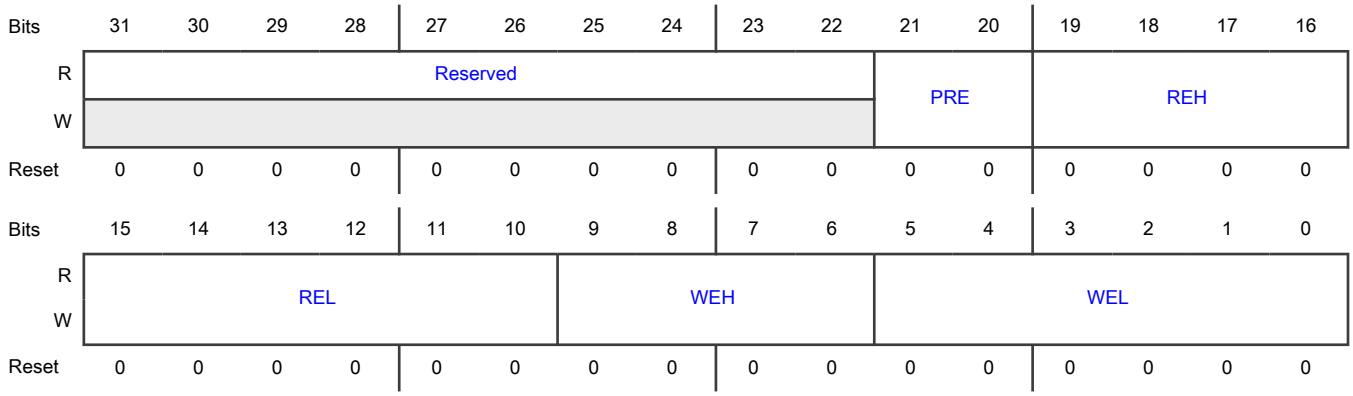
Offset

Register	Offset
SRAMCR1	58h

Function

This register configures SRAM device

Diagram



Fields

Field	Function
31-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-20 PRE	Prescaler timer Prescaler timer controls the time granularity for CES, CEH, AS, AH, TA, WEL, WEH, REL and REH. 00b - Time granularity is 1 clock cycle. 01b - Time granularity is 2 clock cycles. 10b - Time granularity is 3 clock cycles. 11b - Time granularity is 4 clock cycles.
19-16 REH	RE high time RE high time is REH+1 time granularity. The time granularity is configured by SRAMCR1[PRE] field.
15-10 REL	RE low time RE low time is REL+1 time granularity. The time granularity is configured by SRAMCR1[PRE] field.
9-6 WEH	WE high time WE high time is WEH+1 time granularity. The time granularity is configured by SRAMCR1[PRE] field.
5-0 WEL	WE low time WE low time is WEL+1 time granularity. The time granularity is configured by SRAMCR1[PRE] field.

16.6.1.22 Slave stop mode configure register (SLAVE_STOP_MODE_CFG)

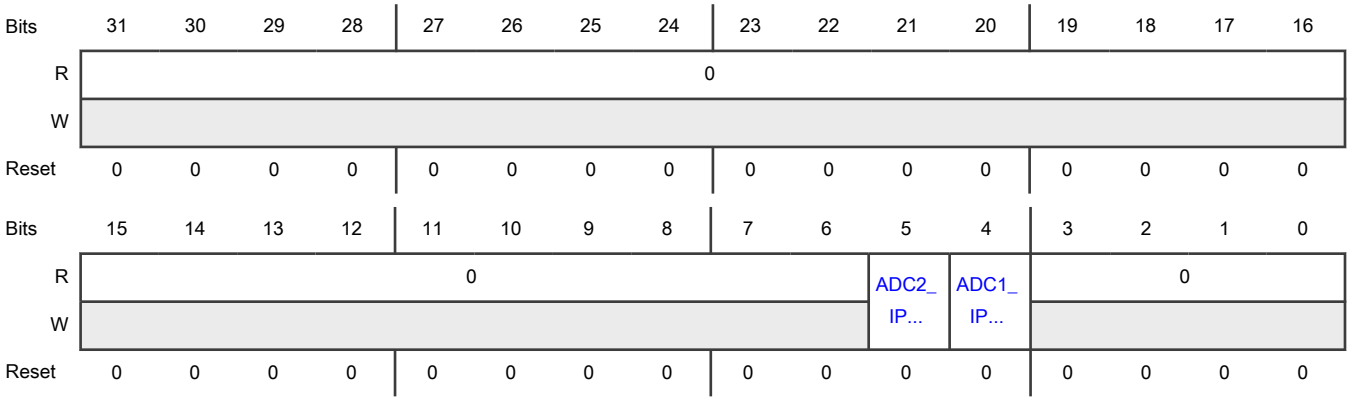
Offset

Register	Offset
SLAVE_STOP_MODE_CFG	60h

Function

Write this register to configure slave behavior under stop mode

Diagram



Fields

Field	Function
31-6 —	Reserved
5 ADC2_IPG_STOP_MODE	ADC2 stop mode selection. 0b - This module is functional in Stop Mode 1b - This module is not functional in Stop Mode
4 ADC1_IPG_STOP_MODE	ADC1 stop mode selection. 0b - This module is functional in Stop Mode 1b - This module is not functional in Stop Mode
3-0 —	Reserved

16.6.1.23 I3C2 async wakeup control register (I3C2_ASYNC_WAKEUP_CTRL)

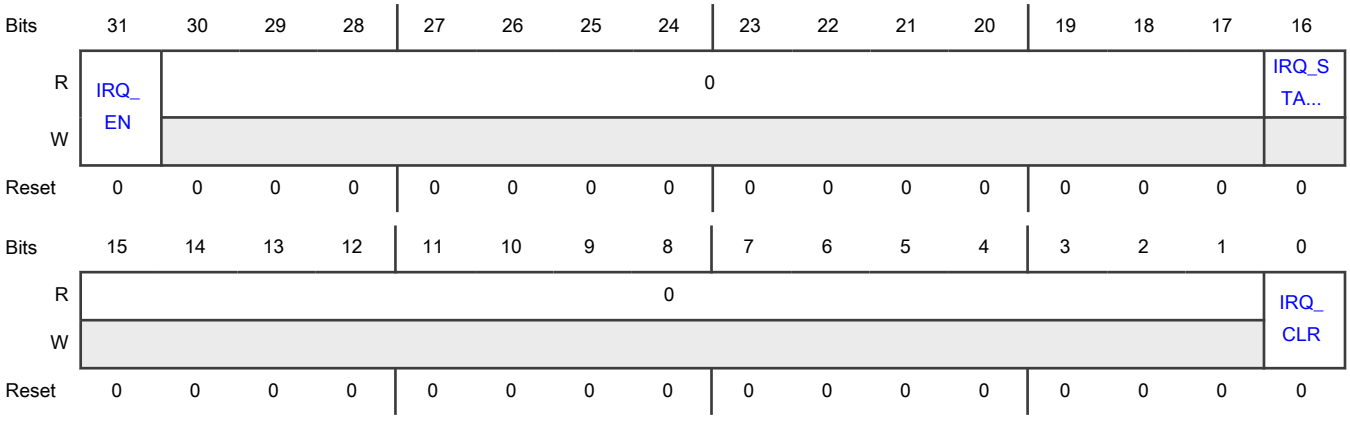
Offset

Register	Offset
I3C2_ASYNC_WAKEUP_CTRL	74h

Function

This register contains status and control for I3C2 async wakeup interrupt

Diagram



Fields

Field	Function
31 IRQ_EN	Master mode async wakeup interrupt enable This bit is to enable I3C2 master mode async wakeup interrupt. If enabled the interrupt happens when I3C2 is in master mode, FCLK is stopped and SCA is driven low by slave. 0b - Interrupt disabled 1b - Interrupt enabled
30-17 —	Reserved
16 IRQ_STATUS	Async wakeup interrupt status This bit indicates the status of I3C2 async wakeup interrupt. For master the interrupt is enabled by SCA_IRQ_EN bit of this register. For slave mode the interrupt is enabled by bit 9 of I3C2 interrupt enable register of the I3C chapter. 0b - Interrupt not asserted 1b - Interrupt asserted
15-1 —	Reserved
0 IRQ_CLR	Async wakeup interrupt clear This bit is to clear I3C2 async wakeup interrupt. The interrupt will be held until this bit is set to 1. Note that this bit need software to clear. Set it to 1 to clear the interrupt and following with writing it to 0 to complete the operation.

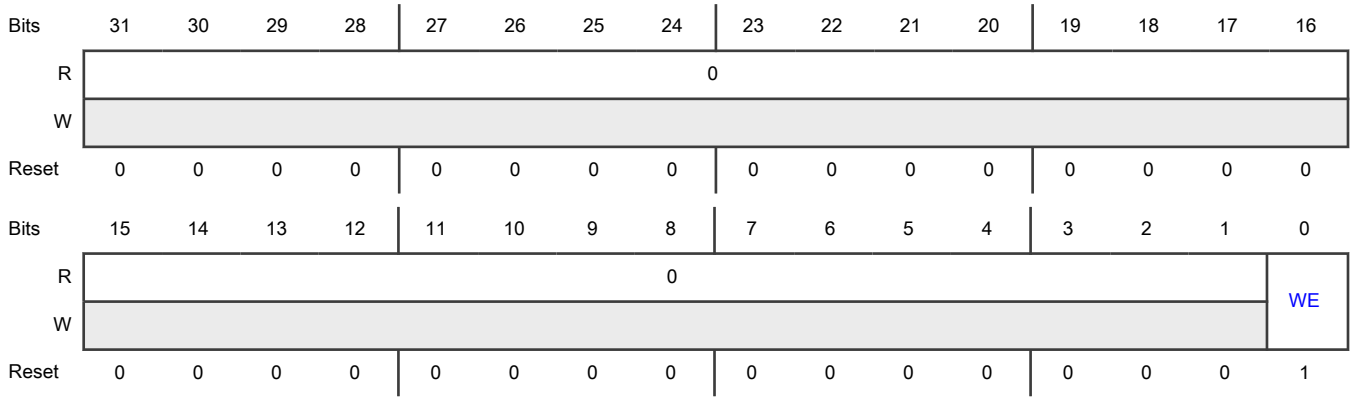
16.6.1.24 XBAR and AOI write protect register (XBAR_AOI_WE)

Offset

Register	Offset
XBAR_AOI_WE	78h

Function
Register write enable bit of XBAR and AOI

Diagram



Fields

Field	Function
31-1 —	Reserved
0 WE	Write Enable to XBAR and AOI When this bit is cleared the register write to AOI1, AOI2, AOI3, AOI1, XBAR1, XBAR2 and XBAR3 will be ignored so that their configuration won't be changed unexpectedly. 0b - Write is disabled 1b - Write is enabled

16.6.1.25 XBAR trigger synchronizer control register1 (XBAR_TRIG_SYNC_CTRL1)

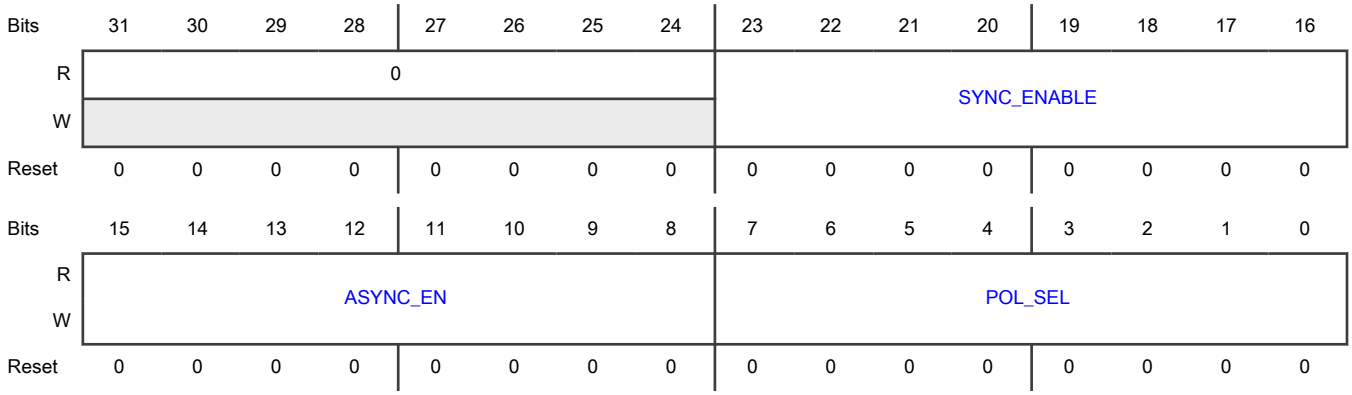
Offset

Register	Offset
XBAR_TRIG_SYNC_CTRL1	7Ch

Function

This register contains XBAR trigger synchronizer configuration bits.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 SYNC_ENABLE	Trigger out synchronizer enable The XBAR trigger synchronizer include eight channels and each channel can synchronize one trigger in to trigger out. The fields include eight bits to enable the corresponding channel. The channel should be disabled with this field before change other fields of XBAR_TRIG_SYNC_CTRL1/2 register. 0000_0000b - Channel is disabled 0000_0001b - Channel is enabled
15-8 ASYNC_EN	Asynchronous trigger in enable Set the corresponding bit of this field to 1 when the channel's trigger in is asynchronous to XBAR clock so that the trigger in can be captured correctly. 0000_0000b - Trigger in is synchronous 0000_0001b - Trigger in is asynchronous
7-0 POL_SEL	Trigger out polarity select This field controls the polarity of the trigger out for the corresponding channel. 0000_0000b - Same as trigger in 0000_0001b - Invert trigger in

16.6.1.26 XBAR trigger synchronizer control register2 (XBAR_TRIG_SYNC_CTRL2)

Offset

Register	Offset
XBAR_TRIG_SYNC_CTRL2	80h

Function

The XBAR trigger synchronizer include eight channels and each channel can synchronize one trigger in to trigger out. This register contains the pulse width control for each channel's trigger out.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PULSE_WIDTH7				PULSE_WIDTH6				PULSE_WIDTH5				PULSE_WIDTH4			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PULSE_WIDTH3				PULSE_WIDTH2				PULSE_WIDTH1				PULSE_WIDTH0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 PULSE_WIDTH7	Pulse width control register of channel7 This field controls the pulse width of the channel7 trigger out when synchronizer is enabled for channel7. The trigger out width is PULSE_WIDTH7+1 cycles of the xbar bus clock.
27-24 PULSE_WIDTH6	Pulse width control register of channel6 This field controls the pulse width of the channel6 trigger out when synchronizer is enabled for channel6. The trigger out width is PULSE_WIDTH6+1 cycles of the xbar bus clock.
23-20 PULSE_WIDTH5	Pulse width control register of channel5 This field controls the pulse width of the channel5 trigger out when synchronizer is enabled for channel5. The trigger out width is PULSE_WIDTH5+1 cycles of the xbar bus clock.
19-16 PULSE_WIDTH4	Pulse width control register of channel4 This field controls the pulse width of the channel4 trigger out when synchronizer is enabled for channel4. The trigger out width is PULSE_WIDTH4+1 cycles of the xbar bus clock.
15-12 PULSE_WIDTH3	Pulse width control register of channel3 This field controls the pulse width of the channel3 trigger out when synchronizer is enabled for channel3. The trigger out width is PULSE_WIDTH3+1 cycles of the xbar bus clock.
11-8 PULSE_WIDTH2	Pulse width control register of channel2 This field controls the pulse width of the channel2 trigger out when synchronizer is enabled for channel2. The trigger out width is PULSE_WIDTH2+1 cycles of the xbar bus clock.
7-4 PULSE_WIDTH1	Pulse width control register of channel1 This field controls the pulse width of the channel1 trigger out when synchronizer is enabled for channel1. The trigger out width is PULSE_WIDTH1+1 cycles of the xbar bus clock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 PULSE_WIDTH0	Pulse width control register of channel0 This field controls the pulse width of the channel0 trigger out when synchronizer is enabled for channel0. The trigger out width is PULSE_WIDTH0+1 cycles of the xbar bus clock.

16.6.1.27 NETC link configuration for port0 (NETC_LINK_CFG0)

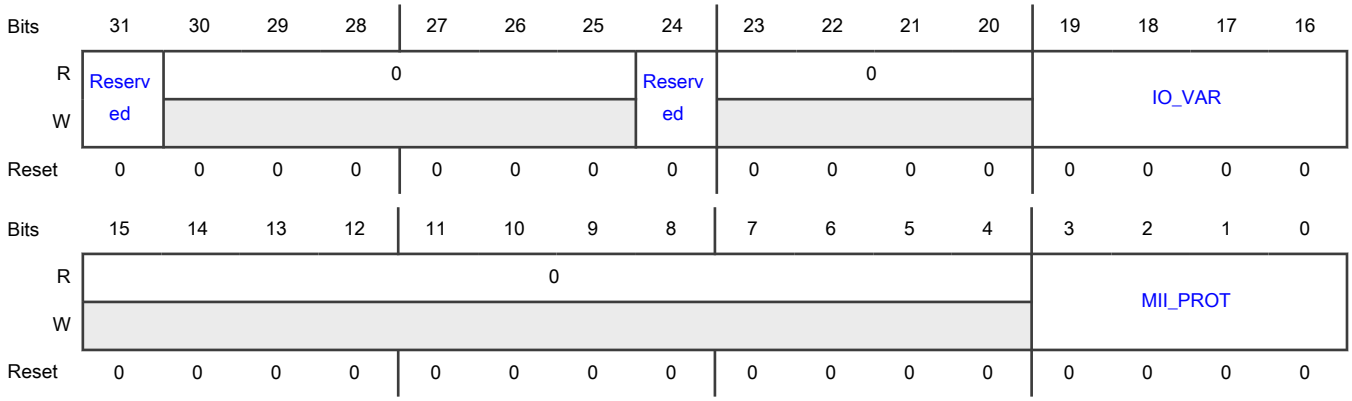
Offset

Register	Offset
NETC_LINK_CFG0	100h

Function

This register provides various link protocol configurations of NETC Port0

Diagram



Fields

Field	Function
31 —	Reserved
30-25 —	Reserved
24 —	Reserved
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-16 IO_VAR	IO variant selection 0000b - None 0001b-1110b - Reserved
15-4 —	Reserved
3-0 MII_PROT	MII protocol selection 0000b - MII 0001b - RMII 0010b - RGMII 0100b-1111b - Reserved

16.6.1.28 NETC link configuration for port1 (NETC_LINK_CFG1)

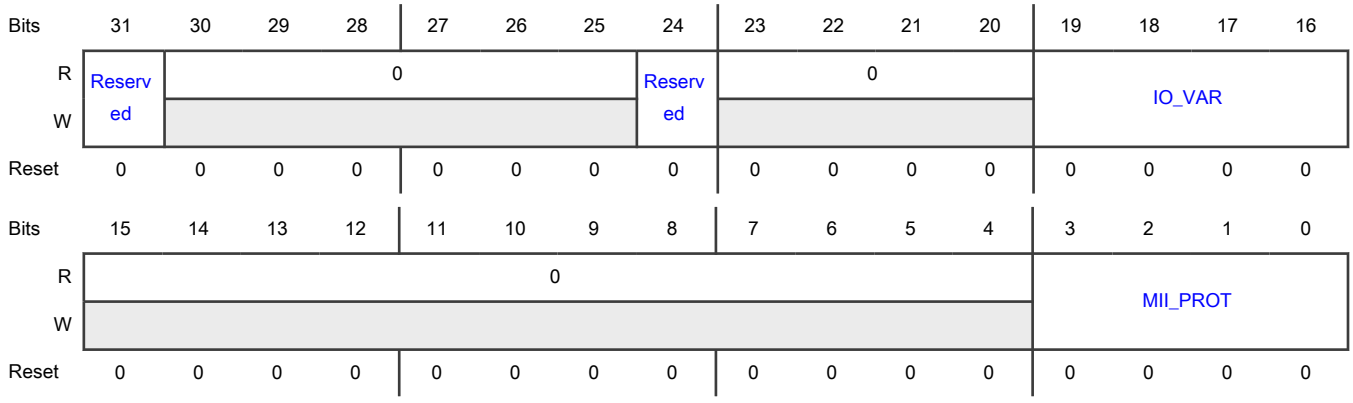
Offset

Register	Offset
NETC_LINK_CFG1	104h

Function

This register provides various link protocol configurations of NETC Port1

Diagram



Fields

Field	Function
31 —	Reserved
30-25 —	Reserved
24 —	Reserved
23-20 —	Reserved
19-16 IO_VAR	IO variant selection 0000b - None 0001b-1110b - Reserved
15-4 —	Reserved
3-0 MII_PROT	MII protocol selection 0000b - MII 0001b - RMII 0010b - RGMII 0100b-1111b - Reserved

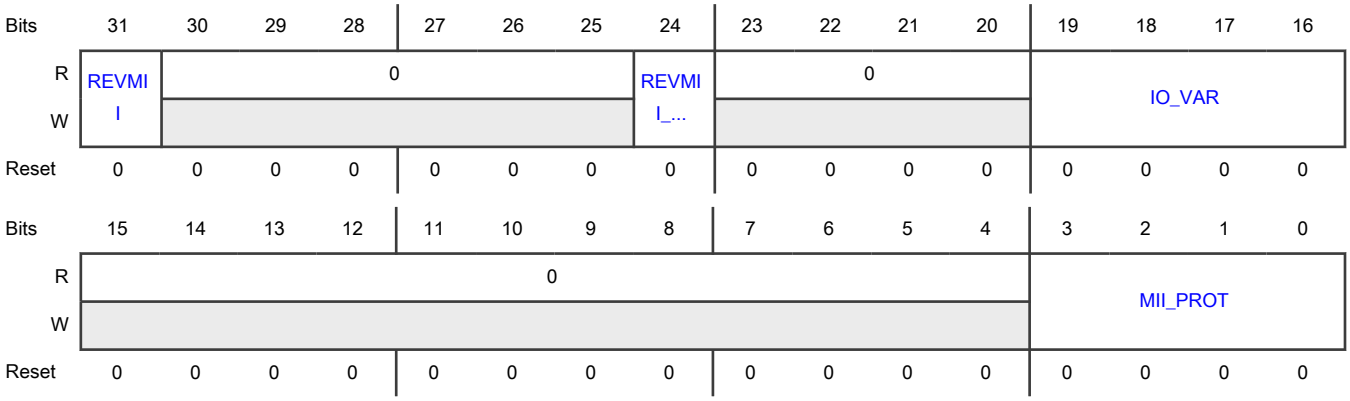
16.6.1.29 NETC link configuration for port2 (NETC_LINK_CFG2)

Offset

Register	Offset
NETC_LINK_CFG2	108h

Function
This register provides various link protocol configurations of NETC Port2

Diagram



Fields

Field	Function
31 REVMII	RevMII selection 0: RevMII not selected 1: RevMII selected MAC to link mapping is hard-coded, so to support RevMII, both the MAC and the SoC link need to support RevMII.
30-25 —	Reserved
24 REVMII_RATE	When REVMII=1 and MII_PROT=MII, this bit configures RevMII rates, otherwise this field has no meaning. 0b - MII interface is operating at 100Mbps 1b - MII interface is operating at 10Mbps
23-20 —	Reserved
19-16 IO_VAR	IO variant selection 0000b - None 0001b-1110b -
15-4 —	Reserved
3-0 MII_PROT	MII protocol selection 0000b - MII 0001b - RMII 0010b - RGMII 0100b-1111b - Reserved

16.6.1.30 NETC link configuration for port3 (NETC_LINK_CFG3)

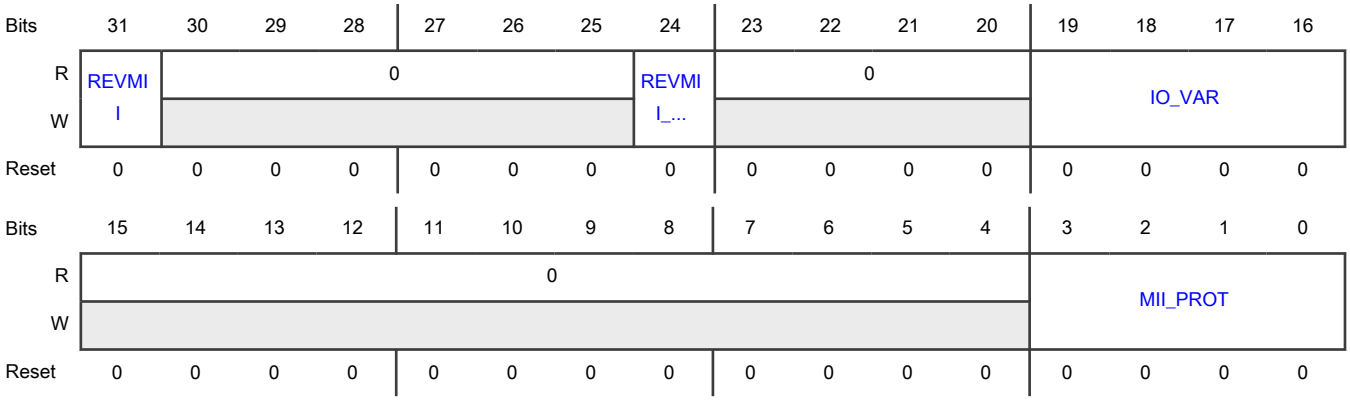
Offset

Register	Offset
NETC_LINK_CFG3	10Ch

Function

This register provides various link protocol configurations of NETC Port3

Diagram



Fields

Field	Function
31 REVMII	RevMII selection 0: RevMII not selected 1: RevMII selected MAC to link mapping is hard-coded, so to support RevMII, both the MAC and the SoC link need to support RevMII.
30-25 —	Reserved
24 REVMII_RATE	When REVMII=1 and MII_PROT=MII, this bit configures RevMII rates, otherwise this field has no meaning. 0: MII interface is operating at 100Mbps 1: MII interface is operating at 10Mbps
23-20 —	Reserved
19-16 IO_VAR	IO variant selection 0: None

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1-14: Reserved
15-4 —	Reserved
3-0 MII_PROT	MII protocol selection 0: MII 1: RMII 2: RGMII All other settings reserved.

16.6.1.31 NETC link configuration for port4 (NETC_LINK_CFG4)

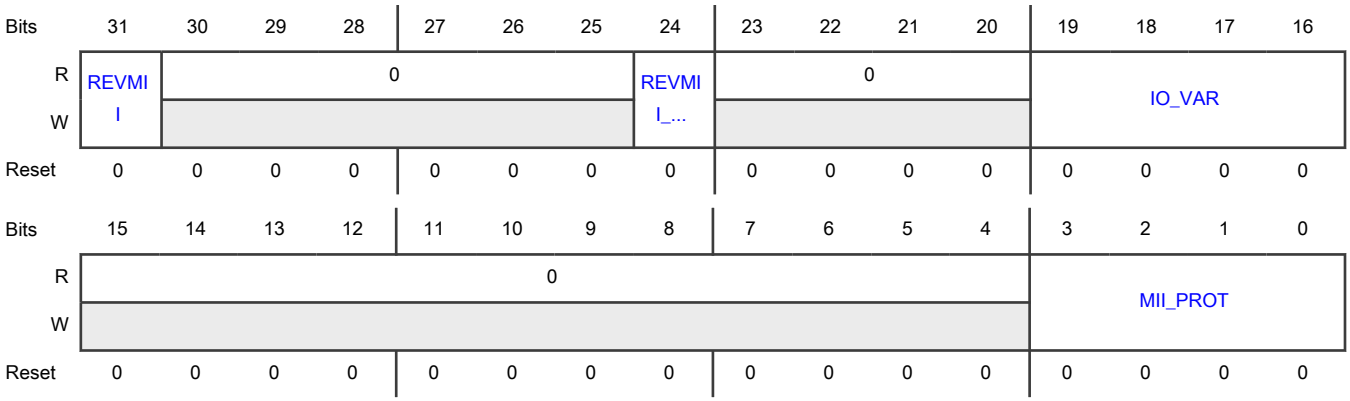
Offset

Register	Offset
NETC_LINK_CFG4	110h

Function

This register provides various link protocol configurations of NETC Port4

Diagram



Fields

Field	Function
31 REVMII	RevMII selection

Table continues on the next page...

Table continued from the previous page...

Field	Function
	MAC to link mapping is hard-coded, so to support RevMII, both the MAC and the SoC link need to support RevMII. 0b - RevMII not selected 1b - RevMII selected
30-25 —	Reserved
24 REVMII_RATE	When REVMII=1 and MII_PROT=MII, this bit configures RevMII rates, otherwise this field has no meaning. 0: MII interface is operating at 100Mbps 1: MII interface is operating at 10Mbps
23-20 —	Reserved
19-16 IO_VAR	IO variant selection 0: None 1-14: Reserved
15-4 —	Reserved
3-0 MII_PROT	MII protocol selection 0: MII 1: RMII 2: RGMII All other settings reserved.

16.6.1.32 NETC RevMII RGMII delay line configuration for port0 (NETC_REVMII_DLL0)

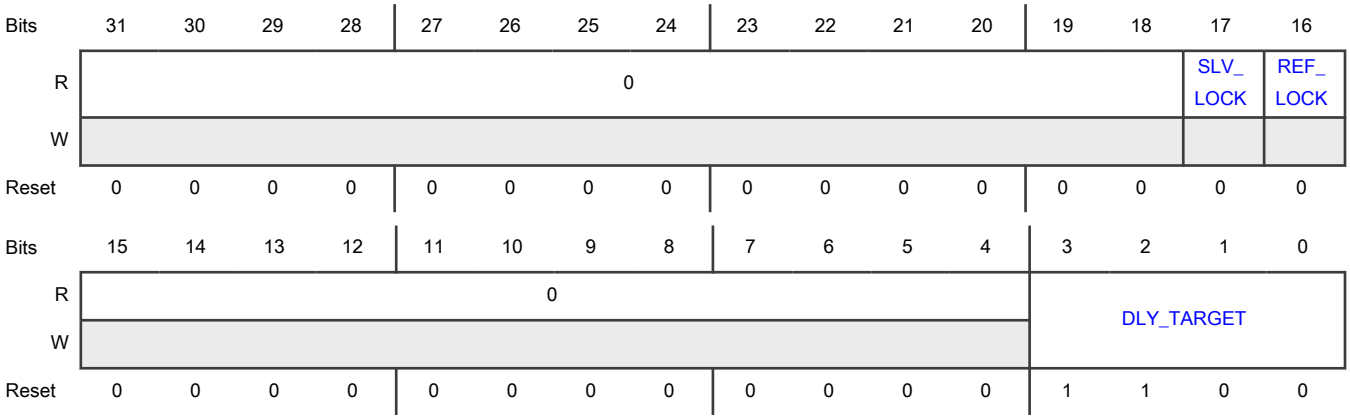
Offset

Register	Offset
NETC_REVMII_DLL0	114h

Function

This register provides delay configuration for NETC Port0 TXC and RXC in RevMII RGMII mode

Diagram



Fields

Field	Function
31-18 —	Reserved
17 SLV_LOCK	Slave delay line lock flag 0b - Slave delay line is not locked 1b - Slave delay line is locked
16 REF_LOCK	Reference delay line lock flag 0b - Reference delay line is not locked 1b - Reference delay line is locked
15-4 —	Reserved
3-0 DLY_TARGET	Delay target of slave delay line The delay target is: ((SLVDLYTARGET+1) * 1/32 * clock cycle of reference clock. Reference clock is fixed to 200MHz clock sourced from PLL 1G.

16.6.1.33 NETC RevMII RGMII delay line configuration for port1 (NETC_REVMII_DLL1)

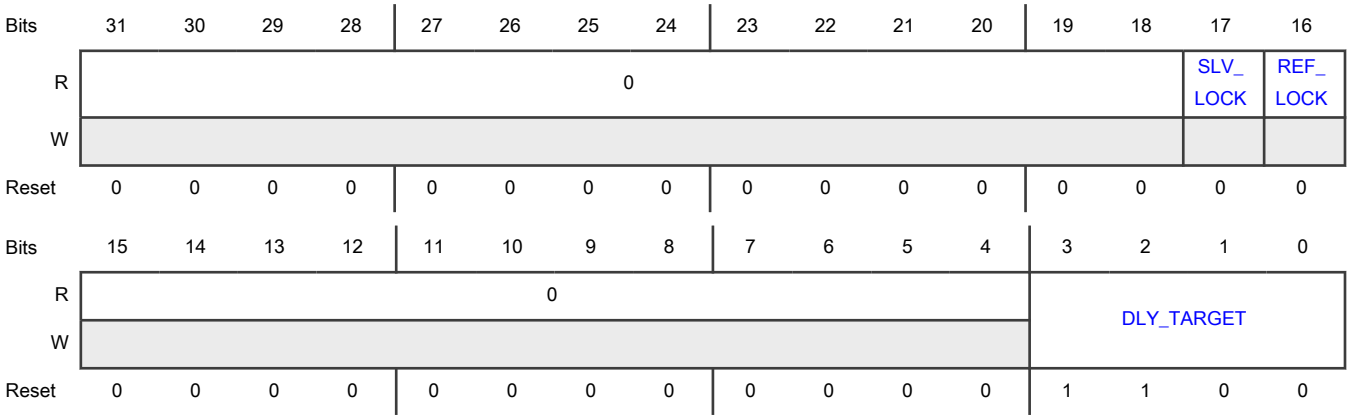
Offset

Register	Offset
NETC_REVMII_DLL1	118h

Function

This register provides delay configuration for NETC Port1 TXC and RXC in RevMII RGMII mode

Diagram



Fields

Field	Function
31-18 —	Reserved
17 SLV_LOCK	Slave delay line lock flag 0b - Slave delay line is not locked 1b - Slave delay line is locked
16 REF_LOCK	Reference delay line lock flag 0b - Reference delay line is not locked 1b - Reference delay line is locked
15-4 —	Reserved
3-0 DLY_TARGET	Delay target of slave delay line The delay target is: ((SLVDLYTARGET+1) * 1/32 * clock cycle of reference clock. Reference clock is fixed to 200MHz clock sourced from PLL 1G.

16.6.1.34 NETC RevMII RGMII delay line configuration for port2 (NETC_REVMII_DLL2)

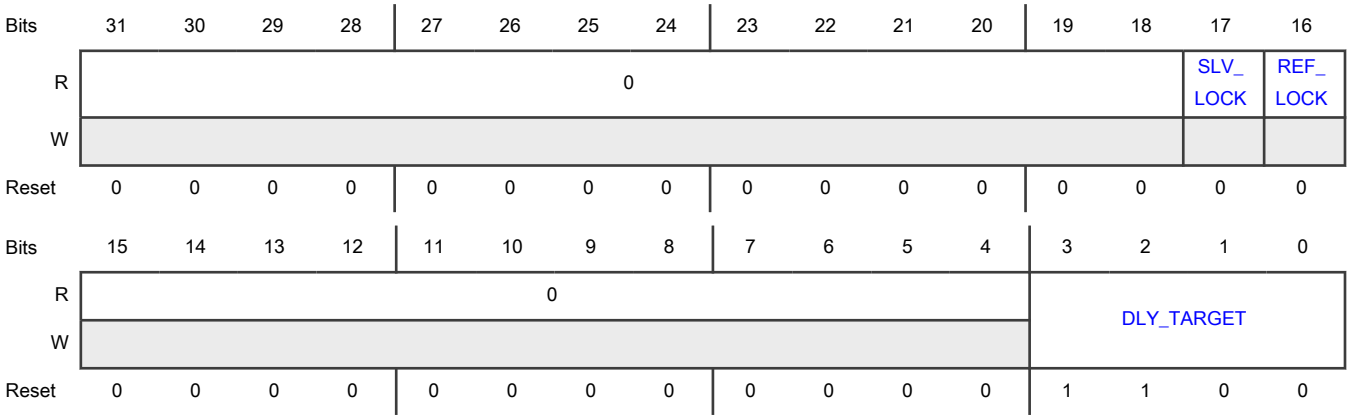
Offset

Register	Offset
NETC_REVMII_DLL2	11Ch

Function

This register provides delay configuration for NETC Port2 TXC and RXC in RevMII RGMII mode

Diagram



Fields

Field	Function
31-18 —	Reserved
17 SLV_LOCK	Slave delay line lock flag 0b - Slave delay line is not locked 1b - Slave delay line is locked
16 REF_LOCK	Reference delay line lock flag 0b - Reference delay line is not locked 1b - Reference delay line is locked
15-4 —	Reserved
3-0 DLY_TARGET	Delay target of slave delay line The delay target is: ((SLVDLYTARGET+1) * 1/32 * clock cycle of reference clock. Reference clock is fixed to 200MHz clock sourced from PLL 1G.

16.6.1.35 NETC RevMII RGMII delay line configuration for port3 (NETC_REVMII_DLL3)

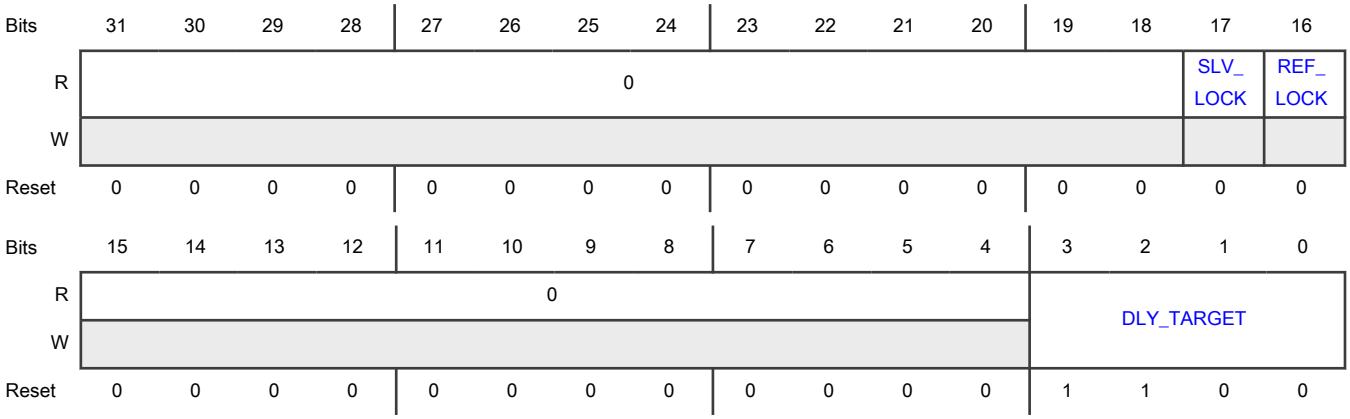
Offset

Register	Offset
NETC_REVMII_DLL3	120h

Function

This register provides delay configuration for NETC Port3 TXC and RXC in RevMII RGMII mode

Diagram



Fields

Field	Function
31-18 —	Reserved
17 SLV_LOCK	Slave delay line lock flag 0b - Slave delay line is not locked 1b - Slave delay line is locked
16 REF_LOCK	Reference delay line lock flag 0b - Reference delay line is not locked 1b - Reference delay line is locked
15-4 —	Reserved
3-0 DLY_TARGET	Delay target of slave delay line The delay target is: ((SLVDLYTARGET+1) * 1/32 * clock cycle of reference clock. Reference clock is fixed to 200MHz clock sourced from PLL 1G.

16.6.1.36 NETC RevMII RGMII delay line configuration for port4 (NETC_REVMII_DLL4)

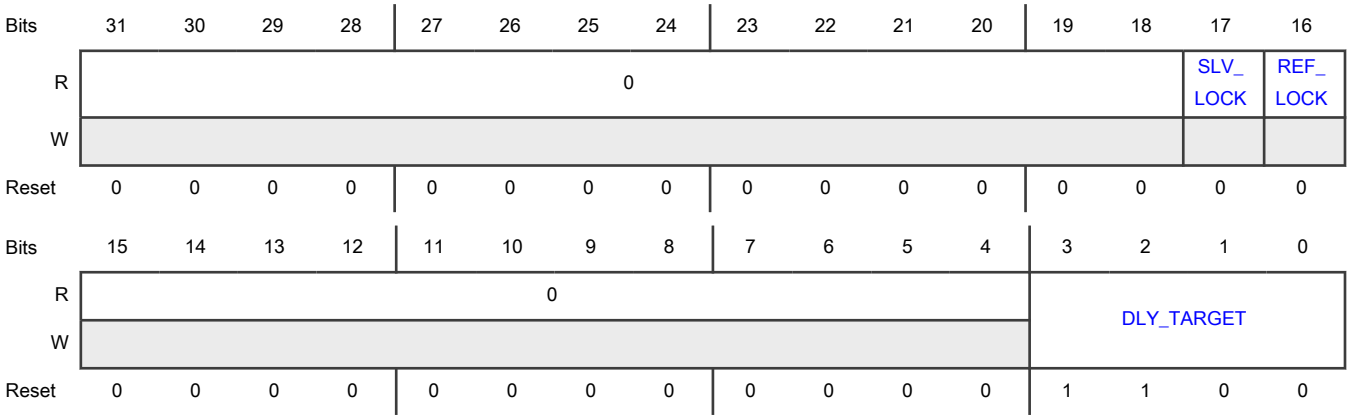
Offset

Register	Offset
NETC_REVMII_DLL4	124h

Function

This register provides delay configuration for NETC Port4 TXC and RXC in RevMII RGMII mode

Diagram



Fields

Field	Function
31-18 —	Reserved
17 SLV_LOCK	Slave delay line lock flag 0b - Slave delay line is not locked 1b - Slave delay line is locked
16 REF_LOCK	Reference delay line lock flag 0b - Reference delay line is not locked 1b - Reference delay line is locked
15-4 —	Reserved
3-0 DLY_TARGET	Delay target of slave delay line The delay target is: ((SLVDLYTARGET+1) * 1/32 * clock cycle of reference clock. Reference clock is fixed to 200MHz clock sourced from PLL 1G.

16.6.1.37 Safety clock monitor control and status register (SAFETY_CLK_MON_CS)

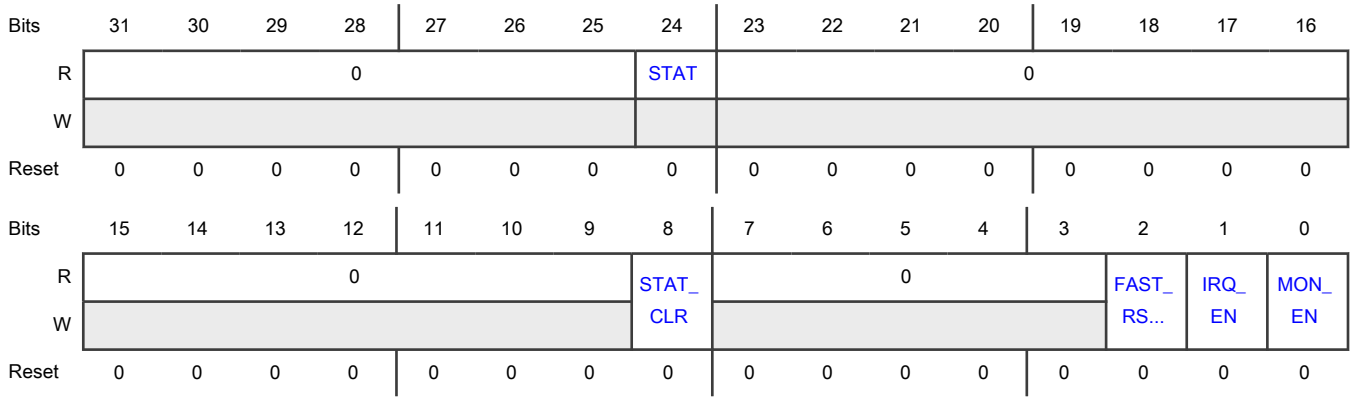
Offset

Register	Offset
SAFETY_CLK_MON_CS	130h

Function

This register provides control and status bits for the safety clock monitor

Diagram



Fields

Field	Function
31-25 —	Reserved
24 STAT	XBAR_OUT220 clock failure status 0b - No failure detected by the monitor 1b - Clock failure has been detected by the monitor
23-9 —	Reserved
8 STAT_CLR	Status clear 0b - No effect to clock failure status bit 1b - Clear clock failure status bit
7-3 —	Reserved
2 FAST_RST_EN	Reset out enable 0b - Clock failure will not assert EWM_OUT 1b - Clock failure will assert EWM_OUT_b immediately regardless EWM state
1 IRQ_EN	Interrupt enable 0b - Clock failure will not assert interrupt 1b - Clock failure will assert interrupt
0 MON_EN	Monitor enable bit 0b - The monitor is off 1b - The monitor is on

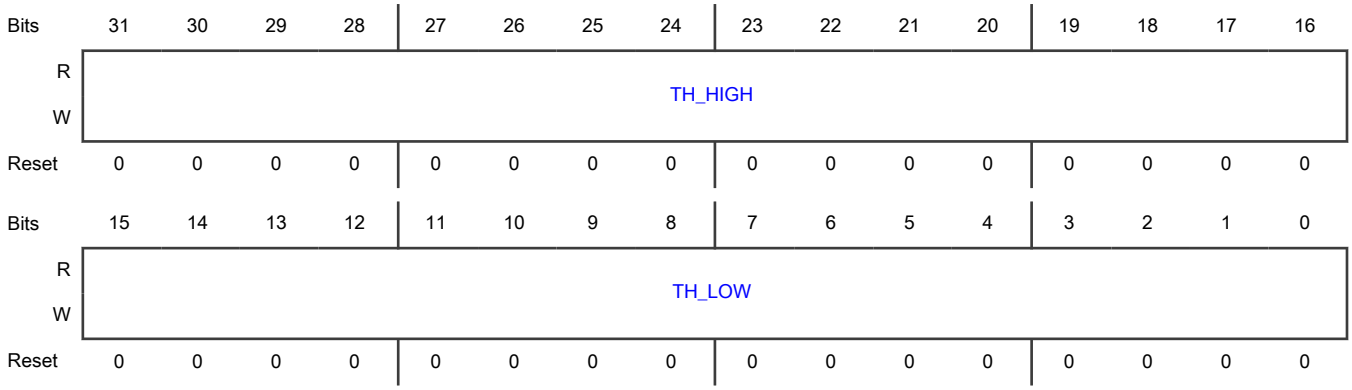
16.6.1.38 Safety clock monitor threshold register (SAFETY_CLK_MON_TH)

Offset

Register	Offset
SAFETY_CLK_MON_TH	134h

Function
This register configures the threshold value for the safety clock monitor

Diagram



Fields

Field	Function
31-16 TH_HIGH	Threshold high value When the cycle number of the monitored clock within one 32KHz clock is above this value, clock failure will be generated
15-0 TH_LOW	Threshold low value When the cycle number of the monitored clock within one 32KHz clock is below this value, clock failure will be generated

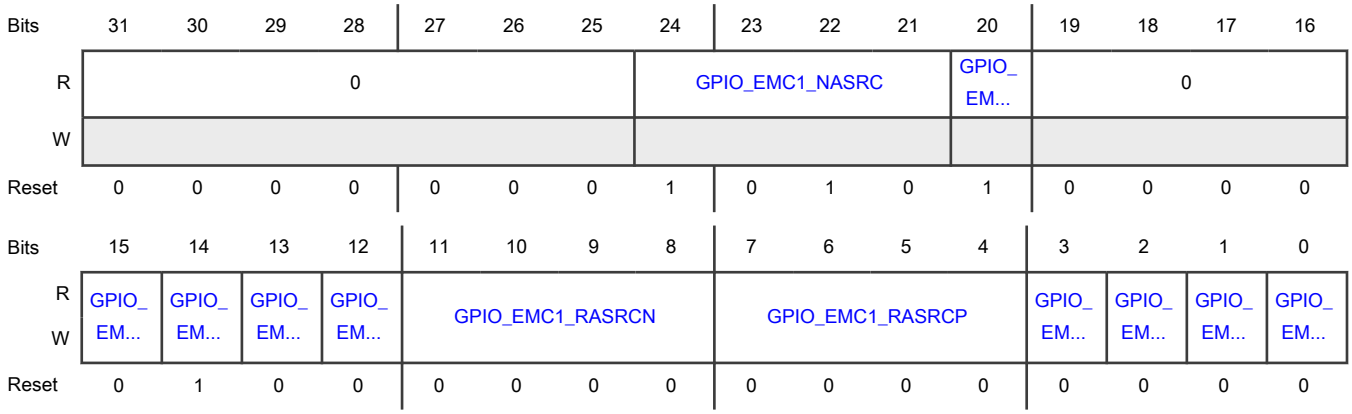
16.6.1.39 GPIO_EMC_B1 bank IO control (EMC_B1_IO_CTRL)

Offset

Register	Offset
EMC_B1_IO_CTRL	140h

Function
This register provides control to IO compensation block of GPIO_EMC_B1 IO bank

Diagram



Fields

Field	Function
31-25 —	Reserved
24-21 GPIO_EMC1_NASRC	GPIO_EMC_B1 IO bank compensation codes 4-bit NMOS compensation codes or 4-bit PMOS compensation codes selected by GPIO_EMC1_SELECT_NASRC. The meaning of codes: PVT ASRCP 3:0 and ASRCN 3:0. Slow 00001111, Typical 10100101, Fast 11110000.
20 GPIO_EMC1_COMPOK	GPIO_EMC_B1 IO bank compensation OK flag It can be high only in the Normal mode and when a new measured code is available.
19-16 —	Reserved
15 GPIO_EMC1_FASTFRZ	Compensation code fast-freeze When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. On the rising edge of FASTFRZ, it fast-freezes the compensation code at its running value. The compensation cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of FREEZE signal.
14 GPIO_EMC1_SUPLYDET_LATCH	GPIO_EMC_B1 IO bank power supply mode latch enable Supply detector cell is used to detect the IO supply range and set IO to 3V3 supply mode or 1V8 supply mode according to the detection. Setting this bit will latch the current detect result.
13 GPIO_EMC1_REFGEN_SLEEP	GPIO_EMC_B1 IO bank reference voltage generator cell sleep enable Reference voltage generator cell provides reference voltages to the IO bank. When this bit is high, it puts the reference voltage generator cell in sleep mode to reduce the static power consumption.
12	GPIO_EMC1_NASRC selection

Table continues on the next page...

Table continued from the previous page...

Field	Function
GPIO_EMC1_SELECT_NASRC	0: Show the 4-bit PMOS compensation codes in GPIO_EMC1_NASRC field 1: Show the 4-bit NMOS compensation codes in GPIO_EMC1_NASRC field
11-8 GPIO_EMC1_RASRCN	GPIO_EMC_B1 IO bank's 4-bit NMOS compensation codes from core These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.
7-4 GPIO_EMC1_RASRCP	GPIO_EMC_B1 IO bank's 4-bit PMOS compensation codes from core These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.
3 GPIO_EMC1_FASTFRZ_EN	Compensation code fast freeze enable When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. Fast Freeze mode is activated on FASTFRZ rising edge. The compensation cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of Freeze mode.
2 GPIO_EMC1_COMPEN	COMPEN and COMPTQ control the operating modes of the compensation cell Normal Mode: In this mode, the macro cell constantly tracks the Process Voltage temperature (PVT) condition of the chip and generates an 8-bit digital code. This 8-bit code is referred to as f(PVT) and it represents current PVT state. In Normal mode, the internal reference current generators are active and power consumption is higher than that in all the other modes. Freeze mode: This mode is used where the current consumption is to be kept low. In this mode, all the internal blocks are switched off to reduce the static power consumption. Internal latches keep the code, which is calculated before the cell enters the Freeze mode. Fast Freeze mode: This mode is used to freeze the digital compensation codes, when the data is transferred from the compensated IOs on a chip to an external device during Burst mode. This ensures signal integrity and eliminates the jitter caused due to modification in IO driver strength. In this mode, the current consumption is comparable to consumption in the Normal mode. Read Mode: In this mode, it is possible to force the digital codes from the chip core logic. The bandgap, measurement block, and comparator blocks are switched off to reduce static power consumption to minimum value. Fixed Code mode: The digital code is forced to a fixed value, obtained at typical PVT conditions and represents typical bit patterns. The bandgap, measurement, and other blocks are switched off to reduce static power consumption to minimum value.

COMPEN	COMPTQ	FREEZE	FASTFRZ_EN	Mode
0	0	0	0	Normal Mode
0	0	1	0	Freeze Mode
0	0	0	1	Fast Freeze Mode
1	1	0	0	Read Mode

Table continued from the previous page...

Field	Function
	<table><tr><th>COMPEN</th><th>COMPTQ</th><th>FREEZE</th><th>FASTFRZ_EN</th><th>Mode</th></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>Fixed Code Mode</td></tr></table>
1 GPIO_EMC1_COMPTQ	COMPEN and COMPTQ control the operating modes of the compensation cell
0 GPIO_EMC1_FREEZE	Compensation code freeze When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, high, low, compensation cell Freeze mode is enabled. On the rising edge of FREEZE, it freezes the compensation code at its running value. The compensation cell delivers refreshed compensation code at a delay after signal falling edge.

16.6.1.40 GPIO_EMC_B2 bank IO control (EMC_B2_IO_CTRL)

Offset

Register	Offset
EMC_B2_IO_CTRL	144h

Function

This register provides control to IO compensation block of GPIO_EMC_B2 IO bank

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							GPIO_EMC2_NASRC				GPIO_EM...	0			
W																
Reset	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPIO_EM...	GPIO_EM...	GPIO_EM...	GPIO_EM...	GPIO_EMC2_RASRCN				GPIO_EMC2_RASRCP				GPIO_EM...	GPIO_EM...	GPIO_EM...	GPIO_EM...
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
24-21 GPIO_EMC2_NASRC	GPIO_EMC_B2 IO bank compensation codes 4-bit NMOS compensation codes or 4-bit PMOS compensation codes selected by GPIO_EMC2_SELECT_NASRC.
20 GPIO_EMC2_COMPOK	GPIO_EMC_B2 IO bank compensation OK flag It can be high only in the Normal mode and when a new measured code is available.
19-16 —	Reserved
15 GPIO_EMC2_FASTFRZ	Compensation code fast-freeze When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. On the rising edge of FASTFRZ, it fast-freezes the compensation code at its running value. The compensation cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of FREEZE signal.
14 GPIO_EMC2_SUPLYDET_LATCH	GPIO_EMC_B2 IO bank power supply mode latch enable Supply detector cell is used to detect the IO supply range and set IO to 3V3 supply mode or 1V8 supply mode according to the detection. Setting this bit will latch the current detect result.
13 GPIO_EMC2_REFGEN_SLEEP	GPIO_EMC_B2 IO bank reference voltage generator cell sleep enable Reference voltage generator cell provides reference voltages to the IO bank. When this bit is high, it puts the reference voltage generator cell in sleep mode to reduce the static power consumption.
12 GPIO_EMC2_SELECT_NASRC	GPIO_EMC2_NASRC selection 0b - Show the 4-bit PMOS compensation codes in GPIO_EMC2_NASRC field 1b - Show the 4-bit NMOS compensation codes in GPIO_EMC2_NASRC field
11-8 GPIO_EMC2_RASRCN	GPIO_EMC_B2 IO bank's 4-bit NMOS compensation codes from core These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.
7-4 GPIO_EMC2_RASRCP	GPIO_EMC_B2 IO bank's 4-bit PMOS compensation codes from core These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.
3 GPIO_EMC2_FASTFRZ_EN	Compensation code fast freeze enable When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. Fast Freeze mode is activated on FASTFRZ rising edge. The compensation cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of Freeze mode.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>2
GPIO_EMC2_COMPEN</td><td>COMPEN and COMPTQ control the operating modes of the compensation cell
Normal Mode: In this mode, the macro cell constantly tracks the PVT condition of the chip and generates an 8-bit digital code. This 8-bit code is referred to as f(PVT) and it represents current PVT state. In Normal mode, the internal reference current generators are active and power consumption is higher than that in all the other modes.
Freeze mode: This mode is used where the current consumption is to be kept low. In this mode, all the internal blocks are switched off to reduce the static power consumption. Internal latches keep the code, which is calculated before the cell enters the Freeze mode.
Fast Freeze mode: This mode is used to freeze the digital compensation codes, when the data is transferred from the compensated IOs on a chip to an external device during Burst mode. This ensures signal integrity and eliminates the jitter caused due to modification in IO driver strength. In this mode, the current consumption is comparable to consumption in the Normal mode.
Read Mode: In this mode, it is possible to force the digital codes from the chip core logic. The bandgap, measurement block, and comparator blocks are switched off to reduce static power consumption to minimum value.
Fixed Code mode: The digital code is forced to a fixed value, obtained at typical PVT conditions and represents typical bit patterns. The bandgap, measurement, and other blocks are switched off to reduce static power consumption to minimum value.
<table>
<tr><th>COMPEN</th><th>COMPTQ</th><th>FREEZE</th><th>FASTFRZ_EN</th><th>Mode</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>Normal Mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>Freeze Mode</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>Fast Freeze Mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>Read Mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>Fixed Code Mode</td></tr>
</table>
</td></tr>
<tr><td>1
GPIO_EMC2_COMPTQ</td><td>COMPEN and COMPTQ control the operating modes of the compensation cell</td></tr>
<tr><td>0
GPIO_EMC2_FREEZE</td><td>Compensation code freeze
When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, high, low, compensation cell Freeze mode is enabled. On the rising edge of FREEZE, it freezes the compensation code at its running value. The compensation cell delivers refreshed compensation code at a delay after signal falling edge.</td></tr>
</table>

16.6.1.41 GPIO_SD_B1 bank IO control (SD_B1_IO_CTRL)

Offset

Register	Offset
SD_B1_IO_CTRL	148h

Function

This register provides control to IO compensation block of GPIO_SD_B1 IO bank

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								GPIO_SD1_NASRC			GPIO_SD...	0			
W																
Reset	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPIO_SD...	GPIO_SD...	GPIO_SD...	GPIO_SD...	GPIO_SD1_RASRCN				GPIO_SD1_RASRCP				GPIO_SD...	GPIO_SD...	GPIO_SD...	GPIO_SD...
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved
24-21 GPIO_SD1_NASRC	GPIO_SD_B1 IO bank compensation codes 4-bit NMOS compensation codes or 4-bit PMOS compensation codes selected by GPIO_SD1_SELECT_NASRC.
20 GPIO_SD1_COMPOK	GPIO_SD_B1 IO bank compensation OK flag It can be high only in the Normal mode and when a new measured code is available.
19-16 —	Reserved
15 GPIO_SD1_FASTFRZ	Compensation code fast-freeze When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. On the rising edge of FASTFRZ, it fast-freezes the compensation code at its running value. The compensation cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of FREEZE signal.
14 GPIO_SD1_SUPLYDET_LATCH	GPIO_SD_B1 IO bank power supply mode latch enable Supply detector cell is used to detect the IO supply range and set IO to 3V3 supply mode or 1V8 supply mode according to the detection. Setting this bit will latch the current detect result.
13	GPIO_SD_B1 IO bank reference voltage generator cell sleep enable

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>GPIO_SD1_REFGEN_SLEEP</td><td>Reference voltage generator cell provides reference voltages to the IO bank. When this bit is high, it puts the reference voltage generator cell in sleep mode to reduce the static power consumption.</td></tr>
<tr><td>12
GPIO_SD1_SELECT_NASRC</td><td>GPIO_SD1_NASRC selection
0: Show the 4-bit PMOS compensation codes in GPIO_SD1_NASRC field
1: Show the 4-bit NMOS compensation codes in GPIO_SD1_NASRC field</td></tr>
<tr><td>11-8
GPIO_SD1_RASRCN</td><td>GPIO_SD_B1 IO bank's 4-bit NMOS compensation codes from core
These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.</td></tr>
<tr><td>7-4
GPIO_SD1_RASRCP</td><td>GPIO_SD_B1 IO bank's 4-bit PMOS compensation codes from core
These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.</td></tr>
<tr><td>3
GPIO_SD1_FASTFRZ_EN</td><td>Compensation code fast freeze enable
When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. Fast Freeze mode is activated on FASTFRZ rising edge. The compensation cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of Freeze mode.</td></tr>
<tr><td>2
GPIO_SD1_COMPEN</td><td>COMPEN and COMPTQ control the operating modes of the compensation cell
Normal Mode: In this mode, the macro cell constantly tracks the PVT condition of the chip and generates an 8-bit digital code. This 8-bit code is referred to as f(PVT) and it represents current PVT state. In Normal mode, the internal reference current generators are active and power consumption is higher than that in all the other modes.
Freeze mode: This mode is used where the current consumption is to be kept low. In this mode, all the internal blocks are switched off to reduce the static power consumption. Internal latches keep the code, which is calculated before the cell enters the Freeze mode.
Fast Freeze mode: This mode is used to freeze the digital compensation codes, when the data is transferred from the compensated IOs on a chip to an external device during Burst mode. This ensures signal integrity and eliminates the jitter caused due to modification in IO driver strength. In this mode, the current consumption is comparable to consumption in the Normal mode.
Read Mode: In this mode, it is possible to force the digital codes from the chip core logic. The bandgap, measurement block, and comparator blocks are switched off to reduce static power consumption to minimum value.
Fixed Code mode: The digital code is forced to a fixed value, obtained at typical PVT conditions and represents typical bit patterns. The bandgap, measurement, and other blocks are switched off to reduce static power consumption to minimum value.
<table>
<tr><th>COMPEN</th><th>COMPTQ</th><th>FREEZE</th><th>FASTFRZ_EN</th><th>Mode</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>Normal Mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>Freeze Mode</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

Field	Function
	<table><tr><th>COMPEN</th><th>COMPTQ</th><th>FREEZE</th><th>FASTFRZ_EN</th><th>Mode</th></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>Fast Freeze Mode</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td><td>Read Mode</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>Fixed Code Mode</td></tr></table>
1 GPIO_SD1_COMPTQ	COMPEN and COMPTQ control the operating modes of the compensation cell
0 GPIO_SD1_FREEZE	Compensation code freeze When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, high, low, compensation cell Freeze mode is enabled. On the rising edge of FREEZE, it freezes the compensation code at its running value. The compensation cell delivers refreshed compensation code at a delay after signal falling edge.

16.6.1.42 GPIO_SD_B2 bank IO control (SD_B2_IO_CTRL)

Offset

Register	Offset
SD_B2_IO_CTRL	14Ch

Function

This register provides control to IO compensation block of GPIO_SD_B2 IO bank

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								GPIO_SD2_NASRC			GPIO_SD...	0			
W																
Reset	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPIO_SD...	GPIO_SD...	GPIO_SD...	GPIO_SD...	GPIO_SD2_RASRCN				GPIO_SD2_RASRCP				GPIO_SD...	GPIO_SD...	GPIO_SD...	GPIO_SD...
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved
24-21 GPIO_SD2_NASRC	GPIO_SD_B2 IO bank compensation codes 4-bit NMOS compensation codes or 4-bit PMOS compensation codes selected by GPIO_SD2_SELECT_NASRC.
20 GPIO_SD2_COMPOK	GPIO_SD_B2 IO bank compensation OK flag It can be high only in the Normal mode and when a new measured code is available.
19-16 —	Reserved
15 GPIO_SD2_FASTFRZ	Compensation code fast-freeze When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. On the rising edge of FASTFRZ, it fast-freezes the compensation code at its running value. The compensation cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of FREEZE signal.
14 GPIO_SD2_SUPLYDET_LATCH	GPIO_SD_B2 IO bank power supply mode latch enable Supply detector cell is used to detect the IO supply range and set IO to 3V3 supply mode or 1V8 supply mode according to the detection. Setting this bit will latch the current detect result.
13 GPIO_SD2_REFGEN_SLEEP	GPIO_SD_B2 IO bank reference voltage generator cell sleep enable Reference voltage generator cell provides reference voltages to the IO bank. When this bit is high, it puts the reference voltage generator cell in sleep mode to reduce the static power consumption.
12 GPIO_SD2_SELECT_NASRC	GPIO_SD2_NASRC selection 0: Show the 4-bit PMOS compensation codes in GPIO_SD2_NASRC field 1: Show the 4-bit NMOS compensation codes in GPIO_SD2_NASRC field
11-8 GPIO_SD2_RASRCN	GPIO_SD_B2 IO bank's 4-bit NMOS compensation codes from core These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.
7-4 GPIO_SD2_RASRCP	GPIO_SD_B2 IO bank's 4-bit PMOS compensation codes from core These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.
3 GPIO_SD2_FASTFRZ_EN	Compensation code fast freeze enable When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. Fast Freeze mode is activated on FASTFRZ rising edge. The compensation

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of Freeze mode.</td></tr>
<tr><td>2

GPIO_SD2_COMPEN</td><td>COMPEN and COMPTQ control the operating modes of the compensation cell

Normal Mode: In this mode, the macro cell constantly tracks the PVT condition of the chip and generates an 8-bit digital code. This 8-bit code is referred to as f(PVT) and it represents current PVT state. In Normal mode, the internal reference current generators are active and power consumption is higher than that in all the other modes.

Freeze mode: This mode is used where the current consumption is to be kept low. In this mode, all the internal blocks are switched off to reduce the static power consumption. Internal latches keep the code, which is calculated before the cell enters the Freeze mode.

Fast Freeze mode: This mode is used to freeze the digital compensation codes, when the data is transferred from the compensated IOs on a chip to an external device during Burst mode. This ensures signal integrity and eliminates the jitter caused due to modification in IO driver strength. In this mode, the current consumption is comparable to consumption in the Normal mode.

Read Mode: In this mode, it is possible to force the digital codes from the chip core logic. The bandgap, measurement block, and comparator blocks are switched off to reduce static power consumption to minimum value.

Fixed Code mode: The digital code is forced to a fixed value, obtained at typical PVT conditions and represents typical bit patterns. The bandgap, measurement, and other blocks are switched off to reduce static power consumption to minimum value.

<table>
<tr><th>COMPEN</th><th>COMPTQ</th><th>FREEZE</th><th>FASTFRZ_EN</th><th>Mode</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>Normal Mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>Freeze Mode</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>Fast Freeze Mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>Read Mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>Fixed Code Mode</td></tr>
</table>
</td></tr>
<tr><td>1

GPIO_SD2_COMPTQ</td><td>COMPEN and COMPTQ control the operating modes of the compensation cell</td></tr>
<tr><td>0

GPIO_SD2_FREEZE</td><td>Compensation code freeze

When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, high, low, compensation cell Freeze mode is enabled. On the rising edge of FREEZE, it freezes the compensation code at its running value. The compensation cell delivers refreshed compensation code at a delay after signal falling edge.</td></tr>
</table>

16.6.1.43 GPIO_B1 bank IO control (GPIO_B1_IO_CTRL)

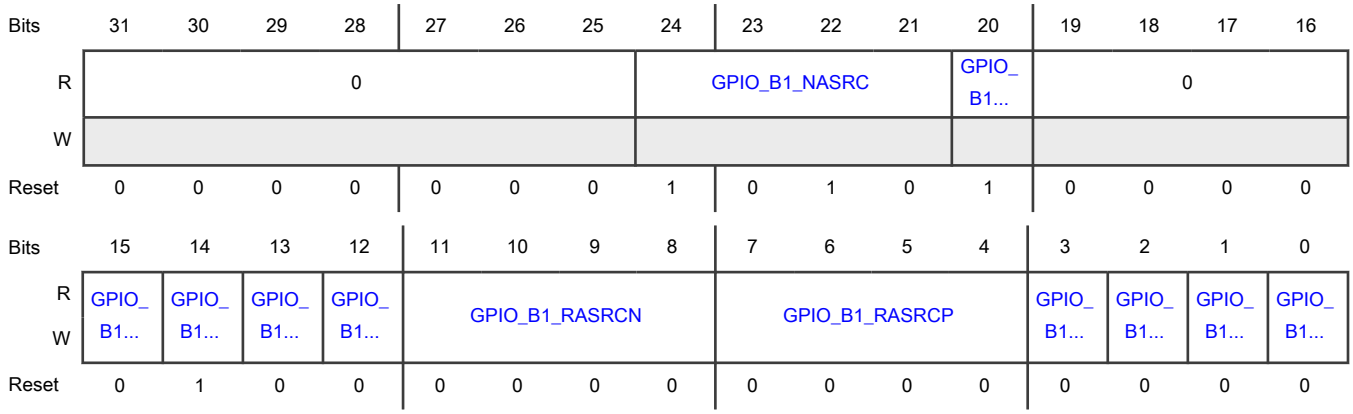
Offset

Register	Offset
GPIO_B1_IO_CTRL	150h

Function

This register provides control to IO compensation block of GPIO_B1 IO bank

Diagram



Fields

Field	Function
31-25 —	Reserved
24-21 GPIO_B1_NASRC	GPIO_B1 IO bank compensation codes 4-bit NMOS compensation codes or 4-bit PMOS compensation codes selected by GPIO_B1_SELECT_NASRC.
20 GPIO_B1_COMPOK	GPIO_B1 IO bank compensation OK flag It can be high only in the Normal mode and when a new measured code is available.
19-16 —	Reserved
15 GPIO_B1_FASTFRZ	Compensation code fast-freeze When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. On the rising edge of FASTFRZ, it fast-freezes the compensation code at its running value. The compensation cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of FREEZE signal.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 GPIO_B1_SUPLYDET_LATCH	GPIO_B1 IO bank power supply mode latch enable Supply detector cell is used to detect the IO supply range and set IO to 3V3 supply mode or 1V8 supply mode according to the detection. Setting this bit will latch the current detect result.
13 GPIO_B1_REFGEN_SLEEP	GPIO_B1 IO bank reference voltage generator cell sleep enable Reference voltage generator cell provides reference voltages to the IO bank. When this bit is high, it puts the reference voltage generator cell in sleep mode to reduce the static power consumption.
12 GPIO_B1_SELECT_NASRC	GPIO_B1_NASRC selection 0b - Show the 4-bit PMOS compensation codes in GPIO_B1_NASRC field 1b - Show the 4-bit NMOS compensation codes in GPIO_B1_NASRC field
11-8 GPIO_B1_RASRCN	GPIO_B1 IO bank's 4-bit NMOS compensation codes from core These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.
7-4 GPIO_B1_RASRCP	GPIO_B1 IO bank's 4-bit PMOS compensation codes from core These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.
3 GPIO_B1_FASTFRZ_EN	Compensation code fast freeze enable When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. Fast Freeze mode is activated on FASTFRZ rising edge. The compensation cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of Freeze mode.
2 GPIO_B1_COMPEN	COMPEN and COMPTQ control the operating modes of the compensation cell Normal Mode: In this mode, the macro cell constantly tracks the PVT condition of the chip and generates an 8-bit digital code. This 8-bit code is referred to as f(PVT) and it represents current PVT state. In Normal mode, the internal reference current generators are active and power consumption is higher than that in all the other modes. Freeze mode: This mode is used where the current consumption is to be kept low. In this mode, all the internal blocks are switched off to reduce the static power consumption. Internal latches keep the code, which is calculated before the cell enters the Freeze mode. Fast Freeze mode: This mode is used to freeze the digital compensation codes, when the data is transferred from the compensated IOs on a chip to an external device during Burst mode. This ensures signal integrity and eliminates the jitter caused due to modification in IO driver strength. In this mode, the current consumption is comparable to consumption in the Normal mode. Read Mode: In this mode, it is possible to force the digital codes from the chip core logic. The bandgap, measurement block, and comparator blocks are switched off to reduce static power consumption to minimum value. Fixed Code mode: The digital code is forced to a fixed value, obtained at typical PVT conditions and represents typical bit patterns. The bandgap, measurement, and other blocks are switched off to reduce static power consumption to minimum value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>COMPEN</th><th>COMPTQ</th><th>FREEZE</th><th>FASTFRZ_EN</th><th>Mode</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>Normal Mode</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td><td>Freeze Mode</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>Fast Freeze Mode</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td><td>Read Mode</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>Fixed Code Mode</td></tr></table>
1 GPIO_B1_COMPTQ	COMPEN and COMPTQ control the operating modes of the compensation cell
0 GPIO_B1_FREEZE	Compensation code freeze When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, high, low, compensation cell Freeze mode is enabled. On the rising edge of FREEZE, it freezes the compensation code at its running value. The compensation cell delivers refreshed compensation code at a delay after signal falling edge.

16.6.1.44 GPIO_B2 bank IO control (GPIO_B2_IO_CTRL)

Offset

Register	Offset
GPIO_B2_IO_CTRL	154h

Function

This register provides control to IO compensation block of GPIO_B2 IO bank

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								GPIO_B2_NASRC			GPIO_B2...	0			
W																
Reset	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPIO_B2...	GPIO_B2...	GPIO_B2...	GPIO_B2...	GPIO_B2_RASRCN				GPIO_B2_RASRCP				GPIO_B2...	GPIO_B2...	GPIO_B2...	GPIO_B2...
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved
24-21 GPIO_B2_NASRC	GPIO_B2 IO bank compensation codes 4-bit NMOS compensation codes or 4-bit PMOS compensation codes selected by GPIO_B2_SELECT_NASRC.
20 GPIO_B2_COMPOK	GPIO_B2 IO bank compensation OK flag It can be high only in the Normal mode and when a new measured code is available.
19-16 —	Reserved
15 GPIO_B2_FASTFRZ	Compensation code fast-freeze When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. On the rising edge of FASTFRZ, it fast-freezes the compensation code at its running value. The compensation cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of FREEZE signal.
14 GPIO_B2_SUPLYDET_LATCH	GPIO_B2 IO bank power supply mode latch enable Supply detector cell is used to detect the IO supply range and set IO to 3V3 supply mode or 1V8 supply mode according to the detection. Setting this bit will latch the current detect result.
13 GPIO_B2_REFGEN_SLEEP	GPIO_B2 IO bank reference voltage generator cell sleep enable Reference voltage generator cell provides reference voltages to the IO bank. When this bit is high, it puts the reference voltage generator cell in sleep mode to reduce the static power consumption.
12 GPIO_B2_SELECT_NASRC	GPIO_B2_NASRC selection 0: Show the 4-bit PMOS compensation codes in GPIO_B2_NASRC field 1: Show the 4-bit NMOS compensation codes in GPIO_B2_NASRC field
11-8 GPIO_B2_RASRCN	GPIO_B2 IO bank's 4-bit NMOS compensation codes from core These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.
7-4 GPIO_B2_RASRCP	GPIO_B2 IO bank's 4-bit PMOS compensation codes from core These codes are used in compensation Read mode when COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is high, high, low, low.
3 GPIO_B2_FASTFRZ_EN	Compensation code fast freeze enable When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, low, high, compensation cell Fast Freeze mode is enabled. Fast Freeze mode is activated on FASTFRZ rising edge. The compensation

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of Freeze mode.</td></tr>
<tr><td>2
GPIO_B2_COMPEN</td><td>COMPEN and COMPTQ control the operating modes of the compensation cell
Normal Mode: In this mode, the macro cell constantly tracks the PVT condition of the chip and generates an 8-bit digital code. This 8-bit code is referred to as f(PVT) and it represents current PVT state. In Normal mode, the internal reference current generators are active and power consumption is higher than that in all the other modes.
Freeze mode: This mode is used where the current consumption is to be kept low. In this mode, all the internal blocks are switched off to reduce the static power consumption. Internal latches keep the code, which is calculated before the cell enters the Freeze mode.
Fast Freeze mode: This mode is used to freeze the digital compensation codes, when the data is transferred from the compensated IOs on a chip to an external device during Burst mode. This ensures signal integrity and eliminates the jitter caused due to modification in IO driver strength. In this mode, the current consumption is comparable to consumption in the Normal mode.
Read Mode: In this mode, it is possible to force the digital codes from the chip core logic. The bandgap, measurement block, and comparator blocks are switched off to reduce static power consumption to minimum value.
Fixed Code mode: The digital code is forced to a fixed value, obtained at typical PVT conditions and represents typical bit patterns. The bandgap, measurement, and other blocks are switched off to reduce static power consumption to minimum value.
<table>
<tr><th>COMPEN</th><th>COMPTQ</th><th>FREEZE</th><th>FASTFRZ_EN</th><th>Mode</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>Normal Mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>Freeze Mode</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>Fast Freeze Mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>Read Mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>Fixed Code Mode</td></tr>
</table>
</td></tr>
<tr><td>1
GPIO_B2_COMPTQ</td><td>COMPEN and COMPTQ control the operating modes of the compensation cell</td></tr>
<tr><td>0
GPIO_B2_FREEZE</td><td>Compensation code freeze
When COMPEN, COMPTQ, FREEZE, FASTFRZ_EN is low, low, high, low, compensation cell Freeze mode is enabled. On the rising edge of FREEZE, it freezes the compensation code at its running value. The compensation cell delivers refreshed compensation code at a delay after signal falling edge.</td></tr>
</table>

16.6.1.45 Miscellaneous control register of IO (MISC_IO_CTRL)

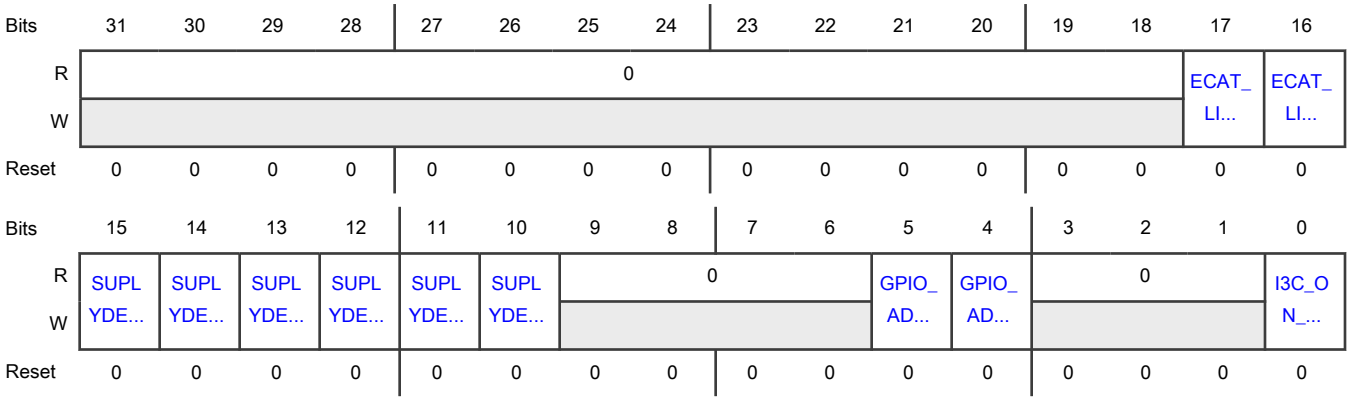
Offset

Register	Offset
MISC_IO_CTRL	158h

Function

This register provides controls to IO supply voltage detector and IO voltage range of particular IO banks

Diagram



Fields

Field	Function
31-18 —	Reserved
17 ECAT_LINK_ACT1_POL	ECAT_LINK_ACT[1] polarity control defines polarity of ECAT_LINK_ACT[1] when 1" Active High, when 0: Active low
16 ECAT_LINK_ACT0_POL	ECAT_LINK_ACT[0] polarity control defines polarity of ECAT_LINK_ACT[0] when 1" Active High, when 0: Active low
15 SUPLYDET_GPIO_B2_SLEEP	GPIO_GPIO_B1 IO bank supply voltage detector sleep mode enable Supply detector cell is used to detect the IO supply range. When this bit is high, it puts the supply detector cell in sleep mode to reduce the static power consumption. The detect result can be latched by setting SUPLYDET_LATCH bit before putting the cell in sleep mode.
14 SUPLYDET_GPIO_B1_SLEEP	GPIO_GPIO_B1 IO bank supply voltage detector sleep mode enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Supply detector cell is used to detect the IO supply range. When this bit is high, it puts the supply detector cell in sleep mode to reduce the static power consumption. The detect result can be latched by setting SUPLYDET_LATCH bit before putting the cell in sleep mode.
13 SUPLYDET_SD2_SLEEP	GPIO_SD_B2 IO bank supply voltage detector sleep mode enable Supply detector cell is used to detect the IO supply range. When this bit is high, it puts the cell in sleep mode to reduce the static power consumption. The detect result can be latched by setting SUPLYDET_LATCH bit before putting the cell in sleep mode.
12 SUPLYDET_SD1_SLEEP	GPIO_SD_B1 IO bank supply voltage detector sleep mode enable Supply detector cell is used to detect the IO supply range. When this bit is high, it puts the cell in sleep mode to reduce the static power consumption. The detect result can be latched by setting SUPLYDET_LATCH bit before putting the cell in sleep mode.
11 SUPLYDET_EMC2_SLEEP	GPIO_EMC_B2 IO bank supply voltage detector sleep mode enable Supply detector cell is used to detect the IO supply range. When this bit is high, it puts the cell in sleep mode to reduce the static power consumption. The detect result can be latched by setting SUPLYDET_LATCH bit before putting the cell in sleep mode.
10 SUPLYDET_EMC1_SLEEP	GPIO_EMC_B1 IO bank supply voltage detector sleep mode enable Supply detector cell is used to detect the IO supply range. When this bit is high, it puts the cell in sleep mode to reduce the static power consumption. The detect result can be latched by setting SUPLYDET_LATCH bit before putting the cell in sleep mode.
9-6 —	Reserved
5 GPIO_AD_LOW_RANGE	GPIO_AD IO bank supply voltage range selection for GPIO_AD_00 to GPIO_AD_17

GPIO_AD_HIGH_RANGE	GPIO_AD_LOW_RANGE	Mode
0	0	GPIO_AD_xx IO will work in continuous range mode with supply voltage in 1.71v-3.6v
0	1	GPIO_AD_xx IO will work in low range mode with supply voltage in 1.71v-1.98v
1	0	GPIO_AD_xx IO will work in high range mode with supply voltage in 3v-3.6v
1	1	Not allowed

4 GPIO_AD_HIGH_RANGE	GPIO_AD IO bank supply voltage range selection for GPIO_AD_00 to GPIO_AD_17

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>GPIO_AD_HIGH_RANGE</th><th>GPIO_AD_LOW_RANGE</th><th>Mode</th></tr>
<tr><td>0</td><td>0</td><td>GPIO_AD_xx IO will work in continuous range mode with supply voltage in 1.71v-3.6v</td></tr>
<tr><td>0</td><td>1</td><td>GPIO_AD_xx IO will work in low range mode with supply voltage in 1.71v-1.98v</td></tr>
<tr><td>1</td><td>0</td><td>GPIO_AD_xx IO will work in high range mode with supply voltage in 3v-3.6v</td></tr>
<tr><td>1</td><td>1</td><td>Not allowed</td></tr>
</table>
</td></tr>
<tr><td>3-1
—</td><td>Reserved</td></tr>
<tr><td>0
I3C_ON_CHIP_STRONG_PULL_DIS</td><td>Disable I3C on-chip strong pull for I3C2
0b - On-chip strong pull is enabled
1b - On-chip strong pull is disabled</td></tr>
</table>

Chapter 17
IOMUX Controller (IOMUXC)

17.1 Chip-specific IOMUXC Information

Table 122. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

17.2 Overview

The IOMUX Controller (IOMUXC), together with the IOMUX, enables the IC to share one pad to several functional blocks. This sharing is done by multiplexing the pad's input and output signals.

Every module requires a specific pad setting (such as pull up or keeper), and for each pad, there are up to 12 muxing options (called ALT modes). The pad settings parameters are controlled by the IOMUXC.

The IOMUX consists only of combinatorial logic combined from several basic IOMUX cells. Each basic IOMUX cell handles only one pad signal's muxing.

17.2.1 Block diagram

The figure below illustrates the IOMUX/IOMUXC connectivity in the system.

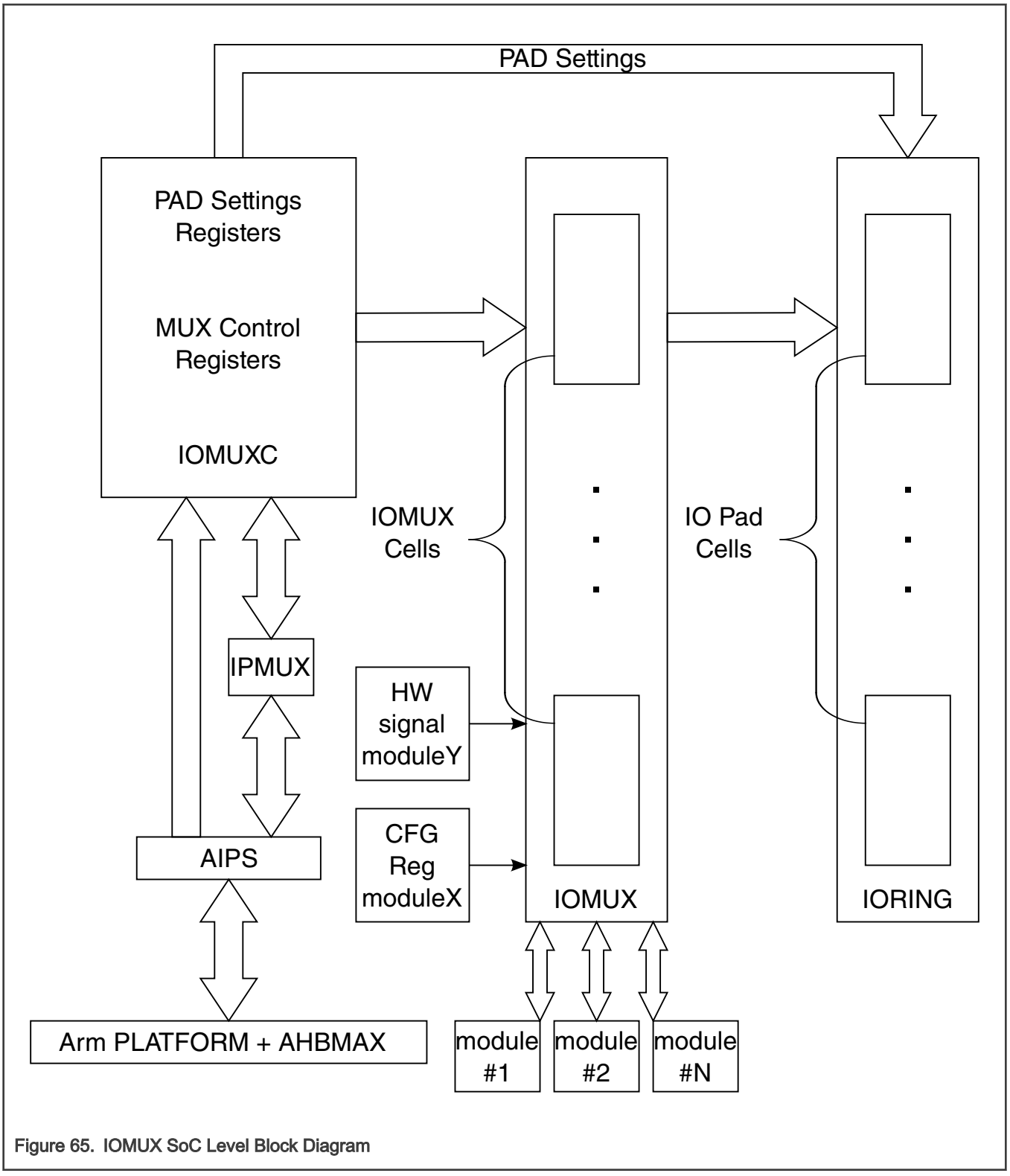


Figure 65. IOMUX SoC Level Block Diagram

17.2.2 Features

The IOMUXC features include:

- 32-bit software mux control registers (SW_MUX_CTL_PAD_<PAD NAME>) to configure 1 of 12 alternate (ALT) MUX_MODE fields of each pad and to enable the forcing of an input path of the pad(s) (SION bit).
- 32-bit software pad control registers (SW_PAD_CTL_PAD_<PAD_NAME>) to configure specific pad settings of each pad.
- 32-bit input select control registers to control the input path to a module when more than one pad drives this module input.

Each SW MUX/PAD CTL IOMUXC register handles only one pad.

Only the minimum number of registers required by software are implemented by hardware. For example, if only ALT0 and ALT1 modes are used on Pad x then only one bit register will be generated as the MUX_MODE control field in the software mux control register of Pad x.

The software mux control registers may allow the forcing of pads to become input (input path enabled) regardless of the functional direction driven. This may be useful for loopback and GPIO data capture.

17.3 Functional description

This section provides a complete functional description of the block.

The IOMUX consists of a number of basic iomux cell units. If only one functional mode is required for a specific pad, it would have been no need for IOMUX and the signals can be connected directly from the module to the I/O. The IOMUX cell is required whenever two or more functional modes are required for a specific pad or when one functional mode and the one test mode are required.

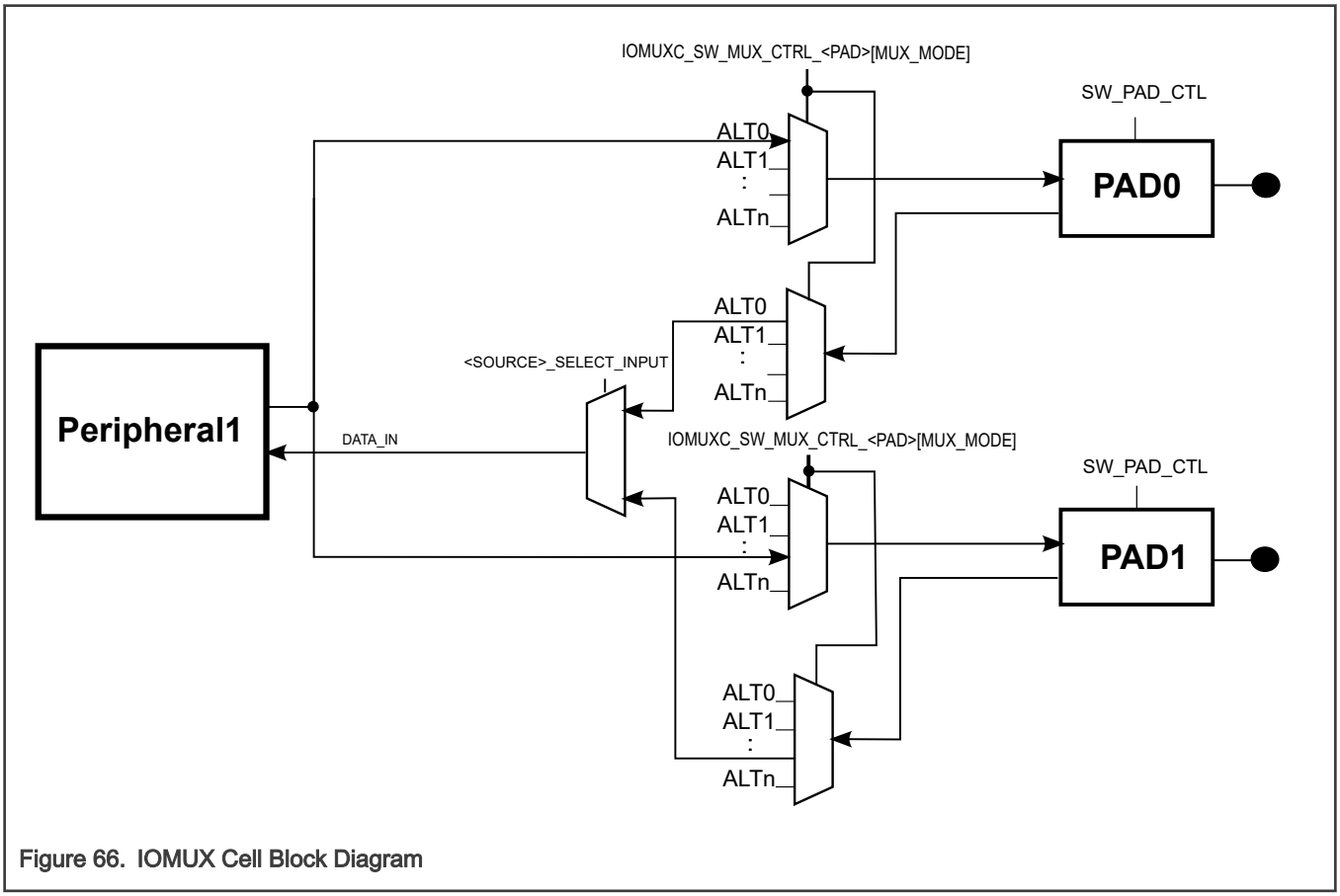


Figure 66. IOMUX Cell Block Diagram

17.3.1 SW Loopback through SION bit

A limited option exists to override the default pad functionality and force the input path to be active regardless of the value driven by the corresponding module. This can be done by setting the SION (Software Input On) bit in the IOMUXC_SW_MUX_CTL register (when available) to "1".

Uses include:

- LoopBack - Module x drives the pad and also receives pad value as an input.

17.3.2 Disable Input Buffer Enable (IBE_OFF)

When pad function is not configured as an output, the Input Buffer Enable (IBE) is asserted and the input buffer enabled. IBE_OFF can force Input Buffer Enable (IBE) to be de-asserted. IBE_OFF can override SION. Once IBE_OFF is 1, SION does not work, SION works only when IBE_OFF is 0.

For ADC/CMP application, IBE_OFF must be set to avoid analog single degrade.

17.3.3 Daisy chain - multi pads driving same module input pin

In some cases, more than one pad may drive a single module input pin. Such cases require the addition of one more level of IOMUXing; all of these input signals are muxed, and a dedicated software controlled register controls the mux in order to select the required input path.

A module port involved in "daisy chain" requires two software configuration commands, one for selecting the mode for this pad (programmable via the SW_MUX_CTL_<PAD> registers) and one for defining it as the input path (via the daisy chain registers).

This means that a module port involved in "daisy chain" requires two software configuration commands, one for selecting the mode for this pad (programmable via the SW_MUX_CTL_<PAD> registers) and one for defining it as the input path (via the daisy chain registers).

The daisy chain is illustrated in the figure below.

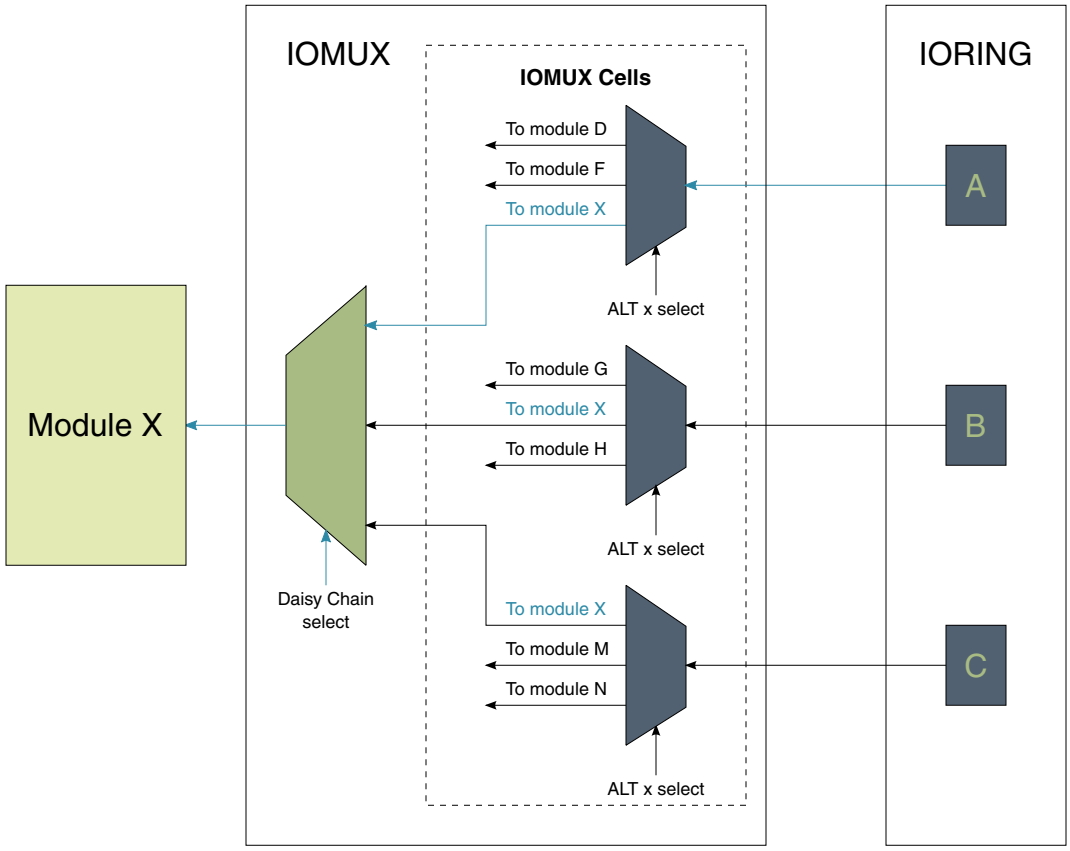


Figure 67. Daisy chain illustration

17.3.4 Clocks

The table found here describes the clock sources for IOMUXC. Please see Clock Controller Module (CCM) chapter for clock setting, configuration and gating information.

Table 123. IOMUXC Clocks

Clock Name	Description
ipg_clk_s	Peripheral access clock

17.4 Memory Map and register definition

This section includes the IOMUXC module memory map and detailed descriptions of all registers.

17.4.1 IOMUXC register descriptions

17.4.1.1 IOMUXC memory map

IOMUXC base address: 42A1_0000h

Offset	Register	Width (In bits)	Access	Reset value
10h	SW_MUX_CTL_PAD_GPIO_EMC_B1_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_00)	32	RW	0000_0005h
14h	SW_MUX_CTL_PAD_GPIO_EMC_B1_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_01)	32	RW	0000_0005h
18h	SW_MUX_CTL_PAD_GPIO_EMC_B1_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_02)	32	RW	0000_0005h
1Ch	SW_MUX_CTL_PAD_GPIO_EMC_B1_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_03)	32	RW	0000_0005h
20h	SW_MUX_CTL_PAD_GPIO_EMC_B1_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_04)	32	RW	0000_0005h
24h	SW_MUX_CTL_PAD_GPIO_EMC_B1_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_05)	32	RW	0000_0005h
28h	SW_MUX_CTL_PAD_GPIO_EMC_B1_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_06)	32	RW	0000_0005h
2Ch	SW_MUX_CTL_PAD_GPIO_EMC_B1_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_07)	32	RW	0000_0005h
30h	SW_MUX_CTL_PAD_GPIO_EMC_B1_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_08)	32	RW	0000_0005h
34h	SW_MUX_CTL_PAD_GPIO_EMC_B1_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_09)	32	RW	0000_0005h
38h	SW_MUX_CTL_PAD_GPIO_EMC_B1_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_10)	32	RW	0000_0005h
3Ch	SW_MUX_CTL_PAD_GPIO_EMC_B1_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_11)	32	RW	0000_0005h
40h	SW_MUX_CTL_PAD_GPIO_EMC_B1_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_12)	32	RW	0000_0005h
44h	SW_MUX_CTL_PAD_GPIO_EMC_B1_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_13)	32	RW	0000_0005h
48h	SW_MUX_CTL_PAD_GPIO_EMC_B1_14 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_14)	32	RW	0000_0005h
4Ch	SW_MUX_CTL_PAD_GPIO_EMC_B1_15 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_15)	32	RW	0000_0005h
50h	SW_MUX_CTL_PAD_GPIO_EMC_B1_16 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_16)	32	RW	0000_0005h
54h	SW_MUX_CTL_PAD_GPIO_EMC_B1_17 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_17)	32	RW	0000_0005h
58h	SW_MUX_CTL_PAD_GPIO_EMC_B1_18 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_18)	32	RW	0000_0005h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5Ch	SW_MUX_CTL_PAD_GPIO_EMC_B1_19 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_19)	32	RW	0000_0005h
60h	SW_MUX_CTL_PAD_GPIO_EMC_B1_20 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_20)	32	RW	0000_0005h
64h	SW_MUX_CTL_PAD_GPIO_EMC_B1_21 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_21)	32	RW	0000_0005h
68h	SW_MUX_CTL_PAD_GPIO_EMC_B1_22 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_22)	32	RW	0000_0005h
6Ch	SW_MUX_CTL_PAD_GPIO_EMC_B1_23 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_23)	32	RW	0000_0005h
70h	SW_MUX_CTL_PAD_GPIO_EMC_B1_24 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_24)	32	RW	0000_0005h
74h	SW_MUX_CTL_PAD_GPIO_EMC_B1_25 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_25)	32	RW	0000_0005h
78h	SW_MUX_CTL_PAD_GPIO_EMC_B1_26 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_26)	32	RW	0000_0005h
7Ch	SW_MUX_CTL_PAD_GPIO_EMC_B1_27 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_27)	32	RW	0000_0005h
80h	SW_MUX_CTL_PAD_GPIO_EMC_B1_28 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_28)	32	RW	0000_0005h
84h	SW_MUX_CTL_PAD_GPIO_EMC_B1_29 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_29)	32	RW	0000_0005h
88h	SW_MUX_CTL_PAD_GPIO_EMC_B1_30 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_30)	32	RW	0000_0005h
8Ch	SW_MUX_CTL_PAD_GPIO_EMC_B1_31 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_31)	32	RW	0000_0005h
90h	SW_MUX_CTL_PAD_GPIO_EMC_B1_32 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_32)	32	RW	0000_0005h
94h	SW_MUX_CTL_PAD_GPIO_EMC_B1_33 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_33)	32	RW	0000_0005h
98h	SW_MUX_CTL_PAD_GPIO_EMC_B1_34 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_34)	32	RW	0000_0005h
9Ch	SW_MUX_CTL_PAD_GPIO_EMC_B1_35 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_35)	32	RW	0000_0005h
A0h	SW_MUX_CTL_PAD_GPIO_EMC_B1_36 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_36)	32	RW	0000_0005h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A4h	SW_MUX_CTL_PAD_GPIO_EMC_B1_37 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_37)	32	RW	0000_0005h
A8h	SW_MUX_CTL_PAD_GPIO_EMC_B1_38 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_38)	32	RW	0000_0005h
ACh	SW_MUX_CTL_PAD_GPIO_EMC_B1_39 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_39)	32	RW	0000_0005h
B0h	SW_MUX_CTL_PAD_GPIO_EMC_B1_40 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_40)	32	RW	0000_0005h
B4h	SW_MUX_CTL_PAD_GPIO_EMC_B1_41 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_41)	32	RW	0000_0005h
B8h	SW_MUX_CTL_PAD_GPIO_EMC_B2_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_00)	32	RW	0000_0005h
BCh	SW_MUX_CTL_PAD_GPIO_EMC_B2_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_01)	32	RW	0000_0005h
C0h	SW_MUX_CTL_PAD_GPIO_EMC_B2_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_02)	32	RW	0000_0005h
C4h	SW_MUX_CTL_PAD_GPIO_EMC_B2_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_03)	32	RW	0000_0005h
C8h	SW_MUX_CTL_PAD_GPIO_EMC_B2_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_04)	32	RW	0000_0005h
CCh	SW_MUX_CTL_PAD_GPIO_EMC_B2_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_05)	32	RW	0000_0005h
D0h	SW_MUX_CTL_PAD_GPIO_EMC_B2_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_06)	32	RW	0000_0005h
D4h	SW_MUX_CTL_PAD_GPIO_EMC_B2_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_07)	32	RW	0000_0005h
D8h	SW_MUX_CTL_PAD_GPIO_EMC_B2_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_08)	32	RW	0000_0005h
DCh	SW_MUX_CTL_PAD_GPIO_EMC_B2_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_09)	32	RW	0000_0005h
E0h	SW_MUX_CTL_PAD_GPIO_EMC_B2_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_10)	32	RW	0000_0005h
E4h	SW_MUX_CTL_PAD_GPIO_EMC_B2_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_11)	32	RW	0000_0005h
E8h	SW_MUX_CTL_PAD_GPIO_EMC_B2_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_12)	32	RW	0000_0005h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
ECh	SW_MUX_CTL_PAD_GPIO_EMC_B2_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_13)	32	RW	0000_0005h
F0h	SW_MUX_CTL_PAD_GPIO_EMC_B2_14 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_14)	32	RW	0000_0005h
F4h	SW_MUX_CTL_PAD_GPIO_EMC_B2_15 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_15)	32	RW	0000_0005h
F8h	SW_MUX_CTL_PAD_GPIO_EMC_B2_16 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_16)	32	RW	0000_0005h
FCh	SW_MUX_CTL_PAD_GPIO_EMC_B2_17 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_17)	32	RW	0000_0005h
100h	SW_MUX_CTL_PAD_GPIO_EMC_B2_18 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_18)	32	RW	0000_0005h
104h	SW_MUX_CTL_PAD_GPIO_EMC_B2_19 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_19)	32	RW	0000_0005h
108h	SW_MUX_CTL_PAD_GPIO_EMC_B2_20 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_20)	32	RW	0000_0005h
10Ch	SW_MUX_CTL_PAD_GPIO_AD_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_00)	32	RW	0000_0005h
110h	SW_MUX_CTL_PAD_GPIO_AD_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_01)	32	RW	0000_0005h
114h	SW_MUX_CTL_PAD_GPIO_AD_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_02)	32	RW	0000_0005h
118h	SW_MUX_CTL_PAD_GPIO_AD_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_03)	32	RW	0000_0005h
11Ch	SW_MUX_CTL_PAD_GPIO_AD_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_04)	32	RW	0000_0005h
120h	SW_MUX_CTL_PAD_GPIO_AD_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_05)	32	RW	0000_0005h
124h	SW_MUX_CTL_PAD_GPIO_AD_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_06)	32	RW	0000_0005h
128h	SW_MUX_CTL_PAD_GPIO_AD_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_07)	32	RW	0000_0005h
12Ch	SW_MUX_CTL_PAD_GPIO_AD_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_08)	32	RW	0000_0005h
130h	SW_MUX_CTL_PAD_GPIO_AD_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_09)	32	RW	0000_0005h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
134h	SW_MUX_CTL_PAD_GPIO_AD_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_10)	32	RW	0000_0005h
138h	SW_MUX_CTL_PAD_GPIO_AD_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_11)	32	RW	0000_0005h
13Ch	SW_MUX_CTL_PAD_GPIO_AD_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_12)	32	RW	0000_0005h
140h	SW_MUX_CTL_PAD_GPIO_AD_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_13)	32	RW	0000_0005h
144h	SW_MUX_CTL_PAD_GPIO_AD_14 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_14)	32	RW	0000_0005h
148h	SW_MUX_CTL_PAD_GPIO_AD_15 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_15)	32	RW	0000_0005h
14Ch	SW_MUX_CTL_PAD_GPIO_AD_16 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_16)	32	RW	0000_0005h
150h	SW_MUX_CTL_PAD_GPIO_AD_17 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_17)	32	RW	0000_0005h
154h	SW_MUX_CTL_PAD_GPIO_AD_18 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_18)	32	RW	0000_0005h
158h	SW_MUX_CTL_PAD_GPIO_AD_19 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_19)	32	RW	0000_0005h
15Ch	SW_MUX_CTL_PAD_GPIO_AD_20 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_20)	32	RW	0000_0005h
160h	SW_MUX_CTL_PAD_GPIO_AD_21 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_21)	32	RW	0000_0005h
164h	SW_MUX_CTL_PAD_GPIO_AD_22 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_22)	32	RW	0000_0005h
168h	SW_MUX_CTL_PAD_GPIO_AD_23 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_23)	32	RW	0000_0005h
16Ch	SW_MUX_CTL_PAD_GPIO_AD_24 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_24)	32	RW	0000_0005h
170h	SW_MUX_CTL_PAD_GPIO_AD_25 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_25)	32	RW	0000_0005h
174h	SW_MUX_CTL_PAD_GPIO_AD_26 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_26)	32	RW	0000_0005h
178h	SW_MUX_CTL_PAD_GPIO_AD_27 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_27)	32	RW	0000_0005h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
17Ch	SW_MUX_CTL_PAD_GPIO_AD_28 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_28)	32	RW	0000_0005h
180h	SW_MUX_CTL_PAD_GPIO_AD_29 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_29)	32	RW	0000_0005h
184h	SW_MUX_CTL_PAD_GPIO_AD_30 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_30)	32	RW	0000_0005h
188h	SW_MUX_CTL_PAD_GPIO_AD_31 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_31)	32	RW	0000_0005h
18Ch	SW_MUX_CTL_PAD_GPIO_AD_32 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_32)	32	RW	0000_0005h
190h	SW_MUX_CTL_PAD_GPIO_AD_33 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_33)	32	RW	0000_0005h
194h	SW_MUX_CTL_PAD_GPIO_AD_34 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_34)	32	RW	0000_0005h
198h	SW_MUX_CTL_PAD_GPIO_AD_35 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_35)	32	RW	0000_0005h
19Ch	SW_MUX_CTL_PAD_GPIO_SD_B1_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_00)	32	RW	0000_0005h
1A0h	SW_MUX_CTL_PAD_GPIO_SD_B1_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_01)	32	RW	0000_0005h
1A4h	SW_MUX_CTL_PAD_GPIO_SD_B1_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_02)	32	RW	0000_0005h
1A8h	SW_MUX_CTL_PAD_GPIO_SD_B1_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_03)	32	RW	0000_0005h
1ACh	SW_MUX_CTL_PAD_GPIO_SD_B1_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_04)	32	RW	0000_0005h
1B0h	SW_MUX_CTL_PAD_GPIO_SD_B1_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_05)	32	RW	0000_0005h
1B4h	SW_MUX_CTL_PAD_GPIO_SD_B2_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_00)	32	RW	0000_0005h
1B8h	SW_MUX_CTL_PAD_GPIO_SD_B2_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_01)	32	RW	0000_0005h
1BCh	SW_MUX_CTL_PAD_GPIO_SD_B2_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_02)	32	RW	0000_0005h
1C0h	SW_MUX_CTL_PAD_GPIO_SD_B2_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_03)	32	RW	0000_0005h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1C4h	SW_MUX_CTL_PAD_GPIO_SD_B2_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_04)	32	RW	0000_0005h
1C8h	SW_MUX_CTL_PAD_GPIO_SD_B2_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_05)	32	RW	0000_0005h
1CCh	SW_MUX_CTL_PAD_GPIO_SD_B2_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_06)	32	RW	0000_0005h
1D0h	SW_MUX_CTL_PAD_GPIO_SD_B2_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_07)	32	RW	0000_0005h
1D4h	SW_MUX_CTL_PAD_GPIO_SD_B2_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_08)	32	RW	0000_0005h
1D8h	SW_MUX_CTL_PAD_GPIO_SD_B2_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_09)	32	RW	0000_0005h
1DCh	SW_MUX_CTL_PAD_GPIO_SD_B2_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_10)	32	RW	0000_0005h
1E0h	SW_MUX_CTL_PAD_GPIO_SD_B2_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_11)	32	RW	0000_0005h
1E4h	SW_MUX_CTL_PAD_GPIO_SD_B2_12_DUMMY SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_12_DUMMY)	32	RW	0000_0000h
1E8h	SW_MUX_CTL_PAD_GPIO_B1_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_00)	32	RW	0000_0005h
1ECh	SW_MUX_CTL_PAD_GPIO_B1_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_01)	32	RW	0000_0005h
1F0h	SW_MUX_CTL_PAD_GPIO_B1_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_02)	32	RW	0000_0005h
1F4h	SW_MUX_CTL_PAD_GPIO_B1_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_03)	32	RW	0000_0005h
1F8h	SW_MUX_CTL_PAD_GPIO_B1_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_04)	32	RW	0000_0005h
1FCh	SW_MUX_CTL_PAD_GPIO_B1_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_05)	32	RW	0000_0005h
200h	SW_MUX_CTL_PAD_GPIO_B1_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_06)	32	RW	0000_0005h
204h	SW_MUX_CTL_PAD_GPIO_B1_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_07)	32	RW	0000_0005h
208h	SW_MUX_CTL_PAD_GPIO_B1_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_08)	32	RW	0000_0005h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20Ch	SW_MUX_CTL_PAD_GPIO_B1_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_09)	32	RW	0000_0005h
210h	SW_MUX_CTL_PAD_GPIO_B1_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_10)	32	RW	0000_0005h
214h	SW_MUX_CTL_PAD_GPIO_B1_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_11)	32	RW	0000_0005h
218h	SW_MUX_CTL_PAD_GPIO_B1_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_12)	32	RW	0000_0005h
21Ch	SW_MUX_CTL_PAD_GPIO_B1_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_13)	32	RW	0000_0005h
220h	SW_MUX_CTL_PAD_GPIO_B2_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_00)	32	RW	0000_0005h
224h	SW_MUX_CTL_PAD_GPIO_B2_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_01)	32	RW	0000_0005h
228h	SW_MUX_CTL_PAD_GPIO_B2_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_02)	32	RW	0000_0005h
22Ch	SW_MUX_CTL_PAD_GPIO_B2_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_03)	32	RW	0000_0005h
230h	SW_MUX_CTL_PAD_GPIO_B2_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_04)	32	RW	0000_0005h
234h	SW_MUX_CTL_PAD_GPIO_B2_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_05)	32	RW	0000_0005h
238h	SW_MUX_CTL_PAD_GPIO_B2_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_06)	32	RW	0000_0005h
23Ch	SW_MUX_CTL_PAD_GPIO_B2_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_07)	32	RW	0000_0005h
240h	SW_MUX_CTL_PAD_GPIO_B2_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_08)	32	RW	0000_0005h
244h	SW_MUX_CTL_PAD_GPIO_B2_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_09)	32	RW	0000_0005h
248h	SW_MUX_CTL_PAD_GPIO_B2_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_10)	32	RW	0000_0005h
24Ch	SW_MUX_CTL_PAD_GPIO_B2_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_11)	32	RW	0000_0005h
250h	SW_MUX_CTL_PAD_GPIO_B2_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_12)	32	RW	0000_0005h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
254h	SW_MUX_CTL_PAD_GPIO_B2_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_13)	32	RW	0000_0005h
258h	SW_PAD_CTL_PAD_GPIO_EMC_B1_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_00)	32	RW	0000_0008h
25Ch	SW_PAD_CTL_PAD_GPIO_EMC_B1_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_01)	32	RW	0000_0008h
260h	SW_PAD_CTL_PAD_GPIO_EMC_B1_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_02)	32	RW	0000_0008h
264h	SW_PAD_CTL_PAD_GPIO_EMC_B1_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_03)	32	RW	0000_0008h
268h	SW_PAD_CTL_PAD_GPIO_EMC_B1_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_04)	32	RW	0000_0008h
26Ch	SW_PAD_CTL_PAD_GPIO_EMC_B1_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_05)	32	RW	0000_0008h
270h	SW_PAD_CTL_PAD_GPIO_EMC_B1_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_06)	32	RW	0000_0008h
274h	SW_PAD_CTL_PAD_GPIO_EMC_B1_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_07)	32	RW	0000_0008h
278h	SW_PAD_CTL_PAD_GPIO_EMC_B1_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_08)	32	RW	0000_0008h
27Ch	SW_PAD_CTL_PAD_GPIO_EMC_B1_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_09)	32	RW	0000_0008h
280h	SW_PAD_CTL_PAD_GPIO_EMC_B1_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_10)	32	RW	0000_0008h
284h	SW_PAD_CTL_PAD_GPIO_EMC_B1_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_11)	32	RW	0000_0008h
288h	SW_PAD_CTL_PAD_GPIO_EMC_B1_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_12)	32	RW	0000_0008h
28Ch	SW_PAD_CTL_PAD_GPIO_EMC_B1_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_13)	32	RW	0000_0008h
290h	SW_PAD_CTL_PAD_GPIO_EMC_B1_14 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_14)	32	RW	0000_0008h
294h	SW_PAD_CTL_PAD_GPIO_EMC_B1_15 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_15)	32	RW	0000_0008h
298h	SW_PAD_CTL_PAD_GPIO_EMC_B1_16 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_16)	32	RW	0000_0008h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
29Ch	SW_PAD_CTL_PAD_GPIO_EMC_B1_17 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_17)	32	RW	0000_0008h
2A0h	SW_PAD_CTL_PAD_GPIO_EMC_B1_18 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_18)	32	RW	0000_0008h
2A4h	SW_PAD_CTL_PAD_GPIO_EMC_B1_19 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_19)	32	RW	0000_0008h
2A8h	SW_PAD_CTL_PAD_GPIO_EMC_B1_20 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_20)	32	RW	0000_0008h
2ACh	SW_PAD_CTL_PAD_GPIO_EMC_B1_21 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_21)	32	RW	0000_0008h
2B0h	SW_PAD_CTL_PAD_GPIO_EMC_B1_22 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_22)	32	RW	0000_0008h
2B4h	SW_PAD_CTL_PAD_GPIO_EMC_B1_23 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_23)	32	RW	0000_0008h
2B8h	SW_PAD_CTL_PAD_GPIO_EMC_B1_24 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_24)	32	RW	0000_0008h
2BCh	SW_PAD_CTL_PAD_GPIO_EMC_B1_25 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_25)	32	RW	0000_0008h
2C0h	SW_PAD_CTL_PAD_GPIO_EMC_B1_26 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_26)	32	RW	0000_0004h
2C4h	SW_PAD_CTL_PAD_GPIO_EMC_B1_27 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_27)	32	RW	0000_0008h
2C8h	SW_PAD_CTL_PAD_GPIO_EMC_B1_28 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_28)	32	RW	0000_0004h
2CCh	SW_PAD_CTL_PAD_GPIO_EMC_B1_29 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_29)	32	RW	0000_0008h
2D0h	SW_PAD_CTL_PAD_GPIO_EMC_B1_30 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_30)	32	RW	0000_0008h
2D4h	SW_PAD_CTL_PAD_GPIO_EMC_B1_31 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_31)	32	RW	0000_0008h
2D8h	SW_PAD_CTL_PAD_GPIO_EMC_B1_32 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_32)	32	RW	0000_0008h
2DCh	SW_PAD_CTL_PAD_GPIO_EMC_B1_33 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_33)	32	RW	0000_0008h
2E0h	SW_PAD_CTL_PAD_GPIO_EMC_B1_34 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_34)	32	RW	0000_0008h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2E4h	SW_PAD_CTL_PAD_GPIO_EMC_B1_35 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_35)	32	RW	0000_0008h
2E8h	SW_PAD_CTL_PAD_GPIO_EMC_B1_36 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_36)	32	RW	0000_0008h
2ECh	SW_PAD_CTL_PAD_GPIO_EMC_B1_37 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_37)	32	RW	0000_0008h
2F0h	SW_PAD_CTL_PAD_GPIO_EMC_B1_38 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_38)	32	RW	0000_0008h
2F4h	SW_PAD_CTL_PAD_GPIO_EMC_B1_39 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_39)	32	RW	0000_0004h
2F8h	SW_PAD_CTL_PAD_GPIO_EMC_B1_40 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_40)	32	RW	0000_0008h
2FCh	SW_PAD_CTL_PAD_GPIO_EMC_B1_41 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_41)	32	RW	0000_0008h
300h	SW_PAD_CTL_PAD_GPIO_EMC_B2_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_00)	32	RW	0000_0008h
304h	SW_PAD_CTL_PAD_GPIO_EMC_B2_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_01)	32	RW	0000_0008h
308h	SW_PAD_CTL_PAD_GPIO_EMC_B2_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_02)	32	RW	0000_0008h
30Ch	SW_PAD_CTL_PAD_GPIO_EMC_B2_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_03)	32	RW	0000_0008h
310h	SW_PAD_CTL_PAD_GPIO_EMC_B2_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_04)	32	RW	0000_0008h
314h	SW_PAD_CTL_PAD_GPIO_EMC_B2_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_05)	32	RW	0000_0008h
318h	SW_PAD_CTL_PAD_GPIO_EMC_B2_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_06)	32	RW	0000_0008h
31Ch	SW_PAD_CTL_PAD_GPIO_EMC_B2_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_07)	32	RW	0000_0008h
320h	SW_PAD_CTL_PAD_GPIO_EMC_B2_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_08)	32	RW	0000_0008h
324h	SW_PAD_CTL_PAD_GPIO_EMC_B2_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_09)	32	RW	0000_0008h
328h	SW_PAD_CTL_PAD_GPIO_EMC_B2_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_10)	32	RW	0000_0008h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
32Ch	SW_PAD_CTL_PAD_GPIO_EMC_B2_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_11)	32	RW	0000_0008h
330h	SW_PAD_CTL_PAD_GPIO_EMC_B2_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_12)	32	RW	0000_0008h
334h	SW_PAD_CTL_PAD_GPIO_EMC_B2_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_13)	32	RW	0000_0008h
338h	SW_PAD_CTL_PAD_GPIO_EMC_B2_14 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_14)	32	RW	0000_0008h
33Ch	SW_PAD_CTL_PAD_GPIO_EMC_B2_15 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_15)	32	RW	0000_0008h
340h	SW_PAD_CTL_PAD_GPIO_EMC_B2_16 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_16)	32	RW	0000_0008h
344h	SW_PAD_CTL_PAD_GPIO_EMC_B2_17 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_17)	32	RW	0000_0008h
348h	SW_PAD_CTL_PAD_GPIO_EMC_B2_18 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_18)	32	RW	0000_0004h
34Ch	SW_PAD_CTL_PAD_GPIO_EMC_B2_19 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_19)	32	RW	0000_0008h
350h	SW_PAD_CTL_PAD_GPIO_EMC_B2_20 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_20)	32	RW	0000_0008h
354h	SW_PAD_CTL_PAD_GPIO_AD_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_00)	32	RW	0000_0006h
358h	SW_PAD_CTL_PAD_GPIO_AD_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_01)	32	RW	0000_0006h
35Ch	SW_PAD_CTL_PAD_GPIO_AD_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_02)	32	RW	0000_0006h
360h	SW_PAD_CTL_PAD_GPIO_AD_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_03)	32	RW	0000_0006h
364h	SW_PAD_CTL_PAD_GPIO_AD_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_04)	32	RW	0000_0006h
368h	SW_PAD_CTL_PAD_GPIO_AD_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_05)	32	RW	0000_0006h
36Ch	SW_PAD_CTL_PAD_GPIO_AD_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_06)	32	RW	0000_0006h
370h	SW_PAD_CTL_PAD_GPIO_AD_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_07)	32	RW	0000_0006h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
374h	SW_PAD_CTL_PAD_GPIO_AD_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_08)	32	RW	0000_0006h
378h	SW_PAD_CTL_PAD_GPIO_AD_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_09)	32	RW	0000_0006h
37Ch	SW_PAD_CTL_PAD_GPIO_AD_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_10)	32	RW	0000_0006h
380h	SW_PAD_CTL_PAD_GPIO_AD_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_11)	32	RW	0000_0006h
384h	SW_PAD_CTL_PAD_GPIO_AD_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_12)	32	RW	0000_000Eh
388h	SW_PAD_CTL_PAD_GPIO_AD_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_13)	32	RW	0000_0006h
38Ch	SW_PAD_CTL_PAD_GPIO_AD_14 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_14)	32	RW	0000_0006h
390h	SW_PAD_CTL_PAD_GPIO_AD_15 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_15)	32	RW	0000_0006h
394h	SW_PAD_CTL_PAD_GPIO_AD_16 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_16)	32	RW	0000_0006h
398h	SW_PAD_CTL_PAD_GPIO_AD_17 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_17)	32	RW	0000_0006h
39Ch	SW_PAD_CTL_PAD_GPIO_AD_18 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_18)	32	RW	0000_0006h
3A0h	SW_PAD_CTL_PAD_GPIO_AD_19 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_19)	32	RW	0000_0006h
3A4h	SW_PAD_CTL_PAD_GPIO_AD_20 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_20)	32	RW	0000_0006h
3A8h	SW_PAD_CTL_PAD_GPIO_AD_21 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_21)	32	RW	0000_0006h
3ACh	SW_PAD_CTL_PAD_GPIO_AD_22 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_22)	32	RW	0000_0006h
3B0h	SW_PAD_CTL_PAD_GPIO_AD_23 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_23)	32	RW	0000_0006h
3B4h	SW_PAD_CTL_PAD_GPIO_AD_24 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_24)	32	RW	0000_0006h
3B8h	SW_PAD_CTL_PAD_GPIO_AD_25 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_25)	32	RW	0000_0006h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3BCh	SW_PAD_CTL_PAD_GPIO_AD_26 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_26)	32	RW	0000_0006h
3C0h	SW_PAD_CTL_PAD_GPIO_AD_27 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_27)	32	RW	0000_0006h
3C4h	SW_PAD_CTL_PAD_GPIO_AD_28 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_28)	32	RW	0000_0006h
3C8h	SW_PAD_CTL_PAD_GPIO_AD_29 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_29)	32	RW	0000_000Eh
3CCh	SW_PAD_CTL_PAD_GPIO_AD_30 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_30)	32	RW	0000_0006h
3D0h	SW_PAD_CTL_PAD_GPIO_AD_31 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_31)	32	RW	0000_0006h
3D4h	SW_PAD_CTL_PAD_GPIO_AD_32 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_32)	32	RW	0000_0006h
3D8h	SW_PAD_CTL_PAD_GPIO_AD_33 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_33)	32	RW	0000_0006h
3DCh	SW_PAD_CTL_PAD_GPIO_AD_34 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_34)	32	RW	0000_0006h
3E0h	SW_PAD_CTL_PAD_GPIO_AD_35 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_35)	32	RW	0000_0006h
3E4h	SW_PAD_CTL_PAD_GPIO_SD_B1_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_00)	32	RW	0000_0008h
3E8h	SW_PAD_CTL_PAD_GPIO_SD_B1_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_01)	32	RW	0000_0008h
3ECh	SW_PAD_CTL_PAD_GPIO_SD_B1_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_02)	32	RW	0000_0004h
3F0h	SW_PAD_CTL_PAD_GPIO_SD_B1_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_03)	32	RW	0000_0004h
3F4h	SW_PAD_CTL_PAD_GPIO_SD_B1_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_04)	32	RW	0000_0004h
3F8h	SW_PAD_CTL_PAD_GPIO_SD_B1_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_05)	32	RW	0000_0004h
3FCh	SW_PAD_CTL_PAD_GPIO_SD_B2_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_00)	32	RW	0000_0008h
400h	SW_PAD_CTL_PAD_GPIO_SD_B2_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_01)	32	RW	0000_0008h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
404h	SW_PAD_CTL_PAD_GPIO_SD_B2_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_02)	32	RW	0000_0008h
408h	SW_PAD_CTL_PAD_GPIO_SD_B2_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_03)	32	RW	0000_0008h
40Ch	SW_PAD_CTL_PAD_GPIO_SD_B2_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_04)	32	RW	0000_0004h
410h	SW_PAD_CTL_PAD_GPIO_SD_B2_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_05)	32	RW	0000_0008h
414h	SW_PAD_CTL_PAD_GPIO_SD_B2_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_06)	32	RW	0000_0004h
418h	SW_PAD_CTL_PAD_GPIO_SD_B2_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_07)	32	RW	0000_0008h
41Ch	SW_PAD_CTL_PAD_GPIO_SD_B2_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_08)	32	RW	0000_0008h
420h	SW_PAD_CTL_PAD_GPIO_SD_B2_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_09)	32	RW	0000_0008h
424h	SW_PAD_CTL_PAD_GPIO_SD_B2_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_10)	32	RW	0000_0008h
428h	SW_PAD_CTL_PAD_GPIO_SD_B2_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_11)	32	RW	0000_0008h
42Ch	SW_PAD_CTL_PAD_GPIO_SD_B2_12_DUMMY SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_12_DUMMY)	32	RW	0000_0008h
430h	SW_PAD_CTL_PAD_GPIO_B1_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_00)	32	RW	0000_0004h
434h	SW_PAD_CTL_PAD_GPIO_B1_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_01)	32	RW	0000_0004h
438h	SW_PAD_CTL_PAD_GPIO_B1_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_02)	32	RW	0000_0004h
43Ch	SW_PAD_CTL_PAD_GPIO_B1_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_03)	32	RW	0000_0008h
440h	SW_PAD_CTL_PAD_GPIO_B1_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_04)	32	RW	0000_0004h
444h	SW_PAD_CTL_PAD_GPIO_B1_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_05)	32	RW	0000_0008h
448h	SW_PAD_CTL_PAD_GPIO_B1_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_06)	32	RW	0000_0008h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
44Ch	SW_PAD_CTL_PAD_GPIO_B1_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_07)	32	RW	0000_0008h
450h	SW_PAD_CTL_PAD_GPIO_B1_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_08)	32	RW	0000_0008h
454h	SW_PAD_CTL_PAD_GPIO_B1_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_09)	32	RW	0000_0008h
458h	SW_PAD_CTL_PAD_GPIO_B1_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_10)	32	RW	0000_0008h
45Ch	SW_PAD_CTL_PAD_GPIO_B1_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_11)	32	RW	0000_0008h
460h	SW_PAD_CTL_PAD_GPIO_B1_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_12)	32	RW	0000_0008h
464h	SW_PAD_CTL_PAD_GPIO_B1_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_13)	32	RW	0000_0008h
468h	SW_PAD_CTL_PAD_GPIO_B2_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_00)	32	RW	0000_0004h
46Ch	SW_PAD_CTL_PAD_GPIO_B2_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_01)	32	RW	0000_0004h
470h	SW_PAD_CTL_PAD_GPIO_B2_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_02)	32	RW	0000_0004h
474h	SW_PAD_CTL_PAD_GPIO_B2_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_03)	32	RW	0000_0008h
478h	SW_PAD_CTL_PAD_GPIO_B2_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_04)	32	RW	0000_0008h
47Ch	SW_PAD_CTL_PAD_GPIO_B2_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_05)	32	RW	0000_0008h
480h	SW_PAD_CTL_PAD_GPIO_B2_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_06)	32	RW	0000_0008h
484h	SW_PAD_CTL_PAD_GPIO_B2_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_07)	32	RW	0000_0008h
488h	SW_PAD_CTL_PAD_GPIO_B2_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_08)	32	RW	0000_0008h
48Ch	SW_PAD_CTL_PAD_GPIO_B2_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_09)	32	RW	0000_0004h
490h	SW_PAD_CTL_PAD_GPIO_B2_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_10)	32	RW	0000_0008h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
494h	SW_PAD_CTL_PAD_GPIO_B2_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_11)	32	RW	0000_0008h
498h	SW_PAD_CTL_PAD_GPIO_B2_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_12)	32	RW	0000_0008h
49Ch	SW_PAD_CTL_PAD_GPIO_B2_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_13)	32	RW	0000_0008h
4A0h	CAN1_IPP_IND_CANRX_SELECT_INPUT DAISY Register (CAN1_IPP_IND_CANRX_SELECT_INPUT)	32	RW	0000_0000h
4A4h	CAN2_IPP_IND_CANRX_SELECT_INPUT DAISY Register (CAN2_IPP_IND_CANRX_SELECT_INPUT)	32	RW	0000_0000h
4A8h	CAN3_IPP_IND_CANRX_SELECT_INPUT DAISY Register (CAN3_IPP_IND_CANRX_SELECT_INPUT)	32	RW	0000_0000h
4ACh	ECAT_ECAT_RX_CLK_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_CLK_0_SELECT_INPUT)	32	RW	0000_0000h
4B0h	ECAT_ECAT_RX_CLK_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_CLK_1_SELECT_INPUT)	32	RW	0000_0000h
4B4h	ECAT_ECAT_RX_DATA0_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA0_0_SELECT_INPUT)	32	RW	0000_0000h
4B8h	ECAT_ECAT_RX_DATA0_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA0_1_SELECT_INPUT)	32	RW	0000_0000h
4BCh	ECAT_ECAT_RX_DATA1_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA1_0_SELECT_INPUT)	32	RW	0000_0000h
4C0h	ECAT_ECAT_RX_DATA1_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA1_1_SELECT_INPUT)	32	RW	0000_0000h
4C4h	ECAT_ECAT_RX_DATA2_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA2_0_SELECT_INPUT)	32	RW	0000_0000h
4C8h	ECAT_ECAT_RX_DATA2_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA2_1_SELECT_INPUT)	32	RW	0000_0000h
4CCh	ECAT_ECAT_RX_DATA3_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA3_0_SELECT_INPUT)	32	RW	0000_0000h
4D0h	ECAT_ECAT_RX_DATA3_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA3_1_SELECT_INPUT)	32	RW	0000_0000h
4D4h	ECAT_ECAT_RX_DV_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DV_0_SELECT_INPUT)	32	RW	0000_0000h
4D8h	ECAT_ECAT_RX_DV_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DV_1_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4DCh	ECAT_ECAT_RX_ER_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_ER_0_SELECT_INPUT)	32	RW	0000_0000h
4E0h	ECAT_ECAT_RX_ER_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_ER_1_SELECT_INPUT)	32	RW	0000_0000h
4E4h	ECAT_ECAT_TX_CLK_0_SELECT_INPUT DAISY Register (ECAT_ECAT_TX_CLK_0_SELECT_INPUT)	32	RW	0000_0000h
4E8h	ECAT_ECAT_TX_CLK_1_SELECT_INPUT DAISY Register (ECAT_ECAT_TX_CLK_1_SELECT_INPUT)	32	RW	0000_0000h
4ECh	ECAT_MDIO_DATA_IN_SELECT_INPUT DAISY Register (ECAT_MDIO_DATA_IN_SELECT_INPUT)	32	RW	0000_0000h
4F0h	ECAT_PROM_DATA_IN_SELECT_INPUT DAISY Register (ECAT_PROM_DATA_IN_SELECT_INPUT)	32	RW	0000_0000h
4F4h	FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_0 DAISY Register (FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_0)	32	RW	0000_0000h
4F8h	FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_1 DAISY Register (FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_1)	32	RW	0000_0000h
4FCh	FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_2 DAISY Register (FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_2)	32	RW	0000_0000h
500h	FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_0 DAISY Register (FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_0)	32	RW	0000_0000h
504h	FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_1 DAISY Register (FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_1)	32	RW	0000_0000h
508h	FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_2 DAISY Register (FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_2)	32	RW	0000_0000h
50Ch	FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_0 DAISY Register (FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_0)	32	RW	0000_0000h
510h	FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_1 DAISY Register (FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_1)	32	RW	0000_0000h
514h	FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_2 DAISY Register (FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_2)	32	RW	0000_0000h
518h	FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_0 DAISY Register (FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_0)	32	RW	0000_0000h
51Ch	FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_1 DAISY Register (FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_1)	32	RW	0000_0000h
520h	FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_2 DAISY Register (FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
524h	FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_0 DAISY Register (FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_0)	32	RW	0000_0000h
528h	FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_1 DAISY Register (FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_1)	32	RW	0000_0000h
52Ch	FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_2 DAISY Register (FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_2)	32	RW	0000_0000h
530h	FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_3 DAISY Register (FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_3)	32	RW	0000_0000h
534h	FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_0 DAISY Register (FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_0)	32	RW	0000_0000h
538h	FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_1 DAISY Register (FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_1)	32	RW	0000_0000h
53Ch	FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_2 DAISY Register (FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_2)	32	RW	0000_0000h
540h	FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_3 DAISY Register (FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_3)	32	RW	0000_0000h
544h	FLEXSPI1_BUS2BIT_IPP_IND_DQS_FA_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_DQS_FA_SELECT_INPUT)	32	RW	0000_0000h
548h	FLEXSPI1_BUS2BIT_IPP_IND_DQS_FB_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_DQS_FB_SELECT_INPUT)	32	RW	0000_0000h
54Ch	FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT0_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT0_SELECT_INPUT)	32	RW	0000_0000h
550h	FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT1_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT1_SELECT_INPUT)	32	RW	0000_0000h
554h	FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT2_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT2_SELECT_INPUT)	32	RW	0000_0000h
558h	FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT3_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT3_SELECT_INPUT)	32	RW	0000_0000h
55Ch	FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT4_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT4_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
560h	FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT5_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT5_SELECT_INPUT)	32	RW	0000_0000h
564h	FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT6_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT6_SELECT_INPUT)	32	RW	0000_0000h
568h	FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT7_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT7_SELECT_INPUT)	32	RW	0000_0000h
56Ch	FLEXSPI1_BUS2BIT_IPP_IND_SCK_FB_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_SCK_FB_SELECT_INPUT)	32	RW	0000_0000h
570h	FLEXSPI2_BUS2BIT_IPP_IND_DQS_FA_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_DQS_FA_SELECT_INPUT)	32	RW	0000_0000h
574h	FLEXSPI2_BUS2BIT_IPP_IND_DQS_FB_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_DQS_FB_SELECT_INPUT)	32	RW	0000_0000h
578h	FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT0_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT0_SELECT_INPUT)	32	RW	0000_0000h
57Ch	FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT1_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT1_SELECT_INPUT)	32	RW	0000_0000h
580h	FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT2_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT2_SELECT_INPUT)	32	RW	0000_0000h
584h	FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT3_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT3_SELECT_INPUT)	32	RW	0000_0000h
588h	FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT0_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT0_SELECT_INPUT)	32	RW	0000_0000h
58Ch	FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT1_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT1_SELECT_INPUT)	32	RW	0000_0000h
590h	FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT2_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT2_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
594h	FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT3_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT3_SELECT_INPUT)	32	RW	0000_0000h
598h	FLEXSPI2_BUS2BIT_IPP_IND_SCK_FA_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_SCK_FA_SELECT_INPUT)	32	RW	0000_0000h
59Ch	FLEXSPI2_BUS2BIT_IPP_IND_SCK_FB_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_SCK_FB_SELECT_INPUT)	32	RW	0000_0000h
5A0h	I3C2_PIN_SCL_IN_SELECT_INPUT DAISY Register (I3C2_PIN_SCL_IN_SELECT_INPUT)	32	RW	0000_0000h
5A4h	I3C2_PIN_SDA_IN_SELECT_INPUT DAISY Register (I3C2_PIN_SDA_IN_SELECT_INPUT)	32	RW	0000_0000h
5A8h	KPP_IPP_IND_COL_SELECT_INPUT_0 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_0)	32	RW	0000_0000h
5ACh	KPP_IPP_IND_COL_SELECT_INPUT_1 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_1)	32	RW	0000_0000h
5B0h	KPP_IPP_IND_COL_SELECT_INPUT_2 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_2)	32	RW	0000_0000h
5B4h	KPP_IPP_IND_COL_SELECT_INPUT_3 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_3)	32	RW	0000_0000h
5B8h	KPP_IPP_IND_COL_SELECT_INPUT_4 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_4)	32	RW	0000_0000h
5BCh	KPP_IPP_IND_COL_SELECT_INPUT_5 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_5)	32	RW	0000_0000h
5C0h	KPP_IPP_IND_COL_SELECT_INPUT_6 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_6)	32	RW	0000_0000h
5C4h	KPP_IPP_IND_COL_SELECT_INPUT_7 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_7)	32	RW	0000_0000h
5C8h	KPP_IPP_IND_ROW_SELECT_INPUT_0 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_0)	32	RW	0000_0000h
5CCh	KPP_IPP_IND_ROW_SELECT_INPUT_1 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_1)	32	RW	0000_0000h
5D0h	KPP_IPP_IND_ROW_SELECT_INPUT_2 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_2)	32	RW	0000_0000h
5D4h	KPP_IPP_IND_ROW_SELECT_INPUT_3 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_3)	32	RW	0000_0000h
5D8h	KPP_IPP_IND_ROW_SELECT_INPUT_4 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5DCh	KPP_IPP_IND_ROW_SELECT_INPUT_5 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_5)	32	RW	0000_0000h
5E0h	KPP_IPP_IND_ROW_SELECT_INPUT_6 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_6)	32	RW	0000_0000h
5E4h	KPP_IPP_IND_ROW_SELECT_INPUT_7 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_7)	32	RW	0000_0000h
5E8h	LPI2C3_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C3_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	RW	0000_0000h
5ECh	LPI2C3_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C3_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	RW	0000_0000h
5F0h	LPI2C4_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C4_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	RW	0000_0000h
5F4h	LPI2C4_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C4_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	RW	0000_0000h
5F8h	LPI2C5_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C5_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	RW	0000_0000h
5FCh	LPI2C5_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C5_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	RW	0000_0000h
600h	LPI2C6_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C6_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	RW	0000_0000h
604h	LPI2C6_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C6_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	RW	0000_0000h
608h	LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)	32	RW	0000_0000h
60Ch	LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_1 DAISY Register (LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_1)	32	RW	0000_0000h
610h	LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_2 DAISY Register (LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_2)	32	RW	0000_0000h
614h	LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_3 DAISY Register (LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_3)	32	RW	0000_0000h
618h	LPSPI3_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI3_IPP_IND_LPSPI_SCK_SELECT_INPUT)	32	RW	0000_0000h
61Ch	LPSPI3_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI3_IPP_IND_LPSPI_SDI_SELECT_INPUT)	32	RW	0000_0000h
620h	LPSPI3_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI3_IPP_IND_LPSPI_SDO_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
624h	LPSPI4_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI4_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)	32	RW	0000_0000h
628h	LPSPI4_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI4_IPP_IND_LPSPI_SCK_SELECT_INPUT)	32	RW	0000_0000h
62Ch	LPSPI4_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI4_IPP_IND_LPSPI_SDI_SELECT_INPUT)	32	RW	0000_0000h
630h	LPSPI4_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI4_IPP_IND_LPSPI_SDO_SELECT_INPUT)	32	RW	0000_0000h
634h	LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)	32	RW	0000_0000h
638h	LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_1 DAISY Register (LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_1)	32	RW	0000_0000h
63Ch	LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_2 DAISY Register (LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_2)	32	RW	0000_0000h
640h	LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_3 DAISY Register (LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_3)	32	RW	0000_0000h
644h	LPSPI5_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI5_IPP_IND_LPSPI_SCK_SELECT_INPUT)	32	RW	0000_0000h
648h	LPSPI5_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI5_IPP_IND_LPSPI_SDI_SELECT_INPUT)	32	RW	0000_0000h
64Ch	LPSPI5_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI5_IPP_IND_LPSPI_SDO_SELECT_INPUT)	32	RW	0000_0000h
650h	LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)	32	RW	0000_0000h
654h	LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_1 DAISY Register (LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_1)	32	RW	0000_0000h
658h	LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_2 DAISY Register (LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_2)	32	RW	0000_0000h
65Ch	LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_3 DAISY Register (LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_3)	32	RW	0000_0000h
660h	LPSPI6_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI6_IPP_IND_LPSPI_SCK_SELECT_INPUT)	32	RW	0000_0000h
664h	LPSPI6_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI6_IPP_IND_LPSPI_SDI_SELECT_INPUT)	32	RW	0000_0000h
668h	LPSPI6_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI6_IPP_IND_LPSPI_SDO_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
66Ch	LPUART10_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART10_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	RW	0000_0000h
670h	LPUART10_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART10_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	RW	0000_0000h
674h	LPUART11_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART11_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	RW	0000_0000h
678h	LPUART11_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART11_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	RW	0000_0000h
67Ch	LPUART3_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART3_IPP_IND_LPUART_CTS_N_SELECT_INPUT)	32	RW	0000_0000h
680h	LPUART3_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART3_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	RW	0000_0000h
684h	LPUART3_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART3_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	RW	0000_0000h
688h	LPUART4_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART4_IPP_IND_LPUART_CTS_N_SELECT_INPUT)	32	RW	0000_0000h
68Ch	LPUART5_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_CTS_N_SELECT_INPUT)	32	RW	0000_0000h
690h	LPUART5_IPP_IND_LPUART_DCD_N_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_DCD_N_SELECT_INPUT)	32	RW	0000_0000h
694h	LPUART5_IPP_IND_LPUART_DSR_N_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_DSR_N_SELECT_INPUT)	32	RW	0000_0000h
698h	LPUART5_IPP_IND_LPUART_RI_N_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_RI_N_SELECT_INPUT)	32	RW	0000_0000h
69Ch	LPUART5_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	RW	0000_0000h
6A0h	LPUART5_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	RW	0000_0000h
6A4h	LPUART6_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_CTS_N_SELECT_INPUT)	32	RW	0000_0000h
6A8h	LPUART6_IPP_IND_LPUART_DCD_N_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_DCD_N_SELECT_INPUT)	32	RW	0000_0000h
6ACh	LPUART6_IPP_IND_LPUART_DSR_N_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_DSR_N_SELECT_INPUT)	32	RW	0000_0000h
6B0h	LPUART6_IPP_IND_LPUART_RI_N_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_RI_N_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
6B4h	LPUART6_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	RW	0000_0000h
6B8h	LPUART6_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	RW	0000_0000h
6BCh	LPUART8_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART8_IPP_IND_LPUART_CTS_N_SELECT_INPUT)	32	RW	0000_0000h
6C0h	LPUART8_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART8_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	RW	0000_0000h
6C4h	LPUART8_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART8_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	RW	0000_0000h
6C8h	LPUART9_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART9_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	RW	0000_0000h
6CCh	LPUART9_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART9_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	RW	0000_0000h
6D0h	MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_0 DAISY Register (MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_0)	32	RW	0000_0000h
6D4h	MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_1 DAISY Register (MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_1)	32	RW	0000_0000h
6D8h	MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_2 DAISY Register (MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_2)	32	RW	0000_0000h
6DCh	MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_3 DAISY Register (MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_3)	32	RW	0000_0000h
798h	NETC_EMDIO_IN_SELECT_INPUT DAISY Register (NETC_EMDIO_IN_SELECT_INPUT)	32	RW	0000_0000h
79Ch	NETC_ETH2_COL_SELECT_INPUT DAISY Register (NETC_ETH2_COL_SELECT_INPUT)	32	RW	0000_0000h
7A0h	NETC_ETH2_CRS_SELECT_INPUT DAISY Register (NETC_ETH2_CRS_SELECT_INPUT)	32	RW	0000_0000h
7A4h	NETC_ETH2_SLV_MDC_IN_SELECT_INPUT DAISY Register (NETC_ETH2_SLV_MDC_IN_SELECT_INPUT)	32	RW	0000_0000h
7A8h	NETC_ETH2_SLV_MDIO_IN_SELECT_INPUT DAISY Register (NETC_ETH2_SLV_MDIO_IN_SELECT_INPUT)	32	RW	0000_0000h
7ACh	NETC_ETH3_COL_SELECT_INPUT DAISY Register (NETC_ETH3_COL_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
7B0h	NETC_ETH3_CRS_SELECT_INPUT DAISY Register (NETC_ETH3_CRS_SELECT_INPUT)	32	RW	0000_0000h
7B4h	NETC_ETH3_SLV_MDC_IN_SELECT_INPUT DAISY Register (NETC_ETH3_SLV_MDC_IN_SELECT_INPUT)	32	RW	0000_0000h
7B8h	NETC_ETH3_SLV_MDIO_IN_SELECT_INPUT DAISY Register (NETC_ETH3_SLV_MDIO_IN_SELECT_INPUT)	32	RW	0000_0000h
7BCh	NETC_ETH4_COL_SELECT_INPUT DAISY Register (NETC_ETH4_COL_SELECT_INPUT)	32	RW	0000_0000h
7C0h	NETC_ETH4_CRS_SELECT_INPUT DAISY Register (NETC_ETH4_CRS_SELECT_INPUT)	32	RW	0000_0000h
7C4h	NETC_ETH4_SLV_MDC_IN_SELECT_INPUT DAISY Register (NETC_ETH4_SLV_MDC_IN_SELECT_INPUT)	32	RW	0000_0000h
7C8h	NETC_ETH4_SLV_MDIO_IN_SELECT_INPUT DAISY Register (NETC_ETH4_SLV_MDIO_IN_SELECT_INPUT)	32	RW	0000_0000h
7CCh	NETC_TMR_TRIG1_SELECT_INPUT DAISY Register (NETC_TMR_TRIG1_SELECT_INPUT)	32	RW	0000_0000h
7D0h	NETC_TMR_TRIG2_SELECT_INPUT DAISY Register (NETC_TMR_TRIG2_SELECT_INPUT)	32	RW	0000_0000h
7D4h	NETC_CLKGEN_IPP_TMR_CLK_SELECT_INPUT DAISY Register (NETC_CLKGEN_IPP_TMR_CLK_SELECT_INPUT)	32	RW	0000_0000h
7D8h	NETC_PINMUX_IPP_IND_ETH0_RX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RX_CLK_SELECT_INPUT)	32	RW	0000_0000h
7DCh	NETC_PINMUX_IPP_IND_ETH0_RX_DV_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RX_DV_SELECT_INPUT)	32	RW	0000_0000h
7E0h	NETC_PINMUX_IPP_IND_ETH0_RX_ER_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RX_ER_SELECT_INPUT)	32	RW	0000_0000h
7E4h	NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_0 DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_0)	32	RW	0000_0000h
7E8h	NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_1 DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_1)	32	RW	0000_0000h
7ECh	NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_2 DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
7F0h	NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_3 DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_3)	32	RW	0000_0000h
7F4h	NETC_PINMUX_IPP_IND_ETH0_TX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH0_TX_CLK_SELECT_INPUT)	32	RW	0000_0000h
7F8h	NETC_PINMUX_IPP_IND_ETH2_RX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RX_CLK_SELECT_INPUT)	32	RW	0000_0000h
7FCh	NETC_PINMUX_IPP_IND_ETH2_RX_DV_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RX_DV_SELECT_INPUT)	32	RW	0000_0000h
800h	NETC_PINMUX_IPP_IND_ETH2_RX_ER_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RX_ER_SELECT_INPUT)	32	RW	0000_0000h
804h	NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_0 DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_0)	32	RW	0000_0000h
808h	NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_1 DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_1)	32	RW	0000_0000h
80Ch	NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_2 DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_2)	32	RW	0000_0000h
810h	NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_3 DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_3)	32	RW	0000_0000h
814h	NETC_PINMUX_IPP_IND_ETH2_TX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH2_TX_CLK_SELECT_INPUT)	32	RW	0000_0000h
818h	NETC_PINMUX_IPP_IND_ETH3_RX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RX_CLK_SELECT_INPUT)	32	RW	0000_0000h
81Ch	NETC_PINMUX_IPP_IND_ETH3_RX_DV_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RX_DV_SELECT_INPUT)	32	RW	0000_0000h
820h	NETC_PINMUX_IPP_IND_ETH3_RX_ER_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RX_ER_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
824h	NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_0 DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_0)	32	RW	0000_0000h
828h	NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_1 DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_1)	32	RW	0000_0000h
82Ch	NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_2 DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_2)	32	RW	0000_0000h
830h	NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_3 DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_3)	32	RW	0000_0000h
834h	NETC_PINMUX_IPP_IND_ETH3_TX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH3_TX_CLK_SELECT_INPUT)	32	RW	0000_0000h
838h	NETC_PINMUX_IPP_IND_ETH4_RX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RX_CLK_SELECT_INPUT)	32	RW	0000_0000h
83Ch	NETC_PINMUX_IPP_IND_ETH4_RX_DV_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RX_DV_SELECT_INPUT)	32	RW	0000_0000h
840h	NETC_PINMUX_IPP_IND_ETH4_RX_ER_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RX_ER_SELECT_INPUT)	32	RW	0000_0000h
844h	NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_0 DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_0)	32	RW	0000_0000h
848h	NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_1 DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_1)	32	RW	0000_0000h
84Ch	NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_2 DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_2)	32	RW	0000_0000h
850h	NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_3 DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_3)	32	RW	0000_0000h
854h	NETC_PINMUX_IPP_IND_ETH4_TX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH4_TX_CLK_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
858h	QTIMER1_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER1_TMR0_INPUT_SELECT_INPUT)	32	RW	0000_0000h
85Ch	QTIMER1_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER1_TMR1_INPUT_SELECT_INPUT)	32	RW	0000_0000h
860h	QTIMER1_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER1_TMR2_INPUT_SELECT_INPUT)	32	RW	0000_0000h
864h	QTIMER2_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER2_TMR0_INPUT_SELECT_INPUT)	32	RW	0000_0000h
868h	QTIMER2_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER2_TMR1_INPUT_SELECT_INPUT)	32	RW	0000_0000h
86Ch	QTIMER2_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER2_TMR2_INPUT_SELECT_INPUT)	32	RW	0000_0000h
870h	QTIMER3_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER3_TMR0_INPUT_SELECT_INPUT)	32	RW	0000_0000h
874h	QTIMER3_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER3_TMR1_INPUT_SELECT_INPUT)	32	RW	0000_0000h
878h	QTIMER3_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER3_TMR2_INPUT_SELECT_INPUT)	32	RW	0000_0000h
87Ch	QTIMER4_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER4_TMR0_INPUT_SELECT_INPUT)	32	RW	0000_0000h
880h	QTIMER4_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER4_TMR1_INPUT_SELECT_INPUT)	32	RW	0000_0000h
884h	QTIMER4_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER4_TMR2_INPUT_SELECT_INPUT)	32	RW	0000_0000h
888h	QTIMER5_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER5_TMR0_INPUT_SELECT_INPUT)	32	RW	0000_0000h
88Ch	QTIMER5_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER5_TMR1_INPUT_SELECT_INPUT)	32	RW	0000_0000h
890h	QTIMER5_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER5_TMR2_INPUT_SELECT_INPUT)	32	RW	0000_0000h
894h	QTIMER6_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER6_TMR0_INPUT_SELECT_INPUT)	32	RW	0000_0000h
898h	QTIMER6_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER6_TMR1_INPUT_SELECT_INPUT)	32	RW	0000_0000h
89Ch	QTIMER6_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER6_TMR2_INPUT_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8A0h	QTIMER7_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER7_TMR0_INPUT_SELECT_INPUT)	32	RW	0000_0000h
8A4h	QTIMER7_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER7_TMR1_INPUT_SELECT_INPUT)	32	RW	0000_0000h
8A8h	QTIMER8_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER8_TMR0_INPUT_SELECT_INPUT)	32	RW	0000_0000h
8ACh	QTIMER8_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER8_TMR1_INPUT_SELECT_INPUT)	32	RW	0000_0000h
8B0h	SAI4_IPG_CLK_SAI_MCLK_SELECT_INPUT DAISY Register (SAI4_IPG_CLK_SAI_MCLK_SELECT_INPUT)	32	RW	0000_0000h
8B4h	SAI4_IPP_IND_SAI_RXBCLK_SELECT_INPUT DAISY Register (SAI4_IPP_IND_SAI_RXBCLK_SELECT_INPUT)	32	RW	0000_0000h
8B8h	SAI4_IPP_IND_SAI_RXDATA_SELECT_INPUT_0 DAISY Register (SAI4_IPP_IND_SAI_RXDATA_SELECT_INPUT_0)	32	RW	0000_0000h
8BCh	SAI4_IPP_IND_SAI_RXDATA_SELECT_INPUT_1 DAISY Register (SAI4_IPP_IND_SAI_RXDATA_SELECT_INPUT_1)	32	RW	0000_0000h
8C0h	SAI4_IPP_IND_SAI_RXSYNC_SELECT_INPUT DAISY Register (SAI4_IPP_IND_SAI_RXSYNC_SELECT_INPUT)	32	RW	0000_0000h
8C4h	SAI4_IPP_IND_SAI_TXBCLK_SELECT_INPUT DAISY Register (SAI4_IPP_IND_SAI_TXBCLK_SELECT_INPUT)	32	RW	0000_0000h
8C8h	SAI4_IPP_IND_SAI_TXSYNC_SELECT_INPUT DAISY Register (SAI4_IPP_IND_SAI_TXSYNC_SELECT_INPUT)	32	RW	0000_0000h
8D4h	SINC1_IPP_IND_EMBIT_SELECT_INPUT_0 DAISY Register (SINC1_IPP_IND_EMBIT_SELECT_INPUT_0)	32	RW	0000_0000h
8D8h	SINC1_IPP_IND_EMBIT_SELECT_INPUT_1 DAISY Register (SINC1_IPP_IND_EMBIT_SELECT_INPUT_1)	32	RW	0000_0000h
8DCh	SINC1_IPP_IND_EMBIT_SELECT_INPUT_2 DAISY Register (SINC1_IPP_IND_EMBIT_SELECT_INPUT_2)	32	RW	0000_0000h
8E0h	SINC1_IPP_IND_EMBIT_SELECT_INPUT_3 DAISY Register (SINC1_IPP_IND_EMBIT_SELECT_INPUT_3)	32	RW	0000_0000h
8E4h	SINC1_IPP_IND_EMCLK_SELECT_INPUT_0 DAISY Register (SINC1_IPP_IND_EMCLK_SELECT_INPUT_0)	32	RW	0000_0000h
8E8h	SINC1_IPP_IND_EMCLK_SELECT_INPUT_1 DAISY Register (SINC1_IPP_IND_EMCLK_SELECT_INPUT_1)	32	RW	0000_0000h
8ECh	SINC1_IPP_IND_EMCLK_SELECT_INPUT_2 DAISY Register (SINC1_IPP_IND_EMCLK_SELECT_INPUT_2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8F0h	SINC1_IPP_IND_EMCLK_SELECT_INPUT_3 DAISY Register (SINC1_IPP_IND_EMCLK_SELECT_INPUT_3)	32	RW	0000_0000h
8F4h	SINC2_IPP_IND_EMBIT_SELECT_INPUT_2 DAISY Register (SINC2_IPP_IND_EMBIT_SELECT_INPUT_2)	32	RW	0000_0000h
8F8h	SINC2_IPP_IND_EMBIT_SELECT_INPUT_3 DAISY Register (SINC2_IPP_IND_EMBIT_SELECT_INPUT_3)	32	RW	0000_0000h
8FCh	SINC2_IPP_IND_EMCLK_SELECT_INPUT_0 DAISY Register (SINC2_IPP_IND_EMCLK_SELECT_INPUT_0)	32	RW	0000_0000h
900h	SINC2_IPP_IND_EMCLK_SELECT_INPUT_2 DAISY Register (SINC2_IPP_IND_EMCLK_SELECT_INPUT_2)	32	RW	0000_0000h
904h	SINC2_IPP_IND_EMCLK_SELECT_INPUT_3 DAISY Register (SINC2_IPP_IND_EMCLK_SELECT_INPUT_3)	32	RW	0000_0000h
908h	SPDIF_SPDIF_IN1_SELECT_INPUT DAISY Register (SPDIF_SPDIF_IN1_SELECT_INPUT)	32	RW	0000_0000h
914h	USB_IPP_IND_OTG2_OC_SELECT_INPUT DAISY Register (USB_IPP_IND_OTG2_OC_SELECT_INPUT)	32	RW	0000_0000h
918h	USB_IPP_IND_OTG_OC_SELECT_INPUT DAISY Register (USB_IPP_IND_OTG_OC_SELECT_INPUT)	32	RW	0000_0000h
91Ch	USBPHY1_USB_ID_SELECT_INPUT DAISY Register (USBPHY1_USB_ID_SELECT_INPUT)	32	RW	0000_0000h
920h	USBPHY2_USB_ID_SELECT_INPUT DAISY Register (USBPHY2_USB_ID_SELECT_INPUT)	32	RW	0000_0000h
924h	USDHC1_IPP_CARD_DET_SELECT_INPUT DAISY Register (USDHC1_IPP_CARD_DET_SELECT_INPUT)	32	RW	0000_0000h
928h	USDHC1_IPP_WP_ON_SELECT_INPUT DAISY Register (USDHC1_IPP_WP_ON_SELECT_INPUT)	32	RW	0000_0000h
92Ch	USDHC2_IPP_CARD_DET_SELECT_INPUT DAISY Register (USDHC2_IPP_CARD_DET_SELECT_INPUT)	32	RW	0000_0000h
930h	USDHC2_IPP_WP_ON_SELECT_INPUT DAISY Register (USDHC2_IPP_WP_ON_SELECT_INPUT)	32	RW	0000_0000h
934h	XBAR1_XBAR_IN_SELECT_INPUT_14 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_14)	32	RW	0000_0000h
938h	XBAR1_XBAR_IN_SELECT_INPUT_15 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_15)	32	RW	0000_0000h
93Ch	XBAR1_XBAR_IN_SELECT_INPUT_17 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_17)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
940h	XBAR1_XBAR_IN_SELECT_INPUT_18 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_18)	32	RW	0000_0000h
944h	XBAR1_XBAR_IN_SELECT_INPUT_19 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_19)	32	RW	0000_0000h
948h	XBAR1_XBAR_IN_SELECT_INPUT_20 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_20)	32	RW	0000_0000h
94Ch	XBAR1_XBAR_IN_SELECT_INPUT_21 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_21)	32	RW	0000_0000h
950h	XBAR1_XBAR_IN_SELECT_INPUT_22 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_22)	32	RW	0000_0000h
954h	XBAR1_XBAR_IN_SELECT_INPUT_23 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_23)	32	RW	0000_0000h
958h	XBAR1_XBAR_IN_SELECT_INPUT_24 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_24)	32	RW	0000_0000h
95Ch	XBAR1_XBAR_IN_SELECT_INPUT_25 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_25)	32	RW	0000_0000h
960h	XBAR1_XBAR_IN_SELECT_INPUT_26 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_26)	32	RW	0000_0000h
964h	XBAR1_XBAR_IN_SELECT_INPUT_27 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_27)	32	RW	0000_0000h
968h	XBAR1_XBAR_IN_SELECT_INPUT_28 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_28)	32	RW	0000_0000h
96Ch	XBAR1_XBAR_IN_SELECT_INPUT_29 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_29)	32	RW	0000_0000h
970h	XBAR1_XBAR_IN_SELECT_INPUT_30 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_30)	32	RW	0000_0000h
974h	XBAR1_XBAR_IN_SELECT_INPUT_31 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_31)	32	RW	0000_0000h
978h	XBAR1_XBAR_IN_SELECT_INPUT_32 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_32)	32	RW	0000_0000h
97Ch	XBAR1_XBAR_IN_SELECT_INPUT_33 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_33)	32	RW	0000_0000h
980h	XBAR1_XBAR_IN_SELECT_INPUT_34 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_34)	32	RW	0000_0000h
984h	XBAR1_XBAR_IN_SELECT_INPUT_35 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_35)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
988h	XBAR1_XBAR_IN_SELECT_INPUT_36 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_36)	32	RW	0000_0000h
98Ch	XBAR1_XBAR_IN_SELECT_INPUT_37 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_37)	32	RW	0000_0000h
A00h	XSPI_SLV_IPP_IND_CS_SELECT_INPUT DAISY Register (XSPI_SLV_IPP_IND_CS_SELECT_INPUT)	32	RW	0000_0000h
A04h	XSPI_SLV_IPP_IND_DQS_SELECT_INPUT DAISY Register (XSPI_SLV_IPP_IND_DQS_SELECT_INPUT)	32	RW	0000_0000h
A08h	XSPI_SLV_IPP_IND_IO_SELECT_INPUT_0 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_0)	32	RW	0000_0000h
A0Ch	XSPI_SLV_IPP_IND_IO_SELECT_INPUT_1 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_1)	32	RW	0000_0000h
A10h	XSPI_SLV_IPP_IND_IO_SELECT_INPUT_2 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_2)	32	RW	0000_0000h
A14h	XSPI_SLV_IPP_IND_IO_SELECT_INPUT_3 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_3)	32	RW	0000_0000h
A18h	XSPI_SLV_IPP_IND_IO_SELECT_INPUT_4 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_4)	32	RW	0000_0000h
A1Ch	XSPI_SLV_IPP_IND_IO_SELECT_INPUT_5 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_5)	32	RW	0000_0000h
A20h	XSPI_SLV_IPP_IND_IO_SELECT_INPUT_6 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_6)	32	RW	0000_0000h
A24h	XSPI_SLV_IPP_IND_IO_SELECT_INPUT_7 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_7)	32	RW	0000_0000h
A28h	XSPI_SLV_IPP_IND_SCK_SELECT_INPUT DAISY Register (XSPI_SLV_IPP_IND_SCK_SELECT_INPUT)	32	RW	0000_0000h

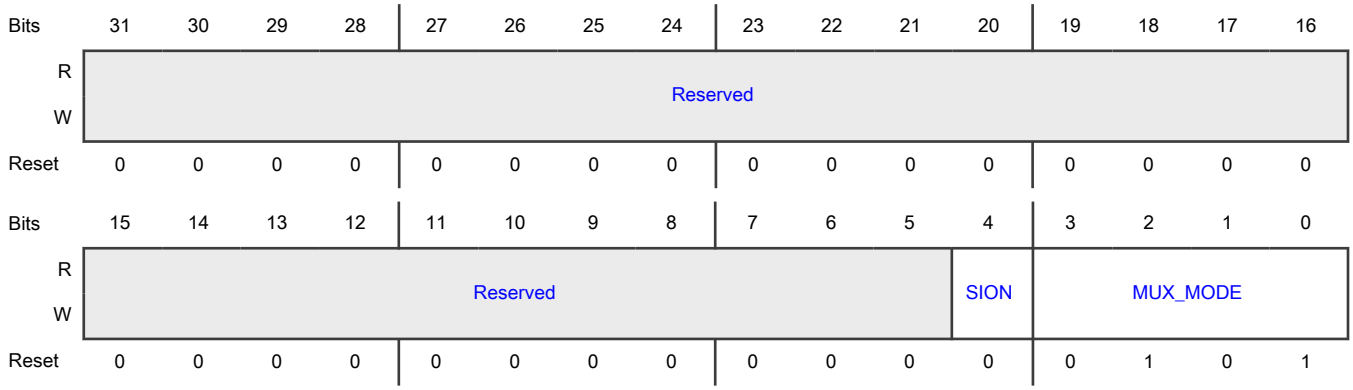
17.4.1.2 SW_MUX_CTL_PAD_GPIO_EMC_B1_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_00)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_00	10h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_00
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_00. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA00 of instance: semc 0001b - Select mux mode: ALT1 mux port: XBAR1_XBAR_INOUT04 of instance: xbar1 0010b - Select mux mode: ALT2 mux port: SINC3_MOD_CLK0 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART3_CTS_B of instance: lpuart3 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_TXD03 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO00 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_ROW03 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO00 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_TXD03 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_TX_DATA3_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA00 of instance: ahb_sramc

17.4.1.3 SW_MUX_CTL_PAD_GPIO_EMC_B1_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_01)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_01	14h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_01
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_01. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA01 of instance: semc 0001b - Select mux mode: ALT1 mux port: XBAR1_XBAR_INOUT05 of instance: xbar1 0010b - Select mux mode: ALT2 mux port: SINC3_MOD_CLK1 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART3_RTS_B of instance: lpuart3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_TXD02 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO01 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_COL03 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO01 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_TXD02 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_TX_DATA2_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA01 of instance: ahb_sramc

17.4.1.4 SW_MUX_CTL_PAD_GPIO_EMC_B1_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_02)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_02	18h

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_02
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_02. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA02 of instance: semc 0001b - Select mux mode: ALT1 mux port: XBAR1_XBAR_INOUT06 of instance: xbar1 0010b - Select mux mode: ALT2 mux port: SINC3_MOD_CLK2 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART3_RX of instance: lpuart3 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_RX_CLK of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO02 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_ROW02 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO02 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_RX_CLK of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_RX_CLK_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA02 of instance: ahb_sramc

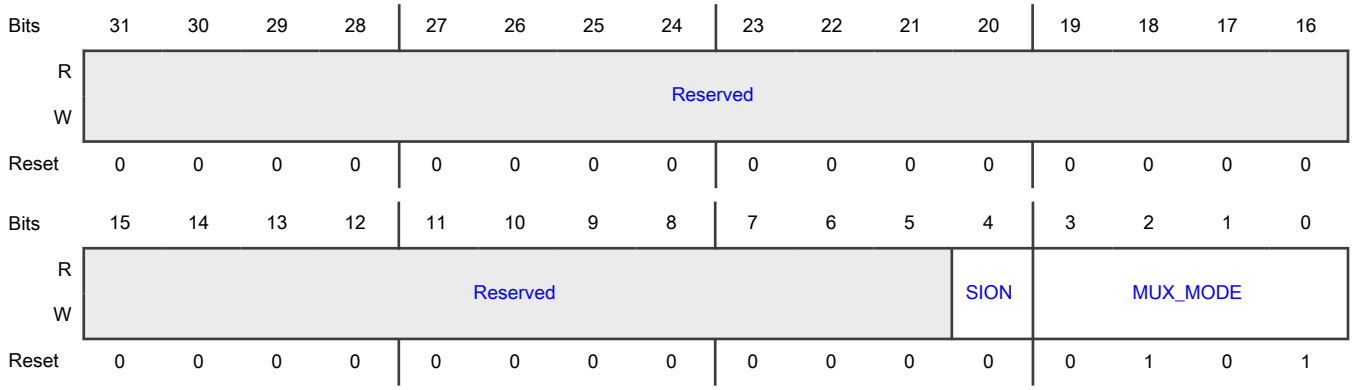
17.4.1.5 SW_MUX_CTL_PAD_GPIO_EMC_B1_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_03)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_03	1Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_03
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_03. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA03 of instance: semc 0001b - Select mux mode: ALT1 mux port: XBAR1_XBAR_INOUT07 of instance: xbar1 0010b - Select mux mode: ALT2 mux port: SINC3_EMCLK00 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART3_TX of instance: lpuart3 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_RXD03 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO03 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_COL02 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO03 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_RXD03 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_RX_DATA3_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA03 of instance: ahb_sramc

17.4.1.6 SW_MUX_CTL_PAD_GPIO_EMC_B1_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_04)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_04	20h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_04
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_04. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA04 of instance: semc 0001b - Select mux mode: ALT1 mux port: XBAR1_XBAR_INOUT08 of instance: xbar1 0010b - Select mux mode: ALT2 mux port: SINC3_EMBIT00 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART3_DSR_B of instance: lpuart3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_RXD02 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO04 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_ROW01 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO04 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_RXD02 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_RX_DATA2_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA04 of instance: ahb_sramc

17.4.1.7 SW_MUX_CTL_PAD_GPIO_EMC_B1_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_05)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_05	24h

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_05
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_05. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA05 of instance: semc 0001b - Select mux mode: ALT1 mux port: XBAR1_XBAR_INOUT09 of instance: xbar1 0010b - Select mux mode: ALT2 mux port: SINC3_EMCLK01 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART3_DCD_B of instance: lpuart3 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_TXD00 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO05 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_ROW07 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO05 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_TXD00 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_TX_DATA0_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA05 of instance: ahb_sramc

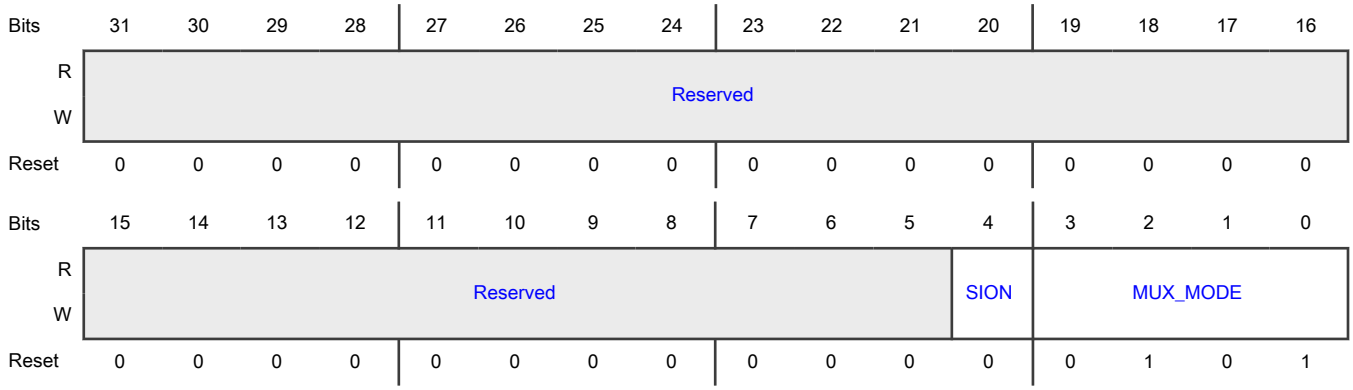
17.4.1.8 SW_MUX_CTL_PAD_GPIO_EMC_B1_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_06)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_06	28h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_06
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_06. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA06 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM4_PWMB03 of instance: flexpwm4 0010b - Select mux mode: ALT2 mux port: SINC3_EMBIT01 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART3_RI_B of instance: lpuart3 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_TXD01 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO06 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_COL07 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO06 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_TXD01 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_TX_DATA1_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA06 of instance: ahb_sramc

17.4.1.9 SW_MUX_CTL_PAD_GPIO_EMC_B1_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_07)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_07	2Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_07
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_07. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA07 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM4_PWMA03 of instance: flexpwm4 0010b - Select mux mode: ALT2 mux port: SINC3_EMCLK02 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART3_DTR_B of instance: lpuart3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_TX_EN of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO07 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_ROW06 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO07 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_TX_EN of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_TX_EN_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA07 of instance: ahb_sramc

17.4.1.10 SW_MUX_CTL_PAD_GPIO_EMC_B1_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_08)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_08	30h

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_08
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_08. 0000b - Select mux mode: ALT0 mux port: SEMC_DM00 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM2_PWMB03 of instance: flexpwm2 0010b - Select mux mode: ALT2 mux port: SINC3_EMBIT02 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART4_DSR_B of instance: lpuart4 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_TX_CLK of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO08 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_COL06 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO08 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_TX_CLK of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_TX_CLK_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_LBB of instance: ahb_sramc

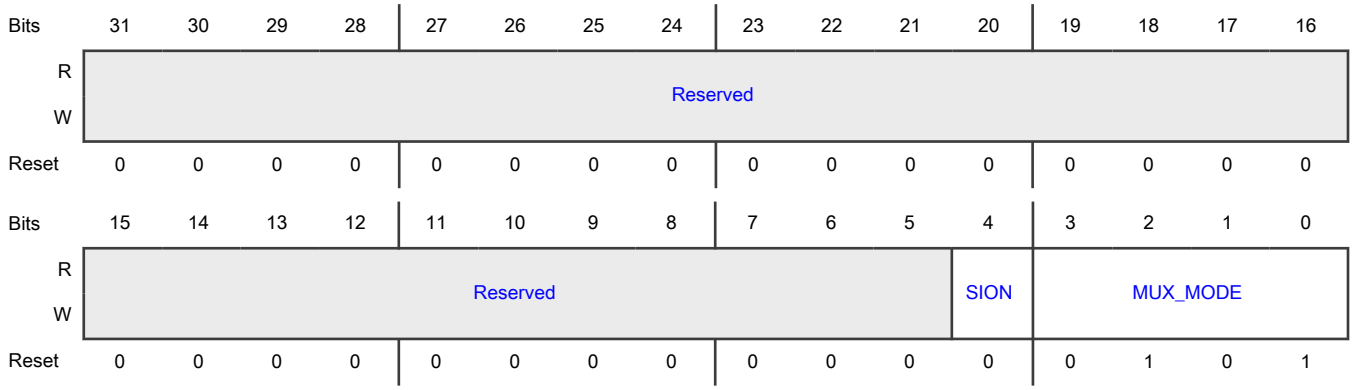
17.4.1.11 SW_MUX_CTL_PAD_GPIO_EMC_B1_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_09)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_09	34h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_09
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_09. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR00 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM2_PWMA03 of instance: flexpwm2 0010b - Select mux mode: ALT2 mux port: SINC3_EMCLK03 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART4_DCD_B of instance: lpuart4 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_RXD00 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO09 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_ROW05 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO09 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_RXD00 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_RX_DATA0_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR00 of instance: ahb_sramc

17.4.1.12 SW_MUX_CTL_PAD_GPIO_EMC_B1_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_10)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_10	38h

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_10
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_10. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR01 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM3_PWMB03 of instance: flexpwm3 0010b - Select mux mode: ALT2 mux port: SINC3_EMBIT03 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART4_RI_B of instance: lpuart4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_RXD01 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO10 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_COL05 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO10 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_RXD01 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_RX_DATA1_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR01 of instance: ahb_sramc

17.4.1.13 SW_MUX_CTL_PAD_GPIO_EMC_B1_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_11)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_11	3Ch

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_11
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_11. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR02 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM3_PWMA03 of instance: flexpwm3 0010b - Select mux mode: ALT2 mux port: SINC_FILTER_GLUE3_BREAK of instance: sinc_filter_glue3 0011b - Select mux mode: ALT3 mux port: LPUART4_DTR_B of instance: lpuart4 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_RX_DV of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO11 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_ROW04 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO11 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_RX_DV of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_RX_DV_0 of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR02 of instance: ahb_sramc

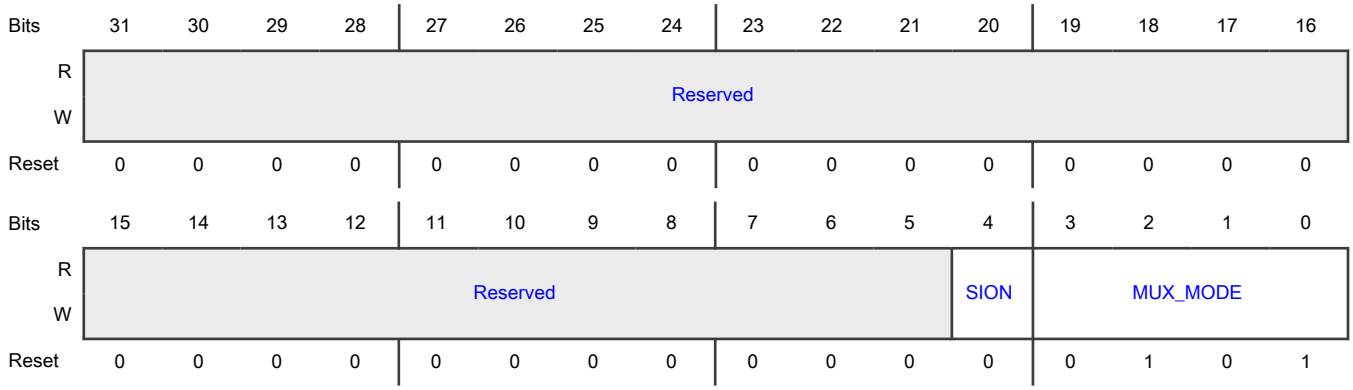
17.4.1.14 SW_MUX_CTL_PAD_GPIO_EMC_B1_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_12)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_12	40h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_12
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B1_12. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR03 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM4_PWMA00 of instance: flexpwm4 0010b - Select mux mode: ALT2 mux port: LPUART4_TX of instance: lpuart4 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_RX_ER of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO12 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_COL04 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO12 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_RX_ER of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: ECAT_PT0_RX_ER of instance: ecat 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR03 of instance: ahb_sramc

17.4.1.15 SW_MUX_CTL_PAD_GPIO_EMC_B1_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_13)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_13	44h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_13
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_13. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR04 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM4_PWMB00 of instance: flexpwm4 0010b - Select mux mode: ALT2 mux port: LPUART4_RX of instance: lpuart4 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH2_RX_DV of instance: netc_pinmux

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH3_TX_ER of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO13 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_COL01 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO13 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH4_TX_ER of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: QTIMER1_TIMER1 of instance: qtimer1 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR04 of instance: ahb_sramc

17.4.1.16 SW_MUX_CTL_PAD_GPIO_EMC_B1_14 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_14)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_14	48h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_14
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_14. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR05 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM4_PWMA01 of instance: flexpwm4 0010b - Select mux mode: ALT2 mux port: LPUART5_TX of instance: lpuart5 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH2_TX_EN of instance: netc_pinmux 0100b - Select mux mode: ALT4 mux port: NETC_ETH3_CRS of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO2_IO14 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_ROW00 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO14 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_ETH4_CRS of instance: netc 1010b - Select mux mode: ALT10 mux port: LPUART4_CTS_B of instance: lpuart4 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR05 of instance: ahb_sramc

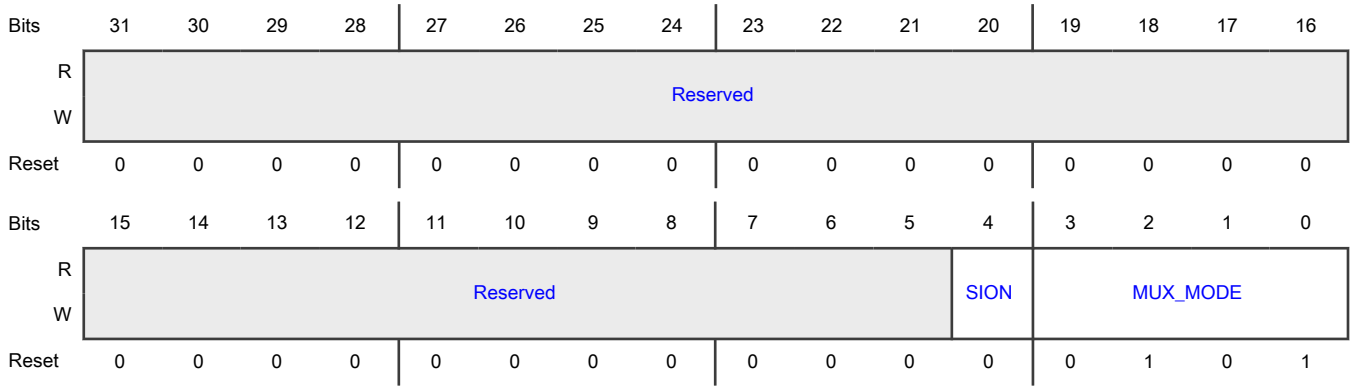
17.4.1.17 SW_MUX_CTL_PAD_GPIO_EMC_B1_15 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_15)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_15	4Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_15
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_15. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR06 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM4_PWMB01 of instance: flexpwm4 0010b - Select mux mode: ALT2 mux port: LPUART5_RX of instance: lpuart5 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH2_TX_CLK of instance: netc_pinmux 0100b - Select mux mode: ALT4 mux port: NETC_ETH3_COL of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO2_IO15 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: KPP_COL00 of instance: kpp 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO15 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_ETH4_COL of instance: netc 1010b - Select mux mode: ALT10 mux port: LPUART4_RTS_B of instance: lpuart4 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR06 of instance: ahb_sramc

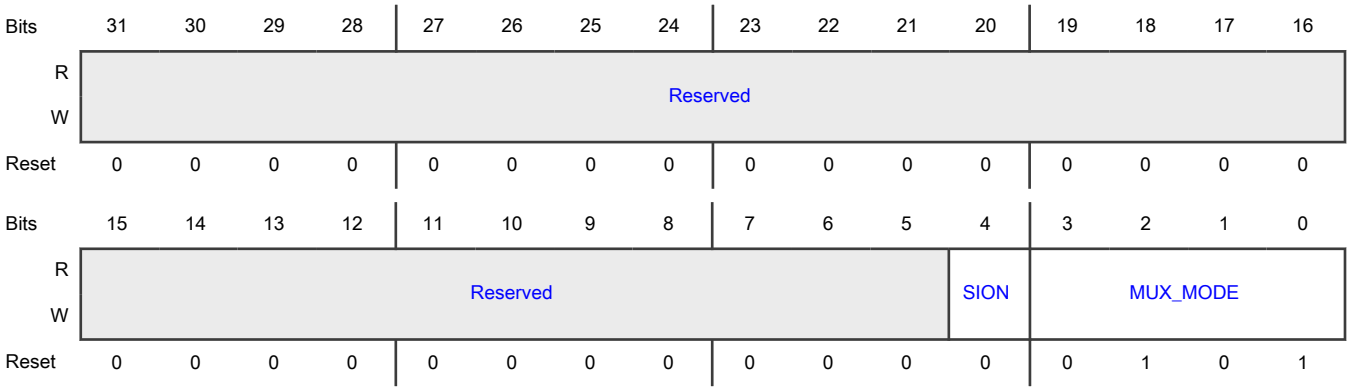
17.4.1.18 SW_MUX_CTL_PAD_GPIO_EMC_B1_16 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_16)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_16	50h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_16
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_16. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR07 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM4_PWMB02 of instance: flexpwm4 0010b - Select mux mode: ALT2 mux port: LPUART9_TX of instance: lpuart9 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH2_RXD00 of instance: netc_pinmux

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_ETH3_SLV_MDC of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO2_IO16 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: NETC_ETH4_SLV_MDC of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO16 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_ETH2_SLV_MDC of instance: netc 1010b - Select mux mode: ALT10 mux port: LPSPI6_PCS2 of instance: lpspi6 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR07 of instance: ahb_sramc

17.4.1.19 SW_MUX_CTL_PAD_GPIO_EMC_B1_17 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_17)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_17	54h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_17
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_17. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR08 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM4_PWMA02 of instance: flexpwm4 0010b - Select mux mode: ALT2 mux port: LPUART9_RX of instance: lpuart9 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH2_RXD01 of instance: netc_pinmux 0100b - Select mux mode: ALT4 mux port: NETC_ETH3_SLV_MDIO of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO2_IO17 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: NETC_ETH4_SLV_MDIO of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO17 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_ETH2_SLV_MDIO of instance: netc 1010b - Select mux mode: ALT10 mux port: LPSPI6_PCS1 of instance: lpspi6 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR08 of instance: ahb_sramc

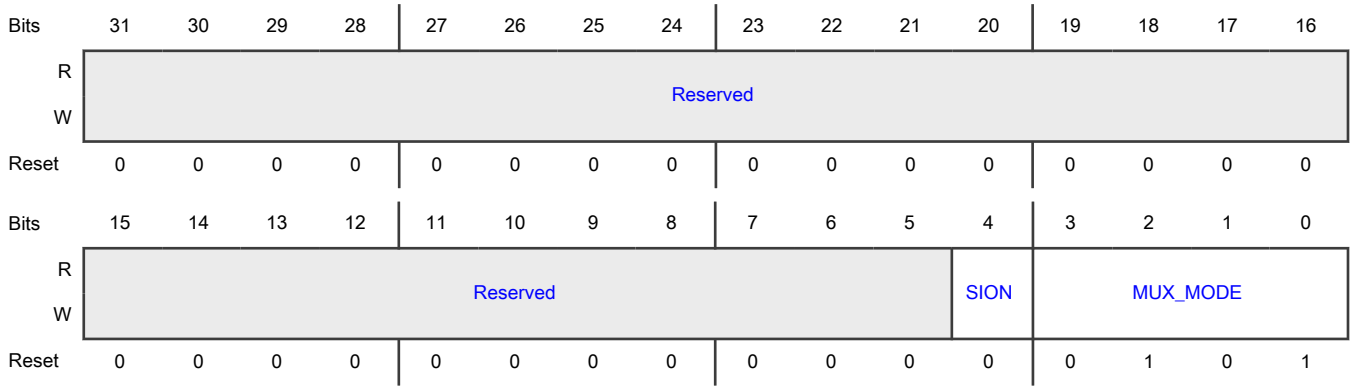
17.4.1.20 SW_MUX_CTL_PAD_GPIO_EMC_B1_18 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_18)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_18	58h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_18
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_EMC_B1_18. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR09 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM2_PWMA00 of instance: flexpwm2 0010b - Select mux mode: ALT2 mux port: QTIMER1_TIMER0 of instance: qtimer1 0011b - Select mux mode: ALT3 mux port: LPSPI6_SCK of instance: lpspi6 0100b - Select mux mode: ALT4 mux port: NETC_ETH2_CRS of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO2_IO18 of instance: gpio2 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO18 of instance: flexio1 1010b - Select mux mode: ALT10 mux port: NETC_EMDC of instance: netc 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR09 of instance: ahb_sramc

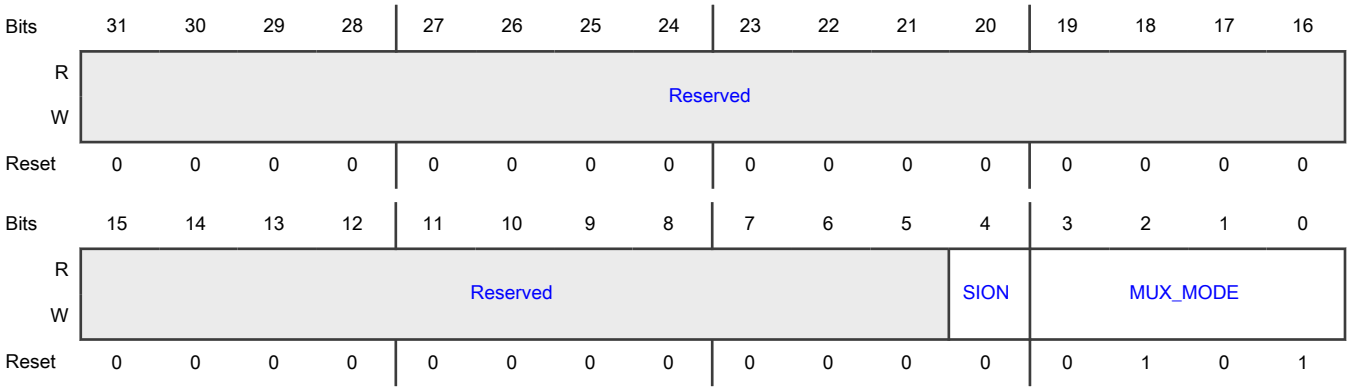
17.4.1.21 SW_MUX_CTL_PAD_GPIO_EMC_B1_19 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_19)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_19	5Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_19
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_EMC_B1_19. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR11 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM2_PWMB00 of instance: flexpwm2 0010b - Select mux mode: ALT2 mux port: QTIMER2_TIMER0 of instance: qtimer2 0011b - Select mux mode: ALT3 mux port: LPSPI6_SDI of instance: lpspi6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_ETH2_COL of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO2_IO19 of instance: gpio2 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO19 of instance: flexio1 1010b - Select mux mode: ALT10 mux port: NETC_EMDIO of instance: netc 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR11 of instance: ahb_sramc

17.4.1.22 SW_MUX_CTL_PAD_GPIO_EMC_B1_20 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_20)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_20	60h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4	Software Input On Field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SION	Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_20
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_EMC_B1_20. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR12 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM2_PWMA01 of instance: flexpwm2 0010b - Select mux mode: ALT2 mux port: QTIMER3_TIMER0 of instance: qtimer3 0011b - Select mux mode: ALT3 mux port: LPSPI6_SDO of instance: lpspi6 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_TX_ER of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO20 of instance: gpio2 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO20 of instance: flexio1 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR12 of instance: ahb_sramc

17.4.1.23 SW_MUX_CTL_PAD_GPIO_EMC_B1_21 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_21)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_21	64h

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_21
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B1_21. 0000b - Select mux mode: ALT0 mux port: SEMC_BA0 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM2_PWMB01 of instance: flexpwm2 0010b - Select mux mode: ALT2 mux port: QTIMER4_TIMER0 of instance: qtimer4 0011b - Select mux mode: ALT3 mux port: LPSPI6_PCS0 of instance: lpspi6 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_RX_CLK of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO21 of instance: gpio2 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO21 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: LPUART4_CTS_B of instance: lpuart4 1010b - Select mux mode: ALT10 mux port: FLEXSPI2_BUS2BIT_B_DQS of instance: flexspi2_bus2bit 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR13 of instance: ahb_sramc

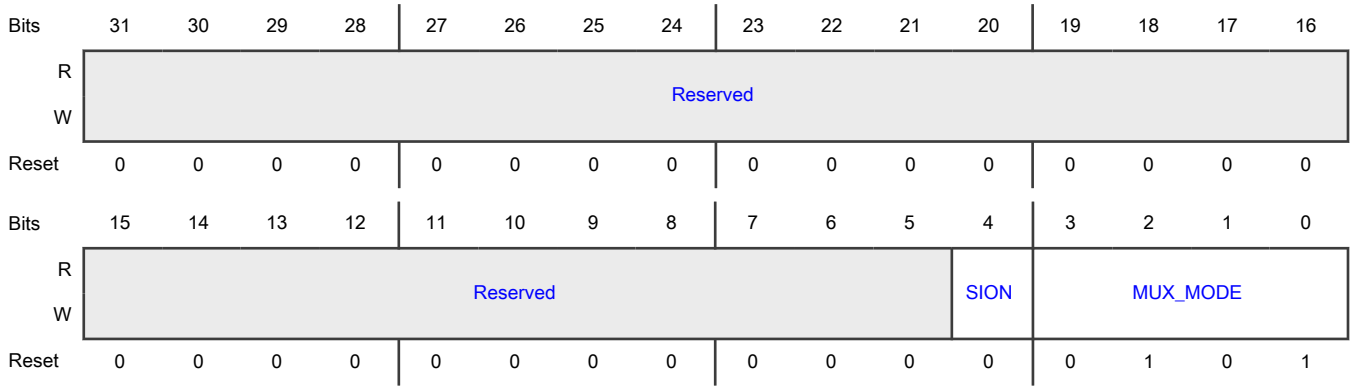
17.4.1.24 SW_MUX_CTL_PAD_GPIO_EMC_B1_22 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_22)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_22	68h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_22
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B1_22. 0000b - Select mux mode: ALT0 mux port: SEMC_BA1 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM2_PWMB02 of instance: flexpwm2 0010b - Select mux mode: ALT2 mux port: QTIMER5_TIMER0 of instance: qtimer5 0011b - Select mux mode: ALT3 mux port: LPSPI4_SCK of instance: lpspi4 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_RXD02 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO22 of instance: gpio2 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO22 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: LPUART4_RTS_B of instance: lpuart4 1010b - Select mux mode: ALT10 mux port: FLEXSPI2_BUS2BIT_B_DATA03 of instance: flexspi2_bus2bit 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR14 of instance: ahb_sramc

17.4.1.25 SW_MUX_CTL_PAD_GPIO_EMC_B1_23 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_23)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_23	6Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_23
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_EMC_B1_23. 0000b - Select mux mode: ALT0 mux port: SEMC_ADDR10 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM2_PWMA02 of instance: flexpwm2 0010b - Select mux mode: ALT2 mux port: QTIMER6_TIMER0 of instance: qtimer6 0011b - Select mux mode: ALT3 mux port: LPSPI4_SDI of instance: lpspi4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_RXD03 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO23 of instance: gpio2 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO23 of instance: flexio1 1010b - Select mux mode: ALT10 mux port: FLEXSPI2_BUS2BIT_B_DATA02 of instance: flexspi2_bus2bit 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR10 of instance: ahb_sramc

17.4.1.26 SW_MUX_CTL_PAD_GPIO_EMC_B1_24 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_24)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_24	70h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_24
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B1_24. 0000b - Select mux mode: ALT0 mux port: SEMC_CAS of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM1_PWMA00 of instance: flexpwm1 0010b - Select mux mode: ALT2 mux port: QTIMER7_TIMER0 of instance: qtimer7 0011b - Select mux mode: ALT3 mux port: LPSPI4_SDO of instance: lpspi4 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_TXD03 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO24 of instance: gpio2 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO24 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_ETH3_SLV_MDC of instance: netc 1010b - Select mux mode: ALT10 mux port: FLEXSPI2_BUS2BIT_B_DATA01 of instance: flexspi2_bus2bit 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR15 of instance: ahb_sramc

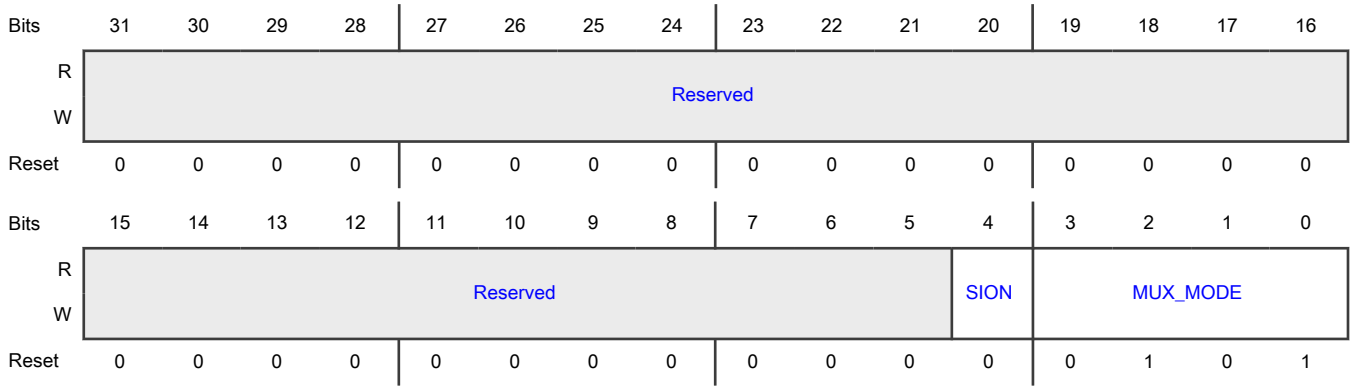
17.4.1.27 SW_MUX_CTL_PAD_GPIO_EMC_B1_25 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_25)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_25	74h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_25
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B1_25. 0000b - Select mux mode: ALT0 mux port: SEMC_RAS of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM1_PWMB00 of instance: flexpwm1 0010b - Select mux mode: ALT2 mux port: QTIMER8_TIMER0 of instance: qtimer8 0011b - Select mux mode: ALT3 mux port: LPSPI4_PCS0 of instance: lpspi4 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_TXD02 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO25 of instance: gpio2 1000b - Select mux mode: ALT8 mux port: FLEXIO1_FLEXIO25 of instance: flexio1 1001b - Select mux mode: ALT9 mux port: NETC_ETH3_SLV_MDIO of instance: netc 1010b - Select mux mode: ALT10 mux port: FLEXSPI2_BUS2BIT_B_DATA00 of instance: flexspi2_bus2bit 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADDR16 of instance: ahb_sramc

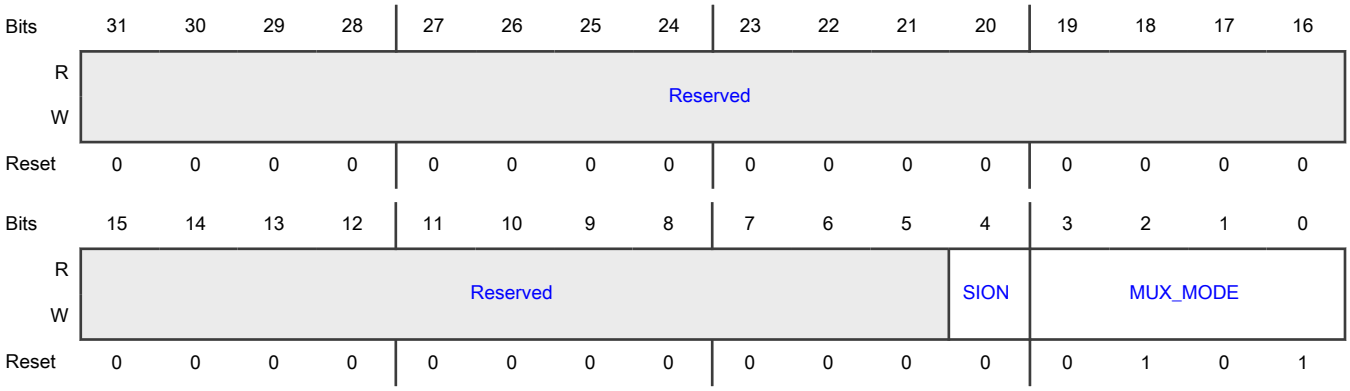
17.4.1.28 SW_MUX_CTL_PAD_GPIO_EMC_B1_26 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_26)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_26	78h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_26
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B1_26. 0000b - Select mux mode: ALT0 mux port: SEMC_CLK of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM1_PWMA01 of instance: flexpwm1 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT10 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_A_SS1_B of instance: flexspi2_bus2bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_TXD01 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO26 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: ECAT_TX_DATA1_1 of instance: ecat 1000b - Reserved 1010b - Select mux mode: ALT10 mux port: LPSPI6_SCK of instance: lpspi6 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_WE of instance: ahb_sramc

17.4.1.29 SW_MUX_CTL_PAD_GPIO_EMC_B1_27 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_27)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_27	7Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_27
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_27. 0000b - Select mux mode: ALT0 mux port: SEMC_CKE of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM1_PWMB01 of instance: flexpwm1 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT11 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_B_SS1_B of instance: flexspi2_bus2bit 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_TXD00 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO27 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: ECAT_TX_DATA0_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: LPUART6_RI_B of instance: lpuart6 1010b - Select mux mode: ALT10 mux port: LPSPI6_SDI of instance: lpspi6 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_OEB of instance: ahb_sramc

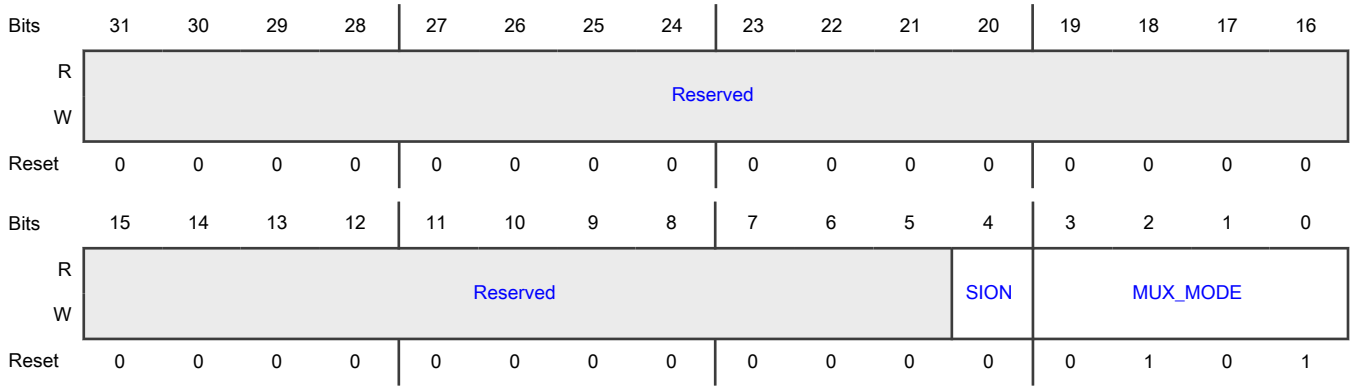
17.4.1.30 SW_MUX_CTL_PAD_GPIO_EMC_B1_28 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_28)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_28	80h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_28
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_28. 0000b - Select mux mode: ALT0 mux port: SEMC_WE of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM1_PWMB02 of instance: flexpwm1 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT12 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_B_SS0_B of instance: flexspi2_bus2bit 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_TX_EN of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO28 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: ECAT_TX_EN_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: LPUART6_DTR_B of instance: lpuart6 1010b - Select mux mode: ALT10 mux port: LPSPI6_SDO of instance: lpspi6 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_ADV of instance: ahb_sramc

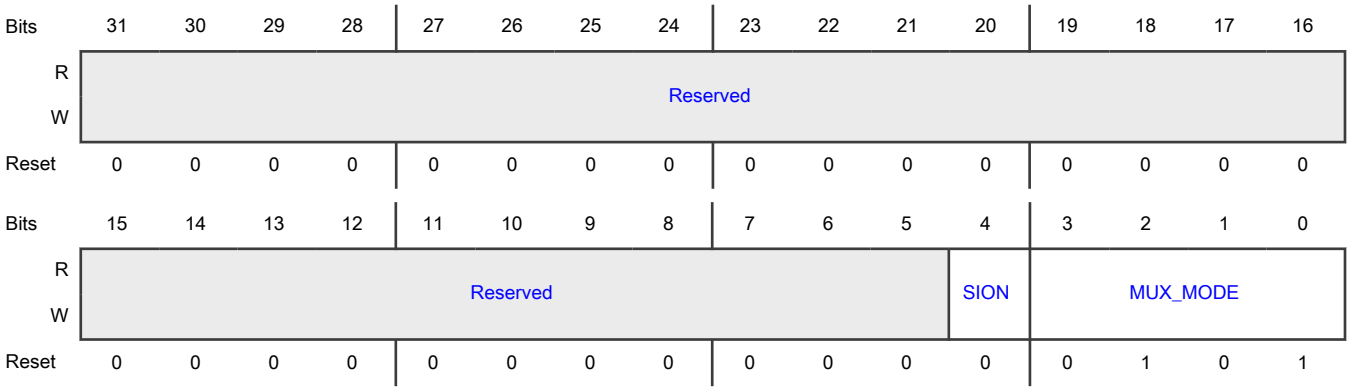
17.4.1.31 SW_MUX_CTL_PAD_GPIO_EMC_B1_29 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_29)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_29	84h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_29
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_29. 0000b - Select mux mode: ALT0 mux port: SEMC_CS0 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM1_PWMA02 of instance: flexpwm1 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT13 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_B_DQS of instance: flexspi2_bus2bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_TX_CLK of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO29 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: ECAT_TX_CLK_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: LPUART6_DCD_B of instance: lpuart6 1010b - Select mux mode: ALT10 mux port: LPSPI6_PCS0 of instance: lpspi6 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_CS0 of instance: ahb_sramc

17.4.1.32 SW_MUX_CTL_PAD_GPIO_EMC_B1_30 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_30)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_30	88h

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_30
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_30. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA08 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM3_PWMA00 of instance: flexpwm3 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT14 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_B_DATA03 of instance: flexspi2_bus2bit 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_RXD00 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO30 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: ECAT_RX_DATA0_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: LPUART6_DSR_B of instance: lpuart6 1010b - Select mux mode: ALT10 mux port: LPSPI6_PCS1 of instance: lpspi6 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA08 of instance: ahb_sramc

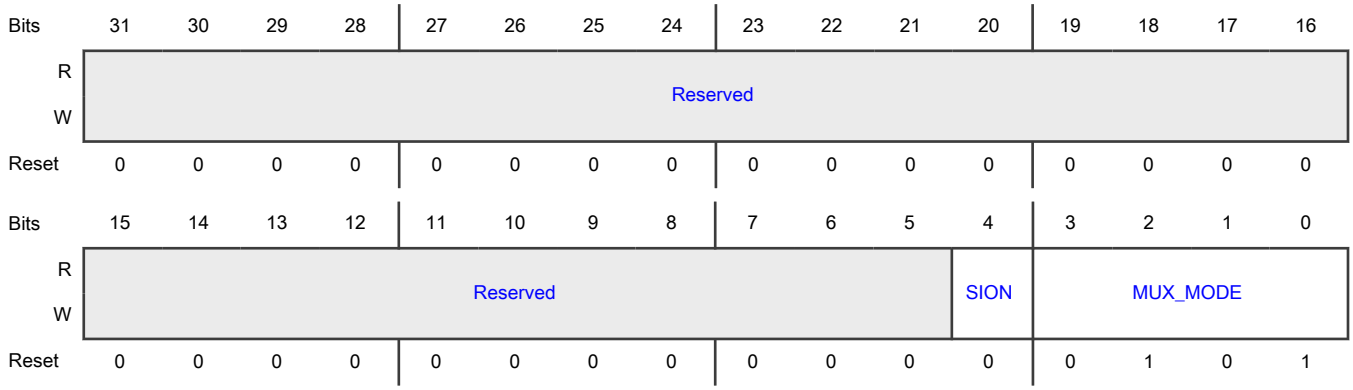
17.4.1.33 SW_MUX_CTL_PAD_GPIO_EMC_B1_31 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_31)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_31	8Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_31
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_31. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA09 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM3_PWMB00 of instance: flexpwm3 0010b - Select mux mode: ALT2 mux port: LPUART6_TX of instance: lpuart6 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_B_DATA02 of instance: flexspi2_bus2bit 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_RXD01 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO2_IO31 of instance: gpio2 0110b - Select mux mode: ALT6 mux port: ECAT_RX_DATA1_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: LPSPI5_SCK of instance: lpspi5 1010b - Select mux mode: ALT10 mux port: LPSPI6_PCS2 of instance: lpspi6 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA09 of instance: ahb_sramc

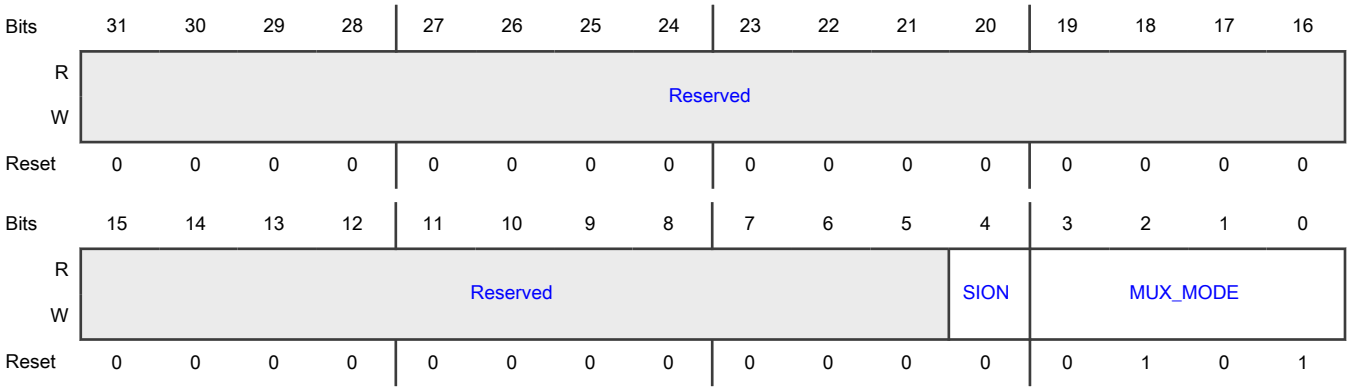
17.4.1.34 SW_MUX_CTL_PAD_GPIO_EMC_B1_32 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_32)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_32	90h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_32
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_32. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA10 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM3_PWMA01 of instance: flexpwm3 0010b - Select mux mode: ALT2 mux port: LPUART6_RX of instance: lpuart6 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_B_DATA01 of instance: flexspi2_bus2bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_RX_DV of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO00 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: ECAT_RX_DV_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: LPSPI5_SDO of instance: lpspi5 1010b - Select mux mode: ALT10 mux port: LPSPI6_PCS3 of instance: lpspi6 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA10 of instance: ahb_sramc

17.4.1.35 SW_MUX_CTL_PAD_GPIO_EMC_B1_33 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_33)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_33	94h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_33
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_33. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA11 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM3_PWMB01 of instance: flexpwm3 0010b - Select mux mode: ALT2 mux port: LPUART6_CTS_B of instance: lpuart6 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_B_DATA00 of instance: flexspi2_bus2bit 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_RX_ER of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO01 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: ECAT_RX_ER_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: LPSPI5_SDI of instance: lpspi5 1010b - Select mux mode: ALT10 mux port: NETC_PINMUX_ETH2_RX_CLK of instance: netc_pinmux 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA11 of instance: ahb_sramc

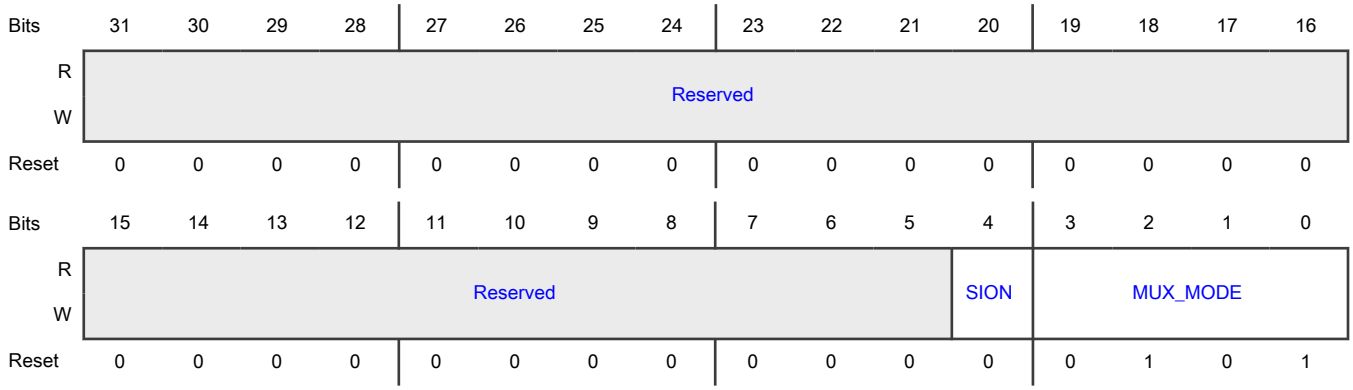
17.4.1.36 SW_MUX_CTL_PAD_GPIO_EMC_B1_34 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_34)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_34	98h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_34
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_34. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA12 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM3_PWMB02 of instance: flexpwm3 0010b - Select mux mode: ALT2 mux port: LPUART6_RTS_B of instance: lpuart6 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_B_SCLK of instance: flexspi2_bus2bit 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_RXD02 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO02 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: ECAT_RX_DATA2_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH0_TXD00 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: LPSPI5_PCS0 of instance: lpspi5 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA12 of instance: ahb_sramc

17.4.1.37 SW_MUX_CTL_PAD_GPIO_EMC_B1_35 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_35)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_35	9Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_35
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_35. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA13 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM3_PWMA02 of instance: flexpwm3 0010b - Select mux mode: ALT2 mux port: LPUART5_TX of instance: lpuart5 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_A_DATA00 of instance: flexspi2_bus2bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_RXD03 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO03 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: ECAT_RX_DATA3_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH0_TXD01 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: LPSPI5_PCS1 of instance: lpspi5 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA13 of instance: ahb_sramc

17.4.1.38 SW_MUX_CTL_PAD_GPIO_EMC_B1_36 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_36)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_36	A0h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_36
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_36. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA14 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM1_PWMA00 of instance: flexpwm1 0010b - Select mux mode: ALT2 mux port: LPUART5_RX of instance: lpuart5 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_A_DATA01 of instance: flexspi2_bus2bit 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_TXD03 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO04 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: ECAT_TX_DATA3_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH0_TX_EN of instance: netc_pinmux 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA14 of instance: ahb_sramc

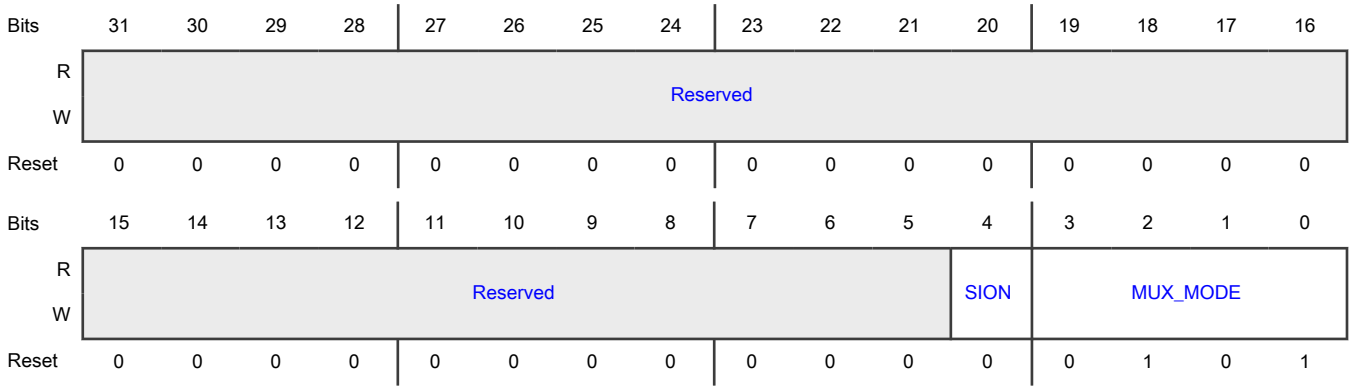
17.4.1.39 SW_MUX_CTL_PAD_GPIO_EMC_B1_37 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_37)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_37	A4h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_37
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_37. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA15 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM1_PWMB00 of instance: flexpwm1 0010b - Select mux mode: ALT2 mux port: LPUART5_CTS_B of instance: lpuart5 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_A_DATA02 of instance: flexspi2_bus2bit 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_TXD02 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO05 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: ECAT_TX_DATA2_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH0_TX_CLK of instance: netc_pinmux 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_DATA15 of instance: ahb_sramc

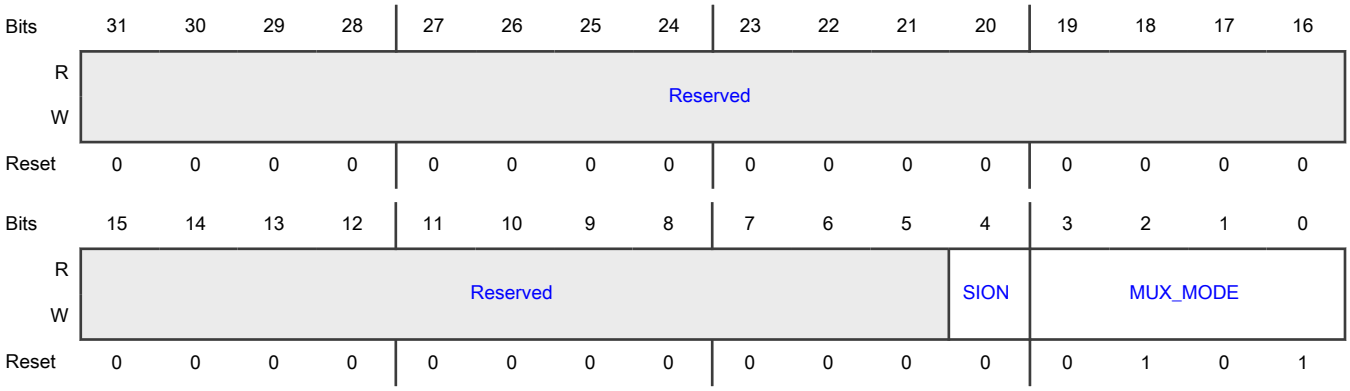
17.4.1.40 SW_MUX_CTL_PAD_GPIO_EMC_B1_38 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_38)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_38	A8h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_38
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_38. 0000b - Select mux mode: ALT0 mux port: SEMC_DM01 of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM1_PWMB03 of instance: flexpwm1 0010b - Select mux mode: ALT2 mux port: LPUART5_RTS_B of instance: lpuart5 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_A_DATA03 of instance: flexspi2_bus2bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_RX_CLK of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO06 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: ECAT_RX_CLK_1 of instance: ecat 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH0_RXD00 of instance: netc_pinmux 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_UBB of instance: ahb_sramc

17.4.1.41 SW_MUX_CTL_PAD_GPIO_EMC_B1_39 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_39)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_39	ACh

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_39
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_39. 0000b - Select mux mode: ALT0 mux port: SEMC_DQS of instance: semc 0001b - Select mux mode: ALT1 mux port: FLEXPWM1_PWMA03 of instance: flexpwm1 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT15 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_A_SS0_B of instance: flexspi2_bus2bit 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH2_TX_ER of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO07 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: QTIMER2_TIMER1 of instance: qtimer2 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH0_RXD01 of instance: netc_pinmux 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_CS1 of instance: ahb_sramc

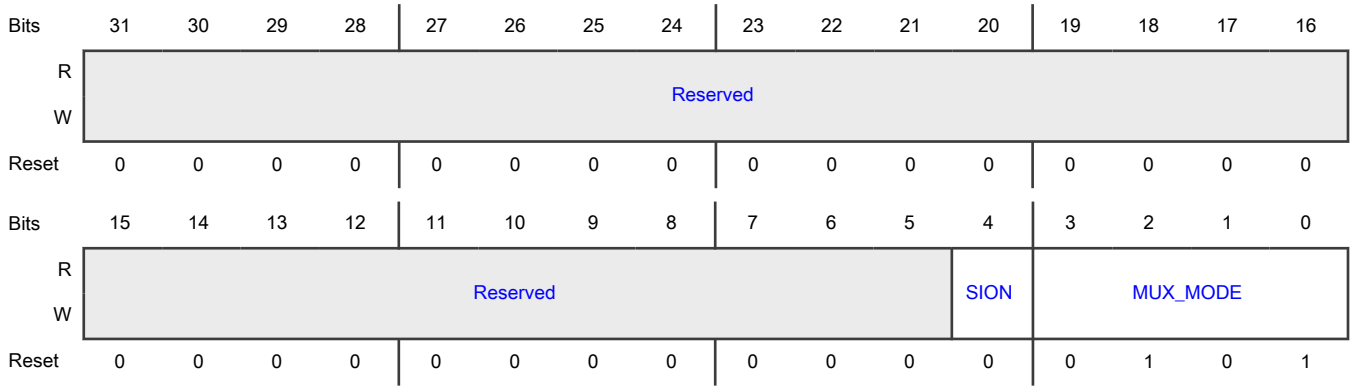
17.4.1.42 SW_MUX_CTL_PAD_GPIO_EMC_B1_40 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_40)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_40	B0h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_40
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_40. 0000b - Select mux mode: ALT0 mux port: SEMC_RDY of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_EMDC of instance: netc 0010b - Select mux mode: ALT2 mux port: NETC_ETH2_SLV_MDC of instance: netc 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_A_DQS of instance: flexspi2_bus2bit 0100b - Select mux mode: ALT4 mux port: NETC_ETH2_CRS of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO3_IO08 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: QTIMER3_TIMER1 of instance: qtimer3 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH0_RX_DV of instance: netc_pinmux 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_CS2 of instance: ahb_sramc

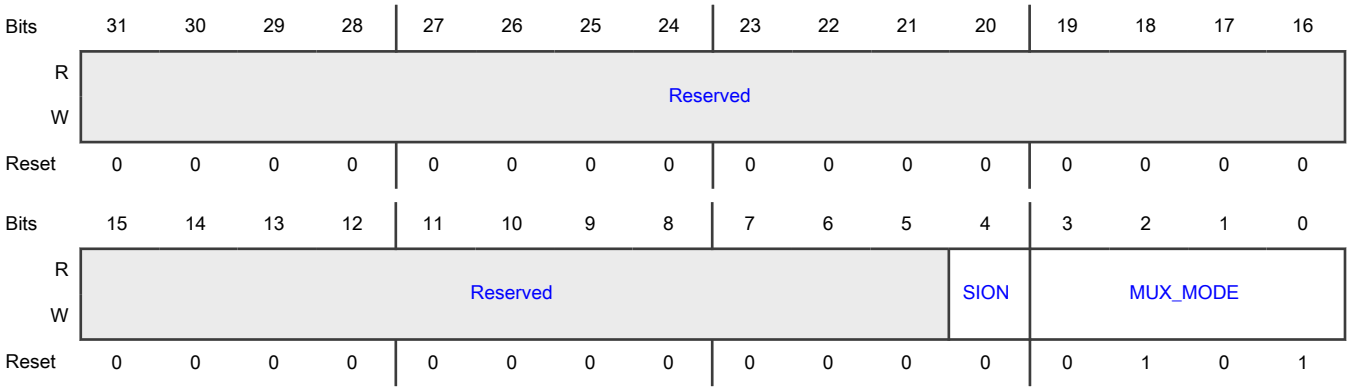
17.4.1.43 SW_MUX_CTL_PAD_GPIO_EMC_B1_41 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B1_41)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B1_41	B4h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B1_41
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B1_41. 0000b - Select mux mode: ALT0 mux port: SEMC_CSX00 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_EMDIO of instance: netc 0010b - Select mux mode: ALT2 mux port: NETC_ETH2_SLV_MDIO of instance: netc 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_A_SCLK of instance: flexspi2_bus2bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_ETH2_COL of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO3_IO09 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: QTIMER4_TIMER1 of instance: qtimer4 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH0_RX_ER of instance: netc_pinmux 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: AHB_SRAMC_CS3 of instance: ahb_sramc

17.4.1.44 SW_MUX_CTL_PAD_GPIO_EMC_B2_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_00)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_00	B8h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_00
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_00. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA16 of instance: semc 0001b - Select mux mode: ALT1 mux port: CCM_ENET_REF_CLK_25M of instance: ccm 0010b - Select mux mode: ALT2 mux port: QTIMER5_TIMER1 of instance: qtimer5 0011b - Select mux mode: ALT3 mux port: NETC_EMDC of instance: netc 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH0_RX_CLK of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO10 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT20 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPSPI5_SCK of instance: lpspi5 1001b - Select mux mode: ALT9 mux port: LPI2C3_SCL of instance: lpi2c3 1010b - Select mux mode: ALT10 mux port: FLEXPWM3_PWMA00 of instance: flexpwm3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_RX_CLK_0 of instance: ecat

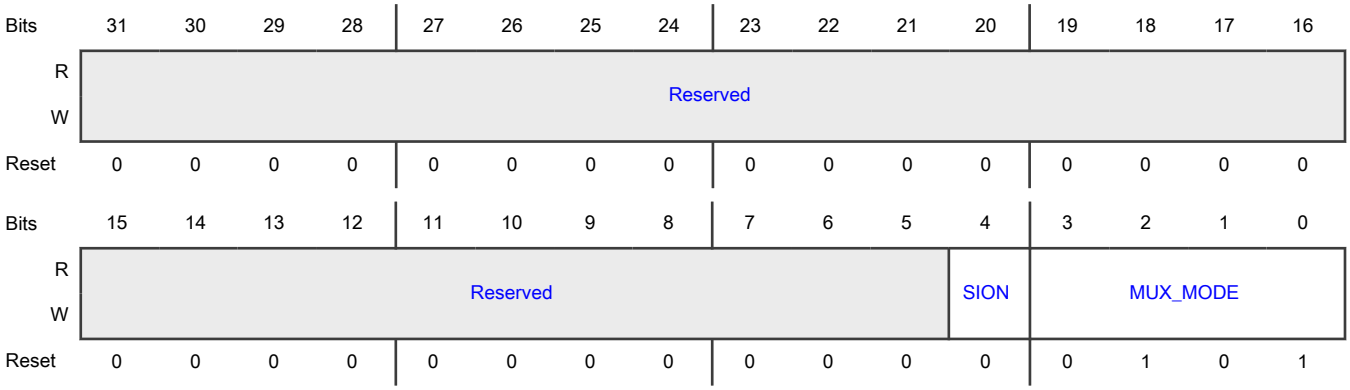
17.4.1.45 SW_MUX_CTL_PAD_GPIO_EMC_B2_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_01)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_01	BCh

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_01
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_01. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA17 of instance: semc 0001b - Select mux mode: ALT1 mux port: USDHC2_CD_B of instance: usdhc2 0010b - Select mux mode: ALT2 mux port: QTIMER6_TIMER1 of instance: qtimer6 0011b - Select mux mode: ALT3 mux port: NETC_EMDIO of instance: netc 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH0_RXD02 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO11 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT21 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPSPI5_PCS0 of instance: lpspi5 1001b - Select mux mode: ALT9 mux port: LPI2C3_SDA of instance: lpi2c3 1010b - Select mux mode: ALT10 mux port: FLEXPWM3_PWMB00 of instance: flexpwm3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DATA2_0 of instance: ecat

17.4.1.46 SW_MUX_CTL_PAD_GPIO_EMC_B2_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_02)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_02	C0h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_02
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B2_02. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA18 of instance: semc 0001b - Select mux mode: ALT1 mux port: USDHC2_WP of instance: usdhc2 0010b - Select mux mode: ALT2 mux port: QTIMER7_TIMER1 of instance: qtimer7 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH0_RXD03 of instance: netc_pinmux

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Select mux mode: ALT5 mux port: GPIO3_IO12 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT22 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPSPI5_SDO of instance: lpspi5 1001b - Select mux mode: ALT9 mux port: CCM_CLKO1 of instance: ccm 1010b - Select mux mode: ALT10 mux port: FLEXPWM3_PWMA01 of instance: flexpwm3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DATA3_0 of instance: ecat

17.4.1.47 SW_MUX_CTL_PAD_GPIO_EMC_B2_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_03)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_03	C4h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_03
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B2_03. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA19 of instance: semc 0001b - Select mux mode: ALT1 mux port: USDHC2_VSELECT of instance: usdhc2 0010b - Select mux mode: ALT2 mux port: QTIMER8_TIMER1 of instance: qtimer8 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH0_TXD02 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO13 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT23 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPSPI5_SDI of instance: lpspi5 1001b - Select mux mode: ALT9 mux port: NETC_ETH3_CRS of instance: netc 1010b - Select mux mode: ALT10 mux port: FLEXPWM3_PWMB01 of instance: flexpwm3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_TX_DATA2_0 of instance: ecat

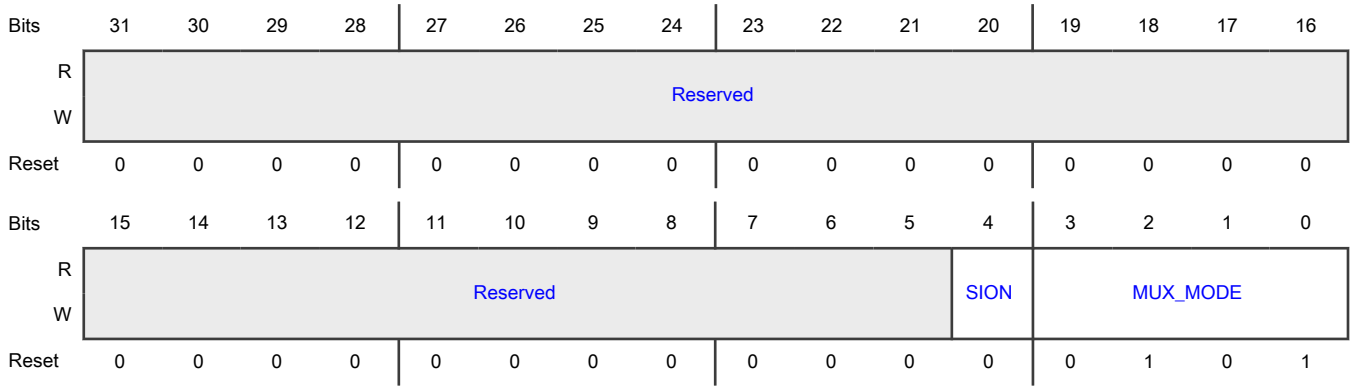
17.4.1.48 SW_MUX_CTL_PAD_GPIO_EMC_B2_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_04)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_04	C8h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_04
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B2_04. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA20 of instance: semc 0001b - Select mux mode: ALT1 mux port: USDHC2_RESET_B of instance: usdhc2 0010b - Select mux mode: ALT2 mux port: SAI2_MCLK of instance: sai2 0100b - Select mux mode: ALT4 mux port: NETC_PINMUX_ETH0_TXD03 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO14 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT24 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPSPI3_SCK of instance: lpspi3 1001b - Select mux mode: ALT9 mux port: NETC_ETH3_COL of instance: netc 1010b - Select mux mode: ALT10 mux port: FLEXPWM3_PWMB02 of instance: flexpwm3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_TX_DATA3_0 of instance: ecat

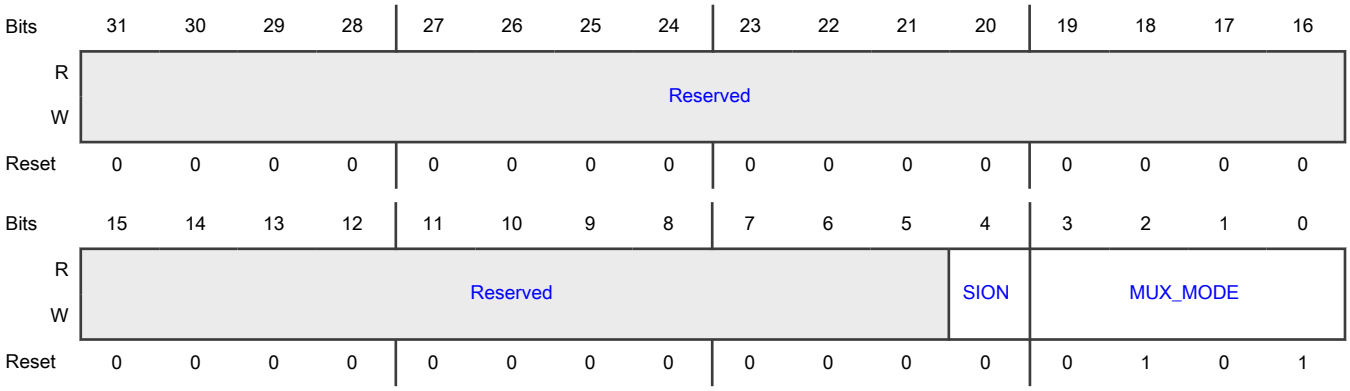
17.4.1.49 SW_MUX_CTL_PAD_GPIO_EMC_B2_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_05)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_05	CCh

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_05
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_05. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA21 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_ETH4_SLV_MDC of instance: netc 0010b - Select mux mode: ALT2 mux port: SAI2_RX_SYNC of instance: sai2 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_TXD00 of instance: netc_pinmux

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_ETH4_CRS of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO3_IO15 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT25 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPSPI3_PCS0 of instance: lpspi3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_TXD00 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: FLEXPWM3_PWMA02 of instance: flexpwm3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_TX_DATA0_0 of instance: ecat

17.4.1.50 SW_MUX_CTL_PAD_GPIO_EMC_B2_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_06)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_06	D0h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_06
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_06. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA22 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_ETH4_SLV_MDIO of instance: netc 0010b - Select mux mode: ALT2 mux port: SAI2_RX_BCLK of instance: sai2 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_TXD01 of instance: netc_pinmux 0100b - Select mux mode: ALT4 mux port: NETC_ETH4_COL of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO3_IO16 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT26 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPSPI3_SDO of instance: lpspi3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_TXD01 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: FLEXPWM3_PWMB03 of instance: flexpwm3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_TX_DATA1_0 of instance: ecat

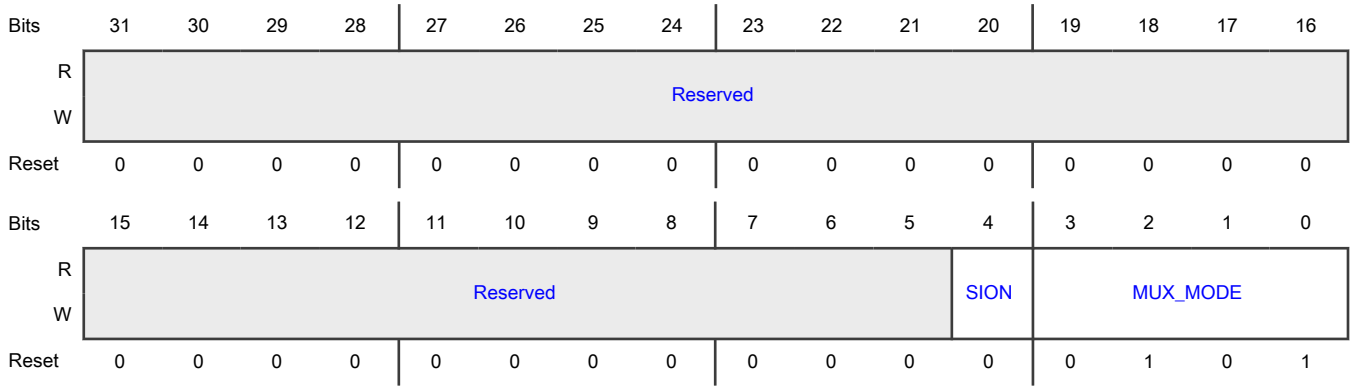
17.4.1.51 SW_MUX_CTL_PAD_GPIO_EMC_B2_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_07)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_07	D4h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_07
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B2_07. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA23 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_TX_ER of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: SAI2_RX_DATA of instance: sai2 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_TX_EN of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO17 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT27 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPSPI3_SDI of instance: lpspi3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_TX_EN of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: FLEXPWM3_PWMA03 of instance: flexpwm3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_TX_EN_0 of instance: ecat

17.4.1.52 SW_MUX_CTL_PAD_GPIO_EMC_B2_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_08)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_08	D8h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_08
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B2_08. 0000b - Select mux mode: ALT0 mux port: SEMC_DM02 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_RX_CLK of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: SAI2_TX_DATA of instance: sai2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_TX_CLK of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO18 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT28 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPSPI3_PCS3 of instance: lpspi3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_TX_CLK of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: CCM_CLKO2 of instance: ccm 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_TX_CLK_0 of instance: ecat

17.4.1.53 SW_MUX_CTL_PAD_GPIO_EMC_B2_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_09)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_09	DCh

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_09
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_EMC_B2_09. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA24 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_RXD03 of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: SAI2_TX_BCLK of instance: sai2 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_RXD00 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO19 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT29 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPSPI3_PCS2 of instance: lpspi3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_RXD00 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: QTIMER1_TIMER0 of instance: qtimer1 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DATA0_0 of instance: ecat

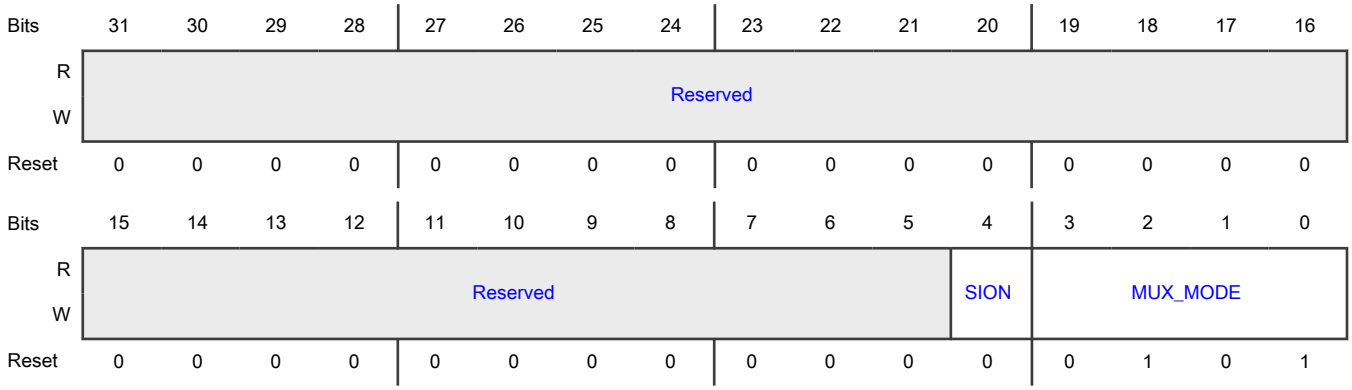
17.4.1.54 SW_MUX_CTL_PAD_GPIO_EMC_B2_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_10)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_10	E0h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_10
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B2_10. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA25 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_RXD02 of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: SAI2_TX_SYNC of instance: sai2 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_RXD01 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO20 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT30 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPSPI3_PCS1 of instance: lpspi3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_RXD01 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: QTIMER1_TIMER1 of instance: qtimer1 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DATA1_0 of instance: ecat

17.4.1.55 SW_MUX_CTL_PAD_GPIO_EMC_B2_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_11)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_11	E4h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_11
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_11. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA26 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_TXD03 of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: SPDIF_OUT of instance: spdif

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_RX_DV of instance: netc_pinmux 0100b - Select mux mode: ALT4 mux port: LPSPI5_PCS3 of instance: lpspi5 0101b - Select mux mode: ALT5 mux port: GPIO3_IO21 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT31 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: SAI3_RX_SYNC of instance: sai3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_RX_DV of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: QTIMER1_TIMER2 of instance: qtimer1 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DV_0 of instance: ecat

17.4.1.56 SW_MUX_CTL_PAD_GPIO_EMC_B2_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_12)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_12	E8h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_12
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_12. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA27 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_TXD02 of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: SPDIF_IN of instance: spdif 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_RX_ER of instance: netc_pinmux 0100b - Select mux mode: ALT4 mux port: LPSPI5_PCS2 of instance: lpspi5 0101b - Select mux mode: ALT5 mux port: GPIO3_IO22 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT32 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: SAI3_RX_BCLK of instance: sai3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_RX_ER of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: QTIMER1_TIMER3 of instance: qtimer1 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_PT0_RX_ER of instance: ecat

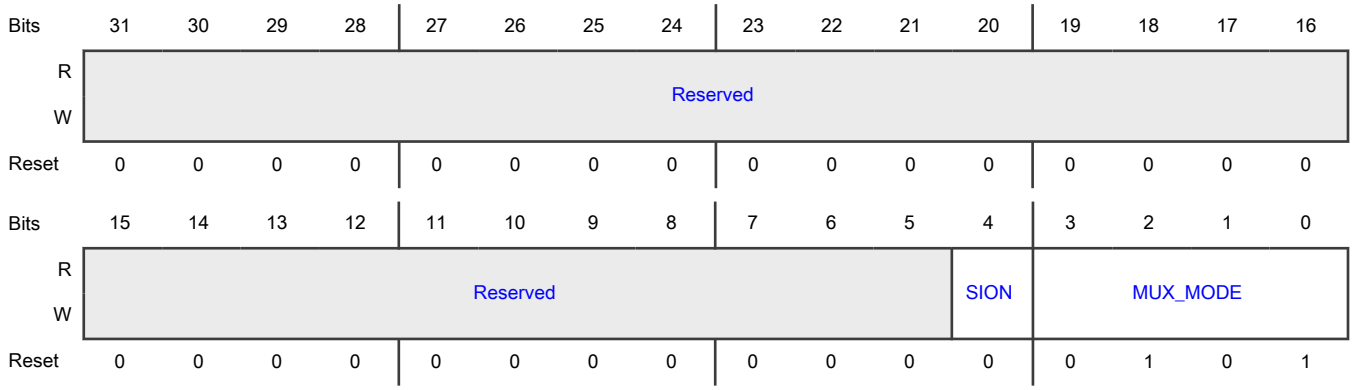
17.4.1.57 SW_MUX_CTL_PAD_GPIO_EMC_B2_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_13)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_13	ECh

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_13
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_13. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA28 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_TXD00 of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: LPUART11_TX of instance: lpuart11 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_TXD03 of instance: netc_pinmux 0100b - Select mux mode: ALT4 mux port: LPSPI5_PCS1 of instance: lpspi5 0101b - Select mux mode: ALT5 mux port: GPIO3_IO23 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT33 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: SAI3_RX_DATA of instance: sai3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_TXD03 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: QTIMER2_TIMER0 of instance: qtimer2 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_TX_DATA0_1 of instance: ecat

17.4.1.58 SW_MUX_CTL_PAD_GPIO_EMC_B2_14 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_14)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_14	F0h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_14
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_14. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA29 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_TXD01 of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: LPUART11_RX of instance: lpuart11

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_TXD02 of instance: netc_pinmux 0100b - Select mux mode: ALT4 mux port: LPUART5_DSR_B of instance: lpuart5 0101b - Select mux mode: ALT5 mux port: GPIO3_IO24 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT34 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: SAI3_TX_DATA of instance: sai3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_TXD02 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: QTIMER2_TIMER1 of instance: qtimer2 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_TX_DATA1_1 of instance: ecat

17.4.1.59 SW_MUX_CTL_PAD_GPIO_EMC_B2_15 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_15)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_15	F4h

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_15
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_15. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA30 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_TX_EN of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: LPUART11_CTS_B of instance: lpuart11 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_RX_CLK of instance: netc_pinmux 0100b - Select mux mode: ALT4 mux port: LPUART5_DCD_B of instance: lpuart5 0101b - Select mux mode: ALT5 mux port: GPIO3_IO25 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT35 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: SAI3_TX_BCLK of instance: sai3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_RX_CLK of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: QTIMER2_TIMER2 of instance: qtimer2 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_TX_EN_1 of instance: ecat

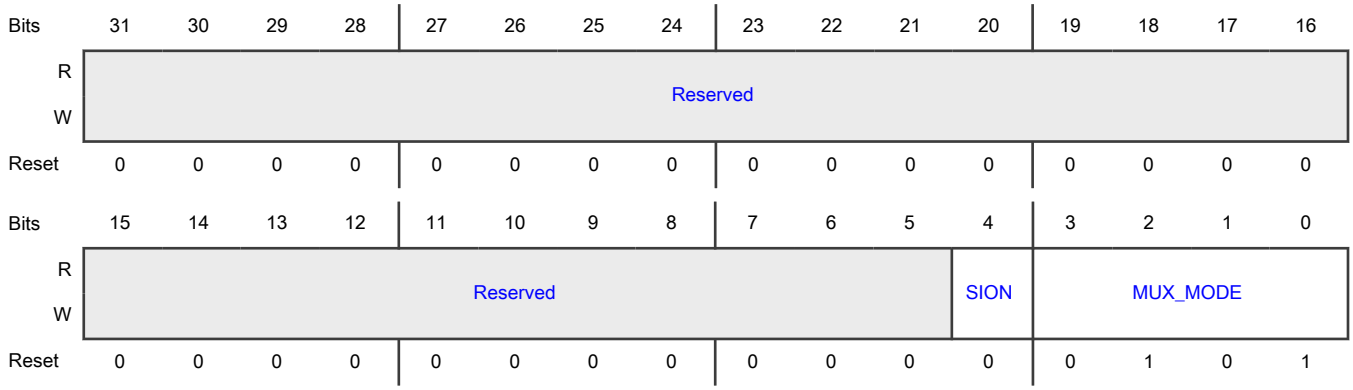
17.4.1.60 SW_MUX_CTL_PAD_GPIO_EMC_B2_16 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_16)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_16	F8h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_16
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_16. 0000b - Select mux mode: ALT0 mux port: SEMC_DATA31 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_TX_CLK of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: LPUART11_RTS_B of instance: lpuart11 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_RXD02 of instance: netc_pinmux 0100b - Select mux mode: ALT4 mux port: LPUART5_DTR_B of instance: lpuart5 0101b - Select mux mode: ALT5 mux port: GPIO3_IO26 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT14 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: SAI3_TX_SYNC of instance: sai3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_RXD02 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: QTIMER2_TIMER3 of instance: qtimer2 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_TX_CLK_1 of instance: ecat

17.4.1.61 SW_MUX_CTL_PAD_GPIO_EMC_B2_17 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_17)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_17	FCh

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_17
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B2_17. 0000b - Select mux mode: ALT0 mux port: SEMC_DM03 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_RXD00 of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: LPUART5_TX of instance: lpuart5

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_RXD03 of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO27 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT15 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: SAI3_MCLK of instance: sai3 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_RXD03 of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: QTIMER3_TIMER0 of instance: qtimer3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DATA0_1 of instance: ecat

17.4.1.62 SW_MUX_CTL_PAD_GPIO_EMC_B2_18 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_18)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_18	100h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_18
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_EMC_B2_18. 0000b - Select mux mode: ALT0 mux port: SEMC_DQS4 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_RXD01 of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: LPUART5_RX of instance: lpuart5 0011b - Select mux mode: ALT3 mux port: NETC_PINMUX_ETH0_TX_ER of instance: netc_pinmux 0101b - Select mux mode: ALT5 mux port: GPIO3_IO28 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT16 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: EWM_EWM_OUT_B of instance: ewm 1001b - Select mux mode: ALT9 mux port: NETC_PINMUX_ETH3_TX_ER of instance: netc_pinmux 1010b - Select mux mode: ALT10 mux port: QTIMER3_TIMER1 of instance: qtimer3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DATA1_1 of instance: ecat

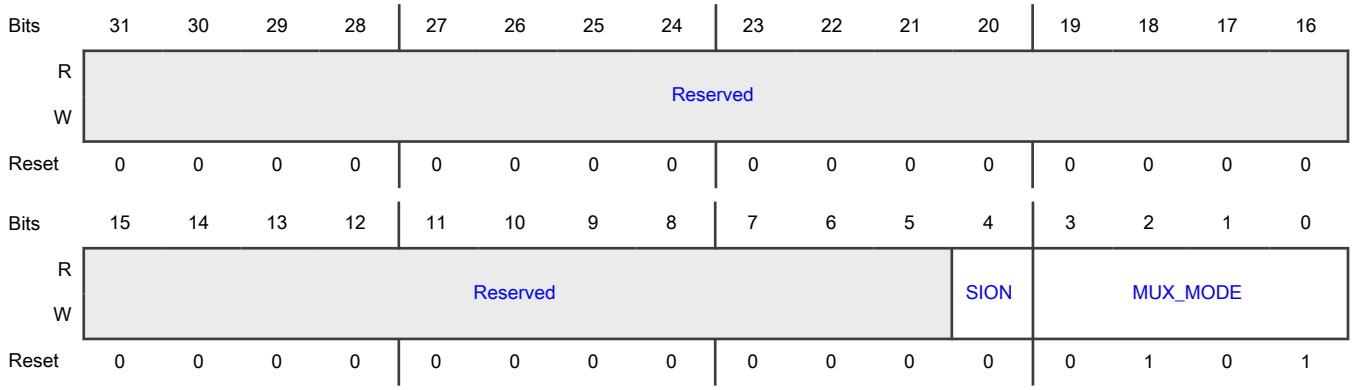
17.4.1.63 SW_MUX_CTL_PAD_GPIO_EMC_B2_19 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_19)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_19	104h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_19
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_19. 0000b - Select mux mode: ALT0 mux port: SEMC_CLKX00 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_RX_DV of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: LPUART5_CTS_B of instance: lpuart5 0011b - Select mux mode: ALT3 mux port: NETC_ETH0_CRS of instance: netc 0100b - Select mux mode: ALT4 mux port: NETC_EMDC of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO3_IO29 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT36 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPI2C3_SCL of instance: lpi2c3 1001b - Select mux mode: ALT9 mux port: NETC_ETH3_SLV_MDC of instance: netc 1010b - Select mux mode: ALT10 mux port: QTIMER3_TIMER2 of instance: qtimer3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DV_1 of instance: ecat

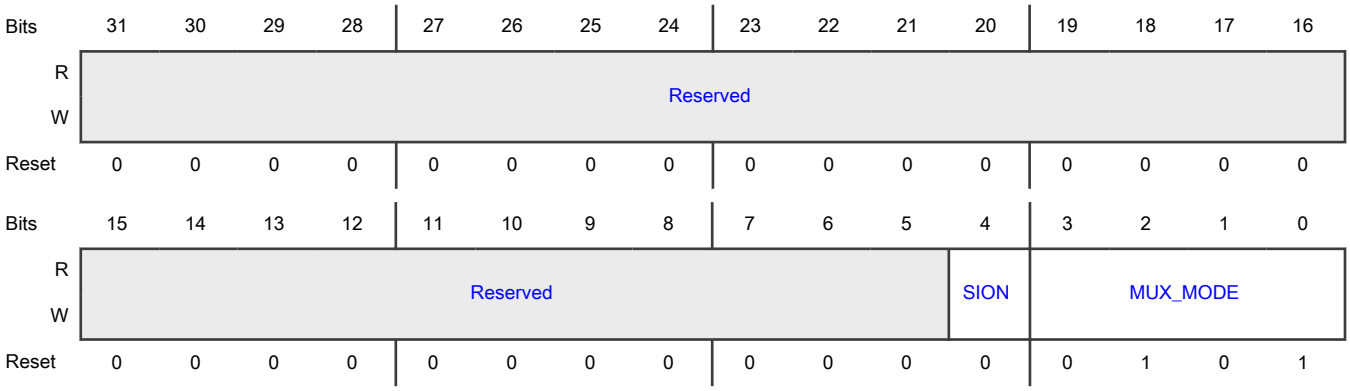
17.4.1.64 SW_MUX_CTL_PAD_GPIO_EMC_B2_20 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_EMC_B2_20)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_EMC_B2_20	108h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_EMC_B2_20
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_EMC_B2_20. 0000b - Select mux mode: ALT0 mux port: SEMC_CLKX01 of instance: semc 0001b - Select mux mode: ALT1 mux port: NETC_PINMUX_ETH4_RX_ER of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: LPUART5_RTS_B of instance: lpuart5 0011b - Select mux mode: ALT3 mux port: NETC_ETH0_COL of instance: netc

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: NETC_EMDIO of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO3_IO30 of instance: gpio3 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT37 of instance: xbar1 1000b - Select mux mode: ALT8 mux port: LPI2C3_SDA of instance: lpi2c3 1001b - Select mux mode: ALT9 mux port: NETC_ETH3_SLV_MDIO of instance: netc 1010b - Select mux mode: ALT10 mux port: QTIMER3_TIMER3 of instance: qtimer3 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_RX_ER_1 of instance: ecat

17.4.1.65 SW_MUX_CTL_PAD_GPIO_AD_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_00)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_00	10Ch

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_00
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_AD_00. 0001b - Select mux mode: ALT1 mux port: CAN2_TX of instance: can2 0010b - Select mux mode: ALT2 mux port: MIC_CLK of instance: mic 0011b - Select mux mode: ALT3 mux port: GPT2_CAPTURE1 of instance: gpt2 0100b - Select mux mode: ALT4 mux port: FLEXPWM1_PWMA00 of instance: flexpwm1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO00 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC1_MOD_CLK0 of instance: sinc1 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO00 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER4_TIMER0 of instance: qtimer4 1010b - Reserved 1011b - Reserved

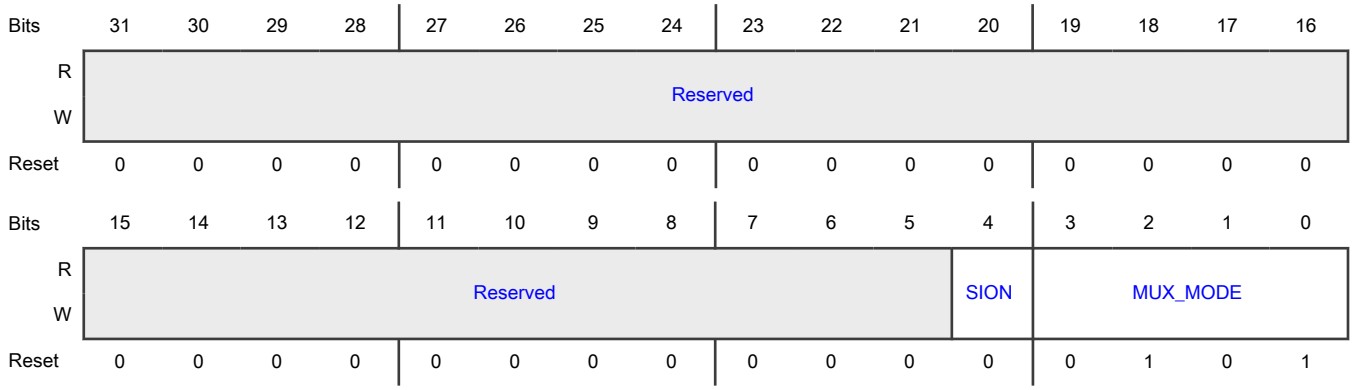
17.4.1.66 SW_MUX_CTL_PAD_GPIO_AD_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_01)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_01	110h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_01
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_AD_01. 0001b - Select mux mode: ALT1 mux port: CAN2_RX of instance: can2 0010b - Select mux mode: ALT2 mux port: MIC_BITSTREAM00 of instance: mic 0011b - Select mux mode: ALT3 mux port: GPT2_CAPTURE2 of instance: gpt2 0100b - Select mux mode: ALT4 mux port: FLEXPWM1_PWMB00 of instance: flexpwm1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO01 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC1_MOD_CLK1 of instance: sinc1 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO01 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER4_TIMER1 of instance: qtimer4 1010b - Reserved 1011b - Reserved

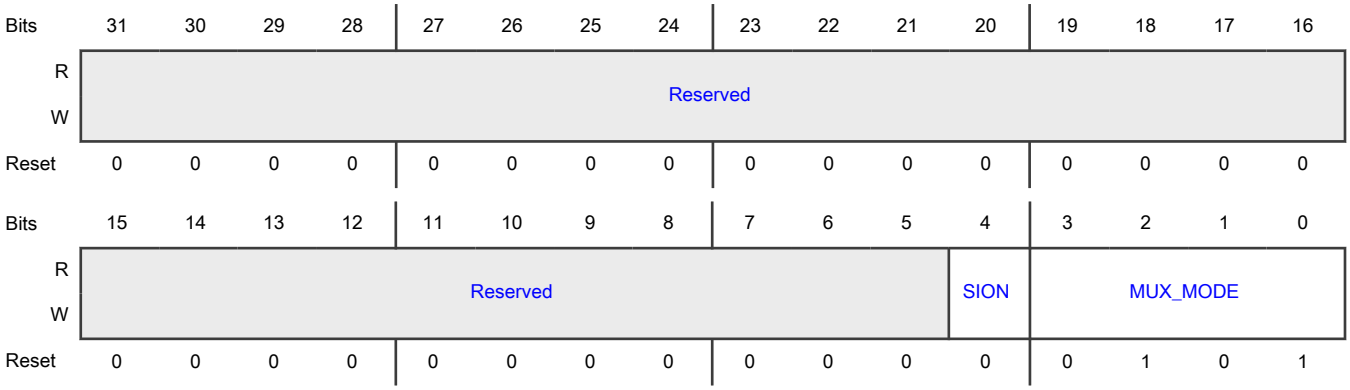
17.4.1.67 SW_MUX_CTL_PAD_GPIO_AD_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_02)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_02	114h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_02
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AD_02. 0010b - Select mux mode: ALT2 mux port: MIC_BITSTREAM01 of instance: mic 0011b - Select mux mode: ALT3 mux port: GPT2_COMPARE1 of instance: gpt2 0100b - Select mux mode: ALT4 mux port: FLEXPWM1_PWMA01 of instance: flexpwm1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO02 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC1_MOD_CLK2 of instance: sinc1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO02 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER4_TIMER2 of instance: qtimer4 1010b - Reserved 1011b - Reserved

17.4.1.68 SW_MUX_CTL_PAD_GPIO_AD_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_03)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_03	118h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_03

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AD_03. 0010b - Select mux mode: ALT2 mux port: MIC_BITSTREAM02 of instance: mic 0011b - Select mux mode: ALT3 mux port: GPT2_COMPARE2 of instance: gpt2 0100b - Select mux mode: ALT4 mux port: FLEXPWM1_PWMB01 of instance: flexpwm1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO03 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC1_EMCLK00 of instance: sinc1 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO03 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER4_TIMER3 of instance: qtimer4 1010b - Reserved 1011b - Reserved

17.4.1.69 SW_MUX_CTL_PAD_GPIO_AD_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_04)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_04	11Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_04
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AD_04. 0010b - Select mux mode: ALT2 mux port: MIC_BITSTREAM03 of instance: mic 0011b - Select mux mode: ALT3 mux port: GPT2_COMPARE3 of instance: gpt2 0100b - Select mux mode: ALT4 mux port: FLEXPWM1_PWMB02 of instance: flexpwm1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO04 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC1_EMBIT00 of instance: sinc1 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO04 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER5_TIMER0 of instance: qtimer5 1010b - Reserved 1011b - Reserved

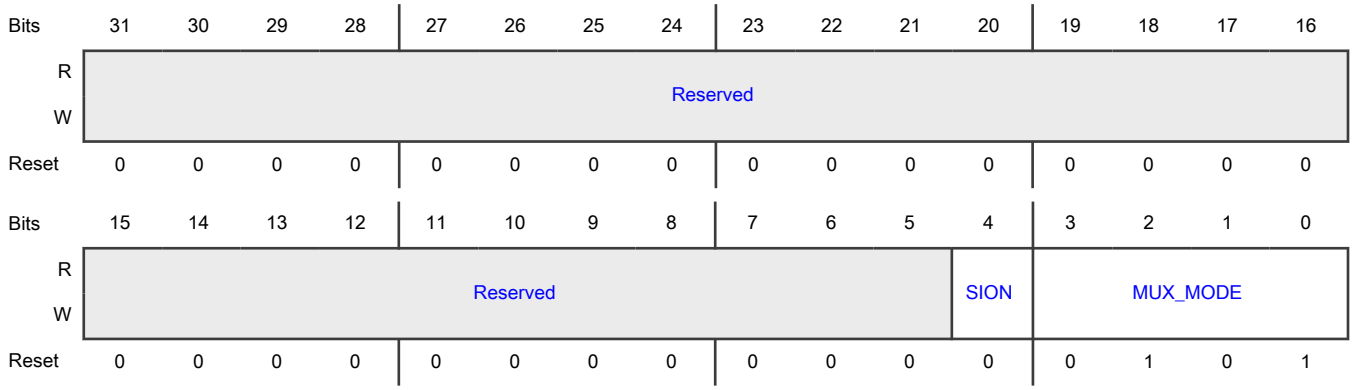
17.4.1.70 SW_MUX_CTL_PAD_GPIO_AD_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_05)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_05	120h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_05
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AD_05. 0011b - Select mux mode: ALT3 mux port: GPT2_CLK of instance: gpt2 0100b - Select mux mode: ALT4 mux port: FLEXPWM1_PWMA02 of instance: flexpwm1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO05 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC1_EMCLK01 of instance: sinc1 0111b - Select mux mode: ALT7 mux port: CCM_ENET_REF_CLK_25M of instance: ccm 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO05 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER5_TIMER1 of instance: qtimer5 1010b - Reserved 1011b - Reserved

17.4.1.71 SW_MUX_CTL_PAD_GPIO_AD_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_06)

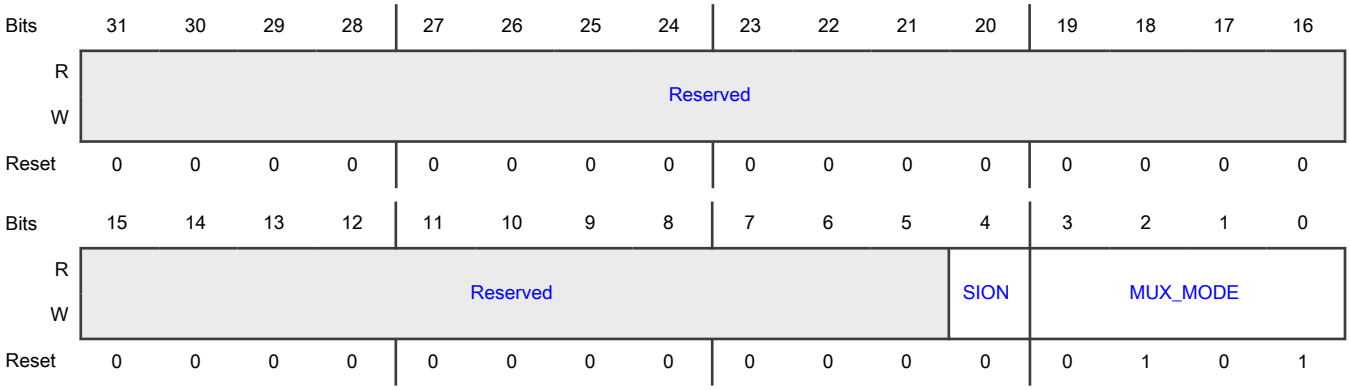
Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_06	124h

Function

SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_06
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AD_06. 0000b - Select mux mode: ALT0 mux port: USB_OTG2_OC of instance: usb 0001b - Select mux mode: ALT1 mux port: CAN3_TX of instance: can3 0100b - Select mux mode: ALT4 mux port: FLEXPWM1_PWMX00 of instance: flexpwm1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO06 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC1_EMBIT01 of instance: sinc1 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO06 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER5_TIMER2 of instance: qtimer5 1010b - Reserved 1011b - Reserved

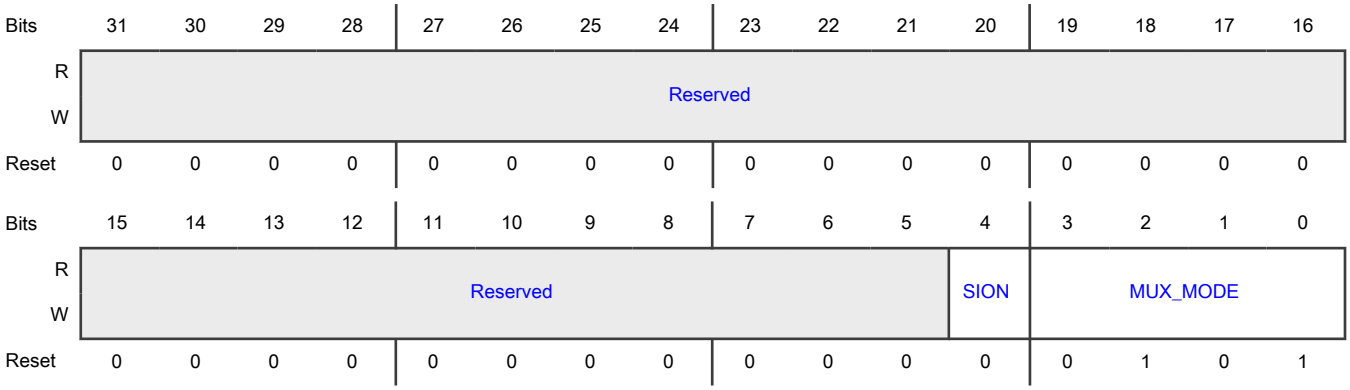
17.4.1.72 SW_MUX_CTL_PAD_GPIO_AD_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_07)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_07	128h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_07
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AD_07. 0000b - Select mux mode: ALT0 mux port: USB_OTG2_PWR of instance: usb 0001b - Select mux mode: ALT1 mux port: CAN3_RX of instance: can3 0100b - Select mux mode: ALT4 mux port: FLEXPWM1_PWMX01 of instance: flexpwm1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO07 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC1_EMCLK02 of instance: sinc1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO07 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER5_TIMER3 of instance: qtimer5 1010b - Reserved 1011b - Reserved

17.4.1.73 SW_MUX_CTL_PAD_GPIO_AD_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_08)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_08	12Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_08

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AD_08. 0000b - Select mux mode: ALT0 mux port: USBPHY2_OTG_ID of instance: usbphy2 0001b - Select mux mode: ALT1 mux port: LPI2C5_SCL of instance: lpi2c5 0100b - Select mux mode: ALT4 mux port: FLEXPWM1_PWMX02 of instance: flexpwm1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO08 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC1_EMBIT02 of instance: sinc1 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO08 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER6_TIMER0 of instance: qtimer6 1010b - Reserved 1011b - Reserved

17.4.1.74 SW_MUX_CTL_PAD_GPIO_AD_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_09)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_09	130h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_09
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AD_09. 0000b - Select mux mode: ALT0 mux port: USBPHY1_OTG_ID of instance: usbphy1 0001b - Select mux mode: ALT1 mux port: LPI2C5_SDA of instance: lpi2c5 0100b - Select mux mode: ALT4 mux port: FLEXPWM1_PWMX03 of instance: flexpwm1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO09 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC1_EMCLK03 of instance: sinc1 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO09 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER6_TIMER1 of instance: qtimer6 1010b - Reserved 1011b - Reserved

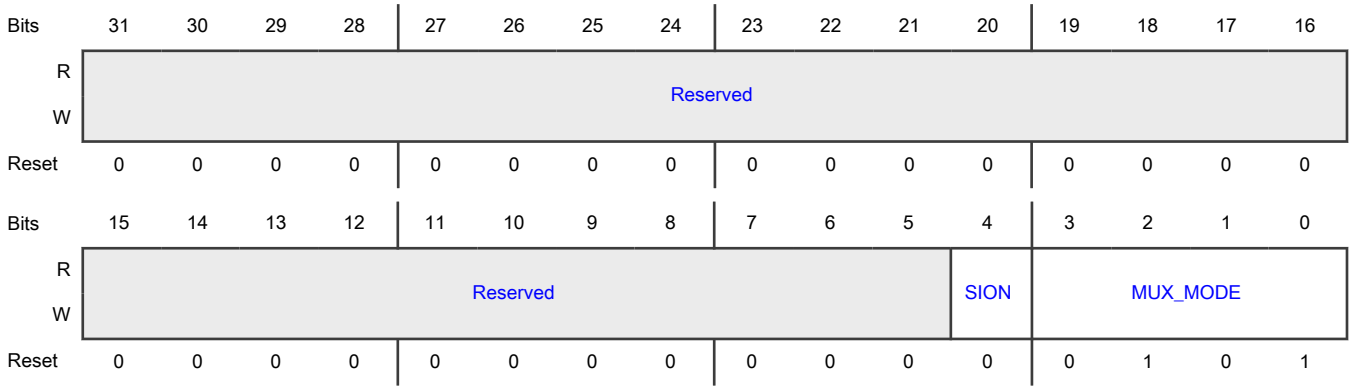
17.4.1.75 SW_MUX_CTL_PAD_GPIO_AD_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_10)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_10	134h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_10
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for pad: GPIO_AD_10. 0000b - Select mux mode: ALT0 mux port: USB_OTG1_PWR of instance: usb 0100b - Select mux mode: ALT4 mux port: FLEXPWM2_PWMX00 of instance: flexpwm2 0101b - Select mux mode: ALT5 mux port: GPIO4_IO10 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC1_EMBIT03 of instance: sinc1 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO10 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER6_TIMER2 of instance: qtimer6 1010b - Reserved 1011b - Reserved

17.4.1.76 SW_MUX_CTL_PAD_GPIO_AD_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_11)

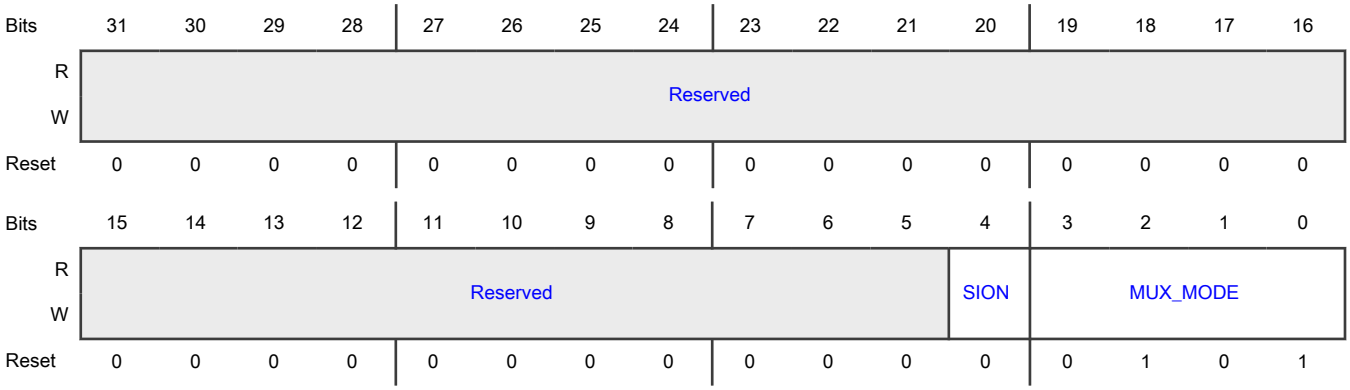
Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_11	138h

Function

SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_11
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for pad: GPIO_AD_11. 0000b - Select mux mode: ALT0 mux port: USB_OTG1_OC of instance: usb 0100b - Select mux mode: ALT4 mux port: FLEXPWM2_PWMX01 of instance: flexpwm2 0101b - Select mux mode: ALT5 mux port: GPIO4_IO11 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: SINC_FILTER_GLUE1_BREAK of instance: sinc_filter_glue1 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO11 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: QTIMER6_TIMER3 of instance: qtimer6 1010b - Reserved 1011b - Reserved

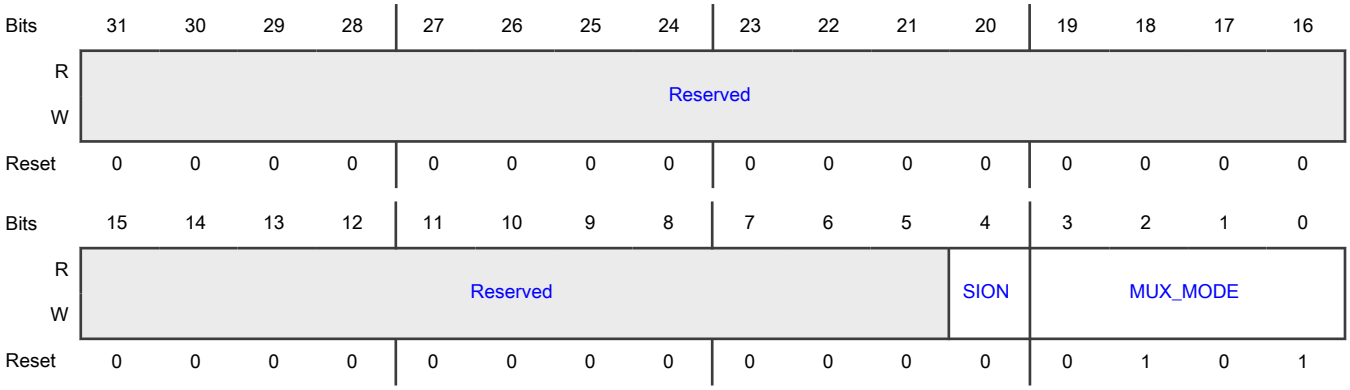
17.4.1.77 SW_MUX_CTL_PAD_GPIO_AD_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_12)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_12	13Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_12
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AD_12. 0000b - Select mux mode: ALT0 mux port: SPDIF_LOCK of instance: spdif 0001b - Select mux mode: ALT1 mux port: LPI2C5_SCLS of instance: lpi2c5 0010b - Select mux mode: ALT2 mux port: GPT1_CAPTURE1 of instance: gpt1 0011b - Select mux mode: ALT3 mux port: KPP_ROW07 of instance: kpp 0100b - Select mux mode: ALT4 mux port: FLEXPWM2_PWMX02 of instance: flexpwm2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Select mux mode: ALT5 mux port: GPIO4_IO12 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: XBAR1_XBAR_INOUT18 of instance: xbar1 0111b - Select mux mode: ALT7 mux port: EWM_EWM_OUT_B of instance: ewm 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO12 of instance: flexio2 1001b - Reserved 1010b - Reserved 1011b - Reserved

17.4.1.78 SW_MUX_CTL_PAD_GPIO_AD_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_13)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_13	140h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4	Software Input On Field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SION	Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_13
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AD_13. 0000b - Select mux mode: ALT0 mux port: SPDIF_SR_CLK of instance: spdif 0001b - Select mux mode: ALT1 mux port: LPI2C5_SDAS of instance: lpi2c5 0010b - Select mux mode: ALT2 mux port: GPT1_CAPTURE2 of instance: gpt1 0011b - Select mux mode: ALT3 mux port: KPP_COL07 of instance: kpp 0100b - Select mux mode: ALT4 mux port: FLEXPWM2_PWMX03 of instance: flexpwm2 0101b - Select mux mode: ALT5 mux port: GPIO4_IO13 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: LPUART3_TX of instance: lpuart3 0111b - Select mux mode: ALT7 mux port: USDHC2_CD_B of instance: usdhc2 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO13 of instance: flexio2 1001b - Reserved 1010b - Reserved 1011b - Reserved

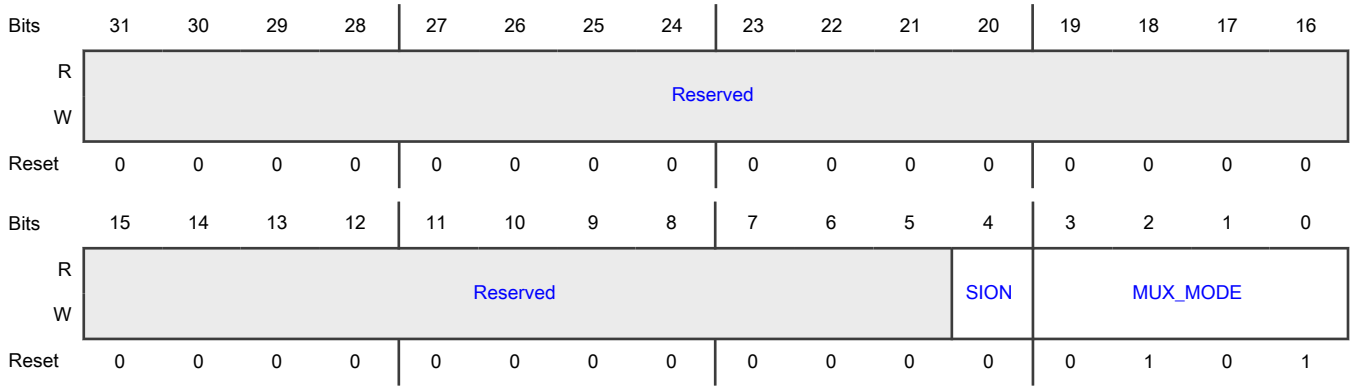
17.4.1.79 SW_MUX_CTL_PAD_GPIO_AD_14 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_14)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_14	144h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_14
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AD_14. 0000b - Select mux mode: ALT0 mux port: SPDIF_EXT_CLK of instance: spdif 0001b - Select mux mode: ALT1 mux port: LPI2C5_HREQ of instance: lpi2c5 0010b - Select mux mode: ALT2 mux port: GPT1_COMPARE1 of instance: gpt1 0011b - Select mux mode: ALT3 mux port: KPP_ROW06 of instance: kpp 0100b - Select mux mode: ALT4 mux port: FLEXPWM3_PWMX00 of instance: flexpwm3 0101b - Select mux mode: ALT5 mux port: GPIO4_IO14 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: LPUART3_RX of instance: lpuart3 0111b - Select mux mode: ALT7 mux port: USDHC2_WP of instance: usdhc2 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO14 of instance: flexio2 1001b - Reserved 1010b - Reserved 1011b - Reserved

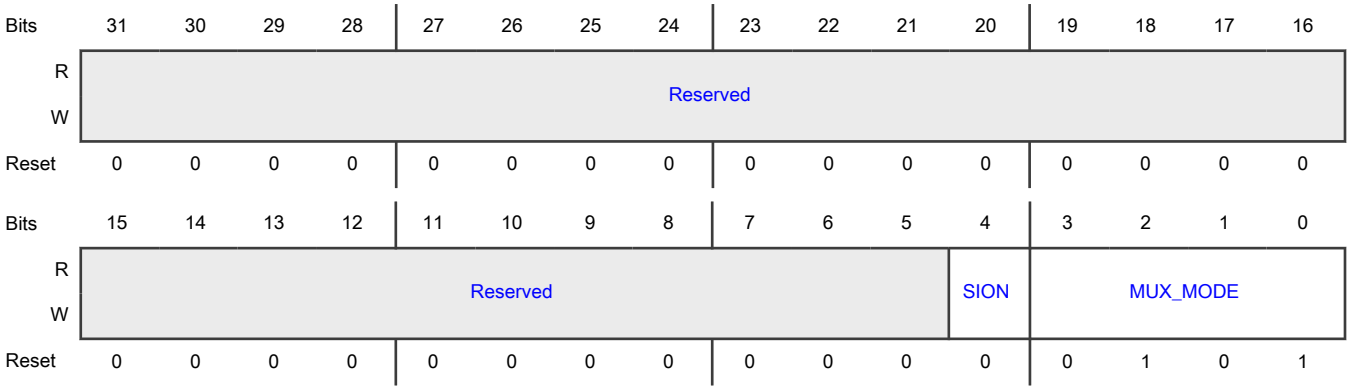
17.4.1.80 SW_MUX_CTL_PAD_GPIO_AD_15 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_15)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_15	148h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_15
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AD_15. 0000b - Select mux mode: ALT0 mux port: SPDIF_IN of instance: spdif 0001b - Select mux mode: ALT1 mux port: LPUART10_TX of instance: lpuart10 0010b - Select mux mode: ALT2 mux port: GPT1_COMPARE2 of instance: gpt1 0011b - Select mux mode: ALT3 mux port: KPP_COL06 of instance: kpp 0100b - Select mux mode: ALT4 mux port: FLEXPWM3_PWMX01 of instance: flexpwm3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Select mux mode: ALT5 mux port: GPIO4_IO15 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: LPUART3_CTS_B of instance: lpuart3 0111b - Select mux mode: ALT7 mux port: LPSPI3_PCS1 of instance: lpspi3 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO15 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: CAN1_TX of instance: can1 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_CLK_ECAT_CLK25 of instance: ecat

17.4.1.81 SW_MUX_CTL_PAD_GPIO_AD_16 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_16)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_16	14Ch

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_16
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_AD_16. 0000b - Select mux mode: ALT0 mux port: SPDIF_OUT of instance: spdif 0001b - Select mux mode: ALT1 mux port: LPUART10_RX of instance: lpuart10 0010b - Select mux mode: ALT2 mux port: GPT1_COMPARE3 of instance: gpt1 0011b - Select mux mode: ALT3 mux port: KPP_ROW05 of instance: kpp 0100b - Select mux mode: ALT4 mux port: FLEXPWM3_PWMX02 of instance: flexpwm3 0101b - Select mux mode: ALT5 mux port: GPIO4_IO16 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: LPUART3_RTS_B of instance: lpuart3 0111b - Select mux mode: ALT7 mux port: LPSPI3_SCK of instance: lpspi3 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO16 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: CAN1_RX of instance: can1 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LINK_0 of instance: ecat

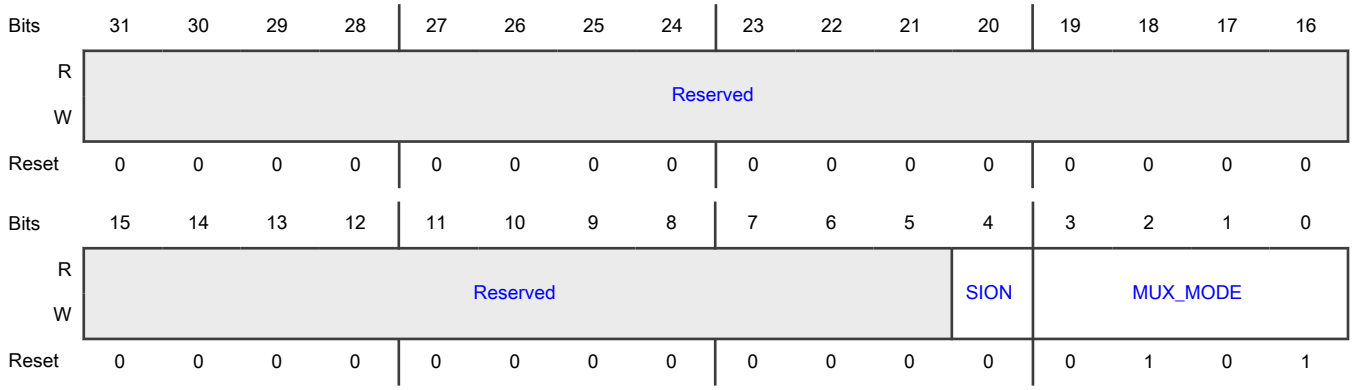
17.4.1.82 SW_MUX_CTL_PAD_GPIO_AD_17 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_17)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_17	150h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_17
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_AD_17. 0000b - Select mux mode: ALT0 mux port: SAI4_MCLK of instance: sai4 0001b - Select mux mode: ALT1 mux port: ACMP1_CMPO of instance: acmp1 0010b - Select mux mode: ALT2 mux port: GPT1_CLK of instance: gpt1 0011b - Select mux mode: ALT3 mux port: KPP_COL05 of instance: kpp 0100b - Select mux mode: ALT4 mux port: FLEXPWM3_PWMX03 of instance: flexpwm3 0101b - Select mux mode: ALT5 mux port: GPIO4_IO17 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: I3C2_PUR of instance: i3c2 0111b - Select mux mode: ALT7 mux port: LPSPI3_PCS0 of instance: lpspi3 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO17 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: LPI2C3_HREQ of instance: lpi2c3 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LINK_1 of instance: ecat

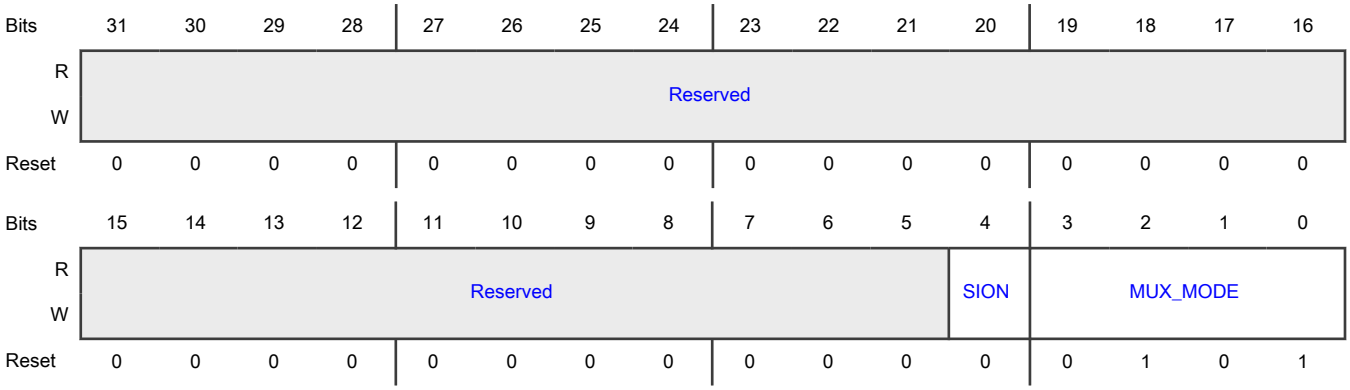
17.4.1.83 SW_MUX_CTL_PAD_GPIO_AD_18 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_18)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_18	154h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_18
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_AD_18. 0000b - Select mux mode: ALT0 mux port: SAI4_RX_SYNC of instance: sai4 0001b - Select mux mode: ALT1 mux port: ACMP2_CMPO of instance: acmp2 0010b - Select mux mode: ALT2 mux port: LPUART5_RI_B of instance: lpuart5 0011b - Select mux mode: ALT3 mux port: KPP_ROW04 of instance: kpp 0100b - Select mux mode: ALT4 mux port: FLEXPWM4_PWMX00 of instance: flexpwm4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Select mux mode: ALT5 mux port: GPIO4_IO18 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: I3C2_SCL of instance: i3c2 0111b - Select mux mode: ALT7 mux port: LPSPI3_SDO of instance: lpspi3 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO18 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: LPI2C3_SCL of instance: lpi2c3 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_PROM_CLK of instance: ecat

17.4.1.84 SW_MUX_CTL_PAD_GPIO_AD_19 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_19)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_19	158h

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_19
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_AD_19. 0000b - Select mux mode: ALT0 mux port: SAI4_RX_BCLK of instance: sai4 0001b - Select mux mode: ALT1 mux port: ACMP3_CMPO of instance: acmp3 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT19 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: KPP_COL04 of instance: kpp 0100b - Select mux mode: ALT4 mux port: FLEXPWM4_PWMX01 of instance: flexpwm4 0101b - Select mux mode: ALT5 mux port: GPIO4_IO19 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: I3C2_SDA of instance: i3c2 0111b - Select mux mode: ALT7 mux port: LPSPI3_SDI of instance: lpspi3 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO19 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: LPI2C3_SDA of instance: lpi2c3 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_PROM_DATA of instance: ecat

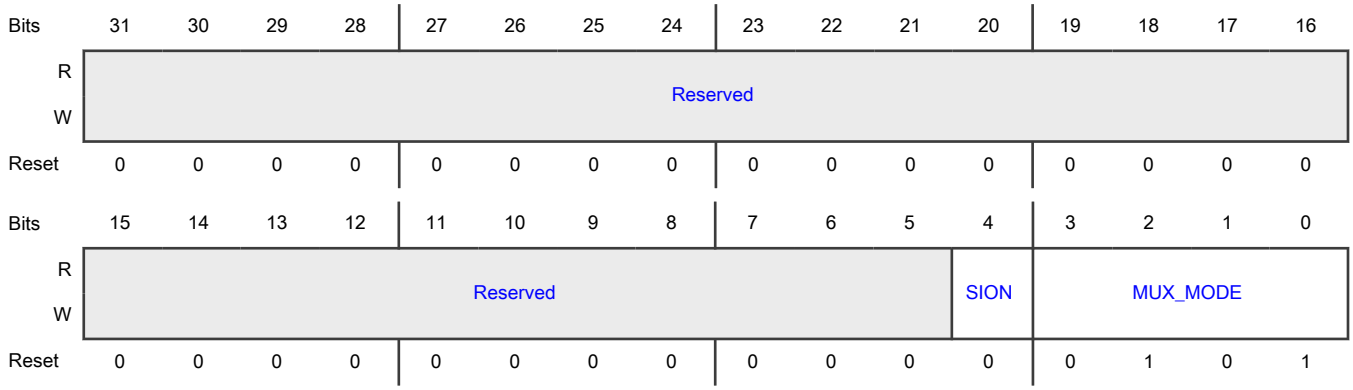
17.4.1.85 SW_MUX_CTL_PAD_GPIO_AD_20 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_20)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_20	15Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_20
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_AD_20. 0000b - Select mux mode: ALT0 mux port: SAI4_RX_DATA00 of instance: sai4 0001b - Select mux mode: ALT1 mux port: ACMP4_CMPO of instance: acmp4 0010b - Select mux mode: ALT2 mux port: LPIT2_TRIGGER00 of instance: lpit2 0011b - Select mux mode: ALT3 mux port: SINC1_EMCLK00 of instance: sinc1 0100b - Select mux mode: ALT4 mux port: FLEXPWM4_PWMX02 of instance: flexpwm4 0101b - Select mux mode: ALT5 mux port: GPIO4_IO20 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: NETC_TMR_TRIG1 of instance: netc 0111b - Select mux mode: ALT7 mux port: NETC_1588_CLK of instance: netc_clkgen 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO20 of instance: flexio2 1010b - Reserved 1011b - Reserved

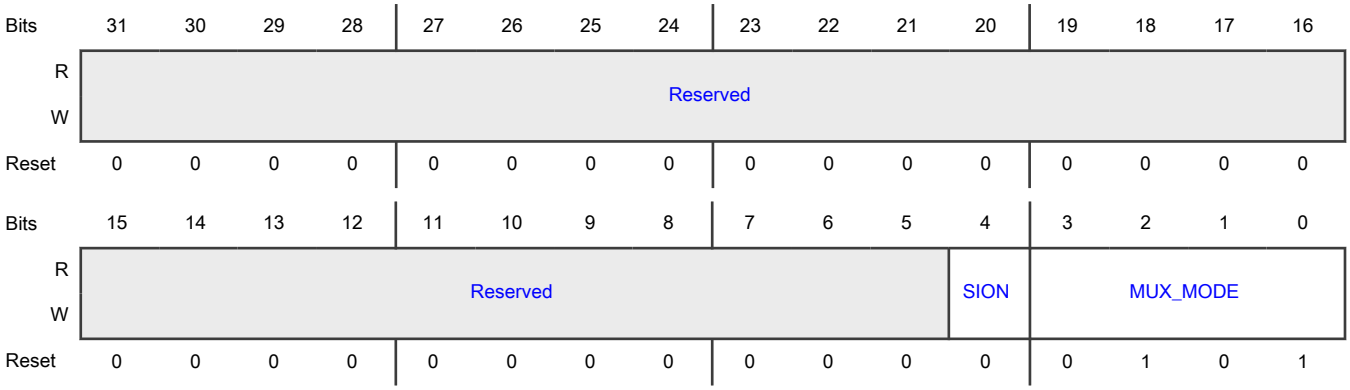
17.4.1.86 SW_MUX_CTL_PAD_GPIO_AD_21 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_21)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_21	160h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_21
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_AD_21. 0000b - Select mux mode: ALT0 mux port: SAI4_TX_DATA00 of instance: sai4 0010b - Select mux mode: ALT2 mux port: LPIT2_TRIGGER01 of instance: lpit2 0011b - Select mux mode: ALT3 mux port: SINC1_EMBIT00 of instance: sinc1 0100b - Select mux mode: ALT4 mux port: FLEXPWM4_PWMX03 of instance: flexpwm4 0101b - Select mux mode: ALT5 mux port: GPIO4_IO21 of instance: gpio4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0110b - Select mux mode: ALT6 mux port: NETC_TMR_TRIG2 of instance: netc 0111b - Select mux mode: ALT7 mux port: NETC_TMR_GCLK of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO21 of instance: flexio2 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LED_RUN of instance: ecat

17.4.1.87 SW_MUX_CTL_PAD_GPIO_AD_22 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_22)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_22	164h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_22
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AD_22. 0000b - Select mux mode: ALT0 mux port: SAI4_TX_BCLK of instance: sai4 0010b - Select mux mode: ALT2 mux port: LPIT2_TRIGGER02 of instance: lpit2 0011b - Select mux mode: ALT3 mux port: SINC1_EMCLK01 of instance: sinc1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO22 of instance: gpio4 0111b - Select mux mode: ALT7 mux port: NETC_TMR_ALARM1 of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO22 of instance: flexio2 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LED_ERR of instance: ecat

17.4.1.88 SW_MUX_CTL_PAD_GPIO_AD_23 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_23)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_23	168h

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_23
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AD_23. 0000b - Select mux mode: ALT0 mux port: SAI4_TX_SYNC of instance: sai4 0010b - Select mux mode: ALT2 mux port: LPIT2_TRIGGER03 of instance: lpit2 0011b - Select mux mode: ALT3 mux port: SINC1_EMBIT01 of instance: sinc1 0101b - Select mux mode: ALT5 mux port: GPIO4_IO23 of instance: gpio4 0111b - Select mux mode: ALT7 mux port: NETC_TMR_ALARM2 of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO23 of instance: flexio2 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LED_STATE_RUN of instance: ecat

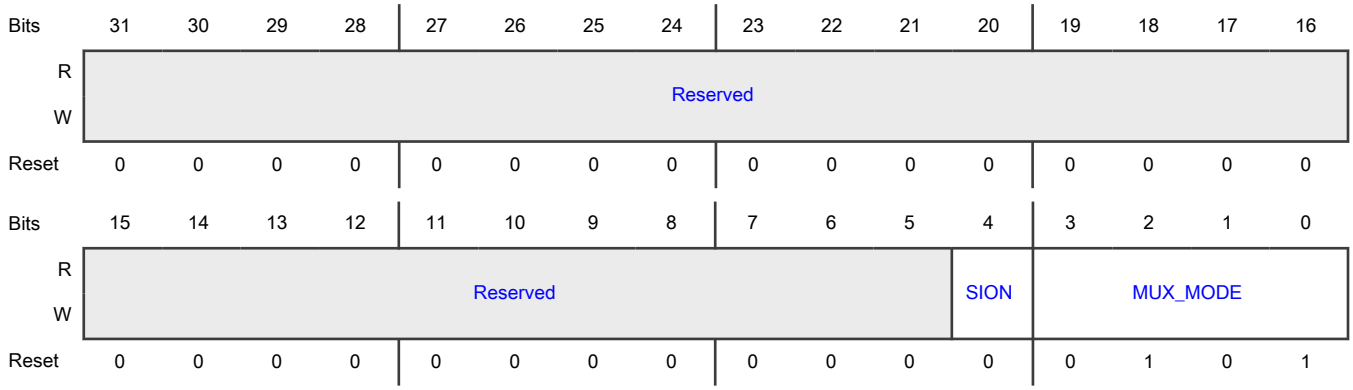
17.4.1.89 SW_MUX_CTL_PAD_GPIO_AD_24 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_24)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_24	16Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_24
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_AD_24. 0000b - Select mux mode: ALT0 mux port: LPUART6_TX of instance: lpuart6 0001b - Select mux mode: ALT1 mux port: LPI2C4_SCL of instance: lpi2c4 0011b - Select mux mode: ALT3 mux port: SINC2_MOD_CLK1 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: FLEXPWM2_PWMA00 of instance: flexpwm2 0101b - Select mux mode: ALT5 mux port: GPIO4_IO24 of instance: gpio4 0111b - Select mux mode: ALT7 mux port: NETC_TMR_TRIG1 of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO24 of instance: flexio2 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LINK_ACT00 of instance: ecat

17.4.1.90 SW_MUX_CTL_PAD_GPIO_AD_25 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_25)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_25	170h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved											SION	MUX_MODE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_25
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_AD_25. 0000b - Select mux mode: ALT0 mux port: LPUART6_RX of instance: lpuart6 0001b - Select mux mode: ALT1 mux port: LPI2C4_SDA of instance: lpi2c4 0010b - Select mux mode: ALT2 mux port: LPSPI5_PCS3 of instance: lpspi5 0011b - Select mux mode: ALT3 mux port: SINC2_MOD_CLK2 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: FLEXPWM2_PWMB00 of instance: flexpwm2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Select mux mode: ALT5 mux port: GPIO4_IO25 of instance: gpio4 0111b - Select mux mode: ALT7 mux port: NETC_TMR_TRIG2 of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO25 of instance: flexio2 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LINK_ACT01 of instance: ecat

17.4.1.91 SW_MUX_CTL_PAD_GPIO_AD_26 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_26)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_26	174h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_26
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_AD_26. 0000b - Select mux mode: ALT0 mux port: LPUART6_CTS_B of instance: lpuart6 0001b - Select mux mode: ALT1 mux port: LPUART5_TX of instance: lpuart5 0010b - Select mux mode: ALT2 mux port: LPSPI5_PCS2 of instance: lpspi5 0011b - Select mux mode: ALT3 mux port: SINC2_EMCLK00 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: FLEXPWM2_PWMA01 of instance: flexpwm2 0101b - Select mux mode: ALT5 mux port: GPIO4_IO26 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: KPP_ROW00 of instance: kpp 0111b - Select mux mode: ALT7 mux port: NETC_TMR_PP1 of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO26 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: USDHC2_CD_B of instance: usdhc2 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: MIC_BITSTREAM02 of instance: mic

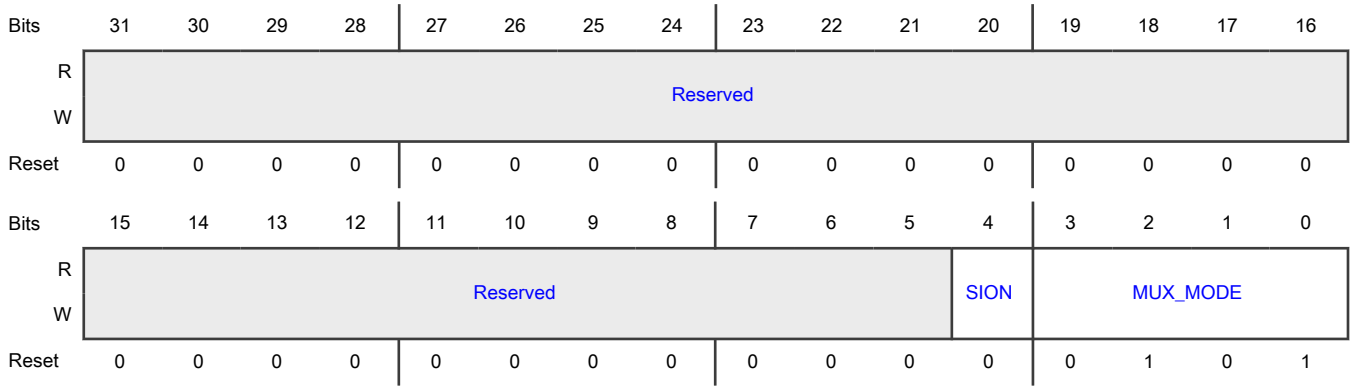
17.4.1.92 SW_MUX_CTL_PAD_GPIO_AD_27 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_27)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_27	178h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_27
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_AD_27. 0000b - Select mux mode: ALT0 mux port: LPUART6_RTS_B of instance: lpuart6 0001b - Select mux mode: ALT1 mux port: LPUART5_RX of instance: lpuart5 0010b - Select mux mode: ALT2 mux port: LPSPI5_PCS1 of instance: lpspi5 0011b - Select mux mode: ALT3 mux port: SINC2_EMBIT00 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: FLEXPWM2_PWMB01 of instance: flexpwm2 0101b - Select mux mode: ALT5 mux port: GPIO4_IO27 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: KPP_COL00 of instance: kpp 0111b - Select mux mode: ALT7 mux port: NETC_TMR_PP2 of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO27 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: USDHC2_WP of instance: usdhc2 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: MIC_CLK of instance: mic

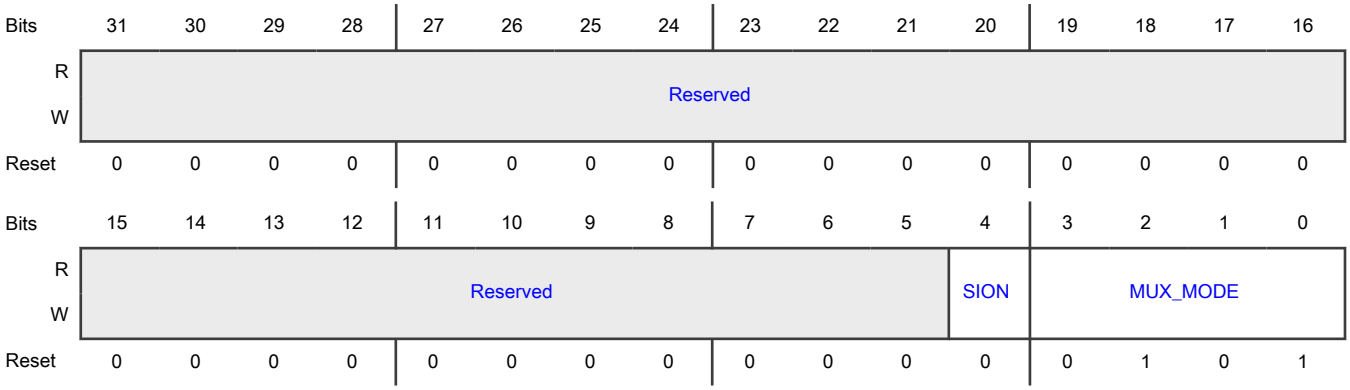
17.4.1.93 SW_MUX_CTL_PAD_GPIO_AD_28 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_28)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_28	17Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_28
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AD_28. 0000b - Select mux mode: ALT0 mux port: LPSPI5_SCK of instance: lpspi5 0010b - Select mux mode: ALT2 mux port: I3C1_PUR of instance: i3c1 0011b - Select mux mode: ALT3 mux port: SINC2_EMCLK01 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: FLEXPWM2_PWMB02 of instance: flexpwm2 0101b - Select mux mode: ALT5 mux port: GPIO4_IO28 of instance: gpio4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0110b - Select mux mode: ALT6 mux port: KPP_ROW03 of instance: kpp 0111b - Select mux mode: ALT7 mux port: NETC_TMR_PP3 of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO28 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: USDHC2_RESET_B of instance: usdhc2 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: MIC_BITSTREAM00 of instance: mic

17.4.1.94 SW_MUX_CTL_PAD_GPIO_AD_29 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_29)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_29	180h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4	Software Input On Field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SION	Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_29
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AD_29. 0000b - Select mux mode: ALT0 mux port: LPSPI5_PCS0 of instance: lpspi5 0010b - Select mux mode: ALT2 mux port: USDHC2_CD_B of instance: usdhc2 0011b - Select mux mode: ALT3 mux port: SINC2_EMBIT01 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: FLEXPWM2_PWMA02 of instance: flexpwm2 0101b - Select mux mode: ALT5 mux port: GPIO4_IO29 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: KPP_COL03 of instance: kpp 0111b - Select mux mode: ALT7 mux port: EWM_EWM_OUT_B of instance: ewm 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO29 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: USDHC2_VSELECT of instance: usdhc2 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: MIC_BITSTREAM01 of instance: mic

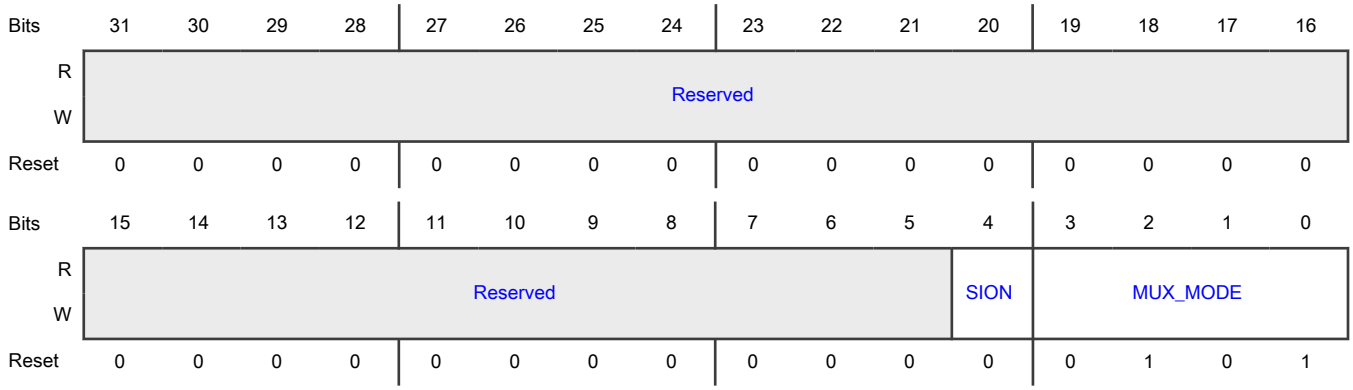
17.4.1.95 SW_MUX_CTL_PAD_GPIO_AD_30 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_30)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_30	184h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_30
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_AD_30. 0000b - Select mux mode: ALT0 mux port: LPSPI5_SDO of instance: lpspi5 0001b - Select mux mode: ALT1 mux port: USB_OTG2_OC of instance: usb 0010b - Select mux mode: ALT2 mux port: CAN2_TX of instance: can2 0011b - Select mux mode: ALT3 mux port: SINC2_EMCLK02 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: LPUART8_TX of instance: lpuart8 0101b - Select mux mode: ALT5 mux port: GPIO4_IO30 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: KPP_ROW02 of instance: kpp 0111b - Select mux mode: ALT7 mux port: NETC_EMDC of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO30 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: XBAR1_XBAR_INOUT24 of instance: xbar1 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_MCLK of instance: ecat

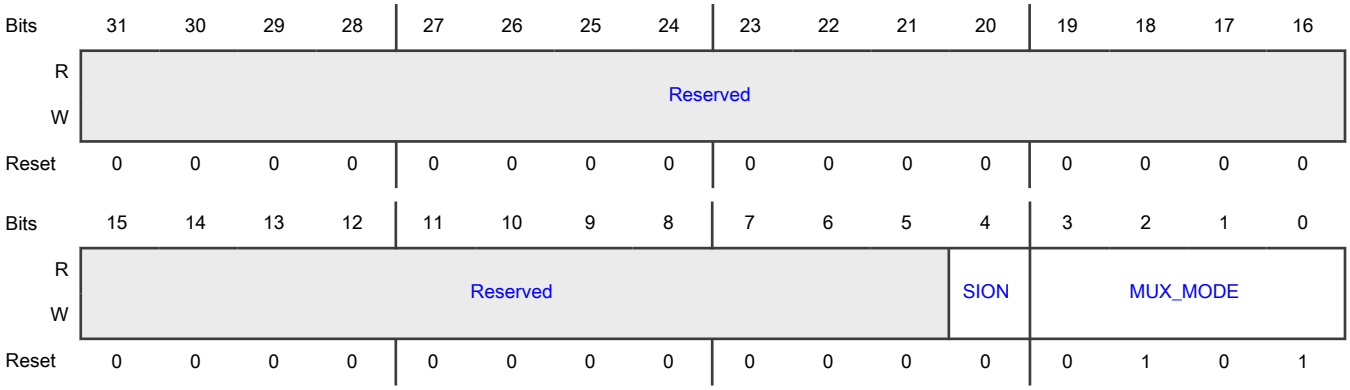
17.4.1.96 SW_MUX_CTL_PAD_GPIO_AD_31 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_31)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_31	188h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_31
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_AD_31. 0000b - Select mux mode: ALT0 mux port: LPSPI5_SDI of instance: lpspi5 0001b - Select mux mode: ALT1 mux port: USB_OTG2_PWR of instance: usb 0010b - Select mux mode: ALT2 mux port: CAN2_RX of instance: can2 0011b - Select mux mode: ALT3 mux port: SINC2_EMBIT02 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: LPUART8_RX of instance: lpuart8

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Select mux mode: ALT5 mux port: GPIO4_IO31 of instance: gpio4 0110b - Select mux mode: ALT6 mux port: KPP_COL02 of instance: kpp 0111b - Select mux mode: ALT7 mux port: NETC_EMDIO of instance: netc 1000b - Select mux mode: ALT8 mux port: FLEXIO2_FLEXIO31 of instance: flexio2 1001b - Select mux mode: ALT9 mux port: XBAR1_XBAR_INOUT25 of instance: xbar1 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_MDIO of instance: ecat

17.4.1.97 SW_MUX_CTL_PAD_GPIO_AD_32 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_32)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_32	18Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_32
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AD_32. 0000b - Select mux mode: ALT0 mux port: LPI2C5_SCL of instance: lpi2c5 0001b - Select mux mode: ALT1 mux port: USBPHY2_OTG_ID of instance: usbphy2 0010b - Select mux mode: ALT2 mux port: GPC_PMIC_RDY of instance: gpc 0011b - Select mux mode: ALT3 mux port: SINC2_EMCLK03 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: USDHC1_CD_B of instance: usdhc1 0101b - Select mux mode: ALT5 mux port: GPIO5_IO00 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: KPP_ROW01 of instance: kpp 0111b - Select mux mode: ALT7 mux port: NETC_TMR_TRIG1 of instance: netc 1000b - Select mux mode: ALT8 mux port: LPUART10_TX of instance: lpuart10 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: MIC_BITSTREAM03 of instance: mic

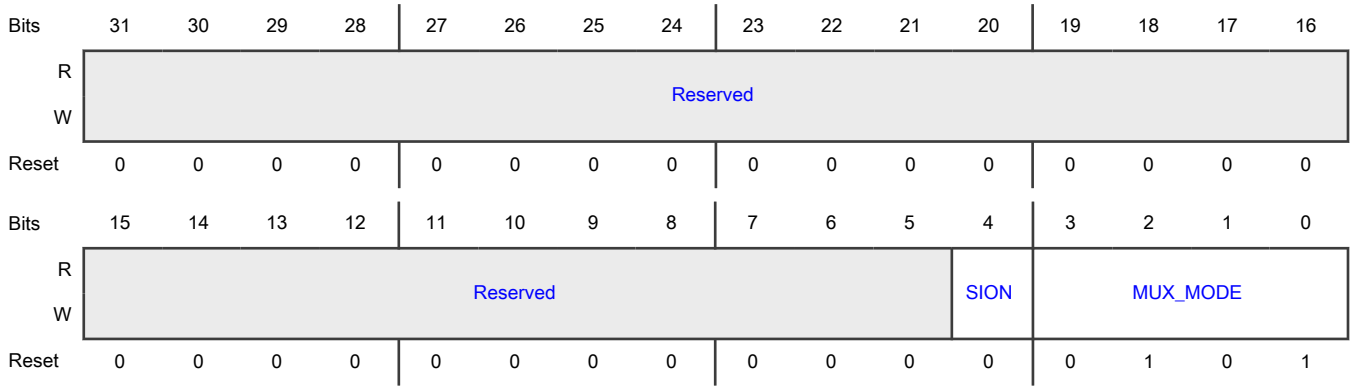
17.4.1.98 SW_MUX_CTL_PAD_GPIO_AD_33 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_33)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_33	190h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_33
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_AD_33. 0000b - Select mux mode: ALT0 mux port: LPI2C5_SDA of instance: lpi2c5 0001b - Select mux mode: ALT1 mux port: USBPHY1_OTG_ID of instance: usbphy1 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT17 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: SINC2_EMBIT03 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: USDHC1_WP of instance: usdhc1 0101b - Select mux mode: ALT5 mux port: GPIO5_IO01 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: KPP_COL01 of instance: kpp 0111b - Select mux mode: ALT7 mux port: NETC_TMR_TRIG2 of instance: netc 1000b - Select mux mode: ALT8 mux port: LPUART10_RX of instance: lpuart10 1010b - Reserved 1011b - Reserved

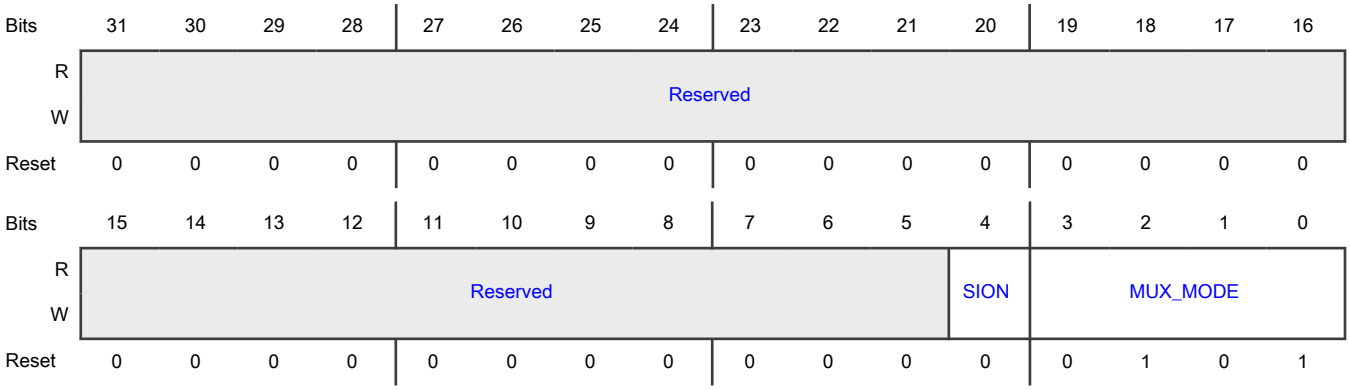
17.4.1.99 SW_MUX_CTL_PAD_GPIO_AD_34 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_34)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_34	194h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_34
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_AD_34. 0000b - Select mux mode: ALT0 mux port: I3C2_SCL of instance: i3c2 0001b - Select mux mode: ALT1 mux port: USB_OTG1_PWR of instance: usb 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT18 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: SINC_FILTER_GLUE2_BREAK of instance: sinc_filter_glue2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: USDHC1_VSELECT of instance: usdhc1 0101b - Select mux mode: ALT5 mux port: GPIO5_IO02 of instance: gpio5 0111b - Select mux mode: ALT7 mux port: NETC_TMR_ALARM1 of instance: netc 1000b - Select mux mode: ALT8 mux port: LPUART10_CTS_B of instance: lpuart10 1010b - Reserved 1011b - Reserved

17.4.1.100 SW_MUX_CTL_PAD_GPIO_AD_35 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AD_35)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AD_35	198h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AD_35
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_AD_35. 0000b - Select mux mode: ALT0 mux port: I3C2_SDA of instance: i3c2 0001b - Select mux mode: ALT1 mux port: USB_OTG1_OC of instance: usb 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT19 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: SINC2_MOD_CLK0 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: USDHC1_RESET_B of instance: usdhc1 0101b - Select mux mode: ALT5 mux port: GPIO5_IO03 of instance: gpio5 0111b - Select mux mode: ALT7 mux port: NETC_TMR_ALARM2 of instance: netc 1000b - Select mux mode: ALT8 mux port: LPUART10_RTS_B of instance: lpuart10 1010b - Reserved 1011b - Reserved

17.4.1.101 SW_MUX_CTL_PAD_GPIO_SD_B1_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_00)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B1_00	19Ch

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B1_00
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_SD_B1_00. 0000b - Select mux mode: ALT0 mux port: USDHC1_CMD of instance: usdhc1 0001b - Select mux mode: ALT1 mux port: SINC1_EMCLK02 of instance: sinc1 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT20 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: LPTMR2_ALT1 of instance: lptmr2 0100b - Select mux mode: ALT4 mux port: XSPI_SLV_CS of instance: xspi_slv 0101b - Select mux mode: ALT5 mux port: GPIO5_IO04 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPSPI3_PCS0 of instance: lpspi3 1000b - Select mux mode: ALT8 mux port: KPP_ROW07 of instance: kpp 1001b - Reserved 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: CCM_CLKO1 of instance: ccm

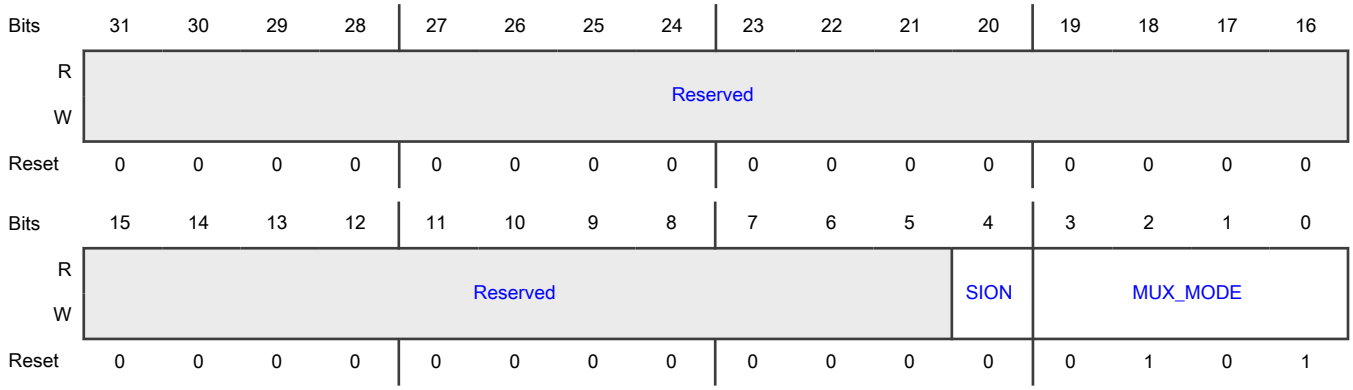
17.4.1.102 SW_MUX_CTL_PAD_GPIO_SD_B1_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_01)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B1_01	1A0h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B1_01
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_SD_B1_01. 0000b - Select mux mode: ALT0 mux port: USDHC1_CLK of instance: usdhc1 0001b - Select mux mode: ALT1 mux port: SINC1_EMBIT02 of instance: sinc1 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT21 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: LPTMR2_ALT2 of instance: lptmr2 0100b - Select mux mode: ALT4 mux port: XSPI_SLV_CLK of instance: xspi_slv 0101b - Select mux mode: ALT5 mux port: GPIO5_IO05 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPSPI3_SCK of instance: lpspi3 1000b - Select mux mode: ALT8 mux port: KPP_COL07 of instance: kpp 1001b - Reserved 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: CCM_CLKO2 of instance: ccm

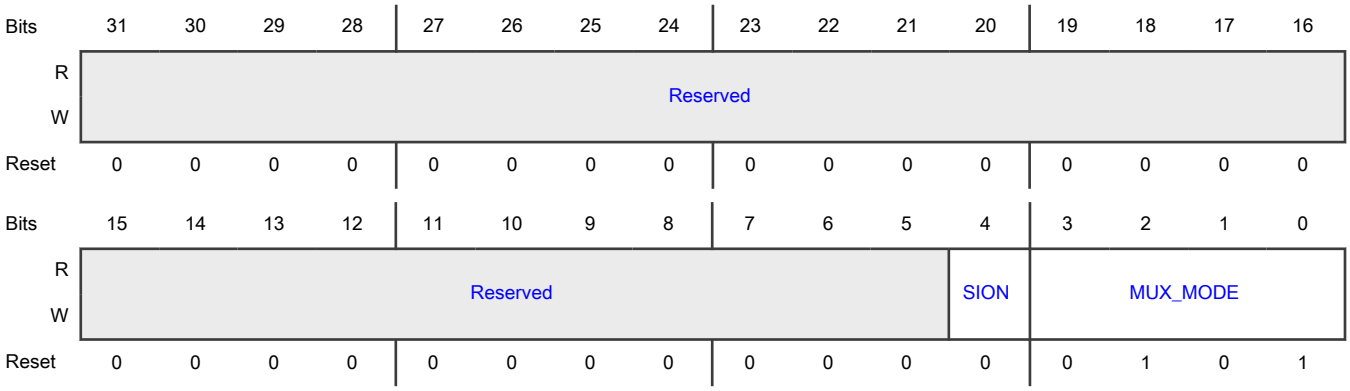
17.4.1.103 SW_MUX_CTL_PAD_GPIO_SD_B1_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_02)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B1_02	1A4h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B1_02
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_SD_B1_02. 0000b - Select mux mode: ALT0 mux port: USDHC1_DATA0 of instance: usdhc1 0001b - Select mux mode: ALT1 mux port: SINC1_EMCLK03 of instance: sinc1 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT22 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: LPTMR2_ALT3 of instance: lptmr2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: XSPI_SLV_DATA04 of instance: xspi_slv 0101b - Select mux mode: ALT5 mux port: GPIO5_IO06 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPSPI3_SDO of instance: lpspi3 1000b - Select mux mode: ALT8 mux port: KPP_ROW06 of instance: kpp 1001b - Select mux mode: ALT9 mux port: FLEXSPI1_BUS2BIT_A_SS1_B of instance: flexspi1_bus2bit 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_RESET_OUT of instance: ecat

17.4.1.104 SW_MUX_CTL_PAD_GPIO_SD_B1_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_03)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B1_03	1A8h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B1_03
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_SD_B1_03. 0000b - Select mux mode: ALT0 mux port: USDHC1_DATA1 of instance: usdhc1 0001b - Select mux mode: ALT1 mux port: SINC1_EMBIT03 of instance: sinc1 0010b - Select mux mode: ALT2 mux port: XBAR1_XBAR_INOUT23 of instance: xbar1 0011b - Select mux mode: ALT3 mux port: LPTMR3_ALT1 of instance: lptmr3 0100b - Select mux mode: ALT4 mux port: XSPI_SLV_DATA05 of instance: xspi_slv 0101b - Select mux mode: ALT5 mux port: GPIO5_IO07 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPSPI3_SDI of instance: lpspi3 1000b - Select mux mode: ALT8 mux port: KPP_COL06 of instance: kpp 1001b - Select mux mode: ALT9 mux port: FLEXSPI1_BUS2BIT_B_SS1_B of instance: flexspi1_bus2bit 1010b - Reserved 1011b - Reserved

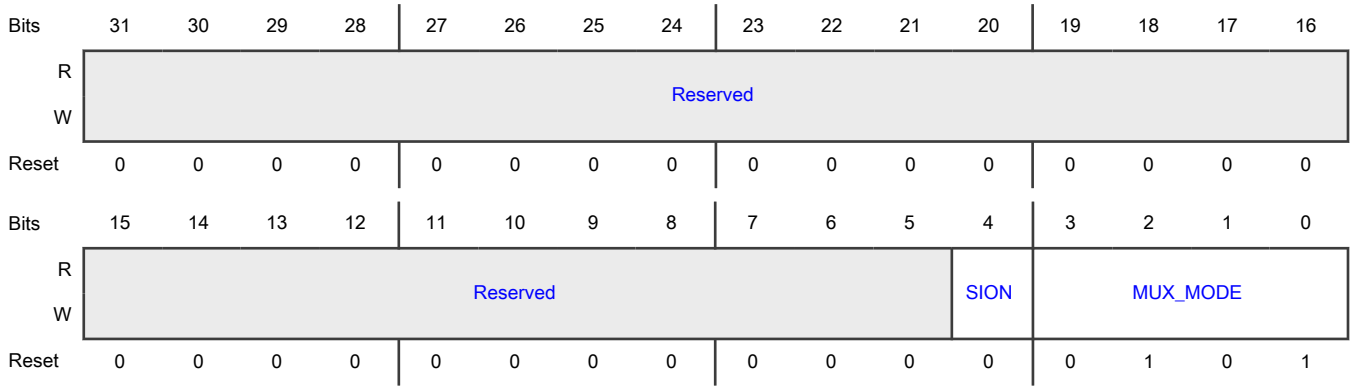
17.4.1.105 SW_MUX_CTL_PAD_GPIO_SD_B1_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_04)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B1_04	1ACh

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B1_04
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_SD_B1_04. 0000b - Select mux mode: ALT0 mux port: USDHC1_DATA2 of instance: usdhc1 0001b - Select mux mode: ALT1 mux port: SINC_FILTER_GLUE1_BREAK of instance: sinc_filter_glue1 0010b - Select mux mode: ALT2 mux port: SINC2_EMCLK02 of instance: sinc2 0011b - Select mux mode: ALT3 mux port: LPTMR3_ALT2 of instance: lptmr3 0100b - Select mux mode: ALT4 mux port: XSPI_SLV_DATA06 of instance: xspi_slv 0101b - Select mux mode: ALT5 mux port: GPIO5_IO08 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPSPI3_PCS1 of instance: lpspi3 1000b - Select mux mode: ALT8 mux port: FLEXSPI1_BUS2BIT_B_SS0_B of instance: flexspi1_bus2bit 1001b - Select mux mode: ALT9 mux port: FLEXSPI1_BUS2BIT_A_SS1_B of instance: flexspi1_bus2bit 1010b - Reserved 1011b - Reserved

17.4.1.106 SW_MUX_CTL_PAD_GPIO_SD_B1_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B1_05)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B1_05	1B0h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B1_05
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_SD_B1_05. 0000b - Select mux mode: ALT0 mux port: USDHC1_DATA3 of instance: usdhc1 0010b - Select mux mode: ALT2 mux port: SINC2_EMBIT02 of instance: sinc2 0011b - Select mux mode: ALT3 mux port: LPTMR3_ALT3 of instance: lptmr3 0100b - Select mux mode: ALT4 mux port: XSPI_SLV_DATA07 of instance: xspi_slv

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Select mux mode: ALT5 mux port: GPIO5_IO09 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPSPI3_PCS2 of instance: lpspi3 1000b - Reserved 1001b - Select mux mode: ALT9 mux port: FLEXSPI1_BUS2BIT_B_SS0_B of instance: flexspi1_bus2bit 1010b - Reserved 1011b - Reserved

17.4.1.107 SW_MUX_CTL_PAD_GPIO_SD_B2_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_00)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_00	1B4h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4	Software Input On Field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SION	Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_00
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_SD_B2_00. 0000b - Select mux mode: ALT0 mux port: USDHC2_DATA3 of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_DATA04 of instance: flexspi1_bus2bit 0010b - Select mux mode: ALT2 mux port: XSPI_SLV_DATA04 of instance: xspi_slv 0011b - Select mux mode: ALT3 mux port: XBAR1_XBAR_INOUT17 of instance: xbar1 0100b - Select mux mode: ALT4 mux port: KPP_ROW01 of instance: kpp 0101b - Select mux mode: ALT5 mux port: GPIO5_IO10 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPSPI3_PCS3 of instance: lpspi3 1000b - Select mux mode: ALT8 mux port: NETC_1588_CLK of instance: netc_clkgen 1001b - Select mux mode: ALT9 mux port: LPUART8_TX of instance: lpuart8 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: MIC_BITSTREAM00 of instance: mic

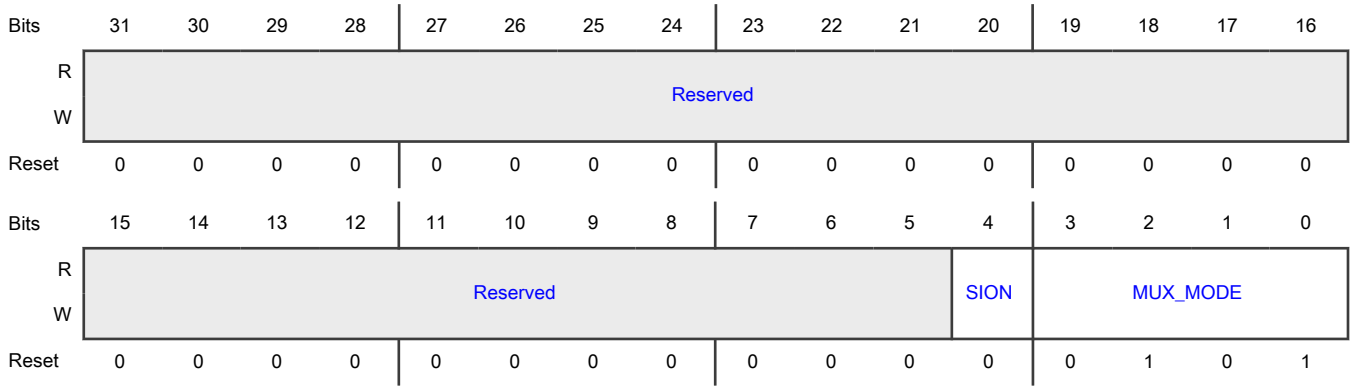
17.4.1.108 SW_MUX_CTL_PAD_GPIO_SD_B2_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_01)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_01	1B8h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_01
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_SD_B2_01. 0000b - Select mux mode: ALT0 mux port: USDHC2_DATA2 of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_DATA05 of instance: flexspi1_bus2bit 0010b - Select mux mode: ALT2 mux port: XSPI_SLV_DATA05 of instance: xspi_slv 0011b - Select mux mode: ALT3 mux port: QTIMER6_TIMER0 of instance: qtimer6 0100b - Select mux mode: ALT4 mux port: KPP_COL01 of instance: kpp 0101b - Select mux mode: ALT5 mux port: GPIO5_IO11 of instance: gpio5 0110b - Reserved 1000b - Select mux mode: ALT8 mux port: NETC_TMR_GCLK of instance: netc 1001b - Select mux mode: ALT9 mux port: LPUART8_RX of instance: lpuart8 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: MIC_BITSTREAM01 of instance: mic

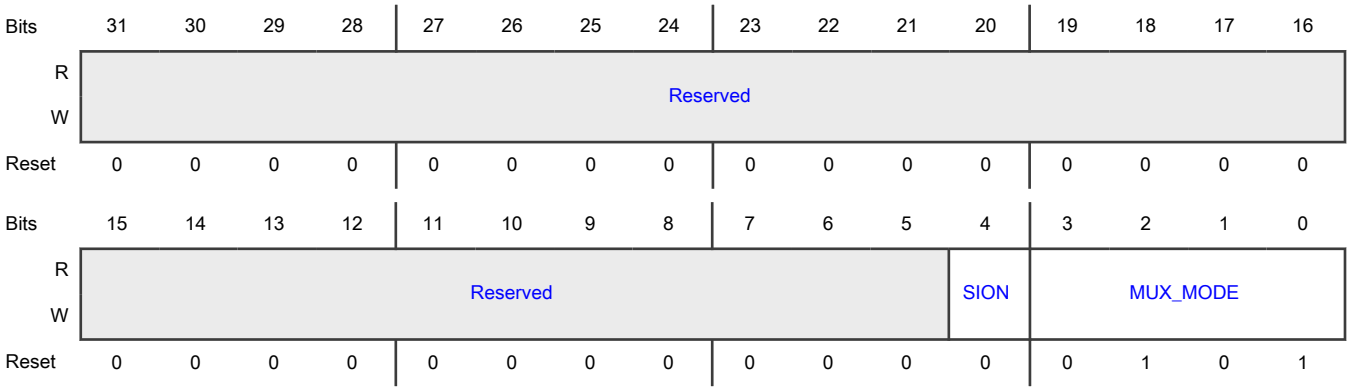
17.4.1.109 SW_MUX_CTL_PAD_GPIO_SD_B2_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_02)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_02	1BCh

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_02
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_SD_B2_02. 0000b - Select mux mode: ALT0 mux port: USDHC2_DATA1 of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_DATA06 of instance: flexspi1_bus2bit 0010b - Select mux mode: ALT2 mux port: XSPI_SLV_DATA06 of instance: xspi_slv 0011b - Select mux mode: ALT3 mux port: QTIMER6_TIMER1 of instance: qtimer6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: KPP_ROW00 of instance: kpp 0101b - Select mux mode: ALT5 mux port: GPIO5_IO12 of instance: gpio5 0110b - Reserved 1000b - Select mux mode: ALT8 mux port: NETC_TMR_ALARM1 of instance: netc 1001b - Select mux mode: ALT9 mux port: LPUART8_CTS_B of instance: lpuart8 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: MIC_BITSTREAM02 of instance: mic

17.4.1.110 SW_MUX_CTL_PAD_GPIO_SD_B2_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_03)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_03	1C0h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_03
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_SD_B2_03. 0000b - Select mux mode: ALT0 mux port: USDHC2_DATA0 of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_DATA07 of instance: flexspi1_bus2bit 0010b - Select mux mode: ALT2 mux port: XSPI_SLV_DATA07 of instance: xspi_slv 0011b - Select mux mode: ALT3 mux port: QTIMER6_TIMER2 of instance: qtimer6 0100b - Select mux mode: ALT4 mux port: KPP_COL00 of instance: kpp 0101b - Select mux mode: ALT5 mux port: GPIO5_IO13 of instance: gpio5 0110b - Reserved 1000b - Select mux mode: ALT8 mux port: NETC_TMR_ALARM2 of instance: netc 1001b - Select mux mode: ALT9 mux port: LPUART8_RTS_B of instance: lpuart8 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: MIC_BITSTREAM03 of instance: mic

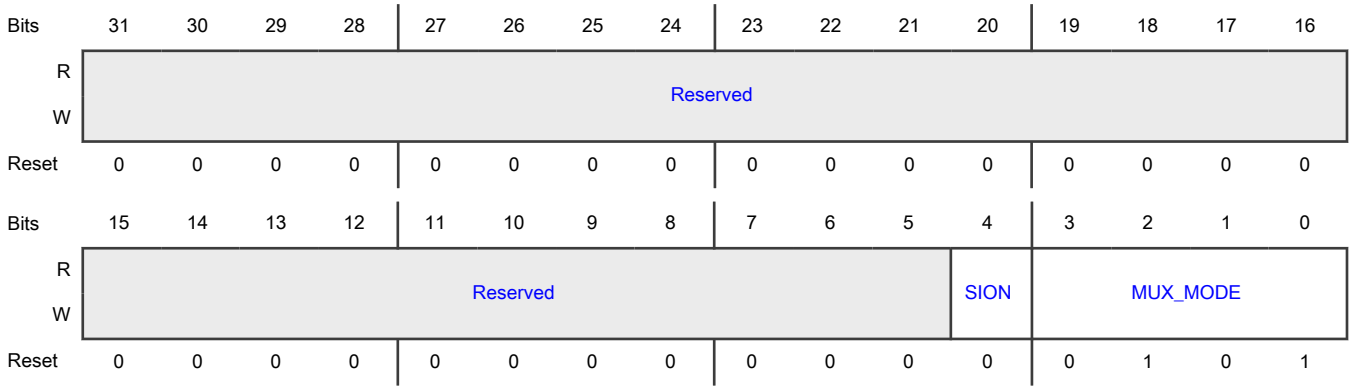
17.4.1.111 SW_MUX_CTL_PAD_GPIO_SD_B2_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_04)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_04	1C4h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_04
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_SD_B2_04. 0000b - Select mux mode: ALT0 mux port: USDHC2_CLK of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_SS1_B of instance: flexspi1_bus2bit 0011b - Select mux mode: ALT3 mux port: QTIMER7_TIMER0 of instance: qtimer7 0100b - Select mux mode: ALT4 mux port: KPP_ROW03 of instance: kpp 0101b - Select mux mode: ALT5 mux port: GPIO5_IO14 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPUART5_RI_B of instance: lpuart5 1000b - Select mux mode: ALT8 mux port: NETC_TMR_PP1 of instance: netc 1001b - Reserved 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: MIC_CLK of instance: mic

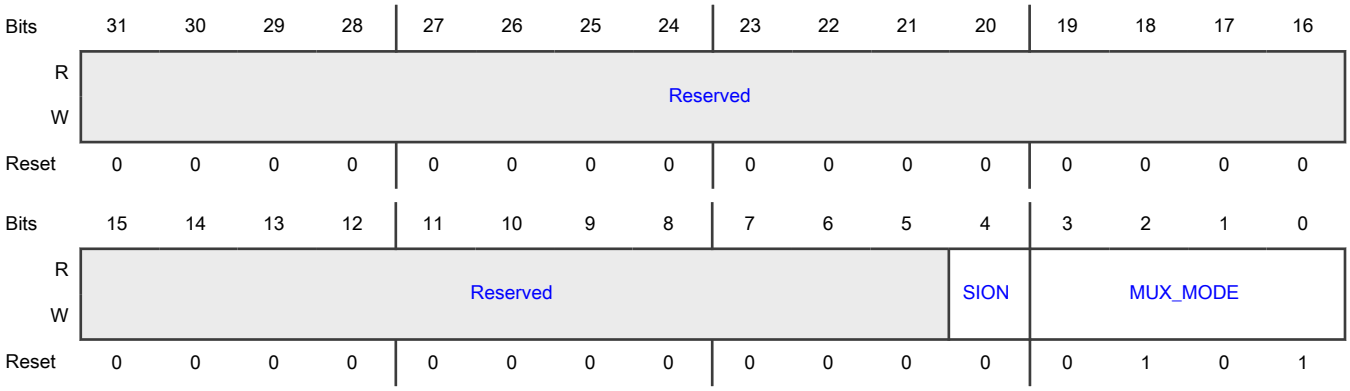
17.4.1.112 SW_MUX_CTL_PAD_GPIO_SD_B2_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_05)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_05	1C8h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_05
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_SD_B2_05. 0000b - Select mux mode: ALT0 mux port: USDHC2_CMD of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_DQS of instance: flexspi1_bus2bit 0010b - Select mux mode: ALT2 mux port: XSPI_SLV_DQS of instance: xspi_slv 0011b - Select mux mode: ALT3 mux port: QTIMER7_TIMER1 of instance: qtimer7

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: LPSPI4_PCS3 of instance: lpspi4 0101b - Select mux mode: ALT5 mux port: GPIO5_IO15 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPUART5_DTR_B of instance: lpuart5 1000b - Select mux mode: ALT8 mux port: NETC_TMR_PP2 of instance: netc 1001b - Reserved 1010b - Reserved 1011b - Reserved

17.4.1.113 SW_MUX_CTL_PAD_GPIO_SD_B2_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_06)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_06	1CCh

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_06
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_SD_B2_06. 0000b - Select mux mode: ALT0 mux port: USDHC2_RESET_B of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_SS0_B of instance: flexspi1_bus2bit 0010b - Select mux mode: ALT2 mux port: XSPI_SLV_CS of instance: xspi_slv 0011b - Select mux mode: ALT3 mux port: QTIMER7_TIMER2 of instance: qtimer7 0100b - Select mux mode: ALT4 mux port: LPSPI4_PCS2 of instance: lpspi4 0101b - Select mux mode: ALT5 mux port: GPIO5_IO16 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPUART5_CTS_B of instance: lpuart5 1000b - Select mux mode: ALT8 mux port: NETC_TMR_PP3 of instance: netc 1001b - Reserved 1010b - Reserved 1011b - Reserved

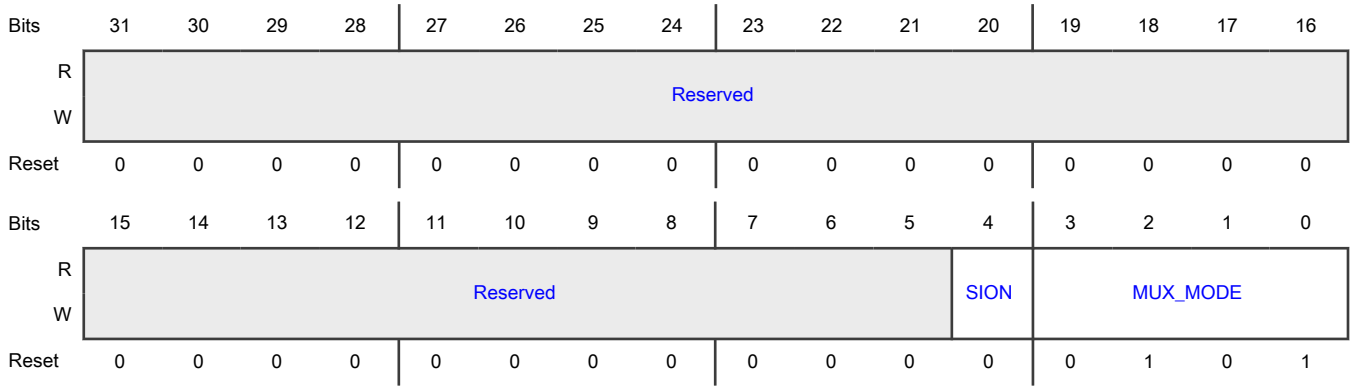
17.4.1.114 SW_MUX_CTL_PAD_GPIO_SD_B2_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_07)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_07	1D0h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_07
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_SD_B2_07. 0000b - Select mux mode: ALT0 mux port: USDHC2_STROBE of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_SCLK of instance: flexspi1_bus2bit 0010b - Select mux mode: ALT2 mux port: XSPI_SLV_CLK of instance: xspi_slv 0011b - Select mux mode: ALT3 mux port: QTIMER7_TIMER3 of instance: qtimer7 0100b - Select mux mode: ALT4 mux port: LPSPI4_PCS1 of instance: lpspi4 0101b - Select mux mode: ALT5 mux port: GPIO5_IO17 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPUART5_RTS_B of instance: lpuart5 1000b - Select mux mode: ALT8 mux port: NETC_TMR_ALARM1 of instance: netc 1001b - Reserved 1010b - Reserved 1011b - Reserved

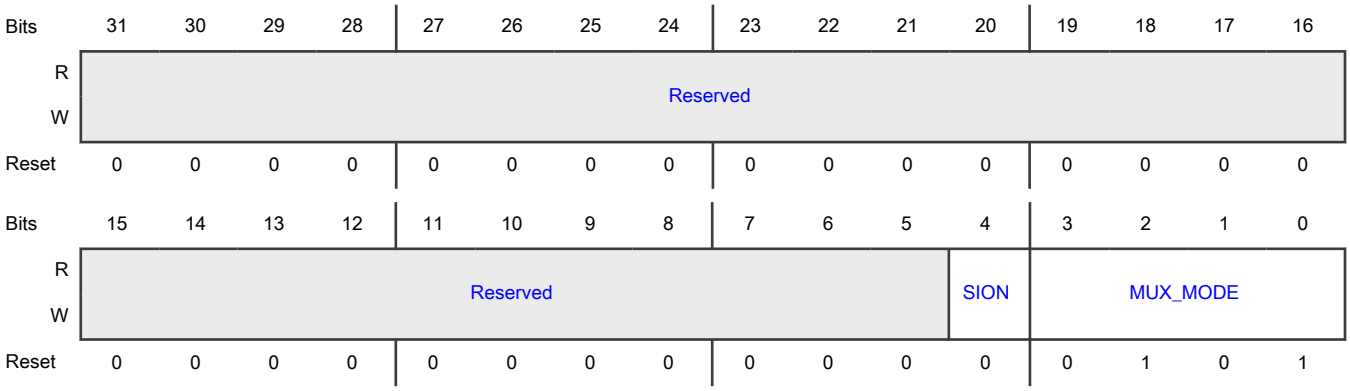
17.4.1.115 SW_MUX_CTL_PAD_GPIO_SD_B2_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_08)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_08	1D4h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_08
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_SD_B2_08. 0000b - Select mux mode: ALT0 mux port: USDHC2_DATA4 of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_DATA00 of instance: flexspi1_bus2bit 0010b - Select mux mode: ALT2 mux port: XSPI_SLV_DATA00 of instance: xspi_slv 0011b - Select mux mode: ALT3 mux port: QTIMER8_TIMER0 of instance: qtimer8

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: LPSPI4_SCK of instance: lpspi4 0101b - Select mux mode: ALT5 mux port: GPIO5_IO18 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPUART5_TX of instance: lpuart5 1000b - Select mux mode: ALT8 mux port: NETC_TMR_ALARM2 of instance: netc 1001b - Select mux mode: ALT9 mux port: NETC_TMR_PP2 of instance: netc 1010b - Reserved 1011b - Reserved

17.4.1.116 SW_MUX_CTL_PAD_GPIO_SD_B2_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_09)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_09	1D8h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_09
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_SD_B2_09. 0000b - Select mux mode: ALT0 mux port: USDHC2_DATA5 of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_DATA01 of instance: flexspi1_bus2bit 0010b - Select mux mode: ALT2 mux port: XSPI_SLV_DATA01 of instance: xspi_slv 0011b - Select mux mode: ALT3 mux port: QTIMER8_TIMER1 of instance: qtimer8 0100b - Select mux mode: ALT4 mux port: LPSPI4_PCS0 of instance: lpspi4 0101b - Select mux mode: ALT5 mux port: GPIO5_IO19 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPUART5_RX of instance: lpuart5 1001b - Select mux mode: ALT9 mux port: NETC_TMR_PP1 of instance: netc 1010b - Reserved 1011b - Reserved

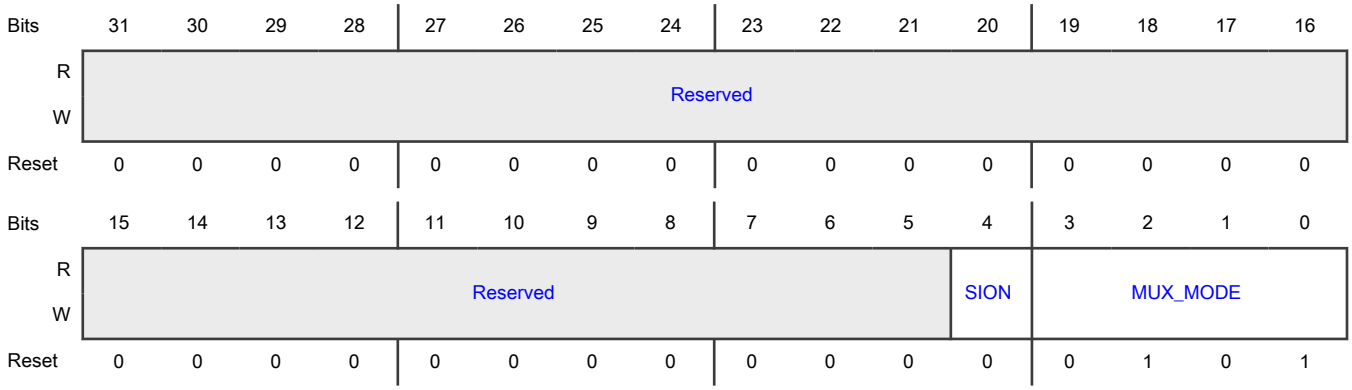
17.4.1.117 SW_MUX_CTL_PAD_GPIO_SD_B2_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_10)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_10	1DCh

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_10
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_SD_B2_10. 0000b - Select mux mode: ALT0 mux port: USDHC2_DATA6 of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_DATA02 of instance: flexspi1_bus2bit 0010b - Select mux mode: ALT2 mux port: XSPI_SLV_DATA02 of instance: xspi_slv 0011b - Select mux mode: ALT3 mux port: QTIMER8_TIMER2 of instance: qtimer8 0100b - Select mux mode: ALT4 mux port: LPSPI4_SDO of instance: lpspi4 0101b - Select mux mode: ALT5 mux port: GPIO5_IO20 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPUART5_DCD_B of instance: lpuart5 1000b - Select mux mode: ALT8 mux port: NETC_TMR_TRIG2 of instance: netc 1001b - Select mux mode: ALT9 mux port: NETC_TMR_PP3 of instance: netc 1010b - Select mux mode: ALT10 mux port: NETC_EMDIO of instance: netc 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_MDIO of instance: ecat

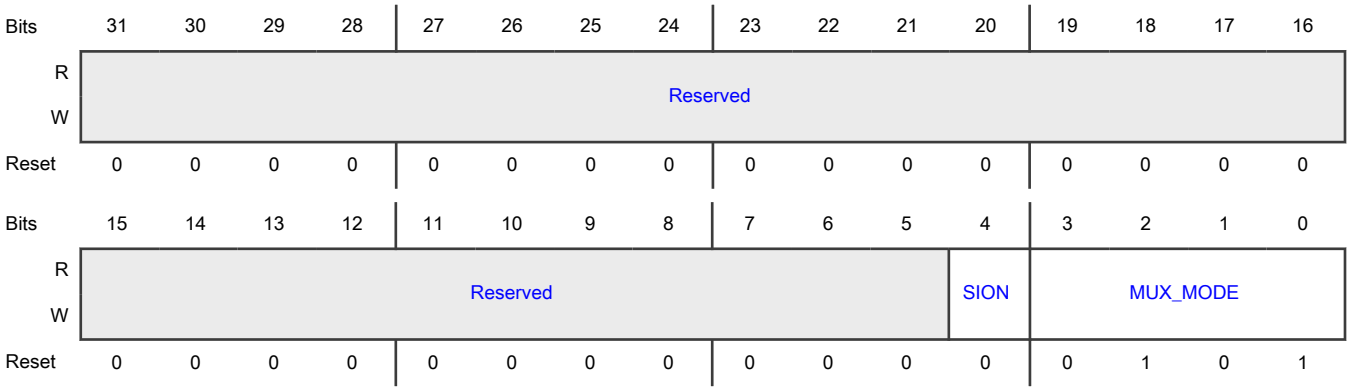
17.4.1.118 SW_MUX_CTL_PAD_GPIO_SD_B2_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_11)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_11	1E0h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_11
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_SD_B2_11. 0000b - Select mux mode: ALT0 mux port: USDHC2_DATA7 of instance: usdhc2 0001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_DATA03 of instance: flexspi1_bus2bit 0010b - Select mux mode: ALT2 mux port: XSPI_SLV_DATA03 of instance: xspi_slv 0011b - Select mux mode: ALT3 mux port: QTIMER8_TIMER3 of instance: qtimer8

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: LPSPI4_SDI of instance: lpspi4 0101b - Select mux mode: ALT5 mux port: GPIO5_IO21 of instance: gpio5 0110b - Select mux mode: ALT6 mux port: LPUART5_DSR_B of instance: lpuart5 0111b - Select mux mode: ALT7 mux port: SFA_ATX_CLK_OUT of instance: sfa 1000b - Select mux mode: ALT8 mux port: NETC_TMR_TRIG1 of instance: netc 1001b - Reserved 1010b - Select mux mode: ALT10 mux port: NETC_EMDC of instance: netc 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_MCLK of instance: ecat

17.4.1.119 SW_MUX_CTL_PAD_GPIO_SD_B2_12_DUMMY SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_SD_B2_12_DUMMY)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_SD_B2_12_DUMMY	1E4h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_SD_B2_12_DUMMY
3 —	- Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: GPIO_SD_B2_12_DUMMY. 000b - Select mux mode: ALT0 mux port: FLEXSPI1_BUS2BIT_A_DQS of instance: flexspi1_bus2bit 001b - Select mux mode: ALT1 mux port: FLEXSPI1_BUS2BIT_B_DQS of instance: flexspi1_bus2bit 101b - Select mux mode: ALT5 mux port: GPIO5_IO22 of instance: gpio5

17.4.1.120 SW_MUX_CTL_PAD_GPIO_B1_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_00)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_00	1E8h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_00
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_B1_00. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_TXD00 of instance: netc_pinmux 0001b - Select mux mode: ALT1 mux port: ADC2_CONV_D00 of instance: adc2 0010b - Select mux mode: ALT2 mux port: SEMC_CSX01 of instance: semc 0011b - Select mux mode: ALT3 mux port: QTIMER1_TIMER0 of instance: qtimer1 0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT26 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO00 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: TPM5_CH00 of instance: tpm5 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_TXD00 of instance: netc_pinmux 1010b - Reserved 1011b - Reserved

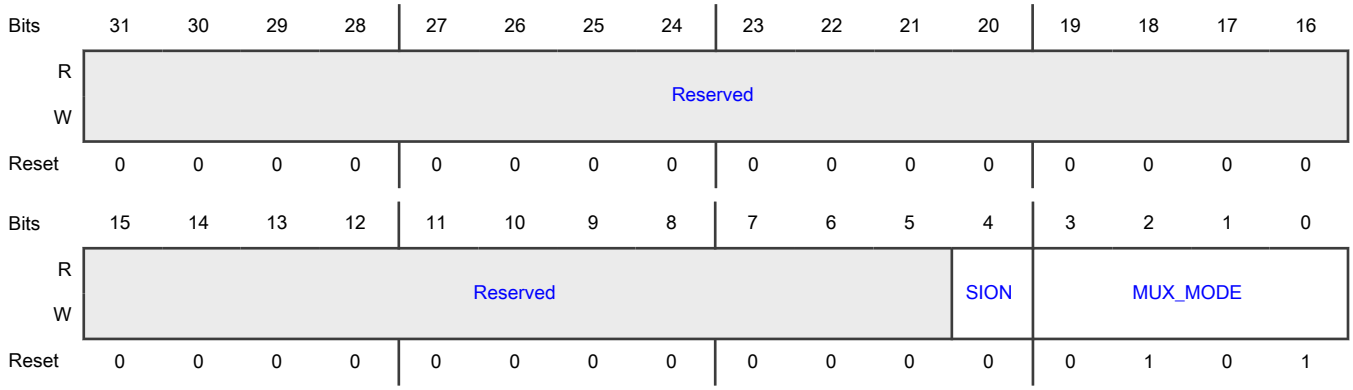
17.4.1.121 SW_MUX_CTL_PAD_GPIO_B1_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_01)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_01	1ECh

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_01
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_B1_01. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_TXD01 of instance: netc_pinmux 0001b - Select mux mode: ALT1 mux port: ADC2_CONV_D01 of instance: adc2 0010b - Select mux mode: ALT2 mux port: SEMC_CSX02 of instance: semc 0011b - Select mux mode: ALT3 mux port: QTIMER1_TIMER1 of instance: qtimer1 0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT27 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO01 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: TPM5_CH01 of instance: tpm5 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_TXD01 of instance: netc_pinmux 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_RX_DATA00 of instance: sai4

17.4.1.122 SW_MUX_CTL_PAD_GPIO_B1_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_02)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_02	1F0h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_02
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_B1_02. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_TX_EN of instance: netc_pinmux 0001b - Select mux mode: ALT1 mux port: ADC2_CONV_D02 of instance: adc2 0010b - Select mux mode: ALT2 mux port: LPI2C6_SCL of instance: lpi2c6 0011b - Select mux mode: ALT3 mux port: QTIMER1_TIMER2 of instance: qtimer1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT28 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO02 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: TPM5_CH02 of instance: tpm5 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_SS1_B of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_TX_EN of instance: netc_pinmux 1001b - Select mux mode: ALT9 mux port: LPUART11_TX of instance: lpuart11 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_RX_DATA01 of instance: sai4

17.4.1.123 SW_MUX_CTL_PAD_GPIO_B1_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_03)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_03	1F4h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_03
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_B1_03. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_TX_CLK of instance: netc_pinmux 0001b - Select mux mode: ALT1 mux port: ADC2_CONV_D03 of instance: adc2 0010b - Select mux mode: ALT2 mux port: LPI2C6_SDA of instance: lpi2c6 0011b - Select mux mode: ALT3 mux port: QTIMER2_TIMER0 of instance: qtimer2 0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT29 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO03 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: TPM5_CH03 of instance: tpm5 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_DQS of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_TX_CLK of instance: netc_pinmux 1001b - Select mux mode: ALT9 mux port: LPUART11_RX of instance: lpuart11 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_RX_DATA02 of instance: sai4

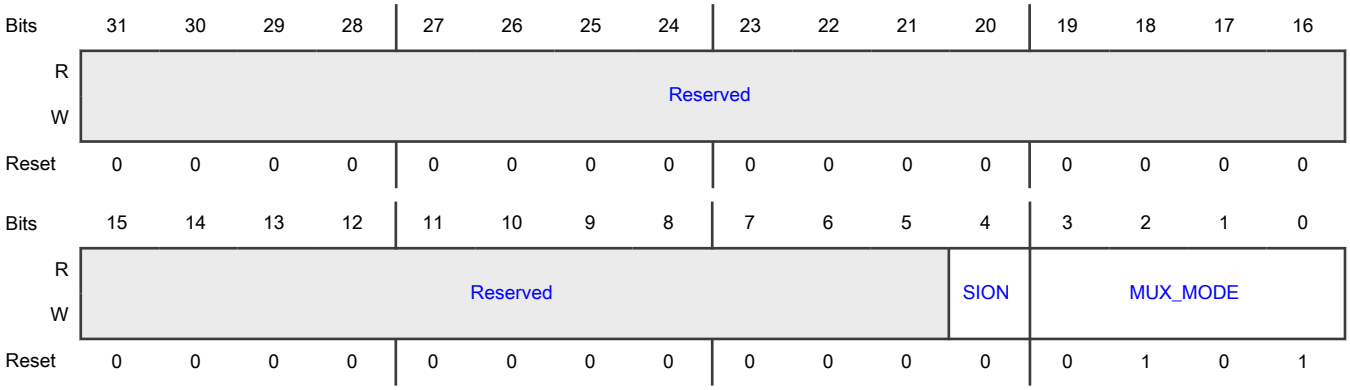
17.4.1.124 SW_MUX_CTL_PAD_GPIO_B1_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_04)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_04	1F8h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_04
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_B1_04. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_RXD00 of instance: netc_pinmux 0001b - Select mux mode: ALT1 mux port: ADC2_CONV_D04 of instance: adc2 0010b - Select mux mode: ALT2 mux port: LPUART9_RX of instance: lpuart9 0011b - Select mux mode: ALT3 mux port: QTIMER2_TIMER1 of instance: qtimer2 0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT30 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO04 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: TPM5_EXTCLK of instance: tpm5 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_SS0_B of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_RXD00 of instance: netc_pinmux 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_RX_DATA03 of instance: sai4

17.4.1.125 SW_MUX_CTL_PAD_GPIO_B1_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_05)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_05	1FCh

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_05
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_B1_05. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_RXD01 of instance: netc_pinmux 0001b - Select mux mode: ALT1 mux port: ADC2_CONV_D05 of instance: adc2 0010b - Select mux mode: ALT2 mux port: LPUART9_CTS_B of instance: lpuart9 0011b - Select mux mode: ALT3 mux port: QTIMER2_TIMER2 of instance: qtimer2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT31 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO05 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: TPM6_EXTCLK of instance: tpm6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_SCLK of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_RXD01 of instance: netc_pinmux 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_MCLK of instance: sai4

17.4.1.126 SW_MUX_CTL_PAD_GPIO_B1_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_06)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_06	200h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_06
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_B1_06. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_RX_DV of instance: netc_pinmux 0001b - Select mux mode: ALT1 mux port: ADC2_CONV_D06 of instance: adc2 0010b - Select mux mode: ALT2 mux port: LPUART9_TX of instance: lpuart9 0011b - Select mux mode: ALT3 mux port: QTIMER3_TIMER0 of instance: qtimer3 0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT32 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO06 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: TPM6_CH00 of instance: tpm6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_DATA07 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_RX_DV of instance: netc_pinmux 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_RX_BCLK of instance: sai4

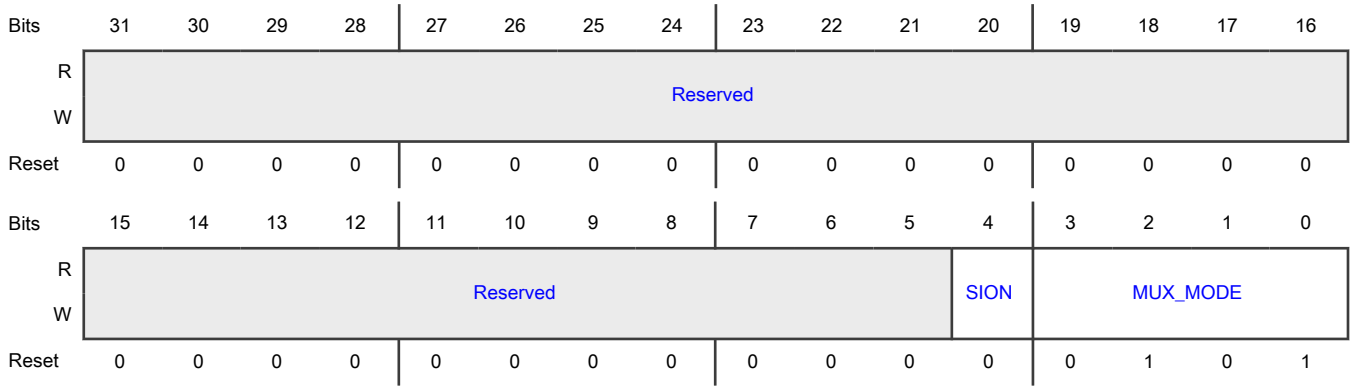
17.4.1.127 SW_MUX_CTL_PAD_GPIO_B1_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_07)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_07	204h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_07
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_B1_07. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_TXD02 of instance: netc_pinmux 0001b - Select mux mode: ALT1 mux port: ADC2_CONV_D07 of instance: adc2 0010b - Select mux mode: ALT2 mux port: LPUART9_RTS_B of instance: lpuart9 0011b - Select mux mode: ALT3 mux port: QTIMER3_TIMER1 of instance: qtimer3 0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT33 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO07 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: TPM6_CH01 of instance: tpm6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_DATA06 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_TXD02 of instance: netc_pinmux 1001b - Select mux mode: ALT9 mux port: LPSPI6_SDI of instance: lpspi6 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_RX_SYNC of instance: sai4

17.4.1.128 SW_MUX_CTL_PAD_GPIO_B1_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_08)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_08	208h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved											SION	MUX_MODE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_08
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_B1_08. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_TXD03 of instance: netc_pinmux 0001b - Select mux mode: ALT1 mux port: ADC2_CONV_RDY_CLK of instance: adc2 0010b - Select mux mode: ALT2 mux port: USDHC1_CD_B of instance: usdhc1 0011b - Select mux mode: ALT3 mux port: QTIMER3_TIMER2 of instance: qtimer3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT36 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO08 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: TPM6_CH02 of instance: tpm6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_DATA05 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_TXD03 of instance: netc_pinmux 1001b - Select mux mode: ALT9 mux port: LPSPI6_SDO of instance: lpspi6 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_TX_BCLK of instance: sai4

17.4.1.129 SW_MUX_CTL_PAD_GPIO_B1_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_09)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_09	20Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_09
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_B1_09. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_RXD02 of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: USDHC1_WP of instance: usdhc1 0011b - Select mux mode: ALT3 mux port: QTIMER4_TIMER0 of instance: qtimer4 0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT37 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO09 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: TPM6_CH03 of instance: tpm6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_DATA04 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_RXD02 of instance: netc_pinmux 1001b - Select mux mode: ALT9 mux port: LPSPI6_PCS1 of instance: lpspi6 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_TX_SYNC of instance: sai4

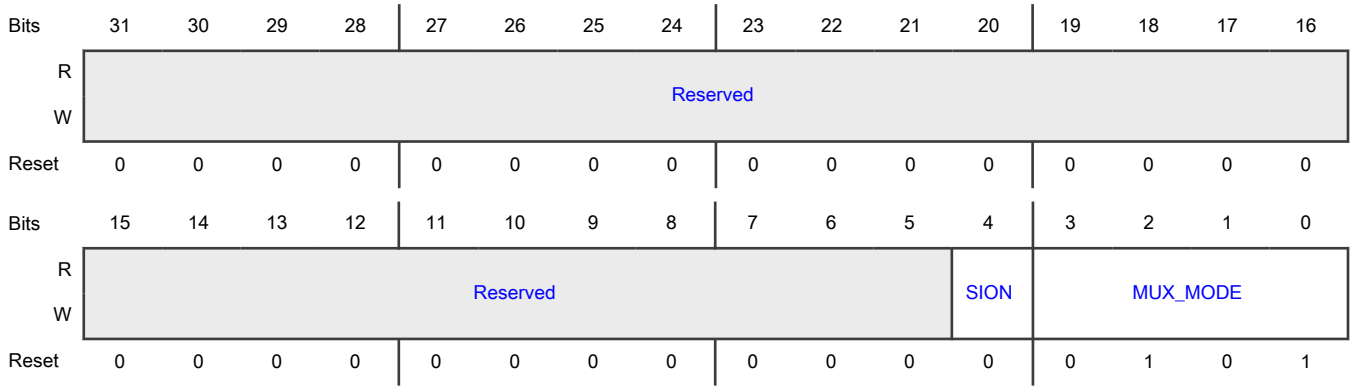
17.4.1.130 SW_MUX_CTL_PAD_GPIO_B1_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_10)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_10	210h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_10
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_B1_10. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_RXD03 of instance: netc_pinmux 0010b - Select mux mode: ALT2 mux port: USDHC1_RESET_B of instance: usdhc1 0011b - Select mux mode: ALT3 mux port: QTIMER4_TIMER1 of instance: qtimer4 0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT34 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO10 of instance: gpio6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_DATA03 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_RXD03 of instance: netc_pinmux 1001b - Select mux mode: ALT9 mux port: LPSPI6_PCS2 of instance: lpspi6 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_TX_DATA00 of instance: sai4

17.4.1.131 SW_MUX_CTL_PAD_GPIO_B1_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_11)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_11	214h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_11
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_B1_11. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_RX_CLK of instance: netc_pinmux 0011b - Select mux mode: ALT3 mux port: QTIMER4_TIMER2 of instance: qtimer4 0100b - Select mux mode: ALT4 mux port: XBAR1_XBAR_INOUT35 of instance: xbar1 0101b - Select mux mode: ALT5 mux port: GPIO6_IO11 of instance: gpio6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_DATA02 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_RX_CLK of instance: netc_pinmux 1001b - Select mux mode: ALT9 mux port: LPSPI6_PCS3 of instance: lpspi6 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_TX_DATA01 of instance: sai4

17.4.1.132 SW_MUX_CTL_PAD_GPIO_B1_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_12)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_12	218h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4	Software Input On Field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SION	Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_12
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_B1_12. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_RX_ER of instance: netc_pinmux 0001b - Select mux mode: ALT1 mux port: NETC_EMDIO of instance: netc 0101b - Select mux mode: ALT5 mux port: GPIO6_IO12 of instance: gpio6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_DATA01 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_RX_ER of instance: netc_pinmux 1001b - Select mux mode: ALT9 mux port: LPSPI6_PCS0 of instance: lpspi6 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_TX_DATA02 of instance: sai4

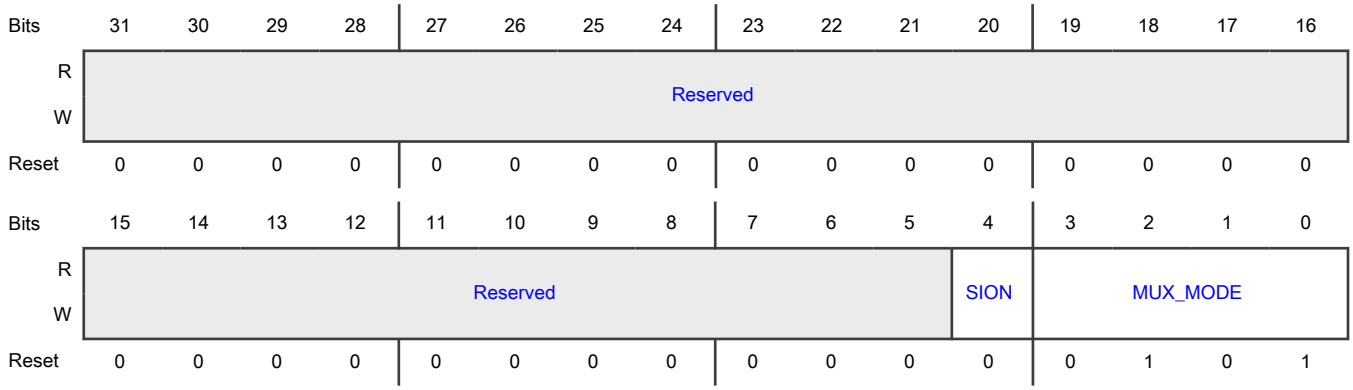
17.4.1.133 SW_MUX_CTL_PAD_GPIO_B1_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B1_13)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B1_13	21Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B1_13
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_B1_13. 0000b - Select mux mode: ALT0 mux port: NETC_PINMUX_ETH1_TX_ER of instance: netc_pinmux 0001b - Select mux mode: ALT1 mux port: NETC_EMDC of instance: netc 0010b - Select mux mode: ALT2 mux port: USDHC1_VSELECT of instance: usdhc1 0011b - Select mux mode: ALT3 mux port: CCM_ENET_REF_CLK_25M of instance: ccm 0101b - Select mux mode: ALT5 mux port: GPIO6_IO13 of instance: gpio6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_B_DATA00 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: NETC_PINMUX_ETH4_TX_ER of instance: netc_pinmux 1001b - Select mux mode: ALT9 mux port: LPSPI6_SCK of instance: lpspi6 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: SAI4_TX_DATA03 of instance: sai4

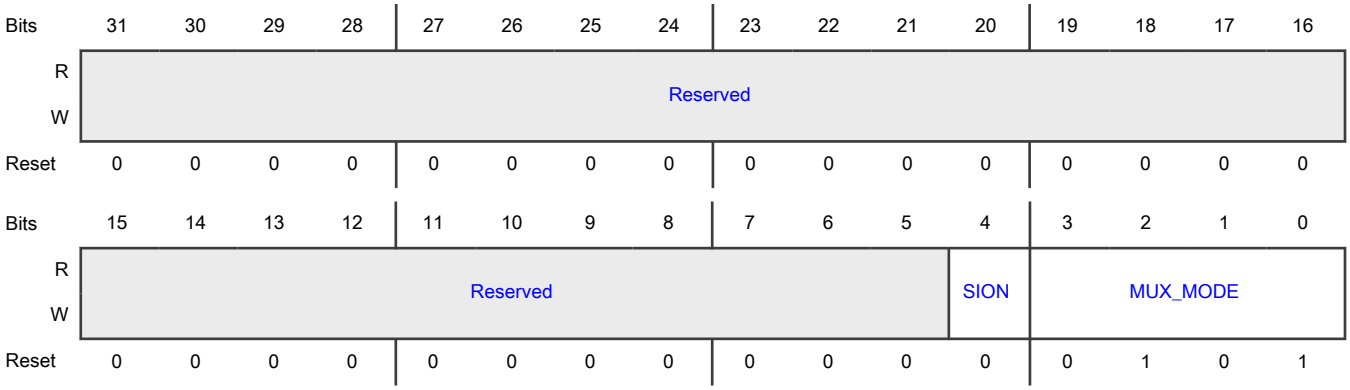
17.4.1.134 SW_MUX_CTL_PAD_GPIO_B2_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_00)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_00	220h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_00
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_B2_00. 0000b - Select mux mode: ALT0 mux port: NETC_ETH1_CRS of instance: netc 0001b - Select mux mode: ALT1 mux port: SEMC_CSX03 of instance: semc 0010b - Select mux mode: ALT2 mux port: LPIT3_TRIGGER00 of instance: lpit3 0100b - Select mux mode: ALT4 mux port: SAI4_MCLK of instance: sai4 0101b - Select mux mode: ALT5 mux port: GPIO6_IO14 of instance: gpio6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0111b - Reserved 1000b - Select mux mode: ALT8 mux port: NETC_ETH4_CRS of instance: netc 1001b - Select mux mode: ALT9 mux port: LPSPI6_SDI of instance: lpspi6 1010b - Select mux mode: ALT10 mux port: NETC_ETH2_SLV_MDIO of instance: netc 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_CLK_ECAT_CLK25 of instance: ecat

17.4.1.135 SW_MUX_CTL_PAD_GPIO_B2_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_01)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_01	224h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_01
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_B2_01. 0000b - Select mux mode: ALT0 mux port: NETC_ETH1_COL of instance: netc 0010b - Select mux mode: ALT2 mux port: LPIT3_TRIGGER01 of instance: lpit3 0100b - Select mux mode: ALT4 mux port: SAI4_TX_BCLK of instance: sai4 0101b - Select mux mode: ALT5 mux port: GPIO6_IO15 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: FLEXSPI1_BUS2BIT_A_SS1_B of instance: flexspi1_bus2bit 0111b - Reserved 1000b - Select mux mode: ALT8 mux port: NETC_ETH4_COL of instance: netc 1001b - Select mux mode: ALT9 mux port: LPSPI6_SDO of instance: lpspi6 1010b - Select mux mode: ALT10 mux port: NETC_ETH2_SLV_MDC of instance: netc 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_RX_ER of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_RX_ER_1 of instance: ecat

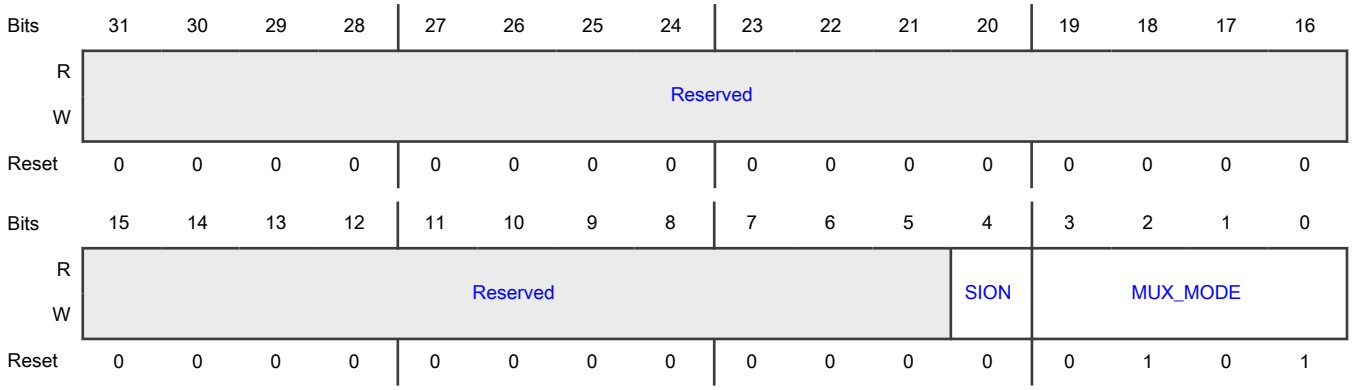
17.4.1.136 SW_MUX_CTL_PAD_GPIO_B2_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_02)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_02	228h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_02
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_B2_02. 0010b - Select mux mode: ALT2 mux port: LPIT3_TRIGGER02 of instance: lpit3 0011b - Select mux mode: ALT3 mux port: NETC_EMDIO of instance: netc 0100b - Select mux mode: ALT4 mux port: SAI4_TX_SYNC of instance: sai4 0101b - Select mux mode: ALT5 mux port: GPIO6_IO16 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: FLEXSPI1_BUS2BIT_B_SCLK of instance: flexspi1_bus2bit 0111b - Reserved 1001b - Select mux mode: ALT9 mux port: CCM_ENET_REF_CLK_25M of instance: ccm 1010b - Select mux mode: ALT10 mux port: EWM_EWM_OUT_B of instance: ewm 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_RXD02 of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DATA2_1 of instance: ecat

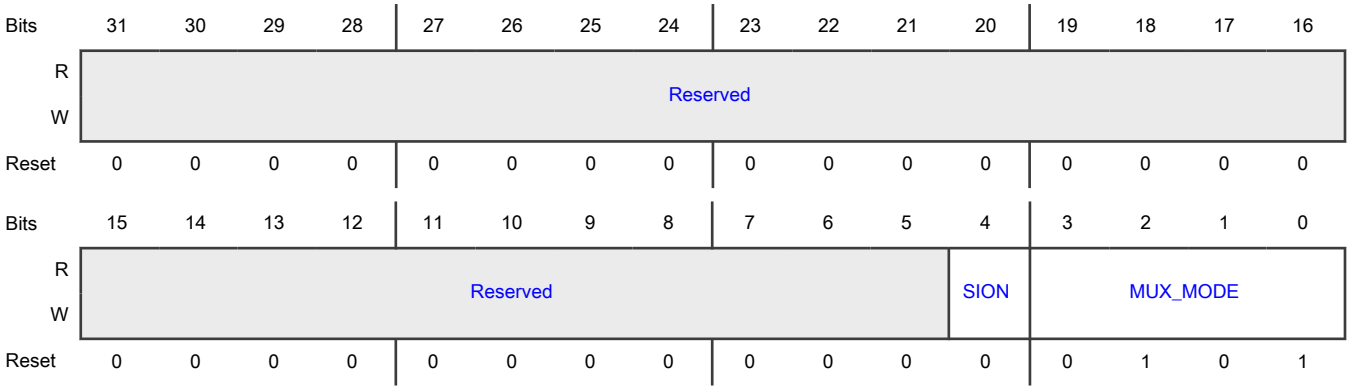
17.4.1.137 SW_MUX_CTL_PAD_GPIO_B2_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_03)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_03	22Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_03
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_B2_03. 0010b - Select mux mode: ALT2 mux port: LPIT3_TRIGGER03 of instance: lpit3 0011b - Select mux mode: ALT3 mux port: NETC_EMDC of instance: netc 0100b - Select mux mode: ALT4 mux port: SAI4_TX_DATA00 of instance: sai4 0101b - Select mux mode: ALT5 mux port: GPIO6_IO17 of instance: gpio6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_A_DATA04 of instance: flexspi1_bus2bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1001b - Reserved 1010b - Select mux mode: ALT10 mux port: XSPI_SLV_DATA04 of instance: xspi_slv 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_RXD03 of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DATA3_1 of instance: ecat

17.4.1.138 SW_MUX_CTL_PAD_GPIO_B2_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_04)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_04	230h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Force input path of pad GPIO_B2_04
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_B2_04. 0000b - Select mux mode: ALT0 mux port: SINC1_MOD_CLK0 of instance: sinc1 0001b - Select mux mode: ALT1 mux port: SINC2_MOD_CLK0 of instance: sinc2 0010b - Select mux mode: ALT2 mux port: SINC3_MOD_CLK0 of instance: sinc3 0100b - Select mux mode: ALT4 mux port: SAI4_RX_SYNC of instance: sai4 0101b - Select mux mode: ALT5 mux port: GPIO6_IO18 of instance: gpio6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_A_DATA05 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: TPM3_EXTCLK of instance: tpm3 1001b - Reserved 1010b - Select mux mode: ALT10 mux port: XSPI_SLV_DATA05 of instance: xspi_slv 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_TXD02 of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_TX_DATA2_1 of instance: ecat

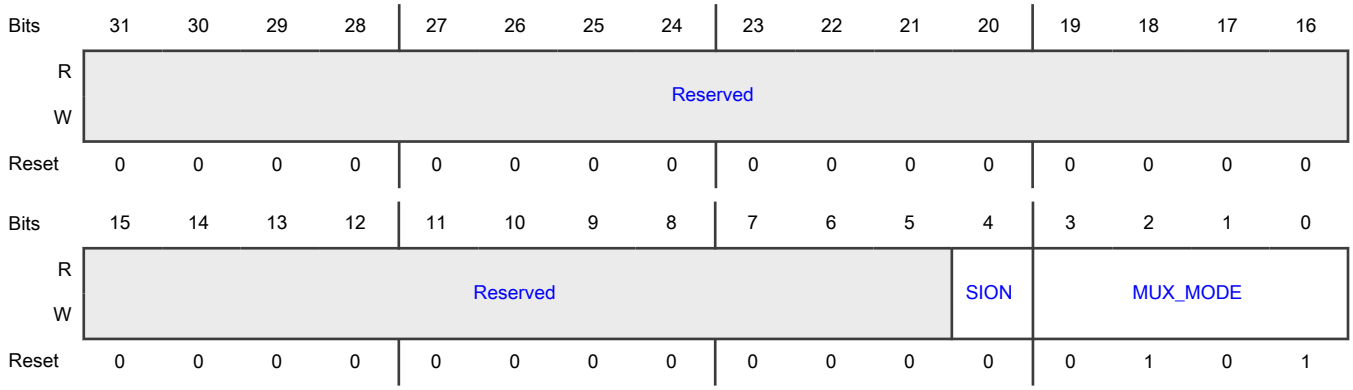
17.4.1.139 SW_MUX_CTL_PAD_GPIO_B2_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_05)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_05	234h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_05
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_B2_05. 0000b - Select mux mode: ALT0 mux port: SINC1_MOD_CLK1 of instance: sinc1 0001b - Select mux mode: ALT1 mux port: SINC2_MOD_CLK1 of instance: sinc2 0010b - Select mux mode: ALT2 mux port: SINC3_MOD_CLK1 of instance: sinc3 0100b - Select mux mode: ALT4 mux port: SAI4_RX_BCLK of instance: sai4 0101b - Select mux mode: ALT5 mux port: GPIO6_IO19 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: MIC_CLK of instance: mic 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_A_DATA06 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: TPM3_CH00 of instance: tpm3 1001b - Reserved 1010b - Select mux mode: ALT10 mux port: XSPI_SLV_DATA06 of instance: xspi_slv 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_TXD03 of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_TX_DATA3_1 of instance: ecat

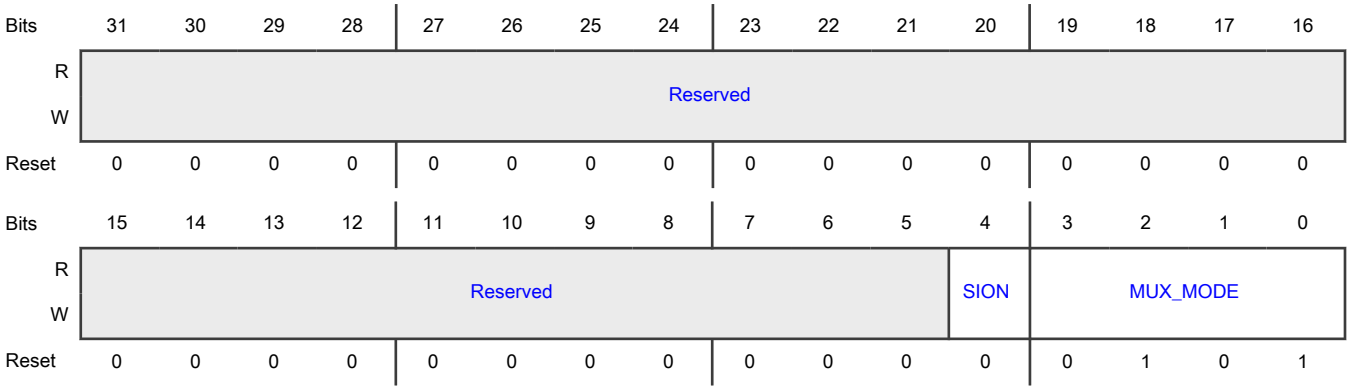
17.4.1.140 SW_MUX_CTL_PAD_GPIO_B2_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_06)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_06	238h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_06
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_B2_06. 0000b - Select mux mode: ALT0 mux port: SINC1_MOD_CLK2 of instance: sinc1 0001b - Select mux mode: ALT1 mux port: SINC2_MOD_CLK2 of instance: sinc2 0010b - Select mux mode: ALT2 mux port: SINC3_MOD_CLK2 of instance: sinc3 0011b - Select mux mode: ALT3 mux port: LPUART6_DSR_B of instance: lpuart6 0100b - Select mux mode: ALT4 mux port: SAI4_RX_DATA00 of instance: sai4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Select mux mode: ALT5 mux port: GPIO6_IO20 of instance: gpio6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_A_DATA07 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: TPM3_CH01 of instance: tpm3 1001b - Select mux mode: ALT9 mux port: LPUART11_TX of instance: lpuart11 1010b - Select mux mode: ALT10 mux port: XSPI_SLV_DATA07 of instance: xspi_slv 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_TXD00 of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_TX_DATA0_1 of instance: ecat

17.4.1.141 SW_MUX_CTL_PAD_GPIO_B2_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_07)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_07	23Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_07
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_B2_07. 0000b - Select mux mode: ALT0 mux port: QTIMER5_TIMER0 of instance: qtimer5 0011b - Select mux mode: ALT3 mux port: LPUART6_DCD_B of instance: lpuart6 0100b - Select mux mode: ALT4 mux port: SAI4_TX_DATA01 of instance: sai4 0101b - Select mux mode: ALT5 mux port: GPIO6_IO21 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: SAI4_RX_DATA01 of instance: sai4 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_A_DQS of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: TPM3_CH02 of instance: tpm3 1001b - Select mux mode: ALT9 mux port: LPUART11_RX of instance: lpuart11 1010b - Select mux mode: ALT10 mux port: XSPI_SLV_DQS of instance: xspi_slv 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_TXD01 of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_TX_DATA1_1 of instance: ecat

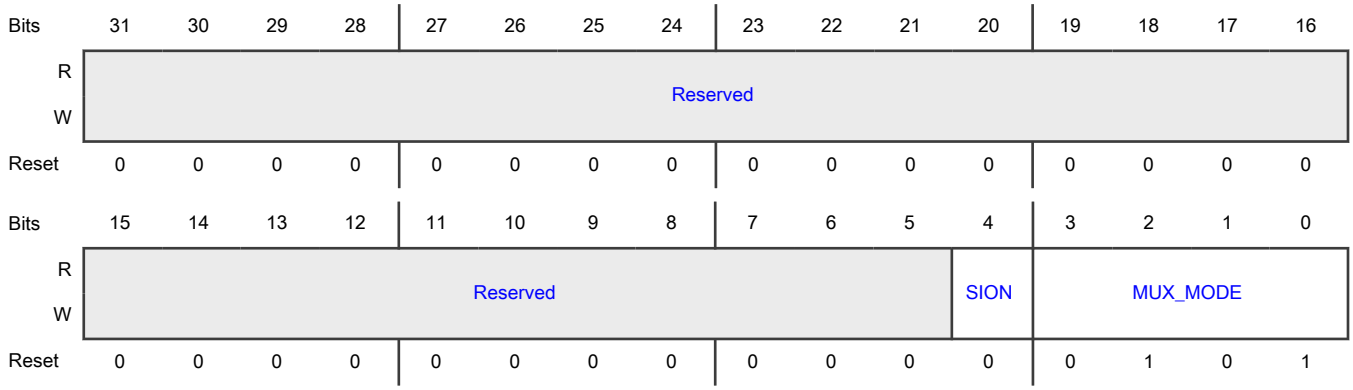
17.4.1.142 SW_MUX_CTL_PAD_GPIO_B2_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_08)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_08	240h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_08
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_B2_08. 0000b - Select mux mode: ALT0 mux port: QTIMER5_TIMER1 of instance: qtimer5 0001b - Select mux mode: ALT1 mux port: SINC2_EMCLK02 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: LPUART6_RI_B of instance: lpuart6 0101b - Select mux mode: ALT5 mux port: GPIO6_IO22 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: LPI2C6_SCL of instance: lpi2c6 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_A_SCLK of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: TPM3_CH03 of instance: tpm3 1001b - Select mux mode: ALT9 mux port: SPDIF_IN of instance: spdif 1010b - Select mux mode: ALT10 mux port: XSPI_SLV_CLK of instance: xspi_slv 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_TX_EN of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_TX_EN_1 of instance: ecat

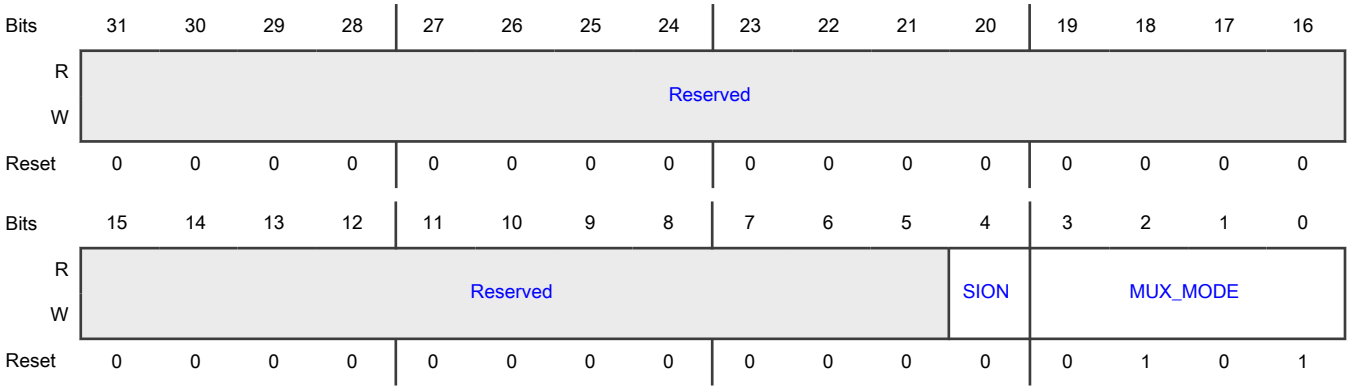
17.4.1.143 SW_MUX_CTL_PAD_GPIO_B2_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_09)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_09	244h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_09
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_B2_09. 0000b - Select mux mode: ALT0 mux port: QTIMER5_TIMER2 of instance: qtimer5 0001b - Select mux mode: ALT1 mux port: SINC2_EMBIT02 of instance: sinc2 0100b - Select mux mode: ALT4 mux port: LPUART6_DTR_B of instance: lpuart6 0101b - Select mux mode: ALT5 mux port: GPIO6_IO23 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: LPI2C6_SDA of instance: lpi2c6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_A_SS0_B of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: TPM4_EXTCLK of instance: tpm4 1001b - Select mux mode: ALT9 mux port: SPDIF_OUT of instance: spdif 1010b - Select mux mode: ALT10 mux port: XSPI_SLV_CS of instance: xspi_slv 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_TX_CLK of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_TX_CLK_1 of instance: ecat

17.4.1.144 SW_MUX_CTL_PAD_GPIO_B2_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_10)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_10	248h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4	Software Input On Field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SION	Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_10
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_B2_10. 0000b - Select mux mode: ALT0 mux port: MIC_BITSTREAM00 of instance: mic 0001b - Select mux mode: ALT1 mux port: SINC2_EMCLK03 of instance: sinc2 0010b - Select mux mode: ALT2 mux port: CAN3_TX of instance: can3 0011b - Select mux mode: ALT3 mux port: LPUART8_CTS_B of instance: lpuart8 0100b - Select mux mode: ALT4 mux port: LPUART6_TX of instance: lpuart6 0101b - Select mux mode: ALT5 mux port: GPIO6_IO24 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: LPI2C4_SCL of instance: lpi2c4 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_A_DATA00 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: TPM4_CH00 of instance: tpm4 1001b - Select mux mode: ALT9 mux port: LPSPI4_SCK of instance: lpspi4 1010b - Select mux mode: ALT10 mux port: XSPI_SLV_DATA00 of instance: xspi_slv 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_RXD00 of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DATA0_1 of instance: ecat

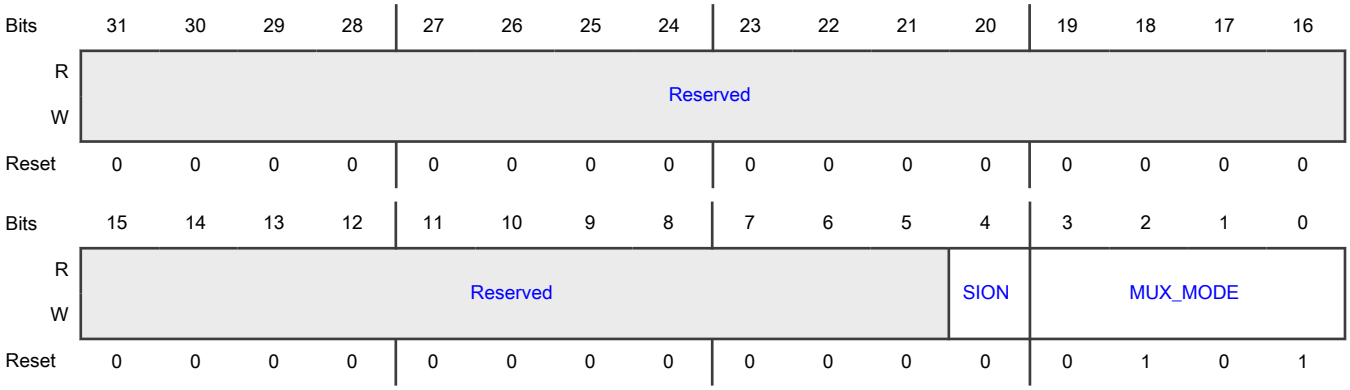
17.4.1.145 SW_MUX_CTL_PAD_GPIO_B2_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_11)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_11	24Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_11
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 13 iomux modes to be used for pad: GPIO_B2_11. 0000b - Select mux mode: ALT0 mux port: MIC_BITSTREAM01 of instance: mic 0001b - Select mux mode: ALT1 mux port: SINC2_EMBIT03 of instance: sinc2 0010b - Select mux mode: ALT2 mux port: CAN3_RX of instance: can3 0011b - Select mux mode: ALT3 mux port: LPUART8_RTS_B of instance: lpuart8 0100b - Select mux mode: ALT4 mux port: LPUART6_RX of instance: lpuart6 0101b - Select mux mode: ALT5 mux port: GPIO6_IO25 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: LPI2C4_SDA of instance: lpi2c4 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_A_DATA01 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: TPM4_CH01 of instance: tpm4 1001b - Select mux mode: ALT9 mux port: LPSPI4_SDI of instance: lpspi4 1010b - Select mux mode: ALT10 mux port: XSPI_SLV_DATA01 of instance: xspi_slv 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_RXD01 of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DATA1_1 of instance: ecat

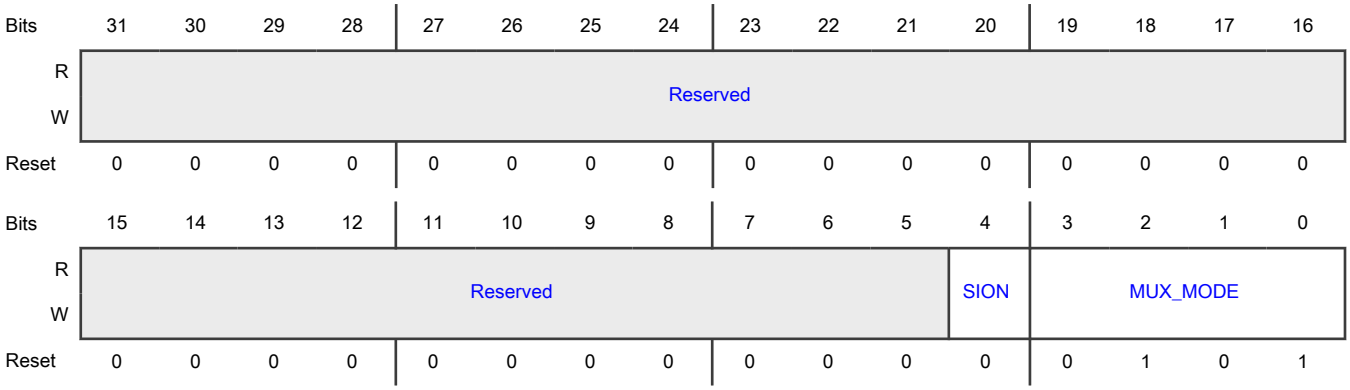
17.4.1.146 SW_MUX_CTL_PAD_GPIO_B2_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_12)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_12	250h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_12
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_B2_12. 0000b - Select mux mode: ALT0 mux port: MIC_BITSTREAM02 of instance: mic 0001b - Select mux mode: ALT1 mux port: SINC_FILTER_GLUE2_BREAK of instance: sinc_filter_glue2 0010b - Select mux mode: ALT2 mux port: LPUART8_TX of instance: lpuart8 0100b - Select mux mode: ALT4 mux port: LPUART6_CTS_B of instance: lpuart6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Select mux mode: ALT5 mux port: GPIO6_IO26 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: CAN3_TX of instance: can3 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_A_DATA02 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: TPM4_CH02 of instance: tpm4 1001b - Select mux mode: ALT9 mux port: LPSPI4_SDO of instance: lpspi4 1010b - Select mux mode: ALT10 mux port: XSPI_SLV_DATA02 of instance: xspi_slv 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_RX_DV of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_RX_DV_1 of instance: ecat

17.4.1.147 SW_MUX_CTL_PAD_GPIO_B2_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_B2_13)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_B2_13	254h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_B2_13
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_B2_13. 0000b - Select mux mode: ALT0 mux port: MIC_BITSTREAM03 of instance: mic 0001b - Select mux mode: ALT1 mux port: SINC2_EMCLK00 of instance: sinc2 0010b - Select mux mode: ALT2 mux port: LPUART8_RX of instance: lpuart8 0100b - Select mux mode: ALT4 mux port: LPUART6_RTS_B of instance: lpuart6 0101b - Select mux mode: ALT5 mux port: GPIO6_IO27 of instance: gpio6 0110b - Select mux mode: ALT6 mux port: CAN3_RX of instance: can3 0111b - Select mux mode: ALT7 mux port: FLEXSPI1_BUS2BIT_A_DATA03 of instance: flexspi1_bus2bit 1000b - Select mux mode: ALT8 mux port: TPM4_CH03 of instance: tpm4 1001b - Select mux mode: ALT9 mux port: LPSPI4_PCS0 of instance: lpspi4 1010b - Select mux mode: ALT10 mux port: XSPI_SLV_DATA03 of instance: xspi_slv 1011b - Select mux mode: ALT11 mux port: NETC_PINMUX_ETH2_RX_CLK of instance: netc_pinmux 1100b - Select mux mode: ALT12 mux port: ECAT_RX_CLK_1 of instance: ecat

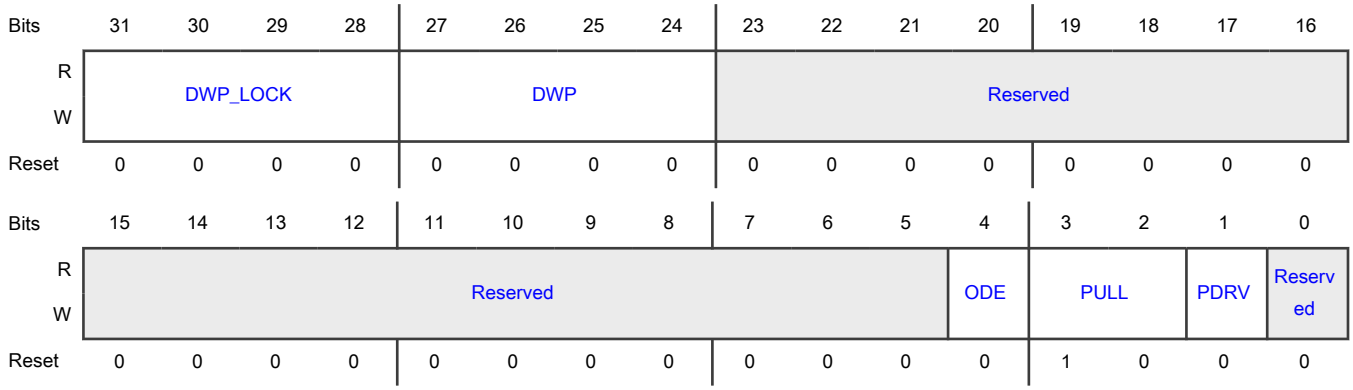
17.4.1.148 SW_PAD_CTL_PAD_GPIO_EMC_B1_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_00)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_00	258h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_00 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_00 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_00

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

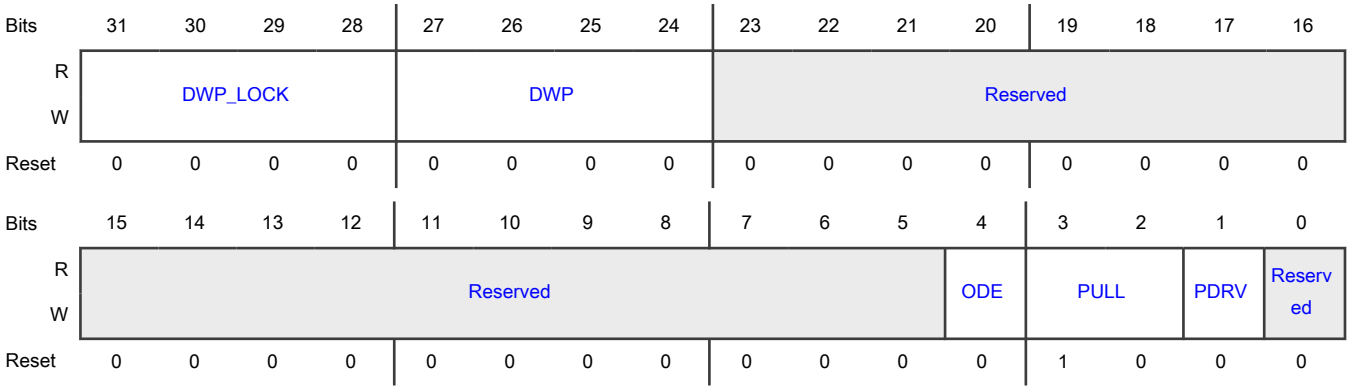
17.4.1.149 SW_PAD_CTL_PAD_GPIO_EMC_B1_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_01)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_01	25Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_01 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_01 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_01 0b - high driver 1b - normal driver
0 —	- Reserved

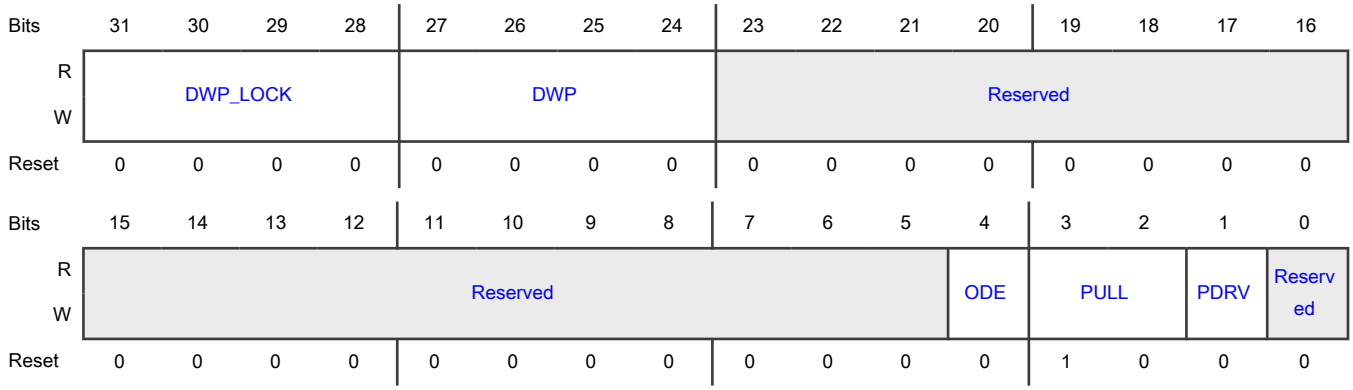
17.4.1.150 SW_PAD_CTL_PAD_GPIO_EMC_B1_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_02)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_02	260h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_02 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_02 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_02

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

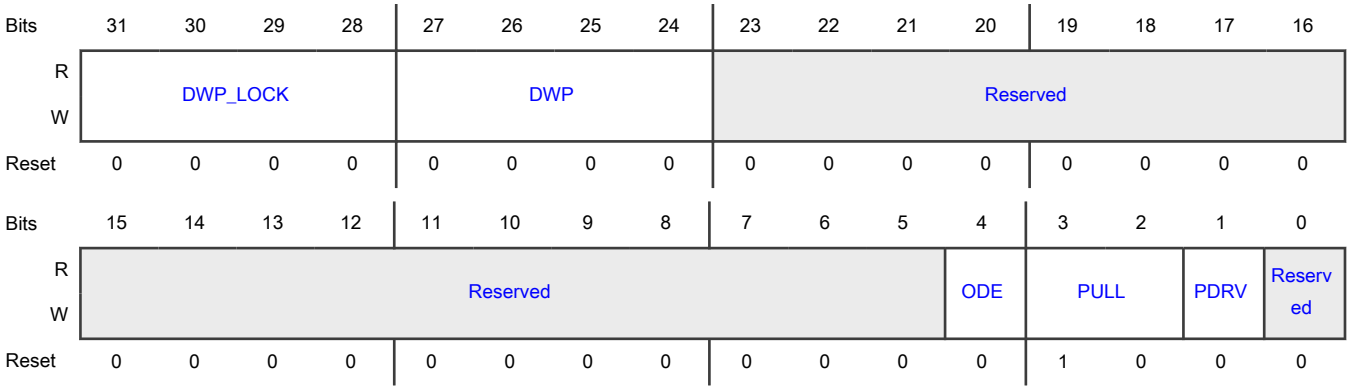
17.4.1.151 SW_PAD_CTL_PAD_GPIO_EMC_B1_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_03)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_03	264h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_03 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_03 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_03 0b - high driver 1b - normal driver
0 —	- Reserved

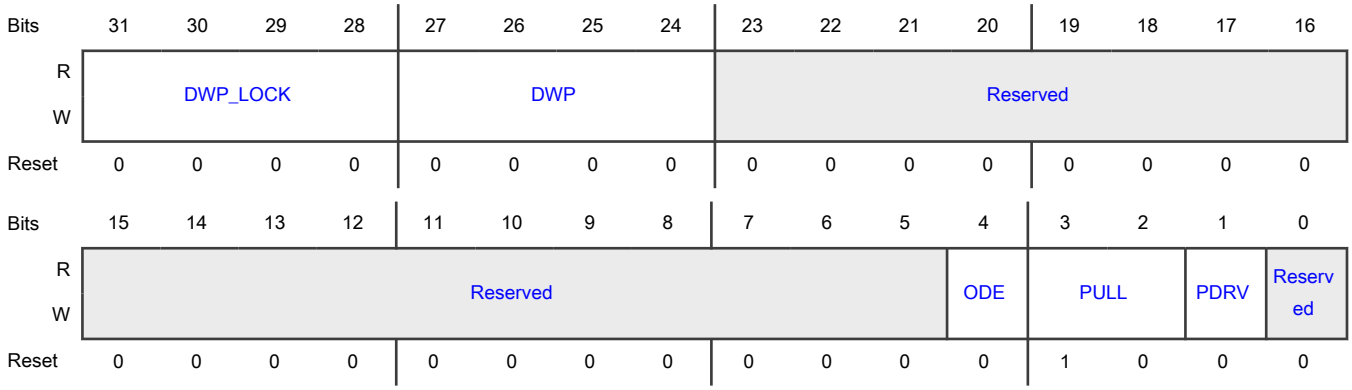
17.4.1.152 SW_PAD_CTL_PAD_GPIO_EMC_B1_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_04)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_04	268h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_04 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_04 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_04

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

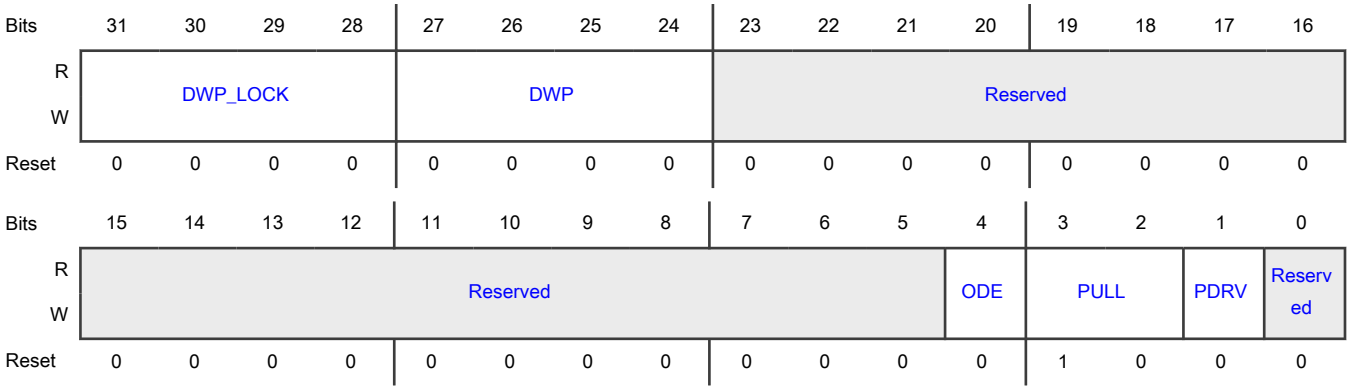
17.4.1.153 SW_PAD_CTL_PAD_GPIO_EMC_B1_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_05)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_05	26Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_05 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_05 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_05 0b - high driver 1b - normal driver
0 —	- Reserved

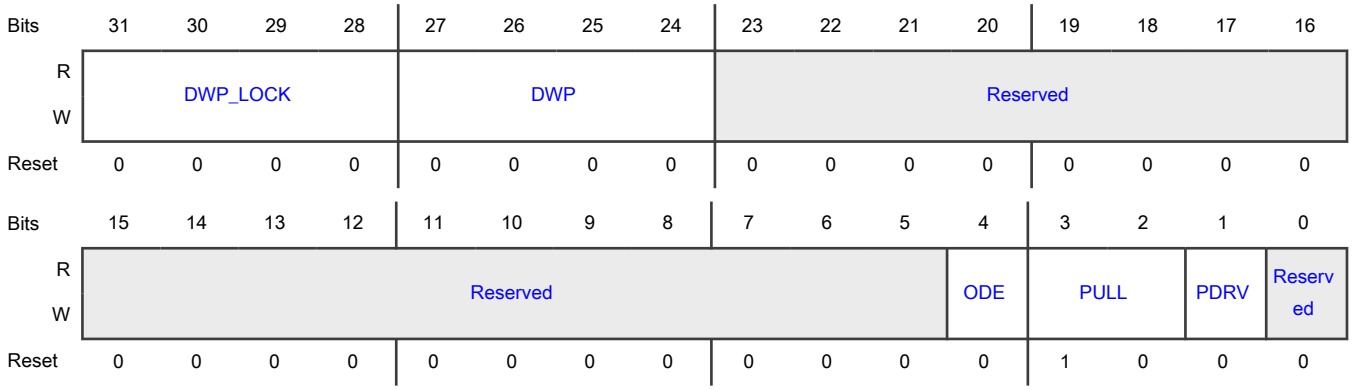
17.4.1.154 SW_PAD_CTL_PAD_GPIO_EMC_B1_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_06)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_06	270h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_06 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_06 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_06

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

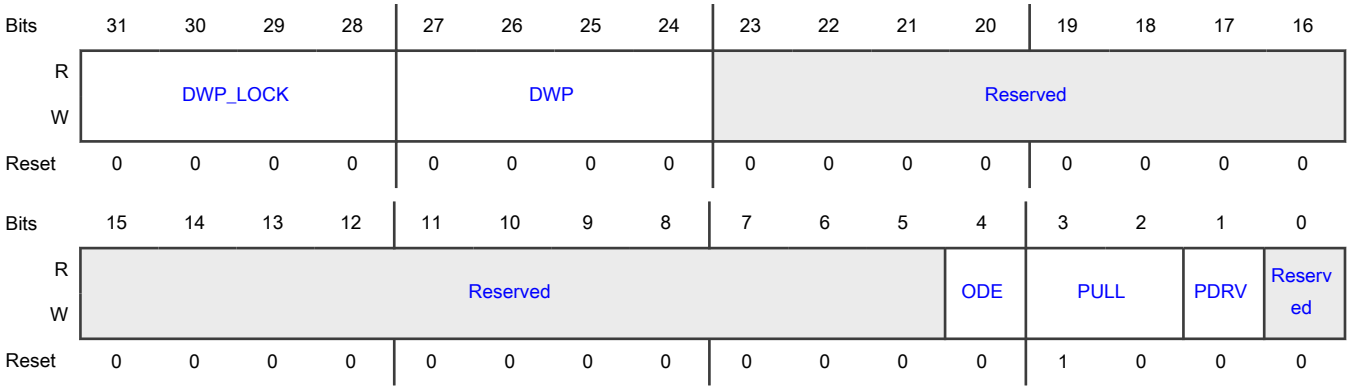
17.4.1.155 SW_PAD_CTL_PAD_GPIO_EMC_B1_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_07)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_07	274h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_07 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_07 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_07 0b - high driver 1b - normal driver
0 —	- Reserved

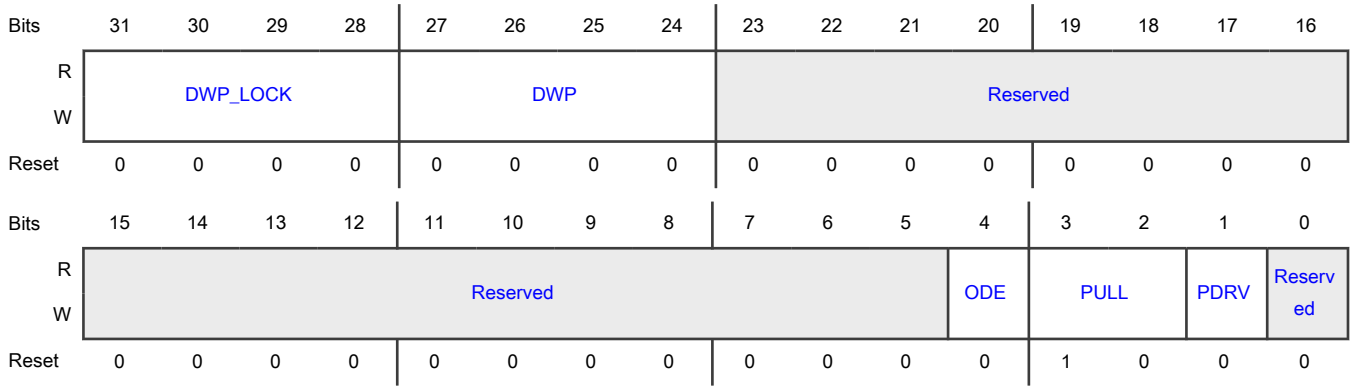
17.4.1.156 SW_PAD_CTL_PAD_GPIO_EMC_B1_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_08)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_08	278h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_08 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_08 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_08

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

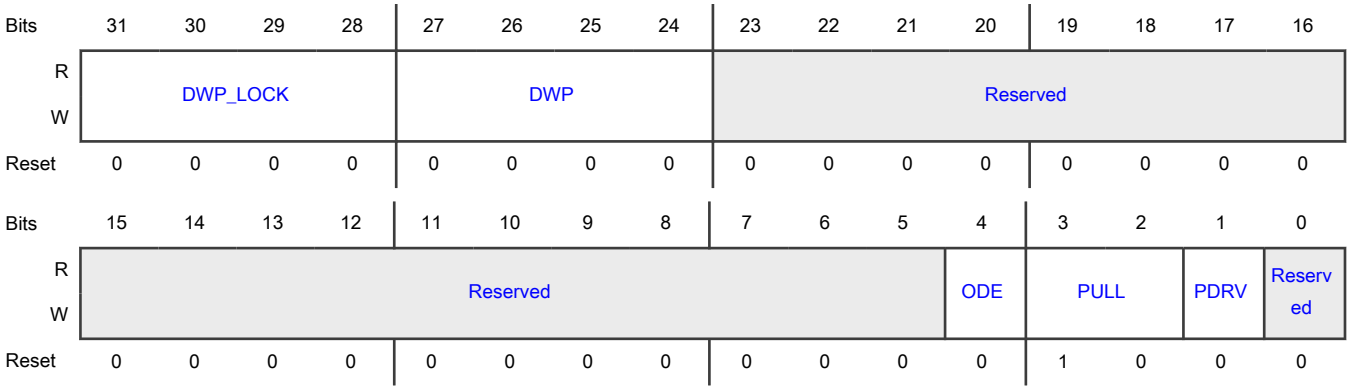
17.4.1.157 SW_PAD_CTL_PAD_GPIO_EMC_B1_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_09)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_09	27Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_09 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_09 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_09 0b - high driver 1b - normal driver
0 —	- Reserved

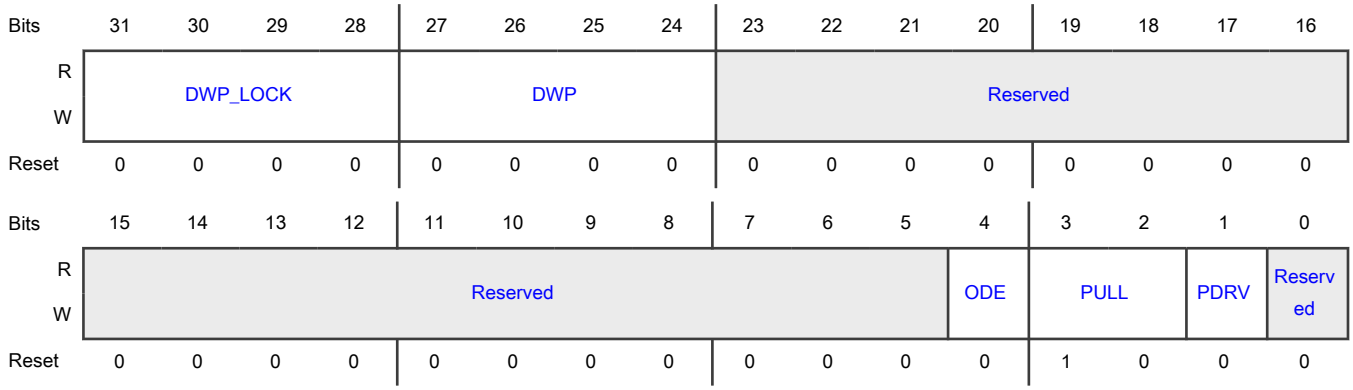
17.4.1.158 SW_PAD_CTL_PAD_GPIO_EMC_B1_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_10)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_10	280h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_10 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_10 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_10

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

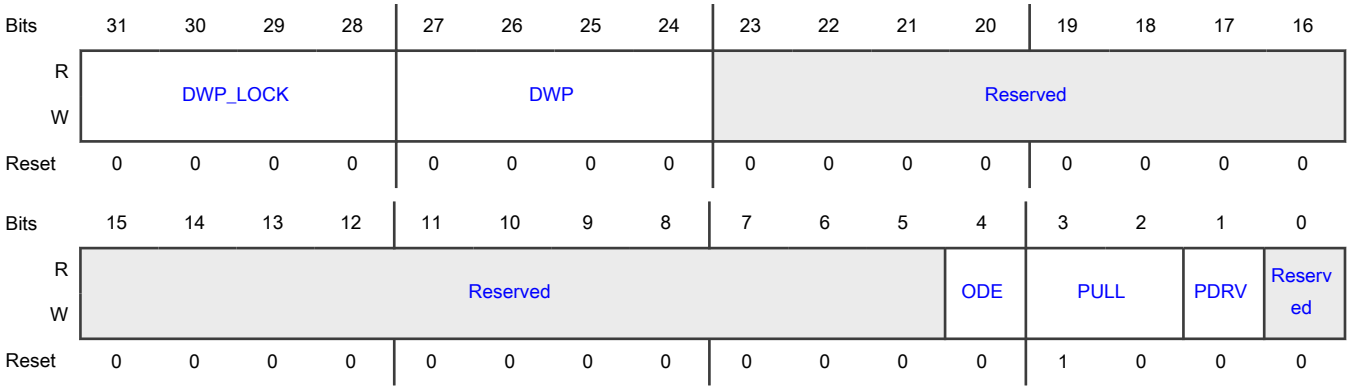
17.4.1.159 SW_PAD_CTL_PAD_GPIO_EMC_B1_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_11)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_11	284h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_11 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_11 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_11 0b - high driver 1b - normal driver
0 —	- Reserved

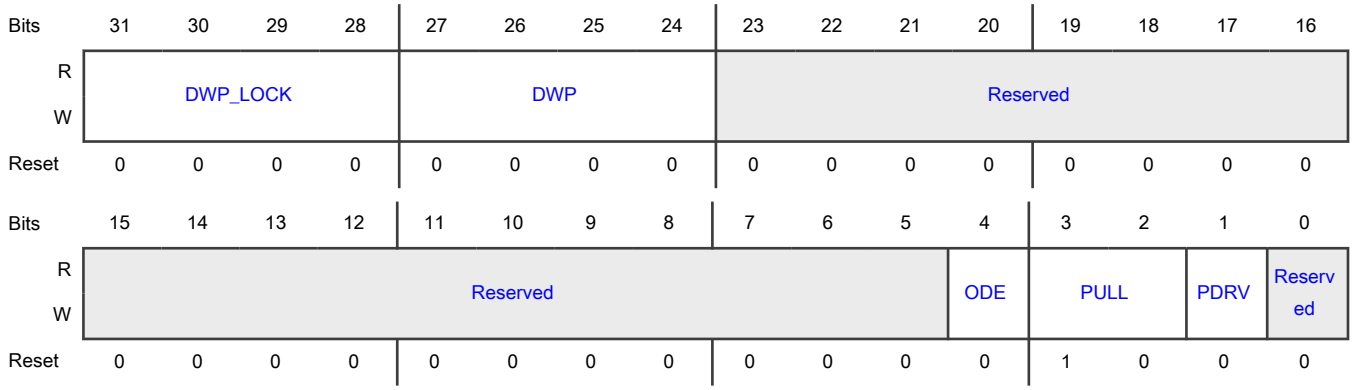
17.4.1.160 SW_PAD_CTL_PAD_GPIO_EMC_B1_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_12)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_12	288h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_12 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_12 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_12

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

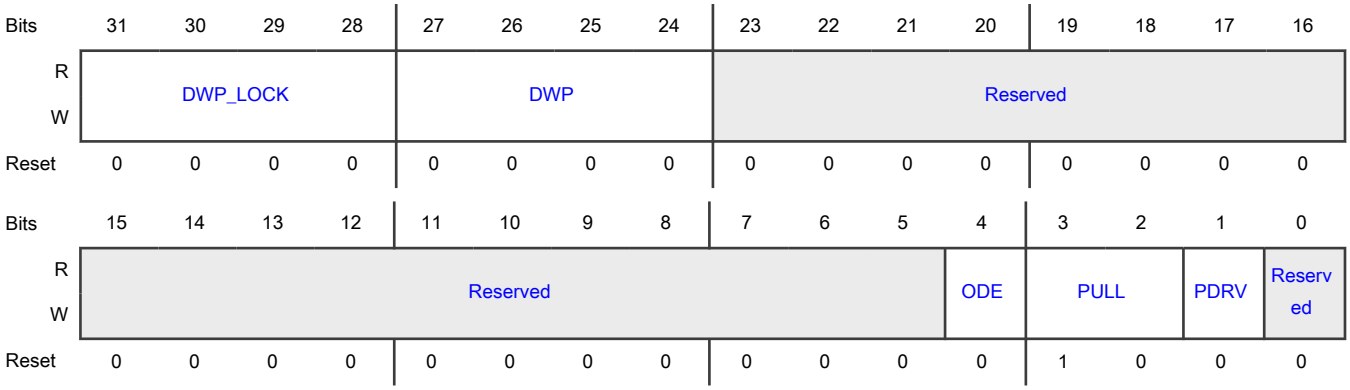
17.4.1.161 SW_PAD_CTL_PAD_GPIO_EMC_B1_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_13)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_13	28Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_13 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_13 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_13 0b - high driver 1b - normal driver
0 —	- Reserved

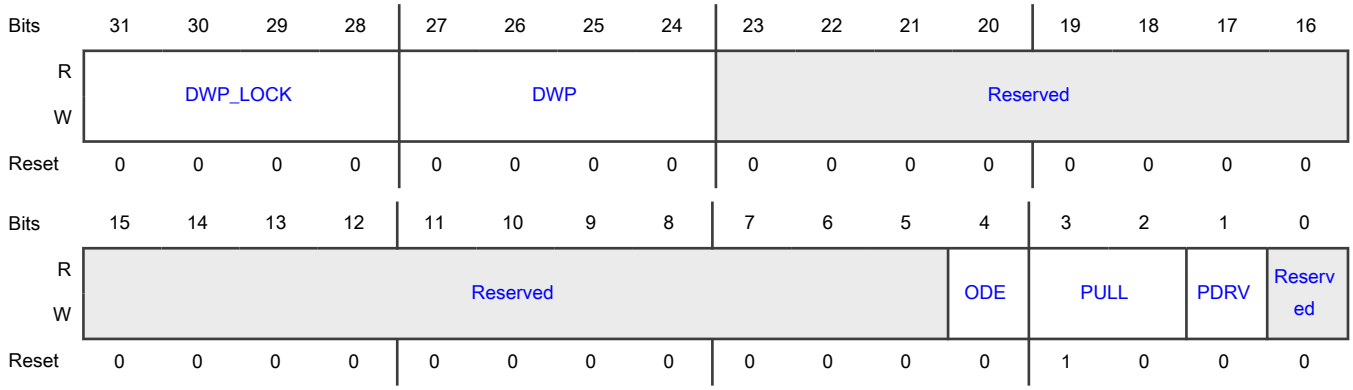
17.4.1.162 SW_PAD_CTL_PAD_GPIO_EMC_B1_14 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_14)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_14	290h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_14 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_14 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_14

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.163 SW_PAD_CTL_PAD_GPIO_EMC_B1_15 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_15)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_15	294h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_15 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_15 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_15 0b - high driver 1b - normal driver
0 —	- Reserved

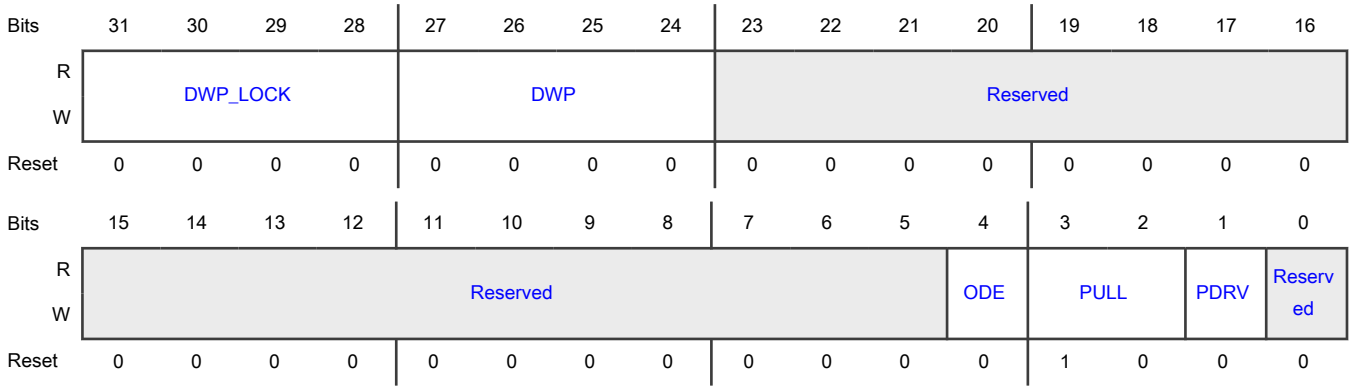
17.4.1.164 SW_PAD_CTL_PAD_GPIO_EMC_B1_16 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_16)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_16	298h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_16 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_16 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_16

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

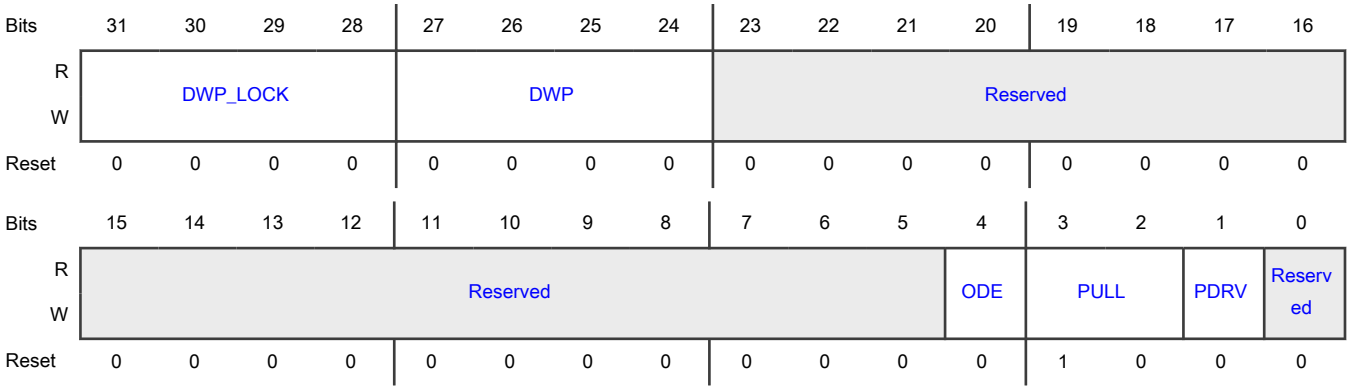
17.4.1.165 SW_PAD_CTL_PAD_GPIO_EMC_B1_17 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_17)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_17	29Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_17 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_17 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_17 0b - high driver 1b - normal driver
0 —	- Reserved

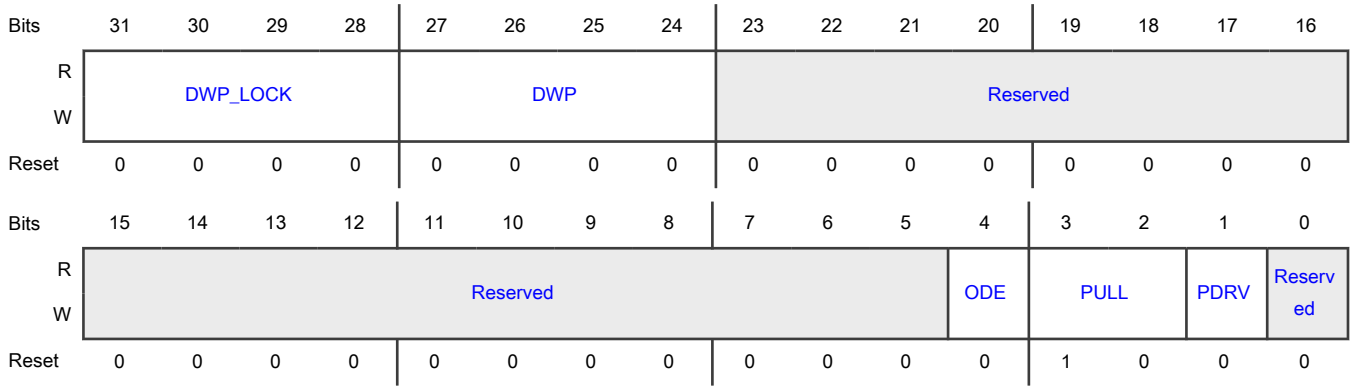
17.4.1.166 SW_PAD_CTL_PAD_GPIO_EMC_B1_18 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_18)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_18	2A0h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_18 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_18 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_18

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

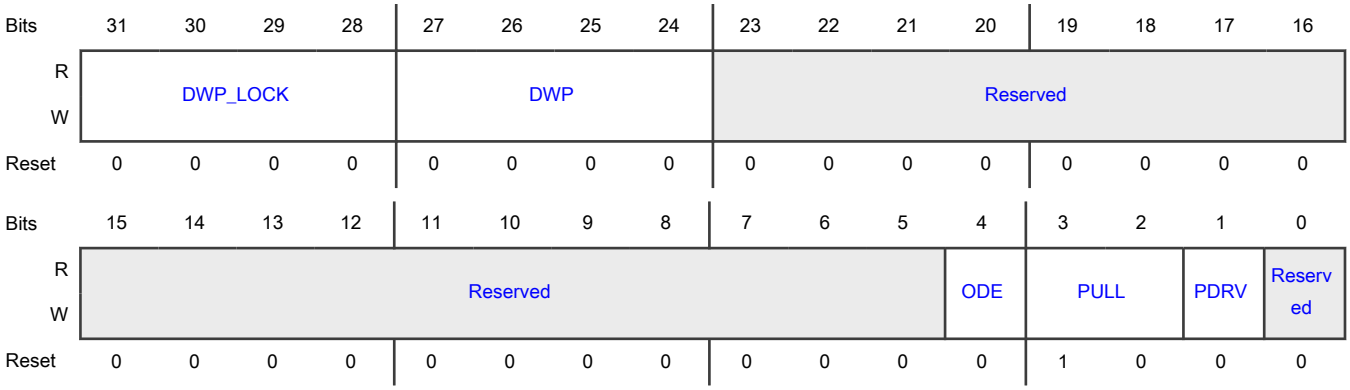
17.4.1.167 SW_PAD_CTL_PAD_GPIO_EMC_B1_19 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_19)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_19	2A4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_19 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_19 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_19 0b - high driver 1b - normal driver
0 —	- Reserved

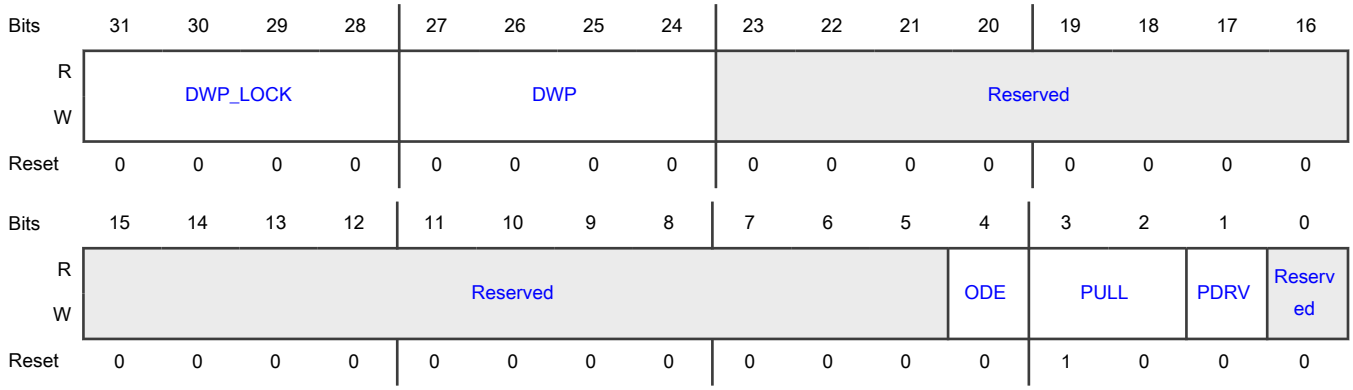
17.4.1.168 SW_PAD_CTL_PAD_GPIO_EMC_B1_20 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_20)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_20	2A8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_20 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_20 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_20

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

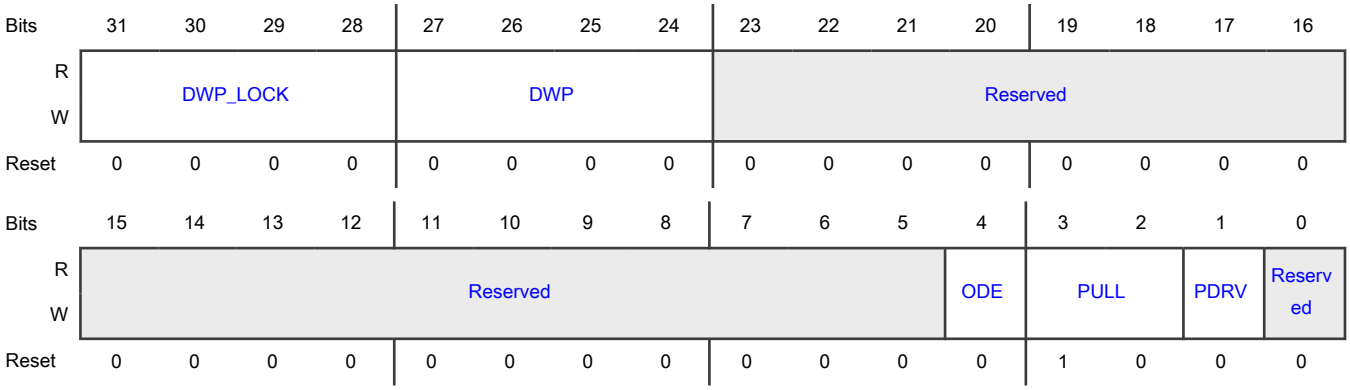
17.4.1.169 SW_PAD_CTL_PAD_GPIO_EMC_B1_21 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_21)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_21	2ACh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_21 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_21 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_21 0b - high driver 1b - normal driver
0 —	- Reserved

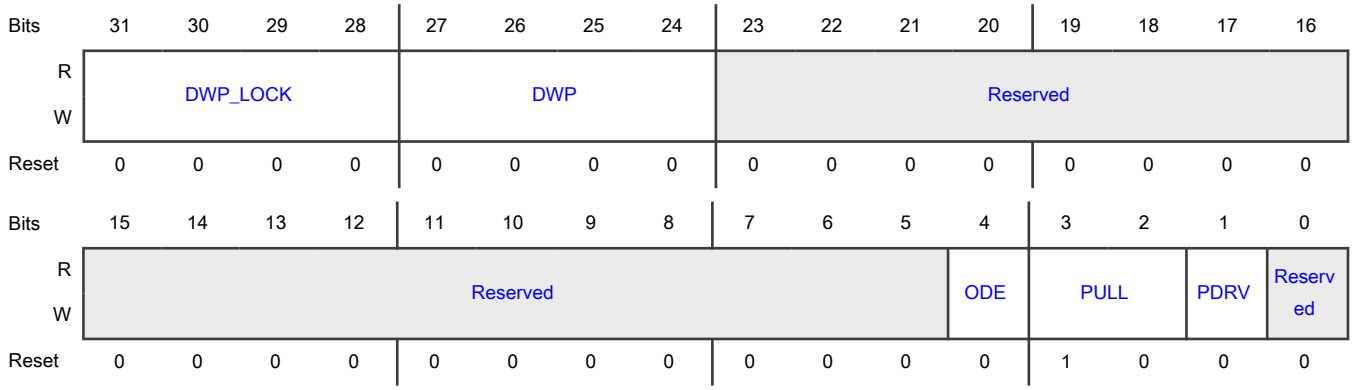
17.4.1.170 SW_PAD_CTL_PAD_GPIO_EMC_B1_22 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_22)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_22	2B0h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_22 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_22 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_22

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

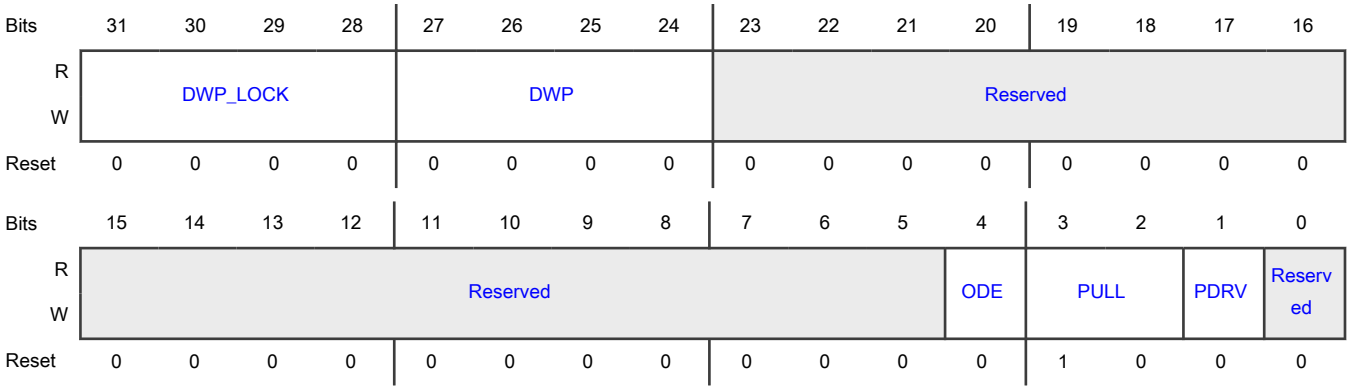
17.4.1.171 SW_PAD_CTL_PAD_GPIO_EMC_B1_23 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_23)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_23	2B4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_23 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_23 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_23 0b - high driver 1b - normal driver
0 —	- Reserved

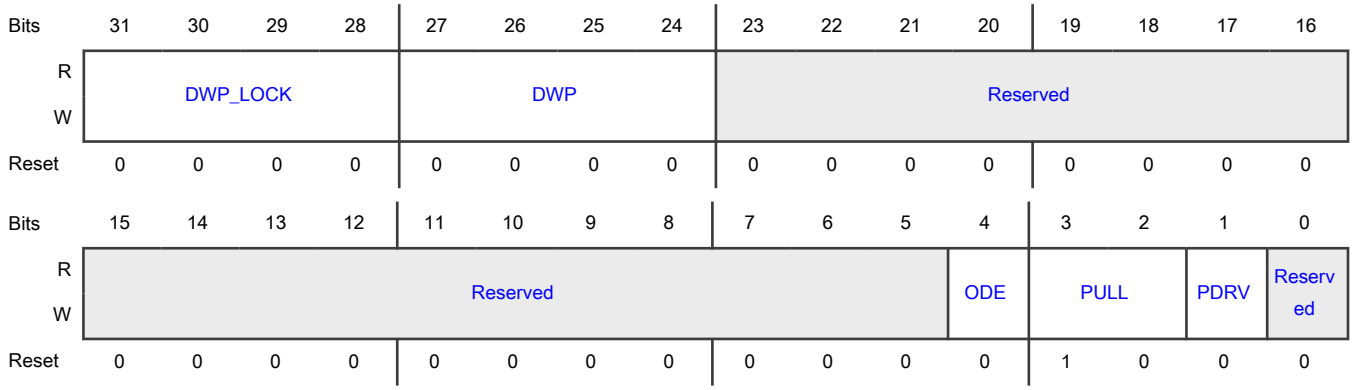
17.4.1.172 SW_PAD_CTL_PAD_GPIO_EMC_B1_24 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_24)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_24	2B8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_24 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_24 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_24

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

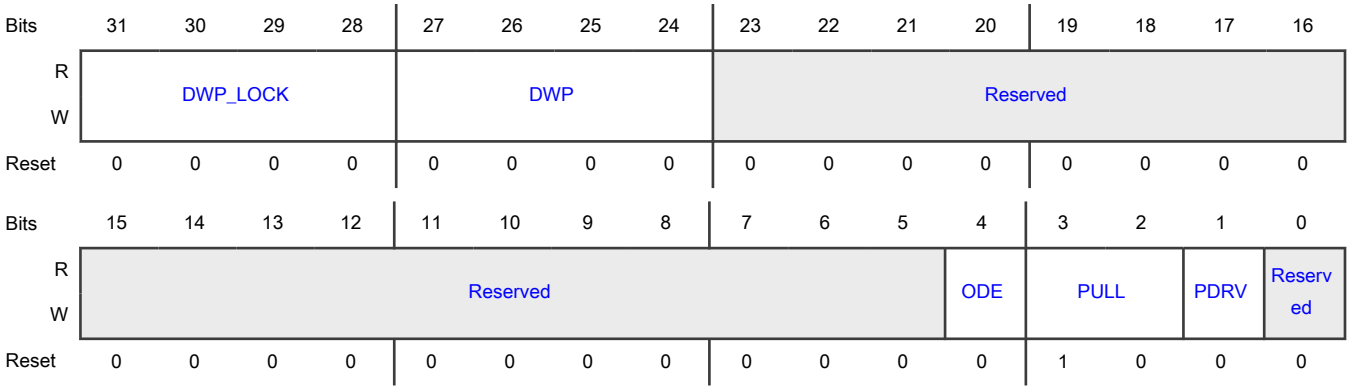
17.4.1.173 SW_PAD_CTL_PAD_GPIO_EMC_B1_25 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_25)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_25	2BCh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_25 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_25 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_25 0b - high driver 1b - normal driver
0 —	- Reserved

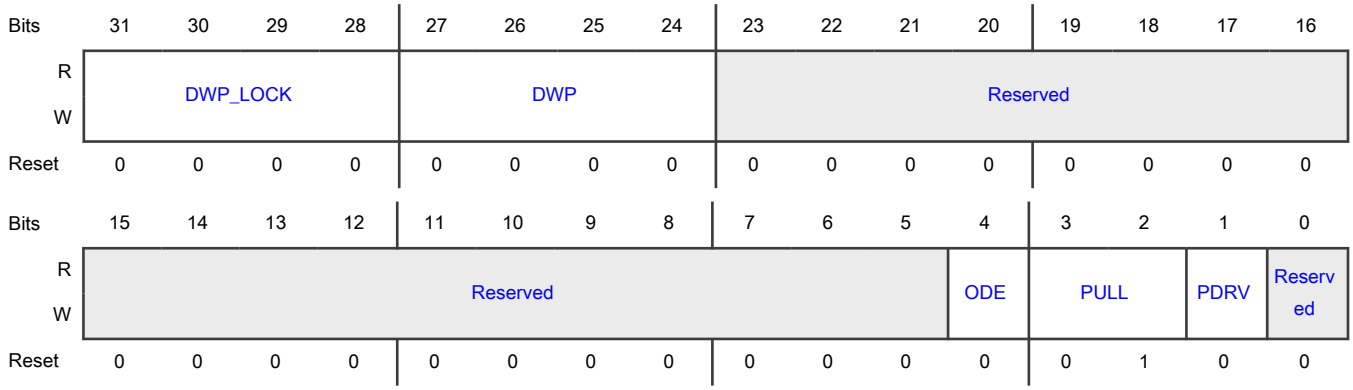
17.4.1.174 SW_PAD_CTL_PAD_GPIO_EMC_B1_26 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_26)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_26	2C0h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_26 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_26 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_26

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

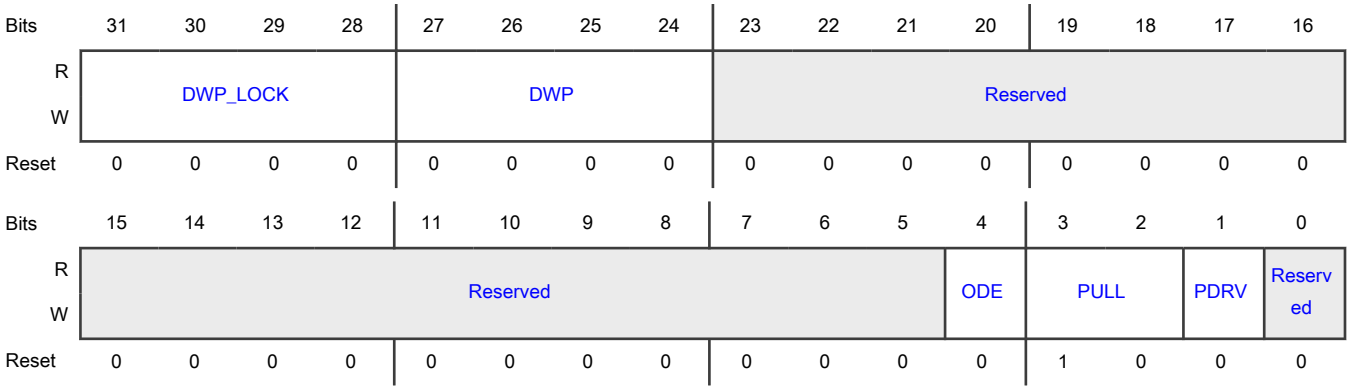
17.4.1.175 SW_PAD_CTL_PAD_GPIO_EMC_B1_27 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_27)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_27	2C4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_27 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_27 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_27 0b - high driver 1b - normal driver
0 —	- Reserved

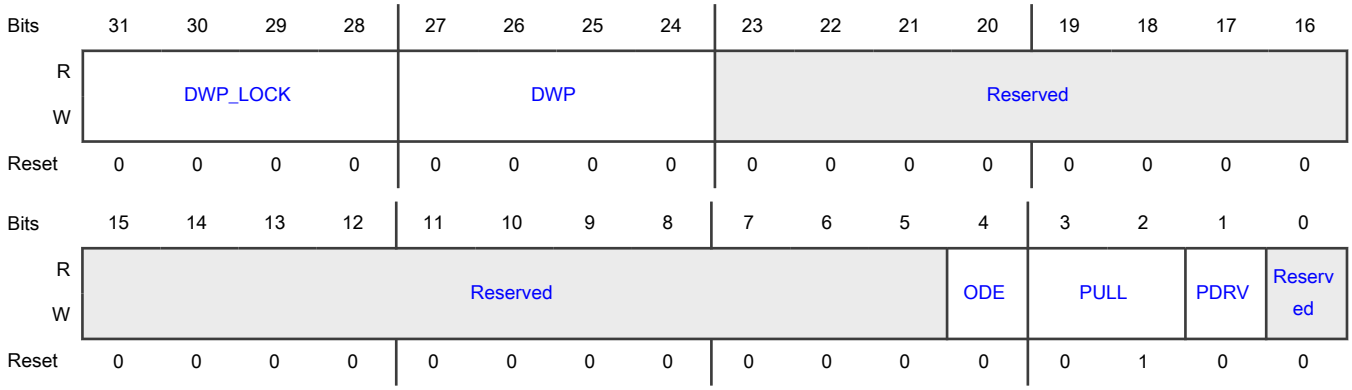
17.4.1.176 SW_PAD_CTL_PAD_GPIO_EMC_B1_28 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_28)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_28	2C8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_28 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_28 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_28

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

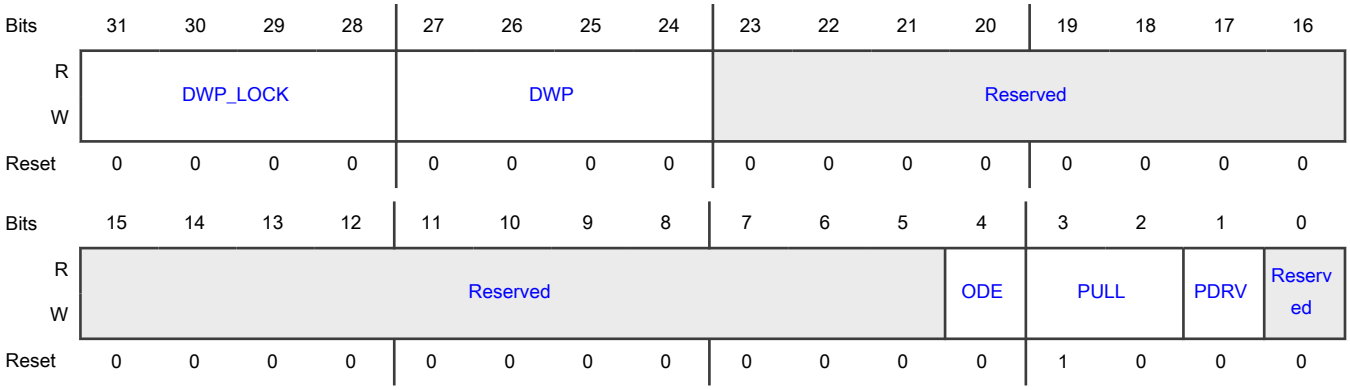
17.4.1.177 SW_PAD_CTL_PAD_GPIO_EMC_B1_29 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_29)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_29	2CCh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_29 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_29 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_29 0b - high driver 1b - normal driver
0 —	- Reserved

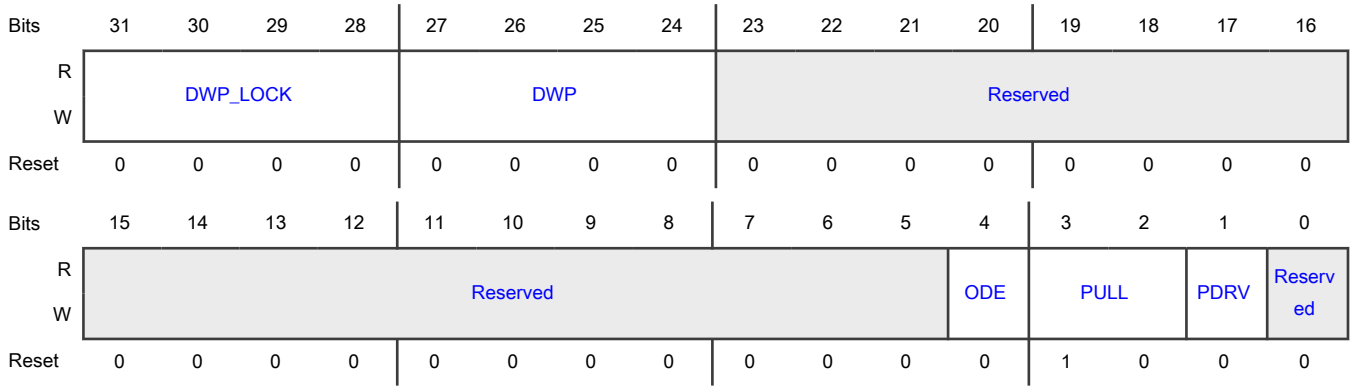
17.4.1.178 SW_PAD_CTL_PAD_GPIO_EMC_B1_30 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_30)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_30	2D0h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_30 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_30 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_30

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

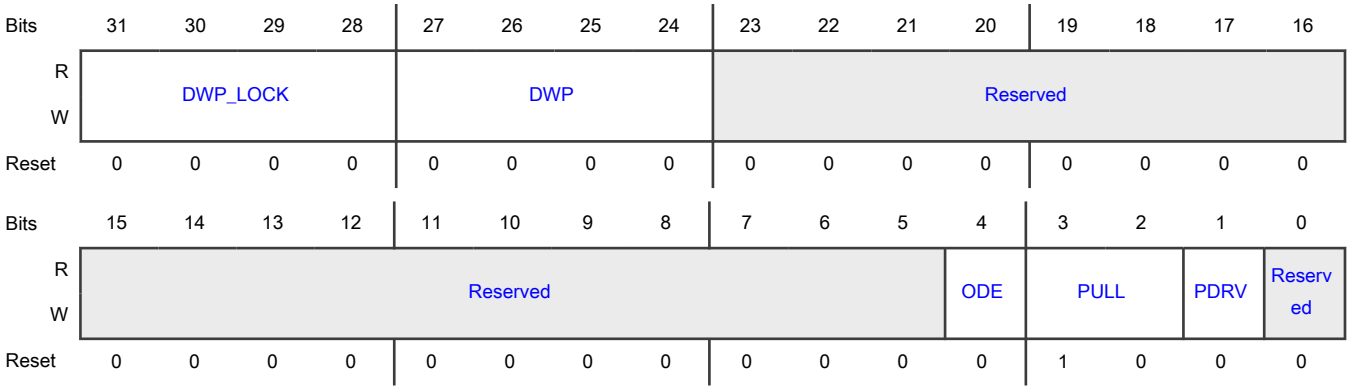
17.4.1.179 SW_PAD_CTL_PAD_GPIO_EMC_B1_31 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_31)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_31	2D4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_31 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_31 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_31 0b - high driver 1b - normal driver
0 —	- Reserved

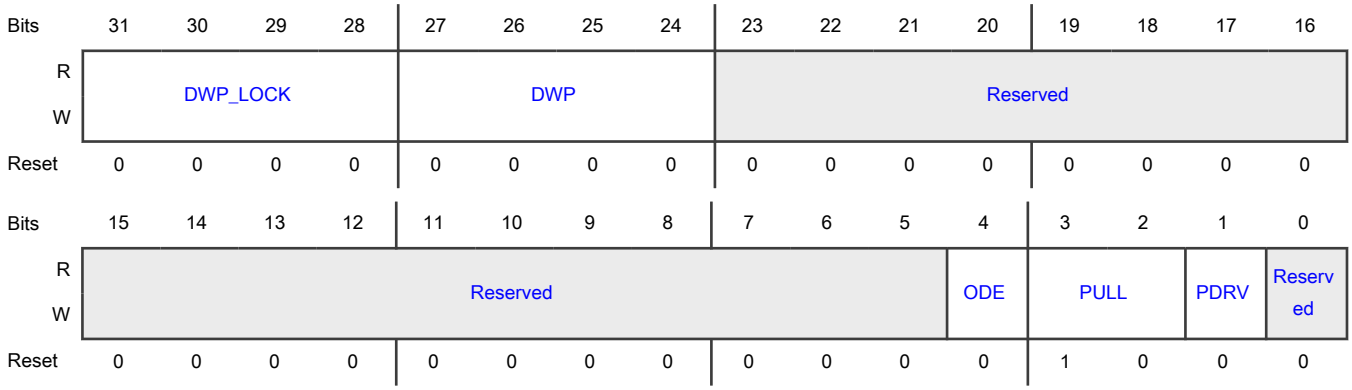
17.4.1.180 SW_PAD_CTL_PAD_GPIO_EMC_B1_32 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_32)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_32	2D8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_32 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_32 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_32

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

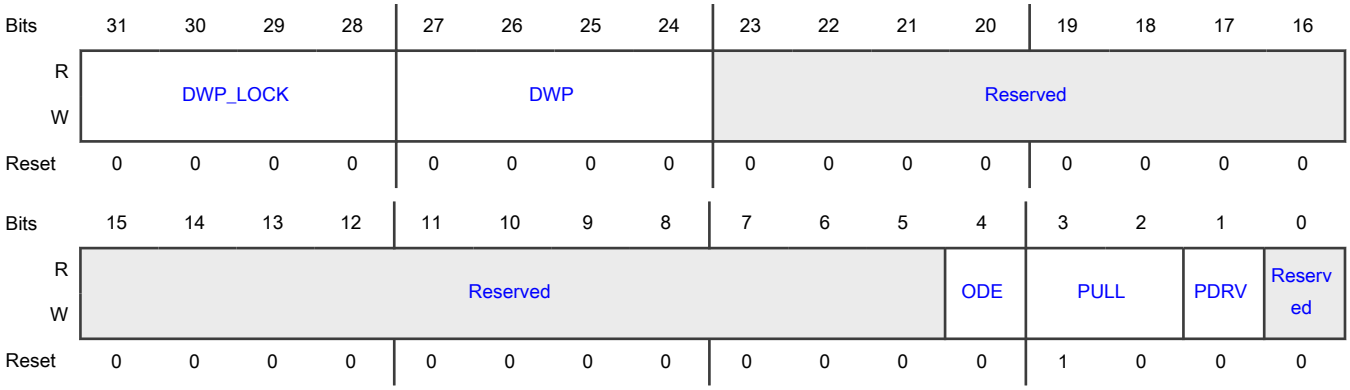
17.4.1.181 SW_PAD_CTL_PAD_GPIO_EMC_B1_33 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_33)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_33	2DCh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_33 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_33 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_33 0b - high driver 1b - normal driver
0 —	- Reserved

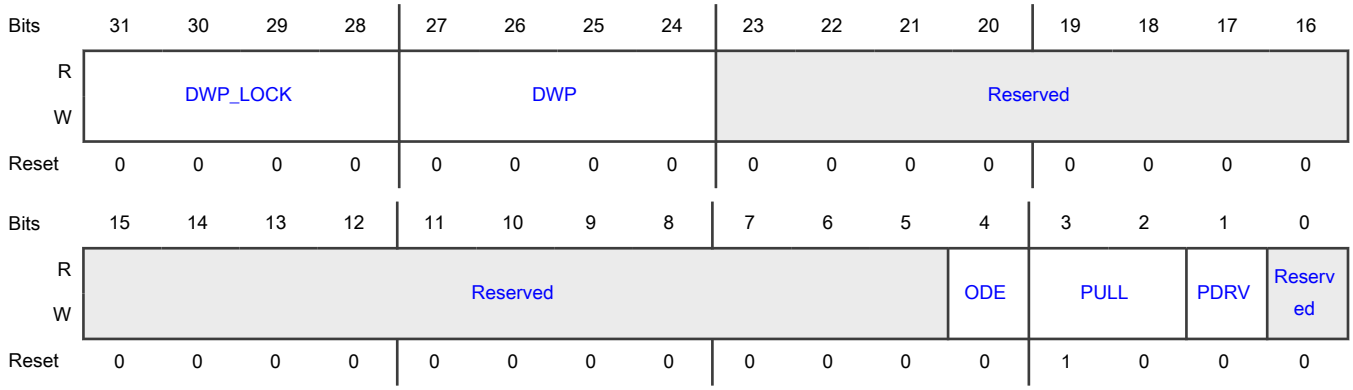
17.4.1.182 SW_PAD_CTL_PAD_GPIO_EMC_B1_34 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_34)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_34	2E0h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_34 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_34 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_34

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.183 SW_PAD_CTL_PAD_GPIO_EMC_B1_35 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_35)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_35	2E4h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_35 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_35 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_35 0b - high driver 1b - normal driver
0 —	- Reserved

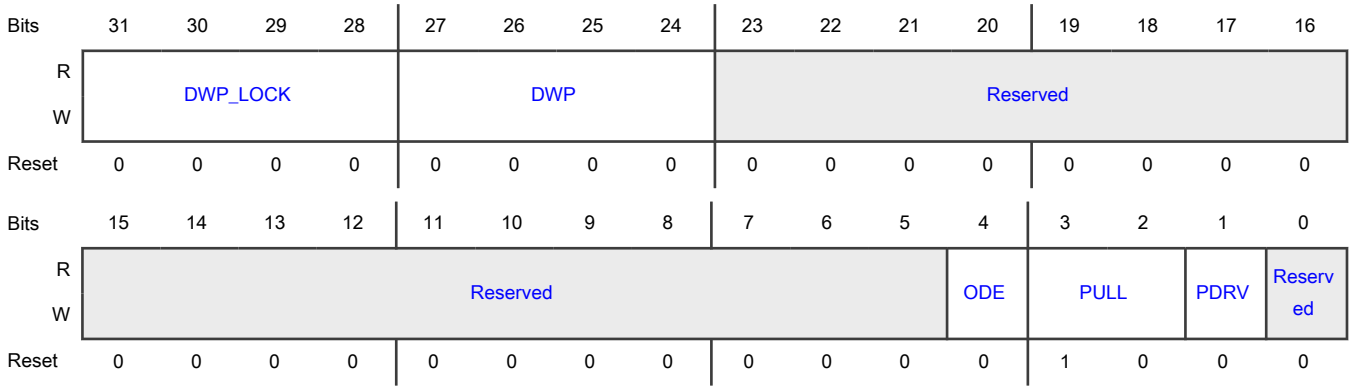
17.4.1.184 SW_PAD_CTL_PAD_GPIO_EMC_B1_36 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_36)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_36	2E8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_36 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_36 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_36

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

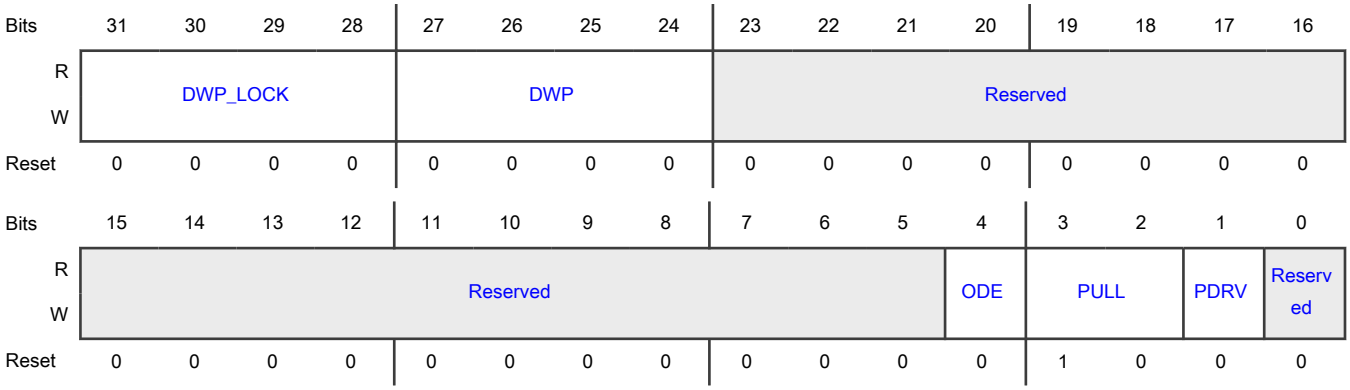
17.4.1.185 SW_PAD_CTL_PAD_GPIO_EMC_B1_37 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_37)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_37	2ECh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_37 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_37 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_37 0b - high driver 1b - normal driver
0 —	- Reserved

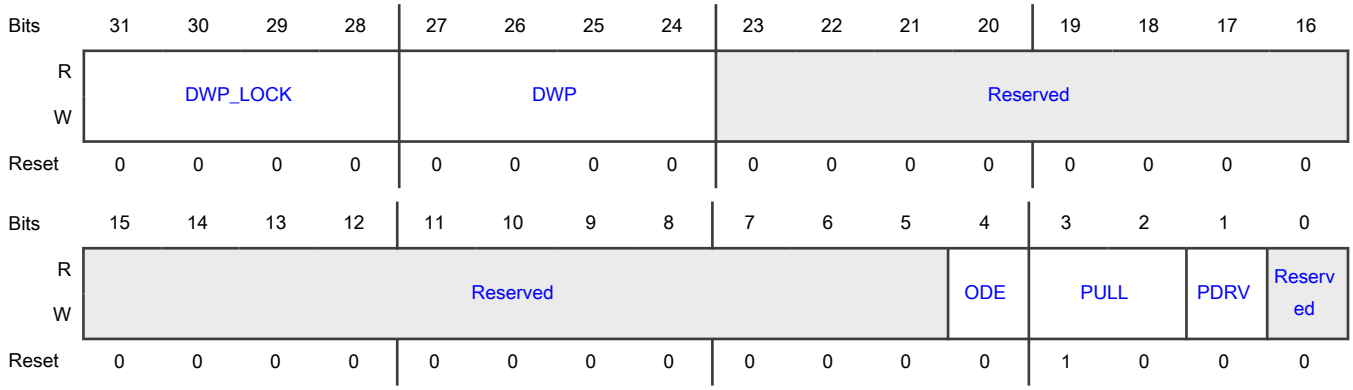
17.4.1.186 SW_PAD_CTL_PAD_GPIO_EMC_B1_38 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_38)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_38	2F0h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_38 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_38 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_38

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.187 SW_PAD_CTL_PAD_GPIO_EMC_B1_39 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_39)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_39	2F4h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_39 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_39 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_39 0b - high driver 1b - normal driver
0 —	- Reserved

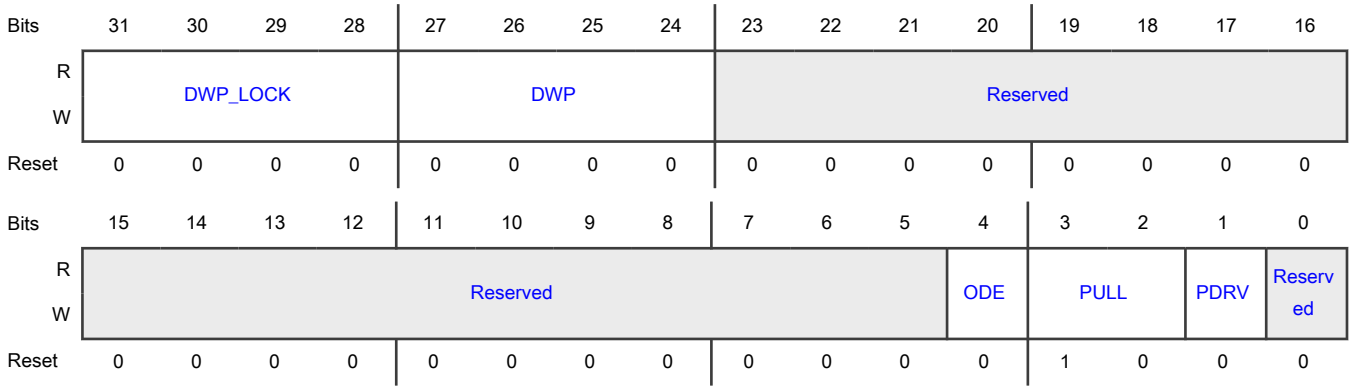
17.4.1.188 SW_PAD_CTL_PAD_GPIO_EMC_B1_40 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_40)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_40	2F8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_40 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_40 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_40

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

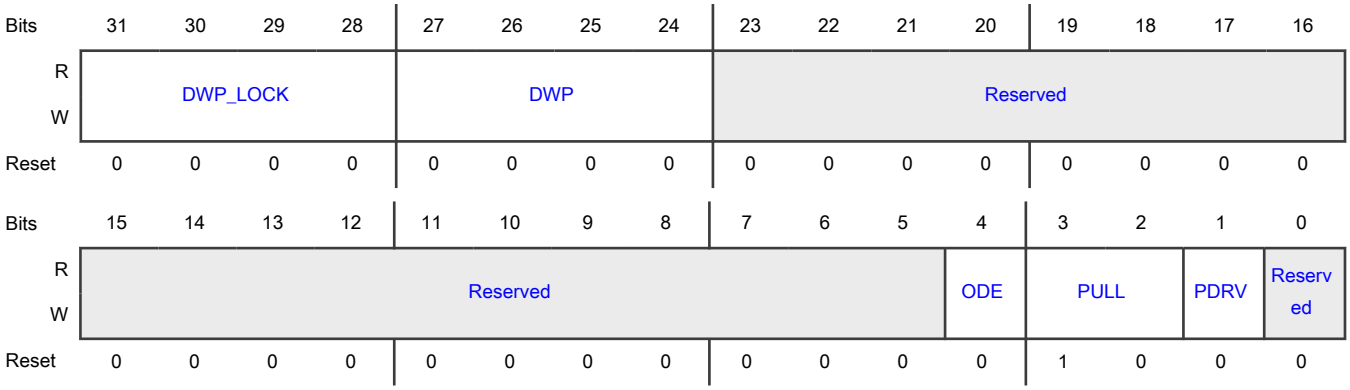
17.4.1.189 SW_PAD_CTL_PAD_GPIO_EMC_B1_41 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B1_41)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B1_41	2FCh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B1_41 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B1_41 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B1_41 0b - high driver 1b - normal driver
0 —	- Reserved

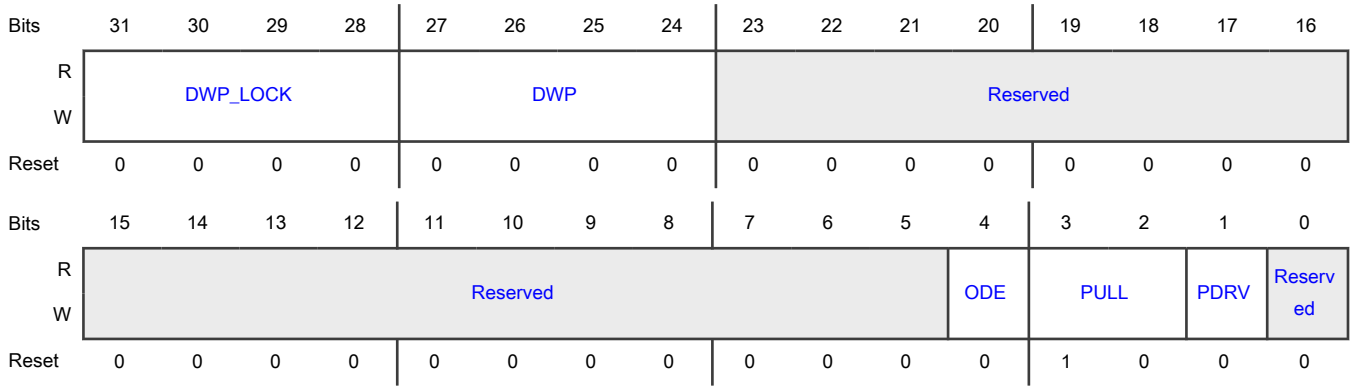
17.4.1.190 SW_PAD_CTL_PAD_GPIO_EMC_B2_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_00)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_00	300h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_00 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_00 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_00

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

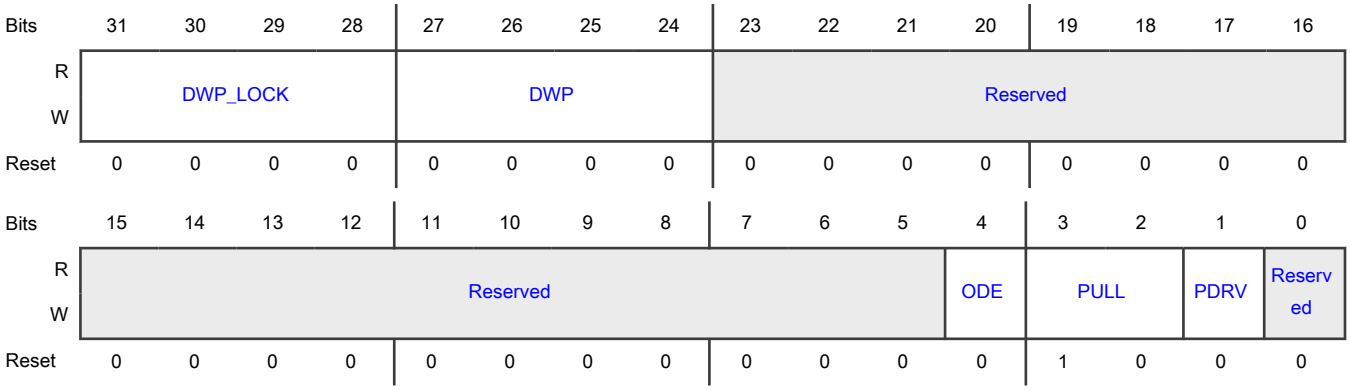
17.4.1.191 SW_PAD_CTL_PAD_GPIO_EMC_B2_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_01)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_01	304h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_01 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_01 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_01 0b - high driver 1b - normal driver
0 —	- Reserved

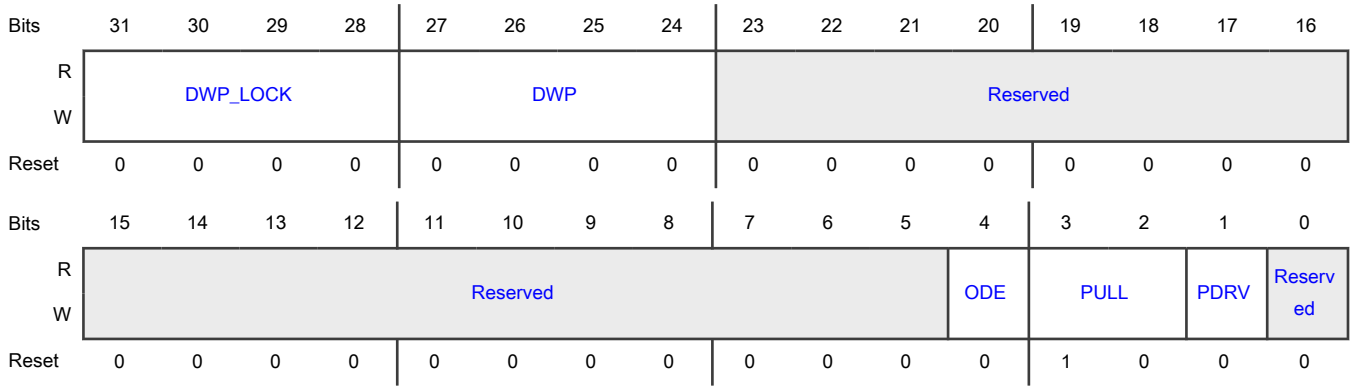
17.4.1.192 SW_PAD_CTL_PAD_GPIO_EMC_B2_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_02)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_02	308h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_02 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_02 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_02

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

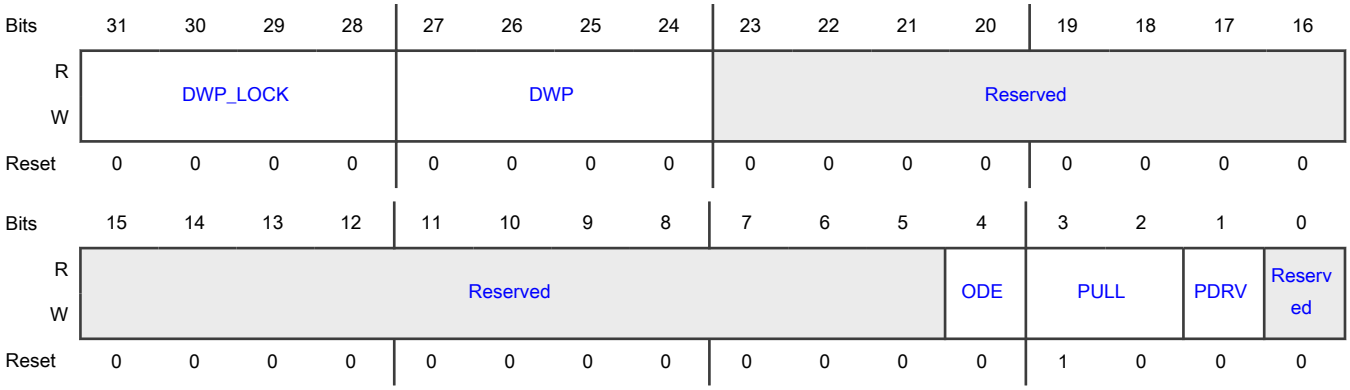
17.4.1.193 SW_PAD_CTL_PAD_GPIO_EMC_B2_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_03)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_03	30Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_03 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_03 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_03 0b - high driver 1b - normal driver
0 —	- Reserved

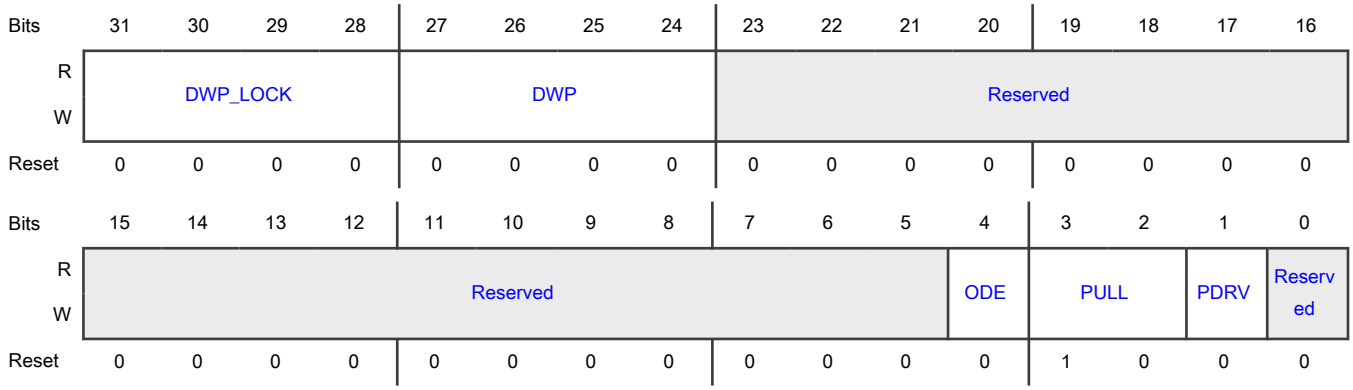
17.4.1.194 SW_PAD_CTL_PAD_GPIO_EMC_B2_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_04)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_04	310h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_04 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_04 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_04

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

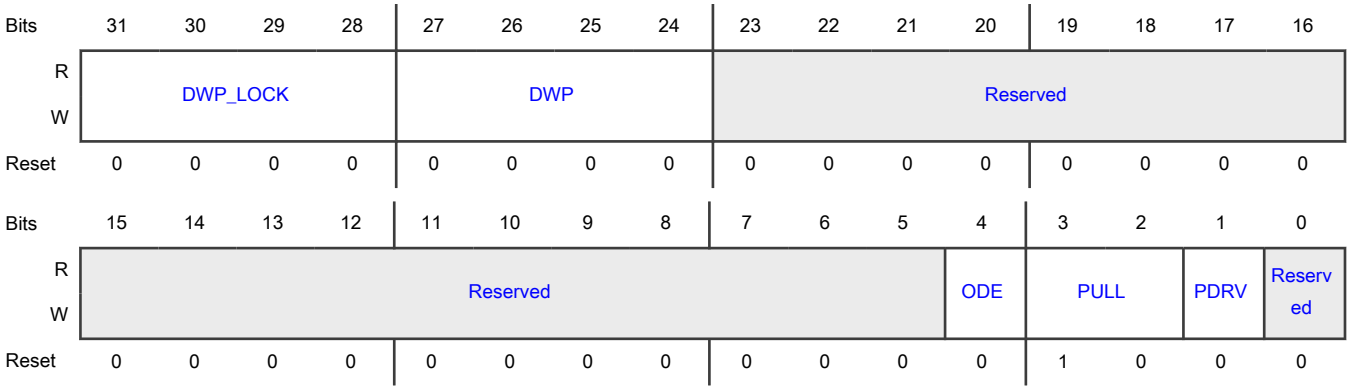
17.4.1.195 SW_PAD_CTL_PAD_GPIO_EMC_B2_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_05)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_05	314h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_05 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_05 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_05 0b - high driver 1b - normal driver
0 —	- Reserved

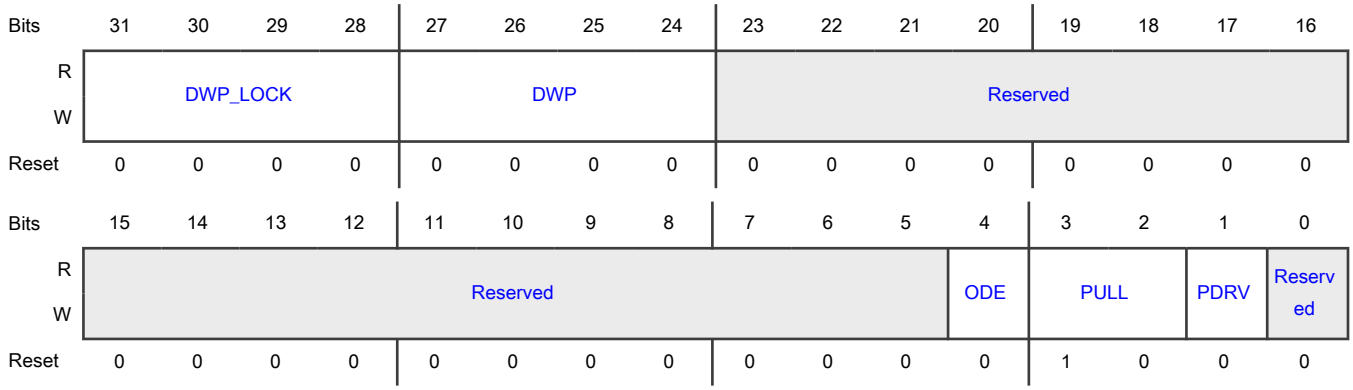
17.4.1.196 SW_PAD_CTL_PAD_GPIO_EMC_B2_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_06)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_06	318h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_06 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_06 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_06

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

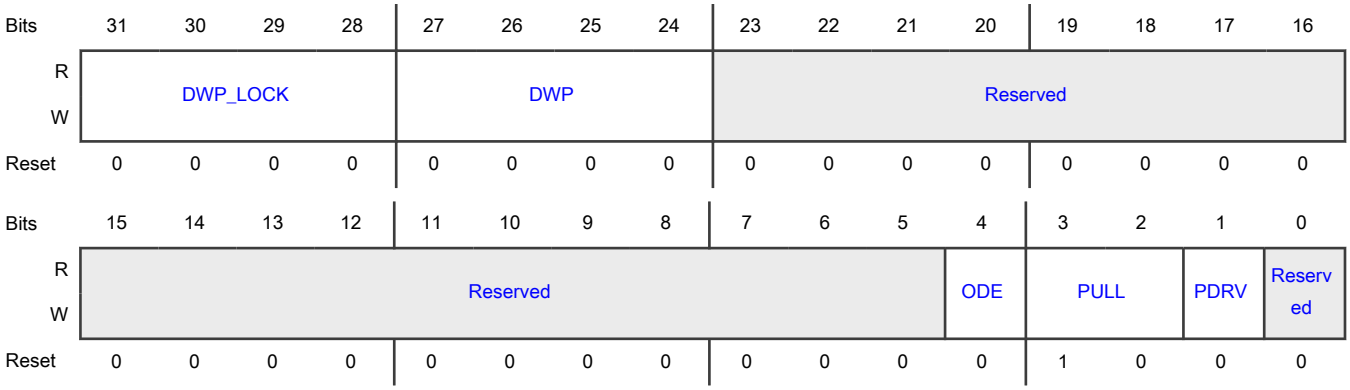
17.4.1.197 SW_PAD_CTL_PAD_GPIO_EMC_B2_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_07)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_07	31Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_07 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_07 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_07 0b - high driver 1b - normal driver
0 —	- Reserved

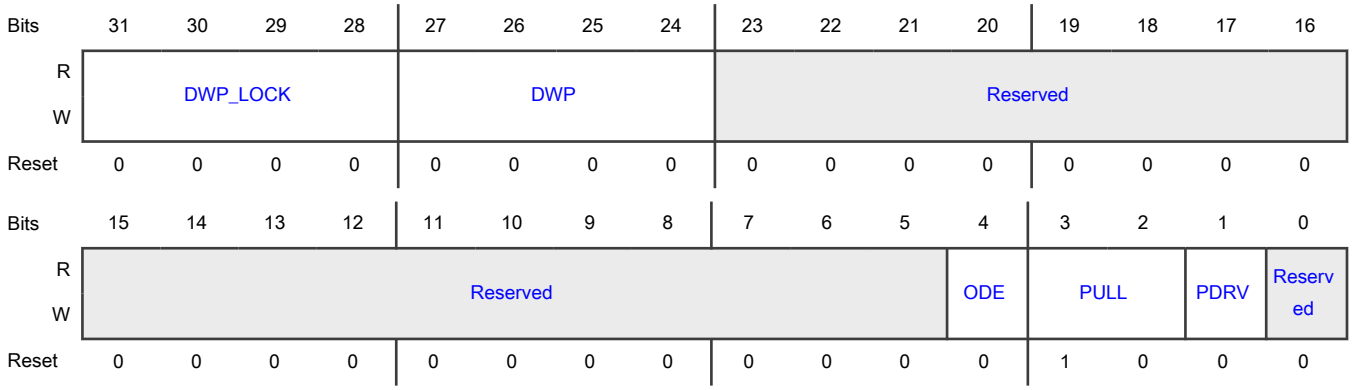
17.4.1.198 SW_PAD_CTL_PAD_GPIO_EMC_B2_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_08)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_08	320h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_08 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_08 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_08

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

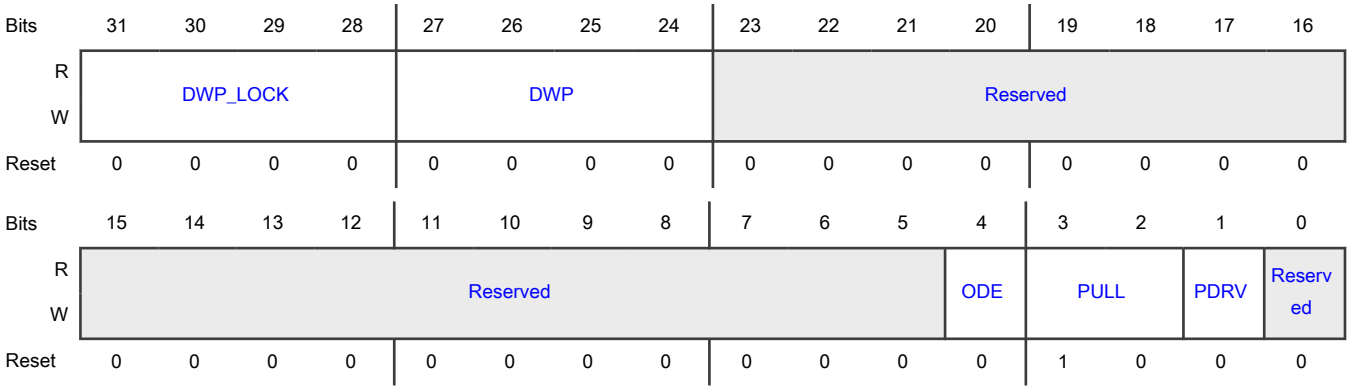
17.4.1.199 SW_PAD_CTL_PAD_GPIO_EMC_B2_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_09)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_09	324h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_09 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_09 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_09 0b - high driver 1b - normal driver
0 —	- Reserved

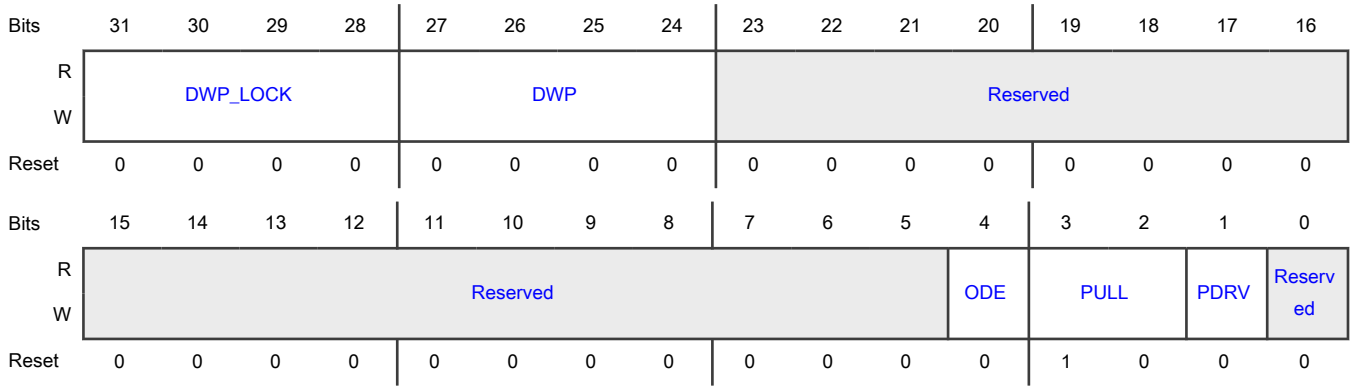
17.4.1.200 SW_PAD_CTL_PAD_GPIO_EMC_B2_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_10)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_10	328h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_10 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_10 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_10

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

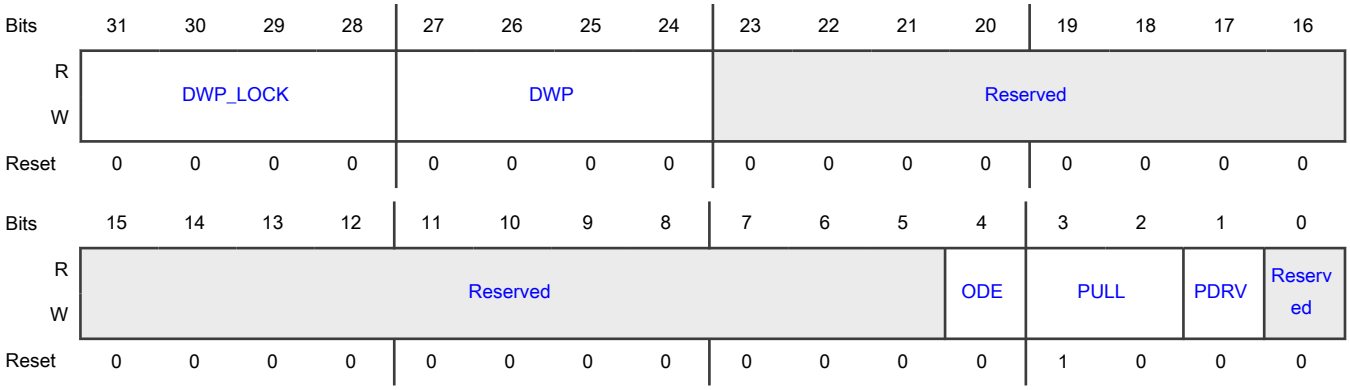
17.4.1.201 SW_PAD_CTL_PAD_GPIO_EMC_B2_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_11)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_11	32Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_11 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_11 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_11 0b - high driver 1b - normal driver
0 —	- Reserved

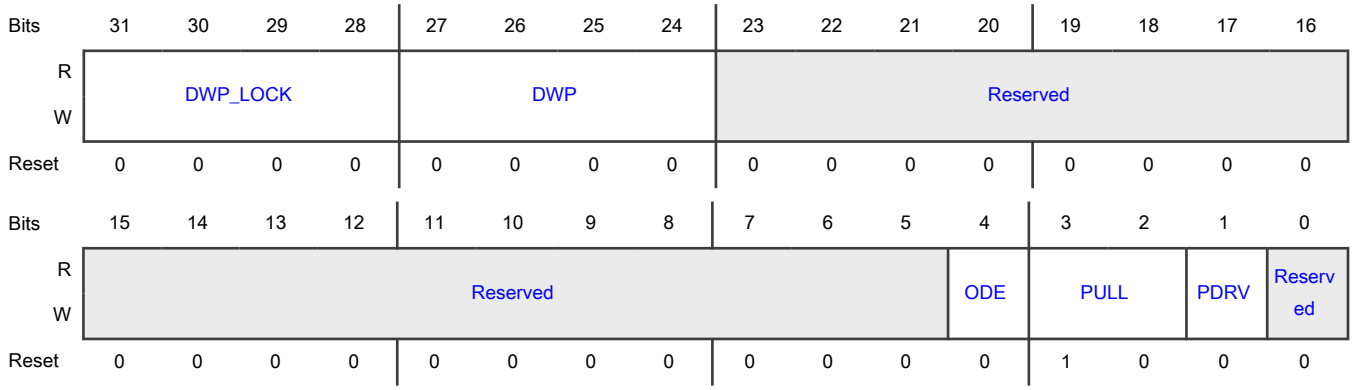
17.4.1.202 SW_PAD_CTL_PAD_GPIO_EMC_B2_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_12)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_12	330h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_12 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_12 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_12

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

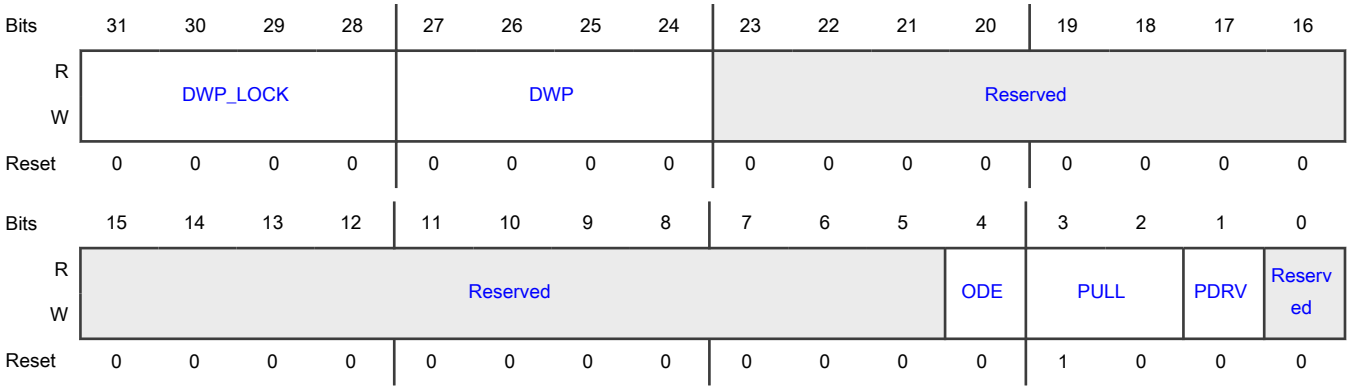
17.4.1.203 SW_PAD_CTL_PAD_GPIO_EMC_B2_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_13)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_13	334h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_13 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_13 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_13 0b - high driver 1b - normal driver
0 —	- Reserved

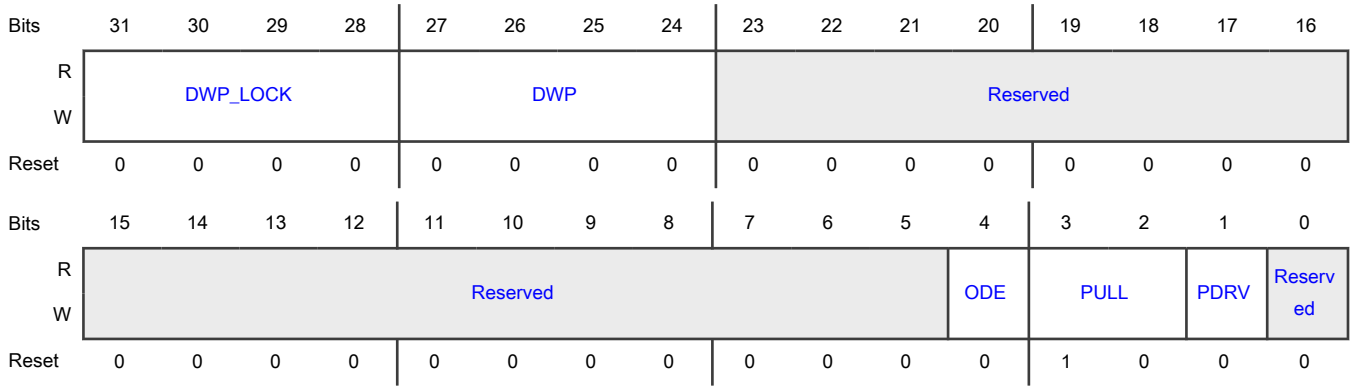
17.4.1.204 SW_PAD_CTL_PAD_GPIO_EMC_B2_14 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_14)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_14	338h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_14 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_14 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_14

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

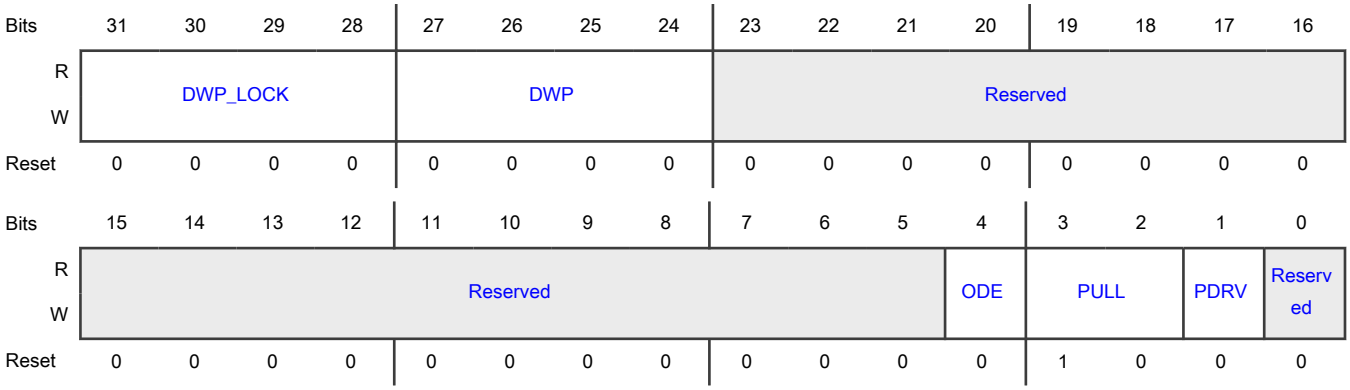
17.4.1.205 SW_PAD_CTL_PAD_GPIO_EMC_B2_15 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_15)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_15	33Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_15 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_15 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_15 0b - high driver 1b - normal driver
0 —	- Reserved

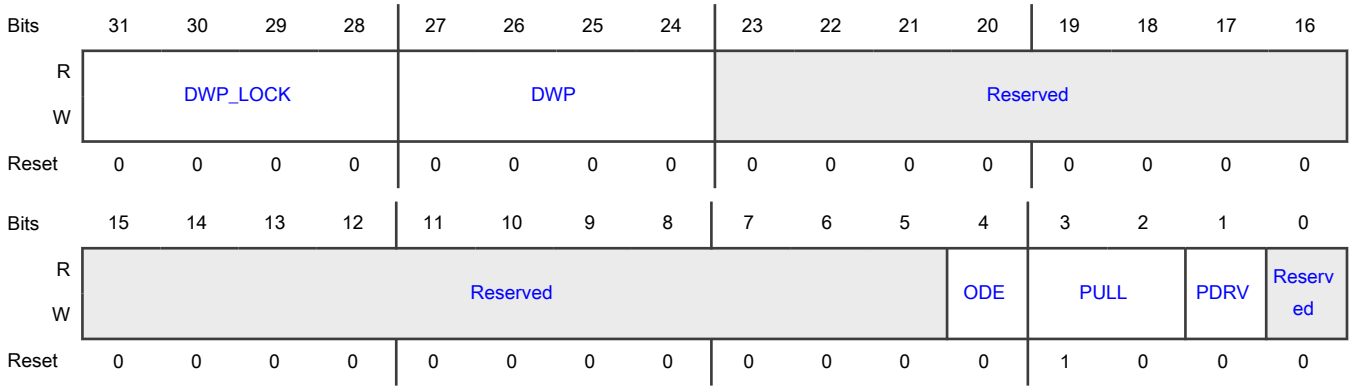
17.4.1.206 SW_PAD_CTL_PAD_GPIO_EMC_B2_16 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_16)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_16	340h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_16 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_16 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_16

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

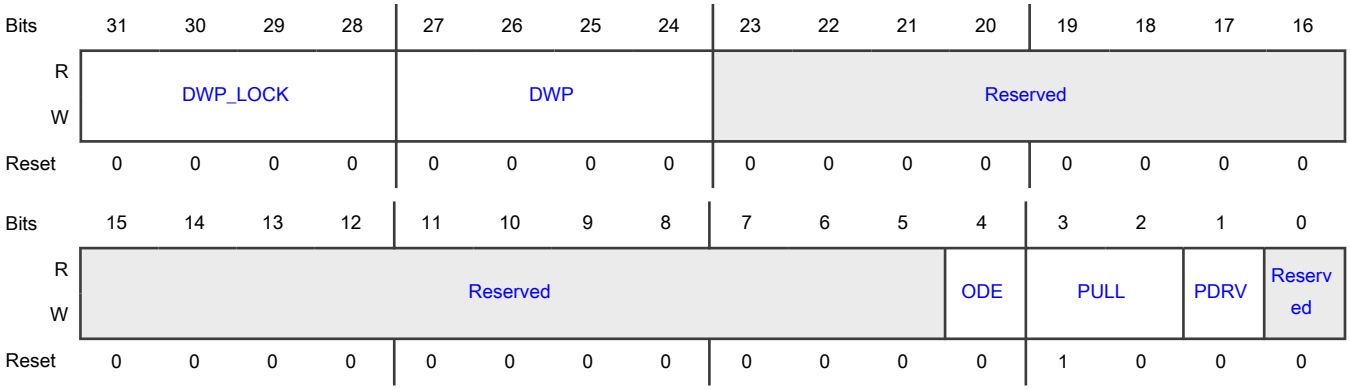
17.4.1.207 SW_PAD_CTL_PAD_GPIO_EMC_B2_17 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_17)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_17	344h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_17 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_17 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_17 0b - high driver 1b - normal driver
0 —	- Reserved

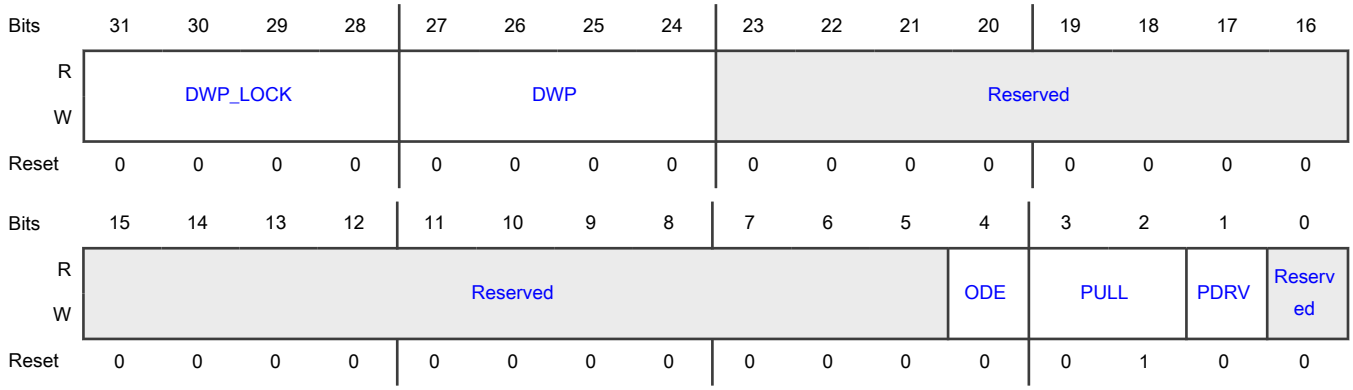
17.4.1.208 SW_PAD_CTL_PAD_GPIO_EMC_B2_18 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_18)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_18	348h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_18 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_18 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_18

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

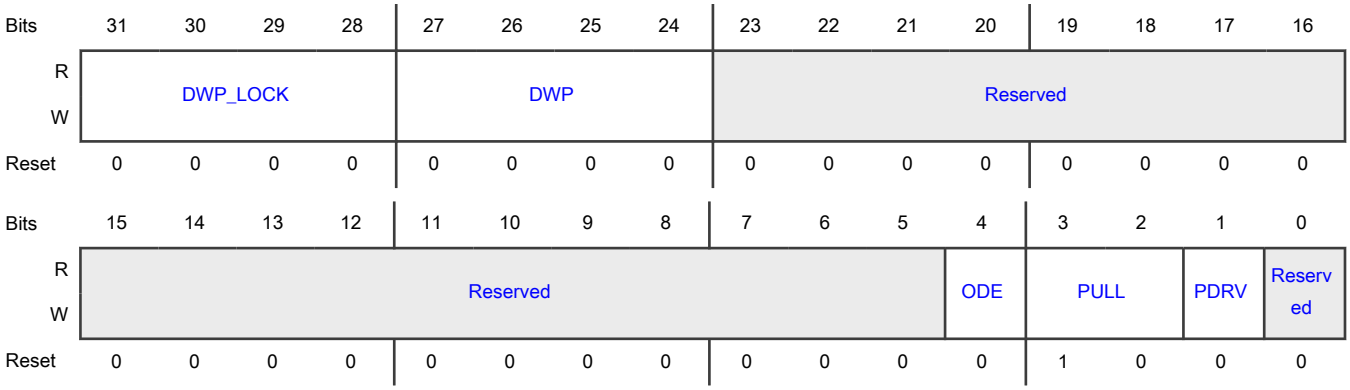
17.4.1.209 SW_PAD_CTL_PAD_GPIO_EMC_B2_19 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_19)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_19	34Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_19 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_19 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_19 0b - high driver 1b - normal driver
0 —	- Reserved

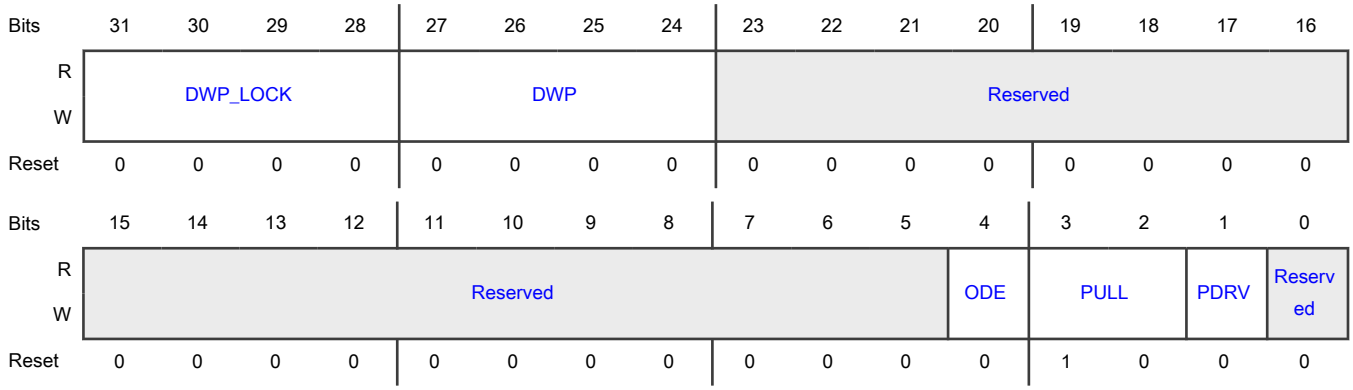
17.4.1.210 SW_PAD_CTL_PAD_GPIO_EMC_B2_20 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_EMC_B2_20)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_EMC_B2_20	350h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_EMC_B2_20 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_EMC_B2_20 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_EMC_B2_20

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

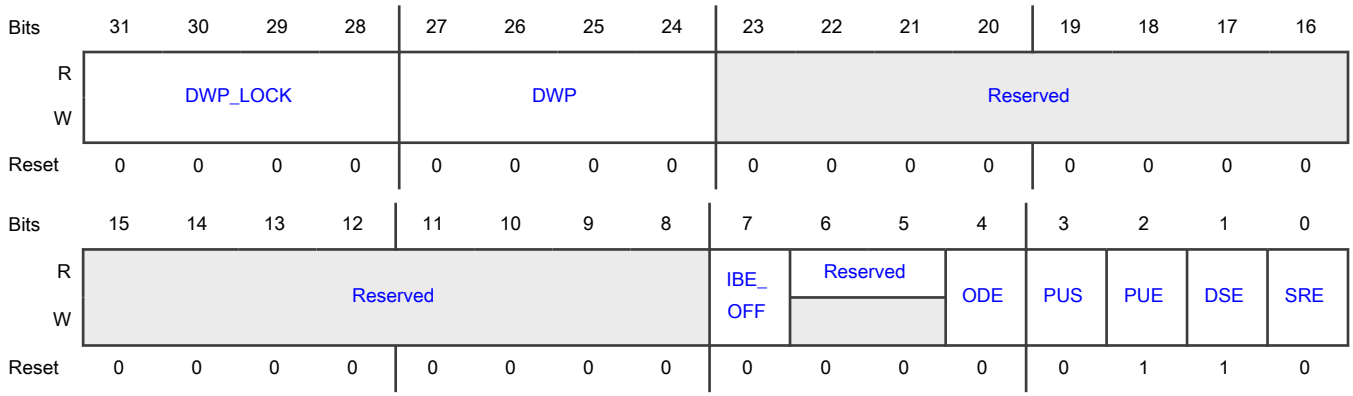
17.4.1.211 SW_PAD_CTL_PAD_GPIO_AD_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_00)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_00	354h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_00 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_00 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_00 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_00 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_00 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_00 0b - Fast Slew Rate 1b - Slow Slew Rate

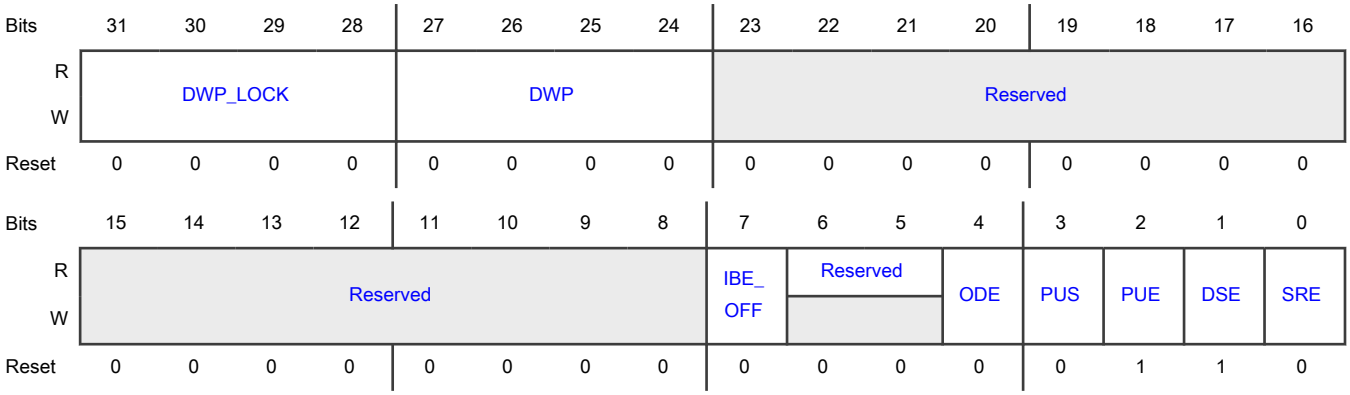
17.4.1.212 SW_PAD_CTL_PAD_GPIO_AD_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_01)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_01	358h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_01 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_01 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_01 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_01 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_01 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_01 0b - Fast Slew Rate 1b - Slow Slew Rate

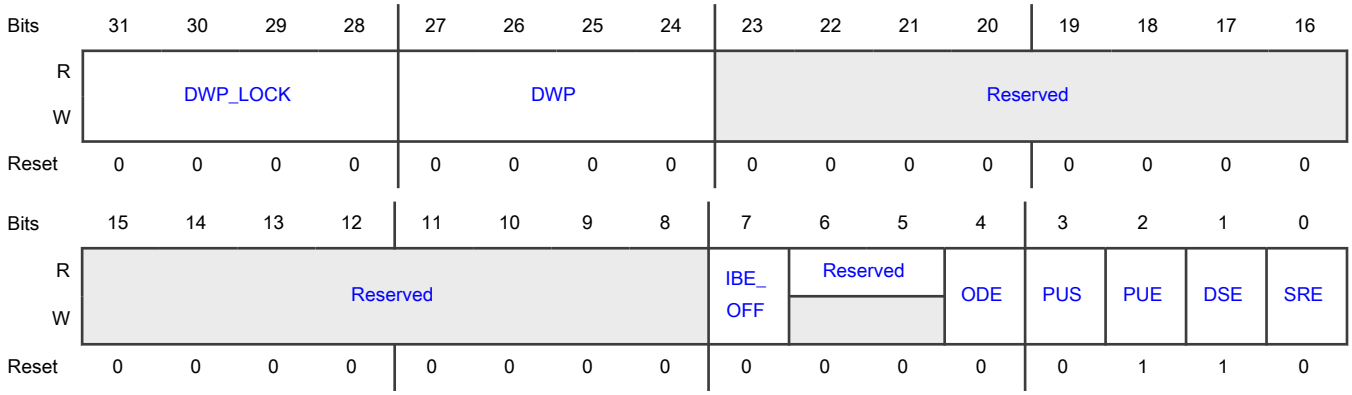
17.4.1.213 SW_PAD_CTL_PAD_GPIO_AD_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_02)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_02	35Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_02 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_02 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_02

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_02 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_02 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_02 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.1.214 SW_PAD_CTL_PAD_GPIO_AD_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_03)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_03	360h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IBE_OFF	Reserved		ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_03 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_03 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_03 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_03 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_03 0b - normal driver

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_03 0b - Fast Slew Rate 1b - Slow Slew Rate

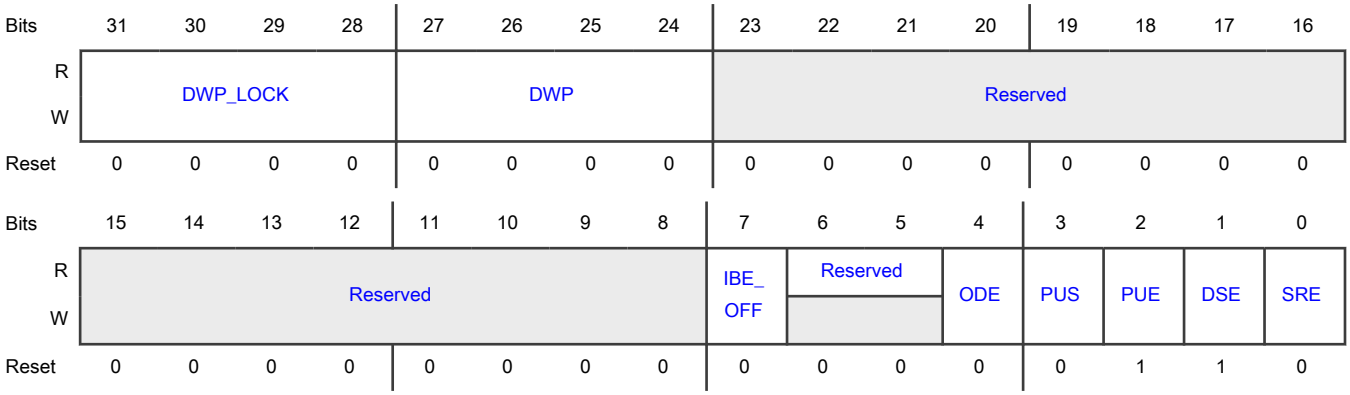
17.4.1.215 SW_PAD_CTL_PAD_GPIO_AD_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_04)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_04	364h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain

Table continues on the next page...

Table continued from the previous page...

Field	Function
	whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_04 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_04 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_04 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_04 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_04 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_04 0b - Fast Slew Rate 1b - Slow Slew Rate

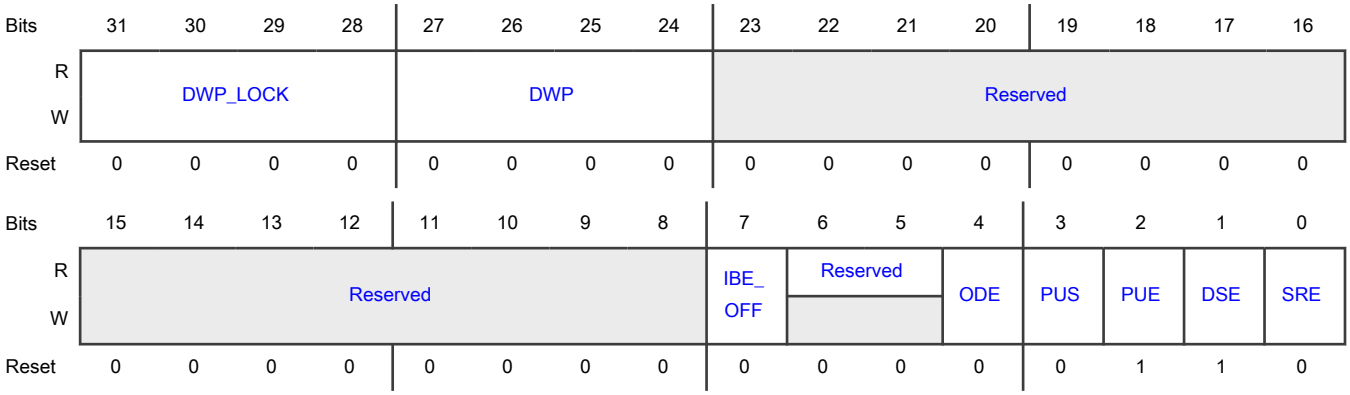
17.4.1.216 SW_PAD_CTL_PAD_GPIO_AD_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_05)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_05	368h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_05 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_05 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_05 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_05 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_05 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_05 0b - Fast Slew Rate 1b - Slow Slew Rate

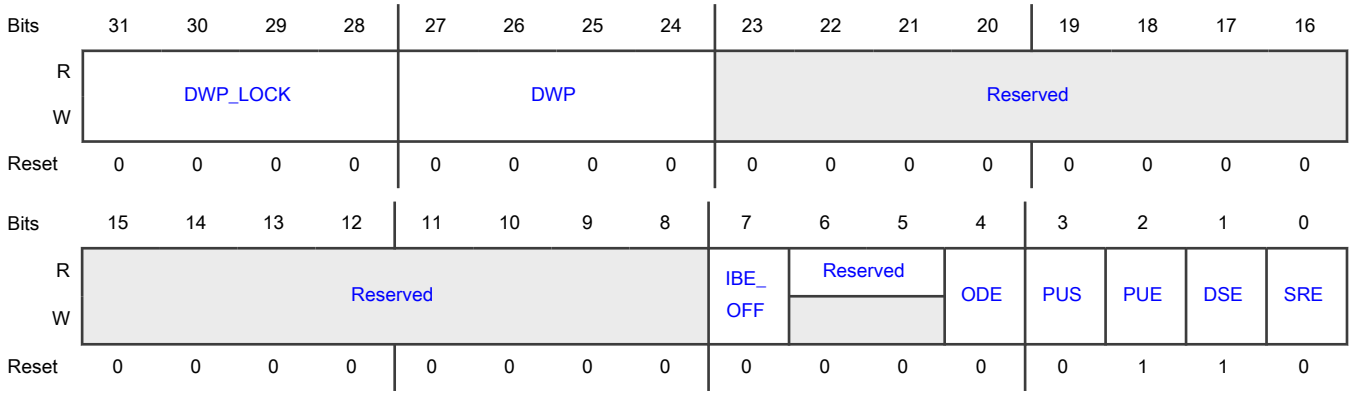
17.4.1.217 SW_PAD_CTL_PAD_GPIO_AD_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_06)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_06	36Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_06 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_06 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_06

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_06 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_06 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_06 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.1.218 SW_PAD_CTL_PAD_GPIO_AD_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_07)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_07	370h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IBE_OFF	Reserved		ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_07 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_07 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_07 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_07 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_07 0b - normal driver

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_07 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.1.219 SW_PAD_CTL_PAD_GPIO_AD_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_08)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_08	374h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IBE_OFF	Reserved		ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain

Table continues on the next page...

Table continued from the previous page...

Field	Function
	whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_08 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_08 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_08 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_08 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_08 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_08 0b - Fast Slew Rate 1b - Slow Slew Rate

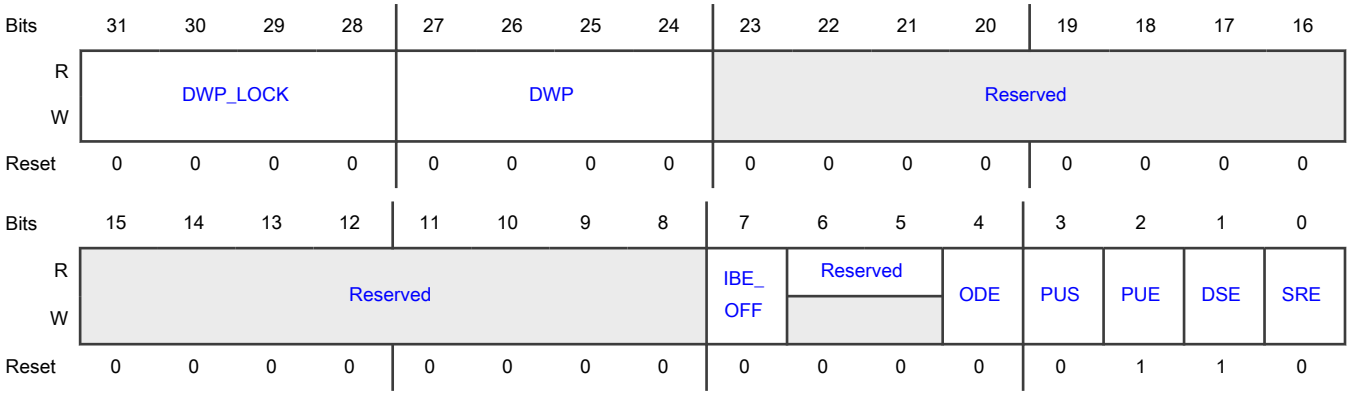
17.4.1.220 SW_PAD_CTL_PAD_GPIO_AD_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_09)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_09	378h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_09 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_09 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_09 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_09 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_09 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_09 0b - Fast Slew Rate 1b - Slow Slew Rate

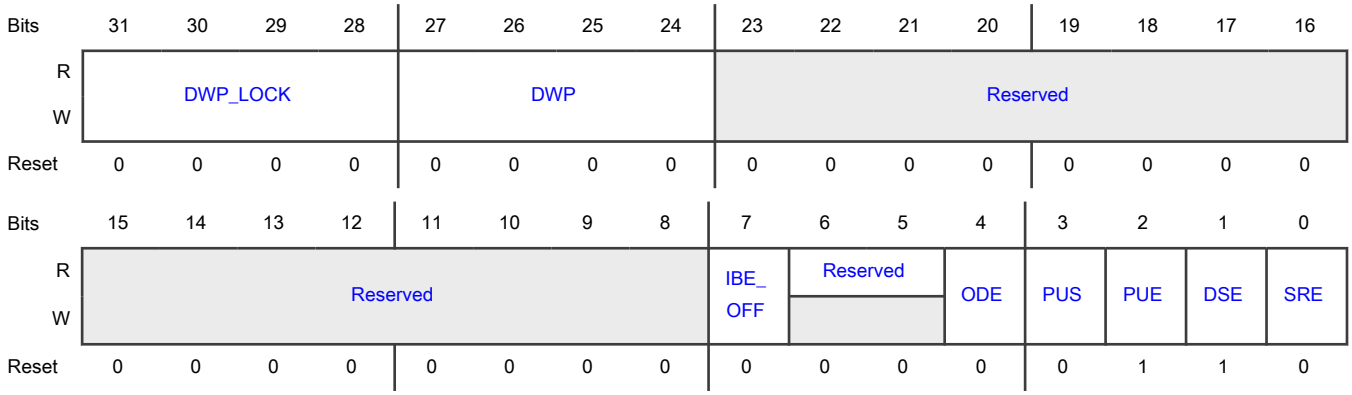
17.4.1.221 SW_PAD_CTL_PAD_GPIO_AD_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_10)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_10	37Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_10 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_10 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_10

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_10 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_10 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_10 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.1.222 SW_PAD_CTL_PAD_GPIO_AD_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_11)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_11	380h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IBE_OFF	Reserved		ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_11 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_11 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_11 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_11 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_11 0b - normal driver

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_11 0b - Fast Slew Rate 1b - Slow Slew Rate

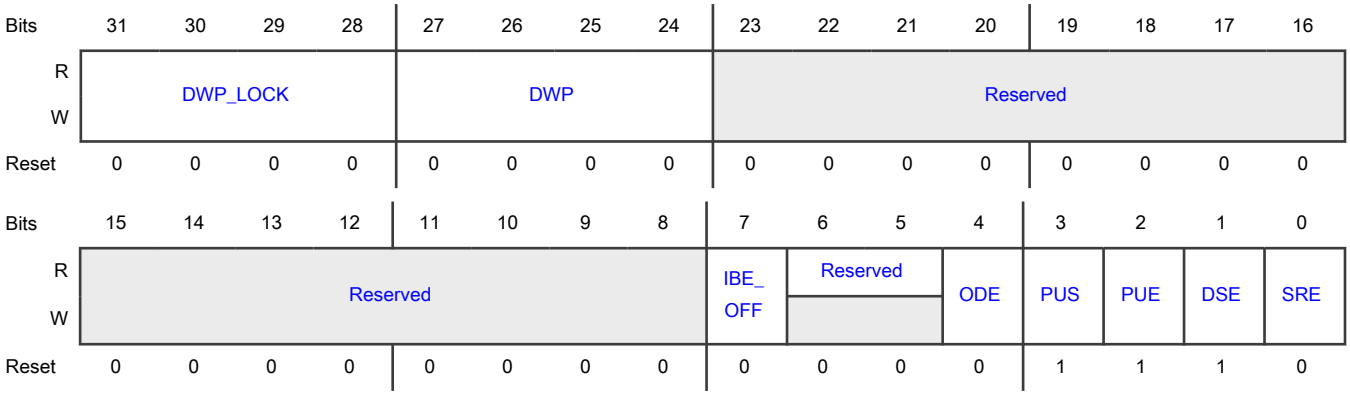
17.4.1.223 SW_PAD_CTL_PAD_GPIO_AD_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_12)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_12	384h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain

Table continues on the next page...

Table continued from the previous page...

Field	Function
	whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_12 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_12 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_12 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_12 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_12 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_12 0b - Fast Slew Rate 1b - Slow Slew Rate

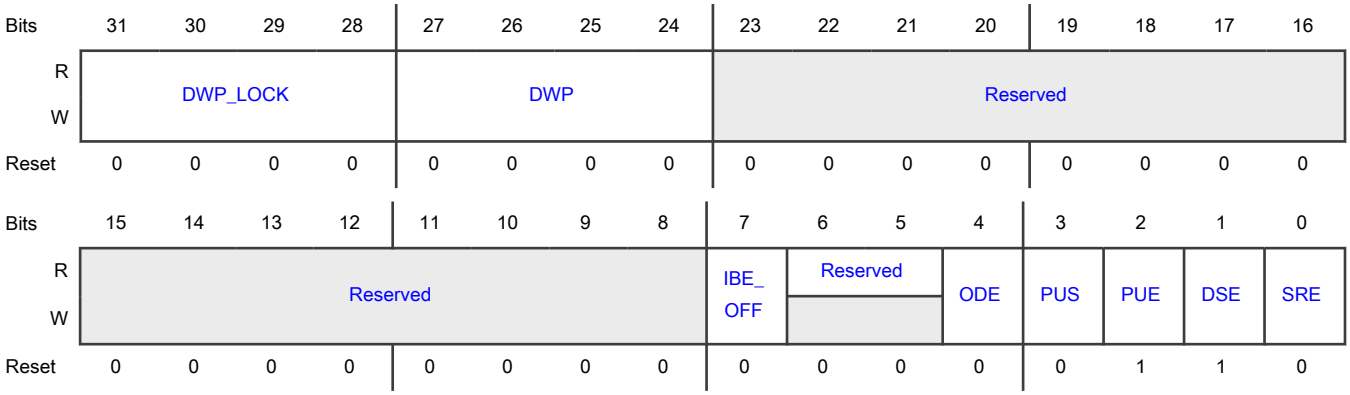
17.4.1.224 SW_PAD_CTL_PAD_GPIO_AD_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_13)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_13	388h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_13 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_13 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_13 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_13 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_13 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_13 0b - Fast Slew Rate 1b - Slow Slew Rate

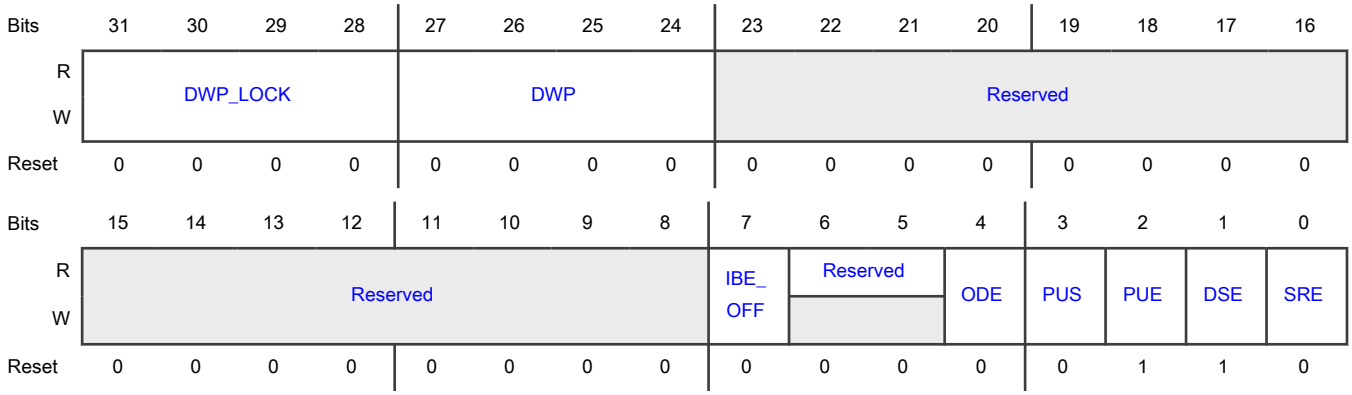
17.4.1.225 SW_PAD_CTL_PAD_GPIO_AD_14 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_14)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_14	38Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_14 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_14 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_14

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_14 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_14 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_14 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.1.226 SW_PAD_CTL_PAD_GPIO_AD_15 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_15)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_15	390h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IBE_OFF	Reserved		ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_15 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_15 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_15 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_15 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_15 0b - normal driver

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_15 0b - Fast Slew Rate 1b - Slow Slew Rate

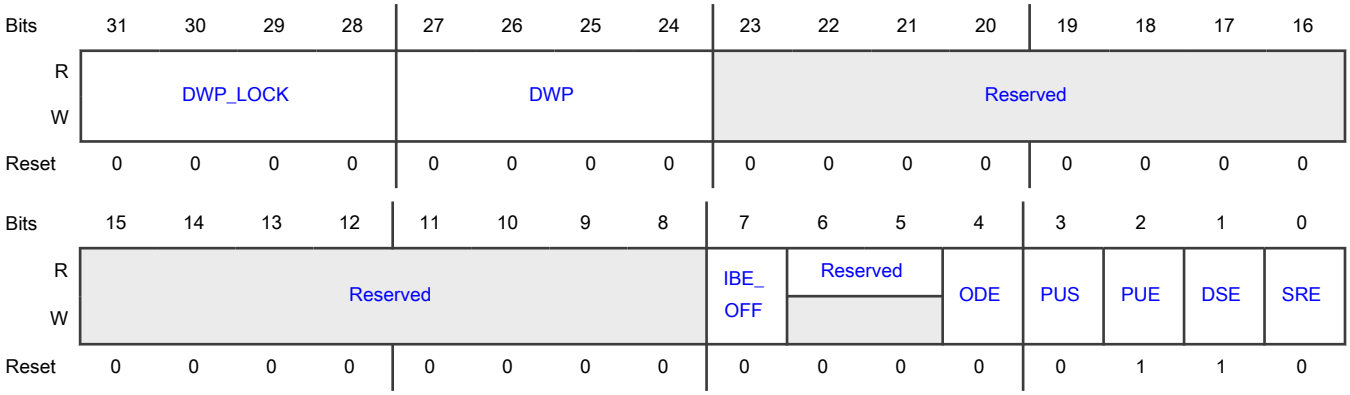
17.4.1.227 SW_PAD_CTL_PAD_GPIO_AD_16 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_16)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_16	394h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain

Table continues on the next page...

Table continued from the previous page...

Field	Function
	whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_16 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_16 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_16 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_16 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_16 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_16 0b - Fast Slew Rate 1b - Slow Slew Rate

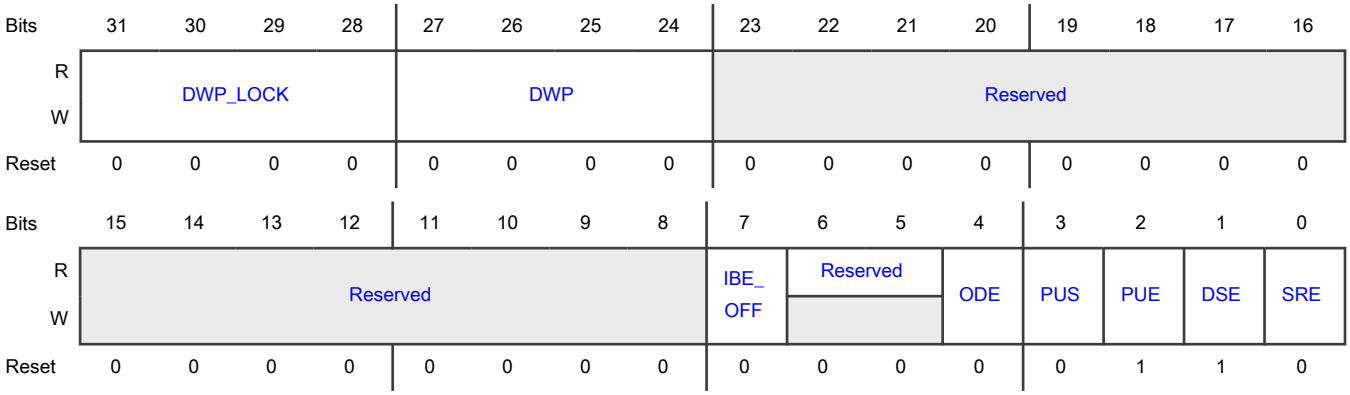
17.4.1.228 SW_PAD_CTL_PAD_GPIO_AD_17 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_17)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_17	398h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_17 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_17 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_17 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_17 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_17 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_17 0b - Fast Slew Rate 1b - Slow Slew Rate

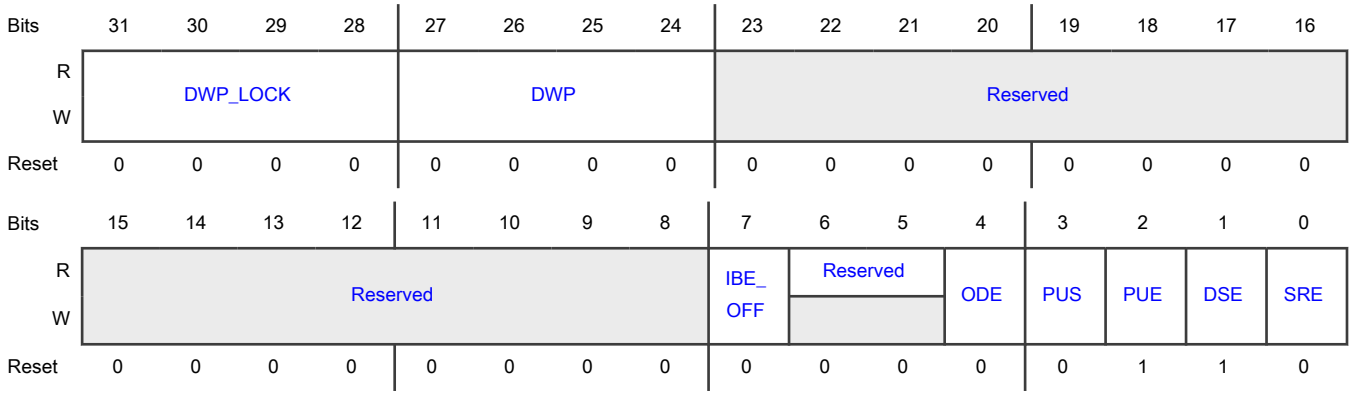
17.4.1.229 SW_PAD_CTL_PAD_GPIO_AD_18 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_18)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_18	39Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_18 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_18 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_18

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_18 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_18 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_18 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.1.230 SW_PAD_CTL_PAD_GPIO_AD_19 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_19)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_19	3A0h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IBE_OFF	Reserved		ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_19 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_19 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_19 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_19 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_19 0b - normal driver

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_19 0b - Fast Slew Rate 1b - Slow Slew Rate

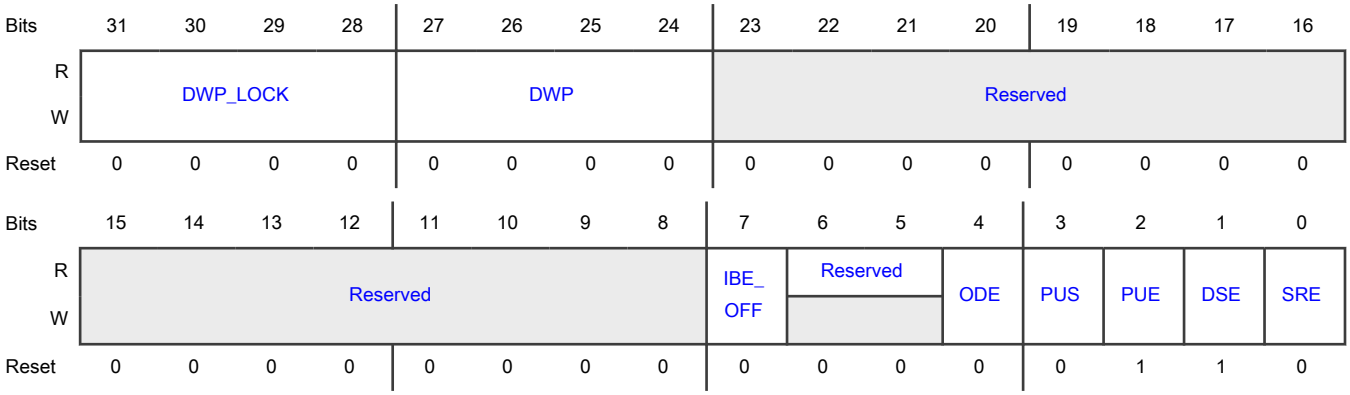
17.4.1.231 SW_PAD_CTL_PAD_GPIO_AD_20 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_20)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_20	3A4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain

Table continues on the next page...

Table continued from the previous page...

Field	Function
	whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_20 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_20 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_20 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_20 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_20 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_20 0b - Fast Slew Rate 1b - Slow Slew Rate

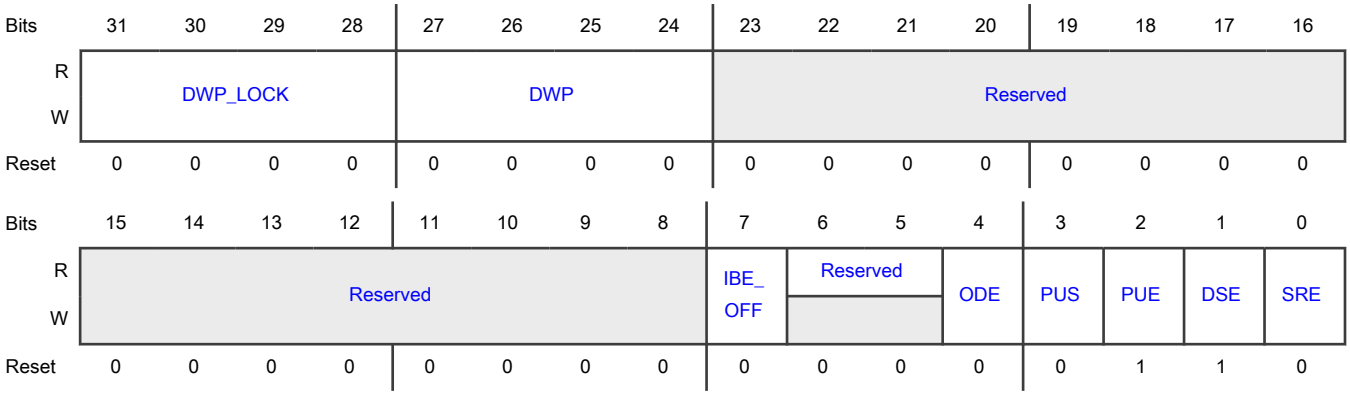
17.4.1.232 SW_PAD_CTL_PAD_GPIO_AD_21 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_21)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_21	3A8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_21 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_21 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_21 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_21 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_21 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_21 0b - Fast Slew Rate 1b - Slow Slew Rate

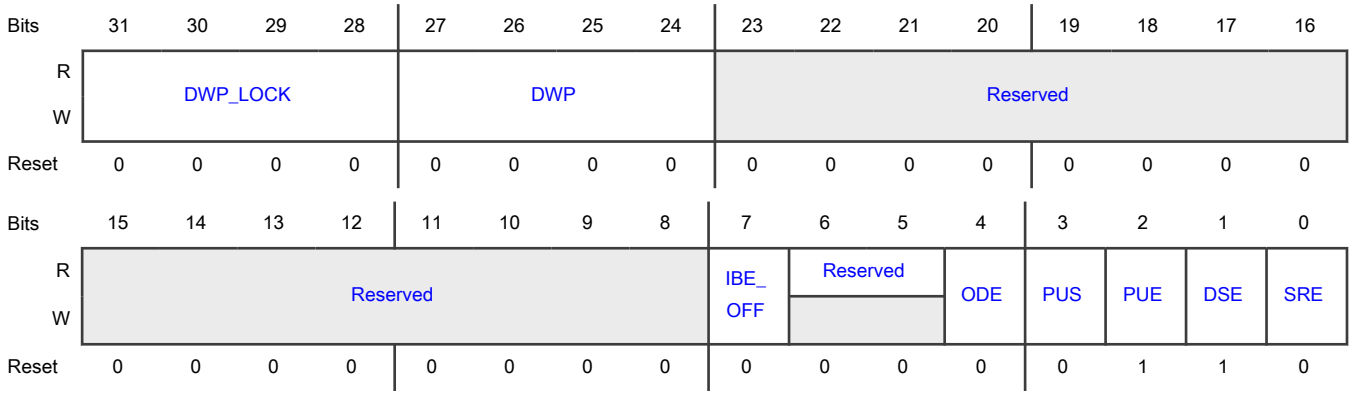
17.4.1.233 SW_PAD_CTL_PAD_GPIO_AD_22 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_22)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_22	3ACh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force Input Buffer Enable (IBE) off Field Select one out of next values for pad: GPIO_AD_22 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_22 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_22

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_22 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_22 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_22 0b - Fast Slew Rate 1b - Slow Slew Rate

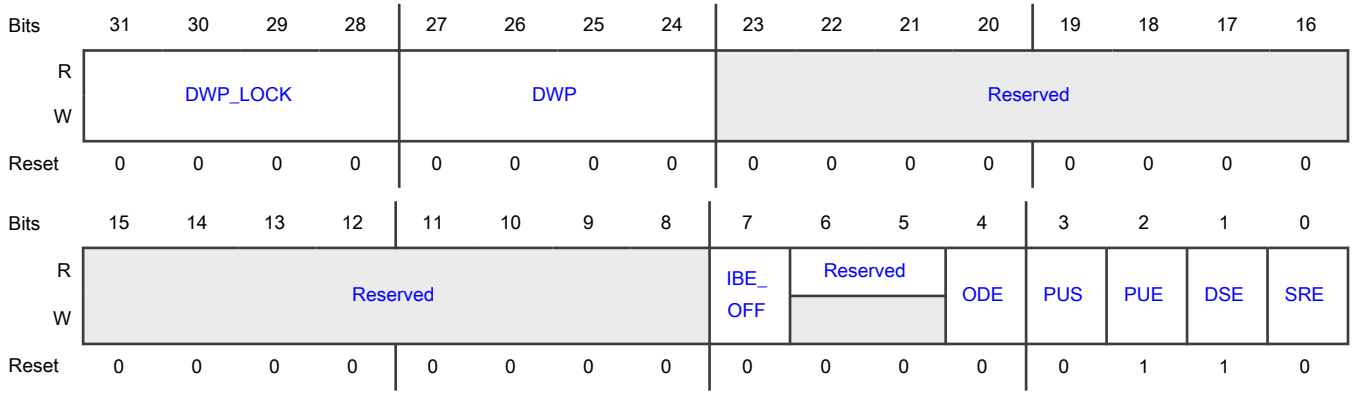
17.4.1.234 SW_PAD_CTL_PAD_GPIO_AD_23 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_23)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_23	3B0h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_23 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_23 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_23 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_23 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_23 0b - normal driver

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_23 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.1.235 SW_PAD_CTL_PAD_GPIO_AD_24 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_24)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_24	3B4h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IBE_OFF	Reserved		ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain

Table continues on the next page...

Table continued from the previous page...

Field	Function
	whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_24 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_24 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_24 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_24 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_24 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_24 0b - Fast Slew Rate 1b - Slow Slew Rate

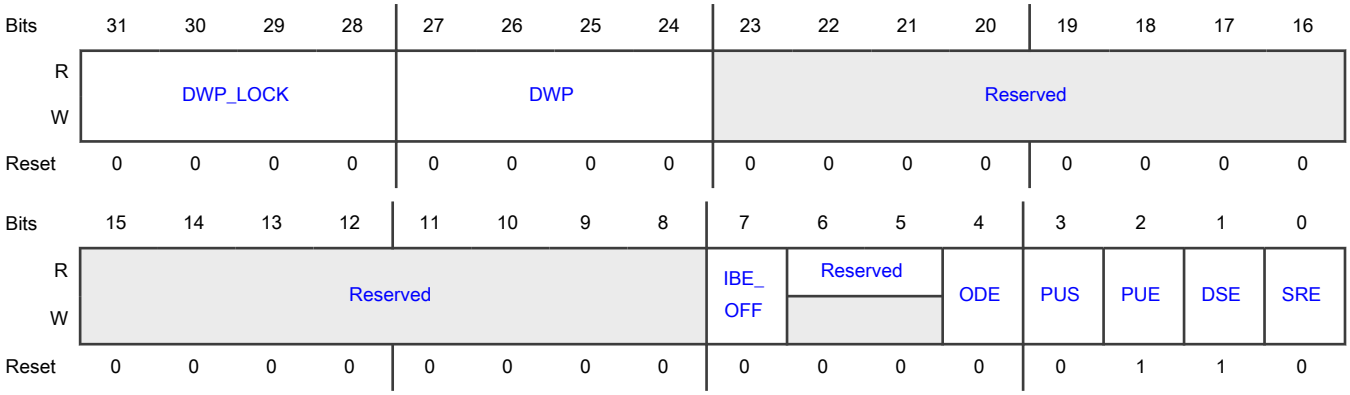
17.4.1.236 SW_PAD_CTL_PAD_GPIO_AD_25 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_25)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_25	3B8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_25 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_25 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_25 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_25 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_25 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_25 0b - Fast Slew Rate 1b - Slow Slew Rate

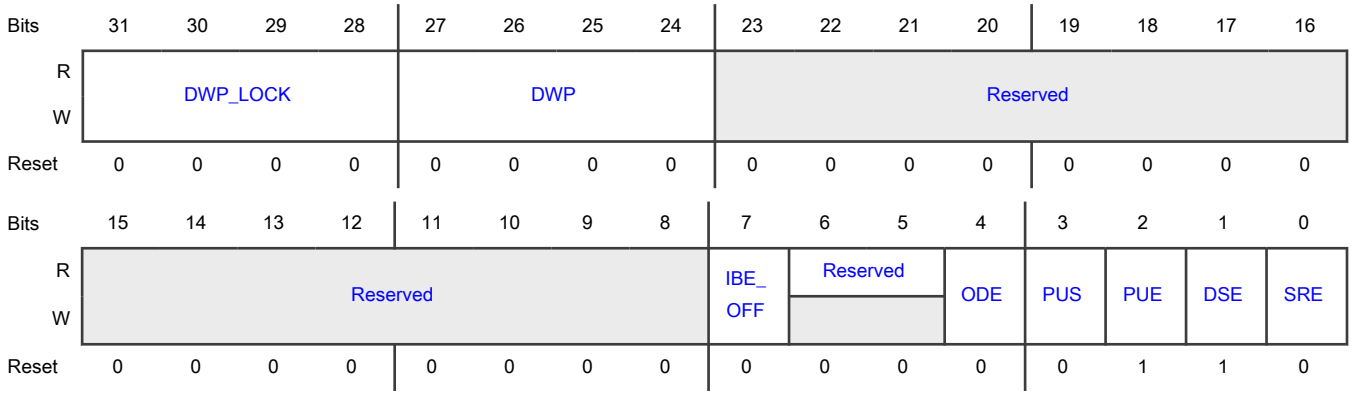
17.4.1.237 SW_PAD_CTL_PAD_GPIO_AD_26 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_26)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_26	3BCh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_26 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_26 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_26

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_26 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_26 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_26 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.1.238 SW_PAD_CTL_PAD_GPIO_AD_27 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_27)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_27	3C0h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IBE_OFF	Reserved		ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_27 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_27 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_27 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_27 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_27 0b - normal driver

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_27 0b - Fast Slew Rate 1b - Slow Slew Rate

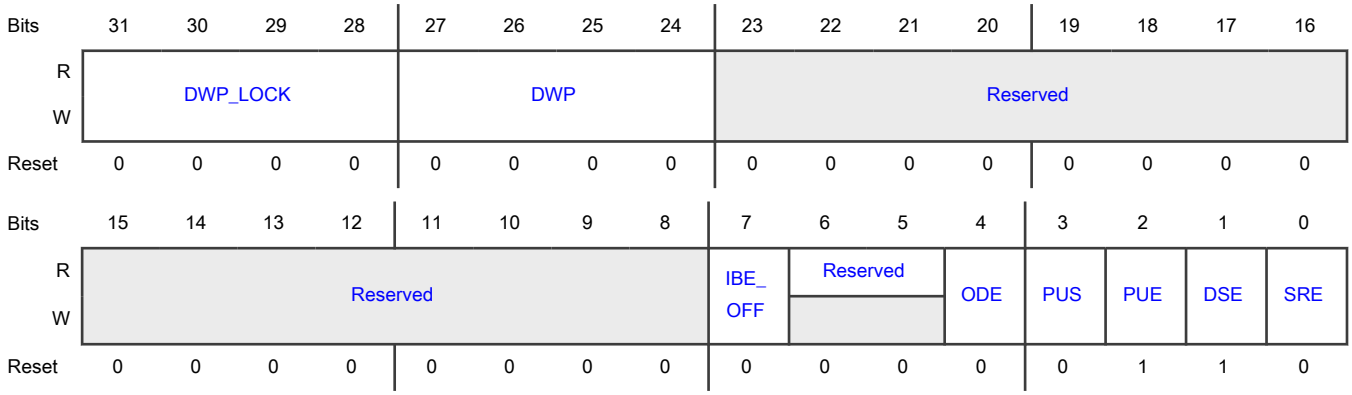
17.4.1.239 SW_PAD_CTL_PAD_GPIO_AD_28 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_28)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_28	3C4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain

Table continues on the next page...

Table continued from the previous page...

Field	Function
	whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_28 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_28 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_28 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_28 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_28 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_28 0b - Fast Slew Rate 1b - Slow Slew Rate

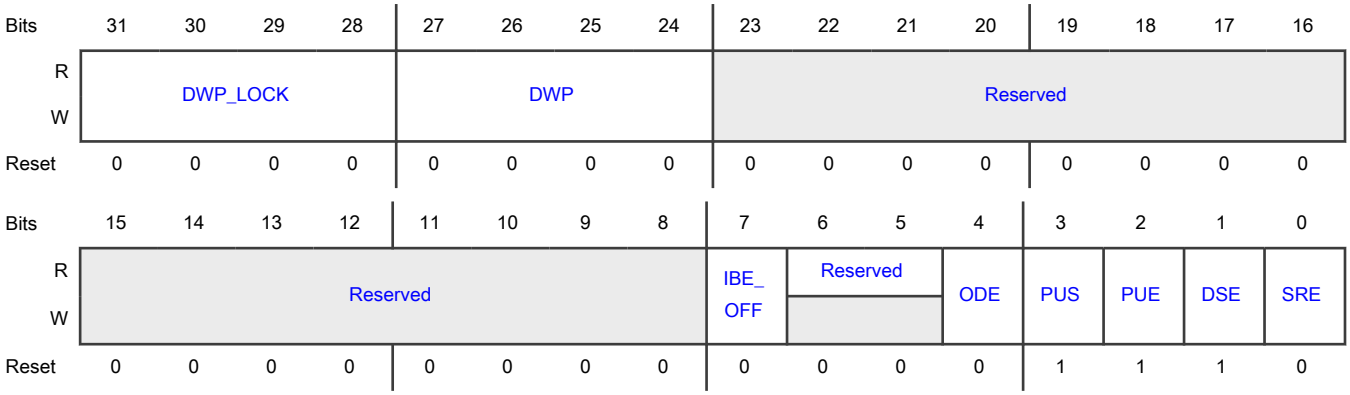
17.4.1.240 SW_PAD_CTL_PAD_GPIO_AD_29 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_29)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_29	3C8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_29 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_29 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_29 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_29 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_29 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_29 0b - Fast Slew Rate 1b - Slow Slew Rate

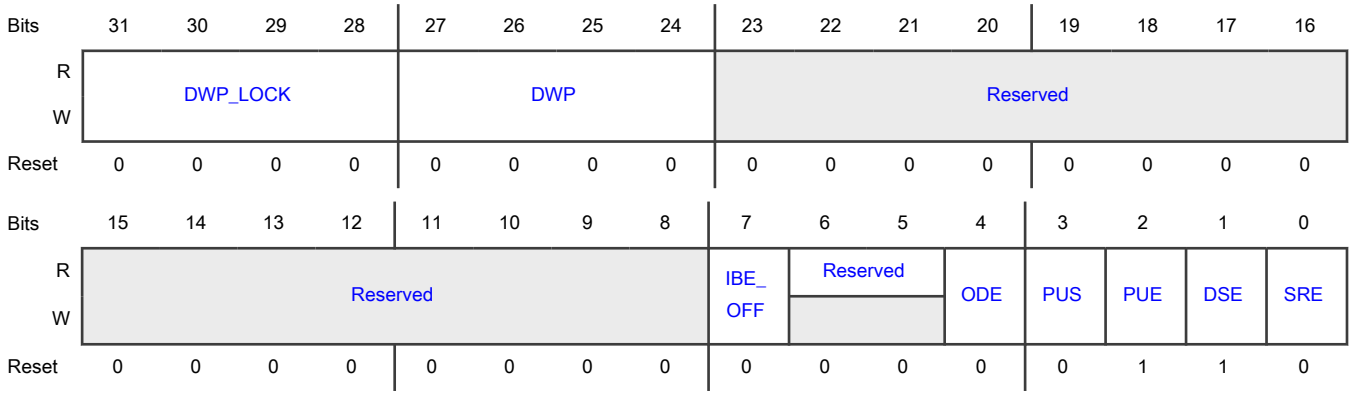
17.4.1.241 SW_PAD_CTL_PAD_GPIO_AD_30 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_30)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_30	3CCh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_30 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_30 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_30

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_30 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_30 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_30 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.1.242 SW_PAD_CTL_PAD_GPIO_AD_31 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_31)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_31	3D0h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IBE_OFF	Reserved		ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_31 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_31 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_31 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_31 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_31 0b - normal driver

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_31 0b - Fast Slew Rate 1b - Slow Slew Rate

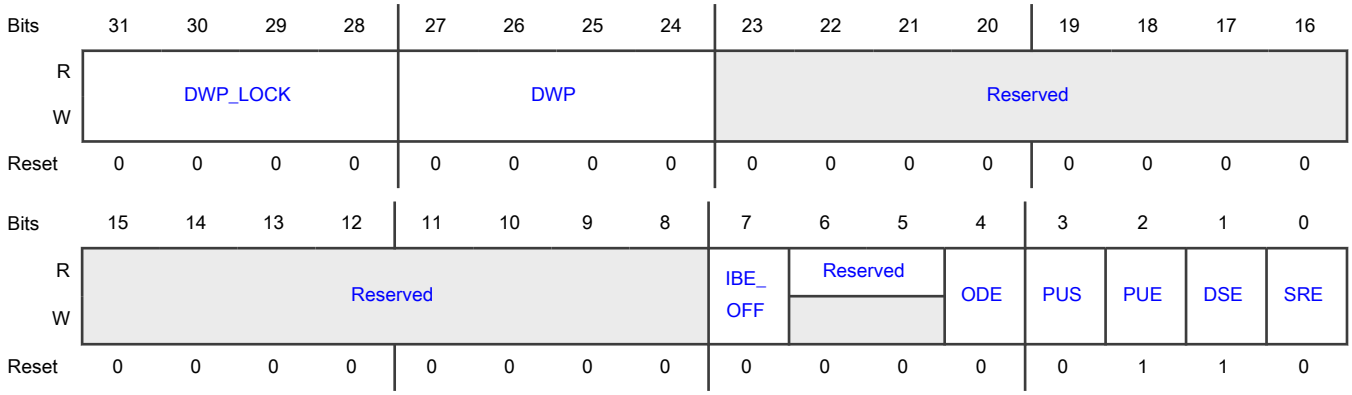
17.4.1.243 SW_PAD_CTL_PAD_GPIO_AD_32 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_32)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_32	3D4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain

Table continues on the next page...

Table continued from the previous page...

Field	Function
	whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_32 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_32 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_32 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_32 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_32 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_32 0b - Fast Slew Rate 1b - Slow Slew Rate

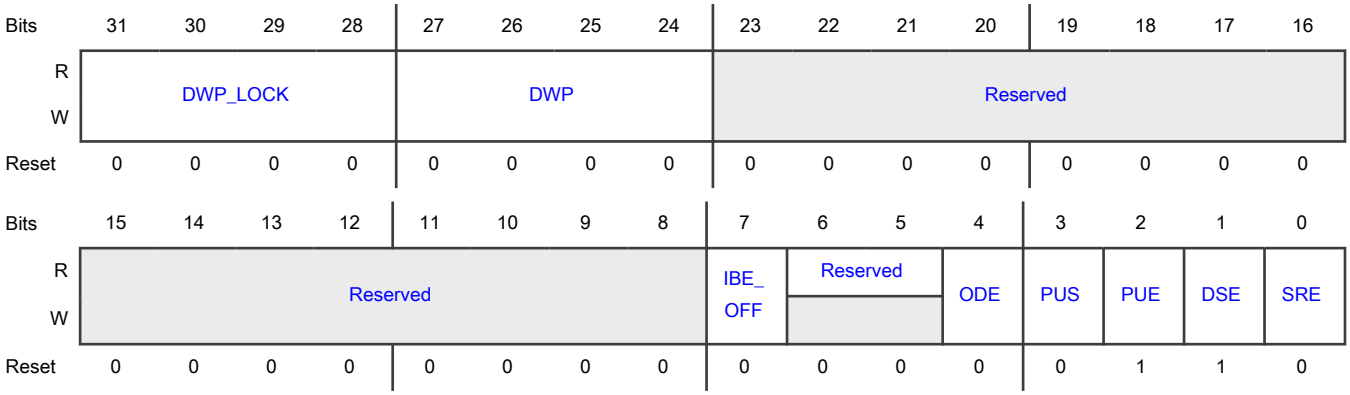
17.4.1.244 SW_PAD_CTL_PAD_GPIO_AD_33 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_33)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_33	3D8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_33 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_33 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_33 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_33 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_33 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_33 0b - Fast Slew Rate 1b - Slow Slew Rate

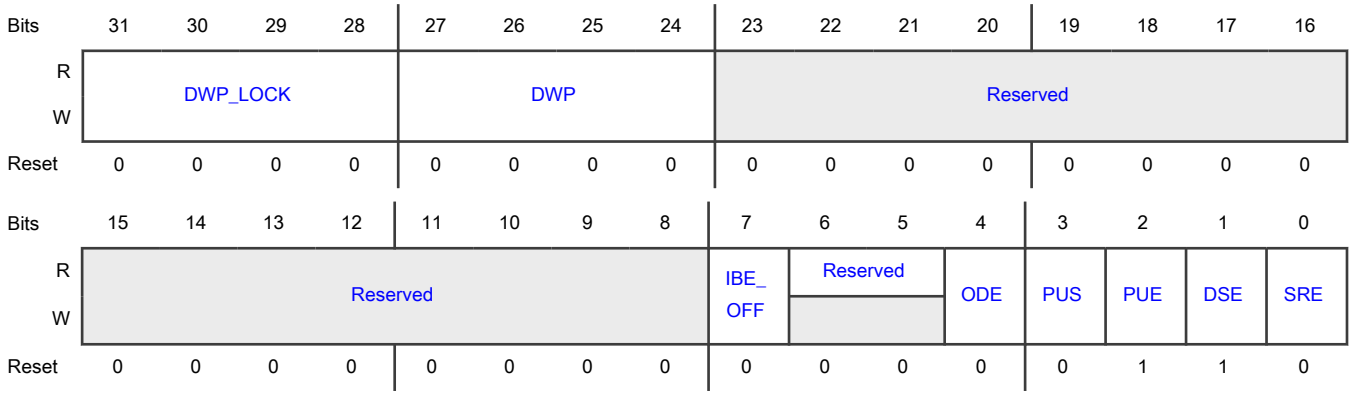
17.4.1.245 SW_PAD_CTL_PAD_GPIO_AD_34 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_34)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_34	3DCh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_34 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_34 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_34

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_34 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_34 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_34 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.1.246 SW_PAD_CTL_PAD_GPIO_AD_35 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AD_35)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AD_35	3E0h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IBE_OFF	Reserved		ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-8 —	- Reserved
7 IBE_OFF	Force ibe off Field Select one out of next values for pad: GPIO_AD_35 0b - Disabled 1b - Enabled
6-5 —	Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AD_35 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AD_35 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AD_35 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AD_35 0b - normal driver

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AD_35 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.1.247 SW_PAD_CTL_PAD_GPIO_SD_B1_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_00)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B1_00	3E4h

Function

SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection

Table continues on the next page...

Table continued from the previous page...

Field	Function
	These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B1_00 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B1_00 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B1_00 0b - high driver 1b - normal driver
0 —	- Reserved

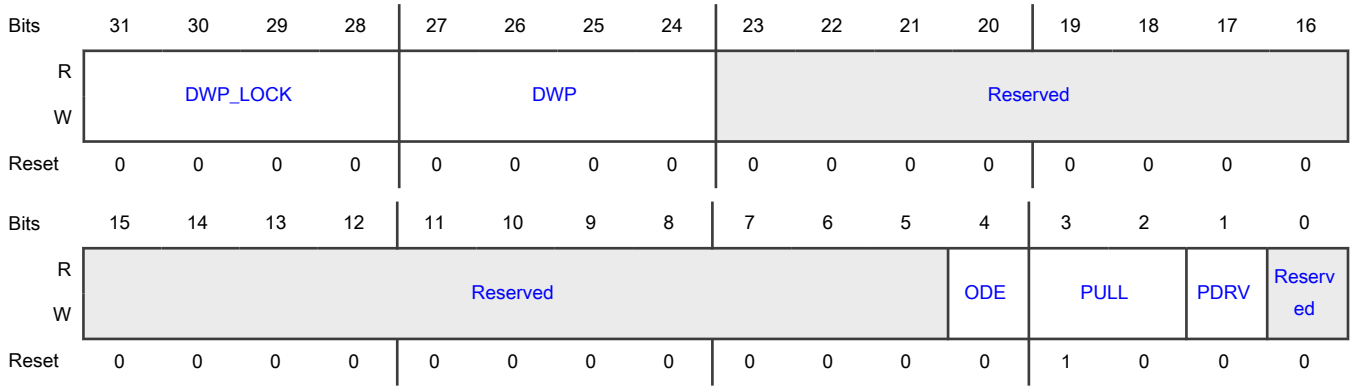
17.4.1.248 SW_PAD_CTL_PAD_GPIO_SD_B1_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_01)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B1_01	3E8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B1_01 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B1_01 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B1_01

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

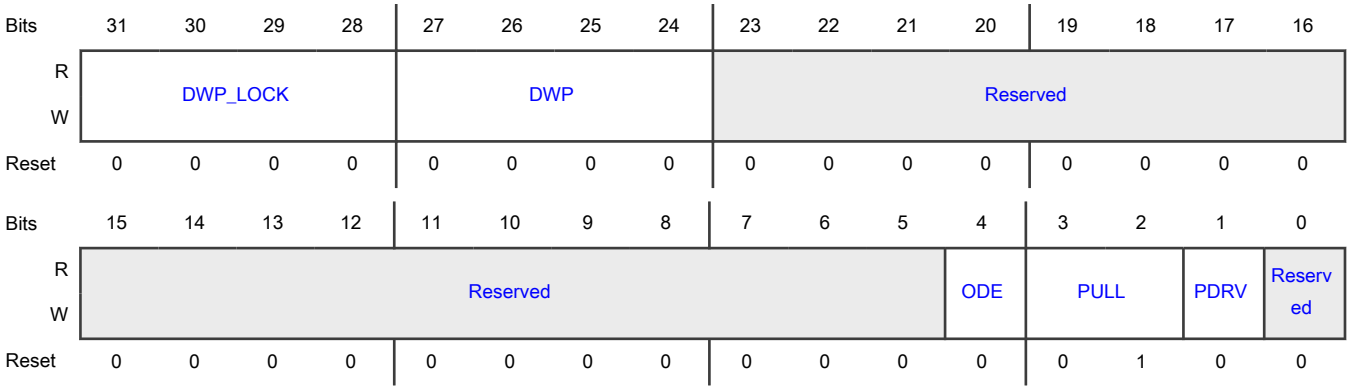
17.4.1.249 SW_PAD_CTL_PAD_GPIO_SD_B1_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_02)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B1_02	3ECh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B1_02 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B1_02 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B1_02 0b - high driver 1b - normal driver
0 —	- Reserved

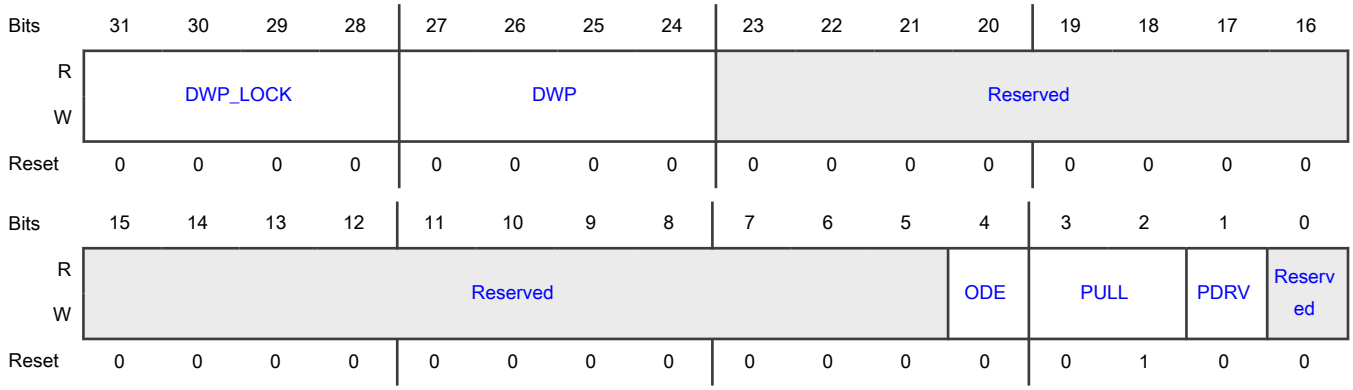
17.4.1.250 SW_PAD_CTL_PAD_GPIO_SD_B1_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_03)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B1_03	3F0h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B1_03 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B1_03 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B1_03

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

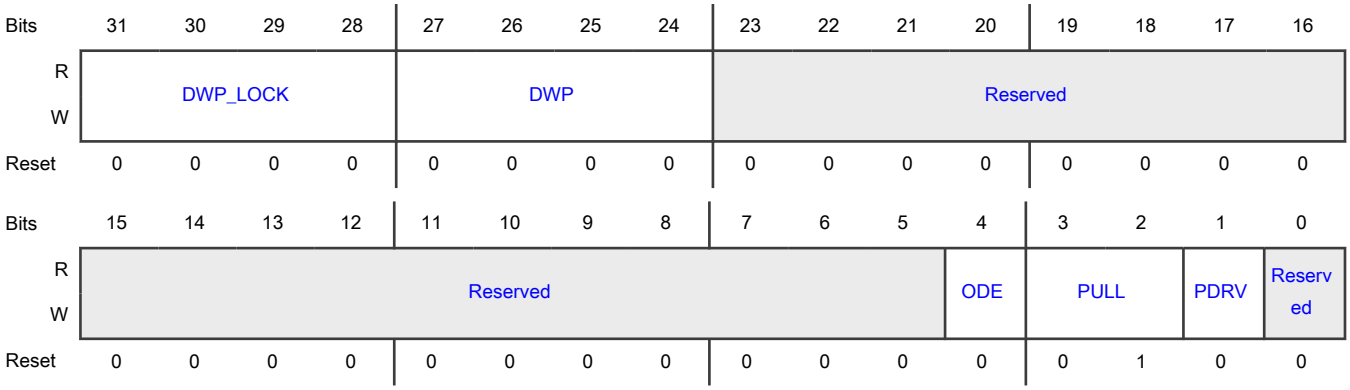
17.4.1.251 SW_PAD_CTL_PAD_GPIO_SD_B1_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_04)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B1_04	3F4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B1_04 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B1_04 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B1_04 0b - high driver 1b - normal driver
0 —	- Reserved

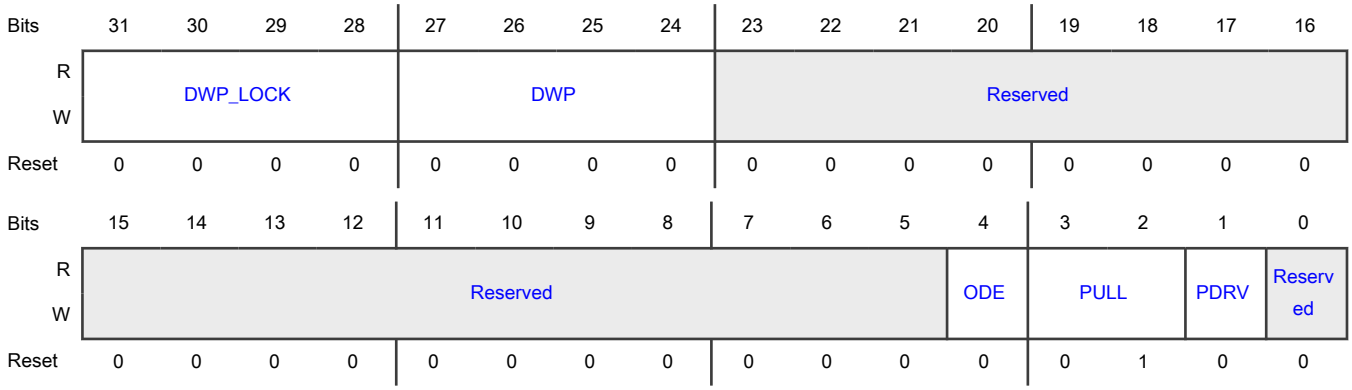
17.4.1.252 SW_PAD_CTL_PAD_GPIO_SD_B1_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B1_05)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B1_05	3F8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B1_05 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B1_05 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B1_05

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

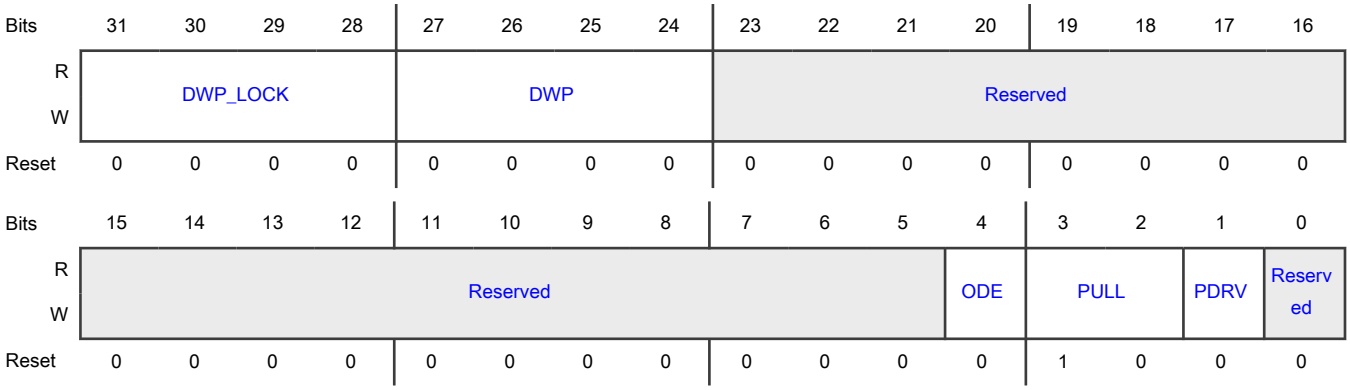
17.4.1.253 SW_PAD_CTL_PAD_GPIO_SD_B2_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_00)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_00	3FCh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_00 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_00 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_00 0b - high driver 1b - normal driver
0 —	- Reserved

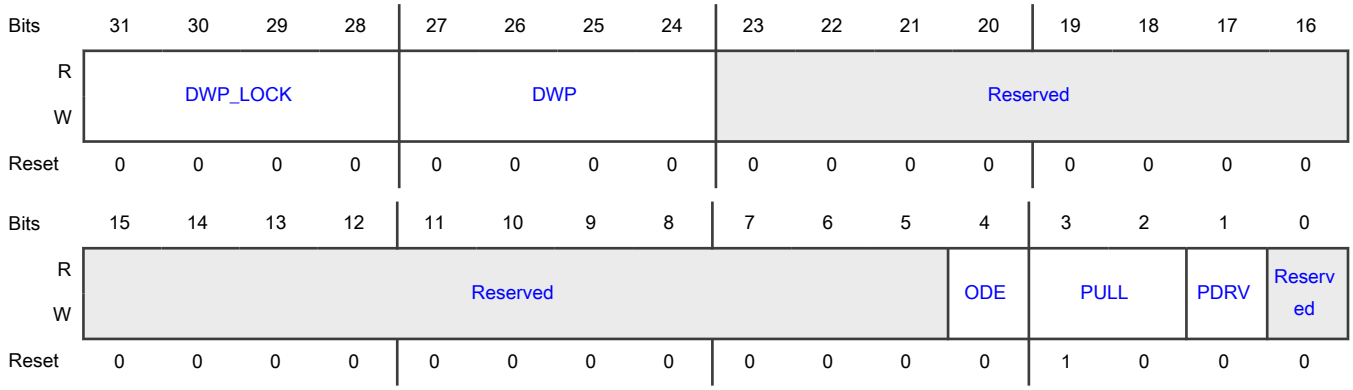
17.4.1.254 SW_PAD_CTL_PAD_GPIO_SD_B2_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_01)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_01	400h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_01 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_01 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_01

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

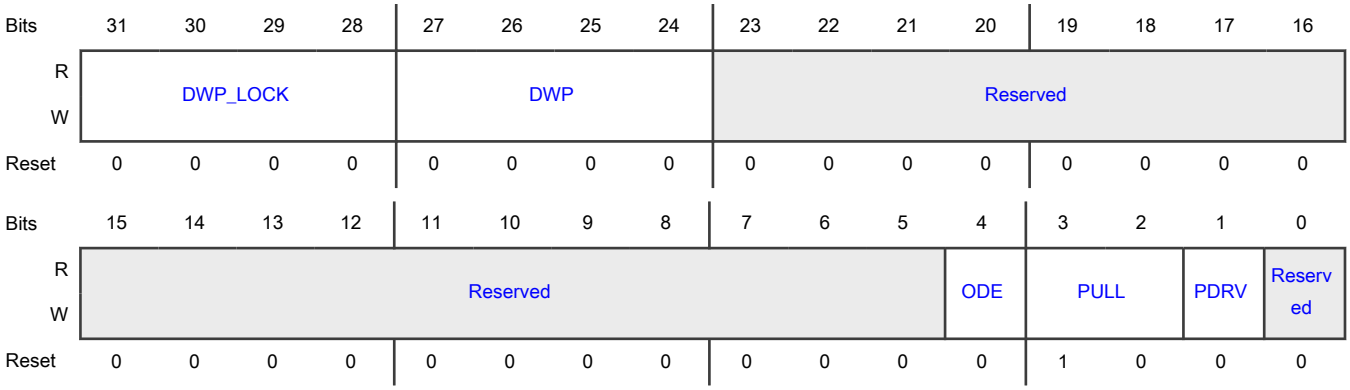
17.4.1.255 SW_PAD_CTL_PAD_GPIO_SD_B2_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_02)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_02	404h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_02 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_02 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_02 0b - high driver 1b - normal driver
0 —	- Reserved

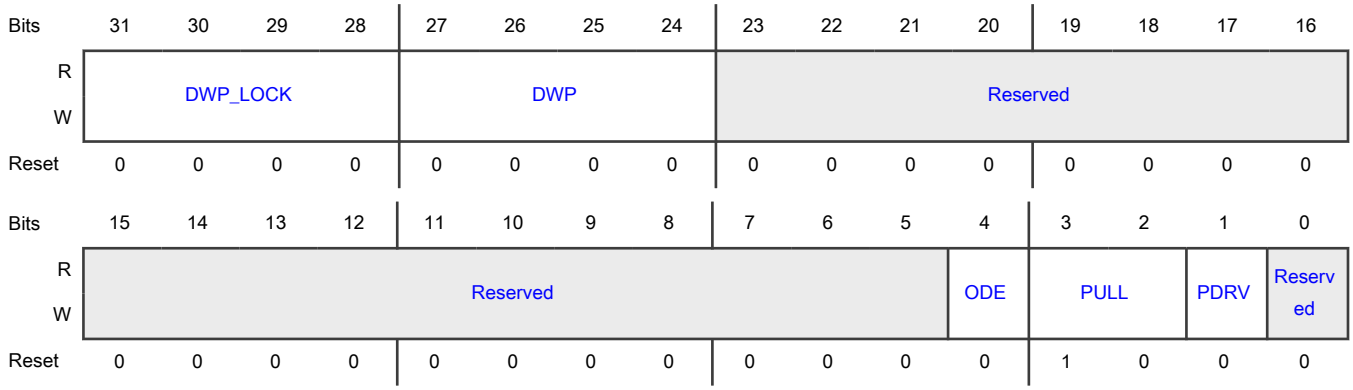
17.4.1.256 SW_PAD_CTL_PAD_GPIO_SD_B2_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_03)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_03	408h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_03 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_03 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_03

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

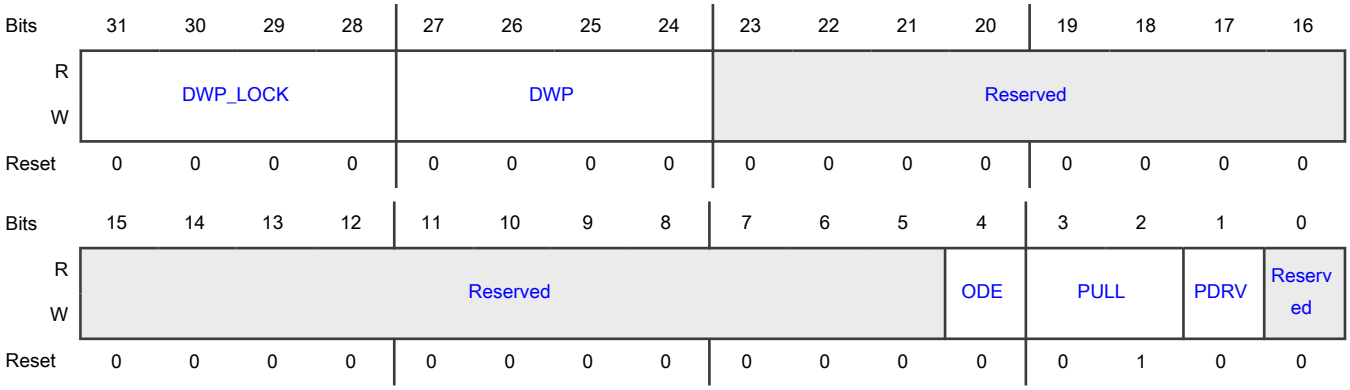
17.4.1.257 SW_PAD_CTL_PAD_GPIO_SD_B2_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_04)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_04	40Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_04 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_04 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_04 0b - high driver 1b - normal driver
0 —	- Reserved

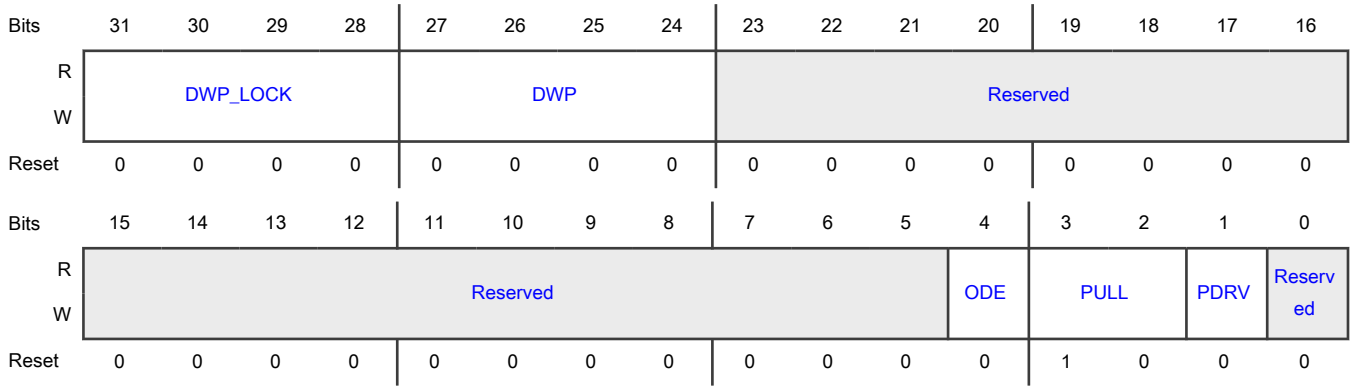
17.4.1.258 SW_PAD_CTL_PAD_GPIO_SD_B2_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_05)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_05	410h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_05 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_05 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_05

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

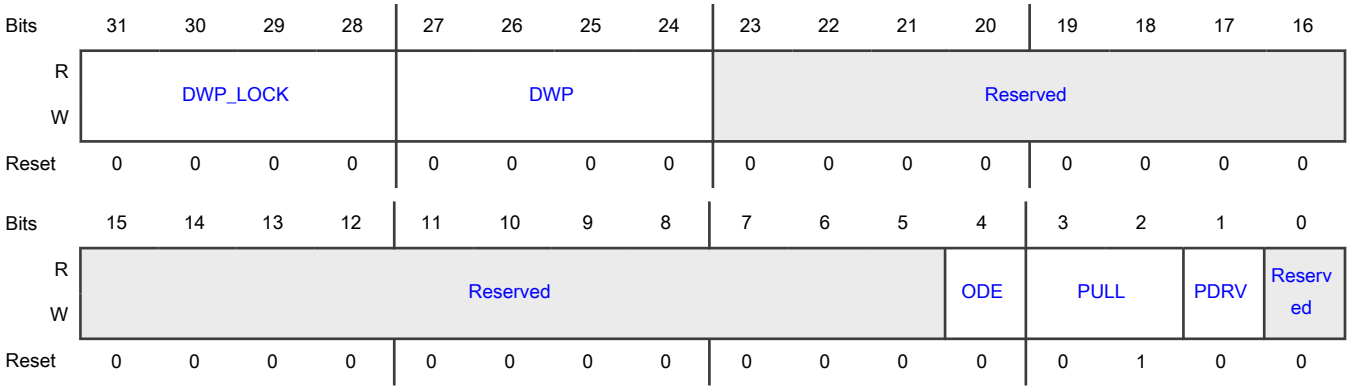
17.4.1.259 SW_PAD_CTL_PAD_GPIO_SD_B2_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_06)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_06	414h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_06 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_06 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_06 0b - high driver 1b - normal driver
0 —	- Reserved

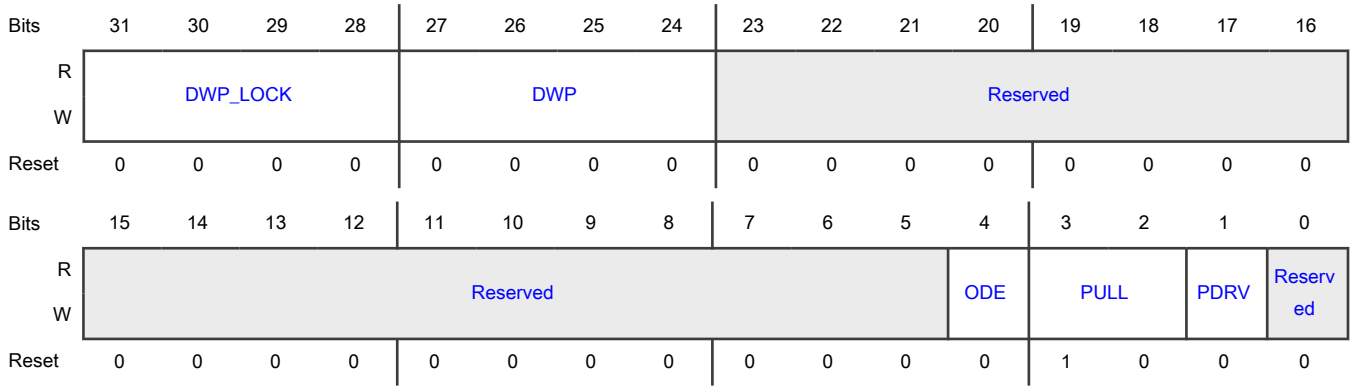
17.4.1.260 SW_PAD_CTL_PAD_GPIO_SD_B2_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_07)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_07	418h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_07 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_07 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_07

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

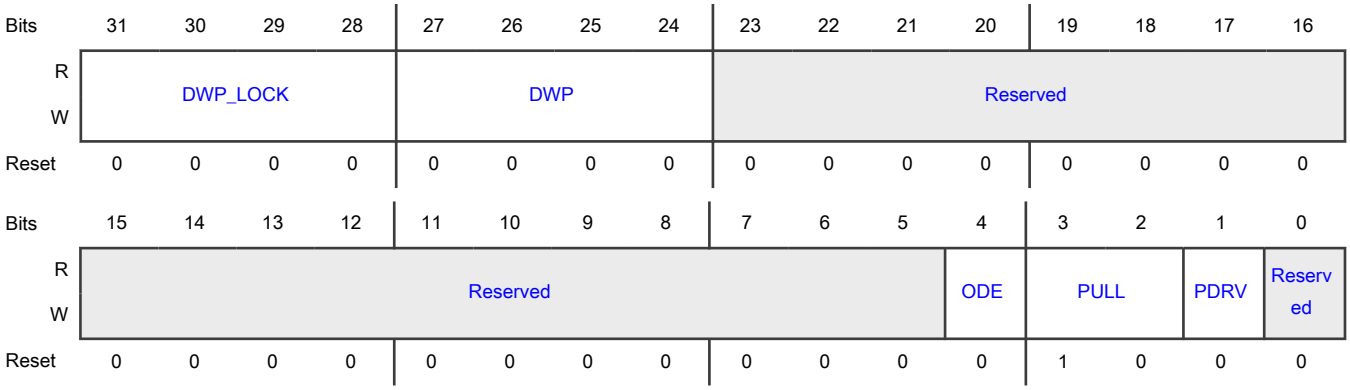
17.4.1.261 SW_PAD_CTL_PAD_GPIO_SD_B2_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_08)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_08	41Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_08 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_08 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_08 0b - high driver 1b - normal driver
0 —	- Reserved

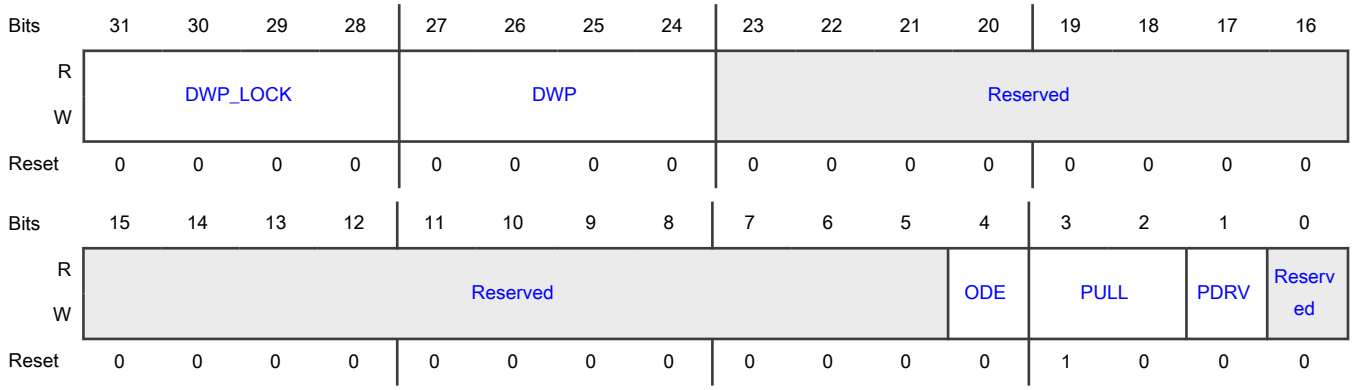
17.4.1.262 SW_PAD_CTL_PAD_GPIO_SD_B2_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_09)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_09	420h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_09 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_09 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_09

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

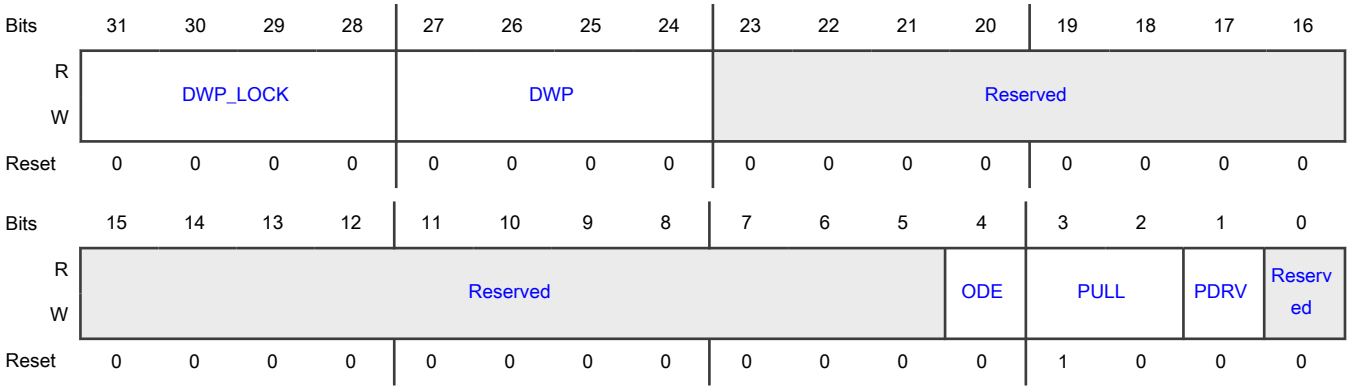
17.4.1.263 SW_PAD_CTL_PAD_GPIO_SD_B2_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_10)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_10	424h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_10 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_10 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_10 0b - high driver 1b - normal driver
0 —	- Reserved

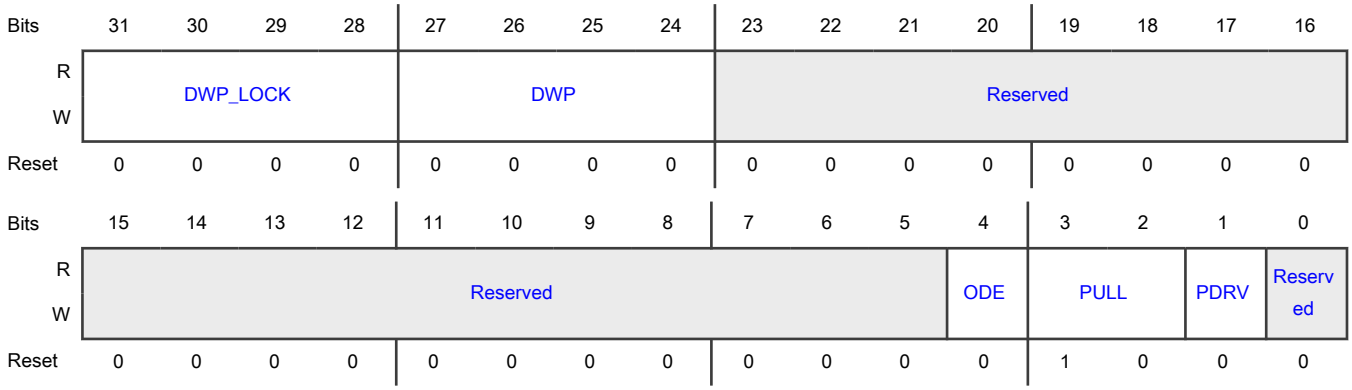
17.4.1.264 SW_PAD_CTL_PAD_GPIO_SD_B2_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_11)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_11	428h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_11 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_11 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_11

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

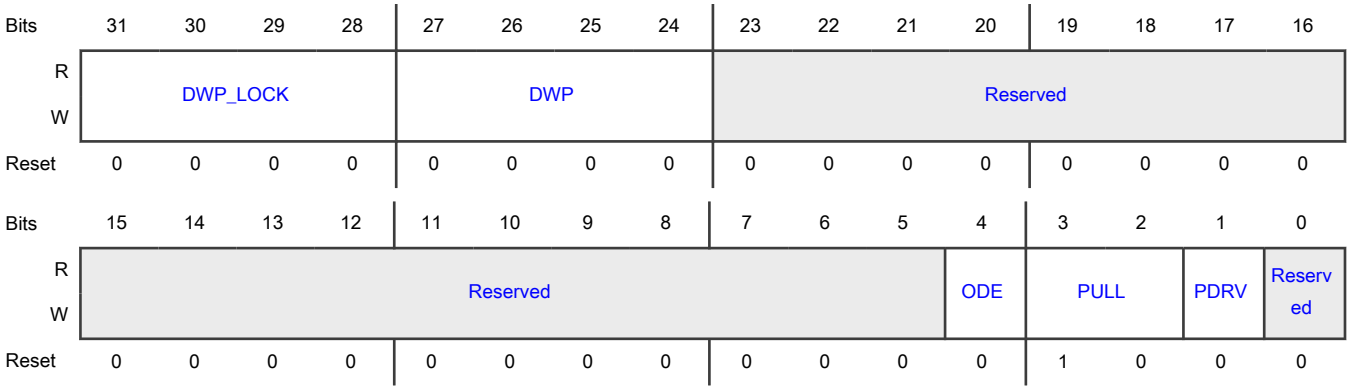
17.4.1.265 SW_PAD_CTL_PAD_GPIO_SD_B2_12_DUMMY SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_SD_B2_12_DUMMY)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_SD_B2_12_DUMMY	42Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_SD_B2_12_DUMMY 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_SD_B2_12_DUMMY 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_SD_B2_12_DUMMY 0b - high driver 1b - normal driver
0 —	- Reserved

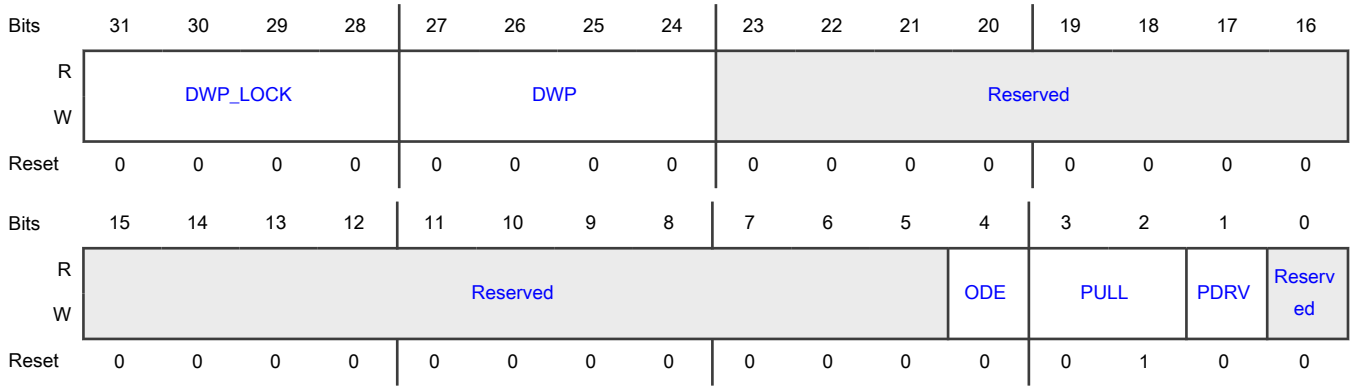
17.4.1.266 SW_PAD_CTL_PAD_GPIO_B1_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_00)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_00	430h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_00 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_00 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_00

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.267 SW_PAD_CTL_PAD_GPIO_B1_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_01)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_01	434h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_01 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_01 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_01 0b - high driver 1b - normal driver
0 —	- Reserved

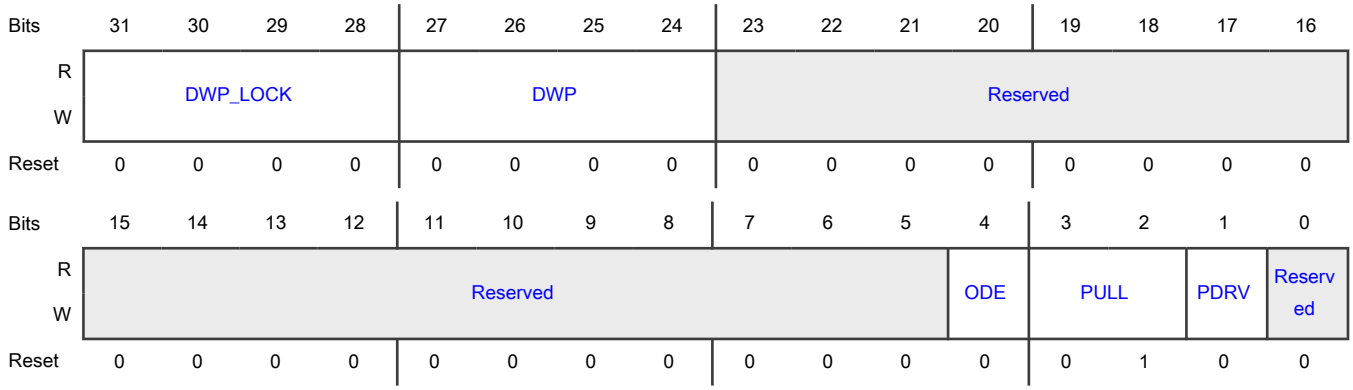
17.4.1.268 SW_PAD_CTL_PAD_GPIO_B1_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_02)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_02	438h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_02 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_02 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_02

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.269 SW_PAD_CTL_PAD_GPIO_B1_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_03)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_03	43Ch

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_03 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_03 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_03 0b - high driver 1b - normal driver
0 —	- Reserved

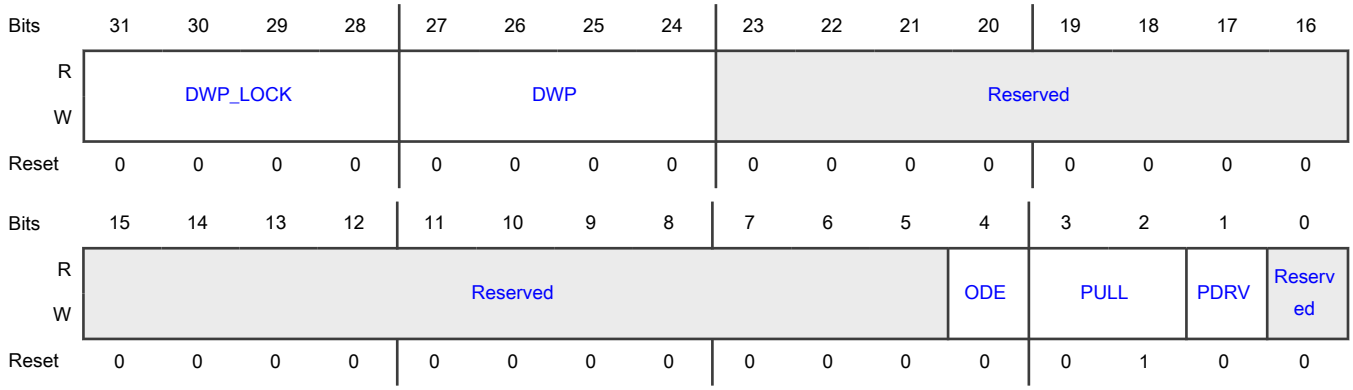
17.4.1.270 SW_PAD_CTL_PAD_GPIO_B1_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_04)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_04	440h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_04 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_04 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_04

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.271 SW_PAD_CTL_PAD_GPIO_B1_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_05)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_05	444h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_05 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_05 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_05 0b - high driver 1b - normal driver
0 —	- Reserved

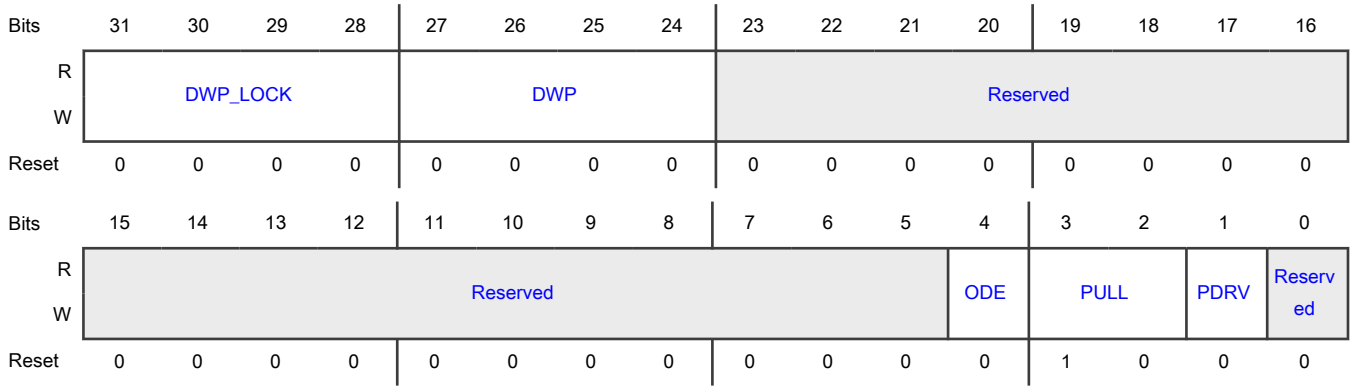
17.4.1.272 SW_PAD_CTL_PAD_GPIO_B1_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_06)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_06	448h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_06 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_06 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_06

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

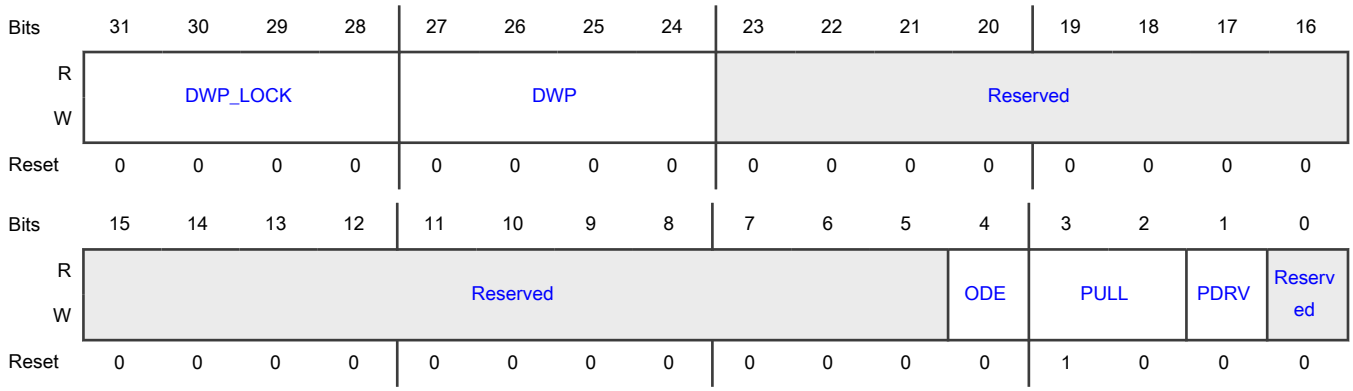
17.4.1.273 SW_PAD_CTL_PAD_GPIO_B1_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_07)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_07	44Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_07 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_07 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_07 0b - high driver 1b - normal driver
0 —	- Reserved

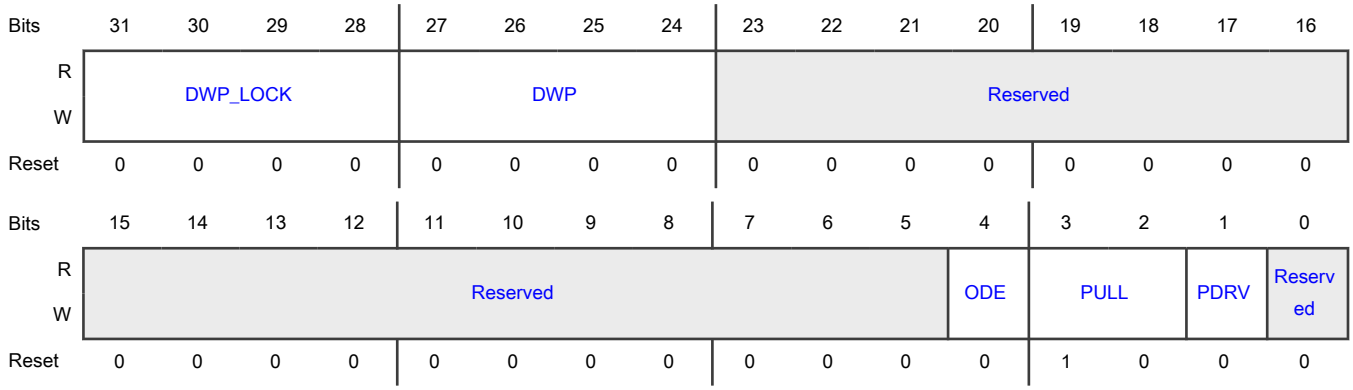
17.4.1.274 SW_PAD_CTL_PAD_GPIO_B1_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_08)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_08	450h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_08 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_08 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_08

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.275 SW_PAD_CTL_PAD_GPIO_B1_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_09)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_09	454h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_09 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_09 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_09 0b - high driver 1b - normal driver
0 —	- Reserved

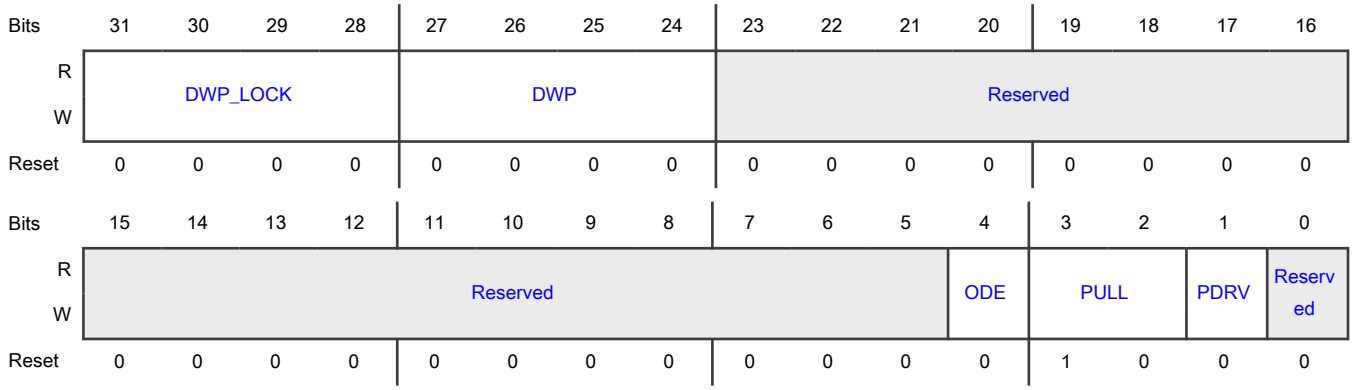
17.4.1.276 SW_PAD_CTL_PAD_GPIO_B1_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_10)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_10	458h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_10 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_10 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_10

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

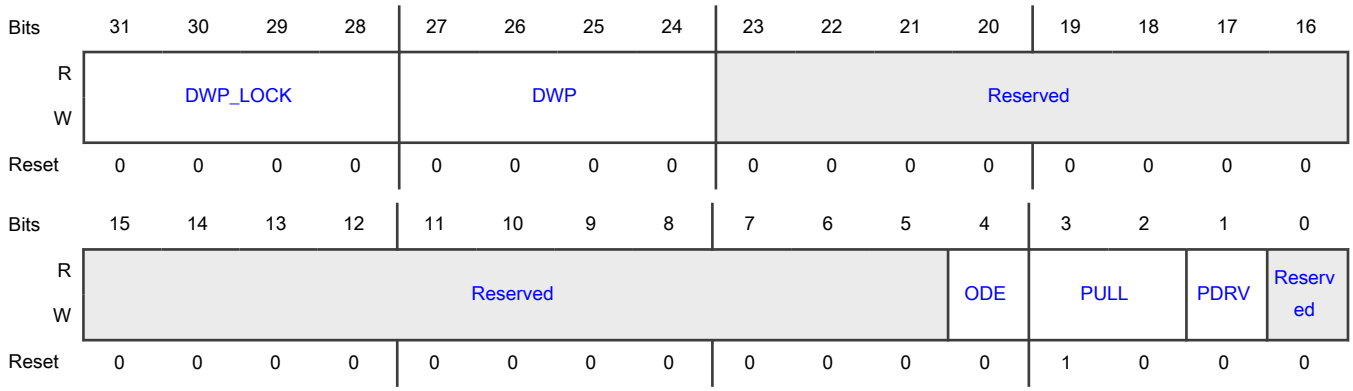
17.4.1.277 SW_PAD_CTL_PAD_GPIO_B1_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_11)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_11	45Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_11 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_11 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_11 0b - high driver 1b - normal driver
0 —	- Reserved

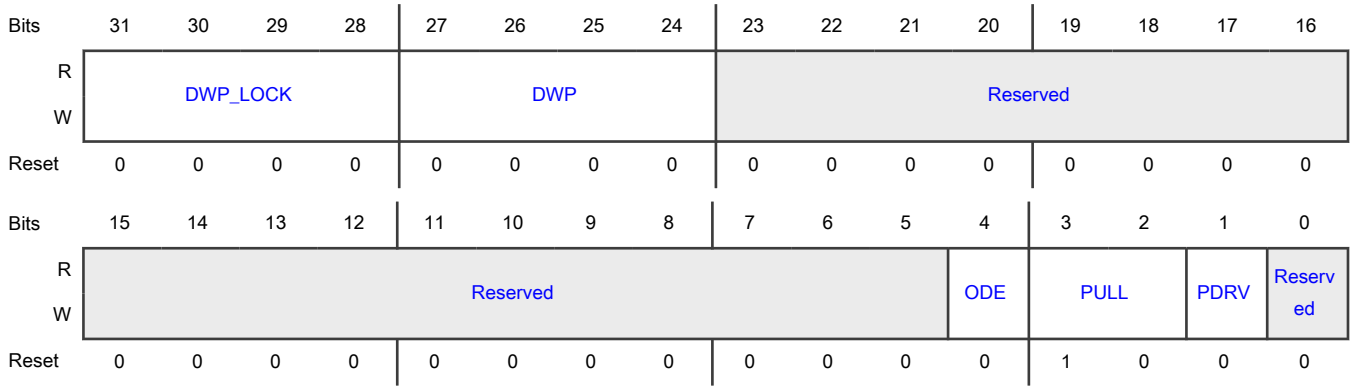
17.4.1.278 SW_PAD_CTL_PAD_GPIO_B1_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_12)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_12	460h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_12 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_12 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_12

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.279 SW_PAD_CTL_PAD_GPIO_B1_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B1_13)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B1_13	464h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B1_13 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B1_13 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B1_13 0b - high driver 1b - normal driver
0 —	- Reserved

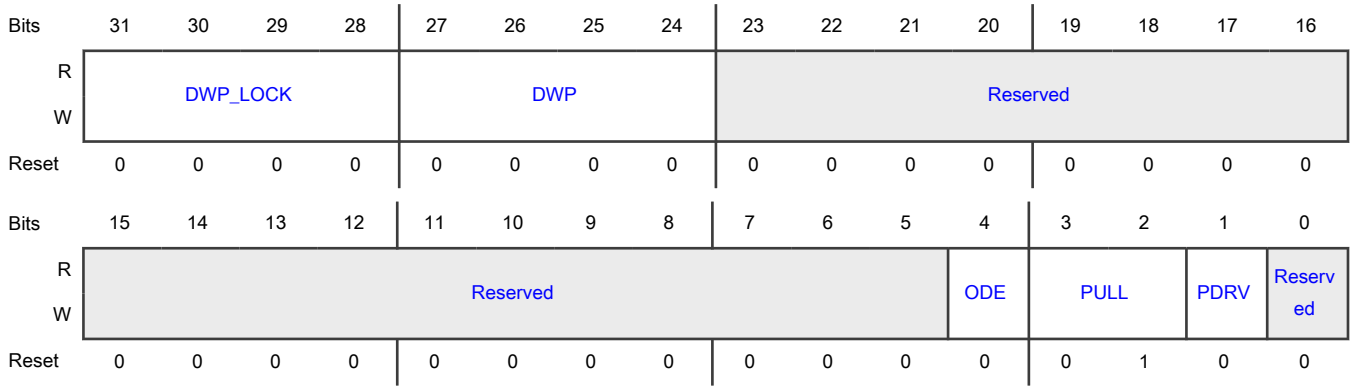
17.4.1.280 SW_PAD_CTL_PAD_GPIO_B2_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_00)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_00	468h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_00 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_00 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_00

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.281 SW_PAD_CTL_PAD_GPIO_B2_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_01)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_01	46Ch

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_01 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_01 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_01 0b - high driver 1b - normal driver
0 —	- Reserved

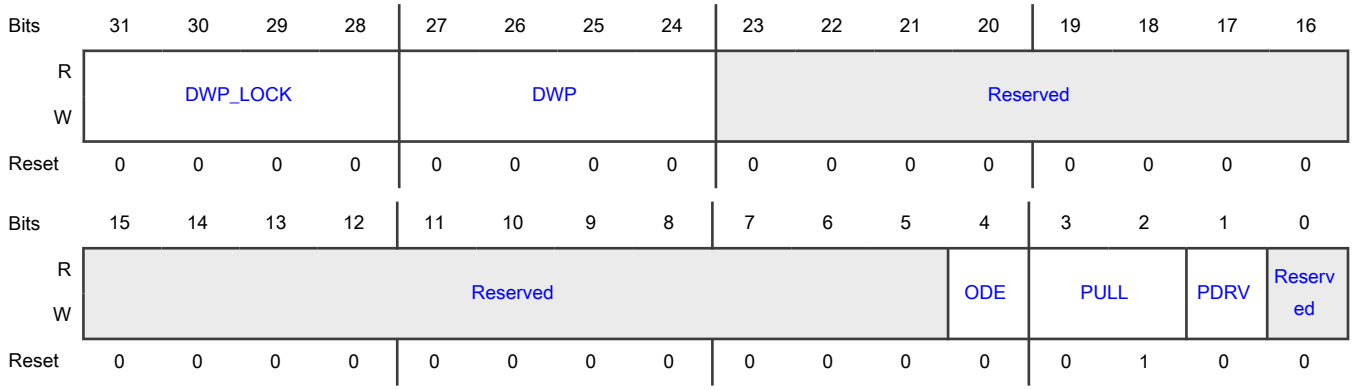
17.4.1.282 SW_PAD_CTL_PAD_GPIO_B2_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_02)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_02	470h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_02 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_02 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_02

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.283 SW_PAD_CTL_PAD_GPIO_B2_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_03)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_03	474h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_03 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_03 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_03 0b - high driver 1b - normal driver
0 —	- Reserved

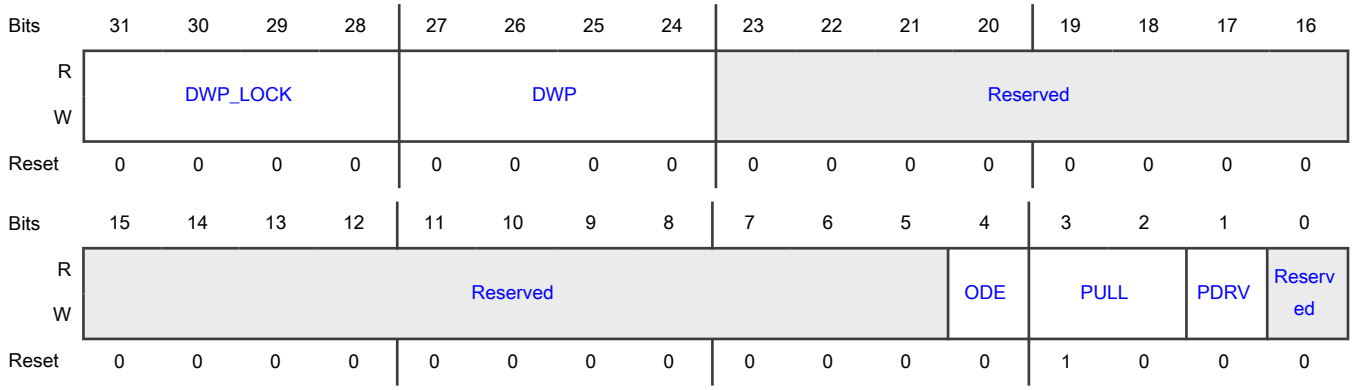
17.4.1.284 SW_PAD_CTL_PAD_GPIO_B2_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_04)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_04	478h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_04 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_04 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_04

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.285 SW_PAD_CTL_PAD_GPIO_B2_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_05)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_05	47Ch

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_05 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_05 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_05 0b - high driver 1b - normal driver
0 —	- Reserved

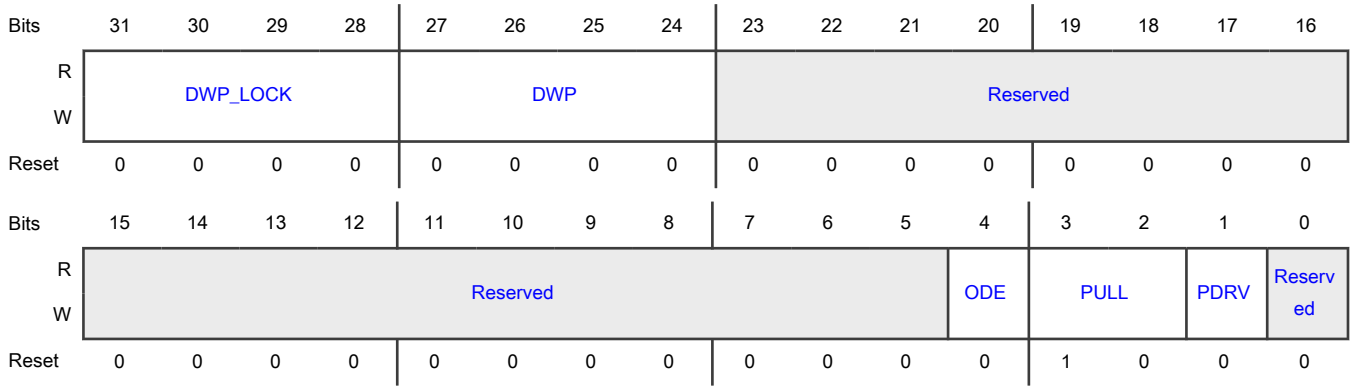
17.4.1.286 SW_PAD_CTL_PAD_GPIO_B2_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_06)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_06	480h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_06 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_06 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_06

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.287 SW_PAD_CTL_PAD_GPIO_B2_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_07)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_07	484h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_07 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_07 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_07 0b - high driver 1b - normal driver
0 —	- Reserved

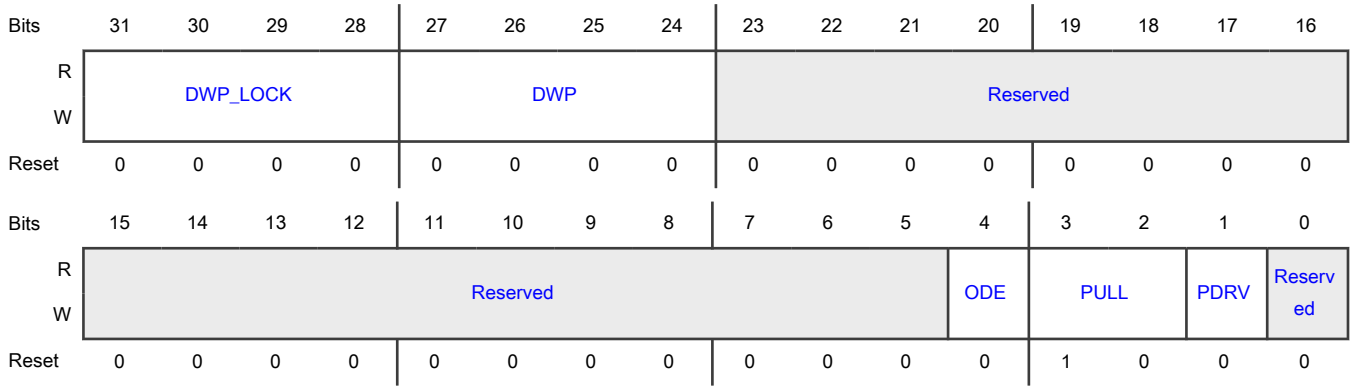
17.4.1.288 SW_PAD_CTL_PAD_GPIO_B2_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_08)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_08	488h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_08 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_08 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_08

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

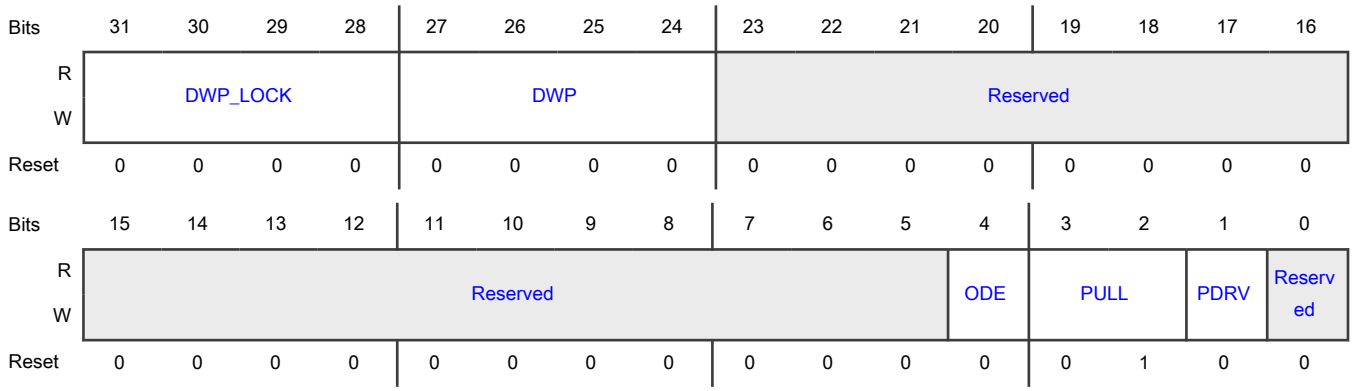
17.4.1.289 SW_PAD_CTL_PAD_GPIO_B2_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_09)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_09	48Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_09 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_09 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_09 0b - high driver 1b - normal driver
0 —	- Reserved

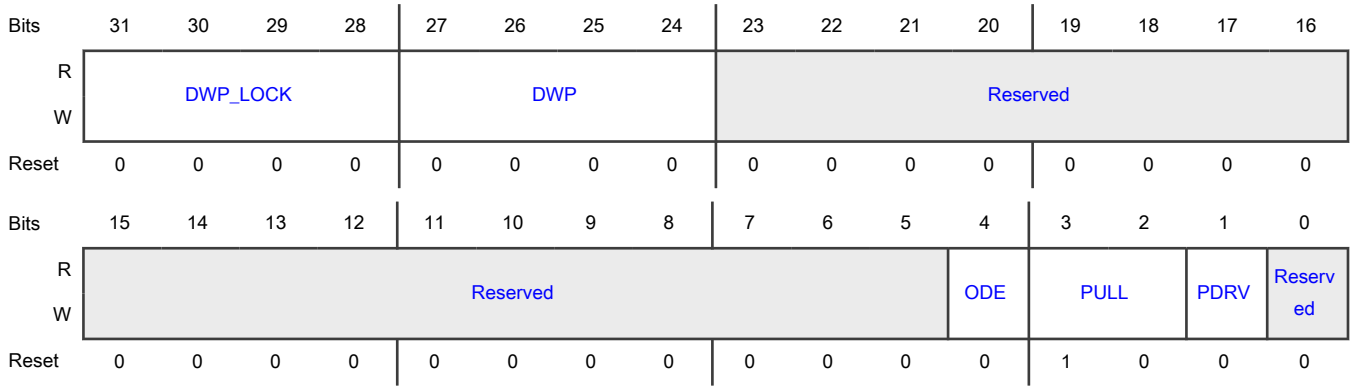
17.4.1.290 SW_PAD_CTL_PAD_GPIO_B2_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_10)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_10	490h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_10 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_10 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_10

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

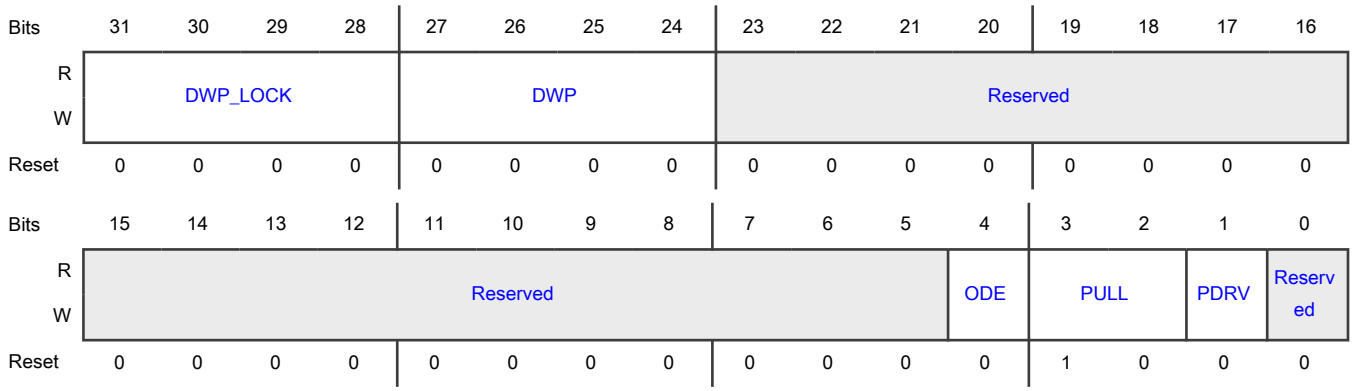
17.4.1.291 SW_PAD_CTL_PAD_GPIO_B2_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_11)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_11	494h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_11 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_11 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_11 0b - high driver 1b - normal driver
0 —	- Reserved

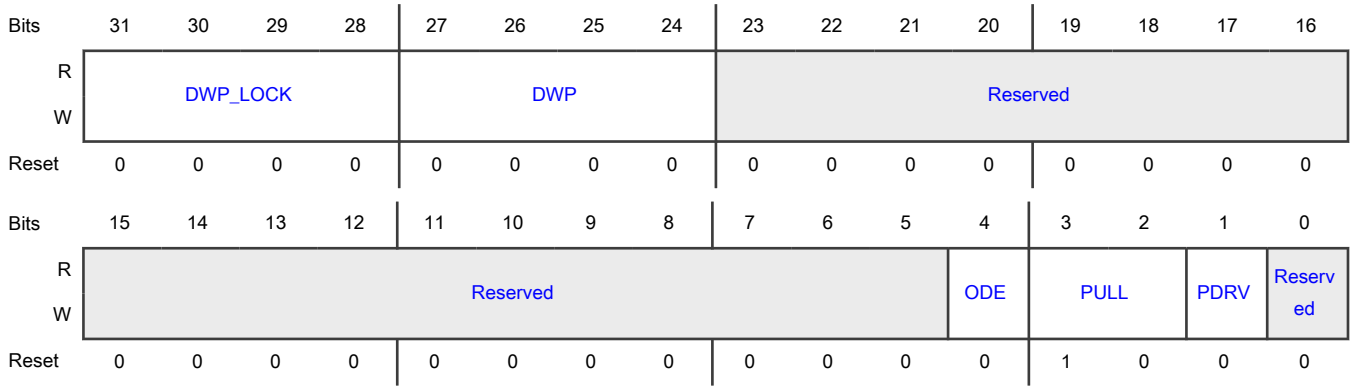
17.4.1.292 SW_PAD_CTL_PAD_GPIO_B2_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_12)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_12	498h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_12 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_12 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_12

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - high driver 1b - normal driver
0 —	- Reserved

17.4.1.293 SW_PAD_CTL_PAD_GPIO_B2_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_B2_13)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_B2_13	49Ch

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PULL		PDRV	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_B2_13 0b - Disabled 1b - Enabled
3-2 PULL	Pull Down Pull Up Field Select one out of next values for pad: GPIO_B2_13 00b - Forbidden 01b - PU 10b - PD 11b - No Pull
1 PDRV	PDRV Field Select one out of next values for pad: GPIO_B2_13 0b - high driver 1b - normal driver
0 —	- Reserved

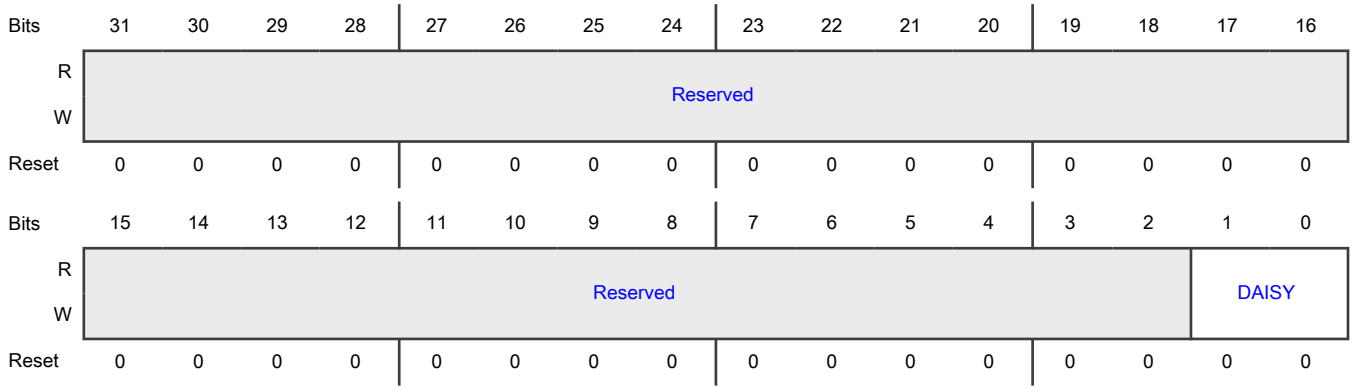
17.4.1.294 CAN1_IPP_IND_CANRX_SELECT_INPUT DAISY Register (CAN1_IPP_IND_CANRX_SELECT_INPUT)

Offset

Register	Offset
CAN1_IPP_IND_CANRX_SELECT_INPUT	4A0h

Function
DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: can1, In Pin: ipp_ind_canrx 00b - Selecting Pad: GPIO_AD_16 for Mode: ALT9 01b - Selecting Pad: GPIO_AON_01 for Mode: ALT1 10b - Selecting Pad: GPIO_AON_07 for Mode: ALT6 11b - Selecting Pad: GPIO_AON_17 for Mode: ALT6

17.4.1.295 CAN2_IPP_IND_CANRX_SELECT_INPUT DAISY Register (CAN2_IPP_IND_CANRX_SELECT_INPUT)

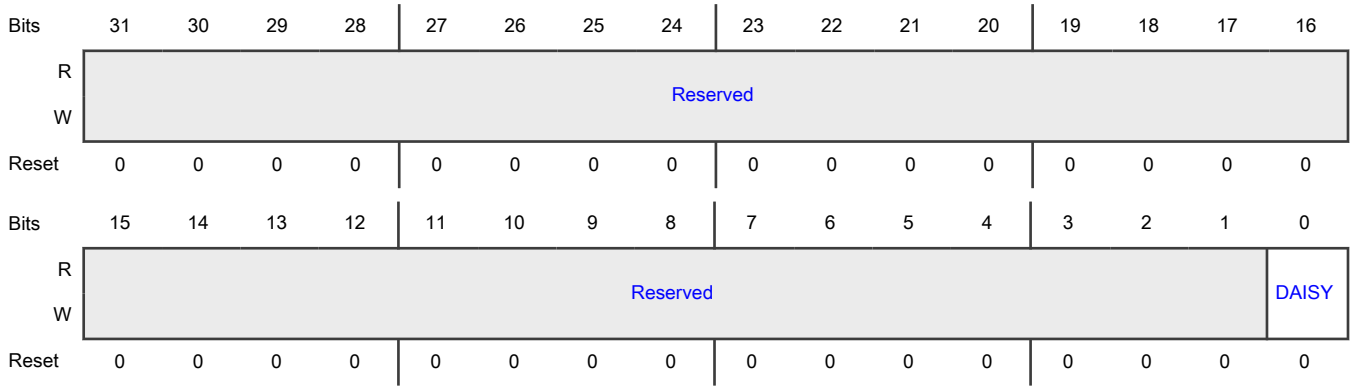
Offset

Register	Offset
CAN2_IPP_IND_CANRX_SELECT_INPUT	4A4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: can2, In Pin: ipp_ind_canrx 0b - Selecting Pad: GPIO_AD_01 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_31 for Mode: ALT2

17.4.1.296 CAN3_IPP_IND_CANRX_SELECT_INPUT DAISY Register (CAN3_IPP_IND_CANRX_SELECT_INPUT)

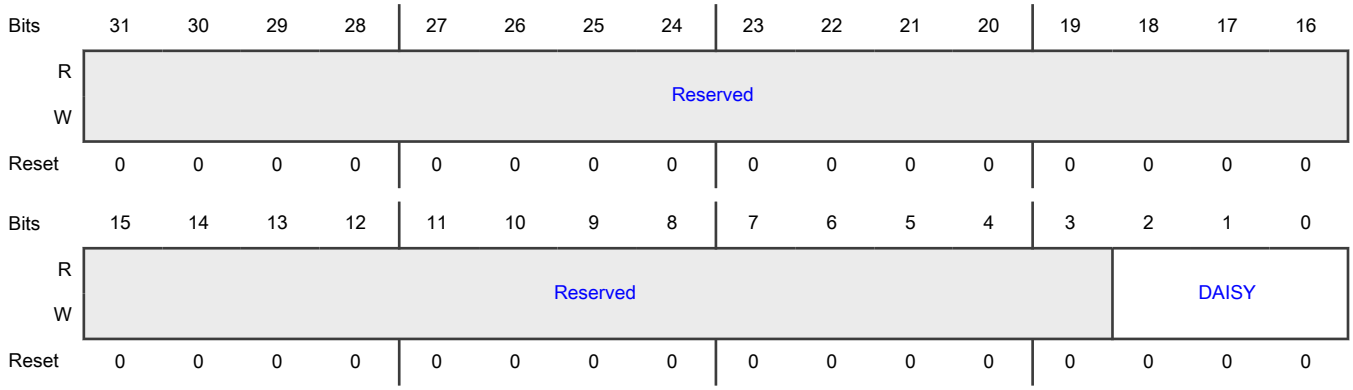
Offset

Register	Offset
CAN3_IPP_IND_CANRX_SELECT_INPUT	4A8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-3 —	- Reserved
2-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: can3, In Pin: ipp_ind_canrx 000b - Selecting Pad: GPIO_AD_07 for Mode: ALT1 001b - Selecting Pad: GPIO_B2_11 for Mode: ALT2 010b - Selecting Pad: GPIO_B2_13 for Mode: ALT6 011b - Selecting Pad: GPIO_AON_03 for Mode: ALT1 100b - Selecting Pad: GPIO_AON_19 for Mode: ALT6

17.4.1.297 ECAT_ECAT_RX_CLK_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_CLK_0_SELECT_INPUT)

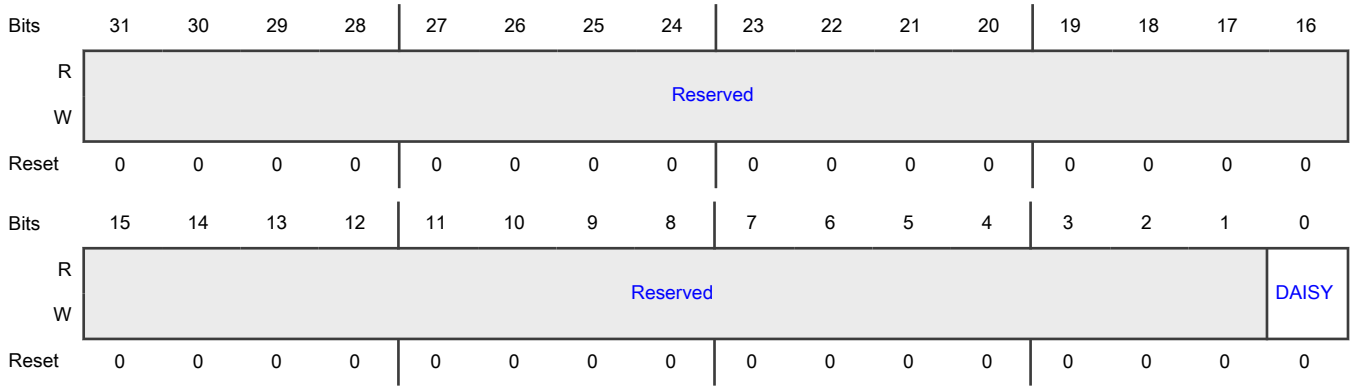
Offset

Register	Offset
ECAT_ECAT_RX_CLK_0_SELECT_INPUT	4ACh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_CLK_0 0b - Selecting Pad: GPIO_EMC_B1_02 for Mode: ALT10 1b - Selecting Pad: GPIO_EMC_B2_00 for Mode: ALT12

17.4.1.298 ECAT_ECAT_RX_CLK_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_CLK_1_SELECT_INPUT)

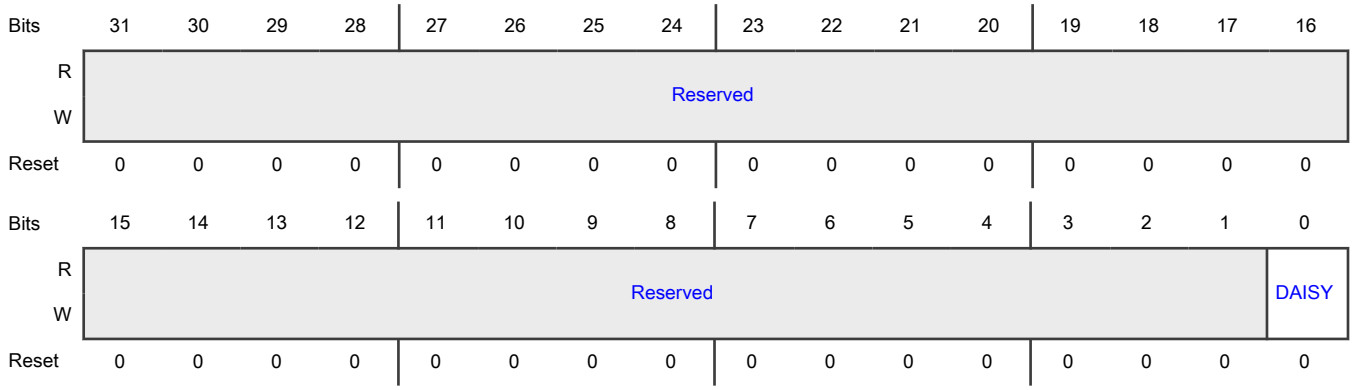
Offset

Register	Offset
ECAT_ECAT_RX_CLK_1_SELECT_INPUT	4B0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_CLK_1 0b - Selecting Pad: GPIO_EMC_B1_38 for Mode: ALT6 1b - Selecting Pad: GPIO_B2_13 for Mode: ALT12

17.4.1.299 ECAT_ECAT_RX_DATA0_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA0_0_SELECT_INPUT)

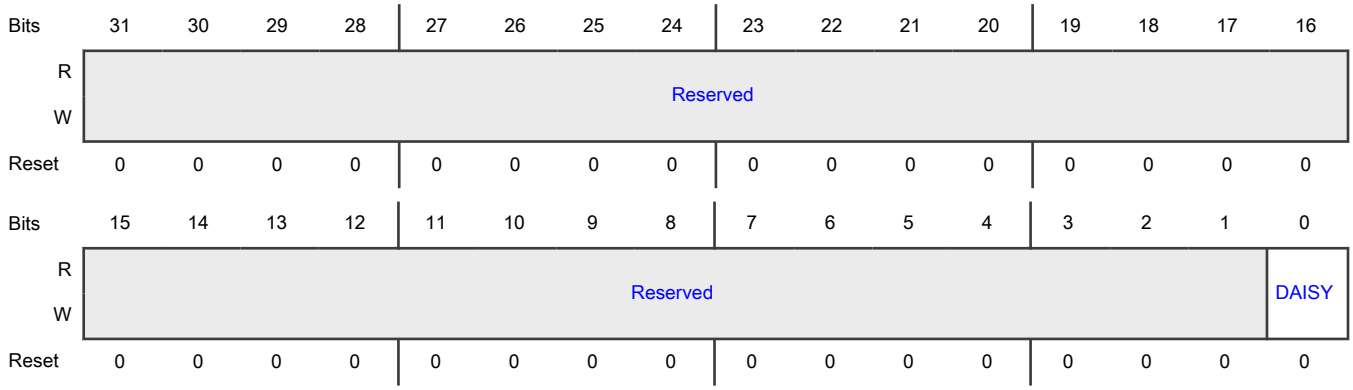
Offset

Register	Offset
ECAT_ECAT_RX_DATA0_0_SELECT_INPUT	4B4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_DATA0_0 0b - Selecting Pad: GPIO_EMC_B1_09 for Mode: ALT10 1b - Selecting Pad: GPIO_EMC_B2_09 for Mode: ALT12

17.4.1.300 ECAT_ECAT_RX_DATA0_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA0_1_SELECT_INPUT)

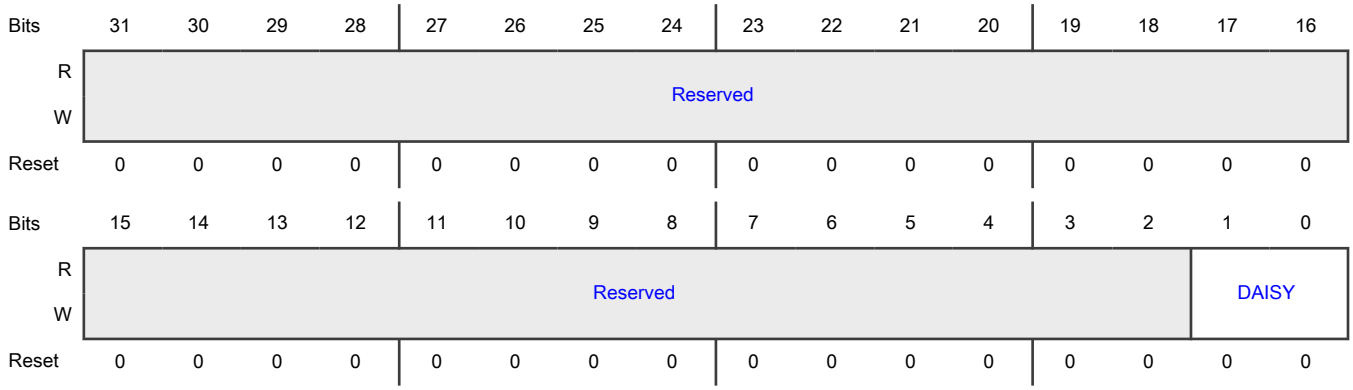
Offset

Register	Offset
ECAT_ECAT_RX_DATA0_1_SELECT_INPUT	4B8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_DATA0_1 00b - Selecting Pad: GPIO_EMC_B1_30 for Mode: ALT6 01b - Selecting Pad: GPIO_EMC_B2_17 for Mode: ALT12 10b - Selecting Pad: GPIO_B2_10 for Mode: ALT12

17.4.1.301 ECAT_ECAT_RX_DATA1_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA1_0_SELECT_INPUT)

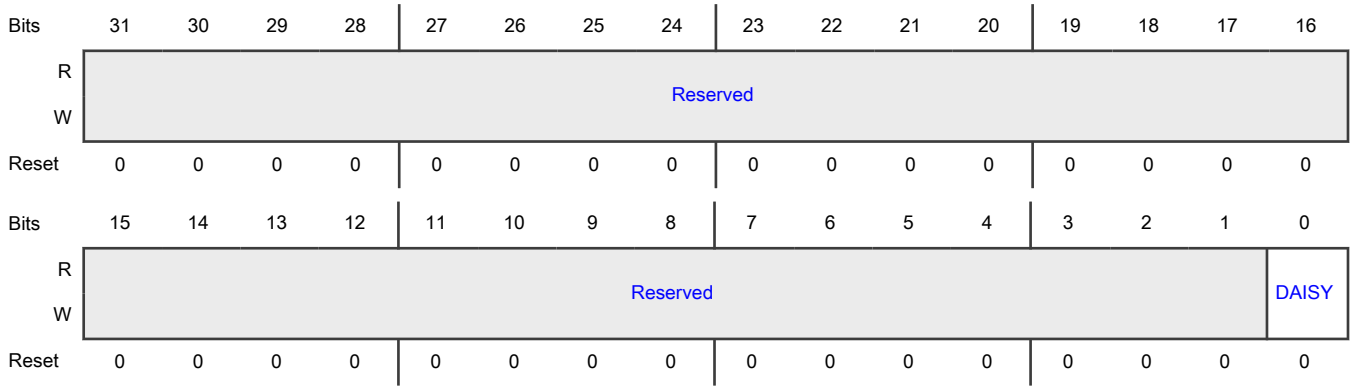
Offset

Register	Offset
ECAT_ECAT_RX_DATA1_0_SELECT_INPUT	4BCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_DATA1_0 0b - Selecting Pad: GPIO_EMC_B1_10 for Mode: ALT10 1b - Selecting Pad: GPIO_EMC_B2_10 for Mode: ALT12

17.4.1.302 ECAT_ECAT_RX_DATA1_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA1_1_SELECT_INPUT)

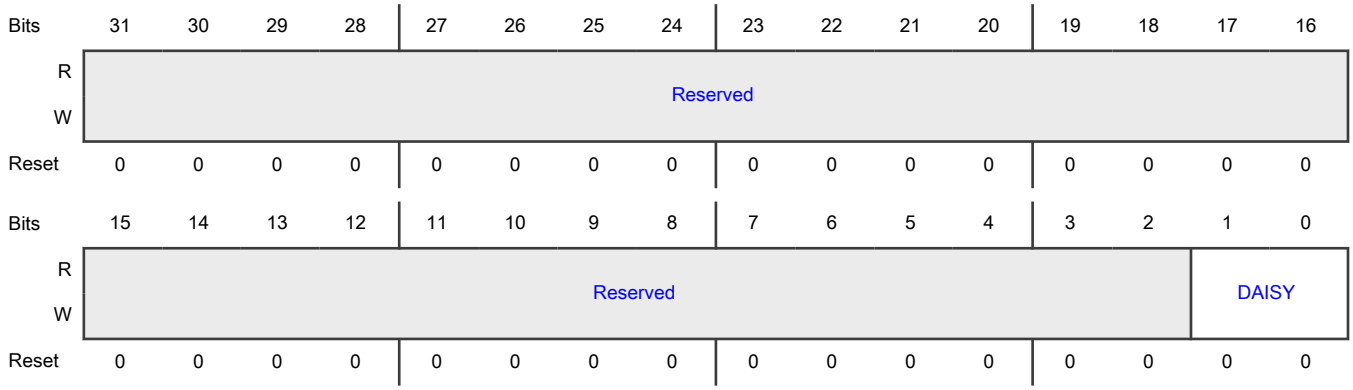
Offset

Register	Offset
ECAT_ECAT_RX_DATA1_1_SELECT_INPUT	4C0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_DATA1_1 00b - Selecting Pad: GPIO_EMC_B1_31 for Mode: ALT6 01b - Selecting Pad: GPIO_EMC_B2_18 for Mode: ALT12 10b - Selecting Pad: GPIO_B2_11 for Mode: ALT12

17.4.1.303 ECAT_ECAT_RX_DATA2_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA2_0_SELECT_INPUT)

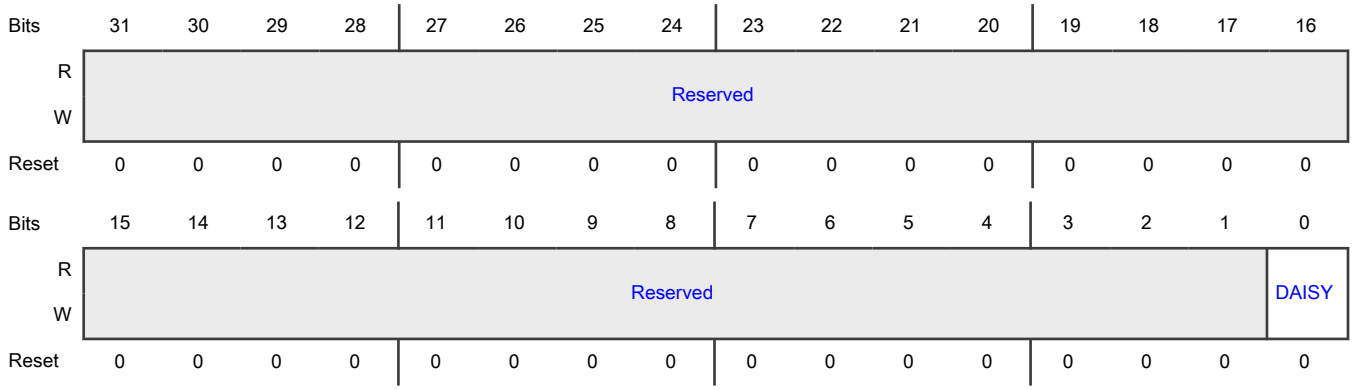
Offset

Register	Offset
ECAT_ECAT_RX_DATA2_0_SELECT_INPUT	4C4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_DATA2_0 0b - Selecting Pad: GPIO_EMC_B1_04 for Mode: ALT10 1b - Selecting Pad: GPIO_EMC_B2_01 for Mode: ALT12

17.4.1.304 ECAT_ECAT_RX_DATA2_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA2_1_SELECT_INPUT)

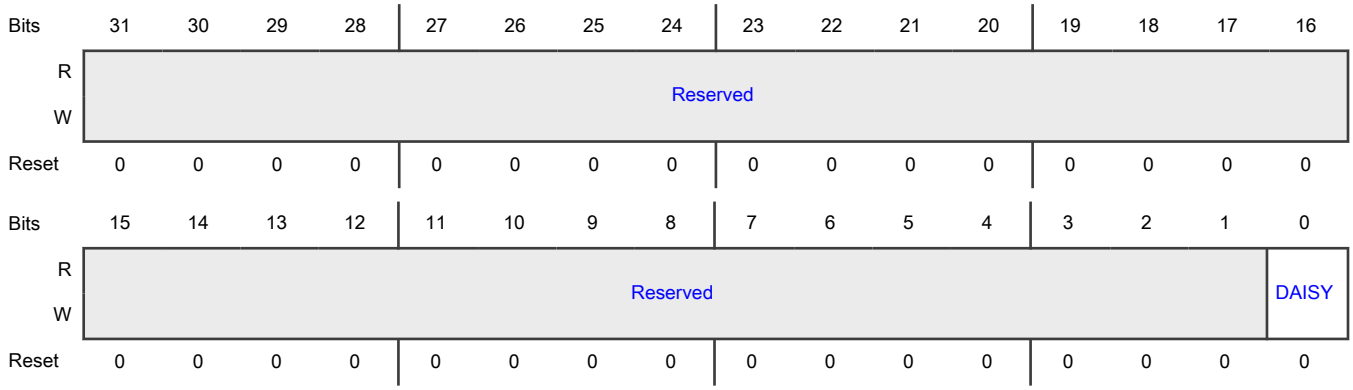
Offset

Register	Offset
ECAT_ECAT_RX_DATA2_1_SELECT_INPUT	4C8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_DATA2_1 0b - Selecting Pad: GPIO_EMC_B1_34 for Mode: ALT6 1b - Selecting Pad: GPIO_B2_02 for Mode: ALT12

17.4.1.305 ECAT_ECAT_RX_DATA3_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA3_0_SELECT_INPUT)

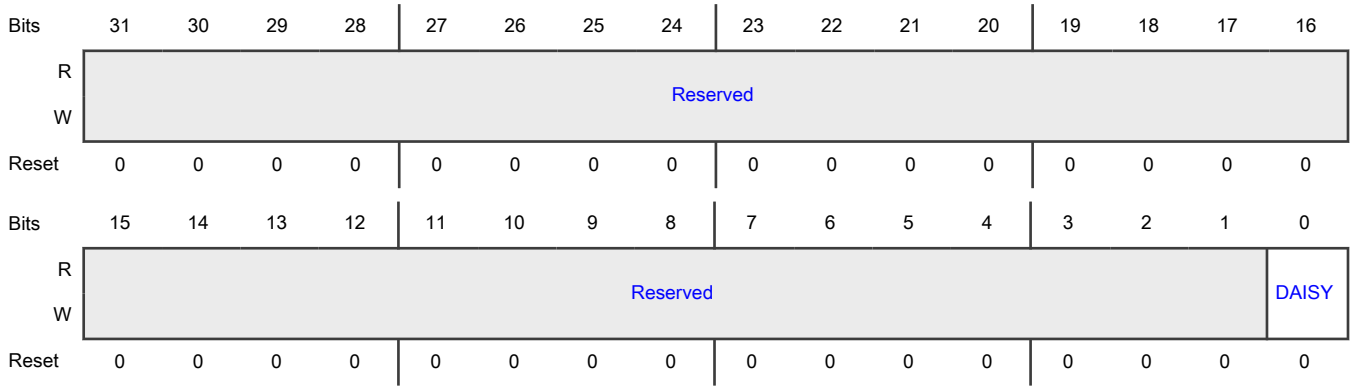
Offset

Register	Offset
ECAT_ECAT_RX_DATA3_0_SELECT_INPUT	4CCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_DATA3_0 0b - Selecting Pad: GPIO_EMC_B1_03 for Mode: ALT10 1b - Selecting Pad: GPIO_EMC_B2_02 for Mode: ALT12

17.4.1.306 ECAT_ECAT_RX_DATA3_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DATA3_1_SELECT_INPUT)

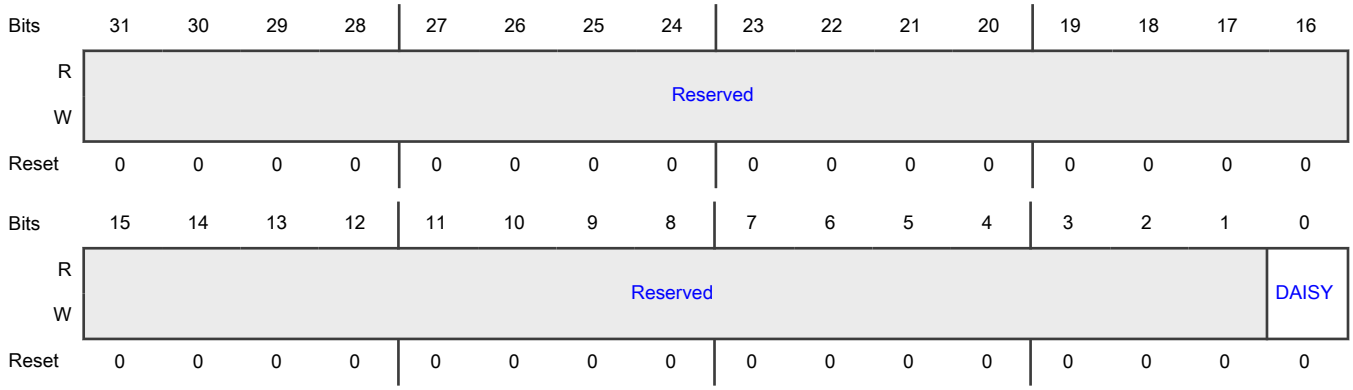
Offset

Register	Offset
ECAT_ECAT_RX_DATA3_1_SELECT_INPUT	4D0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_DATA3_1 0b - Selecting Pad: GPIO_EMC_B1_35 for Mode: ALT6 1b - Selecting Pad: GPIO_B2_03 for Mode: ALT12

17.4.1.307 ECAT_ECAT_RX_DV_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DV_0_SELECT_INPUT)

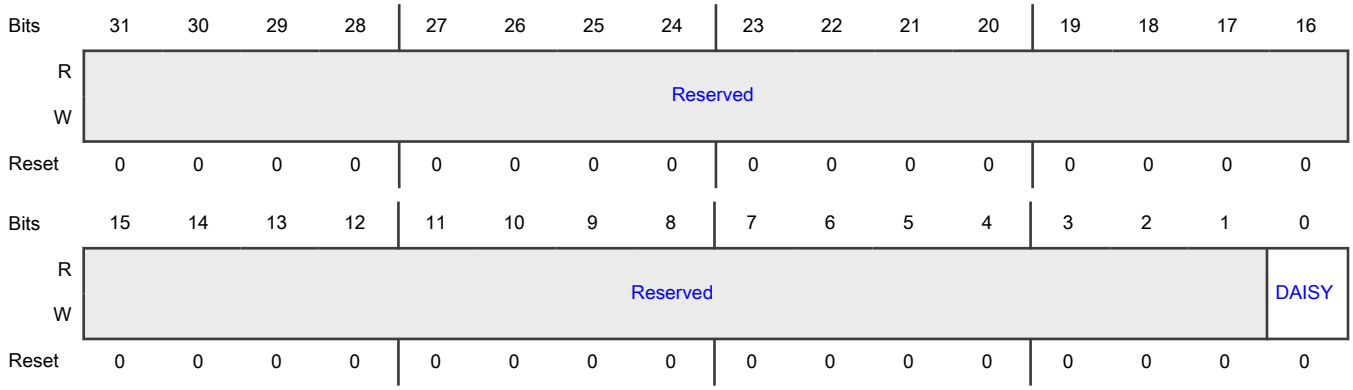
Offset

Register	Offset
ECAT_ECAT_RX_DV_0_SELECT_INPUT	4D4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_DV_0 0b - Selecting Pad: GPIO_EMC_B1_11 for Mode: ALT10 1b - Selecting Pad: GPIO_EMC_B2_11 for Mode: ALT12

17.4.1.308 ECAT_ECAT_RX_DV_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_DV_1_SELECT_INPUT)

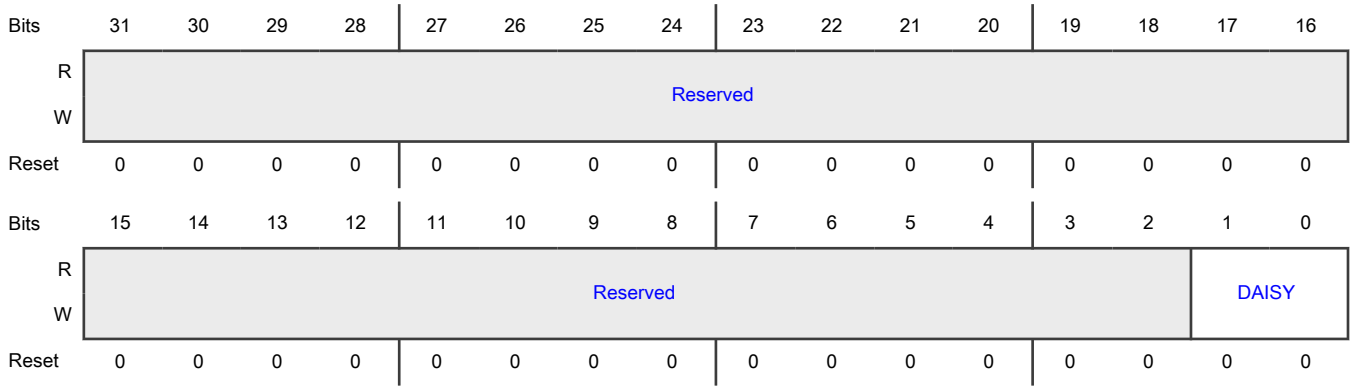
Offset

Register	Offset
ECAT_ECAT_RX_DV_1_SELECT_INPUT	4D8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_DV_1 00b - Selecting Pad: GPIO_EMC_B1_32 for Mode: ALT6 01b - Selecting Pad: GPIO_EMC_B2_19 for Mode: ALT12 10b - Selecting Pad: GPIO_B2_12 for Mode: ALT12

17.4.1.309 ECAT_ECAT_RX_ER_0_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_ER_0_SELECT_INPUT)

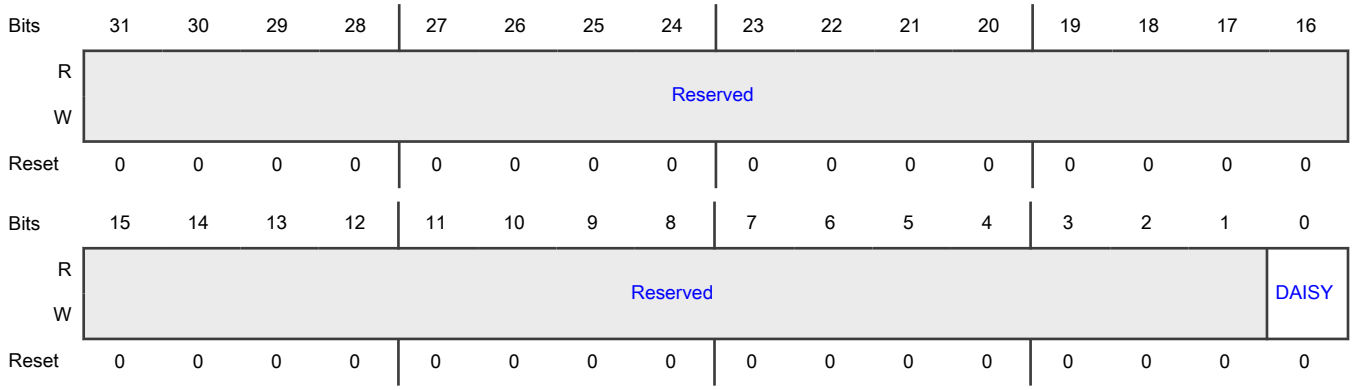
Offset

Register	Offset
ECAT_ECAT_RX_ER_0_SELECT_INPUT	4DCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_ER_0 0b - Selecting Pad: GPIO_EMC_B1_12 for Mode: ALT10 1b - Selecting Pad: GPIO_EMC_B2_12 for Mode: ALT12

17.4.1.310 ECAT_ECAT_RX_ER_1_SELECT_INPUT DAISY Register (ECAT_ECAT_RX_ER_1_SELECT_INPUT)

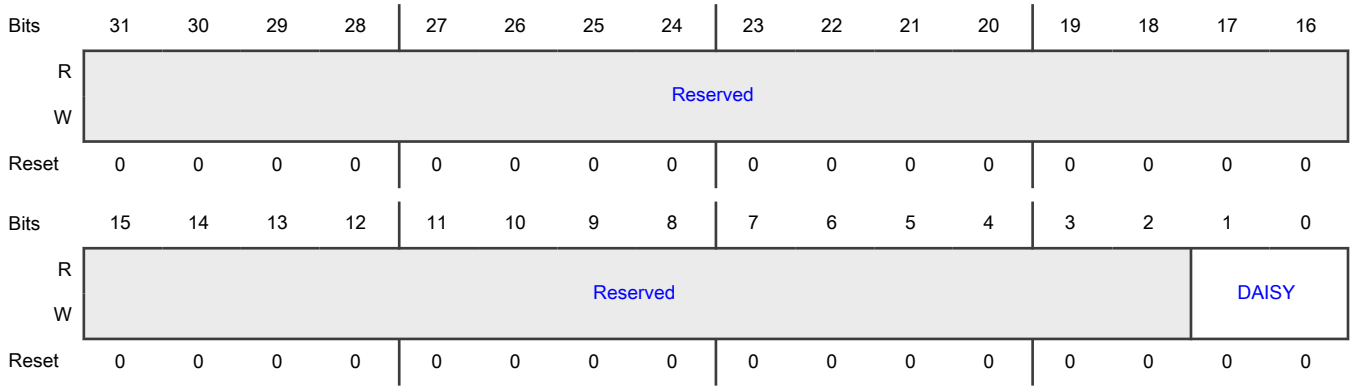
Offset

Register	Offset
ECAT_ECAT_RX_ER_1_SELECT_INPUT	4E0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_RX_ER_1 00b - Selecting Pad: GPIO_EMC_B1_33 for Mode: ALT6 01b - Selecting Pad: GPIO_EMC_B2_20 for Mode: ALT12 10b - Selecting Pad: GPIO_B2_01 for Mode: ALT12

17.4.1.311 ECAT_ECAT_TX_CLK_0_SELECT_INPUT DAISY Register (ECAT_ECAT_TX_CLK_0_SELECT_INPUT)

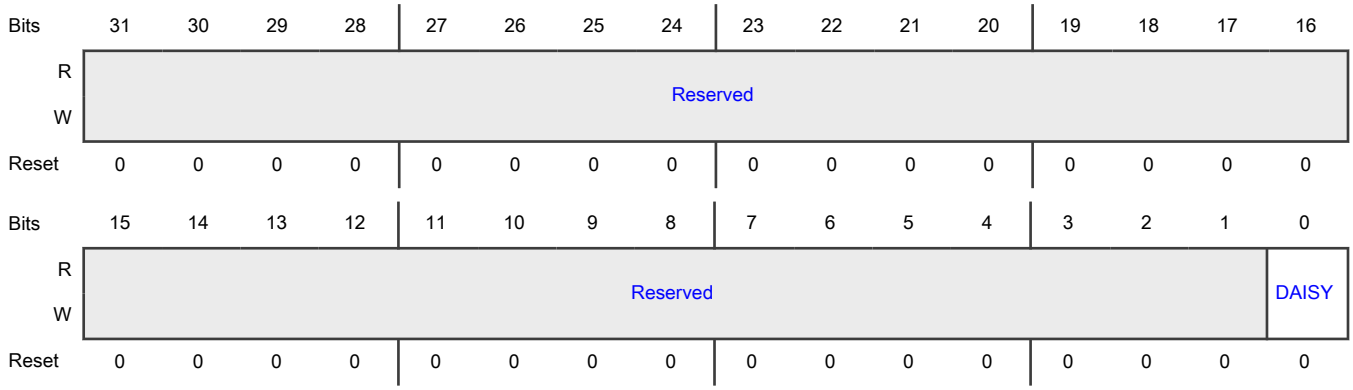
Offset

Register	Offset
ECAT_ECAT_TX_CLK_0_SELECT_INPUT	4E4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_TX_CLK_0 0b - Selecting Pad: GPIO_EMC_B1_08 for Mode: ALT10 1b - Selecting Pad: GPIO_EMC_B2_08 for Mode: ALT12

17.4.1.312 ECAT_ECAT_TX_CLK_1_SELECT_INPUT DAISY Register (ECAT_ECAT_TX_CLK_1_SELECT_INPUT)

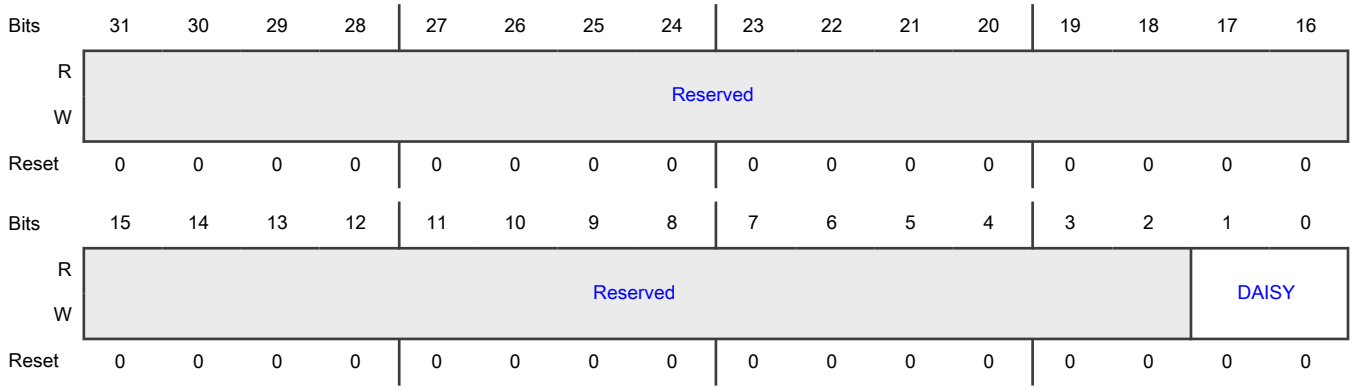
Offset

Register	Offset
ECAT_ECAT_TX_CLK_1_SELECT_INPUT	4E8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: ECAT_TX_CLK_1 00b - Selecting Pad: GPIO_EMC_B1_29 for Mode: ALT6 01b - Selecting Pad: GPIO_EMC_B2_16 for Mode: ALT12 10b - Selecting Pad: GPIO_B2_09 for Mode: ALT12

17.4.1.313 ECAT_MDIO_DATA_IN_SELECT_INPUT DAISY Register (ECAT_MDIO_DATA_IN_SELECT_INPUT)

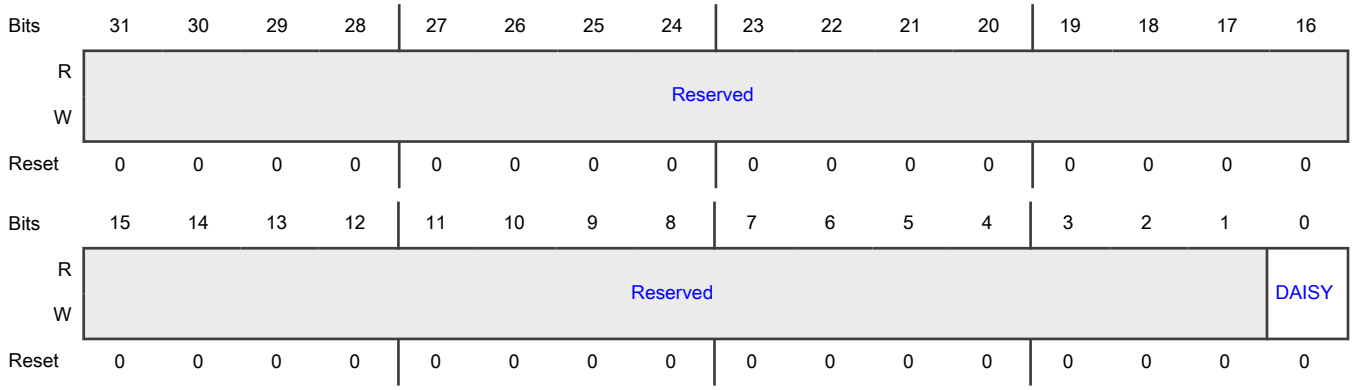
Offset

Register	Offset
ECAT_MDIO_DATA_IN_SELECT_INPUT	4ECh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: MDIO_DATA_IN 0b - Selecting Pad: GPIO_AD_31 for Mode: ALT12 1b - Selecting Pad: GPIO_SD_B2_10 for Mode: ALT12

17.4.1.314 ECAT_PROM_DATA_IN_SELECT_INPUT DAISY Register (ECAT_PROM_DATA_IN_SELECT_INPUT)

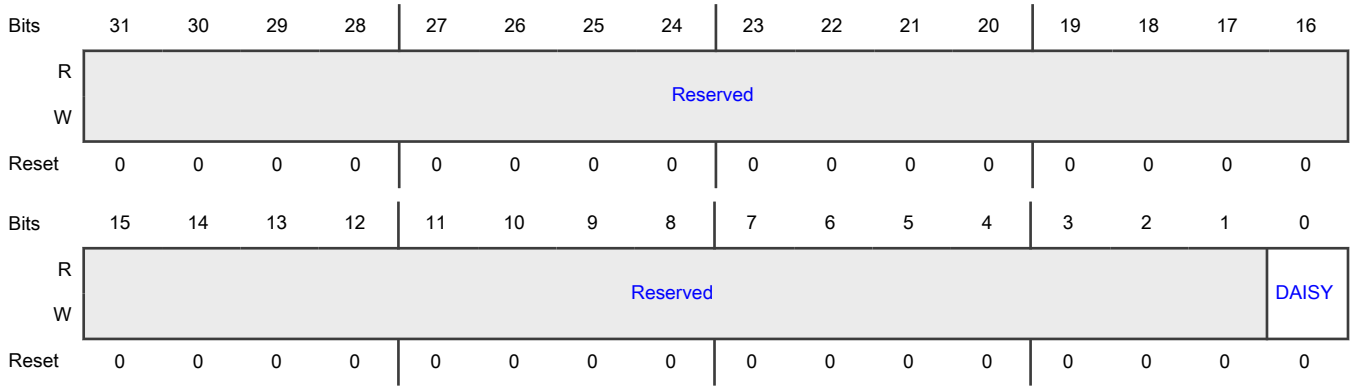
Offset

Register	Offset
ECAT_PROM_DATA_IN_SELECT_INPUT	4F0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecat, In Pin: PROM_DATA_IN 0b - Selecting Pad: GPIO_AD_19 for Mode: ALT12 1b - Selecting Pad: GPIO_AON_06 for Mode: ALT12

17.4.1.315 FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_0 DAISY Register (FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_0)

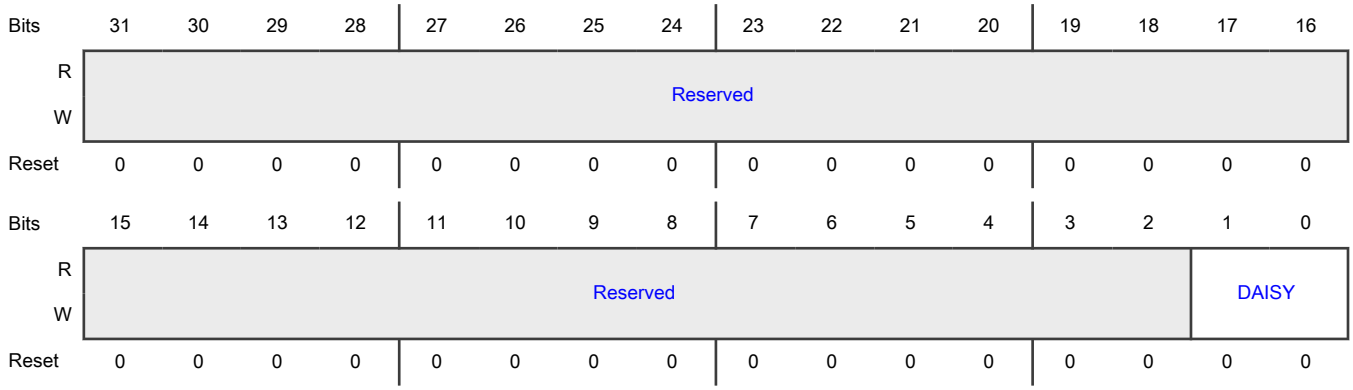
Offset

Register	Offset
FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_0	4F4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm1, In Pin: ipp_ind_pwma0 00b - Selecting Pad: GPIO_EMC_B1_24 for Mode: ALT1 01b - Selecting Pad: GPIO_EMC_B1_36 for Mode: ALT1 10b - Selecting Pad: GPIO_AD_00 for Mode: ALT4

17.4.1.316 FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_1 DAISY Register (FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_1)

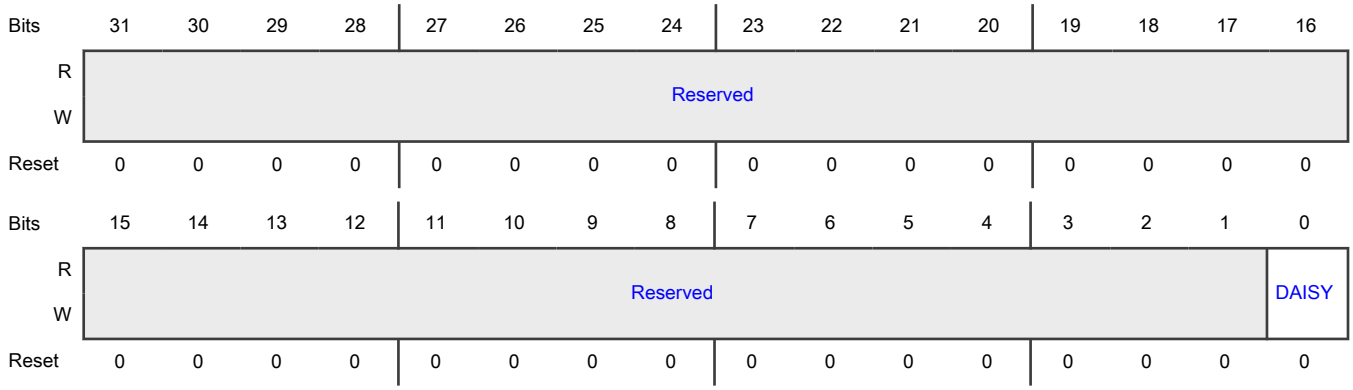
Offset

Register	Offset
FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_1	4F8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm1, In Pin: ipp_ind_pwma1 0b - Selecting Pad: GPIO_EMC_B1_26 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_02 for Mode: ALT4

17.4.1.317 FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_2 DAISY Register (FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_2)

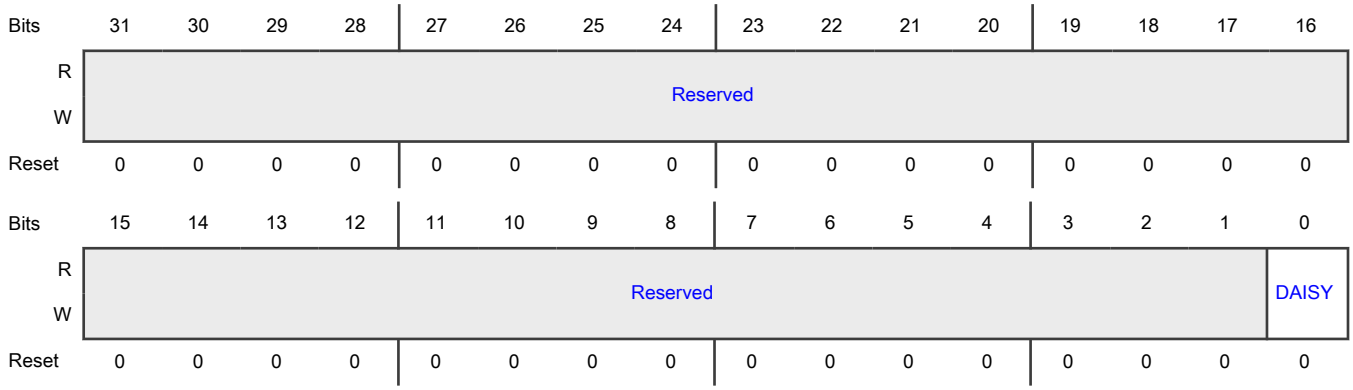
Offset

Register	Offset
FLEXPWM1_IPP_IND_PWMA_SELECT_INPUT_2	4FCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm1, In Pin: ipp_ind_pwma2 0b - Selecting Pad: GPIO_EMC_B1_29 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_05 for Mode: ALT4

17.4.1.318 FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_0 DAISY Register (FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_0)

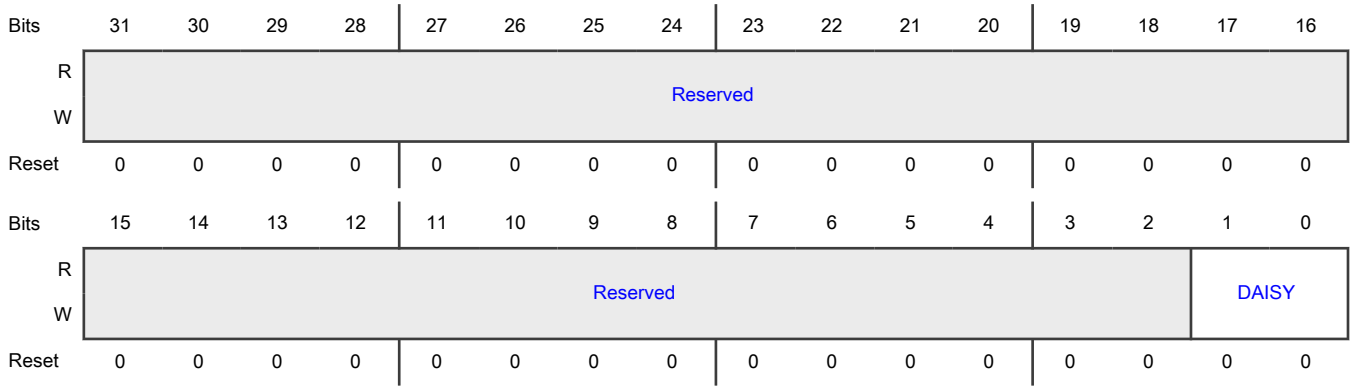
Offset

Register	Offset
FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_0	500h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm1, In Pin: ipp_ind_pwmb0 00b - Selecting Pad: GPIO_EMC_B1_25 for Mode: ALT1 01b - Selecting Pad: GPIO_EMC_B1_37 for Mode: ALT1 10b - Selecting Pad: GPIO_AD_01 for Mode: ALT4

17.4.1.319 FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_1 DAISY Register (FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_1)

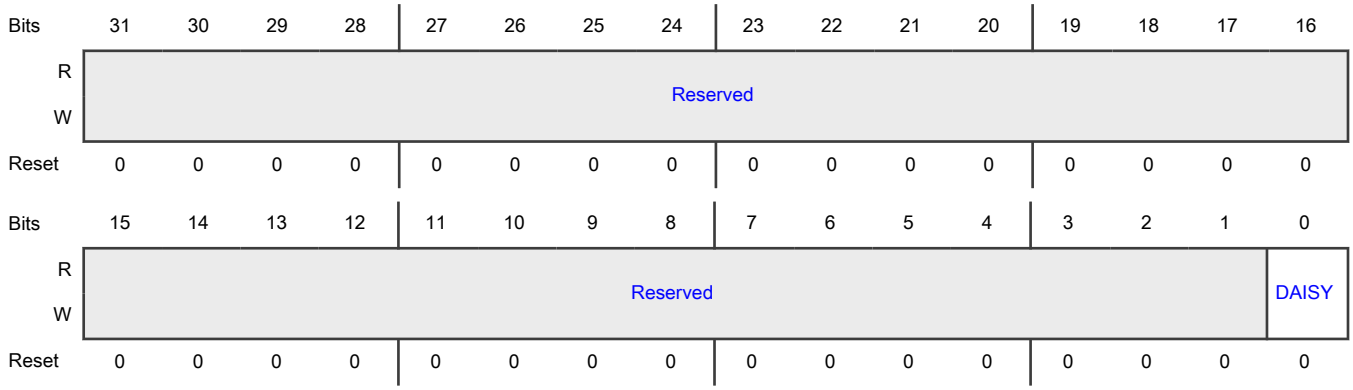
Offset

Register	Offset
FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_1	504h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm1, In Pin: ipp_ind_pwmb1 0b - Selecting Pad: GPIO_EMC_B1_27 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_03 for Mode: ALT4

17.4.1.320 FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_2 DAISY Register (FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_2)

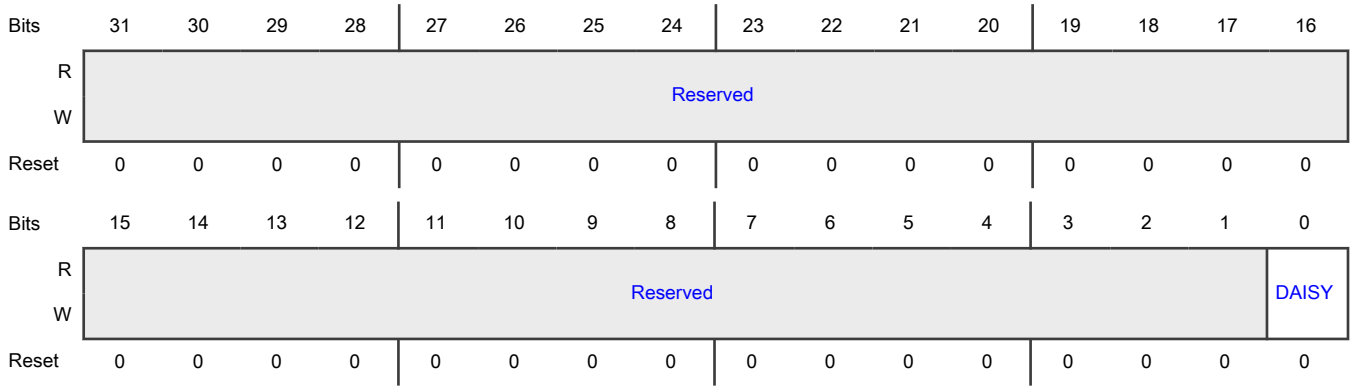
Offset

Register	Offset
FLEXPWM1_IPP_IND_PWMB_SELECT_INPUT_2	508h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm1, In Pin: ipp_ind_pwmb2 0b - Selecting Pad: GPIO_EMC_B1_28 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_04 for Mode: ALT4

17.4.1.321 FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_0 DAISY Register (FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_0)

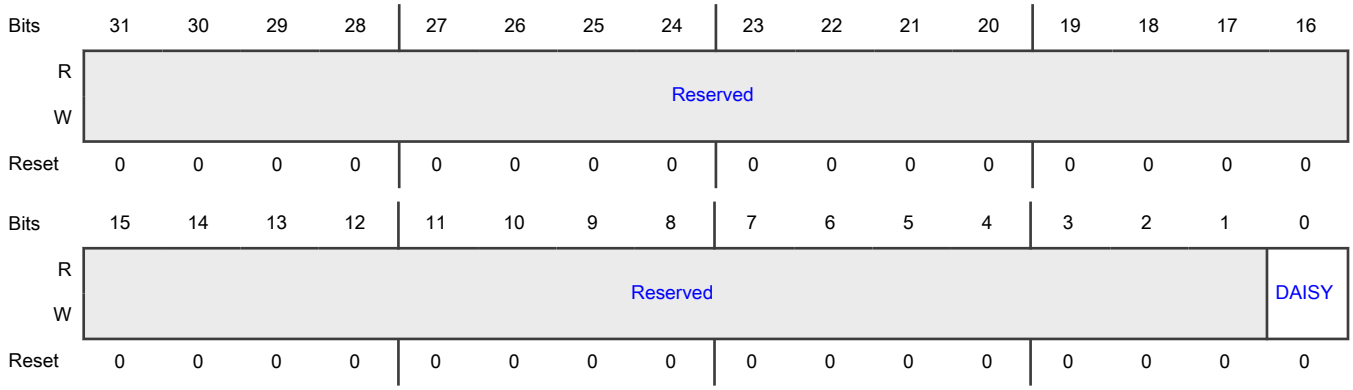
Offset

Register	Offset
FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_0	50Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm2, In Pin: ipp_ind_pwma0 0b - Selecting Pad: GPIO_EMC_B1_18 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_24 for Mode: ALT4

17.4.1.322 FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_1 DAISY Register (FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_1)

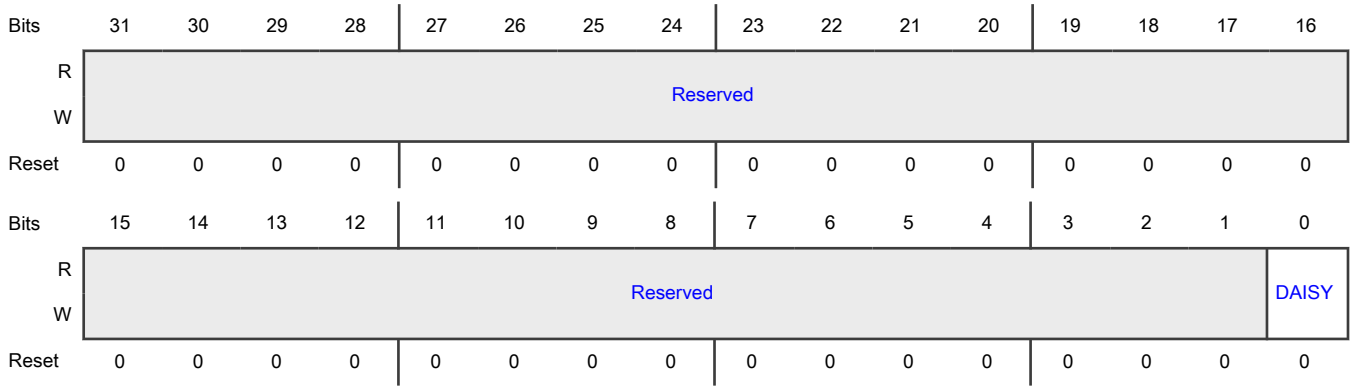
Offset

Register	Offset
FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_1	510h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm2, In Pin: ipp_ind_pwma1 0b - Selecting Pad: GPIO_EMC_B1_20 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_26 for Mode: ALT4

17.4.1.323 FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_2 DAISY Register (FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_2)

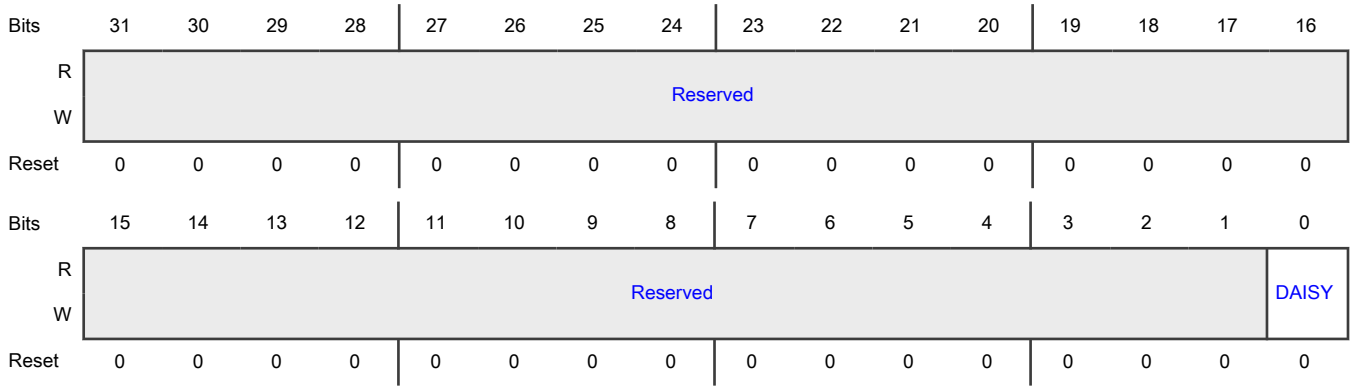
Offset

Register	Offset
FLEXPWM2_IPP_IND_PWMA_SELECT_INPUT_2	514h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm2, In Pin: ipp_ind_pwma2 0b - Selecting Pad: GPIO_EMC_B1_23 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_29 for Mode: ALT4

17.4.1.324 FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_0 DAISY Register (FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_0)

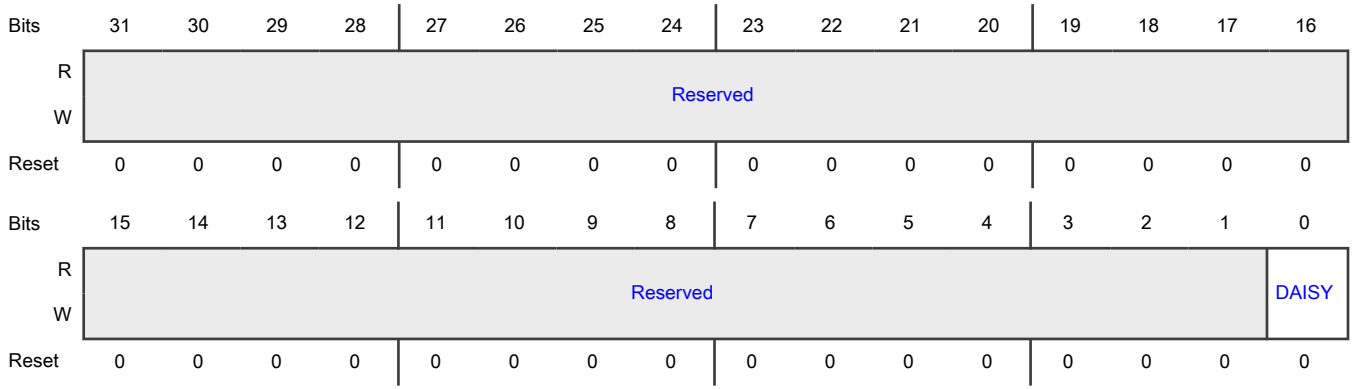
Offset

Register	Offset
FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_0	518h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm2, In Pin: ipp_ind_pwmb0 0b - Selecting Pad: GPIO_EMC_B1_19 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_25 for Mode: ALT4

17.4.1.325 FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_1 DAISY Register (FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_1)

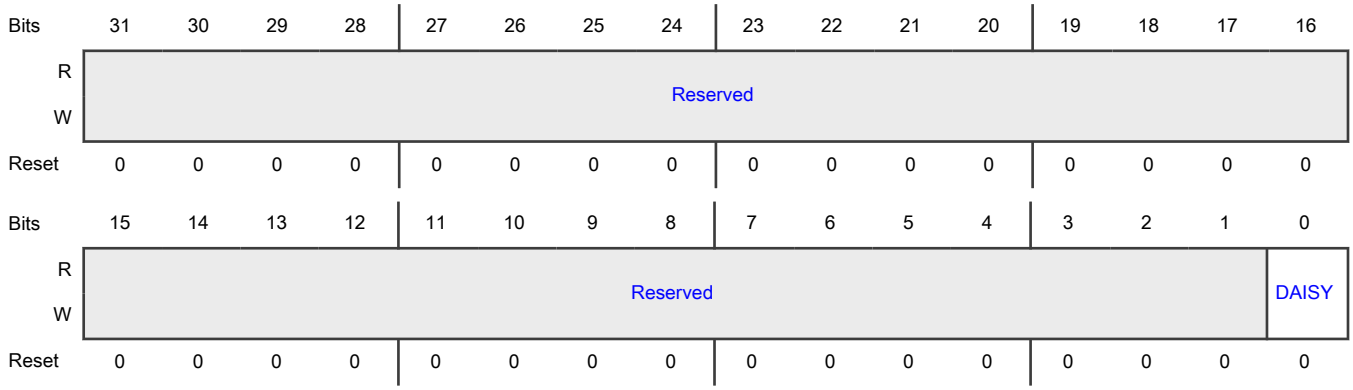
Offset

Register	Offset
FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_1	51Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm2, In Pin: ipp_ind_pwmb1 0b - Selecting Pad: GPIO_EMC_B1_21 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_27 for Mode: ALT4

17.4.1.326 FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_2 DAISY Register (FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_2)

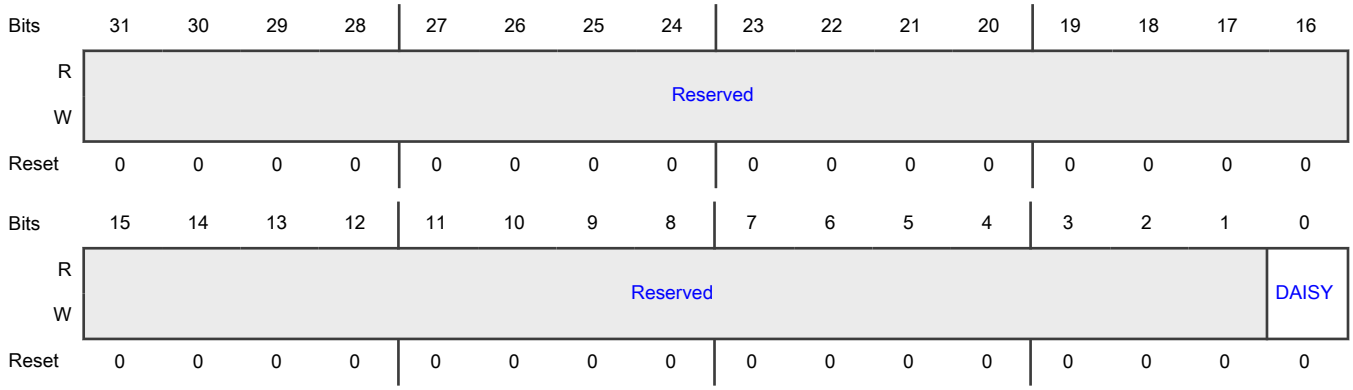
Offset

Register	Offset
FLEXPWM2_IPP_IND_PWMB_SELECT_INPUT_2	520h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm2, In Pin: ipp_ind_pwmb2 0b - Selecting Pad: GPIO_EMC_B1_22 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_28 for Mode: ALT4

17.4.1.327 FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_0 DAISY Register (FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_0)

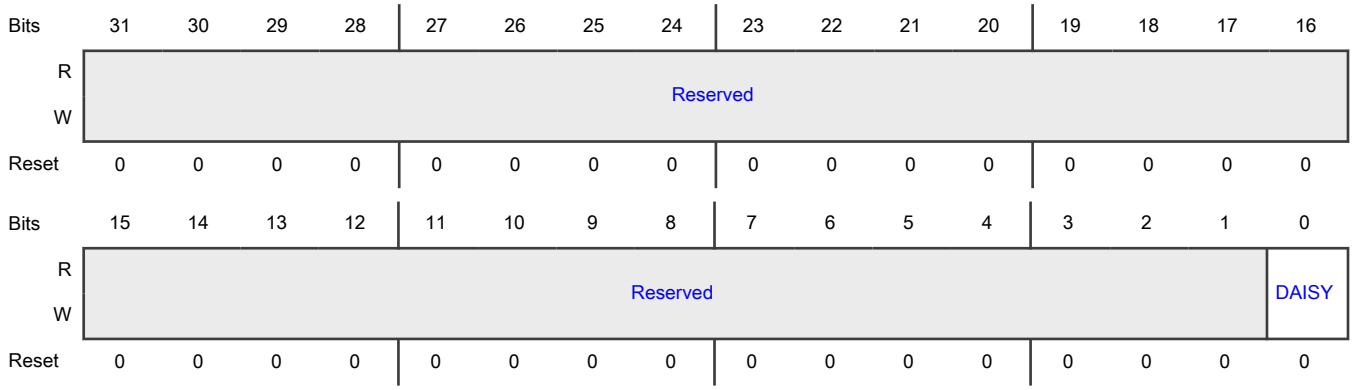
Offset

Register	Offset
FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_0	524h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm3, In Pin: ipp_ind_pwma0 0b - Selecting Pad: GPIO_EMC_B1_30 for Mode: ALT1 1b - Selecting Pad: GPIO_EMC_B2_00 for Mode: ALT10

17.4.1.328 FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_1 DAISY Register (FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_1)

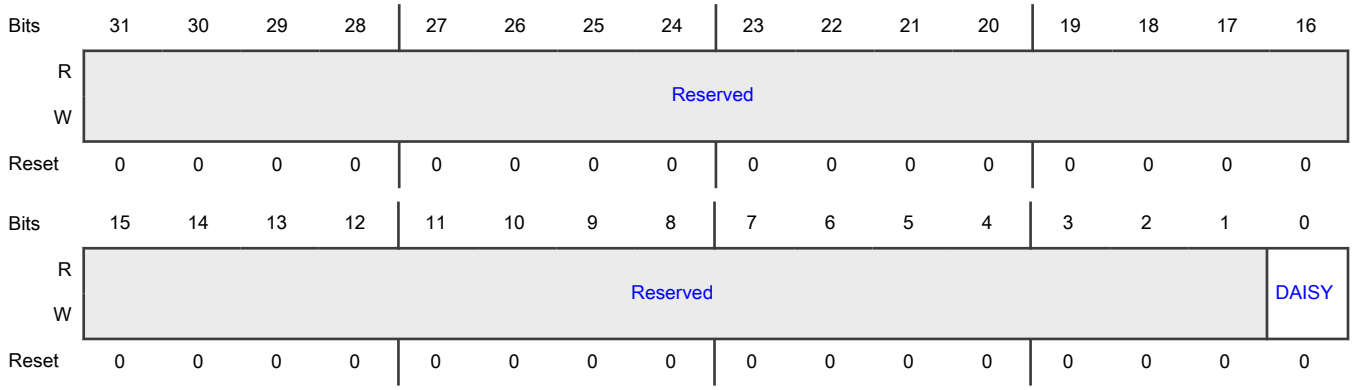
Offset

Register	Offset
FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_1	528h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm3, In Pin: ipp_ind_pwma1 0b - Selecting Pad: GPIO_EMC_B1_32 for Mode: ALT1 1b - Selecting Pad: GPIO_EMC_B2_02 for Mode: ALT10

17.4.1.329 FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_2 DAISY Register (FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_2)

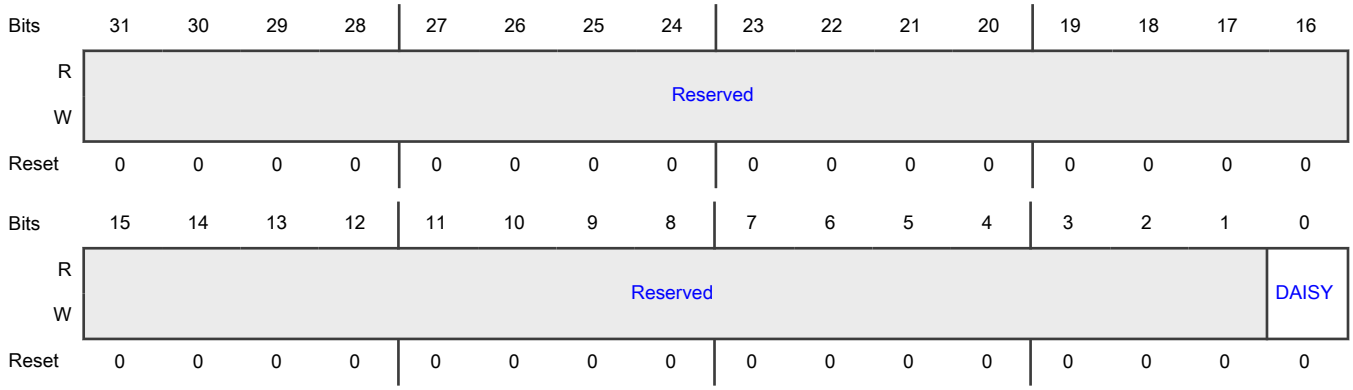
Offset

Register	Offset
FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_2	52Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm3, In Pin: ipp_ind_pwma2 0b - Selecting Pad: GPIO_EMC_B1_35 for Mode: ALT1 1b - Selecting Pad: GPIO_EMC_B2_05 for Mode: ALT10

17.4.1.330 FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_3 DAISY Register (FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_3)

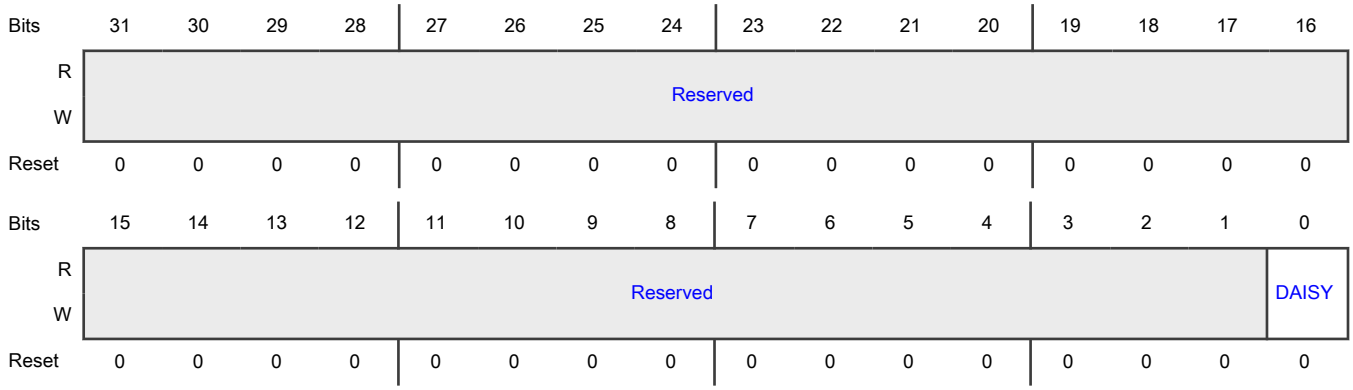
Offset

Register	Offset
FLEXPWM3_IPP_IND_PWMA_SELECT_INPUT_3	530h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm3, In Pin: ipp_ind_pwma3 0b - Selecting Pad: GPIO_EMC_B1_11 for Mode: ALT1 1b - Selecting Pad: GPIO_EMC_B2_07 for Mode: ALT10

17.4.1.331 FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_0 DAISY Register (FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_0)

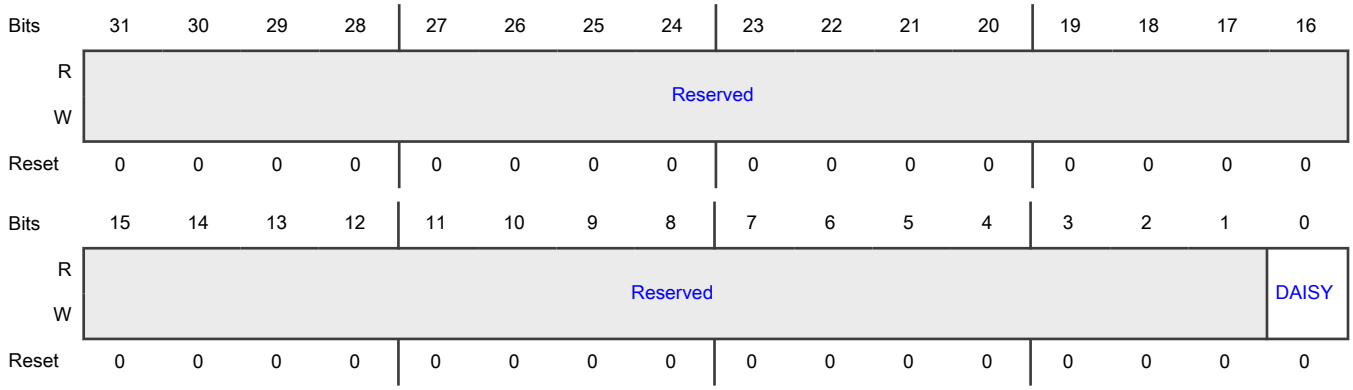
Offset

Register	Offset
FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_0	534h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm3, In Pin: ipp_ind_pwmb0 0b - Selecting Pad: GPIO_EMC_B1_31 for Mode: ALT1 1b - Selecting Pad: GPIO_EMC_B2_01 for Mode: ALT10

17.4.1.332 FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_1 DAISY Register (FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_1)

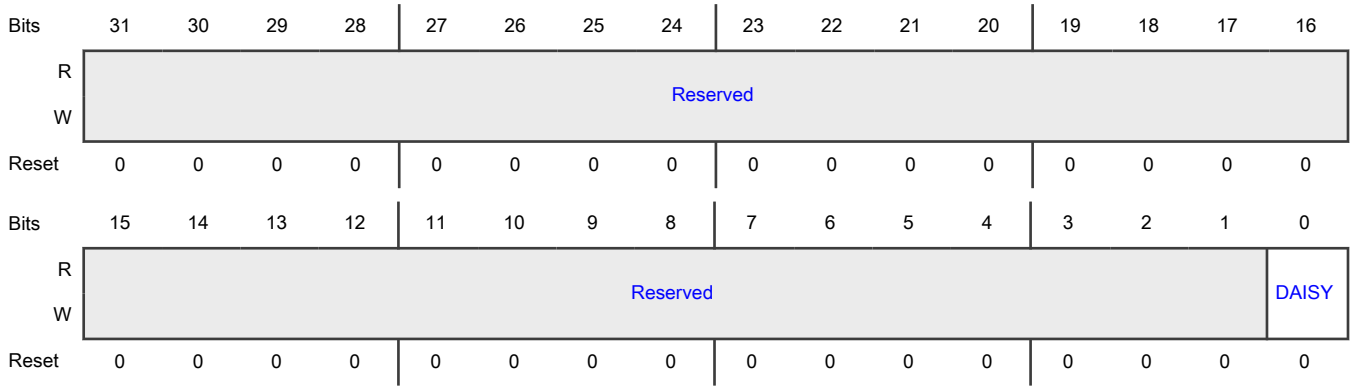
Offset

Register	Offset
FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_1	538h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm3, In Pin: ipp_ind_pwmb1 0b - Selecting Pad: GPIO_EMC_B1_33 for Mode: ALT1 1b - Selecting Pad: GPIO_EMC_B2_03 for Mode: ALT10

17.4.1.333 FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_2 DAISY Register (FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_2)

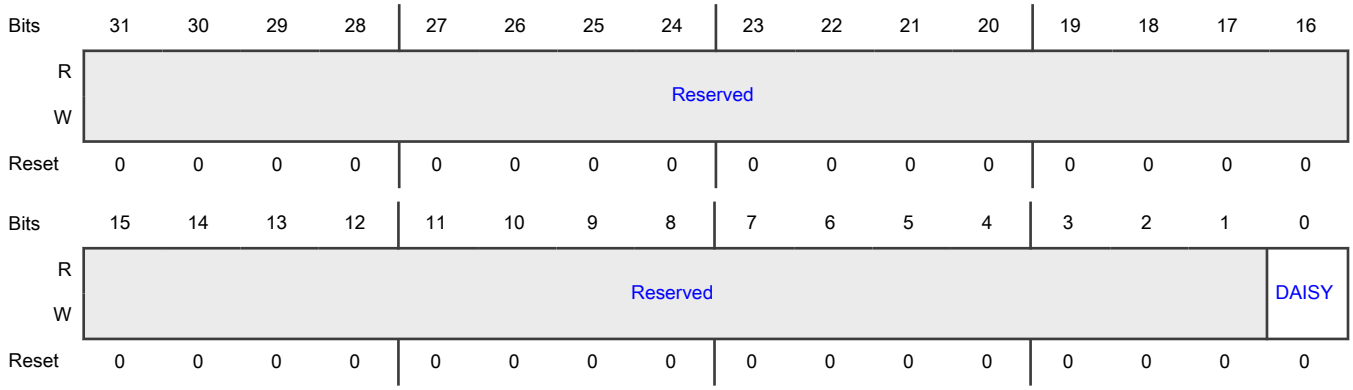
Offset

Register	Offset
FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_2	53Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm3, In Pin: ipp_ind_pwmb2 0b - Selecting Pad: GPIO_EMC_B1_34 for Mode: ALT1 1b - Selecting Pad: GPIO_EMC_B2_04 for Mode: ALT10

17.4.1.334 FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_3 DAISY Register (FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_3)

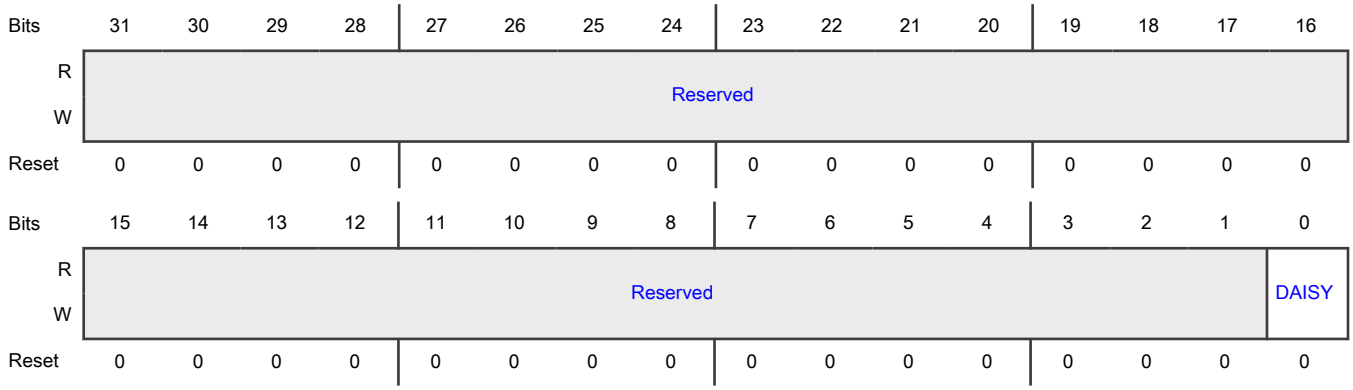
Offset

Register	Offset
FLEXPWM3_IPP_IND_PWMB_SELECT_INPUT_3	540h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexpwm3, In Pin: ipp_ind_pwmb3 0b - Selecting Pad: GPIO_EMC_B1_10 for Mode: ALT1 1b - Selecting Pad: GPIO_EMC_B2_06 for Mode: ALT10

17.4.1.335 FLEXSPI1_BUS2BIT_IPP_IND_DQS_FA_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_DQS_FA_SELECT_INPUT)

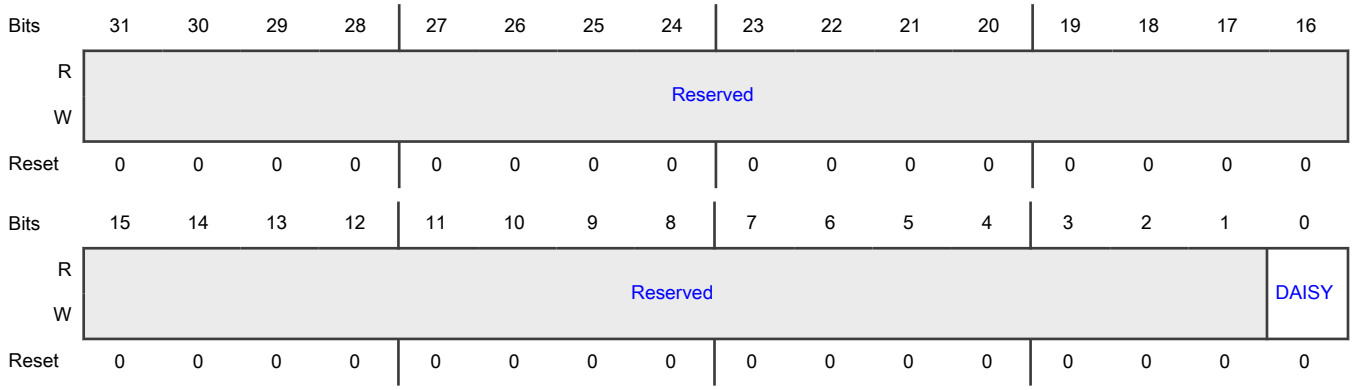
Offset

Register	Offset
FLEXSPI1_BUS2BIT_IPP_IND_DQS_FA_SELECT_INPUT	544h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi1_bus2bit, In Pin: ipp_ind_dqs_fa 0b - Selecting Pad: GPIO_SD_B2_12_DUMMY for Mode: ALT0 1b - Selecting Pad: GPIO_B2_07 for Mode: ALT7

17.4.1.336 FLEXSPI1_BUS2BIT_IPP_IND_DQS_FB_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_DQS_FB_SELECT_INPUT)

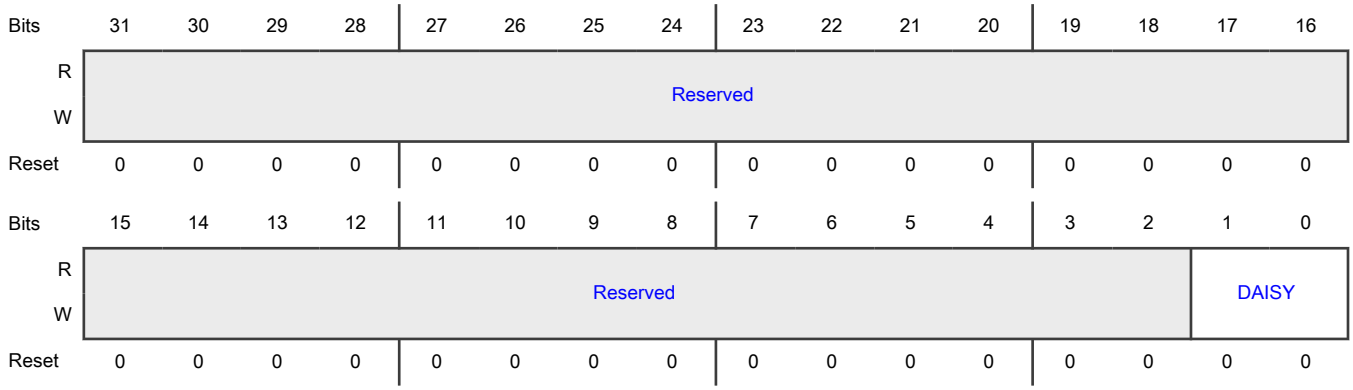
Offset

Register	Offset
FLEXSPI1_BUS2BIT_IPP_IND_DQS_FB_SELECT_INPUT	548h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi1_bus2bit, In Pin: ipp_ind_dqs_fb 00b - Selecting Pad: GPIO_SD_B2_05 for Mode: ALT1 01b - Selecting Pad: GPIO_SD_B2_12_DUMMY for Mode: ALT1 10b - Selecting Pad: GPIO_B1_03 for Mode: ALT7

17.4.1.337 FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT0_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT0_SELECT_INPUT)

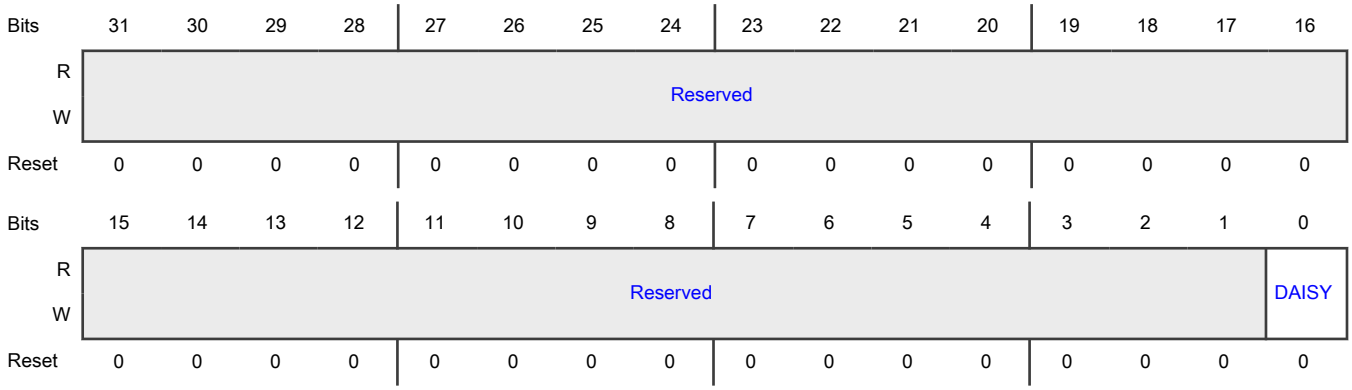
Offset

Register	Offset
FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT0_SELECT_INPUT	54Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi1_bus2bit, In Pin: ipp_ind_io_fb_bit0 0b - Selecting Pad: GPIO_SD_B2_08 for Mode: ALT1 1b - Selecting Pad: GPIO_B1_13 for Mode: ALT7

17.4.1.338 FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT1_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT1_SELECT_INPUT)

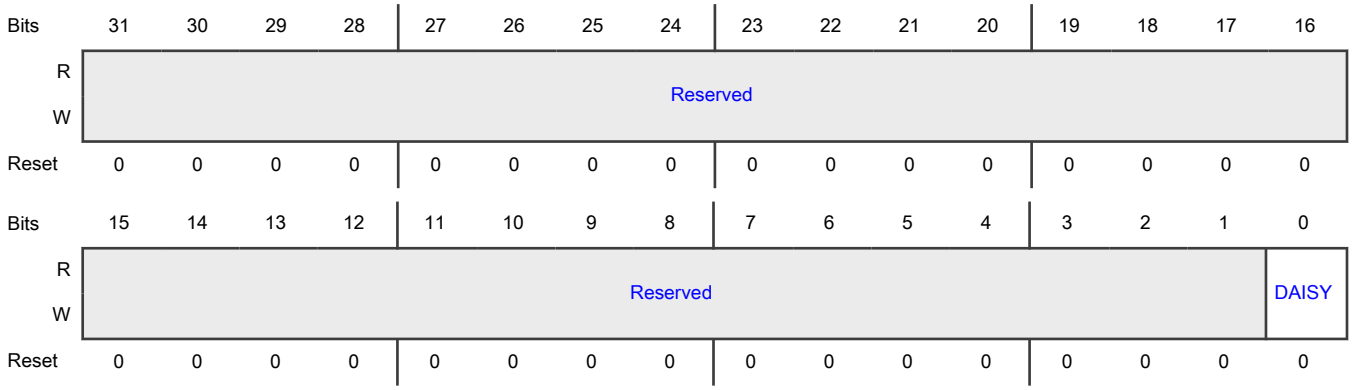
Offset

Register	Offset
FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT1_SELECT_INPUT	550h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi1_bus2bit, In Pin: ipp_ind_io_fb_bit1 0b - Selecting Pad: GPIO_SD_B2_09 for Mode: ALT1 1b - Selecting Pad: GPIO_B1_12 for Mode: ALT7

17.4.1.339 FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT2_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT2_SELECT_INPUT)

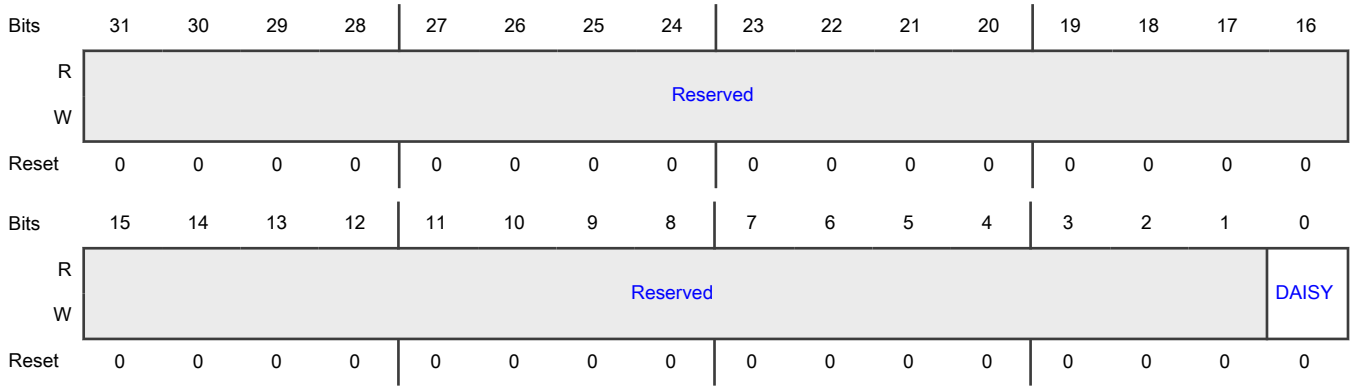
Offset

Register	Offset
FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT2_SELECT_INPUT	554h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi1_bus2bit, In Pin: ipp_ind_io_fb_bit2 0b - Selecting Pad: GPIO_SD_B2_10 for Mode: ALT1 1b - Selecting Pad: GPIO_B1_11 for Mode: ALT7

17.4.1.340 FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT3_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT3_SELECT_INPUT)

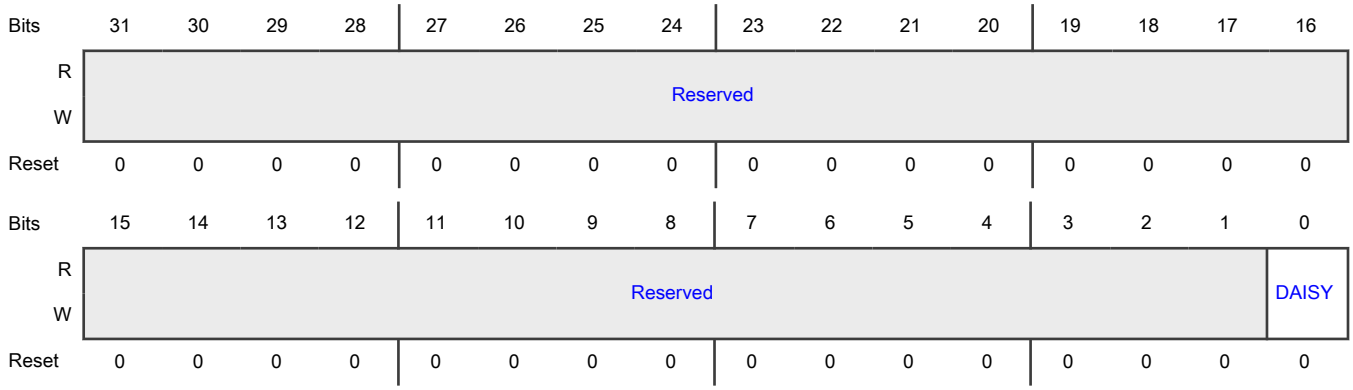
Offset

Register	Offset
FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT3_SELECT_INPUT	558h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi1_bus2bit, In Pin: ipp_ind_io_fb_bit3 0b - Selecting Pad: GPIO_SD_B2_11 for Mode: ALT1 1b - Selecting Pad: GPIO_B1_10 for Mode: ALT7

17.4.1.341 FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT4_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT4_SELECT_INPUT)

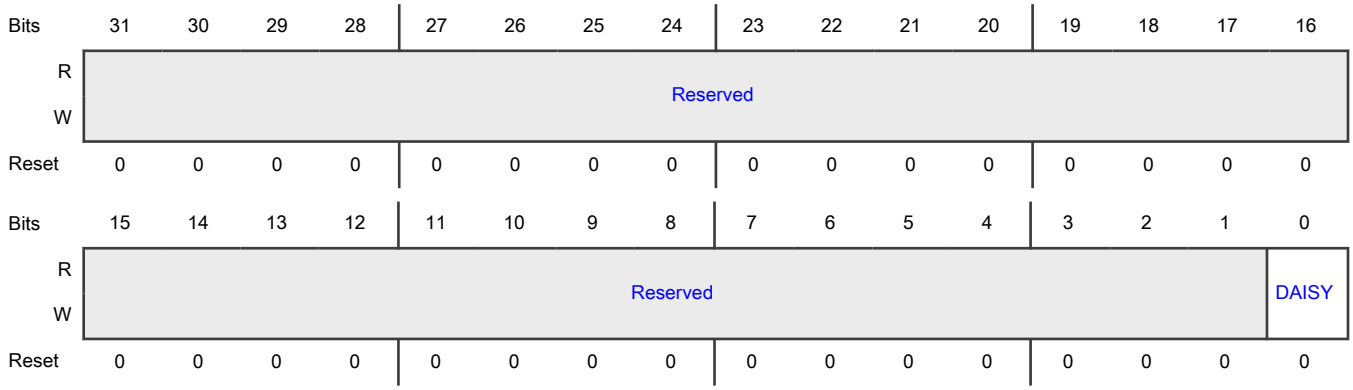
Offset

Register	Offset
FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT4_SELECT_INPUT	55Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi1_bus2bit, In Pin: ipp_ind_io_fb_bit4 0b - Selecting Pad: GPIO_SD_B2_00 for Mode: ALT1 1b - Selecting Pad: GPIO_B1_09 for Mode: ALT7

17.4.1.342 FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT5_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT5_SELECT_INPUT)

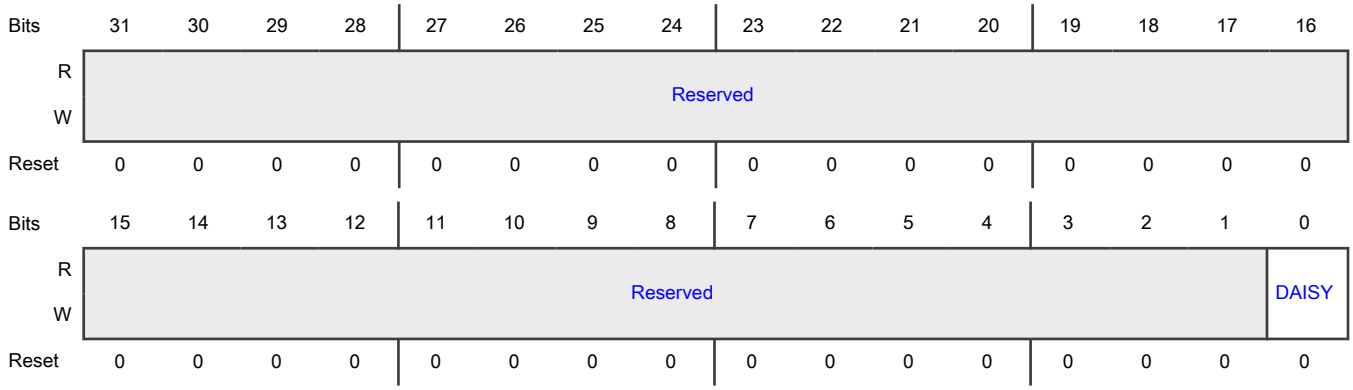
Offset

Register	Offset
FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT5_SELECT_INPUT	560h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi1_bus2bit, In Pin: ipp_ind_io_fb_bit5 0b - Selecting Pad: GPIO_SD_B2_01 for Mode: ALT1 1b - Selecting Pad: GPIO_B1_08 for Mode: ALT7

17.4.1.343 FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT6_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT6_SELECT_INPUT)

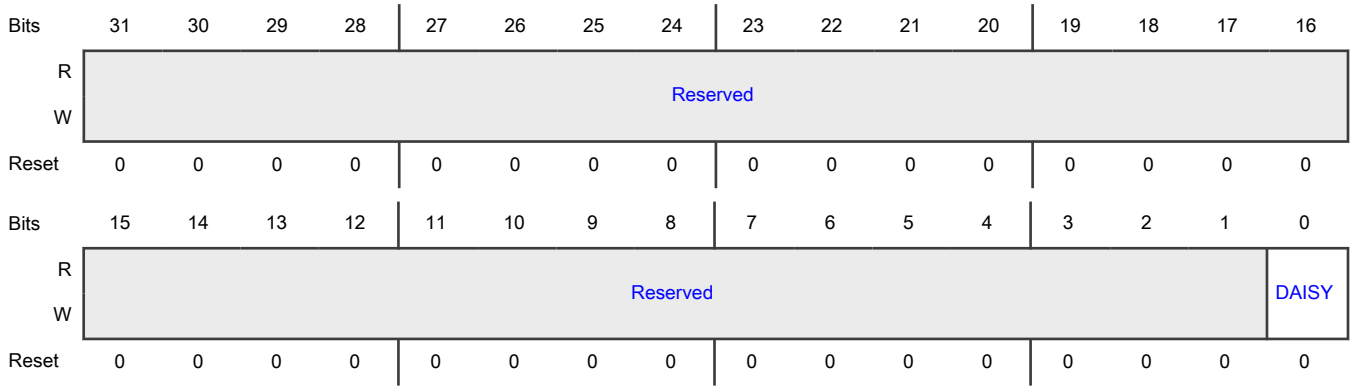
Offset

Register	Offset
FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT6_SELECT_INPUT	564h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi1_bus2bit, In Pin: ipp_ind_io_fb_bit6 0b - Selecting Pad: GPIO_SD_B2_02 for Mode: ALT1 1b - Selecting Pad: GPIO_B1_07 for Mode: ALT7

17.4.1.344 FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT7_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT7_SELECT_INPUT)

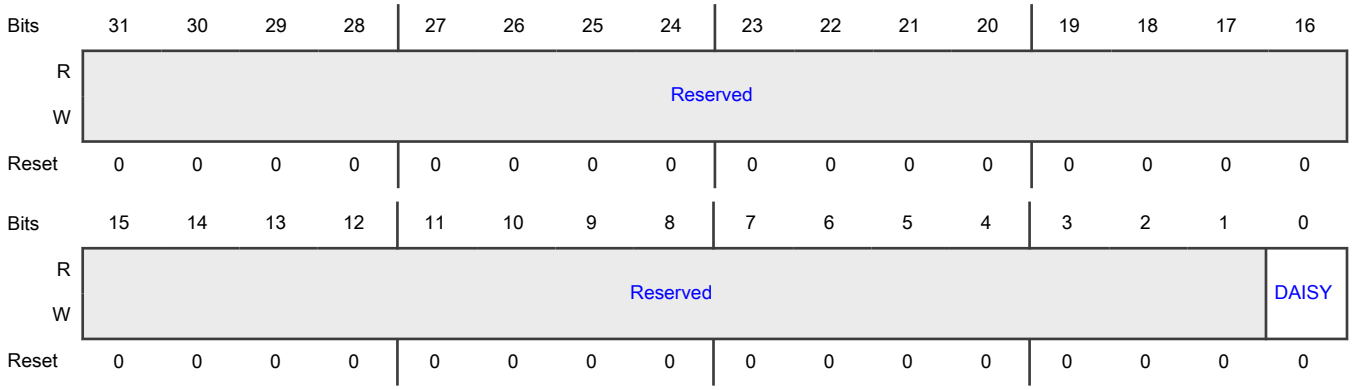
Offset

Register	Offset
FLEXSPI1_BUS2BIT_IPP_IND_IO_FB_BIT7_SELECT_INPUT	568h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi1_bus2bit, In Pin: ipp_ind_io_fb_bit7 0b - Selecting Pad: GPIO_SD_B2_03 for Mode: ALT1 1b - Selecting Pad: GPIO_B1_06 for Mode: ALT7

17.4.1.345 FLEXSPI1_BUS2BIT_IPP_IND_SCK_FB_SELECT_INPUT DAISY Register (FLEXSPI1_BUS2BIT_IPP_IND_SCK_FB_SELECT_INPUT)

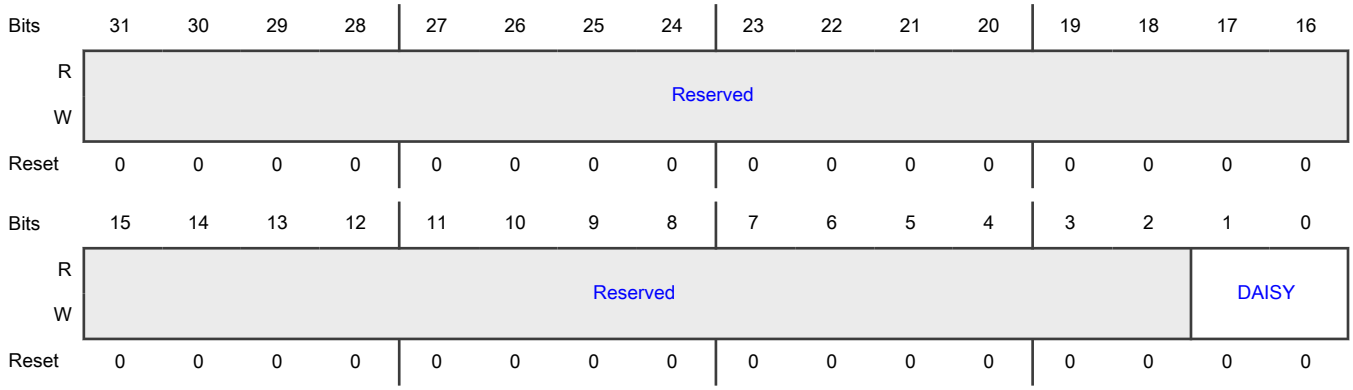
Offset

Register	Offset
FLEXSPI1_BUS2BIT_IPP_IND_SCK_FB_SELECT_INPUT	56Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi1_bus2bit, In Pin: ipp_ind_sck_fb 00b - Selecting Pad: GPIO_SD_B2_07 for Mode: ALT1 01b - Selecting Pad: GPIO_B1_05 for Mode: ALT7 10b - Selecting Pad: GPIO_B2_02 for Mode: ALT6

17.4.1.346 FLEXSPI2_BUS2BIT_IPP_IND_DQS_FA_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_DQS_FA_SELECT_INPUT)

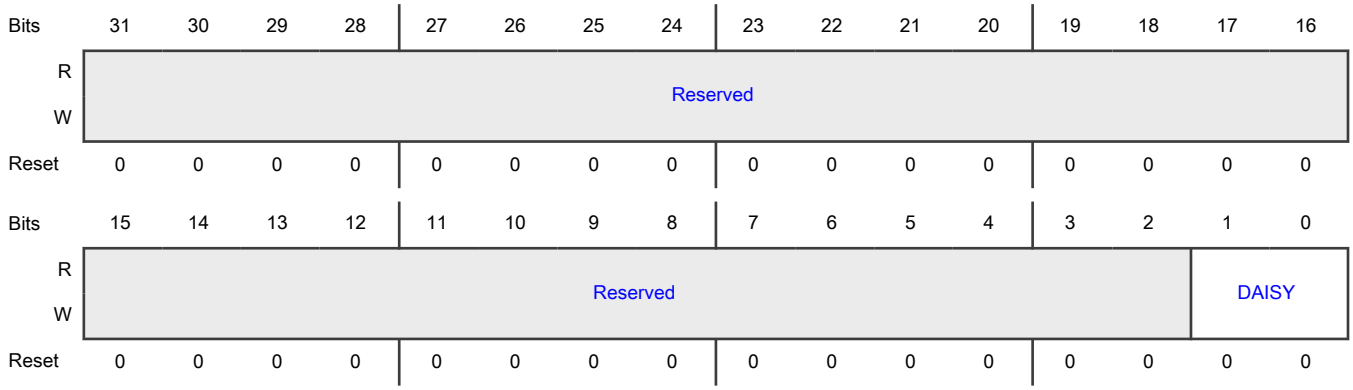
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_DQS_FA_SELECT_INPUT	570h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_dqs_fa 00b - Selecting Pad: GPIO_EMC_B1_40 for Mode: ALT3 01b - Selecting Pad: GPIO_AON_21 for Mode: ALT8 10b - Selecting Pad: GPIO_AON_28_DUMMY for Mode: ALT0

17.4.1.347 FLEXSPI2_BUS2BIT_IPP_IND_DQS_FB_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_DQS_FB_SELECT_INPUT)

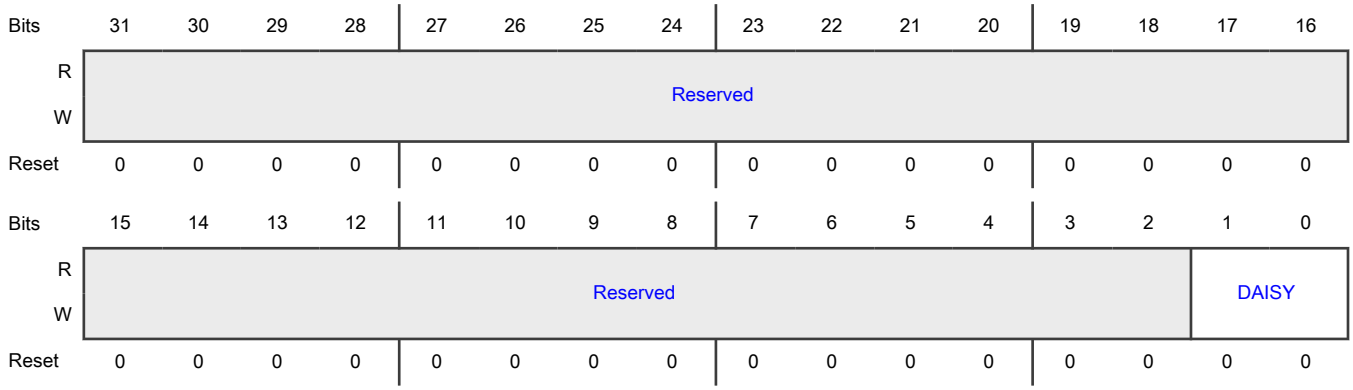
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_DQS_FB_SELECT_INPUT	574h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_dqs_fb 00b - Selecting Pad: GPIO_EMC_B1_21 for Mode: ALT10 01b - Selecting Pad: GPIO_EMC_B1_29 for Mode: ALT3 10b - Selecting Pad: GPIO_AON_20 for Mode: ALT0 11b - Selecting Pad: GPIO_AON_28_DUMMY for Mode: ALT1

17.4.1.348 FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT0_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT0_SELECT_INPUT)

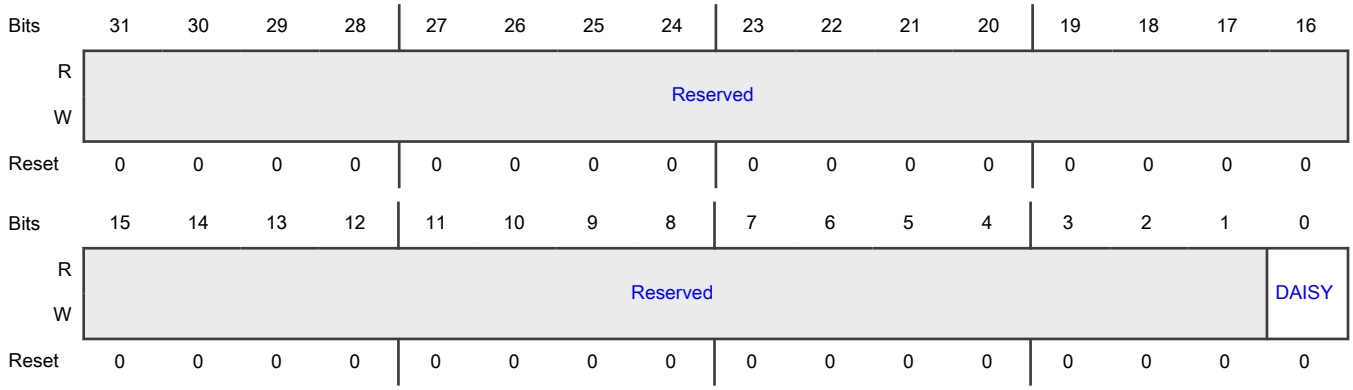
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT0_SELECT_INPUT	578h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_io_fa_bit0 0b - Selecting Pad: GPIO_EMC_B1_35 for Mode: ALT3 1b - Selecting Pad: GPIO_AON_24 for Mode: ALT0

17.4.1.349 FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT1_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT1_SELECT_INPUT)

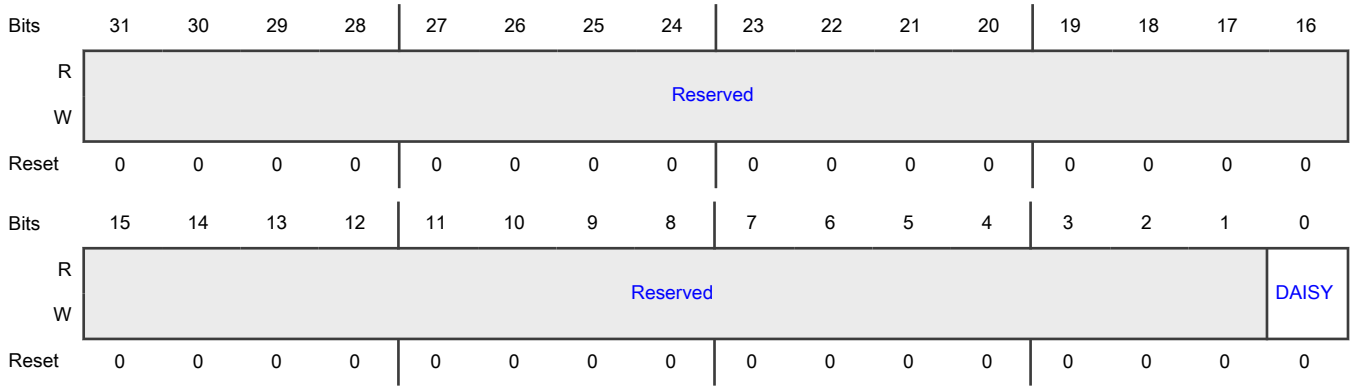
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT1_SELECT_INPUT	57Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_io_fa_bit1 0b - Selecting Pad: GPIO_EMC_B1_36 for Mode: ALT3 1b - Selecting Pad: GPIO_AON_25 for Mode: ALT0

17.4.1.350 FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT2_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT2_SELECT_INPUT)

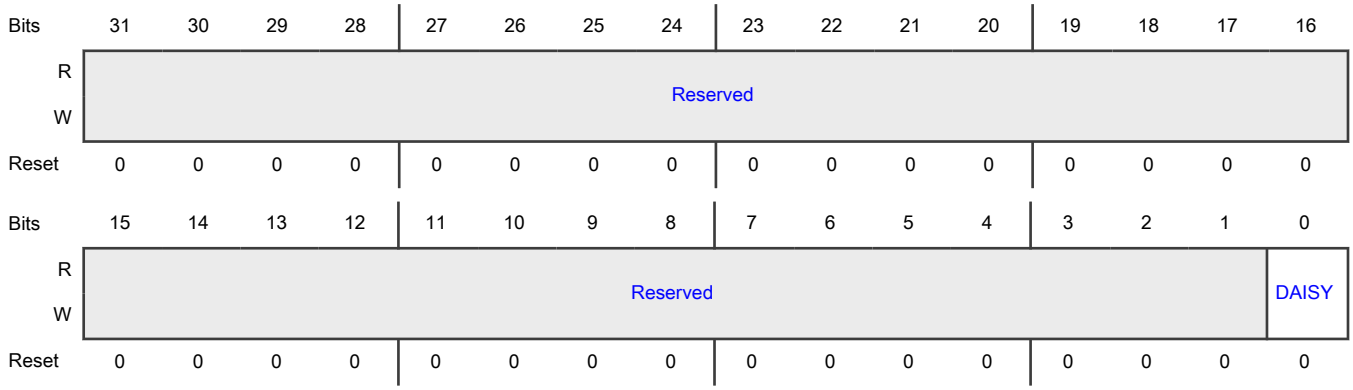
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT2_SELECT_INPUT	580h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_io_fa_bit2 0b - Selecting Pad: GPIO_EMC_B1_37 for Mode: ALT3 1b - Selecting Pad: GPIO_AON_26 for Mode: ALT0

17.4.1.351 FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT3_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT3_SELECT_INPUT)

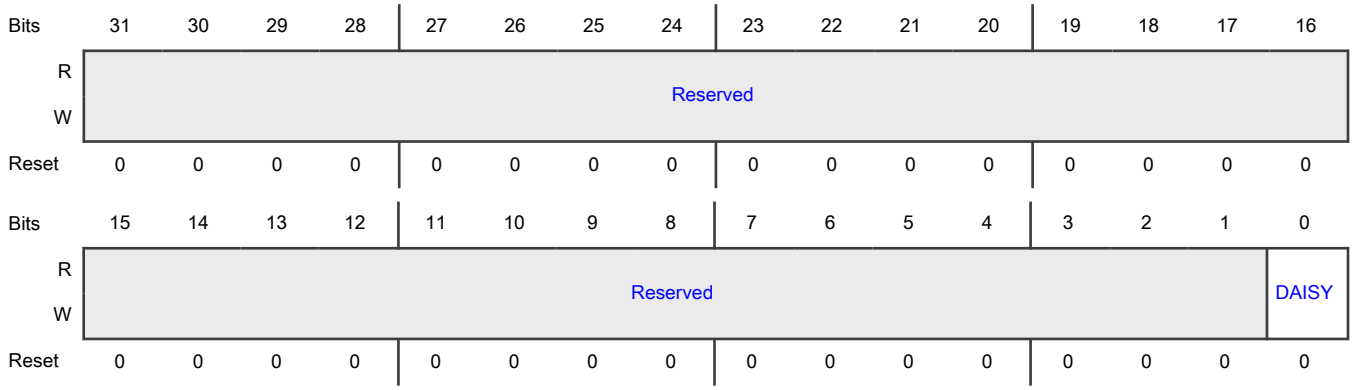
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_IO_FA_BIT3_SELECT_INPUT	584h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_io_fa_bit3 0b - Selecting Pad: GPIO_EMC_B1_38 for Mode: ALT3 1b - Selecting Pad: GPIO_AON_27 for Mode: ALT0

17.4.1.352 FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT0_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT0_SELECT_INPUT)

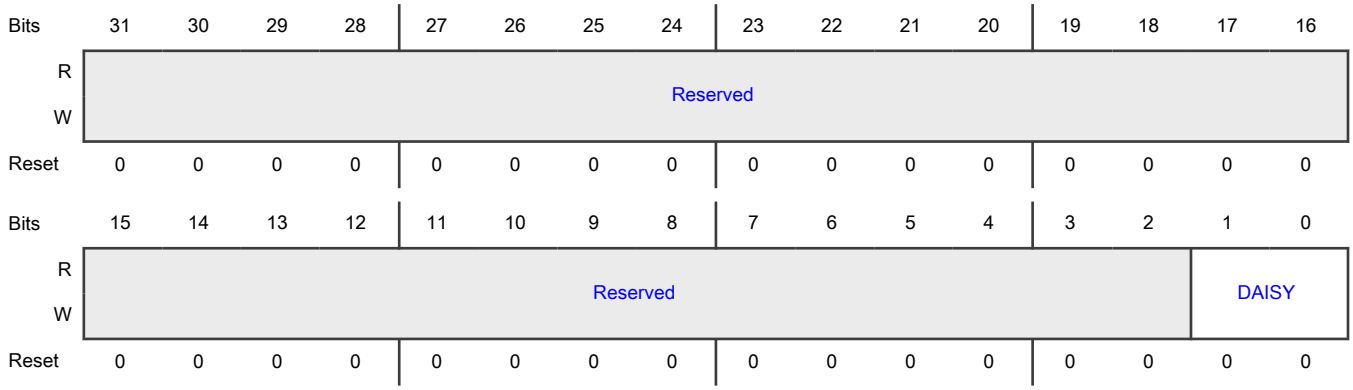
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT0_SELECT_INPUT	588h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_io_fb_bit0 00b - Selecting Pad: GPIO_EMC_B1_25 for Mode: ALT10 01b - Selecting Pad: GPIO_EMC_B1_33 for Mode: ALT3 10b - Selecting Pad: GPIO_AON_18 for Mode: ALT0

17.4.1.353 FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT1_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT1_SELECT_INPUT)

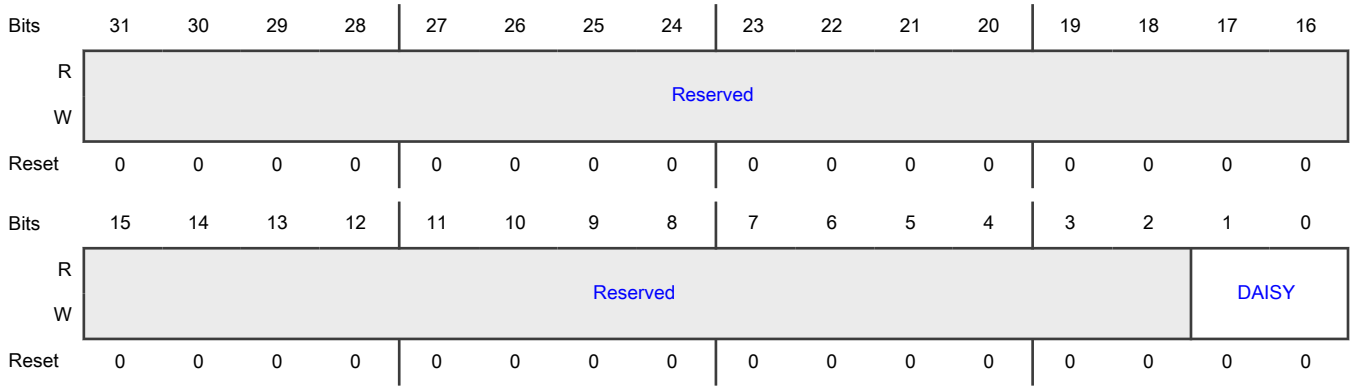
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT1_SELECT_INPUT	58Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_io_fb_bit1 00b - Selecting Pad: GPIO_EMC_B1_24 for Mode: ALT10 01b - Selecting Pad: GPIO_EMC_B1_32 for Mode: ALT3 10b - Selecting Pad: GPIO_AON_17 for Mode: ALT0

17.4.1.354 FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT2_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT2_SELECT_INPUT)

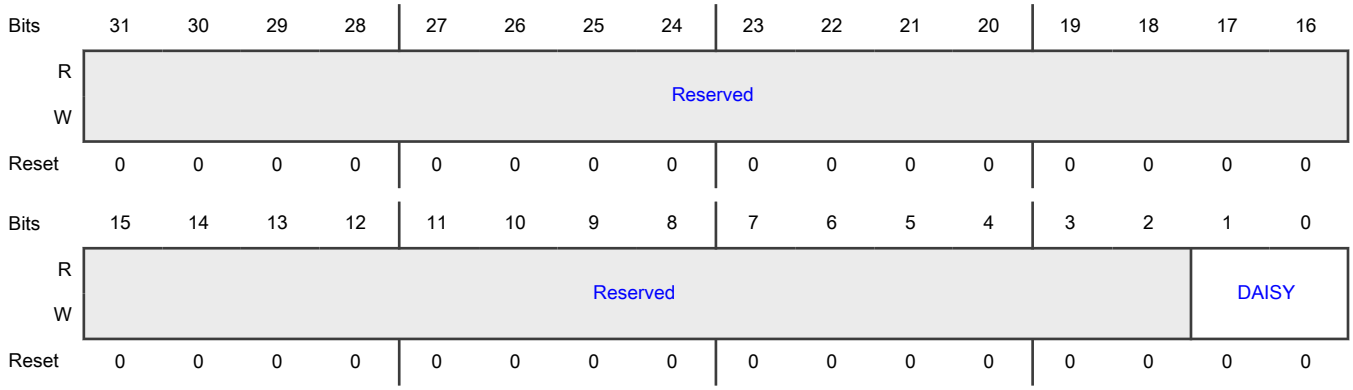
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT2_SELECT_INPUT	590h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_io_fb_bit2 00b - Selecting Pad: GPIO_EMC_B1_23 for Mode: ALT10 01b - Selecting Pad: GPIO_EMC_B1_31 for Mode: ALT3 10b - Selecting Pad: GPIO_AON_16 for Mode: ALT0

17.4.1.355 FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT3_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT3_SELECT_INPUT)

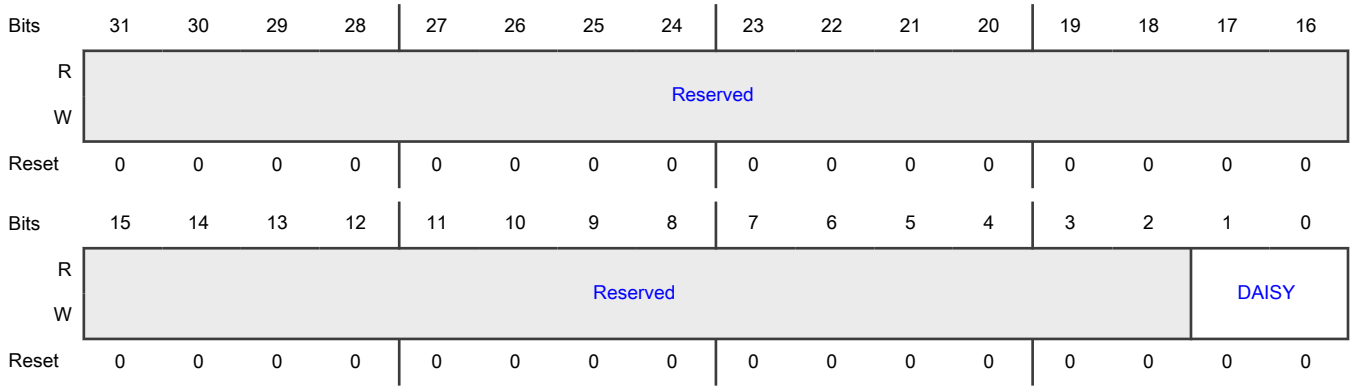
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_IO_FB_BIT3_SELECT_INPUT	594h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_io_fb_bit3 00b - Selecting Pad: GPIO_EMC_B1_22 for Mode: ALT10 01b - Selecting Pad: GPIO_EMC_B1_30 for Mode: ALT3 10b - Selecting Pad: GPIO_AON_15 for Mode: ALT0

17.4.1.356 FLEXSPI2_BUS2BIT_IPP_IND_SCK_FA_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_SCK_FA_SELECT_INPUT)

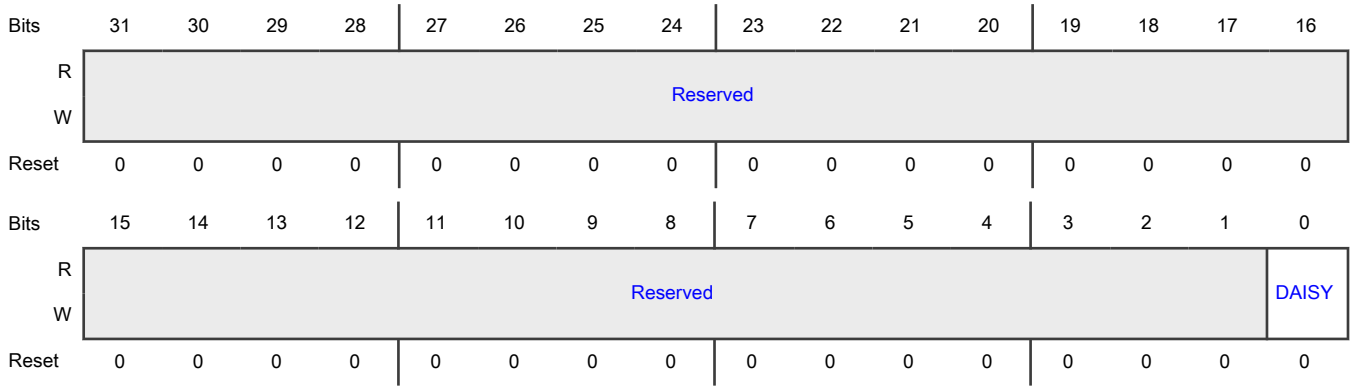
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_SCK_FA_SELECT_INPUT	598h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_sck_fa 0b - Selecting Pad: GPIO_EMC_B1_41 for Mode: ALT3 1b - Selecting Pad: GPIO_AON_23 for Mode: ALT0

17.4.1.357 FLEXSPI2_BUS2BIT_IPP_IND_SCK_FB_SELECT_INPUT DAISY Register (FLEXSPI2_BUS2BIT_IPP_IND_SCK_FB_SELECT_INPUT)

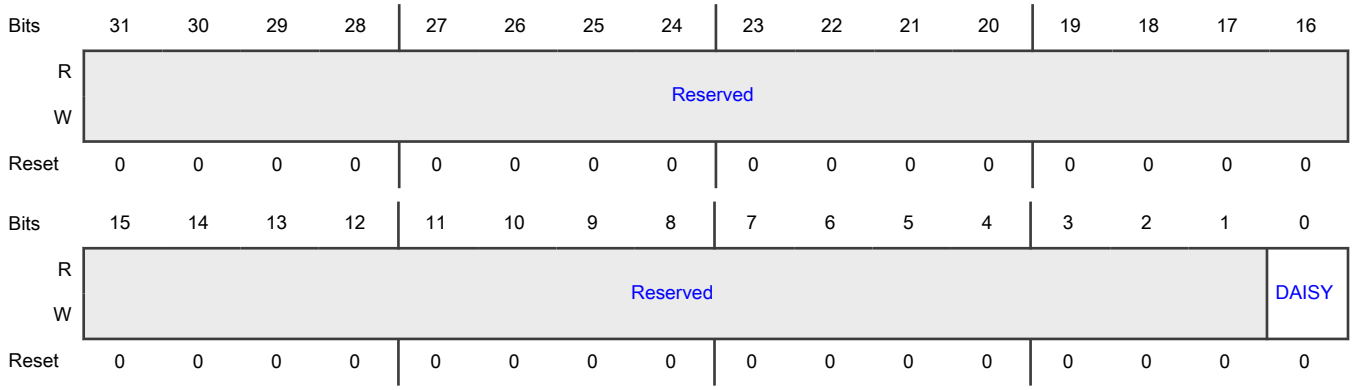
Offset

Register	Offset
FLEXSPI2_BUS2BIT_IPP_IND_SCK_FB_SELECT_INPUT	59Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: flexspi2_bus2bit, In Pin: ipp_ind_sck_fb 0b - Selecting Pad: GPIO_EMC_B1_34 for Mode: ALT3 1b - Selecting Pad: GPIO_AON_19 for Mode: ALT0

17.4.1.358 I3C2_PIN_SCL_IN_SELECT_INPUT DAISY Register (I3C2_PIN_SCL_IN_SELECT_INPUT)

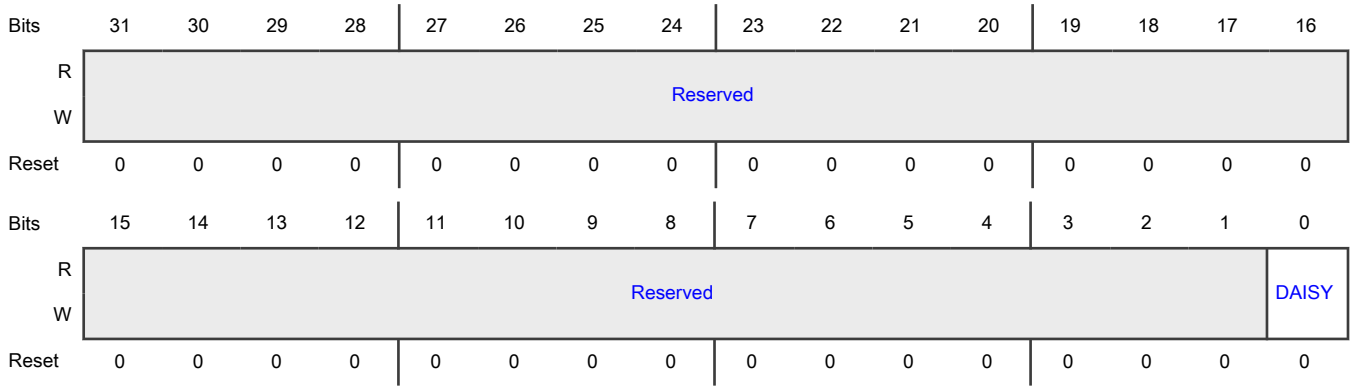
Offset

Register	Offset
I3C2_PIN_SCL_IN_SELECT_INPUT	5A0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: i3c2, In Pin: pin_SCL_in 0b - Selecting Pad: GPIO_AD_18 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_34 for Mode: ALT0

17.4.1.359 I3C2_PIN_SDA_IN_SELECT_INPUT DAISY Register (I3C2_PIN_SDA_IN_SELECT_INPUT)

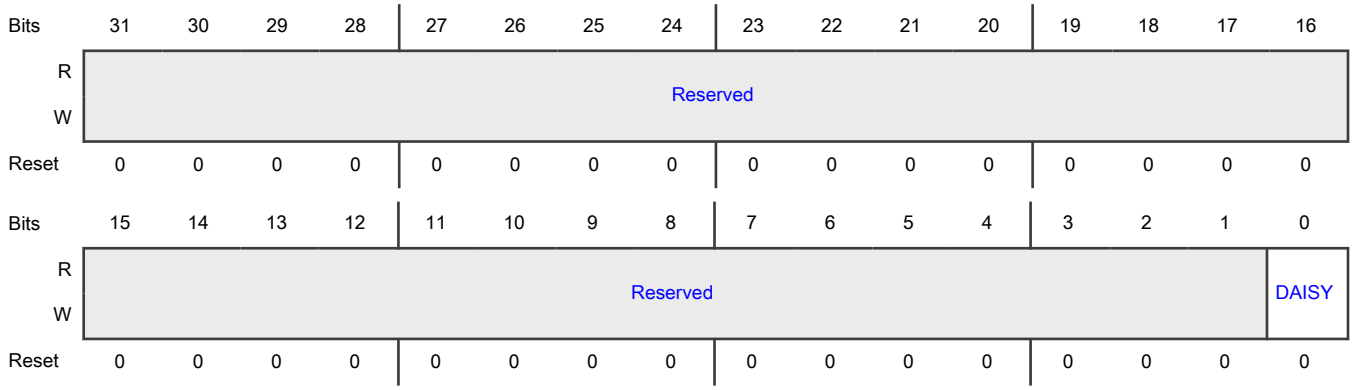
Offset

Register	Offset
I3C2_PIN_SDA_IN_SELECT_INPUT	5A4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: i3c2, In Pin: pin_SDA_in 0b - Selecting Pad: GPIO_AD_19 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_35 for Mode: ALT0

17.4.1.360 KPP_IPP_IND_COL_SELECT_INPUT_0 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_0)

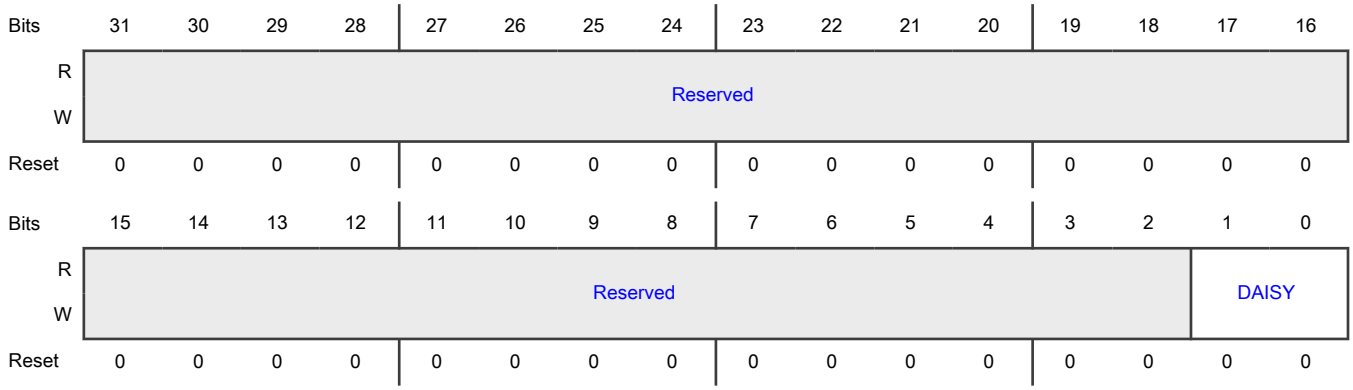
Offset

Register	Offset
KPP_IPP_IND_COL_SELECT_INPUT_0	5A8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_col0 00b - Selecting Pad: GPIO_EMC_B1_15 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_27 for Mode: ALT6 10b - Selecting Pad: GPIO_SD_B2_03 for Mode: ALT4

17.4.1.361 KPP_IPP_IND_COL_SELECT_INPUT_1 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_1)

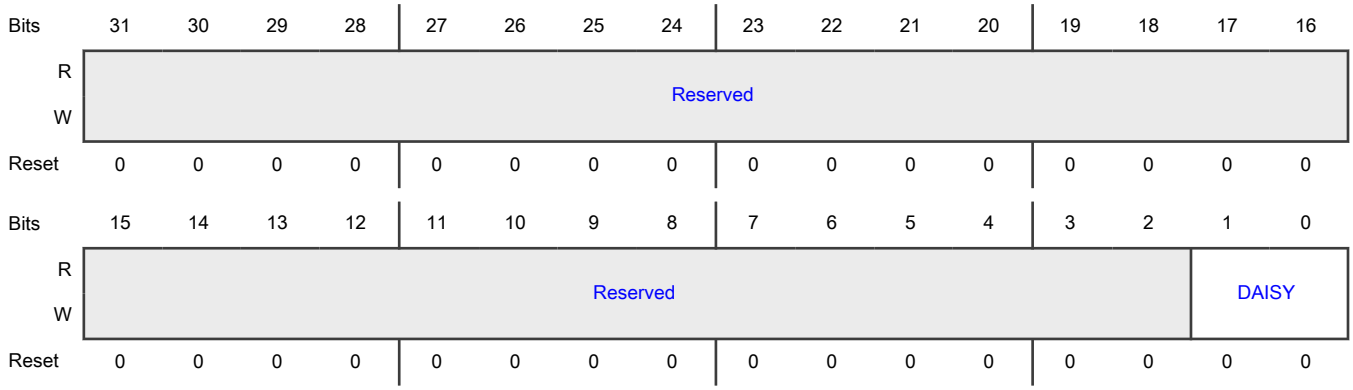
Offset

Register	Offset
KPP_IPP_IND_COL_SELECT_INPUT_1	5ACh

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_col1 00b - Selecting Pad: GPIO_EMC_B1_13 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_33 for Mode: ALT6 10b - Selecting Pad: GPIO_SD_B2_01 for Mode: ALT4

17.4.1.362 KPP_IPP_IND_COL_SELECT_INPUT_2 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_2)

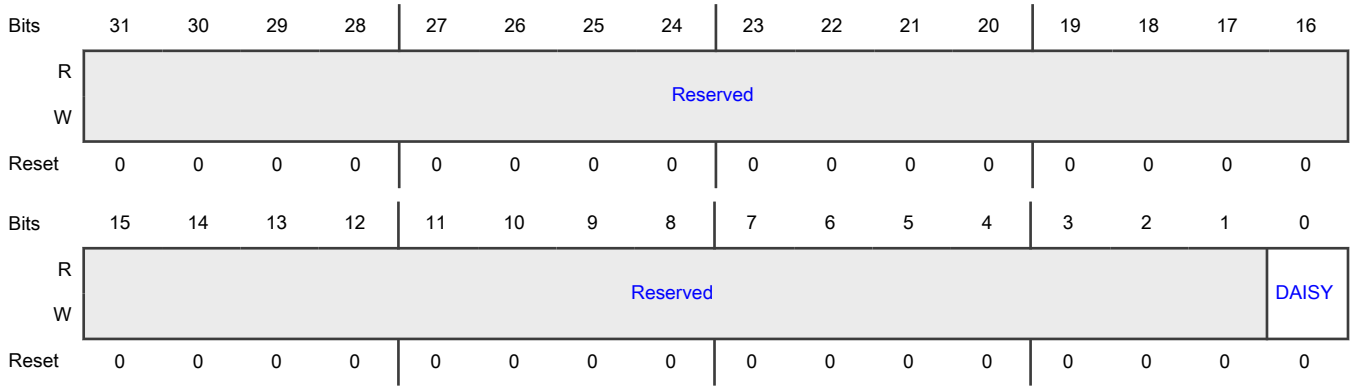
Offset

Register	Offset
KPP_IPP_IND_COL_SELECT_INPUT_2	5B0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_col2 0b - Selecting Pad: GPIO_EMC_B1_03 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_31 for Mode: ALT6

17.4.1.363 KPP_IPP_IND_COL_SELECT_INPUT_3 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_3)

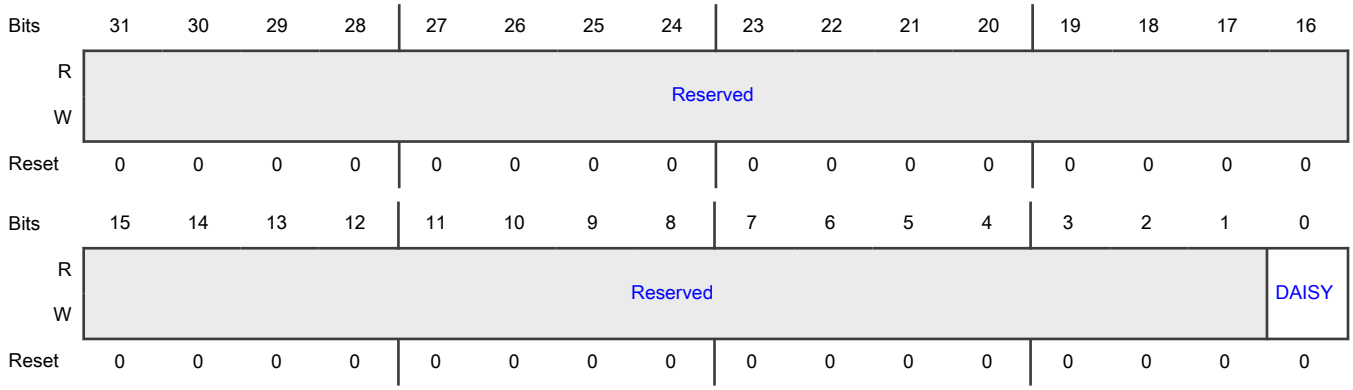
Offset

Register	Offset
KPP_IPP_IND_COL_SELECT_INPUT_3	5B4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_col3 0b - Selecting Pad: GPIO_EMC_B1_01 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_29 for Mode: ALT6

17.4.1.364 KPP_IPP_IND_COL_SELECT_INPUT_4 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_4)

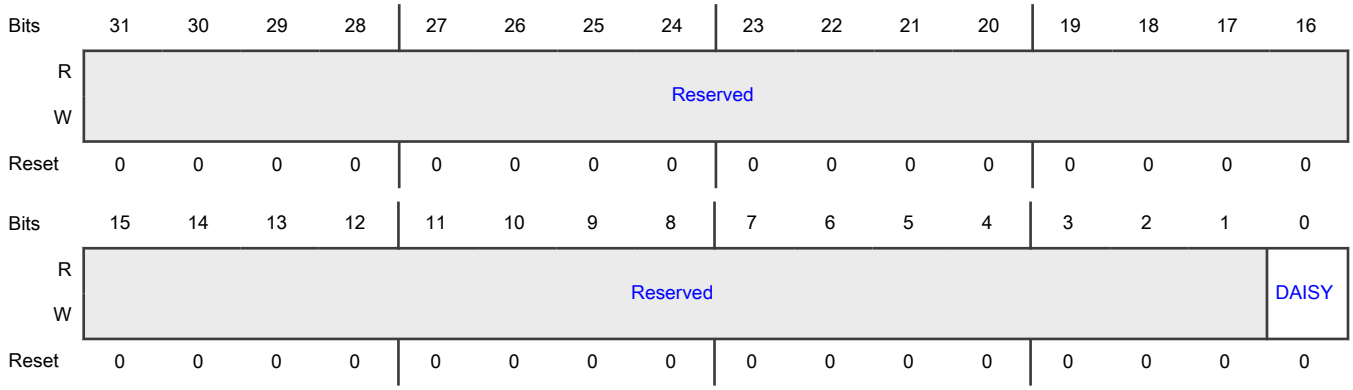
Offset

Register	Offset
KPP_IPP_IND_COL_SELECT_INPUT_4	5B8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_col4 0b - Selecting Pad: GPIO_EMC_B1_12 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_19 for Mode: ALT3

17.4.1.365 KPP_IPP_IND_COL_SELECT_INPUT_5 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_5)

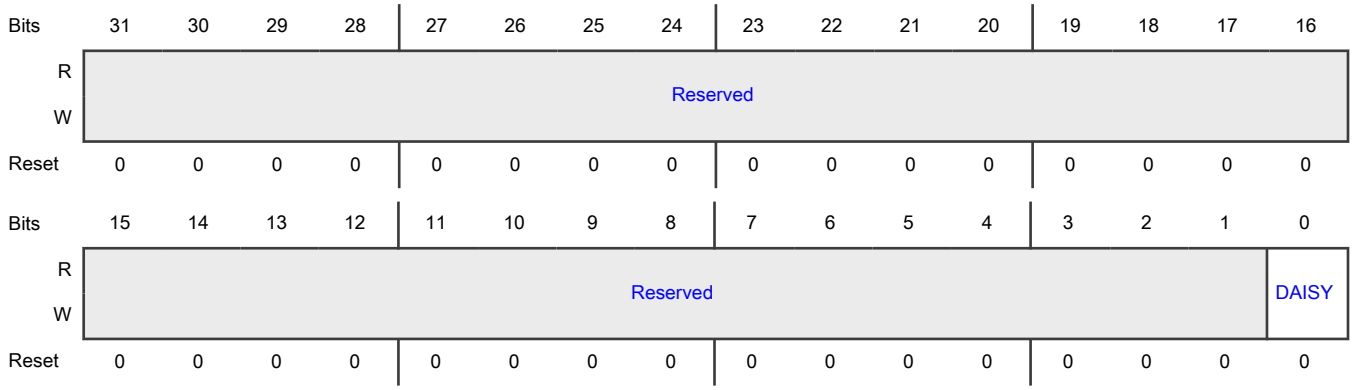
Offset

Register	Offset
KPP_IPP_IND_COL_SELECT_INPUT_5	5BCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_col5 0b - Selecting Pad: GPIO_EMC_B1_10 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_17 for Mode: ALT3

17.4.1.366 KPP_IPP_IND_COL_SELECT_INPUT_6 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_6)

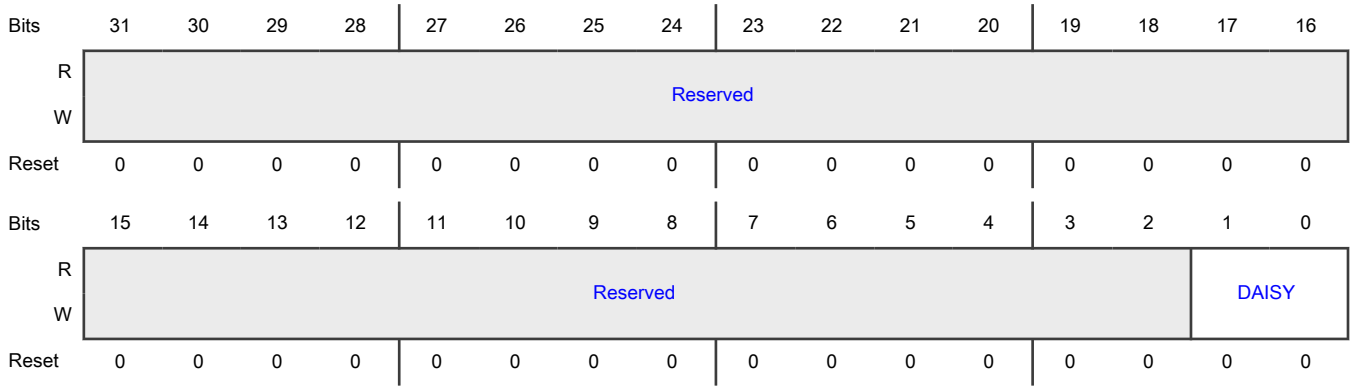
Offset

Register	Offset
KPP_IPP_IND_COL_SELECT_INPUT_6	5C0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_col6 00b - Selecting Pad: GPIO_EMC_B1_08 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_15 for Mode: ALT3 10b - Selecting Pad: GPIO_SD_B1_03 for Mode: ALT8

17.4.1.367 KPP_IPP_IND_COL_SELECT_INPUT_7 DAISY Register (KPP_IPP_IND_COL_SELECT_INPUT_7)

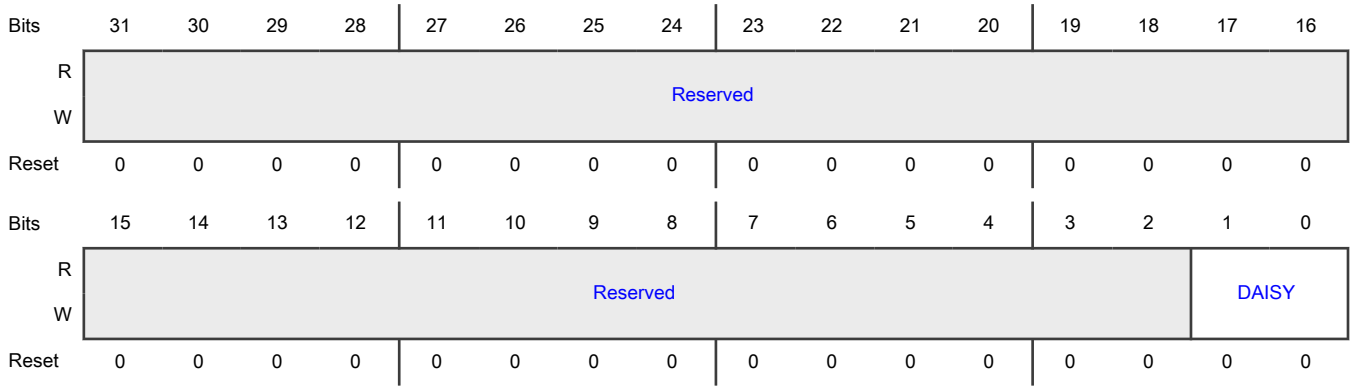
Offset

Register	Offset
KPP_IPP_IND_COL_SELECT_INPUT_7	5C4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_col7 00b - Selecting Pad: GPIO_EMC_B1_06 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_13 for Mode: ALT3 10b - Selecting Pad: GPIO_SD_B1_01 for Mode: ALT8

17.4.1.368 KPP_IPP_IND_ROW_SELECT_INPUT_0 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_0)

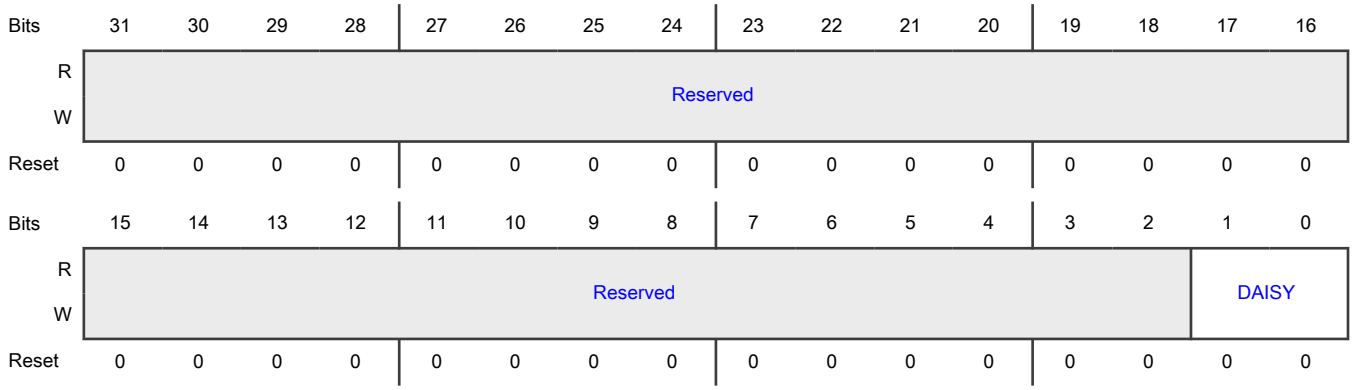
Offset

Register	Offset
KPP_IPP_IND_ROW_SELECT_INPUT_0	5C8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row0 00b - Selecting Pad: GPIO_EMC_B1_14 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_26 for Mode: ALT6 10b - Selecting Pad: GPIO_SD_B2_02 for Mode: ALT4

17.4.1.369 KPP_IPP_IND_ROW_SELECT_INPUT_1 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_1)

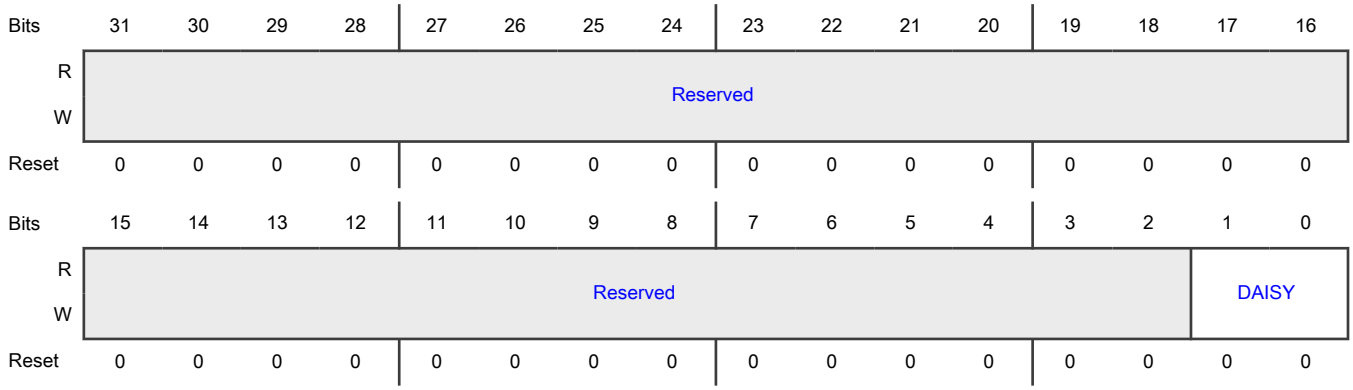
Offset

Register	Offset
KPP_IPP_IND_ROW_SELECT_INPUT_1	5CCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row1 00b - Selecting Pad: GPIO_EMC_B1_04 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_32 for Mode: ALT6 10b - Selecting Pad: GPIO_SD_B2_00 for Mode: ALT4

17.4.1.370 KPP_IPP_IND_ROW_SELECT_INPUT_2 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_2)

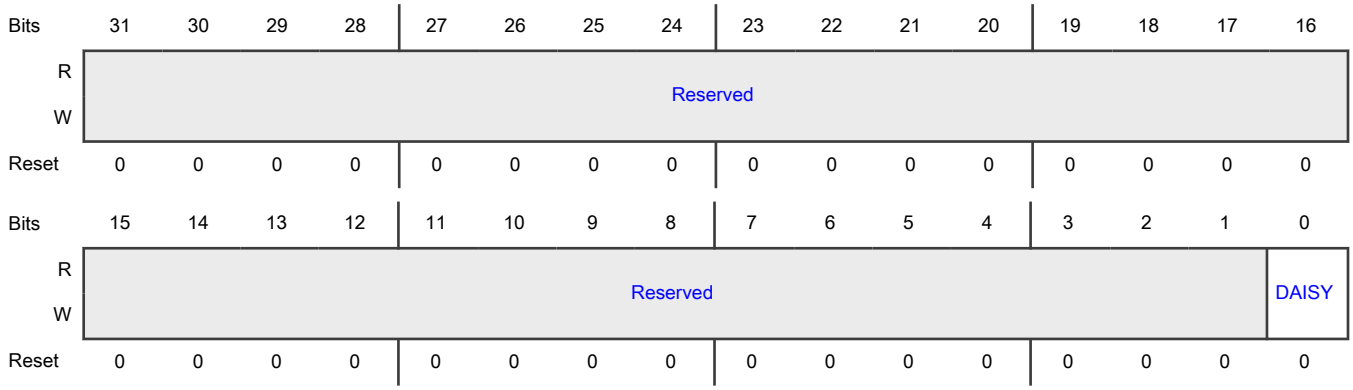
Offset

Register	Offset
KPP_IPP_IND_ROW_SELECT_INPUT_2	5D0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row2 0b - Selecting Pad: GPIO_EMC_B1_02 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_30 for Mode: ALT6

17.4.1.371 KPP_IPP_IND_ROW_SELECT_INPUT_3 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_3)

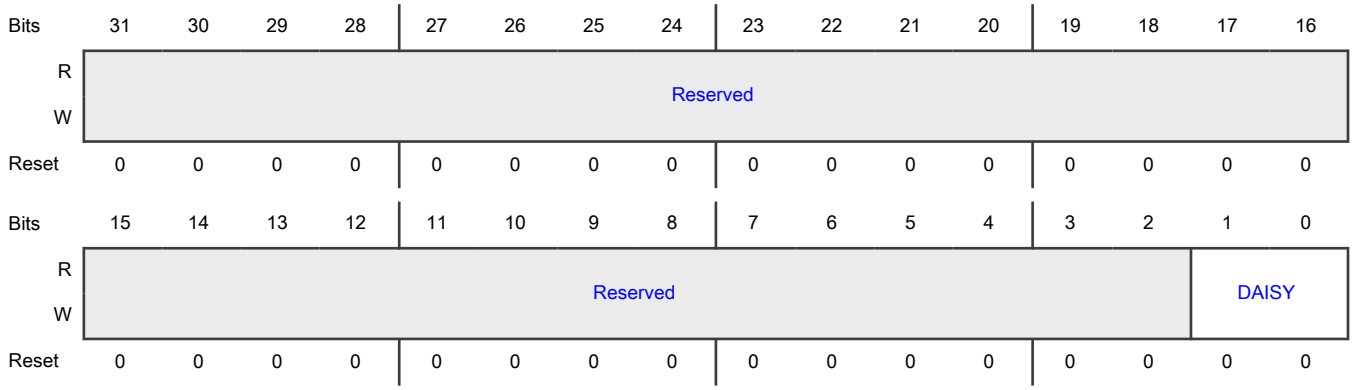
Offset

Register	Offset
KPP_IPP_IND_ROW_SELECT_INPUT_3	5D4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row3 00b - Selecting Pad: GPIO_EMC_B1_00 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_28 for Mode: ALT6 10b - Selecting Pad: GPIO_SD_B2_04 for Mode: ALT4

17.4.1.372 KPP_IPP_IND_ROW_SELECT_INPUT_4 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_4)

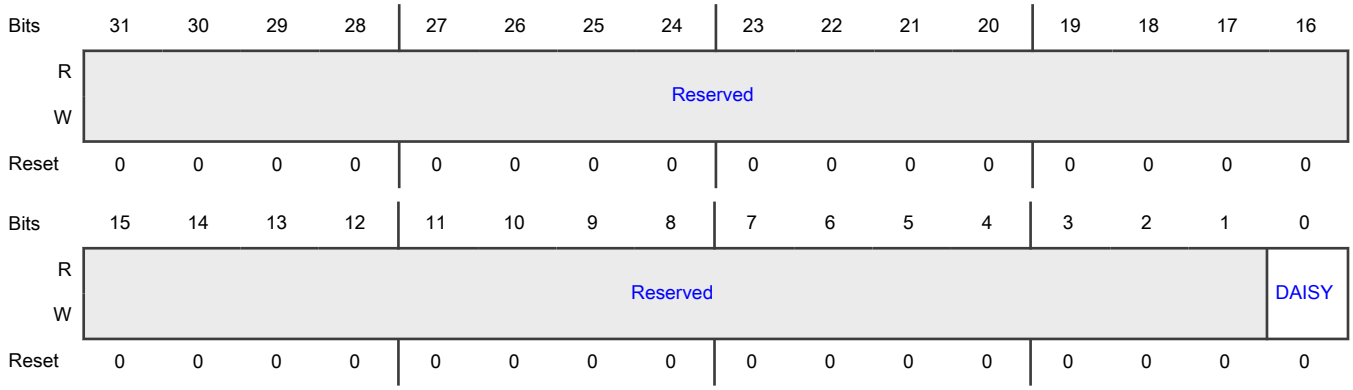
Offset

Register	Offset
KPP_IPP_IND_ROW_SELECT_INPUT_4	5D8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row4 0b - Selecting Pad: GPIO_EMC_B1_11 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_18 for Mode: ALT3

17.4.1.373 KPP_IPP_IND_ROW_SELECT_INPUT_5 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_5)

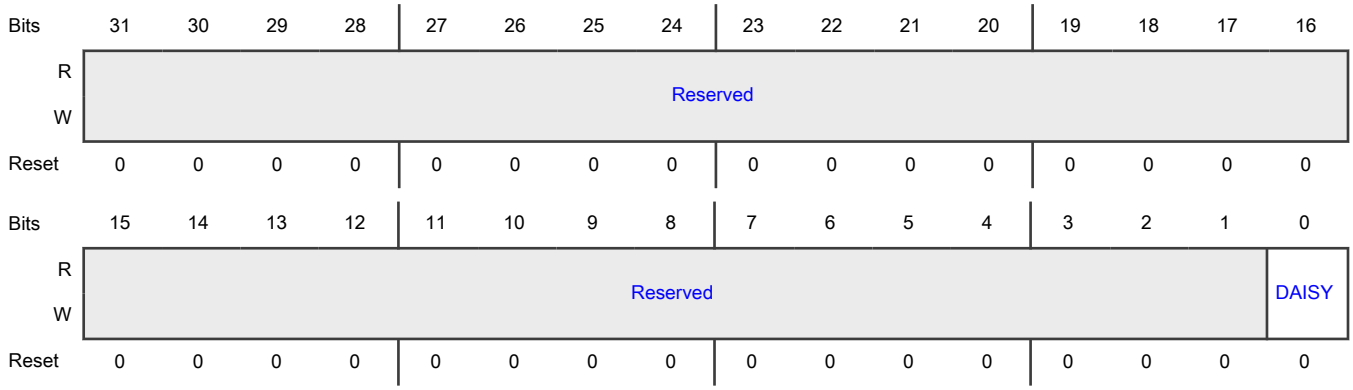
Offset

Register	Offset
KPP_IPP_IND_ROW_SELECT_INPUT_5	5DCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row5 0b - Selecting Pad: GPIO_EMC_B1_09 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_16 for Mode: ALT3

17.4.1.374 KPP_IPP_IND_ROW_SELECT_INPUT_6 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_6)

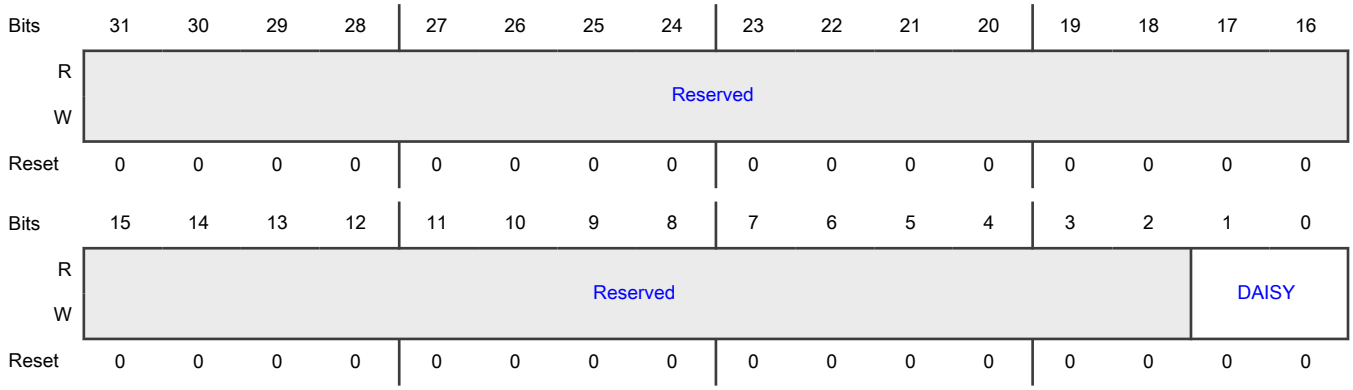
Offset

Register	Offset
KPP_IPP_IND_ROW_SELECT_INPUT_6	5E0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row6 00b - Selecting Pad: GPIO_EMC_B1_07 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_14 for Mode: ALT3 10b - Selecting Pad: GPIO_SD_B1_02 for Mode: ALT8

17.4.1.375 KPP_IPP_IND_ROW_SELECT_INPUT_7 DAISY Register (KPP_IPP_IND_ROW_SELECT_INPUT_7)

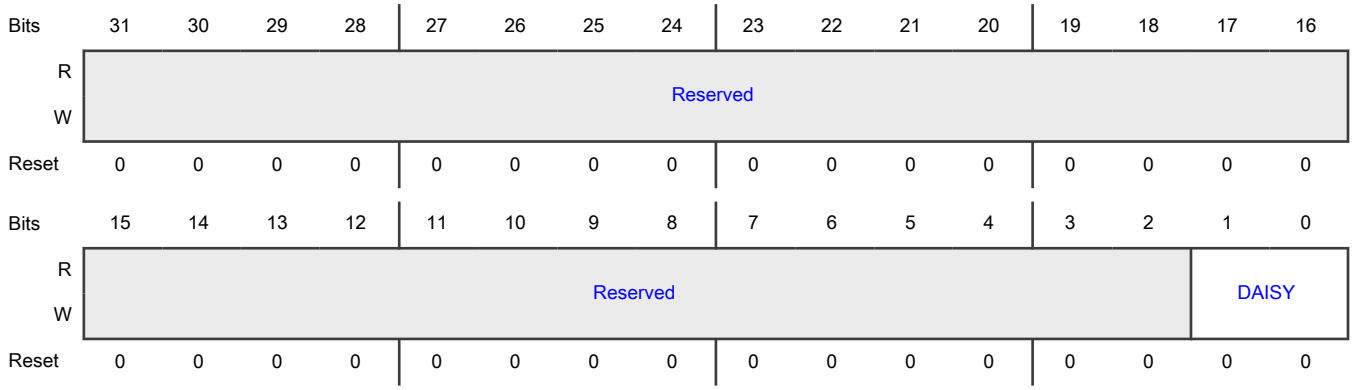
Offset

Register	Offset
KPP_IPP_IND_ROW_SELECT_INPUT_7	5E4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row7 00b - Selecting Pad: GPIO_EMC_B1_05 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_12 for Mode: ALT3 10b - Selecting Pad: GPIO_SD_B1_00 for Mode: ALT8

17.4.1.376 LPI2C3_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C3_IPP_IND_LPI2C_SCL_SELECT_INPUT)

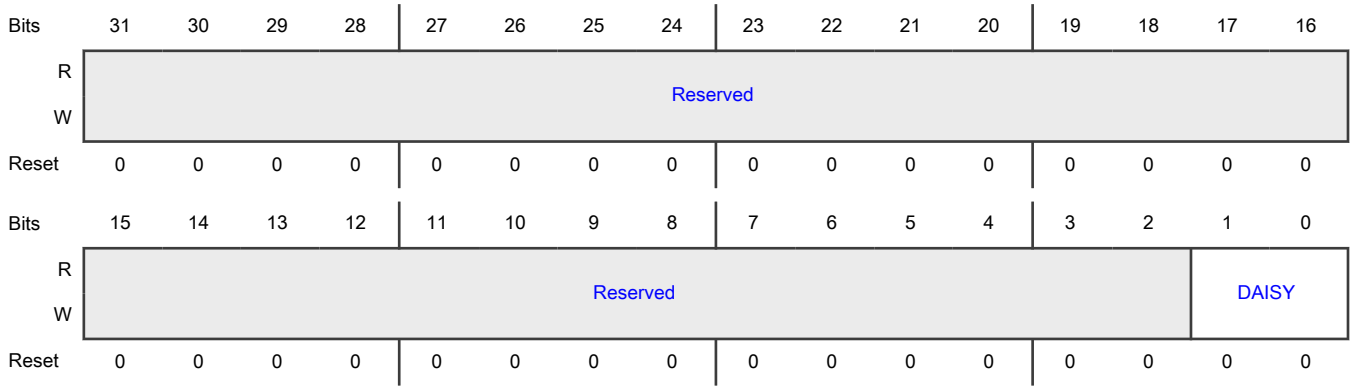
Offset

Register	Offset
LPI2C3_IPP_IND_LPI2C_SCL_SELECT_INPUT	5E8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c3, In Pin: ipp_ind_lpi2c_scl 00b - Selecting Pad: GPIO_EMC_B2_00 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_19 for Mode: ALT8 10b - Selecting Pad: GPIO_AD_18 for Mode: ALT9

17.4.1.377 LPI2C3_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C3_IPP_IND_LPI2C_SDA_SELECT_INPUT)

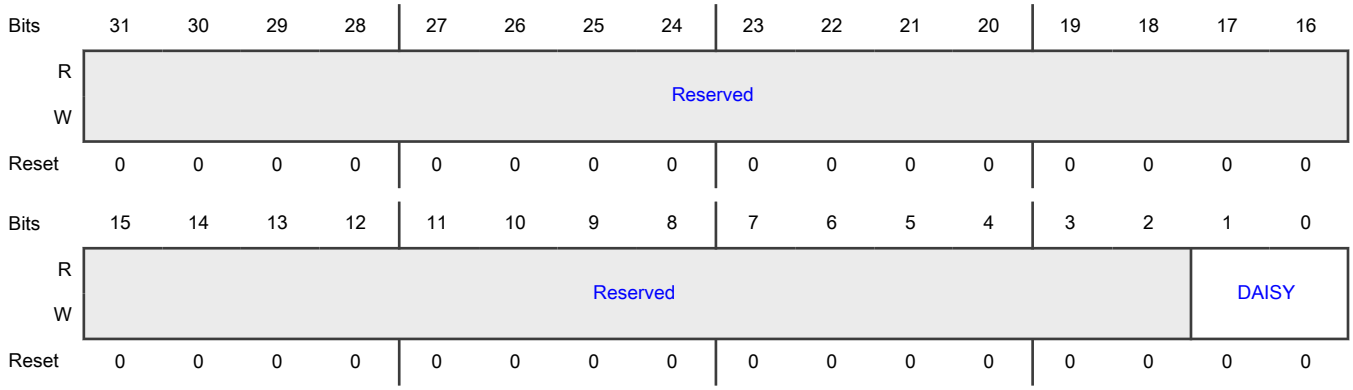
Offset

Register	Offset
LPI2C3_IPP_IND_LPI2C_SDA_SELECT_INPUT	5ECh

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c3, In Pin: ipp_ind_lpi2c_sda 00b - Selecting Pad: GPIO_EMC_B2_01 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_20 for Mode: ALT8 10b - Selecting Pad: GPIO_AD_19 for Mode: ALT9

17.4.1.378 LPI2C4_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C4_IPP_IND_LPI2C_SCL_SELECT_INPUT)

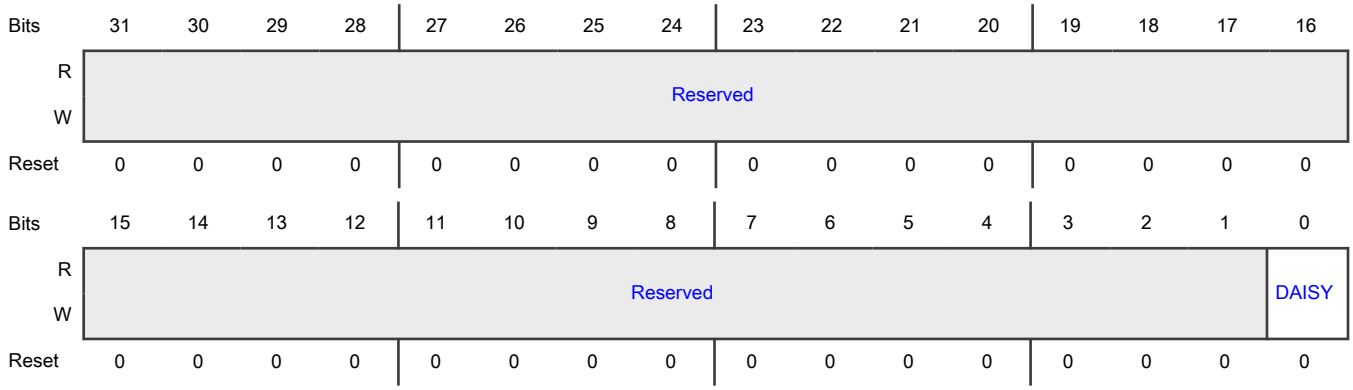
Offset

Register	Offset
LPI2C4_IPP_IND_LPI2C_SCL_SELECT_INPUT	5F0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c4, In Pin: ipp_ind_lpi2c_scl 0b - Selecting Pad: GPIO_AD_24 for Mode: ALT1 1b - Selecting Pad: GPIO_B2_10 for Mode: ALT6

17.4.1.379 LPI2C4_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C4_IPP_IND_LPI2C_SDA_SELECT_INPUT)

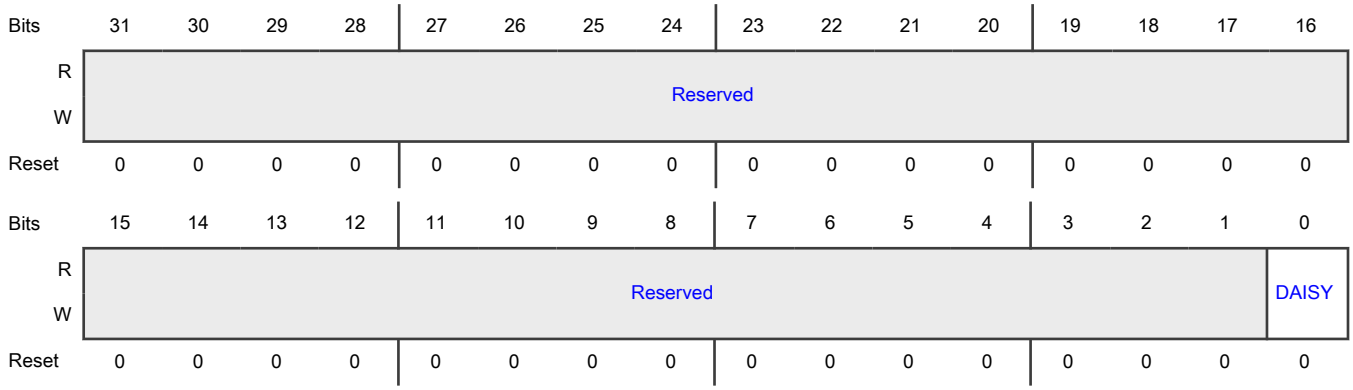
Offset

Register	Offset
LPI2C4_IPP_IND_LPI2C_SDA_SELECT_INPUT	5F4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c4, In Pin: ipp_ind_lpi2c_sda 0b - Selecting Pad: GPIO_AD_25 for Mode: ALT1 1b - Selecting Pad: GPIO_B2_11 for Mode: ALT6

17.4.1.380 LPI2C5_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C5_IPP_IND_LPI2C_SCL_SELECT_INPUT)

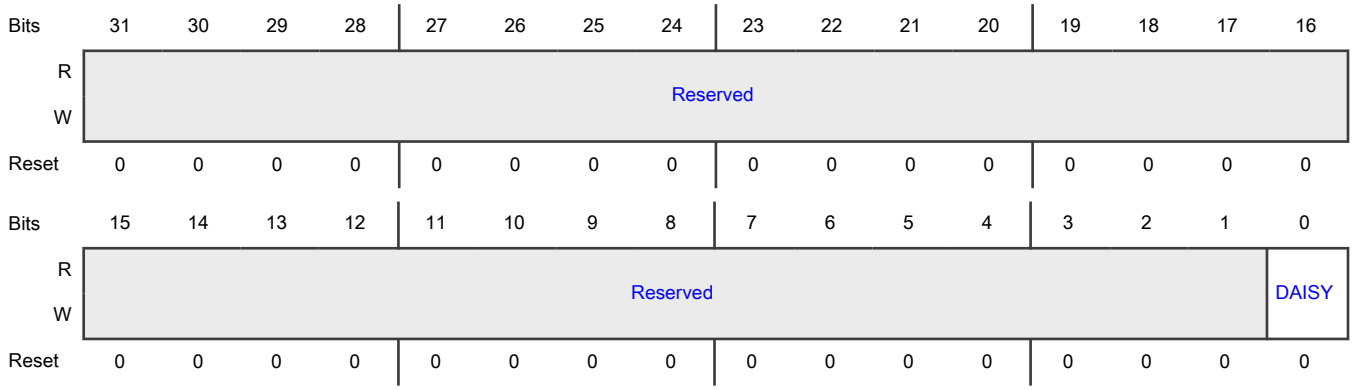
Offset

Register	Offset
LPI2C5_IPP_IND_LPI2C_SCL_SELECT_INPUT	5F8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c5, In Pin: ipp_ind_lpi2c_scl 0b - Selecting Pad: GPIO_AD_08 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_32 for Mode: ALT0

17.4.1.381 LPI2C5_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C5_IPP_IND_LPI2C_SDA_SELECT_INPUT)

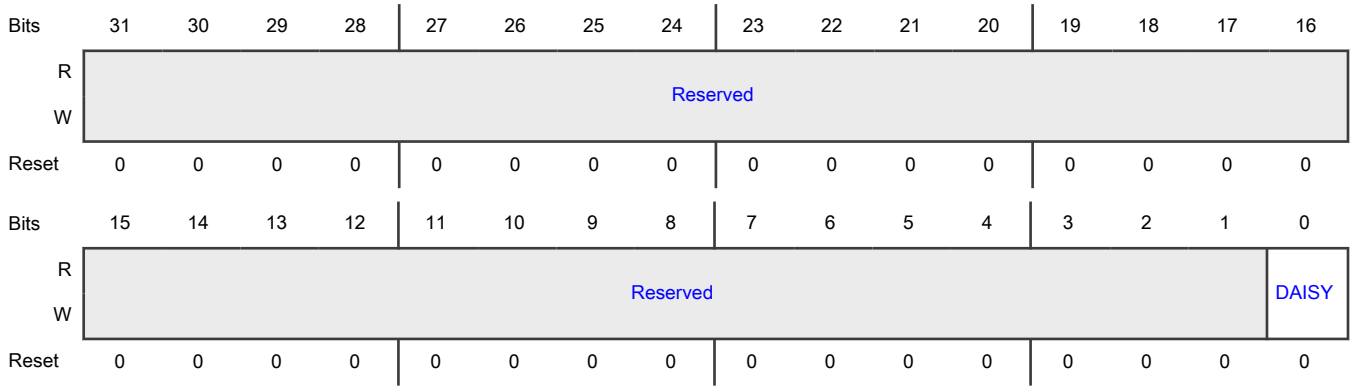
Offset

Register	Offset
LPI2C5_IPP_IND_LPI2C_SDA_SELECT_INPUT	5FCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c5, In Pin: ipp_ind_lpi2c_sda 0b - Selecting Pad: GPIO_AD_09 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_33 for Mode: ALT0

17.4.1.382 LPI2C6_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C6_IPP_IND_LPI2C_SCL_SELECT_INPUT)

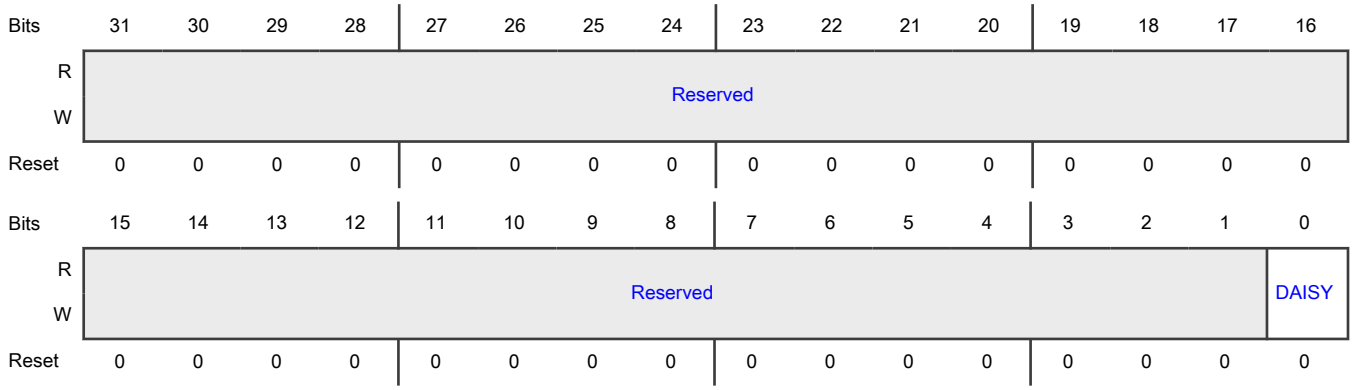
Offset

Register	Offset
LPI2C6_IPP_IND_LPI2C_SCL_SELECT_INPUT	600h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c6, In Pin: ipp_ind_lpi2c_scl 0b - Selecting Pad: GPIO_B1_02 for Mode: ALT2 1b - Selecting Pad: GPIO_B2_08 for Mode: ALT6

17.4.1.383 LPI2C6_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C6_IPP_IND_LPI2C_SDA_SELECT_INPUT)

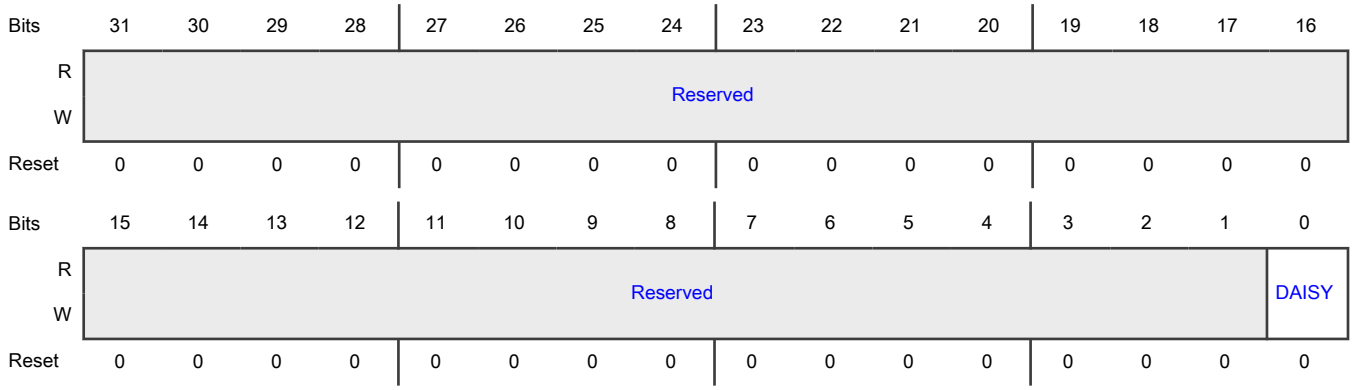
Offset

Register	Offset
LPI2C6_IPP_IND_LPI2C_SDA_SELECT_INPUT	604h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c6, In Pin: ipp_ind_lpi2c_sda 0b - Selecting Pad: GPIO_B1_03 for Mode: ALT2 1b - Selecting Pad: GPIO_B2_09 for Mode: ALT6

17.4.1.384 LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)

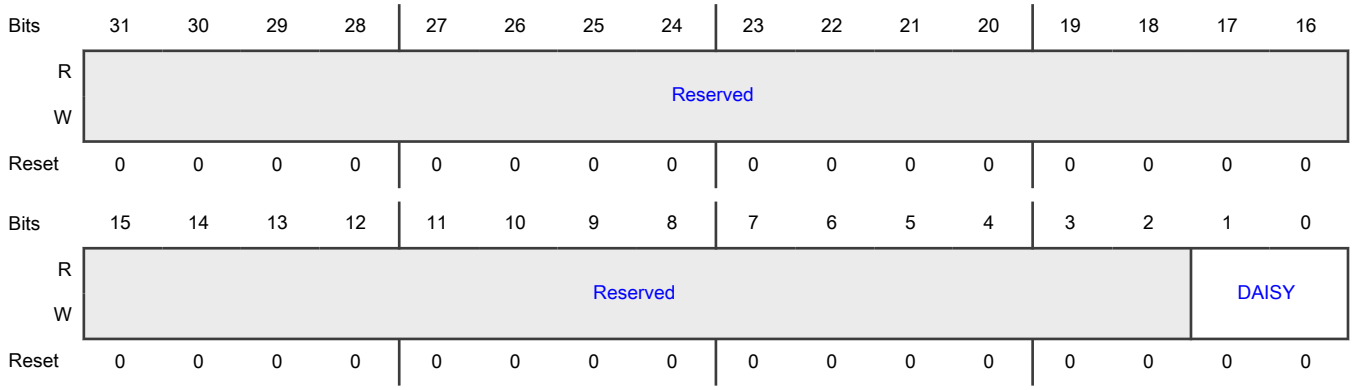
Offset

Register	Offset
LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_0	608h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi3, In Pin: ipp_ind_lpspi_pcs0 00b - Selecting Pad: GPIO_EMC_B2_05 for Mode: ALT8 01b - Selecting Pad: GPIO_AD_17 for Mode: ALT7 10b - Selecting Pad: GPIO_SD_B1_00 for Mode: ALT6

17.4.1.385 LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_1 DAISY Register (LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_1)

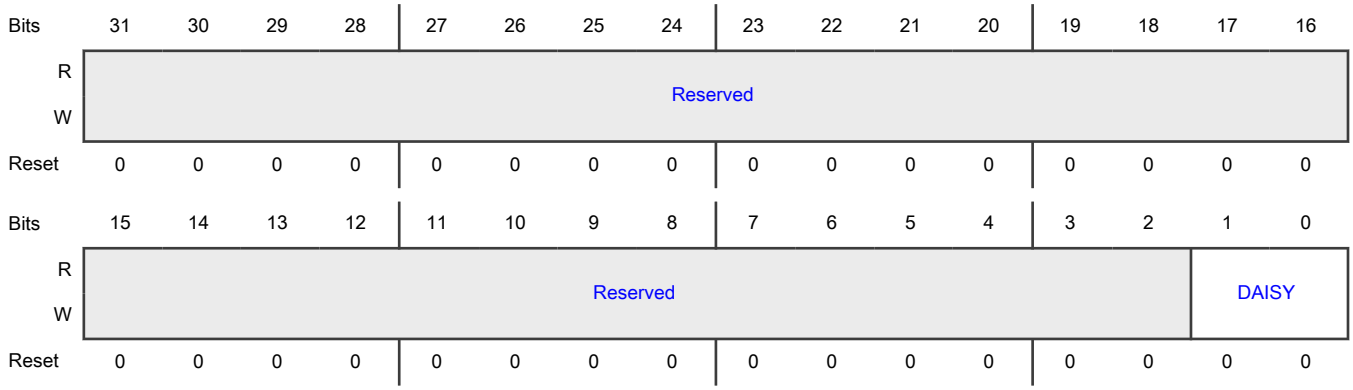
Offset

Register	Offset
LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_1	60Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi3, In Pin: ipp_ind_lpspi_pcs1 00b - Selecting Pad: GPIO_EMC_B2_10 for Mode: ALT8 01b - Selecting Pad: GPIO_AD_15 for Mode: ALT7 10b - Selecting Pad: GPIO_SD_B1_04 for Mode: ALT6

17.4.1.386 LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_2 DAISY Register (LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_2)

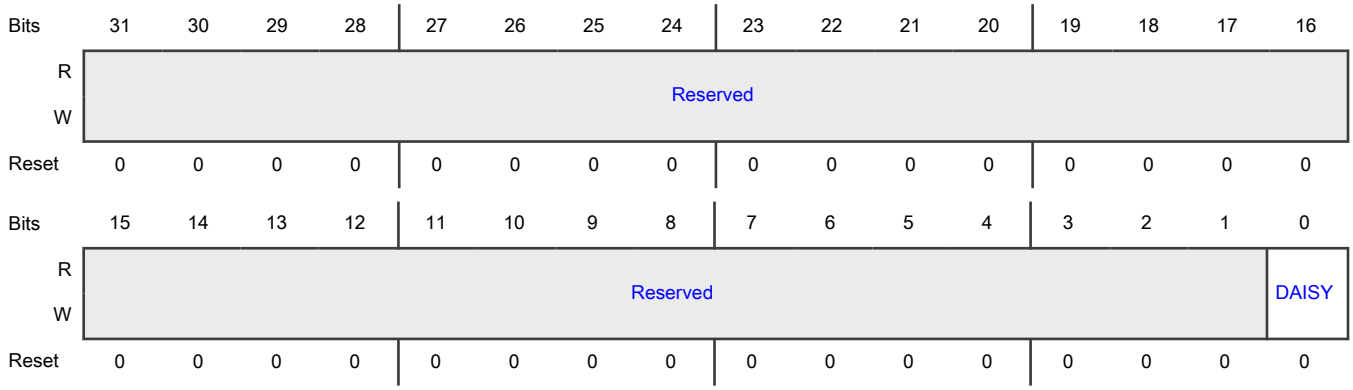
Offset

Register	Offset
LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_2	610h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi3, In Pin: ipp_ind_lpspi_pcs2 0b - Selecting Pad: GPIO_EMC_B2_09 for Mode: ALT8 1b - Selecting Pad: GPIO_SD_B1_05 for Mode: ALT6

17.4.1.387 LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_3 DAISY Register (LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_3)

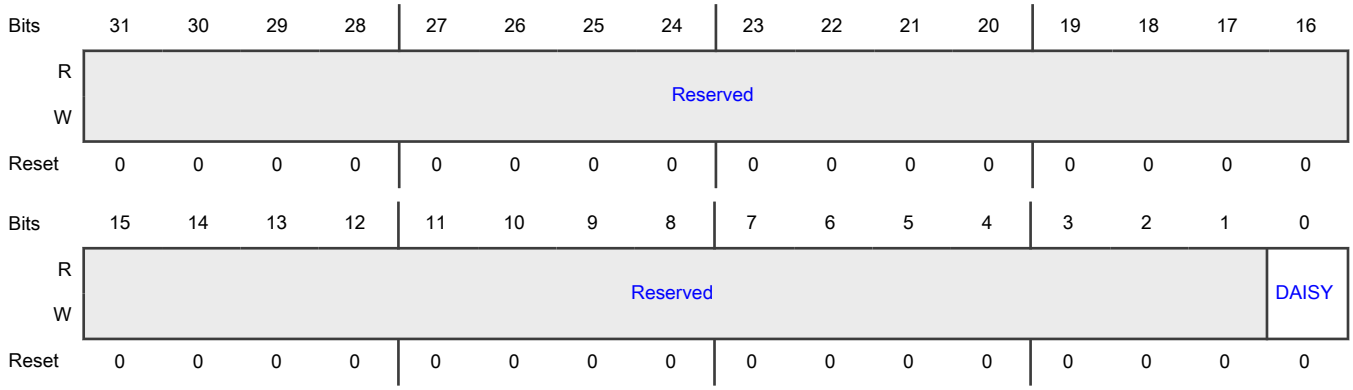
Offset

Register	Offset
LPSPI3_IPP_IND_LPSPI_PCS_SELECT_INPUT_3	614h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi3, In Pin: ipp_ind_lpspi_pcs3 0b - Selecting Pad: GPIO_EMC_B2_08 for Mode: ALT8 1b - Selecting Pad: GPIO_SD_B2_00 for Mode: ALT6

17.4.1.388 LPSPI3_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI3_IPP_IND_LPSPI_SCK_SELECT_INPUT)

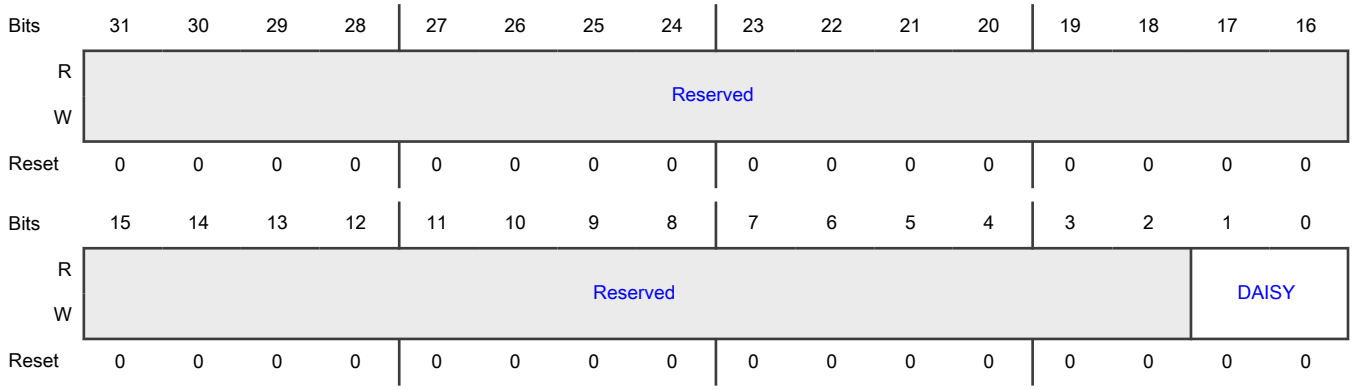
Offset

Register	Offset
LPSPI3_IPP_IND_LPSPI_SCK_SELECT_INPUT	618h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi3, In Pin: ipp_ind_lpspi_sck 00b - Selecting Pad: GPIO_EMC_B2_04 for Mode: ALT8 01b - Selecting Pad: GPIO_AD_16 for Mode: ALT7 10b - Selecting Pad: GPIO_SD_B1_01 for Mode: ALT6

17.4.1.389 LPSPI3_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI3_IPP_IND_LPSPI_SDI_SELECT_INPUT)

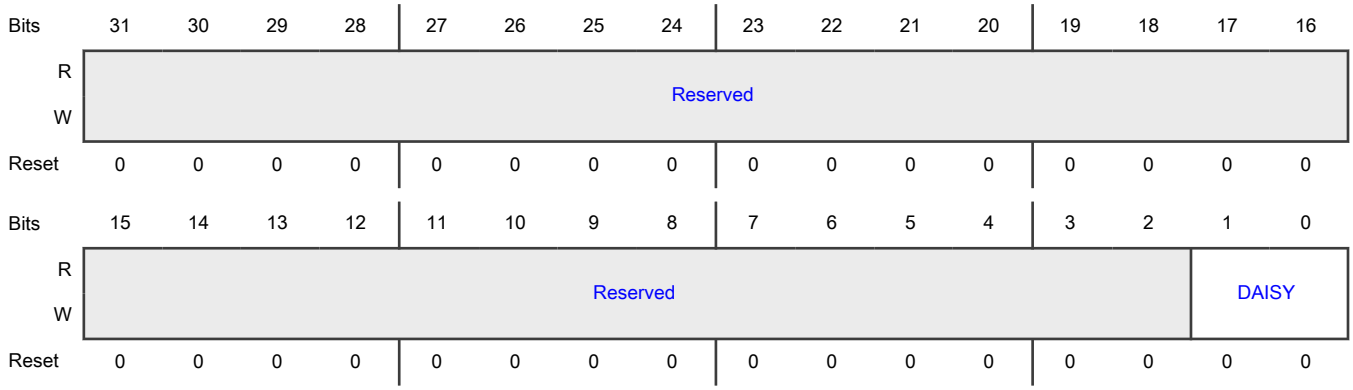
Offset

Register	Offset
LPSPI3_IPP_IND_LPSPI_SDI_SELECT_INPUT	61Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi3, In Pin: ipp_ind_lpspi_sdi 00b - Selecting Pad: GPIO_EMC_B2_07 for Mode: ALT8 01b - Selecting Pad: GPIO_AD_19 for Mode: ALT7 10b - Selecting Pad: GPIO_SD_B1_03 for Mode: ALT6

17.4.1.390 LPSPI3_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI3_IPP_IND_LPSPI_SDO_SELECT_INPUT)

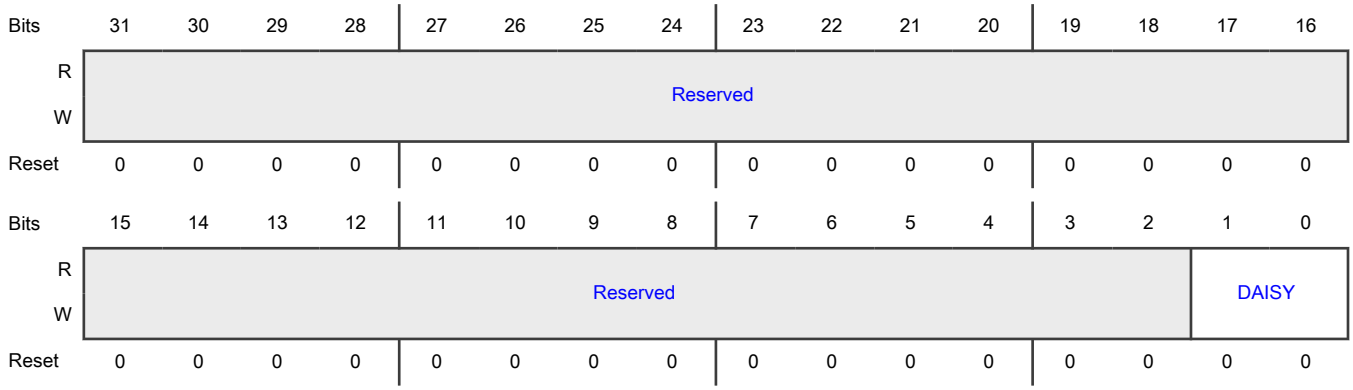
Offset

Register	Offset
LPSPI3_IPP_IND_LPSPI_SDO_SELECT_INPUT	620h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi3, In Pin: ipp_ind_lpspi_sdo 00b - Selecting Pad: GPIO_EMC_B2_06 for Mode: ALT8 01b - Selecting Pad: GPIO_AD_18 for Mode: ALT7 10b - Selecting Pad: GPIO_SD_B1_02 for Mode: ALT6

17.4.1.391 LPSPI4_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI4_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)

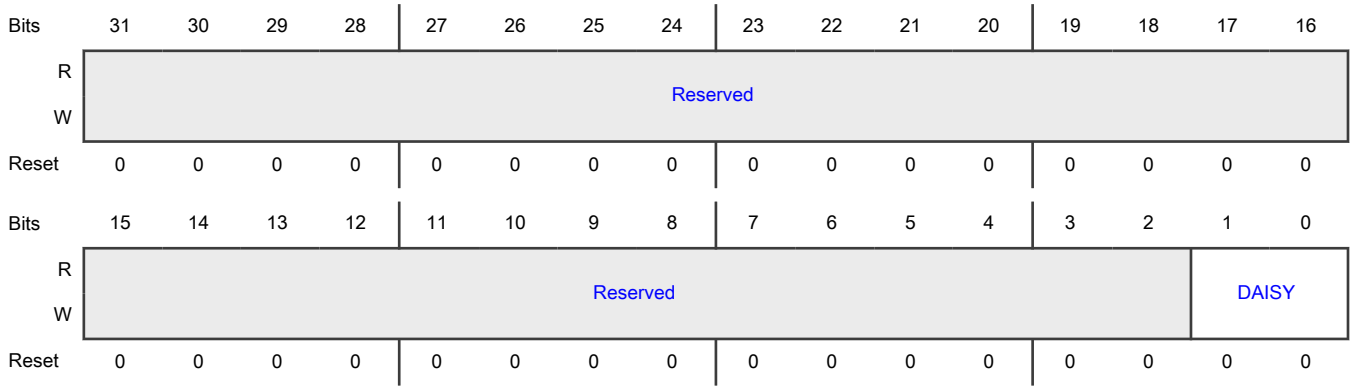
Offset

Register	Offset
LPSPI4_IPP_IND_LPSPI_PCS_SELECT_INPUT_0	624h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi4, In Pin: ipp_ind_lpspi_pcs0 00b - Selecting Pad: GPIO_EMC_B1_25 for Mode: ALT3 01b - Selecting Pad: GPIO_SD_B2_09 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_13 for Mode: ALT9

17.4.1.392 LPSPI4_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI4_IPP_IND_LPSPI_SCK_SELECT_INPUT)

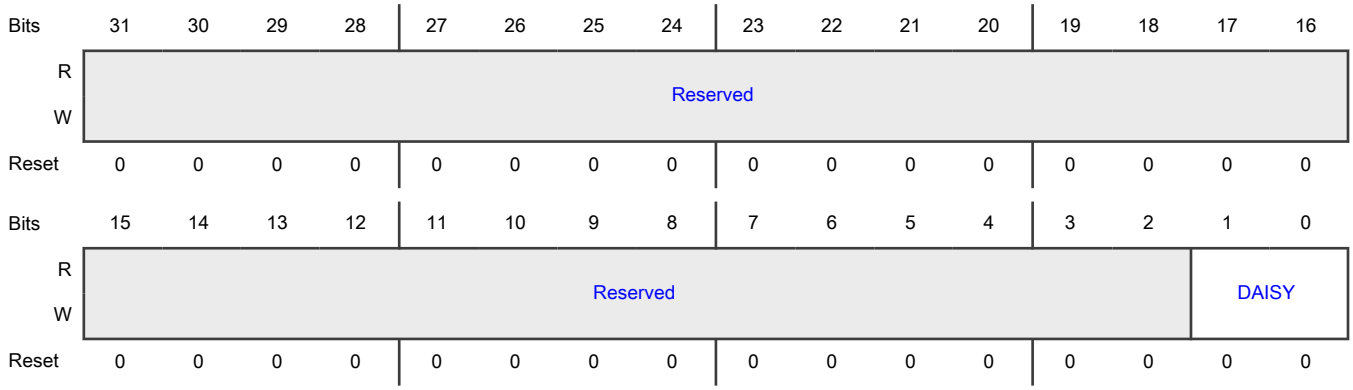
Offset

Register	Offset
LPSPI4_IPP_IND_LPSPI_SCK_SELECT_INPUT	628h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi4, In Pin: ipp_ind_lpspi_sck 00b - Selecting Pad: GPIO_EMC_B1_22 for Mode: ALT3 01b - Selecting Pad: GPIO_SD_B2_08 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_10 for Mode: ALT9

17.4.1.393 LPSPI4_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI4_IPP_IND_LPSPI_SDI_SELECT_INPUT)

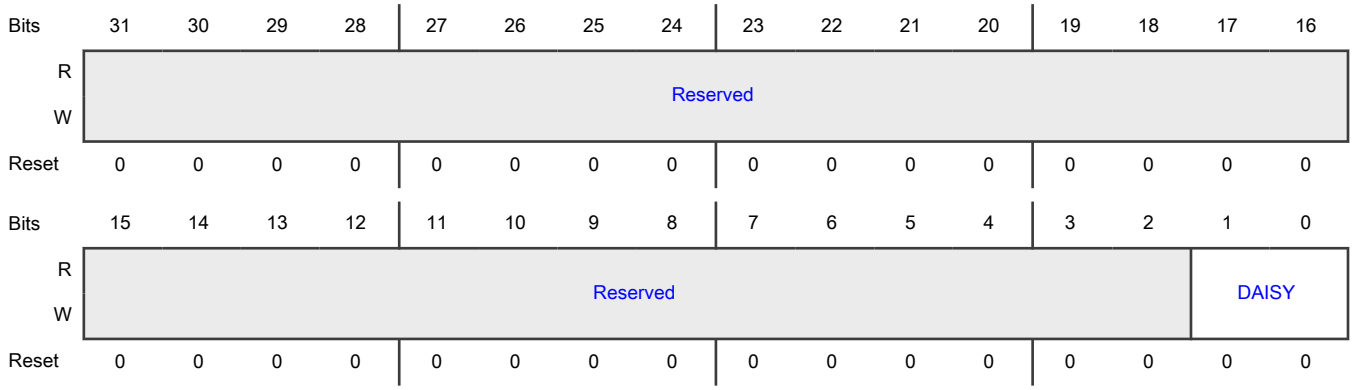
Offset

Register	Offset
LPSPI4_IPP_IND_LPSPI_SDI_SELECT_INPUT	62Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi4, In Pin: ipp_ind_lpspi_sdi 00b - Selecting Pad: GPIO_EMC_B1_23 for Mode: ALT3 01b - Selecting Pad: GPIO_SD_B2_11 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_11 for Mode: ALT9

17.4.1.394 LPSPI4_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI4_IPP_IND_LPSPI_SDO_SELECT_INPUT)

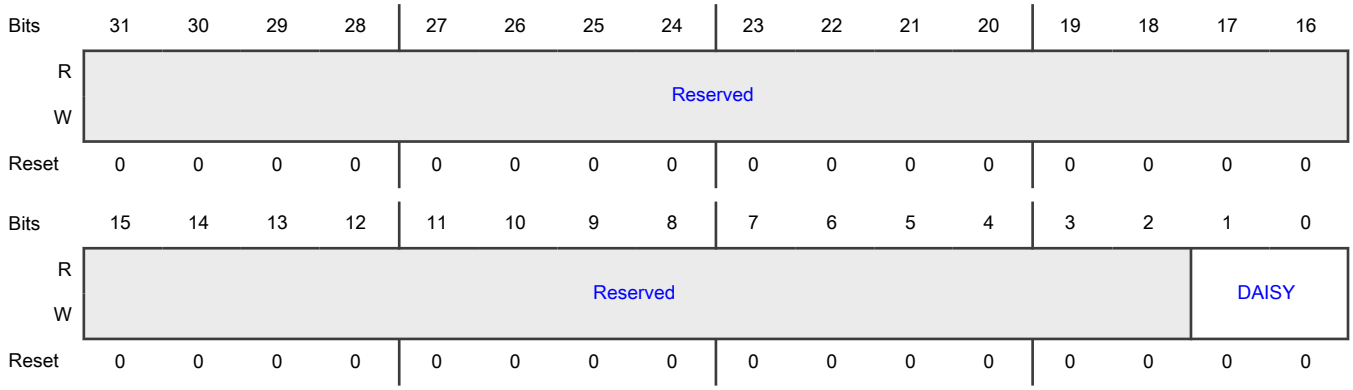
Offset

Register	Offset
LPSPI4_IPP_IND_LPSPI_SDO_SELECT_INPUT	630h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi4, In Pin: ipp_ind_lpspi_sdo 00b - Selecting Pad: GPIO_EMC_B1_24 for Mode: ALT3 01b - Selecting Pad: GPIO_SD_B2_10 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_12 for Mode: ALT9

17.4.1.395 LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)

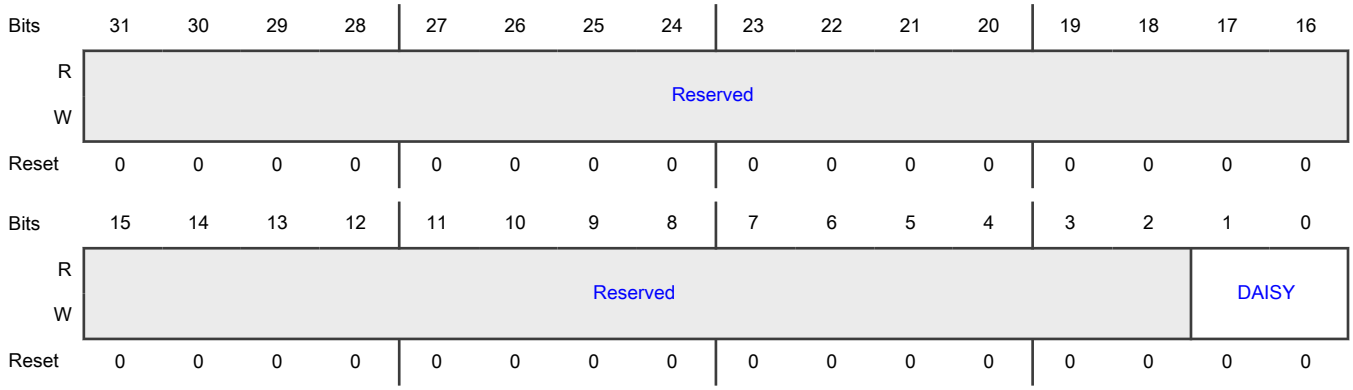
Offset

Register	Offset
LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_0	634h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi5, In Pin: ipp_ind_lpspi_pcs0 00b - Selecting Pad: GPIO_EMC_B1_34 for Mode: ALT10 01b - Selecting Pad: GPIO_EMC_B2_01 for Mode: ALT8 10b - Selecting Pad: GPIO_AD_29 for Mode: ALT0

17.4.1.396 LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_1 DAISY Register (LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_1)

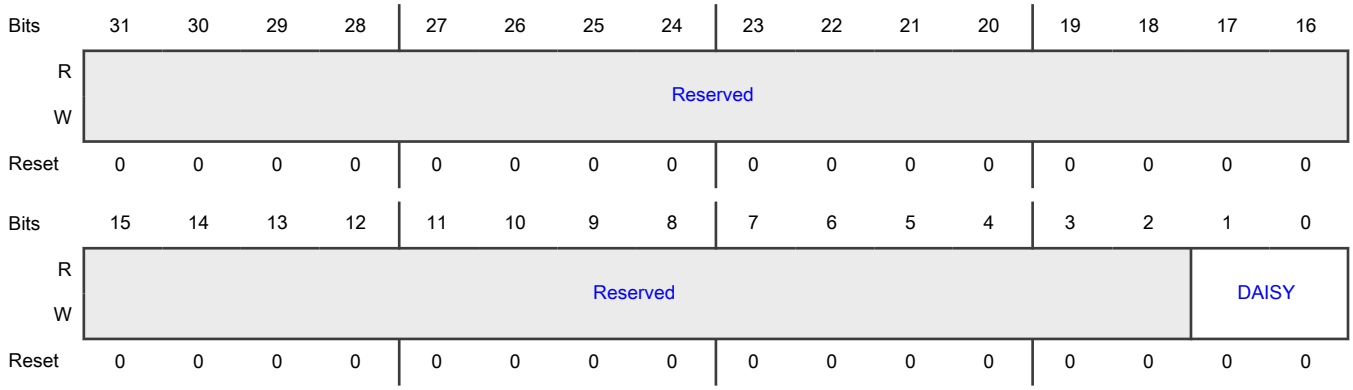
Offset

Register	Offset
LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_1	638h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi5, In Pin: ipp_ind_lpspi_pcs1 00b - Selecting Pad: GPIO_EMC_B1_35 for Mode: ALT10 01b - Selecting Pad: GPIO_EMC_B2_13 for Mode: ALT4 10b - Selecting Pad: GPIO_AD_27 for Mode: ALT2

17.4.1.397 LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_2 DAISY Register (LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_2)

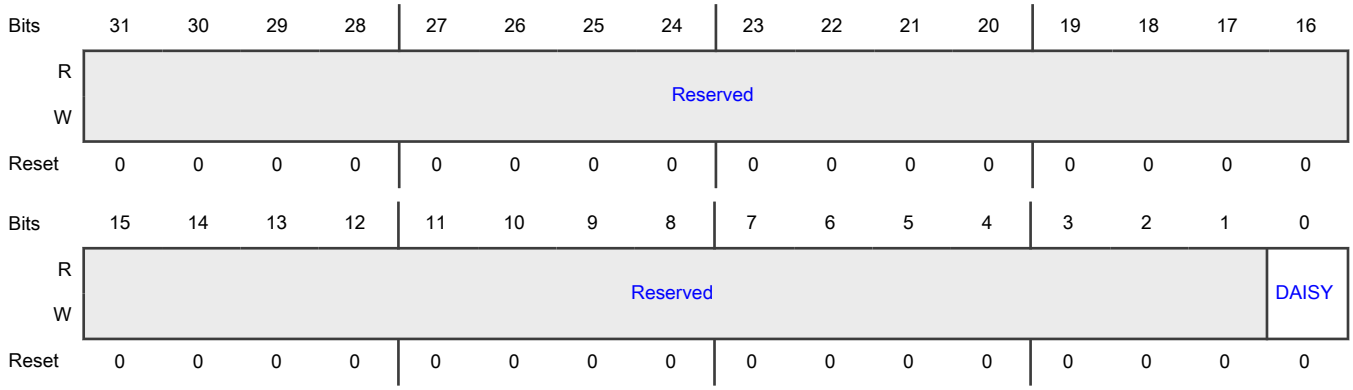
Offset

Register	Offset
LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_2	63Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi5, In Pin: ipp_ind_lpspi_pcs2 0b - Selecting Pad: GPIO_EMC_B2_12 for Mode: ALT4 1b - Selecting Pad: GPIO_AD_26 for Mode: ALT2

17.4.1.398 LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_3 DAISY Register (LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_3)

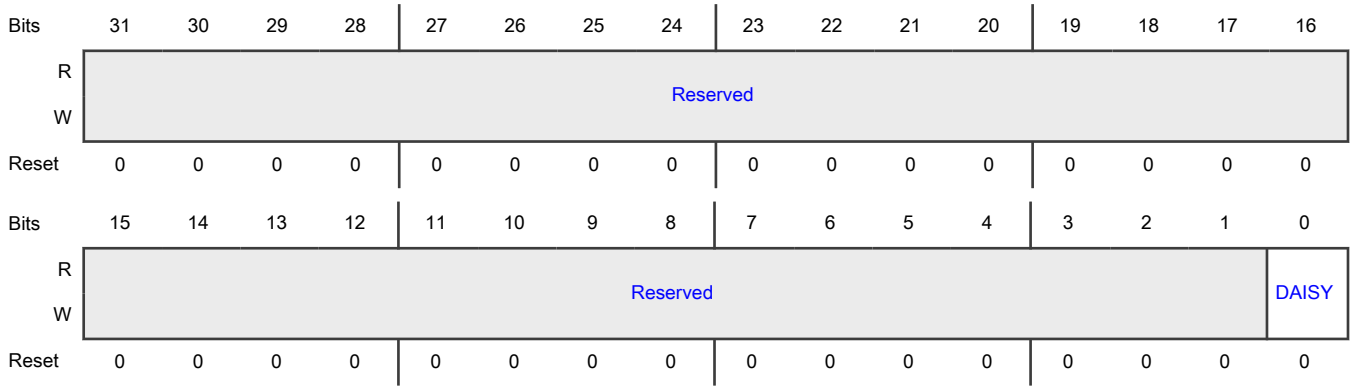
Offset

Register	Offset
LPSPI5_IPP_IND_LPSPI_PCS_SELECT_INPUT_3	640h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi5, In Pin: ipp_ind_lpspi_pcs3 0b - Selecting Pad: GPIO_EMC_B2_11 for Mode: ALT4 1b - Selecting Pad: GPIO_AD_25 for Mode: ALT2

17.4.1.399 LPSPI5_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI5_IPP_IND_LPSPI_SCK_SELECT_INPUT)

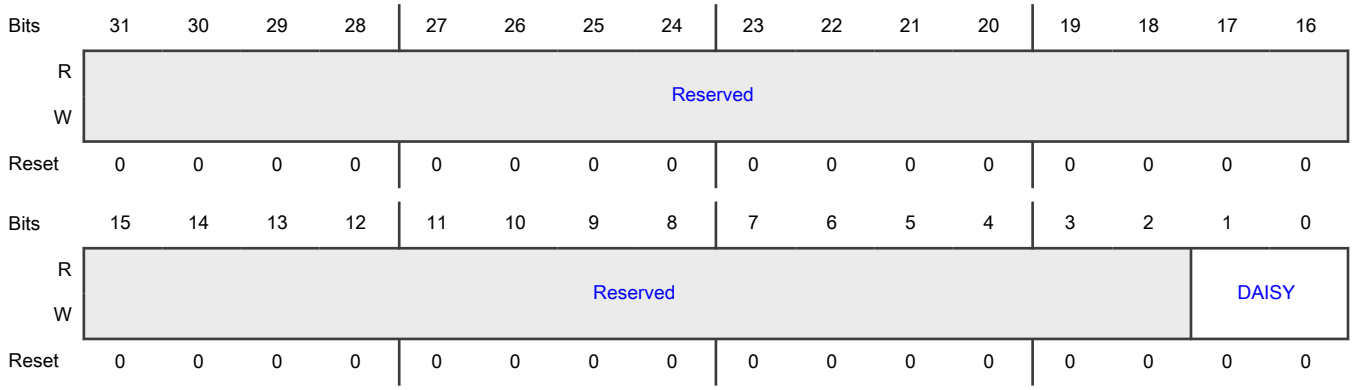
Offset

Register	Offset
LPSPI5_IPP_IND_LPSPI_SCK_SELECT_INPUT	644h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi5, In Pin: ipp_ind_lpspi_sck 00b - Selecting Pad: GPIO_EMC_B1_31 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_00 for Mode: ALT8 10b - Selecting Pad: GPIO_AD_28 for Mode: ALT0

17.4.1.400 LPSPI5_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI5_IPP_IND_LPSPI_SDI_SELECT_INPUT)

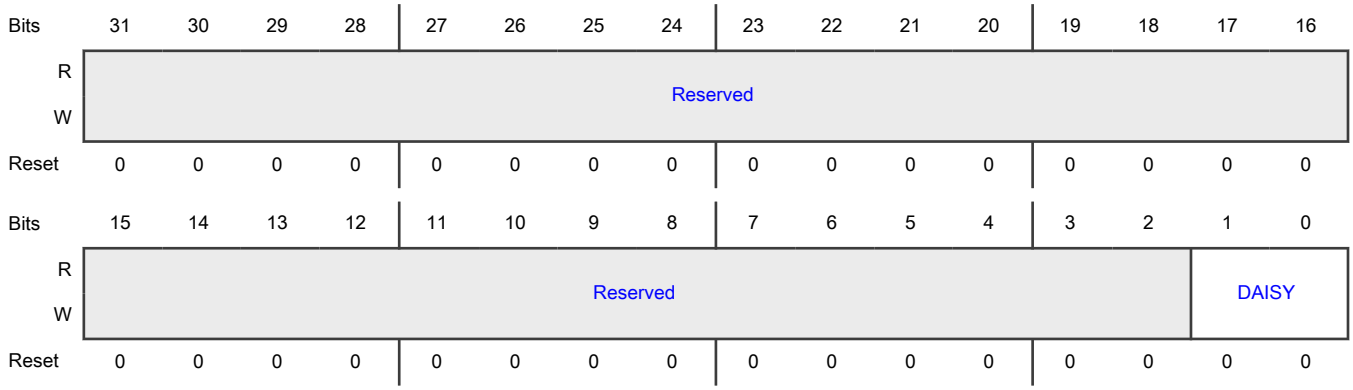
Offset

Register	Offset
LPSPI5_IPP_IND_LPSPI_SDI_SELECT_INPUT	648h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi5, In Pin: ipp_ind_lpspi_sdi 00b - Selecting Pad: GPIO_EMC_B1_33 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_03 for Mode: ALT8 10b - Selecting Pad: GPIO_AD_31 for Mode: ALT0

17.4.1.401 LPSPI5_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI5_IPP_IND_LPSPI_SDO_SELECT_INPUT)

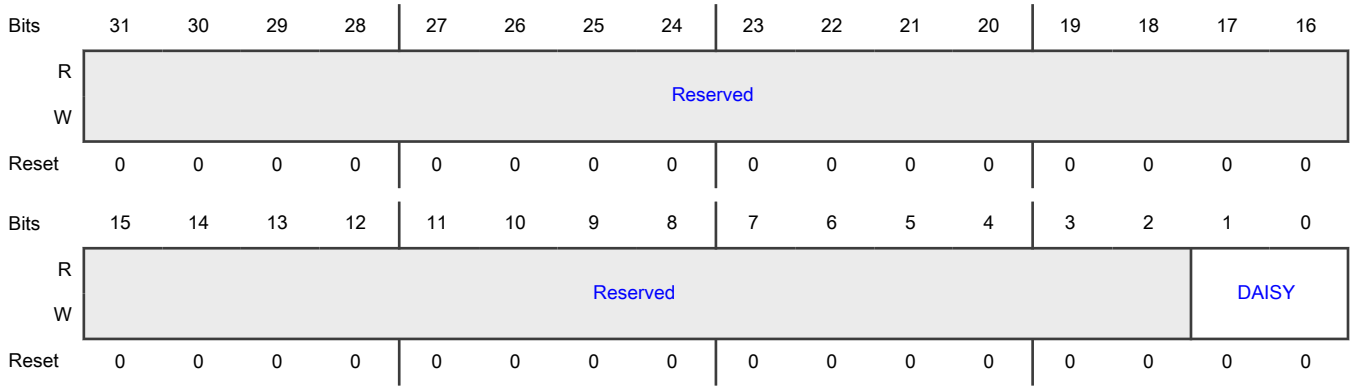
Offset

Register	Offset
LPSPI5_IPP_IND_LPSPI_SDO_SELECT_INPUT	64Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi5, In Pin: ipp_ind_lpspi_sdo 00b - Selecting Pad: GPIO_EMC_B1_32 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_02 for Mode: ALT8 10b - Selecting Pad: GPIO_AD_30 for Mode: ALT0

17.4.1.402 LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)

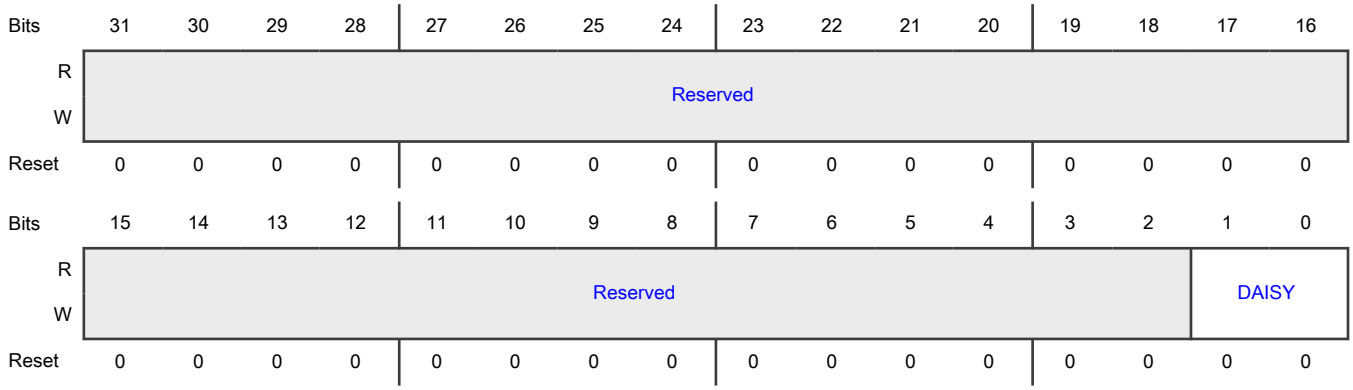
Offset

Register	Offset
LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_0	650h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi6, In Pin: ipp_ind_lpspi_pcs0 00b - Selecting Pad: GPIO_EMC_B1_21 for Mode: ALT3 01b - Selecting Pad: GPIO_EMC_B1_29 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_12 for Mode: ALT9

17.4.1.403 LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_1 DAISY Register (LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_1)

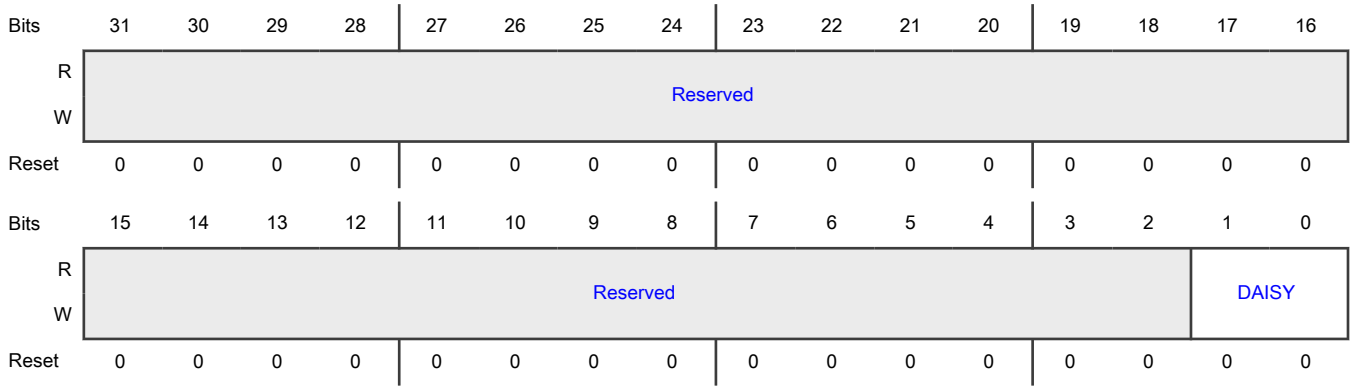
Offset

Register	Offset
LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_1	654h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi6, In Pin: ipp_ind_lpspi_pcs1 00b - Selecting Pad: GPIO_EMC_B1_17 for Mode: ALT10 01b - Selecting Pad: GPIO_EMC_B1_30 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_09 for Mode: ALT9

17.4.1.404 LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_2 DAISY Register (LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_2)

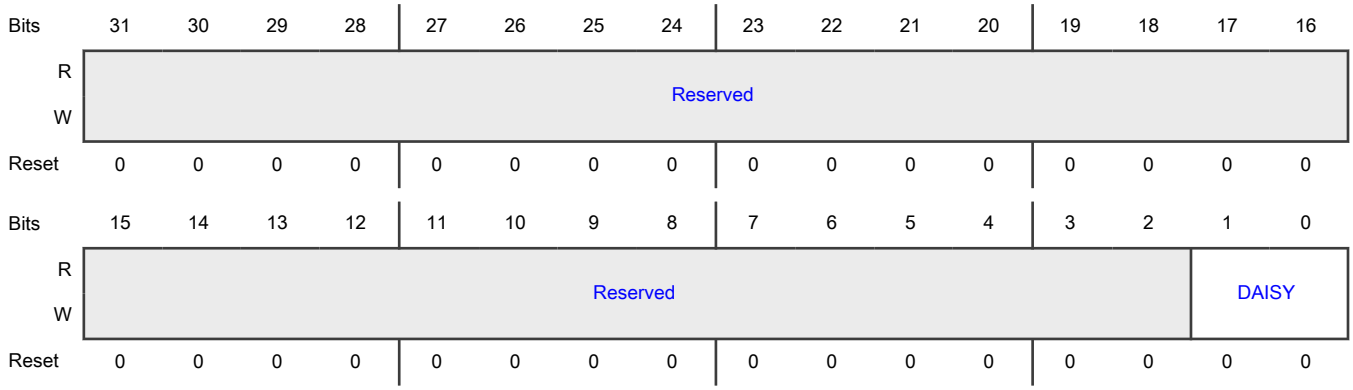
Offset

Register	Offset
LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_2	658h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi6, In Pin: ipp_ind_lpspi_pcs2 00b - Selecting Pad: GPIO_EMC_B1_16 for Mode: ALT10 01b - Selecting Pad: GPIO_EMC_B1_31 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_10 for Mode: ALT9

17.4.1.405 LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_3 DAISY Register (LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_3)

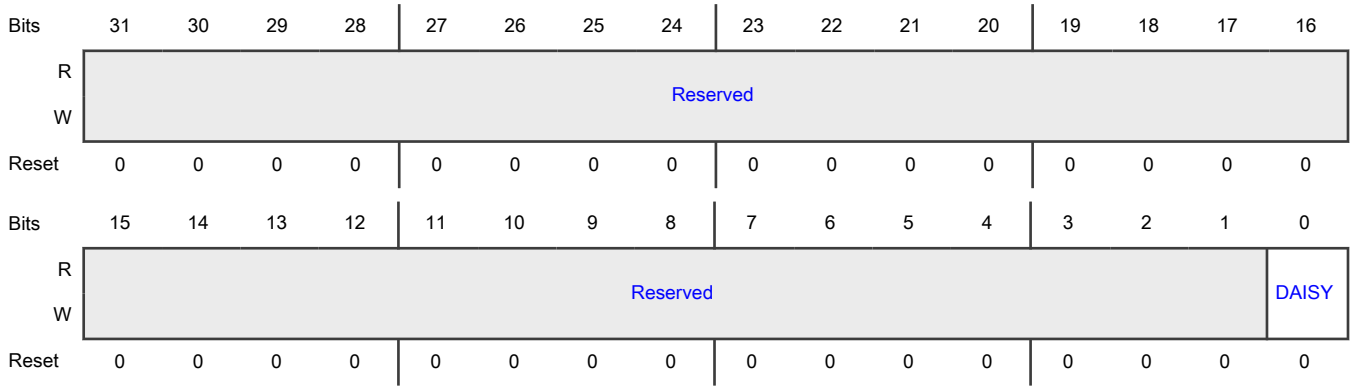
Offset

Register	Offset
LPSPI6_IPP_IND_LPSPI_PCS_SELECT_INPUT_3	65Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi6, In Pin: ipp_ind_lpspi_pcs3 0b - Selecting Pad: GPIO_EMC_B1_32 for Mode: ALT10 1b - Selecting Pad: GPIO_B1_11 for Mode: ALT9

17.4.1.406 LPSPI6_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI6_IPP_IND_LPSPI_SCK_SELECT_INPUT)

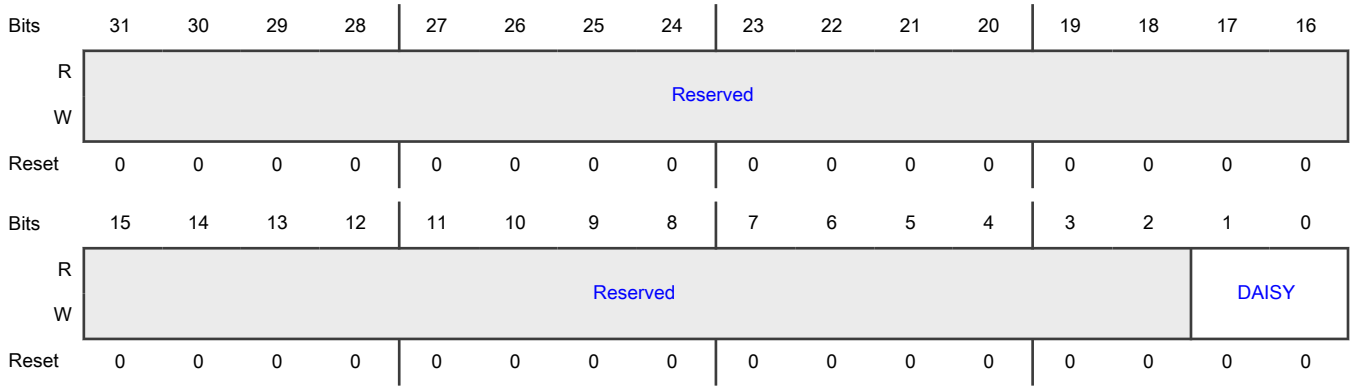
Offset

Register	Offset
LPSPI6_IPP_IND_LPSPI_SCK_SELECT_INPUT	660h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi6, In Pin: ipp_ind_lpspi_sck 00b - Selecting Pad: GPIO_EMC_B1_18 for Mode: ALT3 01b - Selecting Pad: GPIO_EMC_B1_26 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_13 for Mode: ALT9

17.4.1.407 LPSPI6_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI6_IPP_IND_LPSPI_SDI_SELECT_INPUT)

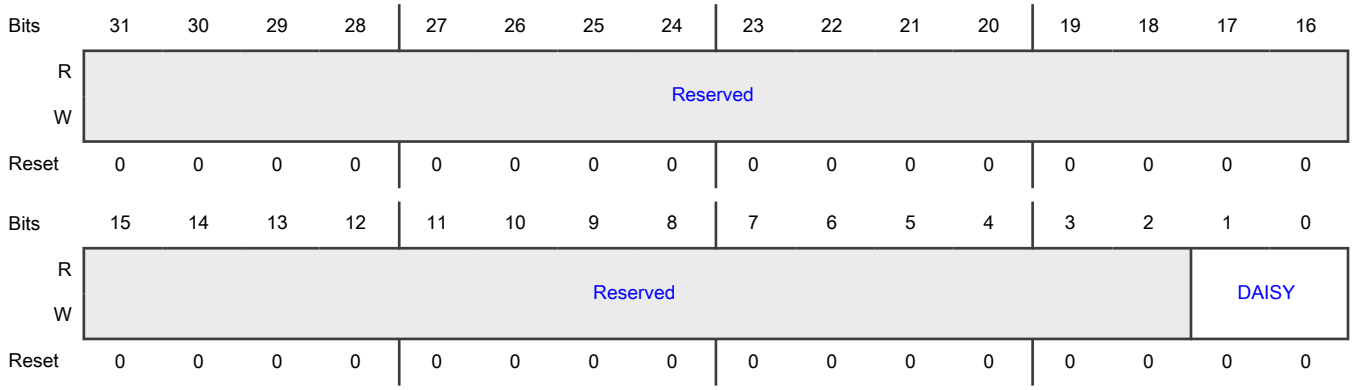
Offset

Register	Offset
LPSPI6_IPP_IND_LPSPI_SDI_SELECT_INPUT	664h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi6, In Pin: ipp_ind_lpspi_sdi 00b - Selecting Pad: GPIO_EMC_B1_19 for Mode: ALT3 01b - Selecting Pad: GPIO_EMC_B1_27 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_07 for Mode: ALT9 11b - Selecting Pad: GPIO_B2_00 for Mode: ALT9

17.4.1.408 LPSPI6_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI6_IPP_IND_LPSPI_SDO_SELECT_INPUT)

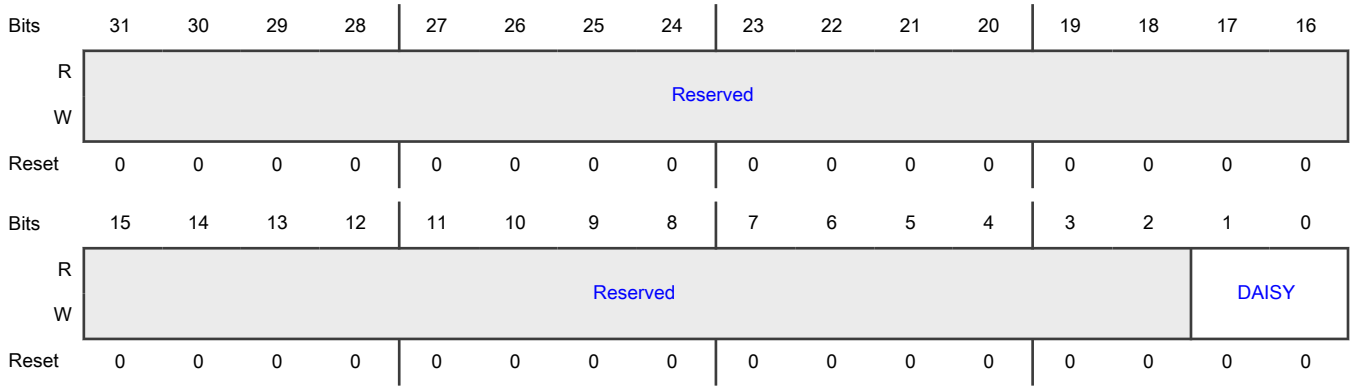
Offset

Register	Offset
LPSPI6_IPP_IND_LPSPI_SDO_SELECT_INPUT	668h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi6, In Pin: ipp_ind_lpspi_sdo 00b - Selecting Pad: GPIO_EMC_B1_20 for Mode: ALT3 01b - Selecting Pad: GPIO_EMC_B1_28 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_08 for Mode: ALT9 11b - Selecting Pad: GPIO_B2_01 for Mode: ALT9

17.4.1.409 LPUART10_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART10_IPP_IND_LPUART_RXD_SELECT_INPUT)

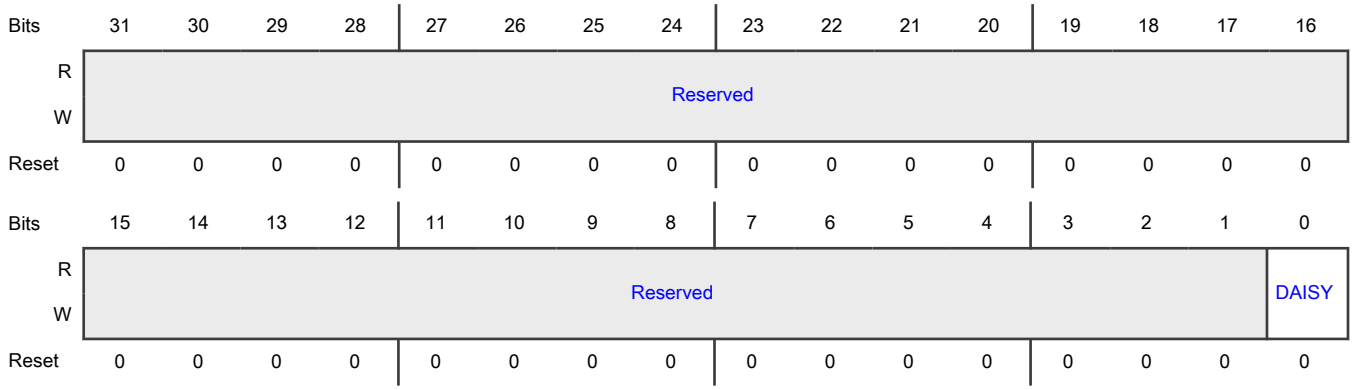
Offset

Register	Offset
LPUART10_IPP_IND_LPUART_RXD_SELECT_INPUT	66Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart10, In Pin: ipp_ind_lpuart_rxd 0b - Selecting Pad: GPIO_AD_16 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_33 for Mode: ALT8

17.4.1.410 LPUART10_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART10_IPP_IND_LPUART_TXD_SELECT_INPUT)

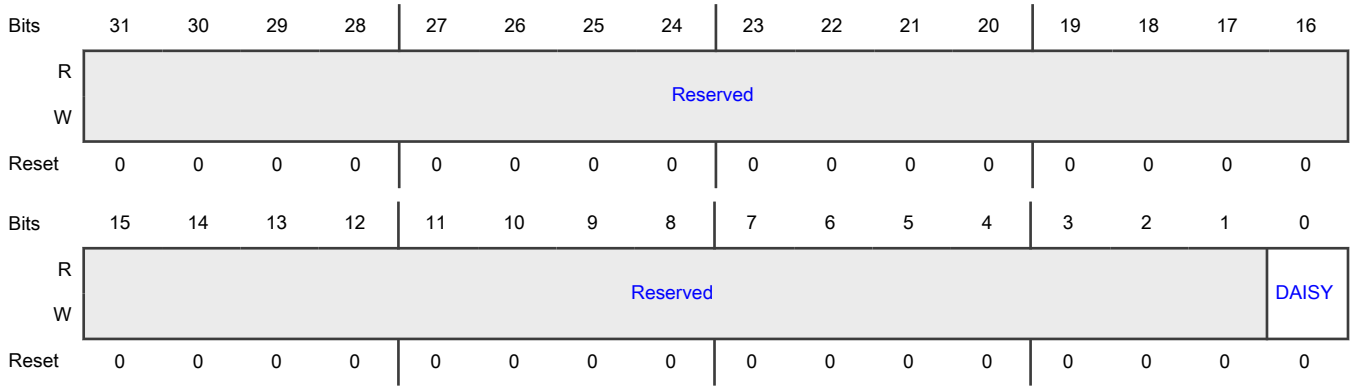
Offset

Register	Offset
LPUART10_IPP_IND_LPUART_TXD_SELECT_INPUT	670h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart10, In Pin: ipp_ind_lpuart_txd 0b - Selecting Pad: GPIO_AD_15 for Mode: ALT1 1b - Selecting Pad: GPIO_AD_32 for Mode: ALT8

17.4.1.411 LPUART11_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART11_IPP_IND_LPUART_RXD_SELECT_INPUT)

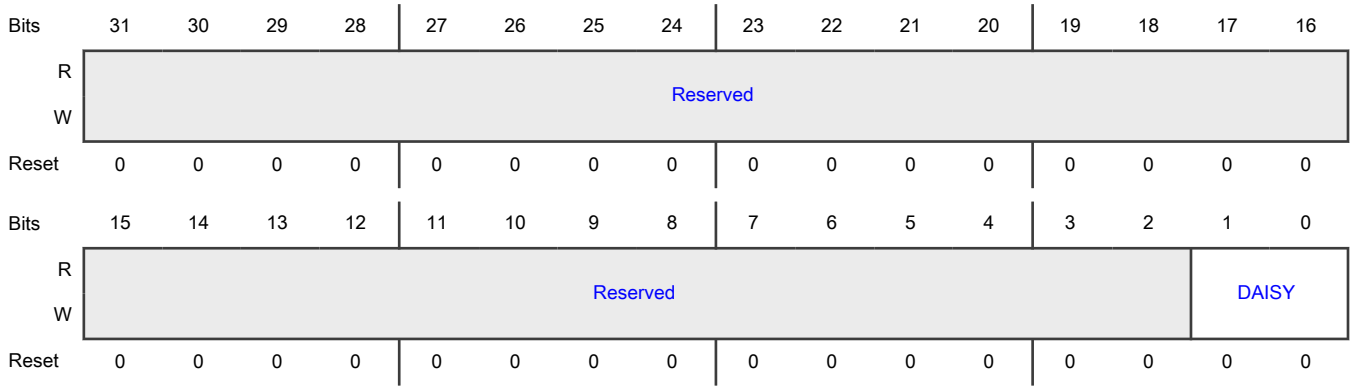
Offset

Register	Offset
LPUART11_IPP_IND_LPUART_RXD_SELECT_INPUT	674h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart11, In Pin: ipp_ind_lpuart_rxd 00b - Selecting Pad: GPIO_EMC_B2_14 for Mode: ALT2 01b - Selecting Pad: GPIO_B1_03 for Mode: ALT9 10b - Selecting Pad: GPIO_B2_07 for Mode: ALT9

17.4.1.412 LPUART11_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART11_IPP_IND_LPUART_TXD_SELECT_INPUT)

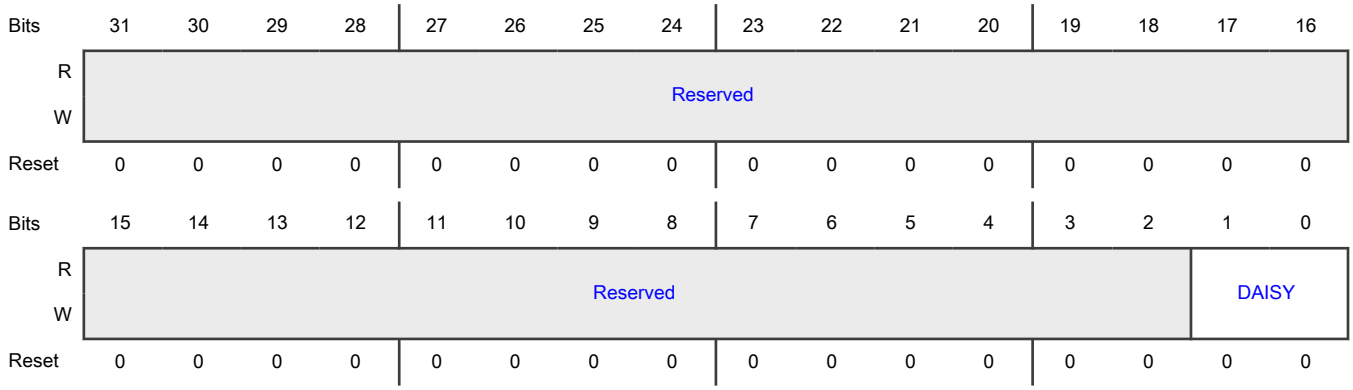
Offset

Register	Offset
LPUART11_IPP_IND_LPUART_TXD_SELECT_INPUT	678h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart11, In Pin: ipp_ind_lpuart_txd 00b - Selecting Pad: GPIO_EMC_B2_13 for Mode: ALT2 01b - Selecting Pad: GPIO_B1_02 for Mode: ALT9 10b - Selecting Pad: GPIO_B2_06 for Mode: ALT9

17.4.1.413 LPUART3_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART3_IPP_IND_LPUART_CTS_N_SELECT_INPUT)

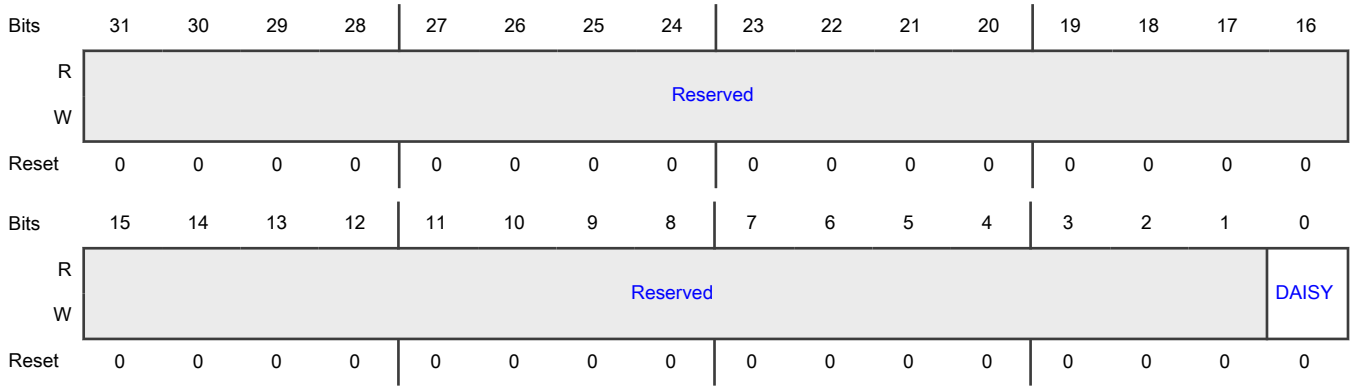
Offset

Register	Offset
LPUART3_IPP_IND_LPUART_CTS_N_SELECT_INPUT	67Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart3, In Pin: ipp_ind_lpuart_cts_n 0b - Selecting Pad: GPIO_EMC_B1_00 for Mode: ALT3 1b - Selecting Pad: GPIO_AD_15 for Mode: ALT6

17.4.1.414 LPUART3_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART3_IPP_IND_LPUART_RXD_SELECT_INPUT)

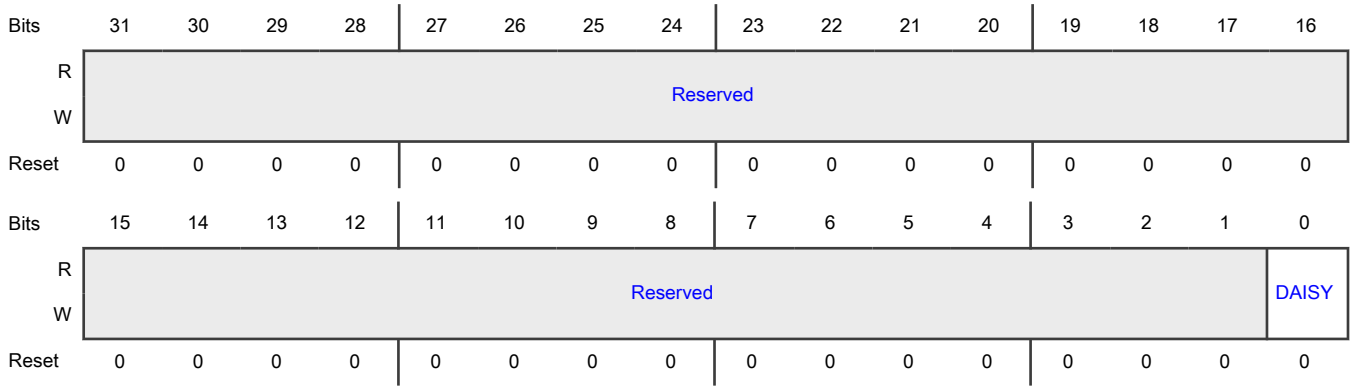
Offset

Register	Offset
LPUART3_IPP_IND_LPUART_RXD_SELECT_INPUT	680h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart3, In Pin: ipp_ind_lpuart_rxd 0b - Selecting Pad: GPIO_EMC_B1_02 for Mode: ALT3 1b - Selecting Pad: GPIO_AD_14 for Mode: ALT6

17.4.1.415 LPUART3_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART3_IPP_IND_LPUART_TXD_SELECT_INPUT)

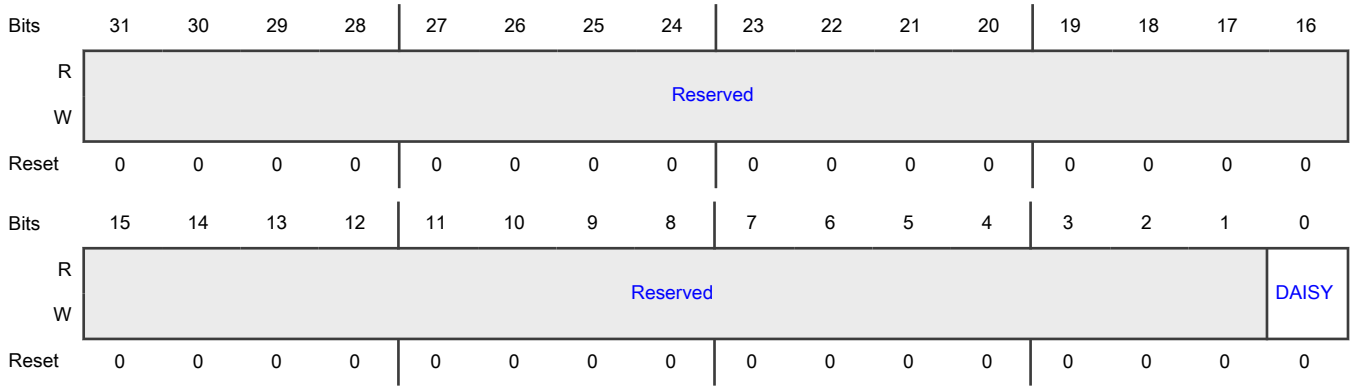
Offset

Register	Offset
LPUART3_IPP_IND_LPUART_TXD_SELECT_INPUT	684h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart3, In Pin: ipp_ind_lpuart_txd 0b - Selecting Pad: GPIO_EMC_B1_03 for Mode: ALT3 1b - Selecting Pad: GPIO_AD_13 for Mode: ALT6

17.4.1.416 LPUART4_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART4_IPP_IND_LPUART_CTS_N_SELECT_INPUT)

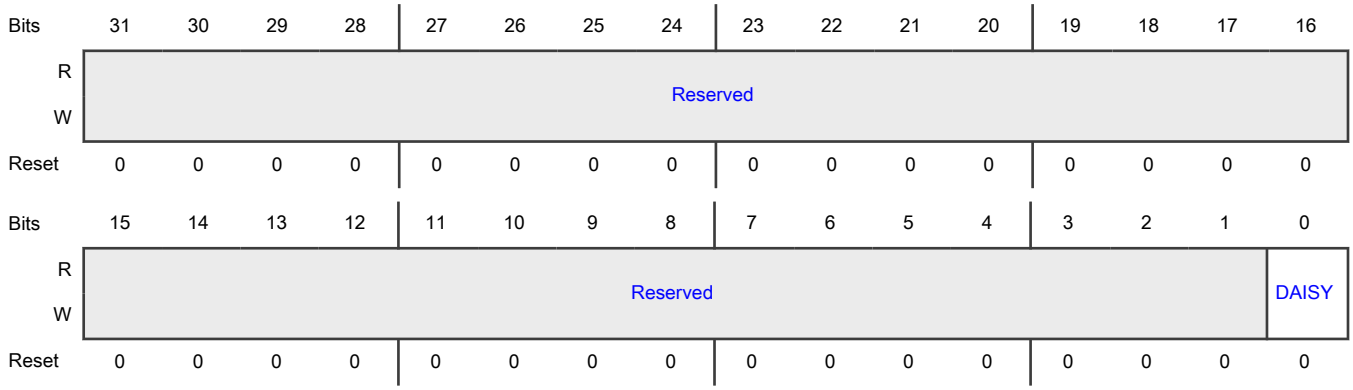
Offset

Register	Offset
LPUART4_IPP_IND_LPUART_CTS_N_SELECT_INPUT	688h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart4, In Pin: ipp_ind_lpuart_cts_n 0b - Selecting Pad: GPIO_EMC_B1_14 for Mode: ALT10 1b - Selecting Pad: GPIO_EMC_B1_21 for Mode: ALT9

17.4.1.417 LPUART5_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_CTS_N_SELECT_INPUT)

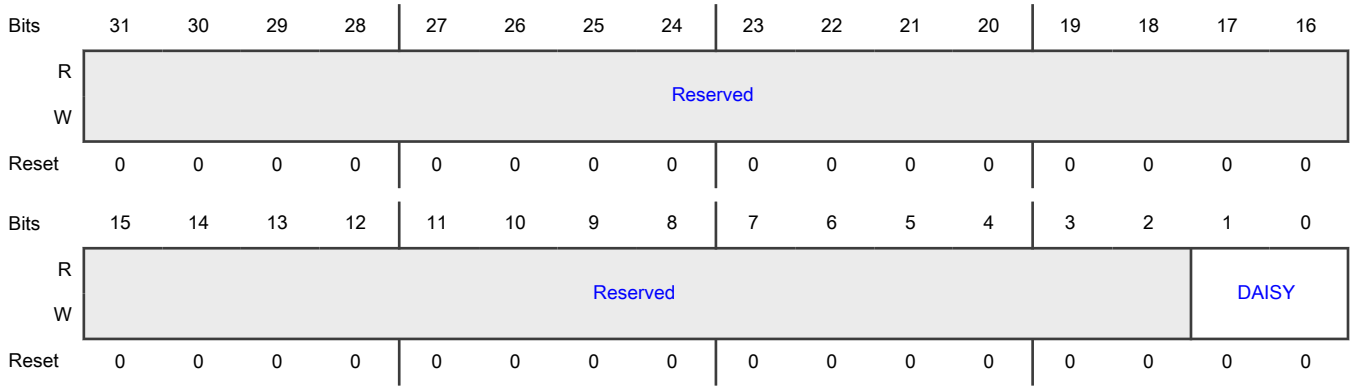
Offset

Register	Offset
LPUART5_IPP_IND_LPUART_CTS_N_SELECT_INPUT	68Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart5, In Pin: ipp_ind_lpuart_cts_n 00b - Selecting Pad: GPIO_EMC_B1_37 for Mode: ALT2 01b - Selecting Pad: GPIO_EMC_B2_19 for Mode: ALT2 10b - Selecting Pad: GPIO_SD_B2_06 for Mode: ALT6

17.4.1.418 LPUART5_IPP_IND_LPUART_DCD_N_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_DCD_N_SELECT_INPUT)

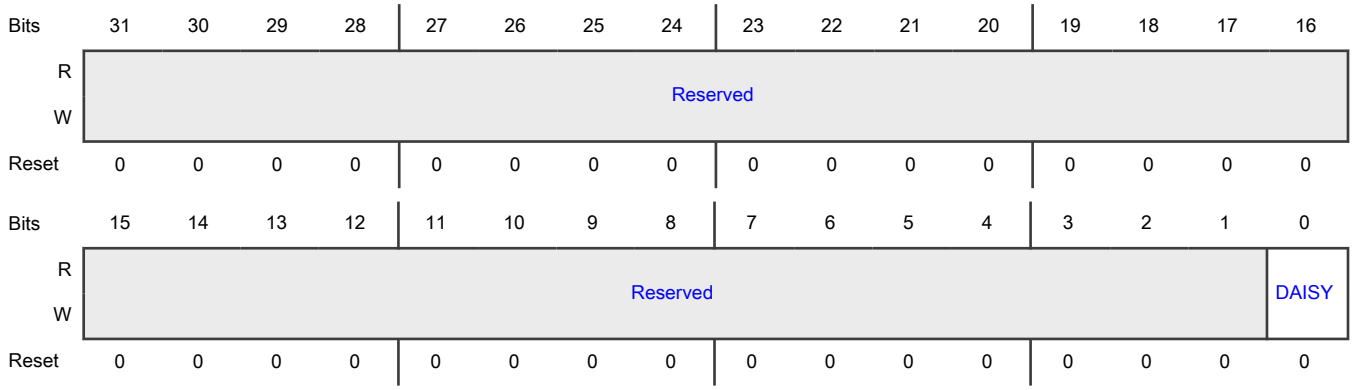
Offset

Register	Offset
LPUART5_IPP_IND_LPUART_DCD_N_SELECT_INPUT	690h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart5, In Pin: ipp_ind_lpuart_dcd_n 0b - Selecting Pad: GPIO_EMC_B2_15 for Mode: ALT4 1b - Selecting Pad: GPIO_SD_B2_10 for Mode: ALT6

17.4.1.419 LPUART5_IPP_IND_LPUART_DSR_N_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_DSR_N_SELECT_INPUT)

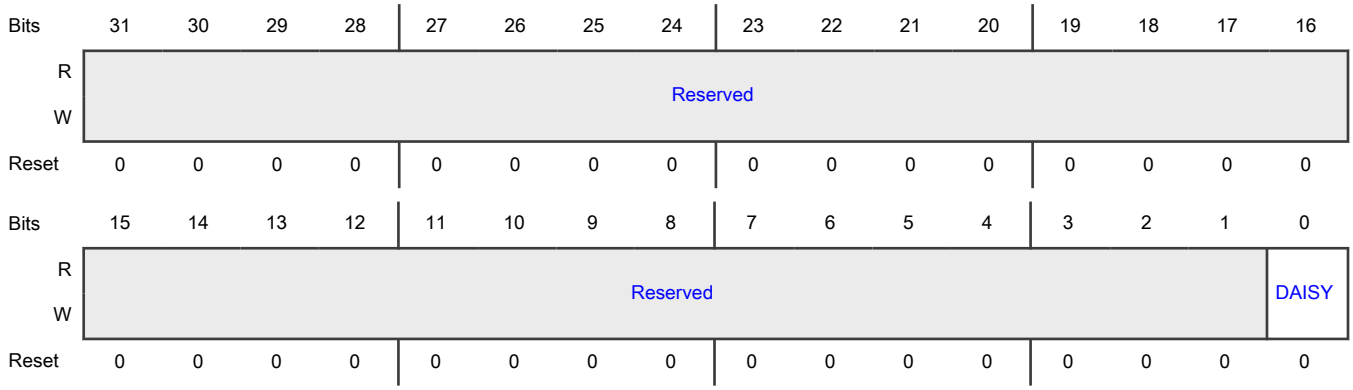
Offset

Register	Offset
LPUART5_IPP_IND_LPUART_DSR_N_SELECT_INPUT	694h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart5, In Pin: ipp_ind_lpuart_dsr_n 0b - Selecting Pad: GPIO_EMC_B2_14 for Mode: ALT4 1b - Selecting Pad: GPIO_SD_B2_11 for Mode: ALT6

17.4.1.420 LPUART5_IPP_IND_LPUART_RI_N_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_RI_N_SELECT_INPUT)

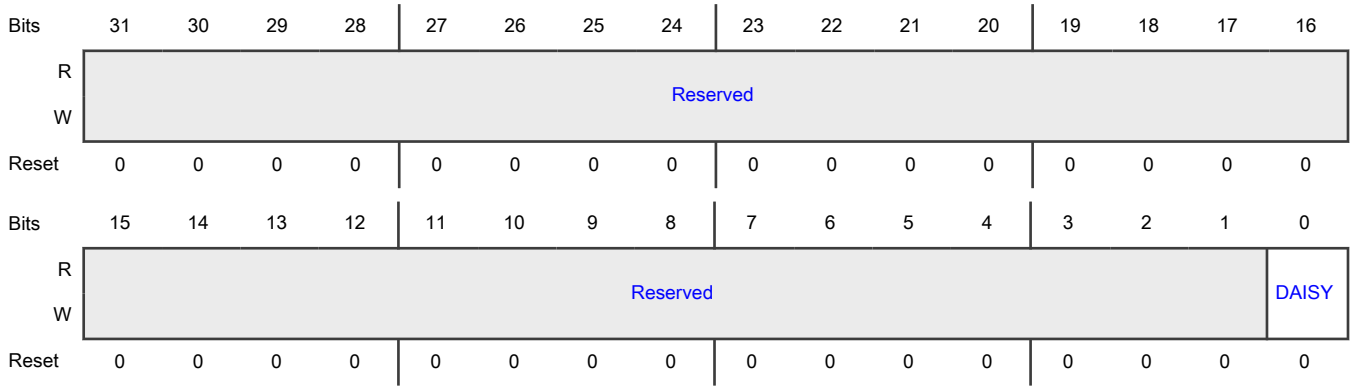
Offset

Register	Offset
LPUART5_IPP_IND_LPUART_RI_N_SELECT_INPUT	698h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart5, In Pin: ipp_ind_lpuart_ri_n 0b - Selecting Pad: GPIO_AD_18 for Mode: ALT2 1b - Selecting Pad: GPIO_SD_B2_04 for Mode: ALT6

17.4.1.421 LPUART5_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_RXD_SELECT_INPUT)

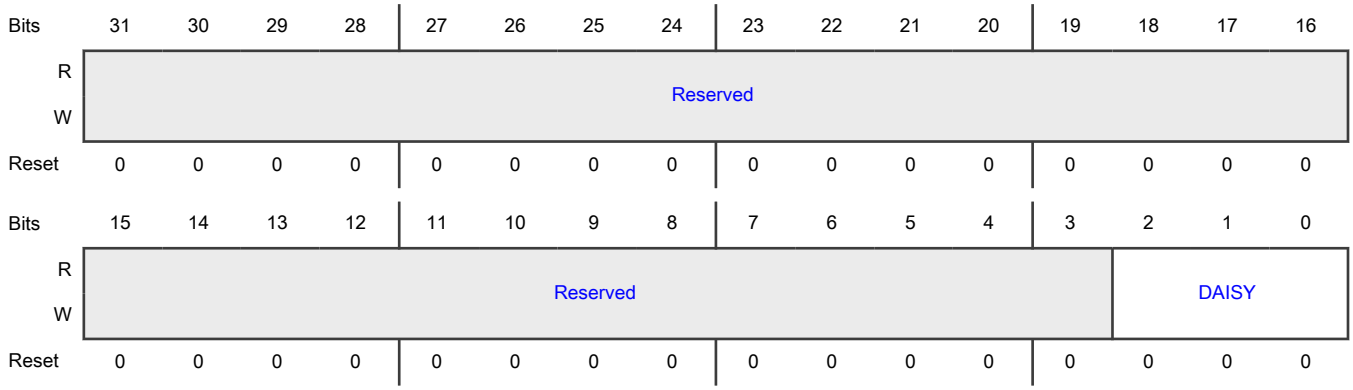
Offset

Register	Offset
LPUART5_IPP_IND_LPUART_RXD_SELECT_INPUT	69Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-3 —	- Reserved
2-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart5, In Pin: ipp_ind_lpuart_rxd 000b - Selecting Pad: GPIO_EMC_B1_15 for Mode: ALT2 001b - Selecting Pad: GPIO_EMC_B1_36 for Mode: ALT2 010b - Selecting Pad: GPIO_EMC_B2_18 for Mode: ALT2 011b - Selecting Pad: GPIO_AD_27 for Mode: ALT1 100b - Selecting Pad: GPIO_SD_B2_09 for Mode: ALT6

17.4.1.422 LPUART5_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART5_IPP_IND_LPUART_TXD_SELECT_INPUT)

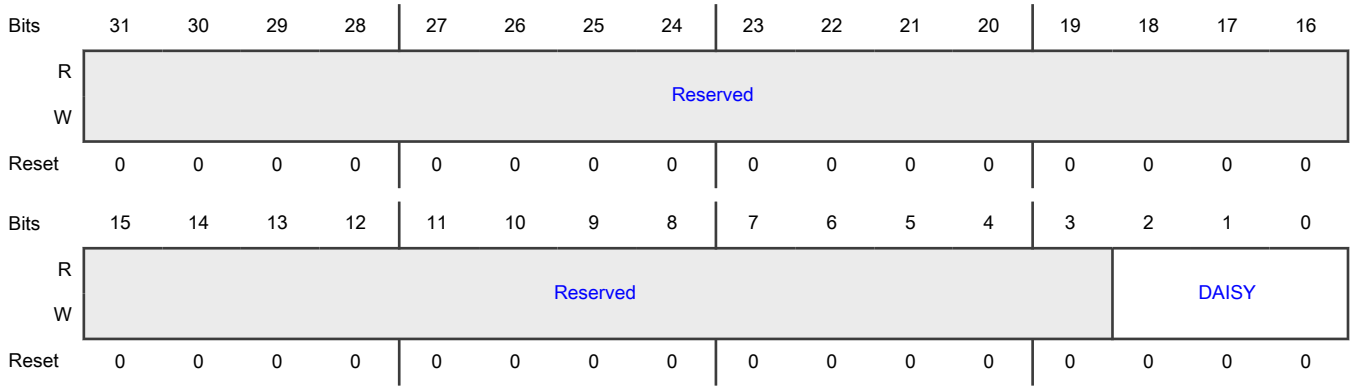
Offset

Register	Offset
LPUART5_IPP_IND_LPUART_TXD_SELECT_INPUT	6A0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-3 —	- Reserved
2-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart5, In Pin: ipp_ind_lpuart_txd 000b - Selecting Pad: GPIO_EMC_B1_14 for Mode: ALT2 001b - Selecting Pad: GPIO_EMC_B1_35 for Mode: ALT2 010b - Selecting Pad: GPIO_EMC_B2_17 for Mode: ALT2 011b - Selecting Pad: GPIO_AD_26 for Mode: ALT1 100b - Selecting Pad: GPIO_SD_B2_08 for Mode: ALT6

17.4.1.423 LPUART6_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_CTS_N_SELECT_INPUT)

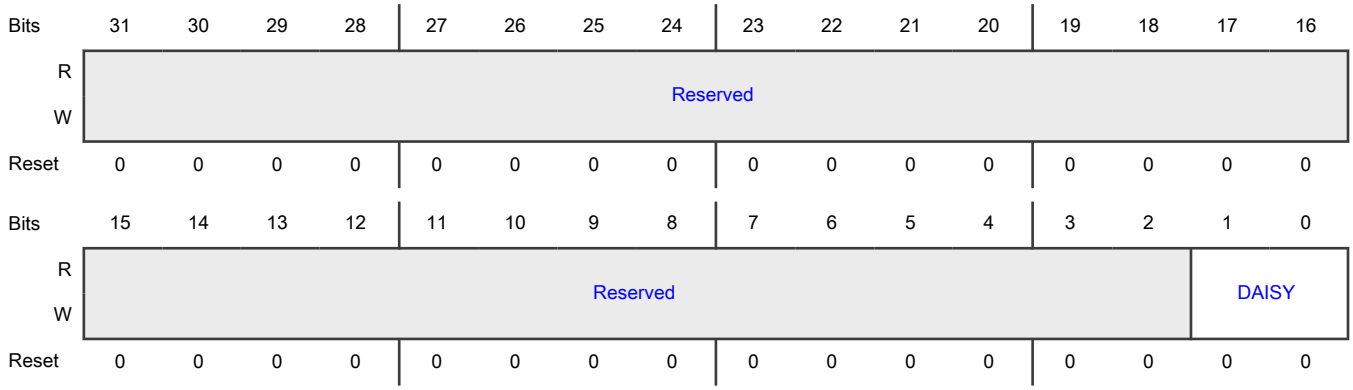
Offset

Register	Offset
LPUART6_IPP_IND_LPUART_CTS_N_SELECT_INPUT	6A4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart6, In Pin: ipp_ind_lpuart_cts_n 00b - Selecting Pad: GPIO_EMC_B1_33 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_26 for Mode: ALT0 10b - Selecting Pad: GPIO_B2_12 for Mode: ALT4

17.4.1.424 LPUART6_IPP_IND_LPUART_DCD_N_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_DCD_N_SELECT_INPUT)

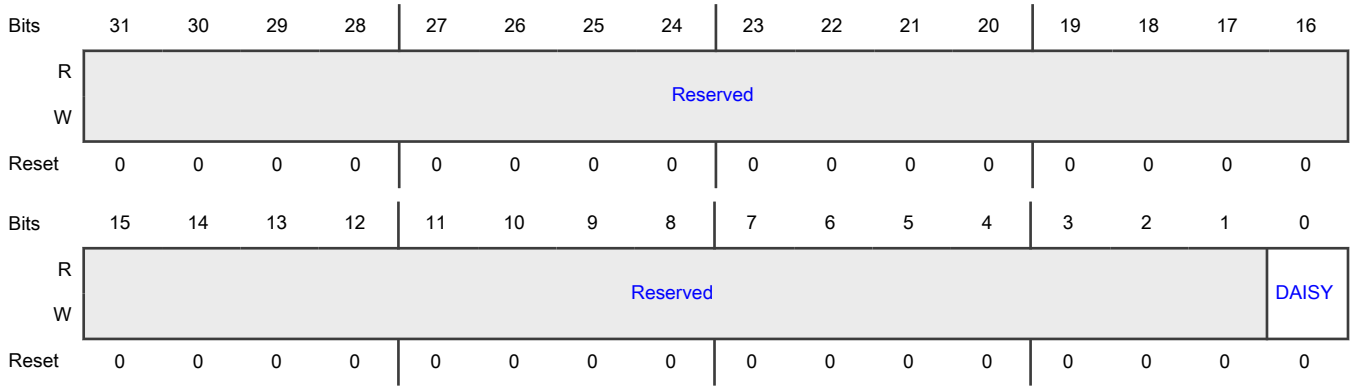
Offset

Register	Offset
LPUART6_IPP_IND_LPUART_DCD_N_SELECT_INPUT	6A8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart6, In Pin: ipp_ind_lpuart_dcd_n 0b - Selecting Pad: GPIO_EMC_B1_29 for Mode: ALT9 1b - Selecting Pad: GPIO_B2_07 for Mode: ALT3

17.4.1.425 LPUART6_IPP_IND_LPUART_DSR_N_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_DSR_N_SELECT_INPUT)

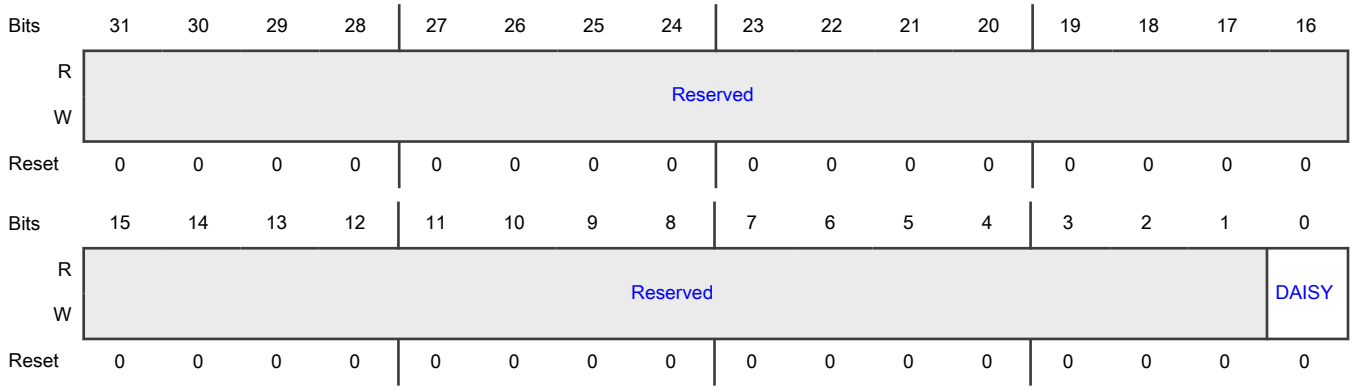
Offset

Register	Offset
LPUART6_IPP_IND_LPUART_DSR_N_SELECT_INPUT	6ACh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart6, In Pin: ipp_ind_lpuart_dsr_n 0b - Selecting Pad: GPIO_EMC_B1_30 for Mode: ALT9 1b - Selecting Pad: GPIO_B2_06 for Mode: ALT3

17.4.1.426 LPUART6_IPP_IND_LPUART_RI_N_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_RI_N_SELECT_INPUT)

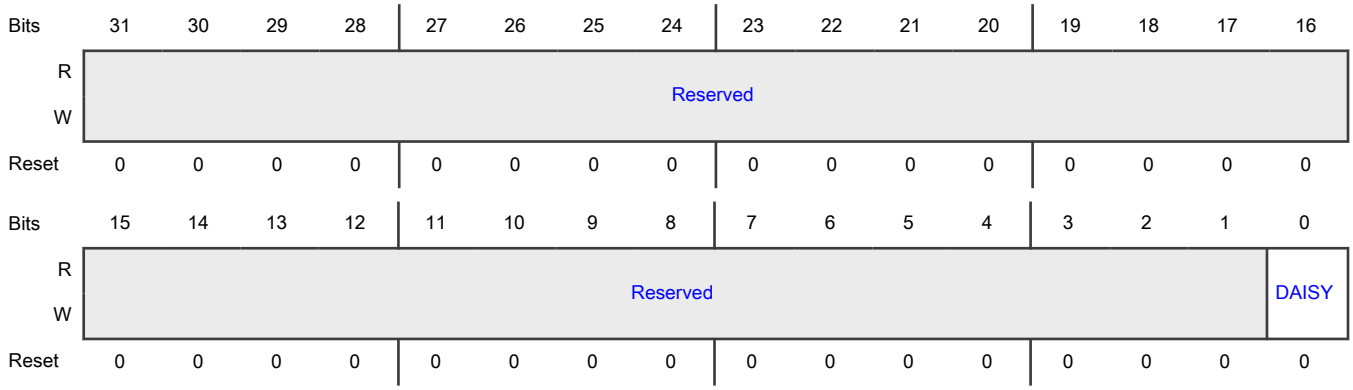
Offset

Register	Offset
LPUART6_IPP_IND_LPUART_RI_N_SELECT_INPUT	6B0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart6, In Pin: ipp_ind_lpuart_ri_n 0b - Selecting Pad: GPIO_EMC_B1_27 for Mode: ALT9 1b - Selecting Pad: GPIO_B2_08 for Mode: ALT4

17.4.1.427 LPUART6_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_RXD_SELECT_INPUT)

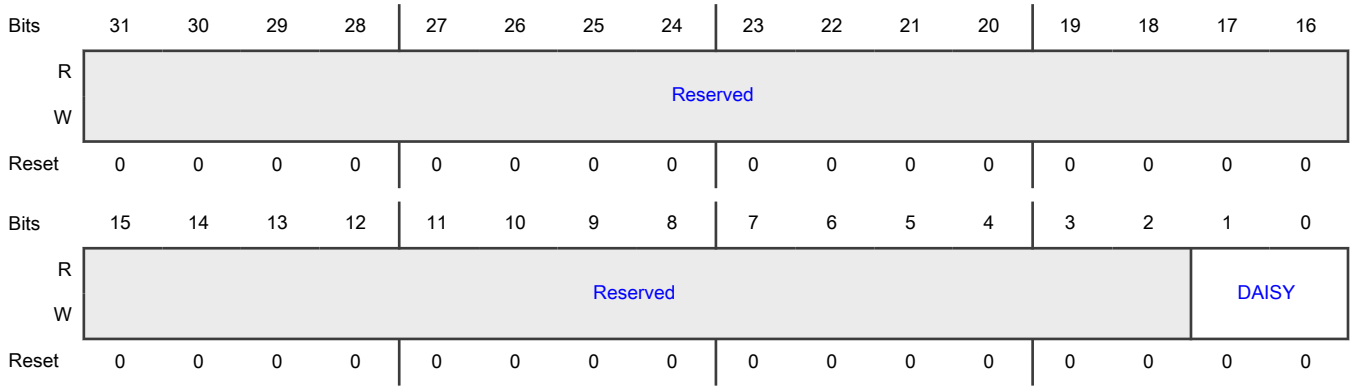
Offset

Register	Offset
LPUART6_IPP_IND_LPUART_RXD_SELECT_INPUT	6B4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart6, In Pin: ipp_ind_lpuart_rxd 00b - Selecting Pad: GPIO_EMC_B1_32 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_25 for Mode: ALT0 10b - Selecting Pad: GPIO_B2_11 for Mode: ALT4

17.4.1.428 LPUART6_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART6_IPP_IND_LPUART_TXD_SELECT_INPUT)

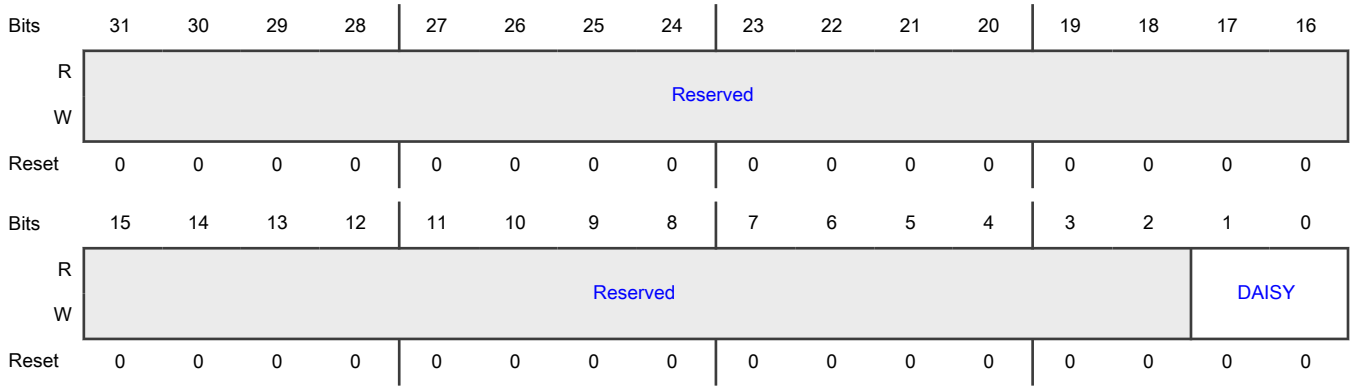
Offset

Register	Offset
LPUART6_IPP_IND_LPUART_TXD_SELECT_INPUT	6B8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart6, In Pin: ipp_ind_lpuart_txd 00b - Selecting Pad: GPIO_EMC_B1_31 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_24 for Mode: ALT0 10b - Selecting Pad: GPIO_B2_10 for Mode: ALT4

17.4.1.429 LPUART8_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART8_IPP_IND_LPUART_CTS_N_SELECT_INPUT)

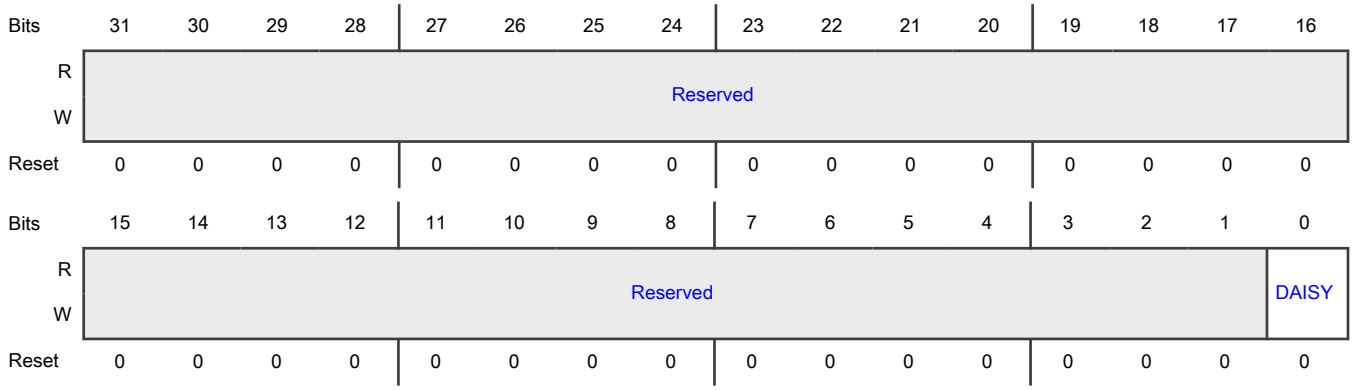
Offset

Register	Offset
LPUART8_IPP_IND_LPUART_CTS_N_SELECT_INPUT	6BCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart8, In Pin: ipp_ind_lpuart_cts_n 0b - Selecting Pad: GPIO_SD_B2_02 for Mode: ALT9 1b - Selecting Pad: GPIO_B2_10 for Mode: ALT3

17.4.1.430 LPUART8_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART8_IPP_IND_LPUART_RXD_SELECT_INPUT)

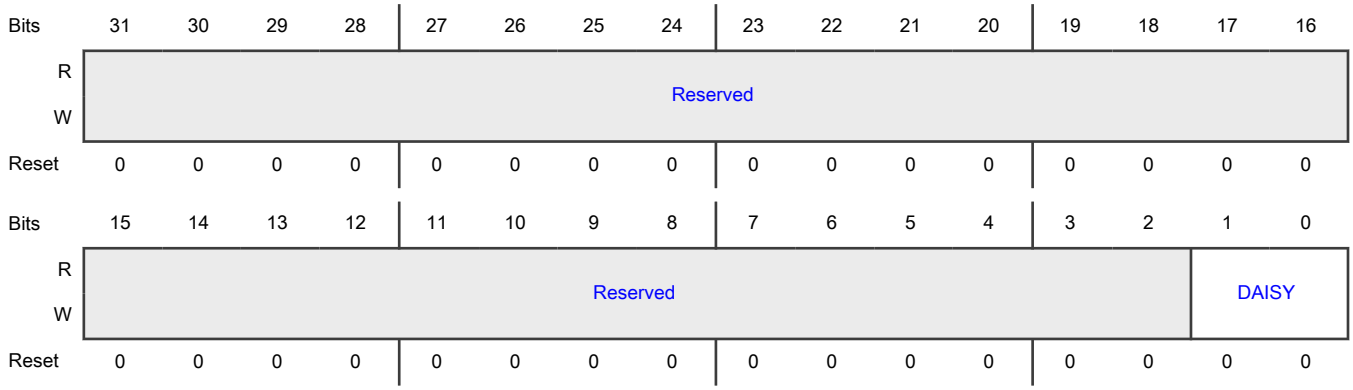
Offset

Register	Offset
LPUART8_IPP_IND_LPUART_RXD_SELECT_INPUT	6C0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart8, In Pin: ipp_ind_lpuart_rxd 00b - Selecting Pad: GPIO_AD_31 for Mode: ALT4 01b - Selecting Pad: GPIO_SD_B2_01 for Mode: ALT9 10b - Selecting Pad: GPIO_B2_13 for Mode: ALT2

17.4.1.431 LPUART8_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART8_IPP_IND_LPUART_TXD_SELECT_INPUT)

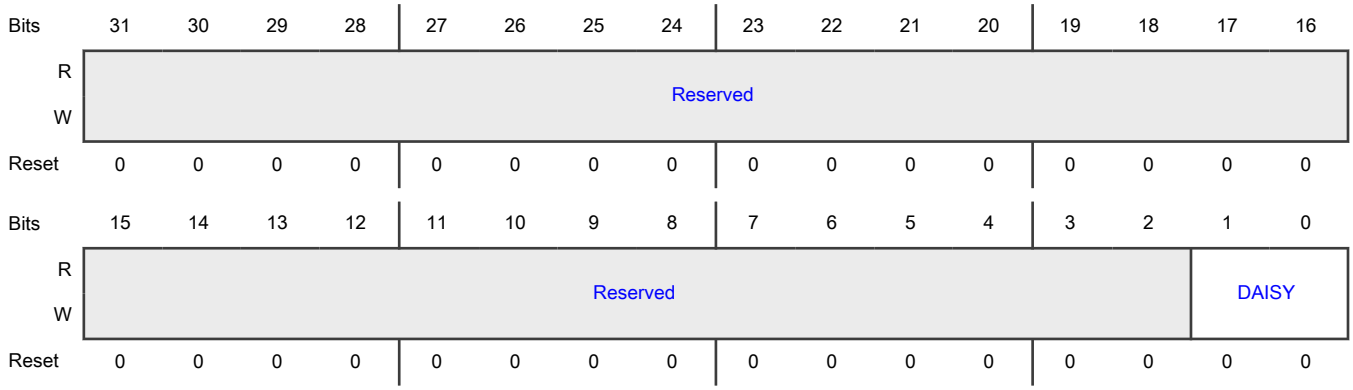
Offset

Register	Offset
LPUART8_IPP_IND_LPUART_TXD_SELECT_INPUT	6C4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart8, In Pin: ipp_ind_lpuart_txd 00b - Selecting Pad: GPIO_AD_30 for Mode: ALT4 01b - Selecting Pad: GPIO_SD_B2_00 for Mode: ALT9 10b - Selecting Pad: GPIO_B2_12 for Mode: ALT2

17.4.1.432 LPUART9_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART9_IPP_IND_LPUART_RXD_SELECT_INPUT)

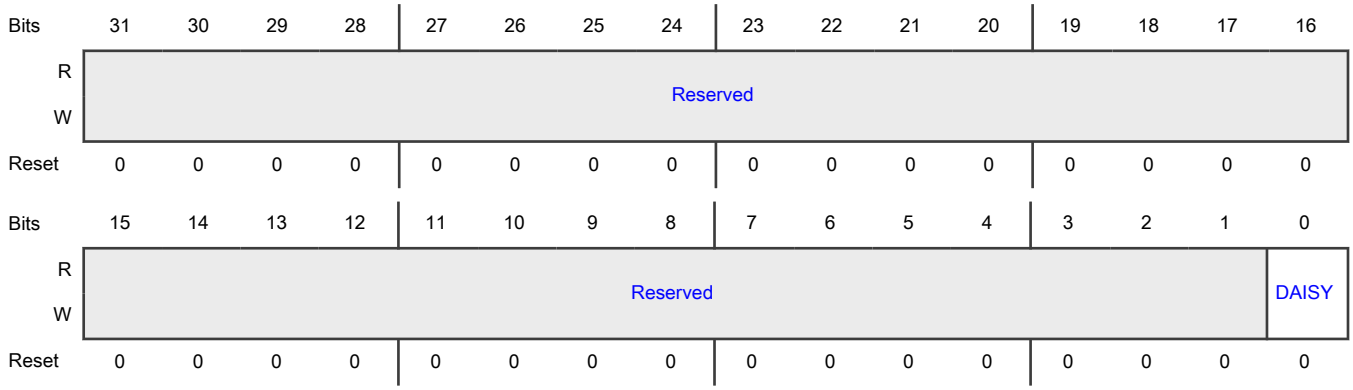
Offset

Register	Offset
LPUART9_IPP_IND_LPUART_RXD_SELECT_INPUT	6C8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart9, In Pin: ipp_ind_lpuart_rxd 0b - Selecting Pad: GPIO_EMC_B1_17 for Mode: ALT2 1b - Selecting Pad: GPIO_B1_04 for Mode: ALT2

17.4.1.433 LPUART9_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART9_IPP_IND_LPUART_TXD_SELECT_INPUT)

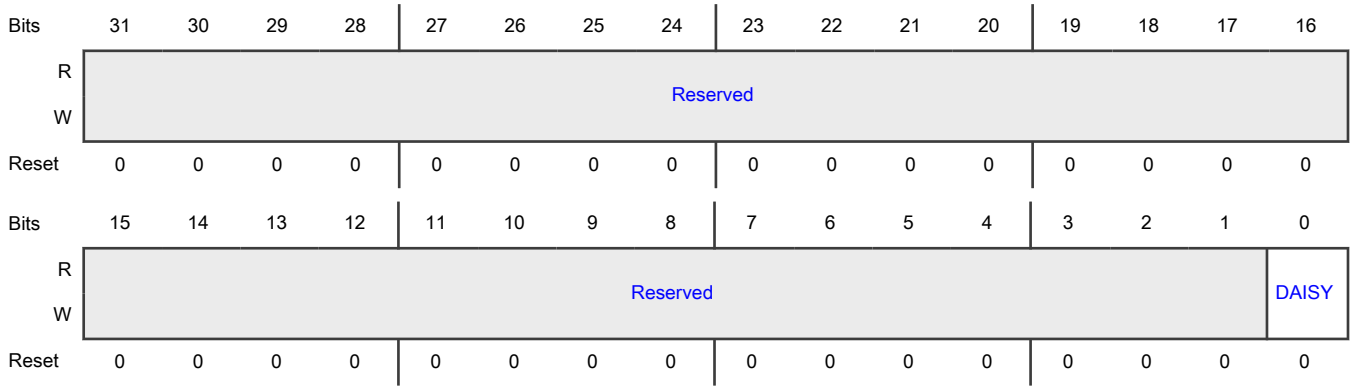
Offset

Register	Offset
LPUART9_IPP_IND_LPUART_TXD_SELECT_INPUT	6CCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart9, In Pin: ipp_ind_lpuart_txd 0b - Selecting Pad: GPIO_EMC_B1_16 for Mode: ALT2 1b - Selecting Pad: GPIO_B1_06 for Mode: ALT2

17.4.1.434 MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_0 DAISY Register (MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_0)

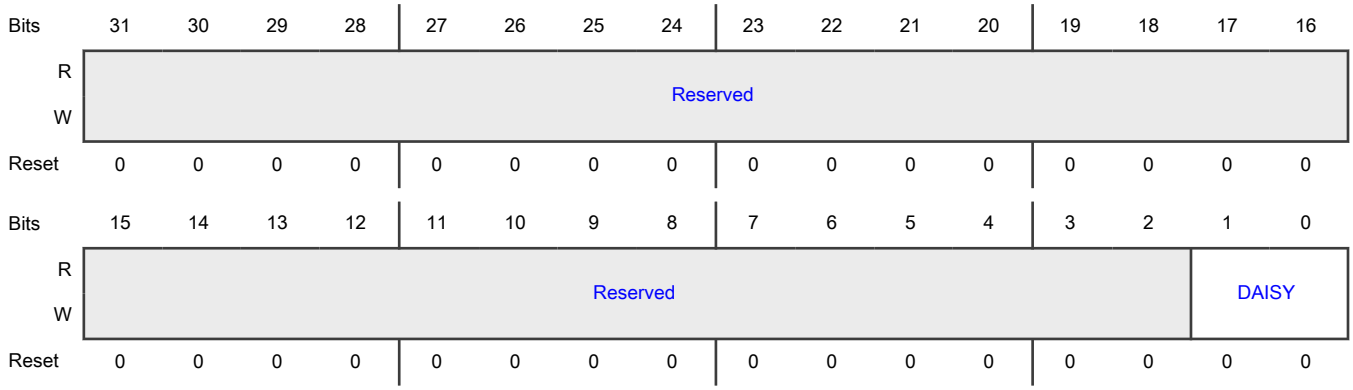
Offset

Register	Offset
MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_0	6D0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: mic, In Pin: ipp_ind_mic_pdm_bitstream0 00b - Selecting Pad: GPIO_AD_01 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_28 for Mode: ALT12 10b - Selecting Pad: GPIO_SD_B2_00 for Mode: ALT12 11b - Selecting Pad: GPIO_B2_10 for Mode: ALT0

17.4.1.435 MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_1 DAISY Register (MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_1)

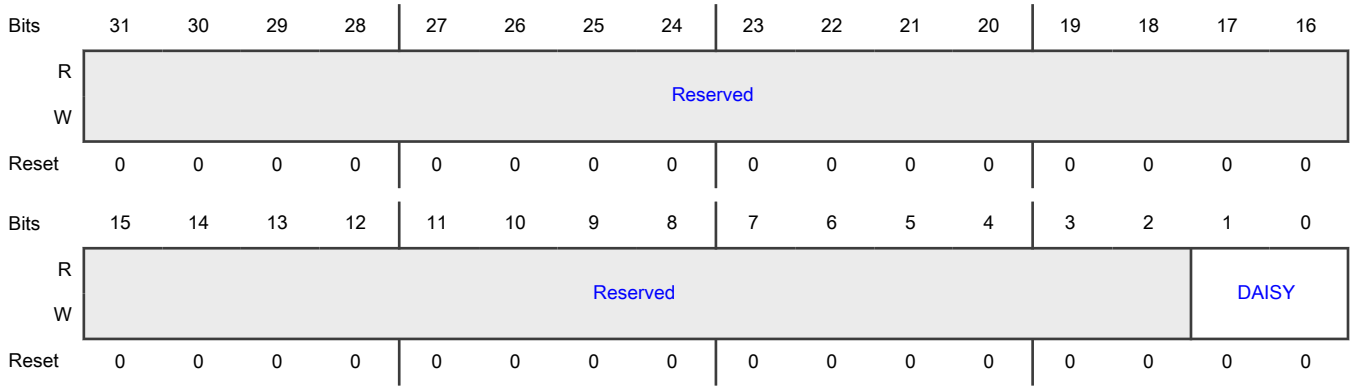
Offset

Register	Offset
MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_1	6D4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: mic, In Pin: ipp_ind_mic_pdm_bitstream1 00b - Selecting Pad: GPIO_AD_02 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_29 for Mode: ALT12 10b - Selecting Pad: GPIO_SD_B2_01 for Mode: ALT12 11b - Selecting Pad: GPIO_B2_11 for Mode: ALT0

17.4.1.436 MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_2 DAISY Register (MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_2)

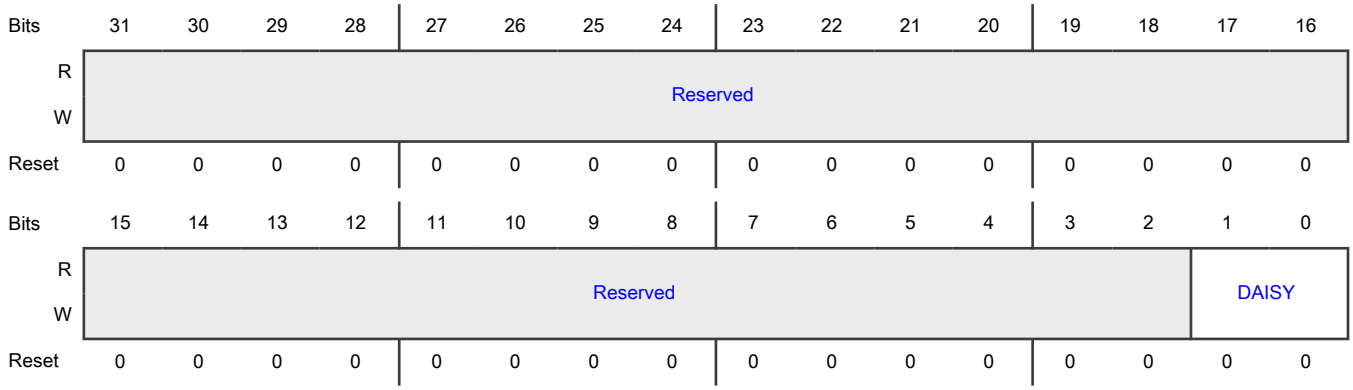
Offset

Register	Offset
MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_2	6D8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: mic, In Pin: ipp_ind_mic_pdm_bitstream2 00b - Selecting Pad: GPIO_AD_03 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_26 for Mode: ALT12 10b - Selecting Pad: GPIO_SD_B2_02 for Mode: ALT12 11b - Selecting Pad: GPIO_B2_12 for Mode: ALT0

17.4.1.437 MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_3 DAISY Register (MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_3)

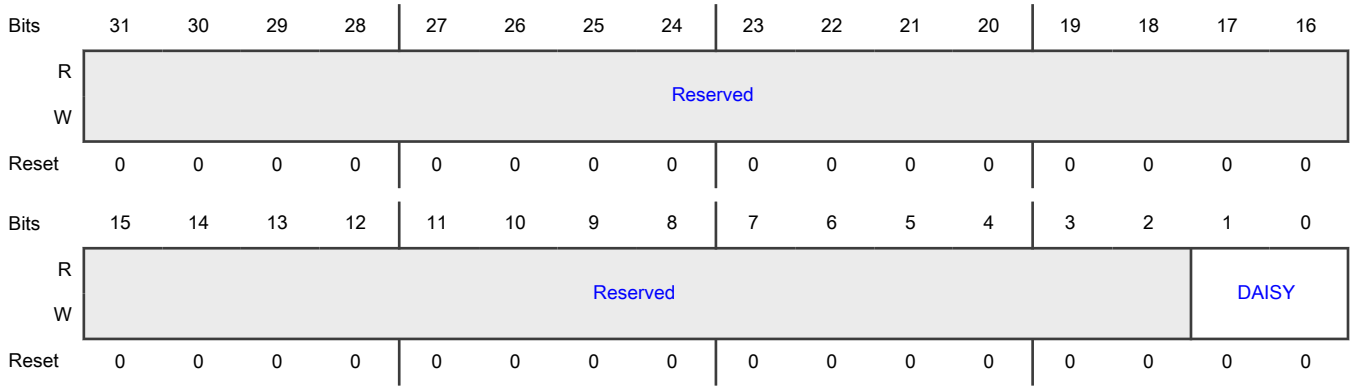
Offset

Register	Offset
MIC_IPP_IND_MIC_PDM_BITSTREAM_SELECT_INPUT_3	6DCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: mic, In Pin: ipp_ind_mic_pdm_bitstream3 00b - Selecting Pad: GPIO_AD_04 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_32 for Mode: ALT12 10b - Selecting Pad: GPIO_SD_B2_03 for Mode: ALT12 11b - Selecting Pad: GPIO_B2_13 for Mode: ALT0

17.4.1.438 NETC_EMDIO_IN_SELECT_INPUT DAISY Register (NETC_EMDIO_IN_SELECT_INPUT)

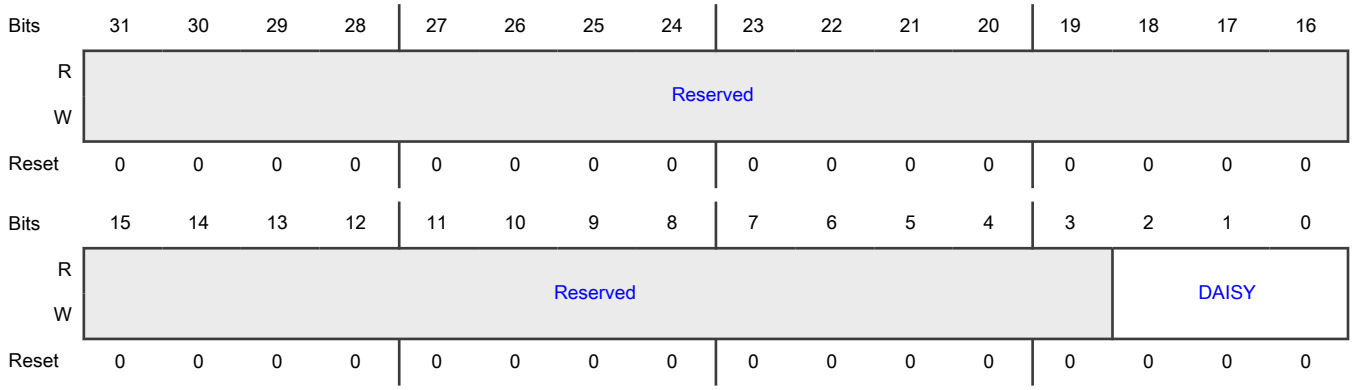
Offset

Register	Offset
NETC_EMDIO_IN_SELECT_INPUT	798h

Function

DAISY Register

Diagram



Fields

Field	Function
31-3 —	- Reserved
2-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: emdio_in 000b - Selecting Pad: GPIO_EMC_B1_19 for Mode: ALT10 001b - Selecting Pad: GPIO_EMC_B1_41 for Mode: ALT1 010b - Selecting Pad: GPIO_EMC_B2_01 for Mode: ALT3 011b - Selecting Pad: GPIO_EMC_B2_20 for Mode: ALT4 100b - Selecting Pad: GPIO_AD_31 for Mode: ALT7 101b - Selecting Pad: GPIO_SD_B2_10 for Mode: ALT10 110b - Selecting Pad: GPIO_B1_12 for Mode: ALT1 111b - Selecting Pad: GPIO_B2_02 for Mode: ALT3

17.4.1.439 NETC_ETH2_COL_SELECT_INPUT DAISY Register (NETC_ETH2_COL_SELECT_INPUT)

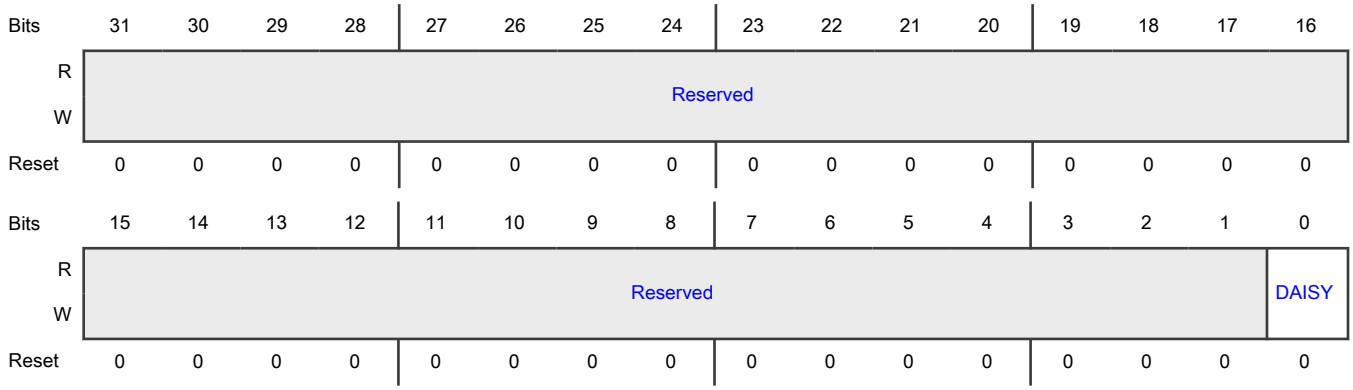
Offset

Register	Offset
NETC_ETH2_COL_SELECT_INPUT	79Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth2_col 0b - Selecting Pad: GPIO_EMC_B1_19 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B1_41 for Mode: ALT4

17.4.1.440 NETC_ETH2_CRS_SELECT_INPUT DAISY Register (NETC_ETH2_CRS_SELECT_INPUT)

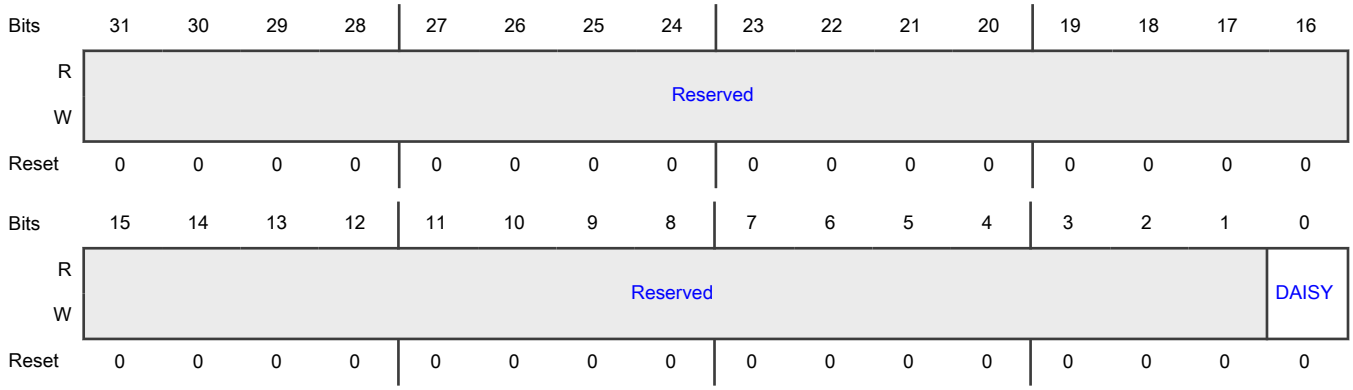
Offset

Register	Offset
NETC_ETH2_CRS_SELECT_INPUT	7A0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth2_crs 0b - Selecting Pad: GPIO_EMC_B1_18 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B1_40 for Mode: ALT4

17.4.1.441 NETC_ETH2_SLV_MDC_IN_SELECT_INPUT DAISY Register (NETC_ETH2_SLV_MDC_IN_SELECT_INPUT)

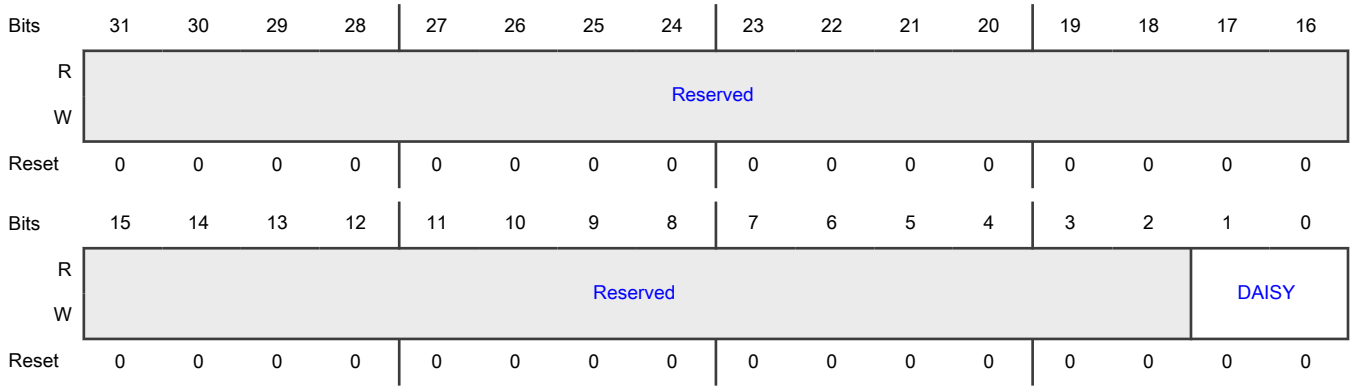
Offset

Register	Offset
NETC_ETH2_SLV_MDC_IN_SELECT_INPUT	7A4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth2_slv_mdc_in 00b - Selecting Pad: GPIO_EMC_B1_16 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B1_40 for Mode: ALT2 10b - Selecting Pad: GPIO_B2_01 for Mode: ALT10

17.4.1.442 NETC_ETH2_SLV_MDIO_IN_SELECT_INPUT DAISY Register (NETC_ETH2_SLV_MDIO_IN_SELECT_INPUT)

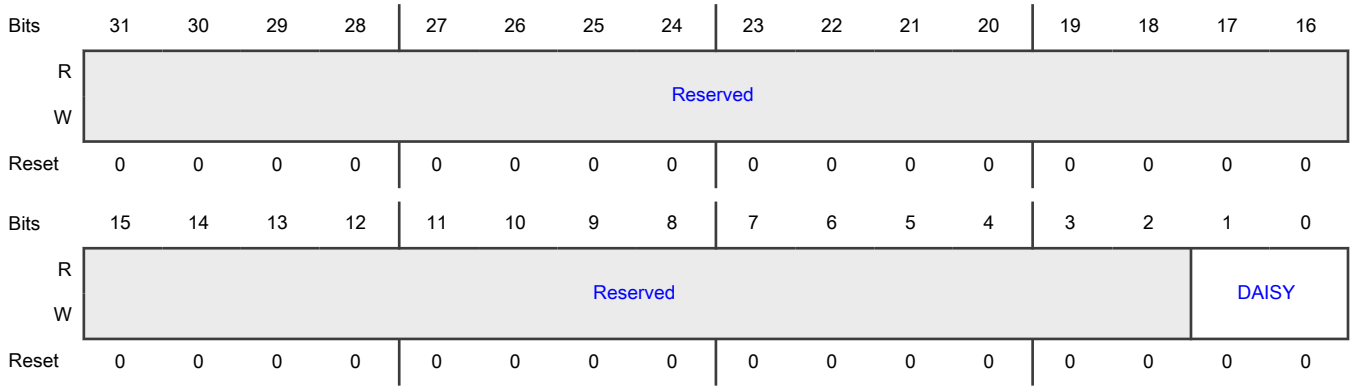
Offset

Register	Offset
NETC_ETH2_SLV_MDIO_IN_SELECT_INPUT	7A8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth2_slv_mdio_in 00b - Selecting Pad: GPIO_EMC_B1_17 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B1_41 for Mode: ALT2 10b - Selecting Pad: GPIO_B2_00 for Mode: ALT10

17.4.1.443 NETC_ETH3_COL_SELECT_INPUT DAISY Register (NETC_ETH3_COL_SELECT_INPUT)

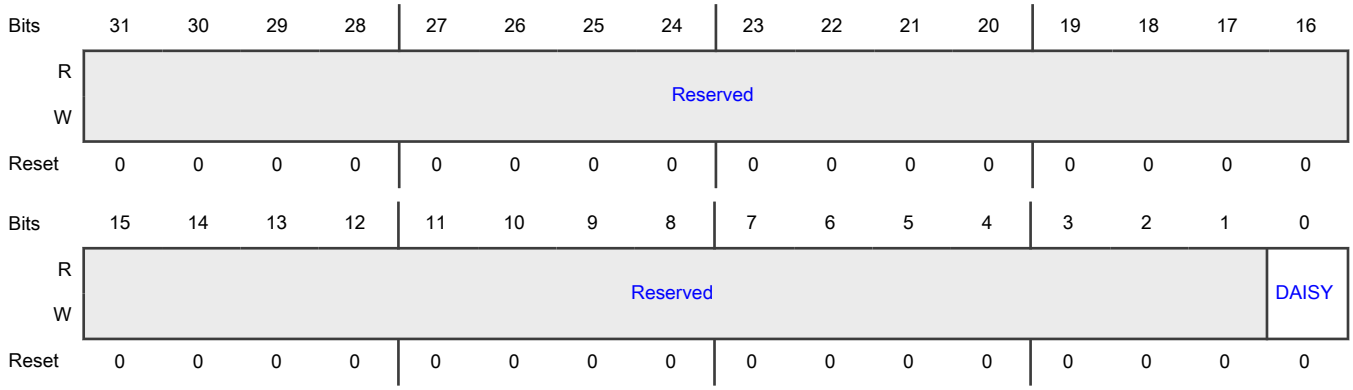
Offset

Register	Offset
NETC_ETH3_COL_SELECT_INPUT	7ACh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth3_col 0b - Selecting Pad: GPIO_EMC_B1_15 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_04 for Mode: ALT9

17.4.1.444 NETC_ETH3_CRS_SELECT_INPUT DAISY Register (NETC_ETH3_CRS_SELECT_INPUT)

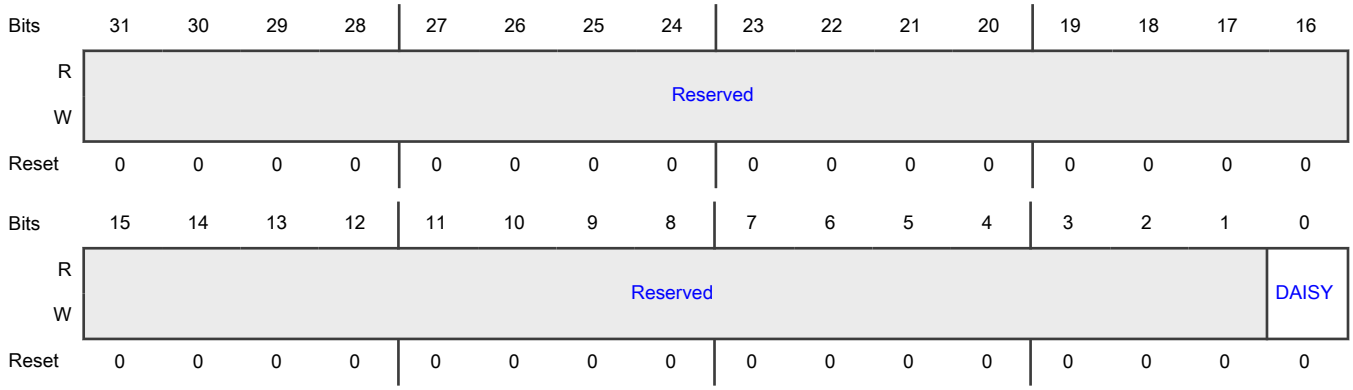
Offset

Register	Offset
NETC_ETH3_CRS_SELECT_INPUT	7B0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth3_crs 0b - Selecting Pad: GPIO_EMC_B1_14 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_03 for Mode: ALT9

17.4.1.445 NETC_ETH3_SLV_MDC_IN_SELECT_INPUT DAISY Register (NETC_ETH3_SLV_MDC_IN_SELECT_INPUT)

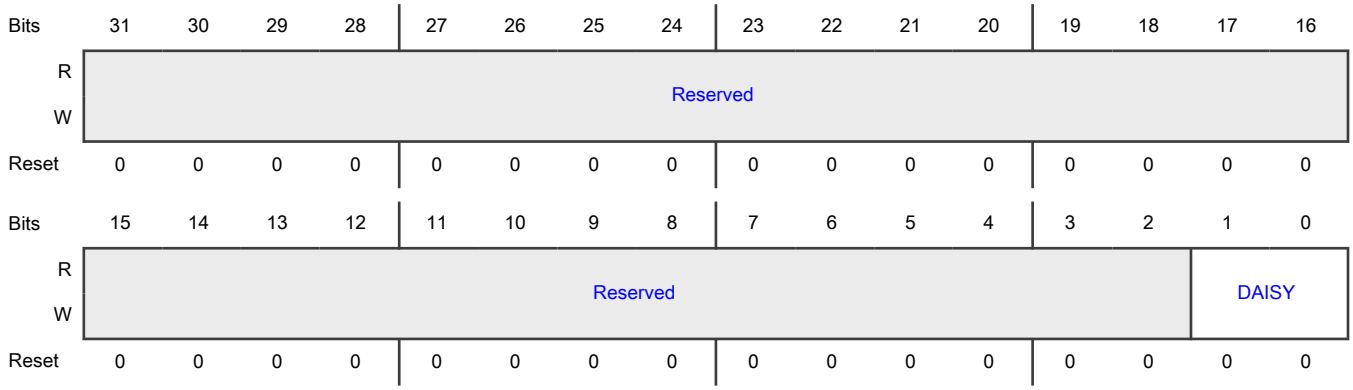
Offset

Register	Offset
NETC_ETH3_SLV_MDC_IN_SELECT_INPUT	7B4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth3_slv_mdc_in 00b - Selecting Pad: GPIO_EMC_B1_16 for Mode: ALT4 01b - Selecting Pad: GPIO_EMC_B1_24 for Mode: ALT9 10b - Selecting Pad: GPIO_EMC_B2_19 for Mode: ALT9

17.4.1.446 NETC_ETH3_SLV_MDIO_IN_SELECT_INPUT DAISY Register (NETC_ETH3_SLV_MDIO_IN_SELECT_INPUT)

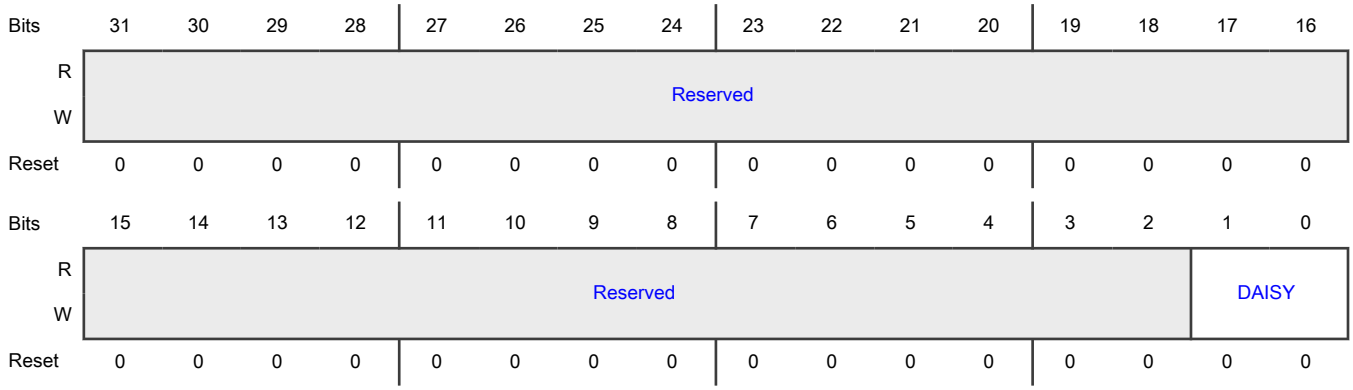
Offset

Register	Offset
NETC_ETH3_SLV_MDIO_IN_SELECT_INPUT	7B8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth3_slv_mdio_in 00b - Selecting Pad: GPIO_EMC_B1_17 for Mode: ALT4 01b - Selecting Pad: GPIO_EMC_B1_25 for Mode: ALT9 10b - Selecting Pad: GPIO_EMC_B2_20 for Mode: ALT9

17.4.1.447 NETC_ETH4_COL_SELECT_INPUT DAISY Register (NETC_ETH4_COL_SELECT_INPUT)

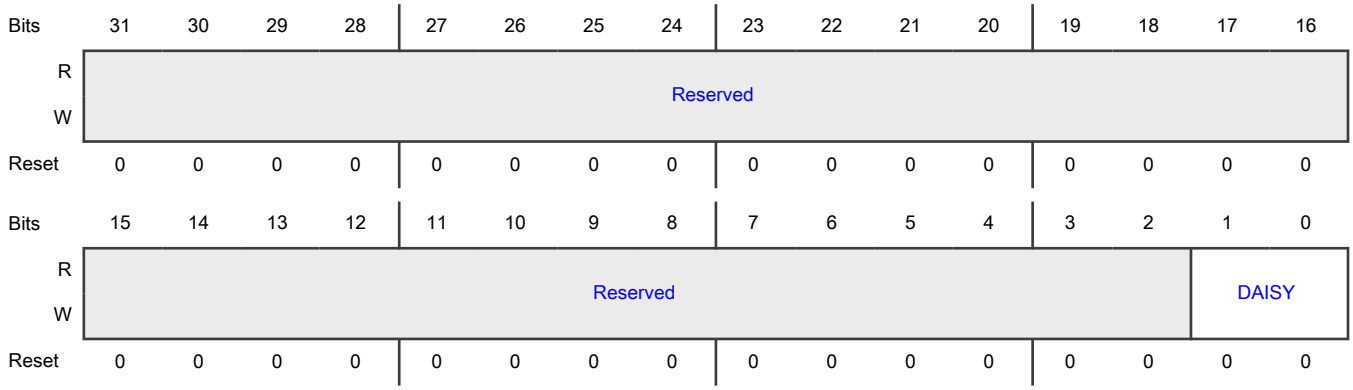
Offset

Register	Offset
NETC_ETH4_COL_SELECT_INPUT	7BCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth4_col 00b - Selecting Pad: GPIO_EMC_B1_15 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_06 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_01 for Mode: ALT8

17.4.1.448 NETC_ETH4_CRS_SELECT_INPUT DAISY Register (NETC_ETH4_CRS_SELECT_INPUT)

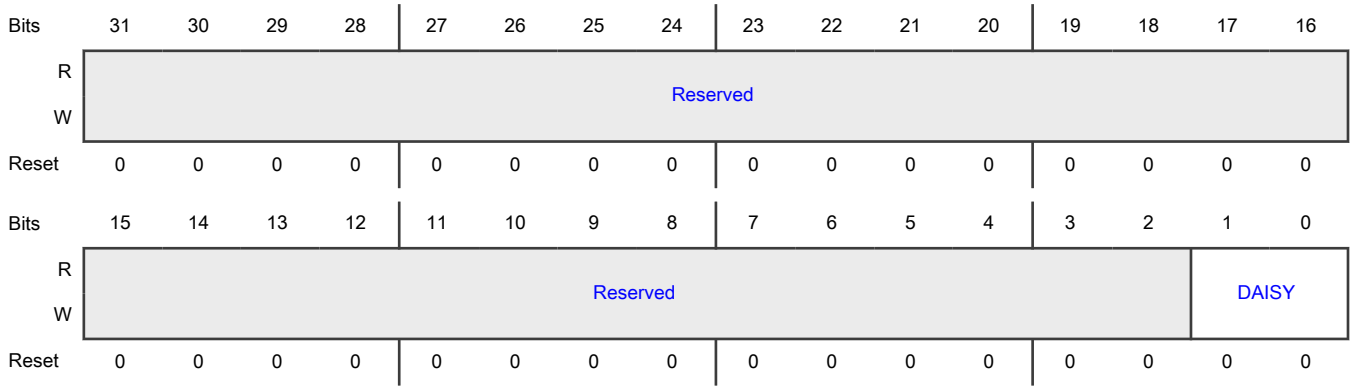
Offset

Register	Offset
NETC_ETH4_CRS_SELECT_INPUT	7C0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth4_crs 00b - Selecting Pad: GPIO_EMC_B1_14 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_05 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_00 for Mode: ALT8

17.4.1.449 NETC_ETH4_SLV_MDC_IN_SELECT_INPUT DAISY Register (NETC_ETH4_SLV_MDC_IN_SELECT_INPUT)

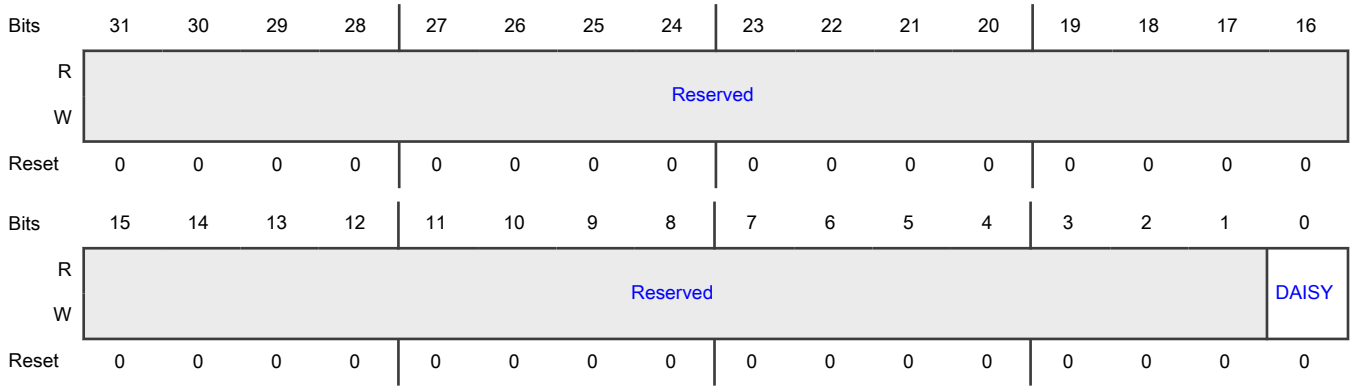
Offset

Register	Offset
NETC_ETH4_SLV_MDC_IN_SELECT_INPUT	7C4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth4_slv_mdc_in 0b - Selecting Pad: GPIO_EMC_B1_16 for Mode: ALT6 1b - Selecting Pad: GPIO_EMC_B2_05 for Mode: ALT1

17.4.1.450 NETC_ETH4_SLV_MDIO_IN_SELECT_INPUT DAISY Register (NETC_ETH4_SLV_MDIO_IN_SELECT_INPUT)

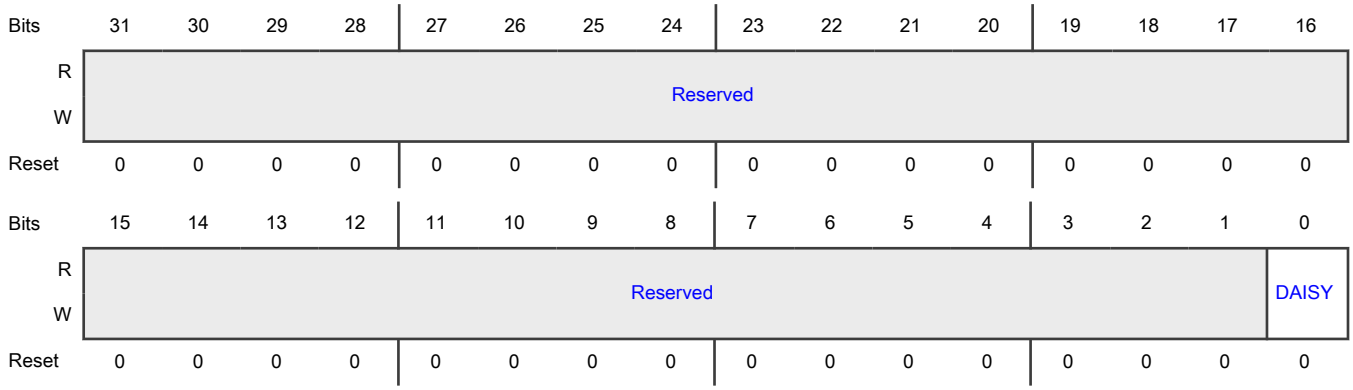
Offset

Register	Offset
NETC_ETH4_SLV_MDIO_IN_SELECT_INPUT	7C8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: eth4_slv_mdio_in 0b - Selecting Pad: GPIO_EMC_B1_17 for Mode: ALT6 1b - Selecting Pad: GPIO_EMC_B2_06 for Mode: ALT1

17.4.1.451 NETC_TMR_TRIG1_SELECT_INPUT DAISY Register (NETC_TMR_TRIG1_SELECT_INPUT)

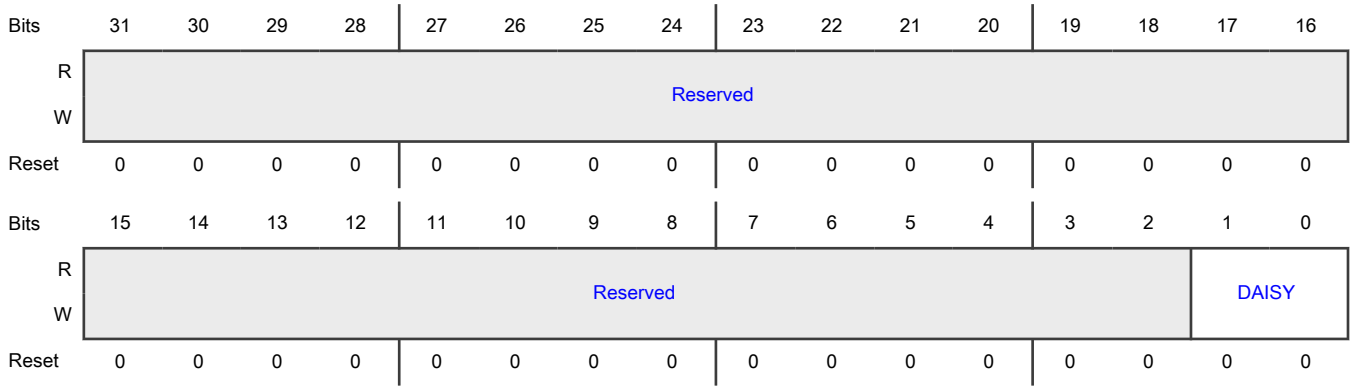
Offset

Register	Offset
NETC_TMR_TRIG1_SELECT_INPUT	7CCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: TMR_TRIG1 00b - Selecting Pad: GPIO_AD_20 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_24 for Mode: ALT7 10b - Selecting Pad: GPIO_AD_32 for Mode: ALT7 11b - Selecting Pad: GPIO_SD_B2_11 for Mode: ALT8

17.4.1.452 NETC_TMR_TRIG2_SELECT_INPUT DAISY Register (NETC_TMR_TRIG2_SELECT_INPUT)

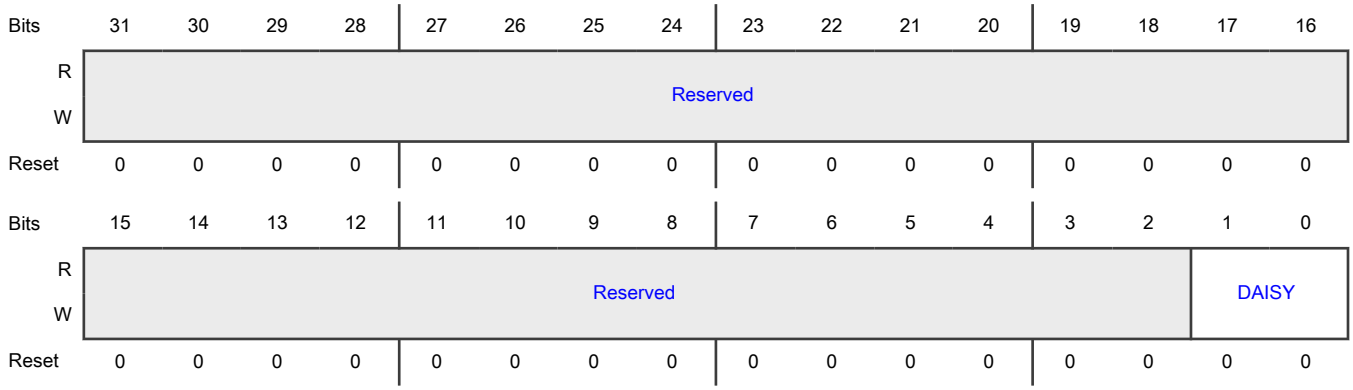
Offset

Register	Offset
NETC_TMR_TRIG2_SELECT_INPUT	7D0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc, In Pin: tmr_1588_trig2 00b - Selecting Pad: GPIO_AD_21 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_25 for Mode: ALT7 10b - Selecting Pad: GPIO_AD_33 for Mode: ALT7 11b - Selecting Pad: GPIO_SD_B2_10 for Mode: ALT8

17.4.1.453 NETC_CLKGEN_IPP_TMR_CLK_SELECT_INPUT DAISY Register (NETC_CLKGEN_IPP_TMR_CLK_SELECT_INPUT)

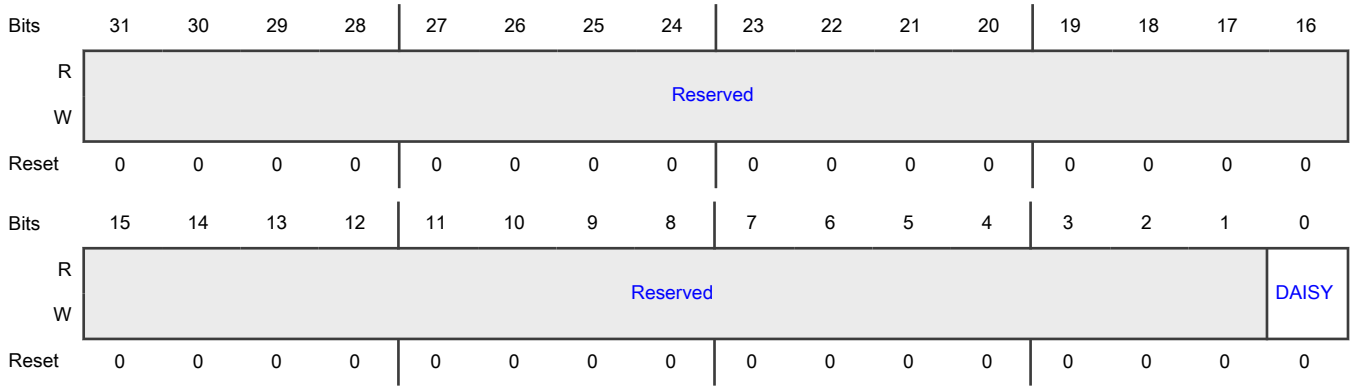
Offset

Register	Offset
NETC_CLKGEN_IPP_TMR_CLK_SELECT_INPUT	7D4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_clkgen, In Pin: ipp_tmr_clk 0b - Selecting Pad: GPIO_AD_20 for Mode: ALT7 1b - Selecting Pad: GPIO_SD_B2_00 for Mode: ALT8

17.4.1.454 NETC_PINMUX_IPP_IND_ETH0_RX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RX_CLK_SELECT_INPUT)

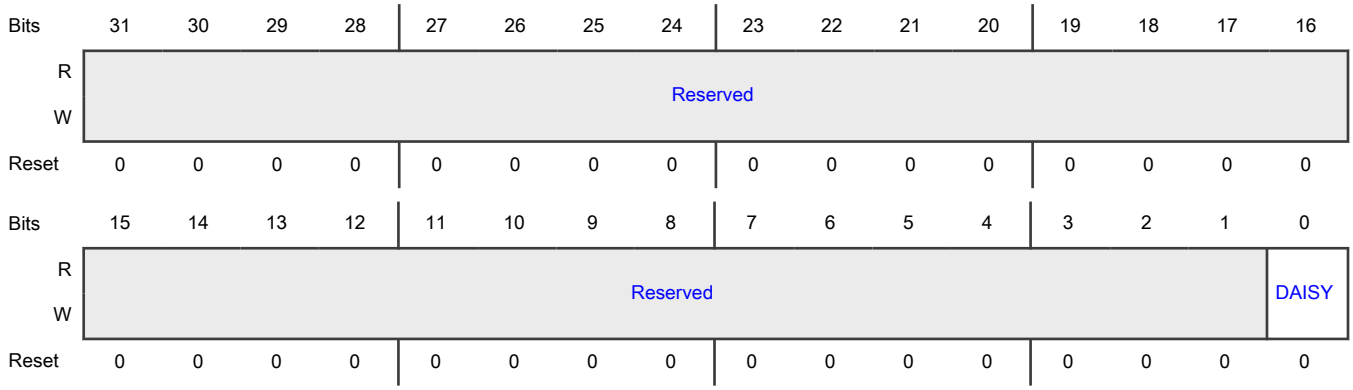
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH0_RX_CLK_SELECT_INPUT	7D8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth0_rx_clk 0b - Selecting Pad: GPIO_EMC_B2_00 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_15 for Mode: ALT3

17.4.1.455 NETC_PINMUX_IPP_IND_ETH0_RX_DV_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RX_DV_SELECT_INPUT)

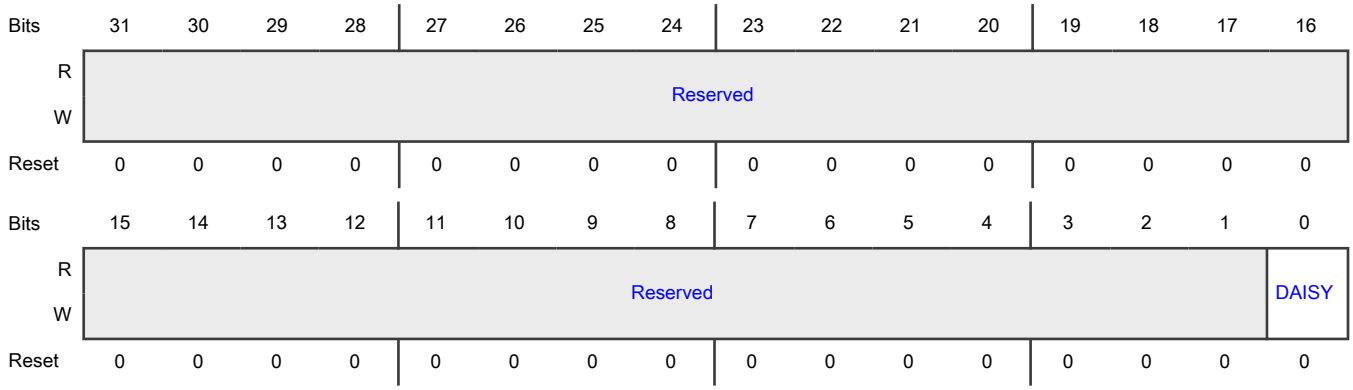
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH0_RX_DV_SELECT_INPUT	7DCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth0_rx_dv 0b - Selecting Pad: GPIO_EMC_B1_40 for Mode: ALT9 1b - Selecting Pad: GPIO_EMC_B2_11 for Mode: ALT3

17.4.1.456 NETC_PINMUX_IPP_IND_ETH0_RX_ER_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RX_ER_SELECT_INPUT)

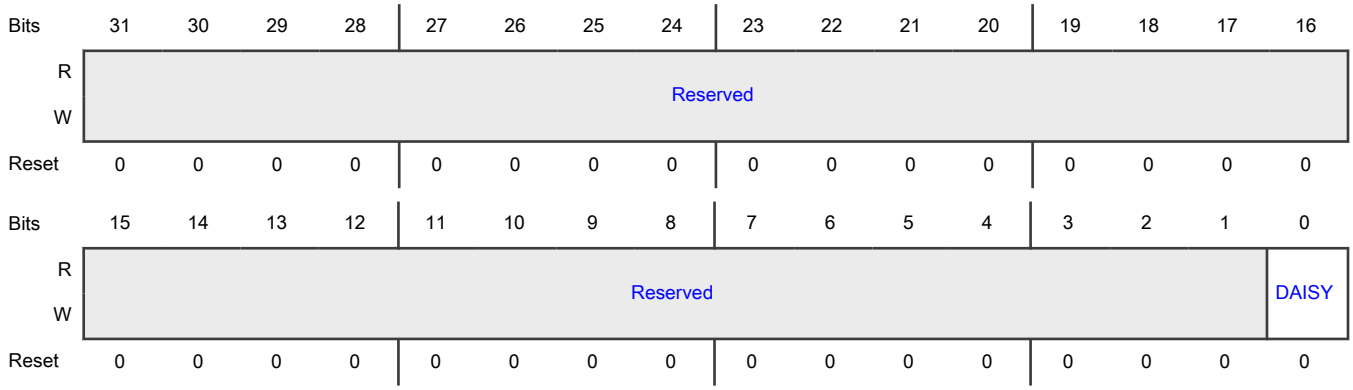
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH0_RX_ER_SELECT_INPUT	7E0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth0_rx_er 0b - Selecting Pad: GPIO_EMC_B1_41 for Mode: ALT9 1b - Selecting Pad: GPIO_EMC_B2_12 for Mode: ALT3

17.4.1.457 NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_0 DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_0)

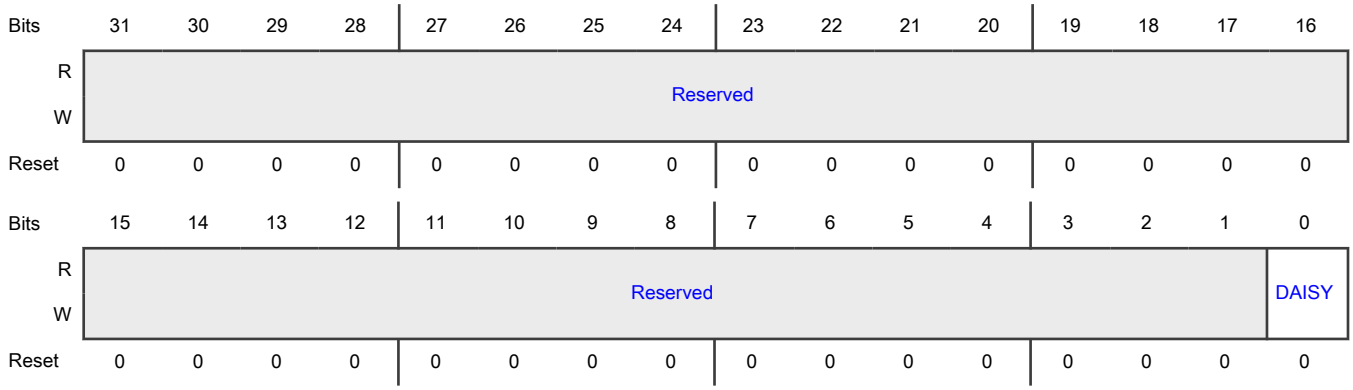
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_0	7E4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth0_rxd0 0b - Selecting Pad: GPIO_EMC_B1_38 for Mode: ALT9 1b - Selecting Pad: GPIO_EMC_B2_09 for Mode: ALT3

17.4.1.458 NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_1 DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_1)

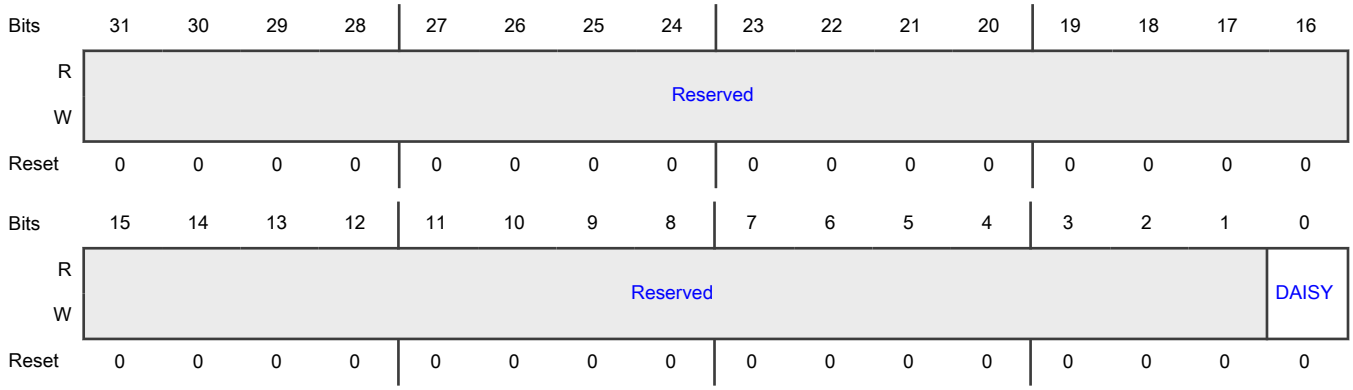
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_1	7E8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth0_rxd1 0b - Selecting Pad: GPIO_EMC_B1_39 for Mode: ALT9 1b - Selecting Pad: GPIO_EMC_B2_10 for Mode: ALT3

17.4.1.459 NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_2 DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_2)

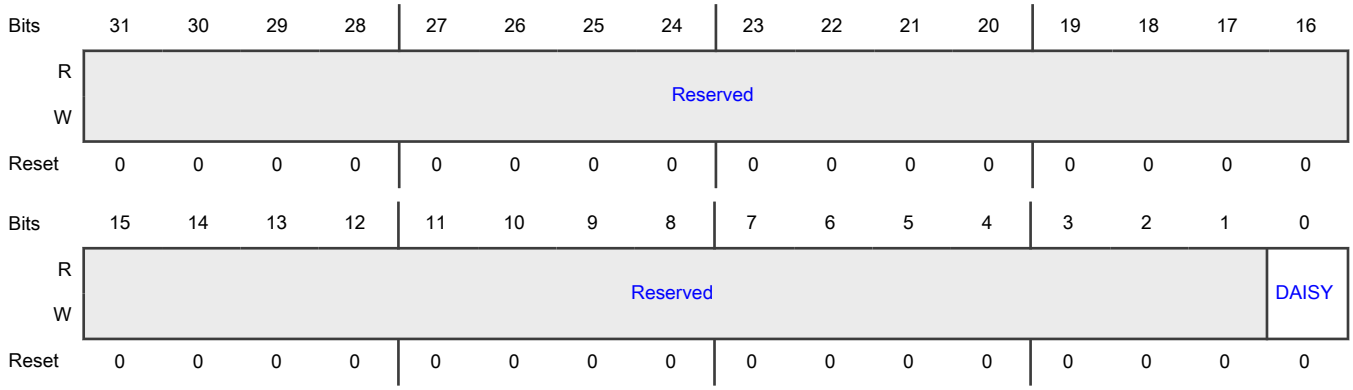
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_2	7ECh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth0_rxd2 0b - Selecting Pad: GPIO_EMC_B2_01 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_16 for Mode: ALT3

17.4.1.460 NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_3 DAISY Register (NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_3)

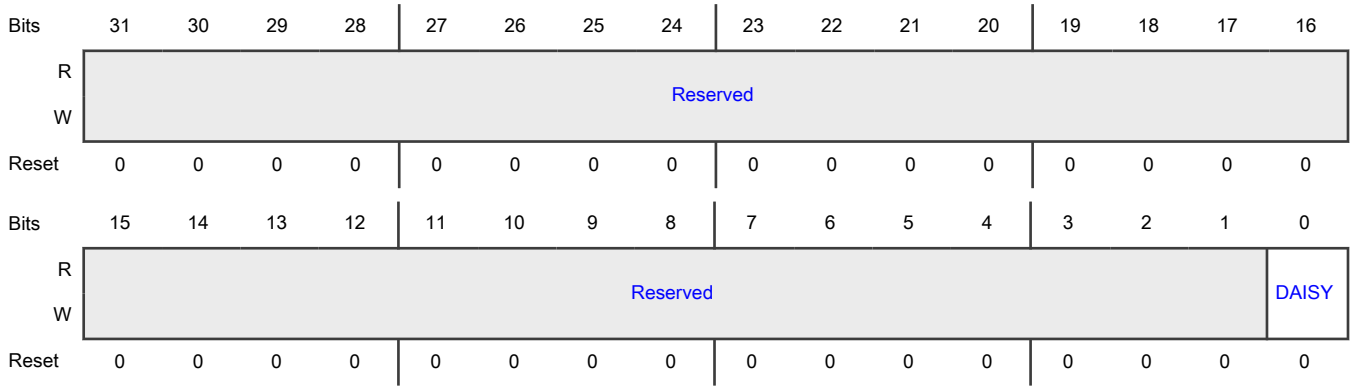
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH0_RXD_SELECT_INPUT_3	7F0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth0_rxd3 0b - Selecting Pad: GPIO_EMC_B2_02 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_17 for Mode: ALT3

17.4.1.461 NETC_PINMUX_IPP_IND_ETH0_TX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH0_TX_CLK_SELECT_INPUT)

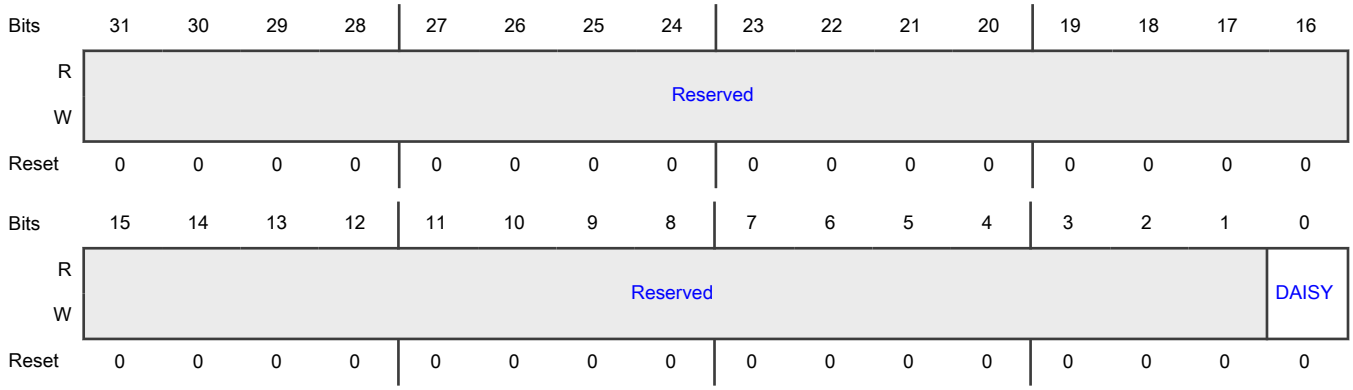
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH0_TX_CLK_SELECT_INPUT	7F4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth0_tx_clk 0b - Selecting Pad: GPIO_EMC_B1_37 for Mode: ALT9 1b - Selecting Pad: GPIO_EMC_B2_08 for Mode: ALT3

17.4.1.462 NETC_PINMUX_IPP_IND_ETH2_RX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RX_CLK_SELECT_INPUT)

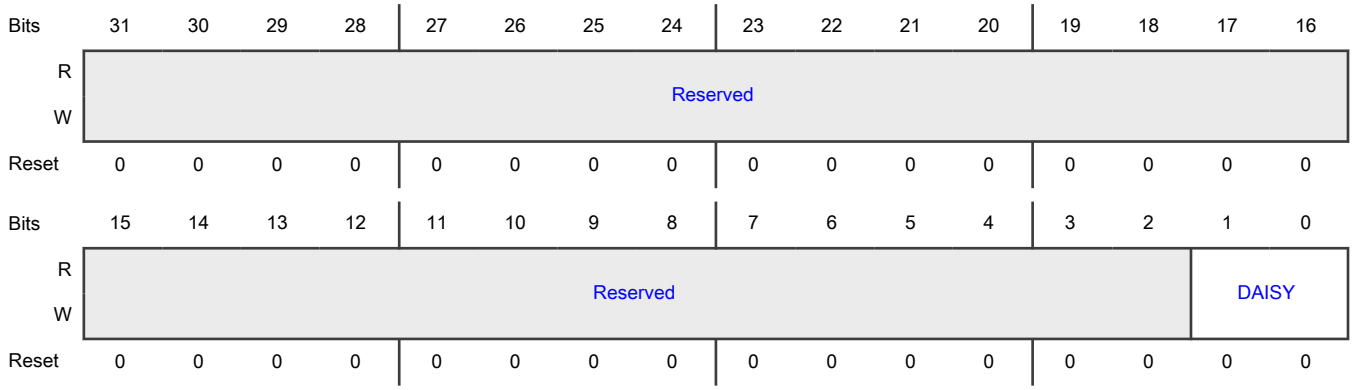
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH2_RX_CLK_SELECT_INPUT	7F8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth2_rx_clk 00b - Selecting Pad: GPIO_EMC_B1_21 for Mode: ALT4 01b - Selecting Pad: GPIO_EMC_B1_33 for Mode: ALT10 10b - Selecting Pad: GPIO_EMC_B1_38 for Mode: ALT4 11b - Selecting Pad: GPIO_B2_13 for Mode: ALT11

17.4.1.463 NETC_PINMUX_IPP_IND_ETH2_RX_DV_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RX_DV_SELECT_INPUT)

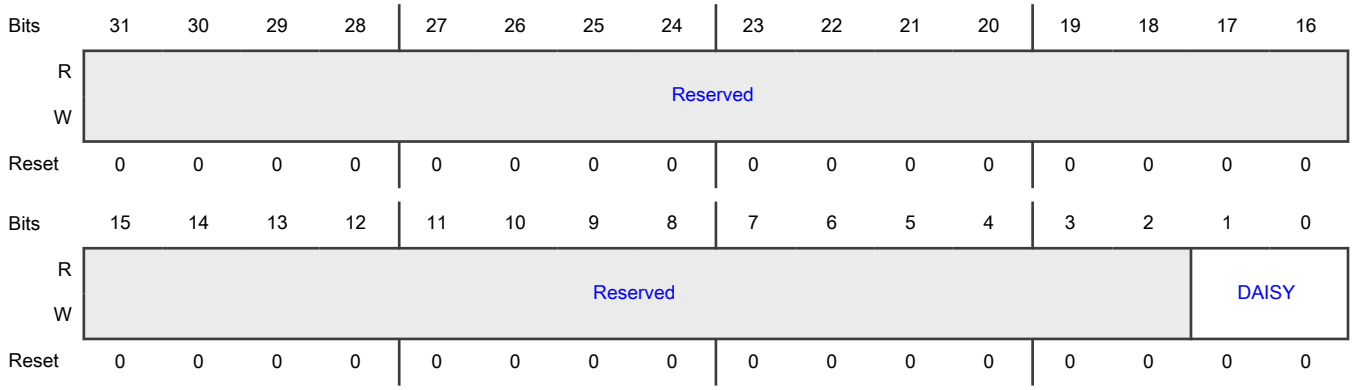
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH2_RX_DV_SELECT_INPUT	7FCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth2_rx_dv 00b - Selecting Pad: GPIO_EMC_B1_13 for Mode: ALT3 01b - Selecting Pad: GPIO_EMC_B1_32 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_12 for Mode: ALT11

17.4.1.464 NETC_PINMUX_IPP_IND_ETH2_RX_ER_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RX_ER_SELECT_INPUT)

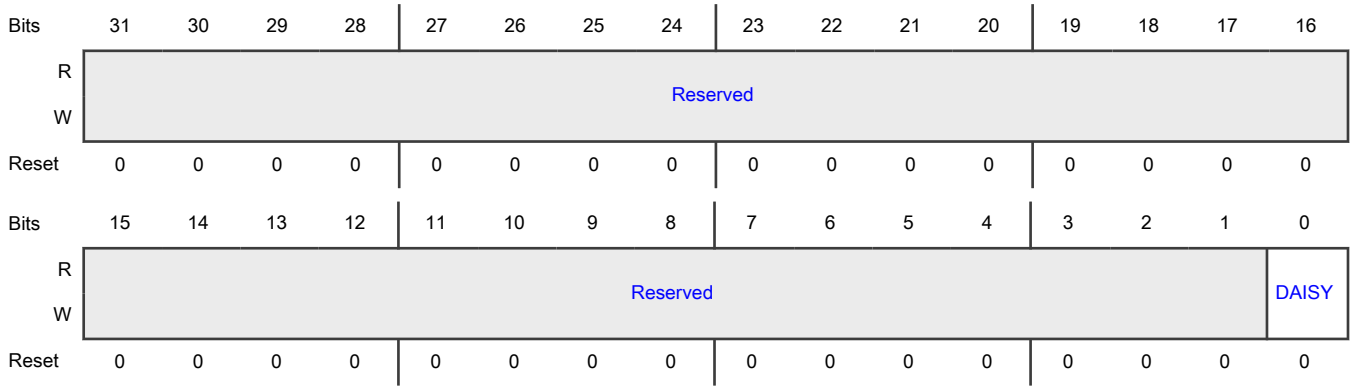
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH2_RX_ER_SELECT_INPUT	800h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth2_rx_er 0b - Selecting Pad: GPIO_EMC_B1_33 for Mode: ALT4 1b - Selecting Pad: GPIO_B2_01 for Mode: ALT11

17.4.1.465 NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_0 DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_0)

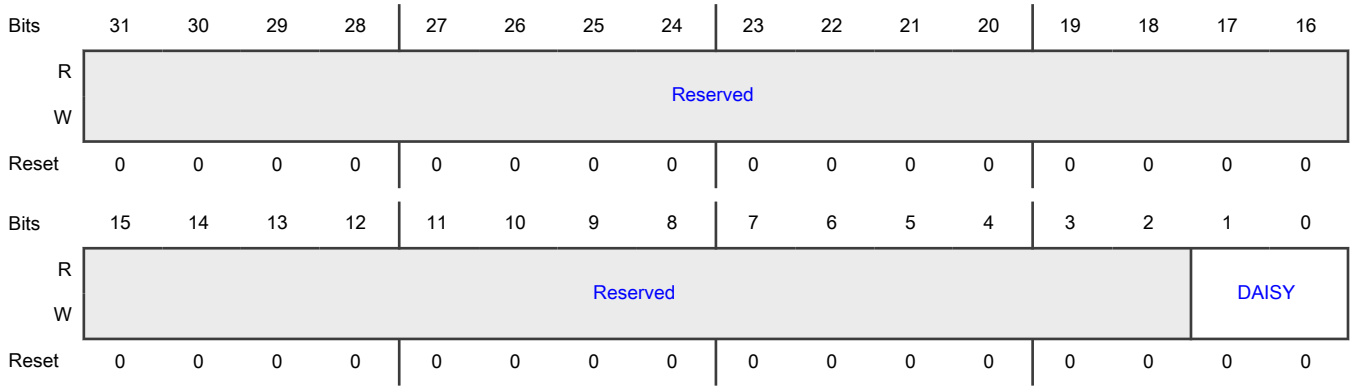
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_0	804h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth2_rxd0 00b - Selecting Pad: GPIO_EMC_B1_16 for Mode: ALT3 01b - Selecting Pad: GPIO_EMC_B1_30 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_10 for Mode: ALT11

17.4.1.466 NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_1 DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_1)

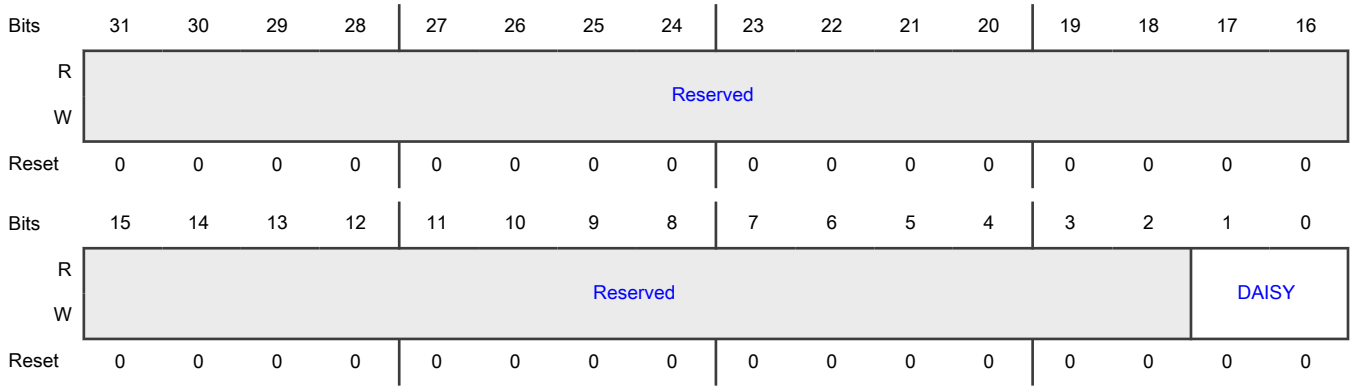
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_1	808h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth2_rxd1 00b - Selecting Pad: GPIO_EMC_B1_17 for Mode: ALT3 01b - Selecting Pad: GPIO_EMC_B1_31 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_11 for Mode: ALT11

17.4.1.467 NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_2 DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_2)

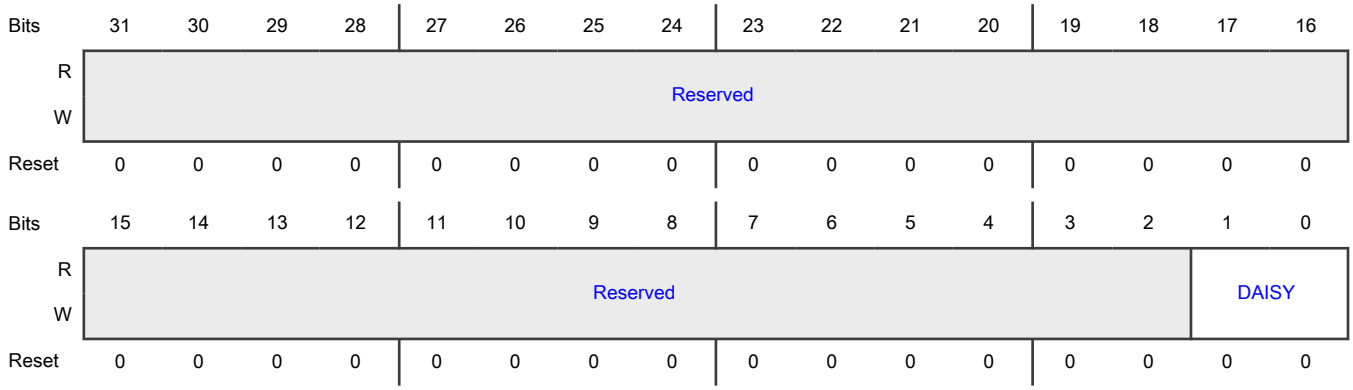
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_2	80Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth2_rxd2 00b - Selecting Pad: GPIO_EMC_B1_22 for Mode: ALT4 01b - Selecting Pad: GPIO_EMC_B1_34 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_02 for Mode: ALT11

17.4.1.468 NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_3 DAISY Register (NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_3)

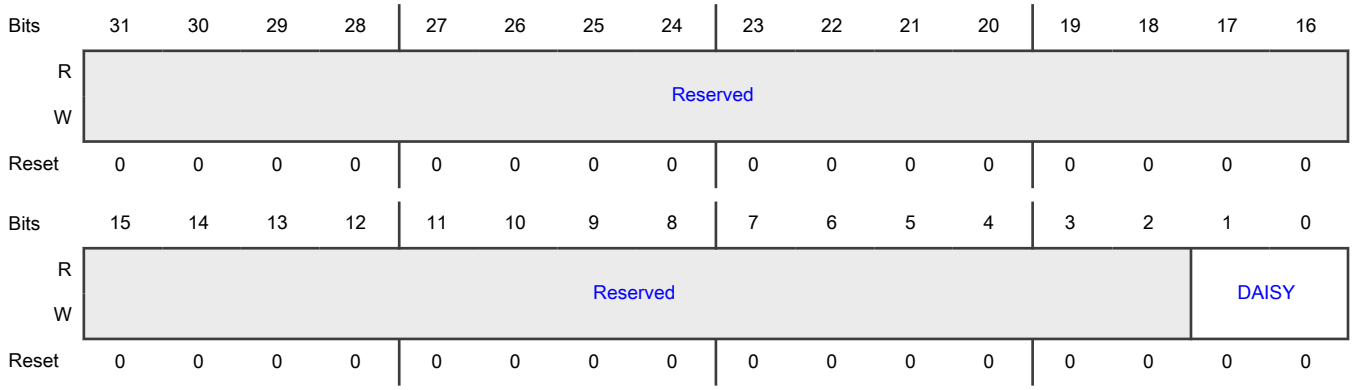
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH2_RXD_SELECT_INPUT_3	810h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth2_rxd3 00b - Selecting Pad: GPIO_EMC_B1_23 for Mode: ALT4 01b - Selecting Pad: GPIO_EMC_B1_35 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_03 for Mode: ALT11

17.4.1.469 NETC_PINMUX_IPP_IND_ETH2_TX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH2_TX_CLK_SELECT_INPUT)

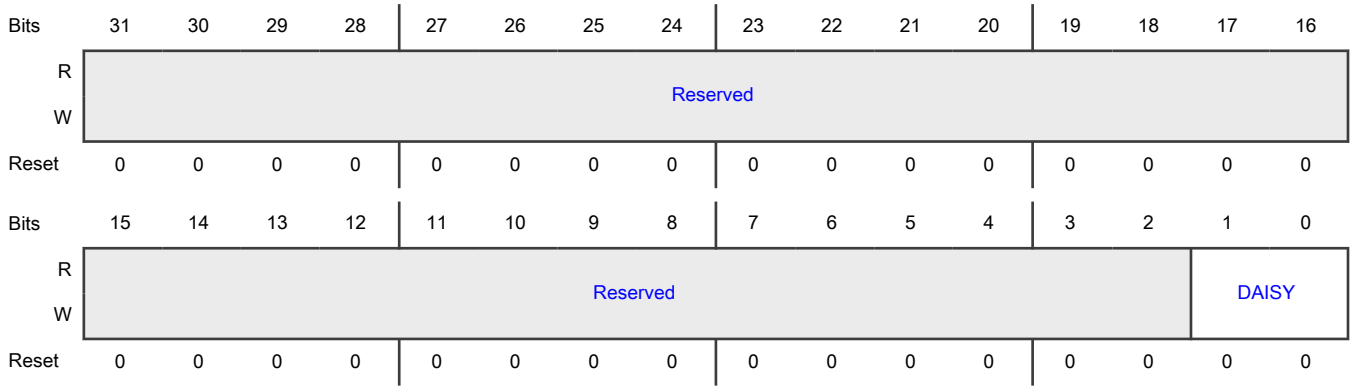
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH2_TX_CLK_SELECT_INPUT	814h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth2_tx_clk 00b - Selecting Pad: GPIO_EMC_B1_15 for Mode: ALT3 01b - Selecting Pad: GPIO_EMC_B1_29 for Mode: ALT4 10b - Selecting Pad: GPIO_B2_09 for Mode: ALT11

17.4.1.470 NETC_PINMUX_IPP_IND_ETH3_RX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RX_CLK_SELECT_INPUT)

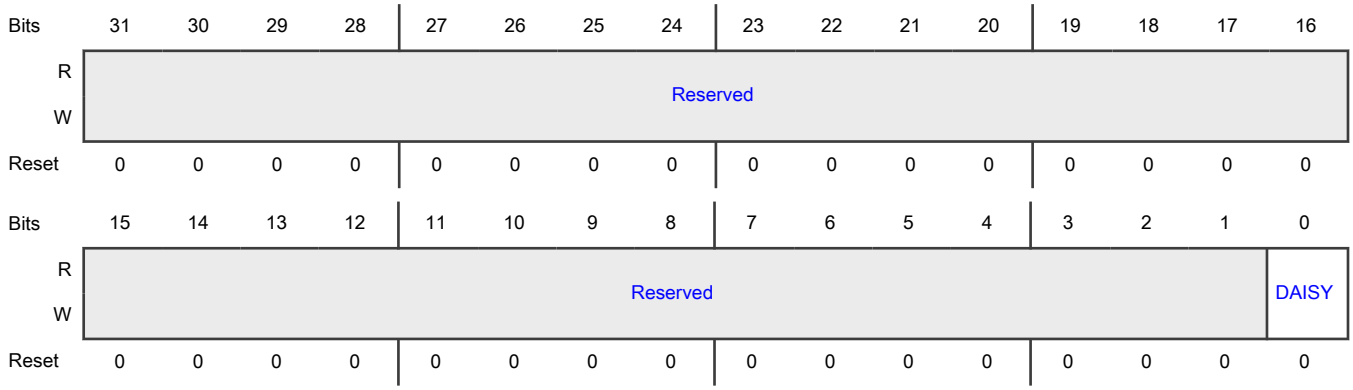
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH3_RX_CLK_SELECT_INPUT	818h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth3_rx_clk 0b - Selecting Pad: GPIO_EMC_B1_02 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_15 for Mode: ALT9

17.4.1.471 NETC_PINMUX_IPP_IND_ETH3_RX_DV_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RX_DV_SELECT_INPUT)

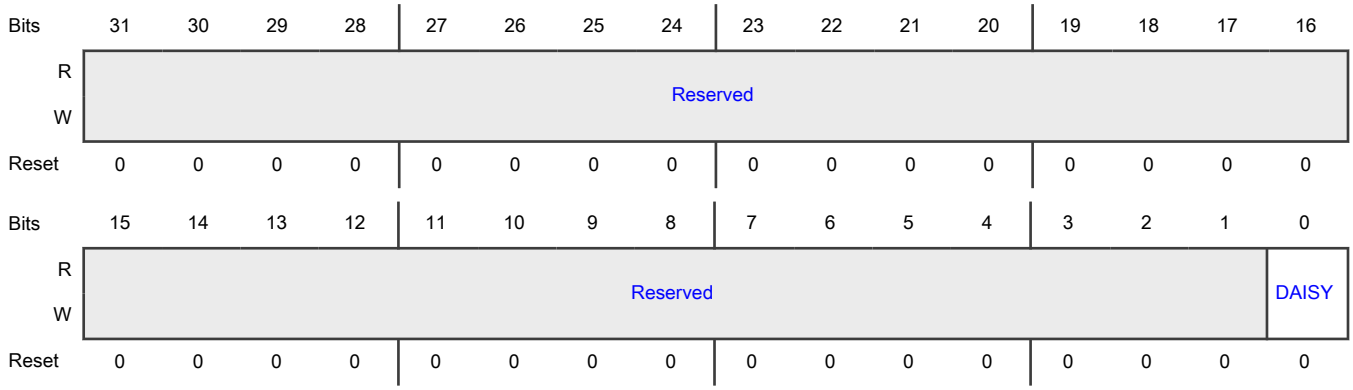
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH3_RX_DV_SELECT_INPUT	81Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth3_rx_dv 0b - Selecting Pad: GPIO_EMC_B1_11 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_11 for Mode: ALT9

17.4.1.472 NETC_PINMUX_IPP_IND_ETH3_RX_ER_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RX_ER_SELECT_INPUT)

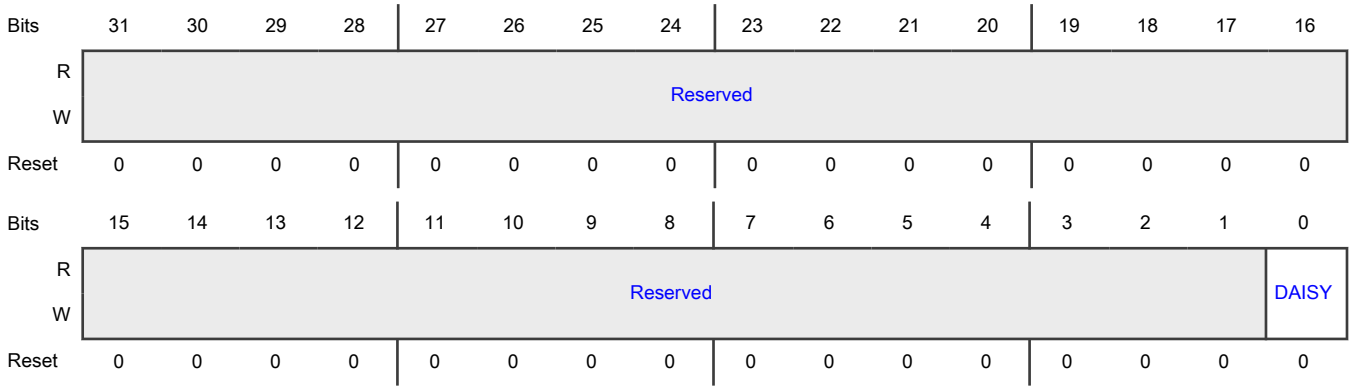
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH3_RX_ER_SELECT_INPUT	820h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth3_rx_er 0b - Selecting Pad: GPIO_EMC_B1_12 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_12 for Mode: ALT9

17.4.1.473 NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_0 DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_0)

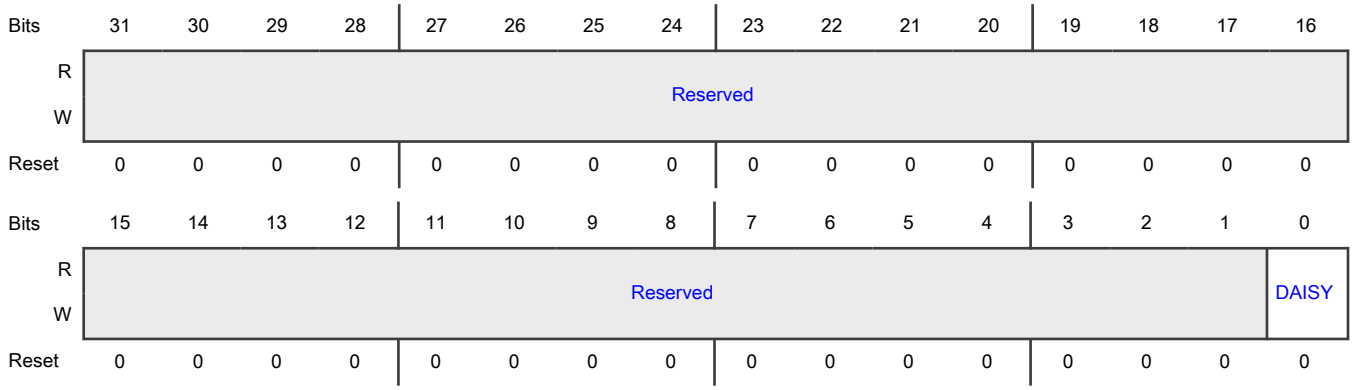
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_0	824h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth3_rxd0 0b - Selecting Pad: GPIO_EMC_B1_09 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_09 for Mode: ALT9

17.4.1.474 NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_1 DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_1)

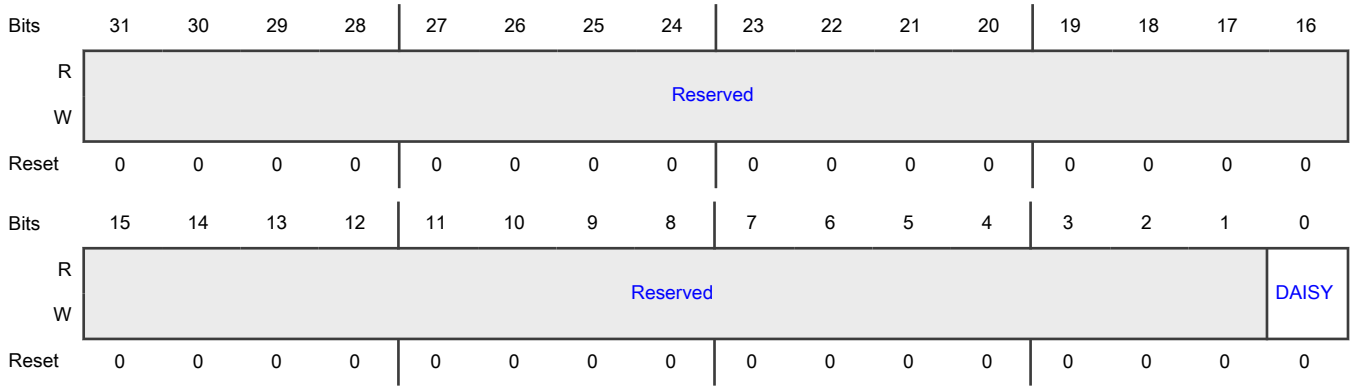
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_1	828h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth3_rxd1 0b - Selecting Pad: GPIO_EMC_B1_10 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_10 for Mode: ALT9

17.4.1.475 NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_2 DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_2)

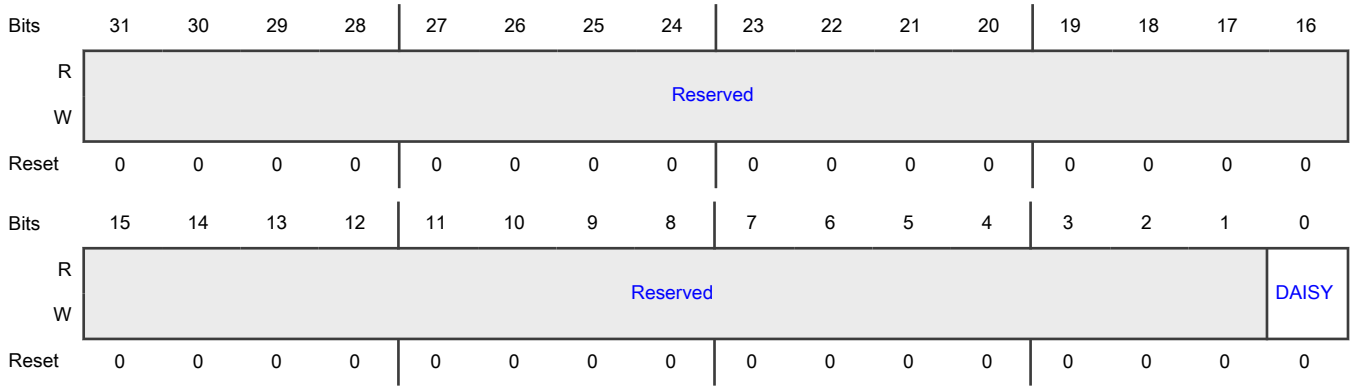
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_2	82Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth3_rxd2 0b - Selecting Pad: GPIO_EMC_B1_04 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_16 for Mode: ALT9

17.4.1.476 NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_3 DAISY Register (NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_3)

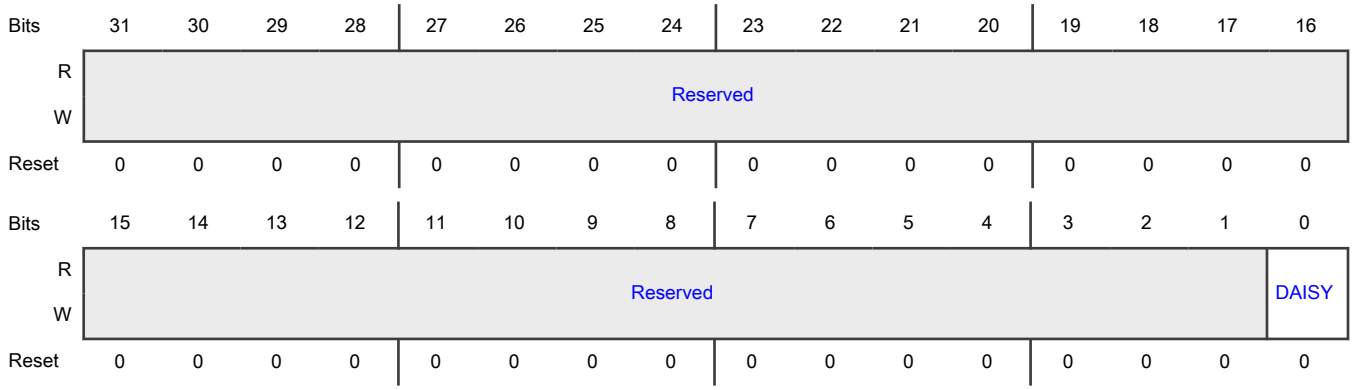
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH3_RXD_SELECT_INPUT_3	830h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth3_rxd3 0b - Selecting Pad: GPIO_EMC_B1_03 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_17 for Mode: ALT9

17.4.1.477 NETC_PINMUX_IPP_IND_ETH3_TX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH3_TX_CLK_SELECT_INPUT)

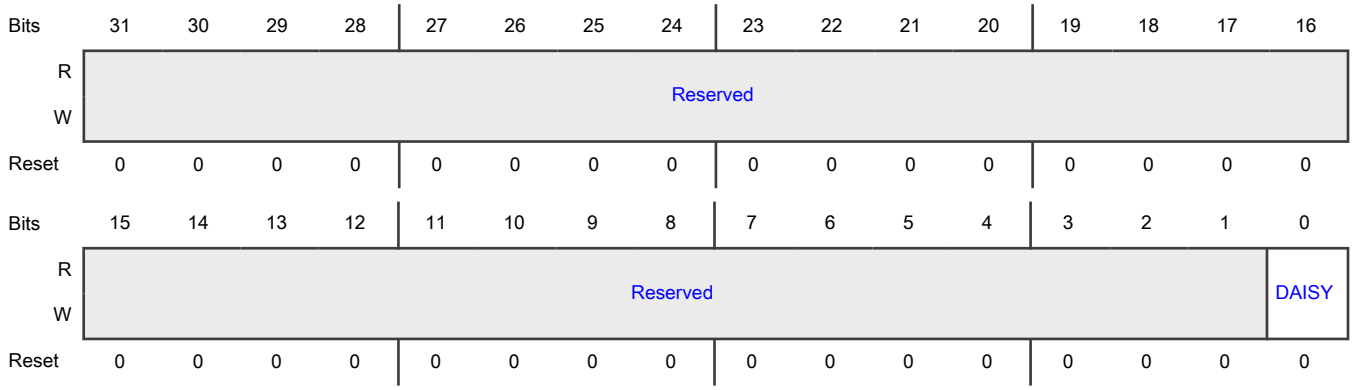
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH3_TX_CLK_SELECT_INPUT	834h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth3_tx_clk 0b - Selecting Pad: GPIO_EMC_B1_08 for Mode: ALT4 1b - Selecting Pad: GPIO_EMC_B2_08 for Mode: ALT9

17.4.1.478 NETC_PINMUX_IPP_IND_ETH4_RX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RX_CLK_SELECT_INPUT)

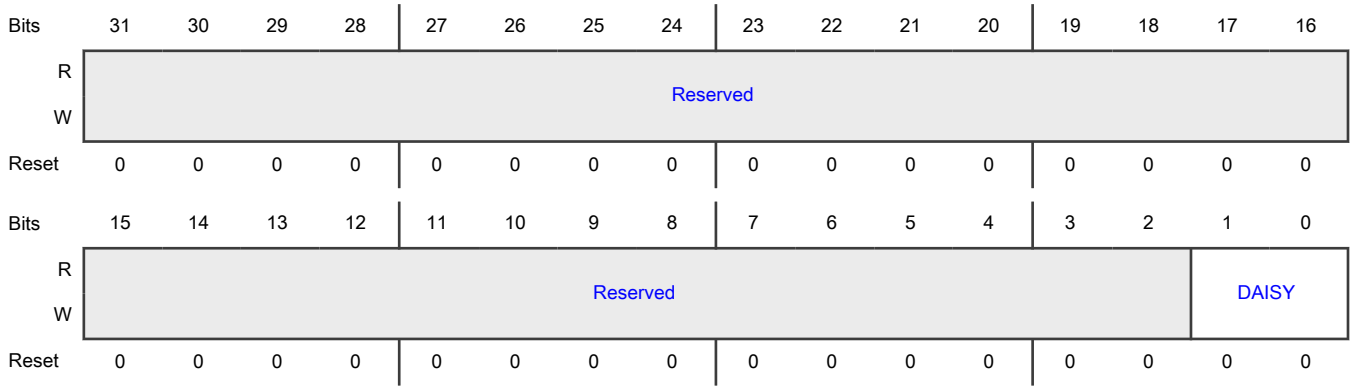
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH4_RX_CLK_SELECT_INPUT	838h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth4_rx_clk 00b - Selecting Pad: GPIO_EMC_B1_02 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_08 for Mode: ALT1 10b - Selecting Pad: GPIO_B1_11 for Mode: ALT8

17.4.1.479 NETC_PINMUX_IPP_IND_ETH4_RX_DV_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RX_DV_SELECT_INPUT)

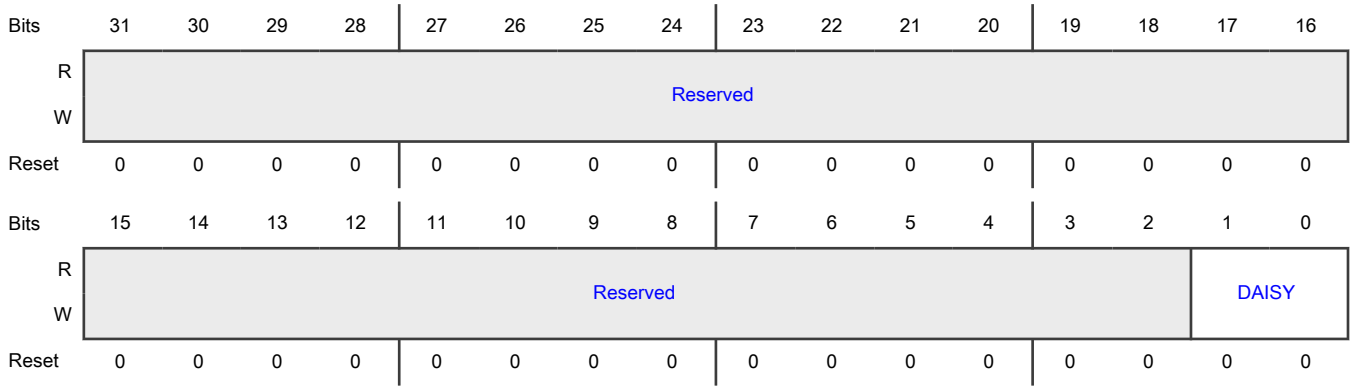
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH4_RX_DV_SELECT_INPUT	83Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth4_rx_dv 00b - Selecting Pad: GPIO_EMC_B1_11 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_19 for Mode: ALT1 10b - Selecting Pad: GPIO_B1_06 for Mode: ALT8

17.4.1.480 NETC_PINMUX_IPP_IND_ETH4_RX_ER_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RX_ER_SELECT_INPUT)

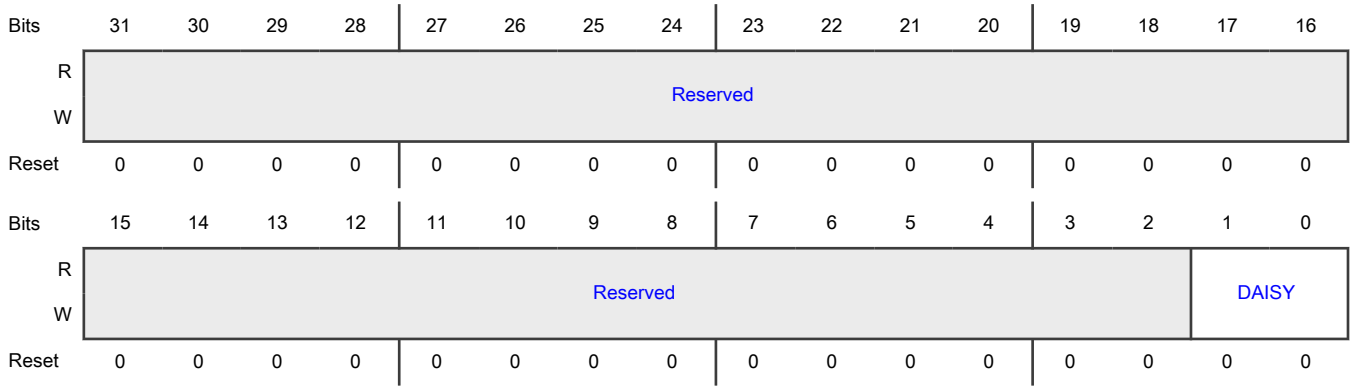
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH4_RX_ER_SELECT_INPUT	840h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth4_rx_er 00b - Selecting Pad: GPIO_EMC_B1_12 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_20 for Mode: ALT1 10b - Selecting Pad: GPIO_B1_12 for Mode: ALT8

17.4.1.481 NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_0 DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_0)

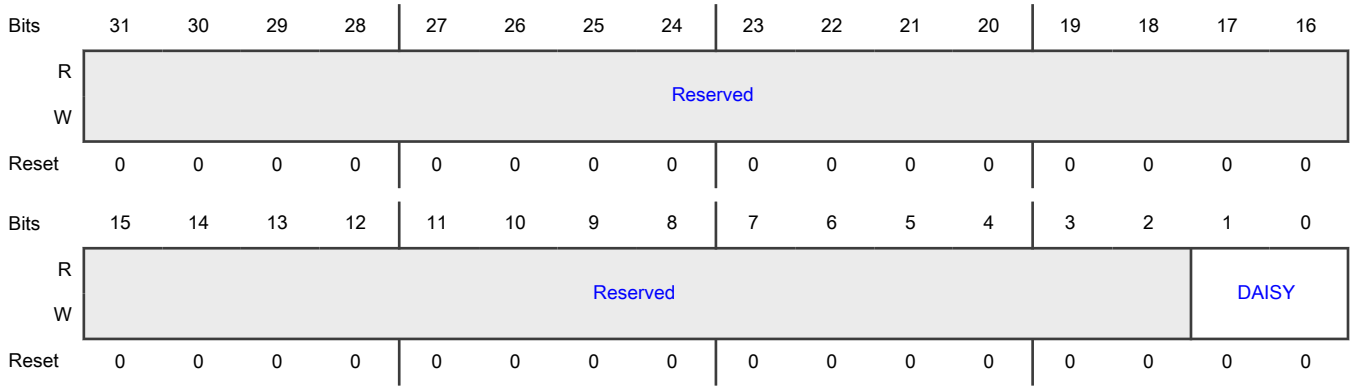
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_0	844h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth4_rxd0 00b - Selecting Pad: GPIO_EMC_B1_09 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_17 for Mode: ALT1 10b - Selecting Pad: GPIO_B1_04 for Mode: ALT8

17.4.1.482 NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_1 DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_1)

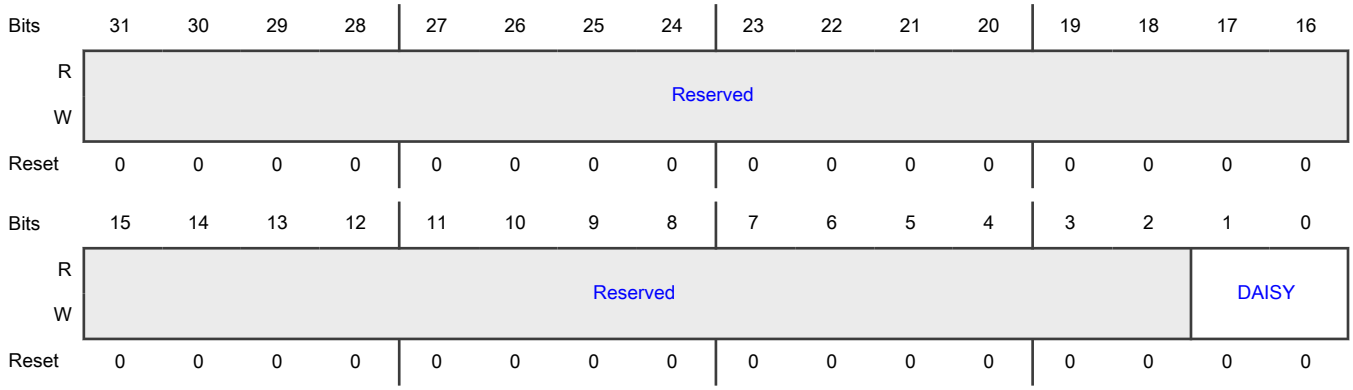
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_1	848h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth4_rxd1 00b - Selecting Pad: GPIO_EMC_B1_10 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_18 for Mode: ALT1 10b - Selecting Pad: GPIO_B1_05 for Mode: ALT8

17.4.1.483 NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_2 DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_2)

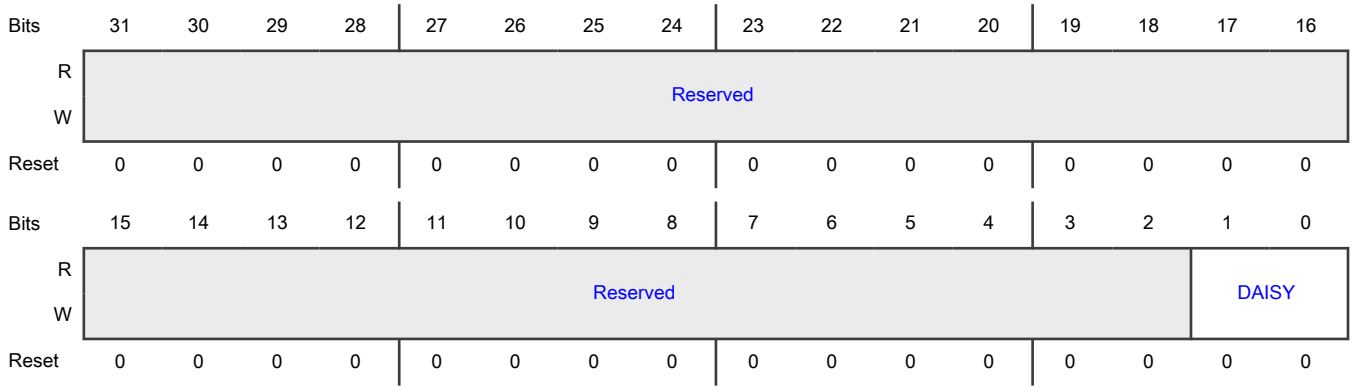
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_2	84Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth4_rxd2 00b - Selecting Pad: GPIO_EMC_B1_04 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_10 for Mode: ALT1 10b - Selecting Pad: GPIO_B1_09 for Mode: ALT8

17.4.1.484 NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_3 DAISY Register (NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_3)

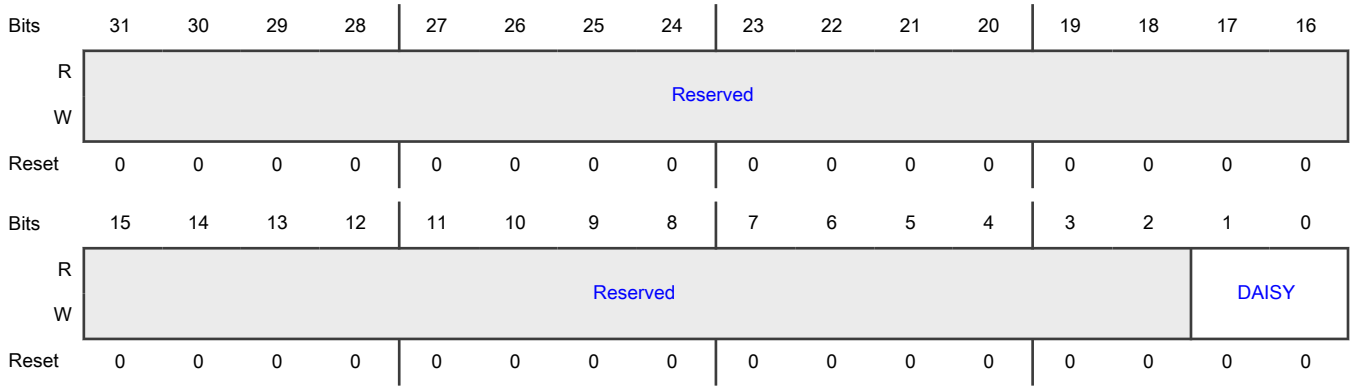
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH4_RXD_SELECT_INPUT_3	850h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth4_rxd3 00b - Selecting Pad: GPIO_EMC_B1_03 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_09 for Mode: ALT1 10b - Selecting Pad: GPIO_B1_10 for Mode: ALT8

17.4.1.485 NETC_PINMUX_IPP_IND_ETH4_TX_CLK_SELECT_INPUT DAISY Register (NETC_PINMUX_IPP_IND_ETH4_TX_CLK_SELECT_INPUT)

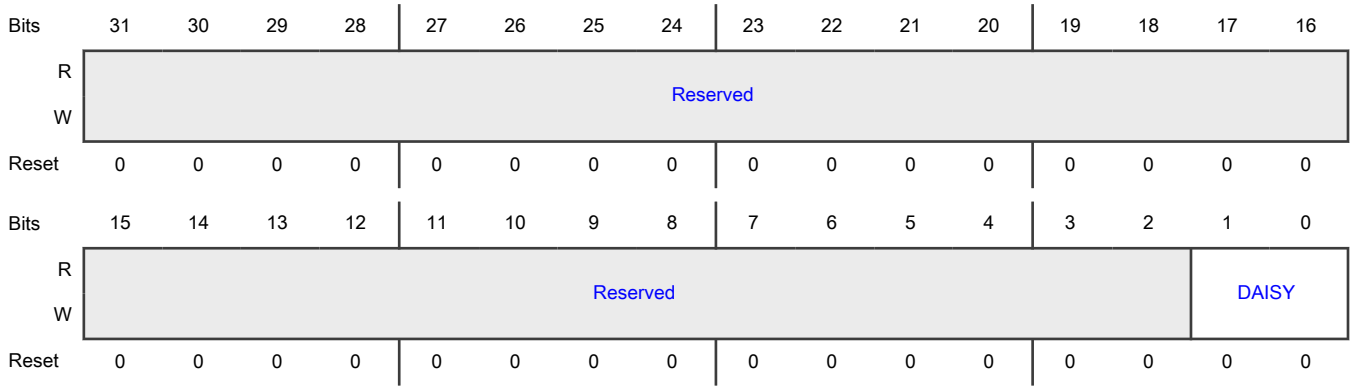
Offset

Register	Offset
NETC_PINMUX_IPP_IND_ETH4_TX_CLK_SELECT_INPUT	854h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: netc_pinmux, In Pin: ipp_ind_eth4_tx_clk 00b - Selecting Pad: GPIO_EMC_B1_08 for Mode: ALT9 01b - Selecting Pad: GPIO_EMC_B2_16 for Mode: ALT1 10b - Selecting Pad: GPIO_B1_03 for Mode: ALT8

17.4.1.486 QTIMER1_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER1_TMR0_INPUT_SELECT_INPUT)

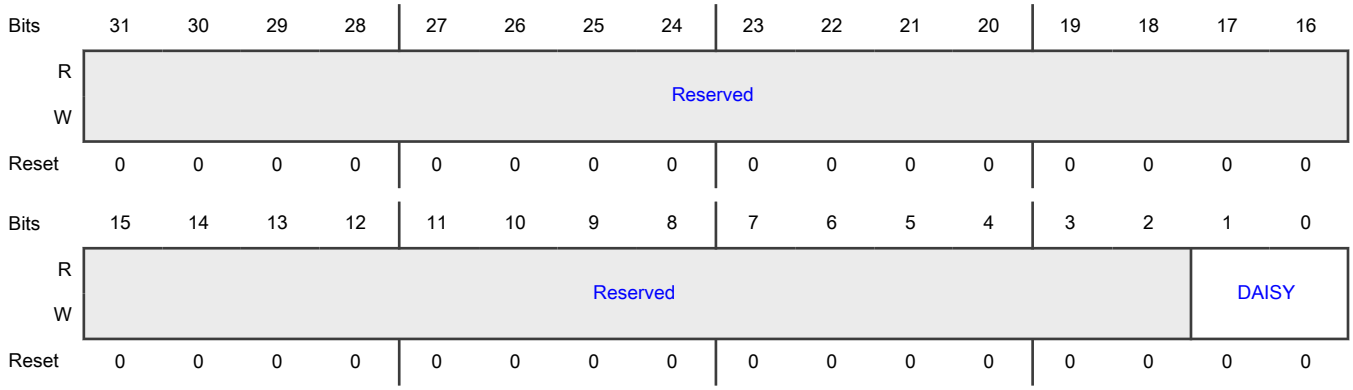
Offset

Register	Offset
QTIMER1_TMR0_INPUT_SELECT_INPUT	858h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer1, In Pin: tmr0_input 00b - Selecting Pad: GPIO_EMC_B1_18 for Mode: ALT2 01b - Selecting Pad: GPIO_EMC_B2_09 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_00 for Mode: ALT3

17.4.1.487 QTIMER1_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER1_TMR1_INPUT_SELECT_INPUT)

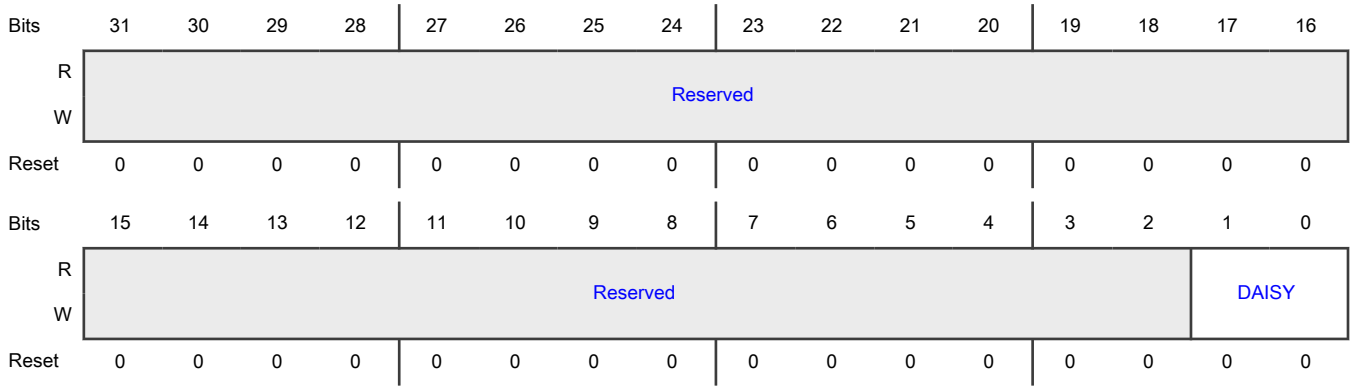
Offset

Register	Offset
QTIMER1_TMR1_INPUT_SELECT_INPUT	85Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer1, In Pin: tmr1_input 00b - Selecting Pad: GPIO_EMC_B1_13 for Mode: ALT10 01b - Selecting Pad: GPIO_EMC_B2_10 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_01 for Mode: ALT3

17.4.1.488 QTIMER1_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER1_TMR2_INPUT_SELECT_INPUT)

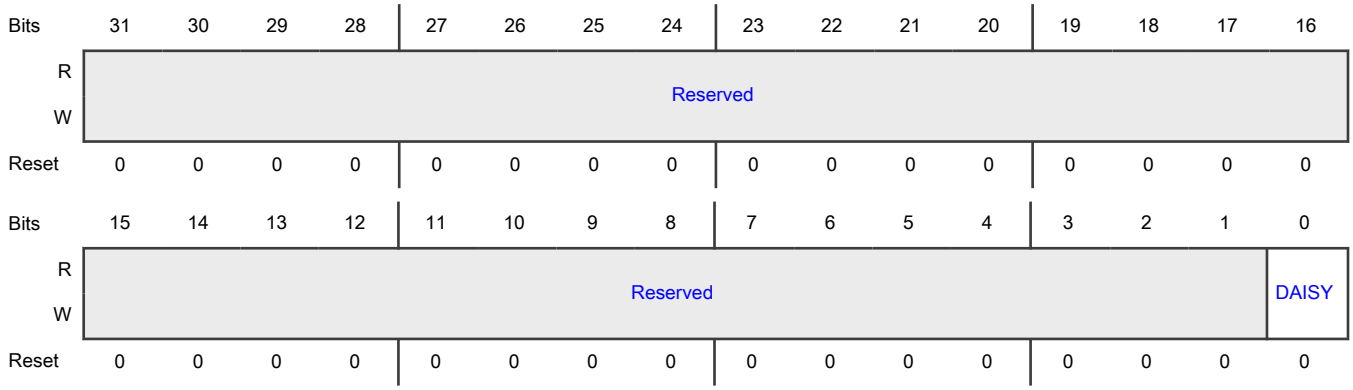
Offset

Register	Offset
QTIMER1_TMR2_INPUT_SELECT_INPUT	860h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer1, In Pin: tmr2_input 0b - Selecting Pad: GPIO_EMC_B2_11 for Mode: ALT10 1b - Selecting Pad: GPIO_B1_02 for Mode: ALT3

17.4.1.489 QTIMER2_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER2_TMR0_INPUT_SELECT_INPUT)

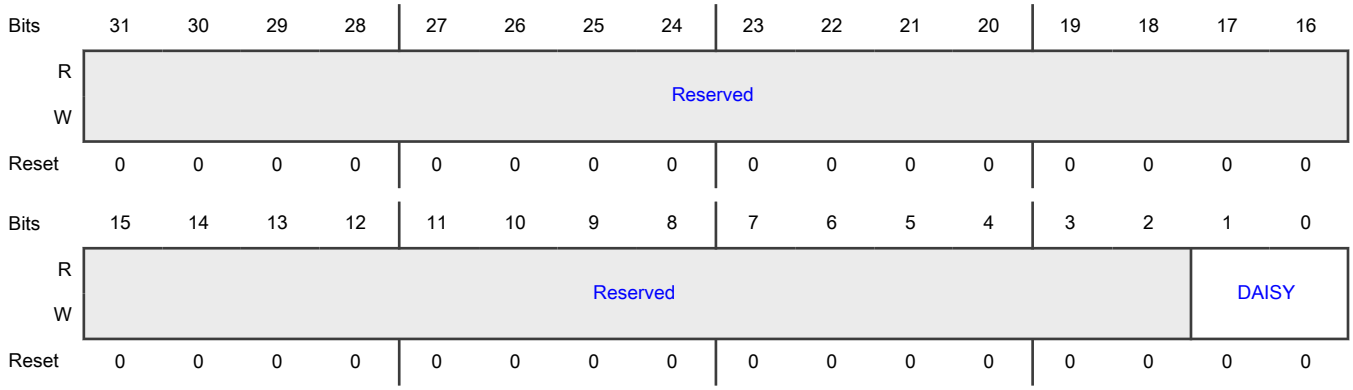
Offset

Register	Offset
QTIMER2_TMR0_INPUT_SELECT_INPUT	864h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer2, In Pin: tmr0_input 00b - Selecting Pad: GPIO_EMC_B1_19 for Mode: ALT2 01b - Selecting Pad: GPIO_EMC_B2_13 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_03 for Mode: ALT3

17.4.1.490 QTIMER2_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER2_TMR1_INPUT_SELECT_INPUT)

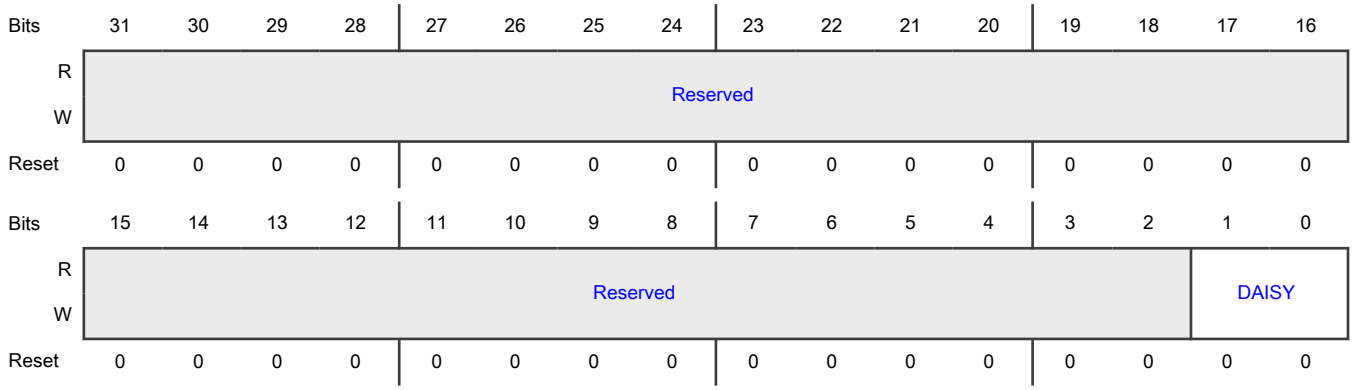
Offset

Register	Offset
QTIMER2_TMR1_INPUT_SELECT_INPUT	868h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer2, In Pin: tmr1_input 00b - Selecting Pad: GPIO_EMC_B1_39 for Mode: ALT6 01b - Selecting Pad: GPIO_EMC_B2_14 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_04 for Mode: ALT3

17.4.1.491 QTIMER2_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER2_TMR2_INPUT_SELECT_INPUT)

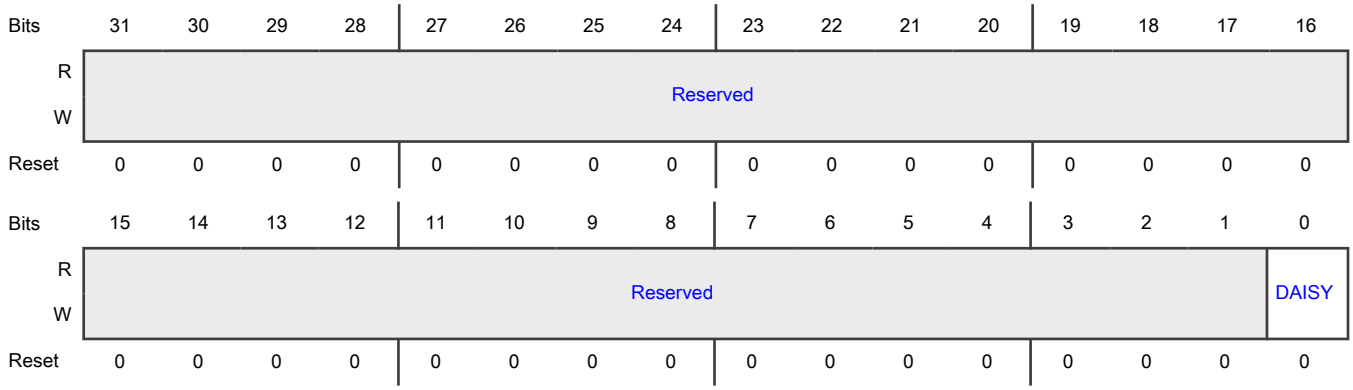
Offset

Register	Offset
QTIMER2_TMR2_INPUT_SELECT_INPUT	86Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer2, In Pin: tmr2_input 0b - Selecting Pad: GPIO_EMC_B2_15 for Mode: ALT10 1b - Selecting Pad: GPIO_B1_05 for Mode: ALT3

17.4.1.492 QTIMER3_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER3_TMR0_INPUT_SELECT_INPUT)

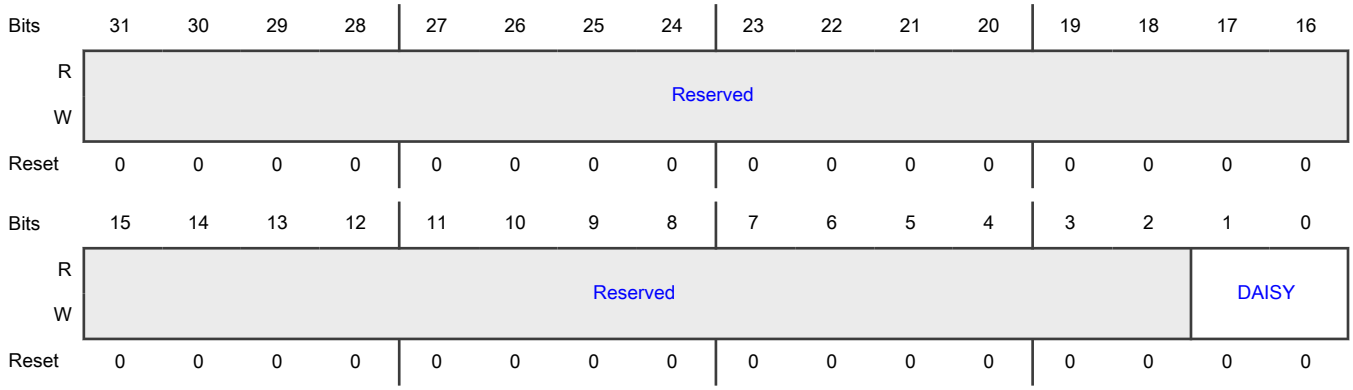
Offset

Register	Offset
QTIMER3_TMR0_INPUT_SELECT_INPUT	870h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer3, In Pin: tmr0_input 00b - Selecting Pad: GPIO_EMC_B1_20 for Mode: ALT2 01b - Selecting Pad: GPIO_EMC_B2_17 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_06 for Mode: ALT3

17.4.1.493 QTIMER3_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER3_TMR1_INPUT_SELECT_INPUT)

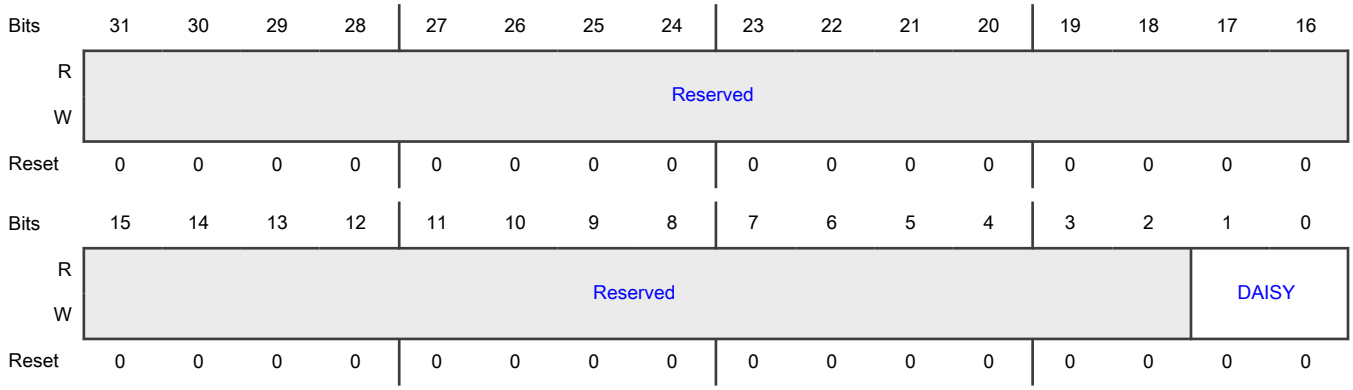
Offset

Register	Offset
QTIMER3_TMR1_INPUT_SELECT_INPUT	874h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer3, In Pin: tmr1_input 00b - Selecting Pad: GPIO_EMC_B1_40 for Mode: ALT6 01b - Selecting Pad: GPIO_EMC_B2_18 for Mode: ALT10 10b - Selecting Pad: GPIO_B1_07 for Mode: ALT3

17.4.1.494 QTIMER3_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER3_TMR2_INPUT_SELECT_INPUT)

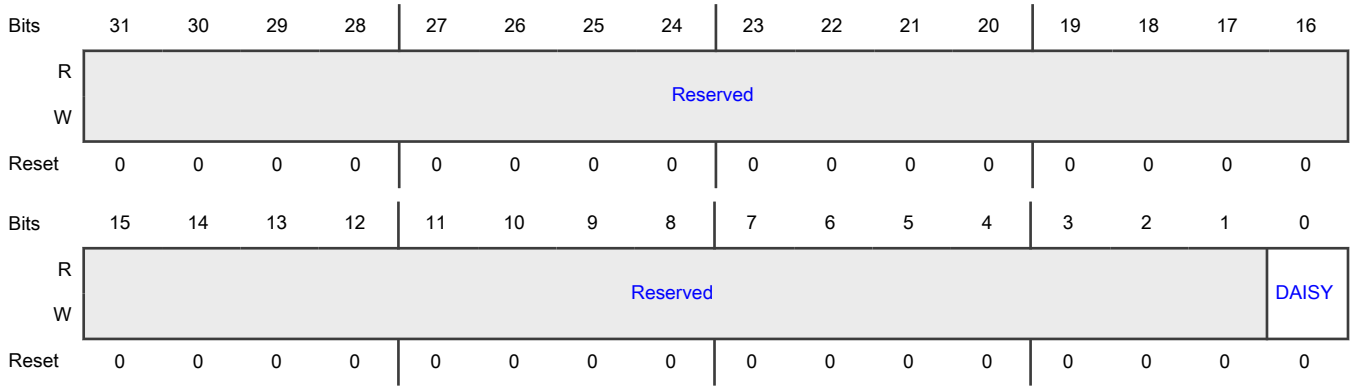
Offset

Register	Offset
QTIMER3_TMR2_INPUT_SELECT_INPUT	878h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer3, In Pin: tmr2_input 0b - Selecting Pad: GPIO_EMC_B2_19 for Mode: ALT10 1b - Selecting Pad: GPIO_B1_08 for Mode: ALT3

17.4.1.495 QTIMER4_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER4_TMR0_INPUT_SELECT_INPUT)

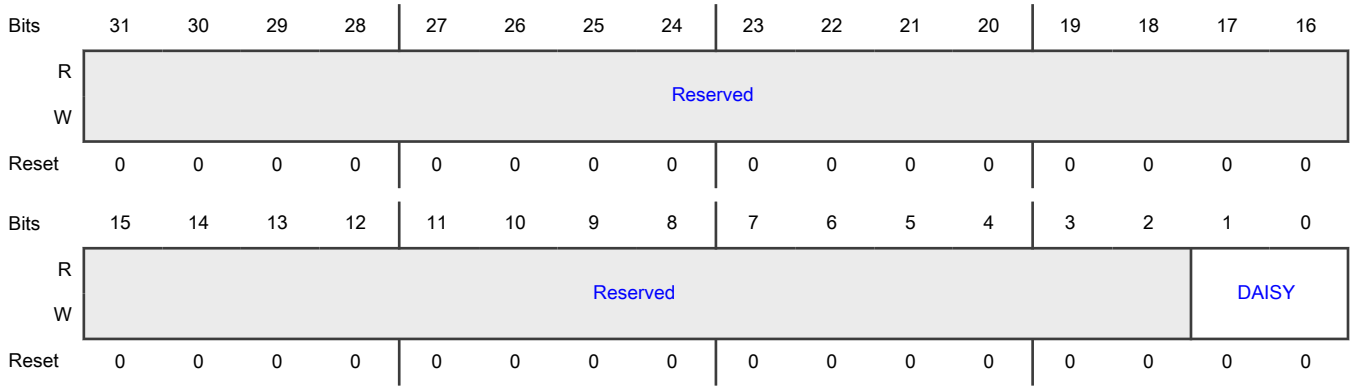
Offset

Register	Offset
QTIMER4_TMR0_INPUT_SELECT_INPUT	87Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer4, In Pin: tmr0_input 00b - Selecting Pad: GPIO_EMC_B1_21 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_00 for Mode: ALT9 10b - Selecting Pad: GPIO_B1_09 for Mode: ALT3

17.4.1.496 QTIMER4_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER4_TMR1_INPUT_SELECT_INPUT)

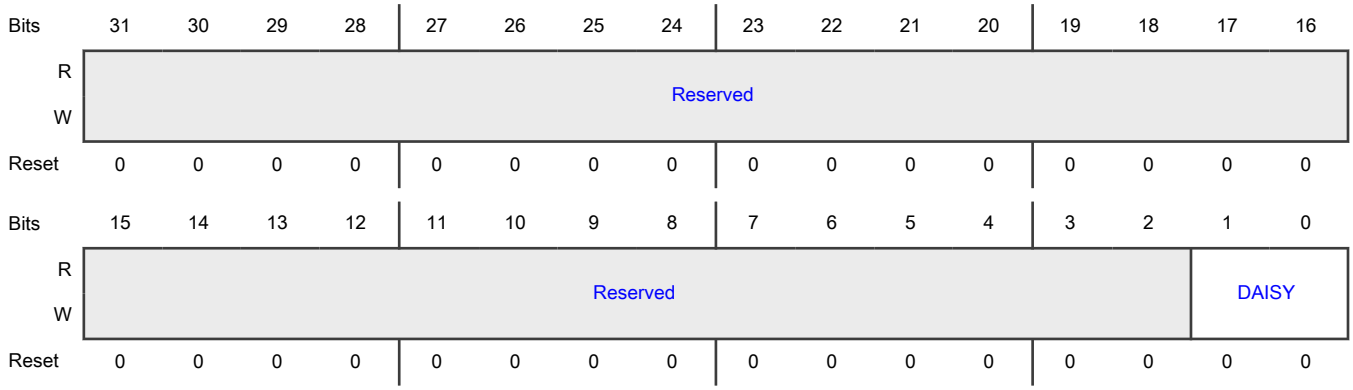
Offset

Register	Offset
QTIMER4_TMR1_INPUT_SELECT_INPUT	880h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer4, In Pin: tmr1_input 00b - Selecting Pad: GPIO_EMC_B1_41 for Mode: ALT6 01b - Selecting Pad: GPIO_AD_01 for Mode: ALT9 10b - Selecting Pad: GPIO_B1_10 for Mode: ALT3

17.4.1.497 QTIMER4_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER4_TMR2_INPUT_SELECT_INPUT)

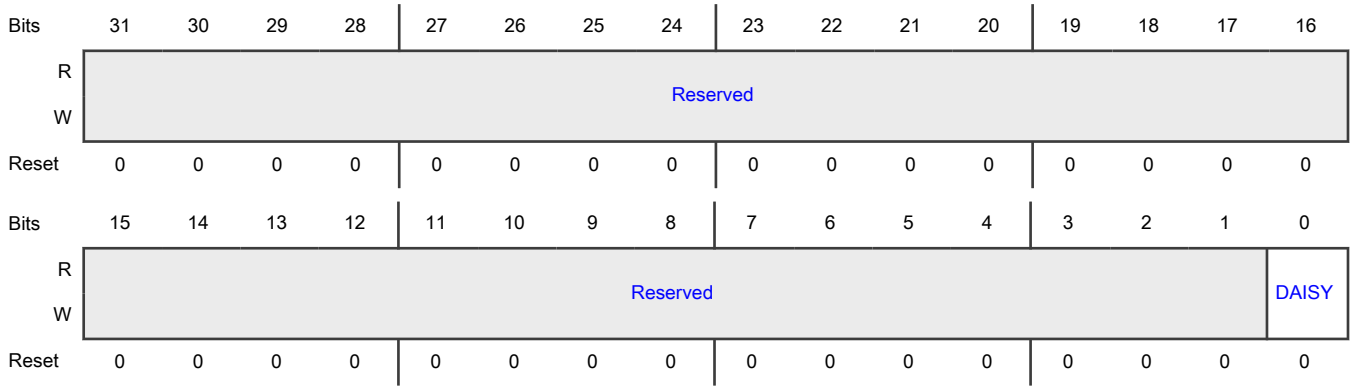
Offset

Register	Offset
QTIMER4_TMR2_INPUT_SELECT_INPUT	884h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer4, In Pin: tmr2_input 0b - Selecting Pad: GPIO_AD_02 for Mode: ALT9 1b - Selecting Pad: GPIO_B1_11 for Mode: ALT3

17.4.1.498 QTIMER5_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER5_TMR0_INPUT_SELECT_INPUT)

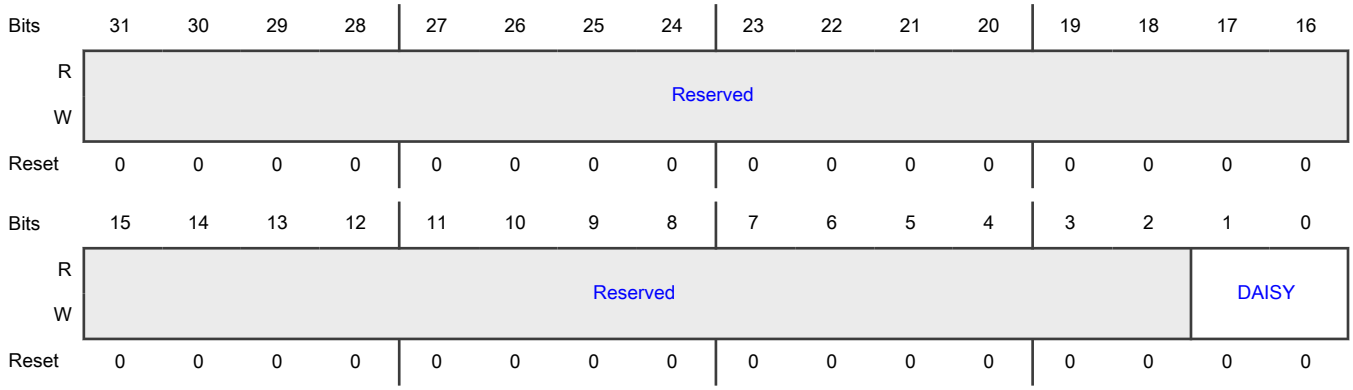
Offset

Register	Offset
QTIMER5_TMR0_INPUT_SELECT_INPUT	888h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer5, In Pin: tmr0_input 00b - Selecting Pad: GPIO_EMC_B1_22 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_04 for Mode: ALT9 10b - Selecting Pad: GPIO_B2_07 for Mode: ALT0

17.4.1.499 QTIMER5_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER5_TMR1_INPUT_SELECT_INPUT)

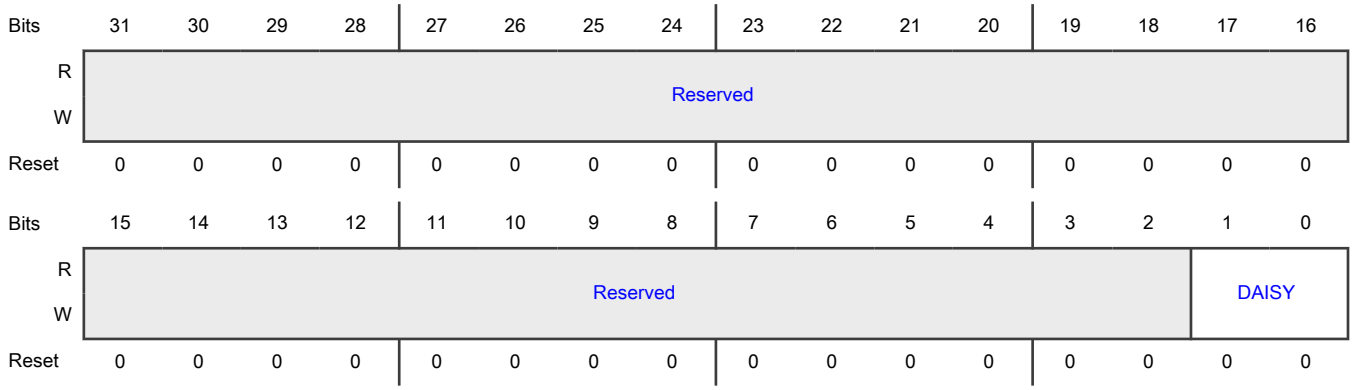
Offset

Register	Offset
QTIMER5_TMR1_INPUT_SELECT_INPUT	88Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer5, In Pin: tmr1_input 00b - Selecting Pad: GPIO_EMC_B2_00 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_05 for Mode: ALT9 10b - Selecting Pad: GPIO_B2_08 for Mode: ALT0

17.4.1.500 QTIMER5_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER5_TMR2_INPUT_SELECT_INPUT)

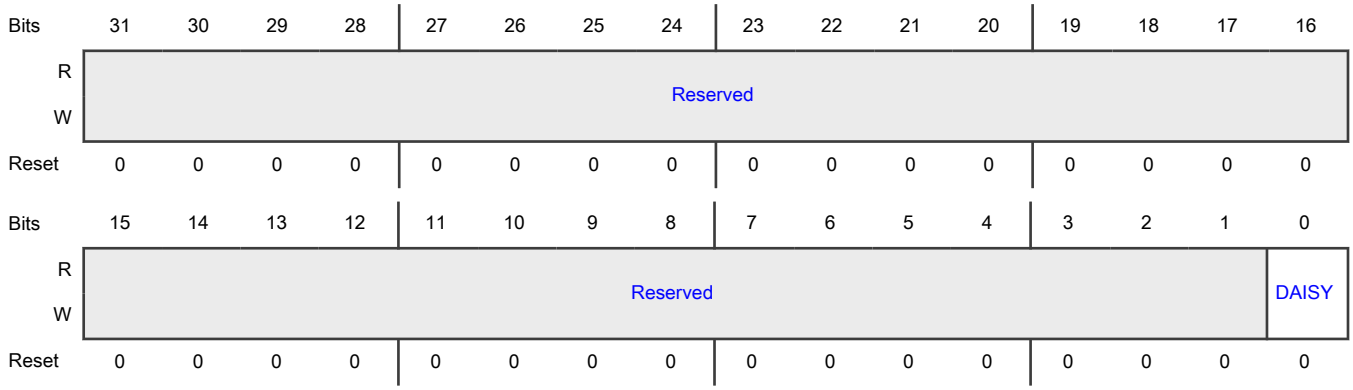
Offset

Register	Offset
QTIMER5_TMR2_INPUT_SELECT_INPUT	890h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer5, In Pin: tmr2_input 0b - Selecting Pad: GPIO_AD_06 for Mode: ALT9 1b - Selecting Pad: GPIO_B2_09 for Mode: ALT0

17.4.1.501 QTIMER6_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER6_TMR0_INPUT_SELECT_INPUT)

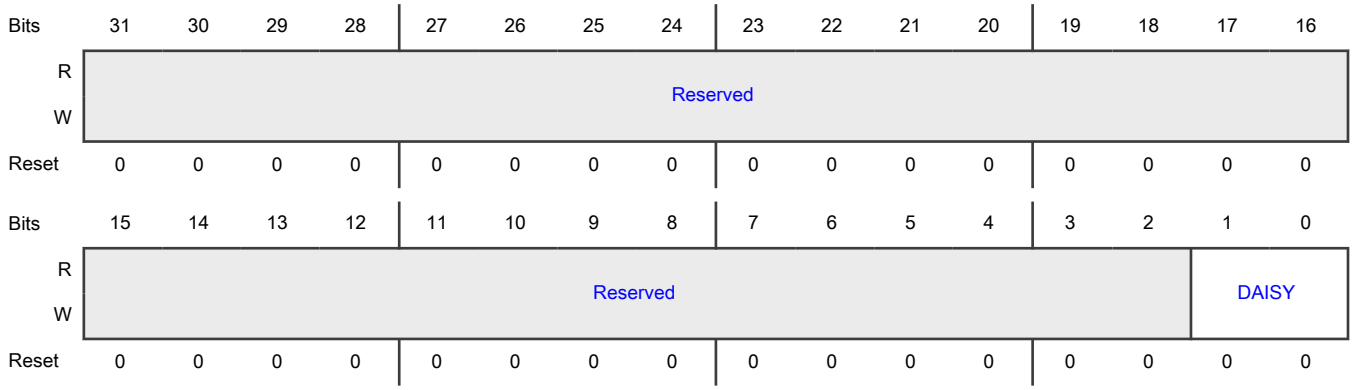
Offset

Register	Offset
QTIMER6_TMR0_INPUT_SELECT_INPUT	894h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer6, In Pin: tmr0_input 00b - Selecting Pad: GPIO_EMC_B1_23 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_08 for Mode: ALT9 10b - Selecting Pad: GPIO_SD_B2_01 for Mode: ALT3

17.4.1.502 QTIMER6_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER6_TMR1_INPUT_SELECT_INPUT)

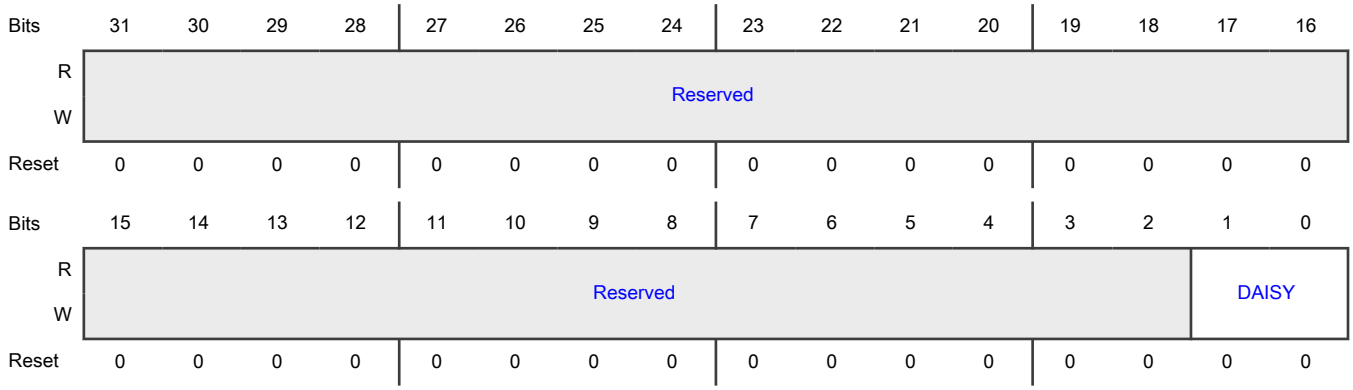
Offset

Register	Offset
QTIMER6_TMR1_INPUT_SELECT_INPUT	898h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer6, In Pin: tmr1_input 00b - Selecting Pad: GPIO_EMC_B2_01 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_09 for Mode: ALT9 10b - Selecting Pad: GPIO_SD_B2_02 for Mode: ALT3

17.4.1.503 QTIMER6_TMR2_INPUT_SELECT_INPUT DAISY Register (QTIMER6_TMR2_INPUT_SELECT_INPUT)

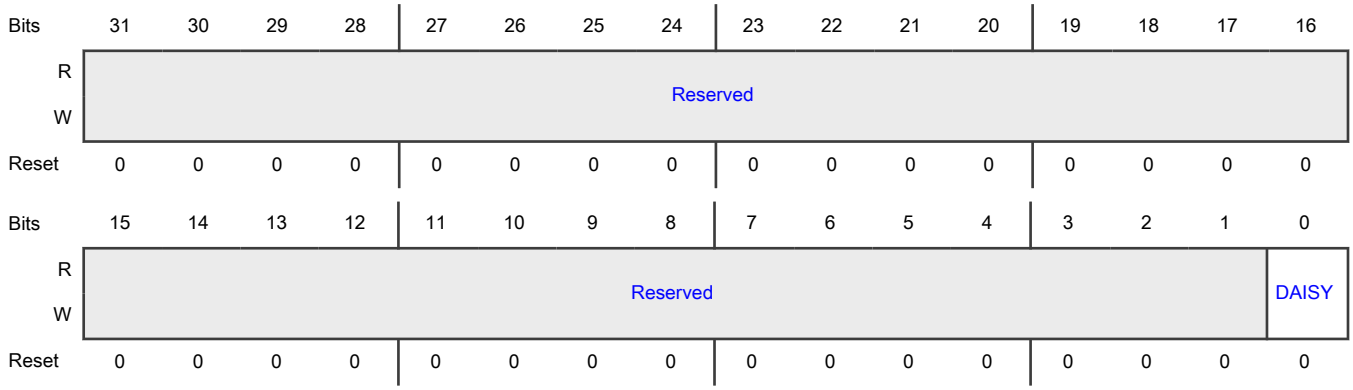
Offset

Register	Offset
QTIMER6_TMR2_INPUT_SELECT_INPUT	89Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer6, In Pin: tmr2_input 0b - Selecting Pad: GPIO_AD_10 for Mode: ALT9 1b - Selecting Pad: GPIO_SD_B2_03 for Mode: ALT3

17.4.1.504 QTIMER7_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER7_TMR0_INPUT_SELECT_INPUT)

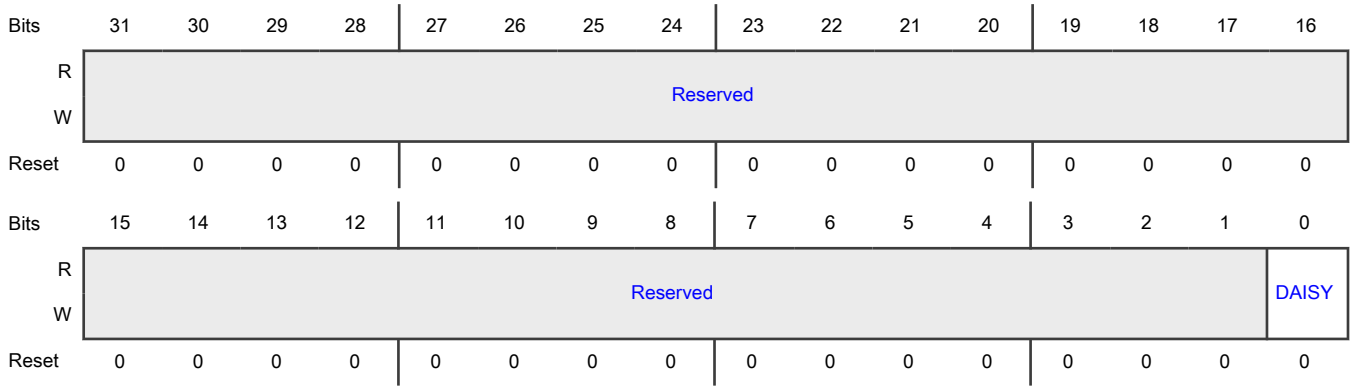
Offset

Register	Offset
QTIMER7_TMR0_INPUT_SELECT_INPUT	8A0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer7, In Pin: tmr0_input 0b - Selecting Pad: GPIO_EMC_B1_24 for Mode: ALT2 1b - Selecting Pad: GPIO_SD_B2_04 for Mode: ALT3

17.4.1.505 QTIMER7_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER7_TMR1_INPUT_SELECT_INPUT)

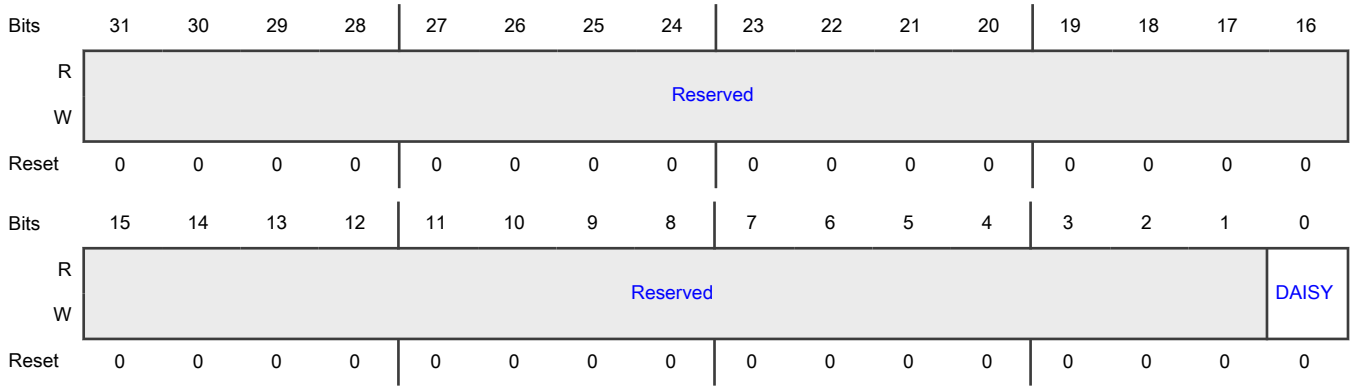
Offset

Register	Offset
QTIMER7_TMR1_INPUT_SELECT_INPUT	8A4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer7, In Pin: tmr1_input 0b - Selecting Pad: GPIO_EMC_B2_02 for Mode: ALT2 1b - Selecting Pad: GPIO_SD_B2_05 for Mode: ALT3

17.4.1.506 QTIMER8_TMR0_INPUT_SELECT_INPUT DAISY Register (QTIMER8_TMR0_INPUT_SELECT_INPUT)

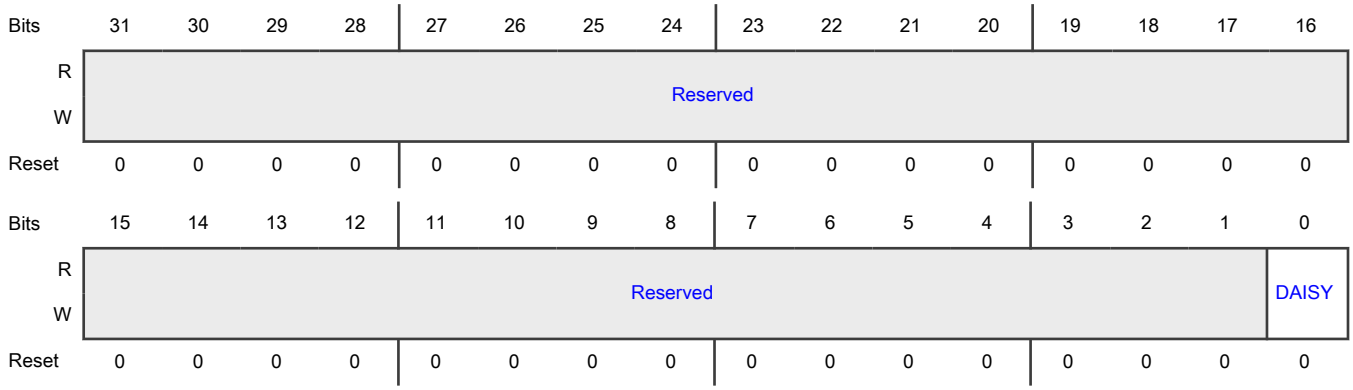
Offset

Register	Offset
QTIMER8_TMR0_INPUT_SELECT_INPUT	8A8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer8, In Pin: tmr0_input 0b - Selecting Pad: GPIO_EMC_B1_25 for Mode: ALT2 1b - Selecting Pad: GPIO_SD_B2_08 for Mode: ALT3

17.4.1.507 QTIMER8_TMR1_INPUT_SELECT_INPUT DAISY Register (QTIMER8_TMR1_INPUT_SELECT_INPUT)

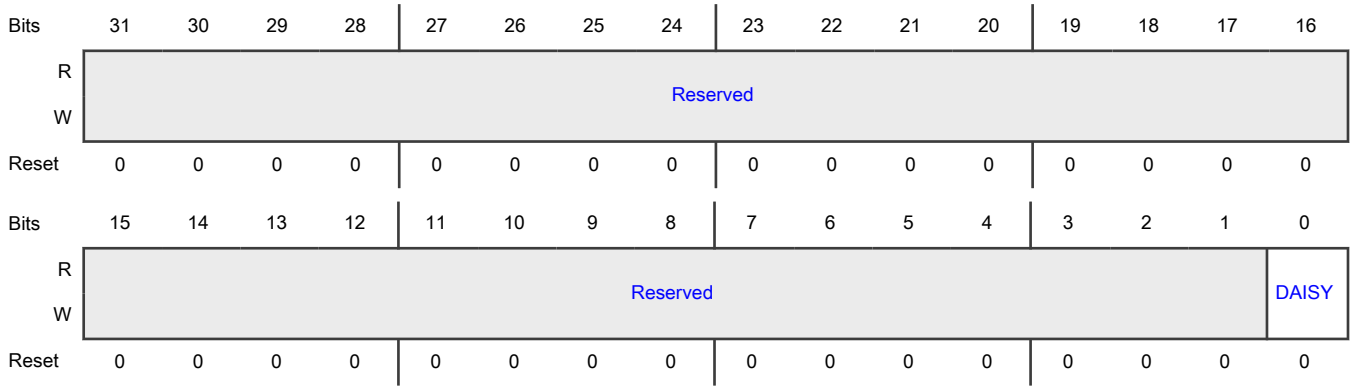
Offset

Register	Offset
QTIMER8_TMR1_INPUT_SELECT_INPUT	8ACh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: qtimer8, In Pin: tmr1_input 0b - Selecting Pad: GPIO_EMC_B2_03 for Mode: ALT2 1b - Selecting Pad: GPIO_SD_B2_09 for Mode: ALT3

17.4.1.508 SAI4_IPG_CLK_SAI_MCLK_SELECT_INPUT DAISY Register (SAI4_IPG_CLK_SAI_MCLK_SELECT_INPUT)

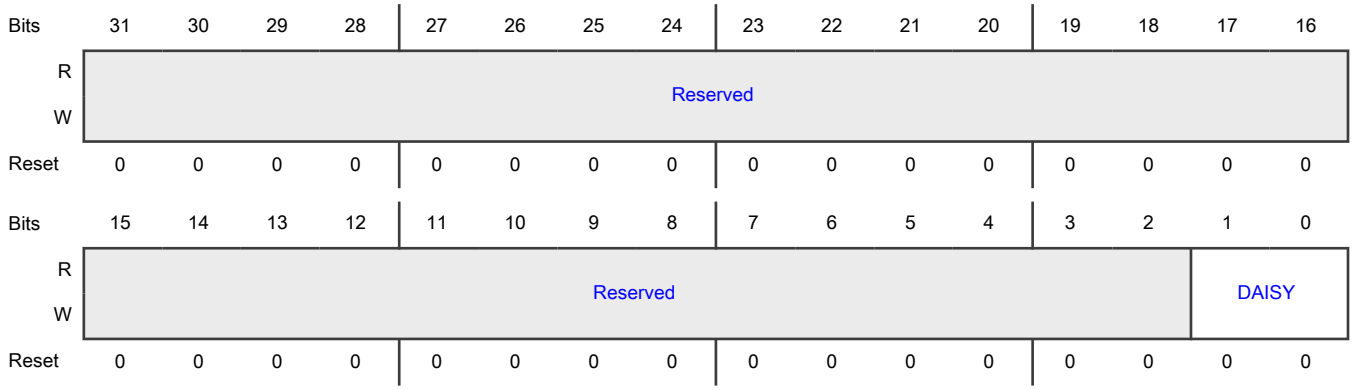
Offset

Register	Offset
SAI4_IPG_CLK_SAI_MCLK_SELECT_INPUT	8B0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai4, In Pin: ipg_clk_sai_mclk 00b - Selecting Pad: GPIO_AD_17 for Mode: ALT0 01b - Selecting Pad: GPIO_B1_05 for Mode: ALT12 10b - Selecting Pad: GPIO_B2_00 for Mode: ALT4

17.4.1.509 SAI4_IPP_IND_SAI_RXBCLK_SELECT_INPUT DAISY Register (SAI4_IPP_IND_SAI_RXBCLK_SELECT_INPUT)

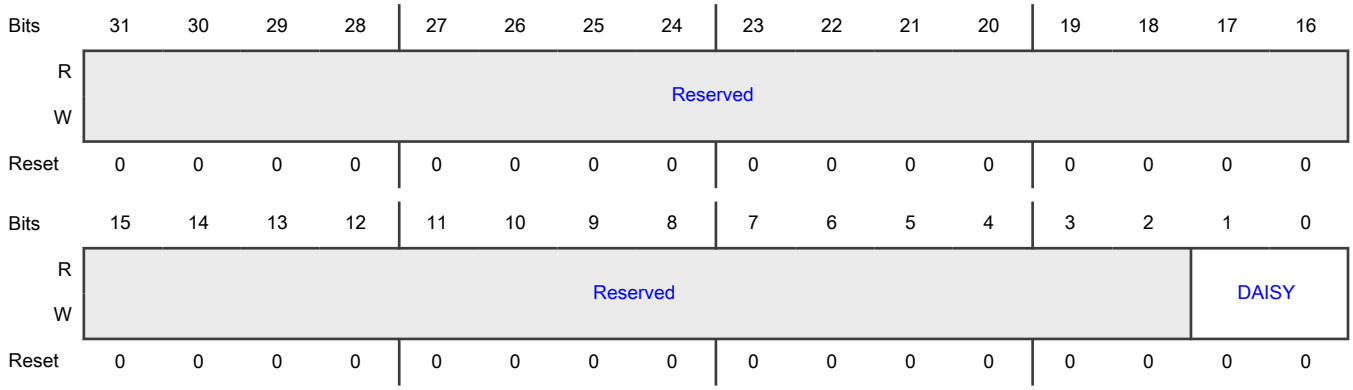
Offset

Register	Offset
SAI4_IPP_IND_SAI_RXBCLK_SELECT_INPUT	8B4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai4, In Pin: ipp_ind_sai_rxbclk 00b - Selecting Pad: GPIO_AD_19 for Mode: ALT0 01b - Selecting Pad: GPIO_B1_06 for Mode: ALT12 10b - Selecting Pad: GPIO_B2_05 for Mode: ALT4

17.4.1.510 SAI4_IPP_IND_SAI_RXDATA_SELECT_INPUT_0 DAISY Register (SAI4_IPP_IND_SAI_RXDATA_SELECT_INPUT_0)

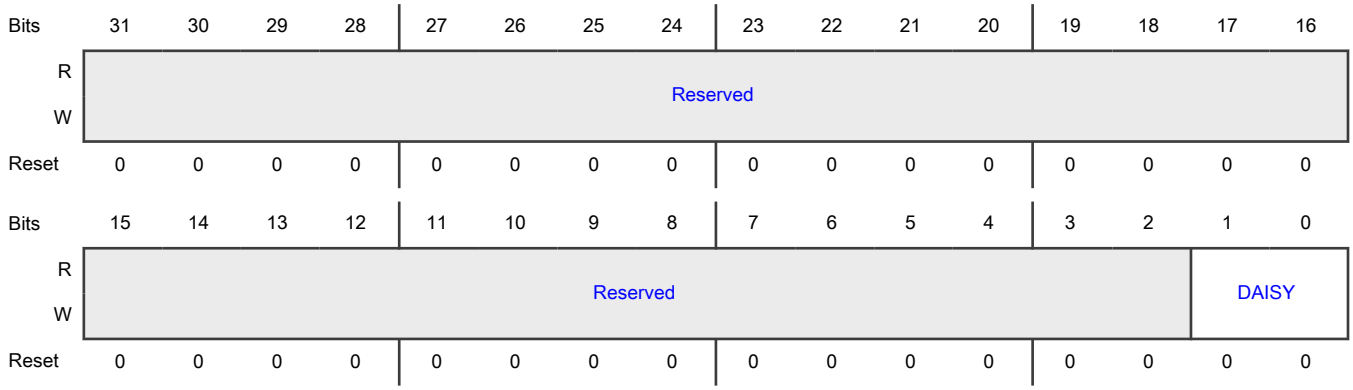
Offset

Register	Offset
SAI4_IPP_IND_SAI_RXDATA_SELECT_INPUT_0	8B8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai4, In Pin: ipp_ind_sai_rxdata0 00b - Selecting Pad: GPIO_AD_20 for Mode: ALT0 01b - Selecting Pad: GPIO_B1_01 for Mode: ALT12 10b - Selecting Pad: GPIO_B2_06 for Mode: ALT4

17.4.1.511 SAI4_IPP_IND_SAI_RXDATA_SELECT_INPUT_1 DAISY Register (SAI4_IPP_IND_SAI_RXDATA_SELECT_INPUT_1)

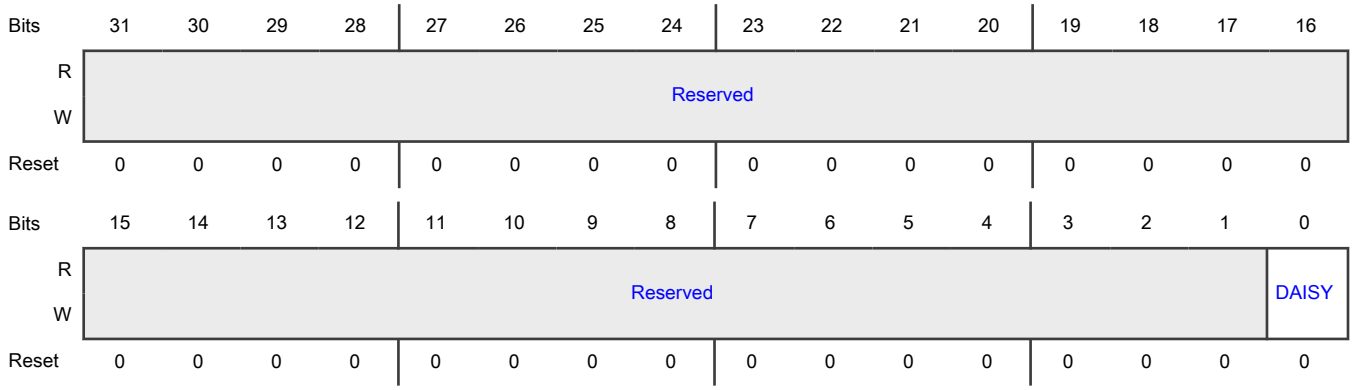
Offset

Register	Offset
SAI4_IPP_IND_SAI_RXDATA_SELECT_INPUT_1	8BCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai4, In Pin: ipp_ind_sai_rxdata1 0b - Selecting Pad: GPIO_B1_02 for Mode: ALT12 1b - Selecting Pad: GPIO_B2_07 for Mode: ALT6

17.4.1.512 SAI4_IPP_IND_SAI_RXSYNC_SELECT_INPUT DAISY Register (SAI4_IPP_IND_SAI_RXSYNC_SELECT_INPUT)

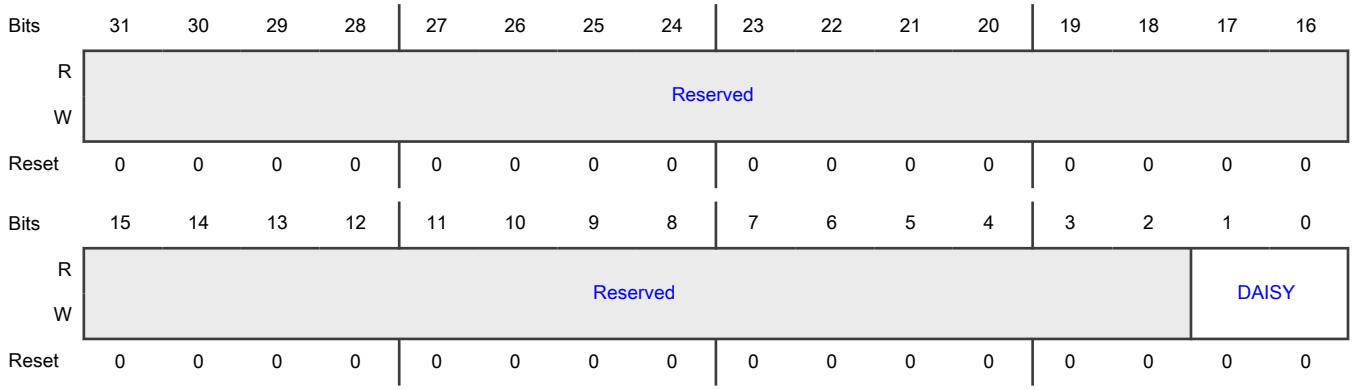
Offset

Register	Offset
SAI4_IPP_IND_SAI_RXSYNC_SELECT_INPUT	8C0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai4, In Pin: ipp_ind_sai_rxsync 00b - Selecting Pad: GPIO_AD_18 for Mode: ALT0 01b - Selecting Pad: GPIO_B1_07 for Mode: ALT12 10b - Selecting Pad: GPIO_B2_04 for Mode: ALT4

17.4.1.513 SAI4_IPP_IND_SAI_TXBCLK_SELECT_INPUT DAISY Register (SAI4_IPP_IND_SAI_TXBCLK_SELECT_INPUT)

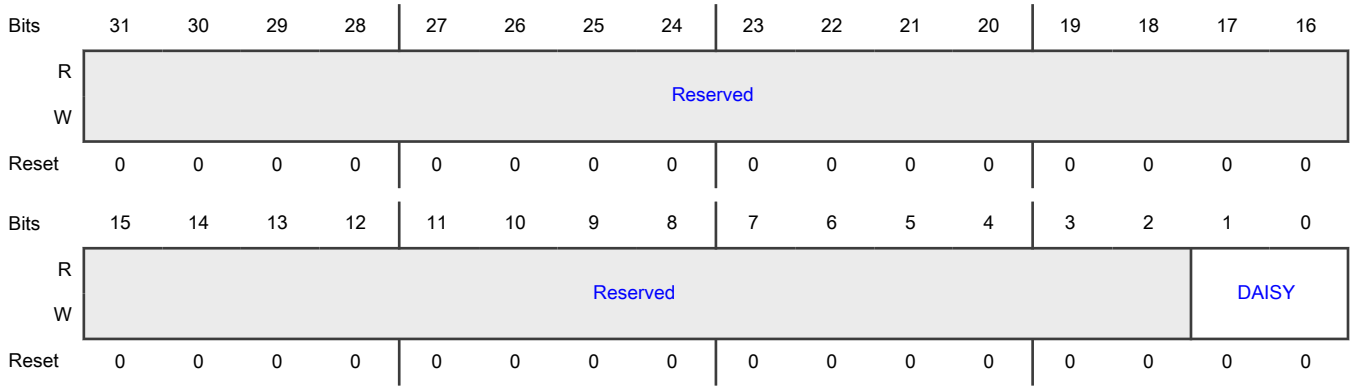
Offset

Register	Offset
SAI4_IPP_IND_SAI_TXBCLK_SELECT_INPUT	8C4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai4, In Pin: ipp_ind_sai_txbclk 00b - Selecting Pad: GPIO_AD_22 for Mode: ALT0 01b - Selecting Pad: GPIO_B1_08 for Mode: ALT12 10b - Selecting Pad: GPIO_B2_01 for Mode: ALT4

17.4.1.514 SAI4_IPP_IND_SAI_TXSYNC_SELECT_INPUT DAISY Register (SAI4_IPP_IND_SAI_TXSYNC_SELECT_INPUT)

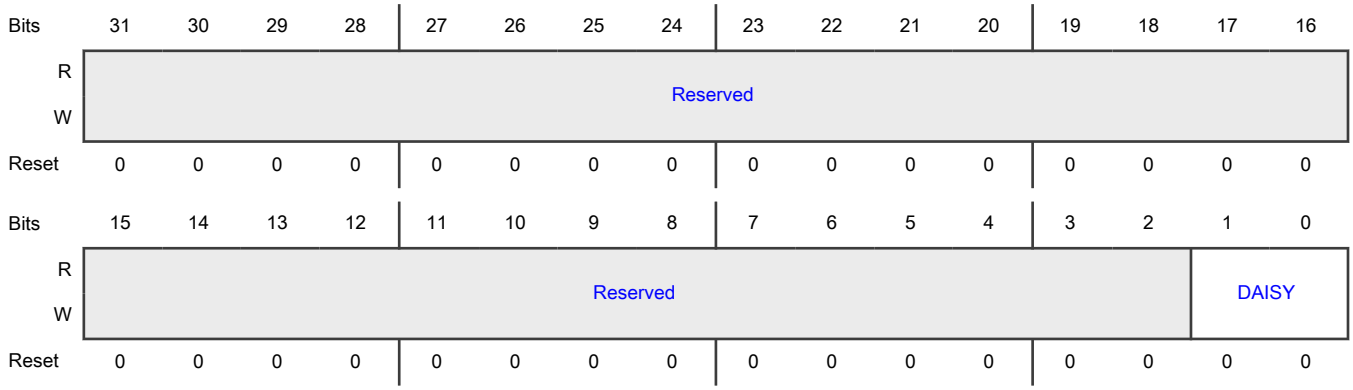
Offset

Register	Offset
SAI4_IPP_IND_SAI_TXSYNC_SELECT_INPUT	8C8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai4, In Pin: ipp_ind_sai_txsync 00b - Selecting Pad: GPIO_AD_23 for Mode: ALT0 01b - Selecting Pad: GPIO_B1_09 for Mode: ALT12 10b - Selecting Pad: GPIO_B2_02 for Mode: ALT4

17.4.1.515 SINC1_IPP_IND_EMBIT_SELECT_INPUT_0 DAISY Register (SINC1_IPP_IND_EMBIT_SELECT_INPUT_0)

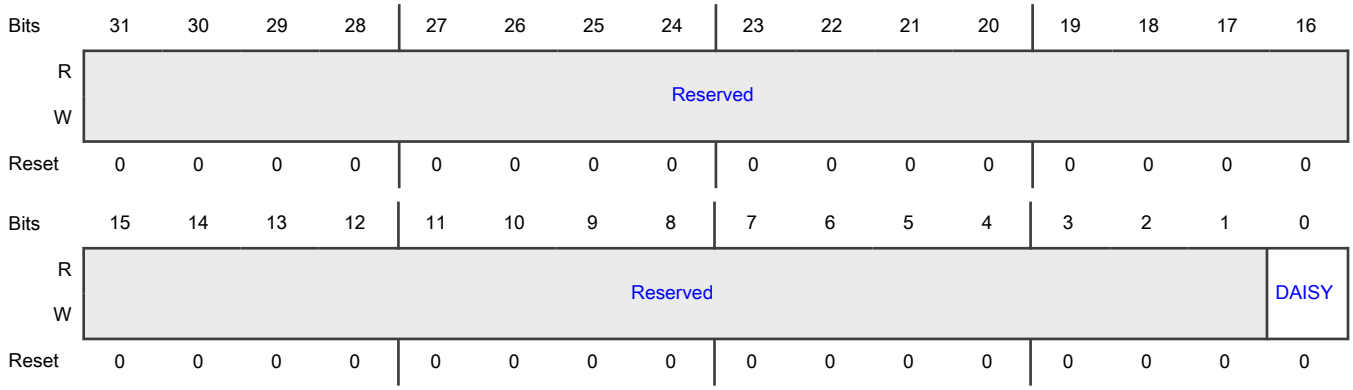
Offset

Register	Offset
SINC1_IPP_IND_EMBIT_SELECT_INPUT_0	8D4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc1, In Pin: ipp_ind_embit0 0b - Selecting Pad: GPIO_AD_04 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_21 for Mode: ALT3

17.4.1.516 SINC1_IPP_IND_EMBIT_SELECT_INPUT_1 DAISY Register (SINC1_IPP_IND_EMBIT_SELECT_INPUT_1)

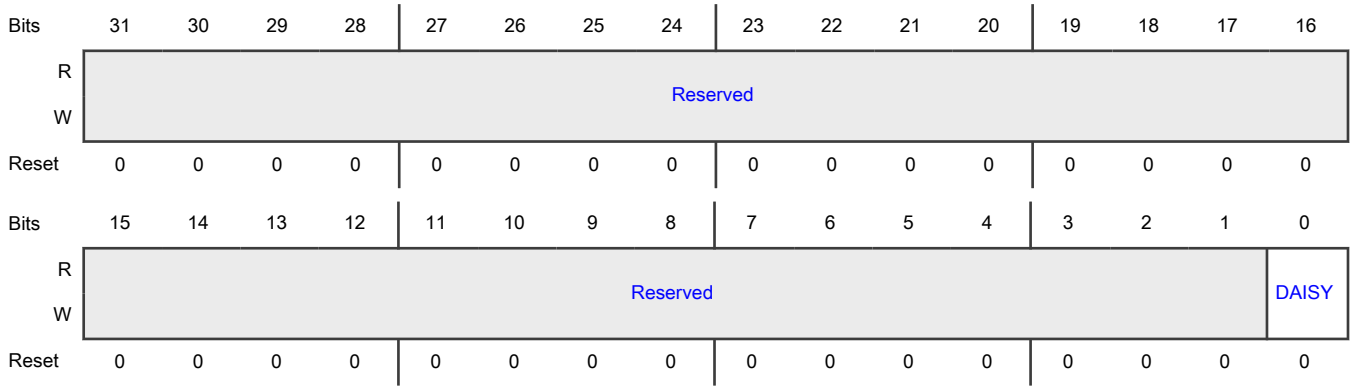
Offset

Register	Offset
SINC1_IPP_IND_EMBIT_SELECT_INPUT_1	8D8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc1, In Pin: ipp_ind_embit1 0b - Selecting Pad: GPIO_AD_06 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_23 for Mode: ALT3

17.4.1.517 SINC1_IPP_IND_EMBIT_SELECT_INPUT_2 DAISY Register (SINC1_IPP_IND_EMBIT_SELECT_INPUT_2)

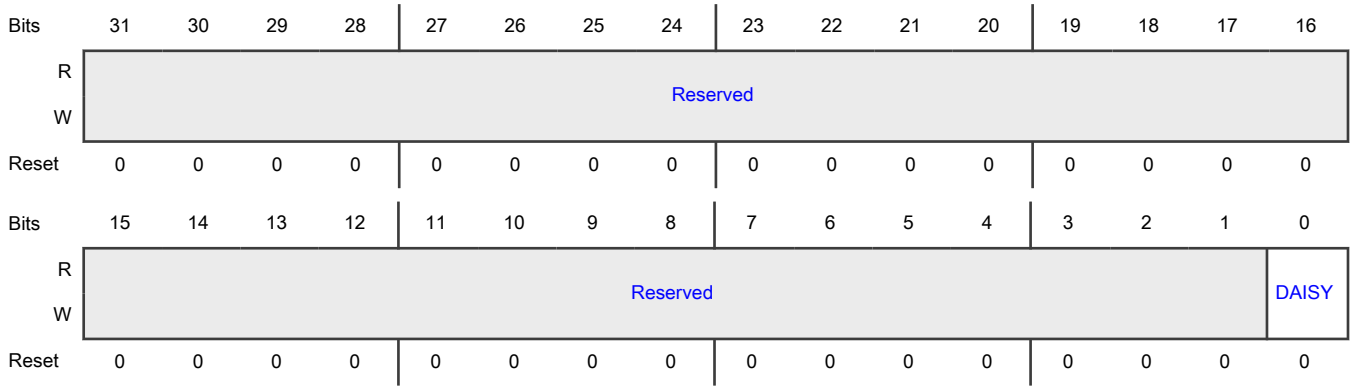
Offset

Register	Offset
SINC1_IPP_IND_EMBIT_SELECT_INPUT_2	8DCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc1, In Pin: ipp_ind_embit2 0b - Selecting Pad: GPIO_AD_08 for Mode: ALT6 1b - Selecting Pad: GPIO_SD_B1_01 for Mode: ALT1

17.4.1.518 SINC1_IPP_IND_EMBIT_SELECT_INPUT_3 DAISY Register (SINC1_IPP_IND_EMBIT_SELECT_INPUT_3)

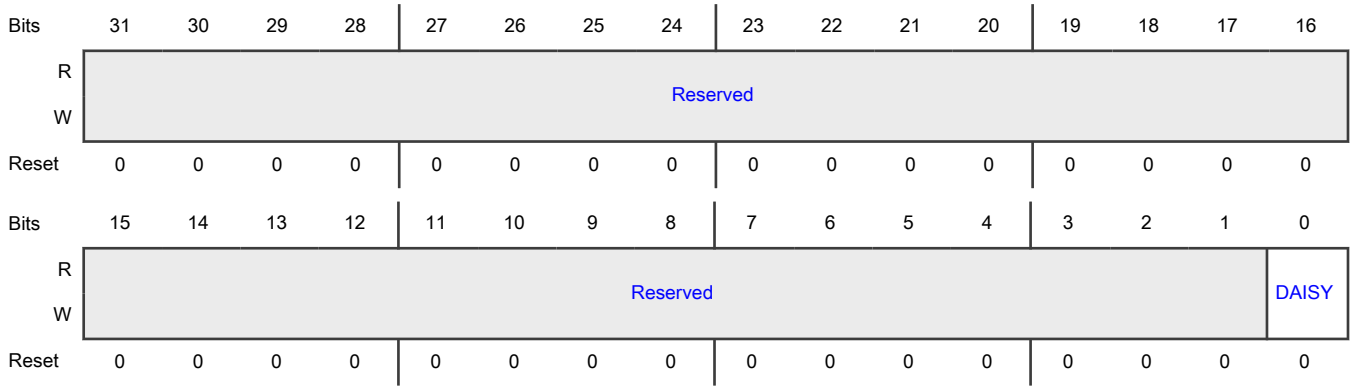
Offset

Register	Offset
SINC1_IPP_IND_EMBIT_SELECT_INPUT_3	8E0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc1, In Pin: ipp_ind_embit3 0b - Selecting Pad: GPIO_AD_10 for Mode: ALT6 1b - Selecting Pad: GPIO_SD_B1_03 for Mode: ALT1

17.4.1.519 SINC1_IPP_IND_EMCLK_SELECT_INPUT_0 DAISY Register (SINC1_IPP_IND_EMCLK_SELECT_INPUT_0)

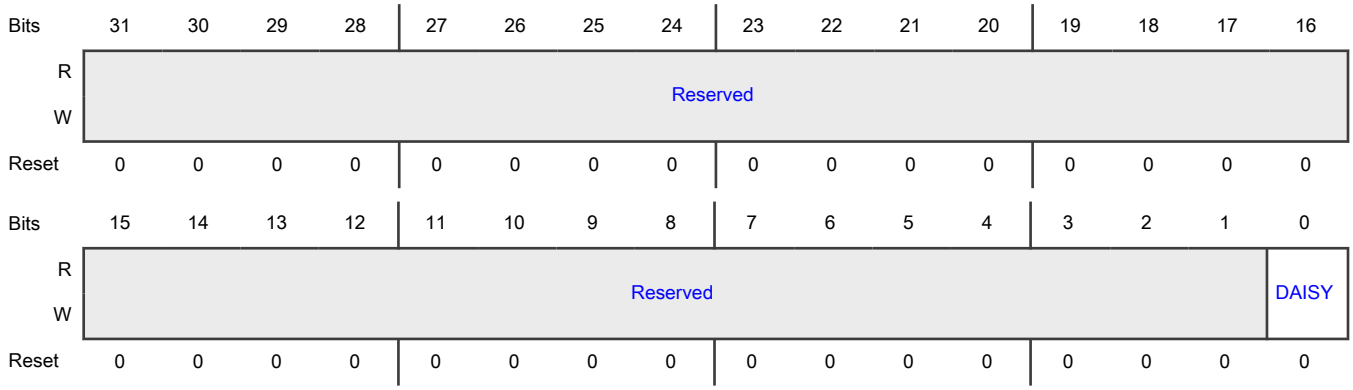
Offset

Register	Offset
SINC1_IPP_IND_EMCLK_SELECT_INPUT_0	8E4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc1, In Pin: ipp_ind_emclk0 0b - Selecting Pad: GPIO_AD_03 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_20 for Mode: ALT3

17.4.1.520 SINC1_IPP_IND_EMCLK_SELECT_INPUT_1 DAISY Register (SINC1_IPP_IND_EMCLK_SELECT_INPUT_1)

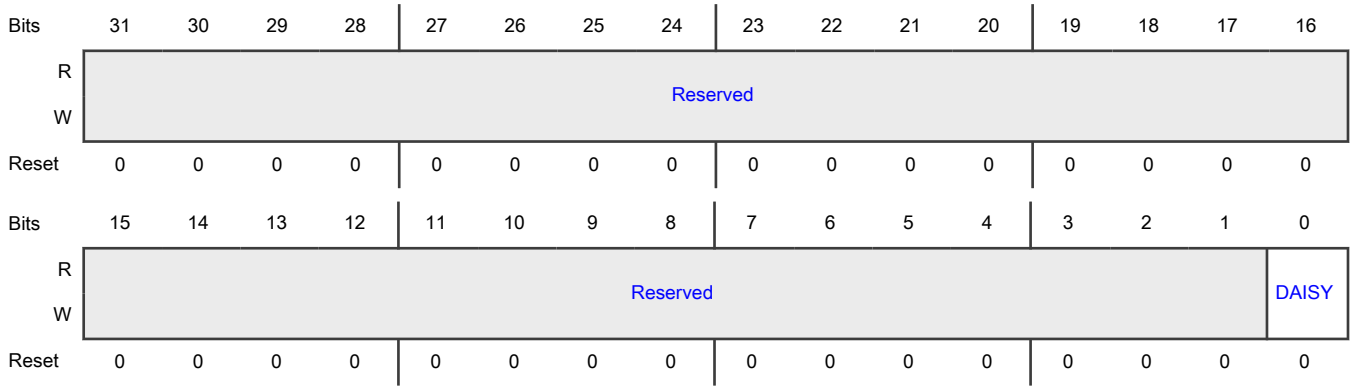
Offset

Register	Offset
SINC1_IPP_IND_EMCLK_SELECT_INPUT_1	8E8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc1, In Pin: ipp_ind_emclk1 0b - Selecting Pad: GPIO_AD_05 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_22 for Mode: ALT3

17.4.1.521 SINC1_IPP_IND_EMCLK_SELECT_INPUT_2 DAISY Register (SINC1_IPP_IND_EMCLK_SELECT_INPUT_2)

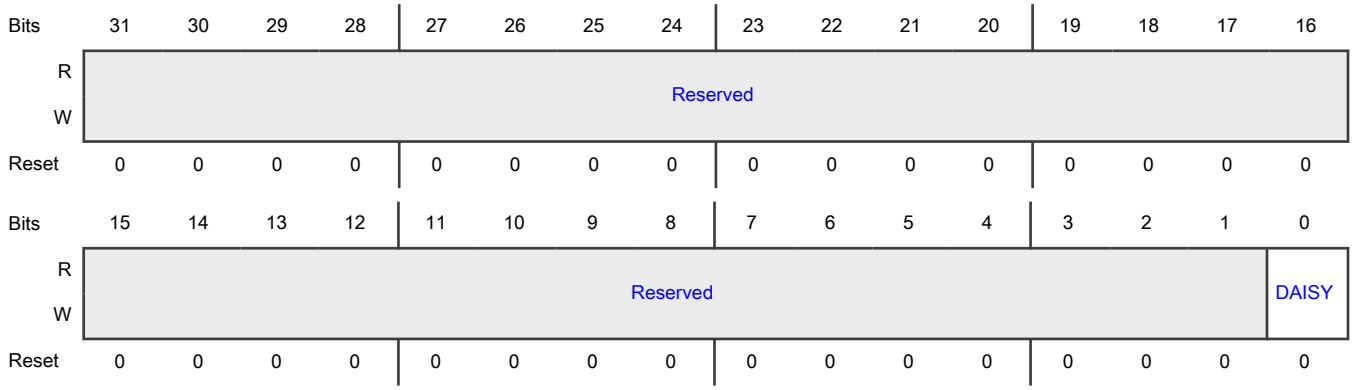
Offset

Register	Offset
SINC1_IPP_IND_EMCLK_SELECT_INPUT_2	8ECh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc1, In Pin: ipp_ind_emclk2 0b - Selecting Pad: GPIO_AD_07 for Mode: ALT6 1b - Selecting Pad: GPIO_SD_B1_00 for Mode: ALT1

17.4.1.522 SINC1_IPP_IND_EMCLK_SELECT_INPUT_3 DAISY Register (SINC1_IPP_IND_EMCLK_SELECT_INPUT_3)

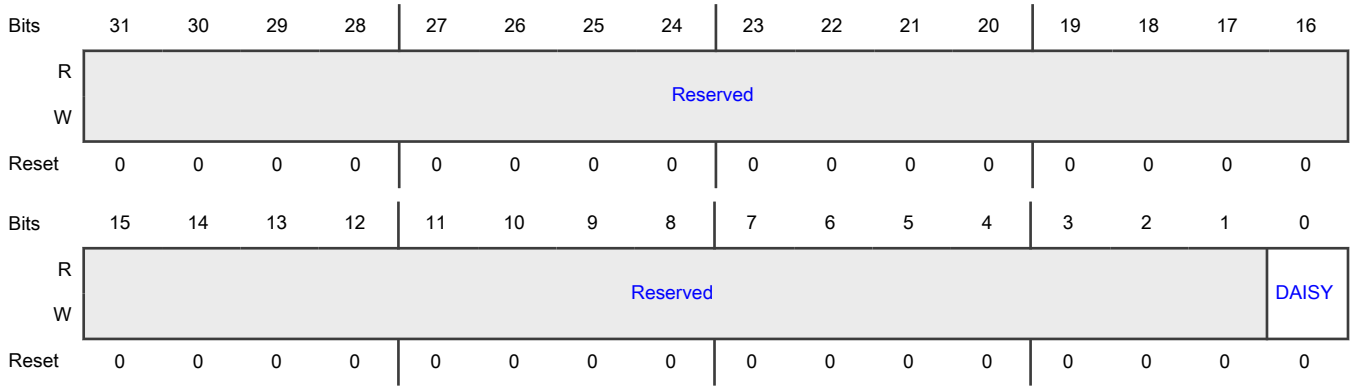
Offset

Register	Offset
SINC1_IPP_IND_EMCLK_SELECT_INPUT_3	8F0h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc1, In Pin: ipp_ind_emclk3 0b - Selecting Pad: GPIO_AD_09 for Mode: ALT6 1b - Selecting Pad: GPIO_SD_B1_02 for Mode: ALT1

17.4.1.523 SINC2_IPP_IND_EMBIT_SELECT_INPUT_2 DAISY Register (SINC2_IPP_IND_EMBIT_SELECT_INPUT_2)

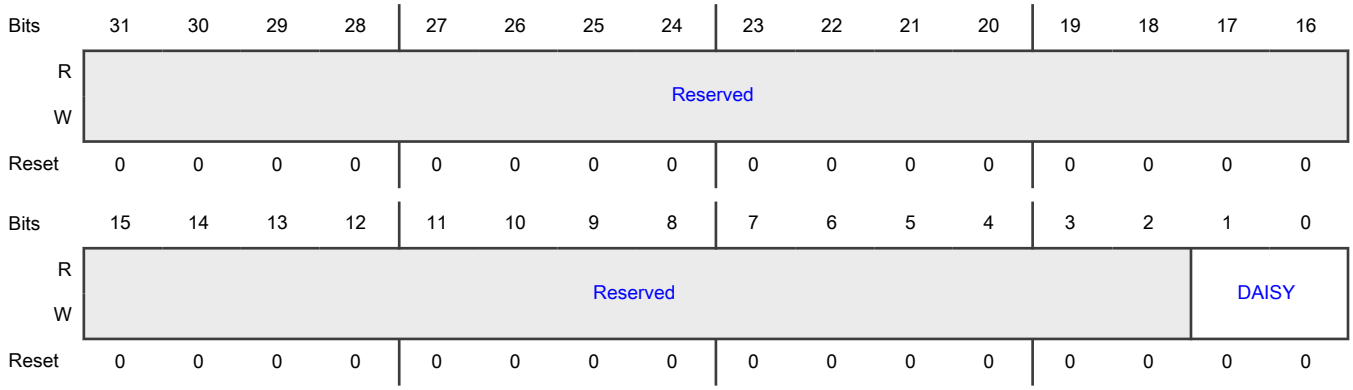
Offset

Register	Offset
SINC2_IPP_IND_EMBIT_SELECT_INPUT_2	8F4h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc2, In Pin: ipp_ind_embit2 00b - Selecting Pad: GPIO_AD_31 for Mode: ALT3 01b - Selecting Pad: GPIO_SD_B1_05 for Mode: ALT2 10b - Selecting Pad: GPIO_B2_09 for Mode: ALT1

17.4.1.524 SINC2_IPP_IND_EMBIT_SELECT_INPUT_3 DAISY Register (SINC2_IPP_IND_EMBIT_SELECT_INPUT_3)

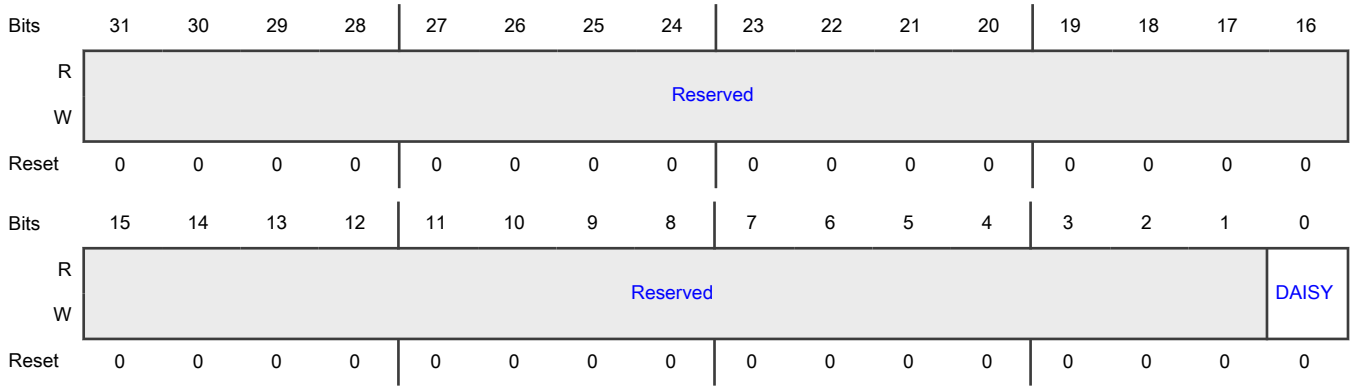
Offset

Register	Offset
SINC2_IPP_IND_EMBIT_SELECT_INPUT_3	8F8h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc2, In Pin: ipp_ind_embit3 0b - Selecting Pad: GPIO_AD_33 for Mode: ALT3 1b - Selecting Pad: GPIO_B2_11 for Mode: ALT1

17.4.1.525 SINC2_IPP_IND_EMCLK_SELECT_INPUT_0 DAISY Register (SINC2_IPP_IND_EMCLK_SELECT_INPUT_0)

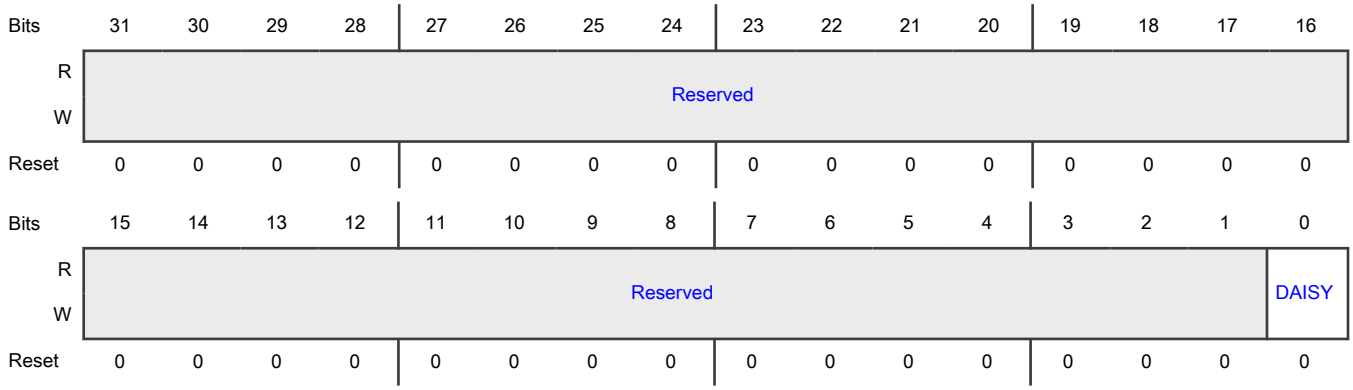
Offset

Register	Offset
SINC2_IPP_IND_EMCLK_SELECT_INPUT_0	8FCh

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc2, In Pin: ipp_ind_emclk0 0b - Selecting Pad: GPIO_AD_26 for Mode: ALT3 1b - Selecting Pad: GPIO_B2_13 for Mode: ALT1

17.4.1.526 SINC2_IPP_IND_EMCLK_SELECT_INPUT_2 DAISY Register (SINC2_IPP_IND_EMCLK_SELECT_INPUT_2)

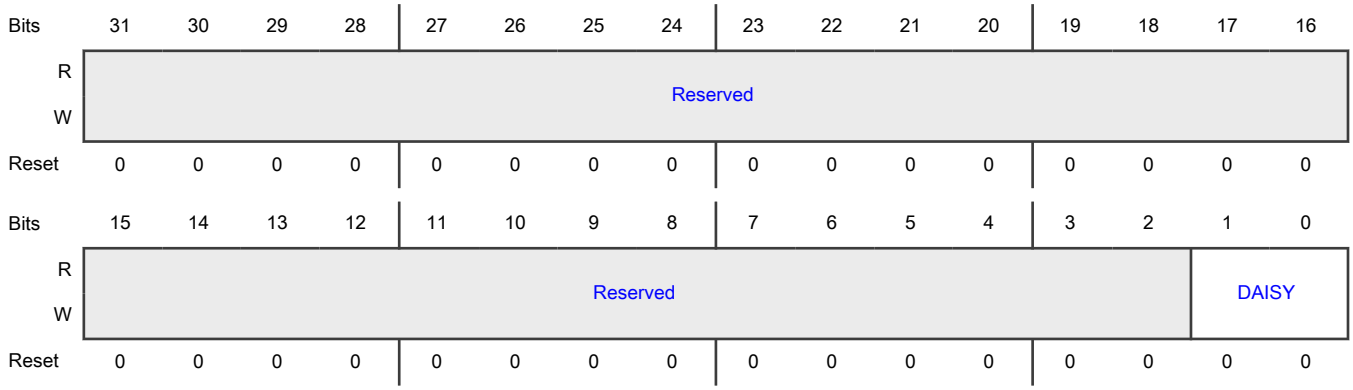
Offset

Register	Offset
SINC2_IPP_IND_EMCLK_SELECT_INPUT_2	900h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc2, In Pin: ipp_ind_emclk2 00b - Selecting Pad: GPIO_AD_30 for Mode: ALT3 01b - Selecting Pad: GPIO_SD_B1_04 for Mode: ALT2 10b - Selecting Pad: GPIO_B2_08 for Mode: ALT1

17.4.1.527 SINC2_IPP_IND_EMCLK_SELECT_INPUT_3 DAISY Register (SINC2_IPP_IND_EMCLK_SELECT_INPUT_3)

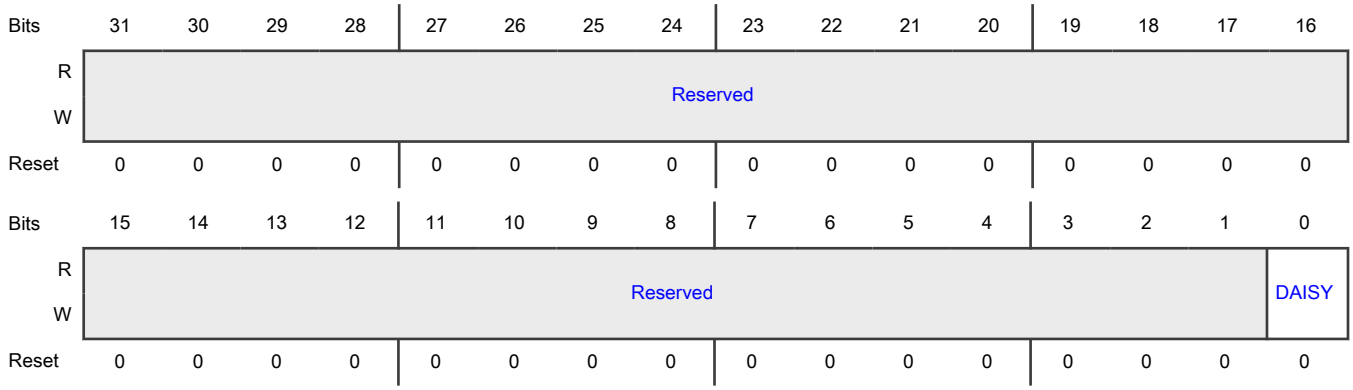
Offset

Register	Offset
SINC2_IPP_IND_EMCLK_SELECT_INPUT_3	904h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sinc2, In Pin: ipp_ind_emclk3 0b - Selecting Pad: GPIO_AD_32 for Mode: ALT3 1b - Selecting Pad: GPIO_B2_10 for Mode: ALT1

17.4.1.528 SPDIF_SPDIF_IN1_SELECT_INPUT DAISY Register (SPDIF_SPDIF_IN1_SELECT_INPUT)

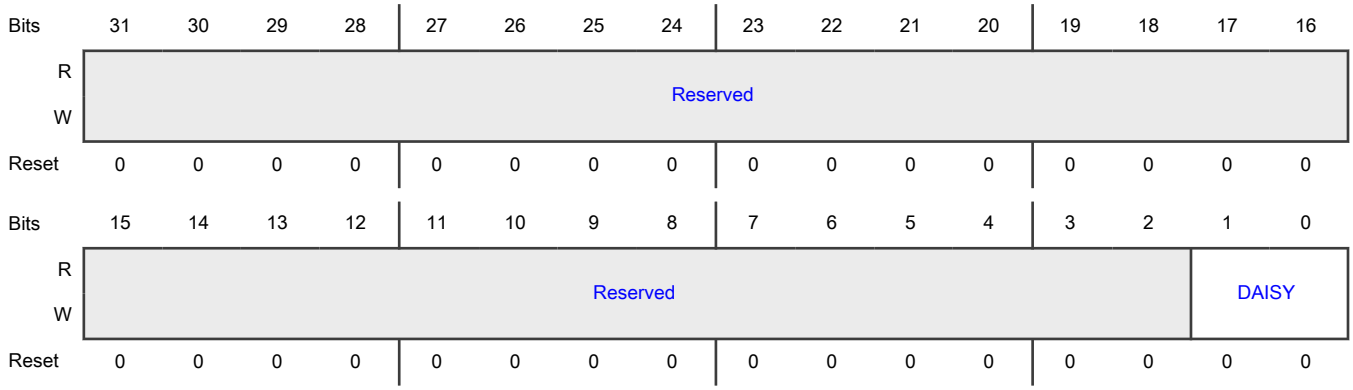
Offset

Register	Offset
SPDIF_SPDIF_IN1_SELECT_INPUT	908h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: spdif, In Pin: spdif_in1 00b - Selecting Pad: GPIO_EMC_B2_12 for Mode: ALT2 01b - Selecting Pad: GPIO_AD_15 for Mode: ALT0 10b - Selecting Pad: GPIO_B2_08 for Mode: ALT9

17.4.1.529 USB_IPP_IND_OTG2_OC_SELECT_INPUT DAISY Register (USB_IPP_IND_OTG2_OC_SELECT_INPUT)

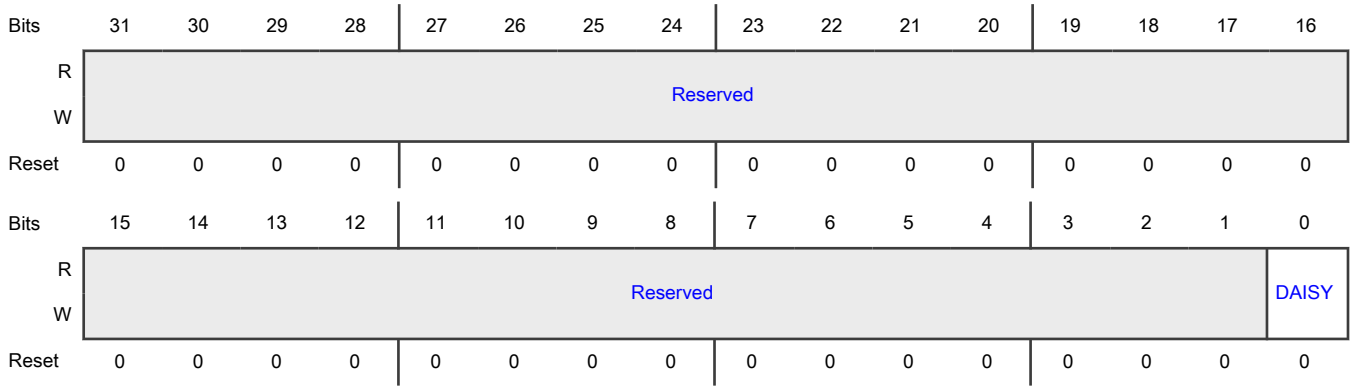
Offset

Register	Offset
USB_IPP_IND_OTG2_OC_SELECT_INPUT	914h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usb, In Pin: ipp_ind_otg2_oc 0b - Selecting Pad: GPIO_AD_06 for Mode: ALT0 1b - Selecting Pad: GPIO_AD_30 for Mode: ALT1

17.4.1.530 USB_IPP_IND_OTG_OC_SELECT_INPUT DAISY Register (USB_IPP_IND_OTG_OC_SELECT_INPUT)

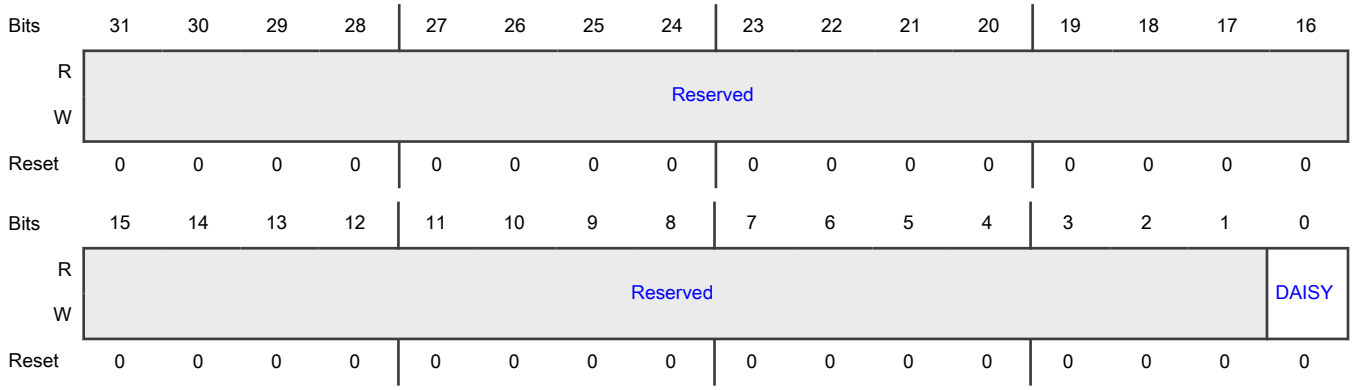
Offset

Register	Offset
USB_IPP_IND_OTG_OC_SELECT_INPUT	918h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usb, In Pin: ipp_ind_otg_oc 0b - Selecting Pad: GPIO_AD_11 for Mode: ALT0 1b - Selecting Pad: GPIO_AD_35 for Mode: ALT1

17.4.1.531 USBPHY1_USB_ID_SELECT_INPUT DAISY Register (USBPHY1_USB_ID_SELECT_INPUT)

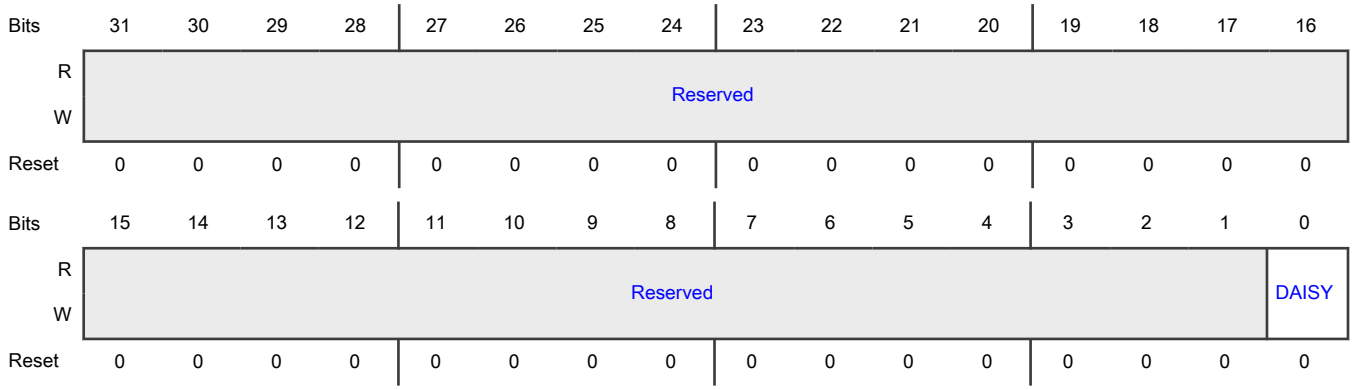
Offset

Register	Offset
USBPHY1_USB_ID_SELECT_INPUT	91Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usbphy1, In Pin: usb_id 0b - Selecting Pad: GPIO_AD_09 for Mode: ALT0 1b - Selecting Pad: GPIO_AD_33 for Mode: ALT1

17.4.1.532 USBPHY2_USB_ID_SELECT_INPUT DAISY Register (USBPHY2_USB_ID_SELECT_INPUT)

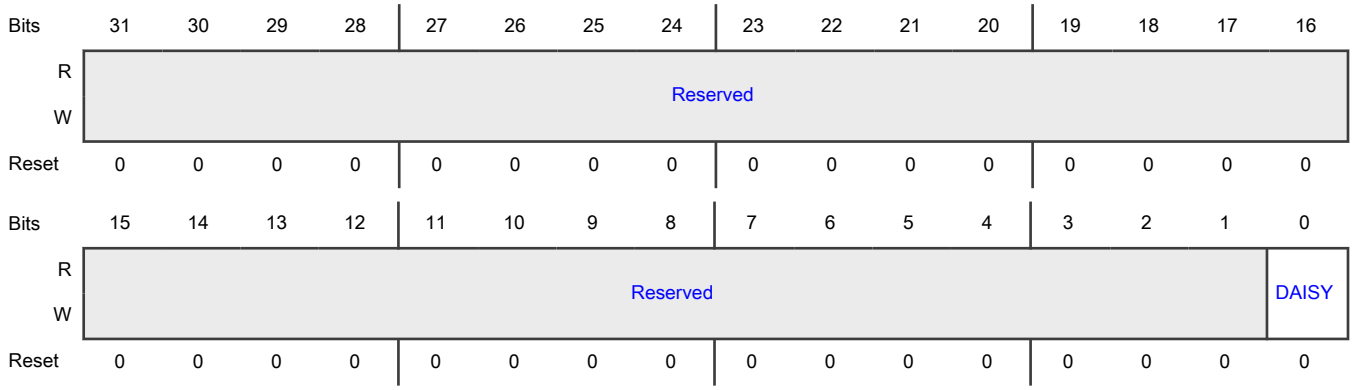
Offset

Register	Offset
USBPHY2_USB_ID_SELECT_INPUT	920h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usbphy2, In Pin: usb_id 0b - Selecting Pad: GPIO_AD_08 for Mode: ALT0 1b - Selecting Pad: GPIO_AD_32 for Mode: ALT1

17.4.1.533 USDHC1_IPP_CARD_DET_SELECT_INPUT DAISY Register (USDHC1_IPP_CARD_DET_SELECT_INPUT)

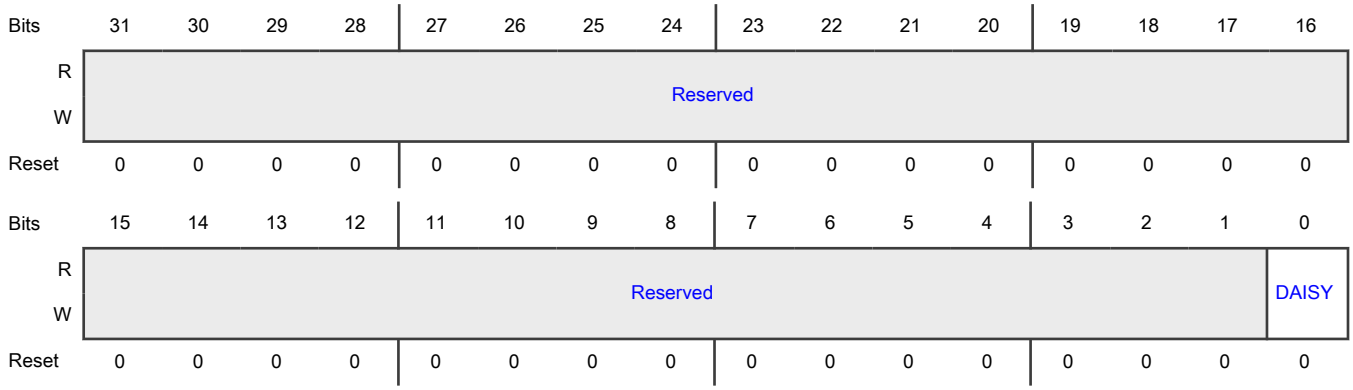
Offset

Register	Offset
USDHC1_IPP_CARD_DET_SELECT_INPUT	924h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usdhc1, In Pin: ipp_card_det 0b - Selecting Pad: GPIO_AD_32 for Mode: ALT4 1b - Selecting Pad: GPIO_B1_08 for Mode: ALT2

17.4.1.534 USDHC1_IPP_WP_ON_SELECT_INPUT DAISY Register (USDHC1_IPP_WP_ON_SELECT_INPUT)

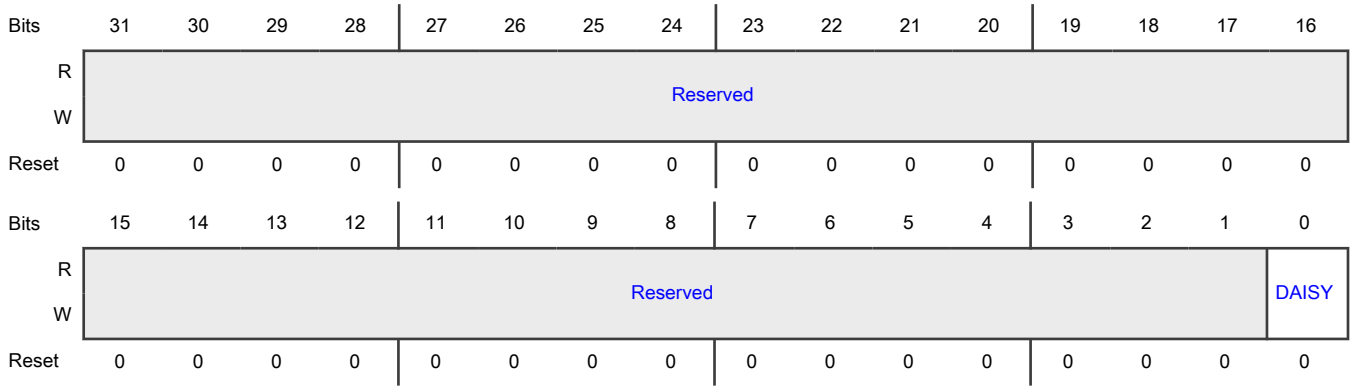
Offset

Register	Offset
USDHC1_IPP_WP_ON_SELECT_INPUT	928h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usdhc1, In Pin: ipp_wp_on 0b - Selecting Pad: GPIO_AD_33 for Mode: ALT4 1b - Selecting Pad: GPIO_B1_09 for Mode: ALT2

17.4.1.535 USDHC2_IPP_CARD_DET_SELECT_INPUT DAISY Register (USDHC2_IPP_CARD_DET_SELECT_INPUT)

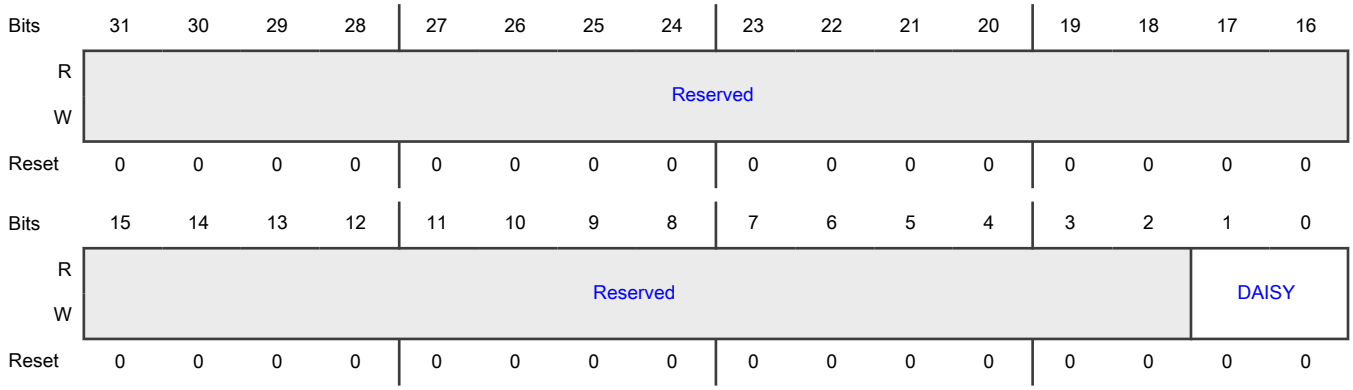
Offset

Register	Offset
USDHC2_IPP_CARD_DET_SELECT_INPUT	92Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usdhc2, In Pin: ipp_card_det 00b - Selecting Pad: GPIO_EMC_B2_01 for Mode: ALT1 01b - Selecting Pad: GPIO_AD_13 for Mode: ALT7 10b - Selecting Pad: GPIO_AD_26 for Mode: ALT9 11b - Selecting Pad: GPIO_AD_29 for Mode: ALT2

17.4.1.536 USDHC2_IPP_WP_ON_SELECT_INPUT DAISY Register (USDHC2_IPP_WP_ON_SELECT_INPUT)

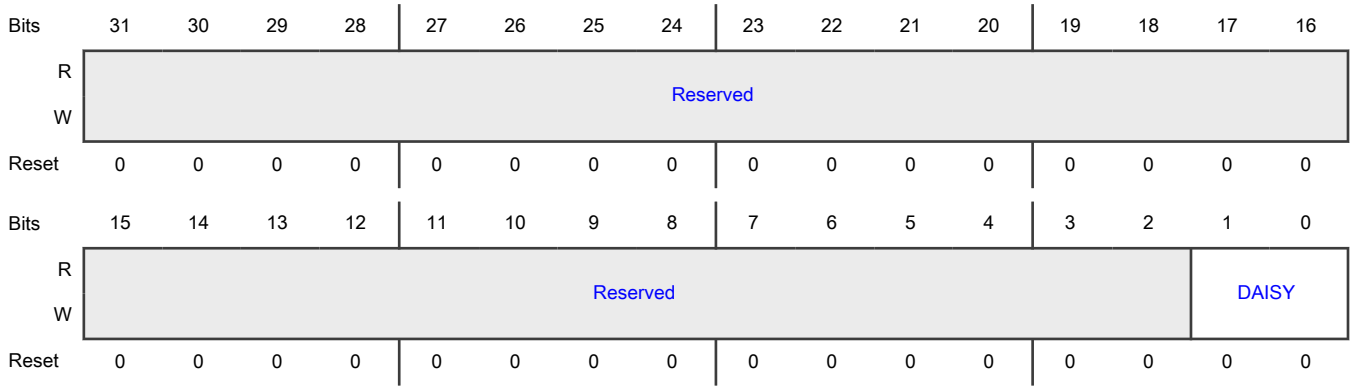
Offset

Register	Offset
USDHC2_IPP_WP_ON_SELECT_INPUT	930h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usdhc2, In Pin: ipp_wp_on 00b - Selecting Pad: GPIO_EMC_B2_02 for Mode: ALT1 01b - Selecting Pad: GPIO_AD_14 for Mode: ALT7 10b - Selecting Pad: GPIO_AD_27 for Mode: ALT9

17.4.1.537 XBAR1_XBAR_IN_SELECT_INPUT_14 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_14)

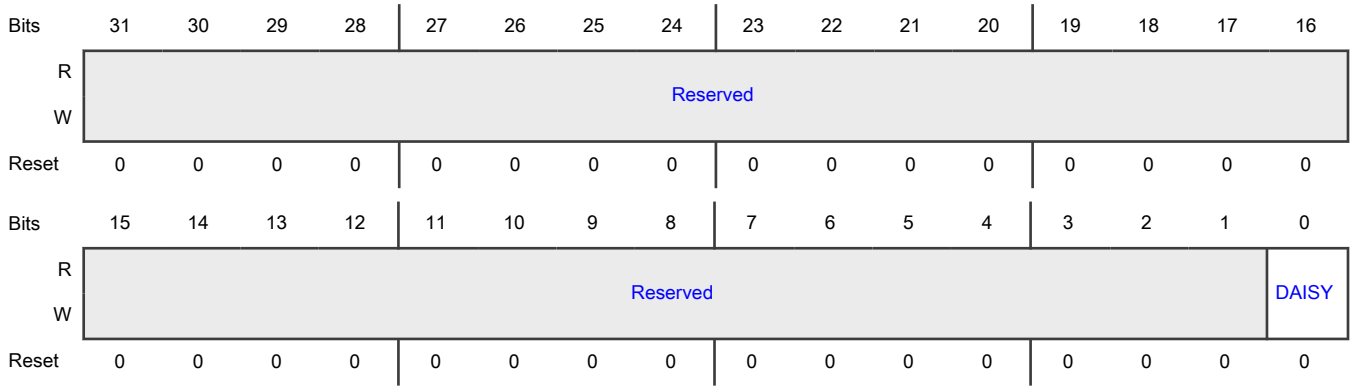
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_14	934h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in14 0b - Selecting Pad: GPIO_EMC_B1_30 for Mode: ALT2 1b - Selecting Pad: GPIO_EMC_B2_16 for Mode: ALT6

17.4.1.538 XBAR1_XBAR_IN_SELECT_INPUT_15 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_15)

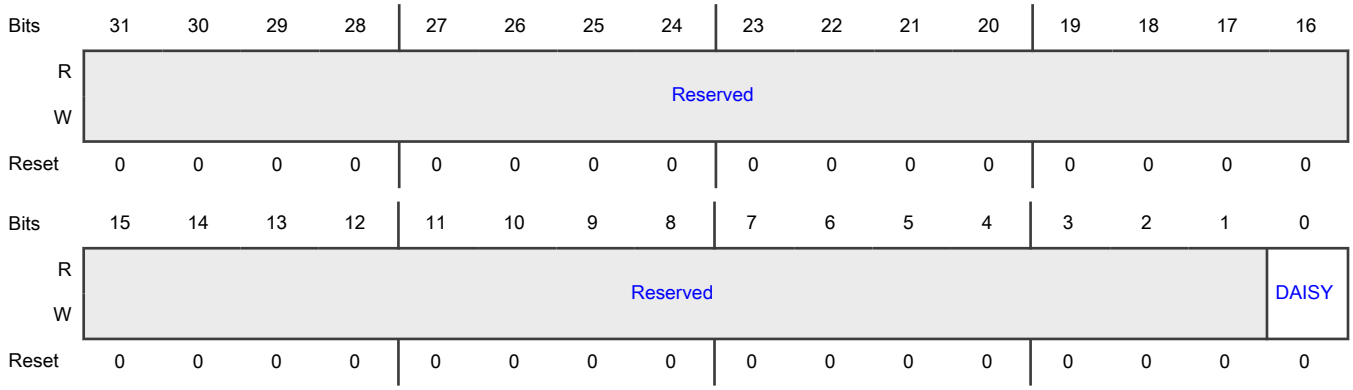
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_15	938h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in15 0b - Selecting Pad: GPIO_EMC_B1_39 for Mode: ALT2 1b - Selecting Pad: GPIO_EMC_B2_17 for Mode: ALT6

17.4.1.539 XBAR1_XBAR_IN_SELECT_INPUT_17 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_17)

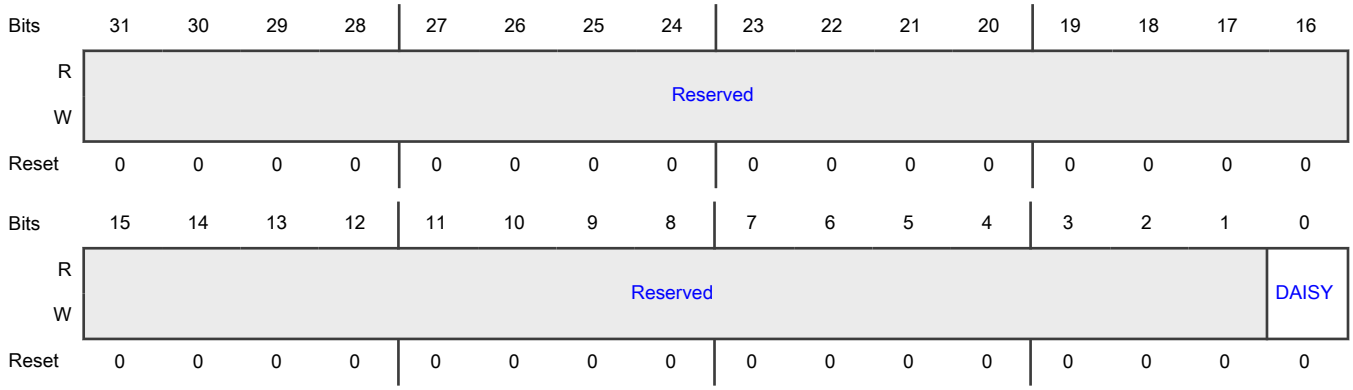
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_17	93Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in17 0b - Selecting Pad: GPIO_AD_33 for Mode: ALT2 1b - Selecting Pad: GPIO_SD_B2_00 for Mode: ALT3

17.4.1.540 XBAR1_XBAR_IN_SELECT_INPUT_18 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_18)

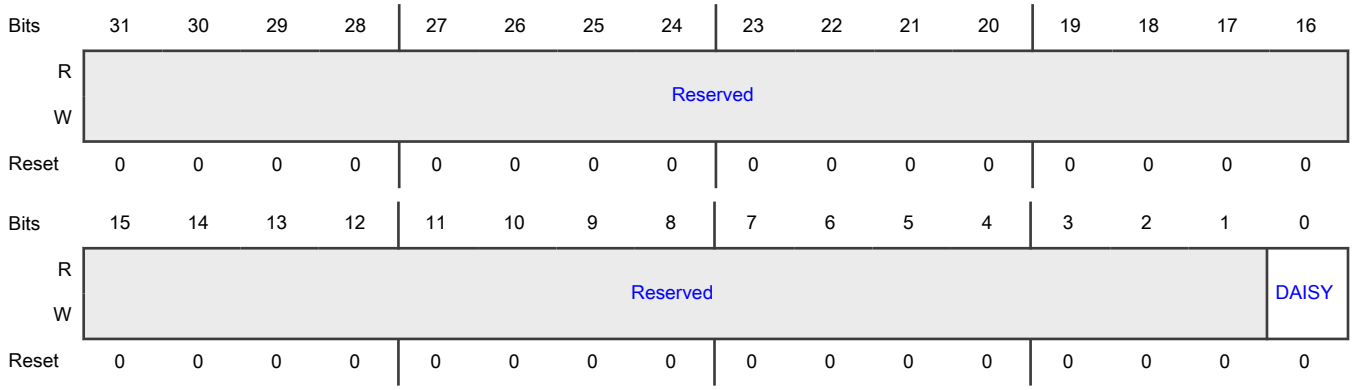
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_18	940h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in18 0b - Selecting Pad: GPIO_AD_12 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_34 for Mode: ALT2

17.4.1.541 XBAR1_XBAR_IN_SELECT_INPUT_19 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_19)

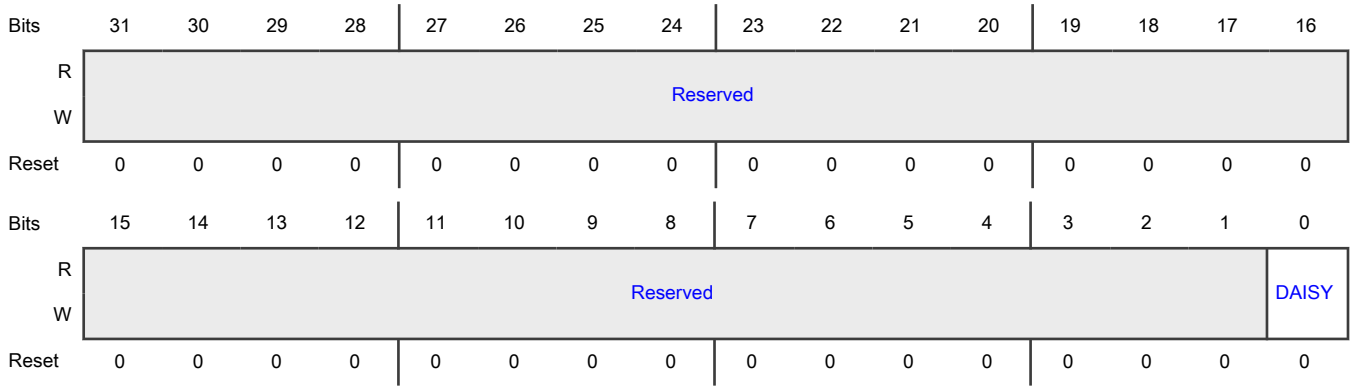
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_19	944h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in19 0b - Selecting Pad: GPIO_AD_19 for Mode: ALT2 1b - Selecting Pad: GPIO_AD_35 for Mode: ALT2

17.4.1.542 XBAR1_XBAR_IN_SELECT_INPUT_20 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_20)

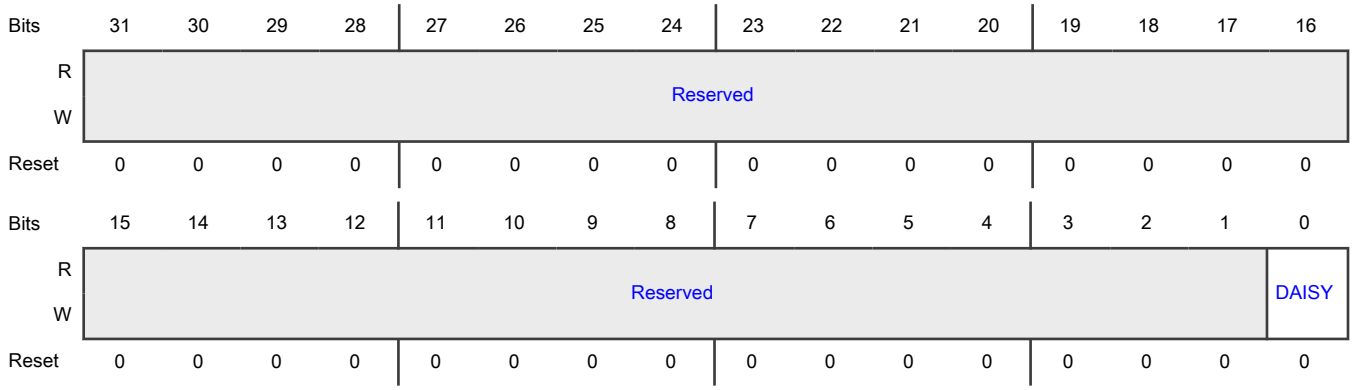
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_20	948h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in20 0b - Selecting Pad: GPIO_EMC_B2_00 for Mode: ALT6 1b - Selecting Pad: GPIO_SD_B1_00 for Mode: ALT2

17.4.1.543 XBAR1_XBAR_IN_SELECT_INPUT_21 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_21)

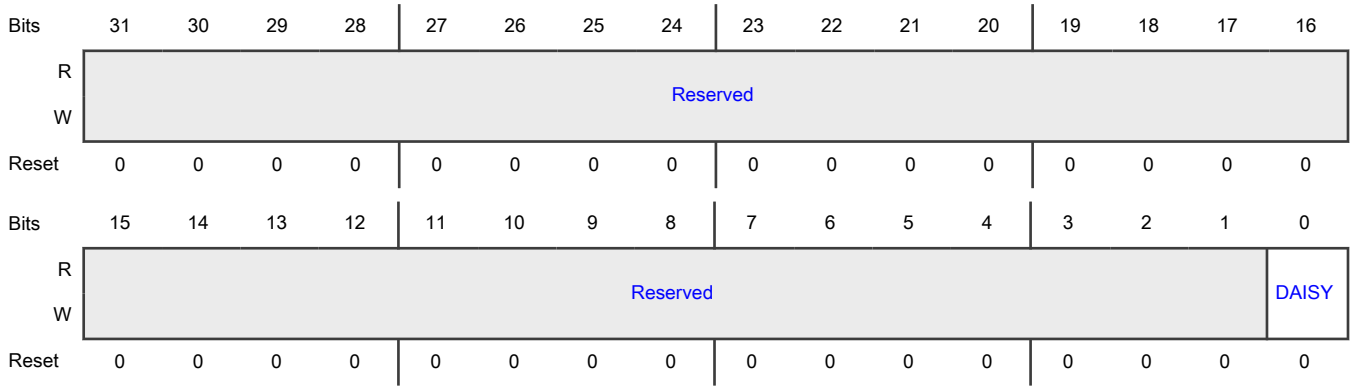
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_21	94Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in21 0b - Selecting Pad: GPIO_EMC_B2_01 for Mode: ALT6 1b - Selecting Pad: GPIO_SD_B1_01 for Mode: ALT2

17.4.1.544 XBAR1_XBAR_IN_SELECT_INPUT_22 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_22)

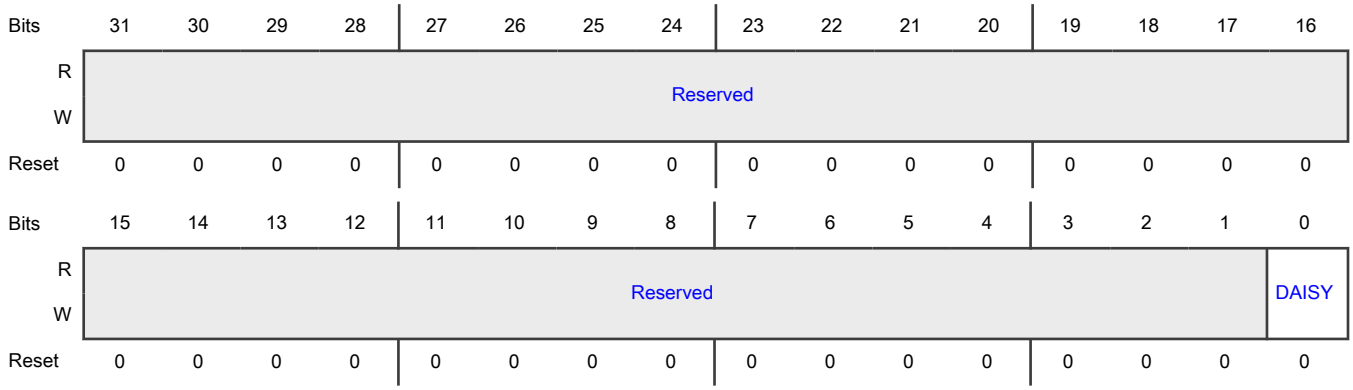
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_22	950h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in22 0b - Selecting Pad: GPIO_EMC_B2_02 for Mode: ALT6 1b - Selecting Pad: GPIO_SD_B1_02 for Mode: ALT2

17.4.1.545 XBAR1_XBAR_IN_SELECT_INPUT_23 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_23)

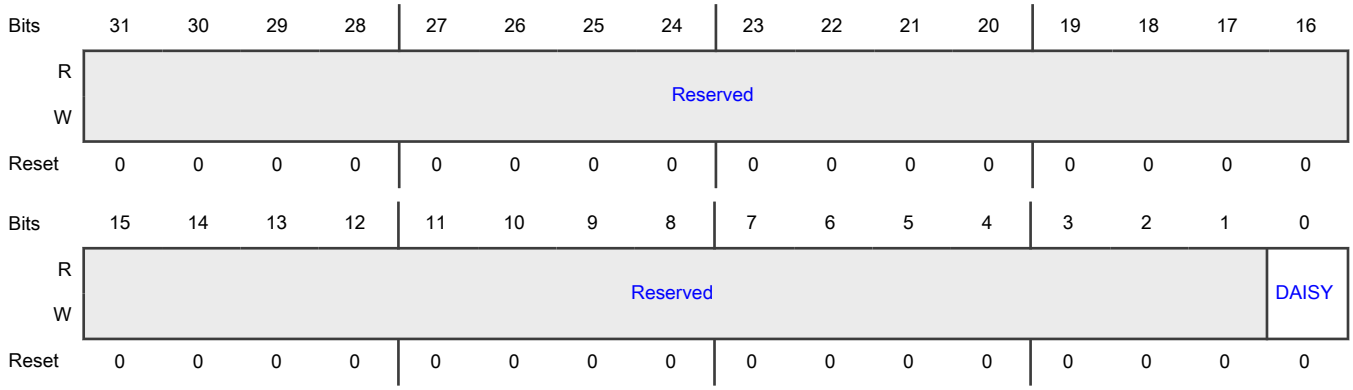
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_23	954h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in23 0b - Selecting Pad: GPIO_EMC_B2_03 for Mode: ALT6 1b - Selecting Pad: GPIO_SD_B1_03 for Mode: ALT2

17.4.1.546 XBAR1_XBAR_IN_SELECT_INPUT_24 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_24)

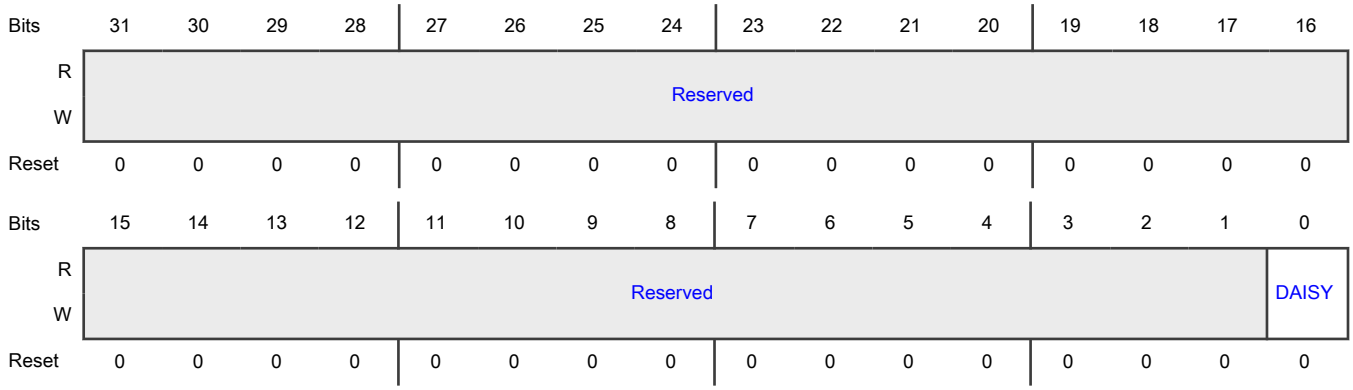
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_24	958h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in24 0b - Selecting Pad: GPIO_EMC_B2_04 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_30 for Mode: ALT9

17.4.1.547 XBAR1_XBAR_IN_SELECT_INPUT_25 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_25)

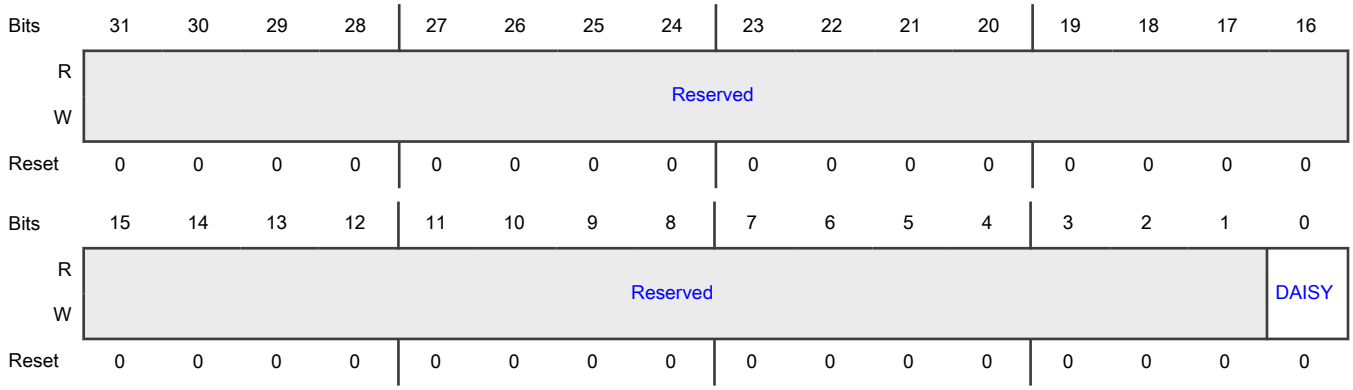
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_25	95Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in25 0b - Selecting Pad: GPIO_EMC_B2_05 for Mode: ALT6 1b - Selecting Pad: GPIO_AD_31 for Mode: ALT9

17.4.1.548 XBAR1_XBAR_IN_SELECT_INPUT_26 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_26)

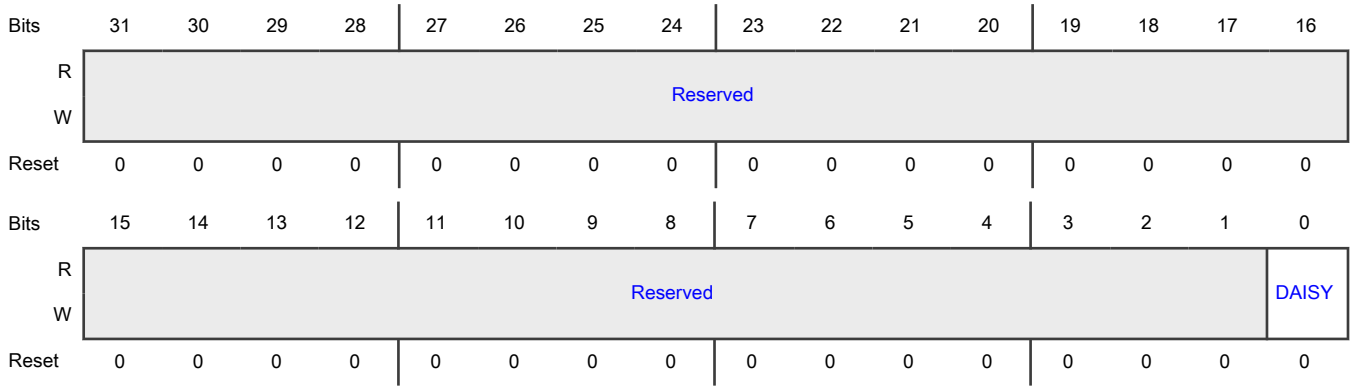
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_26	960h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in26 0b - Selecting Pad: GPIO_EMC_B2_06 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_00 for Mode: ALT4

17.4.1.549 XBAR1_XBAR_IN_SELECT_INPUT_27 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_27)

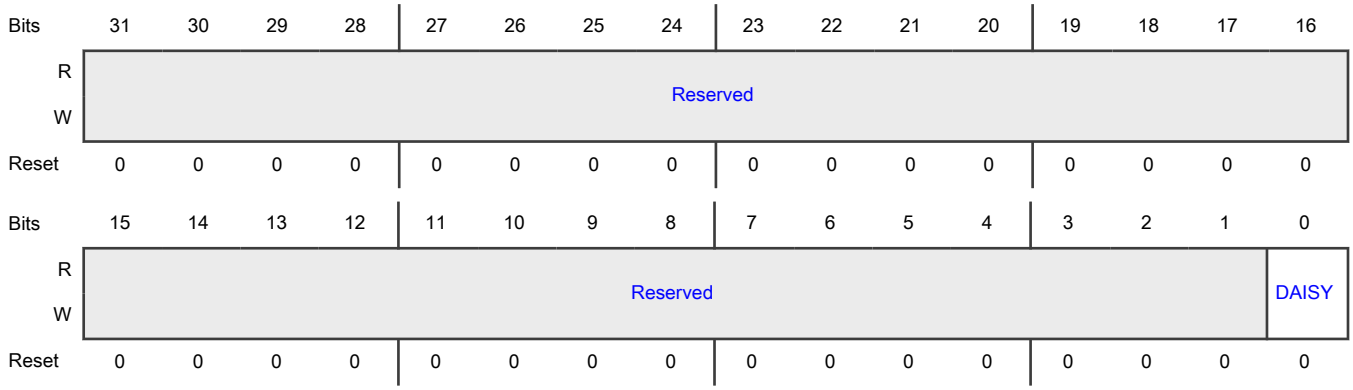
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_27	964h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in27 0b - Selecting Pad: GPIO_EMC_B2_07 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_01 for Mode: ALT4

17.4.1.550 XBAR1_XBAR_IN_SELECT_INPUT_28 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_28)

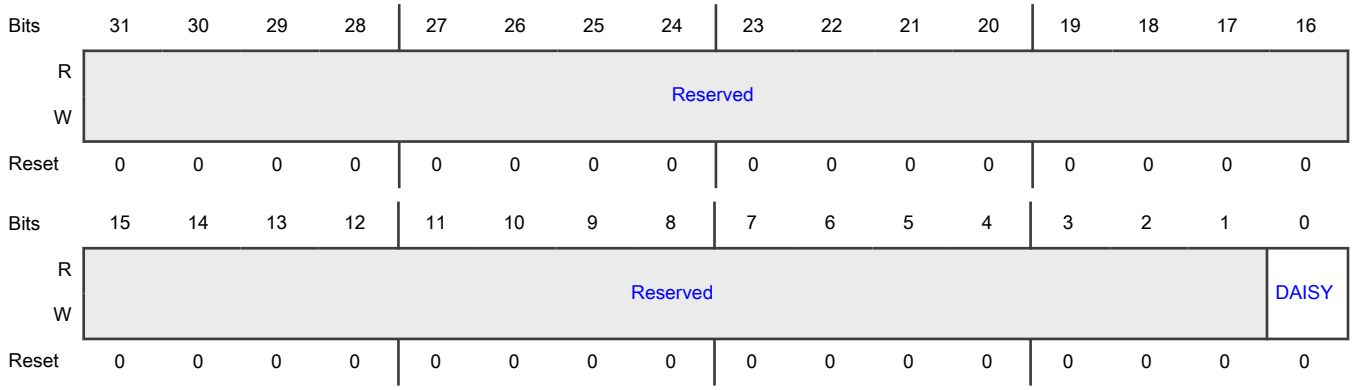
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_28	968h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in28 0b - Selecting Pad: GPIO_EMC_B2_08 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_02 for Mode: ALT4

17.4.1.551 XBAR1_XBAR_IN_SELECT_INPUT_29 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_29)

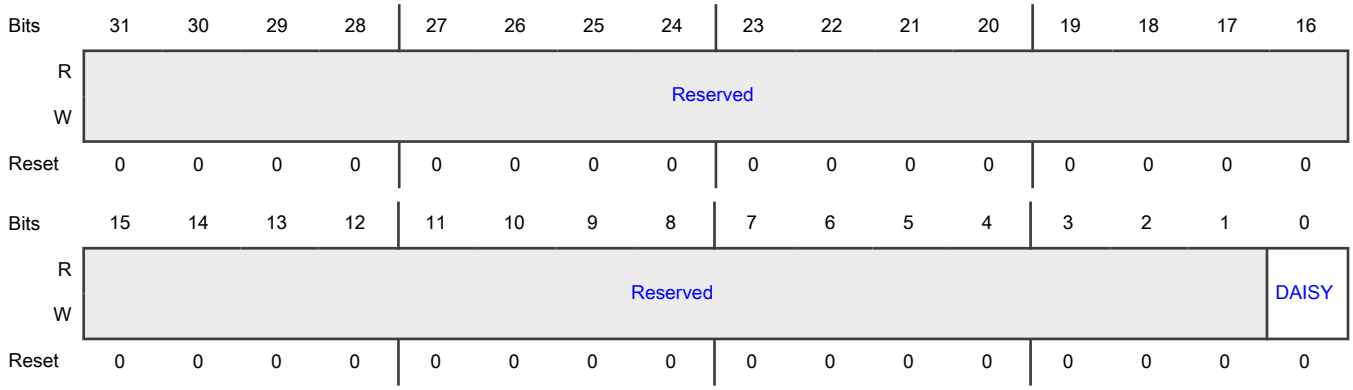
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_29	96Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in29 0b - Selecting Pad: GPIO_EMC_B2_09 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_03 for Mode: ALT4

17.4.1.552 XBAR1_XBAR_IN_SELECT_INPUT_30 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_30)

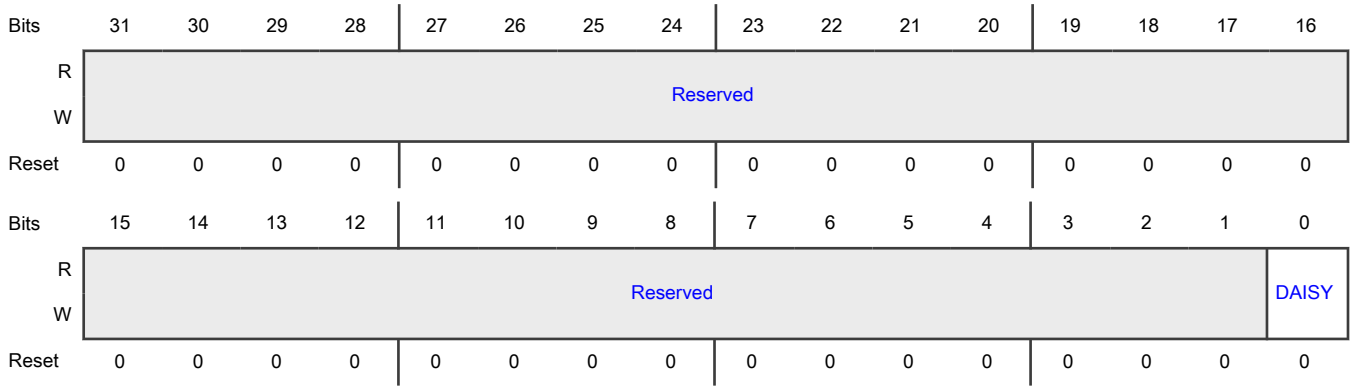
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_30	970h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in30 0b - Selecting Pad: GPIO_EMC_B2_10 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_04 for Mode: ALT4

17.4.1.553 XBAR1_XBAR_IN_SELECT_INPUT_31 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_31)

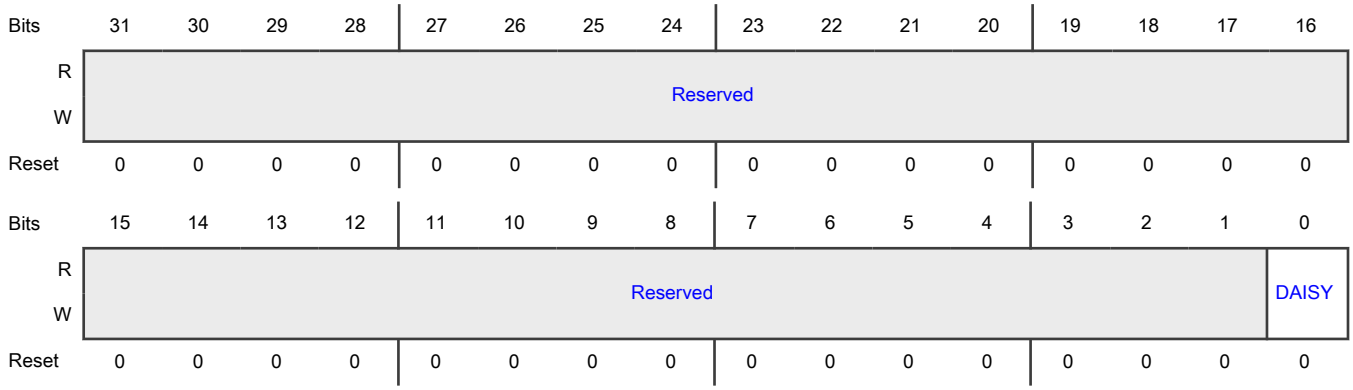
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_31	974h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in31 0b - Selecting Pad: GPIO_EMC_B2_11 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_05 for Mode: ALT4

17.4.1.554 XBAR1_XBAR_IN_SELECT_INPUT_32 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_32)

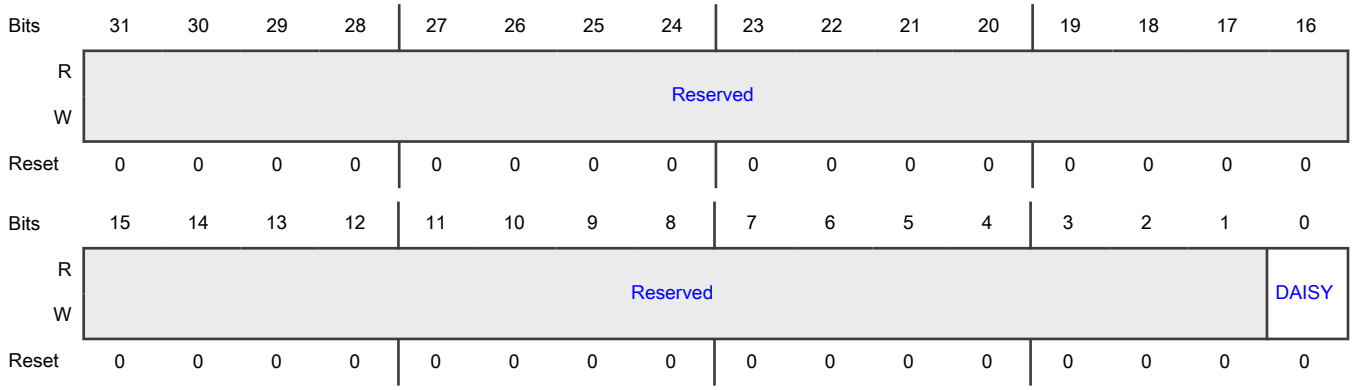
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_32	978h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in32 0b - Selecting Pad: GPIO_EMC_B2_12 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_06 for Mode: ALT4

17.4.1.555 XBAR1_XBAR_IN_SELECT_INPUT_33 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_33)

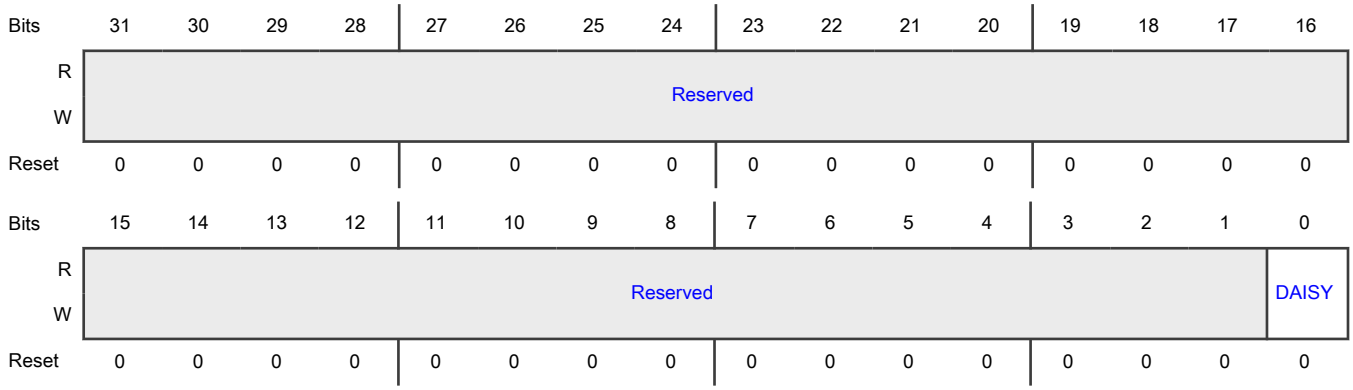
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_33	97Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in33 0b - Selecting Pad: GPIO_EMC_B2_13 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_07 for Mode: ALT4

17.4.1.556 XBAR1_XBAR_IN_SELECT_INPUT_34 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_34)

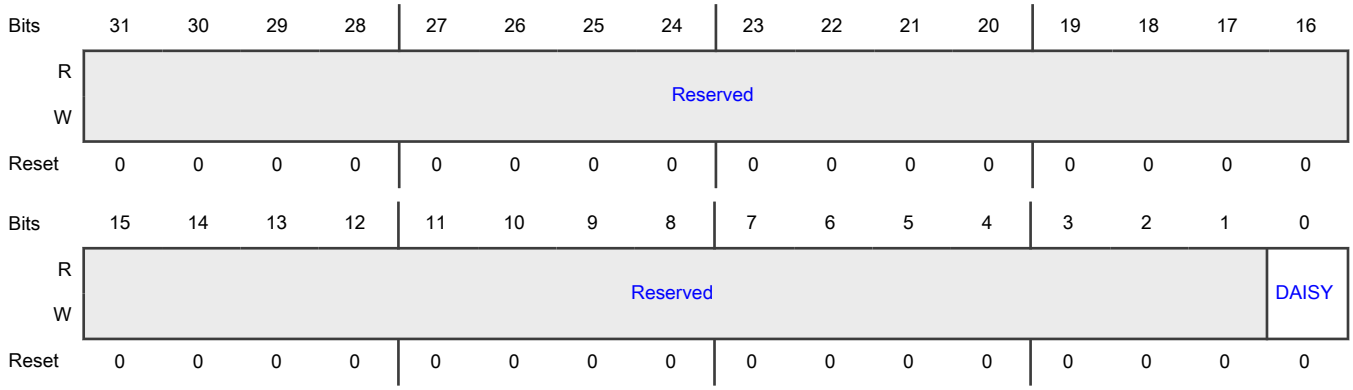
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_34	980h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in34 0b - Selecting Pad: GPIO_EMC_B2_14 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_10 for Mode: ALT4

17.4.1.557 XBAR1_XBAR_IN_SELECT_INPUT_35 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_35)

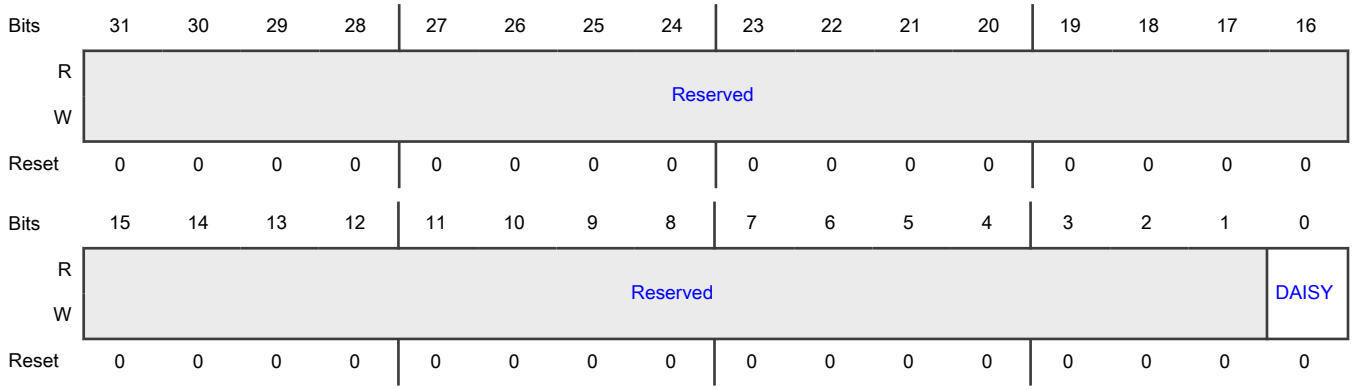
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_35	984h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in35 0b - Selecting Pad: GPIO_EMC_B2_15 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_11 for Mode: ALT4

17.4.1.558 XBAR1_XBAR_IN_SELECT_INPUT_36 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_36)

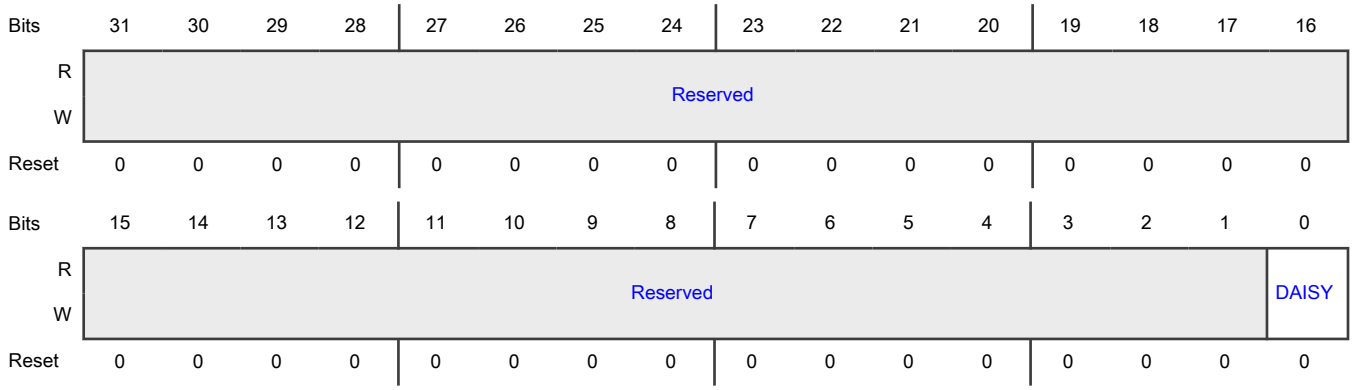
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_36	988h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in36 0b - Selecting Pad: GPIO_EMC_B2_19 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_08 for Mode: ALT4

17.4.1.559 XBAR1_XBAR_IN_SELECT_INPUT_37 DAISY Register (XBAR1_XBAR_IN_SELECT_INPUT_37)

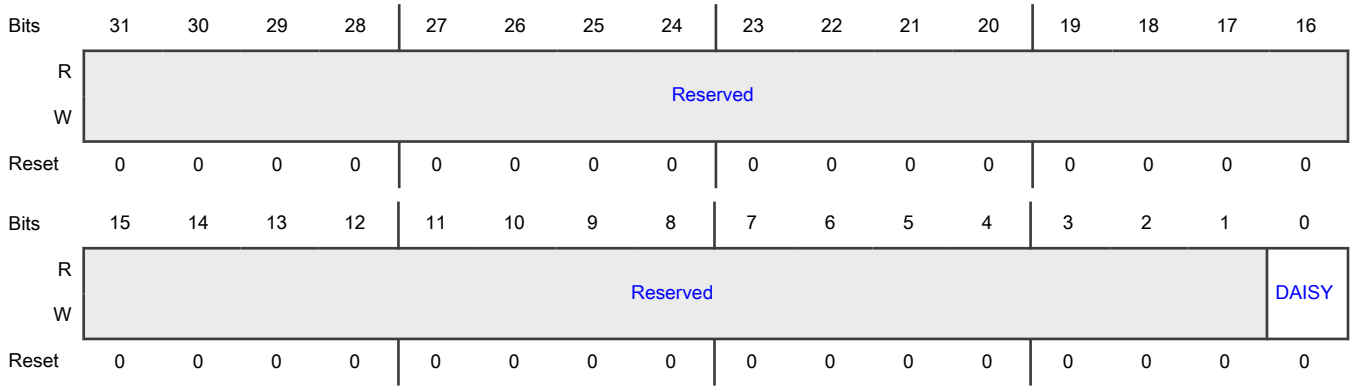
Offset

Register	Offset
XBAR1_XBAR_IN_SELECT_INPUT_37	98Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xbar1, In Pin: xbar_in37 0b - Selecting Pad: GPIO_EMC_B2_20 for Mode: ALT6 1b - Selecting Pad: GPIO_B1_09 for Mode: ALT4

17.4.1.560 XSPI_SLV_IPP_IND_CS_SELECT_INPUT DAISY Register (XSPI_SLV_IPP_IND_CS_SELECT_INPUT)

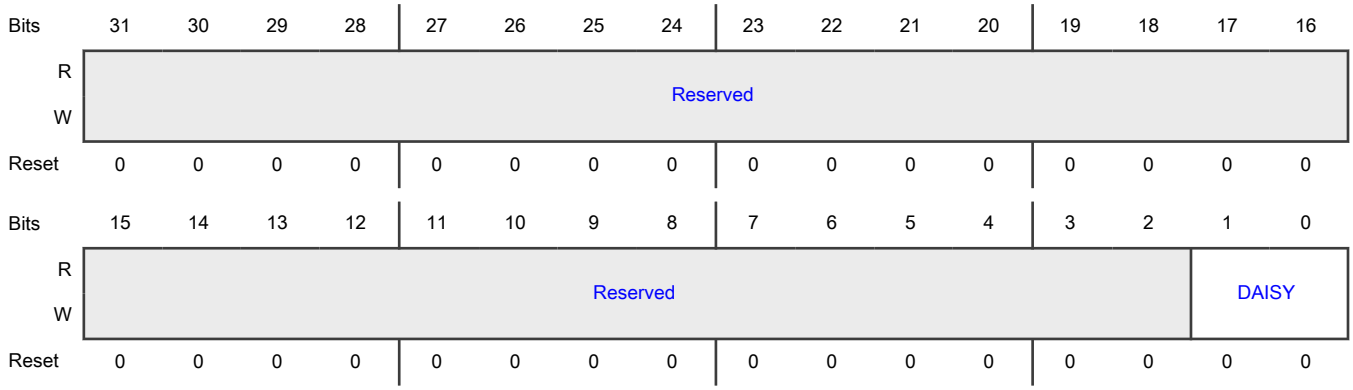
Offset

Register	Offset
XSPI_SLV_IPP_IND_CS_SELECT_INPUT	A00h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xspi_slv, In Pin: ipp_ind_cs 00b - Selecting Pad: GPIO_SD_B1_00 for Mode: ALT4 01b - Selecting Pad: GPIO_SD_B2_06 for Mode: ALT2 10b - Selecting Pad: GPIO_B2_09 for Mode: ALT10

17.4.1.561 XSPI_SLV_IPP_IND_DQS_SELECT_INPUT DAISY Register (XSPI_SLV_IPP_IND_DQS_SELECT_INPUT)

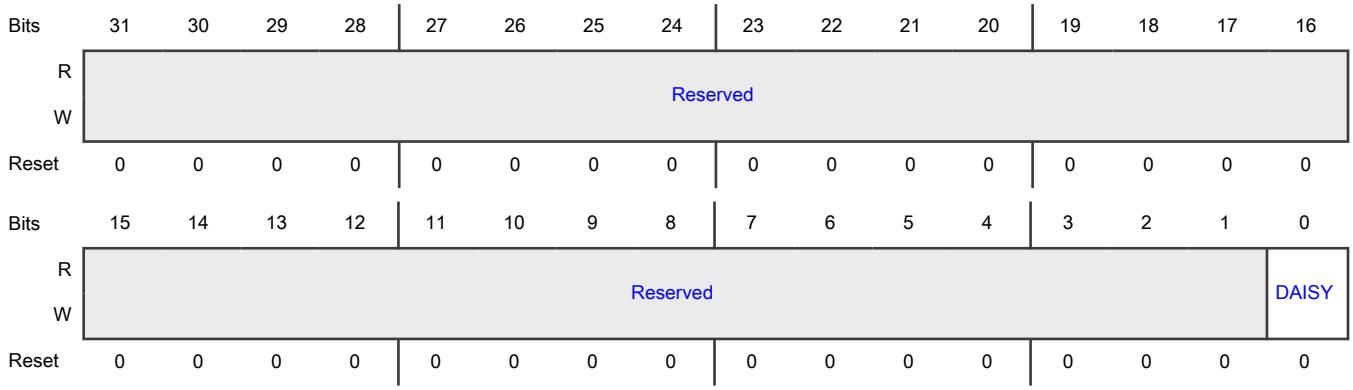
Offset

Register	Offset
XSPI_SLV_IPP_IND_DQS_SELECT_INPUT	A04h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xspi_slv, In Pin: ipp_ind_dqs 0b - Selecting Pad: GPIO_SD_B2_05 for Mode: ALT2 1b - Selecting Pad: GPIO_B2_07 for Mode: ALT10

17.4.1.562 XSPI_SLV_IPP_IND_IO_SELECT_INPUT_0 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_0)

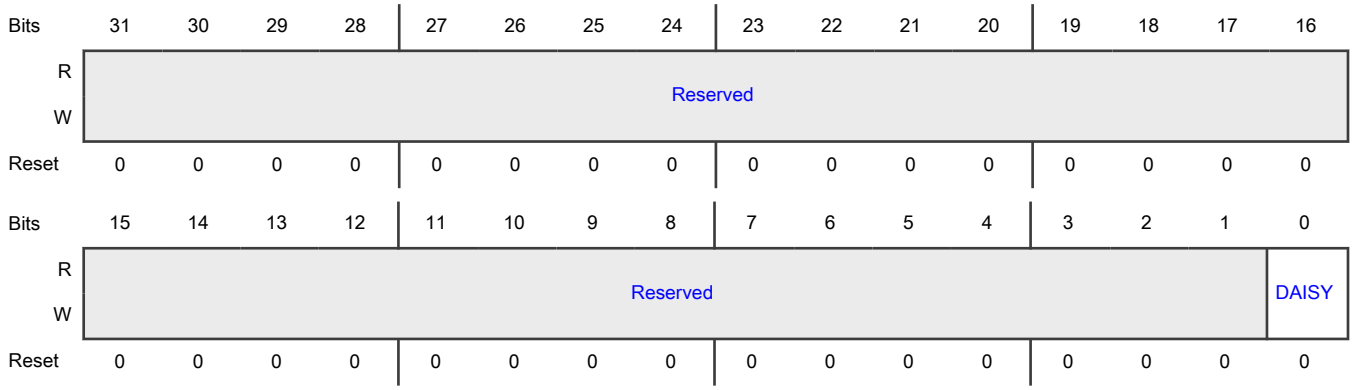
Offset

Register	Offset
XSPI_SLV_IPP_IND_IO_SELECT_INPUT_0	A08h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xspi_slv, In Pin: ipp_ind_io0 0b - Selecting Pad: GPIO_SD_B2_08 for Mode: ALT2 1b - Selecting Pad: GPIO_B2_10 for Mode: ALT10

17.4.1.563 XSPI_SLV_IPP_IND_IO_SELECT_INPUT_1 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_1)

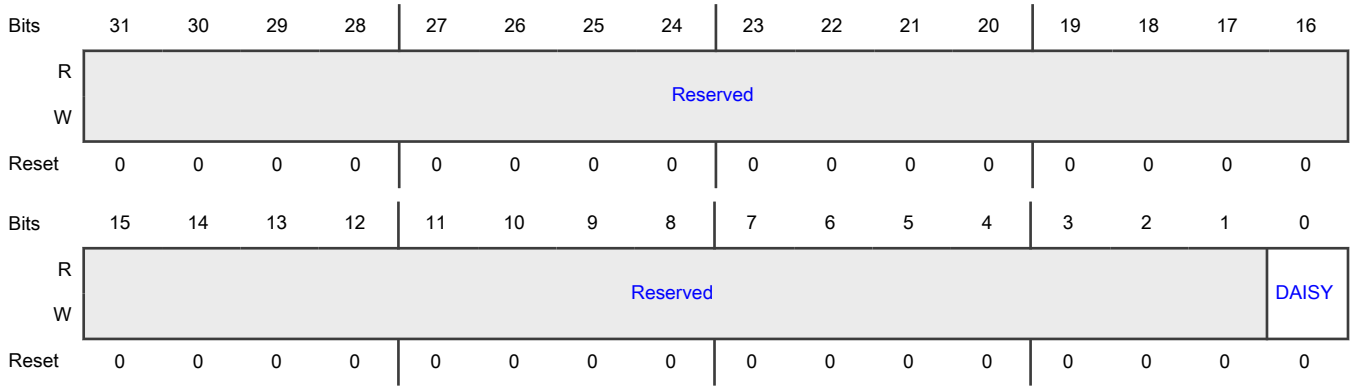
Offset

Register	Offset
XSPI_SLV_IPP_IND_IO_SELECT_INPUT_1	A0Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xspi_slv, In Pin: ipp_ind_io1 0b - Selecting Pad: GPIO_SD_B2_09 for Mode: ALT2 1b - Selecting Pad: GPIO_B2_11 for Mode: ALT10

17.4.1.564 XSPI_SLV_IPP_IND_IO_SELECT_INPUT_2 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_2)

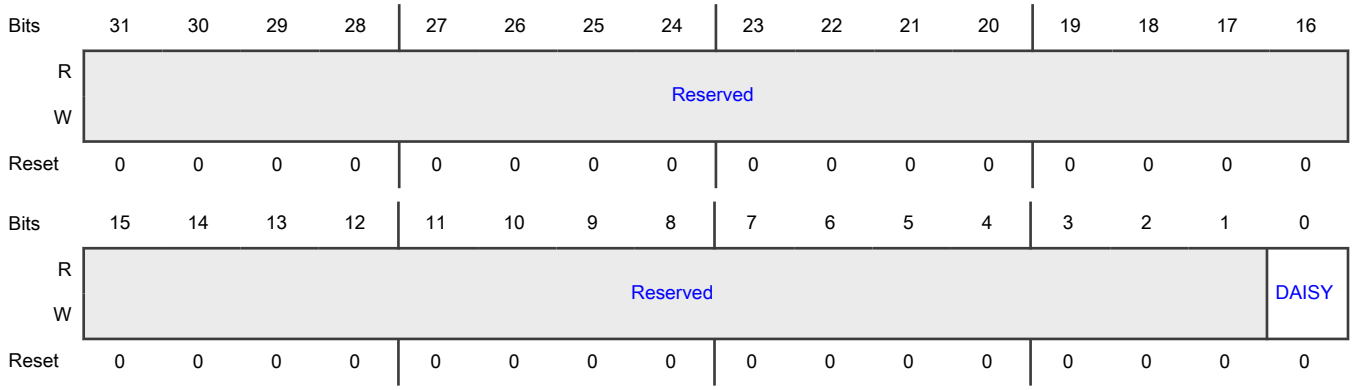
Offset

Register	Offset
XSPI_SLV_IPP_IND_IO_SELECT_INPUT_2	A10h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xspi_slv, In Pin: ipp_ind_io2 0b - Selecting Pad: GPIO_SD_B2_10 for Mode: ALT2 1b - Selecting Pad: GPIO_B2_12 for Mode: ALT10

17.4.1.565 XSPI_SLV_IPP_IND_IO_SELECT_INPUT_3 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_3)

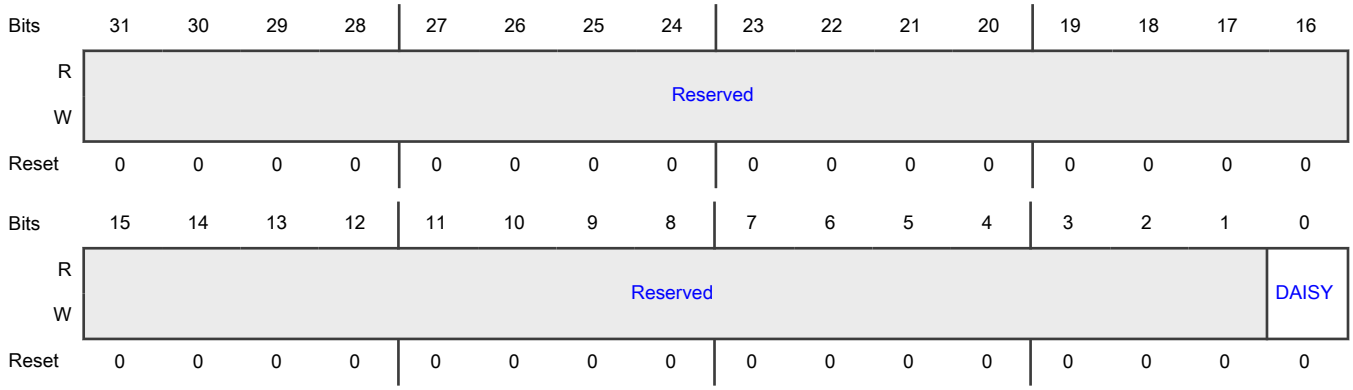
Offset

Register	Offset
XSPI_SLV_IPP_IND_IO_SELECT_INPUT_3	A14h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xspi_slv, In Pin: ipp_ind_io3 0b - Selecting Pad: GPIO_SD_B2_11 for Mode: ALT2 1b - Selecting Pad: GPIO_B2_13 for Mode: ALT10

17.4.1.566 XSPI_SLV_IPP_IND_IO_SELECT_INPUT_4 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_4)

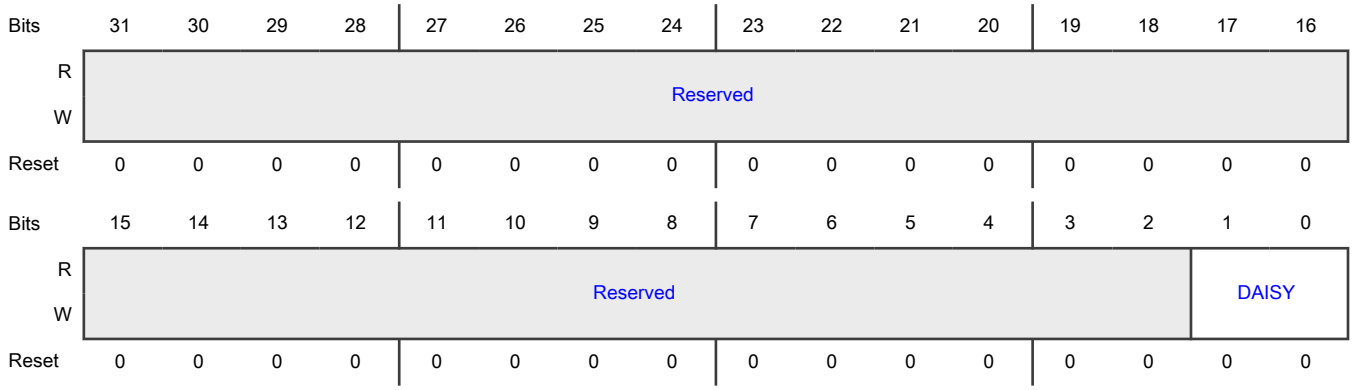
Offset

Register	Offset
XSPI_SLV_IPP_IND_IO_SELECT_INPUT_4	A18h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xspi_slv, In Pin: ipp_ind_io4 00b - Selecting Pad: GPIO_SD_B1_02 for Mode: ALT4 01b - Selecting Pad: GPIO_SD_B2_00 for Mode: ALT2 10b - Selecting Pad: GPIO_B2_03 for Mode: ALT10

17.4.1.567 XSPI_SLV_IPP_IND_IO_SELECT_INPUT_5 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_5)

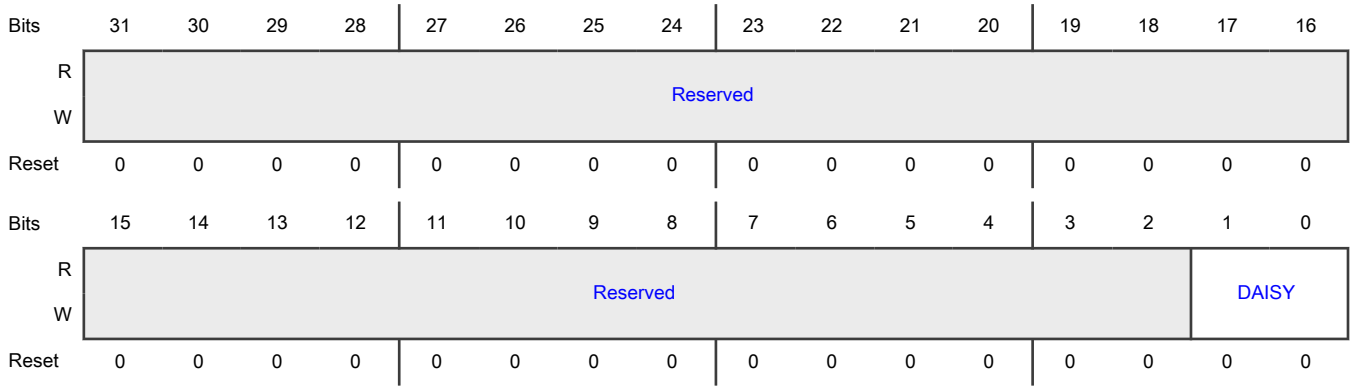
Offset

Register	Offset
XSPI_SLV_IPP_IND_IO_SELECT_INPUT_5	A1Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xspi_slv, In Pin: ipp_ind_io5 00b - Selecting Pad: GPIO_SD_B1_03 for Mode: ALT4 01b - Selecting Pad: GPIO_SD_B2_01 for Mode: ALT2 10b - Selecting Pad: GPIO_B2_04 for Mode: ALT10

17.4.1.568 XSPI_SLV_IPP_IND_IO_SELECT_INPUT_6 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_6)

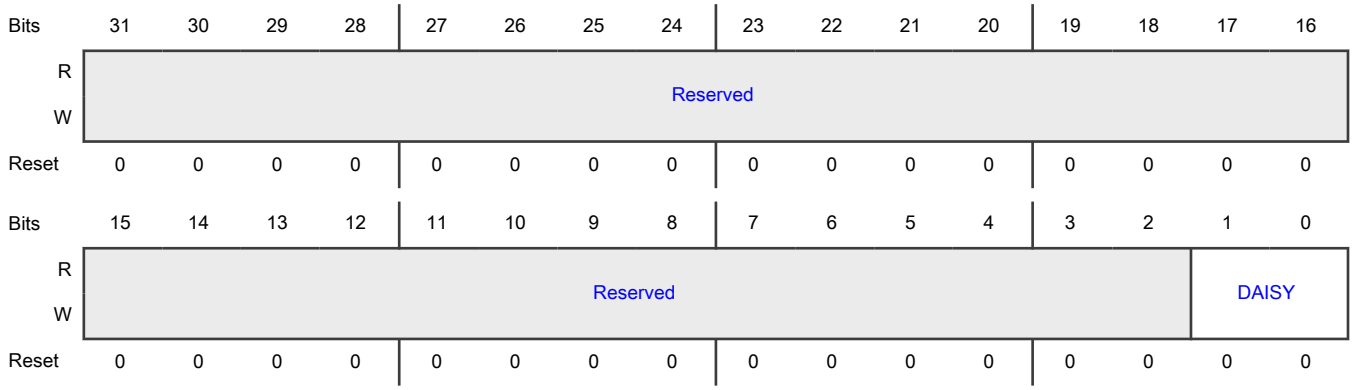
Offset

Register	Offset
XSPI_SLV_IPP_IND_IO_SELECT_INPUT_6	A20h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xspi_slv, In Pin: ipp_ind_io6 00b - Selecting Pad: GPIO_SD_B1_04 for Mode: ALT4 01b - Selecting Pad: GPIO_SD_B2_02 for Mode: ALT2 10b - Selecting Pad: GPIO_B2_05 for Mode: ALT10

17.4.1.569 XSPI_SLV_IPP_IND_IO_SELECT_INPUT_7 DAISY Register (XSPI_SLV_IPP_IND_IO_SELECT_INPUT_7)

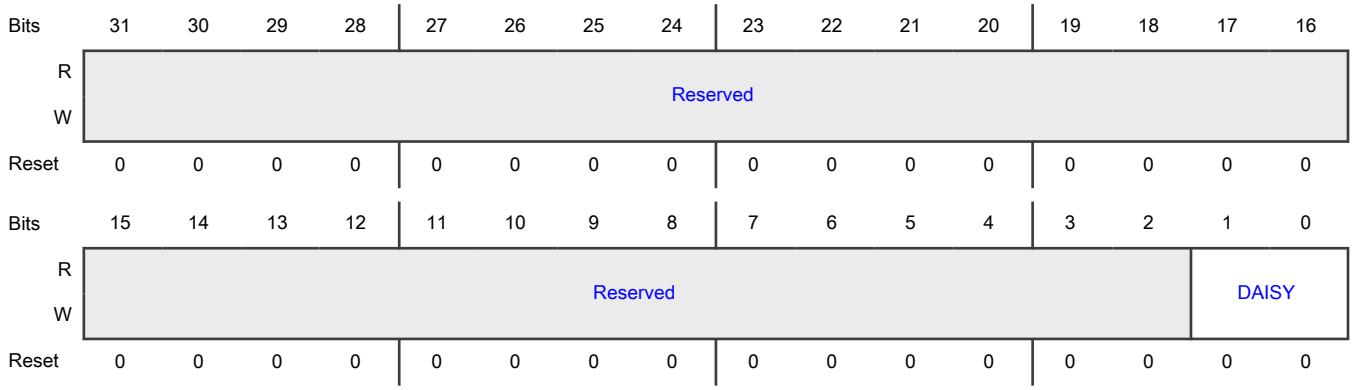
Offset

Register	Offset
XSPI_SLV_IPP_IND_IO_SELECT_INPUT_7	A24h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xspi_slv, In Pin: ipp_ind_io7 00b - Selecting Pad: GPIO_SD_B1_05 for Mode: ALT4 01b - Selecting Pad: GPIO_SD_B2_03 for Mode: ALT2 10b - Selecting Pad: GPIO_B2_06 for Mode: ALT10

17.4.1.570 XSPI_SLV_IPP_IND_SCK_SELECT_INPUT DAISY Register (XSPI_SLV_IPP_IND_SCK_SELECT_INPUT)

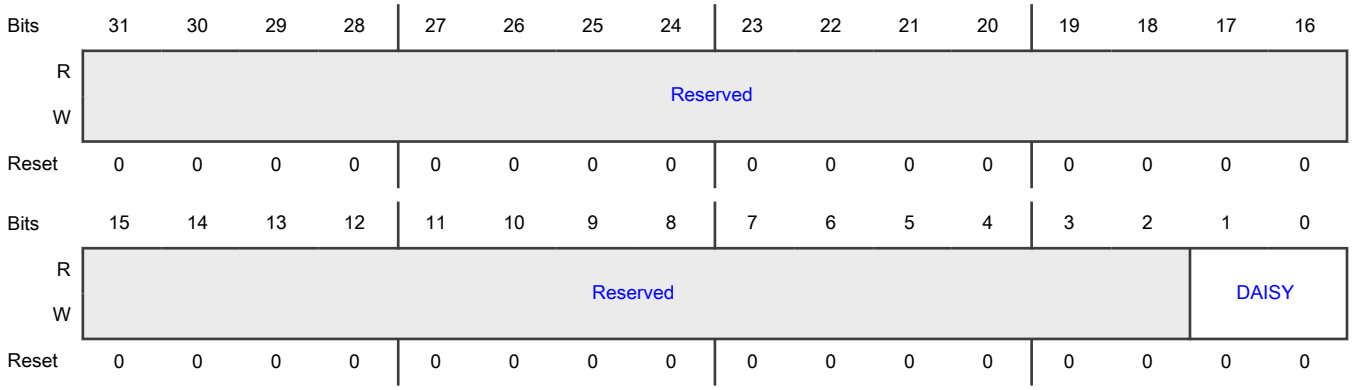
Offset

Register	Offset
XSPI_SLV_IPP_IND_SCK_SELECT_INPUT	A28h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: xspi_slv, In Pin: ipp_ind_sck 00b - Selecting Pad: GPIO_SD_B1_01 for Mode: ALT4 01b - Selecting Pad: GPIO_SD_B2_07 for Mode: ALT2 10b - Selecting Pad: GPIO_B2_08 for Mode: ALT10

17.4.2 IOMUXC_AON register descriptions

17.4.2.1 IOMUXC_AON memory map

IOMUXC_AON base address: 443C_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	SW_MUX_CTL_PAD_GPIO_AON_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_00)	32	RW	0000_0000h
4h	SW_MUX_CTL_PAD_GPIO_AON_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_01)	32	RW	0000_0000h
8h	SW_MUX_CTL_PAD_GPIO_AON_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_02)	32	RW	0000_0000h
Ch	SW_MUX_CTL_PAD_GPIO_AON_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_03)	32	RW	0000_0005h
10h	SW_MUX_CTL_PAD_GPIO_AON_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_04)	32	RW	0000_0005h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
14h	SW_MUX_CTL_PAD_GPIO_AON_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_05)	32	RW	0000_0005h
18h	SW_MUX_CTL_PAD_GPIO_AON_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_06)	32	RW	0000_0005h
1Ch	SW_MUX_CTL_PAD_GPIO_AON_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_07)	32	RW	0000_0005h
20h	SW_MUX_CTL_PAD_GPIO_AON_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_08)	32	RW	0000_0005h
24h	SW_MUX_CTL_PAD_GPIO_AON_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_09)	32	RW	0000_0005h
28h	SW_MUX_CTL_PAD_GPIO_AON_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_10)	32	RW	0000_0000h
2Ch	SW_MUX_CTL_PAD_GPIO_AON_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_11)	32	RW	0000_0000h
30h	SW_MUX_CTL_PAD_GPIO_AON_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_12)	32	RW	0000_0000h
34h	SW_MUX_CTL_PAD_GPIO_AON_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_13)	32	RW	0000_0000h
38h	SW_MUX_CTL_PAD_GPIO_AON_14 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_14)	32	RW	0000_0000h
3Ch	SW_MUX_CTL_PAD_GPIO_AON_15 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_15)	32	RW	0000_0005h
40h	SW_MUX_CTL_PAD_GPIO_AON_16 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_16)	32	RW	0000_0005h
44h	SW_MUX_CTL_PAD_GPIO_AON_17 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_17)	32	RW	0000_0005h
48h	SW_MUX_CTL_PAD_GPIO_AON_18 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_18)	32	RW	0000_0005h
4Ch	SW_MUX_CTL_PAD_GPIO_AON_19 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_19)	32	RW	0000_0005h
50h	SW_MUX_CTL_PAD_GPIO_AON_20 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_20)	32	RW	0000_0005h
54h	SW_MUX_CTL_PAD_GPIO_AON_21 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_21)	32	RW	0000_0005h
58h	SW_MUX_CTL_PAD_GPIO_AON_22 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_22)	32	RW	0000_0005h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5Ch	SW_MUX_CTL_PAD_GPIO_AON_23 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_23)	32	RW	0000_0005h
60h	SW_MUX_CTL_PAD_GPIO_AON_24 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_24)	32	RW	0000_0005h
64h	SW_MUX_CTL_PAD_GPIO_AON_25 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_25)	32	RW	0000_0005h
68h	SW_MUX_CTL_PAD_GPIO_AON_26 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_26)	32	RW	0000_0005h
6Ch	SW_MUX_CTL_PAD_GPIO_AON_27 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_27)	32	RW	0000_0005h
70h	SW_MUX_CTL_PAD_GPIO_AON_28_DUMMY SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_28_DUMMY)	32	RW	0000_0000h
74h	SW_PAD_CTL_PAD_GPIO_AON_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_00)	32	RW	0000_0006h
78h	SW_PAD_CTL_PAD_GPIO_AON_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_01)	32	RW	0000_0006h
7Ch	SW_PAD_CTL_PAD_GPIO_AON_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_02)	32	RW	0000_0006h
80h	SW_PAD_CTL_PAD_GPIO_AON_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_03)	32	RW	0000_0006h
84h	SW_PAD_CTL_PAD_GPIO_AON_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_04)	32	RW	0000_0006h
88h	SW_PAD_CTL_PAD_GPIO_AON_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_05)	32	RW	0000_0006h
8Ch	SW_PAD_CTL_PAD_GPIO_AON_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_06)	32	RW	0000_0006h
90h	SW_PAD_CTL_PAD_GPIO_AON_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_07)	32	RW	0000_0006h
94h	SW_PAD_CTL_PAD_GPIO_AON_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_08)	32	RW	0000_0006h
98h	SW_PAD_CTL_PAD_GPIO_AON_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_09)	32	RW	0000_0006h
9Ch	SW_PAD_CTL_PAD_GPIO_AON_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_10)	32	RW	0000_000Eh
A0h	SW_PAD_CTL_PAD_GPIO_AON_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_11)	32	RW	0000_000Ah

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A4h	SW_PAD_CTL_PAD_GPIO_AON_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_12)	32	RW	0000_000Eh
A8h	SW_PAD_CTL_PAD_GPIO_AON_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_13)	32	RW	0000_0006h
ACh	SW_PAD_CTL_PAD_GPIO_AON_14 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_14)	32	RW	0000_000Eh
B0h	SW_PAD_CTL_PAD_GPIO_AON_15 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_15)	32	RW	0000_0006h
B4h	SW_PAD_CTL_PAD_GPIO_AON_16 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_16)	32	RW	0000_0006h
B8h	SW_PAD_CTL_PAD_GPIO_AON_17 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_17)	32	RW	0000_0006h
BCh	SW_PAD_CTL_PAD_GPIO_AON_18 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_18)	32	RW	0000_0006h
C0h	SW_PAD_CTL_PAD_GPIO_AON_19 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_19)	32	RW	0000_000Eh
C4h	SW_PAD_CTL_PAD_GPIO_AON_20 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_20)	32	RW	0000_000Eh
C8h	SW_PAD_CTL_PAD_GPIO_AON_21 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_21)	32	RW	0000_000Eh
CCh	SW_PAD_CTL_PAD_GPIO_AON_22 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_22)	32	RW	0000_000Eh
D0h	SW_PAD_CTL_PAD_GPIO_AON_23 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_23)	32	RW	0000_0006h
D4h	SW_PAD_CTL_PAD_GPIO_AON_24 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_24)	32	RW	0000_0006h
D8h	SW_PAD_CTL_PAD_GPIO_AON_25 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_25)	32	RW	0000_0006h
DCh	SW_PAD_CTL_PAD_GPIO_AON_26 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_26)	32	RW	0000_0006h
E0h	SW_PAD_CTL_PAD_GPIO_AON_27 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_27)	32	RW	0000_000Eh
E4h	SW_PAD_CTL_PAD_GPIO_AON_28_DUMMY SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_28_DUMMY)	32	RW	0000_0006h
E8h	I3C1_PIN_SCL_IN_SELECT_INPUT DAISY Register (I3C1_PIN_SCL_IN_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
ECh	I3C1_PIN_SDA_IN_SELECT_INPUT DAISY Register (I3C1_PIN_SDA_IN_SELECT_INPUT)	32	RW	0000_0000h
F0h	LPI2C1_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C1_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	RW	0000_0000h
F4h	LPI2C1_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C1_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	RW	0000_0000h
F8h	LPI2C2_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C2_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	RW	0000_0000h
FCh	LPI2C2_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C2_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	RW	0000_0000h
100h	LPSPI1_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI1_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)	32	RW	0000_0000h
104h	LPSPI1_IPP_IND_LPSPI_PCS_SELECT_INPUT_1 DAISY Register (LPSPI1_IPP_IND_LPSPI_PCS_SELECT_INPUT_1)	32	RW	0000_0000h
108h	LPSPI1_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI1_IPP_IND_LPSPI_SCK_SELECT_INPUT)	32	RW	0000_0000h
10Ch	LPSPI1_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI1_IPP_IND_LPSPI_SDI_SELECT_INPUT)	32	RW	0000_0000h
110h	LPSPI1_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI1_IPP_IND_LPSPI_SDO_SELECT_INPUT)	32	RW	0000_0000h
114h	LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)	32	RW	0000_0000h
118h	LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_1 DAISY Register (LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_1)	32	RW	0000_0000h
11Ch	LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_3 DAISY Register (LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_3)	32	RW	0000_0000h
120h	LPSPI2_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI2_IPP_IND_LPSPI_SCK_SELECT_INPUT)	32	RW	0000_0000h
124h	LPSPI2_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI2_IPP_IND_LPSPI_SDI_SELECT_INPUT)	32	RW	0000_0000h
128h	LPSPI2_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI2_IPP_IND_LPSPI_SDO_SELECT_INPUT)	32	RW	0000_0000h
12Ch	LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_1 DAISY Register (LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_1)	32	RW	0000_0000h
130h	LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_2 DAISY Register (LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
134h	LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_3 DAISY Register (LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_3)	32	RW	0000_0000h
138h	LPUART1_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART1_IPP_IND_LPUART_CTS_N_SELECT_INPUT)	32	RW	0000_0000h
13Ch	LPUART1_IPP_IND_LPUART_DCD_N_SELECT_INPUT DAISY Register (LPUART1_IPP_IND_LPUART_DCD_N_SELECT_INPUT)	32	RW	0000_0000h
140h	LPUART1_IPP_IND_LPUART_DSR_N_SELECT_INPUT DAISY Register (LPUART1_IPP_IND_LPUART_DSR_N_SELECT_INPUT)	32	RW	0000_0000h
144h	LPUART12_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART12_IPP_IND_LPUART_CTS_N_SELECT_INPUT)	32	RW	0000_0000h
148h	LPUART12_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART12_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	RW	0000_0000h
14Ch	LPUART12_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART12_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	RW	0000_0000h
150h	LPUART2_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART2_IPP_IND_LPUART_CTS_N_SELECT_INPUT)	32	RW	0000_0000h
154h	LPUART2_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART2_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	RW	0000_0000h
158h	LPUART2_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART2_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	RW	0000_0000h
15Ch	LPUART7_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART7_IPP_IND_LPUART_CTS_N_SELECT_INPUT)	32	RW	0000_0000h
160h	LPUART7_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART7_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	RW	0000_0000h
164h	LPUART7_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART7_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	RW	0000_0000h
168h	SAI1_IPG_CLK_SAI_MCLK_SELECT_INPUT DAISY Register (SAI1_IPG_CLK_SAI_MCLK_SELECT_INPUT)	32	RW	0000_0000h
16Ch	SAI1_IPP_IND_SAI_RXBCLK_SELECT_INPUT DAISY Register (SAI1_IPP_IND_SAI_RXBCLK_SELECT_INPUT)	32	RW	0000_0000h
170h	SAI1_IPP_IND_SAI_RXDATA_SELECT_INPUT_0 DAISY Register (SAI1_IPP_IND_SAI_RXDATA_SELECT_INPUT_0)	32	RW	0000_0000h
174h	SAI1_IPP_IND_SAI_RXDATA_SELECT_INPUT_1 DAISY Register (SAI1_IPP_IND_SAI_RXDATA_SELECT_INPUT_1)	32	RW	0000_0000h
178h	SAI1_IPP_IND_SAI_RXSYNC_SELECT_INPUT DAISY Register (SAI1_IPP_IND_SAI_RXSYNC_SELECT_INPUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
17Ch	SAI1_IPP_IND_SAI_TXBCLK_SELECT_INPUT DAISY Register (SAI1_IPP_IND_SAI_TXBCLK_SELECT_INPUT)	32	RW	0000_0000h
180h	SAI1_IPP_IND_SAI_TXSYNC_SELECT_INPUT DAISY Register (SAI1_IPP_IND_SAI_TXSYNC_SELECT_INPUT)	32	RW	0000_0000h

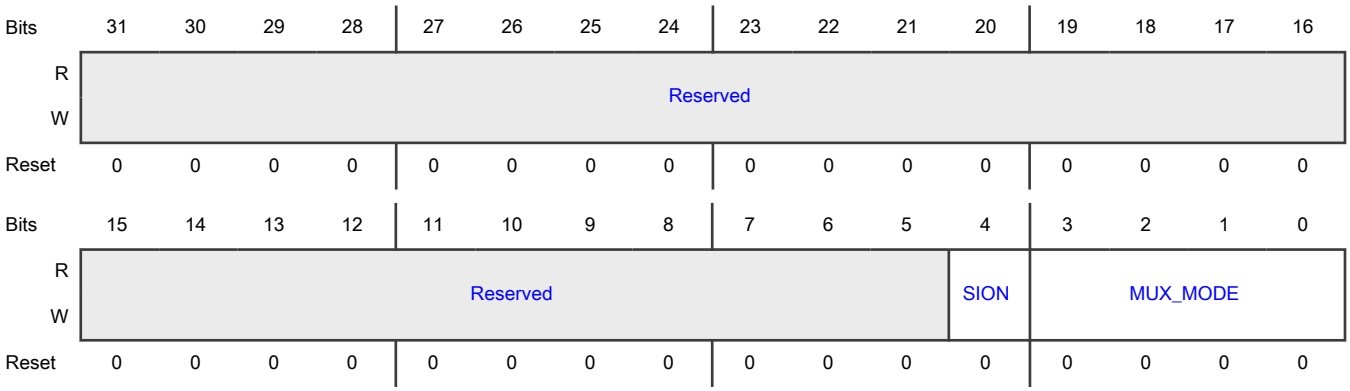
17.4.2.2 SW_MUX_CTL_PAD_GPIO_AON_00 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_00)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_00	0h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_00
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_AON_00. 0000b - Select mux mode: ALT0 mux port: SRC_BOOT_MODE00 of instance: src 0001b - Select mux mode: ALT1 mux port: CAN1_TX of instance: can1 0100b - Select mux mode: ALT4 mux port: LPTMR1_ALT1 of instance: lptmr1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO00 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPUART2_TX of instance: lpuart2 1000b - Select mux mode: ALT8 mux port: TPM1_EXTCLK of instance: tpm1 1001b - Reserved 1010b - Reserved 1011b - Reserved

17.4.2.3 SW_MUX_CTL_PAD_GPIO_AON_01 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_01)

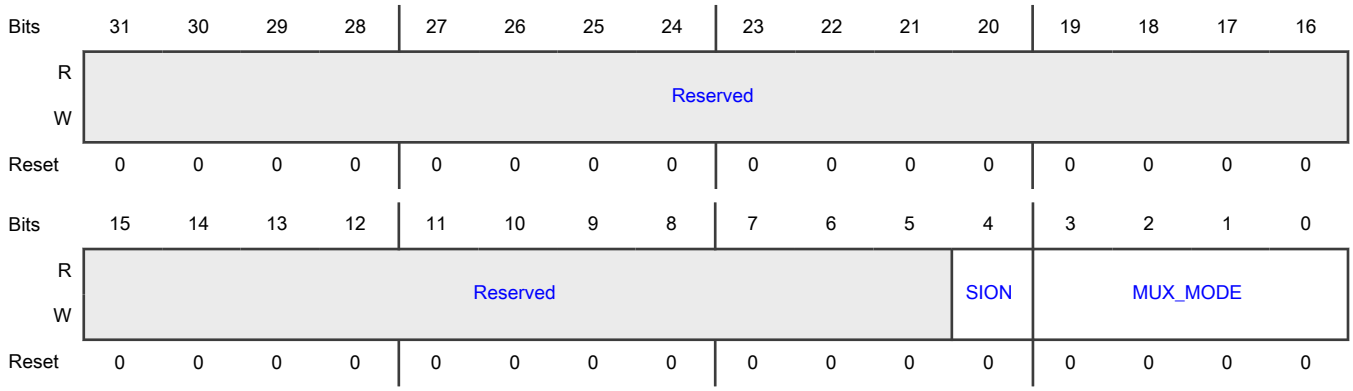
Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_01	4h

Function

SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_01
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_AON_01. 0000b - Select mux mode: ALT0 mux port: SRC_BOOT_MODE01 of instance: src 0001b - Select mux mode: ALT1 mux port: CAN1_RX of instance: can1 0100b - Select mux mode: ALT4 mux port: LPTMR1_ALT2 of instance: lptmr1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO01 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPUART2_RX of instance: lpuart2 1000b - Select mux mode: ALT8 mux port: TPM1_CH00 of instance: tpm1 1001b - Reserved 1010b - Reserved 1011b - Reserved

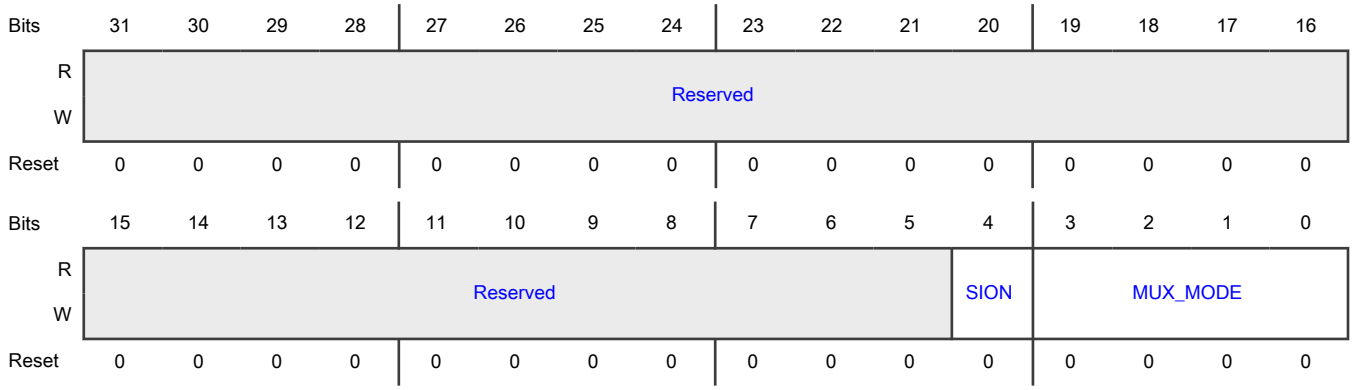
17.4.2.4 SW_MUX_CTL_PAD_GPIO_AON_02 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_02)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_02	8h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_02
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AON_02. 0000b - Select mux mode: ALT0 mux port: SRC_BOOT_MODE02 of instance: src 0001b - Select mux mode: ALT1 mux port: CAN3_TX of instance: can3 0010b - Select mux mode: ALT2 mux port: LPSPI2_PCS3 of instance: lpspi2 0011b - Select mux mode: ALT3 mux port: LPSPI2_SDO of instance: lpspi2 0100b - Select mux mode: ALT4 mux port: LPTMR1_ALT3 of instance: lptmr1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO02 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPUART2_RTS_B of instance: lpuart2 1000b - Select mux mode: ALT8 mux port: TPM1_CH01 of instance: tpm1 1001b - Reserved 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_CLK_ECAT_CLK25 of instance: ecat

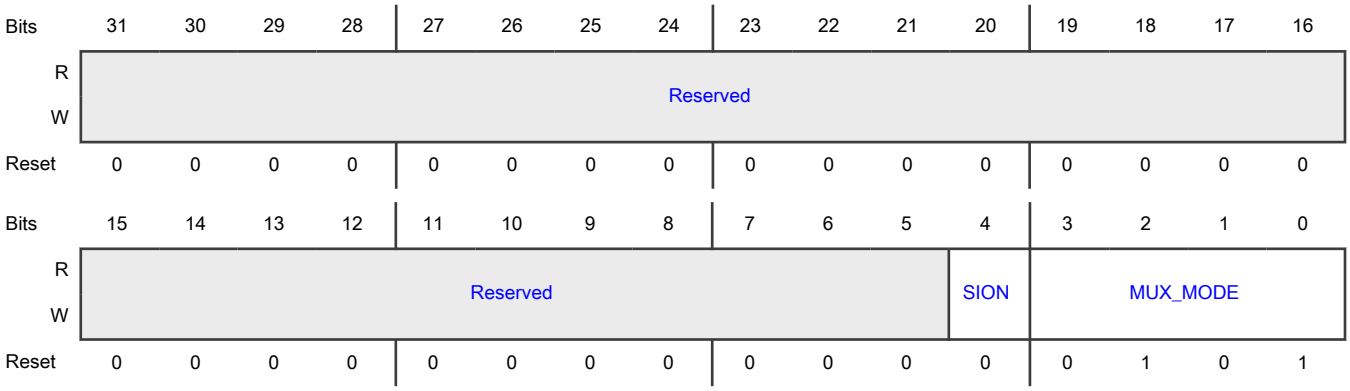
17.4.2.5 SW_MUX_CTL_PAD_GPIO_AON_03 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_03)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_03	Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_03
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AON_03. 0001b - Select mux mode: ALT1 mux port: CAN3_RX of instance: can3 0010b - Select mux mode: ALT2 mux port: LPSPI1_PCS1 of instance: lpspi1 0011b - Select mux mode: ALT3 mux port: LPSPI2_SDI of instance: lpspi2 0100b - Select mux mode: ALT4 mux port: LPSPI1_PCS3 of instance: lpspi1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Select mux mode: ALT5 mux port: GPIO1_IO03 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPUART2_CTS_B of instance: lpuart2 1000b - Select mux mode: ALT8 mux port: TPM1_CH02 of instance: tpm1 1001b - Reserved 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LED_STATE_RUN of instance: ecat

17.4.2.6 SW_MUX_CTL_PAD_GPIO_AON_04 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_04)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_04	10h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_04
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_AON_04. 0000b - Select mux mode: ALT0 mux port: LPSPI1_SCK of instance: lpspi1 0010b - Select mux mode: ALT2 mux port: SAI1_TX_DATA00 of instance: sai1 0011b - Select mux mode: ALT3 mux port: SAI1_RX_DATA01 of instance: sai1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO04 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPUART7_CTS_B of instance: lpuart7 1000b - Select mux mode: ALT8 mux port: TPM1_CH03 of instance: tpm1 1001b - Reserved 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LED_RUN of instance: ecat

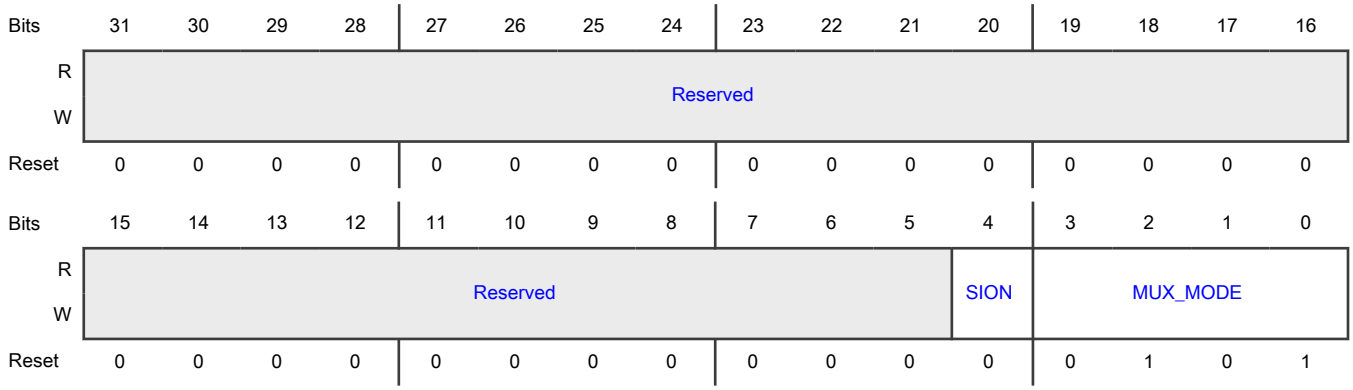
17.4.2.7 SW_MUX_CTL_PAD_GPIO_AON_05 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_05)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_05	14h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_05
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AON_05. 0000b - Select mux mode: ALT0 mux port: LPSPI1_PCS0 of instance: lpspi1 0010b - Select mux mode: ALT2 mux port: SAI1_TX_SYNC of instance: sai1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO05 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPUART7_RTS_B of instance: lpuart7 0111b - Select mux mode: ALT7 mux port: NMI_GLUE_NMI of instance: nmi_glue 1001b - Reserved 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LED_ERR of instance: ecat

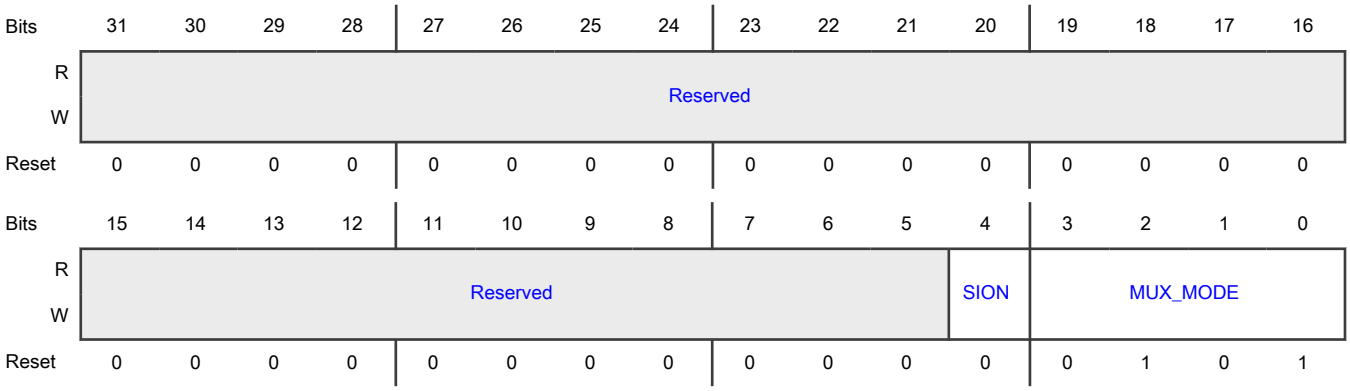
17.4.2.8 SW_MUX_CTL_PAD_GPIO_AON_06 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_06)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_06	18h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_06
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_AON_06. 0000b - Select mux mode: ALT0 mux port: LPSPI1_SDO of instance: lpspi1 0001b - Select mux mode: ALT1 mux port: I3C1_PUR of instance: i3c1 0010b - Select mux mode: ALT2 mux port: SAI1_TX_BCLK of instance: sai1 0011b - Select mux mode: ALT3 mux port: LPI2C1_SDA of instance: lpi2c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Select mux mode: ALT5 mux port: GPIO1_IO06 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: CAN1_TX of instance: can1 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_SDA of instance: ecat

17.4.2.9 SW_MUX_CTL_PAD_GPIO_AON_07 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_07)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_07	1Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_07
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AON_07. 0000b - Select mux mode: ALT0 mux port: LPSPI1_SDI of instance: lpspi1 0010b - Select mux mode: ALT2 mux port: SAI1_MCLK of instance: sai1 0011b - Select mux mode: ALT3 mux port: LPI2C1_SCL of instance: lpi2c1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO07 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: CAN1_RX of instance: can1 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_SCL of instance: ecat

17.4.2.10 SW_MUX_CTL_PAD_GPIO_AON_08 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_08)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_08	20h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_08
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_AON_08. 0000b - Select mux mode: ALT0 mux port: LPUART1_TX of instance: lpuart1 0001b - Select mux mode: ALT1 mux port: S400_TX of instance: s400 0010b - Select mux mode: ALT2 mux port: SAI1_RX_DATA00 of instance: sai1 0011b - Select mux mode: ALT3 mux port: SAI1_TX_DATA01 of instance: sai1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO08 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPI2C1_SDA of instance: lpi2c1 1000b - Select mux mode: ALT8 mux port: LPSPI1_PCS1 of instance: lpspi1 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LINK_ACT00 of instance: ecat

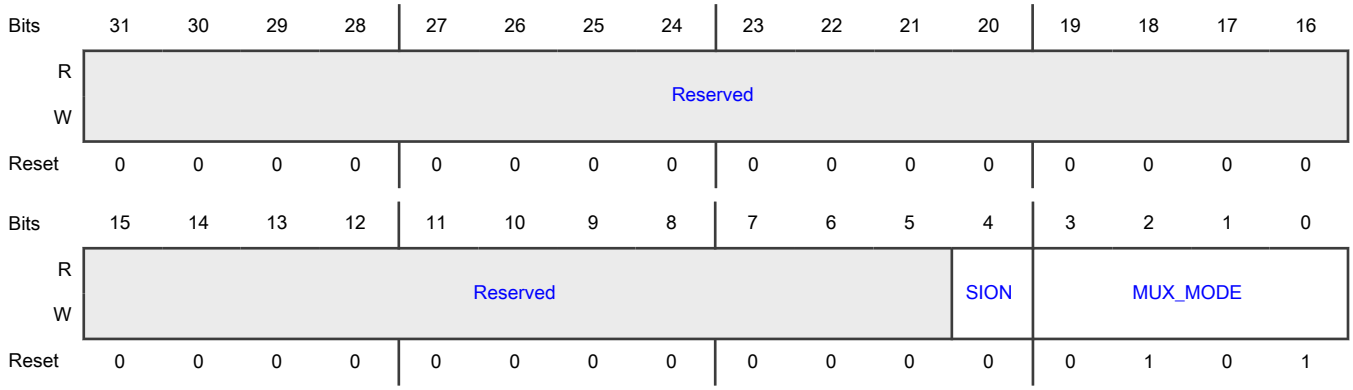
17.4.2.11 SW_MUX_CTL_PAD_GPIO_AON_09 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_09)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_09	24h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_09
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_AON_09. 0000b - Select mux mode: ALT0 mux port: LPUART1_RX of instance: lpuart1 0001b - Select mux mode: ALT1 mux port: S400_RX of instance: s400 0010b - Select mux mode: ALT2 mux port: SAI1_RX_BCLK of instance: sai1 0011b - Select mux mode: ALT3 mux port: LPIT1_TRIGGER00 of instance: lpit1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO09 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPI2C1_SCL of instance: lpi2c1 1000b - Select mux mode: ALT8 mux port: LPSPI1_PCS2 of instance: lpspi1 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ECAT_LINK_ACT01 of instance: ecat

17.4.2.12 SW_MUX_CTL_PAD_GPIO_AON_10 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_10)

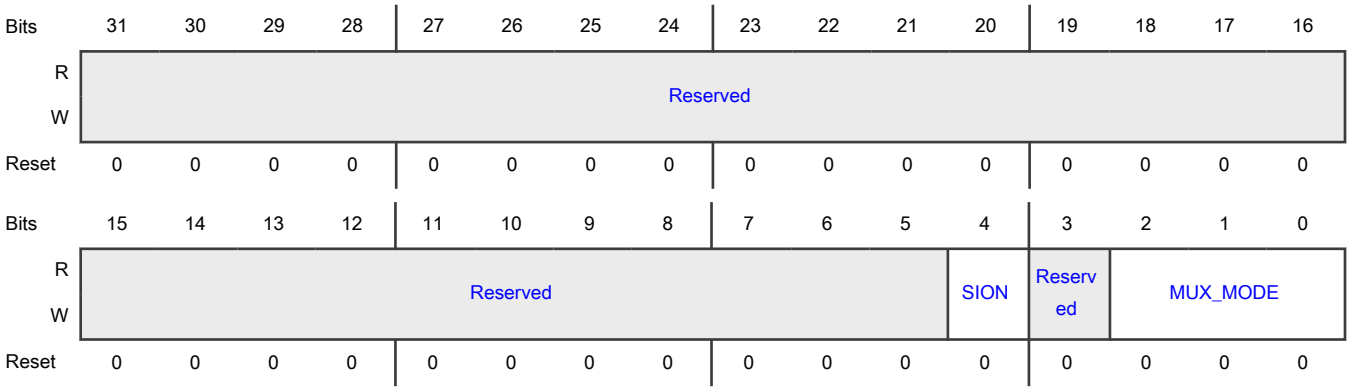
Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_10	28h

Function

SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_10
3 —	- Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for pad: GPIO_AON_10. 000b - Select mux mode: ALT0 mux port: JTAG_MUX_TRSTB of instance: jtag_mux 001b - Select mux mode: ALT1 mux port: LPSPI2_PCS0 of instance: lpspi2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010b - Select mux mode: ALT2 mux port: SAI1_RX_SYNC of instance: sai1 011b - Select mux mode: ALT3 mux port: LPIT1_TRIGGER01 of instance: lpit1 100b - Select mux mode: ALT4 mux port: TPM2_EXTCLK of instance: tpm2 101b - Select mux mode: ALT5 mux port: GPIO1_IO10 of instance: gpio1 110b - Select mux mode: ALT6 mux port: LPI2C1_SCLS of instance: lpi2c1

17.4.2.13 SW_MUX_CTL_PAD_GPIO_AON_11 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_11)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_11	2Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_11
3 —	- Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for pad: GPIO_AON_11. 000b - Select mux mode: ALT0 mux port: JTAG_MUX_TDO of instance: jtag_mux 010b - Select mux mode: ALT2 mux port: LPUART1_CTS_B of instance: lpuart1 011b - Select mux mode: ALT3 mux port: LPIT1_TRIGGER02 of instance: lpit1 100b - Select mux mode: ALT4 mux port: TPM2_CH00 of instance: tpm2 101b - Select mux mode: ALT5 mux port: GPIO1_IO11 of instance: gpio1 110b - Select mux mode: ALT6 mux port: LPI2C1_SDAS of instance: lpi2c1

17.4.2.14 SW_MUX_CTL_PAD_GPIO_AON_12 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_12)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_12	30h

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_12
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for pad: GPIO_AON_12. 0000b - Select mux mode: ALT0 mux port: JTAG_MUX_TDI of instance: jtag_mux 0010b - Select mux mode: ALT2 mux port: LPUART1_RTS_B of instance: lpuart1 0011b - Select mux mode: ALT3 mux port: LPIT1_TRIGGER03 of instance: lpit1 0100b - Select mux mode: ALT4 mux port: TPM2_CH01 of instance: tpm2 0101b - Select mux mode: ALT5 mux port: GPIO1_IO12 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPI2C1_HREQ of instance: lpi2c1 1000b - Select mux mode: ALT8 mux port: LPSPI1_SCK of instance: lpspi1

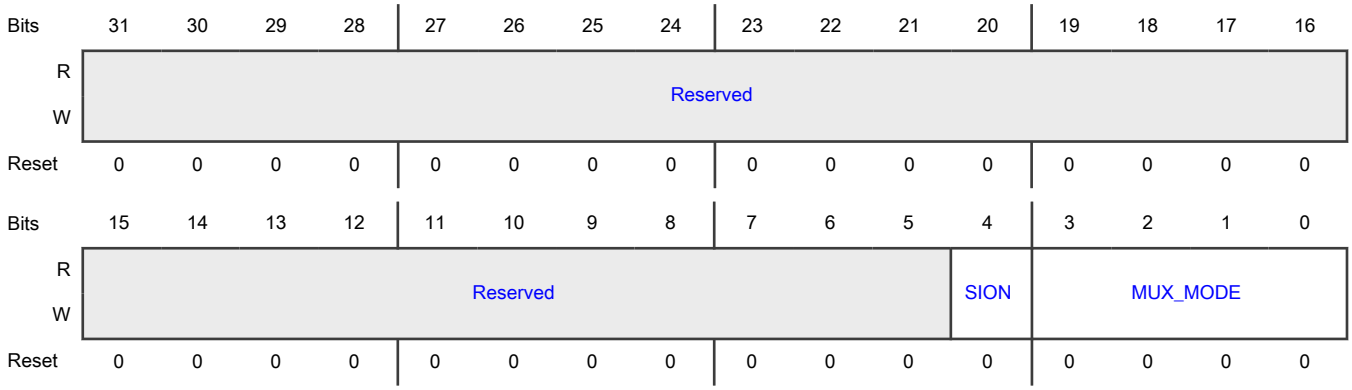
17.4.2.15 SW_MUX_CTL_PAD_GPIO_AON_13 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_13)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_13	34h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_13
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for pad: GPIO_AON_13. 0000b - Select mux mode: ALT0 mux port: JTAG_MUX_TCK of instance: jtag_mux 0010b - Select mux mode: ALT2 mux port: LPUART12_CTS_B of instance: lpuart12 0011b - Select mux mode: ALT3 mux port: LPUART1_DSR_B of instance: lpuart1 0100b - Select mux mode: ALT4 mux port: TPM2_CH02 of instance: tpm2 0101b - Select mux mode: ALT5 mux port: GPIO1_IO13 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPTMR1_ALT1 of instance: lptmr1 1000b - Select mux mode: ALT8 mux port: LPSPI1_PCS0 of instance: lpspi1

17.4.2.16 SW_MUX_CTL_PAD_GPIO_AON_14 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_14)

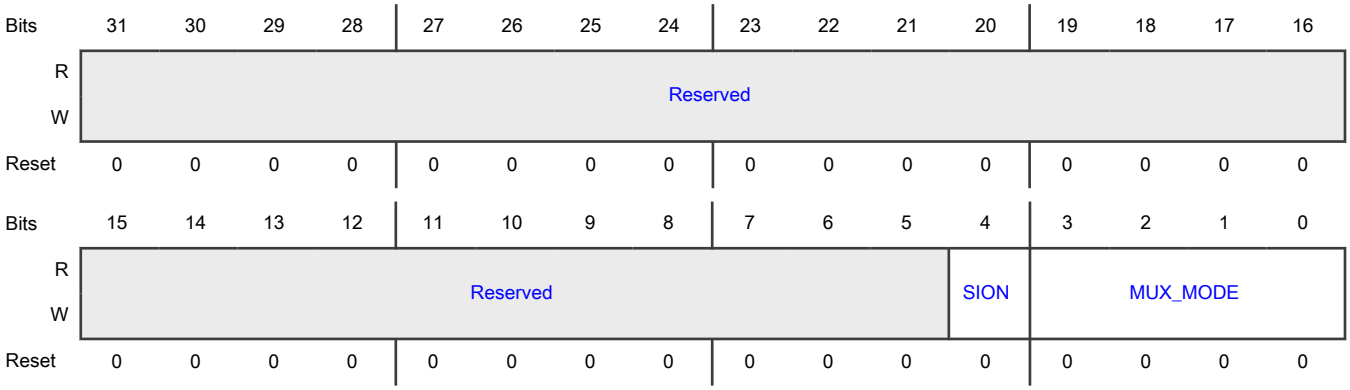
Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_14	38h

Function

SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_14
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for pad: GPIO_AON_14. 0000b - Select mux mode: ALT0 mux port: JTAG_MUX_TMS of instance: jtag_mux 0010b - Select mux mode: ALT2 mux port: LPUART12_RTS_B of instance: lpuart12 0011b - Select mux mode: ALT3 mux port: LPUART1_DCD_B of instance: lpuart1 0100b - Select mux mode: ALT4 mux port: TPM2_CH03 of instance: tpm2 0101b - Select mux mode: ALT5 mux port: GPIO1_IO14 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPTMR1_ALT2 of instance: lptmr1 1000b - Select mux mode: ALT8 mux port: LPSPI1_SDO of instance: lpspi1

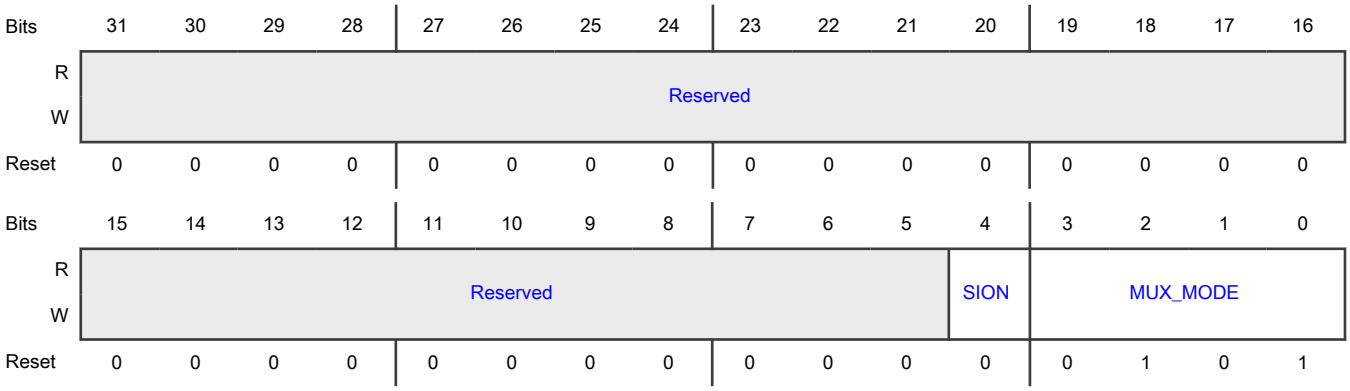
17.4.2.17 SW_MUX_CTL_PAD_GPIO_AON_15 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_15)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_15	3Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_15
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AON_15. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_B_DATA03 of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPSPI2_PCS1 of instance: lpspi2 0010b - Select mux mode: ALT2 mux port: LPUART12_TX of instance: lpuart12 0011b - Select mux mode: ALT3 mux port: LPUART1_RI_B of instance: lpuart1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: LPI2C2_SDA of instance: lpi2c2 0101b - Select mux mode: ALT5 mux port: GPIO1_IO15 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPTMR1_ALT3 of instance: lptmr1 1000b - Select mux mode: ALT8 mux port: LPSPI1_SDI of instance: lpspi1 1001b - Select mux mode: ALT9 mux port: I3C1_SDA of instance: i3c1 1010b - Reserved 1011b - Reserved

17.4.2.18 SW_MUX_CTL_PAD_GPIO_AON_16 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_16)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_16	40h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_16
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AON_16. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_B_DATA02 of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPSPI2_PCS0 of instance: lpspi2 0010b - Select mux mode: ALT2 mux port: LPUART12_RX of instance: lpuart12 0011b - Select mux mode: ALT3 mux port: LPUART1_DTR_B of instance: lpuart1 0100b - Select mux mode: ALT4 mux port: LPI2C2_SCL of instance: lpi2c2 0101b - Select mux mode: ALT5 mux port: GPIO1_IO16 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: CAN1_TX of instance: can1 1000b - Select mux mode: ALT8 mux port: LPUART7_CTS_B of instance: lpuart7 1001b - Select mux mode: ALT9 mux port: I3C1_SCL of instance: i3c1 1010b - Reserved 1011b - Reserved

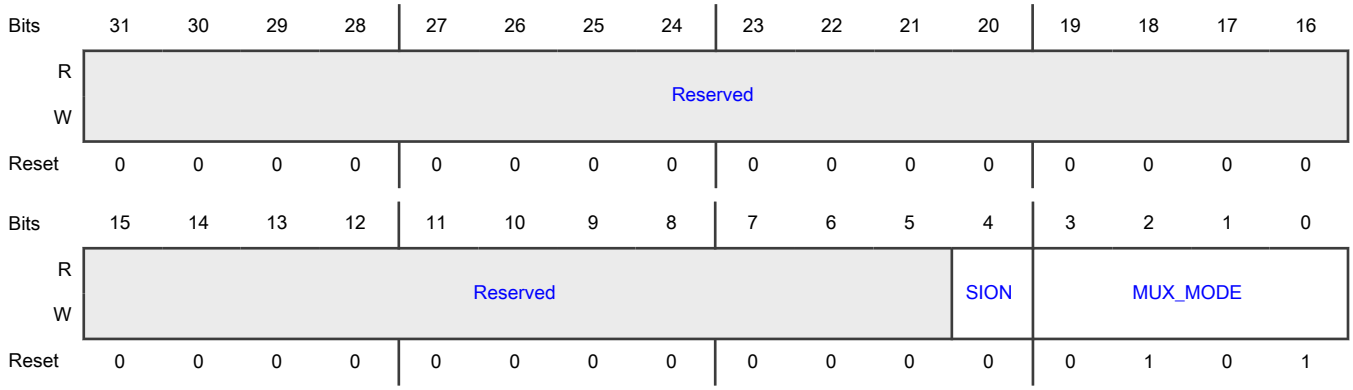
17.4.2.19 SW_MUX_CTL_PAD_GPIO_AON_17 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_17)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_17	44h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_17
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_AON_17. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_B_DATA01 of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPSPI2_SDI of instance: lpspi2 0010b - Select mux mode: ALT2 mux port: LPUART7_TX of instance: lpuart7 0011b - Select mux mode: ALT3 mux port: LPI2C2_SDA of instance: lpi2c2 0100b - Select mux mode: ALT4 mux port: LPUART1_DSR_B of instance: lpuart1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO17 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: CAN1_RX of instance: can1 1010b - Reserved 1011b - Reserved

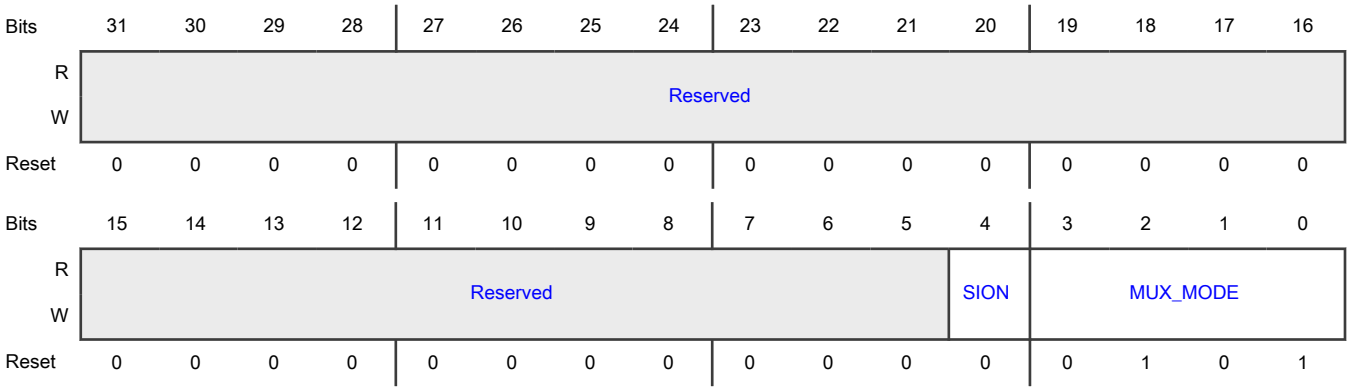
17.4.2.20 SW_MUX_CTL_PAD_GPIO_AON_18 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_18)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_18	48h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_18
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 10 iomux modes to be used for pad: GPIO_AON_18. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_B_DATA00 of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPSPI2_SDO of instance: lpspi2 0010b - Select mux mode: ALT2 mux port: LPUART7_RX of instance: lpuart7 0011b - Select mux mode: ALT3 mux port: LPI2C2_SCL of instance: lpi2c2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: LPUART1_DCD_B of instance: lpuart1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO18 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: CAN3_TX of instance: can3 1010b - Reserved 1011b - Reserved

17.4.2.21 SW_MUX_CTL_PAD_GPIO_AON_19 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_19)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_19	4Ch

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_19
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AON_19. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_B_SCLK of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPSPI2_SCK of instance: lpspi2 0011b - Select mux mode: ALT3 mux port: FLEXSPI2_BUS2BIT_A_SS1_B of instance: flexspi2_bus2bit 0100b - Select mux mode: ALT4 mux port: LPUART1_CTS_B of instance: lpuart1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO19 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: CAN3_RX of instance: can3 1000b - Select mux mode: ALT8 mux port: LPUART7_RTS_B of instance: lpuart7 1001b - Select mux mode: ALT9 mux port: LPUART12_TX of instance: lpuart12 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ADC1_CONV_D00 of instance: adc1

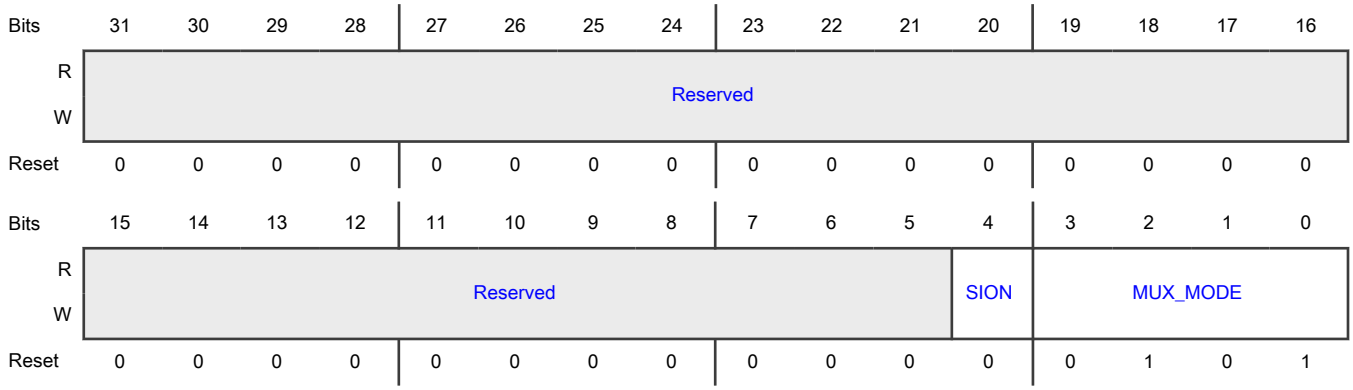
17.4.2.22 SW_MUX_CTL_PAD_GPIO_AON_20 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_20)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_20	50h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_20
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_AON_20. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_B_DQS of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: FLEXSPI2_BUS2BIT_A_SS1_B of instance: flexspi2_bus2bit 0010b - Select mux mode: ALT2 mux port: LPI2C1_SDA of instance: lpi2c1 0011b - Select mux mode: ALT3 mux port: I3C1_SDA of instance: i3c1 0100b - Select mux mode: ALT4 mux port: LPUART1_RTS_B of instance: lpuart1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO20 of instance: gpio1 1001b - Select mux mode: ALT9 mux port: LPUART12_RX of instance: lpuart12 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ADC1_CONV_D01 of instance: adc1

17.4.2.23 SW_MUX_CTL_PAD_GPIO_AON_21 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_21)

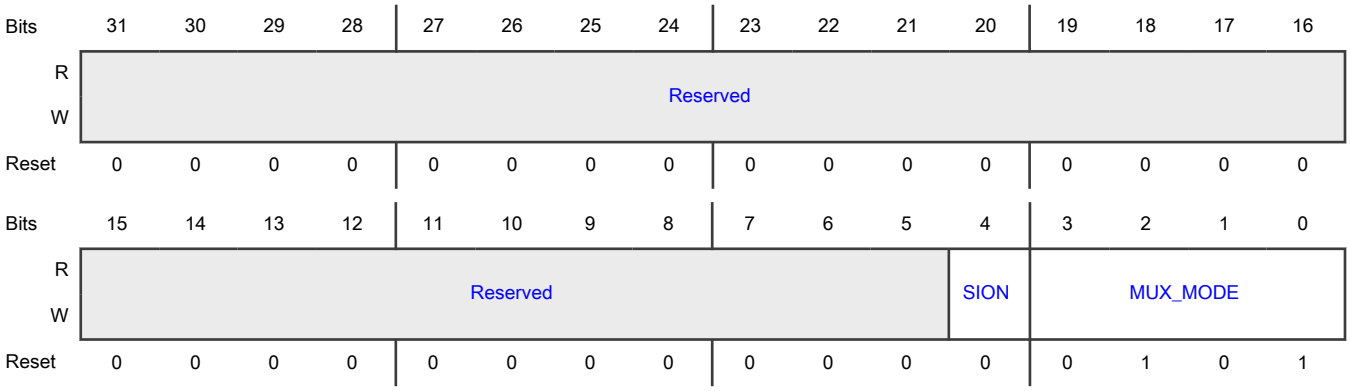
Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_21	54h

Function

SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_21
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 12 iomux modes to be used for pad: GPIO_AON_21. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_B_SS0_B of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPSPI2_PCS1 of instance: lpspi2 0010b - Select mux mode: ALT2 mux port: LPI2C1_SCL of instance: lpi2c1 0011b - Select mux mode: ALT3 mux port: I3C1_SCL of instance: i3c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: SAI1_TX_DATA00 of instance: sai1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO21 of instance: gpio1 1000b - Select mux mode: ALT8 mux port: FLEXSPI2_BUS2BIT_A_DQS of instance: flexspi2_bus2bit 1001b - Select mux mode: ALT9 mux port: SAI1_RX_DATA01 of instance: sai1 1010b - Reserved 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ADC1_CONV_D02 of instance: adc1

17.4.2.24 SW_MUX_CTL_PAD_GPIO_AON_22 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_22)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_22	58h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_22
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_AON_22. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_A_SS0_B of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPI2C2_SDA of instance: lpi2c2 0010b - Select mux mode: ALT2 mux port: LPUART7_TX of instance: lpuart7 0011b - Select mux mode: ALT3 mux port: LPUART12_CTS_B of instance: lpuart12 0100b - Select mux mode: ALT4 mux port: SAI1_TX_SYNC of instance: sai1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO22 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPSPI2_SCK of instance: lpspi2 1001b - Reserved 1010b - Select mux mode: ALT10 mux port: CCMSRCGPC_CCMOBS1 of instance: ccmsrcgpc 1100b - Select mux mode: ALT12 mux port: ADC1_CONV_D03 of instance: adc1

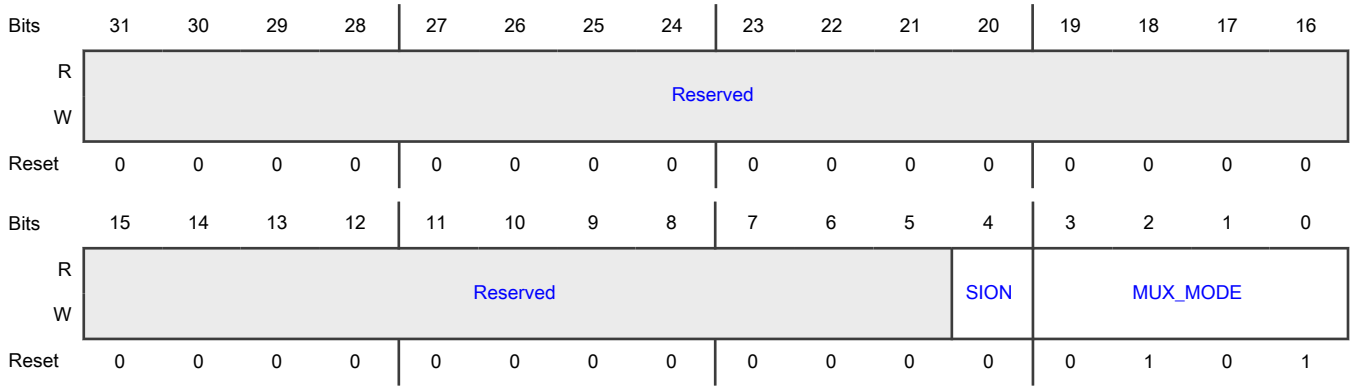
17.4.2.25 SW_MUX_CTL_PAD_GPIO_AON_23 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_23)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_23	5Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_23
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_AON_23. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_A_SCLK of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPI2C2_SCL of instance: lpi2c2 0010b - Select mux mode: ALT2 mux port: LPUART7_RX of instance: lpuart7 0011b - Select mux mode: ALT3 mux port: LPUART12_RTS_B of instance: lpuart12 0100b - Select mux mode: ALT4 mux port: SAI1_TX_BCLK of instance: sai1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO23 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPSPI2_SDO of instance: lpspi2 1001b - Reserved 1010b - Select mux mode: ALT10 mux port: CCMSRCGPC_CCMOBS2 of instance: ccmsrcgpc 1100b - Select mux mode: ALT12 mux port: ADC1_CONV_D04 of instance: adc1

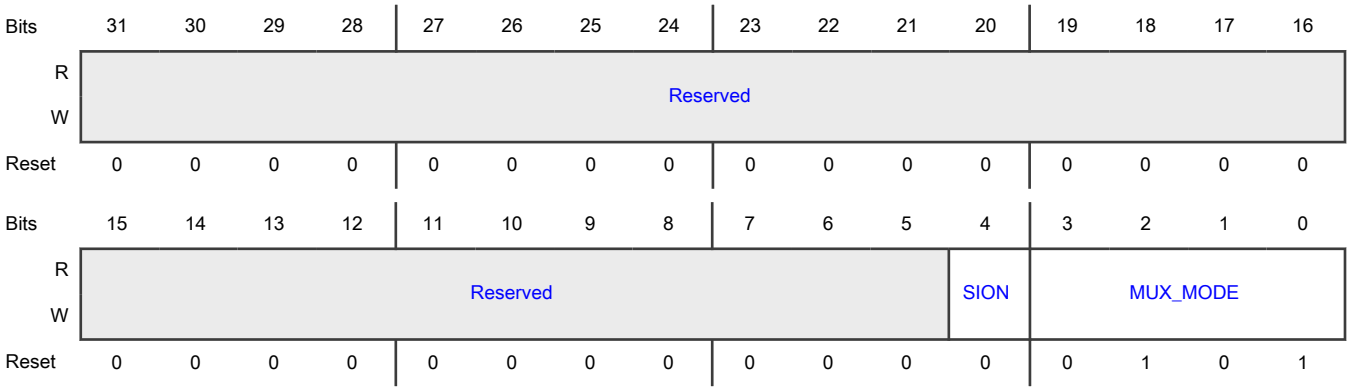
17.4.2.26 SW_MUX_CTL_PAD_GPIO_AON_24 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_24)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_24	60h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_24
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AON_24. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_A_DATA00 of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPI2C1_SDA of instance: lpi2c1 0010b - Select mux mode: ALT2 mux port: LPUART2_RTS_B of instance: lpuart2 0011b - Select mux mode: ALT3 mux port: LPUART7_CTS_B of instance: lpuart7

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - Select mux mode: ALT4 mux port: SAI1_MCLK of instance: sai1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO24 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPSPI2_SDI of instance: lpspi2 1100b - Select mux mode: ALT12 mux port: ADC1_CONV_D05 of instance: adc1

17.4.2.27 SW_MUX_CTL_PAD_GPIO_AON_25 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_25)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_25	64h

Function
SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Force input path of pad GPIO_AON_25
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 11 iomux modes to be used for pad: GPIO_AON_25. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_A_DATA01 of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPI2C1_SCL of instance: lpi2c1 0010b - Select mux mode: ALT2 mux port: LPUART2_CTS_B of instance: lpuart2 0011b - Select mux mode: ALT3 mux port: LPUART7_RTS_B of instance: lpuart7 0100b - Select mux mode: ALT4 mux port: SAI1_RX_DATA00 of instance: sai1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO25 of instance: gpio1 0110b - Select mux mode: ALT6 mux port: LPSPI2_PCS0 of instance: lpspi2 0111b - 1001b - Reserved 1100b - Select mux mode: ALT12 mux port: ADC1_CONV_D06 of instance: adc1

17.4.2.28 SW_MUX_CTL_PAD_GPIO_AON_26 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_26)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_26	68h

Function

SW_MUX_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	MUX_MODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_26
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for pad: GPIO_AON_26. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_A_DATA02 of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPSPI2_PCS2 of instance: lpspi2 0010b - Select mux mode: ALT2 mux port: LPUART2_TX of instance: lpuart2 0100b - Select mux mode: ALT4 mux port: SAI1_RX_BCLK of instance: sai1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO26 of instance: gpio1 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ADC1_CONV_D07 of instance: adc1

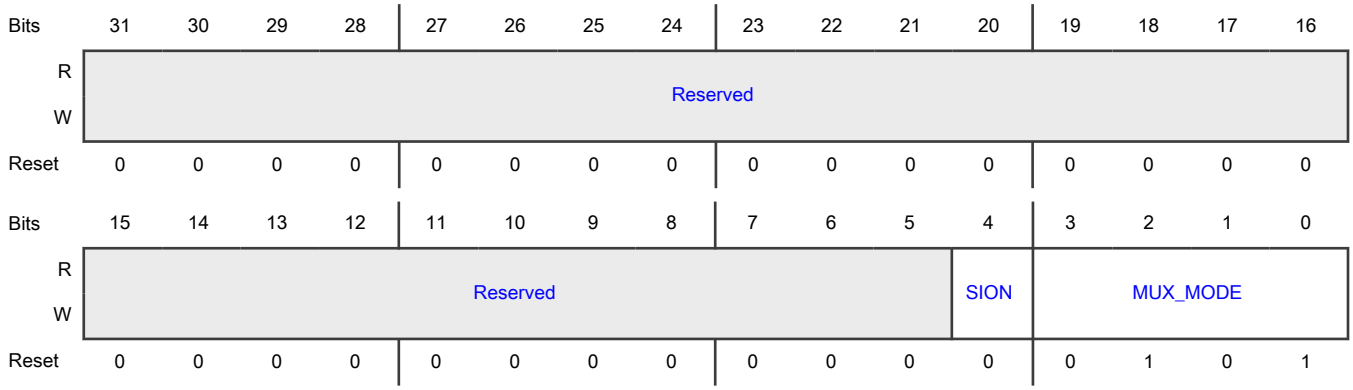
17.4.2.29 SW_MUX_CTL_PAD_GPIO_AON_27 SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_27)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_27	6Ch

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_27
3-0 MUX_MODE	MUX Mode Select Field. Select 1 of 9 iomux modes to be used for pad: GPIO_AON_27. 0000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_A_DATA03 of instance: flexspi2_bus2bit 0001b - Select mux mode: ALT1 mux port: LPSPI2_PCS3 of instance: lpspi2 0010b - Select mux mode: ALT2 mux port: LPUART2_RX of instance: lpuart2 0100b - Select mux mode: ALT4 mux port: SAI1_RX_SYNC of instance: sai1 0101b - Select mux mode: ALT5 mux port: GPIO1_IO27 of instance: gpio1 0111b - Select mux mode: ALT7 mux port: EWM_EWM_OUT_B of instance: ewm 1011b - Reserved 1100b - Select mux mode: ALT12 mux port: ADC1_CONV_RDY_CLK of instance: adc1

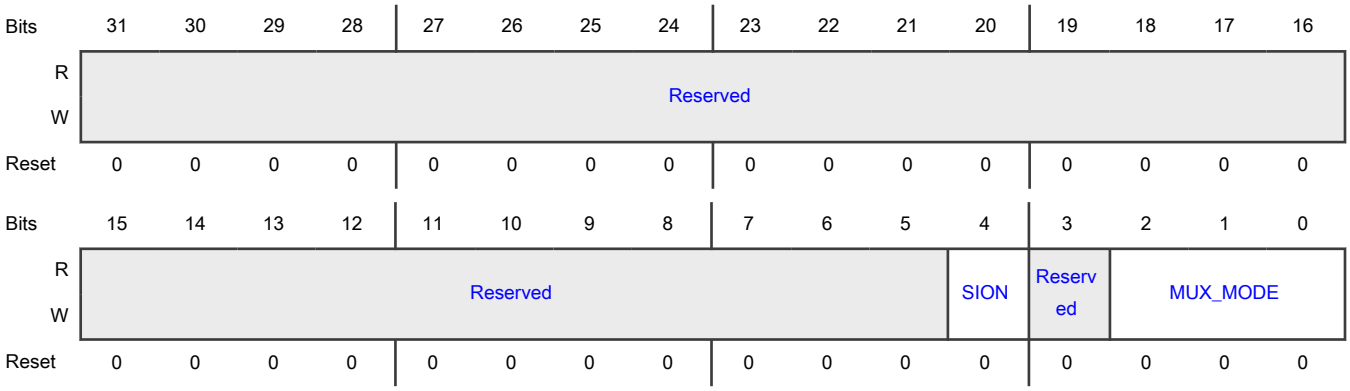
17.4.2.30 SW_MUX_CTL_PAD_GPIO_AON_28_DUMMY SW MUX Control Register (SW_MUX_CTL_PAD_GPIO_AON_28_DUMMY)

Offset

Register	Offset
SW_MUX_CTL_PAD_GPIO_AON_28_DUMMY	70h

Function
SW_MUX_CTL Register

Diagram



Fields

Field	Function
31-5 —	- Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 0b - Input Path is determined by functionality 1b - Force input path of pad GPIO_AON_28_DUMMY
3 —	- Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for pad: GPIO_AON_28_DUMMY. 000b - Select mux mode: ALT0 mux port: FLEXSPI2_BUS2BIT_A_DQS of instance: flexspi2_bus2bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - Select mux mode: ALT1 mux port: FLEXSPI2_BUS2BIT_B_DQS of instance: flexspi2_bus2bit 101b - Select mux mode: ALT5 mux port: GPIO1_IO28 of instance: gpio1

17.4.2.31 SW_PAD_CTL_PAD_GPIO_AON_00 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_00)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_00	74h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_00 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_00 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_00 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_00 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_00 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.32 SW_PAD_CTL_PAD_GPIO_AON_01 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_01)

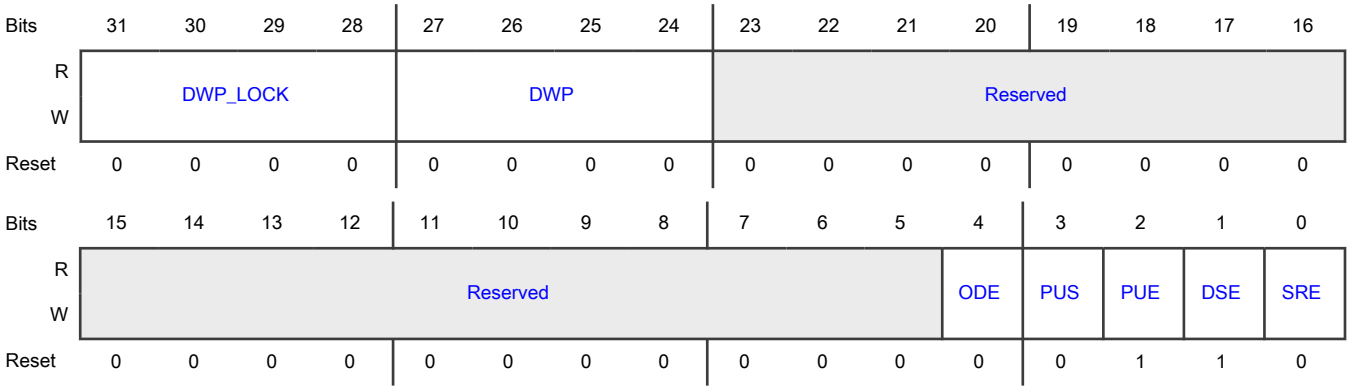
Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_01	78h

Function

SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_01 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_01 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_01

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_01 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_01 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.33 SW_PAD_CTL_PAD_GPIO_AON_02 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_02)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_02	7Ch

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_02 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_02 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_02 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_02 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_02 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.34 SW_PAD_CTL_PAD_GPIO_AON_03 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_03)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_03	80h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_03 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_03 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_03 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_03 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_03 0b - Fast Slew Rate 1b - Slow Slew Rate

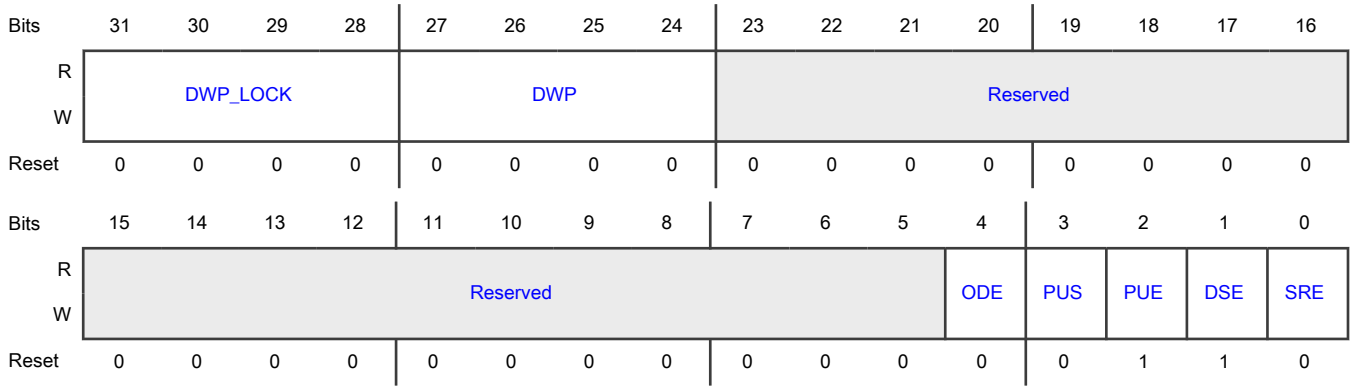
17.4.2.35 SW_PAD_CTL_PAD_GPIO_AON_04 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_04)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_04	84h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_04 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_04 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_04 0b - Pull Disable, Highz 1b - Pull Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_04 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_04 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.36 SW_PAD_CTL_PAD_GPIO_AON_05 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_05)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_05	88h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28	Domain write protection lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
DWP_LOCK	Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_05 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_05 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_05 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_05 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_05 0b - Fast Slew Rate 1b - Slow Slew Rate

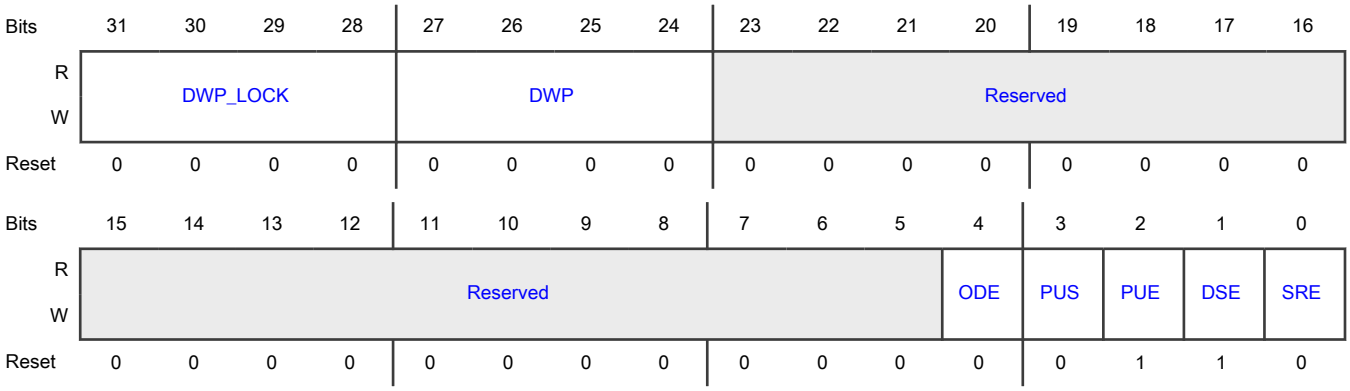
17.4.2.37 SW_PAD_CTL_PAD_GPIO_AON_06 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_06)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_06	8Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_06 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_06 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_06 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_06 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_06 0b - Fast Slew Rate 1b - Slow Slew Rate

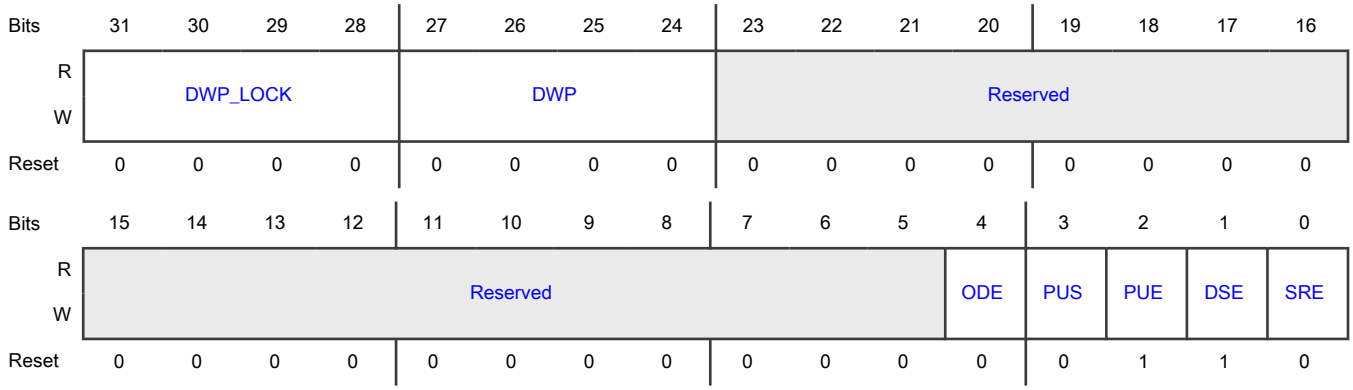
17.4.2.38 SW_PAD_CTL_PAD_GPIO_AON_07 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_07)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_07	90h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_07 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_07 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_07 0b - Pull Disable, Highz 1b - Pull Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_07 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_07 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.39 SW_PAD_CTL_PAD_GPIO_AON_08 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_08)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_08	94h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28	Domain write protection lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
DWP_LOCK	Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_08 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_08 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_08 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_08 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_08 0b - Fast Slew Rate 1b - Slow Slew Rate

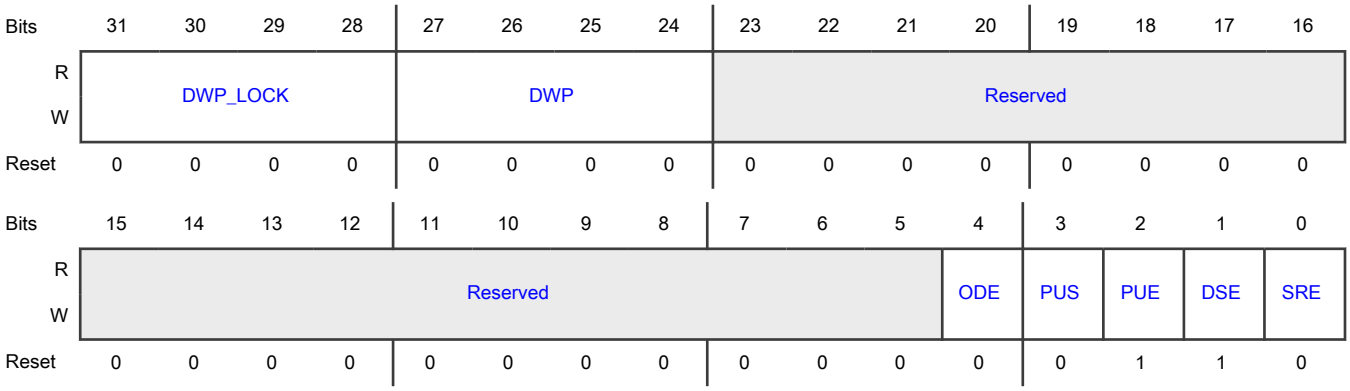
17.4.2.40 SW_PAD_CTL_PAD_GPIO_AON_09 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_09)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_09	98h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_09 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_09 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_09 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_09 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_09 0b - Fast Slew Rate 1b - Slow Slew Rate

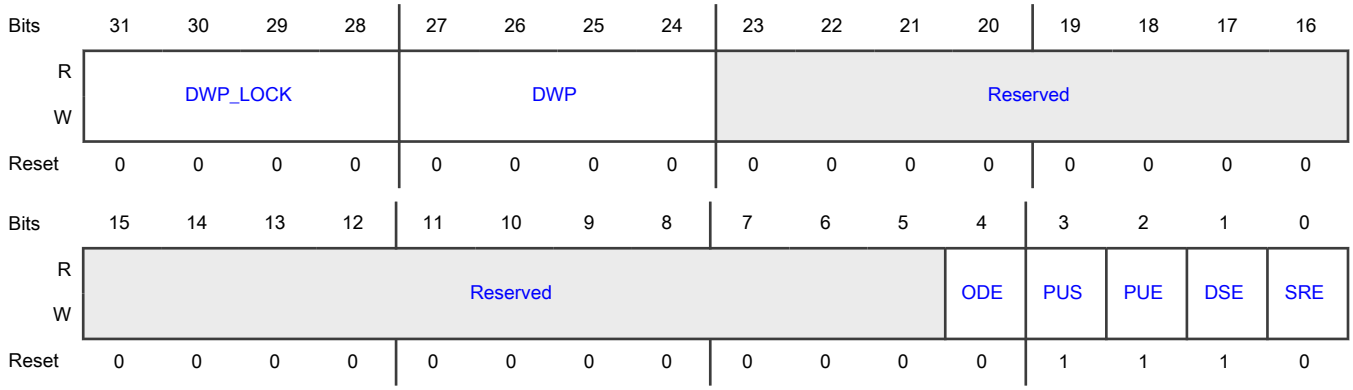
17.4.2.41 SW_PAD_CTL_PAD_GPIO_AON_10 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_10)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_10	9Ch

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_10 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_10 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_10 0b - Pull Disable, Highz 1b - Pull Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_10 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_10 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.42 SW_PAD_CTL_PAD_GPIO_AON_11 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_11)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_11	A0h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

Fields

Field	Function
31-28	Domain write protection lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
DWP_LOCK	Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_11 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_11 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_11 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_11 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_11 0b - Fast Slew Rate 1b - Slow Slew Rate

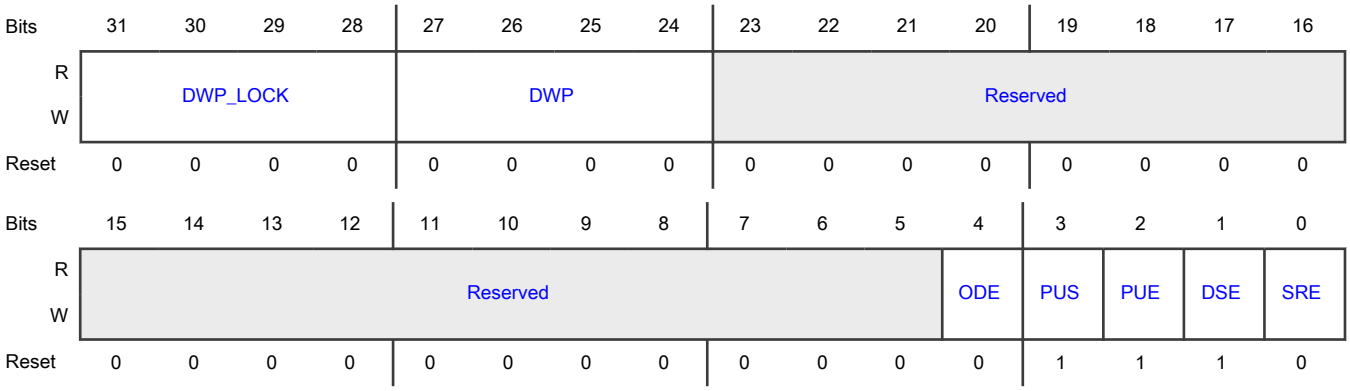
17.4.2.43 SW_PAD_CTL_PAD_GPIO_AON_12 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_12)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_12	A4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_12 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_12 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_12 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_12 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_12 0b - Fast Slew Rate 1b - Slow Slew Rate

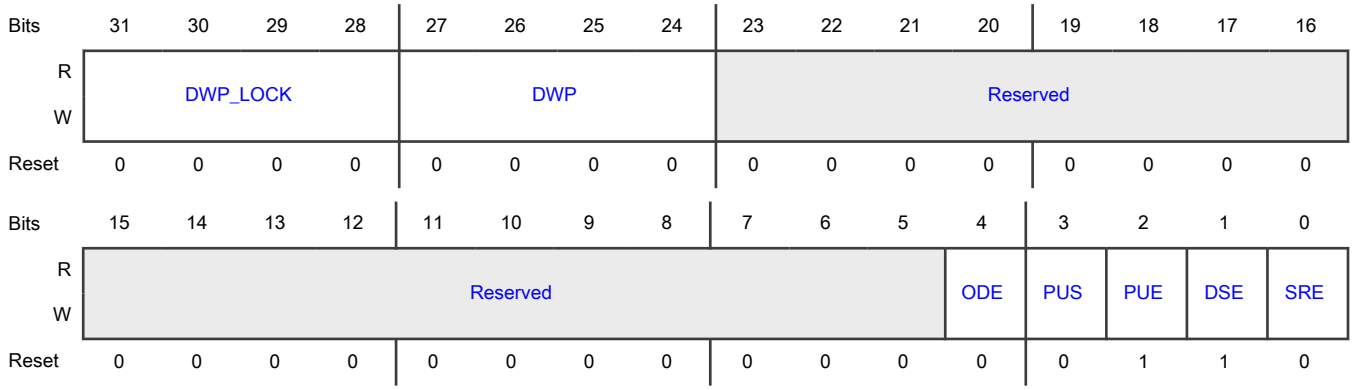
17.4.2.44 SW_PAD_CTL_PAD_GPIO_AON_13 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_13)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_13	A8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_13 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_13 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_13 0b - Pull Disable, Highz 1b - Pull Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_13 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_13 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.45 SW_PAD_CTL_PAD_GPIO_AON_14 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_14)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_14	ACh

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0

Fields

Field	Function
31-28	Domain write protection lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
DWP_LOCK	Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_14 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_14 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_14 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_14 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_14 0b - Fast Slew Rate 1b - Slow Slew Rate

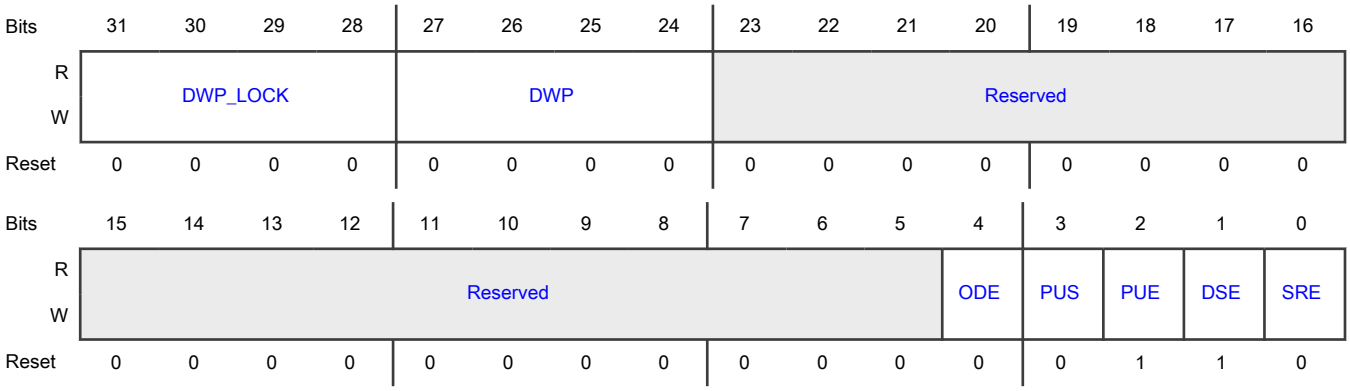
17.4.2.46 SW_PAD_CTL_PAD_GPIO_AON_15 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_15)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_15	B0h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_15 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_15 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_15 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_15 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_15 0b - Fast Slew Rate 1b - Slow Slew Rate

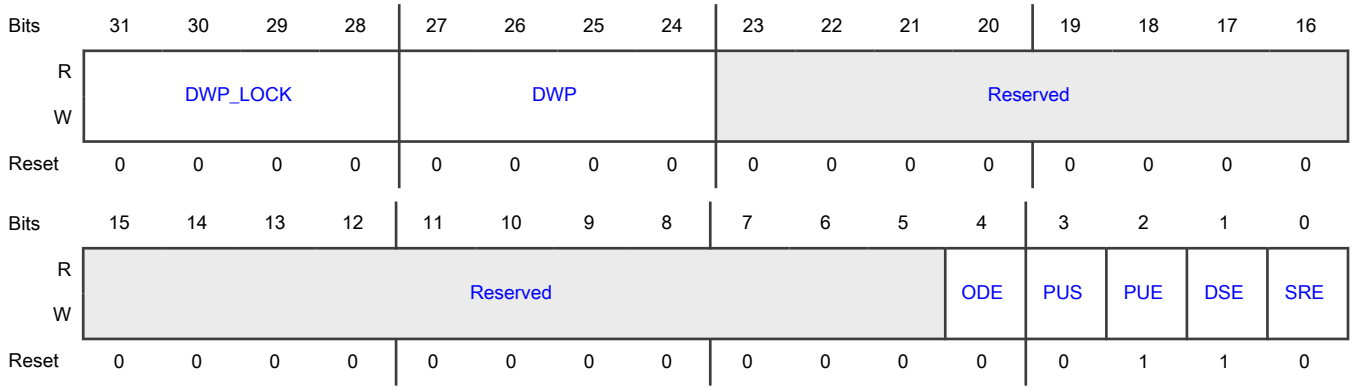
17.4.2.47 SW_PAD_CTL_PAD_GPIO_AON_16 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_16)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_16	B4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_16 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_16 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_16 0b - Pull Disable, Highz 1b - Pull Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_16 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_16 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.48 SW_PAD_CTL_PAD_GPIO_AON_17 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_17)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_17	B8h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28	Domain write protection lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
DWP_LOCK	Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_17 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_17 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_17 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_17 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_17 0b - Fast Slew Rate 1b - Slow Slew Rate

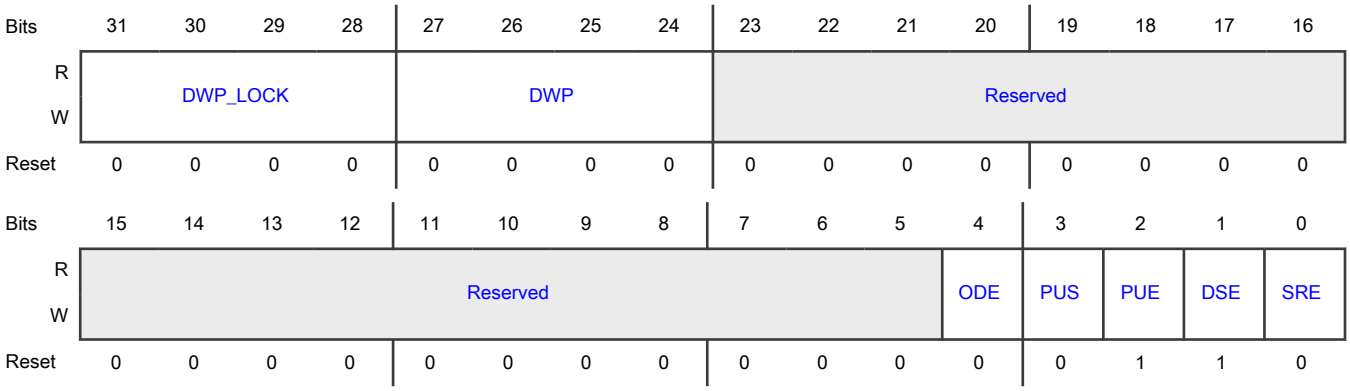
17.4.2.49 SW_PAD_CTL_PAD_GPIO_AON_18 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_18)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_18	BCh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_18 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_18 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_18 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_18 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_18 0b - Fast Slew Rate 1b - Slow Slew Rate

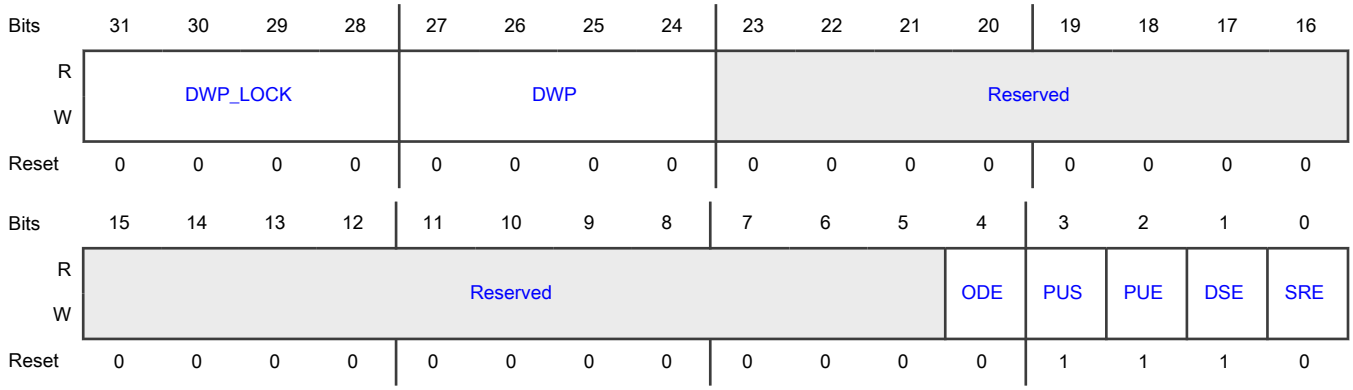
17.4.2.50 SW_PAD_CTL_PAD_GPIO_AON_19 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_19)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_19	C0h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_19 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_19 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_19 0b - Pull Disable, Highz 1b - Pull Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_19 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_19 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.51 SW_PAD_CTL_PAD_GPIO_AON_20 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_20)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_20	C4h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0

Fields

Field	Function
31-28	Domain write protection lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
DWP_LOCK	Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_20 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_20 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_20 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_20 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_20 0b - Fast Slew Rate 1b - Slow Slew Rate

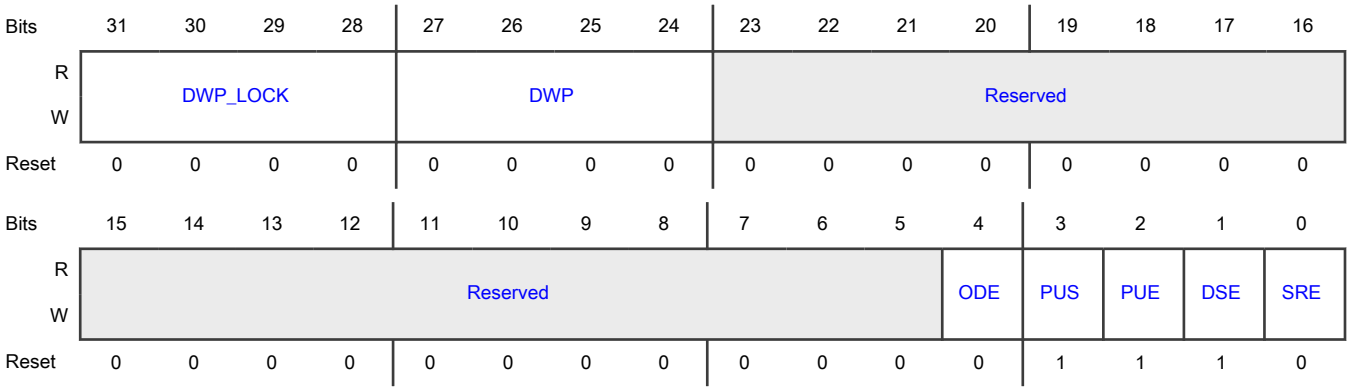
17.4.2.52 SW_PAD_CTL_PAD_GPIO_AON_21 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_21)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_21	C8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_21 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_21 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_21 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_21 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_21 0b - Fast Slew Rate 1b - Slow Slew Rate

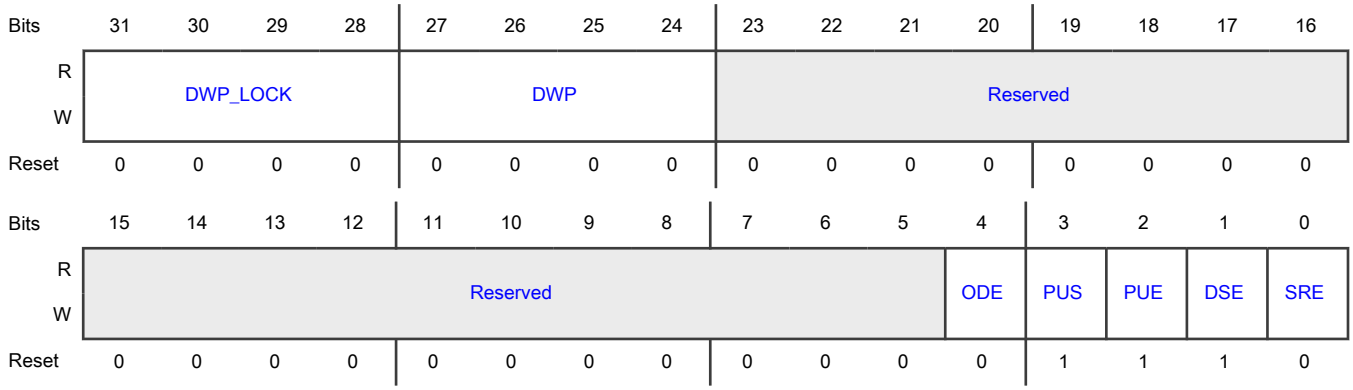
17.4.2.53 SW_PAD_CTL_PAD_GPIO_AON_22 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_22)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_22	CCh

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_22 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_22 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_22 0b - Pull Disable, Highz 1b - Pull Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_22 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_22 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.54 SW_PAD_CTL_PAD_GPIO_AON_23 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_23)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_23	D0h

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28	Domain write protection lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
DWP_LOCK	Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_23 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_23 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_23 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_23 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_23 0b - Fast Slew Rate 1b - Slow Slew Rate

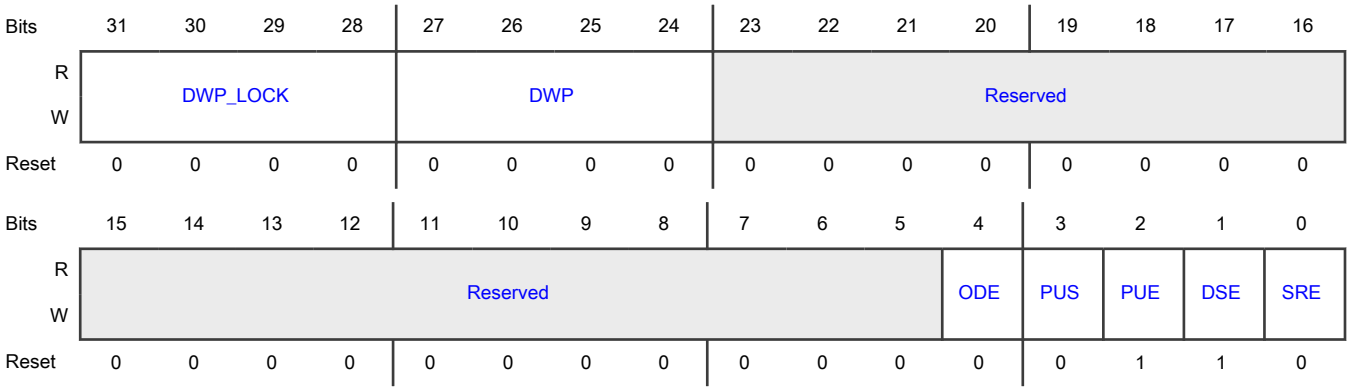
17.4.2.55 SW_PAD_CTL_PAD_GPIO_AON_24 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_24)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_24	D4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_24 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_24 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_24 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_24 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_24 0b - Fast Slew Rate 1b - Slow Slew Rate

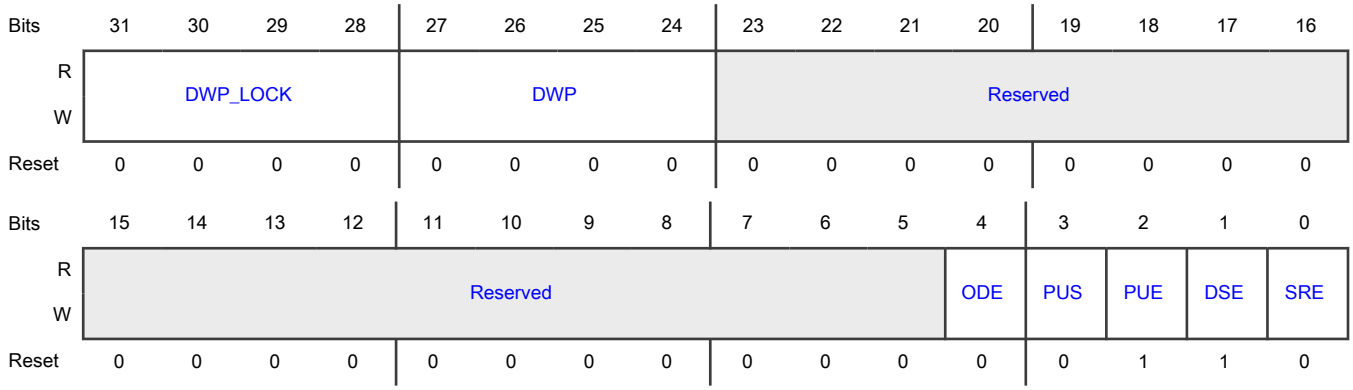
17.4.2.56 SW_PAD_CTL_PAD_GPIO_AON_25 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_25)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_25	D8h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_25 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_25 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_25 0b - Pull Disable, Highz 1b - Pull Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_25 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_25 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.57 SW_PAD_CTL_PAD_GPIO_AON_26 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_26)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_26	DCh

Function
SW_PAD_CTL Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DWP_LOCK				DWP				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ODE	PUS	PUE	DSE	SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-28	Domain write protection lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
DWP_LOCK	Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_26 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_26 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_26 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_26 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_26 0b - Fast Slew Rate 1b - Slow Slew Rate

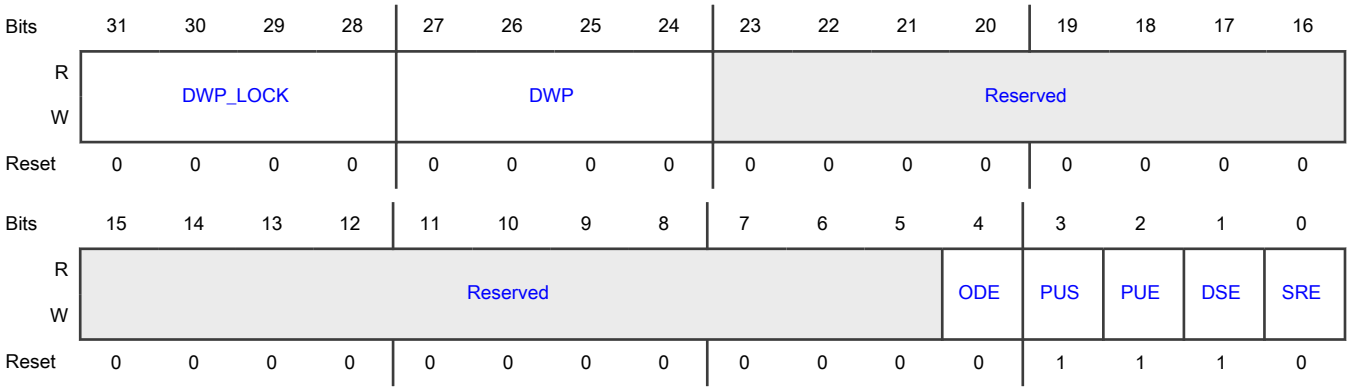
17.4.2.58 SW_PAD_CTL_PAD_GPIO_AON_27 SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_27)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_27	E0h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_27 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_27 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_27 0b - Pull Disable, Highz 1b - Pull Enable
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_27 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_27 0b - Fast Slew Rate 1b - Slow Slew Rate

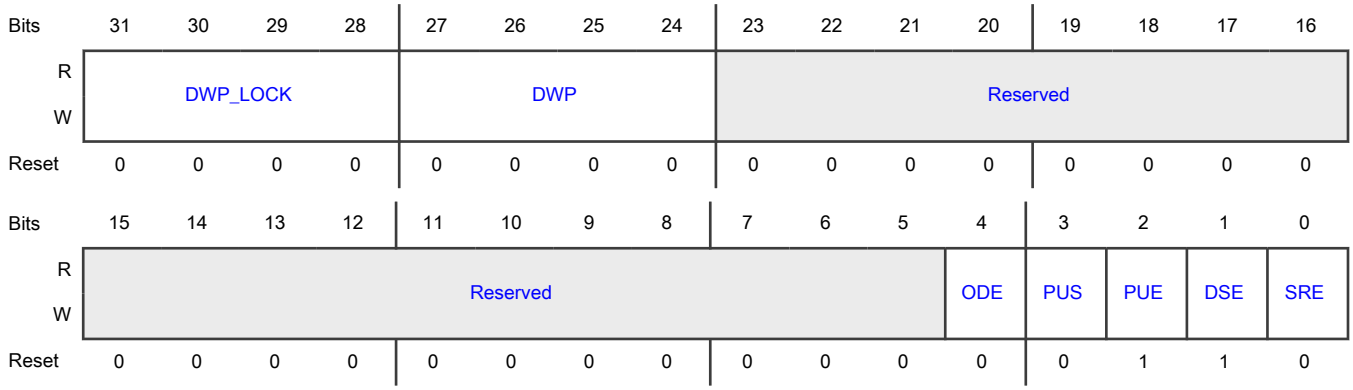
17.4.2.59 SW_PAD_CTL_PAD_GPIO_AON_28_DUMMY SW PAD Control Register (SW_PAD_CTL_PAD_GPIO_AON_28_DUMMY)

Offset

Register	Offset
SW_PAD_CTL_PAD_GPIO_AON_28_DUMMY	E4h

Function
SW_PAD_CTL Register

Diagram



Fields

Field	Function
31-28 DWP_LOCK	Domain write protection lock Once DWP_LOCK bit i is set the write to DWP bit i will be blocked.
27-24 DWP	Domain write protection These 4 bits determine which core is forbidden to change bits 6:0 in this register. When bit 24 is set, access from domain whose domain ID is DID0 is forbidden. When bit 25 is set, access from domain whose domain ID is DID1 is forbidden. When bit 26 is set, access from domain whose domain ID is DID2 is forbidden. When bit 27 is set, access from domain whose domain ID is DID3 is forbidden. DID0/DID1/DID2/DID3 is set from the AON Domain Secure Block Control (BLK_CTRL_S_AON) IOMUXC_DOMAIN_CFG register.
23-5 —	- Reserved
4 ODE	Open Drain Field Select one out of next values for pad: GPIO_AON_28_DUMMY 0b - Disabled 1b - Enabled
3 PUS	Pull Up / Down Config. Field Select one out of next values for pad: GPIO_AON_28_DUMMY 0b - Weak pull down 1b - Weak pull up
2 PUE	Pull / Keep Select Field Select one out of next values for pad: GPIO_AON_28_DUMMY 0b - Pull Disable, Highz 1b - Pull Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DSE	Drive Strength Field Select one out of next values for pad: GPIO_AON_28_DUMMY 0b - normal driver 1b - high driver
0 SRE	Slew Rate Field Select one out of next values for pad: GPIO_AON_28_DUMMY 0b - Fast Slew Rate 1b - Slow Slew Rate

17.4.2.60 I3C1_PIN_SCL_IN_SELECT_INPUT DAISY Register (I3C1_PIN_SCL_IN_SELECT_INPUT)

Offset

Register	Offset
I3C1_PIN_SCL_IN_SELECT_INPUT	E8h

Function
DAISY Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															DAISY
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: i3c1, In Pin: pin_SCL_in 0b - Selecting Pad: GPIO_AON_16 for Mode: ALT9 1b - Selecting Pad: GPIO_AON_21 for Mode: ALT3

17.4.2.61 I3C1_PIN_SDA_IN_SELECT_INPUT DAISY Register (I3C1_PIN_SDA_IN_SELECT_INPUT)

Offset

Register	Offset
I3C1_PIN_SDA_IN_SELECT_INPUT	ECh

Function
DAISY Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															DAISY
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: i3c1, In Pin: pin_SDA_in 0b - Selecting Pad: GPIO_AON_15 for Mode: ALT9 1b - Selecting Pad: GPIO_AON_20 for Mode: ALT3

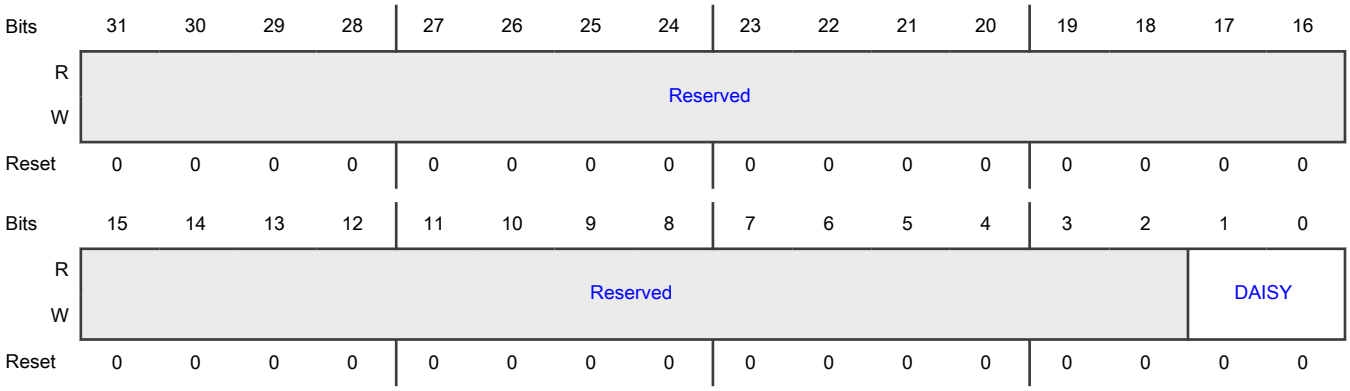
17.4.2.62 LPI2C1_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C1_IPP_IND_LPI2C_SCL_SELECT_INPUT)

Offset

Register	Offset
LPI2C1_IPP_IND_LPI2C_SCL_SELECT_INPUT	F0h

Function
DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c1, In Pin: ipp_ind_lpi2c_scl 00b - Selecting Pad: GPIO_AON_07 for Mode: ALT3 01b - Selecting Pad: GPIO_AON_09 for Mode: ALT6 10b - Selecting Pad: GPIO_AON_21 for Mode: ALT2 11b - Selecting Pad: GPIO_AON_25 for Mode: ALT1

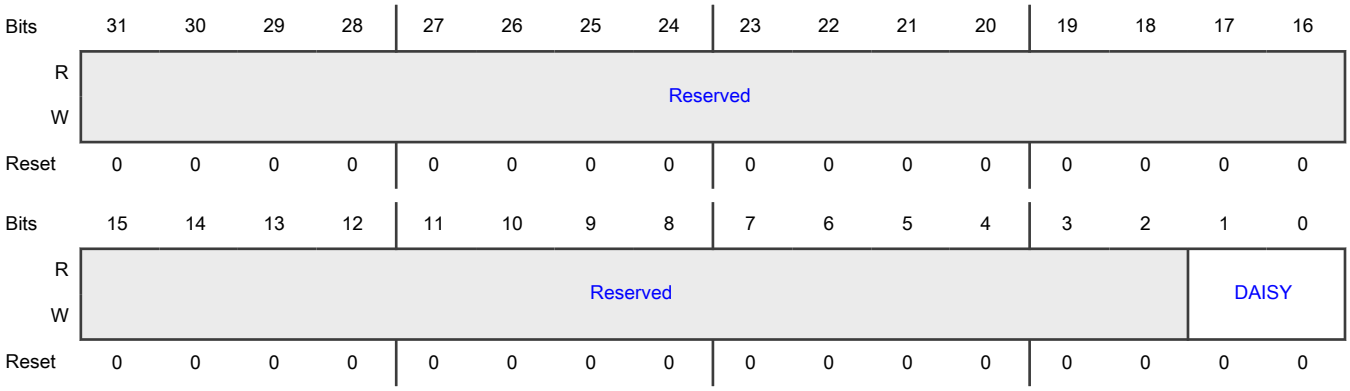
17.4.2.63 LPI2C1_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C1_IPP_IND_LPI2C_SDA_SELECT_INPUT)

Offset

Register	Offset
LPI2C1_IPP_IND_LPI2C_SDA_SELECT_INPUT	F4h

Function
DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c1, In Pin: ipp_ind_lpi2c_sda 00b - Selecting Pad: GPIO_AON_06 for Mode: ALT3 01b - Selecting Pad: GPIO_AON_08 for Mode: ALT6 10b - Selecting Pad: GPIO_AON_20 for Mode: ALT2 11b - Selecting Pad: GPIO_AON_24 for Mode: ALT1

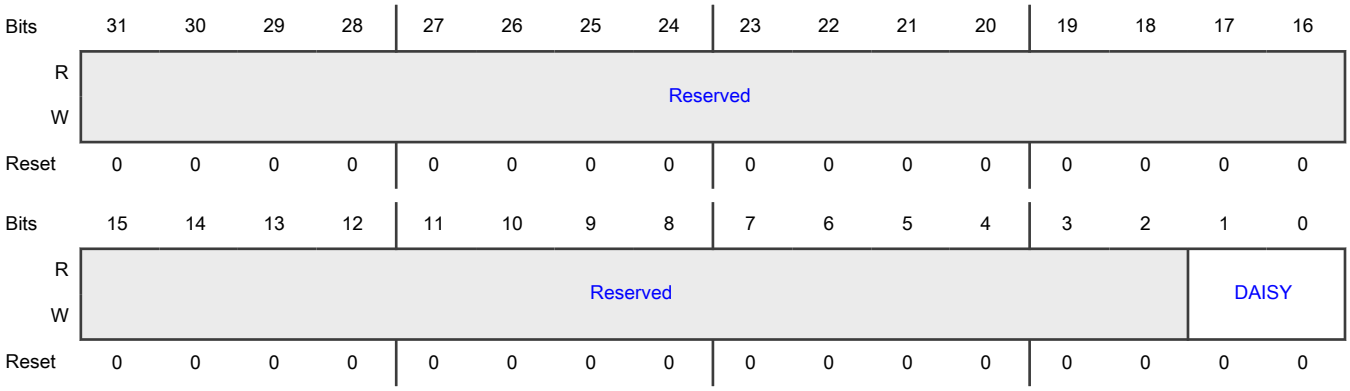
17.4.2.64 LPI2C2_IPP_IND_LPI2C_SCL_SELECT_INPUT DAISY Register (LPI2C2_IPP_IND_LPI2C_SCL_SELECT_INPUT)

Offset

Register	Offset
LPI2C2_IPP_IND_LPI2C_SCL_SELECT_INPUT	F8h

Function
DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c2, In Pin: ipp_ind_lpi2c_scl 00b - Selecting Pad: GPIO_AON_16 for Mode: ALT4 01b - Selecting Pad: GPIO_AON_18 for Mode: ALT3 10b - Selecting Pad: GPIO_AON_23 for Mode: ALT1

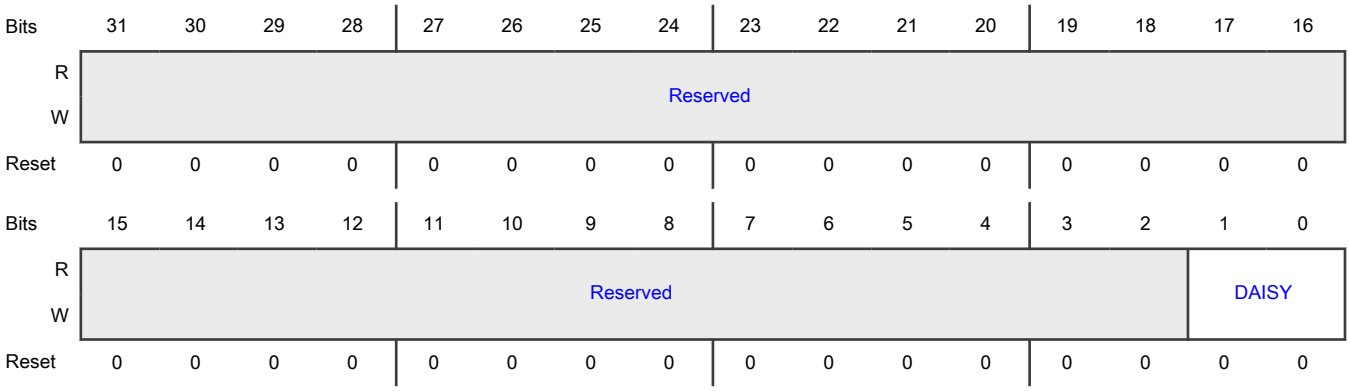
17.4.2.65 LPI2C2_IPP_IND_LPI2C_SDA_SELECT_INPUT DAISY Register (LPI2C2_IPP_IND_LPI2C_SDA_SELECT_INPUT)

Offset

Register	Offset
LPI2C2_IPP_IND_LPI2C_SDA_SELECT_INPUT	FCh

Function
DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpi2c2, In Pin: ipp_ind_lpi2c_sda 00b - Selecting Pad: GPIO_AON_15 for Mode: ALT4 01b - Selecting Pad: GPIO_AON_17 for Mode: ALT3 10b - Selecting Pad: GPIO_AON_22 for Mode: ALT1

17.4.2.66 LPSPI1_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI1_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)

Offset

Register	Offset
LPSPI1_IPP_IND_LPSPI_PCS_SELECT_INPUT_0	100h

Function
DAISY Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															DAISY
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi1, In Pin: ipp_ind_lpspi_pcs0 0b - Selecting Pad: GPIO_AON_05 for Mode: ALT0 1b - Selecting Pad: GPIO_AON_13 for Mode: ALT8

17.4.2.67 LPSPI1_IPP_IND_LPSPI_PCS_SELECT_INPUT_1 DAISY Register (LPSPI1_IPP_IND_LPSPI_PCS_SELECT_INPUT_1)

Offset

Register	Offset
LPSPI1_IPP_IND_LPSPI_PCS_SELECT_INPUT_1	104h

Function
DAISY Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															DAISY
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi1, In Pin: ipp_ind_lpspi_pcs1 0b - Selecting Pad: GPIO_AON_03 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_08 for Mode: ALT8

17.4.2.68 LPSPI1_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI1_IPP_IND_LPSPI_SCK_SELECT_INPUT)

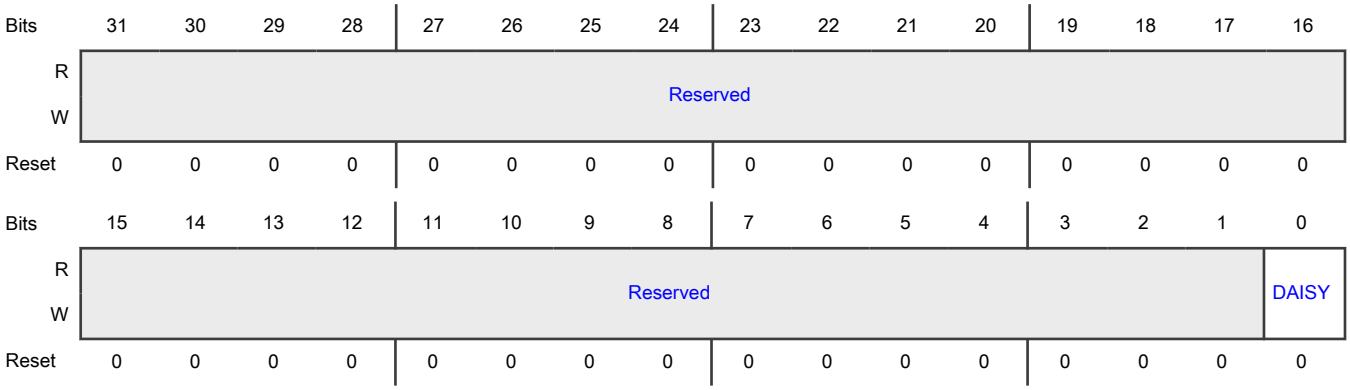
Offset

Register	Offset
LPSPI1_IPP_IND_LPSPI_SCK_SELECT_INPUT	108h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi1, In Pin: ipp_ind_lpspi_sck 0b - Selecting Pad: GPIO_AON_04 for Mode: ALT0 1b - Selecting Pad: GPIO_AON_12 for Mode: ALT8

17.4.2.69 LPSPI1_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI1_IPP_IND_LPSPI_SDI_SELECT_INPUT)

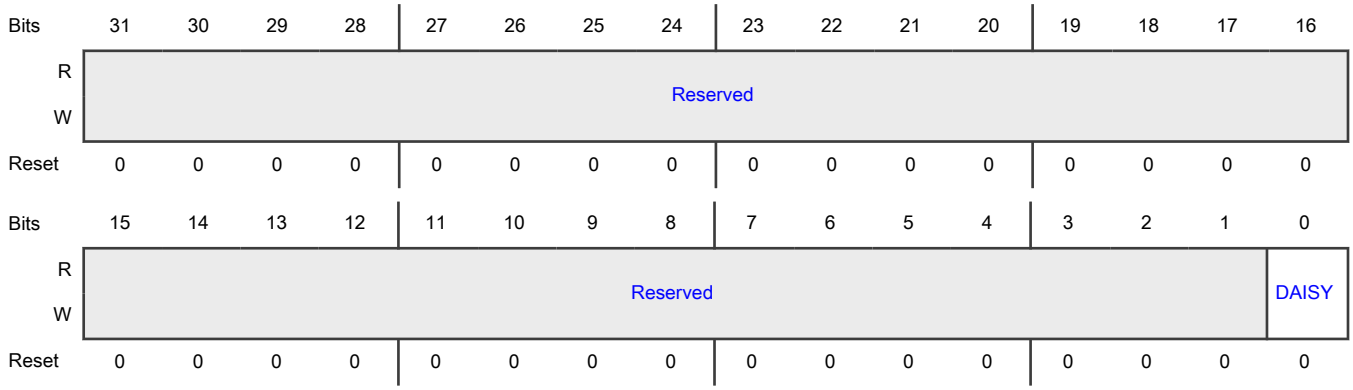
Offset

Register	Offset
LPSPI1_IPP_IND_LPSPI_SDI_SELECT_INPUT	10Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi1, In Pin: ipp_ind_lpspi_sdi 0b - Selecting Pad: GPIO_AON_07 for Mode: ALT0 1b - Selecting Pad: GPIO_AON_15 for Mode: ALT8

17.4.2.70 LPSPI1_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI1_IPP_IND_LPSPI_SDO_SELECT_INPUT)

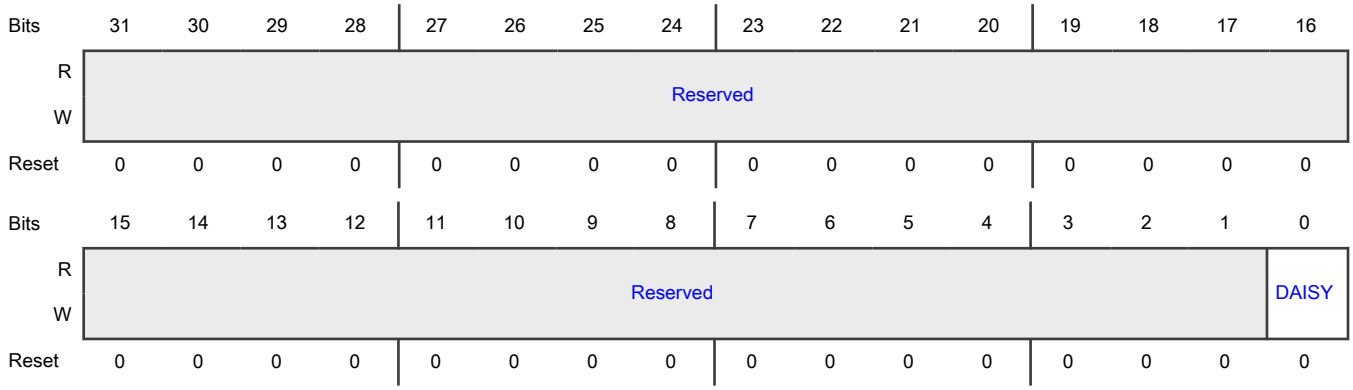
Offset

Register	Offset
LPSPI1_IPP_IND_LPSPI_SDO_SELECT_INPUT	110h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi1, In Pin: ipp_ind_lpspi_sdo 0b - Selecting Pad: GPIO_AON_06 for Mode: ALT0 1b - Selecting Pad: GPIO_AON_14 for Mode: ALT8

17.4.2.71 LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_0 DAISY Register (LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_0)

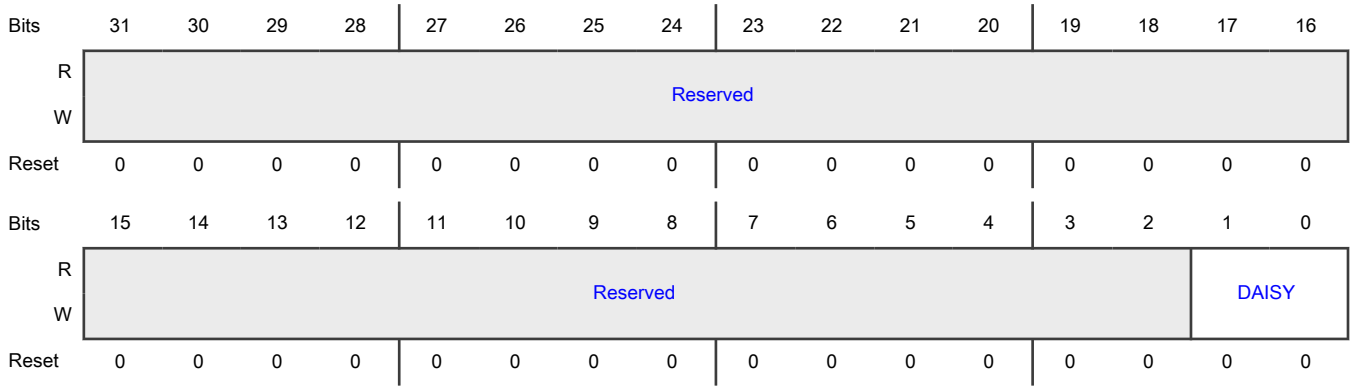
Offset

Register	Offset
LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_0	114h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi2, In Pin: ipp_ind_lpspi_pcs0 00b - Selecting Pad: GPIO_AON_10 for Mode: ALT1 01b - Selecting Pad: GPIO_AON_16 for Mode: ALT1 10b - Selecting Pad: GPIO_AON_25 for Mode: ALT6

17.4.2.72 LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_1 DAISY Register (LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_1)

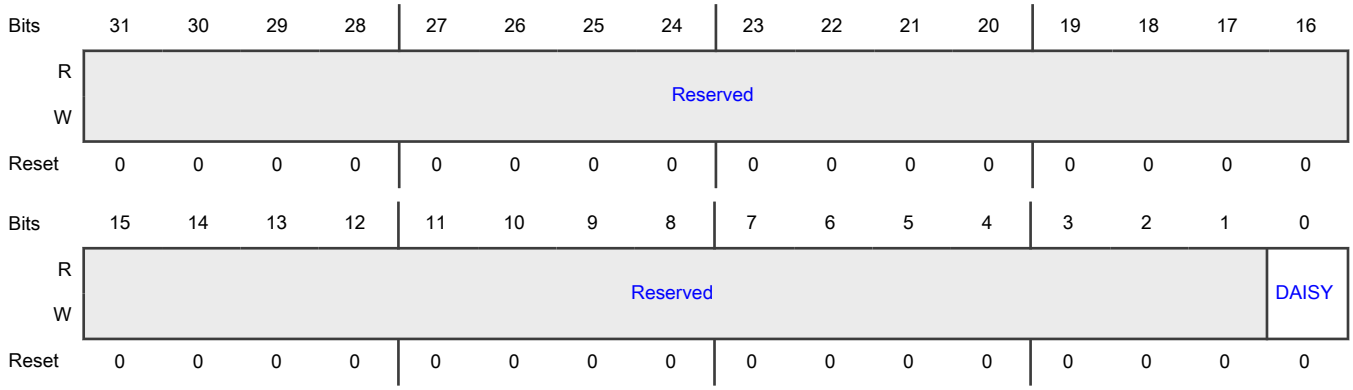
Offset

Register	Offset
LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_1	118h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi2, In Pin: ipp_ind_lpspi_pcs1 0b - Selecting Pad: GPIO_AON_15 for Mode: ALT1 1b - Selecting Pad: GPIO_AON_21 for Mode: ALT1

17.4.2.73 LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_3 DAISY Register (LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_3)

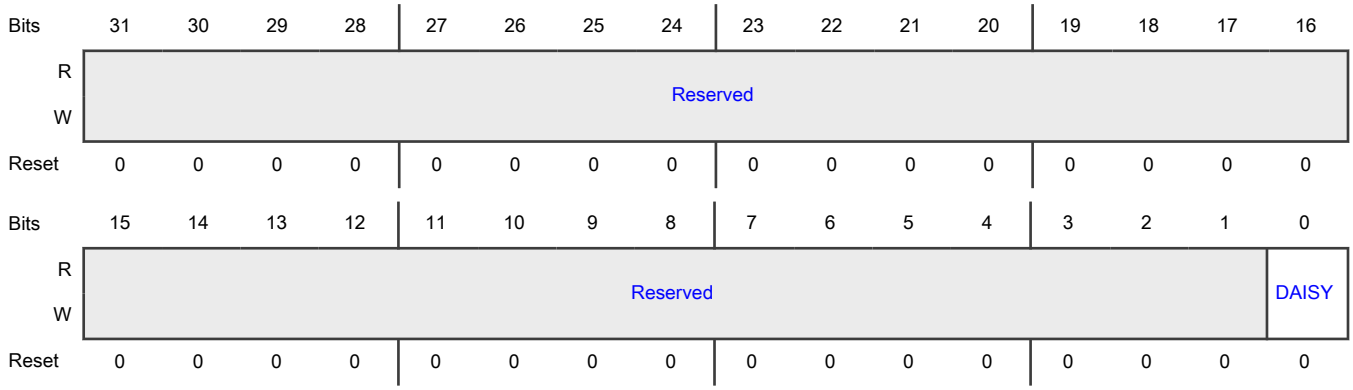
Offset

Register	Offset
LPSPI2_IPP_IND_LPSPI_PCS_SELECT_INPUT_3	11Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi2, In Pin: ipp_ind_lpspi_pcs3 0b - Selecting Pad: GPIO_AON_02 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_27 for Mode: ALT1

17.4.2.74 LPSPI2_IPP_IND_LPSPI_SCK_SELECT_INPUT DAISY Register (LPSPI2_IPP_IND_LPSPI_SCK_SELECT_INPUT)

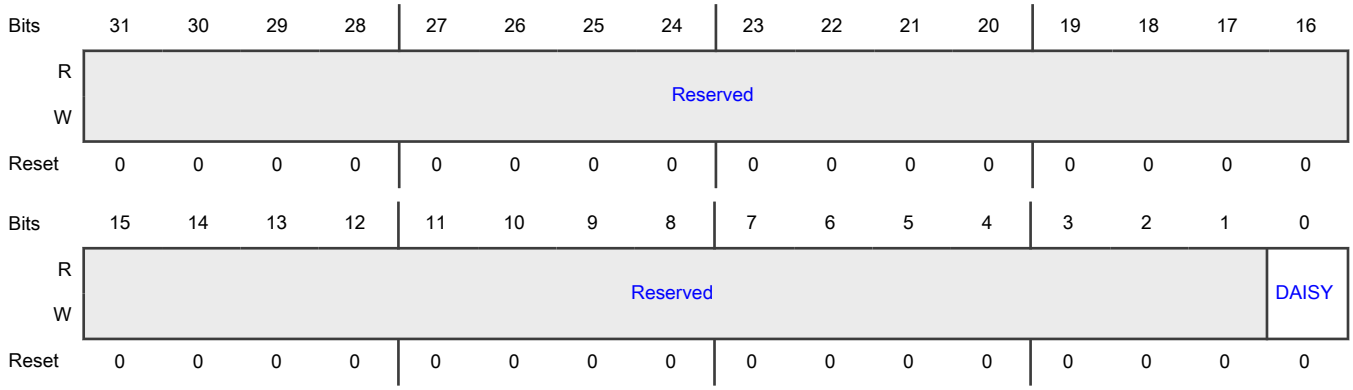
Offset

Register	Offset
LPSPI2_IPP_IND_LPSPI_SCK_SELECT_INPUT	120h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi2, In Pin: ipp_ind_lpspi_sck 0b - Selecting Pad: GPIO_AON_19 for Mode: ALT1 1b - Selecting Pad: GPIO_AON_22 for Mode: ALT6

17.4.2.75 LPSPI2_IPP_IND_LPSPI_SDI_SELECT_INPUT DAISY Register (LPSPI2_IPP_IND_LPSPI_SDI_SELECT_INPUT)

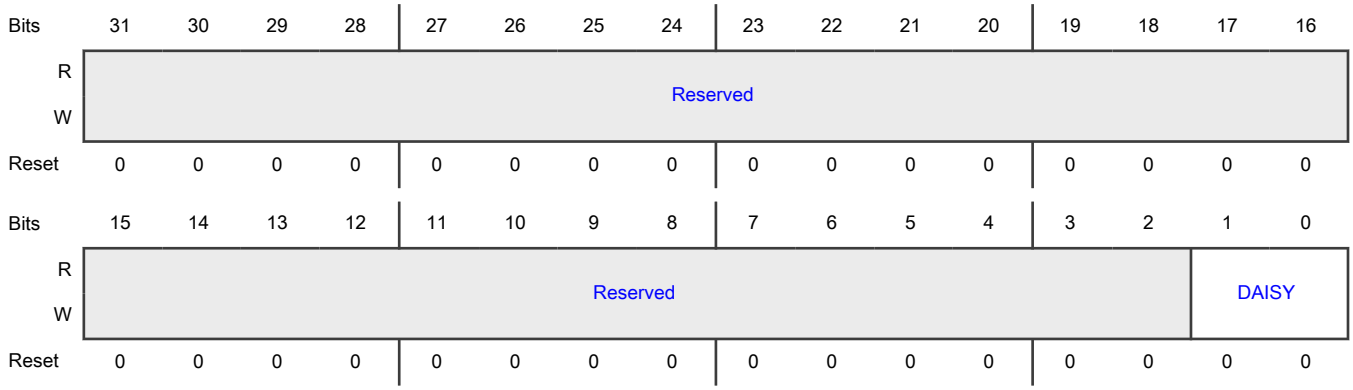
Offset

Register	Offset
LPSPI2_IPP_IND_LPSPI_SDI_SELECT_INPUT	124h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi2, In Pin: ipp_ind_lpspi_sdi 00b - Selecting Pad: GPIO_AON_03 for Mode: ALT3 01b - Selecting Pad: GPIO_AON_17 for Mode: ALT1 10b - Selecting Pad: GPIO_AON_24 for Mode: ALT6

17.4.2.76 LPSPI2_IPP_IND_LPSPI_SDO_SELECT_INPUT DAISY Register (LPSPI2_IPP_IND_LPSPI_SDO_SELECT_INPUT)

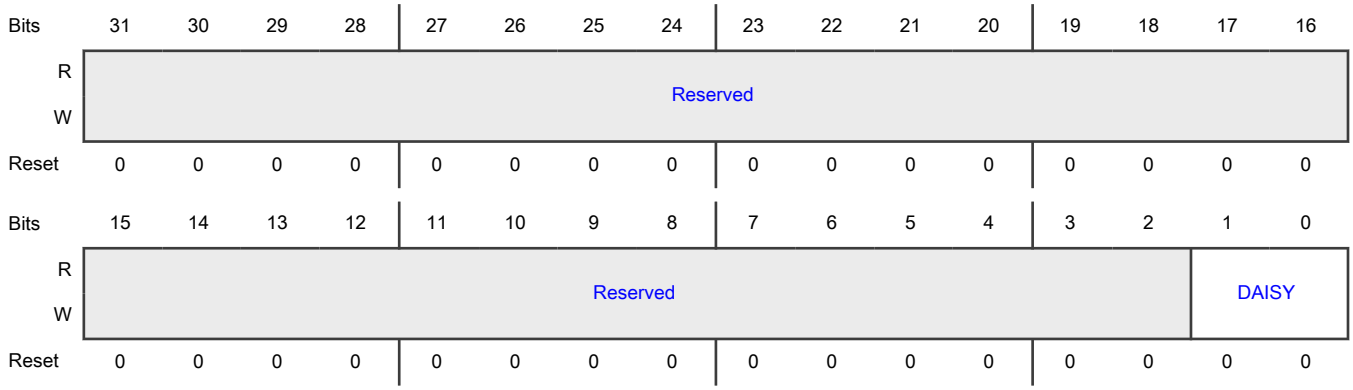
Offset

Register	Offset
LPSPI2_IPP_IND_LPSPI_SDO_SELECT_INPUT	128h

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpspi2, In Pin: ipp_ind_lpspi_sdo 00b - Selecting Pad: GPIO_AON_02 for Mode: ALT3 01b - Selecting Pad: GPIO_AON_18 for Mode: ALT1 10b - Selecting Pad: GPIO_AON_23 for Mode: ALT6

17.4.2.77 LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_1 DAISY Register (LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_1)

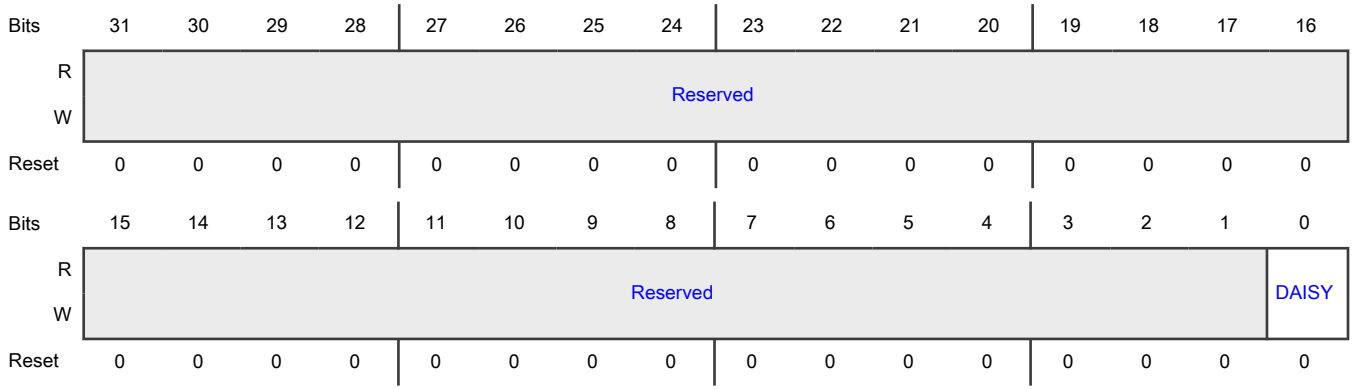
Offset

Register	Offset
LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_1	12Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lptmr1, In Pin: ipp_ind_lptimer1 0b - Selecting Pad: GPIO_AON_00 for Mode: ALT4 1b - Selecting Pad: GPIO_AON_13 for Mode: ALT6

17.4.2.78 LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_2 DAISY Register (LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_2)

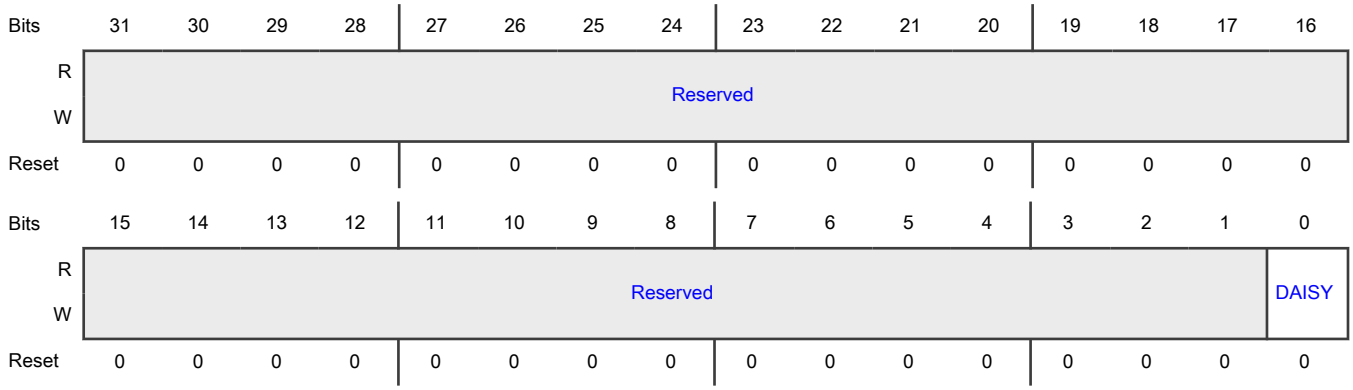
Offset

Register	Offset
LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_2	130h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lptmr1, In Pin: ipp_ind_lptimer2 0b - Selecting Pad: GPIO_AON_01 for Mode: ALT4 1b - Selecting Pad: GPIO_AON_14 for Mode: ALT6

17.4.2.79 LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_3 DAISY Register (LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_3)

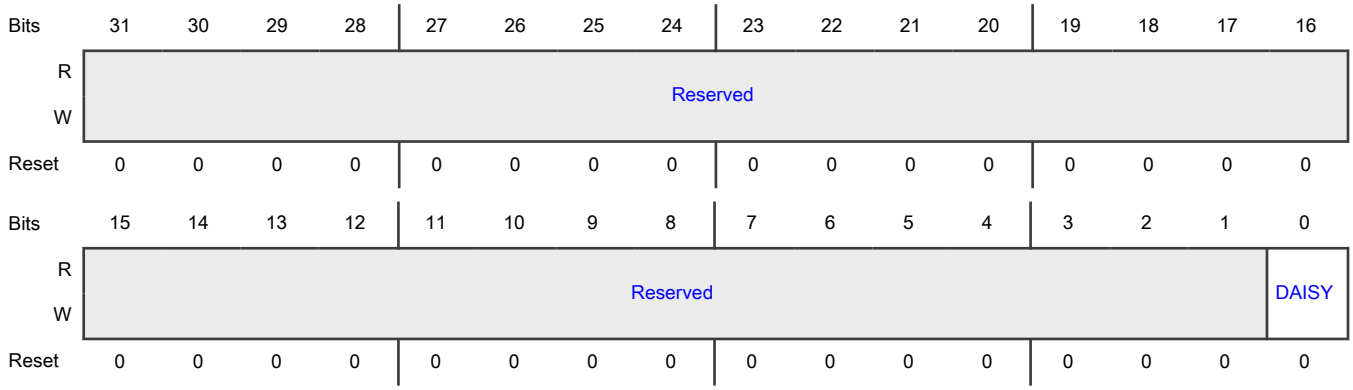
Offset

Register	Offset
LPTMR1_IPP_IND_LPTIMER_SELECT_INPUT_3	134h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lptmr1, In Pin: ipp_ind_lptimer3 0b - Selecting Pad: GPIO_AON_02 for Mode: ALT4 1b - Selecting Pad: GPIO_AON_15 for Mode: ALT6

17.4.2.80 LPUART1_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART1_IPP_IND_LPUART_CTS_N_SELECT_INPUT)

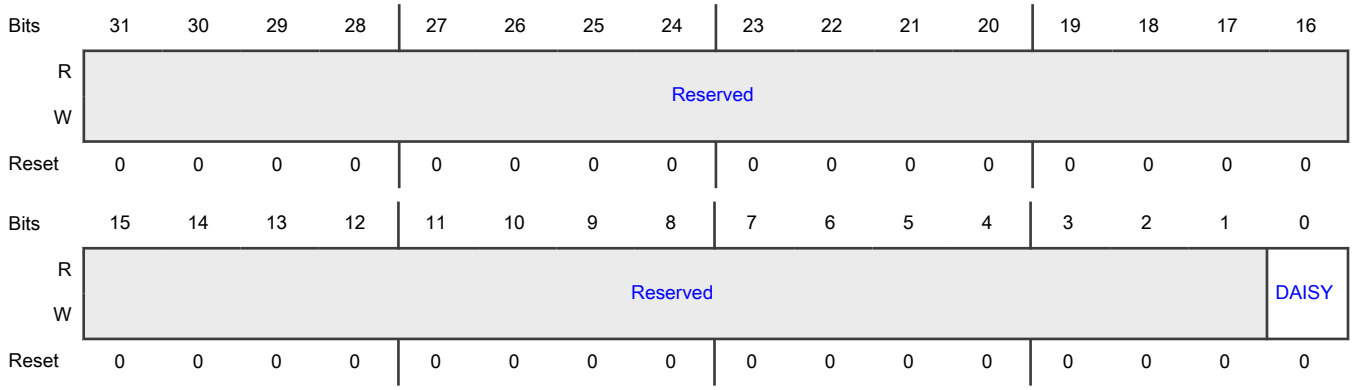
Offset

Register	Offset
LPUART1_IPP_IND_LPUART_CTS_N_SELECT_INPUT	138h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart1, In Pin: ipp_ind_lpuart_cts_n 0b - Selecting Pad: GPIO_AON_11 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_19 for Mode: ALT4

17.4.2.81 LPUART1_IPP_IND_LPUART_DCD_N_SELECT_INPUT DAISY Register (LPUART1_IPP_IND_LPUART_DCD_N_SELECT_INPUT)

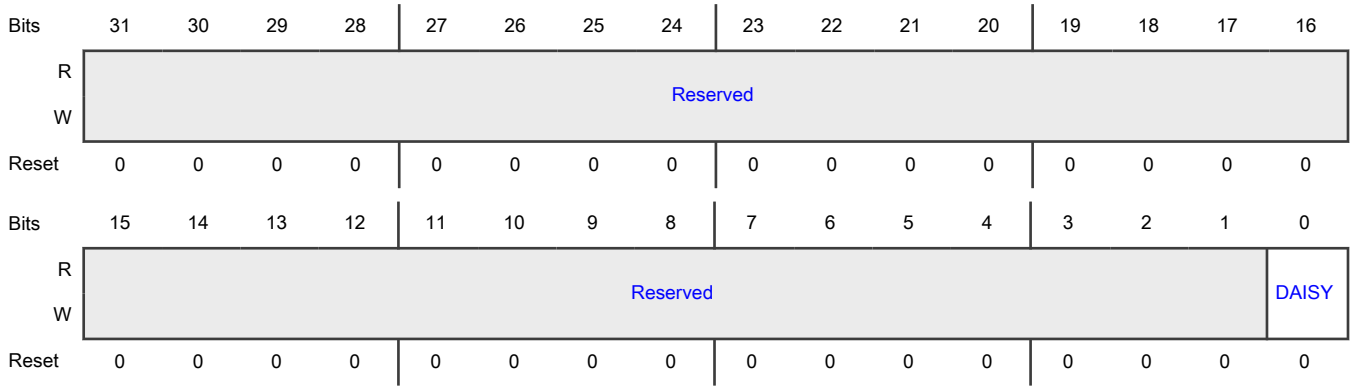
Offset

Register	Offset
LPUART1_IPP_IND_LPUART_DCD_N_SELECT_INPUT	13Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart1, In Pin: ipp_ind_lpuart_dcd_n 0b - Selecting Pad: GPIO_AON_14 for Mode: ALT3 1b - Selecting Pad: GPIO_AON_18 for Mode: ALT4

17.4.2.82 LPUART1_IPP_IND_LPUART_DSR_N_SELECT_INPUT DAISY Register (LPUART1_IPP_IND_LPUART_DSR_N_SELECT_INPUT)

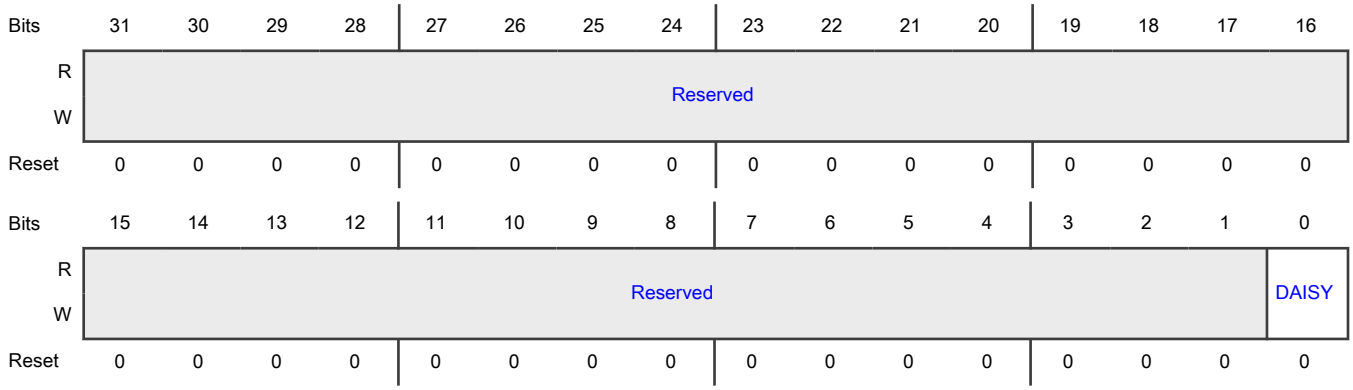
Offset

Register	Offset
LPUART1_IPP_IND_LPUART_DSR_N_SELECT_INPUT	140h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart1, In Pin: ipp_ind_lpuart_dsr_n 0b - Selecting Pad: GPIO_AON_13 for Mode: ALT3 1b - Selecting Pad: GPIO_AON_17 for Mode: ALT4

17.4.2.83 LPUART12_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART12_IPP_IND_LPUART_CTS_N_SELECT_INPUT)

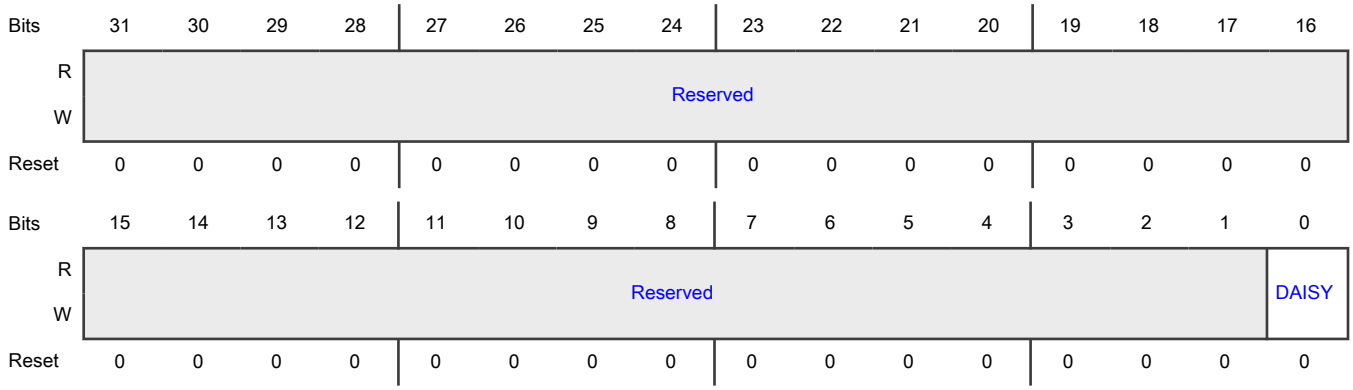
Offset

Register	Offset
LPUART12_IPP_IND_LPUART_CTS_N_SELECT_INPUT	144h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart12, In Pin: ipp_ind_lpuart_cts_n 0b - Selecting Pad: GPIO_AON_13 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_22 for Mode: ALT3

17.4.2.84 LPUART12_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART12_IPP_IND_LPUART_RXD_SELECT_INPUT)

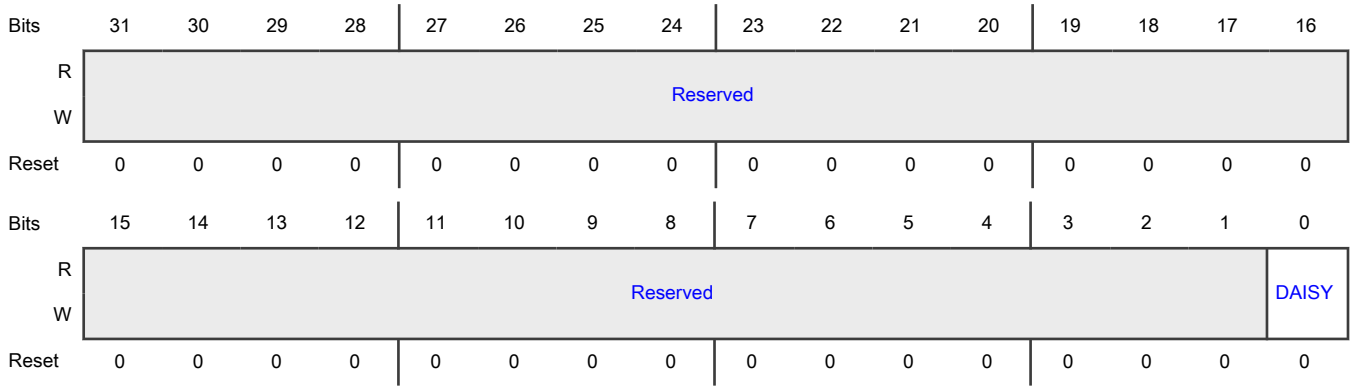
Offset

Register	Offset
LPUART12_IPP_IND_LPUART_RXD_SELECT_INPUT	148h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart12, In Pin: ipp_ind_lpuart_rxd 0b - Selecting Pad: GPIO_AON_16 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_20 for Mode: ALT9

17.4.2.85 LPUART12_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART12_IPP_IND_LPUART_TXD_SELECT_INPUT)

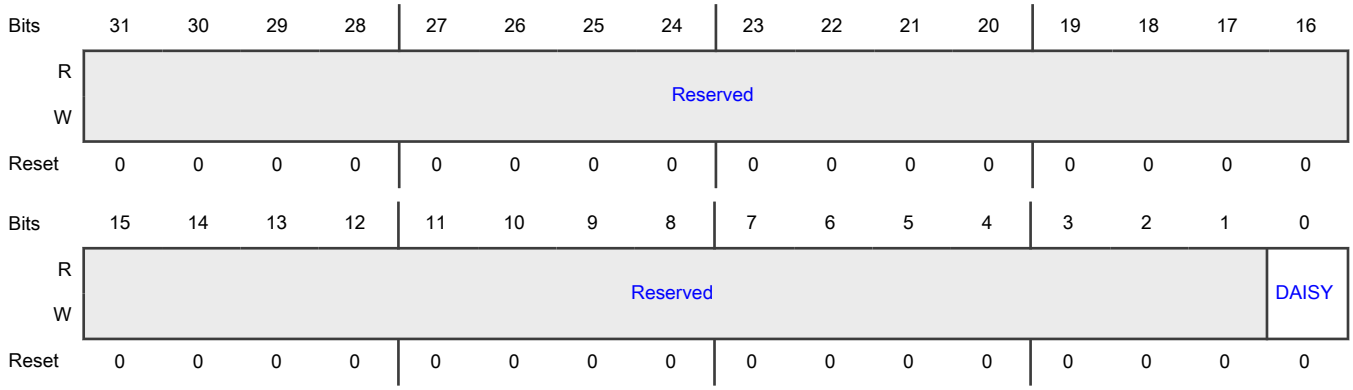
Offset

Register	Offset
LPUART12_IPP_IND_LPUART_TXD_SELECT_INPUT	14Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart12, In Pin: ipp_ind_lpuart_txd 0b - Selecting Pad: GPIO_AON_15 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_19 for Mode: ALT9

17.4.2.86 LPUART2_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART2_IPP_IND_LPUART_CTS_N_SELECT_INPUT)

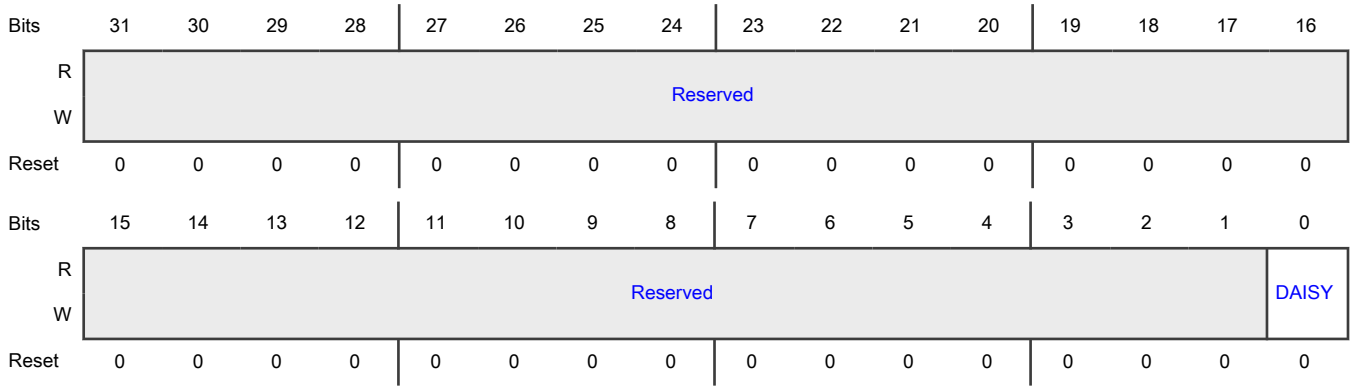
Offset

Register	Offset
LPUART2_IPP_IND_LPUART_CTS_N_SELECT_INPUT	150h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart2, In Pin: ipp_ind_lpuart_cts_n 0b - Selecting Pad: GPIO_AON_03 for Mode: ALT6 1b - Selecting Pad: GPIO_AON_25 for Mode: ALT2

17.4.2.87 LPUART2_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART2_IPP_IND_LPUART_RXD_SELECT_INPUT)

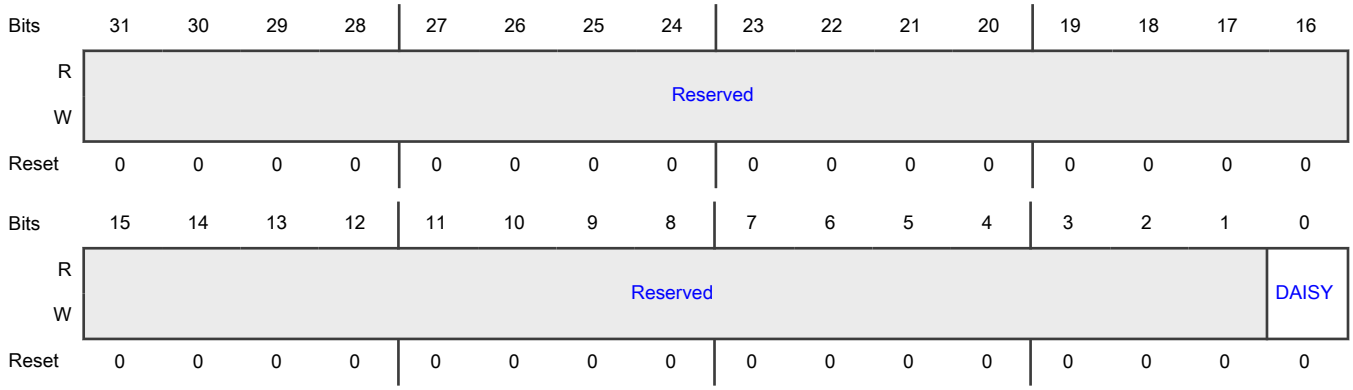
Offset

Register	Offset
LPUART2_IPP_IND_LPUART_RXD_SELECT_INPUT	154h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart2, In Pin: ipp_ind_lpuart_rxd 0b - Selecting Pad: GPIO_AON_01 for Mode: ALT6 1b - Selecting Pad: GPIO_AON_27 for Mode: ALT2

17.4.2.88 LPUART2_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART2_IPP_IND_LPUART_TXD_SELECT_INPUT)

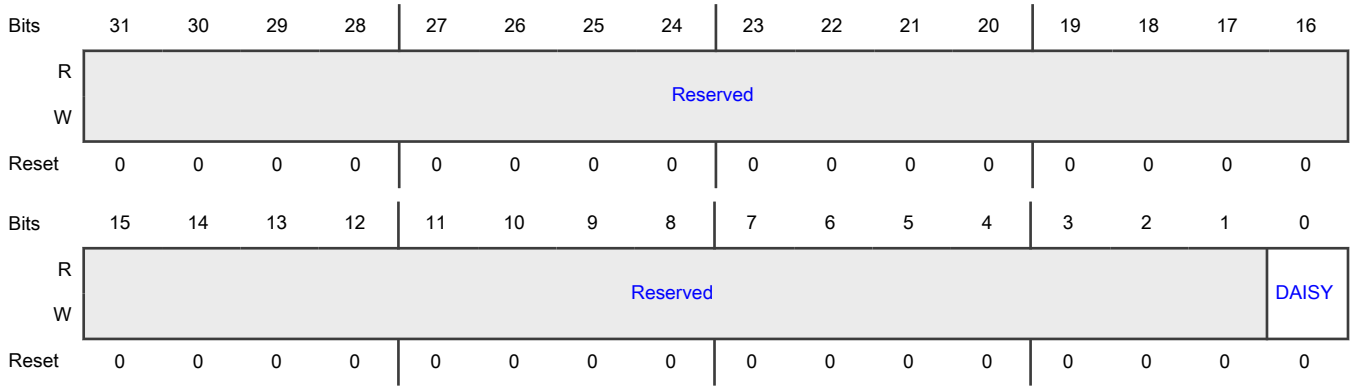
Offset

Register	Offset
LPUART2_IPP_IND_LPUART_TXD_SELECT_INPUT	158h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart2, In Pin: ipp_ind_lpuart_txd 0b - Selecting Pad: GPIO_AON_00 for Mode: ALT6 1b - Selecting Pad: GPIO_AON_26 for Mode: ALT2

17.4.2.89 LPUART7_IPP_IND_LPUART_CTS_N_SELECT_INPUT DAISY Register (LPUART7_IPP_IND_LPUART_CTS_N_SELECT_INPUT)

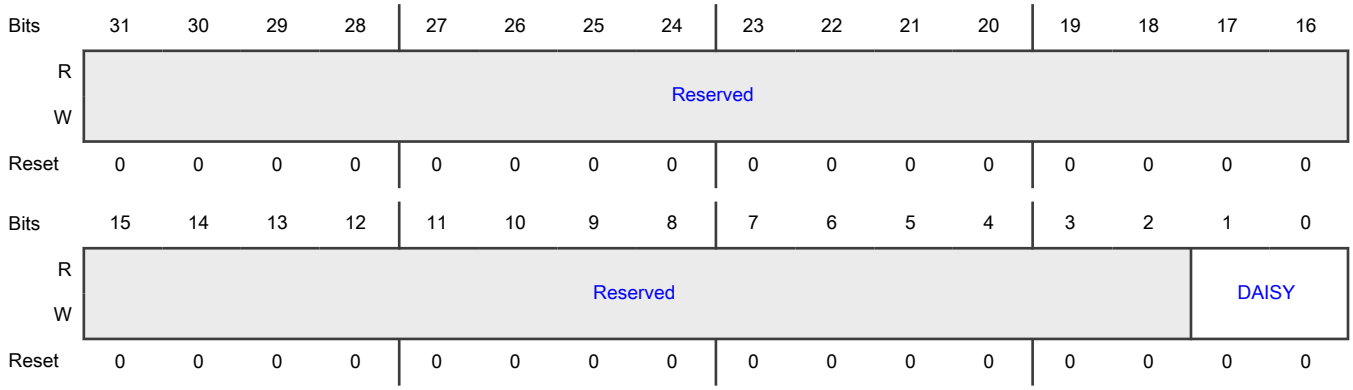
Offset

Register	Offset
LPUART7_IPP_IND_LPUART_CTS_N_SELECT_INPUT	15Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-2 —	- Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart7, In Pin: ipp_ind_lpuart_cts_n 00b - Selecting Pad: GPIO_AON_04 for Mode: ALT6 01b - Selecting Pad: GPIO_AON_16 for Mode: ALT8 10b - Selecting Pad: GPIO_AON_24 for Mode: ALT3

17.4.2.90 LPUART7_IPP_IND_LPUART_RXD_SELECT_INPUT DAISY Register (LPUART7_IPP_IND_LPUART_RXD_SELECT_INPUT)

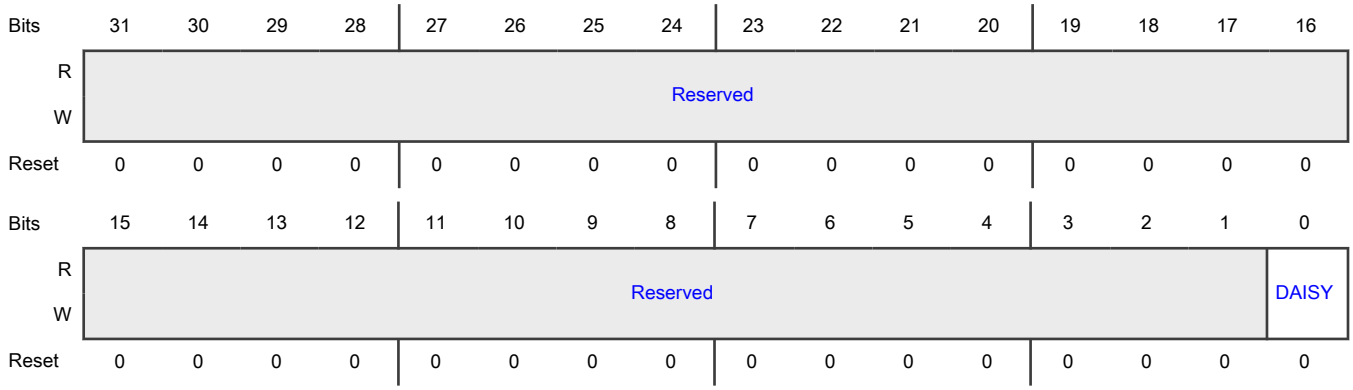
Offset

Register	Offset
LPUART7_IPP_IND_LPUART_RXD_SELECT_INPUT	160h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart7, In Pin: ipp_ind_lpuart_rxd 0b - Selecting Pad: GPIO_AON_18 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_23 for Mode: ALT2

17.4.2.91 LPUART7_IPP_IND_LPUART_TXD_SELECT_INPUT DAISY Register (LPUART7_IPP_IND_LPUART_TXD_SELECT_INPUT)

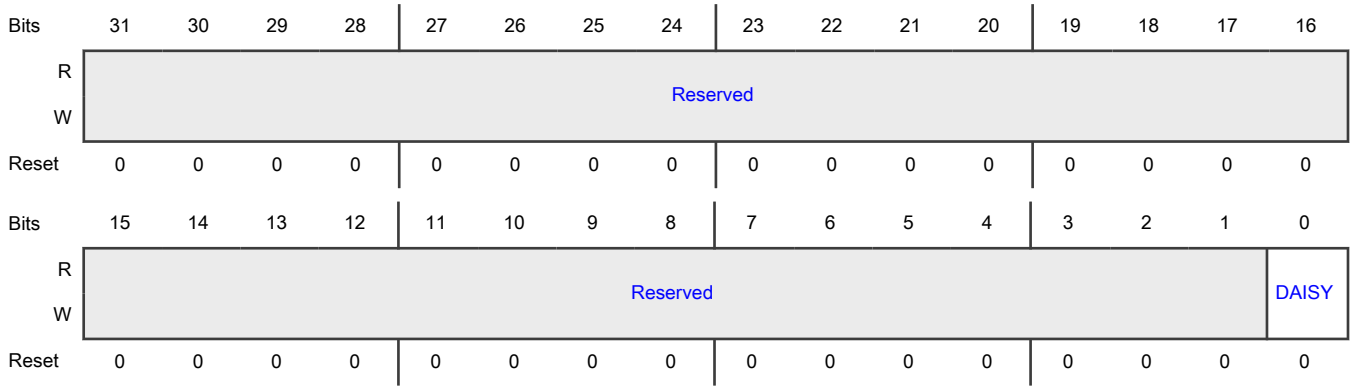
Offset

Register	Offset
LPUART7_IPP_IND_LPUART_TXD_SELECT_INPUT	164h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: lpuart7, In Pin: ipp_ind_lpuart_txd 0b - Selecting Pad: GPIO_AON_17 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_22 for Mode: ALT2

17.4.2.92 SAI1_IPG_CLK_SAI_MCLK_SELECT_INPUT DAISY Register (SAI1_IPG_CLK_SAI_MCLK_SELECT_INPUT)

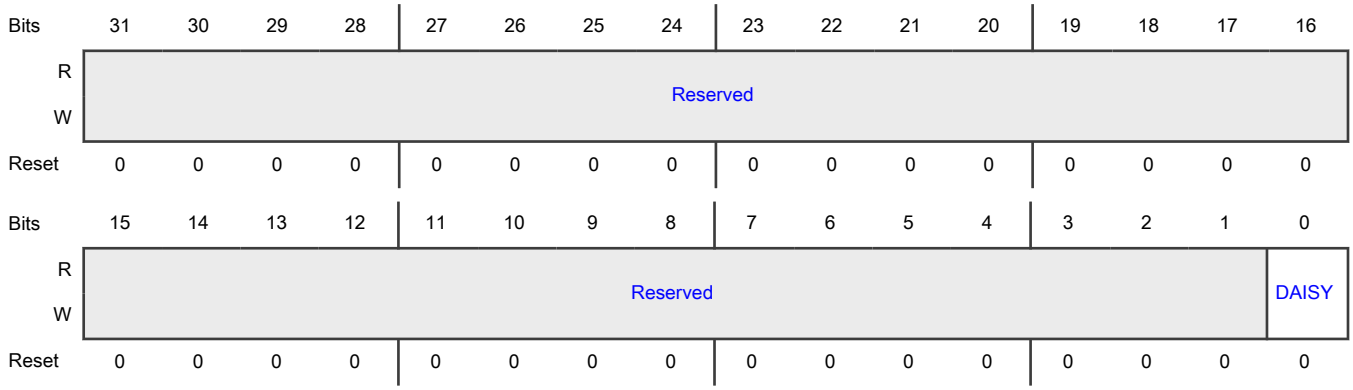
Offset

Register	Offset
SAI1_IPG_CLK_SAI_MCLK_SELECT_INPUT	168h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai1, In Pin: ipg_clk_sai_mclk 0b - Selecting Pad: GPIO_AON_07 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_24 for Mode: ALT4

17.4.2.93 SAI1_IPP_IND_SAI_RXBCLK_SELECT_INPUT DAISY Register (SAI1_IPP_IND_SAI_RXBCLK_SELECT_INPUT)

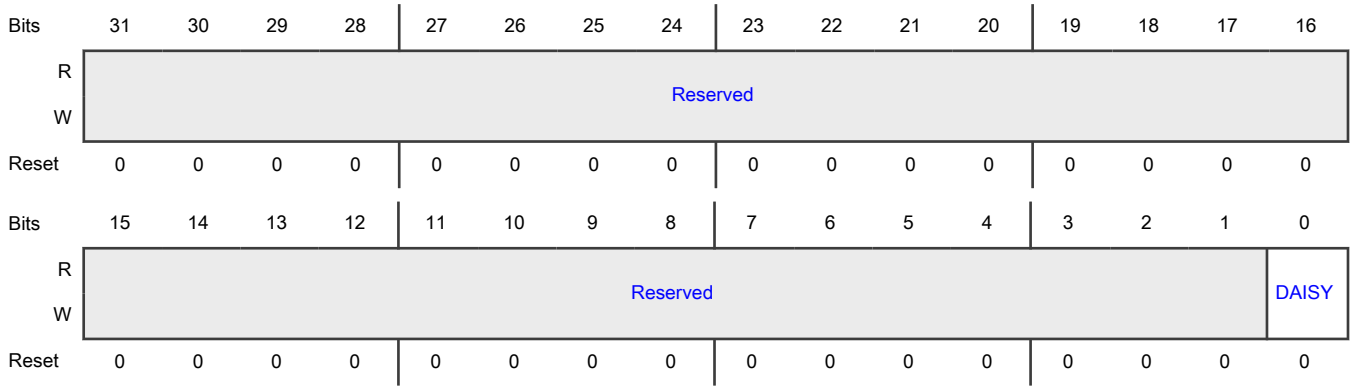
Offset

Register	Offset
SAI1_IPP_IND_SAI_RXBCLK_SELECT_INPUT	16Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai1, In Pin: ipp_ind_sai_rxbclk 0b - Selecting Pad: GPIO_AON_09 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_26 for Mode: ALT4

17.4.2.94 SAI1_IPP_IND_SAI_RXDATA_SELECT_INPUT_0 DAISY Register (SAI1_IPP_IND_SAI_RXDATA_SELECT_INPUT_0)

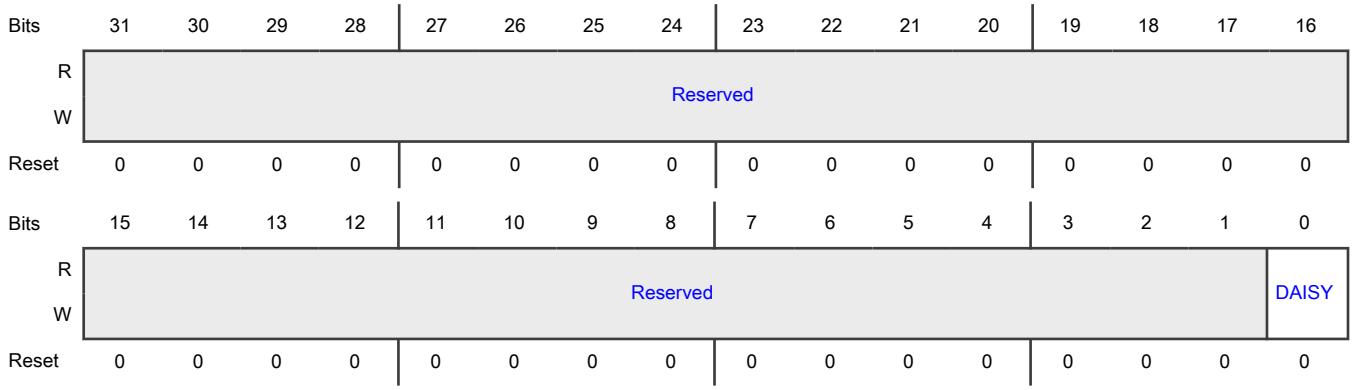
Offset

Register	Offset
SAI1_IPP_IND_SAI_RXDATA_SELECT_INPUT_0	170h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai1, In Pin: ipp_ind_sai_rxdata0 0b - Selecting Pad: GPIO_AON_08 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_25 for Mode: ALT4

17.4.2.95 SAI1_IPP_IND_SAI_RXDATA_SELECT_INPUT_1 DAISY Register (SAI1_IPP_IND_SAI_RXDATA_SELECT_INPUT_1)

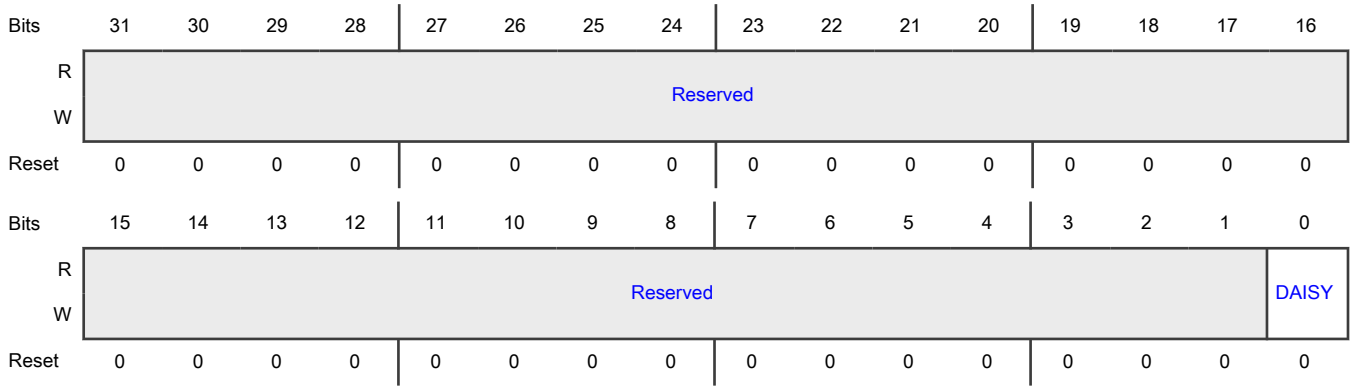
Offset

Register	Offset
SAI1_IPP_IND_SAI_RXDATA_SELECT_INPUT_1	174h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai1, In Pin: ipp_ind_sai_rxdata1 0b - Selecting Pad: GPIO_AON_04 for Mode: ALT3 1b - Selecting Pad: GPIO_AON_21 for Mode: ALT9

17.4.2.96 SAI1_IPP_IND_SAI_RXSYNC_SELECT_INPUT DAISY Register (SAI1_IPP_IND_SAI_RXSYNC_SELECT_INPUT)

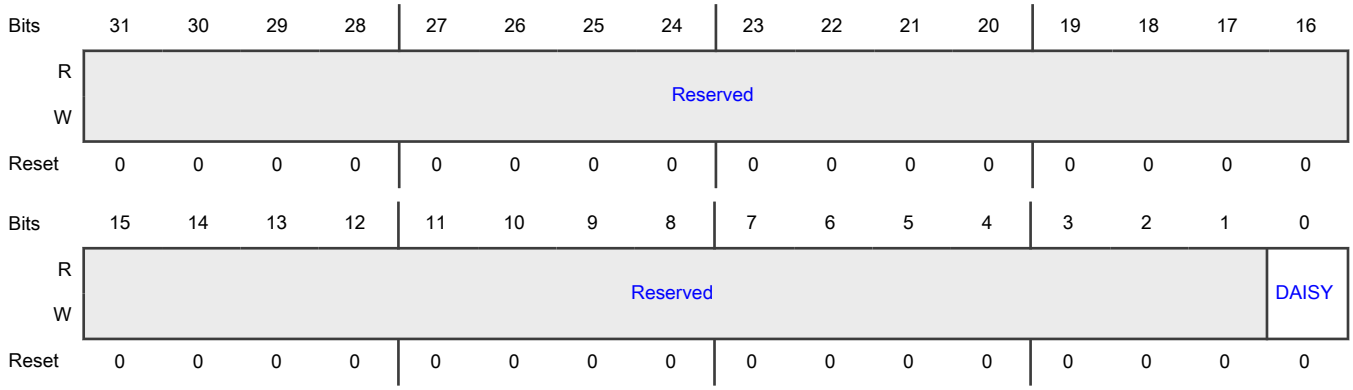
Offset

Register	Offset
SAI1_IPP_IND_SAI_RXSYNC_SELECT_INPUT	178h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai1, In Pin: ipp_ind_sai_rxsync 0b - Selecting Pad: GPIO_AON_10 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_27 for Mode: ALT4

17.4.2.97 SAI1_IPP_IND_SAI_TXBCLK_SELECT_INPUT DAISY Register (SAI1_IPP_IND_SAI_TXBCLK_SELECT_INPUT)

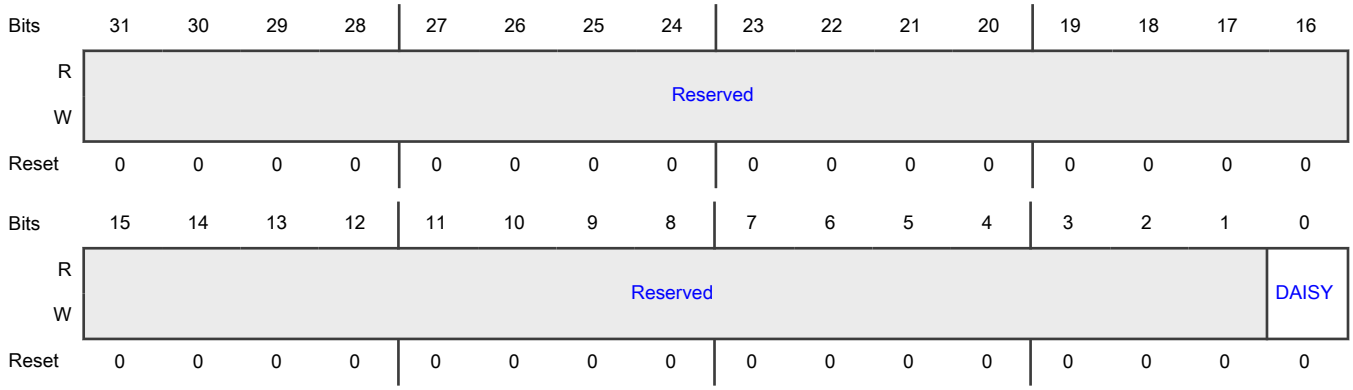
Offset

Register	Offset
SAI1_IPP_IND_SAI_TXBCLK_SELECT_INPUT	17Ch

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai1, In Pin: ipp_ind_sai_txbclk 0b - Selecting Pad: GPIO_AON_06 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_23 for Mode: ALT4

17.4.2.98 SAI1_IPP_IND_SAI_TXSYNC_SELECT_INPUT DAISY Register (SAI1_IPP_IND_SAI_TXSYNC_SELECT_INPUT)

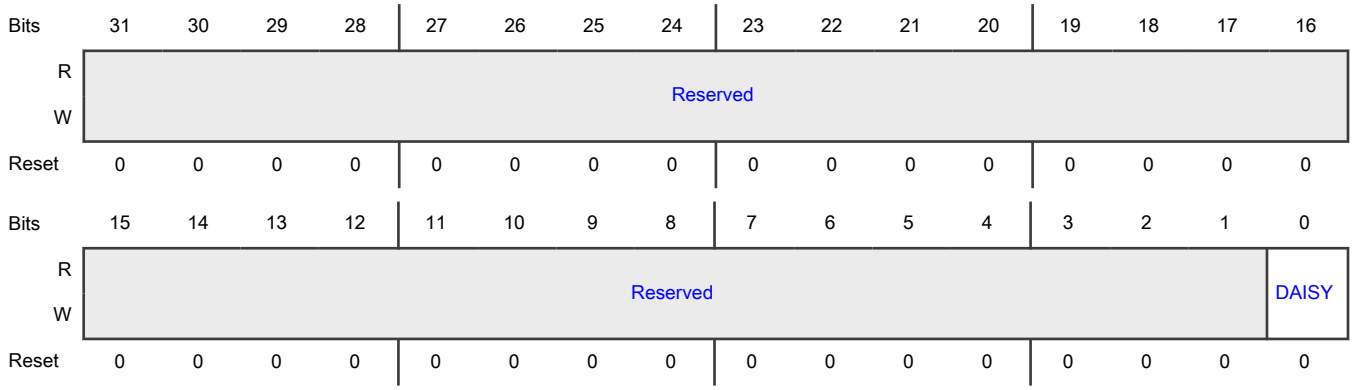
Offset

Register	Offset
SAI1_IPP_IND_SAI_TXSYNC_SELECT_INPUT	180h

Function

DAISY Register

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: sai1, In Pin: ipp_ind_sai_txsync 0b - Selecting Pad: GPIO_AON_05 for Mode: ALT2 1b - Selecting Pad: GPIO_AON_22 for Mode: ALT4

Chapter 18
General-Purpose Input/Output (GPIO)

18.1 Chip-specific GPIO Information

Table 124. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

18.2 Overview

The GPIO module communicates to the processor core via a zero wait-state interface for maximum pin performance.

18.2.1 Block diagram

The following figure shows the block diagram for the GPIO module.

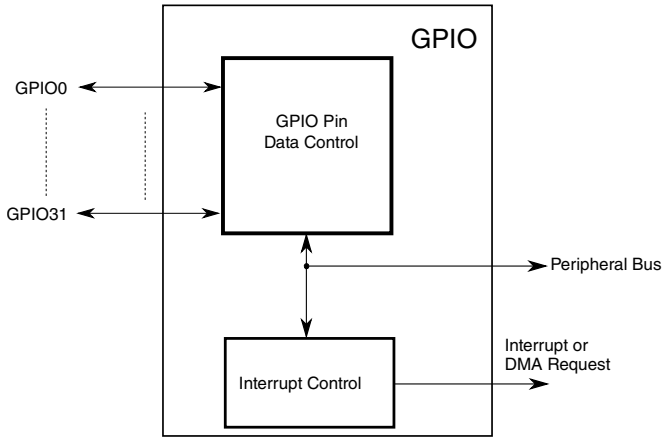


Figure 68. GPIO block diagram

18.2.2 Features

The GPIO module includes the following features:

- Port Data Input (PDIR) register displays the logic value on each pin when the pin is configured for any digital function provided the corresponding Port Control and Interrupt module for that pin are enabled.
- Port Data Output (PDOR) register with corresponding set/clear/toggle registers controls output data of each pin when the pin is configured for the GPIO function.
- Port Data Direction (PDDR) register controls the direction of each pin when the pin is configured for the GPIO function.
- Port Input Disable (PIDR) register controls the disable of the input for each general-purpose pin.
- Pin interrupts
  - Interrupt flag and enable registers for each pin are functional in all digital pin muxing modes.
  - Support for interrupt or DMA request configured per pin.
  - Support for edge sensitive (rising or falling, or both) or level sensitive (low, high) configured per pin.
  - Asynchronous wake-up in Low-Power modes.
  - GPIO module generates a total of 2 interrupts and 2 DMA requests.
  - Each pin can be used to generate a single interrupt or DMA request.
- Protection registers
  - Each pin is configured for Secure or Non-Secure and Privilege/Non-Privilege access.
  - Each interrupt and DMA request domain is configured for Secure or Non-Secure and Privilege/Non-Privilege access.

18.3 Functional description

18.3.1 Low-Power modes

The GPIO module can be configured to exit the Low-Power mode via an asynchronous wake-up signal if an enabled interrupt is detected.

18.3.2 Debug mode

The GPIO module remains functional in debug mode.

18.3.3 General-purpose input

The logic state of each pin is available via the Port Data Input registers (PDIR) if,

- the corresponding bit in the Port Input Disable Register (PIDR) is clear, and
- the pin is configured for a digital function.

18.3.4 General-purpose output

The logic state of each pin is controlled via the Port Data Output Registers(PDOR) and Port Data Direction Registers(PDDR), provided the pin is configured for the GPIO function. The following table depicts the conditions for a pin to be configured as input/output.

If	Then
A pin is configured for the GPIO function and the corresponding PDDR bit is clear.	The pin is configured as an input.

Table continues on the next page...

Table continued from the previous page...

A pin is configured for the GPIO function and the corresponding PDDR bit is set.	The pin is configured as an output and the logic state of the pin is equal to the corresponding PDOR.

For efficient bit manipulation on the general-purpose outputs, pin data set, pin data clear, and pin data toggle registers allows one or more outputs within one port to be set, cleared, or toggled from a single register write, eliminating the need for a read-modify-write operation to prevent changing pins accidentally.

18.3.5 Clocking

The GPIO module receives a single clock, which is used for register access and synchronize external pin inputs. There are no special considerations.

18.3.6 Reset

The GPIO module receives a single reset, which resets the peripheral. There are no special considerations.

18.3.7 External interrupts

The external interrupt capability of the GPIO module is available in all digital pin muxing modes.

GPIO module generates a total of 2 interrupts and 2 DMA requests. Each pin can be individually configured for interrupt or DMA request.

- Each output implements a separate interrupt status flag register for that domain
- Each output generates a single interrupt
- Each output generates a single DMA request

Each pin can be individually configured for any of the following external interrupt modes:

Table 125. Available pin configuration for external interrupt

Signal conditions	Software polling using flags	Interrupts	DMA requests
Rising-edge	Yes	Yes	Yes
Falling-edge	Yes	Yes	Yes
Rising- and falling-edge	Yes	Yes	Yes
High-level	—	Yes	—
Low-level	—	Yes	—

The interrupt status flag is set when the configured edge or level is detected on the pin. Unless the GPIO module is in a Low-Power mode, the input is first synchronized to the system clock to detect the configured level or edge transition.

The GPIO module generates a pin interrupt that asserts when the interrupt status flag is set for any enabled interrupt for that output. The interrupt negates after the interrupt status flags for all enabled interrupts are cleared by writing a logic 1 to the ISF flag in either the ISFR or ICRn registers.

The GPIO module generates a DMA request that asserts when the interrupt status flag is set for any enabled DMA request in that output. The DMA request negates after the DMA transfer is completed, because that clears the interrupt status flags for all enabled DMA requests.

In Low-Power mode, the interrupt status flag for any enabled interrupt is asynchronously set if the required level or edge is detected. This also generates an asynchronous wake-up signal to exit the Low-Power mode.

18.3.8 Global interrupt control

The two global interrupt control registers (GICLR and GICHR) allow a single register write to update the upper 16-bit of the Interrupt Control Register for up to 16 pins, all with the same value.

The global interrupt control registers allow software to quickly configure multiple pins within the same port with the same interrupt configuration.

The global interrupt control registers are write-only registers and always read as 0.

18.3.9 DMA

The GPIO module generates 2 requests. Each pin can be configured to generate one of those DMA requests on either or both pin edge detection. See External interrupts section for further details.

18.3.10 Access protection

The GPIO module implements access protection registers (PCNS and PCNP) for each pin as follows:

Access	Description
Secure	Pins configured for Secure access can only be written or read by software in Secure state.
Non-Secure	Pins configured for Non-Secure access can only be written or read by software in Non-Secure state.
Privilege	Pins configured for Privilege access can only be written by software in Privilege state.
Non-Privilege	Pins configured for Non-Privilege access can be written by software in both Privilege and Non-Privilege states.

The GPIO module implements access protection registers (ICNS and ICNP) for each interrupt and DMA request as follows:

Access	Description
Secure	Outputs configured for Secure access can only be configured by software in Secure state.
Non-Secure	Outputs configured for Non-Secure access can only be configured by software in Non-Secure state.
Privilege	Outputs configured for Privilege access can only be configured by software in Privilege state.
Non-Privilege	Outputs configured for Non-Privilege access can be configured by software in both Privilege and Non-Privilege states.

The access protection registers (PCNS, PCNP, ICNS and ICNP) can only be written by software in secure-privilege state and can be optionally locked until the next reset.

Configuring a pin interrupt/DMA request requires the following access permissions:

- Access permission to configure that pin.
- Access permission to configure to the desired interrupt/DMA request or writing IRQC=0.
- Access permission to configure with the existing interrupt/DMA request or currently IRQC=0.
- The interrupt configuration is not locked.

18.4 External signals

Table 126. GPIO external signals

Signal	Description		Direction
GPIO31 - GPIO0	General-purpose input/output		I/O
	State meaning	Asserted: The pin is logic 1. Deasserted: The pin is logic 0.	
	Timing	Assertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock. Deassertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock.	

NOTE

Not all pins within each GPIO are implemented on each device. See the chapter on signal multiplexing for the number of GPIO pins within each port available in the device.

18.5 Initialization

To initialize the GPIO module:

1. Initialize GPIO pins for output function
    a. Configure the output logic value for each pin by using PDOR register.
    b. Configure the direction for each pin by using PDDR register.
2. Initialize interrupt function
    - Write to ICRn for the corresponding pins and desired configuration.
    - If the pin is previously used for a different function, first write 0x0100_0000 to ICRn register to disable the previous function and clear the flag.

18.6 Application information

The GPIO module has the following applications:

- Read the state of a single pin
    - Perform a byte read of PnDR register.
- Update the state of a single pin
    - Perform a byte write to PnDR register.
- Read the state of multiple pins
    - Read PDIR register.
- Update the state of multiple pins
    - Write to PDOR register bits.
    - Write to PSOR register to set PDOR register bits.
    - Write to PCOR register to clear PDOR register bits.
    - Write to PTOR register to toggle PDOR register bits.

18.7 Memory map and register definition

The GPIO registers support 8-bit, 16-bit, or 32-bit accesses. Any read or write access to the GPIO memory space that is outside the valid memory map results in a bus error.

NOTE

For simplicity, each GPIO port's register appears with the same width of 32 bits, corresponding to 32 pins. The actual number of pins per port (and therefore the number of usable control bits per port register) is chip-specific. See the Signal Multiplexing for exact control bits for each port.

18.7.1 GPIO register descriptions

18.7.1.1 GPIO memory map

RGPIO1 base address: 4740_0000h

RGPIO2 base address: 4381_0000h

RGPIO3 base address: 4382_0000h

RGPIO4 base address: 4383_0000h

RGPIO5 base address: 4384_0000h

RGPIO6 base address: 4385_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0201_0001h
4h	Parameter (PARAM)	32	R	0000_0002h
Ch	Lock (LOCK)	32	RW	0000_0000h
10h	Pin Control Non-Secure (PCNS)	32	RW	0000_0000h
14h	Interrupt Control Non-Secure (ICNS)	32	RW	0000_0000h
18h	Pin Control Non-Privilege (PCNP)	32	RW	0000_0000h
1Ch	Interrupt Control Non-Privilege (ICNP)	32	RW	0000_0000h
40h	Port Data Output Register (PDOR)	32	RW	0000_0000h
44h	Port Set Output Register (PSOR)	32	W	0000_0000h
48h	Port Clear Output Register (PCOR)	32	W	0000_0000h
4Ch	Port Toggle Output Register (PTOR)	32	W	0000_0000h
50h	Port Data Input Register (PDIR)	32	R	0000_0000h
54h	Port Data Direction Register (PDDR)	32	RW	0000_0000h
58h	Port Input Disable Register (PIDR)	32	RW	0000_0000h
60h - 7Fh	Pin Data Register a (P0DR - P31DR)	8	RW	00h
80h - FCh	Interrupt Control Register a (ICR0 - ICR31)	32	RW	0000_0000h
100h	Global Interrupt Control Low Register (GICLR)	32	W	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
104h	Global Interrupt Control High Register (GICHR)	32	W	0000_0000h
120h - 124h	Interrupt Status Flag Register (ISFR0 - ISFR1)	32	RW	0000_0000h

18.7.1.2 Version ID (VERID)

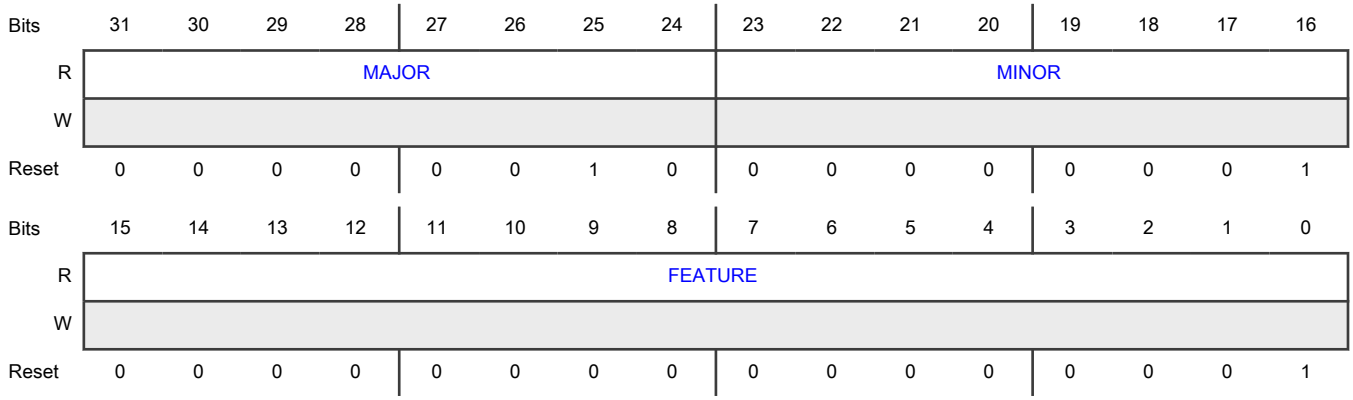
Offset

Register	Offset
VERID	0h

Function

Indicates the version ID number of each GPIO.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the specification.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000_0000_0000_0000b - Basic implementation. 0000_0000_0000_0001b - Protection registers implemented.

18.7.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function
Indicates the interrupt number of each GPIO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												IRQNUM			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-4 —	Reserved
3-0 IRQNUM	Interrupt Number This field indicates the number of Interrupt/DMA request domains.

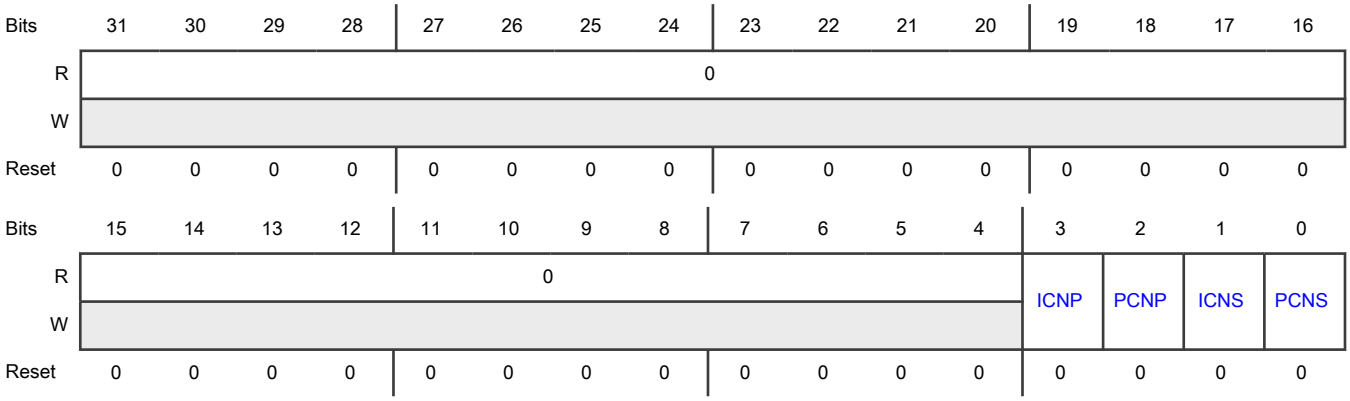
18.7.1.4 Lock (LOCK)

Offset

Register	Offset
LOCK	Ch

Function
Locks the Non-Secure and Non-Privilege access protection registers. This register can be written only by software in Secure-Privilege state.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 ICNP	Lock ICNP This field locks ICNP register. When set, the ICNP register is not writable until the next reset. 0b - ICNP register is writable by software in Secure-Privilege state. 1b - ICNP register is not writable until the next reset.
2 PCNP	Lock PCNP This field locks PCNP register. When set, the PCNP register is not writable until the next reset. 0b - PCNP register is writable by software in Secure-Privilege state. 1b - PCNP register is not writable until the next reset.
1 ICNS	Lock ICNS This field locks ICNS register. When set, the ICNS register is not writable until the next reset. 0b - ICNS register is writable by software in Secure-Privilege state. 1b - ICNS register is not writable until the next reset.
0 PCNS	Lock PCNS This field locks PCNS register. When set, the PCNS register is not writable until the next reset. 0b - PCNS register is writable by software in Secure-Privilege state. 1b - PCNS register is not writable until the next reset.

18.7.1.5 Pin Control Non-Secure (PCNS)

Offset

Register	Offset
PCNS	10h

Function

Configures Secure or Non-Secure access protection for each pin. This register can only be written by software in Secure-Privilege state if it is not locked (LOCK[PCNS] = 0).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NSE31	NSE30	NSE29	NSE28	NSE27	NSE26	NSE25	NSE24	NSE23	NSE22	NSE21	NSE20	NSE19	NSE18	NSE17	NSE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NSE15	NSE14	NSE13	NSE12	NSE11	NSE10	NSE9	NSE8	NSE7	NSE6	NSE5	NSE4	NSE3	NSE2	NSE1	NSE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 NSEn	Non-Secure Enable This field configures the Secure or Non-Secure access protection for each pin. 0b - The pin is configured for Secure access. Read or write access to the corresponding pin's registers and bit fields is only allowed by software in Secure state. When the corresponding pin's registers are accessed by software in Non-Secure state, all bits in the registers related to that pin are read zero and write ignored. 1b - The pin is configured for Non-Secure access. Read or write access to the corresponding pin's registers and bit fields is only allowed by software in Non-Secure state. When the corresponding pin's registers are accessed by software in Secure state, all bits in the registers related to that pin are read zero and write ignored.

18.7.1.6 Interrupt Control Non-Secure (ICNS)

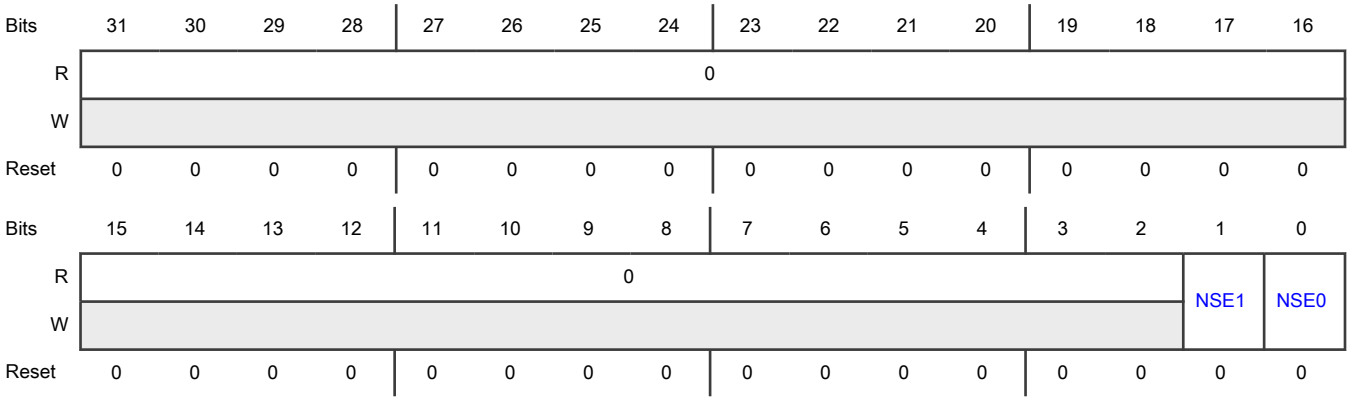
Offset

Register	Offset
ICNS	14h

Function

Configures Secure/Non-Secure access protection for each interrupt or DMA request. This register can only be updated by software in Secure-Privilege state if it is not locked (LOCK[ICNS] = 0).

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 NSEn	Non-Secure Enable This field configures Secure or Non-Secure access protection for each interrupt or DMA request. **NOTE** See GPIO chip-specific info to determine which GPIO instances support interrupt, DMA request, or trigger capabilities. 0b - The interrupt or DMA request is configured for Secure access. Only software in Secure state can configure a pin to use the corresponding interrupt or DMA request or reconfigure a pin that is already configured to use the corresponding interrupt or DMA request. 1b - The interrupt or DMA request is configured for Non-Secure access. Only software in Non-Secure state can configure a pin to use the corresponding interrupt or DMA request or reconfigure a pin that is already configured to use the corresponding interrupt or DMA request.

18.7.1.7 Pin Control Non-Privilege (PCNP)

Offset

Register	Offset
PCNP	18h

Function

Configures either for Privilege or for both (Privilege and Non-Privilege) access protection for each pin. This register can only be updated by software in Secure-Privilege state if it is not locked (LOCK[PCNP] = 0).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NPE31	NPE30	NPE29	NPE28	NPE27	NPE26	NPE25	NPE24	NPE23	NPE22	NPE21	NPE20	NPE19	NPE18	NPE17	NPE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NPE15	NPE14	NPE13	NPE12	NPE11	NPE10	NPE9	NPE8	NPE7	NPE6	NPE5	NPE4	NPE3	NPE2	NPE1	NPE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 NPEn	Non-Privilege Enable This field configures Privilege/Non-Privilege access protection for each pin. 0b - The pin is configured for Privilege access. Write access to the corresponding pin's registers and bit fields is allowed only by software in Privilege state. When the corresponding pin's registers and bit fields are accessed by software in Non-Privilege state, all bits related to that pin in this GPIO are readable but write ignored. 1b - The pin is configured for Non-Privilege access, Read or write access to the corresponding pin's registers is allowed by software in both Privilege and Non-Privilege state.

18.7.1.8 Interrupt Control Non-Privilege (ICNP)

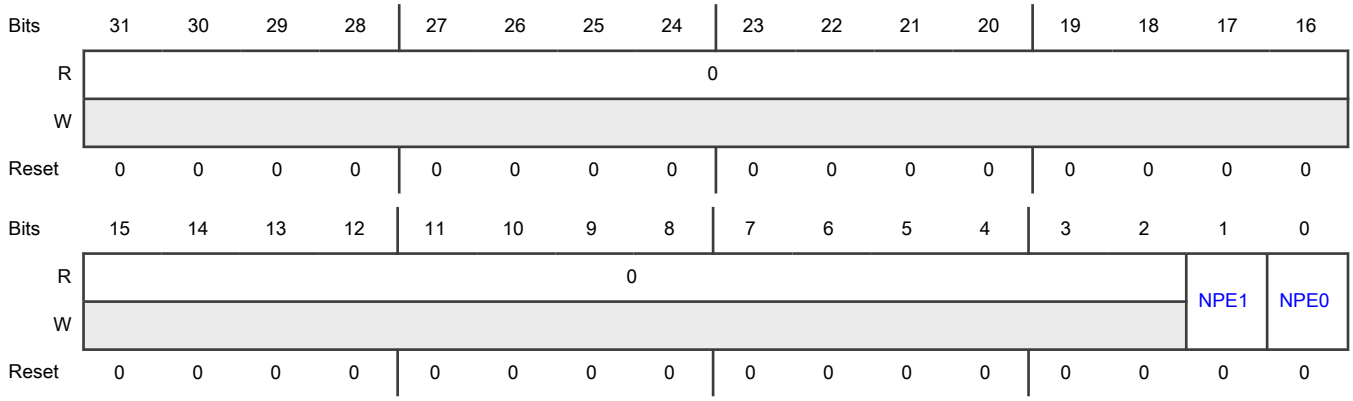
Offset

Register	Offset
ICNP	1Ch

Function

Configures Privilege/Non-Privilege access protection for each interrupt/DMA request. This register can only be updated by software in Secure-Privilege state if it is not locked (LOCK[ICNP] = 0).

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 NPEn	Non-Privilege Enable This field configures Privilege/Non-Privilege access protection for each interrupt/DMA request. **NOTE** See GPIO chip-specific info to determine which GPIO instances support interrupt, DMA request, or trigger capabilities. 0b - The pin is configured for Privilege access. Only software in Privilege state can configure a pin to use the corresponding interrupt/DMA request or reconfigure a pin that is already configured to use the corresponding interrupt/DMA request. 1b - The pin is configured for Non-Privilege access. Software in either Privilege or Non-Privilege state can configure a pin to use the corresponding interrupt/DMA request or reconfigure a pin that is already configured to use the corresponding interrupt/DMA request.

18.7.1.9 Port Data Output Register (PDOR)

Offset

Register	Offset
PDOR	40h

Function

Configures the logic levels that are driven on each general-purpose output pin.

NOTE

Do not modify the pin configuration registers associated with pins that are not available in your selected package. These unbonded pins are default set to Disable state for lowest power consumption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PDO31	PDO30	PDO29	PDO28	PDO27	PDO26	PDO25	PDO24	PDO23	PDO22	PDO21	PDO20	PDO19	PDO18	PDO17	PDO16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PDO15	PDO14	PDO13	PDO12	PDO11	PDO10	PDO9	PDO8	PDO7	PDO6	PDO5	PDO4	PDO3	PDO2	PDO1	PDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PDOn	Port Data Output This field configures the logic level on the pin if it is configured for general-purpose output. **NOTE** Reading the fields for unbonded pins returns an undefined value. 0b - Logic level 0 is driven on pin, if the pin is configured for general-purpose output. 1b - Logic level 1 is driven on pin, if the pin is configured for general-purpose output.

18.7.1.10 Port Set Output Register (PSOR)

Offset

Register	Offset
PSOR	44h

Function

Configures whether to set the corresponding fields of the PDOR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTSO 31	PTSO 30	PTSO 29	PTSO 28	PTSO 27	PTSO 26	PTSO 25	PTSO 24	PTSO 23	PTSO 22	PTSO 21	PTSO 20	PTSO 19	PTSO 18	PTSO 17	PTSO 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTSO 15	PTSO 14	PTSO 13	PTSO 12	PTSO 11	PTSO 10	PTSO 9	PTSO 8	PTSO 7	PTSO 6	PTSO 5	PTSO 4	PTSO 3	PTSO 2	PTSO 1	PTSO 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PTSOn	Port Set Output Writing to this field updates the content of the corresponding field in the PDOR as follows: 0b - Corresponding field of PDOR[PDOn] does not change. 1b - Corresponding field of PDOR[PDOn] is set to logic 1.

18.7.1.11 Port Clear Output Register (PCOR)

Offset

Register	Offset
PCOR	48h

Function

Configures whether to clear the corresponding fields of PDOR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTCO 31	PTCO 30	PTCO 29	PTCO 28	PTCO 27	PTCO 26	PTCO 25	PTCO 24	PTCO 23	PTCO 22	PTCO 21	PTCO 20	PTCO 19	PTCO 18	PTCO 17	PTCO 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTCO 15	PTCO 14	PTCO 13	PTCO 12	PTCO 11	PTCO 10	PTCO 9	PTCO 8	PTCO 7	PTCO 6	PTCO 5	PTCO 4	PTCO 3	PTCO 2	PTCO 1	PTCO 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PTCOn	Port Clear Output Writing to this field updates the content of the corresponding field in the PDOR as follows: 0b - Corresponding field of PDOR[PDOn] does not change. 1b - Corresponding field of PDOR[PDOn] is cleared to logic 0.

18.7.1.12 Port Toggle Output Register (PTOR)

Offset

Register	Offset
PTOR	4Ch

Function
Configures whether to toggle the corresponding fields of PDOR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTTO 31	PTTO 30	PTTO 29	PTTO 28	PTTO 27	PTTO 26	PTTO 25	PTTO 24	PTTO 23	PTTO 22	PTTO 21	PTTO 20	PTTO 19	PTTO 18	PTTO 17	PTTO 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTTO 15	PTTO 14	PTTO 13	PTTO 12	PTTO 11	PTTO 10	PTTO 9	PTTO 8	PTTO 7	PTTO 6	PTTO 5	PTTO 4	PTTO 3	PTTO 2	PTTO 1	PTTO 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PTTOn	Port Toggle Output Writing to this field updates the content of the corresponding field in the PDOR as follows: 0b - Corresponding field of PDOR[PDOn] does not change. 1b - Corresponding field of PDOR[PDOn] is set to the inverse of its current logic state.

18.7.1.13 Port Data Input Register (PDIR)

Offset

Register	Offset
PDIR	50h

Function

Captures the logic levels of each general-purpose input pin.

NOTE

Do not modify the pin configuration registers associated with the pins that are not available in your selected package. These unbonded pins are default set to Disable state for lowest power consumption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PDI31	PDI30	PDI29	PDI28	PDI27	PDI26	PDI25	PDI24	PDI23	PDI22	PDI21	PDI20	PDI19	PDI18	PDI17	PDI16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PDI15	PDI14	PDI13	PDI12	PDI11	PDI10	PDI9	PDI8	PDI7	PDI6	PDI5	PDI4	PDI3	PDI2	PDI1	PDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PDIn	Port Data Input The field indicates the logic level of the pin that is configured for use by digital function. Reads 0 at the unimplemented pins or pins that are not configured for a digital function. 0b - Pin logic level is logic 0 or is not configured for use by digital function. 1b - Pin logic level is logic 1.

18.7.1.14 Port Data Direction Register (PDDR)

Offset

Register	Offset
PDDR	54h

Function

Configures the individual port pins for input or output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PDD31	PDD30	PDD29	PDD28	PDD27	PDD26	PDD25	PDD24	PDD23	PDD22	PDD21	PDD20	PDD19	PDD18	PDD17	PDD16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PDD15	PDD14	PDD13	PDD12	PDD11	PDD10	PDD9	PDD8	PDD7	PDD6	PDD5	PDD4	PDD3	PDD2	PDD1	PDD0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PDDn	Port Data Direction This field configures individual port pins for input or output. 0b - Pin is configured as general-purpose input for the GPIO function. 1b - Pin is configured as general-purpose output for the GPIO function.

18.7.1.15 Port Input Disable Register (PIDR)

Offset

Register	Offset
PIDR	58h

Function

Disables the input for each general-purpose pin, which prevents the value from being reported in the PDIR register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24	PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8	PID7	PID6	PID5	PID4	PID3	PID2	PID1	PID0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PIDn	Port Input Disable This field is used to disable a pin for general-purpose input 0b - Pin is configured for general-purpose input provided, the pin is configured for any digital function. 1b - Pin is disabled for general-purpose input.

18.7.1.16 Pin Data Register a (P0DR - P31DR)

Offset

For a = 0 to 31:

Register	Offset
PaDR	60h + (a × 1h)

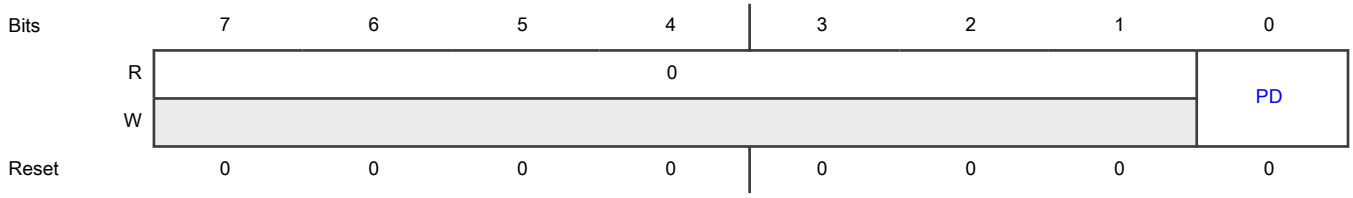
Function

Configures the data feature of a pin. Pin that is unimplemented or not configured for a digital function reads as zero.

NOTE

Do not modify PaDR registers associated with the pins that are not available in your selected package. These unbonded pins are default set to Disable state for lowest power consumption.

Diagram



Fields

Field	Function
7-1 —	Reserved
0 PD	Pin Data (input and output) Writing to this bit field of PaDR register will update corresponding bit a in the PDOR register, reading this bit field of PaDR register will return value in the corresponding bit a of the PDIR register. 0b - Pin logic level is logic zero or not configured for use by digital function. 1b - Pin logic level is logic one.

18.7.1.17 Interrupt Control Register a (ICR0 - ICR31)

Offset

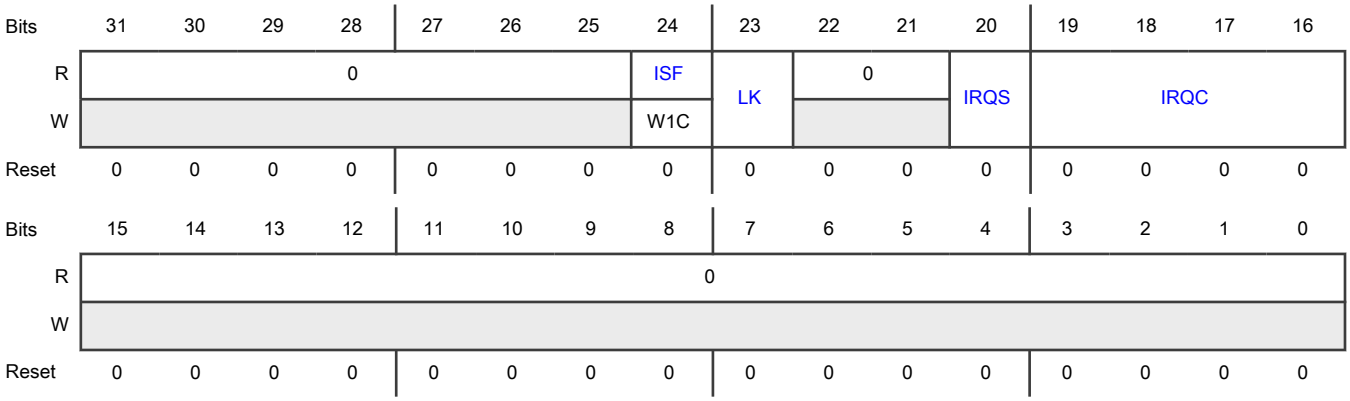
For a = 0 to 31:

Register	Offset
ICRa	80h + (a × 4h)

Function

Configures interrupt features on each pin.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 ISF	Interrupt Status Flag This field indicates whether the configured interrupt is detected. The pin interrupt configuration is valid in all digital pin muxing modes. The bit fields in the ISFR register have the same function. The interrupt status flag can be cleared with either register bit. 0b - Configured interrupt is not detected. 1b - Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.
23 LK	Lock Register This field is used to lock ICR[23:0], and the locked field cannot be updated until the next system reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt configuration by ICR[23:0] is not locked and can be updated. 1b - Interrupt configuration by ICR[23:0] is locked and cannot be updated until next system reset.
22-21 —	Reserved
20 IRQS	Interrupt Select This field configures the interrupt/DMA request. **NOTE** See GPIO chip-specific info to determine which GPIO instances support interrupt, DMA request, or trigger capabilities. 0b - Interrupt/DMA request 0. 1b - Interrupt/DMA request 1.
19-16 IRQC	Interrupt Configuration The pin interrupt configuration is valid in all digital pin muxing modes. When changing the interrupt configuration, it is recommended to first disable the interrupt status flag and then write the new configuration. The corresponding pin is configured to generate interrupt, or DMA request as follows: **NOTE** See GPIO chip-specific info to determine which GPIO instances support interrupt, DMA request, or trigger capabilities. 0000b - Interrupt Status Flag (ISF) is disabled. 0001b - ISF flag and DMA request on rising edge. 0010b - ISF flag and DMA request on falling edge. 0011b - ISF flag and DMA request on either edge. 0100b - Reserved. 0101b - ISF flag sets on rising edge. 0110b - ISF flag sets on falling edge. 0111b - ISF flag sets on either edge. 1000b - ISF flag and Interrupt when logic 0. 1001b - ISF flag and Interrupt on rising-edge. 1010b - ISF flag and Interrupt on falling-edge. 1011b - ISF flag and Interrupt on either edge. 1100b - ISF flag and Interrupt when logic 1. 1101b - Reserved 1110b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1111b - Reserved.
15-0 —	Reserved

18.7.1.18 Global Interrupt Control Low Register (GICLR)

Offset

Register	Offset
GICLR	100h

Function

Only 32-bit writes are supported to this register, any 16-bit or 8-bit writes are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	GIWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	GIWE 15	GIWE 14	GIWE 13	GIWE 12	GIWE 11	GIWE 10	GIWE 9	GIWE 8	GIWE 7	GIWE 6	GIWE 5	GIWE 4	GIWE 3	GIWE 2	GIWE 1	GIWE 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 GIWD	Global Interrupt Write Data Write value that is written to upper 16 bits of the Interrupt Control Registers that are selected by GIWE.
15-0 GIWEn	Global Interrupt Write Enable This field selects whether the upper 16-bit field of the Interrupt Control Register ICR(n) will be updated with the value in GIWD. 0b - Upper 16-bit of corresponding Interrupt Control Register is not updated with the value in GIWD. 1b - Upper 16-bit of corresponding Interrupt Control Register is updated with the value in GIWD.

18.7.1.19 Global Interrupt Control High Register (GICHR)

Offset

Register	Offset
GICHR	104h

Function
Only 32-bit writes are supported to this register, any 16-bit or 8-bit writes are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	GIWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	GIWE 31	GIWE 30	GIWE 29	GIWE 28	GIWE 27	GIWE 26	GIWE 25	GIWE 24	GIWE 23	GIWE 22	GIWE 21	GIWE 20	GIWE 19	GIWE 18	GIWE 17	GIWE 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 GIWD	Global Interrupt Write Data Write value that is written to upper 16 bits of the Interrupt Control Registers that are selected by GIWE.
15-0 GIWEn	Global Interrupt Write Enable This field selects whether the upper 16-bit field of the Interrupt Control Register ICR(n) will be updated with the value in GIWD. 0b - Upper 16-bit of corresponding Interrupt Control Register is not updated with the value in GIWD. 1b - Upper 16-bit of corresponding Interrupt Control Register is updated with the value in GIWD.

18.7.1.20 Interrupt Status Flag Register (ISFR0 - ISFR1)

Offset

Register	Offset
ISFR0	120h
ISFR1	124h

Function

The Interrupt Status Flag Register (ISFR) indicates whether the related configured interrupt is detected on each pin. The pin interrupt configuration is valid in all digital pin muxing modes. The Interrupt Status Flag for each pin is also visible in the corresponding Interrupt Control Register, and each flag can be cleared in either location.

There is a separate Interrupt Status Flag Register for each interrupt or DMA request domain. Each status flag is only visible in the register that corresponds to that flag's domain as configured in IRCa[IRQS].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ISF31	ISF30	ISF29	ISF28	ISF27	ISF26	ISF25	ISF24	ISF23	ISF22	ISF21	ISF20	ISF19	ISF18	ISF17	ISF16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISF15	ISF14	ISF13	ISF12	ISF11	ISF10	ISF9	ISF8	ISF7	ISF6	ISF5	ISF4	ISF3	ISF2	ISF1	ISF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ISFn	Interrupt Status Flag Each bit in the field indicates the detection of the configured interrupt on each pin of the same number. 0b - Configured interrupt is not detected on the pin of the same number. 1b - Configured interrupt is detected on the pin of the same number. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.

Chapter 19
Clock and Power Overview

19.1 Overview

This chip targets many applications with power requirements that include low power consumption, long battery life, always-on capability, instant-on capability, and doesn't require active cooling. To meet these requirements, the chip design focuses on reducing current consumption, while simultaneously enabling the maximum level of peak performance, and also a balanced level of sustained performance. The chip architecture uses a wide range of power-management techniques, used in combination, for system design flexibility.

This chapter describes these architecture details as a high-level clock and power management overview of the chip. This overview chapter covers:

- Structural components of the power and clock management systems
- Clock generation and distribution system
- Power generation and distribution system
- Power mode definition
- Power and clock and management techniques supported.

NOTE

All numerical values in this chapter are either typical values or used as an example. See the data sheet for accurate values.

19.2 Components of Clock and Power Management

The centralized clock generation, power generation and distribution are implemented by the following blocks:

Table 127. Clock and Power Components

IP Module	Description
Crystal OSC (XTALOSC)	The XTALOSC includes a 24 MHz and 32 kHz oscillators. The 24MHz crystal oscillator is the primary clock source for all the PLLs, and clock generation for the CPU and high-speed interfaces. The 32KHz crystal oscillator is the primary clock source for the RTC as well as the low-speed clock source for CCM/SRC/GPC. See the Crystal Oscillator (XTALOSC) section of the ANADIG chapter for details.
PLLs and PFDs	The PLLs and their associated PFDs generate the clocks with various frequencies required to feed the CCM clock generator that supplies the different functional blocks. See the PLL and PFD section for information on the PLL and PFD architecture and functional description. The PLL registers are in the ANADIG chapter.
Clock Control Module (CCM)	The CCM module provides control for clock generation, division, distribution, synchronization, and coarse-level gating. See Clock Controller Module (CCM) for information on the CCM architecture, functional description and programming model.
Low Power Clock Gating (LPCG)	The LPCG distributes the clocks to all blocks in the SoC and handles automated clock gating and block level software-controllable clock gating. See

Table continues on the next page...

Table 127. Clock and Power Components (continued)

IP Module	Description
	Clock Controller Module (CCM) for information on the LPCG architecture and functional description.
General Power Controller (GPC)	This module controls the power state of the SoC. The GPC handles the power gating under low power modes and also manages the power up / power down sequences. See the General Power Controller (GPC) chapter for information on the GPC architecture, functional description and programming model.
Power Management Unit (PMU)	This module generates the internal power supplies distributed to the entire SoC. See the Power Management Unit (PMU) section of the ANADIG chapter for information on the PMU architecture, functional description, and programming model.
System Reset Controller (SRC)	This module generates the reset signals to all the modules in the SoC and controls power gating of power domains and memory low power mode. The SRC will appropriately assert reset signals for power transitions, entry, and exit. See the System Reset Controller (SRC) chapter for information on the SRC architecture, functional description and programming model.
DCDC Converter (DCDC)	This module generates the power supply for chip's core logic. See DCDC converter (DCDC) chapter for information on functional description, and programming model.
Trusted Resource Domain Controller (TRDC)	This module provides domain based resource control architecture for the memory/peripheral resource sharing and isolation between the Arm Cortex M33 core, Arm Cortex-M7 core and other bus masters. See the Trusted Resource Domain Controller (TRDC) chapter for more information.

Together, the modules listed above provide enhanced power-management features with the centralized control for the clock, reset, and power-management signals on the SoC.

19.3 Power Management

The power management design on the chip satisfies many use cases that depend on a key balance of performance and low-power consumption. In order to meet these criteria, the system power management needs to be dynamic and flexible. This is accomplished on a multi-core system by utilizing a power scheme that considers each CPU domain, and arbitrates the appropriate power mode configuration to satisfy the needs of both CPUs, and the system as a whole. The following are additional features supported by the power management architecture:

- Programmable power mode transition rules
- Independent operation mode for individual CPU cores
- Intelligent clock and power management for multi-core architecture
- Unified power management interface (UPI) for all on-chip resources, including clock, power, reset, and peripherals

A high-level view of the power management controllers and connections is shown below:

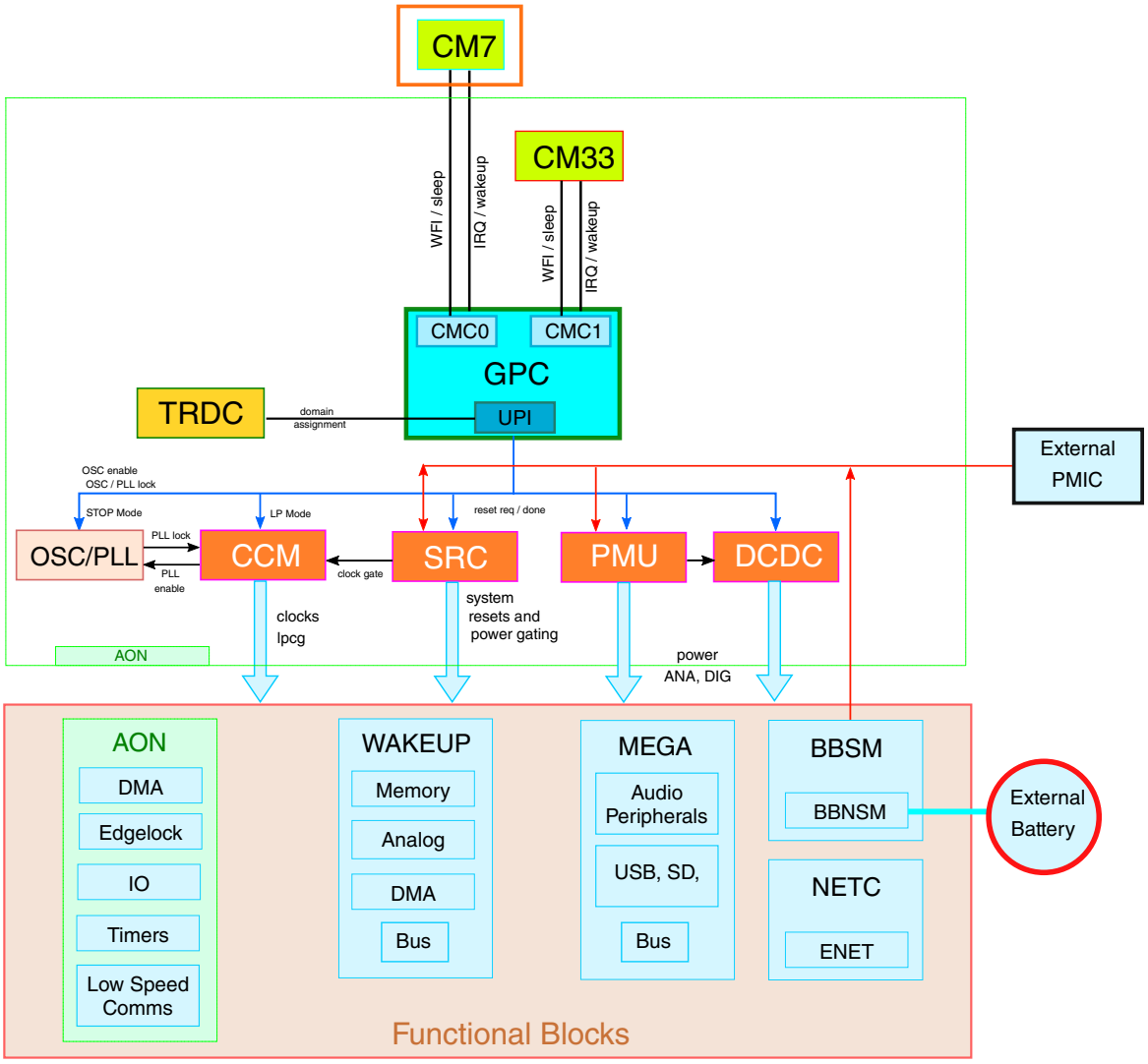


Figure 69. Power Management Block Diagram

19.3.1 Resources and Chip Domains

The chip is divided into resources and domains. Resources are controlled by resource controllers. Domains are grouped by central bus controllers and crossbars. Some domains reside within larger domains, which may have power dependencies. For illustrative representation of the power domains and bus configurations, see the Power Management Block Diagram. The Resource types and Chip Domains are listed in the tables below:

Table 128. Resources

Resource	Description
CPU Platforms	The processor and associated on-platform peripherals that support the complex. The CPU Platforms supported on this chip are the CM7 and the CM33 platforms.

Table continues on the next page...

Table 128. Resources (continued)

Resource	Description
System Controllers	The System Controllers are the associated peripherals that control or define the power management resources for the chip. These modules include GPC, PMU, CCM, SRC, and TRDC.
Functional blocks	The functional blocks are comprised of all the other peripherals, other than System Controllers. These modules include external memory controllers, timers, high-speed and low-speed peripherals, audio peripherals, connectivity, and others.

Table 129. Chip Domains

Domain	Description
CM7	The Cortex-M7 core and platform.
AON	The AON Domain contains the Cortex-M33 core and platform, which includes Tightly Coupled Memory (TCM). It also contains modules: SCTR, WDOG1-2, IOMUX_AON, IOMUXC, MU, MTR DCA_A, TSTMR1, GPIO1, SAI1, CAN1,3, LPI2C1-2, LPSPI1-2, LPUART1,2,7,12, I3C1, GPT1, LPIT1, TPM1-2, DCDC
WAKEUP	The WAKEUP Domain contains the memory controllers and analog on the chip. The specific modules in WAKEUP include: EWM, WDOG3-5, eDMA4, 2x DMAMUX (WAKEUP), ACMP, LPADC1-2, DAC, XBAR1-3, KPP, GPIO2-7, LPUART3,6,8-11, MTR DCA_W, FLEXCAN2 (CANFD), QDC1-8, LPIT2-3, AOI1-2, GPT2, TMR2-3, FlexPWM1-4, LPI2C3-4, I3C2, LPSPI3-4, FlexIO1-2, FlexSPI1, SEMC, IEE
NETC	Contains the NETC subsystem and 5x NETC Ethernet controllers.
MEGA	Contains the connectivity peripherals and audio peripherals. The specific modules include: EtherCAT, uSDHC1-2, 2xUSB, FlexSPI Slave, DMIC, SINC1-3, ASRC, SAI2-4, SPDIF
ANADIG	The ANA Domain contains: XTAL OSC and Control, PLL and Control, Temp Sensor and Control.
CCMSRCGPC	The CCMSRCGPC Domain contains: GPC, SRC, and CCM.
BBSM	The BBSM Domain contains BBNSM, which connects to a coin cell or non-interruptible power source. This domain contains control logic for security, startup, and RTC.

19.3.2 Power Management Unit (PMU)

The PMU has the following components integrated for power management:

- DCDC
  - Generates the core power supply
  - Supports dynamic voltage scaling
- LDOs
  - Internal logic power supply
- Power Switches for sophisticated power mode management

The typical power architecture of the chip is shown below. The use case illustrated shows the whole system working with a single 3.3V power supply and a coin cell. This is the power system configuration of traditional MCU. Depending on the application, other power supply schemes may be used, including bypassing the DCDC converter and using an external voltage regulator or PMIC.

NOTE

The on-chip DCDC converter is suitable for consumer and industrial applications up to the rated temperature specified in their respective market data sheets. Check the data sheet or consult with your NXP representative for appropriate power supply requirements.

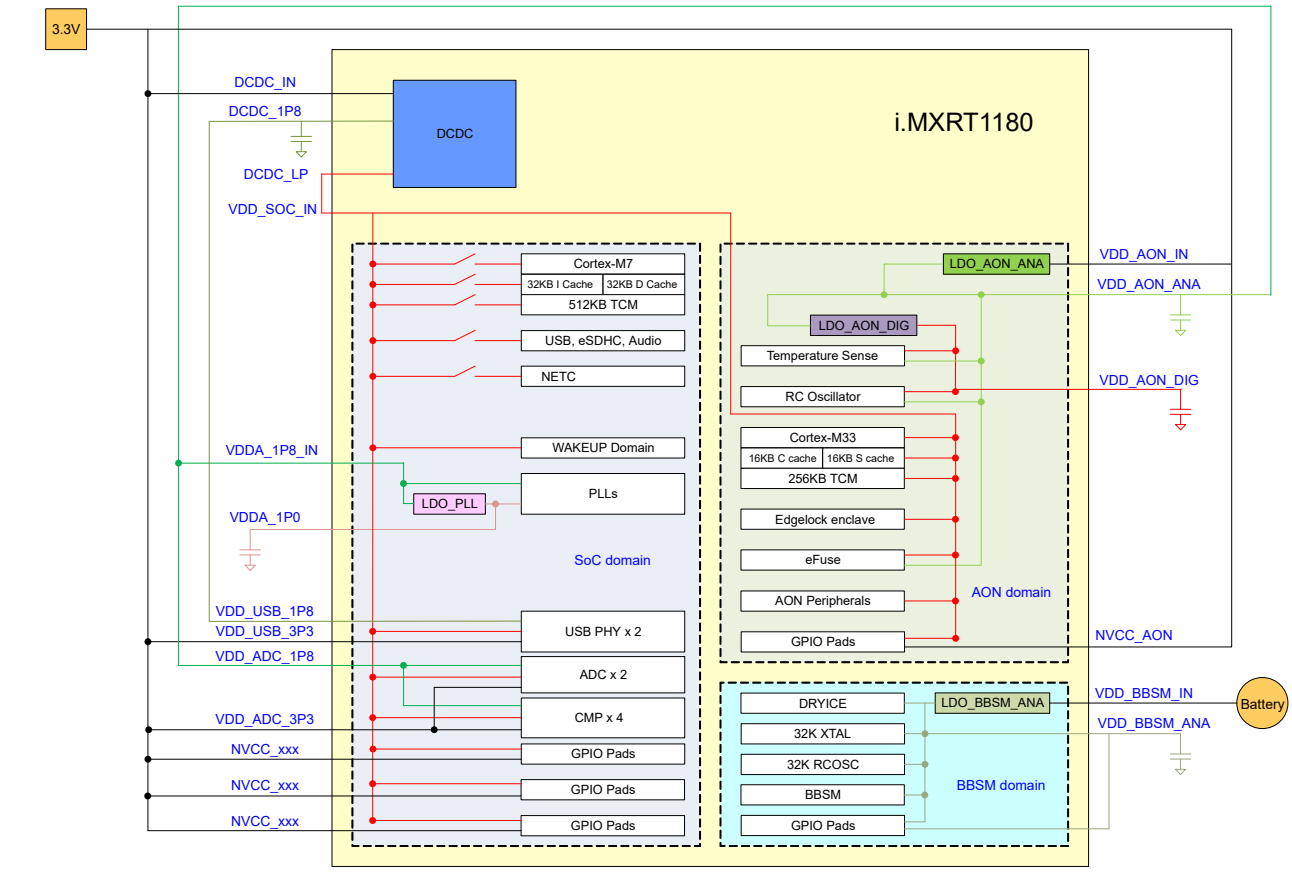


Figure 70. Power Architecture

19.3.3 Body Biasing (BB)

Forward Body Biasing (FBB) is implemented in the CM7 platform in order to achieve higher performance. See PMU for more information.

19.3.4 Power Modes

The chip resources abide by CPU or system-level power modes, depending on their resource assignment. Resources are assigned to one of four categories, which are defined below.

Table 130. Resources Categories

Category	Description
Private	A resource that is fully owned and controlled by one CPU platform. These resources can be controlled by hardware and software.

Table continues on the next page...

Table 130. Resources Categories (continued)

Category	Description
Shared	A resource that is shared by more than one CPU platform. These resources are limited by the other CPU power state. Software must take into regard shared resources, since they are limited by other CPU power states.
Public	A resource that is managed at the system level, which is not owned and controlled by any of the CPU platforms. These resources can only be controlled by hardware.
Unassigned	A resource that is not assigned as private, shared, or public. This is the default state of a resource after reset.

Resources can be assigned to different CPU domains by the TRDC. If one CPU wants to put the chip into a power mode, it must arbitrate with the other CPU to ensure the appropriate power state is chosen for shared resources. Depending on the resource type, varying power control mechanisms are followed.

Table 131. Power modes

Power Modes	CONDITION														Entry	Exit
	M7 (MHz)	M33 (MHz)	S401	AXI (MHz)	AXBS (MHz)	FBB	ECC	AONMIX	WKUPMIX	NETCMIX	MEGAMIX	BBSMMIX	DCDC_1P0	ANALOG		
HSRUN (High Speed Run mode)	800	240	200	240	133	ON	ON	ON	ON	ON	ON	ON	1.1V	FP	SW	Interrupt Reset
RUN (Normal Speed Run mode)	600	240	200	240	133	OFF	ON	ON	ON	ON	ON	ON	1.0V	FP	N/A	Interrupt Reset
LSRUN (Low Speed Run mode)	480	160	100	160	133	OFF	ON	ON	ON	ON	ON	ON	1.0V	FP	SW	Interrupt Reset

Table continues on the next page...

Table 131. Power modes (continued)

Power Modes	CONDITION														Entry	Exit
	M7 (MHz)	M33 (MHz)	S401	AXI (MHz)	AXBS (MHz)	FBB	ECC	AONMIX	WKUPMIX	NETCMIX	MEGAMIX	BBSMMIX	DCDC_1P0	ANALOG		
LVRUN ((Low Voltage Run mode))	360	100	66	100	50	OFF	ON	ON	ON	PG	PG	ON	0.9V	FP	SW	Interrupt Reset
Normal WAIT	CG	CG		240	133	OFF	ON	ON	ON	ON	ON	ON	1.0V	FP	WFI from Normal/ LS RUN	Interrupt Reset
Normal STOP STBY	CG	CG		CG	CG	OFF	ON	CG	CG	CG	CG	ON	1.0V	LP	WFI from Normal/ LS RUN	Interrupt Reset
Normal SUSPEND STBY	PG	PG		CG	CG	OFF	ON	CG	CG	CG	CG	ON	1.0V	LP	WFI from Normal/ LS RUN	Interrupt Reset
LV WAIT	CG	CG		100	66	OFF	ON	ON	ON	ON	ON	ON	0.9V	FP	WFI from LV RUN	Interrupt Reset
LV STOP STBY	CG	CG		CG	CG	OFF	ON	CG	CG	PG	CG	ON	0.9V	LP	WFI from LV RUN	Interrupt Reset
LV SUSPEN	PG	PG		CG	CG	OFF	ON	CG	CG	PG	PG	ON	0.9V	LP	WFI from	Interrupt

Table continues on the next page...

Table 131. Power modes (continued)

Power Modes	CONDITION														Entry	Exit
	M7 (MHz)	M33 (MHz)	S401	AXI (MHz)	AXBS (MHz)	FBB	ECC	AONMIX	WKUPMIX	NETCMIX	MEGAMIX	BBSMMIX	DCDC_1P0	ANALOG		
DSTBY															LVRUN	Reset
BOM(Battery Only Mode)	PG	PG		PG	PG	OFF	OFF	PG	PG	PG	PG	PG	OFF	OFF	SW	WAKE ONOFF

19.3.4.1 CPU Mode

The CPU platforms both support the same power states that are defined as CPU modes. The CPU Modes are detailed below.

Table 132. CPU Mode

CPU Mode	Description
RUN	The CPU is active and running.
WAIT	The CPU is in the WFI state. The CPU core and L1 cache are clock gated, TCM and peripheral can still be alive. The chip can wakeup from this mode from IRQ with a very short latency.
STOP	The CPU is in the WFI state. The CPU core, L1 cache, TCM, and its private peripherals are all clock gated. The chip requires a longer time to wakeup from this mode.
SUSPEND	The CM7 core and L1 cache are power gated. The TCM is in retention mode. SUSPEND mode consumes the least amount of power, while keeping parts of the system alive. But the chip requires a much longer time to wakeup from this mode. The CM33 and CM33 cache is not power gated.

For the 3 types of resources mentioned above, the CPU platform is naturally a private resource, the system resources and functional blocks can be configured as private, public or shared. For example:

- A UART module can be a private resource
- The clock root dedicated for a UART can also be a private resource
- The DCDC which provides power to the entire chip is a public resource
- The AXI bus which provides connection for both CPU platforms is a shared resource

For private resources, they can be controlled by SW, or controlled by HW (based on CPU state). For example:

- A clock for UART can be gated OFF when CPU is in STOP mode
- Audio PLL reserved for one CPU core can be configured by the CPU at any time

For public resources, they need to be controlled by HW through GPC. For example:

- DCDC voltage goes to 0.8V when chip goes to SYSTEM SLEEP mode

- LDO goes to STANDBY mode when chip goes to SYSTEM SLEEP mode

For shared resources, they can be controlled by SW, or controlled by HW (based on CPU state). For example:

- A clock for shared AXI bus can be gated OFF when both CPUs disable this clock
- A clock for shared AXI bus can also be gated OFF when both CPUs are in STOP mode

NOTE

AXI bus clock can NOT be gated off by application.

For unassigned resources, they are expected to be assigned or disabled by SW during chip initialization phase.

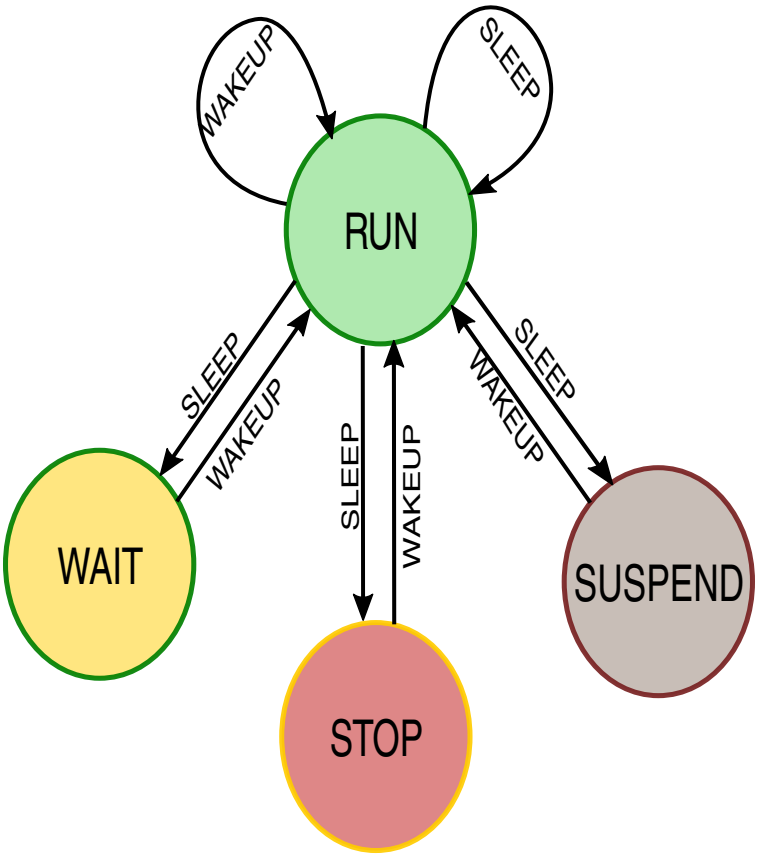


Figure 71. CPU Modes

19.3.4.1.1 Memory Power Connection

A few of the power connections that exist in this chip are shown below:

- M7 Cache and TCM memory:
  - M7 core and the memory interface to the SoC are on the same power domain, which are on/off together. For example, "M7 can be OFF while the memories are ON" means the memory array is powered on, while M7 and the memory interface to the SoC are powered off
  - Supported modes:
    - M7 on, cache and TCM on
    - M7 off, cache and TCM on
    - M7 off, cache off, TCM off

- No access to TCM memory when M7 core is power gated

- M33 Cache & TCM memory:
  - M33 Cache and TCM is in AON Domain, supplied from VDD_SOC_IN, no power gate.
- OCRAM memories:
  - OCRAM is in WAKEUP Domain, supplied from VDD_SOC_IN, no power gate
- RAM in AON Domain:
  - All RAM in AON Domain are supplied by VDD_SOC_IN, no power gate.
- RAM in WAKEUP Domain:
  - All RAM in WAKEUP Domain are supplied by VDD_SOC_IN, no power gate.
- RAM in MEGA Domain:
  - RAM in the MEGA Domain share same power supply as logic cells in the MEGA Domain, when MEGA Domain is power off, RAM in the MEGA Domain are powered off.

19.3.4.2 System Sleep Mode

System sleep mode is a low-power mode that has distinguishing settings outside of CPU mode. When system is in SLEEP mode:

- Analog modules like LDO/BG can be put into their own STANDBY mode
- DCDC can be decreased into lower voltage to reduce leakage
- DCDC module clock can be turned off

System sleep mode is related to the state of all CPU platforms. The system sleep controller in GPC maintains the system sleep sequence. Only when all CPU platforms send system sleep requests, the system enter into system sleep mode. Any interrupt can abort system sleep sequence so that system can be quickly woken up.

19.3.5 Power Sequence

Power Up:

- VDD_BBSM_IN supply must be turned on before all other power supplies, or be connected (shorted) with VDD_AON_IN and DCDC_IN supply
- If a coin cell is used to power VDD_BBSM_IN, then ensure that it is connected before all other supplies are switched on
- When internal DCDC is enabled, external delay circuit is required to delay the “DCDC_PSWITCH” signal 1ms after DCDC_IN is stable
- POR_B should be held low during the entire power up sequence

Power Down:

- VDD_BBSM_IN supply must be turned off after all other power supplies, or be connected (shorted) with VDD_AON_IN and DCDC_IN supply
- If a coin cell is used to power VDD_BBSM_IN, then ensure that it is removed after all other supplies are switched off

19.4 Low-Power Control Architecture

There are 3 types of controllers used in the system for power management:

- **General Power Controller (GPC)** - GPC is the centralized power controller, which controls the power mode of the entire processor. GPC takes the WFI signal from CPU platforms and wakeup events from peripherals, determine the power mode based on power management policy, and controls the power mode transition. GPC does not control the state of the resource directly. During power mode transition, it communicates with the resource controllers through the UPI (Unified Power management Interface) to accomplish the power mode transition.

- **Trusted Resource Domain Controller (TRDC)** - TRDC is used for system resource protection and sharing in multi-core system. The resources of the entire chip can be assigned to different CPU core domains so that resources that belong to one CPU core are not affected by the other core. In power management system, the TRDC delivers the domain assignment information to GPC and other components to support multi-core-aware clock and power management. It does not control the resources directly.
- **System Resource Controllers (SRC)** - System resource controllers are not a single block. It is a set of controllers that manages the system resources such as clock, power and reset. It gets the power mode transition information from UPI interface and get the resource ownership information from TRDC, then change the state of the system resource based on pre-defined policy.

The following blocks are the typical system resource controllers:

- CCM (Clock Controller Module) - Generates the clocks for the entire chip
- SRC (System Reset Controller) - Generates resets for the entire chip
- PMU (Power Management Unit) - Controls the on-chip LDO/DCDC modules
- Other resource controllers

NOTE

Power gating to control ON/OFF of various power domains, as well as memory low-power control for sleep/retention of memories, are performed by the SRC.

19.5 Clock Generation

The clock system is designed to provide a scalable, powerful, and easy-to-use clock solution.

19.5.1 Clock System Diagram

The overall clock system for the chip is shown in the diagram below.

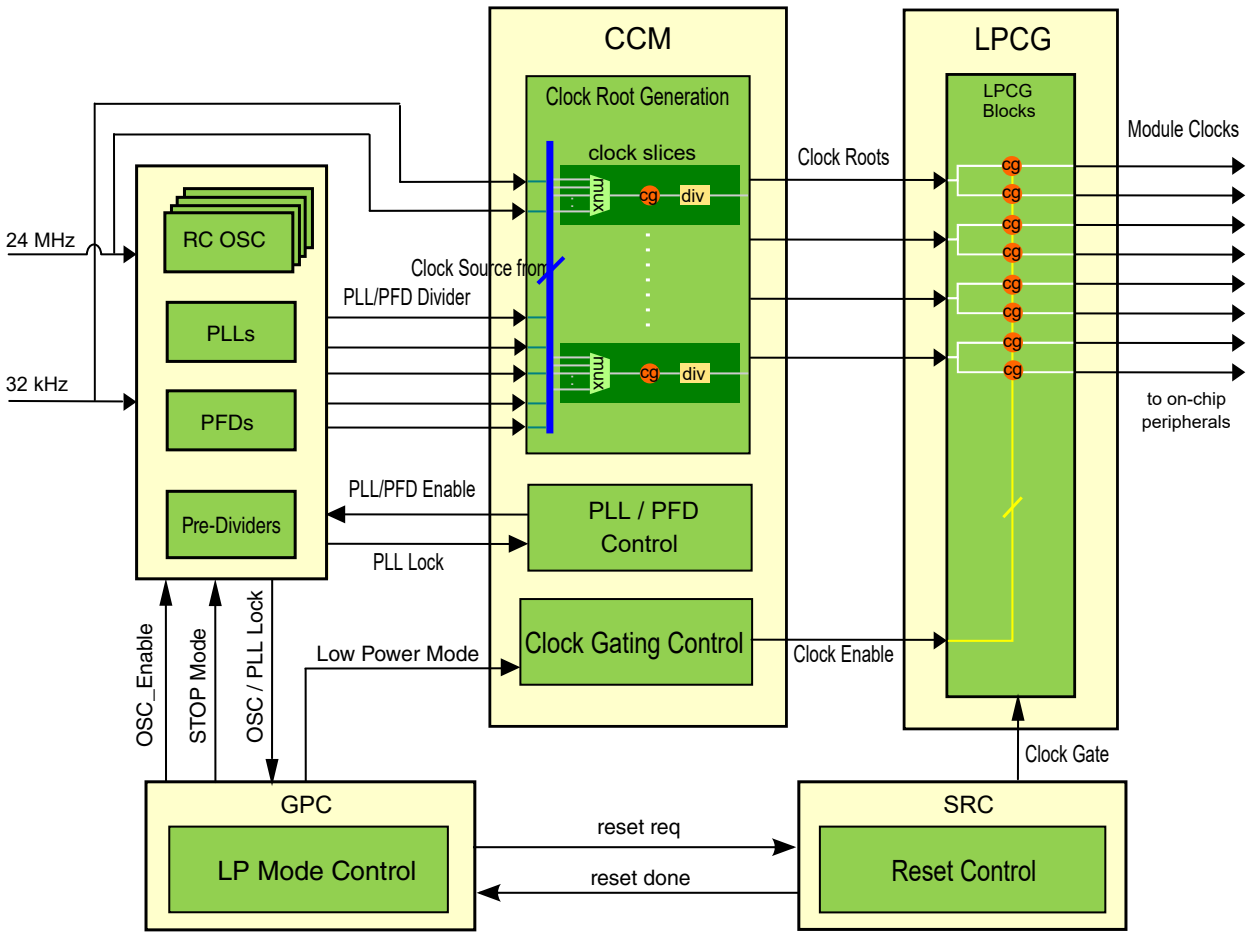


Figure 72. Clock System

The OSCs, PLLs, PFDs and Pre-Dividers generate the clock sources with fixed or variable frequency. The clock sources feed the clock root generation logic inside CCM, which generates numerous clock roots required for core, bus, and peripheral blocks. These clock roots are branched to each module through the LPCG logic, which contains the clock gating cells for each clock. Control signals for the LPCGs are in CCM (clock enable signal). Some of the clocks need to be gated off during reset, because of this, the SRC module will also send clock gate signals to the LPCG.

In low power mode, GPC will drive the low power mode state signal to CCM, then CCM will enable clock gating based on the configuration. At the same time, CCM may also deassert enable signals for PLLs and PFDs, so these modules can power off during low power modes. When the chip enters STOP mode, CCM may deassert the OSC enable signal to shut off the OSC, it may also assert the STOP mode signal to set OSC to low power mode.

The detailed functional description for each block will be described in the following sections.

19.5.2 Oscillator

This chip has two crystal oscillators and three RC oscillators:

Table 133. Oscillators

Oscillator	Description
24 MHz Crystal Oscillator (OSC_24M)	The primary clock source for all PLLs to generate the clocks for CPU, bus, and high-speed interfaces
32 kHz Crystal Oscillator (OSC_32K)	The primary clock source for RTC as well as low-power clock source for CCM, SRC, and GPC. The typical frequency is 32.768KHz.
24 MHz RC Oscillator (OSC_RC_24M)	Generates the 24 MHz clock source for the chip during startup, before the 24MHz crystal oscillator is ready. The OSC_RC_24M can also be used as an alternative 24 MHz clock source in low power mode, when the crystal oscillator is turned off, or in a system that doesn't have a crystal oscillator.
32 kHz RC Oscillator (OSC_RC_32K)	The 32 kHz clock source for the chip during startup when 32 kHz crystal is not ready. It can also be used as alternative 32 kHz clock source when the external 32 kHz crystal oscillator is not stable, or in system which does not have any crystal oscillator. The switch from external 32 kHz to internal 32 kHz RC is controlled by hardware, and it happens automatically when the system detects a loss of the 32 kHz crystal clock.
400 MHz RC Oscillator (OSC_RC_400M)	The clock source during chip boot up, before PLLs are enabled, or in low-power mode when the PLLs are turned off.

See Crystal Oscillator (XTALOSC) chapter for functional description details and the programming model on these blocks.

19.5.3 PLL and PFD

The PLLs are used in the chip to generate the high speed clocks required by the modules in SoC. A few PLLs are equipped with Phase Fractional Dividers (PFDs) in order to generate additional frequencies. The following PLLs are used in this chip:

Table 134. PLLs

PLL	Description	Spread Spectrum	Configurable
ARM_PLL	The ARM_PLL is used to generate the clock for the Arm Cortex-M7 platform. It is a programmable integer-frequency multiplier PLL, capable of an output frequency over 2 GHz.	-	Yes
SYS_PLL1	The SYS_PLL1 is used primarily by the ENET module. It has a dedicated frequency of 1GHz.	Yes	1 GHz
SYS_PLL2	The SYS_PLL2 main output clock has a dedicated frequency of 528MHz, but has several configurable PFDs.	Yes	528 MHz
SYS_PLL3	The SYS_PLL3 main output clock has a dedicated frequency of 480MHz, but has several configurable PFDs.	-	480 MHz
AUDIO_PLL	The AUDIO_PLL used to generate reference clocks, mainly for serial audio interfaces, and external audio codecs. This fractional multiplier PLL provides a low-jitter and high precision audio clock with standardized audio frequencies. The PLL's oscillator frequency range is from 650MHz to 1300MHz, and the frequency resolution is better than 1Hz.	Yes	Yes

NOTE

See the datasheet for all supported maximum PLL frequencies

Each PLL can use one of the following clock sources as its reference clock:

- Primary Option: 24 MHz clock from 24 MHz oscillator (XTAL)
- Secondary Option: 24 MHz clock from 24 MHz RCOSC

The main output clocks for the SYS_PLL2 and SYS_PLL3 (PLLx_CLK) are not configurable, but their PFDs are. Each PFD works independently by interpolating the VCO of the PLL to which it is connected. It effectively takes the PLL VCO frequency and produces 18/N x Fvco at its output where N ranges from 13 to 35. PFD is a completely digital design with no analog components or feedback loops. The frequency switch time is much faster than a PLL because keeping the base PLL locked and changing the integer N only changes the logical combination of the interpolated outputs of the VCO. Note that the PFD not only enables faster frequency changes than a PLL, but also allows the configuration to be safely changed "on-the-fly" without going through the output clock disabling/enabling process.

The basic logical connections for the PLLs, PFDs and Dividers is shown in the diagram below:

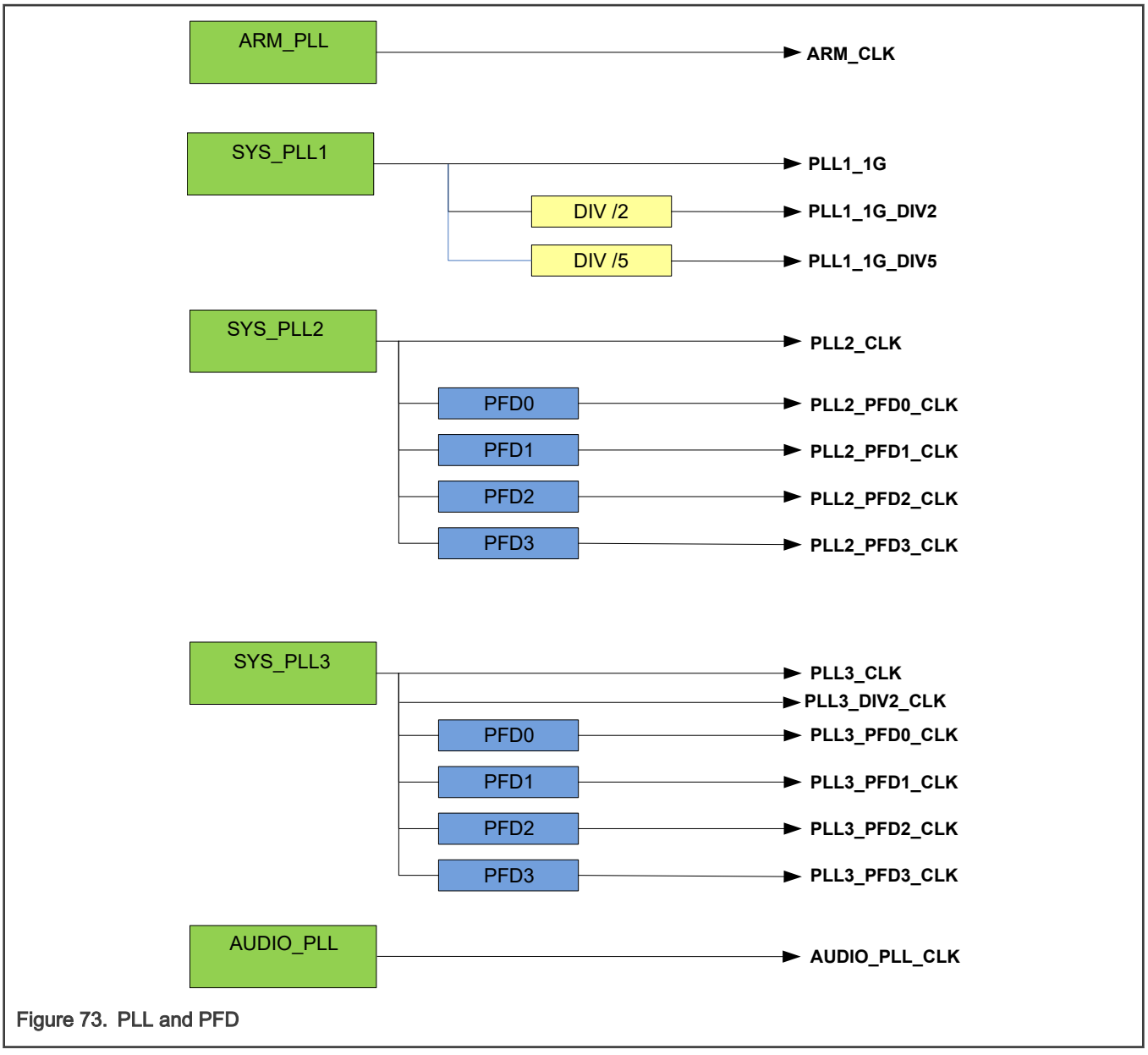


Figure 73. PLL and PFD

PLL configuration and control functions are accessible via configuration and status registers. The ANADIG block contains the detailed descriptions of the memory mapped registers, and control functions of the clock generation sub-module. See ANADIG for more information.

19.5.4 Clock Root Generation

Clock generation in the CCM creates clock roots for on-chip peripherals. Each clock root will have a dedicated clock slice. It takes the clock source from Oscillators, PLL/PFD or Pre-Dividers, and generate the clock root with required frequency.

The CCM can manage on-chip peripheral clocks by controlling the low power clock gating cells (LPCGs) before the clock roots enter the peripherals.

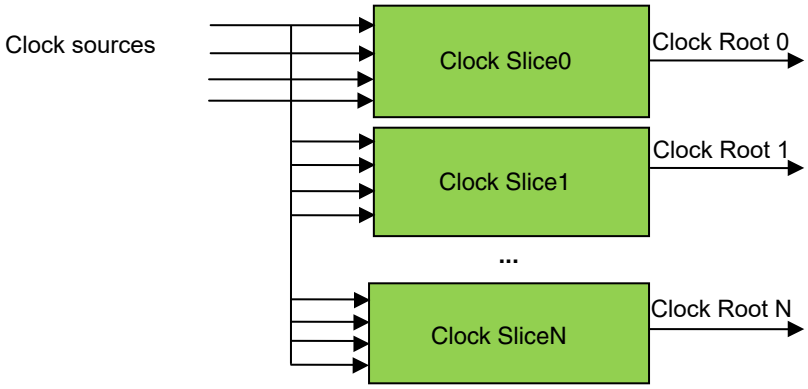


Figure 74. Clock Root Generation from Clock Slice

The LPCGs are implemented to control peripheral functions, and there could be several LPCGs for each function in a peripheral.

For details on the programming model, refer to CCM Memory Map/Register Definition.

19.5.5 Low Power Clock Gating (LPCG)

The clock roots generated inside CCM are distributed to the entire SoC through LPCG. The LPCG block receives the root clocks from CCM and splits them to clock branches for each functional block. The clock branches are individually gated clocks.

See the CCM chapter for more information about clock gating.

Chapter 20
Clock Controller Module (CCM)

20.1 Chip-specific CCM information

Table 135. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

The memory map and register definition for PLLs exist in the ANADIG chapter. See the PLL Memory Map section in the ANADIG chapter for more information.

20.2 Overview

The Clock Control Module (CCM) manages the on-chip module clocks. The oscillators, PLLs, and Phase Fractional Dividers (PFD) will generate clock sources with fixed or variable frequencies.

20.2.1 Block diagram

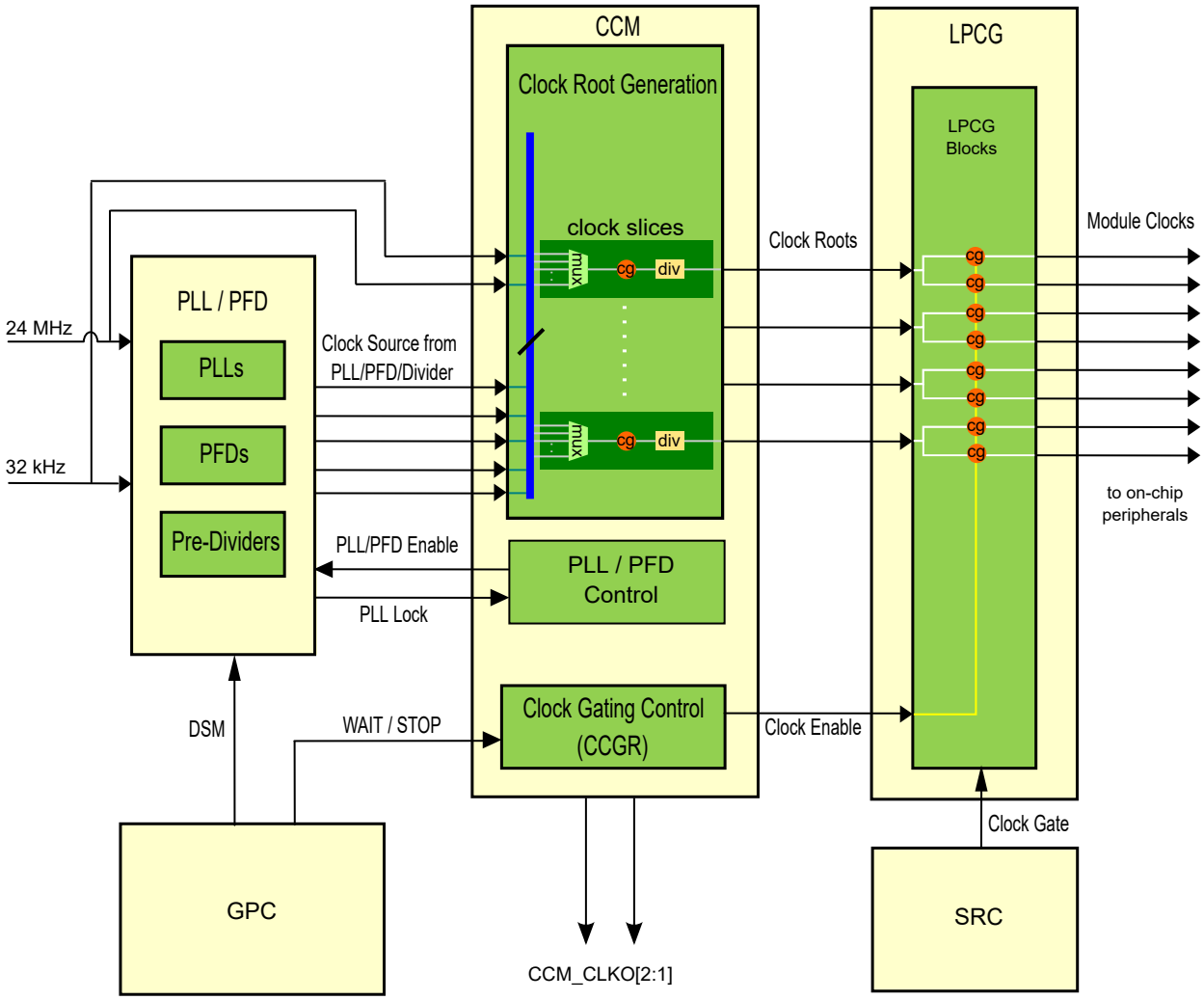


Figure 75. CCM block diagram

The clock root generation logic inside CCM will generate various clock roots required for core, bus, and peripheral blocks. These clock roots will be delivered to each module through Low Power Clock Gate (LPCG), which contains the clock gating logic for each clock.

The control signal for CCGRs in CCM will be the source for clock enable signals. Since some of the clocks need to be gated off during the reset, the System Reset Controller (SRC) will also send clock gate signals to the LPCG.

In low power mode, the General Power Controller(GPC) will drive the low power mode state signal to CCM, and CCM will enable clock gating based on the configuration. At the same time, CCM may also de-assert enable signals for PLLs, so these modules can be powered off during the low power mode.

20.2.2 Features

The CCM includes the following features:

- Clock Root
  - Up to 4 clock sources for each clock root

- 8-bit divider support up to divide by 256 in each clock root
- Each clock root can be shutdown by software

- Peripheral clock can be auto gated according to CPU work mode, software configurations, or system work mode
- PLLs and oscillators can be auto shutdown according to CPU work mode, software configurations, or system work mode
- Access Control
  - Two access control schemes: TrustZone, and Domain
  - Write access to each component can be protected independently
  - Independent setting for user, privileged, secure, non-secure
  - Domain mode access can be controlled by whitelist
  - Access control setting can be locked independently

20.3 System Clocks

The table found here shows the CCM output clocks' system-level connectivity.

Table 136. System Clocks Table

Module	Clock	Clock Root	LPCG control
ACMPn	cmp_fast_clk	acmp_clk_root	LPCG15
	cmp_slow_clkf	OSC_32K	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG15
	cmp_fast_clk	acmp_clk_root	LPCG16
	cmp_slow_clk	OSC_32K	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG16
	cmp_fast_clk	acmp_clk_root	LPCG17
	cmp_slow_clk	OSC_32K	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG17
	cmp_fast_clk	acmp_clk_root	LPCG18
	cmp_slow_clk	OSC_32K	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG18
ADCn	adc_clk	adc1_clk_root	LPCG12
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG12
	adc_clk	adc2_clk_root	LPCG13
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG13
AIPSn	hclk	bus_aon_clk_root	LPCG3
	ipg_clk	bus_aon_clk_root	LPCG3
	hclk	bus_wakeup_clk_root	LPCG4
	ipg_clk	bus_wakeup_clk_root	LPCG4

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
	hclk	bus_wakeup_clk_root	LPCG4
	ipg_clk	bus_wakeup_clk_root	LPCG4
AOI1	ipg_clk	bus_wakeup_clk_root	LPCG114
AOI2	ipg_clk	bus_wakeup_clk_root	LPCG115
AOI3	ipg_clk	bus_wakeup_clk_root	LPCG116
AOI4	ipg_clk	bus_wakeup_clk_root	LPCG117
ASRC	ipg_clk	asrc_clk_root	LPCG141
CANn	ipg_clk_chi	bus_aon_clk_root	NA
	ipg_clk_pe	Selected by CAN1_CTRL1[CLKSRC] 0: can1_clk_root 1: bus_aon_clk_root	LPCG76
	ipg_clk_pe_nogate	Selected by CAN1_CTRL1[CLKSRC] 0: can1_clk_root 1: bus_aon_clk_root	LPCG76
	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG76
	osc_clk	OSC_24M	LPCG76
	ipg_clk_chi	bus_wakeup_clk_root	NA
	ipg_clk_pe	Selected by CAN2_CTRL1[CLKSRC] 0: can2_clk_root 1: bus_wakeup_clk_root	LPCG77
	ipg_clk_pe_nogate	Selected by CAN2_CTRL1[CLKSRC] 0: can2_clk_root 1: bus_wakeup_clk_root	LPCG77
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG77
	ipt_clk	bus_wakeup_clk_root	NA
	osc_clk	OSC_24M	LPCG77
	ipg_clk_chi	bus_aon_clk_root	NA
	ipg_clk_pe	Selected by CAN3_CTRL1[CLKSRC] 0: can3_clk_root	LPCG78

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
		1: bus_aon_clk_root	
	ipg_clk_pe_nogate	Selected by CAN3_CTRL1[CLKSRC] 0: can3_clk_root 1: bus_aon_clk_root	LPCG78
	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG78
	osc_clk	OSC_24M	LPCG78
CM33	ATCLK	m33_clk_root	NA
	FCLK	m33_clk_root	NA
	FCLK_LP	m33_clk_root	LPCG3
	PLAT_HCLK	m33_clk_root	NA
	ipg_clk_perfmon	m33_clk_root	NA
	ipg_clk_trdc	m33_clk_root	NA
	clk	m33_clk_root	LPCG3
	clk	m33_clk_root	LPCG3
DAC	ana8M_clk	OSC_24M	LPCG14
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG14
DCDC	ipg_clk	OSC_RC_24M	LPCG8
ECAT	CLK100	ecat_clk_root	LPCG127
	CLK25	ecat_clk_root	LPCG127
	CLK50	ecat_clk_root	LPCG127
	ipg_clk	ecat_clk_root	LPCG127
eDMAn	hclk	m33_clk_root	LPCG29
	ipg_clk	m33_clk_root	LPCG29
	axi_clk	wakeup_axi_clk_root	LPCG30
	ipg_clk	wakeup_axi_clk_root	LPCG30
EWM	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG24
	lpo_clk[0]	OSC_RC_24M	LPCG24
	lpo_clk[1]	clk_1m_out	LPCG24
	lpo_clk[2]	OSC_32K	NA
	lpo_clk[3]	bus_wakeup_clk_root	NA
FlexIOn	flexio_clk	flexio1_clk_root	LPCG51

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
	pclk	bus_wakeup_clk_root	LPCG51
	flexio_clk	flexio2_clk_root	LPCG52
	pclk	bus_wakeup_clk_root	LPCG52
FlexPWM1	ipg_clk_0	bus_wakeup_clk_root	LPCG123
FlexPWM1	ipg_clk_1	bus_wakeup_clk_root	LPCG123
FlexPWM1	ipg_clk_2	bus_wakeup_clk_root	LPCG123
FlexPWM1	ipg_clk_3	bus_wakeup_clk_root	LPCG123
FlexPWM1	ipg_clk_cfg	bus_wakeup_clk_root	LPCG123
FlexPWM1	ipg_clk_flt	bus_wakeup_clk_root	LPCG123
FlexPWM2	ipg_clk_0	bus_wakeup_clk_root	LPCG124
FlexPWM2	ipg_clk_1	bus_wakeup_clk_root	LPCG124
FlexPWM2	ipg_clk_2	bus_wakeup_clk_root	LPCG124
FlexPWM2	ipg_clk_3	bus_wakeup_clk_root	LPCG124
FlexPWM2	ipg_clk_cfg	bus_wakeup_clk_root	LPCG124
FlexPWM2	ipg_clk_flt	bus_wakeup_clk_root	LPCG124
FlexPWM3	ipg_clk_0	bus_wakeup_clk_root	LPCG125
FlexPWM3	ipg_clk_1	bus_wakeup_clk_root	LPCG125
FlexPWM3	ipg_clk_2	bus_wakeup_clk_root	LPCG125
FlexPWM3	ipg_clk_3	bus_wakeup_clk_root	LPCG125
FlexPWM3	ipg_clk_cfg	bus_wakeup_clk_root	LPCG125
FlexPWM3	ipg_clk_flt	bus_wakeup_clk_root	LPCG125
FlexPWM4	ipg_clk_0	bus_wakeup_clk_root	LPCG126
FlexPWM4	ipg_clk_1	bus_wakeup_clk_root	LPCG126
FlexPWM4	ipg_clk_2	bus_wakeup_clk_root	LPCG126
FlexPWM4	ipg_clk_3	bus_wakeup_clk_root	LPCG126
FlexPWM4	ipg_clk_cfg	bus_wakeup_clk_root	LPCG126
FlexPWM4	ipg_clk_flt	bus_wakeup_clk_root	LPCG126
FlexSPIn	hclk	wakeup_axi_clk_root	LPCG34
	ipg_clk, ipg_clk_s	wakeup_axi_clk_root	LPCG34
	ipg_clk_sfck	flexspi1_clk_root	LPCG34
	hclk	m33_clk_root	LPCG35
	ipg_clk, ipg_clk_s	m33_clk_root	LPCG35

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
	ipg_clk_sfck	flexspi2_clk_root	LPCG35
FlexSPI_FLR	bus_clk	wakeup_axi_clk_root	LPCG36
	ipg_clk	bus_wakeup_clk_root	LPCG36
	root_clk	flexspi_flr_clk_root	LPCG36
GPIOn	hclk	m33_clk_root	LPCG45
	hclk	bus_wakeup_clk_root	LPCG46
	hclk	bus_wakeup_clk_root	LPCG47
	hclk	bus_wakeup_clk_root	LPCG48
	hclk	bus_wakeup_clk_root	LPCG49
	hclk	bus_wakeup_clk_root	LPCG50
GPTn	ipg_clk_24m	OSC_24M	LPCG73
	ipg_clk_32k	OSC_32K	NA
	ipg_clk_highfreq	clk_1m_out	LPCG73
	ipg_clk, ipg_clk_s	gpt1_clk_root	LPCG73
	ipg_clk_24m	OSC_24M	LPCG74
	ipg_clk_32k	OSC_32K	NA
	ipg_clk_highfreq	clk_1m_out	LPCG74
	ipg_clk, ipg_clk_s	gpt2_clk_root	LPCG74
I3Cn	CLK_SLOW	OSC_24M	LPCG103
	CLK_SLOW_TC	OSC_24M	NA
	FCLK	i3c1_clk_root	LPCG103
	PCLK	bus_aon_clk_root	LPCG103
	CLK_SLOW	OSC_24M	LPCG104
	CLK_SLOW_TC	OSC_24M	NA
	FCLK	i3c2_clk_root	LPCG104
	PCLK	bus_wakeup_clk_root	LPCG104
IEE	axi_clk	wakeup_axi_clk_root	NA
	clk	wakeup_axi_clk_root	LPCG40
IOMUXC	ipg_clk_s	bus_wakeup_clk_root	LPCG44 (IOMUXC2)
	ipg_clk_s	bus_aon_clk_root	LPCG43 (IOMUXC1)
KPP	ipg_clk_32k	OSC_32K	NA
	ipg_clk_s	bus_wakeup_clk_root	LPCG122

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
LPI2Cn	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG91
	lpi2c_clk	lpi2c0102_clk_root	LPCG91
	lpi2c_div_clk	lpi2c0102_clk_root	NA
	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG92
	lpi2c_clk	lpi2c0102_clk_root	LPCG92
	lpi2c_div_clk	lpi2c0102_clk_root	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG93
	ipt_clk	bus_wakeup_clk_root	NA
	lpi2c_clk	lpi2c0304_clk_root	LPCG93
	lpi2c_div_clk	lpi2c0304_clk_root	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG94
	lpi2c_clk	lpi2c0304_clk_root	LPCG94
	lpi2c_div_clk	lpi2c0304_clk_root	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG95
	lpi2c_clk	lpi2c0506_clk_root	LPCG95
	lpi2c_div_clk	lpi2c0506_clk_root	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG96
	lpi2c_clk	lpi2c0506_clk_root	LPCG96
	lpi2c_div_clk	lpi2c0506_clk_root	NA
LPITn	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG53 (LPIT1)
	ipg_per_clk	bus_aon_clk_root	NA
	ipg_ungated_per_clk	bus_aon_clk_root	LPCG53 (LPIT1)
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG54 (LPIT2)
	ipg_ungated_per_clk	bus_wakeup_clk_root	LPCG54 (LPIT2)
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG55 (LPIT3)
	ipg_per_clk	lpit3_clk_root	LPCG55 (LPIT3)
	ipg_ungated_per_clk	bus_wakeup_clk_root	LPCG55 (LPIT3)
LPSPIn	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG97
	lpspi_clk	lpspi0102_clk_root	LPCG97
	lpspi_div_clk	lpspi0102_clk_root	NA
	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG98
	lpspi_clk	lpspi0102_clk_root	LPCG98

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
	lpspi_div_clk	lpspi0102_clk_root	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG99
	lpspi_clk	lpspi0304_clk_root	LPCG99
	lpspi_div_clk	lpspi0304_clk_root	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG100
	lpspi_clk	lpspi0304_clk_root	LPCG100
	lpspi_div_clk	lpspi0304_clk_root	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG101
	lpspi_clk	lpspi0506_clk_root	LPCG101
	lpspi_div_clk	lpspi0506_clk_root	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG102
	lpspi_clk	lpspi0506_clk_root	LPCG102
	lpspi_div_clk	lpspi0506_clk_root	NA
LPTMRn	ipg_clk_32khz	OSC_32K	NA
	ipg_clk_irclk	lptmr1_clk_root	LPCG56
	ipg_clk_s	bus_aon_clk_root	LPCG56
	ipg_clk_32khz	OSC_32K	NA
	ipg_clk_irclk	lptmr2_clk_root	LPCG57
	ipg_clk_s	bus_wakeup_clk_root	LPCG57
	ipg_clk_32khz	OSC_32K	NA
	ipg_clk_irclk	lptmr3_clk_root	LPCG58
	ipg_clk_s	bus_wakeup_clk_root	LPCG58
LPUARTn	lpuart_clk_n,lpuart_clk	lpuart0102_clk_root	LPCG79
	pclk	bus_aon_clk_root	LPCG79
	lpuart_clk_n,lpuart_clk	lpuart0910_clk_root	LPCG88
	pclk	bus_wakeup_clk_root	LPCG88
	lpuart_clk_n,lpuart_clk	lpuart1112_clk_root	LPCG89
	pclk	bus_wakeup_clk_root	LPCG89
	lpuart_clk_n,lpuart_clk	lpuart1112_clk_root	LPCG90
	pclk	bus_aon_clk_root	LPCG90
	lpuart_clk_n,lpuart_clk	lpuart0102_clk_root	LPCG80
	pclk	bus_aon_clk_root	LPCG80

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
	lpuart_clk_n,lpuart_clk	lpuart0304_clk_root	LPCG81
	pclk	bus_wakeup_clk_root	LPCG81
	lpuart_clk_n,lpuart_clk	lpuart0304_clk_root	LPCG82
	pclk	bus_wakeup_clk_root	LPCG82
	lpuart_clk_n,lpuart_clk	lpuart0506_clk_root	LPCG83
	pclk	bus_wakeup_clk_root	LPCG83
	lpuart_clk_n,lpuart_clk	lpuart0506_clk_root	LPCG84
	pclk	bus_wakeup_clk_root	LPCG84
	lpuart_clk_n,lpuart_clk	lpuart0708_clk_root	LPCG85
	pclk	bus_aon_clk_root	LPCG85
	lpuart_clk_n,lpuart_clk	lpuart0708_clk_root	LPCG86
	pclk	bus_wakeup_clk_root	LPCG86
	lpuart_clk_n,lpuart_clk	lpuart0910_clk_root	LPCG87
	pclk	bus_wakeup_clk_root	LPCG87
MEC64	clk	wakeup_axi_clk_root	LPCG32
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG32
	clk	wakeup_axi_clk_root	NA
	clk	wakeup_axi_clk_root	LPCG33
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG33
	clk	wakeup_axi_clk_root	NA
MIC (PDM)	ipg_clk_app_nonstop	mic_clk_root	LPCG142
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG142
MUn	ipg_clk_mua	bus_aon_clk_root	LPCG27
	ipg_clk_mub	bus_aon_clk_root	LPCG28
	ipg_clk_s_mua	bus_aon_clk_root	LPCG27
	ipg_clk_s_mub	bus_aon_clk_root	LPCG28
	ipt_clk_mua	bus_aon_clk_root	NA
	ipt_clk_mub	bus_aon_clk_root	NA
	ipg_clk_mua	bus_wakeup_clk_root	LPCG27
	ipg_clk_mub	bus_wakeup_clk_root	LPCG28
	ipg_clk_s_mua	bus_wakeup_clk_root	LPCG27
	ipg_clk_s_mub	bus_wakeup_clk_root	LPCG28

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
	ipt_clk_mua	bus_wakeup_clk_root	NA
	ipt_clk_mub	bus_wakeup_clk_root	NA
NIC	mainclk	wakeup_axi_clk_root	LPCG4
	mainclk_r	wakeup_axi_clk_root	LPCG6
eQDCn	ipg_clk	bus_wakeup_clk_root	LPCG118
	ipg_clk	bus_wakeup_clk_root	LPCG119
	ipg_clk	bus_wakeup_clk_root	LPCG120
	ipg_clk	bus_wakeup_clk_root	LPCG121
TMRn	clk0	bus_wakeup_clk_root	LPCG65
	clk0	bus_wakeup_clk_root	LPCG66
	clk0	bus_wakeup_clk_root	LPCG67
	clk0	bus_wakeup_clk_root	LPCG68
	clk0	bus_wakeup_clk_root	LPCG69
	clk0	bus_wakeup_clk_root	LPCG70
	clk0	bus_wakeup_clk_root	LPCG71
	clk0	bus_wakeup_clk_root	LPCG72
Edgelock enclave	edgelock_clk	edgelock_clk_root	LPCG2
	sys_32k_clk	OSC_32K	NA
	sys_dbg_clk	edgelock_clk_root	NA
	sys_hclk	edgelock_clk_root	LPCG2
	sys_ipt_clk	edgelock_clk_root	NA
	sys_lpuart_baud_clk	lpuart0102_clk_root	NA
	sys_mu_ipg_clk	edgelock_clk_root	NA
	sys_ref1_clk	m33_clk_root	NA
	sys_ref3_clk	OSC_24M	NA
	sys_trust_hclk	edgelock_clk_root	LPCG2
SAIn	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG136
	ipg_clk_sai_mclk	sai1_clk_root	LPCG136
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG137
	ipg_clk_sai_mclk	sai2_clk_root	LPCG137
	sai_mclk_in[3]	Clock selected by SAI2_MCLK_CTRL[SAI2_MCLK3_SEL]	LPCG137

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG138
	ipg_clk_sai_mclk	sai3_clk_root	LPCG138
	sai_mclk_in[3]	Clock selected by SAI2_MCLK_CTRL[SAI2_MCLK3_SEL]	LPCG138
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG139
	ipg_clk_sai_mclk	Clock selected by SAI4_MCLK_CTRL[SAI4_MCLK1_SEL]	LPCG139
	sai_mclk_in[2]	Clock selected by SAI4_MCLK_CTRL[SAI4_MCLK2_SEL]	LPCG139
	sai_mclk_in[3]	Clock selected by SAI4_MCLK_CTRL[SAI4_MCLK3_SEL]	LPCG139
SEMA42	clk	bus_aon_clk_root	LPCG25
	clk	bus_wakeup_clk_root	LPCG26
SEMC	ipg_clk	semc_clk_root	LPCG39
SINCn	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG108
	sinc_func_clk	bus_wakeup_clk_root	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG109
	sinc_func_clk	bus_wakeup_clk_root	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG110
	sinc_func_clk	bus_wakeup_clk_root	NA
SPDIF	extal_clk	OSC_24M	LPCG140
	gclkw_t0, ipg_clk_s	bus_wakeup_clk_root	LPCG140
	tx_clk	spdif_clk_root	LPCG140
	tx_clk5	mic_clk_root	LPCG140
SYS_CTR	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG75
	sys_ctr_base_clk	OSC_24M	LPCG75
	sys_ctr_slow_clk	OSC_32K	NA
TPMn	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG59
	tpm_clk	bus_aon_clk_root	LPCG59
	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG60
	tpm_clk	tpm2_clk_root	LPCG60

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG61
	tpm_clk	bus_wakeup_clk_root	LPCG61
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG62
	tpm_clk	tpm4_clk_root	LPCG62
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG63
	tpm_clk	tpm5_clk_root	LPCG63
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG64
	tpm_clk	tpm6_clk_root	LPCG64
TSTMRn	ipg_clk, ipg_clk_s	bus_aon_clk_root	LPCG75
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG75
USB	ipg_ahb_clk	wakeup_axi_clk_root	LPCG107
	ipg_clk_32khz	OSC_32K	NA
	ipg_clk_s	wakeup_axi_clk_root	NA
USBPHYn	ckil_32k	OSC_32K	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG107
	ckil_32k	OSC_32K	NA
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG107
USDHCn	hclk	wakeup_axi_clk_root	LPCG105
	ipg_clk_32khz	OSC_32K	NA
	ipg_clk_perclk	usdhc1_clk_root	LPCG105
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG105
	hclk	wakeup_axi_clk_root	LPCG106
	ipg_clk_32khz	OSC_32K	NA
	ipg_clk_perclk	usdhc2_clk_root	LPCG106
	ipg_clk, ipg_clk_s	bus_wakeup_clk_root	LPCG106
WDOGn	ext_clk	OSC_RC_24M	LPCG19
	int_clk	bus_aon_clk_root	LPCG19
	ipg_clk_gated	bus_aon_clk_root	LPCG19
	ipg_clk_s	bus_aon_clk_root	LPCG19
	ipg_clk_ungated	bus_aon_clk_root	LPCG19
	ipt_clk	bus_aon_clk_root	NA
	lpo_clk	OSC_32K	NA

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
	ext_clk	OSC_RC_24M	LPCG20
	int_clk	bus_aon_clk_root	LPCG20
	ipg_clk_gated	bus_aon_clk_root	LPCG20
	ipg_clk_s	bus_aon_clk_root	LPCG20
	ipg_clk_ungated	bus_aon_clk_root	LPCG20
	ipt_clk	bus_aon_clk_root	NA
	lpo_clk	OSC_32K	NA
	ext_clk	OSC_RC_24M	LPCG21
	int_clk	bus_wakeup_clk_root	LPCG21
	ipg_clk_gated	bus_wakeup_clk_root	LPCG21
	ipg_clk_s	bus_wakeup_clk_root	LPCG21
	ipg_clk_ungated	bus_wakeup_clk_root	LPCG21
	lpo_clk	OSC_32K	NA
	ext_clk	OSC_RC_24M	LPCG22
	int_clk	bus_wakeup_clk_root	LPCG22
	ipg_clk_gated	bus_wakeup_clk_root	LPCG22
	ipg_clk_s	bus_wakeup_clk_root	LPCG22
	ipg_clk_ungated	bus_wakeup_clk_root	LPCG22
	lpo_clk	OSC_32K	NA
	ext_clk	OSC_RC_24M	LPCG23
	int_clk	bus_wakeup_clk_root	LPCG23
	ipg_clk_gated	bus_wakeup_clk_root	LPCG23
	ipg_clk_s	bus_wakeup_clk_root	LPCG23
	ipg_clk_ungated	bus_wakeup_clk_root	LPCG23
	lpo_clk	OSC_32K	NA
XBARn	ipb_clk,ipb_clk_DUMMY	bus_wakeup_clk_root	LPCG111
	ipb_clk,ipb_clk_DUMMY	bus_wakeup_clk_root	LPCG112
	ipb_clk,ipb_clk_DUMMY	bus_wakeup_clk_root	LPCG113
NETC	ipg_clk	netc_clk_root	LPCG128
	ipg_clk_1	netc_clk_root	LPCG128
	ipg_clk_2	netc_clk_root	LPCG128
	ipg_clk_3	netc_clk_root	LPCG128

Table continues on the next page...

Table 136. System Clocks Table (continued)

Module	Clock	Clock Root	LPCG control
	tmr_1588_clk	tmr_1588_clk_root	LPCG128
	eth0_rmii_ref50_clk	mac0_clk_root	NA
	eth1_rmii_ref50_clk	mac1_clk_root	NA
	eth2_rmii_ref50_clk	mac2_clk_root	NA
	eth3_rmii_ref50_clk	mac3_clk_root	NA
	eth4_rmii_ref50_clk	mac4_clk_root	NA
CM7	CLK,HCLK,FCLK,CLK1,HCLK1,FCLK1	ccm_m7_clk_root	LPCG0

20.4 Functional description

CCM is comprised of several functional blocks:

- Clock Root
- Clock Gate
- Clock Source
- General Purpose Registers (GPR)

These functions can be configured under various CCM modes. The first section will describe the ways to configure these functions under different CCM modes. And the other sections will describe these functions independently.

20.4.1 Configuration of Functions

CCM functional blocks support the following modes:

- Direct Configuration mode
- CPU Low Power mode
- Auto mode (only for clock sources)

The default CCM mode is Direct Configuration mode.

20.4.2 Clock Sources

Clock roots are generated from various clock sources, which can be either oscillators or PLLs. Clock source is implemented to control PLLs and oscillators, to be automatically turned OFF or ON on system low power actions.

The following table shows the clock sources for the CCM:

Table 137. Clock Sources

Clock Source	Description
OSC_RC_24M	24MHz RC OSC output
OSC_RC_400M	400MHz RC OSC output
OSC_24M	Crystal OSC 24MHz clock
ARM_PLL	ARM PLL VCO

Table continues on the next page...

Table 137. Clock Sources (continued)

Clock Source	Description
SYS_PLL2	System PLL2 VCO
SYS_PLL2_PFD0	System PLL2 PFD0 clock
SYS_PLL2_PFD1	System PLL2 PFD1 clock
SYS_PLL2_PFD2	System PLL2 PFD2 clock
SYS_PLL2_PFD3	System PLL2 PFD3 clock
SYS_PLL3	System PLL3 VCO
SYS_PLL3_DIV2	System PLL3 divided by 2 clock
SYS_PLL3_PFD0	System PLL3 PFD0 clock
SYS_PLL3_PFD1	System PLL3 PFD1 clock
SYS_PLL3_PFD2	System PLL3 PFD2 clock
SYS_PLL3_PFD3	System PLL3 PFD3 clock
SYS_PLL1	System PLL1 VCO
SYS_PLL1_DIV2	System PLL1 divided by 2 clock
SYS_PLL1_DIV5	System PLL1 divided by 5 clock
AUDIO_PLL	Audio PLL VCO

NOTE

The clock source calculates whether the clock is required for each CPU domain. If clock source is not needed for any domain, clock will be shutdown.

Clock Source blocks have 3 work mode: Direct Control mode, CPULPM mode and Auto mode. Direct Control mode is the default mode. After reset, all Clock Source blocks work in Direct Control mode.

All registers are readable for any domain in any mode, but write access is restricted for protection. TrustZone unauthenticated access will be ignored.

The OSCPLLn_AUTHEN and OSCPLLn_DIRECT registers can be changed in any mode, while the write access should be from the domain granted by OSCPLLn_AUTHEN[WHITE_LIST] and TrustZone authenticated. The default value of OSCPLLn_AUTHEN[WHITE_LIST] is 0xFFFF, which means all 16 domains are granted.

The OSCPLLn_LPM registers can only be written in CPULPM mode, otherwise the write access will be ignored. And the write access in CPULPM mode should also comply with the TrustZone access control.

In Direct Control mode, the ON/OFF of clock source is determined by OSCPLLn_DIRECT[ON] directly. The default state of OSCPLL is OFF.

In CPULPM mode, the ON/OFF of clock source is dependent on the settings in OSCPLLn_LPMs and the current mode every domain is in. There are 2-bit clock dependent level settings for each domain respectively. The bigger the 2-bit value is, the stronger the clock is dependent. Valid values are as follows:

- 0: Disable in any mode
- 1: Turn on in Run mode
- 2: Turn on in Run and Wait mode
- 3: Enable in Run, Wait, and Stop modes

In CPULPM mode, LPCG will calculate whether the clock is required by any CPU domain. If clock source is not needed for any domain, the clock will be shutdown.

In Auto mode, the specific clock source will be shut down automatically when all clock roots derived from this clock source are OFF, and the OSCPLLn_DIRECT[ON] is de-asserted.

20.4.3 Clock Root

CCM clock root generation contains multiple clock root channels. All clock roots are asynchronous each other, even all the clock root setting are the same.

Each clock channel contains an 4-to-1 MUX, and a 8-bit divider. The clock MUX selects 1 clock out of 4 clock inputs. The 8-bit divider can divide selected clock by up to 256. The clock output of the clock channel can be gated off.

The clock root channel setting can be changed from any value to any value, and at any time. If more than one setting is changed at the same time, internal logic will use a procedure to change the setting. The procedure will make sure the clock output is not faster than the current clock root setting or target clock root setting. If the application does care about the clock during transition, software will need to change the setting one field at a time. This typically happens when a clock must be faster than the given frequency. In this case, the application needs to change each field one at a time and wait for the field to take effect before changing the next field.

Although a clock root channel is designed to be able to switch clocks regardless of clock input status. When the current selected clock input is off, the application needs to avoid changing the clock MUX after the clock input is turned on. This will lead to unstable behavior for the clock root and loading peripherals. To avoid this scenario, the application can either switch the clock MUX before turning on the clock input, or wait until the clock input is stable before changing the MUX option.

The division factor of the divider is equal to CLOCK_ROOTn_CONTROL[DIV]+1.

All registers are readable for any domain in any mode, but write access are restricted. Unauthenticated write access will be ignored. Only the access from the domains on the whitelist can change the value of CLOCK_ROOTn_CONTROL register, after passing TrustZone authentication.

Table 138. Clock Roots

Clock Control Register	Clock Root	Max Freq (MHz)			Source Select CLOCK_ROOTn_CONTROL[MUX]
		HSRUN (High Speed Run mode)	RUN (Run mode)	UDRUN (Underdrive mode)	
CLOCK_ROOT0	m7_clk_root	800	600	360	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_ARM 11 - PLL_480
CLOCK_ROOT1	m33_clk_root	240	240	100	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480 11 - PLL_ARM
CLOCK_ROOT2	edgelock_clk_root	200	200	66	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_1G

Table continues on the next page...

Table 138. Clock Roots (continued)

Clock Control Register	Clock Root	Max Freq (MHz)			Source Select CLOCK_ROOTn_CONTROL[MUX]
		HSRUN (High Speed Run mode)	RUN (Run mode)	UDRUN (Underdrive mode)	
					11 - PLL_528_PFD1
CLOCK_ROOT3	bus_aon_clk_root	133	133	50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_528 11 - PLL_480_PFD2
CLOCK_ROOT4	bus_wakeup_clk_root	133	133	50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_528 11 - PLL_480_PFD1
CLOCK_ROOT5	wakeup_axi_clk_root	240	200	100	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480 11 - PLL_528_PFD1
CLOCK_ROOT6	swo_trace_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_1G_DIV5
CLOCK_ROOT7	m33_systick_clk_root	60	60	30	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - OSC_24M 11 - PLL_480_DIV2
CLOCK_ROOT8	m7_systick_clk_root	60	60	30	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - OSC_24M 11 - PLL_480_DIV2
CLOCK_ROOT9	flexio1_clk_root	120	120	60	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_1G_DIV5

Table continues on the next page...

Table 138. Clock Roots (continued)

Clock Control Register	Clock Root	Max Freq (MHz)			Source Select CLOCK_ROOTn_CONTROL[MUX]
		HSRUN (High Speed Run mode)	RUN (Run mode)	UDRUN (Underdrive mode)	
CLOCK_ROOT10	flexio2_clk_root	120	120	60	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_1G_DIV5
CLOCK_ROOT11	lpit3_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT12	lptmr1_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT13	lptmr2_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT14	lptmr3_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT15	tpm2_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT16	tpm4_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3

Table continues on the next page...

Table 138. Clock Roots (continued)

Clock Control Register	Clock Root	Max Freq (MHz)			Source Select CLOCK_ROOTn_CONTROL[MUX]
		HSRUN (High Speed Run mode)	RUN (Run mode)	UDRUN (Underdrive mode)	
CLOCK_ROOT17	tpm5_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT18	tpm6_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT19	gpt1_clk_root	240	240	120	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT20	gpt2_clk_root	240	240	120	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT21	flexspi1_clk_root	400	400	200	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_PFD0 11 - PLL_528_PFD0
CLOCK_ROOT22	flexspi2_clk_root	333	333	166.50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_PFD2 11 - PLL_528_PFD1
CLOCK_ROOT23	flexspi_slv_clk_root	133	133	66.50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_528 11 - PLL_1G

Table continues on the next page...

Table 138. Clock Roots (continued)

Clock Control Register	Clock Root	Max Freq (MHz)			Source Select CLOCK_ROOTn_CONTROL[MUX]
		HSRUN (High Speed Run mode)	RUN (Run mode)	UDRUN (Underdrive mode)	
CLOCK_ROOT24	can1_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480 11 - OSC_24M
CLOCK_ROOT25	can2_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480 11 - OSC_24M
CLOCK_ROOT26	can3_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480 11 - OSC_24M
CLOCK_ROOT27	lpuart0102_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT28	lpuart0304_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT29	lpuart0506_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT30	lpuart0708_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3

Table continues on the next page...

Table 138. Clock Roots (continued)

Clock Control Register	Clock Root	Max Freq (MHz)			Source Select CLOCK_ROOTn_CONTROL[MUX]
		HSRUN (High Speed Run mode)	RUN (Run mode)	UDRUN (Underdrive mode)	
CLOCK_ROOT31	lpuart0910_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT32	lpuart1112_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT33	lpi2c0102_clk_root	60	60	30	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT34	lpi2c0304_clk_root	60	60	30	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT35	lpi2c0506_clk_root	60	60	30	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT36	lpspi0102_clk_root	135	135	67.50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_PFD1 11 - PLL_528
CLOCK_ROOT37	lpspi0304_clk_root	135	135	67.50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_PFD1 11 - PLL_528

Table continues on the next page...

Table 138. Clock Roots (continued)

Clock Control Register	Clock Root	Max Freq (MHz)			Source Select CLOCK_ROOTn_CONTROL[MUX]
		HSRUN (High Speed Run mode)	RUN (Run mode)	UDRUN (Underdrive mode)	
CLOCK_ROOT38	lpspi0506_clk_root	135	135	67.50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_PFD1 11 - PLL_528
CLOCK_ROOT39	i3c1_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT40	i3c2_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT41	usdhc1_clk_root	200	200	100	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_528_PFD2 11 - PLL_1G_DIV5
CLOCK_ROOT42	usdhc2_clk_root	400	400	200	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_528_PFD2 11 - PLL_1G_DIV5
CLOCK_ROOT43	semc_clk_root	200	200	100	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_1G 11 - PLL_528_PFD0
CLOCK_ROOT44	adc1_clk_root	100	100	50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3

Table continues on the next page...

Table 138. Clock Roots (continued)

Clock Control Register	Clock Root	Max Freq (MHz)			Source Select CLOCK_ROOTn_CONTROL[MUX]
		HSRUN (High Speed Run mode)	RUN (Run mode)	UDRUN (Underdrive mode)	
CLOCK_ROOT45	adc2_clk_root	100	100	50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_528_PFD3
CLOCK_ROOT46	acmp_clk_root	240	240	120	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480 11 - PLL_528_PFD3
CLOCK_ROOT47	ecat_clk_root	100	100	50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_1G_DIV2 11 - PLL_1G_DIV5
CLOCK_ROOT48	enet_refclk_root	125	125	62.50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_1G_DIV2 11 - PLL_1G_DIV5
CLOCK_ROOT49	tmr_1588_clk_root	240	240	NA	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480 11 - PLL_528_PFD3
CLOCK_ROOT50	netc_clk_root	240	240	NA	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_PFD3 11 - PLL_528_PFD1
CLOCK_ROOT51	mac0_clk_root	125	125	NA	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_1G_DIV2 11 - PLL_1G_DIV5

Table continues on the next page...

Table 138. Clock Roots (continued)

Clock Control Register	Clock Root	Max Freq (MHz)			Source Select CLOCK_ROOTn_CONTROL[MUX]
		HSRUN (High Speed Run mode)	RUN (Run mode)	UDRUN (Underdrive mode)	
CLOCK_ROOT52	mac1_clk_root	125	125	NA	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_1G_DIV2 11 - PLL_1G_DIV5
CLOCK_ROOT53	mac2_clk_root	125	125	NA	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_1G_DIV2 11 - PLL_1G_DIV5
CLOCK_ROOT54	mac3_clk_root	125	125	NA	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_1G_DIV2 11 - PLL_1G_DIV5
CLOCK_ROOT55	mac4_clk_root	125	125	NA	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_1G_DIV2 11 - PLL_1G_DIV5
CLOCK_ROOT56 - CLOCK_ROOT64	Reserved	-	-	-	-
CLOCK_ROOT65	sai1_clk_root	66	66	33	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_AUDIO 11 - PLL_480_PFD2
CLOCK_ROOT66	sai2_clk_root	66	66	33	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_AUDIO 11 - PLL_480_PFD2
CLOCK_ROOT67	sai3_clk_root	66	66	33	00 - OSC_RC_24M

Table continues on the next page...

Table 138. Clock Roots (continued)

Clock Control Register	Clock Root	Max Freq (MHz)			Source Select CLOCK_ROOTn_CONTROL[MUX]
		HSRUN (High Speed Run mode)	RUN (Run mode)	UDRUN (Underdrive mode)	
					01 - OSC_RC_400M 10 - PLL_AUDIO 11 - PLL_480_PFD2
CLOCK_ROOT68	sai4_clk_root	66	66	33	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_AUDIO 11 - PLL_480_PFD2
CLOCK_ROOT69	spdif_clk_root	66	66	33	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_AUDIO 11 - PLL_480_PFD2
CLOCK_ROOT70	asrc_clk_root	240	240	120	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480 11 - PLL_AUDIO
CLOCK_ROOT71	mic_clk_root	80	80	40	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_AUDIO
CLOCK_ROOT72	ccm_cko1_clk_root	100	100	50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_480_DIV2 11 - PLL_1G_DIV2
CLOCK_ROOT73	ccm_cko2_clk_root	100	100	50	00 - OSC_RC_24M 01 - OSC_RC_400M 10 - PLL_1G_DIV5 11 - PLL_ARM

NOTE

The frequency values of the cores are dependent on the part configuration. Please see the datasheet for supported core frequencies.

20.4.4 Clock Gate

Before a clock root goes to on-chip peripherals, the clock roots go through the Low Power Clock Gates (LPCG). These LPCG are implemented to perform automatic clock gating when a domain enters and leaves low-power states. The LPCGs are mapped to peripherals so software can be simplified. LPCGs are also implemented to control peripheral functions, and different LPCGs could interface different clocks of the peripheral.

Clock gates have 2 work modes: Direct Control mode and CPULPM mode. Direct Control mode is the implicit mode. After reset, all clock gates work in Direct Control mode.

All registers are readable for any domain in any mode, but write access is restricted for protection. TrustZone unauthenticated access will be ignored.

The LPCGn_AUTHEN and LPCGn_DIRECT registers can be changed in any mode, while the write access should be from the domain granted by LPCGn_AUTHEN[WHITE_LIST] and TrustZone authenticated. The default value of LPCGn_AUTHEN[WHITE_LIST] is 0xFFFF, which means all 16 domains are granted.

The LPCGn_LPM registers can only be written in CPULPM mode, otherwise the write access will be ignored. And the write access in CPULPM mode should also comply with the TrustZone access control.

In Direct Control mode, the ON/OFF of LPCG is determined by LPCGn_DIRECT[ON] directly. The default state of LPCG is OFF.

In CPULPM mode, the ON/OFF of LPCG is dependent on the settings in LPCGn_LPMs and the current mode every domain is in. There are 2-bit clock dependent level settings for each domain respectively. The bigger the 2-bit value is, the stronger the clock is dependent. Valid values are as follows:

- 0: this clock is not needed in any mode
- 1: this clock is needed in RUN mode, but not needed in WAIT, STOP mode
- 2: this clock is needed in RUN and WAIT mode, but not needed in STOP mode
- 3: this clock is needed in RUN, WAIT and STOP mode

In CPULPM mode, LPCG will calculate whether the clock is required by any CPU domain. If clock is not needed for any domain, the clock will be shutdown.

Table 139. Clock Gate Table

Gating Channel	Generated Gate Signal
LPCG0	clk_enable_cm7
LPCG1	clk_enable_cm33
LPCG2	clk_enable_edgelock
LPCG3	clk_enable_sim_aon
LPCG4	clk_enable_sim_wakeup
LPCG5	clk_enable_sim_mega
LPCG6	clk_enable_sim_r
LPCG7	clk_enable_anadig
LPCG8	clk_enable_dcdc
LPCG9	clk_enable_src
LPCG10	clk_enable_ccm
LPCG11	clk_enable_gpc
LPCG12	clk_enable_adc1

Table continues on the next page...

Table 139. Clock Gate Table (continued)

Gating Channel	Generated Gate Signal
LPCG13	clk_enable_adc2
LPCG14	clk_enable_dac
LPCG15	clk_enable_acmp1
LPCG16	clk_enable_acmp2
LPCG17	clk_enable_acmp3
LPCG18	clk_enable_acmp4
LPCG19	clk_enable_wdog1
LPCG20	clk_enable_wdog2
LPCG21	clk_enable_wdog3
LPCG22	clk_enable_wdog4
LPCG23	clk_enable_wdog5
LPCG24	clk_enable_ewm
LPCG25	clk_enable_sema1
LPCG26	clk_enable_sema2
LPCG27	clk_enable_mu_a
LPCG28	clk_enable_mu_b
LPCG29	clk_enable_edma3
LPCG30	clk_enable_edma4
LPCG31	clk_enable_romcp
LPCG32	clk_enable_ocram1
LPCG33	clk_enable_ocram2
LPCG34	clk_enable_flexspi1
LPCG35	clk_enable_flexspi2
LPCG36	clk_enable_flexspi_slv
LPCG37	clk_enable_trdc
LPCG38	clk_enable_ocotp
LPCG39	clk_enable_semc
LPCG40	clk_enable_iee
LPCG41	clk_enable_trace
LPCG42	clk_enable_swo
LPCG43	clk_enable_iomuxc1
LPCG44	clk_enable_iomuxc2

Table continues on the next page...

Table 139. Clock Gate Table (continued)

Gating Channel	Generated Gate Signal
LPCG45	clk_enable_gpio1
LPCG46	clk_enable_gpio2
LPCG47	clk_enable_gpio3
LPCG48	clk_enable_gpio4
LPCG49	clk_enable_gpio5
LPCG50	clk_enable_gpio6
LPCG51	clk_enable_flexio1
LPCG52	clk_enable_flexio2
LPCG53	clk_enable_lpit1
LPCG54	clk_enable_lpit2
LPCG55	clk_enable_lpit3
LPCG56	clk_enable_lptmr1
LPCG57	clk_enable_lptmr2
LPCG58	clk_enable_lptmr3
LPCG59	clk_enable_tpm1
LPCG60	clk_enable_tpm2
LPCG61	clk_enable_tpm3
LPCG62	clk_enable_tpm4
LPCG63	clk_enable_tpm5
LPCG64	clk_enable_tpm6
LPCG65	clk_enable_qtimer1
LPCG66	clk_enable_qtimer2
LPCG67	clk_enable_qtimer3
LPCG68	clk_enable_qtimer4
LPCG69	clk_enable_qtimer5
LPCG70	clk_enable_qtimer6
LPCG71	clk_enable_qtimer7
LPCG72	clk_enable_qtimer8
LPCG73	clk_enable_gpt1
LPCG74	clk_enable_gpt2
LPCG75	clk_enable_tstmr1
LPCG76	clk_enable_can1

Table continues on the next page...

Table 139. Clock Gate Table (continued)

Gating Channel	Generated Gate Signal
LPCG77	clk_enable_can2
LPCG78	clk_enable_can3
LPCG79	clk_enable_lpuart1
LPCG80	clk_enable_lpuart2
LPCG81	clk_enable_lpuart3
LPCG82	clk_enable_lpuart4
LPCG83	clk_enable_lpuart5
LPCG84	clk_enable_lpuart6
LPCG85	clk_enable_lpuart7
LPCG86	clk_enable_lpuart8
LPCG87	clk_enable_lpuart9
LPCG88	clk_enable_lpuart10
LPCG89	clk_enable_lpuart11
LPCG90	clk_enable_lpuart12
LPCG91	clk_enable_lpi2c1
LPCG92	clk_enable_lpi2c2
LPCG93	clk_enable_lpi2c3
LPCG94	clk_enable_lpi2c4
LPCG95	clk_enable_lpi2c5
LPCG96	clk_enable_lpi2c6
LPCG97	clk_enable_lpspi1
LPCG98	clk_enable_lpspi2
LPCG99	clk_enable_lpspi3
LPCG100	clk_enable_lpspi4
LPCG101	clk_enable_lpspi5
LPCG102	clk_enable_lpspi6
LPCG103	clk_enable_i3c1
LPCG104	clk_enable_i3c2
LPCG105	clk_enable_usdhc1
LPCG106	clk_enable_usdhc2
LPCG107	clk_enable_usb
LPCG108	clk_enable_sinc1

Table continues on the next page...

Table 139. Clock Gate Table (continued)

Gating Channel	Generated Gate Signal
LPCG109	clk_enable_sinc2
LPCG110	clk_enable_sinc3
LPCG111	clk_enable_xbar1
LPCG112	clk_enable_xbar2
LPCG113	clk_enable_xbar3
LPCG114	clk_enable_aoi1
LPCG115	clk_enable_aoi2
LPCG116	clk_enable_aoi3
LPCG117	clk_enable_aoi4
LPCG118	clk_enable_enc1
LPCG119	clk_enable_enc2
LPCG120	clk_enable_enc3
LPCG121	clk_enable_enc4
LPCG122	clk_enable_kpp
LPCG123	clk_enable_flexpwm1
LPCG124	clk_enable_flexpwm2
LPCG125	clk_enable_flexpwm3
LPCG126	clk_enable_flexpwm4
LPCG127	clk_enable_ecat
LPCG128	clk_enable_netc
LPCG129	reserved
LPCG130	reserved
LPCG131	reserved
LPCG132	clk_enable_xbus
LPCG133	reserved
LPCG134	reserved
LPCG135	clk_enable_mctrl
LPCG136	clk_enable_sai1
LPCG137	clk_enable_sai2
LPCG138	clk_enable_sai3
LPCG139	clk_enable_sai4
LPCG140	clk_enable_spdif

Table continues on the next page...

Table 139. Clock Gate Table (continued)

Gating Channel	Generated Gate Signal
LPCG141	clk_enable_asrc
LPCG142	clk_enable_mic
LPCG143	clk_enable_vref
LPCG144	clk_enable_bist
LPCG145	clk_enable_w2m7
LPCG146	clk_enable_m72w
LPCG147	clk_enable_w2ao
LPCG148	clk_enable_ao2w

20.4.5 Clock Observe

The clock observe slices are used to observe on-chip clocks or signals by measurement or IO pins.

Each observe channel contains a 512-to-1 MUX, an inverter and an 8-bit divider. The clock MUX selects one observe signal out of 512 inputs.

The 8-bit divider can divide the selected clock by up to 256. The inverter is selectable by software. Raw MUX is used to bypass inverter and divider to observe selected signal directly.

There is also a reset control on for observe selection part. If RAW MUX, inverter signal select MUX or divide factor changed, a logic reset will be automatically reset. If all those setting are kept unchanged, a reset can be force reset by explicitly write "1" to RESET field.

The MUX options are listed in following table. The table listed each option signal name, and present or not in each part of the chip. Any option not shown in the table are tied to "0". Any a tied "0" option can be selected for saving power.

Table 140. Mux options

Number	Name
2	osc_rc_24m
3	osc_rc_400m
5	osc_24m_out
7	pll_arm_out
9	pll_528_out
10	pll_528_pfd0
11	pll_528_pfd1
12	pll_528_pfd2
13	pll_528_pfd3
15	pll_480_out
16	pll_480_div2
17	pll_480_pfd0
18	pll_480_pfd1

Table continues on the next page...

Table 140. Mux options (continued)

Number	Name
19	pll_480_pfd2
20	pll_480_pfd3
22	pll_1g_out
23	pll_1g_div2
24	pll_1g_div5
26	pll_audio_out
34	osc_rc_24m_ch_silent_n
35	osc_rc_400m_ch_silent_n
37	osc_24m_out_ch_silent_n
39	pll_arm_out_ch_silent_n
41	pll_528_out_ch_silent_n
42	pll_528_pfd0_ch_silent_n
43	pll_528_pfd1_ch_silent_n
44	pll_528_pfd2_ch_silent_n
45	pll_528_pfd3_ch_silent_n
47	pll_480_out_ch_silent_n
48	pll_480_div2_ch_silent_n
49	pll_480_pfd0_ch_silent_n
50	pll_480_pfd1_ch_silent_n
51	pll_480_pfd2_ch_silent_n
52	pll_480_pfd3_ch_silent_n
54	pll_1g_out_ch_silent_n
55	pll_1g_div2_ch_silent_n
56	pll_1g_div5_ch_silent_n
58	pll_audio_out_ch_silent_n
66	osc_rc_24m_in_use
67	osc_rc_400m_in_use
69	osc_24m_out_in_use
71	pll_arm_out_in_use
73	pll_528_out_in_use
74	pll_528_pfd0_in_use
75	pll_528_pfd1_in_use

Table continues on the next page...

Table 140. Mux options (continued)

Number	Name
76	pll_528_pfd2_in_use
77	pll_528_pfd3_in_use
79	pll_480_out_in_use
80	pll_480_div2_in_use
81	pll_480_pfd0_in_use
82	pll_480_pfd1_in_use
83	pll_480_pfd2_in_use
84	pll_480_pfd3_in_use
86	pll_1g_out_in_use
87	pll_1g_div2_in_use
88	pll_1g_div5_in_use
90	pll_audio_out_in_use
128	m7_clk_root
129	m33_clk_root
130	sentinel_clk_root
131	bus_aon_clk_root
132	bus_wakeup_clk_root
133	wakeup_axi_clk_root
134	swo_trace_clk_root
135	m33_systick_clk_root
136	m7_systick_clk_root
137	flexio1_clk_root
138	flexio2_clk_root
139	lpit3_clk_root
140	lptmr1_clk_root
141	lptmr2_clk_root
142	lptmr3_clk_root
143	tpm2_clk_root
144	tpm4_clk_root
145	tpm5_clk_root
146	tpm6_clk_root
147	gpt1_clk_root

Table continues on the next page...

Table 140. Mux options (continued)

Number	Name
148	gpt2_clk_root
149	flexspi1_clk_root
150	flexspi2_clk_root
151	flexspi_slv_clk_root
152	can1_clk_root
153	can2_clk_root
154	can3_clk_root
155	lpuart0102_clk_root
156	lpuart0304_clk_root
157	lpuart0506_clk_root
158	lpuart0708_clk_root
159	lpuart0910_clk_root
160	lpuart1112_clk_root
161	lpi2c0102_clk_root
162	lpi2c0304_clk_root
163	lpi2c0506_clk_root
164	lpspi0102_clk_root
165	lpspi0304_clk_root
166	lpspi0506_clk_root
167	i3c1_clk_root
168	i3c2_clk_root
169	usdhc1_clk_root
170	usdhc2_clk_root
171	semc_clk_root
172	adc1_clk_root
173	adc2_clk_root
174	acmp_clk_root
175	ecat_clk_root
176	enet_refclk_root
177	tmr_1588_clk_root
178	netc_clk_root
179	mac0_clk_root

Table continues on the next page...

Table 140. Mux options (continued)

Number	Name
180	mac1_clk_root
181	mac2_clk_root
182	mac3_clk_root
183	mac4_clk_root
184	serdes0_clk_root
185	serdes1_clk_root
186	serdes2_clk_root
187	serdes0_1g_clk_root
188	serdes1_1g_clk_root
189	serdes2_1g_clk_root
190	xcelbusx_clk_root
191	xriocu4_clk_root
192	motorctrl_clk_root
193	sai1_clk_root
194	sai2_clk_root
195	sai3_clk_root
196	sai4_clk_root
197	spdif_clk_root
198	asrc_clk_root
199	mic_clk_root
200	ccm_cko1_clk_root
201	ccm_cko2_clk_root
256	cpu_power_mode_0[0]
257	cpu_power_mode_0[1]
258	cpu_mode_trans_lpcg_done[0]
259	cpu_mode_trans_lpcg_request[0]
260	cpu_mode_trans_oscpll_done[0]
261	cpu_mode_trans_oscpll_request[0]
264	cpu_power_mode_1[0]
265	cpu_power_mode_1[1]
266	cpu_mode_trans_lpcg_done[1]
267	cpu_mode_trans_lpcg_request[1]

Table continues on the next page...

Table 140. Mux options (continued)

Number	Name
268	cpu_mode_trans_oscpll_done[1]
269	cpu_mode_trans_oscpll_request[1]
272	cpu_power_mode_2[0]
273	cpu_power_mode_2[1]
274	cpu_mode_trans_lpcg_done[2]
275	cpu_mode_trans_lpcg_request[2]
276	cpu_mode_trans_oscpll_done[2]
277	cpu_mode_trans_oscpll_request[2]
280	cpu_power_mode_3[0]
281	cpu_power_mode_3[1]
282	cpu_mode_trans_lpcg_done[3]
283	cpu_mode_trans_lpcg_request[3]
284	cpu_mode_trans_oscpll_done[3]
285	cpu_mode_trans_oscpll_request[3]
288	cpu_power_mode_4[0]
289	cpu_power_mode_4[1]
290	cpu_mode_trans_lpcg_done[4]
291	cpu_mode_trans_lpcg_request[4]
292	cpu_mode_trans_oscpll_done[4]
293	cpu_mode_trans_oscpll_request[4]
296	cpu_power_mode_5[0]
297	cpu_power_mode_5[1]
298	cpu_mode_trans_lpcg_done[5]
299	cpu_mode_trans_lpcg_request[5]
300	cpu_mode_trans_oscpll_done[5]
301	cpu_mode_trans_oscpll_request[5]
304	cpu_power_mode_6[0]
305	cpu_power_mode_6[1]
306	cpu_mode_trans_lpcg_done[6]
307	cpu_mode_trans_lpcg_request[6]
308	cpu_mode_trans_oscpll_done[6]
309	cpu_mode_trans_oscpll_request[6]

Table continues on the next page...

Table 140. Mux options (continued)

Number	Name
312	cpu_power_mode_7[0]
313	cpu_power_mode_7[1]
314	cpu_mode_trans_lpcg_done[7]
315	cpu_mode_trans_lpcg_request[7]
316	cpu_mode_trans_oscpll_done[7]
317	cpu_mode_trans_oscpll_request[7]
320	cpu_power_mode_8[0]
321	cpu_power_mode_8[1]
322	cpu_mode_trans_lpcg_done[8]
323	cpu_mode_trans_lpcg_request[8]
324	cpu_mode_trans_oscpll_done[8]
325	cpu_mode_trans_oscpll_request[8]
328	cpu_power_mode_9[0]
329	cpu_power_mode_9[1]
330	cpu_mode_trans_lpcg_done[9]
331	cpu_mode_trans_lpcg_request[9]
332	cpu_mode_trans_oscpll_done[9]
333	cpu_mode_trans_oscpll_request[9]
336	cpu_power_mode_10[0]
337	cpu_power_mode_10[1]
338	cpu_mode_trans_lpcg_done[10]
339	cpu_mode_trans_lpcg_request[10]
340	cpu_mode_trans_oscpll_done[10]
341	cpu_mode_trans_oscpll_request[10]
344	cpu_power_mode_11[0]
345	cpu_power_mode_11[1]
346	cpu_mode_trans_lpcg_done[11]
347	cpu_mode_trans_lpcg_request[11]
348	cpu_mode_trans_oscpll_done[11]
349	cpu_mode_trans_oscpll_request[11]
352	cpu_power_mode_12[0]
353	cpu_power_mode_12[1]

Table continues on the next page...

Table 140. Mux options (continued)

Number	Name
354	cpu_mode_trans_lpcg_done[12]
355	cpu_mode_trans_lpcg_request[12]
356	cpu_mode_trans_oscpll_done[12]
357	cpu_mode_trans_oscpll_request[12]
360	cpu_power_mode_13[0]
361	cpu_power_mode_13[1]
362	cpu_mode_trans_lpcg_done[13]
363	cpu_mode_trans_lpcg_request[13]
364	cpu_mode_trans_oscpll_done[13]
365	cpu_mode_trans_oscpll_request[13]
368	cpu_power_mode_14[0]
369	cpu_power_mode_14[1]
370	cpu_mode_trans_lpcg_done[14]
371	cpu_mode_trans_lpcg_request[14]
372	cpu_mode_trans_oscpll_done[14]
373	cpu_mode_trans_oscpll_request[14]
376	cpu_power_mode_15[0]
377	cpu_power_mode_15[1]
378	cpu_mode_trans_lpcg_done[15]
379	cpu_mode_trans_lpcg_request[15]
380	cpu_mode_trans_oscpll_done[15]
381	cpu_mode_trans_oscpll_request[15]

Observe slice can do frequency, width, and period measurement on selected signals or clocks. On each time observe signal selection change, all measurement data will be reset, and the measurement starts.

The software can explicitly issue a reset by write "1" to RESET filed. During measurement, the register value may change all the time, and read from registers may not be valid. To get the measurement result, software must turn OFF the observe slice before the measurement results are read.

For frequency measurement, 32K clock is used as time reference,and 32K will be divided to 64Hz. So the observe clock counts every 64Hz cycle. The resolution of frequency measurement is 64Hz. And measurement unit will also record maximum and minimum value ever occurred.

For pulse measurement, measurement unit use Ethernet PLL 1GHz clock, the resolution of pulse is 1ns. To observe, the clock need be divide to some frequency lower than 500MHz for correct result. Pulse measurement unit count every positive pulse, negative pulse, and cycle, and update measurement result. At the same time, pulse measurement unit will record maximum and minimum value ever occurred.

Observe signal can be directly read out from status register.

All registers of Clock Observe blocks are readable for any domain. But the write access should be granted by the OBSERVEn_AUTHEN[WHITELIST] and TrustZone authenticated.

20.4.6 General Purpose Registers (GPR)

The general purpose registers provide miscellaneous controls and data saving functions. The GPRs are divided in two regions - shared and private region.

In shared region, all CPU domains access the same registers. If any CPU updates the value in the shared region, all the other CPU domains can read out the new value. The registers have an access control bit field to protect it from being changed unexpectedly.

In private region, each CPU domain has its own register. A CPU domain can only access registers belonging to itself. Since all the registers share the same address, a CPU domain will not have access to registers belonging to another CPU domain.

Shared registers can be read by any domain at any time, while write access to shared registers must be from domains granted by GPR_SHAREDn_AUTHEN[WHITE_LIST] and TrustZone authenticated. Every shared register has independent whitelist.

Private registers can be and only can be read or written by corresponding domain with TrustZone authentication.

Shared registers have two work modes, Unassigned mode and Domain mode. Private registers work in only Unassigned mode. While list is ignored by private registers. Unassigned mode is an implicit mode, if a shared register is not assigned to domain mode. After reset, all registers work in unassigned mode.

GPR_SHAREDn_AUTHEN and GPR_PRIVATEn_AUTHEN registers can be read by any domain at any time, but the write access must be from domains granted by respective whitelist and TrustZone authenticated.

The following figure shows the structure of the CCM GPR shared registers.

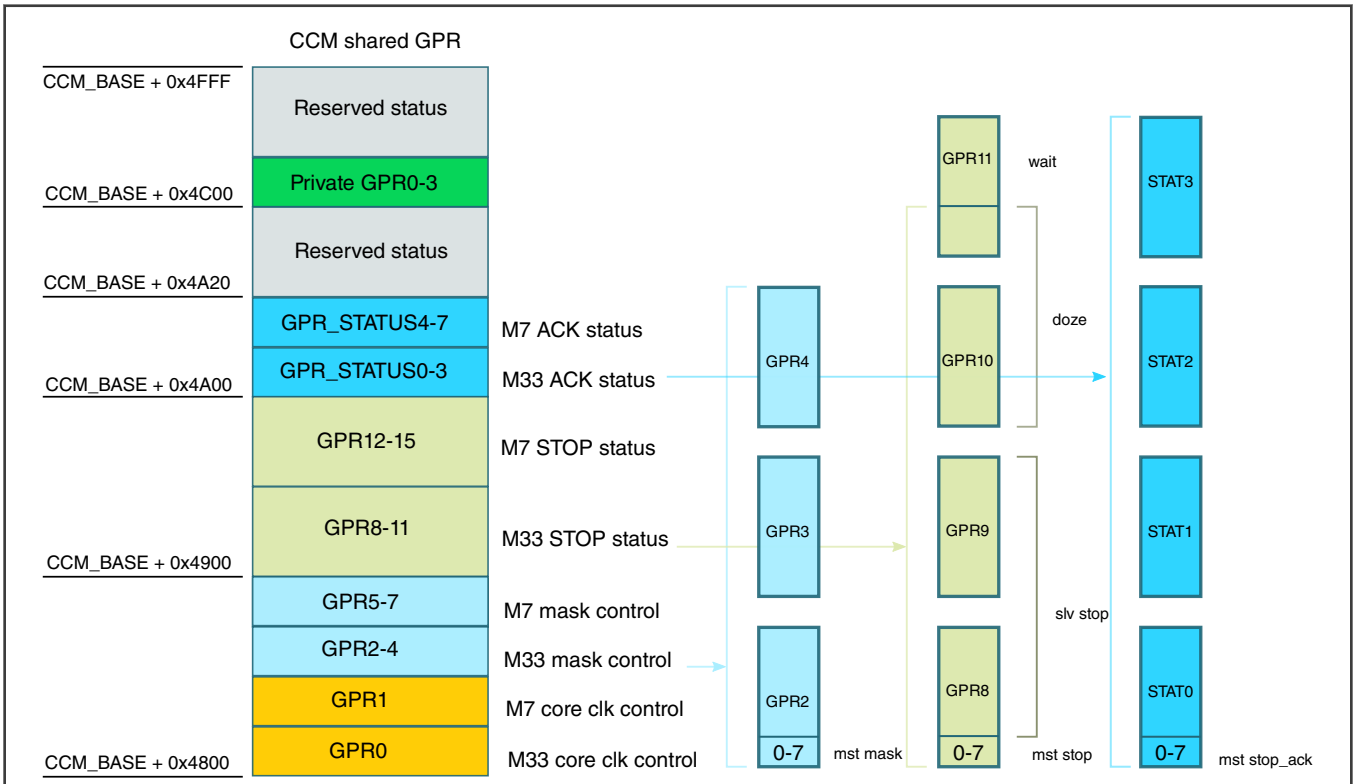


Figure 76. CCM GPR Structure

The following table shows the bitfields for each of the CCM GPR shared registers.

Table 141. CCM GPR shared registers

GPR_CTRL_2 - M33 mask control	
Bit Field Name	**Bit Position**

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m33_mask_cm7	BIT0
m33_mask_cm33	BIT1
m33_mask_edma3	BIT2
m33_mask_edma4	BIT3
m33_mask_netc	BIT4
m33_mask_sim_aon	BIT8
m33_mask_adc1	BIT9
m33_mask_adc2	BIT10
m33_mask_flexspi1	BIT11
m33_mask_flexspi2	BIT12
m33_mask_trdc	BIT13
m33_mask_semc	BIT14
m33_mask_iee	BIT15
m33_mask_gpio1	BIT16
m33_mask_gpio2	BIT17
m33_mask_gpio3	BIT18
m33_mask_gpio4	BIT19
m33_mask_gpio5	BIT20
m33_mask_gpio6	BIT21
m33_mask_flexio1	BIT22
m33_mask_flexio2	BIT23
m33_mask_lpit1	BIT24
m33_mask_lpit2	BIT25
m33_mask_lpit3	BIT26
m33_mask_tpm1	BIT27
m33_mask_tpm2	BIT28
m33_mask_tpm3	BIT29
m33_mask_tpm4	BIT30
m33_mask_tpm5	BIT31
GPR_CTRL_3 - M33 mask control	
Bit Field Name	**Bit Position**
m33_mask_tpm6	BIT0
m33_mask_gpt1	BIT1

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m33_mask_gpt2	BIT2
m33_mask_can1	BIT3
m33_mask_can2	BIT4
m33_mask_can3	BIT5
m33_mask_lpuart1	BIT6
m33_mask_lpuart2	BIT7
m33_mask_lpuart3	BIT8
m33_mask_lpuart4	BIT9
m33_mask_lpuart5	BIT10
m33_mask_lpuart6	BIT11
m33_mask_lpuart7	BIT12
m33_mask_lpuart8	BIT13
m33_mask_lpuart9	BIT14
m33_mask_lpuart10	BIT15
m33_mask_lpuart11	BIT16
m33_mask_lpuart12	BIT17
m33_mask_lpi2c1	BIT18
m33_mask_lpi2c2	BIT19
m33_mask_lpi2c3	BIT20
m33_mask_lpi2c4	BIT21
m33_mask_lpi2c5	BIT22
m33_mask_lpi2c6	BIT23
m33_mask_lpspi1	BIT24
m33_mask_lpspi2	BIT25
m33_mask_lpspi3	BIT26
m33_mask_lpspi4	BIT27
m33_mask_lpspi5	BIT28
m33_mask_lpspi6	BIT29
m33_mask_sinc1	BIT30
m33_mask_sinc2	BIT31
GPR_CTRL_4 - M33 mask control	
Bit Field Name	**Bit Position**
m33_mask_sinc3	BIT0

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m33_mask_sai1	BIT1
m33_mask_sai2	BIT2
m33_mask_sai3	BIT3
m33_mask_sai4	BIT4
m33_mask_mic	BIT5
GPR_CTRL_5 - M7 mask control	
Bit Field Name	**Bit Position**
m7_mask_cm7	BIT0
m7_mask_cm33	BIT1
m7_mask_edma3	BIT2
m7_mask_edma4	BIT3
m7_mask_netc	BIT4
m7_mask_sim_aon	BIT8
m7_mask_adc1	BIT9
m7_mask_adc2	BIT10
m7_mask_flexspi1	BIT11
m7_mask_flexspi2	BIT12
m7_mask_trdc	BIT13
m7_mask_semc	BIT14
m7_mask_iee	BIT15
m7_mask_gpio1	BIT16
m7_mask_gpio2	BIT17
m7_mask_gpio3	BIT18
m7_mask_gpio4	BIT19
m7_mask_gpio5	BIT20
m7_mask_gpio6	BIT21
m7_mask_flexio1	BIT22
m7_mask_flexio2	BIT23
m7_mask_lpit1	BIT24
m7_mask_lpit2	BIT25
m7_mask_lpit3	BIT26
m7_mask_tpm1	BIT27
m7_mask_tpm2	BIT28

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m7_mask_tpm3	BIT29
m7_mask_tpm4	BIT30
m7_mask_tpm5	BIT31
GPR_CTRL_6 - M7 mask control	
Bit Field Name	**Bit Position**
m7_mask_tpm6	BIT0
m7_mask_gpt1	BIT1
m7_mask_gpt2	BIT2
m7_mask_can1	BIT3
m7_mask_can2	BIT4
m7_mask_can3	BIT5
m7_mask_lpuart1	BIT6
m7_mask_lpuart2	BIT7
m7_mask_lpuart3	BIT8
m7_mask_lpuart4	BIT9
m7_mask_lpuart5	BIT10
m7_mask_lpuart6	BIT11
m7_mask_lpuart7	BIT12
m7_mask_lpuart8	BIT13
m7_mask_lpuart9	BIT14
m7_mask_lpuart10	BIT15
m7_mask_lpuart11	BIT16
m7_mask_lpuart12	BIT17
m7_mask_lpi2c1	BIT18
m7_mask_lpi2c2	BIT19
m7_mask_lpi2c3	BIT20
m7_mask_lpi2c4	BIT21
m7_mask_lpi2c5	BIT22
m7_mask_lpi2c6	BIT23
m7_mask_lpspi1	BIT24
m7_mask_lpspi2	BIT25
m7_mask_lpspi3	BIT26
m7_mask_lpspi4	BIT27

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m7_mask_lpspi5	BIT28
m7_mask_lpspi6	BIT29
m7_mask_sinc1	BIT30
m7_mask_sinc2	BIT31
GPR_CTRL_7 - M7 mask control	
Bit Field Name	**Bit Position**
m7_mask_sinc3	BIT0
m7_mask_sai1	BIT1
m7_mask_sai2	BIT2
m7_mask_sai3	BIT3
m7_mask_sai4	BIT4
m7_mask_mic	BIT5
GPR_CTRL_8 - M33 stop/qreq control	
Bit Field Name	**Bit Position**
m33_cm7_ipg_stop	BIT0
m33_cm33_ipg_stop	BIT1
m33_edma3_ipg_stop	BIT2
m33_edma4_ipg_stop	BIT3
m33_netc_ipg_stop	BIT4
m33_sim_aon_ipg_stop	BIT8
m33_adc1_ipg_stop	BIT9
m33_adc2_ipg_stop	BIT10
m33_flexspi1_ipg_stop	BIT11
m33_flexspi2_ipg_stop	BIT12
m33_trdc_ipg_stop	BIT13
m33_semc_ipg_stop	BIT14
m33_iee_ipg_stop	BIT15
m33_gpio1_ipg_stop	BIT16
m33_gpio2_ipg_stop	BIT17
m33_gpio3_ipg_stop	BIT18
m33_gpio4_ipg_stop	BIT19
m33_gpio5_ipg_stop	BIT20
m33_gpio6_ipg_stop	BIT21

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m33_flexio1_ipg_stop	BIT22
m33_flexio2_ipg_stop	BIT23
m33_can1_ipg_stop	BIT24
m33_can2_ipg_stop	BIT25
m33_can3_ipg_stop	BIT26
m33_lpuart1_ipg_stop	BIT27
m33_lpuart2_ipg_stop	BIT28
m33_lpuart3_ipg_stop	BIT29
m33_lpuart4_ipg_stop	BIT30
m33_lpuart5_ipg_stop	BIT31
GPR_CTRL_9 - M33 stop/qreq control	
Bit Field Name	**Bit Position**
m33_lpuart6_ipg_stop	BIT0
m33_lpuart7_ipg_stop	BIT1
m33_lpuart8_ipg_stop	BIT2
m33_lpuart9_ipg_stop	BIT3
m33_lpuart10_ipg_stop	BIT4
m33_lpuart11_ipg_stop	BIT5
m33_lpuart12_ipg_stop	BIT6
m33_lpi2c1_ipg_stop	BIT7
m33_lpi2c2_ipg_stop	BIT8
m33_lpi2c3_ipg_stop	BIT9
m33_lpi2c4_ipg_stop	BIT10
m33_lpi2c5_ipg_stop	BIT11
m33_lpi2c6_ipg_stop	BIT12
m33_lpspi1_ipg_stop	BIT13
m33_lpspi2_ipg_stop	BIT14
m33_lpspi3_ipg_stop	BIT15
m33_lpspi4_ipg_stop	BIT16
m33_lpspi5_ipg_stop	BIT17
m33_lpspi6_ipg_stop	BIT18
m33_sinc1_ipg_stop	BIT19
m33_sinc2_ipg_stop	BIT20

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m33_sinc3_ipg_stop	BIT21
m33_sai1_ipg_stop	BIT22
m33_sai2_ipg_stop	BIT23
m33_sai3_ipg_stop	BIT24
m33_sai4_ipg_stop	BIT25
m33_mic_ipg_stop	BIT26
GPR_CTRL_10 - M33 stop/qreq control	
Bit Field Name	**Bit Position**
m33_adc1_ipg_doze	BIT0
m33_adc2_ipg_doze	BIT1
m33_flexspi1_ipg_doze	BIT2
m33_flexspi2_ipg_doze	BIT3
m33_flexio1_ipg_doze	BIT4
m33_flexio2_ipg_doze	BIT5
m33_lpit1_ipg_doze	BIT6
m33_lpit2_ipg_doze	BIT7
m33_lpit3_ipg_doze	BIT8
m33_tpm1_ipg_doze	BIT9
m33_tpm2_ipg_doze	BIT10
m33_tpm3_ipg_doze	BIT11
m33_tpm4_ipg_doze	BIT12
m33_tpm5_ipg_doze	BIT13
m33_tpm6_ipg_doze	BIT14
m33_gpt1_ipg_doze	BIT15
m33_gpt2_ipg_doze	BIT16
m33_can1_ipg_doze	BIT17
m33_can2_ipg_doze	BIT18
m33_can3_ipg_doze	BIT19
m33_lpuart1_ipg_doze	BIT20
m33_lpuart2_ipg_doze	BIT21
m33_lpuart3_ipg_doze	BIT22
m33_lpuart4_ipg_doze	BIT23
m33_lpuart5_ipg_doze	BIT24

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m33_lpuart6_ipg_doze	BIT25
m33_lpuart7_ipg_doze	BIT26
m33_lpuart8_ipg_doze	BIT27
m33_lpuart9_ipg_doze	BIT28
m33_lpuart10_ipg_doze	BIT29
m33_lpuart11_ipg_doze	BIT30
m33_lpuart12_ipg_doze	BIT31
GPR_CTRL_11 - M33 stop/qreq control	
Bit Field Name	**Bit Position**
m33_lpi2c1_ipg_doze	BIT0
m33_lpi2c2_ipg_doze	BIT1
m33_lpi2c3_ipg_doze	BIT2
m33_lpi2c4_ipg_doze	BIT3
m33_lpi2c5_ipg_doze	BIT4
m33_lpi2c6_ipg_doze	BIT5
m33_lpspi1_ipg_doze	BIT6
m33_lpspi2_ipg_doze	BIT7
m33_lpspi3_ipg_doze	BIT8
m33_lpspi4_ipg_doze	BIT9
m33_lpspi5_ipg_doze	BIT10
m33_lpspi6_ipg_doze	BIT11
m33_sinc1_ipg_doze	BIT12
m33_sinc2_ipg_doze	BIT13
m33_sinc3_ipg_doze	BIT14
m33_mic_ipg_doze	BIT15
GPR_CTRL_12 - M7 stop/qreq control	
Bit Field Name	**Bit Position**
m7_cm7_ipg_stop	BIT0
m7_cm33_ipg_stop	BIT1
m7_edma3_ipg_stop	BIT2
m7_edma4_ipg_stop	BIT3
m7_netc_ipg_stop	BIT4
m7_sim_aon_ipg_stop	BIT8

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m7_adc1_ipg_stop	BIT9
m7_adc2_ipg_stop	BIT10
m7_flexspi1_ipg_stop	BIT11
m7_flexspi2_ipg_stop	BIT12
m7_trdc_ipg_stop	BIT13
m7_semc_ipg_stop	BIT14
m7_iee_ipg_stop	BIT15
m7_gpio1_ipg_stop	BIT16
m7_gpio2_ipg_stop	BIT17
m7_gpio3_ipg_stop	BIT18
m7_gpio4_ipg_stop	BIT19
m7_gpio5_ipg_stop	BIT20
m7_gpio6_ipg_stop	BIT21
m7_flexio1_ipg_stop	BIT22
m7_flexio2_ipg_stop	BIT23
m7_can1_ipg_stop	BIT24
m7_can2_ipg_stop	BIT25
m7_can3_ipg_stop	BIT26
m7_lpuart1_ipg_stop	BIT27
m7_lpuart2_ipg_stop	BIT28
m7_lpuart3_ipg_stop	BIT29
m7_lpuart4_ipg_stop	BIT30
m7_lpuart5_ipg_stop	BIT31
GPR_CTRL_13 - M7 stop/qreq control	
Bit Field Name	**Bit Position**
m7_lpuart6_ipg_stop	BIT0
m7_lpuart7_ipg_stop	BIT1
m7_lpuart8_ipg_stop	BIT2
m7_lpuart9_ipg_stop	BIT3
m7_lpuart10_ipg_stop	BIT4
m7_lpuart11_ipg_stop	BIT5
m7_lpuart12_ipg_stop	BIT6
m7_lpi2c1_ipg_stop	BIT7

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m7_lpi2c2_ipg_stop	BIT8
m7_lpi2c3_ipg_stop	BIT9
m7_lpi2c4_ipg_stop	BIT10
m7_lpi2c5_ipg_stop	BIT11
m7_lpi2c6_ipg_stop	BIT12
m7_lpspi1_ipg_stop	BIT13
m7_lpspi2_ipg_stop	BIT14
m7_lpspi3_ipg_stop	BIT15
m7_lpspi4_ipg_stop	BIT16
m7_lpspi5_ipg_stop	BIT17
m7_lpspi6_ipg_stop	BIT18
m7_sinc1_ipg_stop	BIT19
m7_sinc2_ipg_stop	BIT20
m7_sinc3_ipg_stop	BIT21
m7_sai1_ipg_stop	BIT22
m7_sai2_ipg_stop	BIT23
m7_sai3_ipg_stop	BIT24
m7_sai4_ipg_stop	BIT25
m7_mic_ipg_stop	BIT26
GPR_CTRL_14 - M7 stop/qreq control	
Bit Field Name	**Bit Position**
m7_adc1_ipg_doze	BIT0
m7_adc2_ipg_doze	BIT1
m7_flexspi1_ipg_doze	BIT2
m7_flexspi2_ipg_doze	BIT3
m7_flexio1_ipg_doze	BIT4
m7_flexio2_ipg_doze	BIT5
m7_lpit1_ipg_doze	BIT6
m7_lpit2_ipg_doze	BIT7
m7_lpit3_ipg_doze	BIT8
m7_tpm1_ipg_doze	BIT9
m7_tpm2_ipg_doze	BIT10
m7_tpm3_ipg_doze	BIT11

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m7_tpm4_ipg_doze	BIT12
m7_tpm5_ipg_doze	BIT13
m7_tpm6_ipg_doze	BIT14
m7_gpt1_ipg_doze	BIT15
m7_gpt2_ipg_doze	BIT16
m7_can1_ipg_doze	BIT17
m7_can2_ipg_doze	BIT18
m7_can3_ipg_doze	BIT19
m7_lpuart1_ipg_doze	BIT20
m7_lpuart2_ipg_doze	BIT21
m7_lpuart3_ipg_doze	BIT22
m7_lpuart4_ipg_doze	BIT23
m7_lpuart5_ipg_doze	BIT24
m7_lpuart6_ipg_doze	BIT25
m7_lpuart7_ipg_doze	BIT26
m7_lpuart8_ipg_doze	BIT27
m7_lpuart9_ipg_doze	BIT28
m7_lpuart10_ipg_doze	BIT29
m7_lpuart11_ipg_doze	BIT30
m7_lpuart12_ipg_doze	BIT31
GPR_CTRL_15 - M7 stop/qreq control	
Bit Field Name	**Bit Position**
m7_lpi2c1_ipg_doze	BIT0
m7_lpi2c2_ipg_doze	BIT1
m7_lpi2c3_ipg_doze	BIT2
m7_lpi2c4_ipg_doze	BIT3
m7_lpi2c5_ipg_doze	BIT4
m7_lpi2c6_ipg_doze	BIT5
m7_lpspi1_ipg_doze	BIT6
m7_lpspi2_ipg_doze	BIT7
m7_lpspi3_ipg_doze	BIT8
m7_lpspi4_ipg_doze	BIT9
m7_lpspi5_ipg_doze	BIT10

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m7_lpspi6_ipg_doze	BIT11
m7_sinc1_ipg_doze	BIT12
m7_sinc2_ipg_doze	BIT13
m7_sinc3_ipg_doze	BIT14
m7_mic_ipg_doze	BIT15
GPR_STATUS_0 - M33 stop_ack status	
Bit Field Name	**Bit Position**
m33_ack_cm7	BIT0
m33_ack_cm33	BIT1
m33_ack_edma3	BIT2
m33_ack_edma4	BIT3
m33_ack_netc	BIT4
m33_ack_sim_aon	BIT8
m33_ack_adc1	BIT9
m33_ack_adc2	BIT10
m33_ack_flexspi1	BIT11
m33_ack_flexspi2	BIT12
m33_ack_trdc	BIT13
m33_ack_semc	BIT14
m33_ack_iee	BIT15
m33_ack_gpio1	BIT16
m33_ack_gpio2	BIT17
m33_ack_gpio3	BIT18
m33_ack_gpio4	BIT19
m33_ack_gpio5	BIT20
m33_ack_gpio6	BIT21
m33_ack_flexio1	BIT22
m33_ack_flexio2	BIT23
m33_ack_lpit1	BIT24
m33_ack_lpit2	BIT25
m33_ack_lpit3	BIT26
m33_ack_tpm1	BIT27
m33_ack_tpm2	BIT28

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m33_ack_tpm3	BIT29
m33_ack_tpm4	BIT30
m33_ack_tpm5	BIT31
GPR_STATUS_1 - M33 stop_ack status	
Bit Field Name	**Bit Position**
m33_ack_tpm6	BIT0
m33_ack_gpt1	BIT1
m33_ack_gpt2	BIT2
m33_ack_can1	BIT3
m33_ack_can2	BIT4
m33_ack_can3	BIT5
m33_ack_lpuart1	BIT6
m33_ack_lpuart2	BIT7
m33_ack_lpuart3	BIT8
m33_ack_lpuart4	BIT9
m33_ack_lpuart5	BIT10
m33_ack_lpuart6	BIT11
m33_ack_lpuart7	BIT12
m33_ack_lpuart8	BIT13
m33_ack_lpuart9	BIT14
m33_ack_lpuart10	BIT15
m33_ack_lpuart11	BIT16
m33_ack_lpuart12	BIT17
m33_ack_lpi2c1	BIT18
m33_ack_lpi2c2	BIT19
m33_ack_lpi2c3	BIT20
m33_ack_lpi2c4	BIT21
m33_ack_lpi2c5	BIT22
m33_ack_lpi2c6	BIT23
m33_ack_lpspi1	BIT24
m33_ack_lpspi2	BIT25
m33_ack_lpspi3	BIT26
m33_ack_lpspi4	BIT27

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m33_ack_lpspi5	BIT28
m33_ack_lpspi6	BIT29
m33_ack_sinc1	BIT30
m33_ack_sinc2	BIT31
GPR_STATUS_2 - M33 stop_ack status	
Bit Field Name	**Bit Position**
m33_ack_sinc3	BIT0
m33_ack_sai1	BIT1
m33_ack_sai2	BIT2
m33_ack_sai3	BIT3
m33_ack_sai4	BIT4
m33_ack_mic	BIT5
GPR_STATUS_4 - M7 stop_ack status	
Bit Field Name	**Bit Position**
m7_ack_cm7	BIT0
m7_ack_cm33	BIT1
m7_ack_edma3	BIT2
m7_ack_edma4	BIT3
m7_ack_netc	BIT4
m7_ack_sim_aon	BIT8
m7_ack_adc1	BIT9
m7_ack_adc2	BIT10
m7_ack_flexspi1	BIT11
m7_ack_flexspi2	BIT12
m7_ack_trdc	BIT13
m7_ack_semc	BIT14
m7_ack_iee	BIT15
m7_ack_gpio1	BIT16
m7_ack_gpio2	BIT17
m7_ack_gpio3	BIT18
m7_ack_gpio4	BIT19
m7_ack_gpio5	BIT20
m7_ack_gpio6	BIT21

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m7_ack_flexio1	BIT22
m7_ack_flexio2	BIT23
m7_ack_lpit1	BIT24
m7_ack_lpit2	BIT25
m7_ack_lpit3	BIT26
m7_ack_tpm1	BIT27
m7_ack_tpm2	BIT28
m7_ack_tpm3	BIT29
m7_ack_tpm4	BIT30
m7_ack_tpm5	BIT31
GPR_STATUS_5 - M7 stop_ack status	
Bit Field Name	**Bit Position**
m7_ack_tpm6	BIT0
m7_ack_gpt1	BIT1
m7_ack_gpt2	BIT2
m7_ack_can1	BIT3
m7_ack_can2	BIT4
m7_ack_can3	BIT5
m7_ack_lpuart1	BIT6
m7_ack_lpuart2	BIT7
m7_ack_lpuart3	BIT8
m7_ack_lpuart4	BIT9
m7_ack_lpuart5	BIT10
m7_ack_lpuart6	BIT11
m7_ack_lpuart7	BIT12
m7_ack_lpuart8	BIT13
m7_ack_lpuart9	BIT14
m7_ack_lpuart10	BIT15
m7_ack_lpuart11	BIT16
m7_ack_lpuart12	BIT17
m7_ack_lpi2c1	BIT18
m7_ack_lpi2c2	BIT19
m7_ack_lpi2c3	BIT20

Table continues on the next page...

Table 141. CCM GPR shared registers (continued)

m7_ack_lpi2c4	BIT21
m7_ack_lpi2c5	BIT22
m7_ack_lpi2c6	BIT23
m7_ack_lpspi1	BIT24
m7_ack_lpspi2	BIT25
m7_ack_lpspi3	BIT26
m7_ack_lpspi4	BIT27
m7_ack_lpspi5	BIT28
m7_ack_lpspi6	BIT29
m7_ack_sinc1	BIT30
m7_ack_sinc2	BIT31
GPR_STATUS_6 - M7 stop_ack status	
Bit Field Name	**Bit Position**
m7_ack_sinc3	BIT0
m7_ack_sai1	BIT1
m7_ack_sai2	BIT2
m7_ack_sai3	BIT3
m7_ack_sai4	BIT4
m7_ack_mic	BIT5

20.4.6.1 GPR Shared Control Registers (GPR_SHAREDn)

The following table shows the bitfields for each of the CCM GPR shared control registers.

Table 142. CCM GPR shared control registers

GPR_SHARED2 - M33 mask control	
Bit Field Name	**Bit Position**
m33_mask_cm7	BIT0
m33_mask_cm33	BIT1
m33_mask_edma3	BIT2
m33_mask_edma4	BIT3
m33_mask_netc	BIT4
m33_mask_sim_aon	BIT8
m33_mask_adc1	BIT9
m33_mask_adc2	BIT10

Table continues on the next page...

Table 142. CCM GPR shared control registers (continued)

m33_mask_flexspi1	BIT11
m33_mask_flexspi2	BIT12
m33_mask_trdc	BIT13
m33_mask_semc	BIT14
m33_mask_iee	BIT15
m33_mask_gpio1	BIT16
m33_mask_gpio2	BIT17
m33_mask_gpio3	BIT18
m33_mask_gpio4	BIT19
m33_mask_gpio5	BIT20
m33_mask_gpio6	BIT21
m33_mask_flexio1	BIT22
m33_mask_flexio2	BIT23
m33_mask_lpit1	BIT24
m33_mask_lpit2	BIT25
m33_mask_lpit3	BIT26
m33_mask_tpm1	BIT27
m33_mask_tpm2	BIT28
m33_mask_tpm3	BIT29
m33_mask_tpm4	BIT30
m33_mask_tpm5	BIT31
GPR_SHARED3 - M33 mask control	
Bit Field Name	**Bit Position**
m33_mask_tpm6	BIT0
m33_mask_gpt1	BIT1
m33_mask_gpt2	BIT2
m33_mask_can1	BIT3
m33_mask_can2	BIT4
m33_mask_can3	BIT5
m33_mask_lpuart1	BIT6
m33_mask_lpuart2	BIT7
m33_mask_lpuart3	BIT8
m33_mask_lpuart4	BIT9

Table continues on the next page...

Table 142. CCM GPR shared control registers (continued)

m33_mask_lpuart5	BIT10
m33_mask_lpuart6	BIT11
m33_mask_lpuart7	BIT12
m33_mask_lpuart8	BIT13
m33_mask_lpuart9	BIT14
m33_mask_lpuart10	BIT15
m33_mask_lpuart11	BIT16
m33_mask_lpuart12	BIT17
m33_mask_lpi2c1	BIT18
m33_mask_lpi2c2	BIT19
m33_mask_lpi2c3	BIT20
m33_mask_lpi2c4	BIT21
m33_mask_lpi2c5	BIT22
m33_mask_lpi2c6	BIT23
m33_mask_lpspi1	BIT24
m33_mask_lpspi2	BIT25
m33_mask_lpspi3	BIT26
m33_mask_lpspi4	BIT27
m33_mask_lpspi5	BIT28
m33_mask_lpspi6	BIT29
m33_mask_sinc1	BIT30
m33_mask_sinc2	BIT31
GPR_SHARED4 - M33 mask control	
Bit Field Name	**Bit Position**
m33_mask_sinc3	BIT0
m33_mask_sai1	BIT1
m33_mask_sai2	BIT2
m33_mask_sai3	BIT3
m33_mask_sai4	BIT4
m33_mask_mic	BIT5
GPR_SHARED5 - M7 mask control	
Bit Field Name	**Bit Position**
m7_mask_cm7	BIT0

Table continues on the next page...

Table 142. CCM GPR shared control registers (continued)

m7_mask_cm33	BIT1
m7_mask_edma3	BIT2
m7_mask_edma4	BIT3
m7_mask_netc	BIT4
m7_mask_sim_aon	BIT8
m7_mask_adc1	BIT9
m7_mask_adc2	BIT10
m7_mask_flexspi1	BIT11
m7_mask_flexspi2	BIT12
m7_mask_trdc	BIT13
m7_mask_semc	BIT14
m7_mask_iee	BIT15
m7_mask_gpio1	BIT16
m7_mask_gpio2	BIT17
m7_mask_gpio3	BIT18
m7_mask_gpio4	BIT19
m7_mask_gpio5	BIT20
m7_mask_gpio6	BIT21
m7_mask_flexio1	BIT22
m7_mask_flexio2	BIT23
m7_mask_lpit1	BIT24
m7_mask_lpit2	BIT25
m7_mask_lpit3	BIT26
m7_mask_tpm1	BIT27
m7_mask_tpm2	BIT28
m7_mask_tpm3	BIT29
m7_mask_tpm4	BIT30
m7_mask_tpm5	BIT31
GPR_SHARED6 - M7 mask control	
Bit Field Name	**Bit Position**
m7_mask_tpm6	BIT0
m7_mask_gpt1	BIT1
m7_mask_gpt2	BIT2

Table continues on the next page...

Table 142. CCM GPR shared control registers (continued)

m7_mask_can1	BIT3
m7_mask_can2	BIT4
m7_mask_can3	BIT5
m7_mask_lpuart1	BIT6
m7_mask_lpuart2	BIT7
m7_mask_lpuart3	BIT8
m7_mask_lpuart4	BIT9
m7_mask_lpuart5	BIT10
m7_mask_lpuart6	BIT11
m7_mask_lpuart7	BIT12
m7_mask_lpuart8	BIT13
m7_mask_lpuart9	BIT14
m7_mask_lpuart10	BIT15
m7_mask_lpuart11	BIT16
m7_mask_lpuart12	BIT17
m7_mask_lpi2c1	BIT18
m7_mask_lpi2c2	BIT19
m7_mask_lpi2c3	BIT20
m7_mask_lpi2c4	BIT21
m7_mask_lpi2c5	BIT22
m7_mask_lpi2c6	BIT23
m7_mask_lpspi1	BIT24
m7_mask_lpspi2	BIT25
m7_mask_lpspi3	BIT26
m7_mask_lpspi4	BIT27
m7_mask_lpspi5	BIT28
m7_mask_lpspi6	BIT29
m7_mask_sinc1	BIT30
m7_mask_sinc2	BIT31
GPR_SHARED7 - M7 mask control	
Bit Field Name	**Bit Position**
m7_mask_sinc3	BIT0
m7_mask_sai1	BIT1

Table continues on the next page...

Table 142. CCM GPR shared control registers (continued)

m7_mask_sai2	BIT2
m7_mask_sai3	BIT3
m7_mask_sai4	BIT4
m7_mask_mic	BIT5
GPR_SHARED8 - M33 stop/qreq control	
Bit Field Name	**Bit Position**
m33_cm7_ipg_stop	BIT0
m33_cm33_ipg_stop	BIT1
m33_edma3_ipg_stop	BIT2
m33_edma4_ipg_stop	BIT3
m33_netc_ipg_stop	BIT4
m33_sim_aon_ipg_stop	BIT8
m33_adc1_ipg_stop	BIT9
m33_adc2_ipg_stop	BIT10
m33_flexspi1_ipg_stop	BIT11
m33_flexspi2_ipg_stop	BIT12
m33_trdc_ipg_stop	BIT13
m33_semc_ipg_stop	BIT14
m33_iee_ipg_stop	BIT15
m33_gpio1_ipg_stop	BIT16
m33_gpio2_ipg_stop	BIT17
m33_gpio3_ipg_stop	BIT18
m33_gpio4_ipg_stop	BIT19
m33_gpio5_ipg_stop	BIT20
m33_gpio6_ipg_stop	BIT21
m33_flexio1_ipg_stop	BIT22
m33_flexio2_ipg_stop	BIT23
m33_can1_ipg_stop	BIT24
m33_can2_ipg_stop	BIT25
m33_can3_ipg_stop	BIT26
m33_lpuart1_ipg_stop	BIT27
m33_lpuart2_ipg_stop	BIT28
m33_lpuart3_ipg_stop	BIT29

Table continues on the next page...

Table 142. CCM GPR shared control registers (continued)

m33_lpuart4_ipg_stop	BIT30
m33_lpuart5_ipg_stop	BIT31
GPR_SHARED9 - M33 stop/qreq control	
Bit Field Name	**Bit Position**
m33_lpuart6_ipg_stop	BIT0
m33_lpuart7_ipg_stop	BIT1
m33_lpuart8_ipg_stop	BIT2
m33_lpuart9_ipg_stop	BIT3
m33_lpuart10_ipg_stop	BIT4
m33_lpuart11_ipg_stop	BIT5
m33_lpuart12_ipg_stop	BIT6
Reserved	BITS 3-6
m33_lpi2c1_ipg_stop	BIT7
m33_lpi2c2_ipg_stop	BIT8
m33_lpi2c3_ipg_stop	BIT9
m33_lpi2c4_ipg_stop	BIT10
m33_lpi2c5_ipg_stop	BIT11
m33_lpi2c6_ipg_stop	BIT12
m33_lpspi1_ipg_stop	BIT13
m33_lpspi2_ipg_stop	BIT14
m33_lpspi3_ipg_stop	BIT15
m33_lpspi4_ipg_stop	BIT16
m33_lpspi5_ipg_stop	BIT17
m33_lpspi6_ipg_stop	BIT18
m33_sinc1_ipg_stop	BIT19
m33_sinc2_ipg_stop	BIT20
m33_sinc3_ipg_stop	BIT21
m33_sai1_ipg_stop	BIT22
m33_sai2_ipg_stop	BIT23
m33_sai3_ipg_stop	BIT24
m33_sai4_ipg_stop	BIT25
m33_mic_ipg_stop	BIT26
GPR_SHARED10 - M33 stop/qreq control	

Table continues on the next page...

Table 142. CCM GPR shared control registers (continued)

Bit Field Name	Bit Position
m33_adc1_ipg_doze	BIT0
m33_adc2_ipg_doze	BIT1
m33_flexspi1_ipg_doze	BIT2
m33_flexspi2_ipg_doze	BIT3
m33_flexio1_ipg_doze	BIT4
m33_flexio2_ipg_doze	BIT5
m33_lpit1_ipg_doze	BIT6
m33_lpit2_ipg_doze	BIT7
m33_lpit3_ipg_doze	BIT8
m33_tpm1_ipg_doze	BIT9
m33_tpm2_ipg_doze	BIT10
m33_tpm3_ipg_doze	BIT11
m33_tpm4_ipg_doze	BIT12
m33_tpm5_ipg_doze	BIT13
m33_tpm6_ipg_doze	BIT14
m33_gpt1_ipg_doze	BIT15
m33_gpt2_ipg_doze	BIT16
m33_can1_ipg_doze	BIT17
m33_can2_ipg_doze	BIT18
m33_can3_ipg_doze	BIT19
m33_lpuart1_ipg_doze	BIT20
m33_lpuart2_ipg_doze	BIT21
m33_lpuart3_ipg_doze	BIT22
m33_lpuart4_ipg_doze	BIT23
m33_lpuart5_ipg_doze	BIT24
m33_lpuart6_ipg_doze	BIT25
m33_lpuart7_ipg_doze	BIT26
m33_lpuart8_ipg_doze	BIT27
m33_lpuart9_ipg_doze	BIT28
m33_lpuart10_ipg_doze	BIT29
m33_lpuart11_ipg_doze	BIT30
m33_lpuart12_ipg_doze	BIT31

Table continues on the next page...

Table 142. CCM GPR shared control registers (continued)

Reserved	BITS 28-31
GPR_SHARED11 - M33 stop/qreq control	
Bit Field Name	**Bit Position**
m33_lpi2c1_ipg_doze	BIT0
m33_lpi2c2_ipg_doze	BIT1
m33_lpi2c3_ipg_doze	BIT2
m33_lpi2c4_ipg_doze	BIT3
m33_lpi2c5_ipg_doze	BIT4
m33_lpi2c6_ipg_doze	BIT5
m33_lpspi1_ipg_doze	BIT6
m33_lpspi2_ipg_doze	BIT7
m33_lpspi3_ipg_doze	BIT8
m33_lpspi4_ipg_doze	BIT9
m33_lpspi5_ipg_doze	BIT10
m33_lpspi6_ipg_doze	BIT11
m33_sinc1_ipg_doze	BIT12
m33_sinc2_ipg_doze	BIT13
m33_sinc3_ipg_doze	BIT14
m33_mic_ipg_doze	BIT15
GPR_SHARED12 - M7 stop/qreq control	
Bit Field Name	**Bit Position**
m7_cm7_ipg_stop	BIT0
m7_cm33_ipg_stop	BIT1
m7_edma3_ipg_stop	BIT2
m7_edma4_ipg_stop	BIT3
m7_netc_ipg_stop	BIT4
m7_sim_aon_ipg_stop	BIT8
m7_adc1_ipg_stop	BIT9
m7_adc2_ipg_stop	BIT10
m7_flexspi1_ipg_stop	BIT11
m7_flexspi2_ipg_stop	BIT12
m7_trdc_ipg_stop	BIT13
m7_semc_ipg_stop	BIT14

Table continues on the next page...

Table 142. CCM GPR shared control registers (continued)

m7_iee_ipg_stop	BIT15
m7_gpio1_ipg_stop	BIT16
m7_gpio2_ipg_stop	BIT17
m7_gpio3_ipg_stop	BIT18
m7_gpio4_ipg_stop	BIT19
m7_gpio5_ipg_stop	BIT20
m7_gpio6_ipg_stop	BIT21
m7_flexio1_ipg_stop	BIT22
m7_flexio2_ipg_stop	BIT23
m7_can1_ipg_stop	BIT24
m7_can2_ipg_stop	BIT25
m7_can3_ipg_stop	BIT26
m7_lpuart1_ipg_stop	BIT27
m7_lpuart2_ipg_stop	BIT28
m7_lpuart3_ipg_stop	BIT29
m7_lpuart4_ipg_stop	BIT30
m7_lpuart5_ipg_stop	BIT31
GPR_SHARED13 - M7 stop/qreq control	
Bit Field Name	**Bit Position**
m7_lpuart6_ipg_stop	BIT0
m7_lpuart7_ipg_stop	BIT1
m7_lpuart8_ipg_stop	BIT2
m7_lpuart9_ipg_stop	BIT3
m7_lpuart10_ipg_stop	BIT4
m7_lpuart11_ipg_stop	BIT5
m7_lpuart12_ipg_stop	BIT6
m7_lpi2c1_ipg_stop	BIT7
m7_lpi2c2_ipg_stop	BIT8
m7_lpi2c3_ipg_stop	BIT9
m7_lpi2c4_ipg_stop	BIT10
m7_lpi2c5_ipg_stop	BIT11
m7_lpi2c6_ipg_stop	BIT12
m7_lpspi1_ipg_stop	BIT13

Table continues on the next page...

Table 142. CCM GPR shared control registers (continued)

m7_lpspi2_ipg_stop	BIT14
m7_lpspi3_ipg_stop	BIT15
m7_lpspi4_ipg_stop	BIT16
m7_lpspi5_ipg_stop	BIT17
m7_lpspi6_ipg_stop	BIT18
m7_sinc1_ipg_stop	BIT19
m7_sinc2_ipg_stop	BIT20
m7_sinc3_ipg_stop	BIT21
m7_sai1_ipg_stop	BIT22
m7_sai2_ipg_stop	BIT23
m7_sai3_ipg_stop	BIT24
m7_sai4_ipg_stop	BIT25
m7_mic_ipg_stop	BIT26
GPR_SHARED14 - M7 stop/qreq control	
Bit Field Name	**Bit Position**
m7_adc1_ipg_doze	BIT0
m7_adc2_ipg_doze	BIT1
m7_flexspi1_ipg_doze	BIT2
m7_flexspi2_ipg_doze	BIT3
m7_flexio1_ipg_doze	BIT4
m7_flexio2_ipg_doze	BIT5
m7_lpit1_ipg_doze	BIT6
m7_lpit2_ipg_doze	BIT7
m7_lpit3_ipg_doze	BIT8
m7_tpm1_ipg_doze	BIT9
m7_tpm2_ipg_doze	BIT10
m7_tpm3_ipg_doze	BIT11
m7_tpm4_ipg_doze	BIT12
m7_tpm5_ipg_doze	BIT13
m7_tpm6_ipg_doze	BIT14
m7_gpt1_ipg_doze	BIT15
m7_gpt2_ipg_doze	BIT16
m7_can1_ipg_doze	BIT17

Table continues on the next page...

Table 142. CCM GPR shared control registers (continued)

m7_can2_ipg_doze	BIT18
m7_can3_ipg_doze	BIT19
m7_lpuart1_ipg_doze	BIT20
m7_lpuart2_ipg_doze	BIT21
m7_lpuart3_ipg_doze	BIT22
m7_lpuart4_ipg_doze	BIT23
m7_lpuart5_ipg_doze	BIT24
m7_lpuart6_ipg_doze	BIT25
m7_lpuart7_ipg_doze	BIT26
m7_lpuart8_ipg_doze	BIT27
m7_lpuart9_ipg_doze	BIT28
m7_lpuart10_ipg_doze	BIT29
m7_lpuart11_ipg_doze	BIT30
m7_lpuart12_ipg_doze	BIT31
GPR_SHARED15 - M7 stop/qreq control	
Bit Field Name	**Bit Position**
m7_lpi2c1_ipg_doze	BIT0
m7_lpi2c2_ipg_doze	BIT1
m7_lpi2c3_ipg_doze	BIT2
m7_lpi2c4_ipg_doze	BIT3
m7_lpi2c5_ipg_doze	BIT4
m7_lpi2c6_ipg_doze	BIT5
m7_lpspi1_ipg_doze	BIT6
m7_lpspi2_ipg_doze	BIT7
m7_lpspi3_ipg_doze	BIT8
m7_lpspi4_ipg_doze	BIT9
m7_lpspi5_ipg_doze	BIT10
m7_lpspi6_ipg_doze	BIT11
m7_sinc1_ipg_doze	BIT12
m7_sinc2_ipg_doze	BIT13
m7_sinc3_ipg_doze	BIT14
m7_mic_ipg_doze	BIT15

20.4.6.2 GPR Shared Status Registers (GPR_SHARED_STATUSn)

The following table shows the bitfields for each of the CCM GPR shared status registers.

Table 143. CCM GPR shared status registers

GPR_SHARED_STATUS0 - M33 stop_ack status	
Bit Field Name	**Bit Position**
m33_ack_cm7	BIT0
m33_ack_cm33	BIT1
m33_ack_edma3	BIT2
m33_ack_edma4	BIT3
m33_ack_netc	BIT4
m33_ack_sim_aon	BIT8
m33_ack_adc1	BIT9
m33_ack_adc2	BIT10
m33_ack_flexspi1	BIT11
m33_ack_flexspi2	BIT12
m33_ack_trdc	BIT13
m33_ack_semc	BIT14
m33_ack_iee	BIT15
m33_ack_gpio1	BIT16
m33_ack_gpio2	BIT17
m33_ack_gpio3	BIT18
m33_ack_gpio4	BIT19
m33_ack_gpio5	BIT20
m33_ack_gpio6	BIT21
m33_ack_flexio1	BIT22
m33_ack_flexio2	BIT23
m33_ack_lpit1	BIT24
m33_ack_lpit2	BIT25
m33_ack_lpit3	BIT26
m33_ack_tpm1	BIT27
m33_ack_tpm2	BIT28
m33_ack_tpm3	BIT29
m33_ack_tpm4	BIT30
m33_ack_tpm5	BIT31

Table continues on the next page...

Table 143. CCM GPR shared status registers (continued)

GPR_SHARED_STATUS1 - M33 stop_ack status	
Bit Field Name	**Bit Position**
m33_ack_tpm6	BIT0
m33_ack_gpt1	BIT1
m33_ack_gpt2	BIT2
m33_ack_can1	BIT3
m33_ack_can2	BIT4
m33_ack_can3	BIT5
m33_ack_lpuart1	BIT6
m33_ack_lpuart2	BIT7
m33_ack_lpuart3	BIT8
m33_ack_lpuart4	BIT9
m33_ack_lpuart5	BIT10
m33_ack_lpuart6	BIT11
m33_ack_lpuart7	BIT12
m33_ack_lpuart8	BIT13
m33_ack_lpuart9	BIT14
m33_ack_lpuart10	BIT15
m33_ack_lpuart11	BIT16
m33_ack_lpuart12	BIT17
m33_ack_lpi2c1	BIT18
m33_ack_lpi2c2	BIT19
m33_ack_lpi2c3	BIT20
m33_ack_lpi2c4	BIT21
m33_ack_lpi2c5	BIT22
Reserved	BIT22
m33_ack_lpi2c6	BIT23
m33_ack_lpspi1	BIT24
m33_ack_lpspi2	BIT25
m33_ack_lpspi3	BIT26
m33_ack_lpspi4	BIT27
m33_ack_lpspi5	BIT28
m33_ack_lpspi6	BIT29

Table continues on the next page...

Table 143. CCM GPR shared status registers (continued)

m33_ack_sinc1	BIT30
m33_ack_sinc2	BIT31
GPR_SHARED_STATUS2 - M33 stop_ack status	
Bit Field Name	**Bit Position**
m33_ack_sinc3	BIT0
m33_ack_sai1	BIT1
m33_ack_sai2	BIT2
m33_ack_sai3	BIT3
m33_ack_sai4	BIT4
m33_ack_mic	BIT5
GPR_SHARED_STATUS4 - M7 stop_ack status	
Bit Field Name	**Bit Position**
m7_ack_cm7	BIT0
m7_ack_cm33	BIT1
m7_ack_edma3	BIT2
m7_ack_edma4	BIT3
m7_ack_netc	BIT4
m7_ack_sim_aon	BIT8
m7_ack_adc1	BIT9
m7_ack_adc2	BIT10
m7_ack_flexspi1	BIT11
m7_ack_flexspi2	BIT12
m7_ack_trdc	BIT13
m7_ack_semc	BIT14
m7_ack_iee	BIT15
m7_ack_gpio1	BIT16
m7_ack_gpio2	BIT17
m7_ack_gpio3	BIT18
m7_ack_gpio4	BIT19
m7_ack_gpio5	BIT20
m7_ack_gpio6	BIT21
m7_ack_flexio1	BIT22
m7_ack_flexio2	BIT23

Table continues on the next page...

Table 143. CCM GPR shared status registers (continued)

m7_ack_lpit1	BIT24
m7_ack_lpit2	BIT25
m7_ack_lpit3	BIT26
m7_ack_tpm1	BIT27
m7_ack_tpm2	BIT28
m7_ack_tpm3	BIT29
m7_ack_tpm4	BIT30
m7_ack_tpm5	BIT31
GPR_SHARED_STATUS5 - M7 stop_ack status	
Bit Field Name	**Bit Position**
m7_ack_tpm6	BIT0
m7_ack_gpt1	BIT1
m7_ack_gpt2	BIT2
m7_ack_can1	BIT3
m7_ack_can2	BIT4
m7_ack_can3	BIT5
m7_ack_lpuart1	BIT6
m7_ack_lpuart2	BIT7
m7_ack_lpuart3	BIT8
m7_ack_lpuart4	BIT9
m7_ack_lpuart5	BIT10
m7_ack_lpuart6	BIT11
m7_ack_lpuart7	BIT12
m7_ack_lpuart8	BIT13
m7_ack_lpuart9	BIT14
m7_ack_lpuart10	BIT15
m7_ack_lpuart11	BIT16
m7_ack_lpuart12	BIT17
m7_ack_lpi2c1	BIT18
m7_ack_lpi2c2	BIT19
m7_ack_lpi2c3	BIT20
m7_ack_lpi2c4	BIT21
m7_ack_lpi2c5	BIT22

Table continues on the next page...

Table 143. CCM GPR shared status registers (continued)

m7_ack_lpi2c6	BIT23
m7_ack_lpspi1	BIT24
m7_ack_lpspi2	BIT25
m7_ack_lpspi3	BIT26
m7_ack_lpspi4	BIT27
m7_ack_lpspi5	BIT28
m7_ack_lpspi6	BIT29
m7_ack_sinc1	BIT30
m7_ack_sinc2	BIT31
GPR_SHARED_STATUS6 - M7 stop_ack status	
Bit Field Name	**Bit Position**
m7_ack_sinc3	BIT0
m7_ack_sai1	BIT1
m7_ack_sai2	BIT2
m7_ack_sai3	BIT3
m7_ack_sai4	BIT4
m33_ack_sai4	BIT4
m7_ack_mic	BIT5

20.5 Reset

The reset of CCM will be released along with system early reset and clocks are provided during system reset. For each power domain, early reset of the power domain in which that clock gate resides, will be fed for correct power up and reset flow.

20.6 CCM Memory Map and Register definition

20.6.1 CCM register descriptions

20.6.1.1 D_IP_MXRTC_CCM_SYN memory map

CCM base address: 4445_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Clock Root Control Register (CLOCK_ROOT0_CONTROL)	32	RW	0000_0000h
4h	Clock Root Control Register (CLOCK_ROOT0_CONTROL_SET)	32	RW	0000_0000h
8h	Clock Root Control Register (CLOCK_ROOT0_CONTROL_CLR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
Ch	Clock Root Control Register (CLOCK_ROOT0_CONTROL_TOG)	32	RW	0000_0000h
20h	Clock root working status (CLOCK_ROOT0_STATUS0)	32	R	0000_0000h
30h	Clock root access control (CLOCK_ROOT0_AUTHEN)	32	RW	FFFF_0000h
80h	Clock Root Control Register (CLOCK_ROOT1_CONTROL)	32	RW	0000_0000h
84h	Clock Root Control Register (CLOCK_ROOT1_CONTROL_SET)	32	RW	0000_0000h
88h	Clock Root Control Register (CLOCK_ROOT1_CONTROL_CLR)	32	RW	0000_0000h
8Ch	Clock Root Control Register (CLOCK_ROOT1_CONTROL_TOG)	32	RW	0000_0000h
A0h	Clock root working status (CLOCK_ROOT1_STATUS0)	32	R	0000_0000h
B0h	Clock root access control (CLOCK_ROOT1_AUTHEN)	32	RW	FFFF_0000h
100h	Clock Root Control Register (CLOCK_ROOT2_CONTROL)	32	RW	0000_0000h
104h	Clock Root Control Register (CLOCK_ROOT2_CONTROL_SET)	32	RW	0000_0000h
108h	Clock Root Control Register (CLOCK_ROOT2_CONTROL_CLR)	32	RW	0000_0000h
10Ch	Clock Root Control Register (CLOCK_ROOT2_CONTROL_TOG)	32	RW	0000_0000h
120h	Clock root working status (CLOCK_ROOT2_STATUS0)	32	R	0000_0000h
130h	Clock root access control (CLOCK_ROOT2_AUTHEN)	32	RW	FFFF_0000h
180h	Clock Root Control Register (CLOCK_ROOT3_CONTROL)	32	RW	0000_0000h
184h	Clock Root Control Register (CLOCK_ROOT3_CONTROL_SET)	32	RW	0000_0000h
188h	Clock Root Control Register (CLOCK_ROOT3_CONTROL_CLR)	32	RW	0000_0000h
18Ch	Clock Root Control Register (CLOCK_ROOT3_CONTROL_TOG)	32	RW	0000_0000h
1A0h	Clock root working status (CLOCK_ROOT3_STATUS0)	32	R	0000_0000h
1B0h	Clock root access control (CLOCK_ROOT3_AUTHEN)	32	RW	FFFF_0000h
200h	Clock Root Control Register (CLOCK_ROOT4_CONTROL)	32	RW	0000_0000h
204h	Clock Root Control Register (CLOCK_ROOT4_CONTROL_SET)	32	RW	0000_0000h
208h	Clock Root Control Register (CLOCK_ROOT4_CONTROL_CLR)	32	RW	0000_0000h
20Ch	Clock Root Control Register (CLOCK_ROOT4_CONTROL_TOG)	32	RW	0000_0000h
220h	Clock root working status (CLOCK_ROOT4_STATUS0)	32	R	0000_0000h
230h	Clock root access control (CLOCK_ROOT4_AUTHEN)	32	RW	FFFF_0000h
280h	Clock Root Control Register (CLOCK_ROOT5_CONTROL)	32	RW	0000_0000h
284h	Clock Root Control Register (CLOCK_ROOT5_CONTROL_SET)	32	RW	0000_0000h
288h	Clock Root Control Register (CLOCK_ROOT5_CONTROL_CLR)	32	RW	0000_0000h
28Ch	Clock Root Control Register (CLOCK_ROOT5_CONTROL_TOG)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2A0h	Clock root working status (CLOCK_ROOT5_STATUS0)	32	R	0000_0000h
2B0h	Clock root access control (CLOCK_ROOT5_AUTHEN)	32	RW	FFFF_0000h
300h	Clock Root Control Register (CLOCK_ROOT6_CONTROL)	32	RW	0000_0000h
304h	Clock Root Control Register (CLOCK_ROOT6_CONTROL_SET)	32	RW	0000_0000h
308h	Clock Root Control Register (CLOCK_ROOT6_CONTROL_CLR)	32	RW	0000_0000h
30Ch	Clock Root Control Register (CLOCK_ROOT6_CONTROL_TOG)	32	RW	0000_0000h
320h	Clock root working status (CLOCK_ROOT6_STATUS0)	32	R	0000_0000h
330h	Clock root access control (CLOCK_ROOT6_AUTHEN)	32	RW	FFFF_0000h
380h	Clock Root Control Register (CLOCK_ROOT7_CONTROL)	32	RW	0000_0000h
384h	Clock Root Control Register (CLOCK_ROOT7_CONTROL_SET)	32	RW	0000_0000h
388h	Clock Root Control Register (CLOCK_ROOT7_CONTROL_CLR)	32	RW	0000_0000h
38Ch	Clock Root Control Register (CLOCK_ROOT7_CONTROL_TOG)	32	RW	0000_0000h
3A0h	Clock root working status (CLOCK_ROOT7_STATUS0)	32	R	0000_0000h
3B0h	Clock root access control (CLOCK_ROOT7_AUTHEN)	32	RW	FFFF_0000h
400h	Clock Root Control Register (CLOCK_ROOT8_CONTROL)	32	RW	0000_0000h
404h	Clock Root Control Register (CLOCK_ROOT8_CONTROL_SET)	32	RW	0000_0000h
408h	Clock Root Control Register (CLOCK_ROOT8_CONTROL_CLR)	32	RW	0000_0000h
40Ch	Clock Root Control Register (CLOCK_ROOT8_CONTROL_TOG)	32	RW	0000_0000h
420h	Clock root working status (CLOCK_ROOT8_STATUS0)	32	R	0000_0000h
430h	Clock root access control (CLOCK_ROOT8_AUTHEN)	32	RW	FFFF_0000h
480h	Clock Root Control Register (CLOCK_ROOT9_CONTROL)	32	RW	0000_0000h
484h	Clock Root Control Register (CLOCK_ROOT9_CONTROL_SET)	32	RW	0000_0000h
488h	Clock Root Control Register (CLOCK_ROOT9_CONTROL_CLR)	32	RW	0000_0000h
48Ch	Clock Root Control Register (CLOCK_ROOT9_CONTROL_TOG)	32	RW	0000_0000h
4A0h	Clock root working status (CLOCK_ROOT9_STATUS0)	32	R	0000_0000h
4B0h	Clock root access control (CLOCK_ROOT9_AUTHEN)	32	RW	FFFF_0000h
500h	Clock Root Control Register (CLOCK_ROOT10_CONTROL)	32	RW	0000_0000h
504h	Clock Root Control Register (CLOCK_ROOT10_CONTROL_SET)	32	RW	0000_0000h
508h	Clock Root Control Register (CLOCK_ROOT10_CONTROL_CLR)	32	RW	0000_0000h
50Ch	Clock Root Control Register (CLOCK_ROOT10_CONTROL_TOG)	32	RW	0000_0000h
520h	Clock root working status (CLOCK_ROOT10_STATUS0)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
530h	Clock root access control (CLOCK_ROOT10_AUTHEN)	32	RW	FFFF_0000h
580h	Clock Root Control Register (CLOCK_ROOT11_CONTROL)	32	RW	0000_0000h
584h	Clock Root Control Register (CLOCK_ROOT11_CONTROL_SET)	32	RW	0000_0000h
588h	Clock Root Control Register (CLOCK_ROOT11_CONTROL_CLR)	32	RW	0000_0000h
58Ch	Clock Root Control Register (CLOCK_ROOT11_CONTROL_TOG)	32	RW	0000_0000h
5A0h	Clock root working status (CLOCK_ROOT11_STATUS0)	32	R	0000_0000h
5B0h	Clock root access control (CLOCK_ROOT11_AUTHEN)	32	RW	FFFF_0000h
600h	Clock Root Control Register (CLOCK_ROOT12_CONTROL)	32	RW	0000_0000h
604h	Clock Root Control Register (CLOCK_ROOT12_CONTROL_SET)	32	RW	0000_0000h
608h	Clock Root Control Register (CLOCK_ROOT12_CONTROL_CLR)	32	RW	0000_0000h
60Ch	Clock Root Control Register (CLOCK_ROOT12_CONTROL_TOG)	32	RW	0000_0000h
620h	Clock root working status (CLOCK_ROOT12_STATUS0)	32	R	0000_0000h
630h	Clock root access control (CLOCK_ROOT12_AUTHEN)	32	RW	FFFF_0000h
680h	Clock Root Control Register (CLOCK_ROOT13_CONTROL)	32	RW	0000_0000h
684h	Clock Root Control Register (CLOCK_ROOT13_CONTROL_SET)	32	RW	0000_0000h
688h	Clock Root Control Register (CLOCK_ROOT13_CONTROL_CLR)	32	RW	0000_0000h
68Ch	Clock Root Control Register (CLOCK_ROOT13_CONTROL_TOG)	32	RW	0000_0000h
6A0h	Clock root working status (CLOCK_ROOT13_STATUS0)	32	R	0000_0000h
6B0h	Clock root access control (CLOCK_ROOT13_AUTHEN)	32	RW	FFFF_0000h
700h	Clock Root Control Register (CLOCK_ROOT14_CONTROL)	32	RW	0000_0000h
704h	Clock Root Control Register (CLOCK_ROOT14_CONTROL_SET)	32	RW	0000_0000h
708h	Clock Root Control Register (CLOCK_ROOT14_CONTROL_CLR)	32	RW	0000_0000h
70Ch	Clock Root Control Register (CLOCK_ROOT14_CONTROL_TOG)	32	RW	0000_0000h
720h	Clock root working status (CLOCK_ROOT14_STATUS0)	32	R	0000_0000h
730h	Clock root access control (CLOCK_ROOT14_AUTHEN)	32	RW	FFFF_0000h
780h	Clock Root Control Register (CLOCK_ROOT15_CONTROL)	32	RW	0000_0000h
784h	Clock Root Control Register (CLOCK_ROOT15_CONTROL_SET)	32	RW	0000_0000h
788h	Clock Root Control Register (CLOCK_ROOT15_CONTROL_CLR)	32	RW	0000_0000h
78Ch	Clock Root Control Register (CLOCK_ROOT15_CONTROL_TOG)	32	RW	0000_0000h
7A0h	Clock root working status (CLOCK_ROOT15_STATUS0)	32	R	0000_0000h
7B0h	Clock root access control (CLOCK_ROOT15_AUTHEN)	32	RW	FFFF_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
800h	Clock Root Control Register (CLOCK_ROOT16_CONTROL)	32	RW	0000_0000h
804h	Clock Root Control Register (CLOCK_ROOT16_CONTROL_SET)	32	RW	0000_0000h
808h	Clock Root Control Register (CLOCK_ROOT16_CONTROL_CLR)	32	RW	0000_0000h
80Ch	Clock Root Control Register (CLOCK_ROOT16_CONTROL_TOG)	32	RW	0000_0000h
820h	Clock root working status (CLOCK_ROOT16_STATUS0)	32	R	0000_0000h
830h	Clock root access control (CLOCK_ROOT16_AUTHEN)	32	RW	FFFF_0000h
880h	Clock Root Control Register (CLOCK_ROOT17_CONTROL)	32	RW	0000_0000h
884h	Clock Root Control Register (CLOCK_ROOT17_CONTROL_SET)	32	RW	0000_0000h
888h	Clock Root Control Register (CLOCK_ROOT17_CONTROL_CLR)	32	RW	0000_0000h
88Ch	Clock Root Control Register (CLOCK_ROOT17_CONTROL_TOG)	32	RW	0000_0000h
8A0h	Clock root working status (CLOCK_ROOT17_STATUS0)	32	R	0000_0000h
8B0h	Clock root access control (CLOCK_ROOT17_AUTHEN)	32	RW	FFFF_0000h
900h	Clock Root Control Register (CLOCK_ROOT18_CONTROL)	32	RW	0000_0000h
904h	Clock Root Control Register (CLOCK_ROOT18_CONTROL_SET)	32	RW	0000_0000h
908h	Clock Root Control Register (CLOCK_ROOT18_CONTROL_CLR)	32	RW	0000_0000h
90Ch	Clock Root Control Register (CLOCK_ROOT18_CONTROL_TOG)	32	RW	0000_0000h
920h	Clock root working status (CLOCK_ROOT18_STATUS0)	32	R	0000_0000h
930h	Clock root access control (CLOCK_ROOT18_AUTHEN)	32	RW	FFFF_0000h
980h	Clock Root Control Register (CLOCK_ROOT19_CONTROL)	32	RW	0000_0000h
984h	Clock Root Control Register (CLOCK_ROOT19_CONTROL_SET)	32	RW	0000_0000h
988h	Clock Root Control Register (CLOCK_ROOT19_CONTROL_CLR)	32	RW	0000_0000h
98Ch	Clock Root Control Register (CLOCK_ROOT19_CONTROL_TOG)	32	RW	0000_0000h
9A0h	Clock root working status (CLOCK_ROOT19_STATUS0)	32	R	0000_0000h
9B0h	Clock root access control (CLOCK_ROOT19_AUTHEN)	32	RW	FFFF_0000h
A00h	Clock Root Control Register (CLOCK_ROOT20_CONTROL)	32	RW	0000_0000h
A04h	Clock Root Control Register (CLOCK_ROOT20_CONTROL_SET)	32	RW	0000_0000h
A08h	Clock Root Control Register (CLOCK_ROOT20_CONTROL_CLR)	32	RW	0000_0000h
A0Ch	Clock Root Control Register (CLOCK_ROOT20_CONTROL_TOG)	32	RW	0000_0000h
A20h	Clock root working status (CLOCK_ROOT20_STATUS0)	32	R	0000_0000h
A30h	Clock root access control (CLOCK_ROOT20_AUTHEN)	32	RW	FFFF_0000h
A80h	Clock Root Control Register (CLOCK_ROOT21_CONTROL)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A84h	Clock Root Control Register (CLOCK_ROOT21_CONTROL_SET)	32	RW	0000_0000h
A88h	Clock Root Control Register (CLOCK_ROOT21_CONTROL_CLR)	32	RW	0000_0000h
A8Ch	Clock Root Control Register (CLOCK_ROOT21_CONTROL_TOG)	32	RW	0000_0000h
AA0h	Clock root working status (CLOCK_ROOT21_STATUS0)	32	R	0000_0000h
AB0h	Clock root access control (CLOCK_ROOT21_AUTHEN)	32	RW	FFFF_0000h
B00h	Clock Root Control Register (CLOCK_ROOT22_CONTROL)	32	RW	0000_0000h
B04h	Clock Root Control Register (CLOCK_ROOT22_CONTROL_SET)	32	RW	0000_0000h
B08h	Clock Root Control Register (CLOCK_ROOT22_CONTROL_CLR)	32	RW	0000_0000h
B0Ch	Clock Root Control Register (CLOCK_ROOT22_CONTROL_TOG)	32	RW	0000_0000h
B20h	Clock root working status (CLOCK_ROOT22_STATUS0)	32	R	0000_0000h
B30h	Clock root access control (CLOCK_ROOT22_AUTHEN)	32	RW	FFFF_0000h
B80h	Clock Root Control Register (CLOCK_ROOT23_CONTROL)	32	RW	0000_0000h
B84h	Clock Root Control Register (CLOCK_ROOT23_CONTROL_SET)	32	RW	0000_0000h
B88h	Clock Root Control Register (CLOCK_ROOT23_CONTROL_CLR)	32	RW	0000_0000h
B8Ch	Clock Root Control Register (CLOCK_ROOT23_CONTROL_TOG)	32	RW	0000_0000h
BA0h	Clock root working status (CLOCK_ROOT23_STATUS0)	32	R	0000_0000h
BB0h	Clock root access control (CLOCK_ROOT23_AUTHEN)	32	RW	FFFF_0000h
C00h	Clock Root Control Register (CLOCK_ROOT24_CONTROL)	32	RW	0000_0000h
C04h	Clock Root Control Register (CLOCK_ROOT24_CONTROL_SET)	32	RW	0000_0000h
C08h	Clock Root Control Register (CLOCK_ROOT24_CONTROL_CLR)	32	RW	0000_0000h
C0Ch	Clock Root Control Register (CLOCK_ROOT24_CONTROL_TOG)	32	RW	0000_0000h
C20h	Clock root working status (CLOCK_ROOT24_STATUS0)	32	R	0000_0000h
C30h	Clock root access control (CLOCK_ROOT24_AUTHEN)	32	RW	FFFF_0000h
C80h	Clock Root Control Register (CLOCK_ROOT25_CONTROL)	32	RW	0000_0000h
C84h	Clock Root Control Register (CLOCK_ROOT25_CONTROL_SET)	32	RW	0000_0000h
C88h	Clock Root Control Register (CLOCK_ROOT25_CONTROL_CLR)	32	RW	0000_0000h
C8Ch	Clock Root Control Register (CLOCK_ROOT25_CONTROL_TOG)	32	RW	0000_0000h
CA0h	Clock root working status (CLOCK_ROOT25_STATUS0)	32	R	0000_0000h
CB0h	Clock root access control (CLOCK_ROOT25_AUTHEN)	32	RW	FFFF_0000h
D00h	Clock Root Control Register (CLOCK_ROOT26_CONTROL)	32	RW	0000_0000h
D04h	Clock Root Control Register (CLOCK_ROOT26_CONTROL_SET)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
D08h	Clock Root Control Register (CLOCK_ROOT26_CONTROL_CLR)	32	RW	0000_0000h
D0Ch	Clock Root Control Register (CLOCK_ROOT26_CONTROL_TOG)	32	RW	0000_0000h
D20h	Clock root working status (CLOCK_ROOT26_STATUS0)	32	R	0000_0000h
D30h	Clock root access control (CLOCK_ROOT26_AUTHEN)	32	RW	FFFF_0000h
D80h	Clock Root Control Register (CLOCK_ROOT27_CONTROL)	32	RW	0000_0000h
D84h	Clock Root Control Register (CLOCK_ROOT27_CONTROL_SET)	32	RW	0000_0000h
D88h	Clock Root Control Register (CLOCK_ROOT27_CONTROL_CLR)	32	RW	0000_0000h
D8Ch	Clock Root Control Register (CLOCK_ROOT27_CONTROL_TOG)	32	RW	0000_0000h
DA0h	Clock root working status (CLOCK_ROOT27_STATUS0)	32	R	0000_0000h
DB0h	Clock root access control (CLOCK_ROOT27_AUTHEN)	32	RW	FFFF_0000h
E00h	Clock Root Control Register (CLOCK_ROOT28_CONTROL)	32	RW	0000_0000h
E04h	Clock Root Control Register (CLOCK_ROOT28_CONTROL_SET)	32	RW	0000_0000h
E08h	Clock Root Control Register (CLOCK_ROOT28_CONTROL_CLR)	32	RW	0000_0000h
E0Ch	Clock Root Control Register (CLOCK_ROOT28_CONTROL_TOG)	32	RW	0000_0000h
E20h	Clock root working status (CLOCK_ROOT28_STATUS0)	32	R	0000_0000h
E30h	Clock root access control (CLOCK_ROOT28_AUTHEN)	32	RW	FFFF_0000h
E80h	Clock Root Control Register (CLOCK_ROOT29_CONTROL)	32	RW	0000_0000h
E84h	Clock Root Control Register (CLOCK_ROOT29_CONTROL_SET)	32	RW	0000_0000h
E88h	Clock Root Control Register (CLOCK_ROOT29_CONTROL_CLR)	32	RW	0000_0000h
E8Ch	Clock Root Control Register (CLOCK_ROOT29_CONTROL_TOG)	32	RW	0000_0000h
EA0h	Clock root working status (CLOCK_ROOT29_STATUS0)	32	R	0000_0000h
EB0h	Clock root access control (CLOCK_ROOT29_AUTHEN)	32	RW	FFFF_0000h
F00h	Clock Root Control Register (CLOCK_ROOT30_CONTROL)	32	RW	0000_0000h
F04h	Clock Root Control Register (CLOCK_ROOT30_CONTROL_SET)	32	RW	0000_0000h
F08h	Clock Root Control Register (CLOCK_ROOT30_CONTROL_CLR)	32	RW	0000_0000h
F0Ch	Clock Root Control Register (CLOCK_ROOT30_CONTROL_TOG)	32	RW	0000_0000h
F20h	Clock root working status (CLOCK_ROOT30_STATUS0)	32	R	0000_0000h
F30h	Clock root access control (CLOCK_ROOT30_AUTHEN)	32	RW	FFFF_0000h
F80h	Clock Root Control Register (CLOCK_ROOT31_CONTROL)	32	RW	0000_0000h
F84h	Clock Root Control Register (CLOCK_ROOT31_CONTROL_SET)	32	RW	0000_0000h
F88h	Clock Root Control Register (CLOCK_ROOT31_CONTROL_CLR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
F8Ch	Clock Root Control Register (CLOCK_ROOT31_CONTROL_TOG)	32	RW	0000_0000h
FA0h	Clock root working status (CLOCK_ROOT31_STATUS0)	32	R	0000_0000h
FB0h	Clock root access control (CLOCK_ROOT31_AUTHEN)	32	RW	FFFF_0000h
1000h	Clock Root Control Register (CLOCK_ROOT32_CONTROL)	32	RW	0000_0000h
1004h	Clock Root Control Register (CLOCK_ROOT32_CONTROL_SET)	32	RW	0000_0000h
1008h	Clock Root Control Register (CLOCK_ROOT32_CONTROL_CLR)	32	RW	0000_0000h
100Ch	Clock Root Control Register (CLOCK_ROOT32_CONTROL_TOG)	32	RW	0000_0000h
1020h	Clock root working status (CLOCK_ROOT32_STATUS0)	32	R	0000_0000h
1030h	Clock root access control (CLOCK_ROOT32_AUTHEN)	32	RW	FFFF_0000h
1080h	Clock Root Control Register (CLOCK_ROOT33_CONTROL)	32	RW	0000_0000h
1084h	Clock Root Control Register (CLOCK_ROOT33_CONTROL_SET)	32	RW	0000_0000h
1088h	Clock Root Control Register (CLOCK_ROOT33_CONTROL_CLR)	32	RW	0000_0000h
108Ch	Clock Root Control Register (CLOCK_ROOT33_CONTROL_TOG)	32	RW	0000_0000h
10A0h	Clock root working status (CLOCK_ROOT33_STATUS0)	32	R	0000_0000h
10B0h	Clock root access control (CLOCK_ROOT33_AUTHEN)	32	RW	FFFF_0000h
1100h	Clock Root Control Register (CLOCK_ROOT34_CONTROL)	32	RW	0000_0000h
1104h	Clock Root Control Register (CLOCK_ROOT34_CONTROL_SET)	32	RW	0000_0000h
1108h	Clock Root Control Register (CLOCK_ROOT34_CONTROL_CLR)	32	RW	0000_0000h
110Ch	Clock Root Control Register (CLOCK_ROOT34_CONTROL_TOG)	32	RW	0000_0000h
1120h	Clock root working status (CLOCK_ROOT34_STATUS0)	32	R	0000_0000h
1130h	Clock root access control (CLOCK_ROOT34_AUTHEN)	32	RW	FFFF_0000h
1180h	Clock Root Control Register (CLOCK_ROOT35_CONTROL)	32	RW	0000_0000h
1184h	Clock Root Control Register (CLOCK_ROOT35_CONTROL_SET)	32	RW	0000_0000h
1188h	Clock Root Control Register (CLOCK_ROOT35_CONTROL_CLR)	32	RW	0000_0000h
118Ch	Clock Root Control Register (CLOCK_ROOT35_CONTROL_TOG)	32	RW	0000_0000h
11A0h	Clock root working status (CLOCK_ROOT35_STATUS0)	32	R	0000_0000h
11B0h	Clock root access control (CLOCK_ROOT35_AUTHEN)	32	RW	FFFF_0000h
1200h	Clock Root Control Register (CLOCK_ROOT36_CONTROL)	32	RW	0000_0000h
1204h	Clock Root Control Register (CLOCK_ROOT36_CONTROL_SET)	32	RW	0000_0000h
1208h	Clock Root Control Register (CLOCK_ROOT36_CONTROL_CLR)	32	RW	0000_0000h
120Ch	Clock Root Control Register (CLOCK_ROOT36_CONTROL_TOG)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1220h	Clock root working status (CLOCK_ROOT36_STATUS0)	32	R	0000_0000h
1230h	Clock root access control (CLOCK_ROOT36_AUTHEN)	32	RW	FFFF_0000h
1280h	Clock Root Control Register (CLOCK_ROOT37_CONTROL)	32	RW	0000_0000h
1284h	Clock Root Control Register (CLOCK_ROOT37_CONTROL_SET)	32	RW	0000_0000h
1288h	Clock Root Control Register (CLOCK_ROOT37_CONTROL_CLR)	32	RW	0000_0000h
128Ch	Clock Root Control Register (CLOCK_ROOT37_CONTROL_TOG)	32	RW	0000_0000h
12A0h	Clock root working status (CLOCK_ROOT37_STATUS0)	32	R	0000_0000h
12B0h	Clock root access control (CLOCK_ROOT37_AUTHEN)	32	RW	FFFF_0000h
1300h	Clock Root Control Register (CLOCK_ROOT38_CONTROL)	32	RW	0000_0000h
1304h	Clock Root Control Register (CLOCK_ROOT38_CONTROL_SET)	32	RW	0000_0000h
1308h	Clock Root Control Register (CLOCK_ROOT38_CONTROL_CLR)	32	RW	0000_0000h
130Ch	Clock Root Control Register (CLOCK_ROOT38_CONTROL_TOG)	32	RW	0000_0000h
1320h	Clock root working status (CLOCK_ROOT38_STATUS0)	32	R	0000_0000h
1330h	Clock root access control (CLOCK_ROOT38_AUTHEN)	32	RW	FFFF_0000h
1380h	Clock Root Control Register (CLOCK_ROOT39_CONTROL)	32	RW	0000_0000h
1384h	Clock Root Control Register (CLOCK_ROOT39_CONTROL_SET)	32	RW	0000_0000h
1388h	Clock Root Control Register (CLOCK_ROOT39_CONTROL_CLR)	32	RW	0000_0000h
138Ch	Clock Root Control Register (CLOCK_ROOT39_CONTROL_TOG)	32	RW	0000_0000h
13A0h	Clock root working status (CLOCK_ROOT39_STATUS0)	32	R	0000_0000h
13B0h	Clock root access control (CLOCK_ROOT39_AUTHEN)	32	RW	FFFF_0000h
1400h	Clock Root Control Register (CLOCK_ROOT40_CONTROL)	32	RW	0000_0000h
1404h	Clock Root Control Register (CLOCK_ROOT40_CONTROL_SET)	32	RW	0000_0000h
1408h	Clock Root Control Register (CLOCK_ROOT40_CONTROL_CLR)	32	RW	0000_0000h
140Ch	Clock Root Control Register (CLOCK_ROOT40_CONTROL_TOG)	32	RW	0000_0000h
1420h	Clock root working status (CLOCK_ROOT40_STATUS0)	32	R	0000_0000h
1430h	Clock root access control (CLOCK_ROOT40_AUTHEN)	32	RW	FFFF_0000h
1480h	Clock Root Control Register (CLOCK_ROOT41_CONTROL)	32	RW	0000_0000h
1484h	Clock Root Control Register (CLOCK_ROOT41_CONTROL_SET)	32	RW	0000_0000h
1488h	Clock Root Control Register (CLOCK_ROOT41_CONTROL_CLR)	32	RW	0000_0000h
148Ch	Clock Root Control Register (CLOCK_ROOT41_CONTROL_TOG)	32	RW	0000_0000h
14A0h	Clock root working status (CLOCK_ROOT41_STATUS0)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
14B0h	Clock root access control (CLOCK_ROOT41_AUTHEN)	32	RW	FFFF_0000h
1500h	Clock Root Control Register (CLOCK_ROOT42_CONTROL)	32	RW	0000_0000h
1504h	Clock Root Control Register (CLOCK_ROOT42_CONTROL_SET)	32	RW	0000_0000h
1508h	Clock Root Control Register (CLOCK_ROOT42_CONTROL_CLR)	32	RW	0000_0000h
150Ch	Clock Root Control Register (CLOCK_ROOT42_CONTROL_TOG)	32	RW	0000_0000h
1520h	Clock root working status (CLOCK_ROOT42_STATUS0)	32	R	0000_0000h
1530h	Clock root access control (CLOCK_ROOT42_AUTHEN)	32	RW	FFFF_0000h
1580h	Clock Root Control Register (CLOCK_ROOT43_CONTROL)	32	RW	0000_0000h
1584h	Clock Root Control Register (CLOCK_ROOT43_CONTROL_SET)	32	RW	0000_0000h
1588h	Clock Root Control Register (CLOCK_ROOT43_CONTROL_CLR)	32	RW	0000_0000h
158Ch	Clock Root Control Register (CLOCK_ROOT43_CONTROL_TOG)	32	RW	0000_0000h
15A0h	Clock root working status (CLOCK_ROOT43_STATUS0)	32	R	0000_0000h
15B0h	Clock root access control (CLOCK_ROOT43_AUTHEN)	32	RW	FFFF_0000h
1600h	Clock Root Control Register (CLOCK_ROOT44_CONTROL)	32	RW	0000_0000h
1604h	Clock Root Control Register (CLOCK_ROOT44_CONTROL_SET)	32	RW	0000_0000h
1608h	Clock Root Control Register (CLOCK_ROOT44_CONTROL_CLR)	32	RW	0000_0000h
160Ch	Clock Root Control Register (CLOCK_ROOT44_CONTROL_TOG)	32	RW	0000_0000h
1620h	Clock root working status (CLOCK_ROOT44_STATUS0)	32	R	0000_0000h
1630h	Clock root access control (CLOCK_ROOT44_AUTHEN)	32	RW	FFFF_0000h
1680h	Clock Root Control Register (CLOCK_ROOT45_CONTROL)	32	RW	0000_0000h
1684h	Clock Root Control Register (CLOCK_ROOT45_CONTROL_SET)	32	RW	0000_0000h
1688h	Clock Root Control Register (CLOCK_ROOT45_CONTROL_CLR)	32	RW	0000_0000h
168Ch	Clock Root Control Register (CLOCK_ROOT45_CONTROL_TOG)	32	RW	0000_0000h
16A0h	Clock root working status (CLOCK_ROOT45_STATUS0)	32	R	0000_0000h
16B0h	Clock root access control (CLOCK_ROOT45_AUTHEN)	32	RW	FFFF_0000h
1700h	Clock Root Control Register (CLOCK_ROOT46_CONTROL)	32	RW	0000_0000h
1704h	Clock Root Control Register (CLOCK_ROOT46_CONTROL_SET)	32	RW	0000_0000h
1708h	Clock Root Control Register (CLOCK_ROOT46_CONTROL_CLR)	32	RW	0000_0000h
170Ch	Clock Root Control Register (CLOCK_ROOT46_CONTROL_TOG)	32	RW	0000_0000h
1720h	Clock root working status (CLOCK_ROOT46_STATUS0)	32	R	0000_0000h
1730h	Clock root access control (CLOCK_ROOT46_AUTHEN)	32	RW	FFFF_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1780h	Clock Root Control Register (CLOCK_ROOT47_CONTROL)	32	RW	0000_0000h
1784h	Clock Root Control Register (CLOCK_ROOT47_CONTROL_SET)	32	RW	0000_0000h
1788h	Clock Root Control Register (CLOCK_ROOT47_CONTROL_CLR)	32	RW	0000_0000h
178Ch	Clock Root Control Register (CLOCK_ROOT47_CONTROL_TOG)	32	RW	0000_0000h
17A0h	Clock root working status (CLOCK_ROOT47_STATUS0)	32	R	0000_0000h
17B0h	Clock root access control (CLOCK_ROOT47_AUTHEN)	32	RW	FFFF_0000h
1800h	Clock Root Control Register (CLOCK_ROOT48_CONTROL)	32	RW	0000_0000h
1804h	Clock Root Control Register (CLOCK_ROOT48_CONTROL_SET)	32	RW	0000_0000h
1808h	Clock Root Control Register (CLOCK_ROOT48_CONTROL_CLR)	32	RW	0000_0000h
180Ch	Clock Root Control Register (CLOCK_ROOT48_CONTROL_TOG)	32	RW	0000_0000h
1820h	Clock root working status (CLOCK_ROOT48_STATUS0)	32	R	0000_0000h
1830h	Clock root access control (CLOCK_ROOT48_AUTHEN)	32	RW	FFFF_0000h
1880h	Clock Root Control Register (CLOCK_ROOT49_CONTROL)	32	RW	0000_0000h
1884h	Clock Root Control Register (CLOCK_ROOT49_CONTROL_SET)	32	RW	0000_0000h
1888h	Clock Root Control Register (CLOCK_ROOT49_CONTROL_CLR)	32	RW	0000_0000h
188Ch	Clock Root Control Register (CLOCK_ROOT49_CONTROL_TOG)	32	RW	0000_0000h
18A0h	Clock root working status (CLOCK_ROOT49_STATUS0)	32	R	0000_0000h
18B0h	Clock root access control (CLOCK_ROOT49_AUTHEN)	32	RW	FFFF_0000h
1900h	Clock Root Control Register (CLOCK_ROOT50_CONTROL)	32	RW	0000_0000h
1904h	Clock Root Control Register (CLOCK_ROOT50_CONTROL_SET)	32	RW	0000_0000h
1908h	Clock Root Control Register (CLOCK_ROOT50_CONTROL_CLR)	32	RW	0000_0000h
190Ch	Clock Root Control Register (CLOCK_ROOT50_CONTROL_TOG)	32	RW	0000_0000h
1920h	Clock root working status (CLOCK_ROOT50_STATUS0)	32	R	0000_0000h
1930h	Clock root access control (CLOCK_ROOT50_AUTHEN)	32	RW	FFFF_0000h
1980h	Clock Root Control Register (CLOCK_ROOT51_CONTROL)	32	RW	0000_0000h
1984h	Clock Root Control Register (CLOCK_ROOT51_CONTROL_SET)	32	RW	0000_0000h
1988h	Clock Root Control Register (CLOCK_ROOT51_CONTROL_CLR)	32	RW	0000_0000h
198Ch	Clock Root Control Register (CLOCK_ROOT51_CONTROL_TOG)	32	RW	0000_0000h
19A0h	Clock root working status (CLOCK_ROOT51_STATUS0)	32	R	0000_0000h
19B0h	Clock root access control (CLOCK_ROOT51_AUTHEN)	32	RW	FFFF_0000h
1A00h	Clock Root Control Register (CLOCK_ROOT52_CONTROL)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1A04h	Clock Root Control Register (CLOCK_ROOT52_CONTROL_SET)	32	RW	0000_0000h
1A08h	Clock Root Control Register (CLOCK_ROOT52_CONTROL_CLR)	32	RW	0000_0000h
1A0Ch	Clock Root Control Register (CLOCK_ROOT52_CONTROL_TOG)	32	RW	0000_0000h
1A20h	Clock root working status (CLOCK_ROOT52_STATUS0)	32	R	0000_0000h
1A30h	Clock root access control (CLOCK_ROOT52_AUTHEN)	32	RW	FFFF_0000h
1A80h	Clock Root Control Register (CLOCK_ROOT53_CONTROL)	32	RW	0000_0000h
1A84h	Clock Root Control Register (CLOCK_ROOT53_CONTROL_SET)	32	RW	0000_0000h
1A88h	Clock Root Control Register (CLOCK_ROOT53_CONTROL_CLR)	32	RW	0000_0000h
1A8Ch	Clock Root Control Register (CLOCK_ROOT53_CONTROL_TOG)	32	RW	0000_0000h
1AA0h	Clock root working status (CLOCK_ROOT53_STATUS0)	32	R	0000_0000h
1AB0h	Clock root access control (CLOCK_ROOT53_AUTHEN)	32	RW	FFFF_0000h
1B00h	Clock Root Control Register (CLOCK_ROOT54_CONTROL)	32	RW	0000_0000h
1B04h	Clock Root Control Register (CLOCK_ROOT54_CONTROL_SET)	32	RW	0000_0000h
1B08h	Clock Root Control Register (CLOCK_ROOT54_CONTROL_CLR)	32	RW	0000_0000h
1B0Ch	Clock Root Control Register (CLOCK_ROOT54_CONTROL_TOG)	32	RW	0000_0000h
1B20h	Clock root working status (CLOCK_ROOT54_STATUS0)	32	R	0000_0000h
1B30h	Clock root access control (CLOCK_ROOT54_AUTHEN)	32	RW	FFFF_0000h
1B80h	Clock Root Control Register (CLOCK_ROOT55_CONTROL)	32	RW	0000_0000h
1B84h	Clock Root Control Register (CLOCK_ROOT55_CONTROL_SET)	32	RW	0000_0000h
1B88h	Clock Root Control Register (CLOCK_ROOT55_CONTROL_CLR)	32	RW	0000_0000h
1B8Ch	Clock Root Control Register (CLOCK_ROOT55_CONTROL_TOG)	32	RW	0000_0000h
1BA0h	Clock root working status (CLOCK_ROOT55_STATUS0)	32	R	0000_0000h
1BB0h	Clock root access control (CLOCK_ROOT55_AUTHEN)	32	RW	FFFF_0000h
1C00h	Clock Root Control Register (CLOCK_ROOT56_CONTROL)	32	RW	0000_0000h
1C04h	Clock Root Control Register (CLOCK_ROOT56_CONTROL_SET)	32	RW	0000_0000h
1C08h	Clock Root Control Register (CLOCK_ROOT56_CONTROL_CLR)	32	RW	0000_0000h
1C0Ch	Clock Root Control Register (CLOCK_ROOT56_CONTROL_TOG)	32	RW	0000_0000h
1C20h	Clock root working status (CLOCK_ROOT56_STATUS0)	32	R	0000_0000h
1C30h	Clock root access control (CLOCK_ROOT56_AUTHEN)	32	RW	FFFF_0000h
1C80h	Clock Root Control Register (CLOCK_ROOT57_CONTROL)	32	RW	0000_0000h
1C84h	Clock Root Control Register (CLOCK_ROOT57_CONTROL_SET)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1C88h	Clock Root Control Register (CLOCK_ROOT57_CONTROL_CLR)	32	RW	0000_0000h
1C8Ch	Clock Root Control Register (CLOCK_ROOT57_CONTROL_TOG)	32	RW	0000_0000h
1CA0h	Clock root working status (CLOCK_ROOT57_STATUS0)	32	R	0000_0000h
1CB0h	Clock root access control (CLOCK_ROOT57_AUTHEN)	32	RW	FFFF_0000h
1D00h	Clock Root Control Register (CLOCK_ROOT58_CONTROL)	32	RW	0000_0000h
1D04h	Clock Root Control Register (CLOCK_ROOT58_CONTROL_SET)	32	RW	0000_0000h
1D08h	Clock Root Control Register (CLOCK_ROOT58_CONTROL_CLR)	32	RW	0000_0000h
1D0Ch	Clock Root Control Register (CLOCK_ROOT58_CONTROL_TOG)	32	RW	0000_0000h
1D20h	Clock root working status (CLOCK_ROOT58_STATUS0)	32	R	0000_0000h
1D30h	Clock root access control (CLOCK_ROOT58_AUTHEN)	32	RW	FFFF_0000h
1D80h	Clock Root Control Register (CLOCK_ROOT59_CONTROL)	32	RW	0000_0000h
1D84h	Clock Root Control Register (CLOCK_ROOT59_CONTROL_SET)	32	RW	0000_0000h
1D88h	Clock Root Control Register (CLOCK_ROOT59_CONTROL_CLR)	32	RW	0000_0000h
1D8Ch	Clock Root Control Register (CLOCK_ROOT59_CONTROL_TOG)	32	RW	0000_0000h
1DA0h	Clock root working status (CLOCK_ROOT59_STATUS0)	32	R	0000_0000h
1DB0h	Clock root access control (CLOCK_ROOT59_AUTHEN)	32	RW	FFFF_0000h
1E00h	Clock Root Control Register (CLOCK_ROOT60_CONTROL)	32	RW	0000_0000h
1E04h	Clock Root Control Register (CLOCK_ROOT60_CONTROL_SET)	32	RW	0000_0000h
1E08h	Clock Root Control Register (CLOCK_ROOT60_CONTROL_CLR)	32	RW	0000_0000h
1E0Ch	Clock Root Control Register (CLOCK_ROOT60_CONTROL_TOG)	32	RW	0000_0000h
1E20h	Clock root working status (CLOCK_ROOT60_STATUS0)	32	R	0000_0000h
1E30h	Clock root access control (CLOCK_ROOT60_AUTHEN)	32	RW	FFFF_0000h
1E80h	Clock Root Control Register (CLOCK_ROOT61_CONTROL)	32	RW	0000_0000h
1E84h	Clock Root Control Register (CLOCK_ROOT61_CONTROL_SET)	32	RW	0000_0000h
1E88h	Clock Root Control Register (CLOCK_ROOT61_CONTROL_CLR)	32	RW	0000_0000h
1E8Ch	Clock Root Control Register (CLOCK_ROOT61_CONTROL_TOG)	32	RW	0000_0000h
1EA0h	Clock root working status (CLOCK_ROOT61_STATUS0)	32	R	0000_0000h
1EB0h	Clock root access control (CLOCK_ROOT61_AUTHEN)	32	RW	FFFF_0000h
1F00h	Clock Root Control Register (CLOCK_ROOT62_CONTROL)	32	RW	0000_0000h
1F04h	Clock Root Control Register (CLOCK_ROOT62_CONTROL_SET)	32	RW	0000_0000h
1F08h	Clock Root Control Register (CLOCK_ROOT62_CONTROL_CLR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1F0Ch	Clock Root Control Register (CLOCK_ROOT62_CONTROL_TOG)	32	RW	0000_0000h
1F20h	Clock root working status (CLOCK_ROOT62_STATUS0)	32	R	0000_0000h
1F30h	Clock root access control (CLOCK_ROOT62_AUTHEN)	32	RW	FFFF_0000h
1F80h	Clock Root Control Register (CLOCK_ROOT63_CONTROL)	32	RW	0000_0000h
1F84h	Clock Root Control Register (CLOCK_ROOT63_CONTROL_SET)	32	RW	0000_0000h
1F88h	Clock Root Control Register (CLOCK_ROOT63_CONTROL_CLR)	32	RW	0000_0000h
1F8Ch	Clock Root Control Register (CLOCK_ROOT63_CONTROL_TOG)	32	RW	0000_0000h
1FA0h	Clock root working status (CLOCK_ROOT63_STATUS0)	32	R	0000_0000h
1FB0h	Clock root access control (CLOCK_ROOT63_AUTHEN)	32	RW	FFFF_0000h
2000h	Clock Root Control Register (CLOCK_ROOT64_CONTROL)	32	RW	0000_0000h
2004h	Clock Root Control Register (CLOCK_ROOT64_CONTROL_SET)	32	RW	0000_0000h
2008h	Clock Root Control Register (CLOCK_ROOT64_CONTROL_CLR)	32	RW	0000_0000h
200Ch	Clock Root Control Register (CLOCK_ROOT64_CONTROL_TOG)	32	RW	0000_0000h
2020h	Clock root working status (CLOCK_ROOT64_STATUS0)	32	R	0000_0000h
2030h	Clock root access control (CLOCK_ROOT64_AUTHEN)	32	RW	FFFF_0000h
2080h	Clock Root Control Register (CLOCK_ROOT65_CONTROL)	32	RW	0000_0000h
2084h	Clock Root Control Register (CLOCK_ROOT65_CONTROL_SET)	32	RW	0000_0000h
2088h	Clock Root Control Register (CLOCK_ROOT65_CONTROL_CLR)	32	RW	0000_0000h
208Ch	Clock Root Control Register (CLOCK_ROOT65_CONTROL_TOG)	32	RW	0000_0000h
20A0h	Clock root working status (CLOCK_ROOT65_STATUS0)	32	R	0000_0000h
20B0h	Clock root access control (CLOCK_ROOT65_AUTHEN)	32	RW	FFFF_0000h
2100h	Clock Root Control Register (CLOCK_ROOT66_CONTROL)	32	RW	0000_0000h
2104h	Clock Root Control Register (CLOCK_ROOT66_CONTROL_SET)	32	RW	0000_0000h
2108h	Clock Root Control Register (CLOCK_ROOT66_CONTROL_CLR)	32	RW	0000_0000h
210Ch	Clock Root Control Register (CLOCK_ROOT66_CONTROL_TOG)	32	RW	0000_0000h
2120h	Clock root working status (CLOCK_ROOT66_STATUS0)	32	R	0000_0000h
2130h	Clock root access control (CLOCK_ROOT66_AUTHEN)	32	RW	FFFF_0000h
2180h	Clock Root Control Register (CLOCK_ROOT67_CONTROL)	32	RW	0000_0000h
2184h	Clock Root Control Register (CLOCK_ROOT67_CONTROL_SET)	32	RW	0000_0000h
2188h	Clock Root Control Register (CLOCK_ROOT67_CONTROL_CLR)	32	RW	0000_0000h
218Ch	Clock Root Control Register (CLOCK_ROOT67_CONTROL_TOG)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
21A0h	Clock root working status (CLOCK_ROOT67_STATUS0)	32	R	0000_0000h
21B0h	Clock root access control (CLOCK_ROOT67_AUTHEN)	32	RW	FFFF_0000h
2200h	Clock Root Control Register (CLOCK_ROOT68_CONTROL)	32	RW	0000_0000h
2204h	Clock Root Control Register (CLOCK_ROOT68_CONTROL_SET)	32	RW	0000_0000h
2208h	Clock Root Control Register (CLOCK_ROOT68_CONTROL_CLR)	32	RW	0000_0000h
220Ch	Clock Root Control Register (CLOCK_ROOT68_CONTROL_TOG)	32	RW	0000_0000h
2220h	Clock root working status (CLOCK_ROOT68_STATUS0)	32	R	0000_0000h
2230h	Clock root access control (CLOCK_ROOT68_AUTHEN)	32	RW	FFFF_0000h
2280h	Clock Root Control Register (CLOCK_ROOT69_CONTROL)	32	RW	0000_0000h
2284h	Clock Root Control Register (CLOCK_ROOT69_CONTROL_SET)	32	RW	0000_0000h
2288h	Clock Root Control Register (CLOCK_ROOT69_CONTROL_CLR)	32	RW	0000_0000h
228Ch	Clock Root Control Register (CLOCK_ROOT69_CONTROL_TOG)	32	RW	0000_0000h
22A0h	Clock root working status (CLOCK_ROOT69_STATUS0)	32	R	0000_0000h
22B0h	Clock root access control (CLOCK_ROOT69_AUTHEN)	32	RW	FFFF_0000h
2300h	Clock Root Control Register (CLOCK_ROOT70_CONTROL)	32	RW	0000_0000h
2304h	Clock Root Control Register (CLOCK_ROOT70_CONTROL_SET)	32	RW	0000_0000h
2308h	Clock Root Control Register (CLOCK_ROOT70_CONTROL_CLR)	32	RW	0000_0000h
230Ch	Clock Root Control Register (CLOCK_ROOT70_CONTROL_TOG)	32	RW	0000_0000h
2320h	Clock root working status (CLOCK_ROOT70_STATUS0)	32	R	0000_0000h
2330h	Clock root access control (CLOCK_ROOT70_AUTHEN)	32	RW	FFFF_0000h
2380h	Clock Root Control Register (CLOCK_ROOT71_CONTROL)	32	RW	0000_0000h
2384h	Clock Root Control Register (CLOCK_ROOT71_CONTROL_SET)	32	RW	0000_0000h
2388h	Clock Root Control Register (CLOCK_ROOT71_CONTROL_CLR)	32	RW	0000_0000h
238Ch	Clock Root Control Register (CLOCK_ROOT71_CONTROL_TOG)	32	RW	0000_0000h
23A0h	Clock root working status (CLOCK_ROOT71_STATUS0)	32	R	0000_0000h
23B0h	Clock root access control (CLOCK_ROOT71_AUTHEN)	32	RW	FFFF_0000h
2400h	Clock Root Control Register (CLOCK_ROOT72_CONTROL)	32	RW	0000_0000h
2404h	Clock Root Control Register (CLOCK_ROOT72_CONTROL_SET)	32	RW	0000_0000h
2408h	Clock Root Control Register (CLOCK_ROOT72_CONTROL_CLR)	32	RW	0000_0000h
240Ch	Clock Root Control Register (CLOCK_ROOT72_CONTROL_TOG)	32	RW	0000_0000h
2420h	Clock root working status (CLOCK_ROOT72_STATUS0)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2430h	Clock root access control (CLOCK_ROOT72_AUTHEN)	32	RW	FFFF_0000h
2480h	Clock Root Control Register (CLOCK_ROOT73_CONTROL)	32	RW	0000_0000h
2484h	Clock Root Control Register (CLOCK_ROOT73_CONTROL_SET)	32	RW	0000_0000h
2488h	Clock Root Control Register (CLOCK_ROOT73_CONTROL_CLR)	32	RW	0000_0000h
248Ch	Clock Root Control Register (CLOCK_ROOT73_CONTROL_TOG)	32	RW	0000_0000h
24A0h	Clock root working status (CLOCK_ROOT73_STATUS0)	32	R	0000_0000h
24B0h	Clock root access control (CLOCK_ROOT73_AUTHEN)	32	RW	FFFF_0000h
4400h	Observe control (OBSERVE0_CONTROL)	32	RW	0000_0000h
4404h	Observe control (OBSERVE0_CONTROL_SET)	32	RW	0000_0000h
4408h	Observe control (OBSERVE0_CONTROL_CLR)	32	RW	0000_0000h
440Ch	Observe control (OBSERVE0_CONTROL_TOG)	32	RW	0000_0000h
4420h	Observe status (OBSERVE0_STATUS)	32	R	0000_0000h
4430h	Observe access control (OBSERVE0_AUTHEN)	32	RW	FFFF_0000h
4434h	Observe access control (OBSERVE0_AUTHEN_SET)	32	RW	FFFF_0000h
4438h	Observe access control (OBSERVE0_AUTHEN_CLR)	32	RW	FFFF_0000h
443Ch	Observe access control (OBSERVE0_AUTHEN_TOG)	32	RW	FFFF_0000h
4440h	Current frequency detected (OBSERVE0_FREQUENCY_CURRENT)	32	R	0000_0000h
4444h	Minimum frequency detected (OBSERVE0_FREQUENCY_MIN)	32	R	FFFF_FFC0h
4448h	Maximum frequency detected (OBSERVE0_FREQUENCY_MAX)	32	R	0000_0000h
4450h	Current period time detected (OBSERVE0_PERIOD_CURRENT)	32	R	0000_0000h
4454h	Minimum period time detected (OBSERVE0_PERIOD_MIN)	32	R	FFFF_FFFFh
4458h	Maximum period time detected (OBSERVE0_PERIOD_MAX)	32	R	0000_0000h
4460h	Current high level time detected (OBSERVE0_HIGH_CURRENT)	32	R	0000_0000h
4464h	Minimum high level time detected (OBSERVE0_HIGH_MIN)	32	R	FFFF_FFFFh
4468h	Maximum high level time detected (OBSERVE0_HIGH_MAX)	32	R	0000_0000h
4470h	Current high level time detected (OBSERVE0_LOW_CURRENT)	32	R	0000_0000h
4474h	Minimum high level time detected (OBSERVE0_LOW_MIN)	32	R	FFFF_FFFFh
4478h	Maximum high level time detected (OBSERVE0_LOW_MAX)	32	R	0000_0000h
4480h	Observe control (OBSERVE1_CONTROL)	32	RW	0000_0000h
4484h	Observe control (OBSERVE1_CONTROL_SET)	32	RW	0000_0000h
4488h	Observe control (OBSERVE1_CONTROL_CLR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
448Ch	Observe control (OBSERVE1_CONTROL_TOG)	32	RW	0000_0000h
44A0h	Observe status (OBSERVE1_STATUS)	32	R	0000_0000h
44B0h	Observe access control (OBSERVE1_AUTHEN)	32	RW	FFFF_0000h
44B4h	Observe access control (OBSERVE1_AUTHEN_SET)	32	RW	FFFF_0000h
44B8h	Observe access control (OBSERVE1_AUTHEN_CLR)	32	RW	FFFF_0000h
44BCh	Observe access control (OBSERVE1_AUTHEN_TOG)	32	RW	FFFF_0000h
44C0h	Current frequency detected (OBSERVE1_FREQUENCY_CURRENT)	32	R	0000_0000h
44C4h	Minimum frequency detected (OBSERVE1_FREQUENCY_MIN)	32	R	FFFF_FFC0h
44C8h	Maximum frequency detected (OBSERVE1_FREQUENCY_MAX)	32	R	0000_0000h
44D0h	Current period time detected (OBSERVE1_PERIOD_CURRENT)	32	R	0000_0000h
44D4h	Minimum period time detected (OBSERVE1_PERIOD_MIN)	32	R	FFFF_FFFFh
44D8h	Maximum period time detected (OBSERVE1_PERIOD_MAX)	32	R	0000_0000h
44E0h	Current high level time detected (OBSERVE1_HIGH_CURRENT)	32	R	0000_0000h
44E4h	Minimum high level time detected (OBSERVE1_HIGH_MIN)	32	R	FFFF_FFFFh
44E8h	Maximum high level time detected (OBSERVE1_HIGH_MAX)	32	R	0000_0000h
44F0h	Current high level time detected (OBSERVE1_LOW_CURRENT)	32	R	0000_0000h
44F4h	Minimum high level time detected (OBSERVE1_LOW_MIN)	32	R	FFFF_FFFFh
44F8h	Maximum high level time detected (OBSERVE1_LOW_MAX)	32	R	0000_0000h
4800h	General Purpose Register (GPR_SHARED0)	32	RW	0000_0000h
4804h	General Purpose Register (GPR_SHARED0_SET)	32	RW	0000_0000h
4808h	General Purpose Register (GPR_SHARED0_CLR)	32	RW	0000_0000h
480Ch	General Purpose Register (GPR_SHARED0_TOG)	32	RW	0000_0000h
4810h	GPR access control (GPR_SHARED0_AUTHEN)	32	RW	FFFF_0000h
4814h	GPR access control (GPR_SHARED0_AUTHEN_SET)	32	RW	FFFF_0000h
4818h	GPR access control (GPR_SHARED0_AUTHEN_CLR)	32	RW	FFFF_0000h
481Ch	GPR access control (GPR_SHARED0_AUTHEN_TOG)	32	RW	FFFF_0000h
4820h	General Purpose Register (GPR_SHARED1)	32	RW	0000_0000h
4824h	General Purpose Register (GPR_SHARED1_SET)	32	RW	0000_0000h
4828h	General Purpose Register (GPR_SHARED1_CLR)	32	RW	0000_0000h
482Ch	General Purpose Register (GPR_SHARED1_TOG)	32	RW	0000_0000h
4830h	GPR access control (GPR_SHARED1_AUTHEN)	32	RW	FFFF_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4834h	GPR access control (GPR_SHARED1_AUTHEN_SET)	32	RW	FFFF_0000h
4838h	GPR access control (GPR_SHARED1_AUTHEN_CLR)	32	RW	FFFF_0000h
483Ch	GPR access control (GPR_SHARED1_AUTHEN_TOG)	32	RW	FFFF_0000h
4840h	General Purpose Register (GPR_SHARED2)	32	RW	0000_0000h
4844h	General Purpose Register (GPR_SHARED2_SET)	32	RW	0000_0000h
4848h	General Purpose Register (GPR_SHARED2_CLR)	32	RW	0000_0000h
484Ch	General Purpose Register (GPR_SHARED2_TOG)	32	RW	0000_0000h
4850h	GPR access control (GPR_SHARED2_AUTHEN)	32	RW	FFFF_0000h
4854h	GPR access control (GPR_SHARED2_AUTHEN_SET)	32	RW	FFFF_0000h
4858h	GPR access control (GPR_SHARED2_AUTHEN_CLR)	32	RW	FFFF_0000h
485Ch	GPR access control (GPR_SHARED2_AUTHEN_TOG)	32	RW	FFFF_0000h
4860h	General Purpose Register (GPR_SHARED3)	32	RW	0000_0000h
4864h	General Purpose Register (GPR_SHARED3_SET)	32	RW	0000_0000h
4868h	General Purpose Register (GPR_SHARED3_CLR)	32	RW	0000_0000h
486Ch	General Purpose Register (GPR_SHARED3_TOG)	32	RW	0000_0000h
4870h	GPR access control (GPR_SHARED3_AUTHEN)	32	RW	FFFF_0000h
4874h	GPR access control (GPR_SHARED3_AUTHEN_SET)	32	RW	FFFF_0000h
4878h	GPR access control (GPR_SHARED3_AUTHEN_CLR)	32	RW	FFFF_0000h
487Ch	GPR access control (GPR_SHARED3_AUTHEN_TOG)	32	RW	FFFF_0000h
4880h	General Purpose Register (GPR_SHARED4)	32	RW	0000_0000h
4884h	General Purpose Register (GPR_SHARED4_SET)	32	RW	0000_0000h
4888h	General Purpose Register (GPR_SHARED4_CLR)	32	RW	0000_0000h
488Ch	General Purpose Register (GPR_SHARED4_TOG)	32	RW	0000_0000h
4890h	GPR access control (GPR_SHARED4_AUTHEN)	32	RW	FFFF_0000h
4894h	GPR access control (GPR_SHARED4_AUTHEN_SET)	32	RW	FFFF_0000h
4898h	GPR access control (GPR_SHARED4_AUTHEN_CLR)	32	RW	FFFF_0000h
489Ch	GPR access control (GPR_SHARED4_AUTHEN_TOG)	32	RW	FFFF_0000h
48A0h	General Purpose Register (GPR_SHARED5)	32	RW	0000_0000h
48A4h	General Purpose Register (GPR_SHARED5_SET)	32	RW	0000_0000h
48A8h	General Purpose Register (GPR_SHARED5_CLR)	32	RW	0000_0000h
48ACh	General Purpose Register (GPR_SHARED5_TOG)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
48B0h	GPR access control (GPR_SHARED5_AUTHEN)	32	RW	FFFF_0000h
48B4h	GPR access control (GPR_SHARED5_AUTHEN_SET)	32	RW	FFFF_0000h
48B8h	GPR access control (GPR_SHARED5_AUTHEN_CLR)	32	RW	FFFF_0000h
48BCh	GPR access control (GPR_SHARED5_AUTHEN_TOG)	32	RW	FFFF_0000h
48C0h	General Purpose Register (GPR_SHARED6)	32	RW	0000_0000h
48C4h	General Purpose Register (GPR_SHARED6_SET)	32	RW	0000_0000h
48C8h	General Purpose Register (GPR_SHARED6_CLR)	32	RW	0000_0000h
48CCh	General Purpose Register (GPR_SHARED6_TOG)	32	RW	0000_0000h
48D0h	GPR access control (GPR_SHARED6_AUTHEN)	32	RW	FFFF_0000h
48D4h	GPR access control (GPR_SHARED6_AUTHEN_SET)	32	RW	FFFF_0000h
48D8h	GPR access control (GPR_SHARED6_AUTHEN_CLR)	32	RW	FFFF_0000h
48DCh	GPR access control (GPR_SHARED6_AUTHEN_TOG)	32	RW	FFFF_0000h
48E0h	General Purpose Register (GPR_SHARED7)	32	RW	0000_0000h
48E4h	General Purpose Register (GPR_SHARED7_SET)	32	RW	0000_0000h
48E8h	General Purpose Register (GPR_SHARED7_CLR)	32	RW	0000_0000h
48ECh	General Purpose Register (GPR_SHARED7_TOG)	32	RW	0000_0000h
48F0h	GPR access control (GPR_SHARED7_AUTHEN)	32	RW	FFFF_0000h
48F4h	GPR access control (GPR_SHARED7_AUTHEN_SET)	32	RW	FFFF_0000h
48F8h	GPR access control (GPR_SHARED7_AUTHEN_CLR)	32	RW	FFFF_0000h
48FCh	GPR access control (GPR_SHARED7_AUTHEN_TOG)	32	RW	FFFF_0000h
4900h	General Purpose Register (GPR_SHARED8)	32	RW	0000_0000h
4904h	General Purpose Register (GPR_SHARED8_SET)	32	RW	0000_0000h
4908h	General Purpose Register (GPR_SHARED8_CLR)	32	RW	0000_0000h
490Ch	General Purpose Register (GPR_SHARED8_TOG)	32	RW	0000_0000h
4910h	GPR access control (GPR_SHARED8_AUTHEN)	32	RW	FFFF_0000h
4914h	GPR access control (GPR_SHARED8_AUTHEN_SET)	32	RW	FFFF_0000h
4918h	GPR access control (GPR_SHARED8_AUTHEN_CLR)	32	RW	FFFF_0000h
491Ch	GPR access control (GPR_SHARED8_AUTHEN_TOG)	32	RW	FFFF_0000h
4920h	General Purpose Register (GPR_SHARED9)	32	RW	0000_0000h
4924h	General Purpose Register (GPR_SHARED9_SET)	32	RW	0000_0000h
4928h	General Purpose Register (GPR_SHARED9_CLR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
492Ch	General Purpose Register (GPR_SHARED9_TOG)	32	RW	0000_0000h
4930h	GPR access control (GPR_SHARED9_AUTHEN)	32	RW	FFFF_0000h
4934h	GPR access control (GPR_SHARED9_AUTHEN_SET)	32	RW	FFFF_0000h
4938h	GPR access control (GPR_SHARED9_AUTHEN_CLR)	32	RW	FFFF_0000h
493Ch	GPR access control (GPR_SHARED9_AUTHEN_TOG)	32	RW	FFFF_0000h
4940h	General Purpose Register (GPR_SHARED10)	32	RW	0000_0000h
4944h	General Purpose Register (GPR_SHARED10_SET)	32	RW	0000_0000h
4948h	General Purpose Register (GPR_SHARED10_CLR)	32	RW	0000_0000h
494Ch	General Purpose Register (GPR_SHARED10_TOG)	32	RW	0000_0000h
4950h	GPR access control (GPR_SHARED10_AUTHEN)	32	RW	FFFF_0000h
4954h	GPR access control (GPR_SHARED10_AUTHEN_SET)	32	RW	FFFF_0000h
4958h	GPR access control (GPR_SHARED10_AUTHEN_CLR)	32	RW	FFFF_0000h
495Ch	GPR access control (GPR_SHARED10_AUTHEN_TOG)	32	RW	FFFF_0000h
4960h	General Purpose Register (GPR_SHARED11)	32	RW	0000_0000h
4964h	General Purpose Register (GPR_SHARED11_SET)	32	RW	0000_0000h
4968h	General Purpose Register (GPR_SHARED11_CLR)	32	RW	0000_0000h
496Ch	General Purpose Register (GPR_SHARED11_TOG)	32	RW	0000_0000h
4970h	GPR access control (GPR_SHARED11_AUTHEN)	32	RW	FFFF_0000h
4974h	GPR access control (GPR_SHARED11_AUTHEN_SET)	32	RW	FFFF_0000h
4978h	GPR access control (GPR_SHARED11_AUTHEN_CLR)	32	RW	FFFF_0000h
497Ch	GPR access control (GPR_SHARED11_AUTHEN_TOG)	32	RW	FFFF_0000h
4980h	General Purpose Register (GPR_SHARED12)	32	RW	0000_0000h
4984h	General Purpose Register (GPR_SHARED12_SET)	32	RW	0000_0000h
4988h	General Purpose Register (GPR_SHARED12_CLR)	32	RW	0000_0000h
498Ch	General Purpose Register (GPR_SHARED12_TOG)	32	RW	0000_0000h
4990h	GPR access control (GPR_SHARED12_AUTHEN)	32	RW	FFFF_0000h
4994h	GPR access control (GPR_SHARED12_AUTHEN_SET)	32	RW	FFFF_0000h
4998h	GPR access control (GPR_SHARED12_AUTHEN_CLR)	32	RW	FFFF_0000h
499Ch	GPR access control (GPR_SHARED12_AUTHEN_TOG)	32	RW	FFFF_0000h
49A0h	General Purpose Register (GPR_SHARED13)	32	RW	0000_0000h
49A4h	General Purpose Register (GPR_SHARED13_SET)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
49A8h	General Purpose Register (GPR_SHARED13_CLR)	32	RW	0000_0000h
49ACh	General Purpose Register (GPR_SHARED13_TOG)	32	RW	0000_0000h
49B0h	GPR access control (GPR_SHARED13_AUTHEN)	32	RW	FFFF_0000h
49B4h	GPR access control (GPR_SHARED13_AUTHEN_SET)	32	RW	FFFF_0000h
49B8h	GPR access control (GPR_SHARED13_AUTHEN_CLR)	32	RW	FFFF_0000h
49BCh	GPR access control (GPR_SHARED13_AUTHEN_TOG)	32	RW	FFFF_0000h
49C0h	General Purpose Register (GPR_SHARED14)	32	RW	0000_0000h
49C4h	General Purpose Register (GPR_SHARED14_SET)	32	RW	0000_0000h
49C8h	General Purpose Register (GPR_SHARED14_CLR)	32	RW	0000_0000h
49CCh	General Purpose Register (GPR_SHARED14_TOG)	32	RW	0000_0000h
49D0h	GPR access control (GPR_SHARED14_AUTHEN)	32	RW	FFFF_0000h
49D4h	GPR access control (GPR_SHARED14_AUTHEN_SET)	32	RW	FFFF_0000h
49D8h	GPR access control (GPR_SHARED14_AUTHEN_CLR)	32	RW	FFFF_0000h
49DCh	GPR access control (GPR_SHARED14_AUTHEN_TOG)	32	RW	FFFF_0000h
49E0h	General Purpose Register (GPR_SHARED15)	32	RW	0000_0000h
49E4h	General Purpose Register (GPR_SHARED15_SET)	32	RW	0000_0000h
49E8h	General Purpose Register (GPR_SHARED15_CLR)	32	RW	0000_0000h
49ECh	General Purpose Register (GPR_SHARED15_TOG)	32	RW	0000_0000h
49F0h	GPR access control (GPR_SHARED15_AUTHEN)	32	RW	FFFF_0000h
49F4h	GPR access control (GPR_SHARED15_AUTHEN_SET)	32	RW	FFFF_0000h
49F8h	GPR access control (GPR_SHARED15_AUTHEN_CLR)	32	RW	FFFF_0000h
49FCh	GPR access control (GPR_SHARED15_AUTHEN_TOG)	32	RW	FFFF_0000h
4A00h	General purpose status register for CM33 (GPR_SHARED_STATUS0)	32	R	FF00_0100h
4A04h	General purpose status register for CM33 (GPR_SHARED_STATUS1)	32	R	0000_0007h
4A08h	General purpose status register for CM33 (GPR_SHARED_STATUS2)	32	R	0000_0000h
4A0Ch	General purpose status register for CM33 (GPR_SHARED_STATUS3)	32	R	0000_0000h
4A10h	General status register for CM7 (GPR_SHARED_STATUS4)	32	R	FF00_0100h
4A14h	General purpose status register for CM7 (GPR_SHARED_STATUS5)	32	R	0000_0007h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4A18h	General status register for CM7 (GPR_SHARED_STATUS6)	32	R	0000_0000h
4A1Ch	General purpose status register for CM7 (GPR_SHARED_STATUS7)	32	R	0000_0000h
4C00h	General purpose register (GPR_PRIVATE0)	32	RW	0000_0000h
4C04h	General purpose register (GPR_PRIVATE0_SET)	32	RW	0000_0000h
4C08h	General purpose register (GPR_PRIVATE0_CLR)	32	RW	0000_0000h
4C0Ch	General purpose register (GPR_PRIVATE0_TOG)	32	RW	0000_0000h
4C10h	GPR access control (GPR_PRIVATE0_AUTHEN)	32	RW	FFFF_0000h
4C14h	GPR access control (GPR_PRIVATE0_AUTHEN_SET)	32	RW	FFFF_0000h
4C18h	GPR access control (GPR_PRIVATE0_AUTHEN_CLR)	32	RW	FFFF_0000h
4C1Ch	GPR access control (GPR_PRIVATE0_AUTHEN_TOG)	32	RW	FFFF_0000h
4C20h	General purpose register (GPR_PRIVATE1)	32	RW	0000_0000h
4C24h	General purpose register (GPR_PRIVATE1_SET)	32	RW	0000_0000h
4C28h	General purpose register (GPR_PRIVATE1_CLR)	32	RW	0000_0000h
4C2Ch	General purpose register (GPR_PRIVATE1_TOG)	32	RW	0000_0000h
4C30h	GPR access control (GPR_PRIVATE1_AUTHEN)	32	RW	FFFF_0000h
4C34h	GPR access control (GPR_PRIVATE1_AUTHEN_SET)	32	RW	FFFF_0000h
4C38h	GPR access control (GPR_PRIVATE1_AUTHEN_CLR)	32	RW	FFFF_0000h
4C3Ch	GPR access control (GPR_PRIVATE1_AUTHEN_TOG)	32	RW	FFFF_0000h
4C40h	General purpose register (GPR_PRIVATE2)	32	RW	0000_0000h
4C44h	General purpose register (GPR_PRIVATE2_SET)	32	RW	0000_0000h
4C48h	General purpose register (GPR_PRIVATE2_CLR)	32	RW	0000_0000h
4C4Ch	General purpose register (GPR_PRIVATE2_TOG)	32	RW	0000_0000h
4C50h	GPR access control (GPR_PRIVATE2_AUTHEN)	32	RW	FFFF_0000h
4C54h	GPR access control (GPR_PRIVATE2_AUTHEN_SET)	32	RW	FFFF_0000h
4C58h	GPR access control (GPR_PRIVATE2_AUTHEN_CLR)	32	RW	FFFF_0000h
4C5Ch	GPR access control (GPR_PRIVATE2_AUTHEN_TOG)	32	RW	FFFF_0000h
4C60h	General purpose register (GPR_PRIVATE3)	32	RW	0000_0000h
4C64h	General purpose register (GPR_PRIVATE3_SET)	32	RW	0000_0000h
4C68h	General purpose register (GPR_PRIVATE3_CLR)	32	RW	0000_0000h
4C6Ch	General purpose register (GPR_PRIVATE3_TOG)	32	RW	0000_0000h
4C70h	GPR access control (GPR_PRIVATE3_AUTHEN)	32	RW	FFFF_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4C74h	GPR access control (GPR_PRIVATE3_AUTHEN_SET)	32	RW	FFFF_0000h
4C78h	GPR access control (GPR_PRIVATE3_AUTHEN_CLR)	32	RW	FFFF_0000h
4C7Ch	GPR access control (GPR_PRIVATE3_AUTHEN_TOG)	32	RW	FFFF_0000h
5000h	Clock source direct control (OSCPLL0_DIRECT)	32	RW	0000_0001h
5010h	Clock source low power mode setting (OSCPLL0_LPM0)	32	RW	0000_0000h
5014h	clock source low power mode setting (OSCPLL0_LPM1)	32	RW	0000_0000h
501Ch	LPM setting of current CPU domain (OSCPLL0_LPM_CUR)	32	RW	0000_0000h
5020h	Clock source working status (OSCPLL0_STATUS0)	32	R	0000_0001h
5024h	Clock source domain status (OSCPLL0_STATUS1)	32	R	0000_FFFFh
5030h	Clock Source access control (OSCPLL0_AUTHEN)	32	RW	FFFF_0000h
5040h	Clock source direct control (OSCPLL1_DIRECT)	32	RW	0000_0001h
5050h	Clock source low power mode setting (OSCPLL1_LPM0)	32	RW	0000_0000h
5054h	clock source low power mode setting (OSCPLL1_LPM1)	32	RW	0000_0000h
505Ch	LPM setting of current CPU domain (OSCPLL1_LPM_CUR)	32	RW	0000_0000h
5060h	Clock source working status (OSCPLL1_STATUS0)	32	R	0000_0001h
5064h	Clock source domain status (OSCPLL1_STATUS1)	32	R	0000_FFFFh
5070h	Clock Source access control (OSCPLL1_AUTHEN)	32	RW	FFFF_0000h
5080h	Clock source direct control (OSCPLL2_DIRECT)	32	RW	0000_0001h
5090h	Clock source low power mode setting (OSCPLL2_LPM0)	32	RW	0000_0000h
5094h	clock source low power mode setting (OSCPLL2_LPM1)	32	RW	0000_0000h
509Ch	LPM setting of current CPU domain (OSCPLL2_LPM_CUR)	32	RW	0000_0000h
50A0h	Clock source working status (OSCPLL2_STATUS0)	32	R	0000_0001h
50A4h	Clock source domain status (OSCPLL2_STATUS1)	32	R	0000_FFFFh
50B0h	Clock Source access control (OSCPLL2_AUTHEN)	32	RW	FFFF_0000h
50C0h	Clock source direct control (OSCPLL3_DIRECT)	32	RW	0000_0001h
50D0h	Clock source low power mode setting (OSCPLL3_LPM0)	32	RW	0000_0000h
50D4h	clock source low power mode setting (OSCPLL3_LPM1)	32	RW	0000_0000h
50DCh	LPM setting of current CPU domain (OSCPLL3_LPM_CUR)	32	RW	0000_0000h
50E0h	Clock source working status (OSCPLL3_STATUS0)	32	R	0000_0001h
50E4h	Clock source domain status (OSCPLL3_STATUS1)	32	R	0000_FFFFh
50F0h	Clock Source access control (OSCPLL3_AUTHEN)	32	RW	FFFF_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5100h	Clock source direct control (OSCPLL4_DIRECT)	32	RW	0000_0001h
5110h	Clock source low power mode setting (OSCPLL4_LPM0)	32	RW	0000_0000h
5114h	clock source low power mode setting (OSCPLL4_LPM1)	32	RW	0000_0000h
511Ch	LPM setting of current CPU domain (OSCPLL4_LPM_CUR)	32	RW	0000_0000h
5120h	Clock source working status (OSCPLL4_STATUS0)	32	R	0000_0001h
5124h	Clock source domain status (OSCPLL4_STATUS1)	32	R	0000_FFFFh
5130h	Clock Source access control (OSCPLL4_AUTHEN)	32	RW	FFFF_0000h
5140h	Clock source direct control (OSCPLL5_DIRECT)	32	RW	0000_0001h
5150h	Clock source low power mode setting (OSCPLL5_LPM0)	32	RW	0000_0000h
5154h	clock source low power mode setting (OSCPLL5_LPM1)	32	RW	0000_0000h
515Ch	LPM setting of current CPU domain (OSCPLL5_LPM_CUR)	32	RW	0000_0000h
5160h	Clock source working status (OSCPLL5_STATUS0)	32	R	0000_0001h
5164h	Clock source domain status (OSCPLL5_STATUS1)	32	R	0000_FFFFh
5170h	Clock Source access control (OSCPLL5_AUTHEN)	32	RW	FFFF_0000h
5180h	Clock source direct control (OSCPLL6_DIRECT)	32	RW	0000_0001h
5190h	Clock source low power mode setting (OSCPLL6_LPM0)	32	RW	0000_0000h
5194h	clock source low power mode setting (OSCPLL6_LPM1)	32	RW	0000_0000h
519Ch	LPM setting of current CPU domain (OSCPLL6_LPM_CUR)	32	RW	0000_0000h
51A0h	Clock source working status (OSCPLL6_STATUS0)	32	R	0000_0001h
51A4h	Clock source domain status (OSCPLL6_STATUS1)	32	R	0000_FFFFh
51B0h	Clock Source access control (OSCPLL6_AUTHEN)	32	RW	FFFF_0000h
51C0h	Clock source direct control (OSCPLL7_DIRECT)	32	RW	0000_0001h
51D0h	Clock source low power mode setting (OSCPLL7_LPM0)	32	RW	0000_0000h
51D4h	clock source low power mode setting (OSCPLL7_LPM1)	32	RW	0000_0000h
51DCh	LPM setting of current CPU domain (OSCPLL7_LPM_CUR)	32	RW	0000_0000h
51E0h	Clock source working status (OSCPLL7_STATUS0)	32	R	0000_0001h
51E4h	Clock source domain status (OSCPLL7_STATUS1)	32	R	0000_FFFFh
51F0h	Clock Source access control (OSCPLL7_AUTHEN)	32	RW	FFFF_0000h
5200h	Clock source direct control (OSCPLL8_DIRECT)	32	RW	0000_0001h
5210h	Clock source low power mode setting (OSCPLL8_LPM0)	32	RW	0000_0000h
5214h	clock source low power mode setting (OSCPLL8_LPM1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
521Ch	LPM setting of current CPU domain (OSCPLL8_LPM_CUR)	32	RW	0000_0000h
5220h	Clock source working status (OSCPLL8_STATUS0)	32	R	0000_0001h
5224h	Clock source domain status (OSCPLL8_STATUS1)	32	R	0000_FFFFh
5230h	Clock Source access control (OSCPLL8_AUTHEN)	32	RW	FFFF_0000h
5240h	Clock source direct control (OSCPLL9_DIRECT)	32	RW	0000_0001h
5250h	Clock source low power mode setting (OSCPLL9_LPM0)	32	RW	0000_0000h
5254h	clock source low power mode setting (OSCPLL9_LPM1)	32	RW	0000_0000h
525Ch	LPM setting of current CPU domain (OSCPLL9_LPM_CUR)	32	RW	0000_0000h
5260h	Clock source working status (OSCPLL9_STATUS0)	32	R	0000_0001h
5264h	Clock source domain status (OSCPLL9_STATUS1)	32	R	0000_FFFFh
5270h	Clock Source access control (OSCPLL9_AUTHEN)	32	RW	FFFF_0000h
5280h	Clock source direct control (OSCPLL10_DIRECT)	32	RW	0000_0001h
5290h	Clock source low power mode setting (OSCPLL10_LPM0)	32	RW	0000_0000h
5294h	clock source low power mode setting (OSCPLL10_LPM1)	32	RW	0000_0000h
529Ch	LPM setting of current CPU domain (OSCPLL10_LPM_CUR)	32	RW	0000_0000h
52A0h	Clock source working status (OSCPLL10_STATUS0)	32	R	0000_0001h
52A4h	Clock source domain status (OSCPLL10_STATUS1)	32	R	0000_FFFFh
52B0h	Clock Source access control (OSCPLL10_AUTHEN)	32	RW	FFFF_0000h
52C0h	Clock source direct control (OSCPLL11_DIRECT)	32	RW	0000_0001h
52D0h	Clock source low power mode setting (OSCPLL11_LPM0)	32	RW	0000_0000h
52D4h	clock source low power mode setting (OSCPLL11_LPM1)	32	RW	0000_0000h
52DCh	LPM setting of current CPU domain (OSCPLL11_LPM_CUR)	32	RW	0000_0000h
52E0h	Clock source working status (OSCPLL11_STATUS0)	32	R	0000_0001h
52E4h	Clock source domain status (OSCPLL11_STATUS1)	32	R	0000_FFFFh
52F0h	Clock Source access control (OSCPLL11_AUTHEN)	32	RW	FFFF_0000h
5300h	Clock source direct control (OSCPLL12_DIRECT)	32	RW	0000_0001h
5310h	Clock source low power mode setting (OSCPLL12_LPM0)	32	RW	0000_0000h
5314h	clock source low power mode setting (OSCPLL12_LPM1)	32	RW	0000_0000h
531Ch	LPM setting of current CPU domain (OSCPLL12_LPM_CUR)	32	RW	0000_0000h
5320h	Clock source working status (OSCPLL12_STATUS0)	32	R	0000_0001h
5324h	Clock source domain status (OSCPLL12_STATUS1)	32	R	0000_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5330h	Clock Source access control (OSCPLL12_AUTHEN)	32	RW	FFFF_0000h
5340h	Clock source direct control (OSCPLL13_DIRECT)	32	RW	0000_0001h
5350h	Clock source low power mode setting (OSCPLL13_LPM0)	32	RW	0000_0000h
5354h	clock source low power mode setting (OSCPLL13_LPM1)	32	RW	0000_0000h
535Ch	LPM setting of current CPU domain (OSCPLL13_LPM_CUR)	32	RW	0000_0000h
5360h	Clock source working status (OSCPLL13_STATUS0)	32	R	0000_0001h
5364h	Clock source domain status (OSCPLL13_STATUS1)	32	R	0000_FFFFh
5370h	Clock Source access control (OSCPLL13_AUTHEN)	32	RW	FFFF_0000h
5380h	Clock source direct control (OSCPLL14_DIRECT)	32	RW	0000_0001h
5390h	Clock source low power mode setting (OSCPLL14_LPM0)	32	RW	0000_0000h
5394h	clock source low power mode setting (OSCPLL14_LPM1)	32	RW	0000_0000h
539Ch	LPM setting of current CPU domain (OSCPLL14_LPM_CUR)	32	RW	0000_0000h
53A0h	Clock source working status (OSCPLL14_STATUS0)	32	R	0000_0001h
53A4h	Clock source domain status (OSCPLL14_STATUS1)	32	R	0000_FFFFh
53B0h	Clock Source access control (OSCPLL14_AUTHEN)	32	RW	FFFF_0000h
53C0h	Clock source direct control (OSCPLL15_DIRECT)	32	RW	0000_0001h
53D0h	Clock source low power mode setting (OSCPLL15_LPM0)	32	RW	0000_0000h
53D4h	clock source low power mode setting (OSCPLL15_LPM1)	32	RW	0000_0000h
53DCh	LPM setting of current CPU domain (OSCPLL15_LPM_CUR)	32	RW	0000_0000h
53E0h	Clock source working status (OSCPLL15_STATUS0)	32	R	0000_0001h
53E4h	Clock source domain status (OSCPLL15_STATUS1)	32	R	0000_FFFFh
53F0h	Clock Source access control (OSCPLL15_AUTHEN)	32	RW	FFFF_0000h
5400h	Clock source direct control (OSCPLL16_DIRECT)	32	RW	0000_0001h
5410h	Clock source low power mode setting (OSCPLL16_LPM0)	32	RW	0000_0000h
5414h	clock source low power mode setting (OSCPLL16_LPM1)	32	RW	0000_0000h
541Ch	LPM setting of current CPU domain (OSCPLL16_LPM_CUR)	32	RW	0000_0000h
5420h	Clock source working status (OSCPLL16_STATUS0)	32	R	0000_0001h
5424h	Clock source domain status (OSCPLL16_STATUS1)	32	R	0000_FFFFh
5430h	Clock Source access control (OSCPLL16_AUTHEN)	32	RW	FFFF_0000h
5440h	Clock source direct control (OSCPLL17_DIRECT)	32	RW	0000_0001h
5450h	Clock source low power mode setting (OSCPLL17_LPM0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5454h	clock source low power mode setting (OSCPLL17_LPM1)	32	RW	0000_0000h
545Ch	LPM setting of current CPU domain (OSCPLL17_LPM_CUR)	32	RW	0000_0000h
5460h	Clock source working status (OSCPLL17_STATUS0)	32	R	0000_0001h
5464h	Clock source domain status (OSCPLL17_STATUS1)	32	R	0000_FFFFh
5470h	Clock Source access control (OSCPLL17_AUTHEN)	32	RW	FFFF_0000h
5480h	Clock source direct control (OSCPLL18_DIRECT)	32	RW	0000_0001h
5490h	Clock source low power mode setting (OSCPLL18_LPM0)	32	RW	0000_0000h
5494h	clock source low power mode setting (OSCPLL18_LPM1)	32	RW	0000_0000h
549Ch	LPM setting of current CPU domain (OSCPLL18_LPM_CUR)	32	RW	0000_0000h
54A0h	Clock source working status (OSCPLL18_STATUS0)	32	R	0000_0001h
54A4h	Clock source domain status (OSCPLL18_STATUS1)	32	R	0000_FFFFh
54B0h	Clock Source access control (OSCPLL18_AUTHEN)	32	RW	FFFF_0000h
54C0h	Clock source direct control (OSCPLL19_DIRECT)	32	RW	0000_0001h
54D0h	Clock source low power mode setting (OSCPLL19_LPM0)	32	RW	0000_0000h
54D4h	clock source low power mode setting (OSCPLL19_LPM1)	32	RW	0000_0000h
54DCh	LPM setting of current CPU domain (OSCPLL19_LPM_CUR)	32	RW	0000_0000h
54E0h	Clock source working status (OSCPLL19_STATUS0)	32	R	0000_0001h
54E4h	Clock source domain status (OSCPLL19_STATUS1)	32	R	0000_FFFFh
54F0h	Clock Source access control (OSCPLL19_AUTHEN)	32	RW	FFFF_0000h
5500h	Clock source direct control (OSCPLL20_DIRECT)	32	RW	0000_0001h
5510h	Clock source low power mode setting (OSCPLL20_LPM0)	32	RW	0000_0000h
5514h	clock source low power mode setting (OSCPLL20_LPM1)	32	RW	0000_0000h
551Ch	LPM setting of current CPU domain (OSCPLL20_LPM_CUR)	32	RW	0000_0000h
5520h	Clock source working status (OSCPLL20_STATUS0)	32	R	0000_0001h
5524h	Clock source domain status (OSCPLL20_STATUS1)	32	R	0000_FFFFh
5530h	Clock Source access control (OSCPLL20_AUTHEN)	32	RW	FFFF_0000h
5540h	Clock source direct control (OSCPLL21_DIRECT)	32	RW	0000_0001h
5550h	Clock source low power mode setting (OSCPLL21_LPM0)	32	RW	0000_0000h
5554h	clock source low power mode setting (OSCPLL21_LPM1)	32	RW	0000_0000h
555Ch	LPM setting of current CPU domain (OSCPLL21_LPM_CUR)	32	RW	0000_0000h
5560h	Clock source working status (OSCPLL21_STATUS0)	32	R	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5564h	Clock source domain status (OSCPLL21_STATUS1)	32	R	0000_FFFFh
5570h	Clock Source access control (OSCPLL21_AUTHEN)	32	RW	FFFF_0000h
5580h	Clock source direct control (OSCPLL22_DIRECT)	32	RW	0000_0001h
5590h	Clock source low power mode setting (OSCPLL22_LPM0)	32	RW	0000_0000h
5594h	clock source low power mode setting (OSCPLL22_LPM1)	32	RW	0000_0000h
559Ch	LPM setting of current CPU domain (OSCPLL22_LPM_CUR)	32	RW	0000_0000h
55A0h	Clock source working status (OSCPLL22_STATUS0)	32	R	0000_0001h
55A4h	Clock source domain status (OSCPLL22_STATUS1)	32	R	0000_FFFFh
55B0h	Clock Source access control (OSCPLL22_AUTHEN)	32	RW	FFFF_0000h
55C0h	Clock source direct control (OSCPLL23_DIRECT)	32	RW	0000_0001h
55D0h	Clock source low power mode setting (OSCPLL23_LPM0)	32	RW	0000_0000h
55D4h	clock source low power mode setting (OSCPLL23_LPM1)	32	RW	0000_0000h
55DCh	LPM setting of current CPU domain (OSCPLL23_LPM_CUR)	32	RW	0000_0000h
55E0h	Clock source working status (OSCPLL23_STATUS0)	32	R	0000_0001h
55E4h	Clock source domain status (OSCPLL23_STATUS1)	32	R	0000_FFFFh
55F0h	Clock Source access control (OSCPLL23_AUTHEN)	32	RW	FFFF_0000h
5600h	Clock source direct control (OSCPLL24_DIRECT)	32	RW	0000_0001h
5610h	Clock source low power mode setting (OSCPLL24_LPM0)	32	RW	0000_0000h
5614h	clock source low power mode setting (OSCPLL24_LPM1)	32	RW	0000_0000h
561Ch	LPM setting of current CPU domain (OSCPLL24_LPM_CUR)	32	RW	0000_0000h
5620h	Clock source working status (OSCPLL24_STATUS0)	32	R	0000_0001h
5624h	Clock source domain status (OSCPLL24_STATUS1)	32	R	0000_FFFFh
5630h	Clock Source access control (OSCPLL24_AUTHEN)	32	RW	FFFF_0000h
8000h	LPCG direct control (LPCG0_DIRECT)	32	RW	0000_0001h
8010h	Clock source low power mode setting (LPCG0_LPM0)	32	RW	0000_0000h
8014h	clock source low power mode setting (LPCG0_LPM1)	32	RW	0000_0000h
801Ch	LPM setting of current CPU domain (LPCG0_LPM_CUR)	32	RW	0000_0000h
8020h	LPCG working status (LPCG0_STATUS0)	32	R	0000_0001h
8024h	LPCG domain status (LPCG0_STATUS1)	32	R	0000_FFFFh
8030h	LPCG access control (LPCG0_AUTHEN)	32	RW	FFFF_0000h
8040h	LPCG direct control (LPCG1_DIRECT)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8050h	Clock source low power mode setting (LPCG1_LPM0)	32	RW	0000_0000h
8054h	clock source low power mode setting (LPCG1_LPM1)	32	RW	0000_0000h
805Ch	LPM setting of current CPU domain (LPCG1_LPM_CUR)	32	RW	0000_0000h
8060h	LPCG working status (LPCG1_STATUS0)	32	R	0000_0001h
8064h	LPCG domain status (LPCG1_STATUS1)	32	R	0000_FFFFh
8070h	LPCG access control (LPCG1_AUTHEN)	32	RW	FFFF_0000h
8080h	LPCG direct control (LPCG2_DIRECT)	32	RW	0000_0001h
8090h	Clock source low power mode setting (LPCG2_LPM0)	32	RW	0000_0000h
8094h	clock source low power mode setting (LPCG2_LPM1)	32	RW	0000_0000h
809Ch	LPM setting of current CPU domain (LPCG2_LPM_CUR)	32	RW	0000_0000h
80A0h	LPCG working status (LPCG2_STATUS0)	32	R	0000_0001h
80A4h	LPCG domain status (LPCG2_STATUS1)	32	R	0000_FFFFh
80B0h	LPCG access control (LPCG2_AUTHEN)	32	RW	FFFF_0000h
80C0h	LPCG direct control (LPCG3_DIRECT)	32	RW	0000_0001h
80D0h	Clock source low power mode setting (LPCG3_LPM0)	32	RW	0000_0000h
80D4h	clock source low power mode setting (LPCG3_LPM1)	32	RW	0000_0000h
80DCh	LPM setting of current CPU domain (LPCG3_LPM_CUR)	32	RW	0000_0000h
80E0h	LPCG working status (LPCG3_STATUS0)	32	R	0000_0001h
80E4h	LPCG domain status (LPCG3_STATUS1)	32	R	0000_FFFFh
80F0h	LPCG access control (LPCG3_AUTHEN)	32	RW	FFFF_0000h
8100h	LPCG direct control (LPCG4_DIRECT)	32	RW	0000_0001h
8110h	Clock source low power mode setting (LPCG4_LPM0)	32	RW	0000_0000h
8114h	clock source low power mode setting (LPCG4_LPM1)	32	RW	0000_0000h
811Ch	LPM setting of current CPU domain (LPCG4_LPM_CUR)	32	RW	0000_0000h
8120h	LPCG working status (LPCG4_STATUS0)	32	R	0000_0001h
8124h	LPCG domain status (LPCG4_STATUS1)	32	R	0000_FFFFh
8130h	LPCG access control (LPCG4_AUTHEN)	32	RW	FFFF_0000h
8140h	LPCG direct control (LPCG5_DIRECT)	32	RW	0000_0001h
8150h	Clock source low power mode setting (LPCG5_LPM0)	32	RW	0000_0000h
8154h	clock source low power mode setting (LPCG5_LPM1)	32	RW	0000_0000h
815Ch	LPM setting of current CPU domain (LPCG5_LPM_CUR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8160h	LPCG working status (LPCG5_STATUS0)	32	R	0000_0001h
8164h	LPCG domain status (LPCG5_STATUS1)	32	R	0000_FFFFh
8170h	LPCG access control (LPCG5_AUTHEN)	32	RW	FFFF_0000h
8180h	LPCG direct control (LPCG6_DIRECT)	32	RW	0000_0001h
8190h	Clock source low power mode setting (LPCG6_LPM0)	32	RW	0000_0000h
8194h	clock source low power mode setting (LPCG6_LPM1)	32	RW	0000_0000h
819Ch	LPM setting of current CPU domain (LPCG6_LPM_CUR)	32	RW	0000_0000h
81A0h	LPCG working status (LPCG6_STATUS0)	32	R	0000_0001h
81A4h	LPCG domain status (LPCG6_STATUS1)	32	R	0000_FFFFh
81B0h	LPCG access control (LPCG6_AUTHEN)	32	RW	FFFF_0000h
81C0h	LPCG direct control (LPCG7_DIRECT)	32	RW	0000_0001h
81D0h	Clock source low power mode setting (LPCG7_LPM0)	32	RW	0000_0000h
81D4h	clock source low power mode setting (LPCG7_LPM1)	32	RW	0000_0000h
81DCh	LPM setting of current CPU domain (LPCG7_LPM_CUR)	32	RW	0000_0000h
81E0h	LPCG working status (LPCG7_STATUS0)	32	R	0000_0001h
81E4h	LPCG domain status (LPCG7_STATUS1)	32	R	0000_FFFFh
81F0h	LPCG access control (LPCG7_AUTHEN)	32	RW	FFFF_0000h
8200h	LPCG direct control (LPCG8_DIRECT)	32	RW	0000_0001h
8210h	Clock source low power mode setting (LPCG8_LPM0)	32	RW	0000_0000h
8214h	clock source low power mode setting (LPCG8_LPM1)	32	RW	0000_0000h
821Ch	LPM setting of current CPU domain (LPCG8_LPM_CUR)	32	RW	0000_0000h
8220h	LPCG working status (LPCG8_STATUS0)	32	R	0000_0001h
8224h	LPCG domain status (LPCG8_STATUS1)	32	R	0000_FFFFh
8230h	LPCG access control (LPCG8_AUTHEN)	32	RW	FFFF_0000h
8240h	LPCG direct control (LPCG9_DIRECT)	32	RW	0000_0001h
8250h	Clock source low power mode setting (LPCG9_LPM0)	32	RW	0000_0000h
8254h	clock source low power mode setting (LPCG9_LPM1)	32	RW	0000_0000h
825Ch	LPM setting of current CPU domain (LPCG9_LPM_CUR)	32	RW	0000_0000h
8260h	LPCG working status (LPCG9_STATUS0)	32	R	0000_0001h
8264h	LPCG domain status (LPCG9_STATUS1)	32	R	0000_FFFFh
8270h	LPCG access control (LPCG9_AUTHEN)	32	RW	FFFF_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8280h	LPCG direct control (LPCG10_DIRECT)	32	RW	0000_0001h
8290h	Clock source low power mode setting (LPCG10_LPM0)	32	RW	0000_0000h
8294h	clock source low power mode setting (LPCG10_LPM1)	32	RW	0000_0000h
829Ch	LPM setting of current CPU domain (LPCG10_LPM_CUR)	32	RW	0000_0000h
82A0h	LPCG working status (LPCG10_STATUS0)	32	R	0000_0001h
82A4h	LPCG domain status (LPCG10_STATUS1)	32	R	0000_FFFFh
82B0h	LPCG access control (LPCG10_AUTHEN)	32	RW	FFFF_0000h
82C0h	LPCG direct control (LPCG11_DIRECT)	32	RW	0000_0001h
82D0h	Clock source low power mode setting (LPCG11_LPM0)	32	RW	0000_0000h
82D4h	clock source low power mode setting (LPCG11_LPM1)	32	RW	0000_0000h
82DCh	LPM setting of current CPU domain (LPCG11_LPM_CUR)	32	RW	0000_0000h
82E0h	LPCG working status (LPCG11_STATUS0)	32	R	0000_0001h
82E4h	LPCG domain status (LPCG11_STATUS1)	32	R	0000_FFFFh
82F0h	LPCG access control (LPCG11_AUTHEN)	32	RW	FFFF_0000h
8300h	LPCG direct control (LPCG12_DIRECT)	32	RW	0000_0001h
8310h	Clock source low power mode setting (LPCG12_LPM0)	32	RW	0000_0000h
8314h	clock source low power mode setting (LPCG12_LPM1)	32	RW	0000_0000h
831Ch	LPM setting of current CPU domain (LPCG12_LPM_CUR)	32	RW	0000_0000h
8320h	LPCG working status (LPCG12_STATUS0)	32	R	0000_0001h
8324h	LPCG domain status (LPCG12_STATUS1)	32	R	0000_FFFFh
8330h	LPCG access control (LPCG12_AUTHEN)	32	RW	FFFF_0000h
8340h	LPCG direct control (LPCG13_DIRECT)	32	RW	0000_0001h
8350h	Clock source low power mode setting (LPCG13_LPM0)	32	RW	0000_0000h
8354h	clock source low power mode setting (LPCG13_LPM1)	32	RW	0000_0000h
835Ch	LPM setting of current CPU domain (LPCG13_LPM_CUR)	32	RW	0000_0000h
8360h	LPCG working status (LPCG13_STATUS0)	32	R	0000_0001h
8364h	LPCG domain status (LPCG13_STATUS1)	32	R	0000_FFFFh
8370h	LPCG access control (LPCG13_AUTHEN)	32	RW	FFFF_0000h
8380h	LPCG direct control (LPCG14_DIRECT)	32	RW	0000_0001h
8390h	Clock source low power mode setting (LPCG14_LPM0)	32	RW	0000_0000h
8394h	clock source low power mode setting (LPCG14_LPM1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
839Ch	LPM setting of current CPU domain (LPCG14_LPM_CUR)	32	RW	0000_0000h
83A0h	LPCG working status (LPCG14_STATUS0)	32	R	0000_0001h
83A4h	LPCG domain status (LPCG14_STATUS1)	32	R	0000_FFFFh
83B0h	LPCG access control (LPCG14_AUTHEN)	32	RW	FFFF_0000h
83C0h	LPCG direct control (LPCG15_DIRECT)	32	RW	0000_0001h
83D0h	Clock source low power mode setting (LPCG15_LPM0)	32	RW	0000_0000h
83D4h	clock source low power mode setting (LPCG15_LPM1)	32	RW	0000_0000h
83DCh	LPM setting of current CPU domain (LPCG15_LPM_CUR)	32	RW	0000_0000h
83E0h	LPCG working status (LPCG15_STATUS0)	32	R	0000_0001h
83E4h	LPCG domain status (LPCG15_STATUS1)	32	R	0000_FFFFh
83F0h	LPCG access control (LPCG15_AUTHEN)	32	RW	FFFF_0000h
8400h	LPCG direct control (LPCG16_DIRECT)	32	RW	0000_0001h
8410h	Clock source low power mode setting (LPCG16_LPM0)	32	RW	0000_0000h
8414h	clock source low power mode setting (LPCG16_LPM1)	32	RW	0000_0000h
841Ch	LPM setting of current CPU domain (LPCG16_LPM_CUR)	32	RW	0000_0000h
8420h	LPCG working status (LPCG16_STATUS0)	32	R	0000_0001h
8424h	LPCG domain status (LPCG16_STATUS1)	32	R	0000_FFFFh
8430h	LPCG access control (LPCG16_AUTHEN)	32	RW	FFFF_0000h
8440h	LPCG direct control (LPCG17_DIRECT)	32	RW	0000_0001h
8450h	Clock source low power mode setting (LPCG17_LPM0)	32	RW	0000_0000h
8454h	clock source low power mode setting (LPCG17_LPM1)	32	RW	0000_0000h
845Ch	LPM setting of current CPU domain (LPCG17_LPM_CUR)	32	RW	0000_0000h
8460h	LPCG working status (LPCG17_STATUS0)	32	R	0000_0001h
8464h	LPCG domain status (LPCG17_STATUS1)	32	R	0000_FFFFh
8470h	LPCG access control (LPCG17_AUTHEN)	32	RW	FFFF_0000h
8480h	LPCG direct control (LPCG18_DIRECT)	32	RW	0000_0001h
8490h	Clock source low power mode setting (LPCG18_LPM0)	32	RW	0000_0000h
8494h	clock source low power mode setting (LPCG18_LPM1)	32	RW	0000_0000h
849Ch	LPM setting of current CPU domain (LPCG18_LPM_CUR)	32	RW	0000_0000h
84A0h	LPCG working status (LPCG18_STATUS0)	32	R	0000_0001h
84A4h	LPCG domain status (LPCG18_STATUS1)	32	R	0000_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
84B0h	LPCG access control (LPCG18_AUTHEN)	32	RW	FFFF_0000h
84C0h	LPCG direct control (LPCG19_DIRECT)	32	RW	0000_0001h
84D0h	Clock source low power mode setting (LPCG19_LPM0)	32	RW	0000_0000h
84D4h	clock source low power mode setting (LPCG19_LPM1)	32	RW	0000_0000h
84DCh	LPM setting of current CPU domain (LPCG19_LPM_CUR)	32	RW	0000_0000h
84E0h	LPCG working status (LPCG19_STATUS0)	32	R	0000_0001h
84E4h	LPCG domain status (LPCG19_STATUS1)	32	R	0000_FFFFh
84F0h	LPCG access control (LPCG19_AUTHEN)	32	RW	FFFF_0000h
8500h	LPCG direct control (LPCG20_DIRECT)	32	RW	0000_0001h
8510h	Clock source low power mode setting (LPCG20_LPM0)	32	RW	0000_0000h
8514h	clock source low power mode setting (LPCG20_LPM1)	32	RW	0000_0000h
851Ch	LPM setting of current CPU domain (LPCG20_LPM_CUR)	32	RW	0000_0000h
8520h	LPCG working status (LPCG20_STATUS0)	32	R	0000_0001h
8524h	LPCG domain status (LPCG20_STATUS1)	32	R	0000_FFFFh
8530h	LPCG access control (LPCG20_AUTHEN)	32	RW	FFFF_0000h
8540h	LPCG direct control (LPCG21_DIRECT)	32	RW	0000_0001h
8550h	Clock source low power mode setting (LPCG21_LPM0)	32	RW	0000_0000h
8554h	clock source low power mode setting (LPCG21_LPM1)	32	RW	0000_0000h
855Ch	LPM setting of current CPU domain (LPCG21_LPM_CUR)	32	RW	0000_0000h
8560h	LPCG working status (LPCG21_STATUS0)	32	R	0000_0001h
8564h	LPCG domain status (LPCG21_STATUS1)	32	R	0000_FFFFh
8570h	LPCG access control (LPCG21_AUTHEN)	32	RW	FFFF_0000h
8580h	LPCG direct control (LPCG22_DIRECT)	32	RW	0000_0001h
8590h	Clock source low power mode setting (LPCG22_LPM0)	32	RW	0000_0000h
8594h	clock source low power mode setting (LPCG22_LPM1)	32	RW	0000_0000h
859Ch	LPM setting of current CPU domain (LPCG22_LPM_CUR)	32	RW	0000_0000h
85A0h	LPCG working status (LPCG22_STATUS0)	32	R	0000_0001h
85A4h	LPCG domain status (LPCG22_STATUS1)	32	R	0000_FFFFh
85B0h	LPCG access control (LPCG22_AUTHEN)	32	RW	FFFF_0000h
85C0h	LPCG direct control (LPCG23_DIRECT)	32	RW	0000_0001h
85D0h	Clock source low power mode setting (LPCG23_LPM0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
85D4h	clock source low power mode setting (LPCG23_LPM1)	32	RW	0000_0000h
85DCh	LPM setting of current CPU domain (LPCG23_LPM_CUR)	32	RW	0000_0000h
85E0h	LPCG working status (LPCG23_STATUS0)	32	R	0000_0001h
85E4h	LPCG domain status (LPCG23_STATUS1)	32	R	0000_FFFFh
85F0h	LPCG access control (LPCG23_AUTHEN)	32	RW	FFFF_0000h
8600h	LPCG direct control (LPCG24_DIRECT)	32	RW	0000_0001h
8610h	Clock source low power mode setting (LPCG24_LPM0)	32	RW	0000_0000h
8614h	clock source low power mode setting (LPCG24_LPM1)	32	RW	0000_0000h
861Ch	LPM setting of current CPU domain (LPCG24_LPM_CUR)	32	RW	0000_0000h
8620h	LPCG working status (LPCG24_STATUS0)	32	R	0000_0001h
8624h	LPCG domain status (LPCG24_STATUS1)	32	R	0000_FFFFh
8630h	LPCG access control (LPCG24_AUTHEN)	32	RW	FFFF_0000h
8640h	LPCG direct control (LPCG25_DIRECT)	32	RW	0000_0001h
8650h	Clock source low power mode setting (LPCG25_LPM0)	32	RW	0000_0000h
8654h	clock source low power mode setting (LPCG25_LPM1)	32	RW	0000_0000h
865Ch	LPM setting of current CPU domain (LPCG25_LPM_CUR)	32	RW	0000_0000h
8660h	LPCG working status (LPCG25_STATUS0)	32	R	0000_0001h
8664h	LPCG domain status (LPCG25_STATUS1)	32	R	0000_FFFFh
8670h	LPCG access control (LPCG25_AUTHEN)	32	RW	FFFF_0000h
8680h	LPCG direct control (LPCG26_DIRECT)	32	RW	0000_0001h
8690h	Clock source low power mode setting (LPCG26_LPM0)	32	RW	0000_0000h
8694h	clock source low power mode setting (LPCG26_LPM1)	32	RW	0000_0000h
869Ch	LPM setting of current CPU domain (LPCG26_LPM_CUR)	32	RW	0000_0000h
86A0h	LPCG working status (LPCG26_STATUS0)	32	R	0000_0001h
86A4h	LPCG domain status (LPCG26_STATUS1)	32	R	0000_FFFFh
86B0h	LPCG access control (LPCG26_AUTHEN)	32	RW	FFFF_0000h
86C0h	LPCG direct control (LPCG27_DIRECT)	32	RW	0000_0001h
86D0h	Clock source low power mode setting (LPCG27_LPM0)	32	RW	0000_0000h
86D4h	clock source low power mode setting (LPCG27_LPM1)	32	RW	0000_0000h
86DCh	LPM setting of current CPU domain (LPCG27_LPM_CUR)	32	RW	0000_0000h
86E0h	LPCG working status (LPCG27_STATUS0)	32	R	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
86E4h	LPCG domain status (LPCG27_STATUS1)	32	R	0000_FFFFh
86F0h	LPCG access control (LPCG27_AUTHEN)	32	RW	FFFF_0000h
8700h	LPCG direct control (LPCG28_DIRECT)	32	RW	0000_0001h
8710h	Clock source low power mode setting (LPCG28_LPM0)	32	RW	0000_0000h
8714h	clock source low power mode setting (LPCG28_LPM1)	32	RW	0000_0000h
871Ch	LPM setting of current CPU domain (LPCG28_LPM_CUR)	32	RW	0000_0000h
8720h	LPCG working status (LPCG28_STATUS0)	32	R	0000_0001h
8724h	LPCG domain status (LPCG28_STATUS1)	32	R	0000_FFFFh
8730h	LPCG access control (LPCG28_AUTHEN)	32	RW	FFFF_0000h
8740h	LPCG direct control (LPCG29_DIRECT)	32	RW	0000_0001h
8750h	Clock source low power mode setting (LPCG29_LPM0)	32	RW	0000_0000h
8754h	clock source low power mode setting (LPCG29_LPM1)	32	RW	0000_0000h
875Ch	LPM setting of current CPU domain (LPCG29_LPM_CUR)	32	RW	0000_0000h
8760h	LPCG working status (LPCG29_STATUS0)	32	R	0000_0001h
8764h	LPCG domain status (LPCG29_STATUS1)	32	R	0000_FFFFh
8770h	LPCG access control (LPCG29_AUTHEN)	32	RW	FFFF_0000h
8780h	LPCG direct control (LPCG30_DIRECT)	32	RW	0000_0001h
8790h	Clock source low power mode setting (LPCG30_LPM0)	32	RW	0000_0000h
8794h	clock source low power mode setting (LPCG30_LPM1)	32	RW	0000_0000h
879Ch	LPM setting of current CPU domain (LPCG30_LPM_CUR)	32	RW	0000_0000h
87A0h	LPCG working status (LPCG30_STATUS0)	32	R	0000_0001h
87A4h	LPCG domain status (LPCG30_STATUS1)	32	R	0000_FFFFh
87B0h	LPCG access control (LPCG30_AUTHEN)	32	RW	FFFF_0000h
87C0h	LPCG direct control (LPCG31_DIRECT)	32	RW	0000_0001h
87D0h	Clock source low power mode setting (LPCG31_LPM0)	32	RW	0000_0000h
87D4h	clock source low power mode setting (LPCG31_LPM1)	32	RW	0000_0000h
87DCh	LPM setting of current CPU domain (LPCG31_LPM_CUR)	32	RW	0000_0000h
87E0h	LPCG working status (LPCG31_STATUS0)	32	R	0000_0001h
87E4h	LPCG domain status (LPCG31_STATUS1)	32	R	0000_FFFFh
87F0h	LPCG access control (LPCG31_AUTHEN)	32	RW	FFFF_0000h
8800h	LPCG direct control (LPCG32_DIRECT)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8810h	Clock source low power mode setting (LPCG32_LPM0)	32	RW	0000_0000h
8814h	clock source low power mode setting (LPCG32_LPM1)	32	RW	0000_0000h
881Ch	LPM setting of current CPU domain (LPCG32_LPM_CUR)	32	RW	0000_0000h
8820h	LPCG working status (LPCG32_STATUS0)	32	R	0000_0001h
8824h	LPCG domain status (LPCG32_STATUS1)	32	R	0000_FFFFh
8830h	LPCG access control (LPCG32_AUTHEN)	32	RW	FFFF_0000h
8840h	LPCG direct control (LPCG33_DIRECT)	32	RW	0000_0001h
8850h	Clock source low power mode setting (LPCG33_LPM0)	32	RW	0000_0000h
8854h	clock source low power mode setting (LPCG33_LPM1)	32	RW	0000_0000h
885Ch	LPM setting of current CPU domain (LPCG33_LPM_CUR)	32	RW	0000_0000h
8860h	LPCG working status (LPCG33_STATUS0)	32	R	0000_0001h
8864h	LPCG domain status (LPCG33_STATUS1)	32	R	0000_FFFFh
8870h	LPCG access control (LPCG33_AUTHEN)	32	RW	FFFF_0000h
8880h	LPCG direct control (LPCG34_DIRECT)	32	RW	0000_0001h
8890h	Clock source low power mode setting (LPCG34_LPM0)	32	RW	0000_0000h
8894h	clock source low power mode setting (LPCG34_LPM1)	32	RW	0000_0000h
889Ch	LPM setting of current CPU domain (LPCG34_LPM_CUR)	32	RW	0000_0000h
88A0h	LPCG working status (LPCG34_STATUS0)	32	R	0000_0001h
88A4h	LPCG domain status (LPCG34_STATUS1)	32	R	0000_FFFFh
88B0h	LPCG access control (LPCG34_AUTHEN)	32	RW	FFFF_0000h
88C0h	LPCG direct control (LPCG35_DIRECT)	32	RW	0000_0001h
88D0h	Clock source low power mode setting (LPCG35_LPM0)	32	RW	0000_0000h
88D4h	clock source low power mode setting (LPCG35_LPM1)	32	RW	0000_0000h
88DCh	LPM setting of current CPU domain (LPCG35_LPM_CUR)	32	RW	0000_0000h
88E0h	LPCG working status (LPCG35_STATUS0)	32	R	0000_0001h
88E4h	LPCG domain status (LPCG35_STATUS1)	32	R	0000_FFFFh
88F0h	LPCG access control (LPCG35_AUTHEN)	32	RW	FFFF_0000h
8900h	LPCG direct control (LPCG36_DIRECT)	32	RW	0000_0001h
8910h	Clock source low power mode setting (LPCG36_LPM0)	32	RW	0000_0000h
8914h	clock source low power mode setting (LPCG36_LPM1)	32	RW	0000_0000h
891Ch	LPM setting of current CPU domain (LPCG36_LPM_CUR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8920h	LPCG working status (LPCG36_STATUS0)	32	R	0000_0001h
8924h	LPCG domain status (LPCG36_STATUS1)	32	R	0000_FFFFh
8930h	LPCG access control (LPCG36_AUTHEN)	32	RW	FFFF_0000h
8940h	LPCG direct control (LPCG37_DIRECT)	32	RW	0000_0001h
8950h	Clock source low power mode setting (LPCG37_LPM0)	32	RW	0000_0000h
8954h	clock source low power mode setting (LPCG37_LPM1)	32	RW	0000_0000h
895Ch	LPM setting of current CPU domain (LPCG37_LPM_CUR)	32	RW	0000_0000h
8960h	LPCG working status (LPCG37_STATUS0)	32	R	0000_0001h
8964h	LPCG domain status (LPCG37_STATUS1)	32	R	0000_FFFFh
8970h	LPCG access control (LPCG37_AUTHEN)	32	RW	FFFF_0000h
8980h	LPCG direct control (LPCG38_DIRECT)	32	RW	0000_0001h
8990h	Clock source low power mode setting (LPCG38_LPM0)	32	RW	0000_0000h
8994h	clock source low power mode setting (LPCG38_LPM1)	32	RW	0000_0000h
899Ch	LPM setting of current CPU domain (LPCG38_LPM_CUR)	32	RW	0000_0000h
89A0h	LPCG working status (LPCG38_STATUS0)	32	R	0000_0001h
89A4h	LPCG domain status (LPCG38_STATUS1)	32	R	0000_FFFFh
89B0h	LPCG access control (LPCG38_AUTHEN)	32	RW	FFFF_0000h
89C0h	LPCG direct control (LPCG39_DIRECT)	32	RW	0000_0001h
89D0h	Clock source low power mode setting (LPCG39_LPM0)	32	RW	0000_0000h
89D4h	clock source low power mode setting (LPCG39_LPM1)	32	RW	0000_0000h
89DCh	LPM setting of current CPU domain (LPCG39_LPM_CUR)	32	RW	0000_0000h
89E0h	LPCG working status (LPCG39_STATUS0)	32	R	0000_0001h
89E4h	LPCG domain status (LPCG39_STATUS1)	32	R	0000_FFFFh
89F0h	LPCG access control (LPCG39_AUTHEN)	32	RW	FFFF_0000h
8A00h	LPCG direct control (LPCG40_DIRECT)	32	RW	0000_0001h
8A10h	Clock source low power mode setting (LPCG40_LPM0)	32	RW	0000_0000h
8A14h	clock source low power mode setting (LPCG40_LPM1)	32	RW	0000_0000h
8A1Ch	LPM setting of current CPU domain (LPCG40_LPM_CUR)	32	RW	0000_0000h
8A20h	LPCG working status (LPCG40_STATUS0)	32	R	0000_0001h
8A24h	LPCG domain status (LPCG40_STATUS1)	32	R	0000_FFFFh
8A30h	LPCG access control (LPCG40_AUTHEN)	32	RW	FFFF_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8A40h	LPCG direct control (LPCG41_DIRECT)	32	RW	0000_0001h
8A50h	Clock source low power mode setting (LPCG41_LPM0)	32	RW	0000_0000h
8A54h	clock source low power mode setting (LPCG41_LPM1)	32	RW	0000_0000h
8A5Ch	LPM setting of current CPU domain (LPCG41_LPM_CUR)	32	RW	0000_0000h
8A60h	LPCG working status (LPCG41_STATUS0)	32	R	0000_0001h
8A64h	LPCG domain status (LPCG41_STATUS1)	32	R	0000_FFFFh
8A70h	LPCG access control (LPCG41_AUTHEN)	32	RW	FFFF_0000h
8A80h	LPCG direct control (LPCG42_DIRECT)	32	RW	0000_0001h
8A90h	Clock source low power mode setting (LPCG42_LPM0)	32	RW	0000_0000h
8A94h	clock source low power mode setting (LPCG42_LPM1)	32	RW	0000_0000h
8A9Ch	LPM setting of current CPU domain (LPCG42_LPM_CUR)	32	RW	0000_0000h
8AA0h	LPCG working status (LPCG42_STATUS0)	32	R	0000_0001h
8AA4h	LPCG domain status (LPCG42_STATUS1)	32	R	0000_FFFFh
8AB0h	LPCG access control (LPCG42_AUTHEN)	32	RW	FFFF_0000h
8AC0h	LPCG direct control (LPCG43_DIRECT)	32	RW	0000_0001h
8AD0h	Clock source low power mode setting (LPCG43_LPM0)	32	RW	0000_0000h
8AD4h	clock source low power mode setting (LPCG43_LPM1)	32	RW	0000_0000h
8ADCh	LPM setting of current CPU domain (LPCG43_LPM_CUR)	32	RW	0000_0000h
8AE0h	LPCG working status (LPCG43_STATUS0)	32	R	0000_0001h
8AE4h	LPCG domain status (LPCG43_STATUS1)	32	R	0000_FFFFh
8AF0h	LPCG access control (LPCG43_AUTHEN)	32	RW	FFFF_0000h
8B00h	LPCG direct control (LPCG44_DIRECT)	32	RW	0000_0001h
8B10h	Clock source low power mode setting (LPCG44_LPM0)	32	RW	0000_0000h
8B14h	clock source low power mode setting (LPCG44_LPM1)	32	RW	0000_0000h
8B1Ch	LPM setting of current CPU domain (LPCG44_LPM_CUR)	32	RW	0000_0000h
8B20h	LPCG working status (LPCG44_STATUS0)	32	R	0000_0001h
8B24h	LPCG domain status (LPCG44_STATUS1)	32	R	0000_FFFFh
8B30h	LPCG access control (LPCG44_AUTHEN)	32	RW	FFFF_0000h
8B40h	LPCG direct control (LPCG45_DIRECT)	32	RW	0000_0001h
8B50h	Clock source low power mode setting (LPCG45_LPM0)	32	RW	0000_0000h
8B54h	clock source low power mode setting (LPCG45_LPM1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8B5Ch	LPM setting of current CPU domain (LPCG45_LPM_CUR)	32	RW	0000_0000h
8B60h	LPCG working status (LPCG45_STATUS0)	32	R	0000_0001h
8B64h	LPCG domain status (LPCG45_STATUS1)	32	R	0000_FFFFh
8B70h	LPCG access control (LPCG45_AUTHEN)	32	RW	FFFF_0000h
8B80h	LPCG direct control (LPCG46_DIRECT)	32	RW	0000_0001h
8B90h	Clock source low power mode setting (LPCG46_LPM0)	32	RW	0000_0000h
8B94h	clock source low power mode setting (LPCG46_LPM1)	32	RW	0000_0000h
8B9Ch	LPM setting of current CPU domain (LPCG46_LPM_CUR)	32	RW	0000_0000h
8BA0h	LPCG working status (LPCG46_STATUS0)	32	R	0000_0001h
8BA4h	LPCG domain status (LPCG46_STATUS1)	32	R	0000_FFFFh
8BB0h	LPCG access control (LPCG46_AUTHEN)	32	RW	FFFF_0000h
8BC0h	LPCG direct control (LPCG47_DIRECT)	32	RW	0000_0001h
8BD0h	Clock source low power mode setting (LPCG47_LPM0)	32	RW	0000_0000h
8BD4h	clock source low power mode setting (LPCG47_LPM1)	32	RW	0000_0000h
8BDCh	LPM setting of current CPU domain (LPCG47_LPM_CUR)	32	RW	0000_0000h
8BE0h	LPCG working status (LPCG47_STATUS0)	32	R	0000_0001h
8BE4h	LPCG domain status (LPCG47_STATUS1)	32	R	0000_FFFFh
8BF0h	LPCG access control (LPCG47_AUTHEN)	32	RW	FFFF_0000h
8C00h	LPCG direct control (LPCG48_DIRECT)	32	RW	0000_0001h
8C10h	Clock source low power mode setting (LPCG48_LPM0)	32	RW	0000_0000h
8C14h	clock source low power mode setting (LPCG48_LPM1)	32	RW	0000_0000h
8C1Ch	LPM setting of current CPU domain (LPCG48_LPM_CUR)	32	RW	0000_0000h
8C20h	LPCG working status (LPCG48_STATUS0)	32	R	0000_0001h
8C24h	LPCG domain status (LPCG48_STATUS1)	32	R	0000_FFFFh
8C30h	LPCG access control (LPCG48_AUTHEN)	32	RW	FFFF_0000h
8C40h	LPCG direct control (LPCG49_DIRECT)	32	RW	0000_0001h
8C50h	Clock source low power mode setting (LPCG49_LPM0)	32	RW	0000_0000h
8C54h	clock source low power mode setting (LPCG49_LPM1)	32	RW	0000_0000h
8C5Ch	LPM setting of current CPU domain (LPCG49_LPM_CUR)	32	RW	0000_0000h
8C60h	LPCG working status (LPCG49_STATUS0)	32	R	0000_0001h
8C64h	LPCG domain status (LPCG49_STATUS1)	32	R	0000_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8C70h	LPCG access control (LPCG49_AUTHEN)	32	RW	FFFF_0000h
8C80h	LPCG direct control (LPCG50_DIRECT)	32	RW	0000_0001h
8C90h	Clock source low power mode setting (LPCG50_LPM0)	32	RW	0000_0000h
8C94h	clock source low power mode setting (LPCG50_LPM1)	32	RW	0000_0000h
8C9Ch	LPM setting of current CPU domain (LPCG50_LPM_CUR)	32	RW	0000_0000h
8CA0h	LPCG working status (LPCG50_STATUS0)	32	R	0000_0001h
8CA4h	LPCG domain status (LPCG50_STATUS1)	32	R	0000_FFFFh
8CB0h	LPCG access control (LPCG50_AUTHEN)	32	RW	FFFF_0000h
8CC0h	LPCG direct control (LPCG51_DIRECT)	32	RW	0000_0001h
8CD0h	Clock source low power mode setting (LPCG51_LPM0)	32	RW	0000_0000h
8CD4h	clock source low power mode setting (LPCG51_LPM1)	32	RW	0000_0000h
8CDCh	LPM setting of current CPU domain (LPCG51_LPM_CUR)	32	RW	0000_0000h
8CE0h	LPCG working status (LPCG51_STATUS0)	32	R	0000_0001h
8CE4h	LPCG domain status (LPCG51_STATUS1)	32	R	0000_FFFFh
8CF0h	LPCG access control (LPCG51_AUTHEN)	32	RW	FFFF_0000h
8D00h	LPCG direct control (LPCG52_DIRECT)	32	RW	0000_0001h
8D10h	Clock source low power mode setting (LPCG52_LPM0)	32	RW	0000_0000h
8D14h	clock source low power mode setting (LPCG52_LPM1)	32	RW	0000_0000h
8D1Ch	LPM setting of current CPU domain (LPCG52_LPM_CUR)	32	RW	0000_0000h
8D20h	LPCG working status (LPCG52_STATUS0)	32	R	0000_0001h
8D24h	LPCG domain status (LPCG52_STATUS1)	32	R	0000_FFFFh
8D30h	LPCG access control (LPCG52_AUTHEN)	32	RW	FFFF_0000h
8D40h	LPCG direct control (LPCG53_DIRECT)	32	RW	0000_0001h
8D50h	Clock source low power mode setting (LPCG53_LPM0)	32	RW	0000_0000h
8D54h	clock source low power mode setting (LPCG53_LPM1)	32	RW	0000_0000h
8D5Ch	LPM setting of current CPU domain (LPCG53_LPM_CUR)	32	RW	0000_0000h
8D60h	LPCG working status (LPCG53_STATUS0)	32	R	0000_0001h
8D64h	LPCG domain status (LPCG53_STATUS1)	32	R	0000_FFFFh
8D70h	LPCG access control (LPCG53_AUTHEN)	32	RW	FFFF_0000h
8D80h	LPCG direct control (LPCG54_DIRECT)	32	RW	0000_0001h
8D90h	Clock source low power mode setting (LPCG54_LPM0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8D94h	clock source low power mode setting (LPCG54_LPM1)	32	RW	0000_0000h
8D9Ch	LPM setting of current CPU domain (LPCG54_LPM_CUR)	32	RW	0000_0000h
8DA0h	LPCG working status (LPCG54_STATUS0)	32	R	0000_0001h
8DA4h	LPCG domain status (LPCG54_STATUS1)	32	R	0000_FFFFh
8DB0h	LPCG access control (LPCG54_AUTHEN)	32	RW	FFFF_0000h
8DC0h	LPCG direct control (LPCG55_DIRECT)	32	RW	0000_0001h
8DD0h	Clock source low power mode setting (LPCG55_LPM0)	32	RW	0000_0000h
8DD4h	clock source low power mode setting (LPCG55_LPM1)	32	RW	0000_0000h
8DDCh	LPM setting of current CPU domain (LPCG55_LPM_CUR)	32	RW	0000_0000h
8DE0h	LPCG working status (LPCG55_STATUS0)	32	R	0000_0001h
8DE4h	LPCG domain status (LPCG55_STATUS1)	32	R	0000_FFFFh
8DF0h	LPCG access control (LPCG55_AUTHEN)	32	RW	FFFF_0000h
8E00h	LPCG direct control (LPCG56_DIRECT)	32	RW	0000_0001h
8E10h	Clock source low power mode setting (LPCG56_LPM0)	32	RW	0000_0000h
8E14h	clock source low power mode setting (LPCG56_LPM1)	32	RW	0000_0000h
8E1Ch	LPM setting of current CPU domain (LPCG56_LPM_CUR)	32	RW	0000_0000h
8E20h	LPCG working status (LPCG56_STATUS0)	32	R	0000_0001h
8E24h	LPCG domain status (LPCG56_STATUS1)	32	R	0000_FFFFh
8E30h	LPCG access control (LPCG56_AUTHEN)	32	RW	FFFF_0000h
8E40h	LPCG direct control (LPCG57_DIRECT)	32	RW	0000_0001h
8E50h	Clock source low power mode setting (LPCG57_LPM0)	32	RW	0000_0000h
8E54h	clock source low power mode setting (LPCG57_LPM1)	32	RW	0000_0000h
8E5Ch	LPM setting of current CPU domain (LPCG57_LPM_CUR)	32	RW	0000_0000h
8E60h	LPCG working status (LPCG57_STATUS0)	32	R	0000_0001h
8E64h	LPCG domain status (LPCG57_STATUS1)	32	R	0000_FFFFh
8E70h	LPCG access control (LPCG57_AUTHEN)	32	RW	FFFF_0000h
8E80h	LPCG direct control (LPCG58_DIRECT)	32	RW	0000_0001h
8E90h	Clock source low power mode setting (LPCG58_LPM0)	32	RW	0000_0000h
8E94h	clock source low power mode setting (LPCG58_LPM1)	32	RW	0000_0000h
8E9Ch	LPM setting of current CPU domain (LPCG58_LPM_CUR)	32	RW	0000_0000h
8EA0h	LPCG working status (LPCG58_STATUS0)	32	R	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8EA4h	LPCG domain status (LPCG58_STATUS1)	32	R	0000_FFFFh
8EB0h	LPCG access control (LPCG58_AUTHEN)	32	RW	FFFF_0000h
8EC0h	LPCG direct control (LPCG59_DIRECT)	32	RW	0000_0001h
8ED0h	Clock source low power mode setting (LPCG59_LPM0)	32	RW	0000_0000h
8ED4h	clock source low power mode setting (LPCG59_LPM1)	32	RW	0000_0000h
8EDCh	LPM setting of current CPU domain (LPCG59_LPM_CUR)	32	RW	0000_0000h
8EE0h	LPCG working status (LPCG59_STATUS0)	32	R	0000_0001h
8EE4h	LPCG domain status (LPCG59_STATUS1)	32	R	0000_FFFFh
8EF0h	LPCG access control (LPCG59_AUTHEN)	32	RW	FFFF_0000h
8F00h	LPCG direct control (LPCG60_DIRECT)	32	RW	0000_0001h
8F10h	Clock source low power mode setting (LPCG60_LPM0)	32	RW	0000_0000h
8F14h	clock source low power mode setting (LPCG60_LPM1)	32	RW	0000_0000h
8F1Ch	LPM setting of current CPU domain (LPCG60_LPM_CUR)	32	RW	0000_0000h
8F20h	LPCG working status (LPCG60_STATUS0)	32	R	0000_0001h
8F24h	LPCG domain status (LPCG60_STATUS1)	32	R	0000_FFFFh
8F30h	LPCG access control (LPCG60_AUTHEN)	32	RW	FFFF_0000h
8F40h	LPCG direct control (LPCG61_DIRECT)	32	RW	0000_0001h
8F50h	Clock source low power mode setting (LPCG61_LPM0)	32	RW	0000_0000h
8F54h	clock source low power mode setting (LPCG61_LPM1)	32	RW	0000_0000h
8F5Ch	LPM setting of current CPU domain (LPCG61_LPM_CUR)	32	RW	0000_0000h
8F60h	LPCG working status (LPCG61_STATUS0)	32	R	0000_0001h
8F64h	LPCG domain status (LPCG61_STATUS1)	32	R	0000_FFFFh
8F70h	LPCG access control (LPCG61_AUTHEN)	32	RW	FFFF_0000h
8F80h	LPCG direct control (LPCG62_DIRECT)	32	RW	0000_0001h
8F90h	Clock source low power mode setting (LPCG62_LPM0)	32	RW	0000_0000h
8F94h	clock source low power mode setting (LPCG62_LPM1)	32	RW	0000_0000h
8F9Ch	LPM setting of current CPU domain (LPCG62_LPM_CUR)	32	RW	0000_0000h
8FA0h	LPCG working status (LPCG62_STATUS0)	32	R	0000_0001h
8FA4h	LPCG domain status (LPCG62_STATUS1)	32	R	0000_FFFFh
8FB0h	LPCG access control (LPCG62_AUTHEN)	32	RW	FFFF_0000h
8FC0h	LPCG direct control (LPCG63_DIRECT)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8FD0h	Clock source low power mode setting (LPCG63_LPM0)	32	RW	0000_0000h
8FD4h	clock source low power mode setting (LPCG63_LPM1)	32	RW	0000_0000h
8FDCh	LPM setting of current CPU domain (LPCG63_LPM_CUR)	32	RW	0000_0000h
8FE0h	LPCG working status (LPCG63_STATUS0)	32	R	0000_0001h
8FE4h	LPCG domain status (LPCG63_STATUS1)	32	R	0000_FFFFh
8FF0h	LPCG access control (LPCG63_AUTHEN)	32	RW	FFFF_0000h
9000h	LPCG direct control (LPCG64_DIRECT)	32	RW	0000_0001h
9010h	Clock source low power mode setting (LPCG64_LPM0)	32	RW	0000_0000h
9014h	clock source low power mode setting (LPCG64_LPM1)	32	RW	0000_0000h
901Ch	LPM setting of current CPU domain (LPCG64_LPM_CUR)	32	RW	0000_0000h
9020h	LPCG working status (LPCG64_STATUS0)	32	R	0000_0001h
9024h	LPCG domain status (LPCG64_STATUS1)	32	R	0000_FFFFh
9030h	LPCG access control (LPCG64_AUTHEN)	32	RW	FFFF_0000h
9040h	LPCG direct control (LPCG65_DIRECT)	32	RW	0000_0001h
9050h	Clock source low power mode setting (LPCG65_LPM0)	32	RW	0000_0000h
9054h	clock source low power mode setting (LPCG65_LPM1)	32	RW	0000_0000h
905Ch	LPM setting of current CPU domain (LPCG65_LPM_CUR)	32	RW	0000_0000h
9060h	LPCG working status (LPCG65_STATUS0)	32	R	0000_0001h
9064h	LPCG domain status (LPCG65_STATUS1)	32	R	0000_FFFFh
9070h	LPCG access control (LPCG65_AUTHEN)	32	RW	FFFF_0000h
9080h	LPCG direct control (LPCG66_DIRECT)	32	RW	0000_0001h
9090h	Clock source low power mode setting (LPCG66_LPM0)	32	RW	0000_0000h
9094h	clock source low power mode setting (LPCG66_LPM1)	32	RW	0000_0000h
909Ch	LPM setting of current CPU domain (LPCG66_LPM_CUR)	32	RW	0000_0000h
90A0h	LPCG working status (LPCG66_STATUS0)	32	R	0000_0001h
90A4h	LPCG domain status (LPCG66_STATUS1)	32	R	0000_FFFFh
90B0h	LPCG access control (LPCG66_AUTHEN)	32	RW	FFFF_0000h
90C0h	LPCG direct control (LPCG67_DIRECT)	32	RW	0000_0001h
90D0h	Clock source low power mode setting (LPCG67_LPM0)	32	RW	0000_0000h
90D4h	clock source low power mode setting (LPCG67_LPM1)	32	RW	0000_0000h
90DCh	LPM setting of current CPU domain (LPCG67_LPM_CUR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
90E0h	LPCG working status (LPCG67_STATUS0)	32	R	0000_0001h
90E4h	LPCG domain status (LPCG67_STATUS1)	32	R	0000_FFFFh
90F0h	LPCG access control (LPCG67_AUTHEN)	32	RW	FFFF_0000h
9100h	LPCG direct control (LPCG68_DIRECT)	32	RW	0000_0001h
9110h	Clock source low power mode setting (LPCG68_LPM0)	32	RW	0000_0000h
9114h	clock source low power mode setting (LPCG68_LPM1)	32	RW	0000_0000h
911Ch	LPM setting of current CPU domain (LPCG68_LPM_CUR)	32	RW	0000_0000h
9120h	LPCG working status (LPCG68_STATUS0)	32	R	0000_0001h
9124h	LPCG domain status (LPCG68_STATUS1)	32	R	0000_FFFFh
9130h	LPCG access control (LPCG68_AUTHEN)	32	RW	FFFF_0000h
9140h	LPCG direct control (LPCG69_DIRECT)	32	RW	0000_0001h
9150h	Clock source low power mode setting (LPCG69_LPM0)	32	RW	0000_0000h
9154h	clock source low power mode setting (LPCG69_LPM1)	32	RW	0000_0000h
915Ch	LPM setting of current CPU domain (LPCG69_LPM_CUR)	32	RW	0000_0000h
9160h	LPCG working status (LPCG69_STATUS0)	32	R	0000_0001h
9164h	LPCG domain status (LPCG69_STATUS1)	32	R	0000_FFFFh
9170h	LPCG access control (LPCG69_AUTHEN)	32	RW	FFFF_0000h
9180h	LPCG direct control (LPCG70_DIRECT)	32	RW	0000_0001h
9190h	Clock source low power mode setting (LPCG70_LPM0)	32	RW	0000_0000h
9194h	clock source low power mode setting (LPCG70_LPM1)	32	RW	0000_0000h
919Ch	LPM setting of current CPU domain (LPCG70_LPM_CUR)	32	RW	0000_0000h
91A0h	LPCG working status (LPCG70_STATUS0)	32	R	0000_0001h
91A4h	LPCG domain status (LPCG70_STATUS1)	32	R	0000_FFFFh
91B0h	LPCG access control (LPCG70_AUTHEN)	32	RW	FFFF_0000h
91C0h	LPCG direct control (LPCG71_DIRECT)	32	RW	0000_0001h
91D0h	Clock source low power mode setting (LPCG71_LPM0)	32	RW	0000_0000h
91D4h	clock source low power mode setting (LPCG71_LPM1)	32	RW	0000_0000h
91DCh	LPM setting of current CPU domain (LPCG71_LPM_CUR)	32	RW	0000_0000h
91E0h	LPCG working status (LPCG71_STATUS0)	32	R	0000_0001h
91E4h	LPCG domain status (LPCG71_STATUS1)	32	R	0000_FFFFh
91F0h	LPCG access control (LPCG71_AUTHEN)	32	RW	FFFF_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9200h	LPCG direct control (LPCG72_DIRECT)	32	RW	0000_0001h
9210h	Clock source low power mode setting (LPCG72_LPM0)	32	RW	0000_0000h
9214h	clock source low power mode setting (LPCG72_LPM1)	32	RW	0000_0000h
921Ch	LPM setting of current CPU domain (LPCG72_LPM_CUR)	32	RW	0000_0000h
9220h	LPCG working status (LPCG72_STATUS0)	32	R	0000_0001h
9224h	LPCG domain status (LPCG72_STATUS1)	32	R	0000_FFFFh
9230h	LPCG access control (LPCG72_AUTHEN)	32	RW	FFFF_0000h
9240h	LPCG direct control (LPCG73_DIRECT)	32	RW	0000_0001h
9250h	Clock source low power mode setting (LPCG73_LPM0)	32	RW	0000_0000h
9254h	clock source low power mode setting (LPCG73_LPM1)	32	RW	0000_0000h
925Ch	LPM setting of current CPU domain (LPCG73_LPM_CUR)	32	RW	0000_0000h
9260h	LPCG working status (LPCG73_STATUS0)	32	R	0000_0001h
9264h	LPCG domain status (LPCG73_STATUS1)	32	R	0000_FFFFh
9270h	LPCG access control (LPCG73_AUTHEN)	32	RW	FFFF_0000h
9280h	LPCG direct control (LPCG74_DIRECT)	32	RW	0000_0001h
9290h	Clock source low power mode setting (LPCG74_LPM0)	32	RW	0000_0000h
9294h	clock source low power mode setting (LPCG74_LPM1)	32	RW	0000_0000h
929Ch	LPM setting of current CPU domain (LPCG74_LPM_CUR)	32	RW	0000_0000h
92A0h	LPCG working status (LPCG74_STATUS0)	32	R	0000_0001h
92A4h	LPCG domain status (LPCG74_STATUS1)	32	R	0000_FFFFh
92B0h	LPCG access control (LPCG74_AUTHEN)	32	RW	FFFF_0000h
92C0h	LPCG direct control (LPCG75_DIRECT)	32	RW	0000_0001h
92D0h	Clock source low power mode setting (LPCG75_LPM0)	32	RW	0000_0000h
92D4h	clock source low power mode setting (LPCG75_LPM1)	32	RW	0000_0000h
92DCh	LPM setting of current CPU domain (LPCG75_LPM_CUR)	32	RW	0000_0000h
92E0h	LPCG working status (LPCG75_STATUS0)	32	R	0000_0001h
92E4h	LPCG domain status (LPCG75_STATUS1)	32	R	0000_FFFFh
92F0h	LPCG access control (LPCG75_AUTHEN)	32	RW	FFFF_0000h
9300h	LPCG direct control (LPCG76_DIRECT)	32	RW	0000_0001h
9310h	Clock source low power mode setting (LPCG76_LPM0)	32	RW	0000_0000h
9314h	clock source low power mode setting (LPCG76_LPM1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
931Ch	LPM setting of current CPU domain (LPCG76_LPM_CUR)	32	RW	0000_0000h
9320h	LPCG working status (LPCG76_STATUS0)	32	R	0000_0001h
9324h	LPCG domain status (LPCG76_STATUS1)	32	R	0000_FFFFh
9330h	LPCG access control (LPCG76_AUTHEN)	32	RW	FFFF_0000h
9340h	LPCG direct control (LPCG77_DIRECT)	32	RW	0000_0001h
9350h	Clock source low power mode setting (LPCG77_LPM0)	32	RW	0000_0000h
9354h	clock source low power mode setting (LPCG77_LPM1)	32	RW	0000_0000h
935Ch	LPM setting of current CPU domain (LPCG77_LPM_CUR)	32	RW	0000_0000h
9360h	LPCG working status (LPCG77_STATUS0)	32	R	0000_0001h
9364h	LPCG domain status (LPCG77_STATUS1)	32	R	0000_FFFFh
9370h	LPCG access control (LPCG77_AUTHEN)	32	RW	FFFF_0000h
9380h	LPCG direct control (LPCG78_DIRECT)	32	RW	0000_0001h
9390h	Clock source low power mode setting (LPCG78_LPM0)	32	RW	0000_0000h
9394h	clock source low power mode setting (LPCG78_LPM1)	32	RW	0000_0000h
939Ch	LPM setting of current CPU domain (LPCG78_LPM_CUR)	32	RW	0000_0000h
93A0h	LPCG working status (LPCG78_STATUS0)	32	R	0000_0001h
93A4h	LPCG domain status (LPCG78_STATUS1)	32	R	0000_FFFFh
93B0h	LPCG access control (LPCG78_AUTHEN)	32	RW	FFFF_0000h
93C0h	LPCG direct control (LPCG79_DIRECT)	32	RW	0000_0001h
93D0h	Clock source low power mode setting (LPCG79_LPM0)	32	RW	0000_0000h
93D4h	clock source low power mode setting (LPCG79_LPM1)	32	RW	0000_0000h
93DCh	LPM setting of current CPU domain (LPCG79_LPM_CUR)	32	RW	0000_0000h
93E0h	LPCG working status (LPCG79_STATUS0)	32	R	0000_0001h
93E4h	LPCG domain status (LPCG79_STATUS1)	32	R	0000_FFFFh
93F0h	LPCG access control (LPCG79_AUTHEN)	32	RW	FFFF_0000h
9400h	LPCG direct control (LPCG80_DIRECT)	32	RW	0000_0001h
9410h	Clock source low power mode setting (LPCG80_LPM0)	32	RW	0000_0000h
9414h	clock source low power mode setting (LPCG80_LPM1)	32	RW	0000_0000h
941Ch	LPM setting of current CPU domain (LPCG80_LPM_CUR)	32	RW	0000_0000h
9420h	LPCG working status (LPCG80_STATUS0)	32	R	0000_0001h
9424h	LPCG domain status (LPCG80_STATUS1)	32	R	0000_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9430h	LPCG access control (LPCG80_AUTHEN)	32	RW	FFFF_0000h
9440h	LPCG direct control (LPCG81_DIRECT)	32	RW	0000_0001h
9450h	Clock source low power mode setting (LPCG81_LPM0)	32	RW	0000_0000h
9454h	clock source low power mode setting (LPCG81_LPM1)	32	RW	0000_0000h
945Ch	LPM setting of current CPU domain (LPCG81_LPM_CUR)	32	RW	0000_0000h
9460h	LPCG working status (LPCG81_STATUS0)	32	R	0000_0001h
9464h	LPCG domain status (LPCG81_STATUS1)	32	R	0000_FFFFh
9470h	LPCG access control (LPCG81_AUTHEN)	32	RW	FFFF_0000h
9480h	LPCG direct control (LPCG82_DIRECT)	32	RW	0000_0001h
9490h	Clock source low power mode setting (LPCG82_LPM0)	32	RW	0000_0000h
9494h	clock source low power mode setting (LPCG82_LPM1)	32	RW	0000_0000h
949Ch	LPM setting of current CPU domain (LPCG82_LPM_CUR)	32	RW	0000_0000h
94A0h	LPCG working status (LPCG82_STATUS0)	32	R	0000_0001h
94A4h	LPCG domain status (LPCG82_STATUS1)	32	R	0000_FFFFh
94B0h	LPCG access control (LPCG82_AUTHEN)	32	RW	FFFF_0000h
94C0h	LPCG direct control (LPCG83_DIRECT)	32	RW	0000_0001h
94D0h	Clock source low power mode setting (LPCG83_LPM0)	32	RW	0000_0000h
94D4h	clock source low power mode setting (LPCG83_LPM1)	32	RW	0000_0000h
94DCh	LPM setting of current CPU domain (LPCG83_LPM_CUR)	32	RW	0000_0000h
94E0h	LPCG working status (LPCG83_STATUS0)	32	R	0000_0001h
94E4h	LPCG domain status (LPCG83_STATUS1)	32	R	0000_FFFFh
94F0h	LPCG access control (LPCG83_AUTHEN)	32	RW	FFFF_0000h
9500h	LPCG direct control (LPCG84_DIRECT)	32	RW	0000_0001h
9510h	Clock source low power mode setting (LPCG84_LPM0)	32	RW	0000_0000h
9514h	clock source low power mode setting (LPCG84_LPM1)	32	RW	0000_0000h
951Ch	LPM setting of current CPU domain (LPCG84_LPM_CUR)	32	RW	0000_0000h
9520h	LPCG working status (LPCG84_STATUS0)	32	R	0000_0001h
9524h	LPCG domain status (LPCG84_STATUS1)	32	R	0000_FFFFh
9530h	LPCG access control (LPCG84_AUTHEN)	32	RW	FFFF_0000h
9540h	LPCG direct control (LPCG85_DIRECT)	32	RW	0000_0001h
9550h	Clock source low power mode setting (LPCG85_LPM0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9554h	clock source low power mode setting (LPCG85_LPM1)	32	RW	0000_0000h
955Ch	LPM setting of current CPU domain (LPCG85_LPM_CUR)	32	RW	0000_0000h
9560h	LPCG working status (LPCG85_STATUS0)	32	R	0000_0001h
9564h	LPCG domain status (LPCG85_STATUS1)	32	R	0000_FFFFh
9570h	LPCG access control (LPCG85_AUTHEN)	32	RW	FFFF_0000h
9580h	LPCG direct control (LPCG86_DIRECT)	32	RW	0000_0001h
9590h	Clock source low power mode setting (LPCG86_LPM0)	32	RW	0000_0000h
9594h	clock source low power mode setting (LPCG86_LPM1)	32	RW	0000_0000h
959Ch	LPM setting of current CPU domain (LPCG86_LPM_CUR)	32	RW	0000_0000h
95A0h	LPCG working status (LPCG86_STATUS0)	32	R	0000_0001h
95A4h	LPCG domain status (LPCG86_STATUS1)	32	R	0000_FFFFh
95B0h	LPCG access control (LPCG86_AUTHEN)	32	RW	FFFF_0000h
95C0h	LPCG direct control (LPCG87_DIRECT)	32	RW	0000_0001h
95D0h	Clock source low power mode setting (LPCG87_LPM0)	32	RW	0000_0000h
95D4h	clock source low power mode setting (LPCG87_LPM1)	32	RW	0000_0000h
95DCh	LPM setting of current CPU domain (LPCG87_LPM_CUR)	32	RW	0000_0000h
95E0h	LPCG working status (LPCG87_STATUS0)	32	R	0000_0001h
95E4h	LPCG domain status (LPCG87_STATUS1)	32	R	0000_FFFFh
95F0h	LPCG access control (LPCG87_AUTHEN)	32	RW	FFFF_0000h
9600h	LPCG direct control (LPCG88_DIRECT)	32	RW	0000_0001h
9610h	Clock source low power mode setting (LPCG88_LPM0)	32	RW	0000_0000h
9614h	clock source low power mode setting (LPCG88_LPM1)	32	RW	0000_0000h
961Ch	LPM setting of current CPU domain (LPCG88_LPM_CUR)	32	RW	0000_0000h
9620h	LPCG working status (LPCG88_STATUS0)	32	R	0000_0001h
9624h	LPCG domain status (LPCG88_STATUS1)	32	R	0000_FFFFh
9630h	LPCG access control (LPCG88_AUTHEN)	32	RW	FFFF_0000h
9640h	LPCG direct control (LPCG89_DIRECT)	32	RW	0000_0001h
9650h	Clock source low power mode setting (LPCG89_LPM0)	32	RW	0000_0000h
9654h	clock source low power mode setting (LPCG89_LPM1)	32	RW	0000_0000h
965Ch	LPM setting of current CPU domain (LPCG89_LPM_CUR)	32	RW	0000_0000h
9660h	LPCG working status (LPCG89_STATUS0)	32	R	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9664h	LPCG domain status (LPCG89_STATUS1)	32	R	0000_FFFFh
9670h	LPCG access control (LPCG89_AUTHEN)	32	RW	FFFF_0000h
9680h	LPCG direct control (LPCG90_DIRECT)	32	RW	0000_0001h
9690h	Clock source low power mode setting (LPCG90_LPM0)	32	RW	0000_0000h
9694h	clock source low power mode setting (LPCG90_LPM1)	32	RW	0000_0000h
969Ch	LPM setting of current CPU domain (LPCG90_LPM_CUR)	32	RW	0000_0000h
96A0h	LPCG working status (LPCG90_STATUS0)	32	R	0000_0001h
96A4h	LPCG domain status (LPCG90_STATUS1)	32	R	0000_FFFFh
96B0h	LPCG access control (LPCG90_AUTHEN)	32	RW	FFFF_0000h
96C0h	LPCG direct control (LPCG91_DIRECT)	32	RW	0000_0001h
96D0h	Clock source low power mode setting (LPCG91_LPM0)	32	RW	0000_0000h
96D4h	clock source low power mode setting (LPCG91_LPM1)	32	RW	0000_0000h
96DCh	LPM setting of current CPU domain (LPCG91_LPM_CUR)	32	RW	0000_0000h
96E0h	LPCG working status (LPCG91_STATUS0)	32	R	0000_0001h
96E4h	LPCG domain status (LPCG91_STATUS1)	32	R	0000_FFFFh
96F0h	LPCG access control (LPCG91_AUTHEN)	32	RW	FFFF_0000h
9700h	LPCG direct control (LPCG92_DIRECT)	32	RW	0000_0001h
9710h	Clock source low power mode setting (LPCG92_LPM0)	32	RW	0000_0000h
9714h	clock source low power mode setting (LPCG92_LPM1)	32	RW	0000_0000h
971Ch	LPM setting of current CPU domain (LPCG92_LPM_CUR)	32	RW	0000_0000h
9720h	LPCG working status (LPCG92_STATUS0)	32	R	0000_0001h
9724h	LPCG domain status (LPCG92_STATUS1)	32	R	0000_FFFFh
9730h	LPCG access control (LPCG92_AUTHEN)	32	RW	FFFF_0000h
9740h	LPCG direct control (LPCG93_DIRECT)	32	RW	0000_0001h
9750h	Clock source low power mode setting (LPCG93_LPM0)	32	RW	0000_0000h
9754h	clock source low power mode setting (LPCG93_LPM1)	32	RW	0000_0000h
975Ch	LPM setting of current CPU domain (LPCG93_LPM_CUR)	32	RW	0000_0000h
9760h	LPCG working status (LPCG93_STATUS0)	32	R	0000_0001h
9764h	LPCG domain status (LPCG93_STATUS1)	32	R	0000_FFFFh
9770h	LPCG access control (LPCG93_AUTHEN)	32	RW	FFFF_0000h
9780h	LPCG direct control (LPCG94_DIRECT)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9790h	Clock source low power mode setting (LPCG94_LPM0)	32	RW	0000_0000h
9794h	clock source low power mode setting (LPCG94_LPM1)	32	RW	0000_0000h
979Ch	LPM setting of current CPU domain (LPCG94_LPM_CUR)	32	RW	0000_0000h
97A0h	LPCG working status (LPCG94_STATUS0)	32	R	0000_0001h
97A4h	LPCG domain status (LPCG94_STATUS1)	32	R	0000_FFFFh
97B0h	LPCG access control (LPCG94_AUTHEN)	32	RW	FFFF_0000h
97C0h	LPCG direct control (LPCG95_DIRECT)	32	RW	0000_0001h
97D0h	Clock source low power mode setting (LPCG95_LPM0)	32	RW	0000_0000h
97D4h	clock source low power mode setting (LPCG95_LPM1)	32	RW	0000_0000h
97DCh	LPM setting of current CPU domain (LPCG95_LPM_CUR)	32	RW	0000_0000h
97E0h	LPCG working status (LPCG95_STATUS0)	32	R	0000_0001h
97E4h	LPCG domain status (LPCG95_STATUS1)	32	R	0000_FFFFh
97F0h	LPCG access control (LPCG95_AUTHEN)	32	RW	FFFF_0000h
9800h	LPCG direct control (LPCG96_DIRECT)	32	RW	0000_0001h
9810h	Clock source low power mode setting (LPCG96_LPM0)	32	RW	0000_0000h
9814h	clock source low power mode setting (LPCG96_LPM1)	32	RW	0000_0000h
981Ch	LPM setting of current CPU domain (LPCG96_LPM_CUR)	32	RW	0000_0000h
9820h	LPCG working status (LPCG96_STATUS0)	32	R	0000_0001h
9824h	LPCG domain status (LPCG96_STATUS1)	32	R	0000_FFFFh
9830h	LPCG access control (LPCG96_AUTHEN)	32	RW	FFFF_0000h
9840h	LPCG direct control (LPCG97_DIRECT)	32	RW	0000_0001h
9850h	Clock source low power mode setting (LPCG97_LPM0)	32	RW	0000_0000h
9854h	clock source low power mode setting (LPCG97_LPM1)	32	RW	0000_0000h
985Ch	LPM setting of current CPU domain (LPCG97_LPM_CUR)	32	RW	0000_0000h
9860h	LPCG working status (LPCG97_STATUS0)	32	R	0000_0001h
9864h	LPCG domain status (LPCG97_STATUS1)	32	R	0000_FFFFh
9870h	LPCG access control (LPCG97_AUTHEN)	32	RW	FFFF_0000h
9880h	LPCG direct control (LPCG98_DIRECT)	32	RW	0000_0001h
9890h	Clock source low power mode setting (LPCG98_LPM0)	32	RW	0000_0000h
9894h	clock source low power mode setting (LPCG98_LPM1)	32	RW	0000_0000h
989Ch	LPM setting of current CPU domain (LPCG98_LPM_CUR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
98A0h	LPCG working status (LPCG98_STATUS0)	32	R	0000_0001h
98A4h	LPCG domain status (LPCG98_STATUS1)	32	R	0000_FFFFh
98B0h	LPCG access control (LPCG98_AUTHEN)	32	RW	FFFF_0000h
98C0h	LPCG direct control (LPCG99_DIRECT)	32	RW	0000_0001h
98D0h	Clock source low power mode setting (LPCG99_LPM0)	32	RW	0000_0000h
98D4h	clock source low power mode setting (LPCG99_LPM1)	32	RW	0000_0000h
98DCh	LPM setting of current CPU domain (LPCG99_LPM_CUR)	32	RW	0000_0000h
98E0h	LPCG working status (LPCG99_STATUS0)	32	R	0000_0001h
98E4h	LPCG domain status (LPCG99_STATUS1)	32	R	0000_FFFFh
98F0h	LPCG access control (LPCG99_AUTHEN)	32	RW	FFFF_0000h
9900h	LPCG direct control (LPCG100_DIRECT)	32	RW	0000_0001h
9910h	Clock source low power mode setting (LPCG100_LPM0)	32	RW	0000_0000h
9914h	clock source low power mode setting (LPCG100_LPM1)	32	RW	0000_0000h
991Ch	LPM setting of current CPU domain (LPCG100_LPM_CUR)	32	RW	0000_0000h
9920h	LPCG working status (LPCG100_STATUS0)	32	R	0000_0001h
9924h	LPCG domain status (LPCG100_STATUS1)	32	R	0000_FFFFh
9930h	LPCG access control (LPCG100_AUTHEN)	32	RW	FFFF_0000h
9940h	LPCG direct control (LPCG101_DIRECT)	32	RW	0000_0001h
9950h	Clock source low power mode setting (LPCG101_LPM0)	32	RW	0000_0000h
9954h	clock source low power mode setting (LPCG101_LPM1)	32	RW	0000_0000h
995Ch	LPM setting of current CPU domain (LPCG101_LPM_CUR)	32	RW	0000_0000h
9960h	LPCG working status (LPCG101_STATUS0)	32	R	0000_0001h
9964h	LPCG domain status (LPCG101_STATUS1)	32	R	0000_FFFFh
9970h	LPCG access control (LPCG101_AUTHEN)	32	RW	FFFF_0000h
9980h	LPCG direct control (LPCG102_DIRECT)	32	RW	0000_0001h
9990h	Clock source low power mode setting (LPCG102_LPM0)	32	RW	0000_0000h
9994h	clock source low power mode setting (LPCG102_LPM1)	32	RW	0000_0000h
999Ch	LPM setting of current CPU domain (LPCG102_LPM_CUR)	32	RW	0000_0000h
99A0h	LPCG working status (LPCG102_STATUS0)	32	R	0000_0001h
99A4h	LPCG domain status (LPCG102_STATUS1)	32	R	0000_FFFFh
99B0h	LPCG access control (LPCG102_AUTHEN)	32	RW	FFFF_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
99C0h	LPCG direct control (LPCG103_DIRECT)	32	RW	0000_0001h
99D0h	Clock source low power mode setting (LPCG103_LPM0)	32	RW	0000_0000h
99D4h	clock source low power mode setting (LPCG103_LPM1)	32	RW	0000_0000h
99DCh	LPM setting of current CPU domain (LPCG103_LPM_CUR)	32	RW	0000_0000h
99E0h	LPCG working status (LPCG103_STATUS0)	32	R	0000_0001h
99E4h	LPCG domain status (LPCG103_STATUS1)	32	R	0000_FFFFh
99F0h	LPCG access control (LPCG103_AUTHEN)	32	RW	FFFF_0000h
9A00h	LPCG direct control (LPCG104_DIRECT)	32	RW	0000_0001h
9A10h	Clock source low power mode setting (LPCG104_LPM0)	32	RW	0000_0000h
9A14h	clock source low power mode setting (LPCG104_LPM1)	32	RW	0000_0000h
9A1Ch	LPM setting of current CPU domain (LPCG104_LPM_CUR)	32	RW	0000_0000h
9A20h	LPCG working status (LPCG104_STATUS0)	32	R	0000_0001h
9A24h	LPCG domain status (LPCG104_STATUS1)	32	R	0000_FFFFh
9A30h	LPCG access control (LPCG104_AUTHEN)	32	RW	FFFF_0000h
9A40h	LPCG direct control (LPCG105_DIRECT)	32	RW	0000_0001h
9A50h	Clock source low power mode setting (LPCG105_LPM0)	32	RW	0000_0000h
9A54h	clock source low power mode setting (LPCG105_LPM1)	32	RW	0000_0000h
9A5Ch	LPM setting of current CPU domain (LPCG105_LPM_CUR)	32	RW	0000_0000h
9A60h	LPCG working status (LPCG105_STATUS0)	32	R	0000_0001h
9A64h	LPCG domain status (LPCG105_STATUS1)	32	R	0000_FFFFh
9A70h	LPCG access control (LPCG105_AUTHEN)	32	RW	FFFF_0000h
9A80h	LPCG direct control (LPCG106_DIRECT)	32	RW	0000_0001h
9A90h	Clock source low power mode setting (LPCG106_LPM0)	32	RW	0000_0000h
9A94h	clock source low power mode setting (LPCG106_LPM1)	32	RW	0000_0000h
9A9Ch	LPM setting of current CPU domain (LPCG106_LPM_CUR)	32	RW	0000_0000h
9AA0h	LPCG working status (LPCG106_STATUS0)	32	R	0000_0001h
9AA4h	LPCG domain status (LPCG106_STATUS1)	32	R	0000_FFFFh
9AB0h	LPCG access control (LPCG106_AUTHEN)	32	RW	FFFF_0000h
9AC0h	LPCG direct control (LPCG107_DIRECT)	32	RW	0000_0001h
9AD0h	Clock source low power mode setting (LPCG107_LPM0)	32	RW	0000_0000h
9AD4h	clock source low power mode setting (LPCG107_LPM1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9ADCh	LPM setting of current CPU domain (LPCG107_LPM_CUR)	32	RW	0000_0000h
9AE0h	LPCG working status (LPCG107_STATUS0)	32	R	0000_0001h
9AE4h	LPCG domain status (LPCG107_STATUS1)	32	R	0000_FFFFh
9AF0h	LPCG access control (LPCG107_AUTHEN)	32	RW	FFFF_0000h
9B00h	LPCG direct control (LPCG108_DIRECT)	32	RW	0000_0001h
9B10h	Clock source low power mode setting (LPCG108_LPM0)	32	RW	0000_0000h
9B14h	clock source low power mode setting (LPCG108_LPM1)	32	RW	0000_0000h
9B1Ch	LPM setting of current CPU domain (LPCG108_LPM_CUR)	32	RW	0000_0000h
9B20h	LPCG working status (LPCG108_STATUS0)	32	R	0000_0001h
9B24h	LPCG domain status (LPCG108_STATUS1)	32	R	0000_FFFFh
9B30h	LPCG access control (LPCG108_AUTHEN)	32	RW	FFFF_0000h
9B40h	LPCG direct control (LPCG109_DIRECT)	32	RW	0000_0001h
9B50h	Clock source low power mode setting (LPCG109_LPM0)	32	RW	0000_0000h
9B54h	clock source low power mode setting (LPCG109_LPM1)	32	RW	0000_0000h
9B5Ch	LPM setting of current CPU domain (LPCG109_LPM_CUR)	32	RW	0000_0000h
9B60h	LPCG working status (LPCG109_STATUS0)	32	R	0000_0001h
9B64h	LPCG domain status (LPCG109_STATUS1)	32	R	0000_FFFFh
9B70h	LPCG access control (LPCG109_AUTHEN)	32	RW	FFFF_0000h
9B80h	LPCG direct control (LPCG110_DIRECT)	32	RW	0000_0001h
9B90h	Clock source low power mode setting (LPCG110_LPM0)	32	RW	0000_0000h
9B94h	clock source low power mode setting (LPCG110_LPM1)	32	RW	0000_0000h
9B9Ch	LPM setting of current CPU domain (LPCG110_LPM_CUR)	32	RW	0000_0000h
9BA0h	LPCG working status (LPCG110_STATUS0)	32	R	0000_0001h
9BA4h	LPCG domain status (LPCG110_STATUS1)	32	R	0000_FFFFh
9BB0h	LPCG access control (LPCG110_AUTHEN)	32	RW	FFFF_0000h
9BC0h	LPCG direct control (LPCG111_DIRECT)	32	RW	0000_0001h
9BD0h	Clock source low power mode setting (LPCG111_LPM0)	32	RW	0000_0000h
9BD4h	clock source low power mode setting (LPCG111_LPM1)	32	RW	0000_0000h
9BDCh	LPM setting of current CPU domain (LPCG111_LPM_CUR)	32	RW	0000_0000h
9BE0h	LPCG working status (LPCG111_STATUS0)	32	R	0000_0001h
9BE4h	LPCG domain status (LPCG111_STATUS1)	32	R	0000_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9BF0h	LPCG access control (LPCG111_AUTHEN)	32	RW	FFFF_0000h
9C00h	LPCG direct control (LPCG112_DIRECT)	32	RW	0000_0001h
9C10h	Clock source low power mode setting (LPCG112_LPM0)	32	RW	0000_0000h
9C14h	clock source low power mode setting (LPCG112_LPM1)	32	RW	0000_0000h
9C1Ch	LPM setting of current CPU domain (LPCG112_LPM_CUR)	32	RW	0000_0000h
9C20h	LPCG working status (LPCG112_STATUS0)	32	R	0000_0001h
9C24h	LPCG domain status (LPCG112_STATUS1)	32	R	0000_FFFFh
9C30h	LPCG access control (LPCG112_AUTHEN)	32	RW	FFFF_0000h
9C40h	LPCG direct control (LPCG113_DIRECT)	32	RW	0000_0001h
9C50h	Clock source low power mode setting (LPCG113_LPM0)	32	RW	0000_0000h
9C54h	clock source low power mode setting (LPCG113_LPM1)	32	RW	0000_0000h
9C5Ch	LPM setting of current CPU domain (LPCG113_LPM_CUR)	32	RW	0000_0000h
9C60h	LPCG working status (LPCG113_STATUS0)	32	R	0000_0001h
9C64h	LPCG domain status (LPCG113_STATUS1)	32	R	0000_FFFFh
9C70h	LPCG access control (LPCG113_AUTHEN)	32	RW	FFFF_0000h
9C80h	LPCG direct control (LPCG114_DIRECT)	32	RW	0000_0001h
9C90h	Clock source low power mode setting (LPCG114_LPM0)	32	RW	0000_0000h
9C94h	clock source low power mode setting (LPCG114_LPM1)	32	RW	0000_0000h
9C9Ch	LPM setting of current CPU domain (LPCG114_LPM_CUR)	32	RW	0000_0000h
9CA0h	LPCG working status (LPCG114_STATUS0)	32	R	0000_0001h
9CA4h	LPCG domain status (LPCG114_STATUS1)	32	R	0000_FFFFh
9CB0h	LPCG access control (LPCG114_AUTHEN)	32	RW	FFFF_0000h
9CC0h	LPCG direct control (LPCG115_DIRECT)	32	RW	0000_0001h
9CD0h	Clock source low power mode setting (LPCG115_LPM0)	32	RW	0000_0000h
9CD4h	clock source low power mode setting (LPCG115_LPM1)	32	RW	0000_0000h
9CDCh	LPM setting of current CPU domain (LPCG115_LPM_CUR)	32	RW	0000_0000h
9CE0h	LPCG working status (LPCG115_STATUS0)	32	R	0000_0001h
9CE4h	LPCG domain status (LPCG115_STATUS1)	32	R	0000_FFFFh
9CF0h	LPCG access control (LPCG115_AUTHEN)	32	RW	FFFF_0000h
9D00h	LPCG direct control (LPCG116_DIRECT)	32	RW	0000_0001h
9D10h	Clock source low power mode setting (LPCG116_LPM0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9D14h	clock source low power mode setting (LPCG116_LPM1)	32	RW	0000_0000h
9D1Ch	LPM setting of current CPU domain (LPCG116_LPM_CUR)	32	RW	0000_0000h
9D20h	LPCG working status (LPCG116_STATUS0)	32	R	0000_0001h
9D24h	LPCG domain status (LPCG116_STATUS1)	32	R	0000_FFFFh
9D30h	LPCG access control (LPCG116_AUTHEN)	32	RW	FFFF_0000h
9D40h	LPCG direct control (LPCG117_DIRECT)	32	RW	0000_0001h
9D50h	Clock source low power mode setting (LPCG117_LPM0)	32	RW	0000_0000h
9D54h	clock source low power mode setting (LPCG117_LPM1)	32	RW	0000_0000h
9D5Ch	LPM setting of current CPU domain (LPCG117_LPM_CUR)	32	RW	0000_0000h
9D60h	LPCG working status (LPCG117_STATUS0)	32	R	0000_0001h
9D64h	LPCG domain status (LPCG117_STATUS1)	32	R	0000_FFFFh
9D70h	LPCG access control (LPCG117_AUTHEN)	32	RW	FFFF_0000h
9D80h	LPCG direct control (LPCG118_DIRECT)	32	RW	0000_0001h
9D90h	Clock source low power mode setting (LPCG118_LPM0)	32	RW	0000_0000h
9D94h	clock source low power mode setting (LPCG118_LPM1)	32	RW	0000_0000h
9D9Ch	LPM setting of current CPU domain (LPCG118_LPM_CUR)	32	RW	0000_0000h
9DA0h	LPCG working status (LPCG118_STATUS0)	32	R	0000_0001h
9DA4h	LPCG domain status (LPCG118_STATUS1)	32	R	0000_FFFFh
9DB0h	LPCG access control (LPCG118_AUTHEN)	32	RW	FFFF_0000h
9DC0h	LPCG direct control (LPCG119_DIRECT)	32	RW	0000_0001h
9DD0h	Clock source low power mode setting (LPCG119_LPM0)	32	RW	0000_0000h
9DD4h	clock source low power mode setting (LPCG119_LPM1)	32	RW	0000_0000h
9DDCh	LPM setting of current CPU domain (LPCG119_LPM_CUR)	32	RW	0000_0000h
9DE0h	LPCG working status (LPCG119_STATUS0)	32	R	0000_0001h
9DE4h	LPCG domain status (LPCG119_STATUS1)	32	R	0000_FFFFh
9DF0h	LPCG access control (LPCG119_AUTHEN)	32	RW	FFFF_0000h
9E00h	LPCG direct control (LPCG120_DIRECT)	32	RW	0000_0001h
9E10h	Clock source low power mode setting (LPCG120_LPM0)	32	RW	0000_0000h
9E14h	clock source low power mode setting (LPCG120_LPM1)	32	RW	0000_0000h
9E1Ch	LPM setting of current CPU domain (LPCG120_LPM_CUR)	32	RW	0000_0000h
9E20h	LPCG working status (LPCG120_STATUS0)	32	R	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9E24h	LPCG domain status (LPCG120_STATUS1)	32	R	0000_FFFFh
9E30h	LPCG access control (LPCG120_AUTHEN)	32	RW	FFFF_0000h
9E40h	LPCG direct control (LPCG121_DIRECT)	32	RW	0000_0001h
9E50h	Clock source low power mode setting (LPCG121_LPM0)	32	RW	0000_0000h
9E54h	clock source low power mode setting (LPCG121_LPM1)	32	RW	0000_0000h
9E5Ch	LPM setting of current CPU domain (LPCG121_LPM_CUR)	32	RW	0000_0000h
9E60h	LPCG working status (LPCG121_STATUS0)	32	R	0000_0001h
9E64h	LPCG domain status (LPCG121_STATUS1)	32	R	0000_FFFFh
9E70h	LPCG access control (LPCG121_AUTHEN)	32	RW	FFFF_0000h
9E80h	LPCG direct control (LPCG122_DIRECT)	32	RW	0000_0001h
9E90h	Clock source low power mode setting (LPCG122_LPM0)	32	RW	0000_0000h
9E94h	clock source low power mode setting (LPCG122_LPM1)	32	RW	0000_0000h
9E9Ch	LPM setting of current CPU domain (LPCG122_LPM_CUR)	32	RW	0000_0000h
9EA0h	LPCG working status (LPCG122_STATUS0)	32	R	0000_0001h
9EA4h	LPCG domain status (LPCG122_STATUS1)	32	R	0000_FFFFh
9EB0h	LPCG access control (LPCG122_AUTHEN)	32	RW	FFFF_0000h
9EC0h	LPCG direct control (LPCG123_DIRECT)	32	RW	0000_0001h
9ED0h	Clock source low power mode setting (LPCG123_LPM0)	32	RW	0000_0000h
9ED4h	clock source low power mode setting (LPCG123_LPM1)	32	RW	0000_0000h
9EDCh	LPM setting of current CPU domain (LPCG123_LPM_CUR)	32	RW	0000_0000h
9EE0h	LPCG working status (LPCG123_STATUS0)	32	R	0000_0001h
9EE4h	LPCG domain status (LPCG123_STATUS1)	32	R	0000_FFFFh
9EF0h	LPCG access control (LPCG123_AUTHEN)	32	RW	FFFF_0000h
9F00h	LPCG direct control (LPCG124_DIRECT)	32	RW	0000_0001h
9F10h	Clock source low power mode setting (LPCG124_LPM0)	32	RW	0000_0000h
9F14h	clock source low power mode setting (LPCG124_LPM1)	32	RW	0000_0000h
9F1Ch	LPM setting of current CPU domain (LPCG124_LPM_CUR)	32	RW	0000_0000h
9F20h	LPCG working status (LPCG124_STATUS0)	32	R	0000_0001h
9F24h	LPCG domain status (LPCG124_STATUS1)	32	R	0000_FFFFh
9F30h	LPCG access control (LPCG124_AUTHEN)	32	RW	FFFF_0000h
9F40h	LPCG direct control (LPCG125_DIRECT)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9F50h	Clock source low power mode setting (LPCG125_LPM0)	32	RW	0000_0000h
9F54h	clock source low power mode setting (LPCG125_LPM1)	32	RW	0000_0000h
9F5Ch	LPM setting of current CPU domain (LPCG125_LPM_CUR)	32	RW	0000_0000h
9F60h	LPCG working status (LPCG125_STATUS0)	32	R	0000_0001h
9F64h	LPCG domain status (LPCG125_STATUS1)	32	R	0000_FFFFh
9F70h	LPCG access control (LPCG125_AUTHEN)	32	RW	FFFF_0000h
9F80h	LPCG direct control (LPCG126_DIRECT)	32	RW	0000_0001h
9F90h	Clock source low power mode setting (LPCG126_LPM0)	32	RW	0000_0000h
9F94h	clock source low power mode setting (LPCG126_LPM1)	32	RW	0000_0000h
9F9Ch	LPM setting of current CPU domain (LPCG126_LPM_CUR)	32	RW	0000_0000h
9FA0h	LPCG working status (LPCG126_STATUS0)	32	R	0000_0001h
9FA4h	LPCG domain status (LPCG126_STATUS1)	32	R	0000_FFFFh
9FB0h	LPCG access control (LPCG126_AUTHEN)	32	RW	FFFF_0000h
9FC0h	LPCG direct control (LPCG127_DIRECT)	32	RW	0000_0001h
9FD0h	Clock source low power mode setting (LPCG127_LPM0)	32	RW	0000_0000h
9FD4h	clock source low power mode setting (LPCG127_LPM1)	32	RW	0000_0000h
9FDCh	LPM setting of current CPU domain (LPCG127_LPM_CUR)	32	RW	0000_0000h
9FE0h	LPCG working status (LPCG127_STATUS0)	32	R	0000_0001h
9FE4h	LPCG domain status (LPCG127_STATUS1)	32	R	0000_FFFFh
9FF0h	LPCG access control (LPCG127_AUTHEN)	32	RW	FFFF_0000h
A000h	LPCG direct control (LPCG128_DIRECT)	32	RW	0000_0001h
A010h	Clock source low power mode setting (LPCG128_LPM0)	32	RW	0000_0000h
A014h	clock source low power mode setting (LPCG128_LPM1)	32	RW	0000_0000h
A01Ch	LPM setting of current CPU domain (LPCG128_LPM_CUR)	32	RW	0000_0000h
A020h	LPCG working status (LPCG128_STATUS0)	32	R	0000_0001h
A024h	LPCG domain status (LPCG128_STATUS1)	32	R	0000_FFFFh
A030h	LPCG access control (LPCG128_AUTHEN)	32	RW	FFFF_0000h
A040h	LPCG direct control (LPCG129_DIRECT)	32	RW	0000_0001h
A050h	Clock source low power mode setting (LPCG129_LPM0)	32	RW	0000_0000h
A054h	clock source low power mode setting (LPCG129_LPM1)	32	RW	0000_0000h
A05Ch	LPM setting of current CPU domain (LPCG129_LPM_CUR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A060h	LPCG working status (LPCG129_STATUS0)	32	R	0000_0001h
A064h	LPCG domain status (LPCG129_STATUS1)	32	R	0000_FFFFh
A070h	LPCG access control (LPCG129_AUTHEN)	32	RW	FFFF_0000h
A080h	LPCG direct control (LPCG130_DIRECT)	32	RW	0000_0001h
A090h	Clock source low power mode setting (LPCG130_LPM0)	32	RW	0000_0000h
A094h	clock source low power mode setting (LPCG130_LPM1)	32	RW	0000_0000h
A09Ch	LPM setting of current CPU domain (LPCG130_LPM_CUR)	32	RW	0000_0000h
A0A0h	LPCG working status (LPCG130_STATUS0)	32	R	0000_0001h
A0A4h	LPCG domain status (LPCG130_STATUS1)	32	R	0000_FFFFh
A0B0h	LPCG access control (LPCG130_AUTHEN)	32	RW	FFFF_0000h
A0C0h	LPCG direct control (LPCG131_DIRECT)	32	RW	0000_0001h
A0D0h	Clock source low power mode setting (LPCG131_LPM0)	32	RW	0000_0000h
A0D4h	clock source low power mode setting (LPCG131_LPM1)	32	RW	0000_0000h
A0DCh	LPM setting of current CPU domain (LPCG131_LPM_CUR)	32	RW	0000_0000h
A0E0h	LPCG working status (LPCG131_STATUS0)	32	R	0000_0001h
A0E4h	LPCG domain status (LPCG131_STATUS1)	32	R	0000_FFFFh
A0F0h	LPCG access control (LPCG131_AUTHEN)	32	RW	FFFF_0000h
A100h	LPCG direct control (LPCG132_DIRECT)	32	RW	0000_0001h
A110h	Clock source low power mode setting (LPCG132_LPM0)	32	RW	0000_0000h
A114h	clock source low power mode setting (LPCG132_LPM1)	32	RW	0000_0000h
A11Ch	LPM setting of current CPU domain (LPCG132_LPM_CUR)	32	RW	0000_0000h
A120h	LPCG working status (LPCG132_STATUS0)	32	R	0000_0001h
A124h	LPCG domain status (LPCG132_STATUS1)	32	R	0000_FFFFh
A130h	LPCG access control (LPCG132_AUTHEN)	32	RW	FFFF_0000h
A140h	LPCG direct control (LPCG133_DIRECT)	32	RW	0000_0001h
A150h	Clock source low power mode setting (LPCG133_LPM0)	32	RW	0000_0000h
A154h	clock source low power mode setting (LPCG133_LPM1)	32	RW	0000_0000h
A15Ch	LPM setting of current CPU domain (LPCG133_LPM_CUR)	32	RW	0000_0000h
A160h	LPCG working status (LPCG133_STATUS0)	32	R	0000_0001h
A164h	LPCG domain status (LPCG133_STATUS1)	32	R	0000_FFFFh
A170h	LPCG access control (LPCG133_AUTHEN)	32	RW	FFFF_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A180h	LPCG direct control (LPCG134_DIRECT)	32	RW	0000_0001h
A190h	Clock source low power mode setting (LPCG134_LPM0)	32	RW	0000_0000h
A194h	clock source low power mode setting (LPCG134_LPM1)	32	RW	0000_0000h
A19Ch	LPM setting of current CPU domain (LPCG134_LPM_CUR)	32	RW	0000_0000h
A1A0h	LPCG working status (LPCG134_STATUS0)	32	R	0000_0001h
A1A4h	LPCG domain status (LPCG134_STATUS1)	32	R	0000_FFFFh
A1B0h	LPCG access control (LPCG134_AUTHEN)	32	RW	FFFF_0000h
A1C0h	LPCG direct control (LPCG135_DIRECT)	32	RW	0000_0001h
A1D0h	Clock source low power mode setting (LPCG135_LPM0)	32	RW	0000_0000h
A1D4h	clock source low power mode setting (LPCG135_LPM1)	32	RW	0000_0000h
A1DCh	LPM setting of current CPU domain (LPCG135_LPM_CUR)	32	RW	0000_0000h
A1E0h	LPCG working status (LPCG135_STATUS0)	32	R	0000_0001h
A1E4h	LPCG domain status (LPCG135_STATUS1)	32	R	0000_FFFFh
A1F0h	LPCG access control (LPCG135_AUTHEN)	32	RW	FFFF_0000h
A200h	LPCG direct control (LPCG136_DIRECT)	32	RW	0000_0001h
A210h	Clock source low power mode setting (LPCG136_LPM0)	32	RW	0000_0000h
A214h	clock source low power mode setting (LPCG136_LPM1)	32	RW	0000_0000h
A21Ch	LPM setting of current CPU domain (LPCG136_LPM_CUR)	32	RW	0000_0000h
A220h	LPCG working status (LPCG136_STATUS0)	32	R	0000_0001h
A224h	LPCG domain status (LPCG136_STATUS1)	32	R	0000_FFFFh
A230h	LPCG access control (LPCG136_AUTHEN)	32	RW	FFFF_0000h
A240h	LPCG direct control (LPCG137_DIRECT)	32	RW	0000_0001h
A250h	Clock source low power mode setting (LPCG137_LPM0)	32	RW	0000_0000h
A254h	clock source low power mode setting (LPCG137_LPM1)	32	RW	0000_0000h
A25Ch	LPM setting of current CPU domain (LPCG137_LPM_CUR)	32	RW	0000_0000h
A260h	LPCG working status (LPCG137_STATUS0)	32	R	0000_0001h
A264h	LPCG domain status (LPCG137_STATUS1)	32	R	0000_FFFFh
A270h	LPCG access control (LPCG137_AUTHEN)	32	RW	FFFF_0000h
A280h	LPCG direct control (LPCG138_DIRECT)	32	RW	0000_0001h
A290h	Clock source low power mode setting (LPCG138_LPM0)	32	RW	0000_0000h
A294h	clock source low power mode setting (LPCG138_LPM1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A29Ch	LPM setting of current CPU domain (LPCG138_LPM_CUR)	32	RW	0000_0000h
A2A0h	LPCG working status (LPCG138_STATUS0)	32	R	0000_0001h
A2A4h	LPCG domain status (LPCG138_STATUS1)	32	R	0000_FFFFh
A2B0h	LPCG access control (LPCG138_AUTHEN)	32	RW	FFFF_0000h
A2C0h	LPCG direct control (LPCG139_DIRECT)	32	RW	0000_0001h
A2D0h	Clock source low power mode setting (LPCG139_LPM0)	32	RW	0000_0000h
A2D4h	clock source low power mode setting (LPCG139_LPM1)	32	RW	0000_0000h
A2DCh	LPM setting of current CPU domain (LPCG139_LPM_CUR)	32	RW	0000_0000h
A2E0h	LPCG working status (LPCG139_STATUS0)	32	R	0000_0001h
A2E4h	LPCG domain status (LPCG139_STATUS1)	32	R	0000_FFFFh
A2F0h	LPCG access control (LPCG139_AUTHEN)	32	RW	FFFF_0000h
A300h	LPCG direct control (LPCG140_DIRECT)	32	RW	0000_0001h
A310h	Clock source low power mode setting (LPCG140_LPM0)	32	RW	0000_0000h
A314h	clock source low power mode setting (LPCG140_LPM1)	32	RW	0000_0000h
A31Ch	LPM setting of current CPU domain (LPCG140_LPM_CUR)	32	RW	0000_0000h
A320h	LPCG working status (LPCG140_STATUS0)	32	R	0000_0001h
A324h	LPCG domain status (LPCG140_STATUS1)	32	R	0000_FFFFh
A330h	LPCG access control (LPCG140_AUTHEN)	32	RW	FFFF_0000h
A340h	LPCG direct control (LPCG141_DIRECT)	32	RW	0000_0001h
A350h	Clock source low power mode setting (LPCG141_LPM0)	32	RW	0000_0000h
A354h	clock source low power mode setting (LPCG141_LPM1)	32	RW	0000_0000h
A35Ch	LPM setting of current CPU domain (LPCG141_LPM_CUR)	32	RW	0000_0000h
A360h	LPCG working status (LPCG141_STATUS0)	32	R	0000_0001h
A364h	LPCG domain status (LPCG141_STATUS1)	32	R	0000_FFFFh
A370h	LPCG access control (LPCG141_AUTHEN)	32	RW	FFFF_0000h
A380h	LPCG direct control (LPCG142_DIRECT)	32	RW	0000_0001h
A390h	Clock source low power mode setting (LPCG142_LPM0)	32	RW	0000_0000h
A394h	clock source low power mode setting (LPCG142_LPM1)	32	RW	0000_0000h
A39Ch	LPM setting of current CPU domain (LPCG142_LPM_CUR)	32	RW	0000_0000h
A3A0h	LPCG working status (LPCG142_STATUS0)	32	R	0000_0001h
A3A4h	LPCG domain status (LPCG142_STATUS1)	32	R	0000_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A3B0h	LPCG access control (LPCG142_AUTHEN)	32	RW	FFFF_0000h
A3C0h	LPCG direct control (LPCG143_DIRECT)	32	RW	0000_0001h
A3D0h	Clock source low power mode setting (LPCG143_LPM0)	32	RW	0000_0000h
A3D4h	clock source low power mode setting (LPCG143_LPM1)	32	RW	0000_0000h
A3DCh	LPM setting of current CPU domain (LPCG143_LPM_CUR)	32	RW	0000_0000h
A3E0h	LPCG working status (LPCG143_STATUS0)	32	R	0000_0001h
A3E4h	LPCG domain status (LPCG143_STATUS1)	32	R	0000_FFFFh
A3F0h	LPCG access control (LPCG143_AUTHEN)	32	RW	FFFF_0000h
A400h	LPCG direct control (LPCG144_DIRECT)	32	RW	0000_0001h
A410h	Clock source low power mode setting (LPCG144_LPM0)	32	RW	0000_0000h
A414h	clock source low power mode setting (LPCG144_LPM1)	32	RW	0000_0000h
A41Ch	LPM setting of current CPU domain (LPCG144_LPM_CUR)	32	RW	0000_0000h
A420h	LPCG working status (LPCG144_STATUS0)	32	R	0000_0001h
A424h	LPCG domain status (LPCG144_STATUS1)	32	R	0000_FFFFh
A430h	LPCG access control (LPCG144_AUTHEN)	32	RW	FFFF_0000h
A440h	LPCG direct control (LPCG145_DIRECT)	32	RW	0000_0001h
A450h	Clock source low power mode setting (LPCG145_LPM0)	32	RW	0000_0000h
A454h	clock source low power mode setting (LPCG145_LPM1)	32	RW	0000_0000h
A45Ch	LPM setting of current CPU domain (LPCG145_LPM_CUR)	32	RW	0000_0000h
A460h	LPCG working status (LPCG145_STATUS0)	32	R	0000_0001h
A464h	LPCG domain status (LPCG145_STATUS1)	32	R	0000_FFFFh
A470h	LPCG access control (LPCG145_AUTHEN)	32	RW	FFFF_0000h
A480h	LPCG direct control (LPCG146_DIRECT)	32	RW	0000_0001h
A490h	Clock source low power mode setting (LPCG146_LPM0)	32	RW	0000_0000h
A494h	clock source low power mode setting (LPCG146_LPM1)	32	RW	0000_0000h
A49Ch	LPM setting of current CPU domain (LPCG146_LPM_CUR)	32	RW	0000_0000h
A4A0h	LPCG working status (LPCG146_STATUS0)	32	R	0000_0001h
A4A4h	LPCG domain status (LPCG146_STATUS1)	32	R	0000_FFFFh
A4B0h	LPCG access control (LPCG146_AUTHEN)	32	RW	FFFF_0000h
A4C0h	LPCG direct control (LPCG147_DIRECT)	32	RW	0000_0001h
A4D0h	Clock source low power mode setting (LPCG147_LPM0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A4D4h	clock source low power mode setting (LPCG147_LPM1)	32	RW	0000_0000h
A4DCh	LPM setting of current CPU domain (LPCG147_LPM_CUR)	32	RW	0000_0000h
A4E0h	LPCG working status (LPCG147_STATUS0)	32	R	0000_0001h
A4E4h	LPCG domain status (LPCG147_STATUS1)	32	R	0000_FFFFh
A4F0h	LPCG access control (LPCG147_AUTHEN)	32	RW	FFFF_0000h
A500h	LPCG direct control (LPCG148_DIRECT)	32	RW	0000_0001h
A510h	Clock source low power mode setting (LPCG148_LPM0)	32	RW	0000_0000h
A514h	clock source low power mode setting (LPCG148_LPM1)	32	RW	0000_0000h
A51Ch	LPM setting of current CPU domain (LPCG148_LPM_CUR)	32	RW	0000_0000h
A520h	LPCG working status (LPCG148_STATUS0)	32	R	0000_0001h
A524h	LPCG domain status (LPCG148_STATUS1)	32	R	0000_FFFFh
A530h	LPCG access control (LPCG148_AUTHEN)	32	RW	FFFF_0000h

20.6.1.2 Clock Root Control Register (CLOCK_ROOT0_CONTROL - CLOCK_ROOT73_CONTROL)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

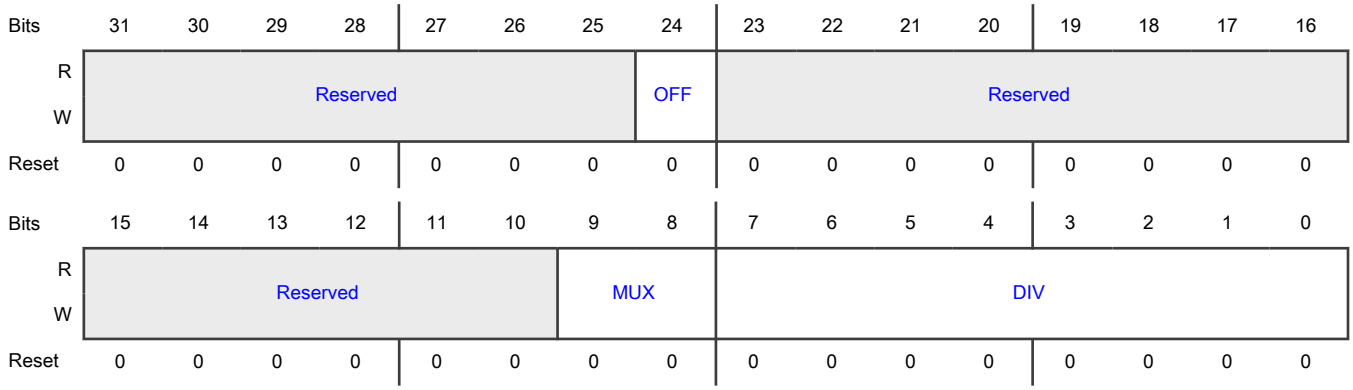
For a = 0 to 73:

Register	Offset	Description
CLOCK_ROOTa_CONTROL	0h + (a × 80h)	Clock Root Control Register
CLOCK_ROOTa_CONTROL_SET	4h + (a × 80h)	Writing 1 to a bit in this register ensures that the corresponding bit in CLOCK_ROOTa_CONTROL is 1
CLOCK_ROOTa_CONTROL_CLR	8h + (a × 80h)	Writing 1 to a bit in this register ensures that the corresponding bit in CLOCK_ROOTa_CONTROL is 0
CLOCK_ROOTa_CONTROL_TOG	Ch + (a × 80h)	Writing 1 to a bit in this register inverts the value of the corresponding bit in CLOCK_ROOTa_CONTROL

Function

This register controls clock root generation, including enable, MUX selection and division factor.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 OFF	Shutdown clock root. Turn off the generated clock root or not. 0b - Clock root is enabled 1b - Clock root is disabled
23-10 —	Reserved
9-8 MUX	Clock multiplexer. Select clock from 4 clock sources to generate the clock root. 00b - Select clock source 0 01b - Select clock source 1 10b - Select clock source 2 11b - Select clock source 3
7-0 DIV	Clock division fraction. The period of generated clock root is DIV + 1 times longer than selected clock source.

20.6.1.3 Clock root working status (CLOCK_ROOT0_STATUS0 - CLOCK_ROOT73_STATUS0)

Offset

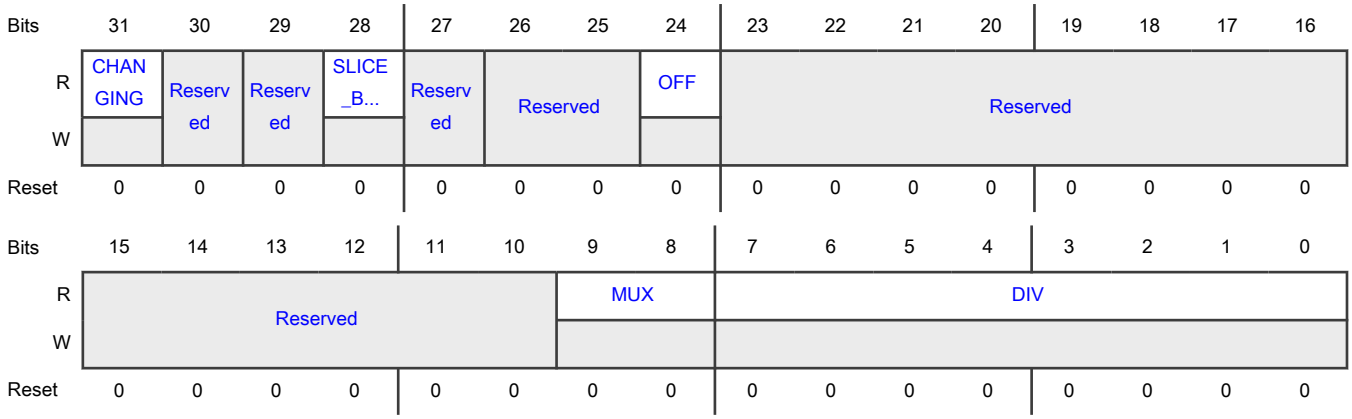
For a = 0 to 73:

Register	Offset
CLOCK_ROOTa_STATUS0	20h + (a × 80h)

Function

This register shows current clock root running status.

Diagram



Fields

Field	Function
31 CHANGING	Indication for clock root internal logic is updating. This status is a combination of UPDATE_FORWARD and SLICE_BUSY. 0b - Clock Status is not updating currently 1b - Clock generation logic is currently updating
30 —	Reserved
29 —	Internal status synchronization to clock generation logic Indication for clock status is synchronizing for clock root. 0b - Synchronization not in process 1b - Synchronization in process
28 SLICE_BUSY	Internal updating in generation logic Indication for clock generation logic is applying new setting. 0b - Clock generation logic is not busy 1b - Clock generation logic is applying the new setting
27 —	Reserved
26-25	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
24 OFF	Current clock root OFF setting Current running state of OFF field for clock root. 0b - Clock root is enabled 1b - Clock root is disabled
23-10 —	Reserved
9-8 MUX	Current clock root MUX setting Current running state of MUX field for clock root. See Clock Root for more info.
7-0 DIV	Current clock root DIV setting Current running state of DIV field for clock root. The 8-bit divider can divide selected clock by up to 256. The Divider value = DIV+1.

20.6.1.4 Clock root access control (CLOCK_ROOT0_AUTHEN - CLOCK_ROOT73_AUTHEN)

Offset

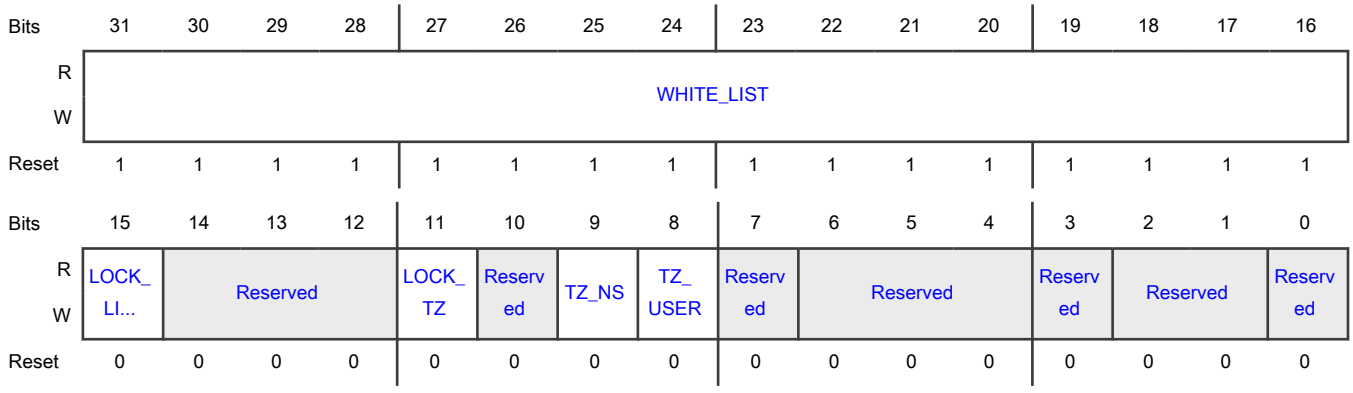
For a = 0 to 73:

Register	Offset
CLOCK_ROOTa_AUTHEN	30h + (a × 80h)

Function

This register controls the access to this Clock Root slice. The access control includes 2 parts: TrustZone control and domain based granting.

Diagram



Fields

Field	Function
31-16 WHITE_LIST	Whitelist settings Each bit in this field represent for one domain. Bit16~Bit31 represent for DOMAIN0~DOMAIN15 respectively. Only the domains the corresponding bit of which is set to 1 can change the registers of this Clock Root.
15 LOCK_LIST	Lock white list This bit locks whitelist of this Clock Root. When this bit is set, CLOCK_ROOTn_AUTHEN[WHITE_LIST] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset. 0b - Whitelist is not locked. 1b - Whitelist is locked.
14-12 —	Reserved
11 LOCK_TZ	Lock TrustZone settings This bit locks TrustZone settings. When this bit is set, CLOCK_ROOTn_AUTHEN[TZ_USER] and CLOCK_ROOTn_AUTHEN[TZ_NS] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset. 0b - TrustZone settings is not locked. 1b - TrustZone settings is locked.
10 —	Reserved
9 TZ_NS	Non-secure access permission Whether the registers of this Clock Root can be changed when CPU is in non-secure mode. The default value is 0, that means only the access from CPU in secure mode can change the registers of this Clock Root. 0b - Cannot be changed in Non-secure mode. 1b - Can be changed in Non-secure mode.
8 TZ_USER	User access permission Whether the registers of this Clock Root can be changed when CPU is in user mode. The default value is 0, that means only the access from CPU in supervisor mode can change the registers of this Clock Root. 0b - Clock Root settings cannot be changed in user mode. 1b - Clock Root settings can be changed in user mode.
7 —	Reserved
6-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3 —	Reserved
2-1 —	Reserved
0 —	Reserved

20.6.1.5 Observe control (OBSERVE0_CONTROL - OBSERVE1_CONTROL)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

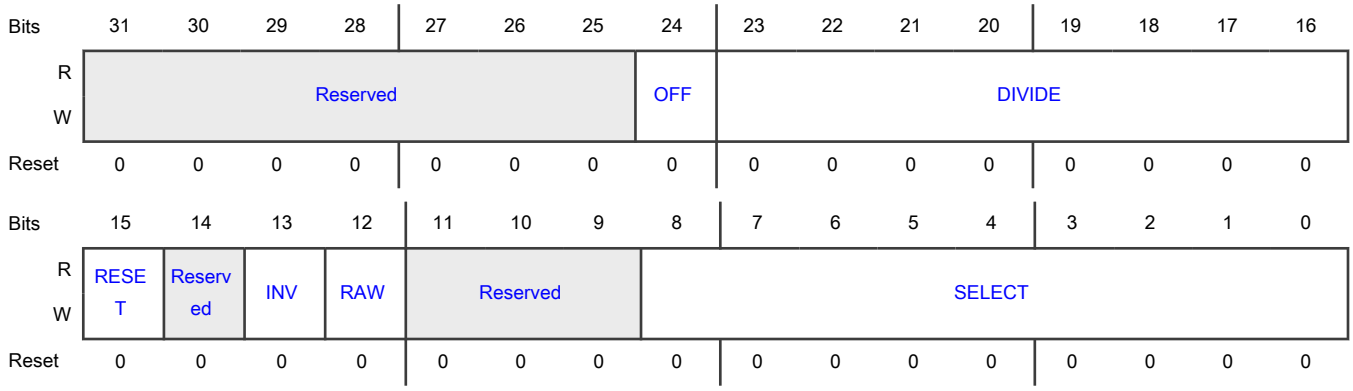
For a = 0 to 1:

Register	Offset	Description
OBSERVEa_CONTROL	4400h + (a × 80h)	Observe control
OBSERVEa_CONTROL_SET	4404h + (a × 80h)	Writing 1 to a bit in this register ensures that the corresponding bit in OBSERVEa_CONTROL is 1
OBSERVEa_CONTROL_CLR	4408h + (a × 80h)	Writing 1 to a bit in this register ensures that the corresponding bit in OBSERVEa_CONTROL is 0
OBSERVEa_CONTROL_TOG	440Ch + (a × 80h)	Writing 1 to a bit in this register inverts the value of the corresponding bit in OBSERVEa_CONTROL

Function

This register contains several control settings of observe slice, such as observed signal selection, disivion factor of the divider for the signals, software reset of the divider and the ON/OFF of observe slice.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 OFF	Turn off Turn off slice to save power. 0b - observe slice is on 1b - observe slice is off
23-16 DIVIDE	Division factor of the divider for observed signal This field sets the division factor of divider before measurement or send out to IO. The divison factor is equal to DIVIDE+1. **NOTE** If divider selected, first serveral toggles maynot be observed. Divider is typically used when observed signal is a continuous clock.
15 RESET	Reset observe divider Reset observe divider. Setting to 1 change will reset the observe divider. **NOTE** This bit is not self-clear, write 0 to release the divider. 0b - Reset deasserts 1b - Reset asserts
14 —	Reserved
13 INV	Invert input signal phase. 0b - Clock phase remain same. 1b - Invert clock phase before measurement or send to IO.
12 RAW	Observe raw signal Select from raw signal and divided signal. 0b - Select divided signal. 1b - Select raw signal.
11-9 —	Reserved
8-0 SELECT	Observe signal selector Selector that controls which observe signals will be selected and observed. The observe slice supports up to 512 signals.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The observe functionality is for debug purposes only, and the accuracy is not guaranteed. Use of this field should be limited to room temperature, and frequency limited to less than 400 MHz.

20.6.1.6 Observe status (OBSERVE0_STATUS - OBSERVE1_STATUS)

Offset

Register	Offset
OBSERVE0_STATUS	4420h
OBSERVE1_STATUS	44A0h

Function

This register contains several current settings of OBSERVEn_CONTROL, and another work status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CHANGING	Reserved	UPDATED...	BUSY	Reserved	Reserved	FREQ_ME...	OFF	DIVIDE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RESET	Reserved	INV	RAW	Reserved				SELECT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 CHANGING	Busy Current observe is busy in updating.
30 —	Reserved
29	Update Forward

Table continues on the next page...

Table continued from the previous page...

Field	Function
UPDATED_FORWARD	
28 BUSY	Busy Current observe is busy updating. 0b - Current observe is not busy 1b - Current observe is busy
27 —	Reserved
26 —	Reserved
25 FREQ_MEASURE_DONE	frequency measurement done flag Frequency measurement is done or on-going/not started. It can be cleared by OBSERVEn_CONTROL[OFF] or OBSERVEn_CONTROL[RESET]. 0b - Frequency measurement is on-going or not started 1b - Frequency measurement is done.
24 OFF	Turn off slice 0b - observe slice is on 1b - observe slice is off
23-16 DIVIDE	Divider for observe signal Divider before measurement or send out to IO. Divider by DIVIDE+1. **NOTE** If divider selected, first several toggles may not be observed. Divider is typically be used when observe signal is a continuous clock.
15 RESET	Reset state 0b - Observe divider is not in reset state 1b - Observe divider is in reset state
14 —	Reserved
13 INV	Invert Invert input signal phase. 0b - Clock phase remain same. 1b - Invert clock phase before measurement or send to IO.

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 RAW	Observe raw signal Select from raw signal and divided signal. 0b - Select divided signal. 1b - Select raw signal.
11-9 —	Reserved
8-0 SELECT	Observe signal selector Selector that controls which observe signals will be selected and observed.

20.6.1.7 Observe access control (OBSERVE0_AUTHEN - OBSERVE1_AUTHEN)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

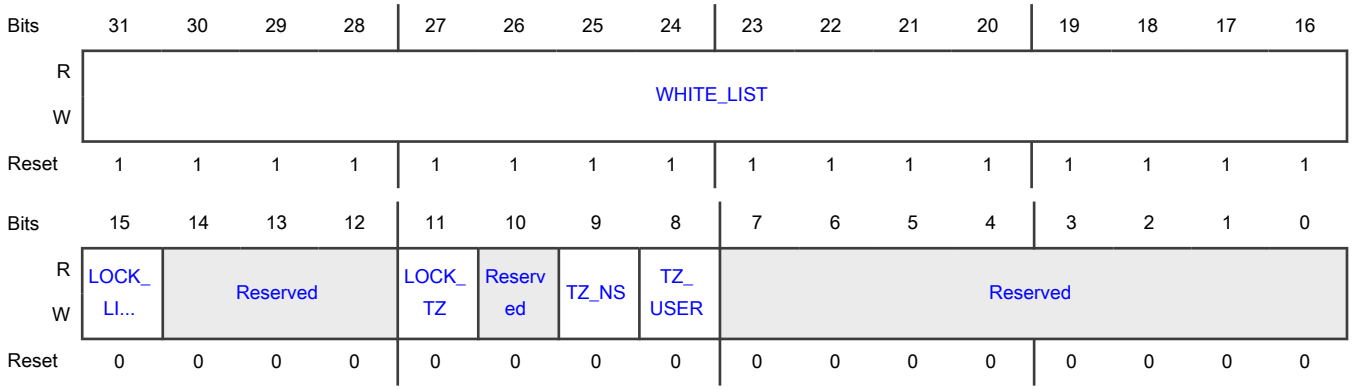
For a = 0 to 1:

Register	Offset	Description
OBSERVEa_AUTHEN	4430h + (a × 80h)	Observe access control
OBSERVEa_AUTHEN_SET	4434h + (a × 80h)	Writing 1 to a bit in this register ensures that the corresponding bit in OBSERVEa_AUTHEN is 1
OBSERVEa_AUTHEN_CLR	4438h + (a × 80h)	Writing 1 to a bit in this register ensures that the corresponding bit in OBSERVEa_AUTHEN is 0
OBSERVEa_AUTHEN_TOG	443Ch + (a × 80h)	Writing 1 to a bit in this register inverts the value of the corresponding bit in OBSERVEa_AUTHEN

Function

This register controls the access to this observe slice. The access control includes 2 parts: TrustZone control and domain based granting.

Diagram



Fields

Field	Function
31-16 WHITE_LIST	Whitelist settings Each bit in this field represent for one domain. Bit16~Bit31 represent for DOMAIN0~DOMAIN15 respectively. Only the domains the corresponding bit of which is set to 1 can change the registers of this observe slice. 0000_0000_0000_0000b - No domain can change. 0000_0000_0000_0001b - Domain 0 can change. 0000_0000_0000_0010b - Domain 1 can change. 0000_0000_0000_0011b - Domain 0 and domain 1 can change. 0000_0000_0000_0100b - Domain 2 can change. 0000_0000_0000_1111b - All domain can change.
15 LOCK_LIST	Lock white list This bit locks white list of this observe slice. When this bit is set, OBSERVEn_AUTHEN[WHITE_LIST] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset. 0b - White list is not locked. 1b - White list is locked.
14-12 —	Reserved
11 LOCK_TZ	Lock TrustZone setting This bit locks TrustZone settings. When this bit is set, OBSERVEn_AUTHEN[TZ_USER] and OBSERVEn_AUTHEN[TZ_NS] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset. 0b - TrustZone settings is not locked. 1b - TrustZone settings is locked.
10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 TZ_NS	Non-secure access permission Whether the registers of this observe slice can be changed when CPU is in non-secure mode. The default value is 0, that means only the access from CPU in secure mode can change the registers of this observe slice. 0b - Cannot be changed in non-secure mode. 1b - Can be changed in non-secure mode.
8 TZ_USER	User access permission Whether the registers of this observe slice can be changed when CPU is in user mode. The default value is 0, that means only the access from CPU in supervisor mode can change the registers of this observe slice. 0b - Observe slice settings cannot be changed in user mode. 1b - Observe slice settings can be changed in user mode.
7-0 —	Reserved

20.6.1.8 Current frequency detected (OBSERVE0_FREQUENCY_CURRENT - OBSERVE1_FREQUENCY_CURRENT)

Offset

Register	Offset
OBSERVE0_FREQUENCY_CURRENT	4440h
OBSERVE1_FREQUENCY_CURRENT	44C0h

Function

This register shows the current frequency detected since last update or reset.

NOTE

Frequency related measurment requires 32K crystal clock, and the gauged signal is generally a clock.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FREQUENCY															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FREQUENCY															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 FREQUENCY	Frequency Frequency of observed signal in Hz.

20.6.1.9 Minimum frequency detected (OBSERVE0_FREQUENCY_MIN - OBSERVE1_FREQUENCY_MIN)

Offset

Register	Offset
OBSERVE0_FREQUENCY_MIN	4444h
OBSERVE1_FREQUENCY_MIN	44C4h

Function

This register shows the minimum frequency detected since last update or reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FREQUENCY															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FREQUENCY															
W																
Reset	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0

Fields

Field	Function
31-0 FREQUENCY	Frequency Frequency of observed signal in Hz.

20.6.1.10 Maximum frequency detected (OBSERVE0_FREQUENCY_MAX - OBSERVE1_FREQUENCY_MAX)

Offset

Register	Offset
OBSERVE0_FREQUENCY_MAX	4448h
OBSERVE1_FREQUENCY_MAX	44C8h

Function

This register shows the maximum frequency detected since last update or reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FREQUENCY															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FREQUENCY															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 FREQUENCY	Frequency Frequency of observed signal in Hz.

20.6.1.11 Current period time detected (OBSERVE0_PERIOD_CURRENT - OBSERVE1_PERIOD_CURRENT)

Offset

Register	Offset
OBSERVE0_PERIOD_CURRENT	4450h
OBSERVE1_PERIOD_CURRENT	44D0h

Function

This register shows the current period detected since last update or reset.

NOTE

Period related measurment requires 1G Hz PLL clock, and the measured signal is treated as a pulse.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERIOD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PERIOD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PERIOD	Period time Period time of observed signal in ns.

20.6.1.12 Minimum period time detected (OBSERVE0_PERIOD_MIN - OBSERVE1_PERIOD_MIN)

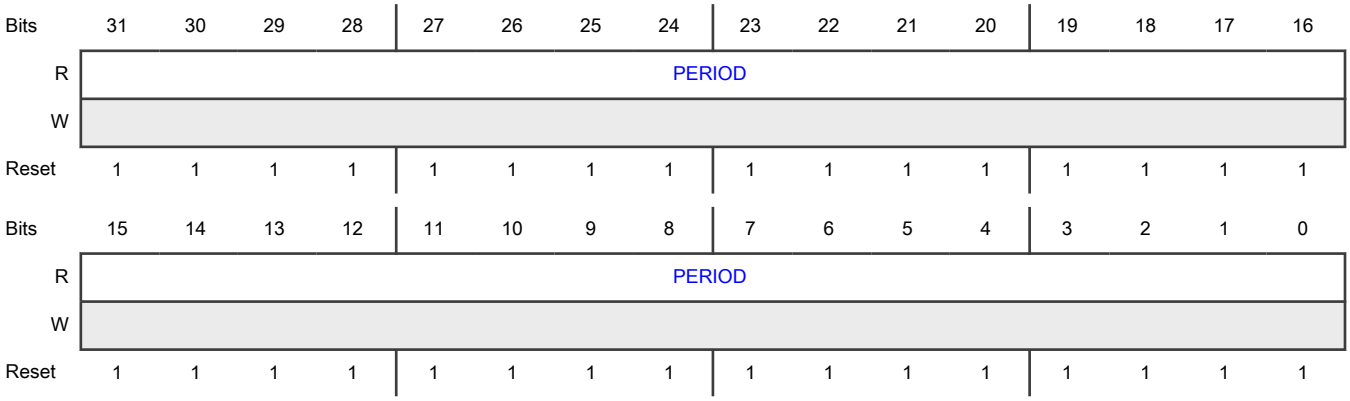
Offset

Register	Offset
OBSERVE0_PERIOD_MIN	4454h
OBSERVE1_PERIOD_MIN	44D4h

Function

This register shows the minimum period time detected since last update or reset.

Diagram



Fields

Field	Function
31-0 PERIOD	Period time Period time of observed signal in ns.

20.6.1.13 Maximum period time detected (OBSERVE0_PERIOD_MAX - OBSERVE1_PERIOD_MAX)

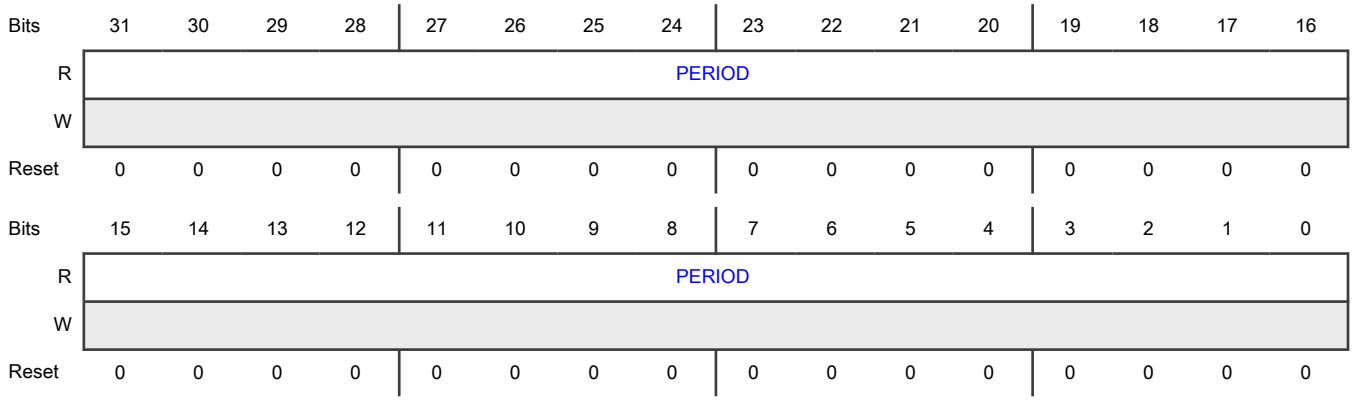
Offset

Register	Offset
OBSERVE0_PERIOD_MAX	4458h
OBSERVE1_PERIOD_MAX	44D8h

Function

This register shows the maximum period time detected since last update or reset.

Diagram



Fields

Field	Function
31-0 PERIOD	Period time Period time of observed signal in ns.

20.6.1.14 Current high level time detected (OBSERVE0_HIGH_CURRENT - OBSERVE1_HIGH_CURRENT)

Offset

Register	Offset
OBSERVE0_HIGH_CURRENT	4460h
OBSERVE1_HIGH_CURRENT	44E0h

Function

This register shows the current high level time detected since last update or reset.

NOTE

Period related measurment requires 1G Hz PLL clock, and the measured signal is treated as a pulse.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 HIGH	High level time High level time of observed signal in ns.

20.6.1.15 Minimum high level time detected (OBSERVE0_HIGH_MIN - OBSERVE1_HIGH_MIN)

Offset

Register	Offset
OBSERVE0_HIGH_MIN	4464h
OBSERVE1_HIGH_MIN	44E4h

Function

This register shows the minimum high level time detected since last update or reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HIGH															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HIGH															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 HIGH	High level time High level time of observed signal in ns.

20.6.1.16 Maximum high level time detected (OBSERVE0_HIGH_MAX - OBSERVE1_HIGH_MAX)

Offset

Register	Offset
OBSERVE0_HIGH_MAX	4468h
OBSERVE1_HIGH_MAX	44E8h

Function
This register shows the maximum high level time detected since last update or reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 HIGH	High level time High level time of observed signal in ns.

20.6.1.17 Current high level time detected (OBSERVE0_LOW_CURRENT - OBSERVE1_LOW_CURRENT)

Offset

Register	Offset
OBSERVE0_LOW_CURRENT	4470h
OBSERVE1_LOW_CURRENT	44F0h

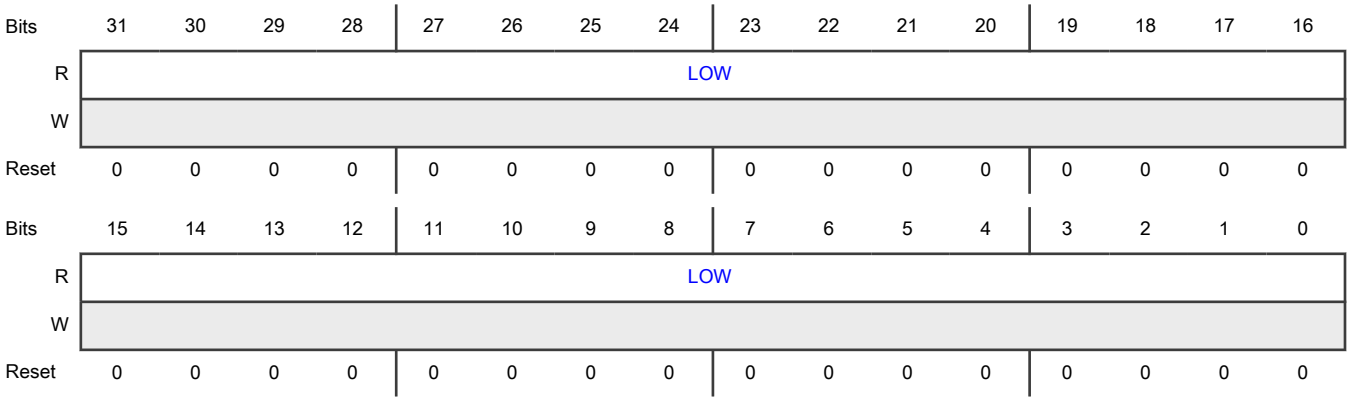
Function

This register shows the current low level time detected since last update or reset.

NOTE

Period related measurment requires 1G Hz PLL clock, and the measured signal is treated as a pulse.

Diagram



Fields

Field	Function
31-0 LOW	High level time Low level time of observed signal in ns.

20.6.1.18 Minimum high level time detected (OBSERVE0_LOW_MIN - OBSERVE1_LOW_MIN)

Offset

Register	Offset
OBSERVE0_LOW_MIN	4474h
OBSERVE1_LOW_MIN	44F4h

Function

This register shows the minimum low level time detected since lastest update or reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOW															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOW															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 LOW	High level time Low level time of observed signal in ns.

20.6.1.19 Maximum high level time detected (OBSERVE0_LOW_MAX - OBSERVE1_LOW_MAX)

Offset

Register	Offset
OBSERVE0_LOW_MAX	4478h
OBSERVE1_LOW_MAX	44F8h

Function

This register shows the maximum low level time detected since last update or reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 LOW	High level time Low level time of observed signal in ns.

20.6.1.20 General Purpose Register (GPR_SHARED0)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED0	4800h	General Purpose Register
GPR_SHARED0_SET	4804h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED0 is 1
GPR_SHARED0_CLR	4808h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED0 is 0
GPR_SHARED0_TOG	480Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED0

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

20.6.1.21 GPR access control (GPR_SHARED0_AUTHEN - GPR_SHARED15_AUTHEN)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

For a = 0 to 15:

Register	Offset	Description
GPR_SHAREDa_AUTHEN	4810h + (a × 20h)	GPR access control
GPR_SHAREDa_AUTHEN_SET	4814h + (a × 20h)	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHAREDa_AUTHEN is 1
GPR_SHAREDa_AUTHEN_CLR	4818h + (a × 20h)	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHAREDa_AUTHEN is 0
GPR_SHAREDa_AUTHEN_TOG	481Ch + (a × 20h)	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHAREDa_AUTHEN

Function

This register manages GPR access control. The access control includes 2 parts: TrustZone control and domain based granting.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WHITE_LIST															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LOCK_LI...	Reserved			LOCK_TZ	Reserved	TZ_NS	TZ_USER	Reserved	Reserved			Reserved	Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 WHITE_LIST	Whitelist settings Each bit in this field represent for one domain. Bit16~Bit31 represent for DOMAIN0~DOMAIN15 respectively. Only the domains the corresponding bit of which is set to 1 can change the registers of this shared GPR.
15 LOCK_LIST	Lock white list This bit lock whitelist. When this bit is set, GPR_SHAREDn_AUTHEN[WHITE_LIST] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Whitelist is not locked. 1b - Whitelist is locked.
14-12 —	Reserved
11 LOCK_TZ	Lock TrustZone settings This bit lock TrustZone settings. When this bit is set, GPR_SHAREDn_AUTHEN[TZ_USER] and GPR_SHAREDn_AUTHEN[TZ_NS] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset. 0b - TrustZone settings is not locked. 1b - TrustZone settings is locked.
10 —	Reserved
9 TZ_NS	Non-secure access permission Whether the registers of this shared GPR can be changed when CPU is in non-secure mode. The default value is 0, that means only the access from CPU in secure mode can change the registers of this shared GPR. 0b - Cannot be changed in Non-secure mode. 1b - Can be changed in Non-secure mode.
8 TZ_USER	User access permission Whether the registers of this shared GPR can be changed when CPU is in user mode. The default value is 0, that means only the access from CPU in supervisor mode can change the registers of this shared GPR. 0b - Registers of shared GPR slice cannot be changed in user mode. 1b - Registers of shared GPR slice can be changed in user mode.
7 —	Reserved
6-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

20.6.1.22 General Purpose Register (GPR_SHARED1)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED1	4820h	General Purpose Register
GPR_SHARED1_SET	4824h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED1 is 1
GPR_SHARED1_CLR	4828h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED1 is 0
GPR_SHARED1_TOG	482Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED1

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

20.6.1.23 General Purpose Register (GPR_SHARED2)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED2	4840h	General Purpose Register
GPR_SHARED2_SET	4844h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED2 is 1
GPR_SHARED2_CLR	4848h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED2 is 0
GPR_SHARED2_TOG	484Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED2

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	Reserved			m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 m33_mask_tpm5	m33_mask_tpm5 Mask low power control to tpm5 from M33 side
30 m33_mask_tpm4	m33_mask_tpm4 Mask low power control to tpm4 from M33 side
29 m33_mask_tpm3	m33_mask_tpm3 Mask low power control to tpm3 from M33 side
28	m33_mask_tpm2 Mask low power control to tpm2 from M33 side

Table continues on the next page...

Table continued from the previous page...

Field	Function
m33_mask_tpm2	
27 m33_mask_tpm1	m33_mask_tpm1 Mask low power control to tpm1 from M33 side
26 m33_mask_lpit3	m33_mask_lpit3 Mask low power control to lpit3 from M33 side
25 m33_mask_lpit2	m33_mask_lpit2 Mask low power control to lpit2 from M33 side
24 m33_mask_lpit1	m33_mask_lpit1 Mask low power control to lpit1 from M33 side
23 m33_mask_flexio2	m33_mask_flexio2 Mask low power control to flexio2 from M33 side
22 m33_mask_flexio1	m33_mask_flexio1 Mask low power control to flexio1 from M33 side
21 m33_mask_gpio6	m33_mask_gpio6 Mask low power control to gpio6 from M33 side
20 m33_mask_gpio5	m33_mask_gpio5 Mask low power control to gpio5 from M33 side
19 m33_mask_gpio4	m33_mask_gpio4 Mask low power control to gpio4 from M33 side
18 m33_mask_gpio3	m33_mask_gpio3 Mask low power control to gpio3 from M33 side
17 m33_mask_gpio2	m33_mask_gpio2 Mask low power control to gpio2 from M33 side

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 m33_mask_gpio1	m33_mask_gpio1 Mask low power control to gpio1 from M33 side
15 m33_mask_iee	m33_mask_iee Mask low power control to iee from M33 side
14 m33_mask_semc	m33_mask_semc Mask low power control to semc from M33 side
13 m33_mask_trdc	m33_mask_trdc Mask low power control to trdc from M33 side
12 m33_mask_flexspi2	m33_mask_flexspi2 Mask low power control to flexspi2 from M33 side
11 m33_mask_flexspi1	m33_mask_flexspi1 Mask low power control to flexspi1 from M33 side
10 m33_mask_adc2	m33_mask_adc2 Mask low power control to adc2 from M33 side
9 m33_mask_adc1	m33_mask_adc1 Mask low power control to adc1 from M33 side
8 m33_mask_sim_aon	m33_mask_sim_aon Mask low power control to sim_aon from M33 side
7-5 —	Reserved
4 m33_mask_netc	m33_mask_netc Mask low power control to netc from M33 side
3 m33_mask_edma4	m33_mask_edma4 Mask low power control to edma4 from M33 side

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 m33_mask_edma3	m33_mask_edma3 Mask low power control to edma3 from M33 side
1 m33_mask_cm33	m33_mask_cm33 Mask low power control to cm33 from M33 side
0 m33_mask_cm7	m33_mask_cm7 Mask low power control to cm7 from M33 side.

20.6.1.24 General Purpose Register (GPR_SHARED3)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED3	4860h	General Purpose Register
GPR_SHARED3_SET	4864h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED3 is 1
GPR_SHARED3_CLR	4868h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED3 is 0
GPR_SHARED3_TOG	486Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED3

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...	m33_mas...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 m33_mask_sinc2	m33_mask_sinc2 Mask low power control to sinc2 from M33 side
30 m33_mask_sinc1	m33_mask_sinc1 Mask low power control to sinc1 from M33 side
29 m33_mask_lpspi6	m33_mask_lpspi6 Mask low power control to lpspi6 from M33 side
28 m33_mask_lpspi5	m33_mask_lpspi5 Mask low power control to lpspi5 from M33 side
27 m33_mask_lpspi4	m33_mask_lpspi4 Mask low power control to lpspi4 from M33 side
26 m33_mask_lpspi3	m33_mask_lpspi3 Mask low power control to lpspi3 from M33 side
25 m33_mask_lpspi2	m33_mask_lpspi2 Mask low power control to lpspi2 from M33 side
24 m33_mask_lpspi1	m33_mask_lpspi1 Mask low power control to lpspi1 from M33 side
23 m33_mask_lpi2c6	m33_mask_lpi2c6 Mask low power control to lpi2c6 from M33 side
22 m33_mask_lpi2c5	m33_mask_lpi2c5 Mask low power control to lpi2c5 from M33 side
21 m33_mask_lpi2c4	m33_mask_lpi2c4 Mask low power control to lpi2c4 from M33 side

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 m33_mask_lpi2c3	m33_mask_lpi2c3 Mask low power control to lpi2c3 from M33 side
19 m33_mask_lpi2c2	m33_mask_lpi2c2 Mask low power control to lpi2c2 from M33 side
18 m33_mask_lpi2c1	m33_mask_lpi2c1 Mask low power control to lpi2c1 from M33 side
17 m33_mask_lpuart12	m33_mask_lpuart12 Mask low power control to lpuart12 from M33 side
16 m33_mask_lpuart11	m33_mask_lpuart11 Mask low power control to lpuart11 from M33 side
15 m33_mask_lpuart10	m33_mask_lpuart10 Mask low power control to lpuart10 from M33 side
14 m33_mask_lpuart9	m33_mask_lpuart9 Mask low power control to lpuart9 from M33 side
13 m33_mask_lpuart8	m33_mask_lpuart8 Mask low power control to lpuart8 from M33 side
12 m33_mask_lpuart7	m33_mask_lpuart7 Mask low power control to lpuart7 from M33 side
11 m33_mask_lpuart6	m33_mask_lpuart6 Mask low power control to lpuart6 from M33 side
10 m33_mask_lpuart5	m33_mask_lpuart5 Mask low power control to lpuart5 from M33 side

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 m33_mask_lpuart4	m33_mask_lpuart4 Mask low power control to lpuart4 from M33 side
8 m33_mask_lpuart3	m33_mask_lpuart3 Mask low power control to lpuart3 from M33 side
7 m33_mask_lpuart2	m33_mask_lpuart2 Mask low power control to lpuart2 from M33 side
6 m33_mask_lpuart1	m33_mask_lpuart1 Mask low power control to lpuart1 from M33 side
5 m33_mask_can3	m33_mask_can3 Mask low power control to can3 from M33 side
4 m33_mask_can2	m33_mask_can2 Mask low power control to can2 from M33 side
3 m33_mask_can1	m33_mask_can1 Mask low power control to can1 from M33 side
2 m33_mask_gpt2	m33_mask_gpt2 Mask low power control to gpt2 from M33 side
1 m33_mask_gpt1	m33_mask_gpt1 Mask low power control to gpt1 from M33 side
0 m33_mask_tpm6	m33_mask_tpm6 Mask low power control to tpm6 from M33 side

20.6.1.25 General Purpose Register (GPR_SHARED4)

Offset

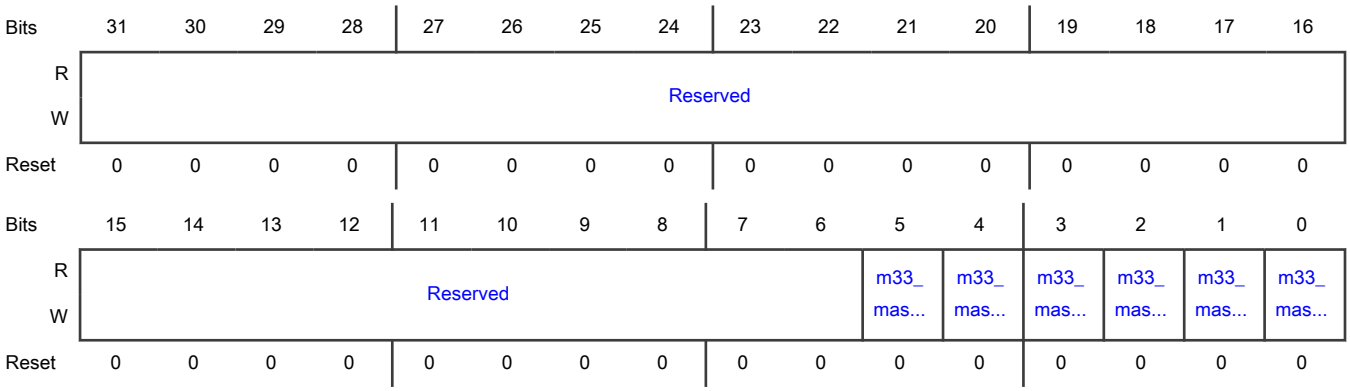
This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED4	4880h	General Purpose Register
GPR_SHARED4_SET	4884h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED4 is 1
GPR_SHARED4_CLR	4888h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED4 is 0
GPR_SHARED4_TOG	488Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED4

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 m33_mask_mic	m33_mask_mic Mask low power control to mic from M33 side
4 m33_mask_sai4	m33_mask_sai4 Mask low power control to sai4 from M33 side
3 m33_mask_sai3	m33_mask_sai3 Mask low power control to sai3 from M33 side
2 m33_mask_sai2	m33_mask_sai2 Mask low power control to sai2 from M33 side

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 m33_mask_sai1	m33_mask_sai1 Mask low power control to sai1 from M33 side
0 m33_mask_sinc3	m33_mask_sinc3 Mask low power control to sinc3 from M33 side

20.6.1.26 General Purpose Register (GPR_SHARED5)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED5	48A0h	General Purpose Register
GPR_SHARED5_SET	48A4h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED5 is 1
GPR_SHARED5_CLR	48A8h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED5 is 0
GPR_SHARED5_TOG	48ACh	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED5

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	Reserved			m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 m7_mask_tpm5	m7_mask_tpm5 Mask low power control to tpm5 from M7 side
30 m7_mask_tpm4	m7_mask_tpm4 Mask low power control to tpm4 from M7 side
29 m7_mask_tpm3	m7_mask_tpm3 Mask low power control to tpm3 from M7 side
28 m7_mask_tpm2	m7_mask_tpm2 Mask low power control to tpm2 from M7 side
27 m7_mask_tpm1	m7_mask_tpm1 Mask low power control to tpm1 from M7 side
26 m7_mask_lpit3	m7_mask_lpit3 Mask low power control to lpit3 from M7 side
25 m7_mask_lpit2	m7_mask_lpit2 Mask low power control to lpit2 from M7 side
24 m7_mask_lpit1	m7_mask_lpit1 Mask low power control to lpit1 from M7 side
23 m7_mask_flexio2	m7_mask_flexio2 Mask low power control to flexio2 from M7 side
22 m7_mask_flexio1	m7_mask_flexio1 Mask low power control to flexio1 from M7 side
21 m7_mask_gpio6	m7_mask_gpio6 Mask low power control to gpio6 from M7 side
20 m7_mask_gpio5	m7_mask_gpio5 Mask low power control to gpio5 from M7 side
19 m7_mask_gpio4	m7_mask_gpio4 Mask low power control to gpio4 from M7 side
18 m7_mask_gpio3	m7_mask_gpio3 Mask low power control to gpio3 from M7 side

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 m7_mask_gpio2	m7_mask_gpio2 Mask low power control to gpio2 from M7 side
16 m7_mask_gpio1	m7_mask_gpio1 Mask low power control to gpio1 from M7 side
15 m7_mask_iee	m7_mask_iee Mask low power control to iee from M7 side
14 m7_mask_semc	m7_mask_semc Mask low power control to semc from M7 side
13 m7_mask_trdc	m7_mask_trdc Mask low power control to trdc from M7 side
12 m7_mask_flexspi2	m7_mask_flexspi2 Mask low power control to flexspi2 from M7 side
11 m7_mask_flexspi1	m7_mask_flexspi1 Mask low power control to flexspi1 from M7 side
10 m7_mask_adc2	m7_mask_adc2 Mask low power control to adc2 from M7 side
9 m7_mask_adc1	m7_mask_adc1 Mask low power control to adc1 from M7 side
8 m7_mask_sim_aon	m7_mask_sim_aon Mask low power control to sim_aon from M7 side
7-5 —	Reserved
4 m7_mask_netc	m7_mask_netc Mask low power control to netc from M7 side
3 m7_mask_edma4	m7_mask_edma4 Mask low power control to edma4 from M7 side

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 m7_mask_edma3	m7_mask_edma3 Mask low power control to edma3 from M7 side
1 m7_mask_cm33	m7_mask_cm33 Mask low power control to cm33 from M7 side
0 m7_mask_cm7	m7_mask_cm7 Mask low power control to cm7 from M7 side.

20.6.1.27 General Purpose Register (GPR_SHARED6)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED6	48C0h	General Purpose Register
GPR_SHARED6_SET	48C4h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED6 is 1
GPR_SHARED6_CLR	48C8h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED6 is 0
GPR_SHARED6_TOG	48CCh	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED6

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...	m7_mask...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 m7_mask_sinc2	m7_mask_sinc2 Mask low power control to sinc2 from M7 side
30 m7_mask_sinc1	m7_mask_sinc1 Mask low power control to sinc1 from M7 side
29 m7_mask_lpspi6	m7_mask_lpspi6 Mask low power control to lpspi6 from M7 side
28 m7_mask_lpspi5	m7_mask_lpspi5 Mask low power control to lpspi5 from M7 side
27 m7_mask_lpspi4	m7_mask_lpspi4 Mask low power control to lpspi4 from M7 side
26 m7_mask_lpspi3	m7_mask_lpspi3 Mask low power control to lpspi3 from M7 side
25 m7_mask_lpspi2	m7_mask_lpspi2 Mask low power control to lpspi2 from M7 side
24 m7_mask_lpspi1	m7_mask_lpspi1 Mask low power control to lpspi1 from M7 side
23 m7_mask_lpi2c6	m7_mask_lpi2c6 Mask low power control to lpi2c6 from M7 side
22 m7_mask_lpi2c5	m7_mask_lpi2c5 Mask low power control to lpi2c5 from M7 side
21 m7_mask_lpi2c4	m7_mask_lpi2c4 Mask low power control to lpi2c4 from M7 side
20	m7_mask_lpi2c3 Mask low power control to lpi2c3 from M7 side

Table continues on the next page...

Table continued from the previous page...

Field	Function
m7_mask_lpi2c3	
19 m7_mask_lpi2c2	m7_mask_lpi2c2 Mask low power control to lpi2c2 from M7 side
18 m7_mask_lpi2c1	m7_mask_lpi2c1 Mask low power control to lpi2c1 from M7 side
17 m7_mask_lpuart12	m7_mask_lpuart12 Mask low power control to lpuart12 from M7 side
16 m7_mask_lpuart11	m7_mask_lpuart11 Mask low power control to lpuart11 from M7 side
15 m7_mask_lpuart10	m7_mask_lpuart10 Mask low power control to lpuart10 from M7 side
14 m7_mask_lpuart9	m7_mask_lpuart9 Mask low power control to lpuart9 from M7 side
13 m7_mask_lpuart8	m7_mask_lpuart8 Mask low power control to lpuart8 from M7 side
12 m7_mask_lpuart7	m7_mask_lpuart7 Mask low power control to lpuart7 from M7 side
11 m7_mask_lpuart6	m7_mask_lpuart6 Mask low power control to lpuart6 from M7 side
10 m7_mask_lpuart5	m7_mask_lpuart5 Mask low power control to lpuart5 from M7 side
9	m7_mask_lpuart4

Table continues on the next page...

Table continued from the previous page...

Field	Function
m7_mask_lpuart4	Mask low power control to lpuart4 from M7 side
8 m7_mask_lpuart3	m7_mask_lpuart3 Mask low power control to lpuart3 from M7 side
7 m7_mask_lpuart2	m7_mask_lpuart2 Mask low power control to lpuart2 from M7 side
6 m7_mask_lpuart1	m7_mask_lpuart1 Mask low power control to lpuart1 from M7 side
5 m7_mask_can3	m7_mask_can3 Mask low power control to can3 from M7 side
4 m7_mask_can2	m7_mask_can2 Mask low power control to can2 from M7 side
3 m7_mask_can1	m7_mask_can1 Mask low power control to can1 from M7 side
2 m7_mask_gpt2	m7_mask_gpt2 Mask low power control to gpt2 from M7 side
1 m7_mask_gpt1	m7_mask_gpt1 Mask low power control to gpt1 from M7 side
0 m7_mask_tpm6	m7_mask_tpm6 Mask low power control to tpm6 from M7 side

20.6.1.28 General Purpose Register (GPR_SHARED7)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED7	48E0h	General Purpose Register
GPR_SHARED7_SET	48E4h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED7 is 1

Table continues on the next page...

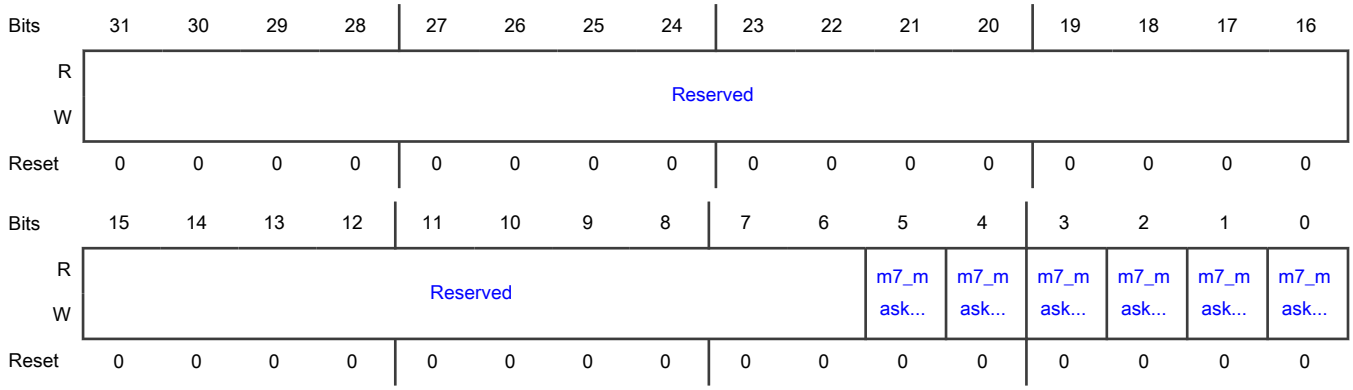
Table continued from the previous page...

Register	Offset	Description
GPR_SHARED7_CLR	48E8h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED7 is 0
GPR_SHARED7_TOG	48ECh	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED7

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 m7_mask_mic	m7_mask_mic Mask low power control to mic from M7 side
4 m7_mask_sai4	m7_mask_sai4 Mask low power control to sai4 from M7 side
3 m7_mask_sai3	m7_mask_sai3 Mask low power control to sai3 from M7 side
2 m7_mask_sai2	m7_mask_sai2 Mask low power control to sai2 from M7 side
1	m7_mask_sai1

Table continues on the next page...

Table continued from the previous page...

Field	Function
m7_mask_sai1	Mask low power control to sai1 from M7 side
0 m7_mask_sinc3	m7_mask_sinc3 Mask low power control to sinc3 from M7 side

20.6.1.29 General Purpose Register (GPR_SHARED8)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED8	4900h	General Purpose Register
GPR_SHARED8_SET	4904h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED8 is 1
GPR_SHARED8_CLR	4908h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED8 is 0
GPR_SHARED8_TOG	490Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED8

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_can...	m33_can...	m33_can...	m33_fle...	m33_fle...	m33_gpi...	m33_gpi...	m33_gpi...	m33_gpi...	m33_gpi...	m33_gpi...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	m33_ieee...	m33_sem...	m33_trd...	m33_fle...	m33_fle...	m33_adc...	m33_adc...	m33_sim...	Reserved			m33_net...	m33_edm...	m33_edm...	m33_cm3...	m33_cm7...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 m33_lpuart5_ipg_stop	m33_lpuart5_ipg_stop M33 request lpuart5 to ipg_stop mode, can be masked by m33_lpuart5_mask
30 m33_lpuart4_ipg_stop	m33_lpuart4_ipg_stop M33 request lpuart4 to ipg_stop mode, can be masked by m33_lpuart4_mask
29 m33_lpuart3_ipg_stop	m33_lpuart3_ipg_stop M33 request lpuart3 to ipg_stop mode, can be masked by m33_lpuart3_mask
28 m33_lpuart2_ipg_stop	m33_lpuart2_ipg_stop M33 request lpuart2 to ipg_stop mode, can be masked by m33_lpuart2_mask
27 m33_lpuart1_ipg_stop	m33_lpuart1_ipg_stop M33 request lpuart1 to ipg_stop mode, can be masked by m33_lpuart1_mask
26 m33_can3_ipg_stop	m33_can3_ipg_stop M33 request can3 to ipg_stop mode, can be masked by m33_can3_mask
25 m33_can2_ipg_stop	m33_can2_ipg_stop M33 request can2 to ipg_stop mode, can be masked by m33_can2_mask
24 m33_can1_ipg_stop	m33_can1_ipg_stop M33 request can1 to ipg_stop mode, can be masked by m33_can1_mask
23 m33_flexio2_ipg_stop	m33_flexio2_ipg_stop M33 request flexio2 to ipg_stop mode, can be masked by m33_flexio2_mask
22 m33_flexio1_ipg_stop	m33_flexio1_ipg_stop M33 request flexio1 to ipg_stop mode, can be masked by m33_flexio1_mask
21 m33_gpio6_ipg_stop	m33_gpio6_ipg_stop M33 request gpio6 to ipg_stop mode, can be masked by m33_gpio6_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 m33_gpio5_ipg_stop	m33_gpio5_ipg_stop M33 request gpio5 to ipg_stop mode, can be masked by m33_gpio5_mask
19 m33_gpio4_ipg_stop	m33_gpio4_ipg_stop M33 request gpio4 to ipg_stop mode, can be masked by m33_gpio4_mask
18 m33_gpio3_ipg_stop	m33_gpio3_ipg_stop M33 request gpio3 to ipg_stop mode, can be masked by m33_gpio3_mask
17 m33_gpio2_ipg_stop	m33_gpio2_ipg_stop M33 request gpio2 to ipg_stop mode, can be masked by m33_gpio2_mask
16 m33_gpio1_ipg_stop	m33_gpio1_ipg_stop M33 request gpio1 to ipg_stop mode, can be masked by m33_gpio1_mask
15 m33_iee_ipg_stop	m33_iee_ipg_stop M33 request iee to ipg_stop mode, can be masked by m33_iee_mask
14 m33_semc_ipg_stop	m33_semc_ipg_stop M33 request semc to ipg_stop mode, can be masked by m33_semc_mask
13 m33_trdc_ipg_stop	m33_trdc_ipg_stop M33 request trdc to ipg_stop mode, can be masked by m33_trdc_mask
12 m33_flexspi2_ipg_stop	m33_flexspi2_ipg_stop M33 request flexspi2 to ipg_stop mode, can be masked by m33_flexspi2_mask
11 m33_flexspi1_ipg_stop	m33_flexspi1_ipg_stop M33 request flexspi1 to ipg_stop mode, can be masked by m33_flexspi1_mask
10 m33_adc2_ipg_stop	m33_adc2_ipg_stop M33 request adc2 to ipg_stop mode, can be masked by m33_adc2_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 m33_adc1_ipg_stop	m33_adc1_ipg_stop M33 request adc1 to ipg_stop mode, can be masked by m33_adc1_mask
8 m33_sim_aon_ipg_stop	m33_sim_aon_ipg_stop M33 request sim_aon to ipg_stop mode, can be masked by m33_sim_aon_mask
7-5 —	Reserved
4 m33_netc_ipg_stop	m33_netc_ipg_stop M33 request netc to ipg_stop mode, can be masked by m33_netc_mask
3 m33_edma4_ipg_stop	m33_edma4_ipg_stop M33 request edma4 to ipg_stop mode, can be masked by m33_edma4_mask
2 m33_edma3_ipg_stop	m33_edma3_ipg_stop M33 request edma3 to ipg_stop mode, can be masked by m33_edma3_mask
1 m33_cm33_ipg_stop	m33_cm33_ipg_stop M33 request cm33 to ipg_stop mode, can be masked by m33_cm33_mask
0 m33_cm7_ipg_stop	m33_cm7_ipg_stop M33 request cm7 to ipg_stop mode, can be masked by m33_cm7_mask

20.6.1.30 General Purpose Register (GPR_SHARED9)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED9	4920h	General Purpose Register
GPR_SHARED9_SET	4924h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED9 is 1

Table continues on the next page...

Table continued from the previous page...

Register	Offset	Description
GPR_SHARED9_CLR	4928h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED9 is 0
GPR_SHARED9_TOG	492Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED9

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					m33_mic...	m33_sai...	m33_sai...	m33_sai...	m33_sai...	m33_sin...	m33_sin...	m33_sin...	m33_lps...	m33_lps...	m33_lps...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	m33_lps...	m33_lps...	m33_lps...	m33_lpi...	m33_lpi...	m33_lpi...	m33_lpi...	m33_lpi...	m33_lpi...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-27 —	Reserved
26 m33_mic_ipg_stop	m33_mic_ipg_stop M33 request mic to ipg_stop mode, can be masked by m33_mic_mask
25 m33_sai4_ipg_stop	m33_sai4_ipg_stop M33 request sai4 to ipg_stop mode, can be masked by m33_sai4_mask
24 m33_sai3_ipg_stop	m33_sai3_ipg_stop M33 request sai3 to ipg_stop mode, can be masked by m33_sai3_mask
23	m33_sai2_ipg_stop M33 request sai2 to ipg_stop mode, can be masked by m33_sai2_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
m33_sai2_ipg_stop	
22 m33_sai1_ipg_stop	m33_sai1_ipg_stop M33 request sai1 to ipg_stop mode, can be masked by m33_sai1_mask
21 m33_sinc3_ipg_stop	m33_sinc3_ipg_stop M33 request sinc3 to ipg_stop mode, can be masked by m33_sinc3_mask
20 m33_sinc2_ipg_stop	m33_sinc2_ipg_stop M33 request sinc2 to ipg_stop mode, can be masked by m33_sinc2_mask
19 m33_sinc1_ipg_stop	m33_sinc1_ipg_stop M33 request sinc1 to ipg_stop mode, can be masked by m33_sinc1_mask
18 m33_lpspi6_ipg_stop	m33_lpspi6_ipg_stop M33 request lpspi6 to ipg_stop mode, can be masked by m33_lpspi6_mask
17 m33_lpspi5_ipg_stop	m33_lpspi5_ipg_stop M33 request lpspi5 to ipg_stop mode, can be masked by m33_lpspi5_mask
16 m33_lpspi4_ipg_stop	m33_lpspi4_ipg_stop M33 request lpspi4 to ipg_stop mode, can be masked by m33_lpspi4_mask
15 m33_lpspi3_ipg_stop	m33_lpspi3_ipg_stop M33 request lpspi3 to ipg_stop mode, can be masked by m33_lpspi3_mask
14 m33_lpspi2_ipg_stop	m33_lpspi2_ipg_stop M33 request lpspi2 to ipg_stop mode, can be masked by m33_lpspi2_mask
13 m33_lpspi1_ipg_stop	m33_lpspi1_ipg_stop M33 request lpspi1 to ipg_stop mode, can be masked by m33_lpspi1_mask
12	m33_lpi2c6_ipg_stop

Table continues on the next page...

Table continued from the previous page...

Field	Function
m33_lpi2c6_ipg_stop	M33 request lpi2c6 to ipg_stop mode, can be masked by m33_lpi2c6_mask
11 m33_lpi2c5_ipg_stop	m33_lpi2c5_ipg_stop M33 request lpi2c5 to ipg_stop mode, can be masked by m33_lpi2c5_mask
10 m33_lpi2c4_ipg_stop	m33_lpi2c4_ipg_stop M33 request lpi2c4 to ipg_stop mode, can be masked by m33_lpi2c4_mask
9 m33_lpi2c3_ipg_stop	m33_lpi2c3_ipg_stop M33 request lpi2c3 to ipg_stop mode, can be masked by m33_lpi2c3_mask
8 m33_lpi2c2_ipg_stop	m33_lpi2c2_ipg_stop M33 request lpi2c2 to ipg_stop mode, can be masked by m33_lpi2c2_mask
7 m33_lpi2c1_ipg_stop	m33_lpi2c1_ipg_stop M33 request lpi2c1 to ipg_stop mode, can be masked by m33_lpi2c1_mask
6 m33_lpuart12_ipg_stop	m33_lpuart12_ipg_stop M33 request lpuart12 to ipg_stop mode, can be masked by m33_lpuart12_mask
5 m33_lpuart11_ipg_stop	m33_lpuart11_ipg_stop M33 request lpuart11 to ipg_stop mode, can be masked by m33_lpuart11_mask
4 m33_lpuart10_ipg_stop	m33_lpuart10_ipg_stop M33 request lpuart10 to ipg_stop mode, can be masked by m33_lpuart10_mask
3 m33_lpuart9_ipg_stop	m33_lpuart9_ipg_stop M33 request lpuart9 to ipg_stop mode, can be masked by m33_lpuart9_mask
2 m33_lpuart8_ipg_stop	m33_lpuart8_ipg_stop M33 request lpuart8 to ipg_stop mode, can be masked by m33_lpuart8_mask
1	m33_lpuart7_ipg_stop

Table continues on the next page...

Table continued from the previous page...

Field	Function
m33_lpuart7_ipg_stop	M33 request lpuart7 to ipg_stop mode, can be masked by m33_lpuart7_mask
0 m33_lpuart6_ipg_stop	m33_lpuart6_ipg_stop M33 request lpuart6 to ipg_stop mode, can be masked by m33_lpuart6_mask

20.6.1.31 General Purpose Register (GPR_SHARED10)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED10	4940h	General Purpose Register
GPR_SHARED10_SET	4944h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED10 is 1
GPR_SHARED10_CLR	4948h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED10 is 0
GPR_SHARED10_TOG	494Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED10

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_lpu...	m33_can...	m33_can...	m33_can...	m33_gpt...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	m33_gpt...	m33_tpm...	m33_tpm...	m33_tpm...	m33_tpm...	m33_tpm...	m33_tpm...	m33_lpi...	m33_lpi...	m33_lpi...	m33_fle...	m33_fle...	m33_fle...	m33_fle...	m33_adc...	m33_adc...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 m33_lpuart12_ipg_doze	m33_lpuart12_ipg_doze M33 request lpuart12 to ipg_doze mode, can be masked by m33_lpuart12_mask
30 m33_lpuart11_ipg_doze	m33_lpuart11_ipg_doze M33 request lpuart11 to ipg_doze mode, can be masked by m33_lpuart11_mask
29 m33_lpuart10_ipg_doze	m33_lpuart10_ipg_doze M33 request lpuart10 to ipg_doze mode, can be masked by m33_lpuart10_mask
28 m33_lpuart9_ipg_doze	m33_lpuart9_ipg_doze M33 request lpuart9 to ipg_doze mode, can be masked by m33_lpuart9_mask
27 m33_lpuart8_ipg_doze	m33_lpuart8_ipg_doze M33 request lpuart8 to ipg_doze mode, can be masked by m33_lpuart8_mask
26 m33_lpuart7_ipg_doze	m33_lpuart7_ipg_doze M33 request lpuart7 to ipg_doze mode, can be masked by m33_lpuart7_mask
25 m33_lpuart6_ipg_doze	m33_lpuart6_ipg_doze M33 request lpuart6 to ipg_doze mode, can be masked by m33_lpuart6_mask
24 m33_lpuart5_ipg_doze	m33_lpuart5_ipg_doze M33 request lpuart5 to ipg_doze mode, can be masked by m33_lpuart5_mask
23 m33_lpuart4_ipg_doze	m33_lpuart4_ipg_doze M33 request lpuart4 to ipg_doze mode, can be masked by m33_lpuart4_mask
22 m33_lpuart3_ipg_doze	m33_lpuart3_ipg_doze M33 request lpuart3 to ipg_doze mode, can be masked by m33_lpuart3_mask
21 m33_lpuart2_ipg_doze	m33_lpuart2_ipg_doze M33 request lpuart2 to ipg_doze mode, can be masked by m33_lpuart2_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 m33_lpuart1_ipg_doze	m33_lpuart1_ipg_doze M33 request lpuart1 to ipg_doze mode, can be masked by m33_lpuart1_mask
19 m33_can3_ipg_doze	m33_can3_ipg_doze M33 request can3 to ipg_doze mode, can be masked by m33_can3_mask
18 m33_can2_ipg_doze	m33_can2_ipg_doze M33 request can2 to ipg_doze mode, can be masked by m33_can2_mask
17 m33_can1_ipg_doze	m33_can1_ipg_doze M33 request can1 to ipg_doze mode, can be masked by m33_can1_mask
16 m33_gpt2_ipg_doze	m33_gpt2_ipg_doze M33 request gpt2 to ipg_doze mode, can be masked by m33_gpt2_mask
15 m33_gpt1_ipg_doze	m33_gpt1_ipg_doze M33 request gpt1 to ipg_doze mode, can be masked by m33_gpt1_mask
14 m33_tpm6_ipg_doze	m33_tpm6_ipg_doze M33 request tpm6 to ipg_doze mode, can be masked by m33_tpm6_mask
13 m33_tpm5_ipg_doze	m33_tpm5_ipg_doze M33 request tpm5 to ipg_doze mode, can be masked by m33_tpm5_mask
12 m33_tpm4_ipg_doze	m33_tpm4_ipg_doze M33 request tpm4 to ipg_doze mode, can be masked by m33_tpm4_mask
11 m33_tpm3_ipg_doze	m33_tpm3_ipg_doze M33 request tpm3 to ipg_doze mode, can be masked by m33_tpm3_mask
10 m33_tpm2_ipg_doze	m33_tpm2_ipg_doze M33 request tpm2 to ipg_doze mode, can be masked by m33_tpm2_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 m33_tpm1_ipg_doze	m33_tpm1_ipg_doze M33 request tpm1 to ipg_doze mode, can be masked by m33_tpm1_mask
8 m33_lpit3_ipg_doze	m33_lpit3_ipg_doze M33 request lpit3 to ipg_doze mode, can be masked by m33_lpit3_mask
7 m33_lpit2_ipg_doze	m33_lpit2_ipg_doze M33 request lpit2 to ipg_doze mode, can be masked by m33_lpit2_mask
6 m33_lpit1_ipg_doze	m33_lpit1_ipg_doze M33 request lpit1 to ipg_doze mode, can be masked by m33_lpit1_mask
5 m33_flexio2_ipg_doze	m33_flexio2_ipg_doze M33 request flexio2 to ipg_doze mode, can be masked by m33_flexio2_mask
4 m33_flexio1_ipg_doze	m33_flexio1_ipg_doze M33 request flexio1 to ipg_doze mode, can be masked by m33_flexio1_mask
3 m33_flexspi2_ipg_doze	m33_flexspi2_ipg_doze M33 request flexspi2 to ipg_doze mode, can be masked by m33_flexspi2_mask
2 m33_flexspi1_ipg_doze	m33_flexspi1_ipg_doze M33 request flexspi1 to ipg_doze mode, can be masked by m33_flexspi1_mask
1 m33_adc2_ipg_doze	m33_adc2_ipg_doze M33 request adc2 to ipg_doze mode, can be masked by m33_adc2_mask
0 m33_adc1_ipg_doze	m33_adc1_ipg_doze M33 request adc1 to ipg_doze mode, can be masked by m33_adc1_mask

20.6.1.32 General Purpose Register (GPR_SHARED11)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED11	4960h	General Purpose Register
GPR_SHARED11_SET	4964h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED11 is 1
GPR_SHARED11_CLR	4968h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED11 is 0
GPR_SHARED11_TOG	496Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED11

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	m33_mic...	m33_sin...	m33_sin...	m33_sin...	m33_lps...	m33_lps...	m33_lps...	m33_lps...	m33_lps...	m33_lps...	m33_lpi...	m33_lpi...	m33_lpi...	m33_lpi...	m33_lpi...	m33_lpi...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 m33_mic_ipg_doze	m33_mic_ipg_doze M33 request mic to ipg_doze mode, can be masked by m33_mic_mask
14 m33_sinc3_ipg_doze	m33_sinc3_ipg_doze M33 request sinc3 to ipg_doze mode, can be masked by m33_sinc3_mask
13 m33_sinc2_ipg_doze	m33_sinc2_ipg_doze M33 request sinc2 to ipg_doze mode, can be masked by m33_sinc2_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 m33_sinc1_ipg_doze	m33_sinc1_ipg_doze M33 request sinc1 to ipg_doze mode, can be masked by m33_sinc1_mask
11 m33_lpspi6_ipg_doze	m33_lpspi6_ipg_doze M33 request lpspi6 to ipg_doze mode, can be masked by m33_lpspi6_mask
10 m33_lpspi5_ipg_doze	m33_lpspi5_ipg_doze M33 request lpspi5 to ipg_doze mode, can be masked by m33_lpspi5_mask
9 m33_lpspi4_ipg_doze	m33_lpspi4_ipg_doze M33 request lpspi4 to ipg_doze mode, can be masked by m33_lpspi4_mask
8 m33_lpspi3_ipg_doze	m33_lpspi3_ipg_doze M33 request lpspi3 to ipg_doze mode, can be masked by m33_lpspi3_mask
7 m33_lpspi2_ipg_doze	m33_lpspi2_ipg_doze M33 request lpspi2 to ipg_doze mode, can be masked by m33_lpspi2_mask
6 m33_lpspi1_ipg_doze	m33_lpspi1_ipg_doze M33 request lpspi1 to ipg_doze mode, can be masked by m33_lpspi1_mask
5 m33_lpi2c6_ipg_doze	m33_lpi2c6_ipg_doze M33 request lpi2c6 to ipg_doze mode, can be masked by m33_lpi2c6_mask
4 m33_lpi2c5_ipg_doze	m33_lpi2c5_ipg_doze M33 request lpi2c5 to ipg_doze mode, can be masked by m33_lpi2c5_mask
3 m33_lpi2c4_ipg_doze	m33_lpi2c4_ipg_doze M33 request lpi2c4 to ipg_doze mode, can be masked by m33_lpi2c4_mask
2 m33_lpi2c3_ipg_doze	m33_lpi2c3_ipg_doze M33 request lpi2c3 to ipg_doze mode, can be masked by m33_lpi2c3_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 m33_lpi2c2_ipg_doze	m33_lpi2c2_ipg_doze M33 request lpi2c2 to ipg_doze mode, can be masked by m33_lpi2c2_mask
0 m33_lpi2c1_ipg_doze	m33_lpi2c1_ipg_doze M33 request lpi2c1 to ipg_doze mode, can be masked by m33_lpi2c1_mask

20.6.1.33 General Purpose Register (GPR_SHARED12)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED12	4980h	General Purpose Register
GPR_SHARED12_SET	4984h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED12 is 1
GPR_SHARED12_CLR	4988h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED12 is 0
GPR_SHARED12_TOG	498Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED12

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_can3...	m7_can2...	m7_can1...	m7_flex...	m7_flex...	m7_gpio...	m7_gpio...	m7_gpio...	m7_gpio...	m7_gpio...	m7_gpio...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	m7_iee_...	m7_semc...	m7_trdc...	m7_flex...	m7_flex...	m7_adc2...	m7_adc1...	m7_sim_...	Reserved			m7_netc...	m7_edma...	m7_edma...	m7_cm33...	m7_cm7_...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 m7_lpuart5_ipg_stop	m7_lpuart5_ipg_stop M7 request lpuart5 to ipg_stop mode, can be masked by m7_lpuart5_mask
30 m7_lpuart4_ipg_stop	m7_lpuart4_ipg_stop M7 request lpuart4 to ipg_stop mode, can be masked by m7_lpuart4_mask
29 m7_lpuart3_ipg_stop	m7_lpuart3_ipg_stop M7 request lpuart3 to ipg_stop mode, can be masked by m7_lpuart3_mask
28 m7_lpuart2_ipg_stop	m7_lpuart2_ipg_stop M7 request lpuart2 to ipg_stop mode, can be masked by m7_lpuart2_mask
27 m7_lpuart1_ipg_stop	m7_lpuart1_ipg_stop M7 request lpuart1 to ipg_stop mode, can be masked by m7_lpuart1_mask
26 m7_can3_ipg_stop	m7_can3_ipg_stop M7 request can3 to ipg_stop mode, can be masked by m7_can3_mask
25 m7_can2_ipg_stop	m7_can2_ipg_stop M7 request can2 to ipg_stop mode, can be masked by m7_can2_mask
24 m7_can1_ipg_stop	m7_can1_ipg_stop M7 request can1 to ipg_stop mode, can be masked by m7_can1_mask
23 m7_flexio2_ipg_stop	m7_flexio2_ipg_stop M7 request flexio2 to ipg_stop mode, can be masked by m7_flexio2_mask
22 m7_flexio1_ipg_stop	m7_flexio1_ipg_stop M7 request flexio1 to ipg_stop mode, can be masked by m7_flexio1_mask
21 m7_gpio6_ipg_stop	m7_gpio6_ipg_stop M7 request gpio6 to ipg_stop mode, can be masked by m7_gpio6_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 m7_gpio5_ipg_stop	m7_gpio5_ipg_stop M7 request gpio5 to ipg_stop mode, can be masked by m7_gpio5_mask
19 m7_gpio4_ipg_stop	m7_gpio4_ipg_stop M7 request gpio4 to ipg_stop mode, can be masked by m7_gpio4_mask
18 m7_gpio3_ipg_stop	m7_gpio3_ipg_stop M7 request gpio3 to ipg_stop mode, can be masked by m7_gpio3_mask
17 m7_gpio2_ipg_stop	m7_gpio2_ipg_stop M7 request gpio2 to ipg_stop mode, can be masked by m7_gpio2_mask
16 m7_gpio1_ipg_stop	m7_gpio1_ipg_stop M7 request gpio1 to ipg_stop mode, can be masked by m7_gpio1_mask
15 m7_iee_ipg_stop	m7_iee_ipg_stop M7 request iee to ipg_stop mode, can be masked by m7_iee_mask
14 m7_semc_ipg_stop	m7_semc_ipg_stop M7 request semc to ipg_stop mode, can be masked by m7_semc_mask
13 m7_trdc_ipg_stop	m7_trdc_ipg_stop M7 request trdc to ipg_stop mode, can be masked by m7_trdc_mask
12 m7_flexspi2_ipg_stop	m7_flexspi2_ipg_stop M7 request flexspi2 to ipg_stop mode, can be masked by m7_flexspi2_mask
11 m7_flexspi1_ipg_stop	m7_flexspi1_ipg_stop M7 request flexspi1 to ipg_stop mode, can be masked by m7_flexspi1_mask
10 m7_adc2_ipg_stop	m7_adc2_ipg_stop M7 request adc2 to ipg_stop mode, can be masked by m7_adc2_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 m7_adc1_ipg_stop	m7_adc1_ipg_stop M7 request adc1 to ipg_stop mode, can be masked by m7_adc1_mask
8 m7_sim_aon_ipg_stop	m7_sim_aon_ipg_stop M7 request sim_aon to ipg_stop mode, can be masked by m7_sim_aon_mask
7-5 —	Reserved
4 m7_netc_ipg_stop	m7_netc_ipg_stop M7 request netc to ipg_stop mode, can be masked by m7_netc_mask
3 m7_edma4_ipg_stop	m7_edma4_ipg_stop M7 request edma4 to ipg_stop mode, can be masked by m7_edma4_mask
2 m7_edma3_ipg_stop	m7_edma3_ipg_stop M7 request edma3 to ipg_stop mode, can be masked by m7_edma3_mask
1 m7_cm33_ipg_stop	m7_cm33_ipg_stop M7 request cm33 to ipg_stop mode, can be masked by m7_cm33_mask
0 m7_cm7_ipg_stop	m7_cm7_ipg_stop M7 request cm7 to ipg_stop mode, can be masked by m7_cm7_mask

20.6.1.34 General Purpose Register (GPR_SHARED13)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED13	49A0h	General Purpose Register
GPR_SHARED13_SET	49A4h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED13 is 1

Table continues on the next page...

Table continued from the previous page...

Register	Offset	Description
GPR_SHARED13_CLR	49A8h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED13 is 0
GPR_SHARED13_TOG	49ACh	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED13

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					m7_mic_...	m7_sai4...	m7_sai3...	m7_sai2...	m7_sai1...	m7_sinc...	m7_sinc...	m7_sinc...	m7_lpsp...	m7_lpsp...	m7_lpsp...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	m7_lpsp...	m7_lpsp...	m7_lpsp...	m7_lpi2...	m7_lpi2...	m7_lpi2...	m7_lpi2...	m7_lpi2...	m7_lpi2...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-27 —	Reserved
26 m7_mic_ipg_stop	m7_mic_ipg_stop M7 request mic to ipg_stop mode, can be masked by m7_mic_mask
25 m7_sai4_ipg_stop	m7_sai4_ipg_stop M7 request sai4 to ipg_stop mode, can be masked by m7_sai4_mask
24 m7_sai3_ipg_stop	m7_sai3_ipg_stop M7 request sai3 to ipg_stop mode, can be masked by m7_sai3_mask
23	m7_sai2_ipg_stop M7 request sai2 to ipg_stop mode, can be masked by m7_sai2_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
m7_sai2_ipg_stop	
22 m7_sai1_ipg_stop	m7_sai1_ipg_stop M7 request sai1 to ipg_stop mode, can be masked by m7_sai1_mask
21 m7_sinc3_ipg_stop	m7_sinc3_ipg_stop M7 request sinc3 to ipg_stop mode, can be masked by m7_sinc3_mask
20 m7_sinc2_ipg_stop	m7_sinc2_ipg_stop M7 request sinc2 to ipg_stop mode, can be masked by m7_sinc2_mask
19 m7_sinc1_ipg_stop	m7_sinc1_ipg_stop M7 request sinc1 to ipg_stop mode, can be masked by m7_sinc1_mask
18 m7_lpspi6_ipg_stop	m7_lpspi6_ipg_stop M7 request lpspi6 to ipg_stop mode, can be masked by m7_lpspi6_mask
17 m7_lpspi5_ipg_stop	m7_lpspi5_ipg_stop M7 request lpspi5 to ipg_stop mode, can be masked by m7_lpspi5_mask
16 m7_lpspi4_ipg_stop	m7_lpspi4_ipg_stop M7 request lpspi4 to ipg_stop mode, can be masked by m7_lpspi4_mask
15 m7_lpspi3_ipg_stop	m7_lpspi3_ipg_stop M7 request lpspi3 to ipg_stop mode, can be masked by m7_lpspi3_mask
14 m7_lpspi2_ipg_stop	m7_lpspi2_ipg_stop M7 request lpspi2 to ipg_stop mode, can be masked by m7_lpspi2_mask
13 m7_lpspi1_ipg_stop	m7_lpspi1_ipg_stop M7 request lpspi1 to ipg_stop mode, can be masked by m7_lpspi1_mask
12	m7_lpi2c6_ipg_stop

Table continues on the next page...

Table continued from the previous page...

Field	Function
m7_lpi2c6_ipg_stop	M7 request lpi2c6 to ipg_stop mode, can be masked by m7_lpi2c6_mask
11 m7_lpi2c5_ipg_stop	m7_lpi2c5_ipg_stop M7 request lpi2c5 to ipg_stop mode, can be masked by m7_lpi2c5_mask
10 m7_lpi2c4_ipg_stop	m7_lpi2c4_ipg_stop M7 request lpi2c4 to ipg_stop mode, can be masked by m7_lpi2c4_mask
9 m7_lpi2c3_ipg_stop	m7_lpi2c3_ipg_stop M7 request lpi2c3 to ipg_stop mode, can be masked by m7_lpi2c3_mask
8 m7_lpi2c2_ipg_stop	m7_lpi2c2_ipg_stop M7 request lpi2c2 to ipg_stop mode, can be masked by m7_lpi2c2_mask
7 m7_lpi2c1_ipg_stop	m7_lpi2c1_ipg_stop M7 request lpi2c1 to ipg_stop mode, can be masked by m7_lpi2c1_mask
6 m7_lpuart12_ipg_stop	m7_lpuart12_ipg_stop M7 request lpuart12 to ipg_stop mode, can be masked by m7_lpuart12_mask
5 m7_lpuart11_ipg_stop	m7_lpuart11_ipg_stop M7 request lpuart11 to ipg_stop mode, can be masked by m7_lpuart11_mask
4 m7_lpuart10_ipg_stop	m7_lpuart10_ipg_stop M7 request lpuart10 to ipg_stop mode, can be masked by m7_lpuart10_mask
3 m7_lpuart9_ipg_stop	m7_lpuart9_ipg_stop M7 request lpuart9 to ipg_stop mode, can be masked by m7_lpuart9_mask
2 m7_lpuart8_ipg_stop	m7_lpuart8_ipg_stop M7 request lpuart8 to ipg_stop mode, can be masked by m7_lpuart8_mask
1	m7_lpuart7_ipg_stop

Table continues on the next page...

Table continued from the previous page...

Field	Function
m7_lpuart7_ipg_stop	M7 request lpuart7 to ipg_stop mode, can be masked by m7_lpuart7_mask
0 m7_lpuart6_ipg_stop	m7_lpuart6_ipg_stop M7 request lpuart6 to ipg_stop mode, can be masked by m7_lpuart6_mask

20.6.1.35 General Purpose Register (GPR_SHARED14)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED14	49C0h	General Purpose Register
GPR_SHARED14_SET	49C4h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED14 is 1
GPR_SHARED14_CLR	49C8h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED14 is 0
GPR_SHARED14_TOG	49CCh	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED14

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_lpua...	m7_can3...	m7_can2...	m7_can1...	m7_gpt2...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	m7_gpt1...	m7_tpm6...	m7_tpm5...	m7_tpm4...	m7_tpm3...	m7_tpm2...	m7_tpm1...	m7_lpit...	m7_lpit...	m7_lpit...	m7_flex...	m7_flex...	m7_flex...	m7_flex...	m7_adc2...	m7_adc1...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 m7_lpuart12_ipg_doze	m7_lpuart12_ipg_doze M7 request lpuart12 to ipg_doze mode, can be masked by m7_lpuart12_mask
30 m7_lpuart11_ipg_doze	m7_lpuart11_ipg_doze M7 request lpuart11 to ipg_doze mode, can be masked by m7_lpuart11_mask
29 m7_lpuart10_ipg_doze	m7_lpuart10_ipg_doze M7 request lpuart10 to ipg_doze mode, can be masked by m7_lpuart10_mask
28 m7_lpuart9_ipg_doze	m7_lpuart9_ipg_doze M7 request lpuart9 to ipg_doze mode, can be masked by m7_lpuart9_mask
27 m7_lpuart8_ipg_doze	m7_lpuart8_ipg_doze M7 request lpuart8 to ipg_doze mode, can be masked by m7_lpuart8_mask
26 m7_lpuart7_ipg_doze	m7_lpuart7_ipg_doze M7 request lpuart7 to ipg_doze mode, can be masked by m7_lpuart7_mask
25 m7_lpuart6_ipg_doze	m7_lpuart6_ipg_doze M7 request lpuart6 to ipg_doze mode, can be masked by m7_lpuart6_mask
24 m7_lpuart5_ipg_doze	m7_lpuart5_ipg_doze M7 request lpuart5 to ipg_doze mode, can be masked by m7_lpuart5_mask
23 m7_lpuart4_ipg_doze	m7_lpuart4_ipg_doze M7 request lpuart4 to ipg_doze mode, can be masked by m7_lpuart4_mask
22 m7_lpuart3_ipg_doze	m7_lpuart3_ipg_doze M7 request lpuart3 to ipg_doze mode, can be masked by m7_lpuart3_mask
21 m7_lpuart2_ipg_doze	m7_lpuart2_ipg_doze M7 request lpuart2 to ipg_doze mode, can be masked by m7_lpuart2_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 m7_lpuart1_ipg_doze	m7_lpuart1_ipg_doze M7 request lpuart1 to ipg_doze mode, can be masked by m7_lpuart1_mask
19 m7_can3_ipg_doze	m7_can3_ipg_doze M7 request can3 to ipg_doze mode, can be masked by m7_can3_mask
18 m7_can2_ipg_doze	m7_can2_ipg_doze M7 request can2 to ipg_doze mode, can be masked by m7_can2_mask
17 m7_can1_ipg_doze	m7_can1_ipg_doze M7 request can1 to ipg_doze mode, can be masked by m7_can1_mask
16 m7_gpt2_ipg_doze	m7_gpt2_ipg_doze M7 request gpt2 to ipg_doze mode, can be masked by m7_gpt2_mask
15 m7_gpt1_ipg_doze	m7_gpt1_ipg_doze M7 request gpt1 to ipg_doze mode, can be masked by m7_gpt1_mask
14 m7_tpm6_ipg_doze	m7_tpm6_ipg_doze M7 request tpm6 to ipg_doze mode, can be masked by m7_tpm6_mask
13 m7_tpm5_ipg_doze	m7_tpm5_ipg_doze M7 request tpm5 to ipg_doze mode, can be masked by m7_tpm5_mask
12 m7_tpm4_ipg_doze	m7_tpm4_ipg_doze M7 request tpm4 to ipg_doze mode, can be masked by m7_tpm4_mask
11 m7_tpm3_ipg_doze	m7_tpm3_ipg_doze M7 request tpm3 to ipg_doze mode, can be masked by m7_tpm3_mask
10 m7_tpm2_ipg_doze	m7_tpm2_ipg_doze M7 request tpm2 to ipg_doze mode, can be masked by m7_tpm2_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 m7_tpm1_ipg_doze	m7_tpm1_ipg_doze M7 request tpm1 to ipg_doze mode, can be masked by m7_tpm1_mask
8 m7_lpit3_ipg_doze	m7_lpit3_ipg_doze M7 request lpit3 to ipg_doze mode, can be masked by m7_lpit3_mask
7 m7_lpit2_ipg_doze	m7_lpit2_ipg_doze M7 request lpit2 to ipg_doze mode, can be masked by m7_lpit2_mask
6 m7_lpit1_ipg_doze	m7_lpit1_ipg_doze M7 request lpit1 to ipg_doze mode, can be masked by m7_lpit1_mask
5 m7_flexio2_ipg_doze	m7_flexio2_ipg_doze M7 request flexio2 to ipg_doze mode, can be masked by m7_flexio2_mask
4 m7_flexio1_ipg_doze	m7_flexio1_ipg_doze M7 request flexio1 to ipg_doze mode, can be masked by m7_flexio1_mask
3 m7_flexspi2_ipg_doze	m7_flexspi2_ipg_doze M7 request flexspi2 to ipg_doze mode, can be masked by m7_flexspi2_mask
2 m7_flexspi1_ipg_doze	m7_flexspi1_ipg_doze M7 request flexspi1 to ipg_doze mode, can be masked by m7_flexspi1_mask
1 m7_adc2_ipg_doze	m7_adc2_ipg_doze M7 request adc2 to ipg_doze mode, can be masked by m7_adc2_mask
0 m7_adc1_ipg_doze	m7_adc1_ipg_doze M7 request adc1 to ipg_doze mode, can be masked by m7_adc1_mask

20.6.1.36 General Purpose Register (GPR_SHARED15)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
GPR_SHARED15	49E0h	General Purpose Register
GPR_SHARED15_SET	49E4h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED15 is 1
GPR_SHARED15_CLR	49E8h	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_SHARED15 is 0
GPR_SHARED15_TOG	49ECh	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_SHARED15

Function

These registers are shared general purpose registers. See General Purpose Registers (GPR) and GPR Shared Control Registers (GPR_SHAREDn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	m7_mic_...	m7_sinc...	m7_sinc...	m7_sinc...	m7_lpsp...	m7_lpsp...	m7_lpsp...	m7_lpsp...	m7_lpsp...	m7_lpsp...	m7_lpi2...	m7_lpi2...	m7_lpi2...	m7_lpi2...	m7_lpi2...	m7_lpi2...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 m7_mic_ipg_doze	m7_mic_ipg_doze M7 request mic to ipg_doze mode, can be masked by m7_mic_mask
14 m7_sinc3_ipg_doze	m7_sinc3_ipg_doze M7 request sinc3 to ipg_doze mode, can be masked by m7_sinc3_mask
13 m7_sinc2_ipg_doze	m7_sinc2_ipg_doze M7 request sinc2 to ipg_doze mode, can be masked by m7_sinc2_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 m7_sinc1_ipg_doze	m7_sinc1_ipg_doze M7 request sinc1 to ipg_doze mode, can be masked by m7_sinc1_mask
11 m7_lpspi6_ipg_doze	m7_lpspi6_ipg_doze M7 request lpspi6 to ipg_doze mode, can be masked by m7_lpspi6_mask
10 m7_lpspi5_ipg_doze	m7_lpspi5_ipg_doze M7 request lpspi5 to ipg_doze mode, can be masked by m7_lpspi5_mask
9 m7_lpspi4_ipg_doze	m7_lpspi4_ipg_doze M7 request lpspi4 to ipg_doze mode, can be masked by m7_lpspi4_mask
8 m7_lpspi3_ipg_doze	m7_lpspi3_ipg_doze M7 request lpspi3 to ipg_doze mode, can be masked by m7_lpspi3_mask
7 m7_lpspi2_ipg_doze	m7_lpspi2_ipg_doze M7 request lpspi2 to ipg_doze mode, can be masked by m7_lpspi2_mask
6 m7_lpspi1_ipg_doze	m7_lpspi1_ipg_doze M7 request lpspi1 to ipg_doze mode, can be masked by m7_lpspi1_mask
5 m7_lpi2c6_ipg_doze	m7_lpi2c6_ipg_doze M7 request lpi2c6 to ipg_doze mode, can be masked by m7_lpi2c6_mask
4 m7_lpi2c5_ipg_doze	m7_lpi2c5_ipg_doze M7 request lpi2c5 to ipg_doze mode, can be masked by m7_lpi2c5_mask
3 m7_lpi2c4_ipg_doze	m7_lpi2c4_ipg_doze M7 request lpi2c4 to ipg_doze mode, can be masked by m7_lpi2c4_mask
2 m7_lpi2c3_ipg_doze	m7_lpi2c3_ipg_doze M7 request lpi2c3 to ipg_doze mode, can be masked by m7_lpi2c3_mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 m7_lpi2c2_ipg_doze	m7_lpi2c2_ipg_doze M7 request lpi2c2 to ipg_doze mode, can be masked by m7_lpi2c2_mask
0 m7_lpi2c1_ipg_doze	m7_lpi2c1_ipg_doze M7 request lpi2c1 to ipg_doze mode, can be masked by m7_lpi2c1_mask

20.6.1.37 General purpose status register for CM33 (GPR_SHARED_STATUS0)

Offset

Register	Offset
GPR_SHARED_STATUS0	4A00h

Function

This register contains the acknowledge indicators for low power handshake. See General Purpose Registers (GPR) and GPR Shared Status Registers (GPR_SHARED_STATUSn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GPR_STATUS															
W																
Reset	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPR_STATUS															
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GPR_STATUS	Acknowledge indicators for low power handshake

20.6.1.38 General purpose status register for CM33 (GPR_SHARED_STATUS1)

Offset

Register	Offset
GPR_SHARED_STATUS1	4A04h

Function

This register contains the acknowledge indicators for low power handshake. See General Purpose Registers (GPR) and GPR Shared Status Registers (GPR_SHARED_STATUSn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GPR_STATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPR_STATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31-0 GPR_STATUS	Acknowledge indicators for low power handshake

20.6.1.39 General purpose status register for CM33 (GPR_SHARED_STATUS2)

Offset

Register	Offset
GPR_SHARED_STATUS2	4A08h

Function

This register contains the acknowledge indicators for low power handshake. See General Purpose Registers (GPR) and GPR Shared Status Registers (GPR_SHARED_STATUSn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GPR_STATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPR_STATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GPR_STATUS	Acknowledge indicators for low power handshake

20.6.1.40 General purpose status register for CM33 (GPR_SHARED_STATUS3)

Offset

Register	Offset
GPR_SHARED_STATUS3	4A0Ch

Function

This register contains the acknowledge indicators for low power handshake. See General Purpose Registers (GPR) and GPR Shared Status Registers (GPR_SHARED_STATUSn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GPR_STATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPR_STATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GPR_STATUS	Acknowledge indicators for low power handshake

20.6.1.41 General status register for CM7 (GPR_SHARED_STATUS4)

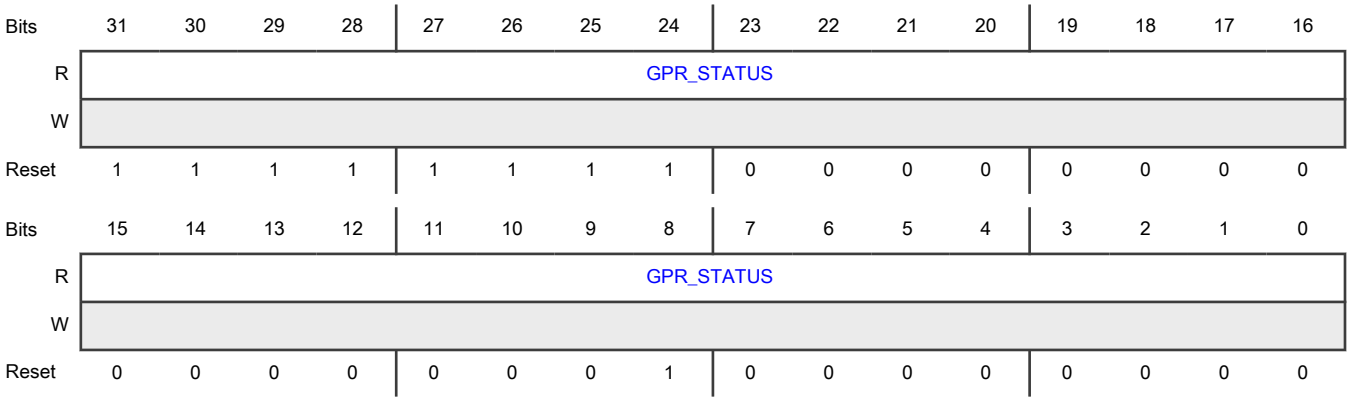
Offset

Register	Offset
GPR_SHARED_STATUS4	4A10h

Function

This register contains the acknowledge indicators for low power handshake. See General Purpose Registers (GPR) and GPR Shared Status Registers (GPR_SHARED_STATUSn) for more information.

Diagram



Fields

Field	Function
31-0 GPR_STATUS	Acknowledge indicators for low power handshake

20.6.1.42 General purpose status register for CM7 (GPR_SHARED_STATUS5)

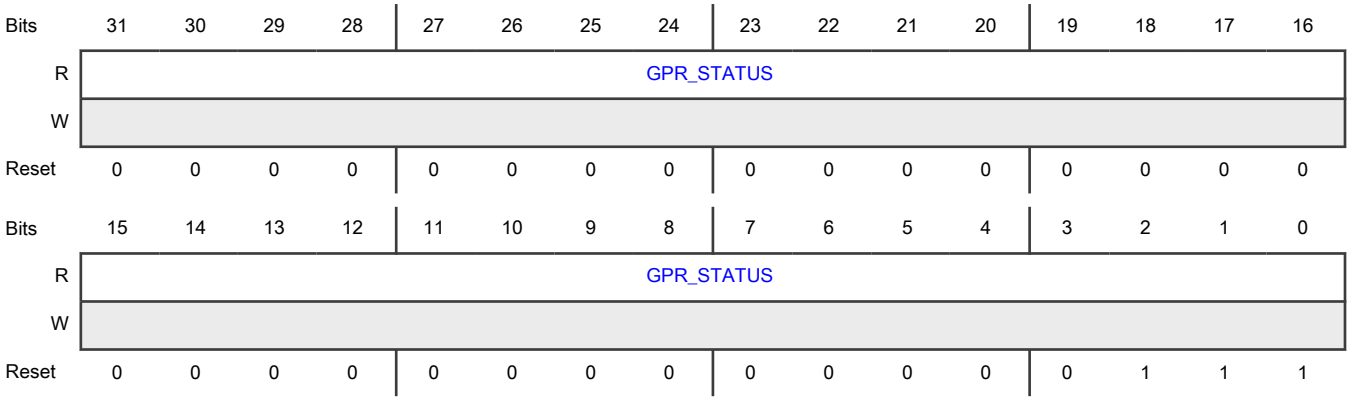
Offset

Register	Offset
GPR_SHARED_STATUS5	4A14h

Function

This register contains the acknowledge indicators for low power handshake. See General Purpose Registers (GPR) and GPR Shared Status Registers (GPR_SHARED_STATUSn) for more information.

Diagram



Fields

Field	Function
31-0 GPR_STATUS	Acknowledge indicators for low power handshake

20.6.1.43 General status register for CM7 (GPR_SHARED_STATUS6)

Offset

Register	Offset
GPR_SHARED_STATUS6	4A18h

Function

This register contains the acknowledge indicators for low power handshake. See General Purpose Registers (GPR) and GPR Shared Status Registers (GPR_SHARED_STATUSn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GPR_STATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPR_STATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GPR_STATUS	Acknowledge indicators for low power handshake

20.6.1.44 General purpose status register for CM7 (GPR_SHARED_STATUS7)

Offset

Register	Offset
GPR_SHARED_STATUS7	4A1Ch

Function

This register contains the acknowledge indicators for low power handshake. See General Purpose Registers (GPR) and GPR Shared Status Registers (GPR_SHARED_STATUSn) for more information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GPR_STATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPR_STATUS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GPR_STATUS	Acknowledge indicators for low power handshake

20.6.1.45 General purpose register (GPR_PRIVATE0 - GPR_PRIVATE3)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

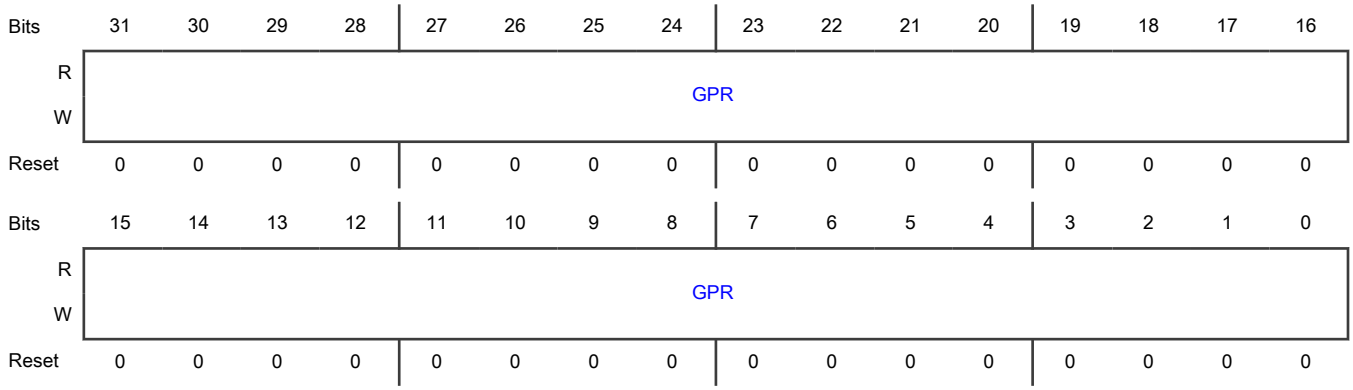
For a = 0 to 3:

Register	Offset	Description
GPR_PRIVATEa	4C00h + (a × 20h)	General purpose register
GPR_PRIVATEa_SET	4C04h + (a × 20h)	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_PRIVATEa is 1
GPR_PRIVATEa_CLR	4C08h + (a × 20h)	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_PRIVATEa is 0
GPR_PRIVATEa_TOG	4C0Ch + (a × 20h)	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_PRIVATEa

Function

These registers are general purpose registers.

Diagram



Fields

Field	Function
31-0 GPR	GP register General purpose register. This register has dedicated bits for each domain.

20.6.1.46 GPR access control (GPR_PRIVATE0_AUTHEN - GPR_PRIVATE3_AUTHEN)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

For a = 0 to 3:

Register	Offset	Description
GPR_PRIVATEa_AUTHEN	4C10h + (a × 20h)	GPR access control
GPR_PRIVATEa_AUTHEN_SET	4C14h + (a × 20h)	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_PRIVATEa_AUTHEN is 1
GPR_PRIVATEa_AUTHEN_CLR	4C18h + (a × 20h)	Writing 1 to a bit in this register ensures that the corresponding bit in GPR_PRIVATEa_AUTHEN is 0
GPR_PRIVATEa_AUTHEN_TOG	4C1Ch + (a × 20h)	Writing 1 to a bit in this register inverts the value of the corresponding bit in GPR_PRIVATEa_AUTHEN

Function

This register controls the access to this private GPR slice. The access control includes 2 parts: TrustZone control and domain based granting.

NOTE

The domain based granting only influence the access to GPR_PRIVATEn_AUTHEN.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WHITE_LIST															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LOCK_LI...	Reserved			LOCK_TZ	Reserved	TZ_NS	TZ_USER	Reserved	Reserved			Reserved	Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 WHITE_LIST	Whitelist settings Each bit in this field represent for one domain. Bit16~Bit31 represent for DOMAIN0~DOMAIN15 respectively. Only the domains the corresponding bit of which is set to 1 can change the registers of this private GPR.
15	Lock white list

Table continues on the next page...

Table continued from the previous page...

Field	Function
LOCK_LIST	This bit lock whitelist. When this bit is set, GPR_PRIVATEn_AUTHEN[WHITE_LIST] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset. 0b - Whitelist is not locked. 1b - Whitelist is locked.
14-12 —	Reserved
11 LOCK_TZ	Lock TrustZone settings This bit lock TrustZone settings. When this bit is set, GPR_PRIVATEn_AUTHEN[TZ_USER] and GPR_PRIVATEn_AUTHEN[TZ_NS] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset. 0b - TrustZone settings is not locked. 1b - TrustZone settings is locked.
10 —	Reserved
9 TZ_NS	Non-secure access permission Whether the registers of this private GPR can be changed when CPU is in non-secure mode. The default value is 0, that means only the access from CPU in secure mode can change the registers of this private GPR. 0b - Cannot be changed in Non-secure mode. 1b - Can be changed in Non-secure mode.
8 TZ_USER	User access permission Whether the registers of this private GPR can be changed when CPU is in user mode. The default value is 0, that means only the access from CPU in supervisor mode can change the registers of this private GPR. 0b - Registers of private GPR cannot be changed in user mode. 1b - Registers of private GPR can be changed in user mode.
7 —	Reserved
6-4 —	Reserved
3 —	Reserved
2-0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

20.6.1.47 Clock source direct control (OSCPLL0_DIRECT - OSCPLL24_DIRECT)

Offset

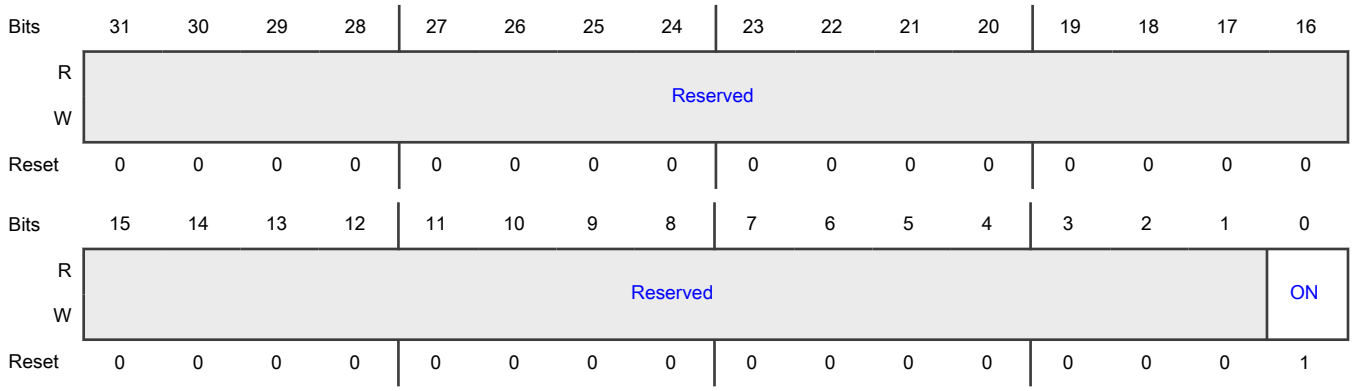
For a = 0 to 24:

Register	Offset
OSCPLLa_DIRECT	5000h + (a × 40h)

Function

This register controls the ON/OFF of clock source when clock source is working in Direct Control mode.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 ON	Turn on clock source This bit controls ON/OFF of clock source. 0b - Clock source is OFF. 1b - Clock source is ON.

20.6.1.48 Clock source low power mode setting (OSCPLL0_LPM0 - OSCPLL24_LPM0)

Offset

For a = 0 to 24:

Register	Offset
OSCPLLa_LPM0	5010h + (a × 40h)

Function

Clock source low power mode setting for DOMAIN0 ~ DOMAIM7.

NOTE

The settings in this register are valid only when this Clock Source slice is working in CPULPM mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	LPM_SETTING_D7			Reserved	LPM_SETTING_D6			Reserved	LPM_SETTING_D5			Reserved	LPM_SETTING_D4		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	LPM_SETTING_D3			Reserved	LPM_SETTING_D2			Reserved	LPM_SETTING_D1			Reserved	LPM_SETTING_D0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 LPM_SETTING_D7	Clock Source LPM in DOMAIN7 Clock Source low power mode setting in DOMAIN7. **NOTE** This field only can be changed by DOMAIN7. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
27	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
26-24 LPM_SETTING_D6	Clock Source LPM in DOMAIN6 Clock Source low power mode setting in DOMAIN6. **NOTE** This field only can be changed by DOMAIN6. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
23 —	Reserved
22-20 LPM_SETTING_D5	Clock Source LPM in DOMAIN5 Clock Source low power mode setting in DOMAIN5. **NOTE** This field only can be changed by DOMAIN5. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
19 —	Reserved
18-16 LPM_SETTING_D4	Clock Source LPM in DOMAIN4 Clock Source low power mode setting in DOMAIN4. **NOTE** This field only can be changed by DOMAIN4. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-12 LPM_SETTING_D3	Clock Source LPM in DOMAIN3 Clock Source low power mode setting in DOMAIN3. **NOTE** This field only can be changed by DOMAIN3. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
11 —	Reserved
10-8 LPM_SETTING_D2	Clock Source LPM in DOMAIN2 Clock Source low power mode setting in DOMAIN2. **NOTE** This field only can be changed by DOMAIN2. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
7 —	Reserved
6-4 LPM_SETTING_D1	Clock Source LPM in DOMAIN1 Clock Source low power mode setting in DOMAIN1. **NOTE** This field only can be changed by DOMAIN1. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
3 —	Reserved
2-0	Clock Source LPM in DOMAIN0

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPM_SETTING_D0	Clock Source low power mode setting in DOMAIN0. **NOTE** This field only can be changed by DOMAIN0. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.

20.6.1.49 clock source low power mode setting (OSCPLL0_LPM1 - OSCPLL24_LPM1)

Offset

For a = 0 to 24:

Register	Offset
OSCPLLa_LPM1	5014h + (a × 40h)

Function

Clock source low power mode setting for DOMAIN8 ~ DOMAIM15.

NOTE

The settings in this register are valid only when this Clock Source slice is working in CPULPM mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	LPM_SETTING_D15			Reserved	LPM_SETTING_D14			Reserved	LPM_SETTING_D13			Reserved	LPM_SETTING_D12		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	LPM_SETTING_D11			Reserved	LPM_SETTING_D10			Reserved	LPM_SETTING_D9			Reserved	LPM_SETTING_D8		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
30-28 LPM_SETTING_D15	Clock Source LPM in DOMAIN15 Clock Source low power mode setting in DOMAIN15. **NOTE** This field only can be changed by DOMAIN15. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
27 —	Reserved
26-24 LPM_SETTING_D14	Clock Source LPM in DOMAIN14 Clock Source low power mode setting in DOMAIN14. **NOTE** This field only can be changed by DOMAIN14. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
23 —	Reserved
22-20 LPM_SETTING_D13	Clock Source LPM in DOMAIN13 Clock Source low power mode setting in DOMAIN13. **NOTE** This field only can be changed by DOMAIN13. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 LPM_SETTING_D12	Clock Source LPM in DOMAIN12 Clock Source low power mode setting in DOMAIN12. **NOTE** This field only can be changed by DOMAIN12. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
15 —	Reserved
14-12 LPM_SETTING_D11	Clock Source LPM in DOMAIN11 Clock Source low power mode setting in DOMAIN11. **NOTE** This field only can be changed by DOMAIN11. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
11 —	Reserved
10-8 LPM_SETTING_D10	Clock Source LPM in DOMAIN10 Clock Source low power mode setting in DOMAIN10. **NOTE** This field only can be changed by DOMAIN10. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
7 —	Reserved
6-4	Clock Source LPM in DOMAIN9

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPM_SETTING_D9	Clock Source low power mode setting in DOMAIN9. **NOTE** This field only can be changed by DOMAIN9. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
3 —	Reserved
2-0 LPM_SETTING_D8	Clock Source LPM in DOMAIN8 Clock Source low power mode setting in DOMAIN8. **NOTE** This field only can be changed by DOMAIN8. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.

20.6.1.50 LPM setting of current CPU domain (OSCPLL0_LPM_CUR - OSCPLL24_LPM_CUR)

Offset

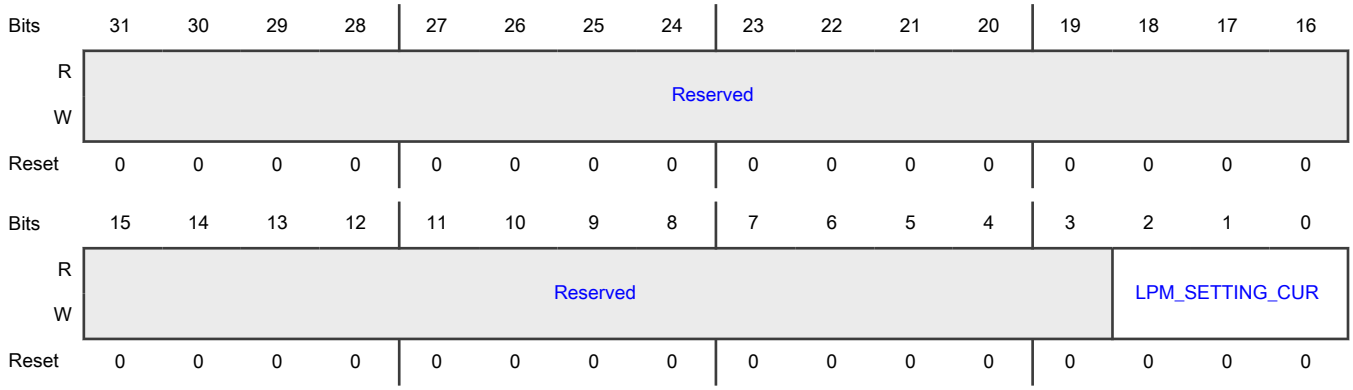
For a = 0 to 24:

Register	Offset
OSCPLLa_LPM_CUR	501Ch + (a × 40h)

Function

Reading this field will return OSCPLLn_LPM[LPM_SETTING_Dx], and x represent the CPU domain which launched the read access.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 LPM_SETTING_CUR	LPM settings value for current CPU domain that is reading this register. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.

20.6.1.51 Clock source working status (OSCPLL0_STATUS0 - OSCPLL24_STATUS0)

Offset

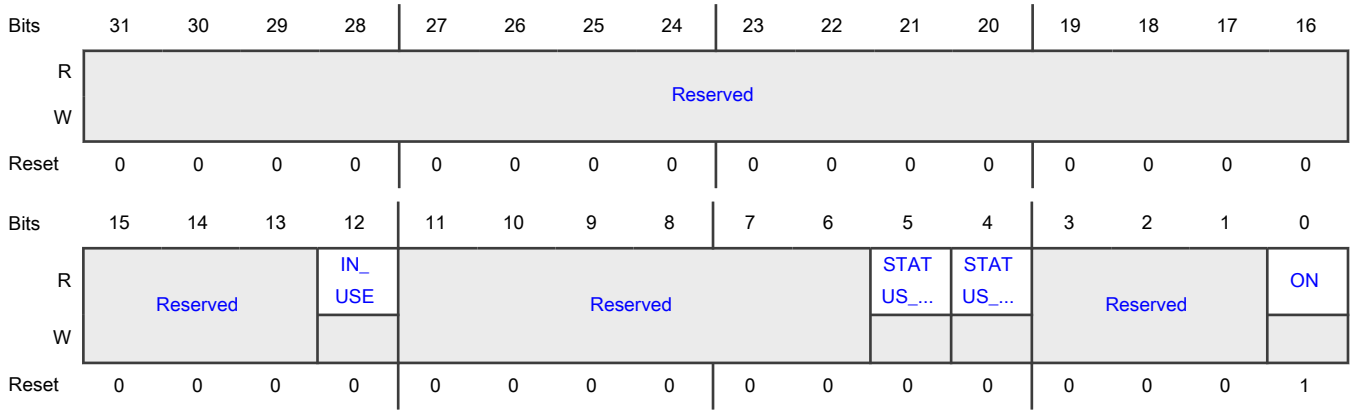
For a = 0 to 24:

Register	Offset
OSCPLLa_STATUS0	5020h + (a × 40h)

Function

This register shows current clock source running status.

Diagram



Fields

Field	Function
31-13 —	Reserved
12 IN_USE	This Clock Source is being used or not. This field will be 1, if any Clock Root derived from this Clock Source is enabled. 0b - Clock Source is not being used. 1b - Clock Source is being used.
11-6 —	Reserved
5 STATUS_LATE	Clock source ready This status bit indicates clock source is ready to use. 0b - Clock source is not ready to use 1b - Clock source is ready to use
4 STATUS_EARLY	Clock source active This status bit indicating clock source is active. 0b - Clock source is not active 1b - Clock source is active
3-1 —	Reserved
0 ON	Clock source current state Clock source running status. 0b - Clock source is OFF. 1b - Clock source is ON.

20.6.1.52 Clock source domain status (OSCPLL0_STATUS1 - OSCPLL24_STATUS1)

Offset

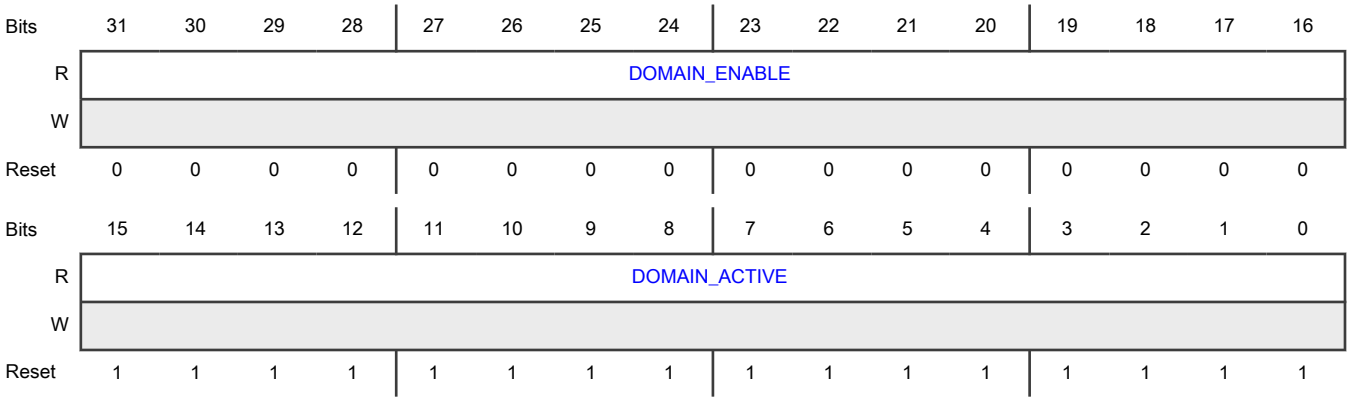
For a = 0 to 24:

Register	Offset
OSCPLLa_STATUS1	5024h + (a × 40h)

Function

This register shows domain status of clock source.

Diagram



Fields

Field	Function
31-16 DOMAIN_ENABLE	Domain enable Enable status for each domain. This clock source should be ON for DOMAINx. BIT16~BIT31 are corresponding with DOMAIN0~DOMAIN15.
15-0 DOMAIN_ACTIVE	Domain active This field is equal to OSCPLLn_AUTHEN[WHITE_LIST].

20.6.1.53 Clock Source access control (OSCPLL0_AUTHEN - OSCPLL24_AUTHEN)

Offset

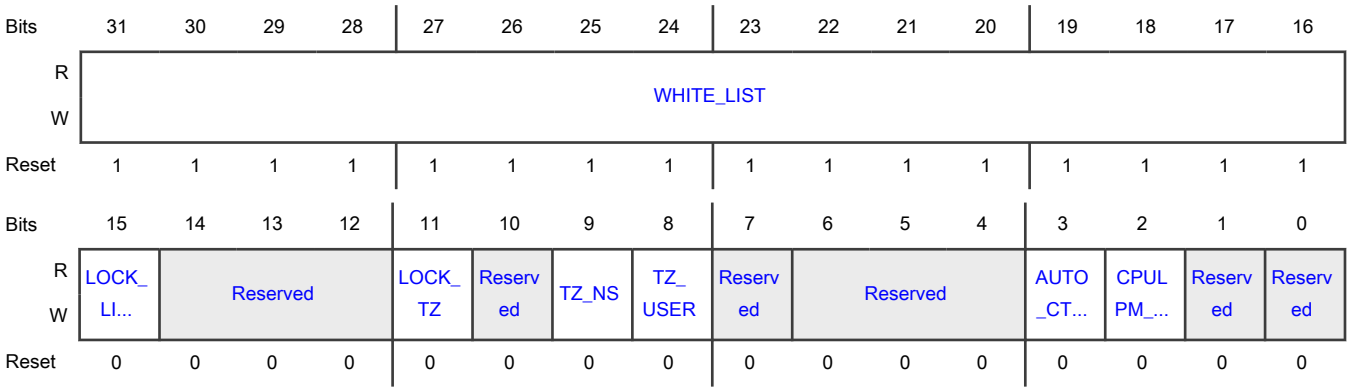
For a = 0 to 24:

Register	Offset
OSCPLLa_AUTHEN	5030h + (a × 40h)

Function

This register controls the access to this Clock Source slice. The access control includes 2 parts: TrustZone control and domain based granting.

Diagram



Fields

Field	Function
31-16 WHITE_LIST	Whitelist Each bit in this field represent for one domain. Bit16~Bit31 represent for DOMAIN0~DOMAIN15 respectively. Only the domains the corresponding bit of which is set to 1 can change the registers of this Clock Source.
15 LOCK_LIST	Lock white list This bit lock white list. When this bit is set, OSCPLLn_AUTHEN[WHITE_LIST] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset. 0b - Whitelist is not locked. 1b - Whitelist is locked.
14-12 —	Reserved
11 LOCK_TZ	Lock TrustZone settings This bit locks TrustZone settings. When this bit is set, OSCPLLn_AUTHEN[TZ_USER] and OSCPLLn_AUTHEN[TZ_NS] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset. 0b - TrustZone settings is not locked. 1b - TrustZone settings is locked.
10 —	Reserved
9 TZ_NS	Non-secure access permission

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Whether the registers of this Clock Source can be changed when CPU is in non-secure mode. The default value is 0, that means only the access from CPU in secure mode can change the registers of this Clock Source. 0b - Cannot be changed in Non-secure mode. 1b - Can be changed in Non-secure mode.
8 TZ_USER	User access permission Whether the registers of this Clock Source can be changed when CPU is in user mode. The default value is 0, that means only the access from CPU in supervisor mode can change the registers of this Clock Source. 0b - Clock Source settings cannot be changed in user mode. 1b - Clock Source settings can be changed in user mode.
7 —	Reserved
6-4 —	Reserved
3 AUTO_CTRL	Auto mode enable This field enables the Auto mode. Auto mode means if a clock source is used by any root, it will be automatically enabled. **NOTE** The actual working mode of this Clock Source is determined by OSCPLLn_AUTHEN[CPULPM_MODE] and OSCPLLn_AUTHEN[AUTO_CTRL]. 0b - Disable Auto mode 1b - Enable Auto mode
2 CPULPM_MODE	CPULPM mode enable This field enables the CPULPM mode. Once CPULPM mode is set, auto_ctrl is disabled. **NOTE** The actual working mode of this Clock Source is determined by OSCPLLn_AUTHEN[CPULPM_MODE] and OSCPLLn_AUTHEN[AUTO_CTRL]. 0b - Disable CPULPM mode. 1b - Enable CPULPM mode.
1 —	Reserved
0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

20.6.1.54 LPCG direct control (LPCG0_DIRECT - LPCG148_DIRECT)

Offset

For a = 0 to 148:

Register	Offset
LPCGa_DIRECT	8000h + (a × 40h)

Function

This register controls the ON/OFF of LPCG when LPCG slice is working in Direct Control mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved													CLKOFF_...	Reserved	ON
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-3 —	Reserved
2 CLKOFF_ACK_TIMEOUT_EN	Clock off handshake timeout enable This bit enables time out mechanism for clock off handshake, when enabled, clock off acknowledge will be ignored and wait for 16-cycle IPG clock timeout, when disabled, wait for clock off acknowledge. 0b - disable 1b - enable
1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 ON	Turn on LPCG This bit controls ON/OFF of LPCG. 0b - LPCG is OFF. 1b - LPCG is ON.

20.6.1.55 Clock source low power mode setting (LPCG0_LPM0 - LPCG148_LPM0)

Offset

For a = 0 to 148:

Register	Offset
LPCGa_LPM0	8010h + (a × 40h)

Function

Clock source low power mode setting for DOMAIN0 ~ DOMAIM7.

NOTE

The settings in this register are valid only when this Clock Source slice is working in CPULPM mode.

NOTE

The LPCG clock levels (LPM setting) of GPC, CCM, SRC and ANADIG should be set as 4 before CPU enter SUSPEND mode, otherwise CPU cannot be awakened once it entered SUSPEND mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	LPM_SETTING_D7			Reserved	LPM_SETTING_D6			Reserved	LPM_SETTING_D5			Reserved	LPM_SETTING_D4		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	LPM_SETTING_D3			Reserved	LPM_SETTING_D2			Reserved	LPM_SETTING_D1			Reserved	LPM_SETTING_D0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
30-28 LPM_SETTING_D7	Clock Source LPM in DOMAIN7 Clock Source low power mode setting in DOMAIN7. **NOTE** This field only can be changed by DOMAIN7. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
27 —	Reserved
26-24 LPM_SETTING_D6	Clock Source LPM in DOMAIN6 Clock Source low power mode setting in DOMAIN6. **NOTE** This field only can be changed by DOMAIN6. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
23 —	Reserved
22-20 LPM_SETTING_D5	Clock Source LPM in DOMAIN5 Clock Source low power mode setting in DOMAIN5. **NOTE** This field only can be changed by DOMAIN5. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 LPM_SETTING_D4	Clock Source LPM in DOMAIN4 Clock Source low power mode setting in DOMAIN4. **NOTE** This field only can be changed by DOMAIN4. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
15 —	Reserved
14-12 LPM_SETTING_D3	Clock Source LPM in DOMAIN3 Clock Source low power mode setting in DOMAIN3. **NOTE** This field only can be changed by DOMAIN3. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
11 —	Reserved
10-8 LPM_SETTING_D2	Clock Source LPM in DOMAIN2 Clock Source low power mode setting in DOMAIN2. **NOTE** This field only can be changed by DOMAIN2. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
7 —	Reserved
6-4	Clock Source LPM in DOMAIN1

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPM_SETTING_D1	Clock Source low power mode setting in DOMAIN1. **NOTE** This field only can be changed by DOMAIN1. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
3 —	Reserved
2-0 LPM_SETTING_D0	Clock Source LPM in DOMAIN0 Clock Source low power mode setting in DOMAIN0. **NOTE** This field only can be changed by DOMAIN0. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.

20.6.1.56 clock source low power mode setting (LPCG0_LPM1 - LPCG148_LPM1)

Offset

For a = 0 to 148:

Register	Offset
LPCGa_LPM1	8014h + (a × 40h)

Function

Clock source low power mode setting for DOMAIN8 ~ DOMAIM15.

NOTE

The settings in this register are valid only when this Clock Source slice is working in CPULPM mode.

NOTE

The LPCG clock levels (LPM setting) of GPC, CCM, SRC and ANADIG should be set as 4 before CPU enter SUSPEND mode, otherwise CPU cannot be awakened once it entered SUSPEND mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	LPM_SETTING_D15			Reserved	LPM_SETTING_D14			Reserved	LPM_SETTING_D13			Reserved	LPM_SETTING_D12		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	LPM_SETTING_D11			Reserved	LPM_SETTING_D10			Reserved	LPM_SETTING_D9			Reserved	LPM_SETTING_D8		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 LPM_SETTING_D15	Clock Source LPM in DOMAIN15 Clock Source low power mode setting in DOMAIN15. **NOTE** This field only can be changed by DOMAIN15. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
27 —	Reserved
26-24 LPM_SETTING_D14	Clock Source LPM in DOMAIN14 Clock Source low power mode setting in DOMAIN14. **NOTE** This field only can be changed by DOMAIN14. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
23 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
22-20 LPM_SETTING_D13	Clock Source LPM in DOMAIN13 Clock Source low power mode setting in DOMAIN13. **NOTE** This field only can be changed by DOMAIN13. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
19 —	Reserved
18-16 LPM_SETTING_D12	Clock Source LPM in DOMAIN12 Clock Source low power mode setting in DOMAIN12. **NOTE** This field only can be changed by DOMAIN12. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
15 —	Reserved
14-12 LPM_SETTING_D11	Clock Source LPM in DOMAIN11 Clock Source low power mode setting in DOMAIN11. **NOTE** This field only can be changed by DOMAIN11. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
11 —	Reserved
10-8	Clock Source LPM in DOMAIN10

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPM_SETTING_D10	Clock Source low power mode setting in DOMAIN10. **NOTE** This field only can be changed by DOMAIN10. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
7 —	Reserved
6-4 LPM_SETTING_D9	Clock Source LPM in DOMAIN9 Clock Source low power mode setting in DOMAIN9. **NOTE** This field only can be changed by DOMAIN9. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.
3 —	Reserved
2-0 LPM_SETTING_D8	Clock Source LPM in DOMAIN8 Clock Source low power mode setting in DOMAIN8. **NOTE** This field only can be changed by DOMAIN8. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.

20.6.1.57 LPM setting of current CPU domain (LPCG0_LPM_CUR - LPCG148_LPM_CUR)

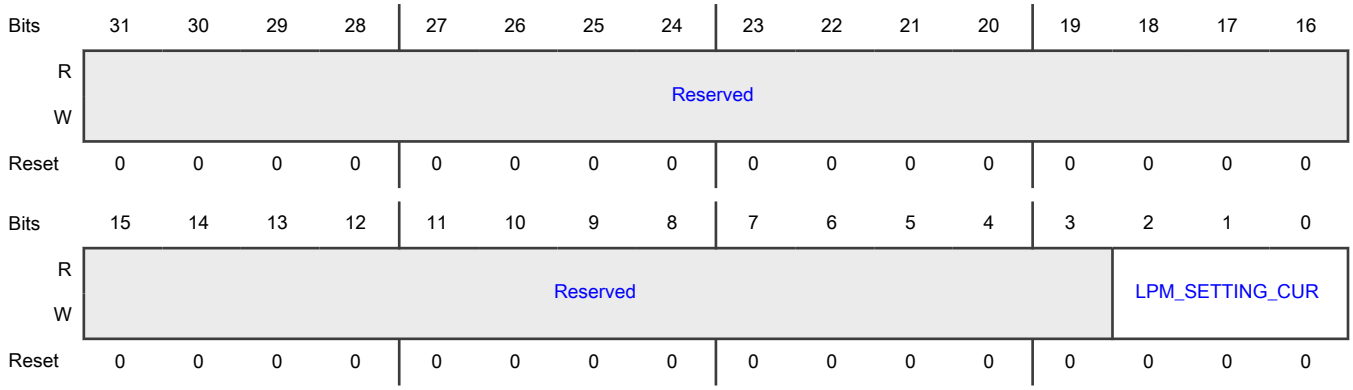
Offset

For a = 0 to 148:

Register	Offset
LPCGa_LPM_CUR	801Ch + (a × 40h)

Function
Reading this field will return LPCGn_LPM[LPM_SETTING_Dx], and x represent the CPU domain which launched the read access.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 LPM_SETTING_CUR	LPM settings value for current CPU domain that is reading this register. 000b - Clock Source will be OFF in any CPU mode. 001b - Clock Source will be ON in RUN mode, OFF in WAIT/STOP/SUSPEND mode. 010b - Clock Source will be ON in RUN/WAIT mode, OFF in STOP/SUSPEND mode. 011b - Clock Source will be ON in RUN/WAIT/STOP mode, OFF in SUSPEND mode.

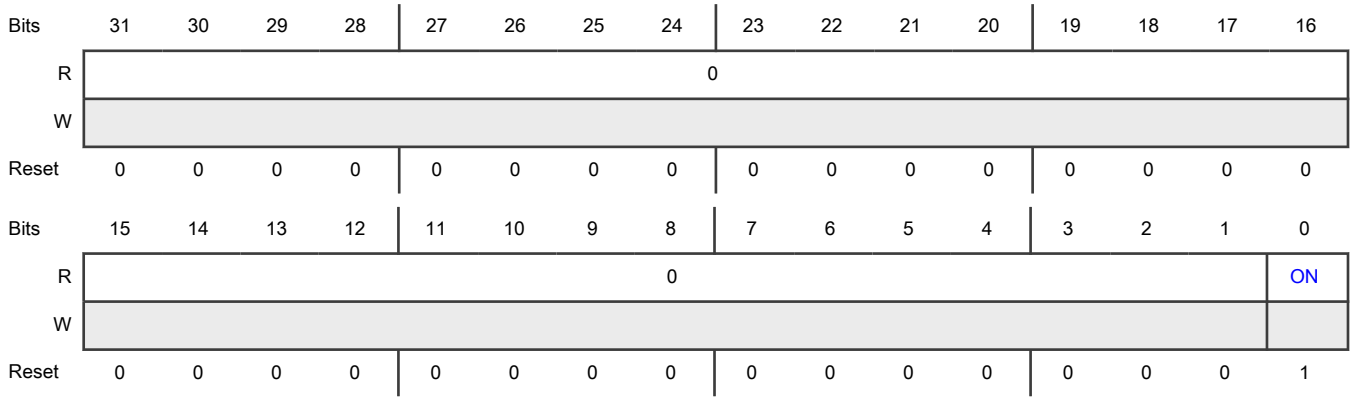
20.6.1.58 LPCG working status (LPCG0_STATUS0 - LPCG148_STATUS0)

Offset
For a = 0 to 148:

Register	Offset
LPCGa_STATUS0	8020h + (a × 40h)

Function
This register shows current LPCG running status.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 ON	LPCG work status 0b - LPCG is OFF. 1b - LPCG is ON.

20.6.1.59 LPCG domain status (LPCG0_STATUS1 - LPCG148_STATUS1)

Offset

For a = 0 to 148:

Register	Offset
LPCGa_STATUS1	8024h + (a × 40h)

Function

This register shows domains status of LPCG.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DOMAIN_ENABLE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DOMAIN_ACTIVE															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-16 DOMAIN_ENABLE	Domain enable Enable status for each domain. This clock source should be ON for DOMAINx. BIT16~BIT31 are corresponding with DOMAIN0~DOMAIN15.
15-0 DOMAIN_ACTIVE	Domain active This field is equal to LPCGn_AUTHEN[WHITE_LIST].

20.6.1.60 LPCG access control (LPCG0_AUTHEN - LPCG148_AUTHEN)

Offset

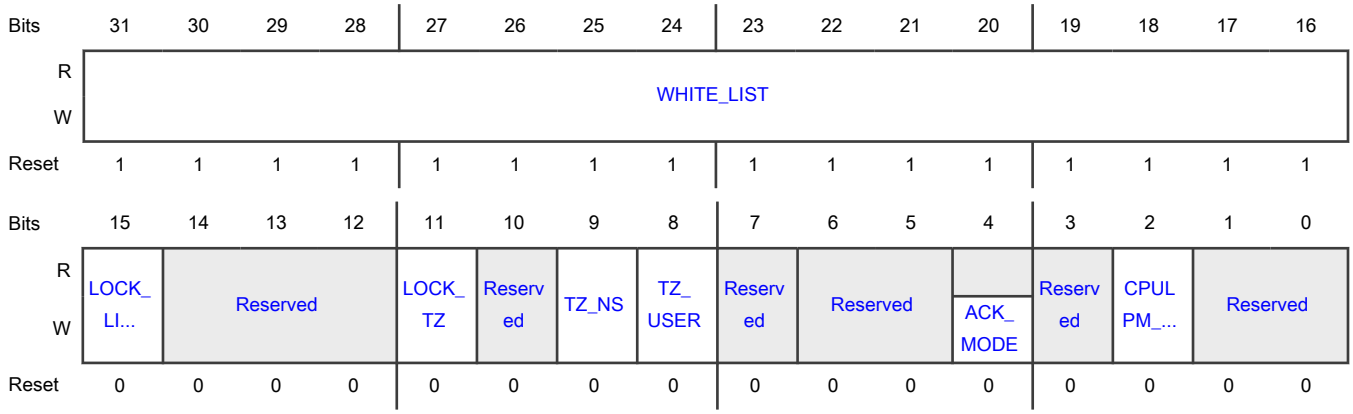
For a = 0 to 148:

Register	Offset
LPCGa_AUTHEN	8030h + (a × 40h)

Function

This register controls the access to this LPCG slice. The access control includes 2 parts: TrustZone control and domain based granting.

Diagram



Fields

Field	Function
31-16 WHITE_LIST	Whitelist Each bit in this field represent for one domain. Bit16~Bit31 represent for DOMAIN0~DOMAIN15 respectively. Only the domains the corresponding bit of which is set to 1 can change the registers of this LPCG slice.
15 LOCK_LIST	Lock white list This bit lock white list. When this bit is set, LPCGn_AUTHEN[WHITE_LIST] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset. 0b - Whitelist is not locked. 1b - Whitelist is locked.
14-12 —	Reserved
11 LOCK_TZ	Lock TrustZone settings This bit locks TrustZone settings. When this bit is set, LPCGn_AUTHEN[TZ_USER] and LPCGn_AUTHEN[TZ_NS] cannot be changed. Once this bit is set, it cannot be cleared, until next system reset. 0b - TrustZone settings is not locked. 1b - TrustZone settings is locked.
10 —	Reserved
9 TZ_NS	Non-secure access permission Whether the registers of this LPCG can be changed when CPU is in non-secure mode. The default value is 0, that means only the access from CPU in secure mode can change the registers of this LPCG. 0b - Cannot be changed in Non-secure mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Can be changed in Non-secure mode.
8 TZ_USER	User access permission Whether the registers of this LPCG can be changed when CPU is in user mode. The default value is 0, that means only the access from CPU in supervisor mode can change the registers of this LPCG. 0b - LPCG settings cannot be changed in user mode. 1b - LPCG settings can be changed in user mode.
7 —	Reserved
6-5 —	Reserved
4 ACK_MODE	CPULPM mode enable This field enables the ACK mode. **NOTE** ACK mode means a handshake between CCM and peripherals will be implemented if turning off its clock. 0b - Disable ACK mode. 1b - Enable ACK mode.
3 —	Reserved
2 CPULPM_MODE	CPULPM mode enable This field enables the CPULPM mode. **NOTE** The actual working mode of this LPCG is determined by OSCPLLn_AUTHEN[CPULPM_MODE] and OSCPLLn_AUTHEN[AUTO_CTRL]. 0b - Disable CPULPM mode, this LPCG is in Direct Control mode. 1b - Enable CPULPM mode, this LPCG is in CPULPM mode.
1-0 —	Reserved

Chapter 21
General Power Controller (GPC)

21.1 Chip-specific GPC Information

Table 144. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

21.2 Introduction

The General Power Controller (GPC) is the centralized power controller, which controls the power mode of the processor(s). The GPC takes the Wait For Interrupt (WFI) signal from CPU platforms, and wakeup events from peripherals, to determine the power mode based on the GPC power management policy. The GPC can also control the power mode transition.

GPC does not control the state of the resource directly. During power mode transition, it communicates with the resource controllers through the UPI (Unified Power Management Interface) to accomplish the power mode transition.

21.2.1 Block diagram

The GPC block consists of the following sub-modules:

- CPU Mode Control (CMC): It contains CPU mode controllers, one for each CPU platform. They control CPU mode of CPU platforms and their private resources.
- UPI CM Mapping: The connection between CMC and CPU platform is directly hard-wired, but CPU platform can be assigned to any domain in the system. This block is used to map CPU mode control signals into the correct domain.
- System Sleep Control (SSC): It is used to indicate resource controllers into and out of system sleep mode.

The figure below shows the block diagram of the General Power Controller.

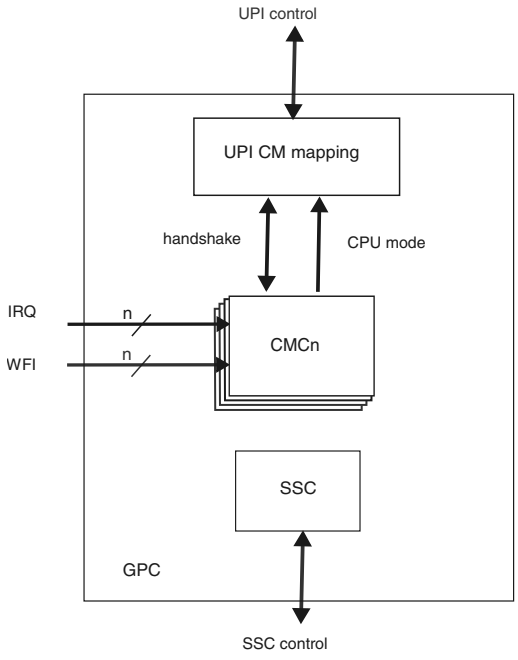


Figure 77. GPC block diagram

21.2.2 Features

The GPC includes the following features:

- Independent CPU modes for each CPU platform
- System sleep management
- Domain access control
- Controllable steps of sleep and wakeup sequence

21.3 Functional description

The following sections describe functional details of the GPC module.

21.3.1 CPU Mode Control (CMC)

The CPU mode is a power mode of the CPU platform. The following are the four CPU modes:

Table 145. CPU Modes

CPU Mode	Description
RUN	The CPU core is active and running under normal operation. All the blocks inside the CPU platform can be accessed when needed. The state of private resources are fully controlled by software configuration.
WAIT	The CPU is in WFI/WFE state, but can to get back to RUN mode with very short latency. In typical applications, the CPU will enter WAIT mode whenever there is no active thread running. In WAIT mode, the clock to the CPU core is gated off, the cache is clock gated, and the TCM is still active since there are other modules, such as DMA, that still needs access to it.

Table continues on the next page...

Table 145. CPU Modes (continued)

CPU Mode	Description
STOP	The CPU is in WFI/WFE state, and does not require an extremely short exit time. In STOP mode, the clocks to the CPU core, Cache, and TCM are all gated off. The clocks to the bus and peripherals are also gated off. This is the lowest power consumption mode without losing the state of the peripheral. When exiting from STOP mode, there is no need to re-initialize the peripherals.
SUSPEND	Entering SUSPEND mode is for lowest power consumption, and exit time is not critical. In SUSPEND mode, CPU, Cache, and peripherals are all power gated. The biggest difference between SUSPEND and STOP mode is the peripherals will be power gated. Because the peripherals are power gated, entering SUSPEND mode requires the CPU to save the state of the peripherals. When exiting from SUSPEND mode, the CPU needs to restore these states.

CPU Mode transitions happen when the following events occur:

- Sleep Event: CPU enters sleep state with WFI/WFE instruction
- Wakeup Event: Any unmasked IRQ wakeup

On a Sleep Event, the CPU has the option to enter WAIT/STOP/SUSPEND mode, or stay in RUN mode. The transition to WAIT/STOP/SUSPEND is called the sleep sequence, and the next state is configured by CM_MODE_CTRL[CPU_MODE_TARGET].

When CPU is in WAIT/STOP/SUSPEND mode, a wakeup request can wake the CPU, and bring it back to RUN mode. This is transition is called the wakeup sequence. The CM_IRQ_WAKEUP_MASK_n register selects which IRQs can wakeup the CPU platform. There are also non-IRQ requests, which can wakeup the CPU. These non-IRQ requests are configured by the CM_NON_IRQ_WAKEUP_MASK register.

During a CPU mode transition, the GPC communicates with the system resource controllers by sending the target CPU mode and transition step. Please see the UPI interface for more details.

If a resource is set as private to a single CPU platform, the resource's power state will change with the CPU platform's mode. If a resource is set as shared by multiple CPU platforms, the resource controller needs to look at all the current modes of the CPU platforms to determine the power state of the resource.

21.3.2 System Sleep Mode

System sleep mode is the second kind of low power mode besides CPU mode. System sleep mode is related to the state of all CPU platforms. When all CPU platforms are in low power mode, the system can further shut off some analog modules or put them into their own STANDBY mode. This chip status is called SYSTEM SLEEP. For details of each system sleep transition step, refer to Table 147.

The CMC can request send a system sleep when the CPU is in WAIT/STOP/SUSPEND modes by configuring the CM_SYS_SLEEP_CTRL[SS_WAIT, SS_STOP, SS_SUSPEND].

The SSC maintains the system sleep status. When all the CPU platforms send a system sleep request, the system is put into system sleep mode. Any interrupt can abort a system sleep sequence so that the system can be quickly woken up. If a CPU is disabled by FUSE, the GPC considers it to be always in the system sleep mode.

21.3.3 Unified Power Management Interface (UPI)

GPC does not control the state of the resource directly. During power mode transition, it communicates with the resource controllers through the UPI (Unified Power management Interface) to accomplish the power mode transition.

GPC and resource controllers implement a handshaking mechanism for mode transitions.

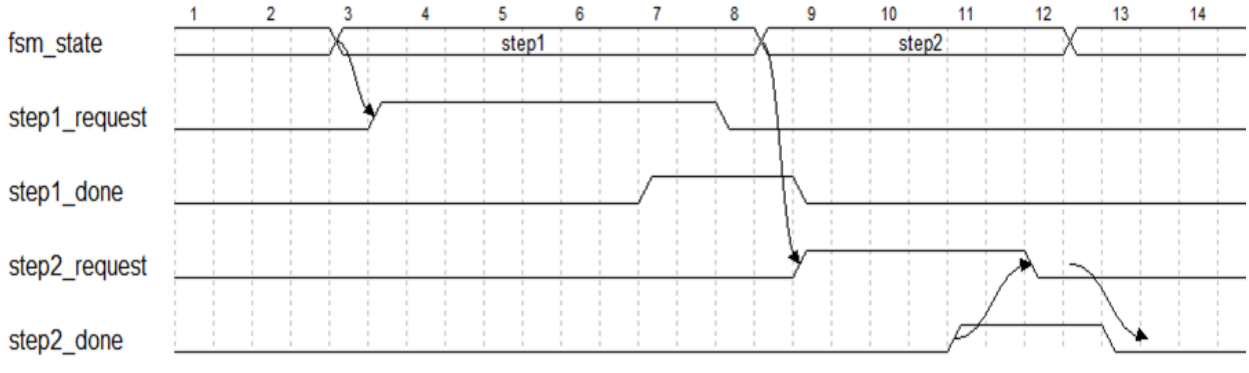


Figure 78. Handshake method

21.3.4 Low Power Sequence

21.3.4.1 CPU mode transition flow

There are two CPU mode control instances in iMX RT1180 GPC: CMC0 for CM33 and CMC1 for CM7. Each CMC has a state machine to control the sequence of CPU mode transition.

When CMC receives a sleep request from CPU platform, it starts the sleep sequence. The flow cannot be interrupted by a wakeup request until the SLEEP_POWER request completes. In SLEEP_SYS step, the SSC runs the system sleep transition flow. After the sleep sequence finishes, the CMC stays at IDLE_SLEEP and waits for a wakeup event to start wakeup sequence, which is shown in the left part of the diagram.

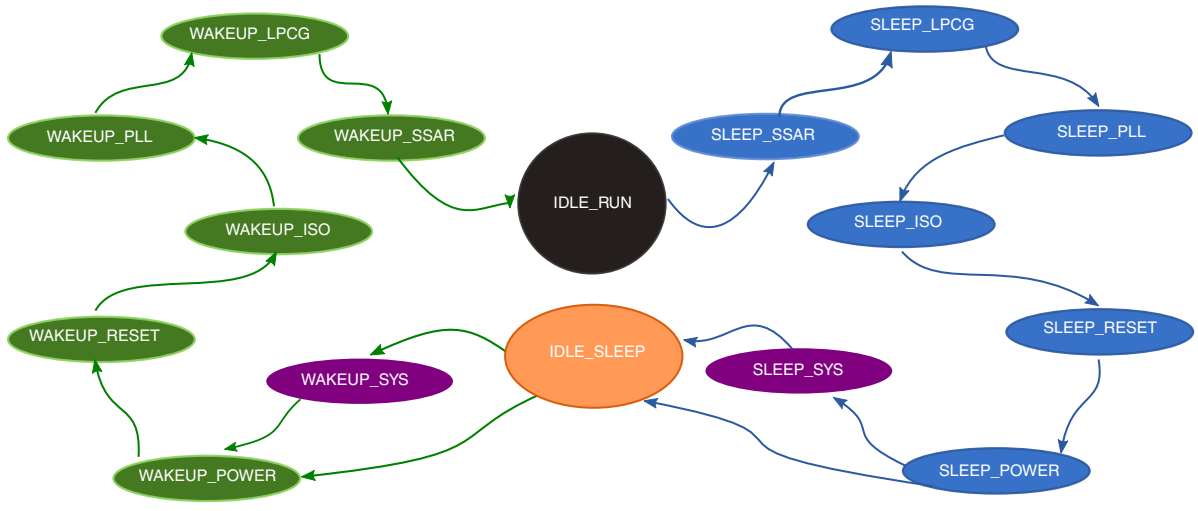


Figure 79. CPU mode transition flow

Table 146. CPU mode transition steps

State	Description
IDLE_RUN	CPU is in RUN mode

Table continues on the next page...

Table 146. CPU mode transition steps (continued)

State	Description
SLEEP_SSAR	GPC asserts command to SRC to implement handshake with ELE for power-off.
SLEEP_LPCG	GPC asserts command to CCM to disable LPCGs
SLEEP_PLL	GPC asserts command to CCM to turn PLLs off
SLEEP_ISO	GPC asserts command to SRC to enable isolation signals
SLEEP_RESET	GPC asserts command to SRC to assert reset signals
SLEEP_POWER	GPC asserts command to SRC to turn off power switch signals
SLEEP_SYS	SSC runs the system sleep transition flow
IDLE_SLEEP	CMC waits for wakeup event to start wakeup sequence
WAKEUP_SYS	SSC runs the wakeup flow
WAKEUP_RESET	GPC asserts command to SRC to turn on power switch signals
WAKEUP_POWER	GPC asserts command to SRC to de-assert reset signals
WAKEUP_ISO	GPC asserts command to SRC to disable isolation signals
WAKEUP_PLL	GPC asserts command to CCM to turn PLLs on
WAKEUP_LPCG	GPC asserts command to CCM to enable LPCGs
WAKEUP_SSAR	GPC asserts command to SRC to implement handshake with ELE for power-on.

21.3.4.2 System sleep transition flow

If all CPUs are in low power state and they all request system sleep mode, a system sleep sequence will be triggered.

There are two state machines in SSC: the first in 24MHz ROSC clock domain, the second in 32KHz clock domain. The first SSC state machine starts when CMC sends a system sleep request, it looks at all CMC system sleep status, if any CPU platform is not in sleep mode or not allow system sleep, this system sleep request will be refused. When this state machine transits to ROSC_OFF and there is no wakeup request, the second state machine starts. The system sleep in sequence can be aborted after any step completes because of wakeup time requirement. In the 24MHz state machine, step 0 is for BIAS, step 1 is for PLDO, step 2 is for BANDGAP, and step 3 is for LDO.

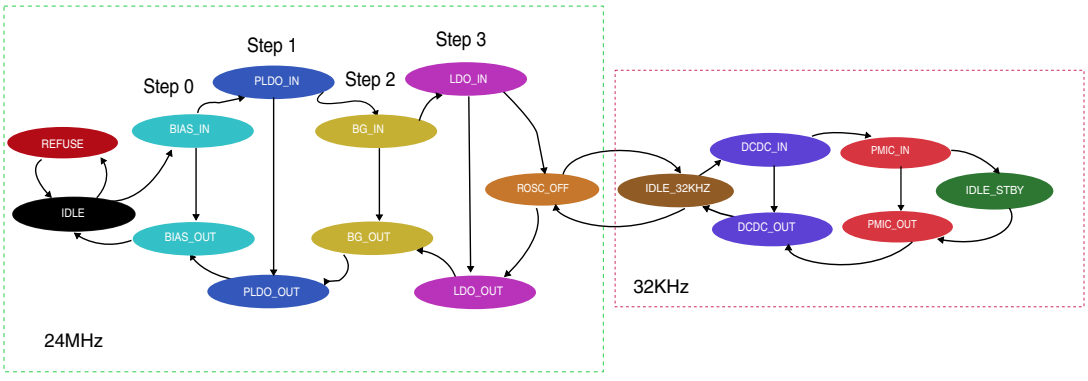


Figure 80. System sleep transition flow

Table 147. System sleep transition steps

State	Description
REFUSE	Condition for system sleep is not met. FSM goes to REFUSE state, then automatically returns to IDLE again.
IDLE	System sleep mode is not requested. 24MHz FSM is in IDLE mode.
BIAS_IN	GPC asserts command to disable BIAS.
PLDO_IN	GPC asserts command to disable PHY LDO.
BG_IN	GPC asserts command to power down BANDGAP.
LDO_IN	GPC asserts command to put LDO into low power mode.
ROSC_OFF	GPC asserts command to disable 24MHz OSC.
IDLE_32KHZ	32KHz FSM is in IDLE mode.
DCDC_IN	GPC asserts command to DCDC to select low power target trim value to analog part.
PMIC_IN	GPC asserts PMIC_STBY_REQ to external PMIC to adjust voltage.
IDLE_STBY	Chip is in SYSTEM SLEEP mode.
*_OUT	The steps named as *_OUT consist of a system sleep wakeup sequence.

21.3.5 System Domain

21.3.5.1 Domain assignment

Each CPU platform and the CMC connected to it can be assigned to the same domain.

The GPC_CPUn_DOMAIN[CPUn_DOMAIN] is used to select the domain assignments for CMCn.

Based on the domain assignment, a mapping mechanism between CPU platform and system domain is implemented inside GPC. The CMCn's power mode and each handshake step request will be mapped to all the domains it belongs to. When all its domains' Done response are asserted, the step handshake completes.

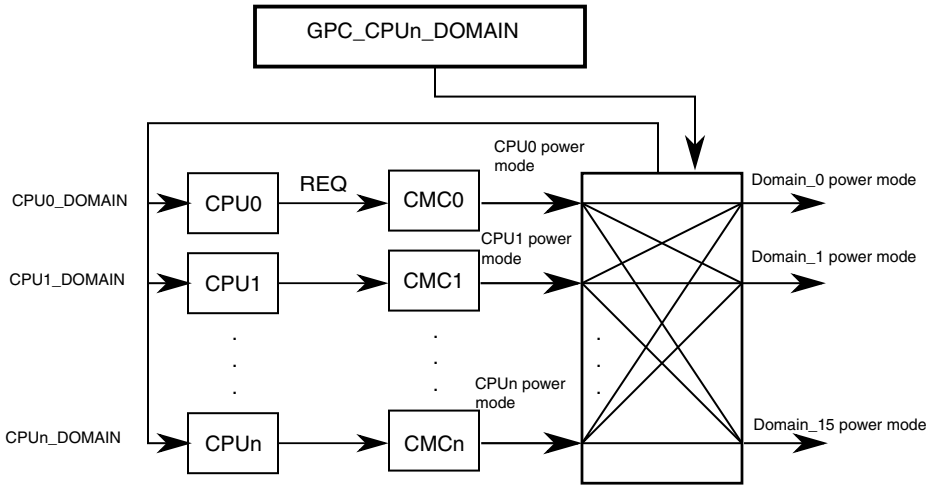


Figure 81. Domain Assignment

When more than one CMCs is assigned to same domain, a two stage priority selects which CMC can control the domain's UPI output. The first stage is CPUn_MASTER flag from GLOBAL which is enabled by register GPC_MASTER[CPUn_MASTER]. Only one CMC with CPUn_MASTER flag owns the domain UPI output. The second stage is for the case when more than one CMC has CPUn_MASTER flag or no CMC has this flag. In this case, the lower indexed CMC has higher priority.

21.3.5.2 Register Access Permissions

The GPC register access is controlled by the following registers and bitfields:

- CM_AUTHEN_CTRL register for CMC access permission
- SS_AUTHEN_CTRL register for SSC access permission
- GLOBAL_AUTHEN_CTRL register to control GLOBAL access permission
- x_AUTHEN_CTRL[WHITE_LIST] field configures which domain IDs can write to the registers
- x_AUTHEN_CTRL[USER] and x_AUTHEN_CTRL[NONSECURE] fields are used to set the privilege and security permission
- x_AUTHEN_CTRL[LOCK_CFG] locks configuration fields and should only be set at initialization
- x_AUTHEN_CTRL[LOCK_LIST] locks WHITE_LIST fields and should only be set at initialization

The GPC returns an error response when a register write access has any access violations.

21.3.6 Clocks

This section describes clocks and special clocking requirements of the GPC module.

Table 148. Clocks

Clock	Description
ipg_clk	Main GPC clock. This clock originates from RCOSC 24 MHz and can be turned off by setting GPC_ROSC_CTRL[ROSC_OFF_EN]. When the ROSC_OFF_EN = 1, the 24 MHz system sleep state machine transitions to the ROSC_OFF state and the GPC shuts off the ROSC 24 MHz block. The 32 KHz clock is then used for the 32 KHz state machine transition.
32k_clk	32 kHz clock used to wakeup GPC and RCOSC. The 32 kHz clock is not needed by ipg_clk since they are fully asynchronous

21.3.7 Interrupts

The interrupt inputs of CMC0 and CMC1 are independent and the same as the CPU platforms they are connected to. Each CMC has up to 256 IRQ input to wake up the core or system.

21.4 External Signals

The following table describes the GPC external signals.

Table 149. External Signals

Signal	Description	Direction
PMIC_STBY_REQ	Asserted by the GPC to the external PMIC to adjust the voltage.	Output
PMIC_READY	Indicates that the external PMIC is ready.	Input

21.5 Initialization

See Low Power Sequence for low power entry and exit flow and considerations.

21.5.1 CPU Sleep Hold

There is a certain delay between the time the core sleeps and when the core clock stops. If a wakeup IRQ appears inside this delay window, the core will process the IRQ and wake up, but the GPC sleep sequence is still shutting off the core clock. Since the sleep sequence is not interruptable, this scenario may lead to unpredictable behavior.

To avoid this scenario, a CPU Sleep Hold is implemented to hold the core in the sleep state until the sleep sequence is finished. The GPC will then send a signal to release the core. The CM_MISC[SLEEP_HOLD_EN] field is used to enable this function.

The core clock can also be gated by a WFI event without any GPC sequence. The GPC uses a signal to clear this gate when a wakeup trigger occurs.

21.6 Memory map and register definition

This section includes the GPC module memory map and detailed descriptions of all registers.

21.6.1 register descriptions

21.6.1.1 gpc_cpu_ctrl memory map

GPC_CPU_CTRL base address: 4447_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	CM Authentication Control (CPU0_CM_AUTHEN_CTRL)	32	RW	FFFF_0000h
Ch	Miscellaneous (CPU0_CM_MISC)	32	RW	0000_000Eh
10h	CPU mode control (CPU0_CM_MODE_CTRL)	32	RW	0000_0000h
14h	CM CPU mode Status (CPU0_CM_MODE_STAT)	32	R	0000_0000h
100h	CM IRQ0~31 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
104h	CM IRQ32~63 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_1)	32	RW	0000_0000h
108h	CM IRQ64~95 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_2)	32	RW	0000_0000h
10Ch	CM IRQ96~127 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_3)	32	RW	0000_0000h
110h	CM IRQ128~159 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_4)	32	RW	0000_0000h
114h	CM IRQ160~191 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_5)	32	RW	0000_0000h
118h	CM IRQ192~223 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_6)	32	RW	0000_0000h
11Ch	CM IRQ224~255 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_7)	32	RW	0000_0000h
140h	CM non-IRQ wakeup mask (CPU0_CM_NON_IRQ_WAKEUP_MASK)	32	RW	0000_0001h
150h	CM IRQ0~31 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_0)	32	R	0000_0000h
154h	CM IRQ32~63 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_1)	32	R	0000_0000h
158h	CM IRQ64~95 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_2)	32	R	0000_0000h
15Ch	CM IRQ96~127 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_3)	32	R	0000_0000h
160h	CM IRQ128~159 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_4)	32	R	0000_0000h
164h	CM IRQ160~191 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_5)	32	R	0000_0000h
168h	CM IRQ192~223 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_6)	32	R	0000_0000h
16Ch	CM IRQ224~255 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_7)	32	R	0000_0000h
190h	CM non-irq wakeup status (CPU0_CM_NON_IRQ_WAKEUP_STAT)	32	R	0000_0000h
200h	CM sleep SSAR control (CPU0_CM_SLEEP_SSAR_CTRL)	32	RW	0000_0004h
208h	CM sleep LPCG control (CPU0_CM_SLEEP_LPCG_CTRL)	32	RW	0000_0004h
210h	CM sleep PLL control (CPU0_CM_SLEEP_PLL_CTRL)	32	RW	0000_0004h
218h	CM sleep isolation control (CPU0_CM_SLEEP_ISO_CTRL)	32	RW	0000_0004h
220h	CM sleep reset control (CPU0_CM_SLEEP_RESET_CTRL)	32	RW	0000_0004h
228h	CM sleep power control (CPU0_CM_SLEEP_POWER_CTRL)	32	RW	0000_0004h
290h	CM wakeup power control (CPU0_CM_WAKEUP_POWER_CTRL)	32	RW	0000_0004h
298h	CM wakeup reset control (CPU0_CM_WAKEUP_RESET_CTRL)	32	RW	0000_0004h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2A0h	CM wakeup isolation control (CPU0_CM_WAKEUP_ISO_CTRL)	32	RW	0000_0004h
2A8h	CM wakeup PLL control (CPU0_CM_WAKEUP_PLL_CTRL)	32	RW	0000_0004h
2B0h	CM wakeup LPCG control (CPU0_CM_WAKEUP_LPCG_CTRL)	32	RW	0000_0004h
2C0h	CM wakeup SSAR control (CPU0_CM_WAKEUP_SSAR_CTRL)	32	RW	0000_0004h
380h	CM system sleep control (CPU0_CM_SYS_SLEEP_CTRL)	32	RW	0000_0004h
804h	CM Authentication Control (CPU1_CM_AUTHEN_CTRL)	32	RW	FFFF_0000h
80Ch	Miscellaneous (CPU1_CM_MISC)	32	RW	0000_000Eh
810h	CPU mode control (CPU1_CM_MODE_CTRL)	32	RW	0000_0000h
814h	CM CPU mode Status (CPU1_CM_MODE_STAT)	32	R	0000_0000h
900h	CM IRQ0~31 wakeup mask (CPU1_CM_IRQ_WAKEUP_MASK_0)	32	RW	0000_0000h
904h	CM IRQ32~63 wakeup mask (CPU1_CM_IRQ_WAKEUP_MASK_1)	32	RW	0000_0000h
908h	CM IRQ64~95 wakeup mask (CPU1_CM_IRQ_WAKEUP_MASK_2)	32	RW	0000_0000h
90Ch	CM IRQ96~127 wakeup mask (CPU1_CM_IRQ_WAKEUP_MASK_3)	32	RW	0000_0000h
910h	CM IRQ128~159 wakeup mask (CPU1_CM_IRQ_WAKEUP_MASK_4)	32	RW	0000_0000h
914h	CM IRQ160~191 wakeup mask (CPU1_CM_IRQ_WAKEUP_MASK_5)	32	RW	0000_0000h
918h	CM IRQ192~223 wakeup mask (CPU1_CM_IRQ_WAKEUP_MASK_6)	32	RW	0000_0000h
91Ch	CM IRQ224~255 wakeup mask (CPU1_CM_IRQ_WAKEUP_MASK_7)	32	RW	0000_0000h
940h	CM non-IRQ wakeup mask (CPU1_CM_NON_IRQ_WAKEUP_MASK)	32	RW	0000_0001h
950h	CM IRQ0~31 wakeup status (CPU1_CM_IRQ_WAKEUP_STAT_0)	32	R	0000_0000h
954h	CM IRQ32~63 wakeup status (CPU1_CM_IRQ_WAKEUP_STAT_1)	32	R	0000_0000h
958h	CM IRQ64~95 wakeup status (CPU1_CM_IRQ_WAKEUP_STAT_2)	32	R	0000_0000h
95Ch	CM IRQ96~127 wakeup status (CPU1_CM_IRQ_WAKEUP_STAT_3)	32	R	0000_0000h
960h	CM IRQ128~159 wakeup status (CPU1_CM_IRQ_WAKEUP_STAT_4)	32	R	0000_0000h
964h	CM IRQ160~191 wakeup status (CPU1_CM_IRQ_WAKEUP_STAT_5)	32	R	0000_0000h
968h	CM IRQ192~223 wakeup status (CPU1_CM_IRQ_WAKEUP_STAT_6)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
96Ch	CM IRQ224~255 wakeup status (CPU1_CM_IRQ_WAKEUP_STAT_7)	32	R	0000_0000h
990h	CM non-irq wakeup status (CPU1_CM_NON_IRQ_WAKEUP_STAT)	32	R	0000_0000h
A00h	CM sleep SSAR control (CPU1_CM_SLEEP_SSAR_CTRL)	32	RW	0000_0004h
A08h	CM sleep LPCG control (CPU1_CM_SLEEP_LPCG_CTRL)	32	RW	0000_0004h
A10h	CM sleep PLL control (CPU1_CM_SLEEP_PLL_CTRL)	32	RW	0000_0004h
A18h	CM sleep isolation control (CPU1_CM_SLEEP_ISO_CTRL)	32	RW	0000_0004h
A20h	CM sleep reset control (CPU1_CM_SLEEP_RESET_CTRL)	32	RW	0000_0004h
A28h	CM sleep power control (CPU1_CM_SLEEP_POWER_CTRL)	32	RW	0000_0004h
A90h	CM wakeup power control (CPU1_CM_WAKEUP_POWER_CTRL)	32	RW	0000_0004h
A98h	CM wakeup reset control (CPU1_CM_WAKEUP_RESET_CTRL)	32	RW	0000_0004h
AA0h	CM wakeup isolation control (CPU1_CM_WAKEUP_ISO_CTRL)	32	RW	0000_0004h
AA8h	CM wakeup PLL control (CPU1_CM_WAKEUP_PLL_CTRL)	32	RW	0000_0004h
AB0h	CM wakeup LPCG control (CPU1_CM_WAKEUP_LPCG_CTRL)	32	RW	0000_0004h
AC0h	CM wakeup SSAR control (CPU1_CM_WAKEUP_SSAR_CTRL)	32	RW	0000_0004h
B80h	CM system sleep control (CPU1_CM_SYS_SLEEP_CTRL)	32	RW	0000_0004h

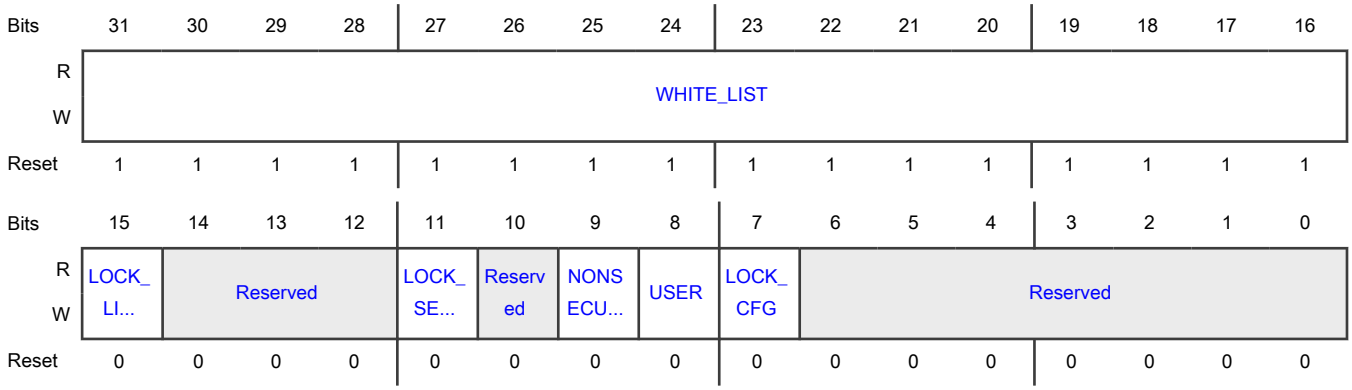
21.6.1.2 CM Authentication Control (CPU0_CM_AUTHEN_CTRL - CPU1_CM_AUTHEN_CTRL)

Offset

Register	Offset
CPU0_CM_AUTHEN_CTRL	4h
CPU1_CM_AUTHEN_CTRL	804h

Function
Authentication control for CMC.

Diagram



Fields

Field	Function
31-16 WHITE_LIST	Domain ID white list When bit is set to 1, the corresponding domain ID can access CPU mode control registers. bit[0] for Domain 0, bit[1] for Domain 1, bit[2] for Domain 2, and bit[3] for Domain 3. Multiple bits with a value of 1 is permitted.
15 LOCK_LIST	White list lock This bit locks the WHITE_LIST. When this bit is set, WHITE_LIST cannot be changed. Once this bit is set, it cannot be cleared until the next system reset. 0b - WHITE_LIST is not locked 1b - WHITE_LIST is locked
14-12 —	Reserved
11 LOCK_SETTING	Lock NONSECURE and USER This bit locks the NONSECURE and USER fields. When this bit is set, LOCK_SETTING cannot be changed. Once this bit is set, it cannot be cleared until the next system reset. 0b - NONSECURE and USER fields are not locked 1b - NONSECURE and USER fields are locked
10 —	Reserved
9 NONSECURE	Allow non-secure mode access 0b - Allow only secure mode to access CPU mode control 1b - Allow both secure and non-secure mode to access CPU mode control registers
8 USER	Allow user mode access 0b - Allow only privilege mode to access CPU mode control registers

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Allow both privilege and user mode to access CPU mode control registers
7 LOCK_CFG	Configuration lock This bit locks the value of low power configuration fields. Once this bit is set, it cannot be cleared until the next system reset. 0b - The value of low power configuration fields are not locked. 1b - The value of low power configuration fields are locked. It locks the CPUx_CM registers which are marked as "Locked by LOCK_CFG field" in the function field.
6-0 —	Reserved

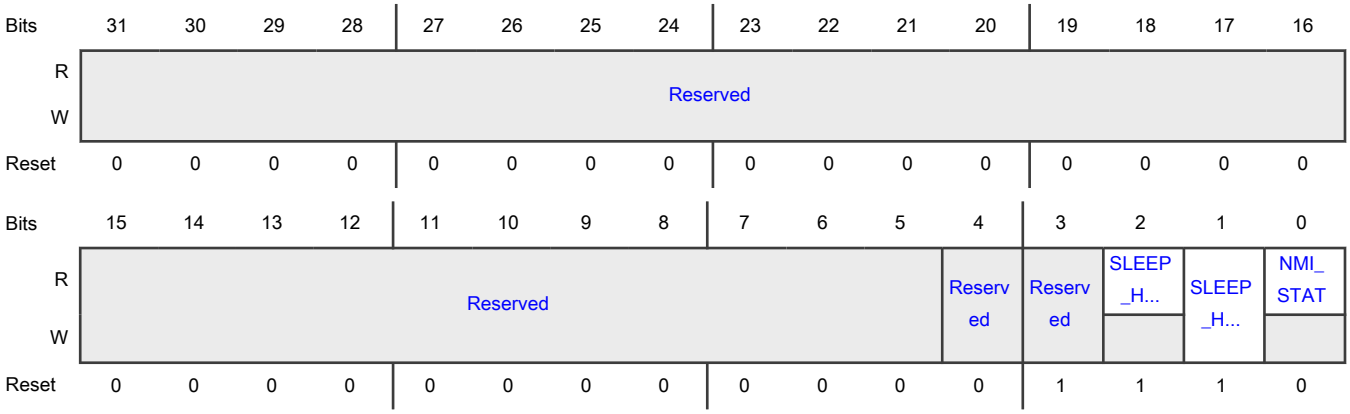
21.6.1.3 Miscellaneous (CPU0_CM_MISC - CPU1_CM_MISC)

Offset

Register	Offset
CPU0_CM_MISC	Ch
CPU1_CM_MISC	80Ch

Function
Miscellaneous register.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 SLEEP_HOLD_STAT	CPU sleep hold status 0b - CPU sleep hold is acknowledged 1b - CPU is not in sleep hold
1 SLEEP_HOLD_EN	Allow cpu_sleep_hold_req to assert during CPU low power status 0b - Disable cpu_sleep_hold_req 1b - Allow cpu_sleep_hold_req to assert during CPU low power status
0 NMI_STAT	Non-masked interrupt status 0b - NMI is not asserted 1b - NMI is asserted

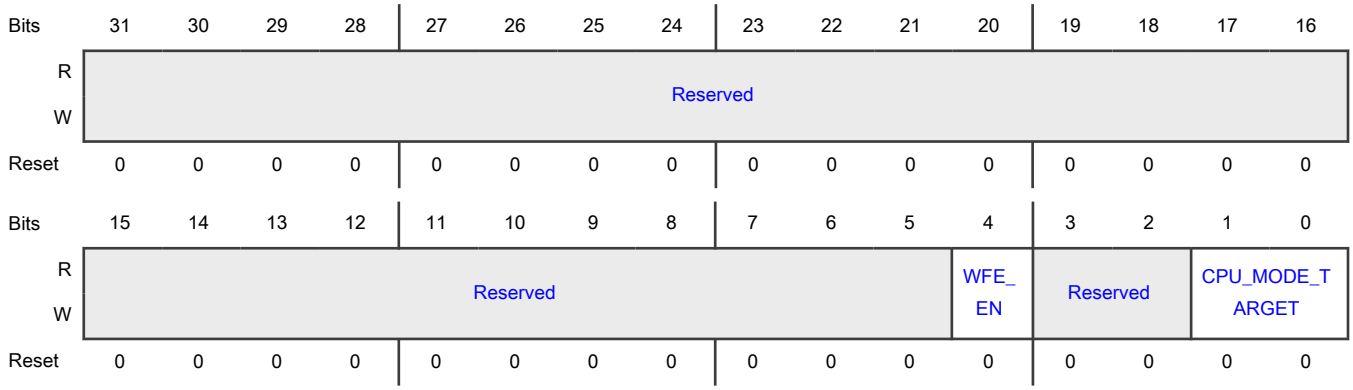
21.6.1.4 CPU mode control (CPU0_CM_MODE_CTRL - CPU1_CM_MODE_CTRL)

Offset

Register	Offset
CPU0_CM_MODE_CTRL	10h
CPU1_CM_MODE_CTRL	810h

Function
CPU mode control register.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 WFE_EN	WFE assertion can be sleep event 0b - WFE assertion can not trigger low power 1b - WFE assertion can trigger low power
3-2 —	Reserved
1-0 CPU_MODE_TARGET	The CPU mode the CPU platform should transit to on next sleep event Important: Core MUST re-configure this field EACH TIME before entering sleep 00b - Stay in RUN mode 01b - Transit to WAIT mode 10b - Transit to STOP mode 11b - Transit to SUSPEND mode

21.6.1.5 CM CPU mode Status (CPU0_CM_MODE_STAT - CPU1_CM_MODE_STAT)

Offset

Register	Offset
CPU0_CM_MODE_STAT	14h
CPU1_CM_MODE_STAT	814h

Function

CPU mode status register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				Reserved				Reserved					Reserved	Reserved	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					Reserved	Reserved	Reserved	Reserved				CPU_MODE_PREVIO...		CPU_MODE_CURRENT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 —	Reserved
23-19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15-11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 —	Reserved
3-2 CPU_MODE_PREVIOUS	Previous CPU mode 00b - CPU was previously in RUN mode 01b - CPU was previously in WAIT mode 10b - CPU was previously in STOP mode 11b - CPU was previously in SUSPEND mode
1-0 CPU_MODE_CURRENT	Current CPU mode 00b - CPU is currently in RUN mode 01b - CPU is currently in WAIT mode 10b - CPU is currently in STOP mode 11b - CPU is currently in SUSPEND mode

21.6.1.6 CM IRQ0~31 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_0 - CPU1_CM_IRQ_WAKEUP_MASK_0)

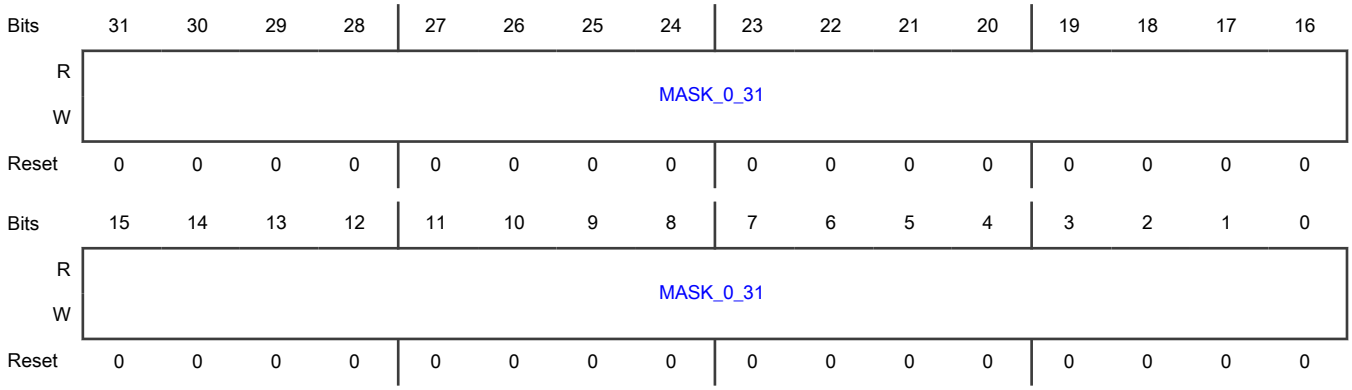
Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_MASK_0	100h
CPU1_CM_IRQ_WAKEUP_MASK_0	900h

Function

IRQ0~31 mask register.

Diagram



Fields

Field	Function
31-0 MASK_0_31	"1" means the IRQ cannot wakeup CPU platform

21.6.1.7 CM IRQ32~63 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_1 - CPU1_CM_IRQ_WAKEUP_MASK_1)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_MASK_1	104h
CPU1_CM_IRQ_WAKEUP_MASK_1	904h

Function
IRQ32~63 mask register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MASK_32_63															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MASK_32_63															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MASK_32_63	"1" means the IRQ cannot wakeup CPU platform

21.6.1.8 CM IRQ64~95 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_2 - CPU1_CM_IRQ_WAKEUP_MASK_2)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_MASK_2	108h
CPU1_CM_IRQ_WAKEUP_MASK_2	908h

Function
IRQ64~95 mask register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MASK_64_95															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MASK_64_95															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MASK_64_95	"1" means the IRQ cannot wakeup CPU platform

21.6.1.9 CM IRQ96~127 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_3 - CPU1_CM_IRQ_WAKEUP_MASK_3)

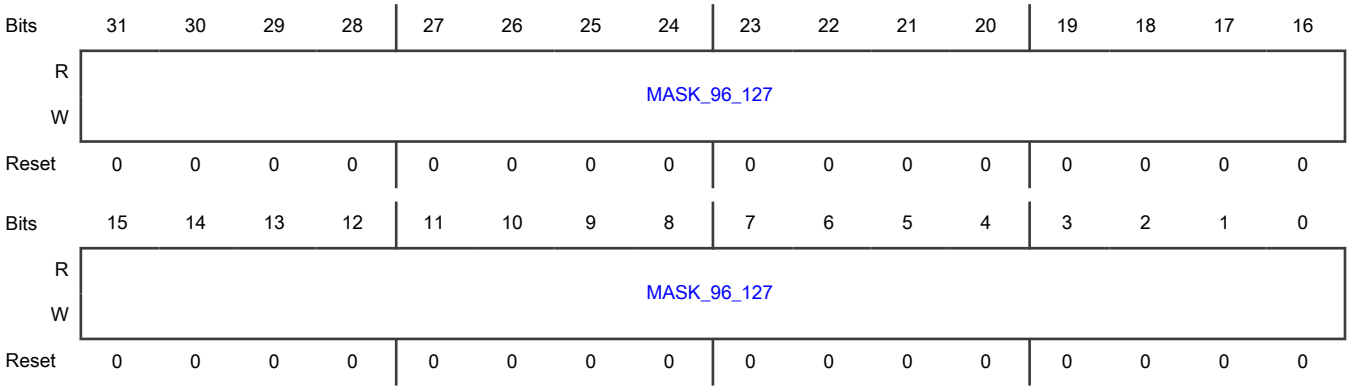
Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_MASK_3	10Ch
CPU1_CM_IRQ_WAKEUP_MASK_3	90Ch

Function

IRQ96~127 mask register.

Diagram



Fields

Field	Function
31-0 MASK_96_127	"1" means the IRQ cannot wakeup CPU platform

21.6.1.10 CM IRQ128~159 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_4 - CPU1_CM_IRQ_WAKEUP_MASK_4)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_MASK_4	110h
CPU1_CM_IRQ_WAKEUP_MASK_4	910h

Function

IRQ128~159 mask register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MASK_128_159															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MASK_128_159															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MASK_128_159	"1" means the IRQ cannot wakeup CPU platform

21.6.1.11 CM IRQ160~191 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_5 - CPU1_CM_IRQ_WAKEUP_MASK_5)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_MASK_5	114h
CPU1_CM_IRQ_WAKEUP_MASK_5	914h

Function

IRQ160~191 mask register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MASK_160_191															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MASK_160_191															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MASK_160_191	"1" means the IRQ cannot wakeup CPU platform

21.6.1.12 CM IRQ192~223 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_6 - CPU1_CM_IRQ_WAKEUP_MASK_6)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_MASK_6	118h
CPU1_CM_IRQ_WAKEUP_MASK_6	918h

Function
IRQ192~223 mask register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MASK_192_223															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MASK_192_223															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MASK_192_223	"1" means the IRQ cannot wakeup CPU platform

21.6.1.13 CM IRQ224~255 wakeup mask (CPU0_CM_IRQ_WAKEUP_MASK_7 - CPU1_CM_IRQ_WAKEUP_MASK_7)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_MASK_7	11Ch
CPU1_CM_IRQ_WAKEUP_MASK_7	91Ch

Function
IRQ224~255 mask register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MASK_224_255															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MASK_224_255															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MASK_224_255	"1" means the IRQ cannot wakeup CPU platform

21.6.1.14 CM non-IRQ wakeup mask (CPU0_CM_NON_IRQ_WAKEUP_MASK - CPU1_CM_NON_IRQ_WAKEUP_MASK)

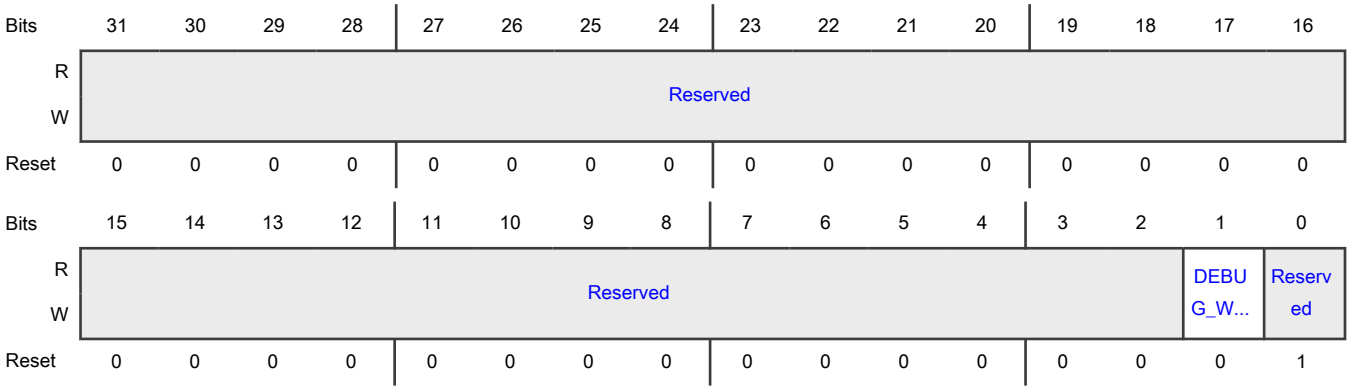
Offset

Register	Offset
CPU0_CM_NON_IRQ_WAKEUP_MASK	140h
CPU1_CM_NON_IRQ_WAKEUP_MASK	940h

Function

This register is used to mask wakeup events which are not interrupt.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 DEBUG_WAKEUP_MASK	"1" means the debug_wakeup_request cannot wakeup CPU platform
0 —	Reserved

21.6.1.15 CM IRQ0~31 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_0 - CPU1_CM_IRQ_WAKEUP_STAT_0)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_STAT_0	150h
CPU1_CM_IRQ_WAKEUP_STAT_0	950h

Function

Interrupt 0~31 status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STAT_0_31															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STAT_0_31															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STAT_0_31	IRQ status A bit set to 1 means the corresponding IRQ has a valid pending interrupt to wakeup the CPU platform.

21.6.1.16 CM IRQ32~63 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_1 - CPU1_CM_IRQ_WAKEUP_STAT_1)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_STAT_1	154h
CPU1_CM_IRQ_WAKEUP_STAT_1	954h

Function

Interrupt 32~63 status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STAT_32_63															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STAT_32_63															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STAT_32_63	IRQ status A bit set to 1 means the corresponding IRQ has a valid pending interrupt to wakeup the CPU platform.

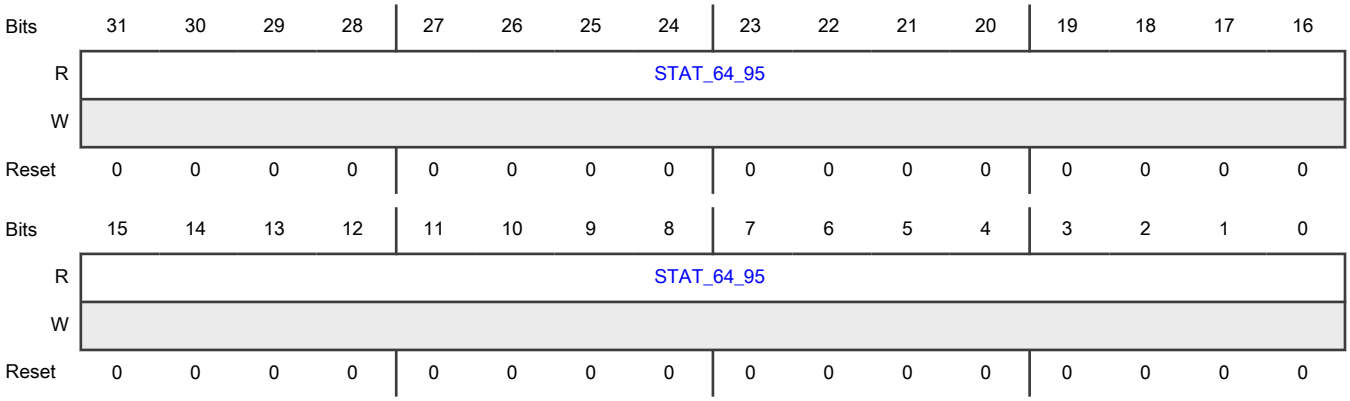
21.6.1.17 CM IRQ64~95 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_2 - CPU1_CM_IRQ_WAKEUP_STAT_2)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_STAT_2	158h
CPU1_CM_IRQ_WAKEUP_STAT_2	958h

Function
Interrupt 64~95 status.

Diagram



Fields

Field	Function
31-0 STAT_64_95	IRQ status A bit set to 1 means the corresponding IRQ has a valid pending interrupt to wakeup the CPU platform.

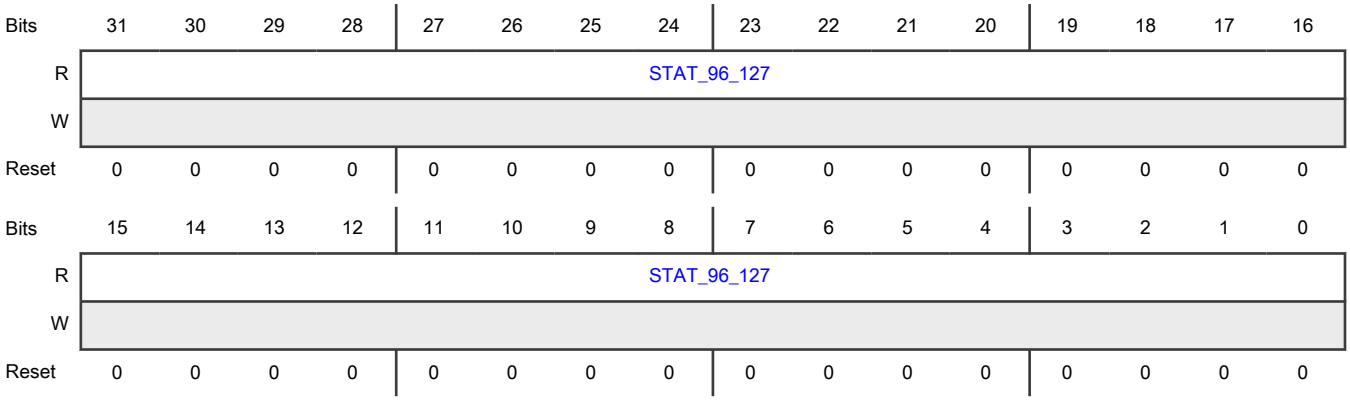
21.6.1.18 CM IRQ96~127 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_3 - CPU1_CM_IRQ_WAKEUP_STAT_3)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_STAT_3	15Ch
CPU1_CM_IRQ_WAKEUP_STAT_3	95Ch

Function
Interrupt 96~127 status.

Diagram



Fields

Field	Function
31-0 STAT_96_127	IRQ status A bit set to 1 means the corresponding IRQ has a valid pending interrupt to wakeup the CPU platform.

21.6.1.19 CM IRQ128~159 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_4 - CPU1_CM_IRQ_WAKEUP_STAT_4)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_STAT_4	160h
CPU1_CM_IRQ_WAKEUP_STAT_4	960h

Function

Interrupt 128~159 status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STAT_128_159															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STAT_128_159															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STAT_128_159	IRQ status A bit set to 1 means the corresponding IRQ has a valid pending interrupt to wakeup the CPU platform.

21.6.1.20 CM IRQ160~191 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_5 - CPU1_CM_IRQ_WAKEUP_STAT_5)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_STAT_5	164h
CPU1_CM_IRQ_WAKEUP_STAT_5	964h

Function

Interrupt 160~191 status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STAT_160_191															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STAT_160_191															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STAT_160_191	IRQ status A bit set to 1 means the corresponding IRQ has a valid pending interrupt to wakeup the CPU platform.

21.6.1.21 CM IRQ192~223 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_6 - CPU1_CM_IRQ_WAKEUP_STAT_6)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_STAT_6	168h
CPU1_CM_IRQ_WAKEUP_STAT_6	968h

Function

Interrupt 192~223 status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STAT_192_223															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STAT_192_223															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STAT_192_223	IRQ status A bit set to 1 means the corresponding IRQ has a valid pending interrupt to wakeup the CPU platform.

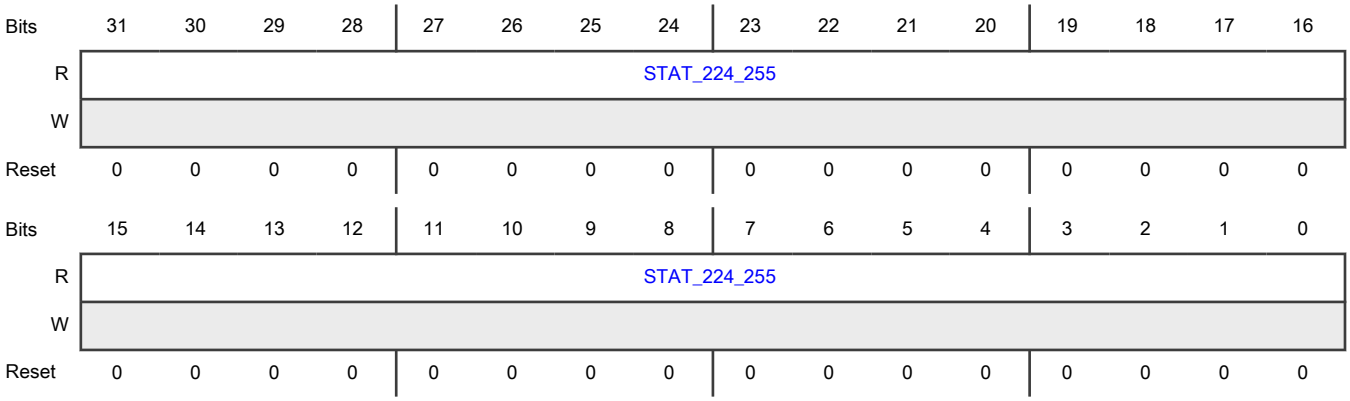
21.6.1.22 CM IRQ224~255 wakeup status (CPU0_CM_IRQ_WAKEUP_STAT_7 - CPU1_CM_IRQ_WAKEUP_STAT_7)

Offset

Register	Offset
CPU0_CM_IRQ_WAKEUP_STAT_7	16Ch
CPU1_CM_IRQ_WAKEUP_STAT_7	96Ch

Function
Interrupt 224~255 status.

Diagram



Fields

Field	Function
31-0 STAT_224_255	IRQ status A bit set to 1 means the corresponding IRQ has a valid pending interrupt to wakeup the CPU platform.

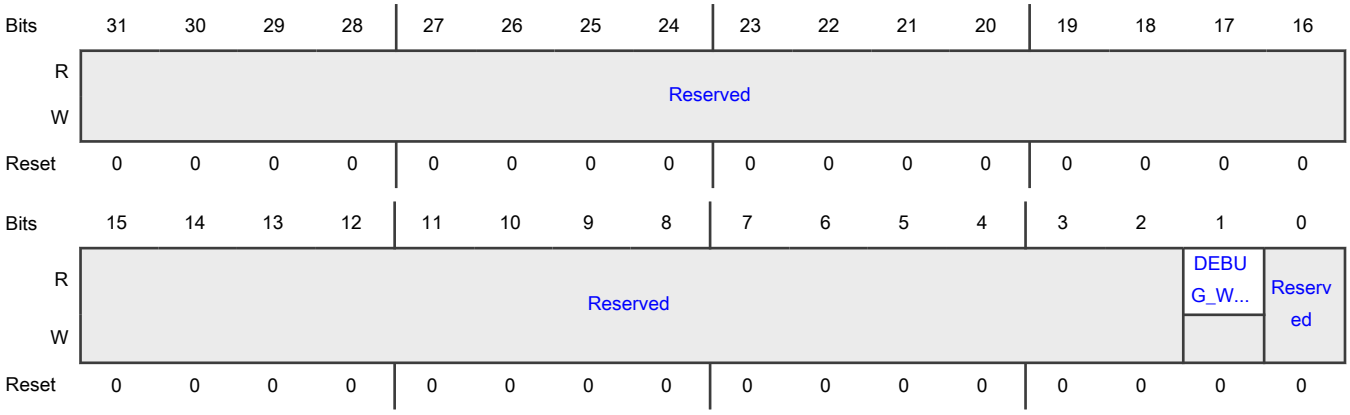
21.6.1.23 CM non-irq wakeup status (CPU0_CM_NON_IRQ_WAKEUP_STAT - CPU1_CM_NON_IRQ_WAKEUP_STAT)

Offset

Register	Offset
CPU0_CM_NON_IRQ_WAKEUP_STAT	190h
CPU1_CM_NON_IRQ_WAKEUP_STAT	990h

Function
Wakeup event status.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 DEBUG_WAKEUP_STAT	Debug wakeup status 0b - No debug wakeup is requested 1b - Debug wakeup is requested
0 —	Reserved

21.6.1.24 CM sleep SSAR control (CPU0_CM_SLEEP_SSAR_CTRL - CPU1_CM_SLEEP_SSAR_CTRL)

Offset

Register	Offset
CPU0_CM_SLEEP_SSAR_CTRL	200h
CPU1_CM_SLEEP_SSAR_CTRL	A00h

Function
Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.1.25 CM sleep LPCG control (CPU0_CM_SLEEP_LPCG_CTRL - CPU1_CM_SLEEP_LPCG_CTRL)

Offset

Register	Offset
CPU0_CM_SLEEP_LPCG_CTRL	208h
CPU1_CM_SLEEP_LPCG_CTRL	A08h

Function
Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.1.26 CM sleep PLL control (CPU0_CM_SLEEP_PLL_CTRL - CPU1_CM_SLEEP_PLL_CTRL)

Offset

Register	Offset
CPU0_CM_SLEEP_PLL_CTRL	210h
CPU1_CM_SLEEP_PLL_CTRL	A10h

Function
Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.1.27 CM sleep isolation control (CPU0_CM_SLEEP_ISO_CTRL - CPU1_CM_SLEEP_ISO_CTRL)

Offset

Register	Offset
CPU0_CM_SLEEP_ISO_CTRL	218h
CPU1_CM_SLEEP_ISO_CTRL	A18h

Function
Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.1.28 CM sleep reset control (CPU0_CM_SLEEP_RESET_CTRL - CPU1_CM_SLEEP_RESET_CTRL)

Offset

Register	Offset
CPU0_CM_SLEEP_RESET_CTRL	220h
CPU1_CM_SLEEP_RESET_CTRL	A20h

Function
Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.1.29 CM sleep power control (CPU0_CM_SLEEP_POWER_CTRL - CPU1_CM_SLEEP_POWER_CTRL)

Offset

Register	Offset
CPU0_CM_SLEEP_POWER_CTRL	228h
CPU1_CM_SLEEP_POWER_CTRL	A28h

Function
Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.1.30 CM wakeup power control (CPU0_CM_WAKEUP_POWER_CTRL - CPU1_CM_WAKEUP_POWER_CTRL)

Offset

Register	Offset
CPU0_CM_WAKEUP_POWER_CTRL	290h
CPU1_CM_WAKEUP_POWER_CTRL	A90h

Function

Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.1.31 CM wakeup reset control (CPU0_CM_WAKEUP_RESET_CTRL - CPU1_CM_WAKEUP_RESET_CTRL)

Offset

Register	Offset
CPU0_CM_WAKEUP_RESET_CTRL	298h
CPU1_CM_WAKEUP_RESET_CTRL	A98h

Function

Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

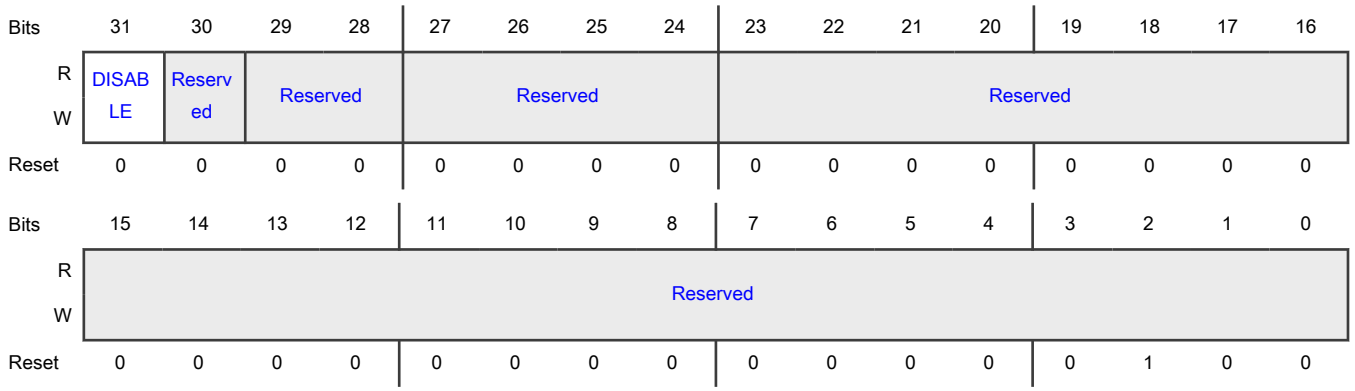
21.6.1.32 CM wakeup isolation control (CPU0_CM_WAKEUP_ISO_CTRL - CPU1_CM_WAKEUP_ISO_CTRL)

Offset

Register	Offset
CPU0_CM_WAKEUP_ISO_CTRL	2A0h
CPU1_CM_WAKEUP_ISO_CTRL	AA0h

Function
Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

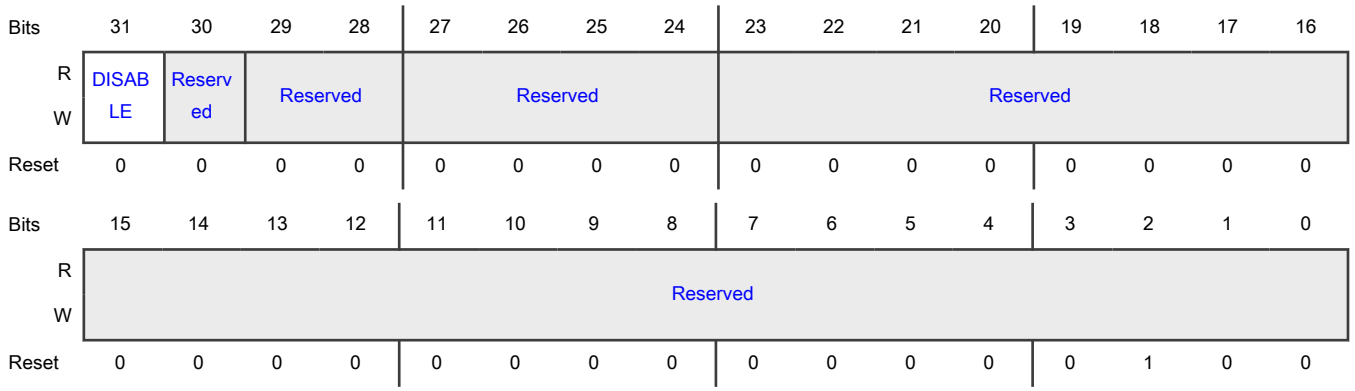
21.6.1.33 CM wakeup PLL control (CPU0_CM_WAKEUP_PLL_CTRL - CPU1_CM_WAKEUP_PLL_CTRL)

Offset

Register	Offset
CPU0_CM_WAKEUP_PLL_CTRL	2A8h
CPU1_CM_WAKEUP_PLL_CTRL	AA8h

Function
Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.1.34 CM wakeup LPCG control (CPU0_CM_WAKEUP_LPCG_CTRL - CPU1_CM_WAKEUP_LPCG_CTRL)

Offset

Register	Offset
CPU0_CM_WAKEUP_LPCG_CTRL	2B0h
CPU1_CM_WAKEUP_LPCG_CTRL	AB0h

Function
Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.1.35 CM wakeup SSAR control (CPU0_CM_WAKEUP_SSAR_CTRL - CPU1_CM_WAKEUP_SSAR_CTRL)

Offset

Register	Offset
CPU0_CM_WAKEUP_SSAR_CTRL	2C0h
CPU1_CM_WAKEUP_SSAR_CTRL	AC0h

Function
Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.1.36 CM system sleep control (CPU0_CM_SYS_SLEEP_CTRL - CPU1_CM_SYS_SLEEP_CTRL)

Offset

Register	Offset
CPU0_CM_SYS_SLEEP_CTRL	380h
CPU1_CM_SYS_SLEEP_CTRL	B80h

Function
Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved														Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved													SS_SUSP...	SS_STOP	SS_WAIT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-18 —	Reserved
17 —	Reserved
16 —	Reserved
15-3 —	Reserved
2 SS_SUSPEND	Request system sleep when CPU is in SUSPEND mode 0b - Do not request system sleep when CPU is in SUSPEND mode 1b - Request system sleep when CPU is in SUSPEND mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 SS_STOP	Request system sleep when CPU is in STOP mode 0b - Do not request system sleep when CPU is in STOP mode 1b - Request system sleep when CPU is in STOP mode
0 SS_WAIT	Request system sleep when CPU is in WAIT mode 0b - Do not request system sleep when CPU is in WAIT mode 1b - Request system sleep when CPU is in WAIT mode

21.6.2 register descriptions

21.6.2.1 gpc_global memory map

GPC_GLOBAL base address: 4447_2000h

Offset	Register	Width (In bits)	Access	Reset value
4h	GPC Global Authentication Control (AUTHEN_CTRL)	32	RW	FFFF_0000h
10h	GPC CPU0 domain assignment (GPC_CPU0_DOMAIN)	32	RW	0000_0000h
14h	GPC CPU1 domain assignment (GPC_CPU1_DOMAIN)	32	RW	0000_0000h
24h	GPC master CPU configuration (GPC_MASTER)	32	RW	0000_0000h
200h	RCOSC control (GPC_ROSC_CTRL)	32	RW	0000_0000h

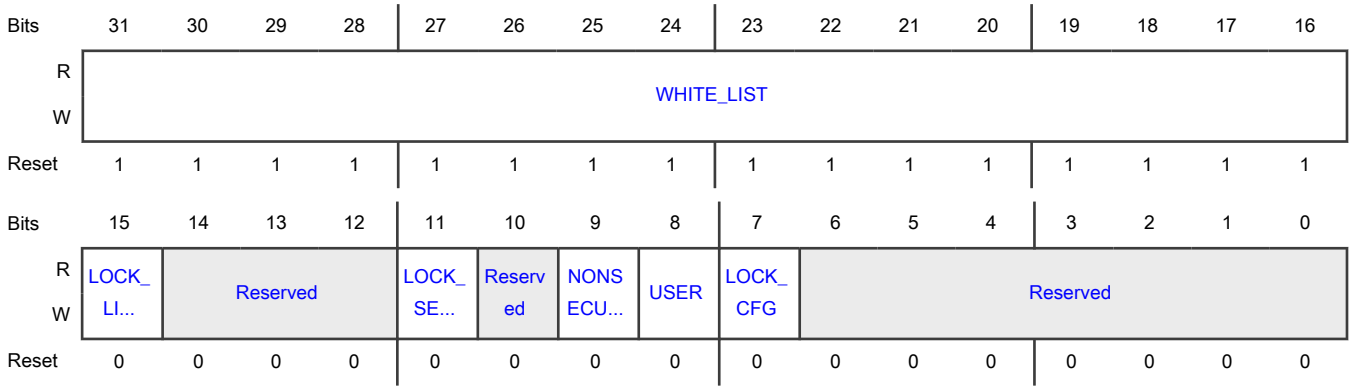
21.6.2.2 GPC Global Authentication Control (AUTHEN_CTRL)

Offset

Register	Offset
AUTHEN_CTRL	4h

Function
Authentication control for GPC GLOBAL.

Diagram



Fields

Field	Function
31-16 WHITE_LIST	Domain ID white list When bit is set to 1, the corresponding domain ID can access CPU mode control registers. bit[0] for Domain 0, bit[1] for Domain 1, bit[2] for Domain 2, and bit[3] for Domain 3. Multiple bits with a value of 1 is permitted.
15 LOCK_LIST	White list lock This bit locks the WHITE_LIST. When this bit is set, WHITE_LIST cannot be changed. Once this bit is set, it cannot be cleared until the next system reset. 0b - WHITE_LIST is not locked 1b - WHITE_LIST is locked
14-12 —	Reserved
11 LOCK_SETTING	Lock NONSECURE and USER This bit locks the NONSECURE and USER fields. When this bit is set, LOCK_SETTING cannot be changed. Once this bit is set, it cannot be cleared until the next system reset. 0b - NONSECURE and USER fields are not locked 1b - NONSECURE and USER fields are locked
10 —	Reserved
9 NONSECURE	Allow non-secure mode access 0b - Allow only secure mode to access CPU mode registers 1b - Allow both secure and non-secure mode to access CPU mode control registers.
8 USER	Allow user mode access 0b - Allow only privilege mode to access CPU mode control registers

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Allow both privilege and user mode to access CPU mode control registers
7 LOCK_CFG	Configuration lock This bit locks the value of low power configuration fields. Once this bit is set, it cannot be cleared until the next system reset. 0b - The value of low power configuration fields are not locked. 1b - The value of low power configuration fields are locked. Refer to the function field of each gpc_global registers.
6-0 —	Reserved

21.6.2.3 GPC CPU0 domain assignment (GPC_CPU0_DOMAIN)

Offset

Register	Offset
GPC_CPU0_DOMAIN	10h

Function
Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CPU0_DOMAIN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 CPU0_DOMAIN	CPU0 domain assignment

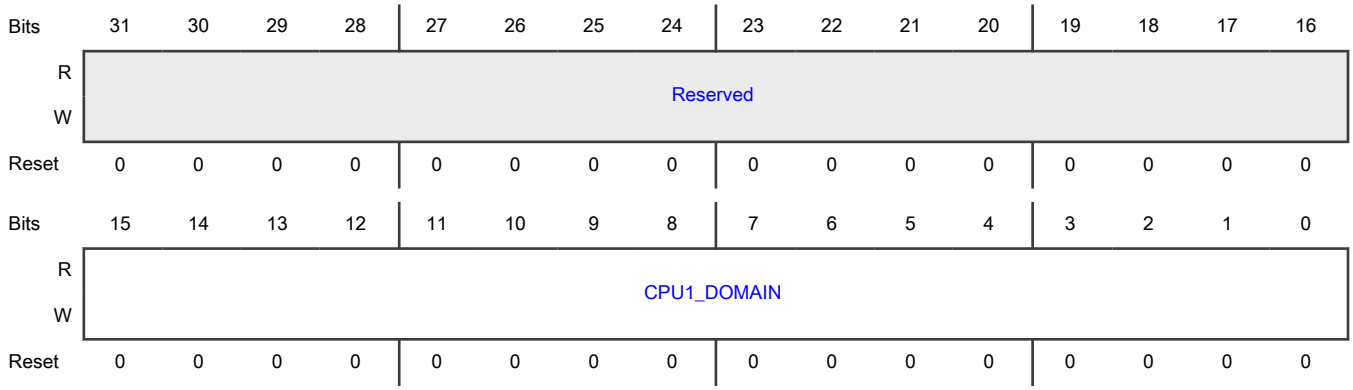
21.6.2.4 GPC CPU1 domain assignment (GPC_CPU1_DOMAIN)

Offset

Register	Offset
GPC_CPU1_DOMAIN	14h

Function
Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 CPU1_DOMAIN	CPU1 domain assignment

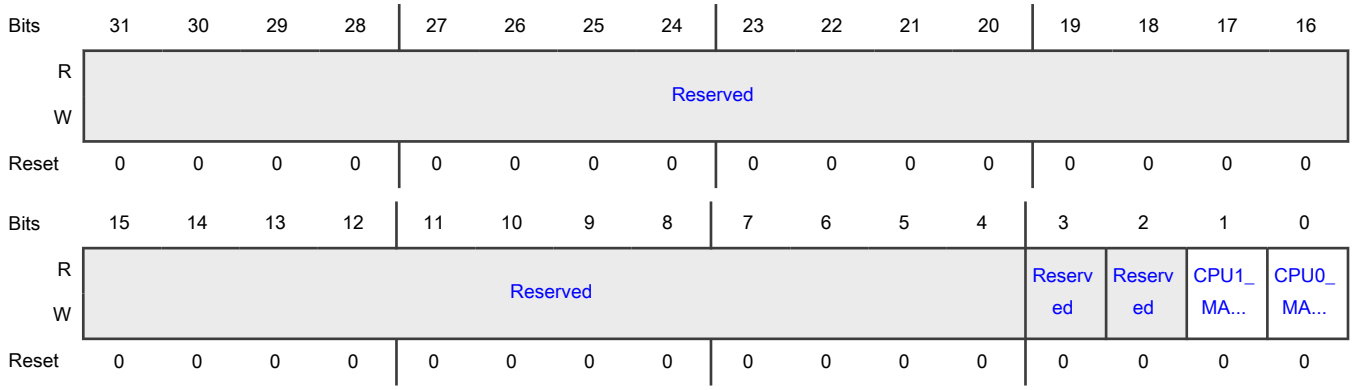
21.6.2.5 GPC master CPU configuration (GPC_MASTER)

Offset

Register	Offset
GPC_MASTER	24h

Function
Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 —	Reserved
2 —	Reserved
1 CPU1_MASTER	Setting to 1 means CPU1 is the master CPU of its domain 0b - CPU1 is not the master CPU of its domain 1b - CPU1 is the master CPU of its domain
0 CPU0_MASTER	Setting to 1 means CPU0 is the master CPU of its domain 0b - CPU0 is not the master CPU of its domain 1b - CPU0 is the master CPU of its domain

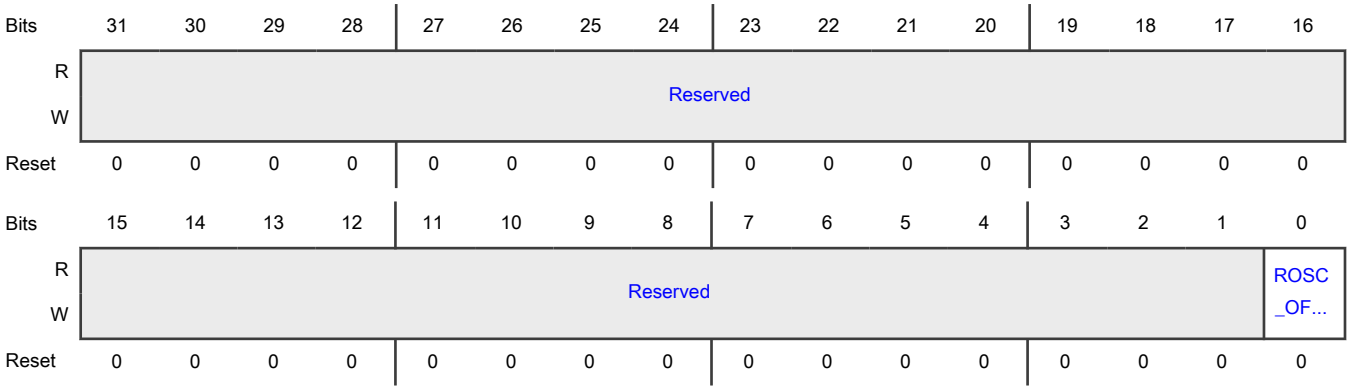
21.6.2.6 RCOSC control (GPC_ROSC_CTRL)

Offset

Register	Offset
GPC_ROSC_CTRL	200h

Function
Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 ROSC_OFF_EN	Shut off the 24 MHz RCOSC clock when system sleep 0b - Keep 24 MHz ROSC clock running during system sleep 1b - Shut off 24 MHz ROSC clock during system sleep

21.6.3 register descriptions

21.6.3.1 gpc_sys_sleep_ctrl memory map

GPC_SYS_SLEEP_CTRL base address: 4447_2800h

Offset	Register	Width (In bits)	Access	Reset value
4h	System Sleep Authentication Control (SS_AUTHEN_CTRL)	32	RW	0000_0000h
Ch	System Sleep Misc (SS_MISC)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
40h	PMIC standby control (PMIC_CTRL)	32	RW	0000_0006h
F0h	System Sleep STEP0 (BIAS) in control (SS_STEP0_IN_CTRL)	32	RW	0000_0004h
100h	System Sleep STEP1 (PLDO) in control (SS_STEP1_IN_CTRL)	32	RW	0000_0004h
110h	System Sleep STEP2 (BANDGAP) in control (SS_STEP2_IN_CTRL)	32	RW	0000_0004h
120h	System Sleep STEP3 (LDO) in control (SS_STEP3_IN_CTRL)	32	RW	0000_0004h
140h	System Sleep DCDC in control (SS_DCDC_IN_CTRL)	32	RW	0000_0004h
150h	System Sleep PMIC in control (SS_PMIC_IN_CTRL)	32	RW	0000_0004h
200h	System Sleep PMIC out control (SS_PMIC_OUT_CTRL)	32	RW	0000_0004h
210h	System Sleep DCDC out control (SS_DCDC_OUT_CTRL)	32	RW	0000_0004h
238h	System Sleep STEP3 (LDO) out control (SS_STEP3_OUT_CTRL)	32	RW	0000_0004h
240h	System Sleep STEP2 (BANDGAP) out control (SS_STEP2_OUT_CTRL)	32	RW	0000_0004h
250h	System Sleep STEP1 (PLDO) out control (SS_STEP1_OUT_CTRL)	32	RW	0000_0004h
260h	System Sleep STEP0 (BIAS) out control (SS_STEP0_OUT_CTRL)	32	RW	0000_0004h

21.6.3.2 System Sleep Authentication Control (SS_AUTHEN_CTRL)

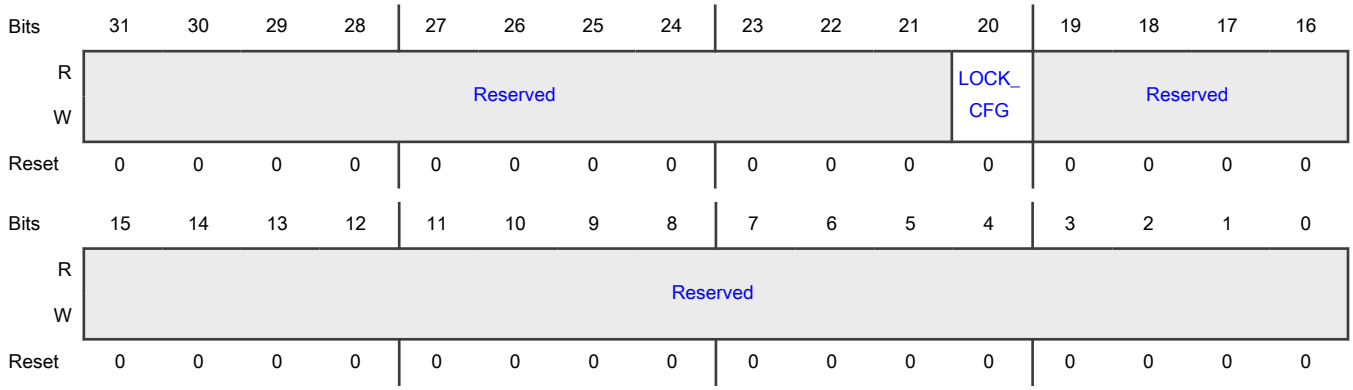
Offset

Register	Offset
SS_AUTHEN_CTRL	4h

Function

Authentication control for the system sleep controller.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 LOCK_CFG	Configuration lock This bit locks the value of low power configuration fields. Once this bit is set, it cannot be cleared until the next system reset. 0b - The value of low power configuration fields are not locked. 1b - The value of low power configuration fields are locked. Refer to the function field of each gpc_sys_sleep_ctrl registers.
19-0 —	Reserved

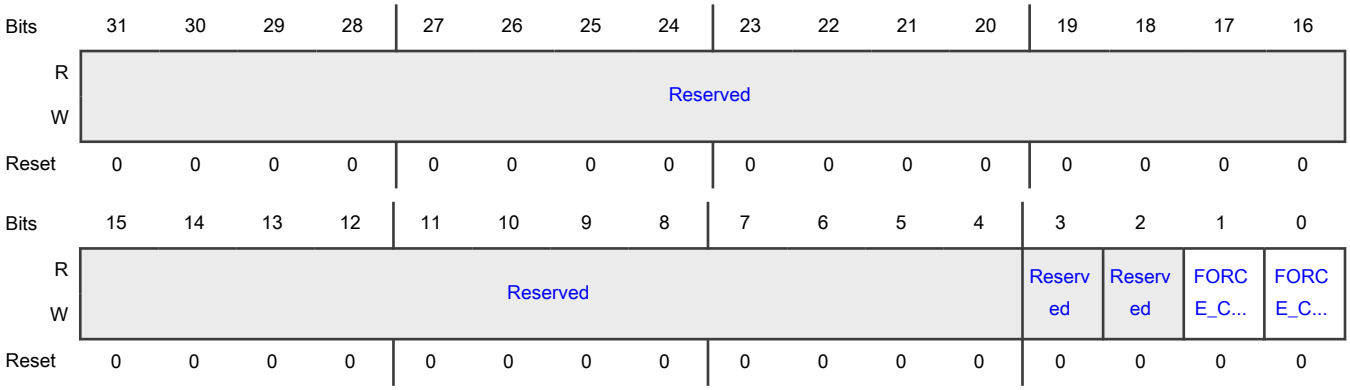
21.6.3.3 System Sleep Misc (SS_MISC)

Offset

Register	Offset
SS_MISC	Ch

Function
Miscellaneous register.

Diagram



Fields

Field	Function
31-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3 —	Reserved
2 —	Reserved
1 FORCE_CPU1_SYS_SLEEP	Force CPU1 to request system sleep mode If CPU1 exists but is not used and is always in reset status, users can set this bit to force CPU1 to request system sleep mode 0b - Do not force CPU1 to request system sleep mode 1b - Force CPU1 to request system sleep mode
0 FORCE_CPU0_SYS_SLEEP	Force CPU0 to request system sleep mode If CPU0 exists but is not used and is always in reset status, users can set this bit to force CPU0 to request system sleep mode 0b - Do not force CPU0 to request system sleep mode 1b - Force CPU0 to request system sleep mode

21.6.3.4 PMIC standby control (PMIC_CTRL)

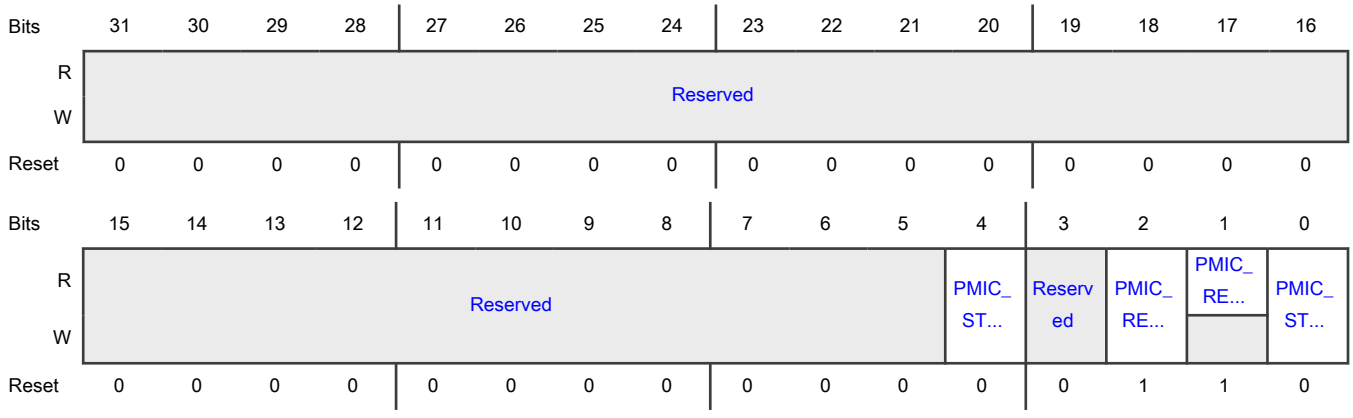
Offset

Register	Offset
PMIC_CTRL	40h

Function

PMIC standby control register. Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 PMIC_STBY_SOFT	Software PMIC standby trigger 0b - Exit PMIC standby 1b - Trigger PMIC standby
3 —	Reserved
2 PMIC_READY_EXIST	PMIC_READY is driven from pad If 0, SS_PMIC_IN_CTRL and SS_PMIC_OUT_CTRL's CNT_MODE can only be 2 or 3. 0b - PMIC_READY is not driven. 1b - PMIC_READY is driven from pad. If PMIC_READY_EXIST = 1, PMIC_READY = 1 means external PMIC drives PMIC_READY pin.
1 PMIC_READY	PMIC_READY pin status 0b - PMIC_READY not asserted 1b - PMIC_READY asserted
0 PMIC_STBY_EN	Assert the PMIC standby request when system sleep 0b - Do not assert PMIC_STBY_REQ when system sleep is entered 1b - Assert PMIC_STBY_REQ when system sleep is entered

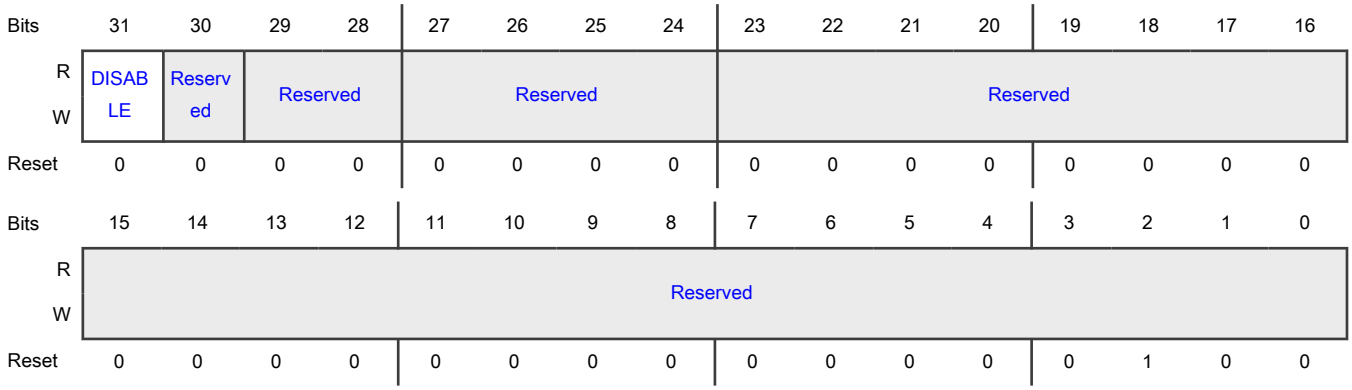
21.6.3.5 System Sleep STEP0 (BIAS) in control (SS_STEP0_IN_CTRL)

Offset

Register	Offset
SS_STEP0_IN_CTRL	F0h

Function
BIAS_IN step control. Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.3.6 System Sleep STEP1 (PLDO) in control (SS_STEP1_IN_CTRL)

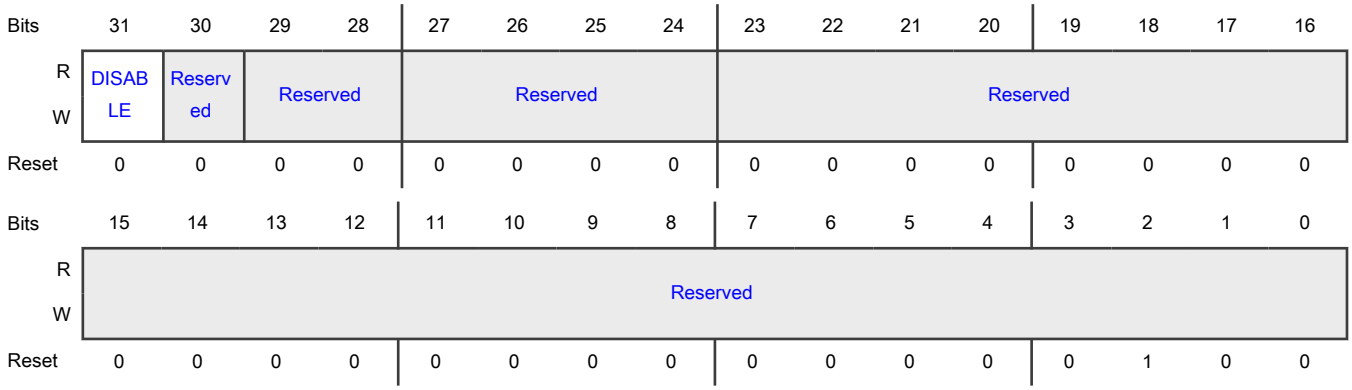
Offset

Register	Offset
SS_STEP1_IN_CTRL	100h

Function

PLDO_IN step control. Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.3.7 System Sleep STEP2 (BANDGAP) in control (SS_STEP2_IN_CTRL)

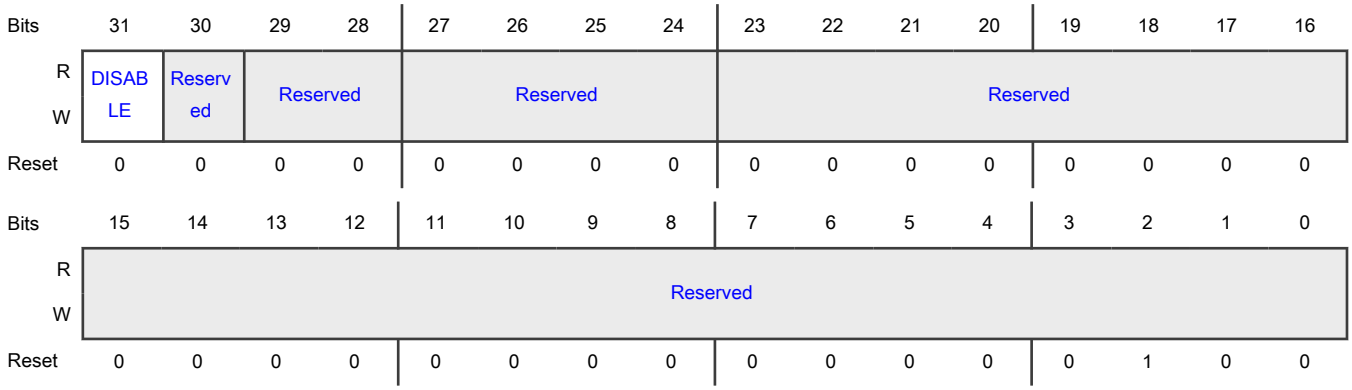
Offset

Register	Offset
SS_STEP2_IN_CTRL	110h

Function

BANDGAP_IN step control. Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.3.8 System Sleep STEP3 (LDO) in control (SS_STEP3_IN_CTRL)

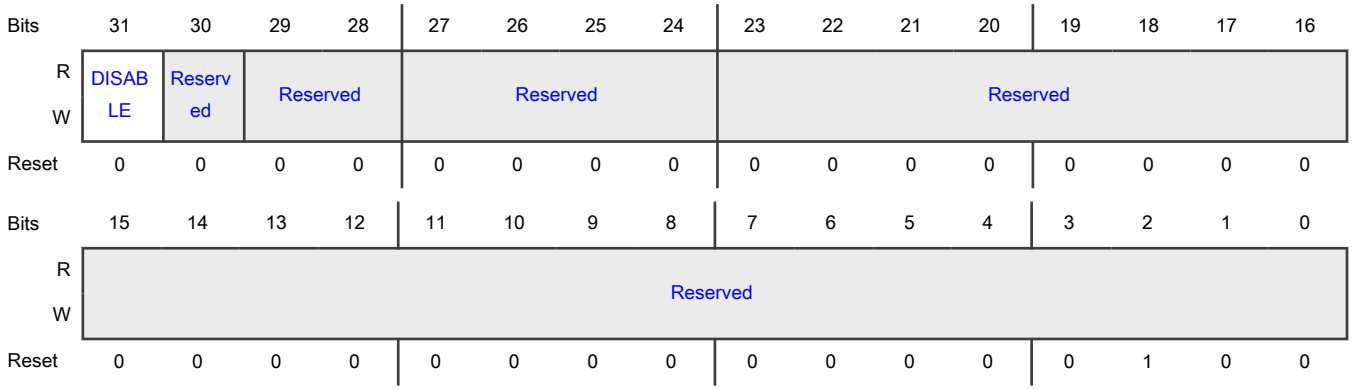
Offset

Register	Offset
SS_STEP3_IN_CTRL	120h

Function

LDO_IN step control. Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.3.9 System Sleep DCDC in control (SS_DCDC_IN_CTRL)

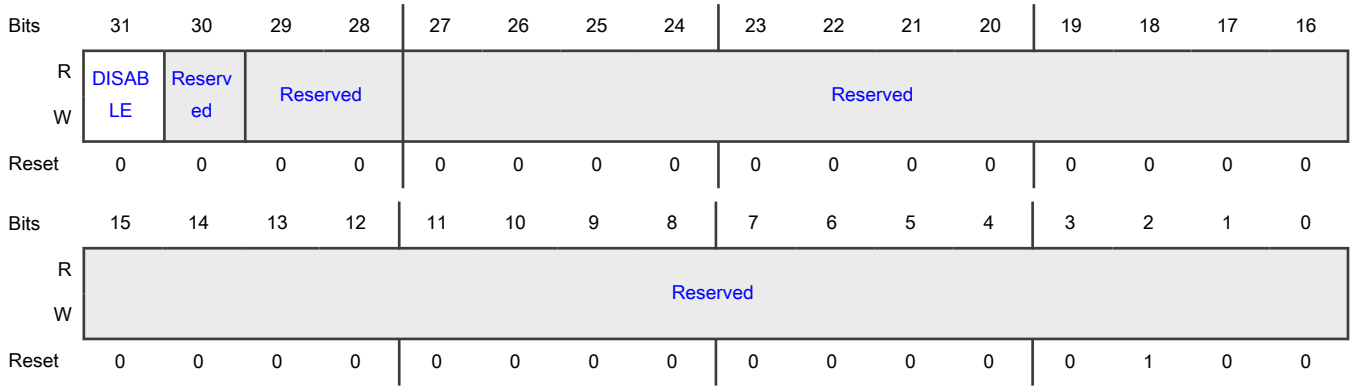
Offset

Register	Offset
SS_DCDC_IN_CTRL	140h

Function

DCDC_IN step control. Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-16 —	Reserved
15-0 —	Reserved

21.6.3.10 System Sleep PMIC in control (SS_PMIC_IN_CTRL)

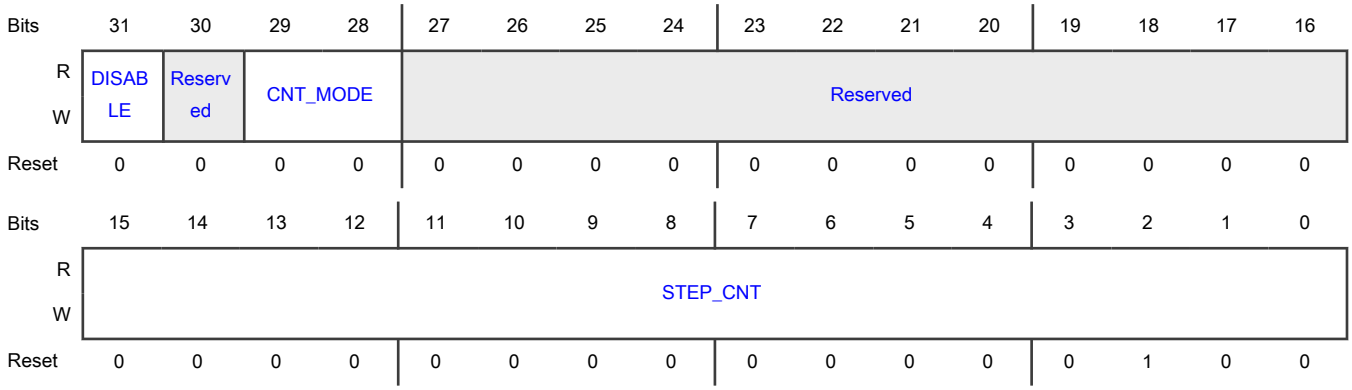
Offset

Register	Offset
SS_PMIC_IN_CTRL	150h

Function

PMIC_IN step control. Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 CNT_MODE	Count mode Configure the step counter working mode 00b - Counter disable mode: not use step counter, step completes once receiving step_done 01b - Counter delay mode: delay after receiving step_done, delay cycle number is STEP_CNT 10b - Ignore step_done response, the counter starts to count once step begins, when counter reaches STEP_CNT value, the step completes 11b - Time out mode, the counter starts to count once step begins, the step completes when either step_done received or counting to STEP_CNT value
27-16 —	Reserved
15-0 STEP_CNT	Step count, useage depends on CNT_MODE Must be greater than 2 under CNT_MODE 2 and 3

21.6.3.11 System Sleep PMIC out control (SS_PMIC_OUT_CTRL)

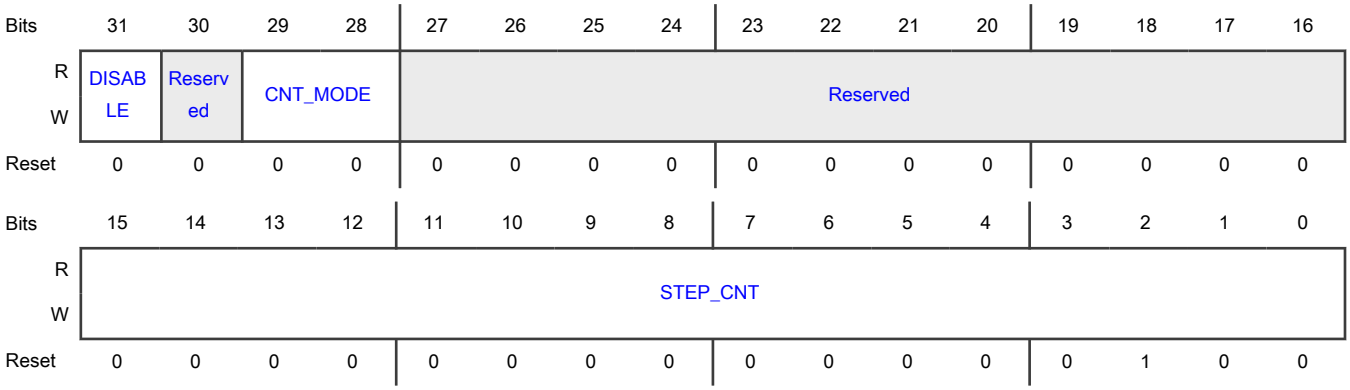
Offset

Register	Offset
SS_PMIC_OUT_CTRL	200h

Function

PMIC_OUT step control. Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 CNT_MODE	Count mode Configure the step counter working mode 00b - Counter disable mode: not use step counter, step completes once receiving step_done 01b - Counter delay mode: delay after receiving step_done, delay cycle number is STEP_CNT 10b - Ignore step_done response, the counter starts to count once step begins, when counter reaches STEP_CNT value, the step completes 11b - Time out mode, the counter starts to count once step begins, the step completes when either step_done received or counting to STEP_CNT value
27-16 —	Reserved
15-0 STEP_CNT	Step count, useage depends on CNT_MODE Must be greater than 2 under CNT_MODE 2 and 3

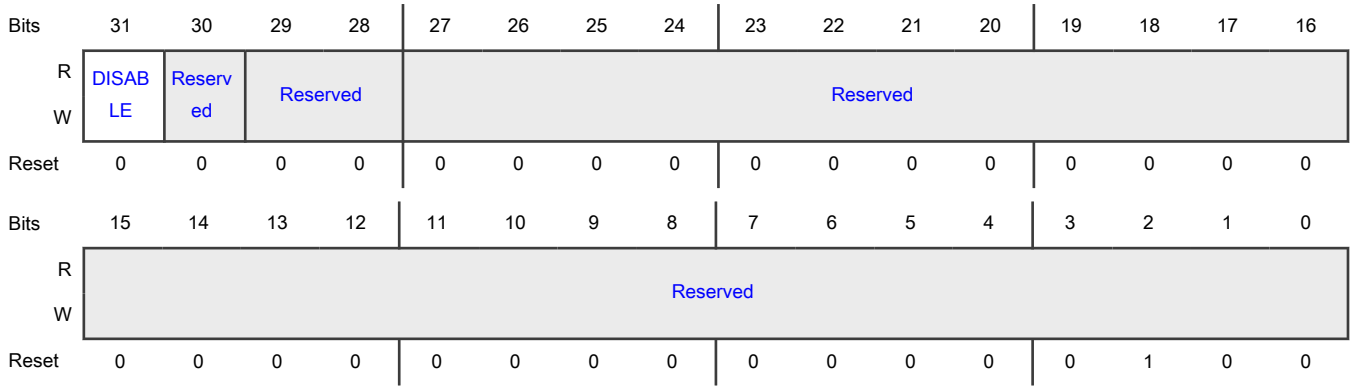
21.6.3.12 System Sleep DCDC out control (SS_DCDC_OUT_CTRL)

Offset

Register	Offset
SS_DCDC_OUT_CTRL	210h

Function
DCDC_OUT step control. Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-16 —	Reserved
15-0 —	Reserved

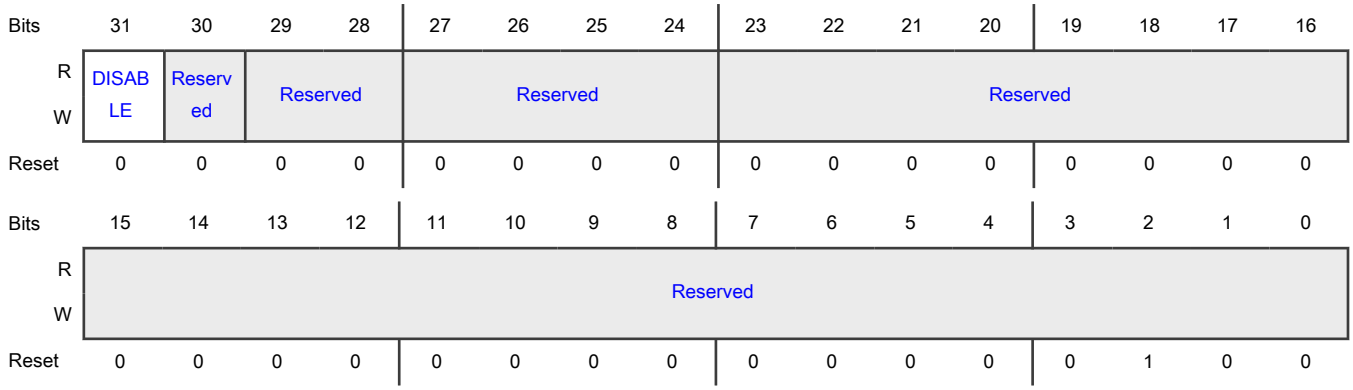
21.6.3.13 System Sleep STEP3 (LDO) out control (SS_STEP3_OUT_CTRL)

Offset

Register	Offset
SS_STEP3_OUT_CTRL	238h

Function
LDO_OUT step control. Locked by LOCK_CFG field.

Diagram



Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.3.14 System Sleep STEP2 (BANDGAP) out control (SS_STEP2_OUT_CTRL)

Offset

Register	Offset
SS_STEP2_OUT_CTRL	240h

Function

BANDGAP_OUT step control. Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.3.15 System Sleep STEP1 (PLDO) out control (SS_STEP1_OUT_CTRL)

Offset

Register	Offset
SS_STEP1_OUT_CTRL	250h

Function

PLDO_OUT step control. Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

21.6.3.16 System Sleep STEP0 (BIAS) out control (SS_STEP0_OUT_CTRL)

Offset

Register	Offset
SS_STEP0_OUT_CTRL	260h

Function

BIAS_OUT step control. Locked by LOCK_CFG field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DISABLE	Reserved	Reserved		Reserved				Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 DISABLE	Disable this step When set to 1, GPC will skip this step and not send any request 0b - This step is enabled. 1b - This step is disabled. GPC will skip this step and not send any request.
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-0 —	Reserved

Chapter 22
DCDC Converter (DCDC)

22.1 Chip-specific DCDC Information

Table 150. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

22.2 Overview

The DCDC Converter(DCDC) is a synchronous buck mode DCDC converter, which is used for generating the power supply for chip's core logic. It also includes a LDO which outputs 1.8V to supply for the analog module of DCDC itself and some of the other 1.8V domain blocks of the chip.

22.2.1 Block diagram

The block diagram of the DCDC module is provided below:

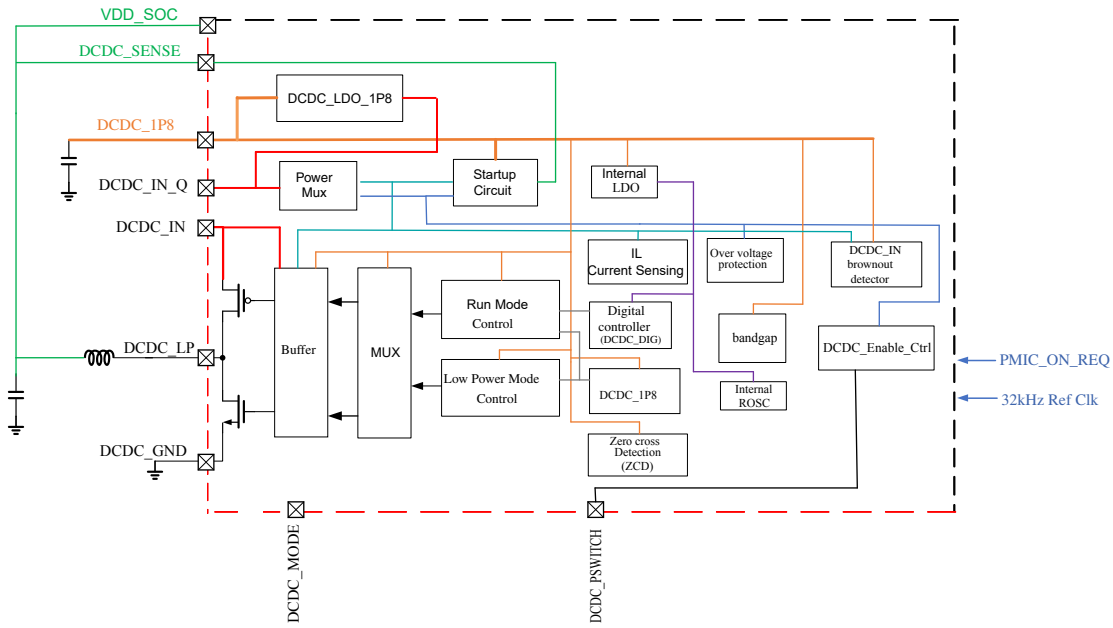


Figure 82. DCDC block diagram

22.2.2 Features

The DCDC module includes the following features:

- Buck mode only, 3.3V input with 1.0V output
- LDO with 3.3V input and 1.8V output
- DCDC_PSWITCH pin to turn on/off DCDC
- Continuous conduction mode (CCM) operation
- Automatic pulse skipping
- RUN mode and LP (low-power) mode
- RUN mode supports CCM
- Overcurrent and overvoltage protection
- Low-input voltage detection
- Low-input voltage warning

22.3 Functional description

The DCDC converter is a synchronous step-down converter with one output. In run mode, it can operate on continuous conduction mode or discontinuous conduction mode, depending on the level of the load current. Discontinuous conduction mode needs to be enabled through register configuration.

DCDC can be configured to operate in Low Power (LP) mode when the load current is less than 50mA. During the LP mode, the converter operates in pulse mode with a minimum quiescent current to maintain high efficiency.

22.3.1 Application Modes

DCDC has only one application mode, which is configured by DCDC_MODE:

- Application Mode 0: DCDC_IN is connected to 3.3V, DCDC outputs 1.0V and LDO output 1.8V. DCDC_MODE is tied to ground

22.3.2 Functional Modes

- Run Mode
  - Continuous Conduction Mode
  - Discontinuous Conduction Mode - improved efficiency and same working state as Continuous Conduction Mode
- Low-Power Mode
  - Improved efficiency
  - Ripple is a little bit higher than Run Mode

22.3.3 Operation

The DCDC is a synchronous buck converter. An illustration of the power switch arrangement is shown below.

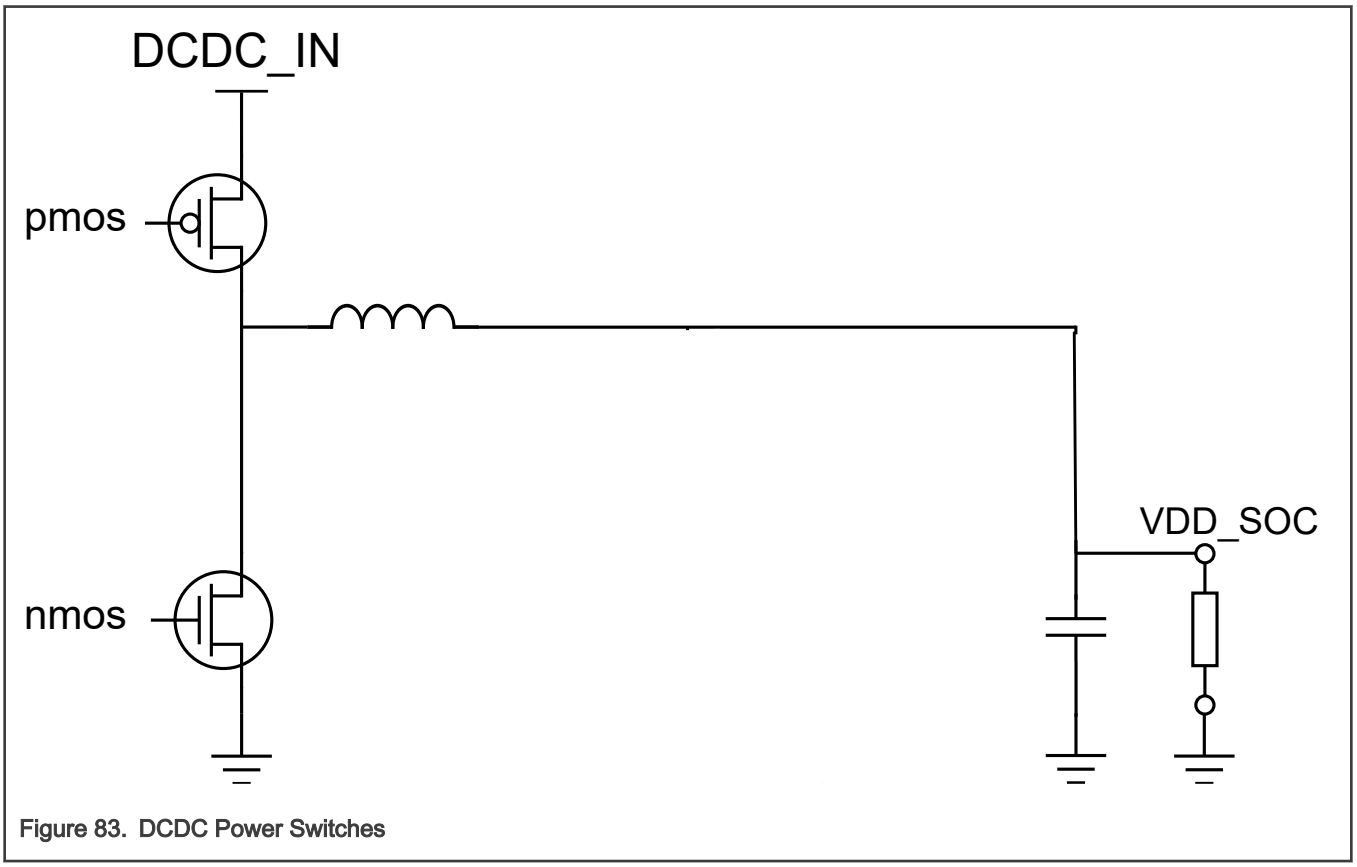


Figure 83. DCDC Power Switches

The DCDC features zero cross detection for Discontinuous Conduction Mode. This functionality is enabled by the REG0[PWD_ZCD] bit.

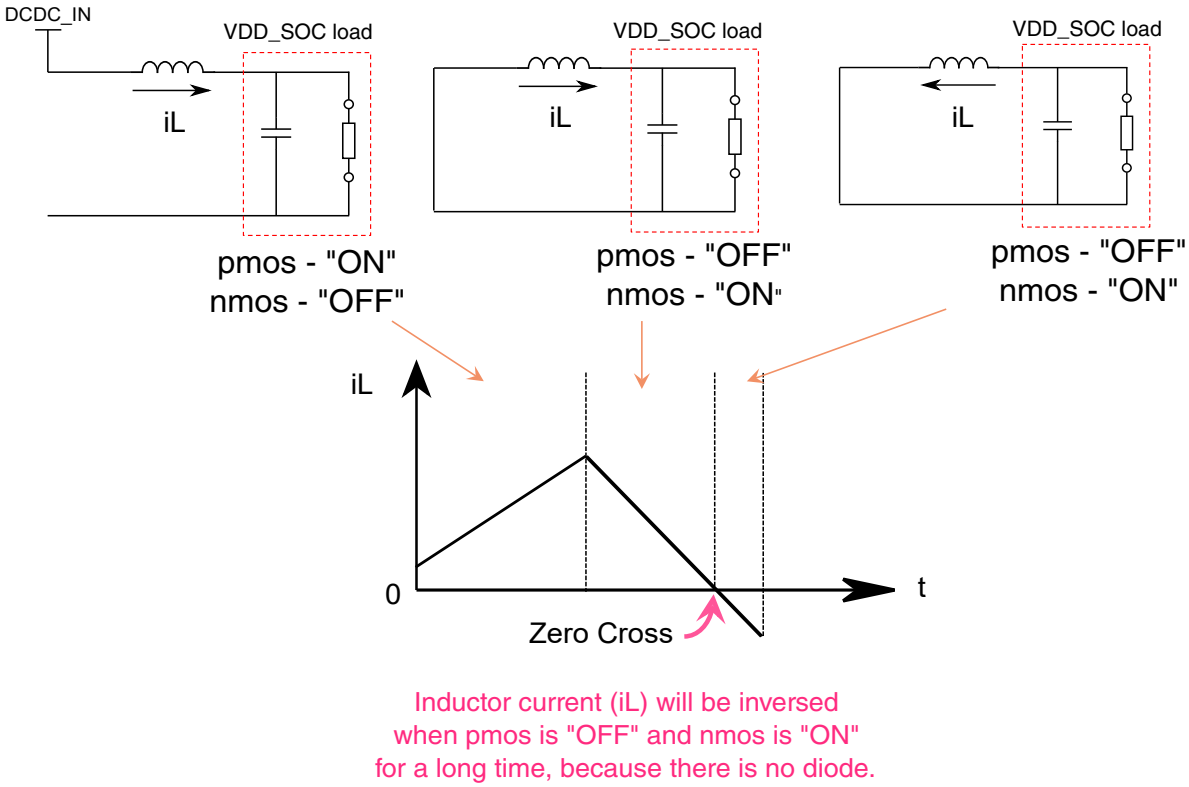


Figure 84. Zero Cross Detection

Working as the CCM, the inductor current (iL) will be inversed during the cycle of pmos is "OFF" and nmos is "ON" if the current loading is half of the current ripple of the inductor.

22.3.4 Overcurrent, Overvoltage, and Low Voltage Detection

DCDC supports the following three protection functions:

1. Overcurrent protection
   - In run mode, DCDC will shut down when detecting abnormal large current in the P-type power switch. See REG0[OVERCUR_TRIG_ADJ] for details.
   - In LP mode, DCDC will stop charging the inductor when detecting large current in the P-type power switch.
2. Overvoltage protection - DCDC will shut down when detecting high voltage on either of the two outputs.
3. Low-Voltage Detection - DCDC will shut down when detecting the input voltage is too low.
4. Low-Voltage Warning - DCDC will assert interrupt when detecting the input voltage below threshold.

NOTE

Please see the Chip Datasheet for the specified values.

22.3.5 Clocks

This section describes clocks and special clocking requirements for the DCDC module.

Table 151. Clocks

Clock	Description
ipg_clk	Peripheral Clock
clk_24M	24 MHz reference clock

Besides using an external 24 MHz clock from outside of DCDC. There is an internal 24 MHz clock inside the DCDC analog. If the internal 24 MHz clock is used, auto clock switching can be used. The diagram below illustrates how Select XTAL clock(24 MHz clock) works whenever auto clock switching is enabled or disabled:

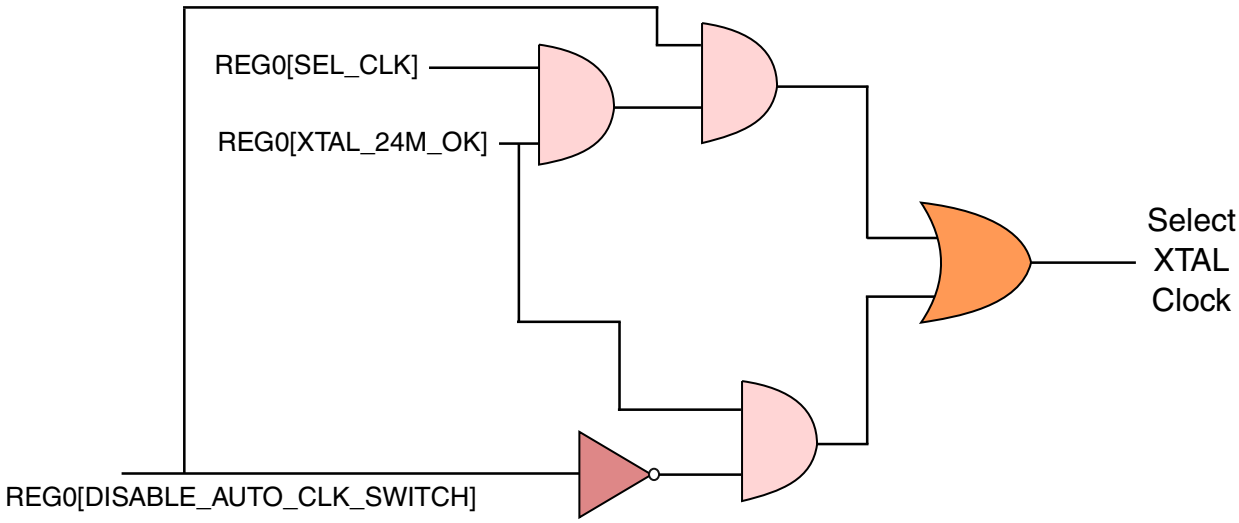


Figure 85. 24M Clock Selection

22.3.6 Interrupts

Please see the IRQ chapter for any assigned interrupts.

22.4 External Signals

Table 152. External Signals

External Signals	Power Domain	Description	Direction
DCDC_1P8	DCDC_1P8	The PAD used to provide 1.8 V power to DCDC analog module and some of the other 1.8V domain blocks of the chip.. It is connected to the 1.8 V LDO output.	O
DCDC_SENSE	VDD_SOC	Connected to DCDC_SENSE PAD, provide the sense point of DCDC output voltage for the feedback loop.	I
DCDC_IN	DCDC_IN	The DCDC input which is connected to the power switch of DCDC	I
DCDC_LP	DCDC_IN	The PAD connected to one terminal of the external inductor on the board.	IO

Table continues on the next page...

Table 152. External Signals (continued)

External Signals	Power Domain	Description	Direction
DCDC_GND	DCDC_IN	Ground PAD which is connected to DCDC power switch	O
DCDC_IN_Q	DCDC_IN	Analog power supply used by some analog modules of DCDC. It is from the same source as DCDC_IN, but is split from DCDC_IN, in order to make it more quiet.	I
DCDC_PSWITCH	DCDC_IN	The PAD which can bypass the DCDC	I
DCDC_MODE	DCDC_IN	The PAD which is used to distinguish between application mode “0” and “1”. Please refer to the Application Mode section for more information.	I

22.5 Initialization

To enable DCDC, the following conditions must be met:

1. Power supply VDD_BBSM_IN and DCDC_IN are available
2. 32 kHz clock in the VDD_BBSM_IN domain is available
3. Provide DCDC with 1ms of reset time after DCDC_IN powers up
4. PMIC_ON_REQ && DCDC_PSWITCH

The Power up sequence is:

- Power VDD_BBSM_IN
- DCDC_IN powered up
- Delay 1 ms reset time. Ensure DCDC_PSWITCH to 0 during 1 ms reset time.
- PMIC_ON_REQ && DCDC_PSWITCH
- DCDC startup

22.6 Application information

This section describes applications supported by the DCDC module.

22.6.1 Typical Application

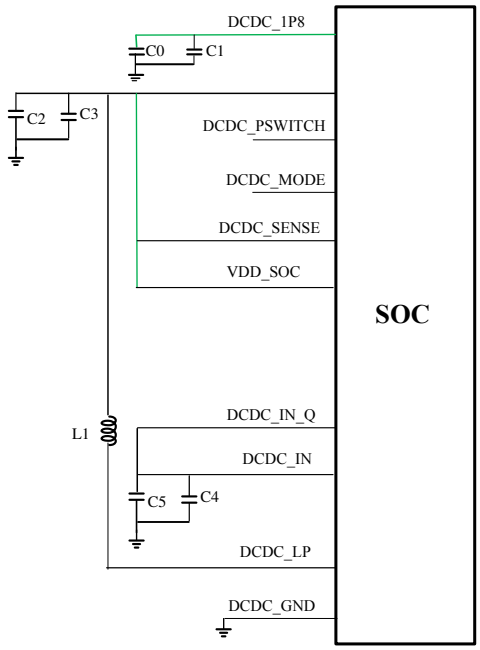


Figure 86. DCDC Typical Application

Table 153. DCDC Component List

Component	Description	Recommendation
L1	Inductance needed by DCDC	4.7uH with low DC resistance (less than 0.1 ohm ESR) inductor to minimize conduction losses.
C0	High-frequency decoupling capacitor	At least 0.1uF.
C1	Filter capacitor on DCDC_1P8 output	A 10uF with ESR less than 0.1 ohm capacitor for small ripple.
C2	High frequency decoupling capacitor	At least 0.1uF.
C3	Filter capacitors of DCDC output for VDD_SOC	A 33uF with low ESR (less than 0.02 ohm ESR) capacitor for small ripple.
C4	High-frequency decoupling capacitor for DCDC_IN	At least 0.1uF. The parasitic inductance of the routing from C4 to DCDC_IN, of the chip on the board, should be less than 0.5nH.
C5	Filter capacitor of DCDC_IN.	A 10uF (or greater) with low ESR capacitor. Increase capacitance (no limit) for better input-voltage filtering.

NOTE

It is very important to place C0, C1, C2, C3, C4 as close to the associated BGA power supply ball as possible.

NOTE

Please see the chip datasheet for specification details.

22.6.2 Recommended configuration

To improve loading ability under heavy load:

- Set CTRL0[DEBUG_BITS] = 0x1

22.7 Memory Map and register definition

This section includes the DCDC module memory map and detailed descriptions of all registers.

22.7.1 DCDC register descriptions

22.7.1.1 DCDC memory map

DCDC base address: 4452_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	DCDC Register 0 (REG0)	32	RW	0403_0501h
4h	DCDC Register 1 (REG1)	32	RW	01CD_4410h
8h	DCDC Register 2 (REG2)	32	RW	0210_8089h
Ch	DCDC Register 3 (REG3)	32	RW	0098_0000h
10h	DCDC Control Register 0 (CTRL0)	32	RW	0000_0000h
14h	OK CNT (OK_CNT)	32	RW	0000_00A4h
18h	CURRENT TARGET VALUE for DCDC ANALOG (CURRENT_TRG)	32	RW	0010_0C10h
1Ch	FILTER CNT (FILTER_CNT)	32	RW	0000_0008h
20h	TRG_0 Authentication Control (TRG_0_AUTHEN)	32	RW	FFFF_0000h
24h	Target SW Control for CORE 0 (TRG_SW_0)	32	RW	0010_0C10h
28h	Target GPC Control for CORE 0 (TRG_GPC_0)	32	RW	0010_0C10h
30h	TRG_1 Authentication Control (TRG_1_AUTHEN)	32	RW	FFFF_0000h
34h	Target SW Control for CORE 1 (TRG_SW_1)	32	RW	0010_0C10h
38h	Target GPC Control for CORE 1 (TRG_GPC_1)	32	RW	0010_0C10h

22.7.1.2 DCDC Register 0 (REG0)

Offset

Register	Offset
REG0	0h

Function

DCDC register 0

This register controls various high-level functions of the DCDC

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STS_DC_...	Reserved	XTAL_24...	Reserved	XTALOK_...	PWD_CMP...	Reserved	Reserved							PWD_HIG...	PWD_HIG...
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				PWD_CMP...	OVERCUR_TRIG_ADJ		PWD_OVE...	Reserved				PWD_OSC...	SEL_CLK	DISABLE...	PWD_ZCD
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1

Fields

Field	Function
31 STS_DC_OK	DCDC Output OK This is the status bit indicates if the DCDC output voltage is stable. The lock time depends on the loading and the DCDC mode. 0b - DCDC is settling 1b - DCDC already settled
30 —	Reserved
29 XTAL_24M_OK	24M XTAL OK This bit determines if the DCDC uses the internal ring oscillator or the xtal 24 M. See Figure 85 for details. 0b - DCDC uses internal ring OSC 1b - DCDC uses xtal 24M
28 —	Reserved
27 XTALOK_DISABLE	Disable xtalok detection circuit This is a debug bit which should not be changed in real case. This bit disables and bypasses the clock 24 MHz reference clock detector. 0b - Enable xtalok detection circuit 1b - Disable xtalok detection circuit and always outputs OK signal "1"
26 PWD_CMP_OFFSET	power down the out-of-range detection comparator The out-of-range comparator monitors the output voltage of DCDC. When the DCDC output voltage is out of range, this comparator will speed up DCDC control loop in an attempt to bring the DCDC output voltage back in range.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	PWD_CMP_OFFSET must be set to 0 (out-of-range comparator on) if LOOPCTRL_EN_RCSCALE is to be used to speed up the transient response. 0b - Out-of-range comparator powered up 1b - Out-of-range comparator powered down
25 —	Reserved
24-18 —	Reserved
17 PWD_HIGH_VDD1P0_DET	Power Down High Voltage Detection for VDD1P0 Power down overvoltage detection comparator for the VDD1P0 output. 0b - Overvoltage detection comparator for the VDD1P0 output is enabled 1b - Overvoltage detection comparator for the VDD1P0 output is disabled
16 PWD_HIGH_VDD1P8_DET	Power Down High Voltage Detection for VDD1P8 Power down overvoltage detection comparator for the VDD1P8 output. 0b - Overvoltage detection comparator for the VDD1P8 output is enabled 1b - Overvoltage detection comparator for the VDD1P8 output is disabled
15-12 —	Reserved
11 PWD_CMP_DCDC_IN_DET	Set to "1" to power down the low voltage detection comparator. Set and clear this bit can reset this comparator and its 2 output flags: 1. IN_BROWNOUT; 2 IN_BROWNOUT_WARN. 0b - Low voltage detection comparator is enabled 1b - Low voltage detection comparator is disabled
10-9 OVERCUR_TRIG_ADJ	Overcurrent Trigger Adjust Set the threshold of overcurrent detector in run mode and LP mode, if the peak current of the inductor exceeds the threshold, the current detector will assert. OVERCUR_TRIG_ADJ[1] is used to set overcurrent threshold in RUN mode - 0 is 1.5 A, 1 is 2 A. OVERCUR_TRIG_ADJ[0] is used to set overcurrent threshold in LP(low-power) mode - 0 is 150 mA, 1 is 130 mA. 00b - In Run Mode, 1.5 A. In LP Mode, 150 mA 01b - In Run Mode, 1.5 A. In LP Mode, 130 mA 10b - In Run Mode, 2 A. In LP Mode, 150 mA

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - In Run Mode, 2 A. In LP Mode, 130 mA
8 PWD_OVERCUR_DET	Power down overcurrent detection comparator 0b - Overcurrent detection comparator is enabled 1b - Overcurrent detection comparator is disabled
7-4 —	Reserved
3 PWD_OSC_INT	Power down internal osc Only set this bit when 24 MHz crystal osc is available. 0b - Internal oscillator powered up 1b - Internal oscillator powered down
2 SEL_CLK	Select Clock Select 24 MHz Crystal clock for DCDC, **when DISABLE_AUTO_CLK_SWITCH is set.** If DISABLE_AUTO_CLK_SWITCH is set to 0 and 24M xtal is OK, the clock source will switch from internal ring OSC to 24M xtal automatically. If DISABLE_AUTO_CLK_SWITCH is set to 1, SEL_CLK will determine which clock source the DCDC uses: if SEL_CLK=0, DCDC will use internal ring OSC, if SEL_CLK=1, DCDC will use 24M xtal. See Figure 85 for details. 0b - DCDC uses internal ring oscillator 1b - DCDC uses 24M xtal
1 DISABLE_AUTO_CLK_SWITCH	Disable Auto Clock Switch Disable automatic clock switch from internal osc to xtal clock. 0b - If DISABLE_AUTO_CLK_SWITCH is set to 0 and 24M xtal is OK, the clock source will switch from internal ring OSC to 24M xtal automatically 1b - If DISABLE_AUTO_CLK_SWITCH is set to 1, SEL_CLK will determine which clock source the DCDC uses
0 PWD_ZCD	Power Down Zero Cross Detection Power down the zero cross detection function for discontinuous conductor mode. 0b - Zero cross detection function powered up 1b - Zero cross detection function powered down

22.7.1.3 DCDC Register 1 (REG1)

Offset

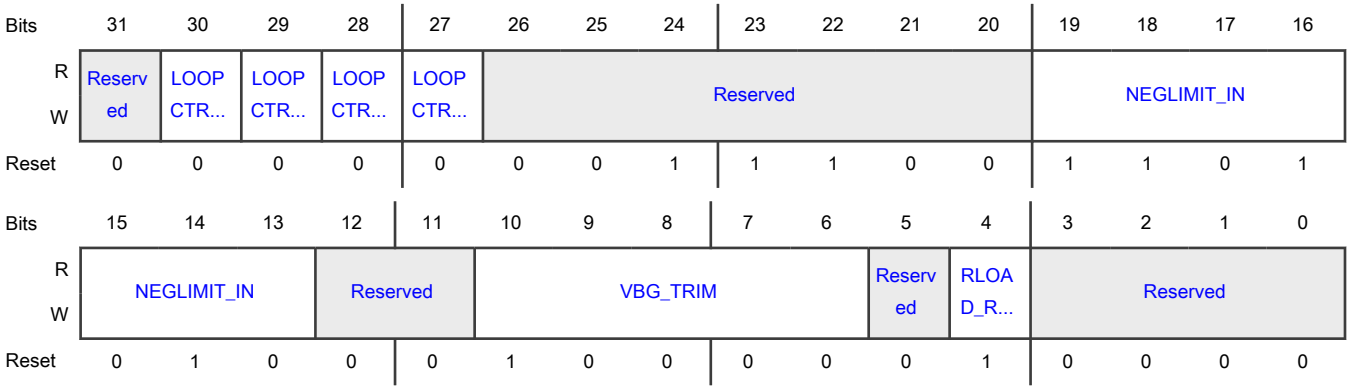
Register	Offset
REG1	4h

Function

DCDC register 1

This register controls various high-level functions of the DCDC

Diagram



Fields

Field	Function
31 —	Reserved
30 LOOPCTRL_EN_DF_HYST	Enable hysteresis in switching converter differential mode analog comparators. This feature will improve transient supply ripple and efficiency 0b - Disable hysteresis in switching converter differential mode analog comparators 1b - Enable hysteresis in switching converter differential mode analog comparators
29 LOOPCTRL_EN_CM_HYST	Enable hysteresis in switching converter common mode analog comparators. This feature will improve transient supply ripple and efficiency 0b - Disable hysteresis in switching converter common mode analog comparators 1b - Enable hysteresis in switching converter common mode analog comparators
28 LOOPCTRL_DF_HST_THRESH	Increase Threshold Detection Increase the threshold detection for differential mode analog comparators. 0b - Disable increase the threshold detection for differential mode analog comparators.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable increase the threshold detection for differential mode analog comparators.
27 LOOPCTRL_CM_HST_THRESH	Increase Threshold Detection Increase the threshold detection for common mode analog comparators. 0b - Disable increase the threshold detection for common mode analog comparators. 1b - Enable increase the threshold detection for common mode analog comparators.
26-20 —	Reserved
19-13 NEGLIMIT_IN	Negative duty cycle limit of DC-DC converter
12-11 —	Reserved
10-6 VBG_TRIM	Trim Bandgap Voltage Trim step is 3mV. 00000 - 0.452V 10000 - 0.5V 11111 - 0.545V
5 —	Reserved
4 RLOAD_REG_EN	Resistor Load of Regulator Enable This controls the load resistor of the internal regulator of DCDC. Connecting the load resistor to the internal LDO in RUN mode, improves the startup ripple of the LDO, but the load resistor will consume additional current. Once the REG0[STS_DC_OK] asserts, the load resistor can be disconnected to save the current caused by the resistor loading. 0b - Resistor load disconnected 1b - Resistor load connected
3-0 —	Reserved

22.7.1.4 DCDC Register 2 (REG2)

Offset

Register	Offset
REG2	8h

Function

DCDC register 2

This register controls various high-level functions of the DCDC

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	LOOPCTR...	Reserved	Reserved	Reserved				BATTMONITOR_BATT_VAL							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BATTMON...	Reserved	LOOPCTR...	LOOPCTR...	LOOPCTRL_EN_RCSCALE				LOOPCTRL_DC_FF				LOOPCTRL_DC_R			LOOPCTRL_DC_C
W																
Reset	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1

Fields

Field	Function
31 —	Reserved
30 LOOPCTRL_TOGGLE_DIF	Set high to enable supply stepping to change only after the differential control loop has toggled as well. This should eliminate any chance of large transients when supply voltage changes are made. 0b - Disable supply stepping to change. 1b - Enable supply stepping to change.
29 —	Reserved
28 —	Reserved
27-26 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-16 BATTMONITOR_BATT_VAL	Software should be configured to place the battery voltage in this register measured with an 8-mV LSB resolution through the ADC. This value is used by the DC-DC converter and must be correct before setting BATTMONITOR_EN_BATADJ.
15 BATTMONITOR_EN_BATADJ	This bit enables the DC-DC to improve efficiency and minimize ripple using the information from the BATTMONITOR_BATT_VAL field. It is very important that the BATTMONITOR_BATT_VAL contain accurate information before setting BATTMONITOR_EN_BATADJ. 0b - Disable the improvement function. 1b - Enable the improvement function.
14 —	Reserved
13 LOOPCTRL_HYST_SIGN	Invert the sign of the hysteresis in DC-DC analog comparators. 0b - Disable the invert function. 1b - Enable the invert function.
12 LOOPCTRL_RCSCALE_THRSH	Increase the threshold detection for RC scale circuit. 0b - Disable increasing the threshold detection function. 1b - Enable increasing the threshold detection function.
11-9 LOOPCTRL_EN_RCSCALE	Enable RC Scale This enables the analog circuit of the DCDC to respond faster under transient load conditions. Higher the value, faster the response. 0x5 is the highest setting. 0x6 - 0x7: Not available
8-6 LOOPCTRL_DC_FF	Two's complement feed forward step in duty cycle in the switching DC-DC converter. Each time this field makes a transition from 0x0, the loop filter of the DC-DC converter is stepped once by a value proportional to the change. This can be used to force a certain control loop behavior, such as improving response under known heavy load transients.
5-2 LOOPCTRL_DC_R	Magnitude of proportional control parameter in the switching DC-DC converter control loop.
1-0 LOOPCTRL_DC_C	Ratio of integral control parameter to proportional control parameter in the switching DC-DC converter, and can be used to optimize efficiency and loop response.

22.7.1.5 DCDC Register 3 (REG3)

Offset

Register	Offset
REG3	Ch

Function

DCDC register 3

This register controls various high-level functions of the DCDC

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		VDD1P0C...	Reserved	MISC_DE...	MINPWR_...	Reserved		REG_FBK_SEL		DOUBLE_...	DISABLE...	DISABLE...	ENABLE_...	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		Reserved				IN_BROW...	Reserved								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 VDD1P0CTRL_DISABLE_STEP	Disable Step for VDD1P0 Disable stepping for VDD1P0. Must set this bit before entering low power modes. 0b - Enable stepping for VDD1P0 1b - Disable stepping for VDD1P0
28 —	Reserved
27 MISC_DELAY_TIMING	Miscellaneous Delay Timing Adjust delay to reduce ground noise. 0b - Disable the function. 1b - Enable the function.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 MINPWR_HALF_FETS	Use half switch FET 0b - Donot use half switch FET. 1b - Use half switch FET.
25-24 —	Reserved
23-22 REG_FBK_SEL	Select the feedback point of the internal regulator. No need to change this field in user mode.
21 DOUBLE_IBIAS_CMP_LP	Double the bias current of the comparator for low-voltage detector in LP (low-power) mode. 0b - Disable the function. 1b - Enable the function.
20 DISABLE_IDLE_SKIP	The DCDC will be idle when out-of-range comparator detects the output voltage is higher than the target by 25mV. This function requires the out-of-range comparator to be enabled (PWD_CMP_OFFSET=0). 0b - Enable the idle skip function. 1b - Disable the idle skip function.
19 DISABLE_PULSE_SKIP	Disable Pulse Skip 0 - Stop charging if the duty cycle is lower than what is set by REG1[NEGLIMIT_IN] 1 - Enable charging if the duty cycle is lower than what is set by REG1[NEGLIMIT_IN]
18 ENABLE_FF	Enable feed-forward (FF) function that can speed up transient settling. 0b - Enable the FF function. 1b - Disable the FF function.
17-14 —	Reserved
13-10 —	Reserved
9 IN_BROWNOUT_WARN	The voltage on DCDC_IN is lower than 2.8V 0b - The voltage on DCDC_IN raises up to 2.8V. 1b - The voltage on DCDC_IN is lower than 2.8V. Once this bit sets, the bit must be cleared by software write one clear action while the voltage on DCDC_IN raises up to 2.8V. Writing "0" to this bit has no effect. Writing "1" to this bit has no effect while it's "0".
8-0 —	Reserved

22.7.1.6 DCDC Control Register 0 (CTRL0)

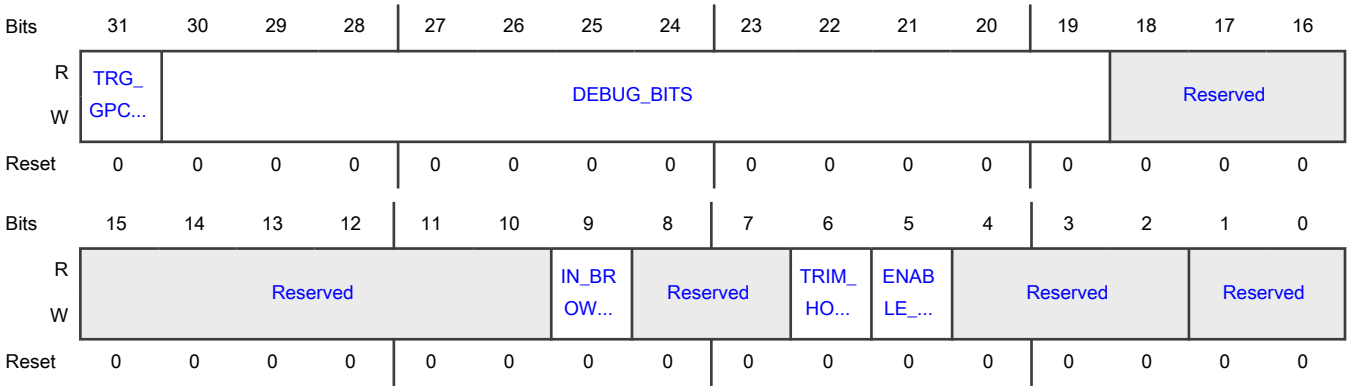
Offset

Register	Offset
CTRL0	10h

Function

DCDC Control register 0

Diagram



Fields

Field	Function
31 TRG_GPC_EN	TRG_GPC_EN: used to enable TRG_GPC_* value or not. 0b - No matter there is GPC stby request or not, value in TRG_SW_* register will always be used as DCDC analog target value. 1b - When there is a GPC stby request, value in TRG_GPC_* register will be used as DCDC analog target value instead of TRG_SW_*'s
30-19 DEBUG_BITS	Set to 0x1: To improve loading ability under heavy load.
18-10 —	Reserved
9 IN_BROWNOUT_WARN_EN	IN_BROWNOUT_WARN_EN 0b - Disable IN_BROWNOUT_WARN int flag bit output to CORE as an interrupt resource. 1b - Enable IN_BROWNOUT_WARN int flag bit output to CORE as an interrupt resource.
8-7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 TRIM_HOLD	Hold trim input 0b - Sample trim value from FUSE or value from REG1[VBG_TRIM] depending on FUSE select bit. 1b - Use value from REG1[VBG_TRIM] as trim value.
5 ENABLE_OK_CNT	Enable internal count for DCDC_OK timeout 0b - Wait DCDC_OK for ACK 1b - Enable internal count for DCDC_OK timeout
4-2 —	Reserved
1-0 —	Reserved

22.7.1.7 OK CNT (OK_CNT)

Offset

Register	Offset
OK_CNT	14h

Function

This register value is only used when CTRL0[ENABLE_OK_CNT] bit is 1. This register is internal counter initial value. When this counter decreases to 0 by 32KHz clock rising edge, DCDC state machine will jump to next state instead of waiting for DCDC analog's status OK hardware acknowledge. Reset value is (0xA4)164*0.030517=5ms.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OK_COUNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OK_COUNT															
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0

Fields

Field	Function
31-0 OK_COUNT	OK_COUNT Internal count for dcdc_ok timeout The value smaller than 8 is forbidden to write to this register. When this happens, 8 will be written to this register.

22.7.1.8 CURRENT TARGET VALUE for DCDC ANALOG (CURRENT_TRG)

Offset

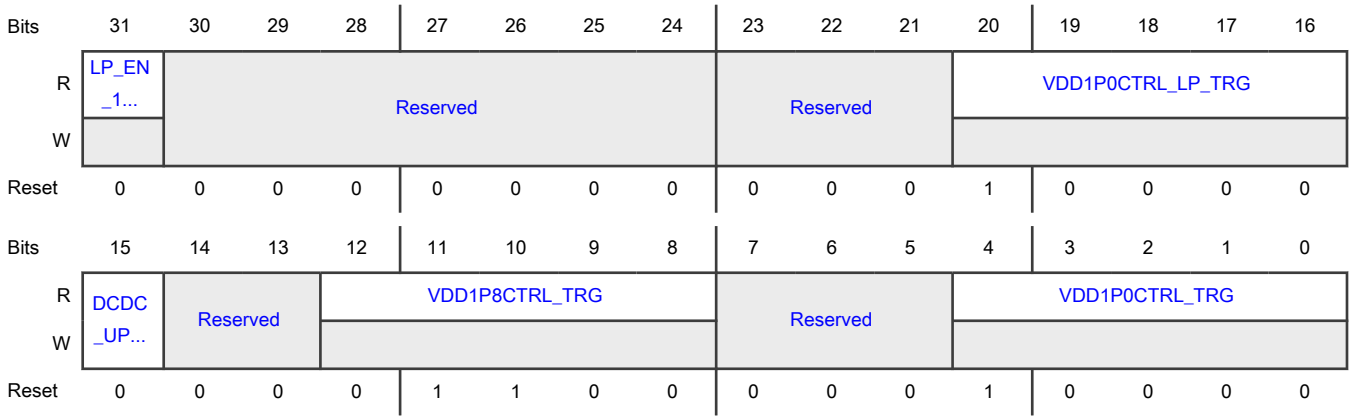
Register	Offset
CURRENT_TRG	18h

Function

This register value is current value used by DCDC analog. Since both TRG_SW_0 and TRG_SW_1 can control DCDC analog, hardware implements an arbitration strategy between them. For voltage target value, only the higher value between TRG_SW_0 and TRG_SW_1 can take effect. For LP_EN_1P0, only the lower value between them can take effect.

Notice: TRG_GPC_0 and TRG_GPC_1 also use this strategy to control DCDC analog when CTRL0[TRG_GPC_EN]=1 and there is a GPC stby request. But when TRG GPC takes effect, we can not get current value used by DCDC analog through reading this register. Because CORE is always in wait or stop mode at this time.

Diagram



Fields

Field	Function
31 LP_EN_1P0	This value comes from the smaller one between TRG_SW_0 and TRG_SW_1. This bit only controls 1P0. 1P8 is always controlled by VDD1P8CTRL_TRG

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DCDC 1P0 works in run mode. Its output voltage is controlled by VDD1P0CTRL_TRG. 1b - DCDC 1P0 works in low power mode. Its output voltage is controlled by VDD1P0CTRL_LP_TRG and its output current is less than 50mA.
30-24 —	Reserved
23-21 —	Reserved
20-16 VDD1P0CTRL_LP_TRG	This value is only valid when LP_EN_1P0 =1. The value comes from the bigger one between TRG_SW_0 and TRG_SW_1.
15 DCDC_UPDATING	Notice: Only if the time of the bit's 1 is too long(more than several seconds) and REG0[STS_DC_OK]=1, then you can write 1 to clear it. Else writing to this bit is forbidden. Writing to this bit is only for debug. 0b - Last DCDC change has been done. New value can be written to TRG register to trigger a new change of DCDC voltage. 1b - Last DCDC change is still not done. New value should not be written to TRG register to trigger a new change of DCDC voltage.
14-13 —	Reserved
12-8 VDD1P8CTRL_TRG	This value is current value used by DCDC analog. The value comes from the bigger one between TRG_SW_0 and TRG_SW_1.
7-5 —	Reserved
4-0 VDD1P0CTRL_TRG	This value is current value used by DCDC analog. The value comes from the bigger one between TRG_SW_0 and TRG_SW_1.

22.7.1.9 FILTER CNT (FILTER_CNT)

Offset

Register	Offset
FILTER_CNT	1Ch

Function

Control dcdc_sts_ok signal's filter function. This register can force DCDC analog status output signal dcdc_sts_ok to 0. Cannot write value smaller than 2.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	FILTER_CNT_CFG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FILTER_CNT_CFG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-0 FILTER_CNT_CFG	FILTER_CNT_CFG Filter counter timeout value. Any DCDC change requirement can trigger filter counter begin to count by 32K clock, and it will stop at this value and go back to 0 at next clock cycle. When this counter is not 0, dcdc_sts_ok will be forced to 0, otherwise dcdc_sts_ok will be released to its original value.

22.7.1.10 TRG_0 Authentication Control (TRG_0_AUTHEN)

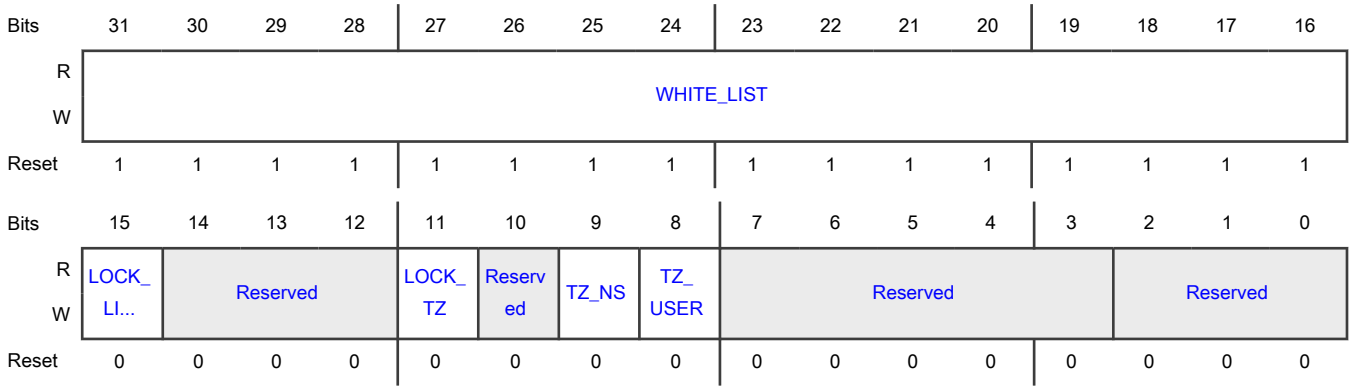
Offset

Register	Offset
TRG_0_AUTHEN	20h

Function

This register controls TRG_SW_0, TRG_GPC_0 and its own access control for different CORE.

Diagram



Fields

Field	Function
31-16 WHITE_LIST	Domain ID white list Each master(Core or non Core) can be assigned to any domains among domain0~domain15 by TRDC. Once its domains are assigned, then its accesses will have its unique domain ID. Each bit in this field represent for one domain. Bit16~bit31 represent for domain0~domain15 respectively. Only the domains whose corresponding bits are 1 can write TRG_0 registers. Write violation will fail without any exception or information. Read violation can still get read data. 0000_0000_0000_0001b - Core with domain ID=0 can write TRG_0 registers. 0000_0000_0000_0010b - Core with domain ID=1 can write TRG_0 registers. 0000_0000_0000_0100b - Core with domain ID=2 can write TRG_0 registers. 0000_0000_0000_1000b - Core with domain ID=3 can write TRG_0 registers. 0000_0000_0001_0000b - Core with domain ID=4 can write TRG_0 registers. 0000_0000_0010_0000b - Core with domain ID=5 can write TRG_0 registers. 0000_0000_0100_0000b - Core with domain ID=6 can write TRG_0 registers. 0000_0000_1000_0000b - Core with domain ID=7 can write TRG_0 registers. 0000_0001_0000_0000b - Core with domain ID=8 can write TRG_0 registers. 0000_0010_0000_0000b - Core with domain ID=9 can write TRG_0 registers. 0000_0100_0000_0000b - Core with domain ID=10 can write TRG_0 registers. 0000_1000_0000_0000b - Core with domain ID=11 can write TRG_0 registers. 0001_0000_0000_0000b - Core with domain ID=12 can write TRG_0 registers. 0010_0000_0000_0000b - Core with domain ID=13 can write TRG_0 registers. 0100_0000_0000_0000b - Core with domain ID=14 can write TRG_0 registers. 1000_0000_0000_0000b - Core with domain ID=15 can write TRG_0 registers.
15 LOCK_LIST	White list lock This field can lock itself and WHITE_LIST.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - WHITE_LIST value can be changed. 1b - LOCK_LIST and WHITE_LIST value cannot be changed.
14-12 —	Reserved
11 LOCK_TZ	Lock TZ_NS and TZ_USER This field can lock itself, TZ_NS and TZ_USER. 0b - TZ_NS and TZ_USER value can be changed. 1b - LOCK_TZ, TZ_NS and TZ_USER value cannot be changed.
10 —	Reserved
9 TZ_NS	Allow non-secure mode access Allow either secure mode or non-secure mode to write TRG_0 registers. 0b - TRG_0 registers can only be written in secure mode. 1b - TRG_0 registers can be written either in secure mode or non-secure mode.
8 TZ_USER	Allow user mode write Allow either privilege mode or user mode to write TRG_0 registers. 0b - TRG_0 registers can only be written in privilege mode. 1b - TRG_0 registers can be written either in privilege mode or user mode.
7-3 —	Reserved
2-0 —	Reserved

22.7.1.11 Target SW Control for CORE 0 (TRG_SW_0)

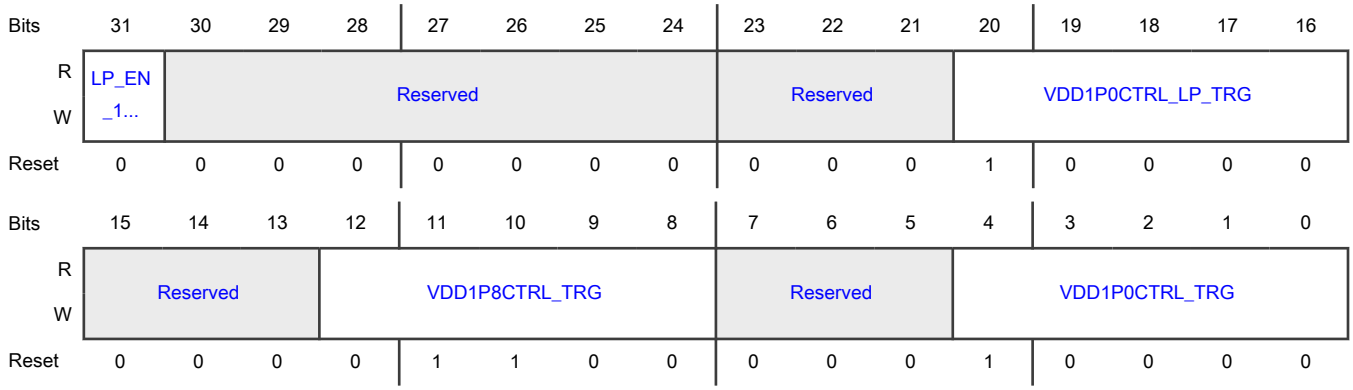
Offset

Register	Offset
TRG_SW_0	24h

Function

DCDC target software control register for CORE 0.

Diagram



Fields

Field	Function
31 LP_EN_1P0	LP_EN_1P0 only controls 1P0. 1P8 is always controlled by VDD1P8CTRL_TRG 1b - DCDC 1P0 works in low power mode. Its output voltage is controlled by VDD1P0CTRL_LP_TRG and its output current is less than 50mA. 0b - DCDC 1P0 works in run mode. Its output voltage is controlled by VDD1P0CTRL_TRG.
30-24 —	Reserved
23-21 —	Reserved
20-16 VDD1P0CTRL_LP_TRG	This value is only valid when LP_EN_1P0 =1. Target value of VDD1P0 in low power mode, 25mV each step from 0x00 to 0x1F: 0x1F: 1.375V 0x10: 1.0V 0x00: 0.6V
15-13 —	Reserved
12-8 VDD1P8CTRL_TRG	Target value of VDD1P8 in run mode, 25mV each step from 0x00 to 0x1F: 0x1F: 2.275V 0x0C: 1.8V 0x00: 1.5V
7-5 —	Reserved
4-0	Target value of VDD1P0 in run mode, 25mV each step from 0x00 to 0x1F:

Table continues on the next page...

Table continued from the previous page...

Field	Function
VDD1P0CTRL_TRG	0x1F: 1.375V 0x10: 1.0V 0x00: 0.6V

22.7.1.12 Target GPC Control for CORE 0 (TRG_GPC_0)

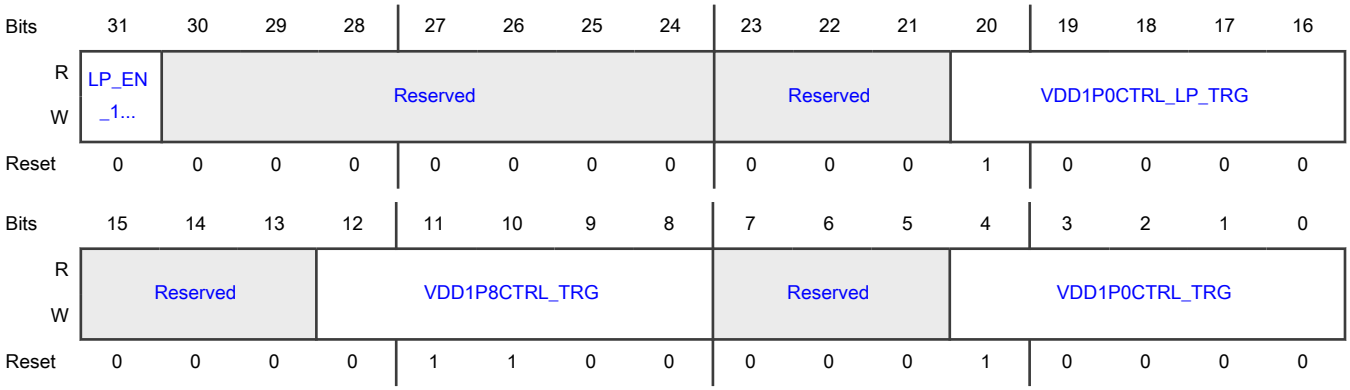
Offset

Register	Offset
TRG_GPC_0	28h

Function

DCDC target GPC control register for CORE 0. This register value only take effect when TRG_GPC_EN = 1 and there is a GPC stby request, else TRG_SW_0 will take effect.

Diagram



Fields

Field	Function
31 LP_EN_1P0	LP_EN_1P0 only controls 1P0. 1P8 is always controlled by VDD1P8CTRL_TRG 1b - DCDC 1P0 works in low power mode. Its output voltage is controlled by VDD1P0CTRL_LP_TRG and its output current is less than 50mA. 0b - DCDC 1P0 works in run mode. Its output voltage is controlled by VDD1P0CTRL_TRG.
30-24 —	Reserved
23-21	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
20-16 VDD1P0CTRL_LP_TRG	This value is only valid when LP_EN_1P0 =1. Target value of VDD1P0 in low power mode, 25mV each step from 0x00 to 0x1F: 0x1F: 1.375V 0x10: 1.0V 0x00: 0.6V
15-13 —	Reserved
12-8 VDD1P8CTRL_TRG	Target value of VDD1P8 in run mode, 25mV each step from 0x00 to 0x1F: 0x1F: 2.275V 0x0C: 1.8V 0x00: 1.5V
7-5 —	Reserved
4-0 VDD1P0CTRL_TRG	Target value of VDD1P0 in run mode, 25mV each step from 0x00 to 0x1F: 0x1F: 1.375V 0x10: 1.0V 0x00: 0.6V

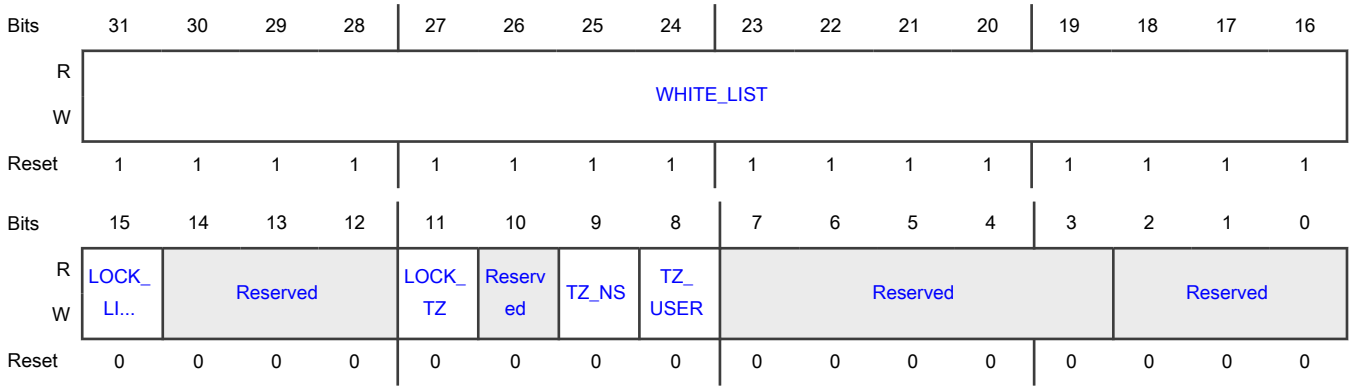
22.7.1.13 TRG_1 Authentication Control (TRG_1_AUTHEN)

Offset

Register	Offset
TRG_1_AUTHEN	30h

Function
This register controls TRG_SW_1, TRG_GPC_1 and its own access control for different CORE.

Diagram



Fields

Field	Function
31-16 WHITE_LIST	Domain ID white list Each master(Core or non Core) can be assigned to any domains among domain0~domain15 by TRDC. Once its domains are assigned, then its accesses will have its unique domain ID. Each bit in this field represent for one domain. Bit16~bit31 represent for domain0~domain15 respectively. Only the domains whose corresponding bits are 1 can write TRG_1 registers. Write violation will fail without any exception or information. Read violation can still get read data. 0000_0000_0000_0001b - Core with domain ID=0 can write TRG_1 registers. 0000_0000_0000_0010b - Core with domain ID=1 can write TRG_1 registers. 0000_0000_0000_0100b - Core with domain ID=2 can write TRG_1 registers. 0000_0000_0000_1000b - Core with domain ID=3 can write TRG_1 registers. 0000_0000_0001_0000b - Core with domain ID=4 can write TRG_1 registers. 0000_0000_0010_0000b - Core with domain ID=5 can write TRG_1 registers. 0000_0000_0100_0000b - Core with domain ID=6 can write TRG_1 registers. 0000_0000_1000_0000b - Core with domain ID=7 can write TRG_1 registers. 0000_0001_0000_0000b - Core with domain ID=8 can write TRG_1 registers. 0000_0010_0000_0000b - Core with domain ID=9 can write TRG_1 registers. 0000_0100_0000_0000b - Core with domain ID=10 can write TRG_1 registers. 0000_1000_0000_0000b - Core with domain ID=11 can write TRG_1 registers. 0001_0000_0000_0000b - Core with domain ID=12 can write TRG_1 registers. 0010_0000_0000_0000b - Core with domain ID=13 can write TRG_1 registers. 0100_0000_0000_0000b - Core with domain ID=14 can write TRG_1 registers. 1000_0000_0000_0000b - Core with domain ID=15 can write TRG_1 registers.
15 LOCK_LIST	White list lock This field can lock itself and WHITE_LIST.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - WHITE_LIST value can be changed. 1b - LOCK_LIST and WHITE_LIST value cannot be changed.
14-12 —	Reserved
11 LOCK_TZ	Lock TZ_NS and TZ_USER This field can lock itself, TZ_NS and TZ_USER. 0b - TZ_NS and TZ_USER value can be changed. 1b - LOCK_TZ, TZ_NS and TZ_USER value cannot be changed.
10 —	Reserved
9 TZ_NS	Allow non-secure mode access Allow either secure mode or non-secure mode to write TRG_1 registers. 0b - TRG_1 registers can only be written in secure mode. 1b - TRG_1 registers can be written either in secure mode or non-secure mode.
8 TZ_USER	Allow user mode write Allow either privilege mode or user mode to write TRG_1 registers. 0b - TRG_1 registers can only be written in privilege mode. 1b - TRG_1 registers can be written either in privilege mode or user mode.
7-3 —	Reserved
2-0 —	Reserved

22.7.1.14 Target SW Control for CORE 1 (TRG_SW_1)

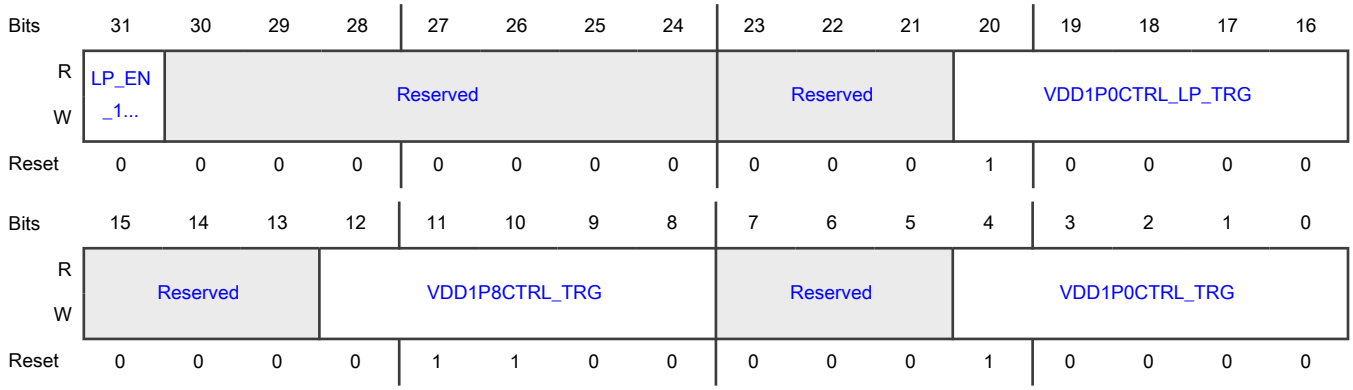
Offset

Register	Offset
TRG_SW_1	34h

Function

DCDC target software control register for CORE 1.

Diagram



Fields

Field	Function
31 LP_EN_1P0	LP_EN_1P0 only controls 1P0. 1P8 is always controlled by VDD1P8CTRL_TRG 1b - DCDC 1P0 works in low power mode. Its output voltage is controlled by VDD1P0CTRL_LP_TRG and its output current is less than 50mA. 0b - DCDC 1P0 works in run mode. Its output voltage is controlled by VDD1P0CTRL_TRG.
30-24 —	Reserved
23-21 —	Reserved
20-16 VDD1P0CTRL_LP_TRG	This value is only valid when LP_EN_1P0 =1. Target value of VDD1P0 in low power mode, 25mV each step from 0x00 to 0x1F: 0x1F: 1.375V 0x10: 1.0V 0x00: 0.6V
15-13 —	Reserved
12-8 VDD1P8CTRL_TRG	Target value of VDD1P8 in run mode, 25mV each step from 0x00 to 0x1F: 0x1F: 2.275V 0x0C: 1.8V 0x00: 1.5V
7-5 —	Reserved
4-0	Target value of VDD1P0 in run mode, 25mV each step from 0x00 to 0x1F:

Table continues on the next page...

Table continued from the previous page...

Field	Function
VDD1P0CTRL_TRG	0x1F: 1.375V 0x10: 1.0V 0x00: 0.6V

22.7.1.15 Target GPC Control for CORE 1 (TRG_GPC_1)

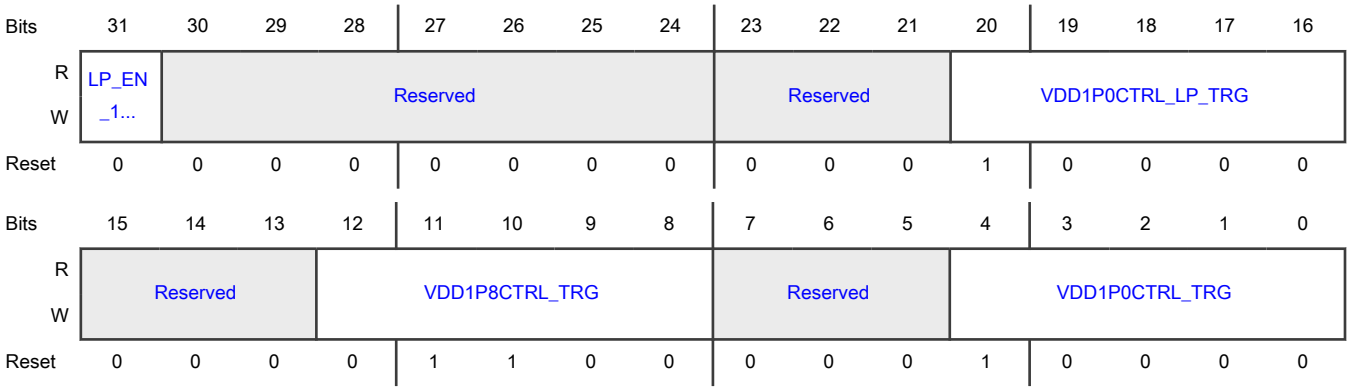
Offset

Register	Offset
TRG_GPC_1	38h

Function

DCDC target GPC control register for CORE 1. This register value only take effect when TRG_GPC_EN = 1 and there is a GPC stby request, else TRG_SW_0 will take effect.

Diagram



Fields

Field	Function
31 LP_EN_1P0	LP_EN_1P0 only controls 1P0. 1P8 is always controlled by VDD1P8CTRL_TRG 1b - DCDC 1P0 works in low power mode. Its output voltage is controlled by VDD1P0CTRL_LP_TRG and its output current is less than 50mA. 0b - DCDC 1P0 works in run mode. Its output voltage is controlled by VDD1P0CTRL_TRG.
30-24 —	Reserved
23-21	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
20-16 VDD1P0CTRL_LP_TRG	This value is only valid when LP_EN_1P0 =1. Target value of VDD1P0 in low power mode, 25mV each step from 0x00 to 0x1F: 0x1F: 1.375V 0x10: 1.0V 0x00: 0.6V
15-13 —	Reserved
12-8 VDD1P8CTRL_TRG	Target value of VDD1P8 in run mode, 25mV each step from 0x00 to 0x1F: 0x1F: 2.275V 0x0C: 1.8V 0x00: 1.5V
7-5 —	Reserved
4-0 VDD1P0CTRL_TRG	Target value of VDD1P0 in run mode, 25mV each step from 0x00 to 0x1F: 0x1F: 1.375V 0x10: 1.0V 0x00: 0.6V

Chapter 23
Temperature Sensor (TMPSNS)

23.1 Chip-specific TMPSNS information

Table 154. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

The TMPSNS analog registers are in the ANADIG chapter. The TEMPSNS_OTP_TRIM_VALUE[TEMPSNS_TEMP_VAL] in ANADIG is used in the temperature equation.

23.2 Overview

TMPSNS implements a temperature sensor and conversion function based on a temperature-dependent voltage-to-time conversion.

23.2.1 Block diagram

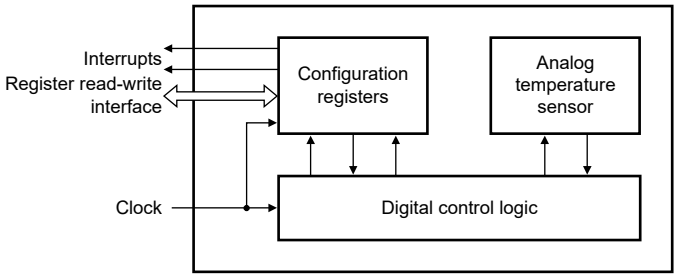


Figure 87. Block diagram

23.2.2 Features

- Single Measurement mode
  - Performs one measurement for each start indication from software
  - Asserts STATUS0[FINISH] on every measurement completion
- Continuous Measurement mode
  - Performs continuous measurements after being initiated by software
  - Allows frequency of measurements to be programmed by software
  - Asserts STATUS0[FINISH] on every measurement completion, if previously cleared
- Contains temperature alarm interrupts to indicate if the temperature is beyond programmed thresholds
  - Supports three tier alarms: Low, High, Panic

23.3 Functional description

TMPSNS features alarm functions that can raise independent interrupt signals if the temperature is above either of two high-temperature thresholds, or below a low-temperature threshold. These temperature thresholds are programmable and designated as low, high, and panic temperatures.

The panic threshold is a special programmable threshold. If the temperature increases above this value and the temperature-panic-reset interrupt is enabled in the System Reset Controller, then the hardware assumes that software no longer has control over the thermal situation and initiates a reset of the chip. In order to avoid resets initiated by a false panic temperature, the panic alarm is not triggered until the temperature panic condition has been met for four consecutive conversion cycles. You can also program a self-repeating mode that executes a temperature-sensing operation based on a programmed delay.

Because the high and low temperature thresholds are programmable, they form a sliding temperature bracket that can be tailored to the needs of each application. For example, at startup, you can set the low-temperature threshold to the minimum temperature code, and the high-temperature threshold to a maximum operating temperature for the chip.

The chip can then use TMPSNS to monitor the on-die temperature and take appropriate action, such as throttling back the core frequency when the high-temperature interrupt is set. After you have set the high-temperature interrupt, the chip can program the low-temperature threshold to a desired cool-down temperature. The chip can then switch to monitoring the low-temperature alarm and wait for its interrupt to be set.

With this scheme, after the low-temperature interrupt is set, TMPSNS can determine that the temperature has cooled down to a safe level. The process can then be repeated.

23.3.1 Modes of operation

The following modes are supported in TMPSNS:

- Single Measurement mode
  - Performs one measurement per start indication from software.
  - Asserts STATUS0[FINISH] on completion of every measurement.
- Continuous Measurement mode
  - Performs continuous measurements (after being initiated by software) at the rate defined by CTRL1[FREQ].
  - Allows you to program the frequency of measurements through software.
  - Asserts STATUS0[FINISH] on every measurement completion, if previously cleared.

23.3.2 Temperature sensing

Use Equation 1 to calculate the initial value of the temperature in *C.

$$temp_{diode} = \frac{[-ts_{21} - \sqrt{ts_{21}^2 - 4 \times ts_{22} \times (ts_{20} + ts_{25C} - ts_{meas})}]}{[2 \times ts_{22}]}$$

Where:

$ts_{21} = -5.39$
$ts_{22} = 0.002$
$ts_{20} = 133.6$

ts_{meas} = the measured code output from the temperature sensor
$ts_{25C} = ts_{offset} + 1915$, where ts_{offset} is a value in two's-complement format that comes from fuses

Equation 1. Temperature formula

23.3.3 Clocking

Clock	Description
TEMP_24MCLK	Reference clock

23.3.4 Reset

All resets are controlled through the Analog IP (AI) interface.

23.3.5 Interrupts

All interrupts are controlled through the Analog IP (AI) interface.

23.4 External signals

TMPSNS has no external signals.

23.5 Initialization

There are no special considerations for initializing TMPSNS. Use CTRL1[PWD] and CTRL1[PWD_FULL] to power down the temperature sensor.

23.6 Application information

23.6.1 Configuration for Single Measurement mode

1. Write the appropriate threshold values in Range 0 (RANGE0) and Range 1 (RANGE1) for the Panic, High, and Low Temperature alarms to trigger. See Temperature sensing to determine the value to program, based on the selected temperature value.
2. Enable appropriate interrupts by writing 1 to the corresponding interrupt enable fields in Control 1 (CTRL1).
3. Power up the analog section by writing 0 to CTRL1[PWD] and CTRL1[PWD_FULL].
4. Ensure that CTRL1[FREQ] = 0.
5. Enable the measurement process by writing 1 to CTRL1[START].
6. Determine the completion of the measurement by polling STATUS0[FINISH].
7. Read the measured value in STATUS0[TEMP_VAL].
8. Clear CTRL1[FINISH_IE] by writing 1 to it.
9. Clear CTRL1[START] by writing 0 to it.

Repeat steps 4–8 to start another measurement cycle.

23.6.2 Configuration for Multiple Measurement mode

1. Write the appropriate threshold values in Range 0 (RANGE0) and Range 1 (RANGE1) for the Panic, High, and Low Temperature alarms to trigger. See Temperature sensing to determine the value to program, based on the selected temperature value.
2. Enable appropriate interrupts by writing 1 to the corresponding interrupt enable fields in Control 1 (CTRL1).
3. Power up the analog section by writing 0 to CTRL1[PWD] and CTRL1[PWD_FULL].
4. Set the selected frequency of measurement by writing a non-zero value to CTRL1[FREQ]. This field determines the number of idle clock cycles between two conversions. The next measurement starts after counting these clock cycles.
5. Enable the measurement process by writing 1 to CTRL1[START].
6. Determine the completion of the measurement by polling STATUS0[FINISH].
7. Read the measured value in STATUS0[TEMP_VAL].
8. Clear CTRL1[FINISH_IE] by writing 1 to it.
9. Repeat steps 5 to 7 to continue receiving continuous measurements.
10. To exit Multiple Measurement mode, write 0 to CTRL1[START].

23.7 Memory map and register definitions

23.7.1 TMPSNS register descriptions

TMPSNS memory consists of 32-bit–aligned registers that can be accessed only via 8-, 16-, or 32-bit accesses. Write access to reserved locations generates a transfer error. Read access to reserved locations also generates a transfer error, and the read data bus shows all zeros.

TMPSNS does not check for correctness of programmed values in the registers. You must program software to ensure that the values are valid.

NOTE

All register fields are shadowed except for those in Status 0 (STATUS0). You can change shadowed fields only when CTRL1[START] = 0 or CTRL1[PWD] = 1. Changing shadowed fields at any other time will take effect only when CTRL1[START] is toggled from 0 to 1, or CTRL1[PWD] is toggled from 1 to 0. You can change any STATUS0 field regardless of the values of CTRL1[START] and CTRL1[PWD].

23.7.1.1 TMPSNS memory map

TMPSNS base address: 4448_4580h

Offset	Register	Width (In bits)	Access	Reset value
10h	Control 1 (CTRL1)	32	RW	8080_0000h
14h	Control 1 (CTRL1_SET)	32	RW	8080_0000h
18h	Control 1 (CTRL1_CLR)	32	RW	8080_0000h
1Ch	Control 1 (CTRL1_TOG)	32	RW	8080_0000h
20h	Range 0 (RANGE0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
24h	Range 0 (RANGE0_SET)	32	RW	0000_0000h
28h	Range 0 (RANGE0_CLR)	32	RW	0000_0000h
2Ch	Range 0 (RANGE0_TOG)	32	RW	0000_0000h
30h	Range 1 (RANGE1)	32	RW	0000_0000h
34h	Range 1 (RANGE1_SET)	32	RW	0000_0000h
38h	Range 1 (RANGE1_CLR)	32	RW	0000_0000h
3Ch	Range 1 (RANGE1_TOG)	32	RW	0000_0000h
50h	Status 0 (STATUS0)	32	RW	0000_0000h

23.7.1.2 Control 1 (CTRL1)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
CTRL1	10h	Control 1
CTRL1_SET	14h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL1 is 1
CTRL1_CLR	18h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL1 is 0
CTRL1_TOG	1Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in CTRL1

Function

Configures the digital temperature sensor control and mode of operation (Single Read or Continuous Read).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PWD_FULL	0							PWD	START	0		PANIC_T...	HIGH_TE...	LOW_TEM...	FINISH_...
W																
Reset	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FREQ															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PWD_FULL	Power Down Powers down TMPSNS. If this field = 0, TMPSNS is active. If this field = 1, TMPSNS is powered down. 0b - Active 1b - Inactive
30-24 —	Reserved
23 PWD	Power Down Except Bias Current Powers down TMPSNS except for the bias current. If this field = 0, TMPSNS is active. If this field = 1, TMPSNS is powered down except for the bias current. 0b - Active 1b - Inactive
22 START	Start Temperature Measurement Initiates the temperature reading in either Single Read mode or Continuous Read mode. For single reading, you must write 0 this field and then write 1 for every read. For continuous reading, this field must remain 1. Write 0 to this field when you want to stop measurement. 0b - No read 1b - New read
21-20 —	Reserved
19 PANIC_TEMP_IE	Panic Temperature Interrupt Enable Enables assertion of an interrupt when the measured temperature is greater than or equal to the programmed panic temperature value. 0b - Disable 1b - Enable
18 HIGH_TEMP_IE	High Temperature Interrupt Enable Enables assertion of an interrupt when the measured temperature is greater than or equal to the programmed high temperature value. 0b - Disable 1b - Enable
17 LOW_TEMP_IE	Low Temperature Interrupt Enable Enables assertion of an interrupt when the measured temperature is lower than or equal to the programmed low temperature value. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
16 FINISH_IE	Measurement Finished Interrupt Enable Enables assertion of an interrupt when the temperature measurement is complete. 0b - Disable 1b - Enable
15-0 FREQ	Temperature Measurement Frequency Represents the number of clock cycles of the 24 MHz clock that TMPSNS counts between two temperature readings. If this field = 0, TMPSNS works in Single Reading mode, where a new temperature reading is available every time START = 1. For non-zero values of this field, TMPSNS works in Continuous Reading mode, where it initiates a new temperature reading when the programmed period expires after the end of the current temperature reading. The value of START must remain 1 for continuous conversion. 0000_0000_0000_0000b - Single Reading Mode. A new reading is available every time you change START from 0 to 1. 0000_0000_0000_0001b-1111_1111_1111_1111b - Continuous Reading Mode. TMPSNS takes the next temperature reading when it completes the programmed number of cycles after the current reading.

23.7.1.3 Range 0 (RANGE0)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
RANGE0	20h	Range 0
RANGE0_SET	24h	Writing 1 to a bit in this register ensures that the corresponding bit in RANGE0 is 1
RANGE0_CLR	28h	Writing 1 to a bit in this register ensures that the corresponding bit in RANGE0 is 0
RANGE0_TOG	2Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in RANGE0

Function

Configures the ranges for generation of High and Low temperature alarms.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				HIGH_TEMP_VAL											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				LOW_TEMP_VAL											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-16 HIGH_TEMP_VAL	High Temperature Threshold Value Contains the high temperature threshold value (number of clock counts) below which you must write 1 to STATUS0[HIGH_TEMP].
15-12 —	Reserved
11-0 LOW_TEMP_VAL	Low Temperature Threshold Value Contains the low temperature threshold value (number of clock counts) above which you must set STATUS0[LOW_TEMP].

23.7.1.4 Range 1 (RANGE1)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
RANGE1	30h	Range 1
RANGE1_SET	34h	Writing 1 to a bit in this register ensures that the corresponding bit in RANGE1 is 1
RANGE1_CLR	38h	Writing 1 to a bit in this register ensures that the corresponding bit in RANGE1 is 0
RANGE1_TOG	3Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in RANGE1

Function

Configures the ranges for generation of a Panic temperature alarm.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				PANIC_TEMP_VAL											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 PANIC_TEMP_VAL	Panic Temperature Threshold Value Contains the panic temperature threshold value (number of clock counts) below which you must write 1 to STATUS0[PANIC_TEMP].

23.7.1.5 Status 0 (STATUS0)

Offset

Register	Offset
STATUS0	50h

Function

Provides the status and the measured value from the temperature sensor.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												PANIC_T...	HIGH_TE...	LOW_TEMP	FINISH
W													W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TEMP_VAL											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 PANIC_TEMP	Panic Temperature Alarm Asserts when the measured temperature is above the programmed panic temperature threshold. 0b - No alert 1b - Alert
18 HIGH_TEMP	High Temperature Alarm Asserts when the measured temperature is above the programmed high temperature threshold. 0b - No alert 1b - Alert
17 LOW_TEMP	Low Temperature Alarm Asserts when the measured temperature is below the programmed low temperature threshold. 0b - No alert 1b - Alert
16 FINISH	Temperature Measurement Complete Indicates that the temperature sensor has finished measuring the temperature value, and the measured value is available for reading in STATUS0[TEMP_VAL]. You must clear FINISH by writing 1 to acknowledge that the temperature reading has been read out. If this field = 0, then the temperature sensor is either busy (if CTRL1[START] = 1) or else no new reading has been initiated (if CTRL1[START] = 0). If this field = 1, the temperature reading is complete. 0b - No read 1b - New read
15-12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11-0 TEMP_VAL	Measured Temperature Value Contains the clock count corresponding to the temperature measured by the sensor. This field is valid when STATUS0[FINISH] = 1. To calculate the actual temperature value, you must convert this count using the formula shown in Temperature sensing.

Chapter 24
ANADIG

24.1 Chip-specific ANADIG Information

Table 155. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

24.2 Overview

The ANADIG block contains the Power Management Unit (PMU), Crystal Oscillator (XTALOSC), Temp Sensor (TMPSNS), PLL controls for CCM, and several other miscellaneous system-level control and status registers.

NOTE

AON_ANA_LDO and AON_1P8_LDO is used interchangeably throughout this document.

24.2.1 Block Diagram

The following shows a block diagram of ANADIG.

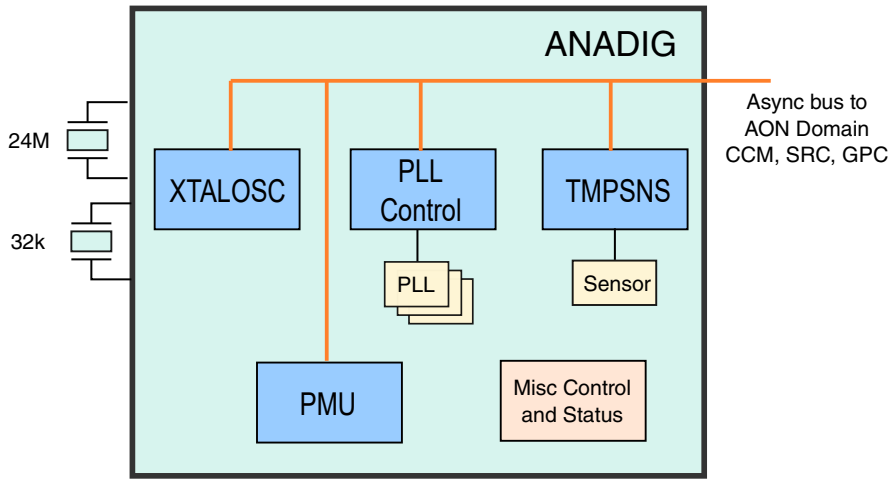


Figure 88. ANADIG Block Diagram

24.2.2 Features

Main features of ANADIG are:

- Integrated crystal ,RC oscillators and PLLs for system generation
- Integrated Power Management Unit (PMU) analog component: LDOs and Power Switches
- Integrated Temperature Monitor for system over heat detection

24.3 Functional Description

The following sections describe the XTALOSC, PMU, PLL, and TMPSNS modules.

24.3.1 Crystal Oscillator (XTALOSC)

This chip contains two crystal oscillators for clock generation: 24 MHz and 32 kHz. In addition, it also contains three RC oscillators: 24 MHz, 32 kHz and 400 MHz.

24.3.1.1 XTALOSC Block Diagram

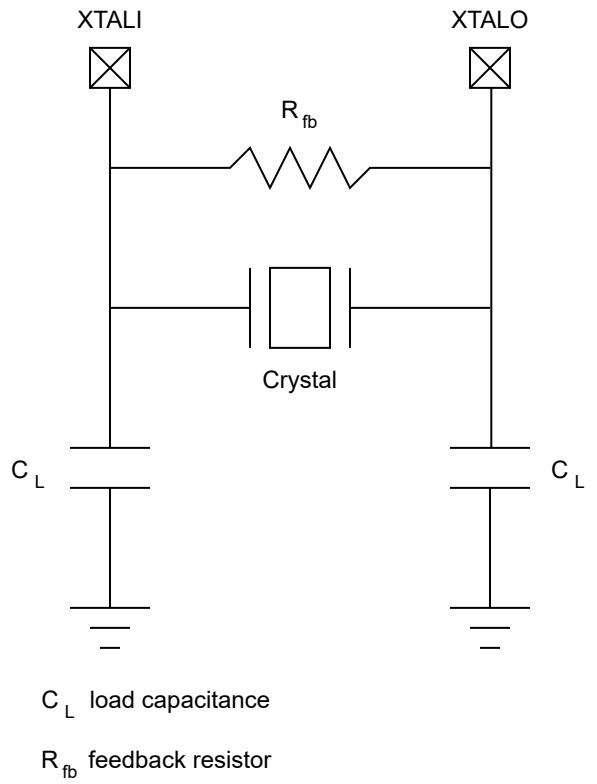


Figure 89. Crystal Oscillator block diagram

NOTE

The Rfb resistor is used in high gain mode. It is not used in low power mode.

24.3.1.2 XTALOSC Functional Description

This section details the oscillator clock sources to the chip.

24.3.1.2.1 Crystal Oscillators

The XTALOSC has two external crystal oscillators:

- **24 MHz XTAL oscillator**
  - Primary clock source for all PLLs that generate the clocks for the CPU, bus, and high-speed interfaces.
  - This clock source can be bypassed by the user, and replaced with another external clock source.
- **32 kHz XTAL oscillator**
  - Primary clock source for RTC as well as low-speed clock source for CCM/SRC/GPC.

24.3.1.2.1.1 Bypass configuration (24 MHz XTAL oscillator)

If it is desired to drive the chip with an external clock source, then it should be connected to XTALI and OSC_24M_CTRL[BYPASS_EN] should be set to 1. XTALO should be left externally floating.

24.3.1.2.2 RC Oscillators

The XTALOSC is supported by RC oscillators (RCOSC). This chip has three RCOSC:

- **24 MHz RC oscillator**
  - Generates the 24MHz clock source for the chip during start-up when the external 24 MHz crystal is not ready.
  - Alternative 24 MHz clock source in low power mode when the crystal oscillator is turned off, or in system which does not have any crystal oscillator.
- **32 kHz RC oscillator**
  - 32 kHz clock source for the chip during start-up when the external 32 kHz crystal is not ready.
  - Alternative 32 kHz clock source when the external 32 kHz crystal oscillator is not stable, or in system which does not have any crystal oscillator.
- **400 MHz RC oscillator**
  - Clock source during chip boot up before PLL is enabled or in low-power mode when PLL is turned off.

NOTE

- The switch from external 32 kHz XTALOSC to internal 32 kHz RCOSC is controlled by hardware and occurs automatically when the system detects a loss of 32 kHz crystal clock.
- To properly switch between 24 MHz XTALOSC and 24 MHz internal RCOSC, ensure the PLL is powered down before the switch. After switching the clock, power up PLL.

24.3.1.3 XTALOSC External Signals

The table found here describes the external signals of XTALOSC:

Table 156. XTALOSC External Signals

Signal	Description	Pad	Mode	Direction
RTC_XTALI	Real-time clock crystal oscillator input	RTC_XTALI	No muxing	I
RTC_XTALO	Real-time clock crystal oscillator output	RTC_XTALO	No muxing	O
XTALI	Crystal oscillator input signal	XTALI	No muxing	I
XTALO	Crystal oscillator output signal	XTALO	No muxing	O

24.3.1.4 XTALOSC Initialization

In software control mode, the enable sequence for OSC24M is:

1. Set OSC_24M_CTRL[OSC_EN] = 1
2. Wait for OSC_24M_CTRL[OSC_24M_STABLE] = 1
3. Set OSC_24M_CTRL[OSC_24M_GATE] = 0

The disable sequence for OSC24M is:

1. Set OSC_24M_CTRL[OSC_24M_GATE] = 1
2. Set OSC_24M_CTRL[OSC_EN] = 0

In software control mode, the enable sequence for RC_24M is:

- Set OSC_RC24M_CTRL[TEN] = 1

The disable sequence for RC_24M is:

- Set OSC_RC24M_CTRL[TEN] = 0

In hardware control mode, the power-on/power-off of the oscillators are fully controlled by CCM/GPC. The other settings remain under the control of XTALOSC.

24.3.1.5 XTALOSC Memory Map and register definition

Please see the OSC Memory Map section in this ANADIG chapter for more information about the XTALOSC registers.

NOTE

For related IPS Domain Slot control, see IPS Domain Registers in this chapter.

24.3.2 Power Management Unit (PMU)

The Power Management Unit (PMU) is designed to simplify the external power interface.

24.3.2.1 PMU Block Diagram

The power architecture of the chip is illustrated in the figure below, which shows the typical use case of a single 3.3V power supply and a coin cell battery. Depending on the application, different power supply schemes can apply (for example, DCDC bypassed).

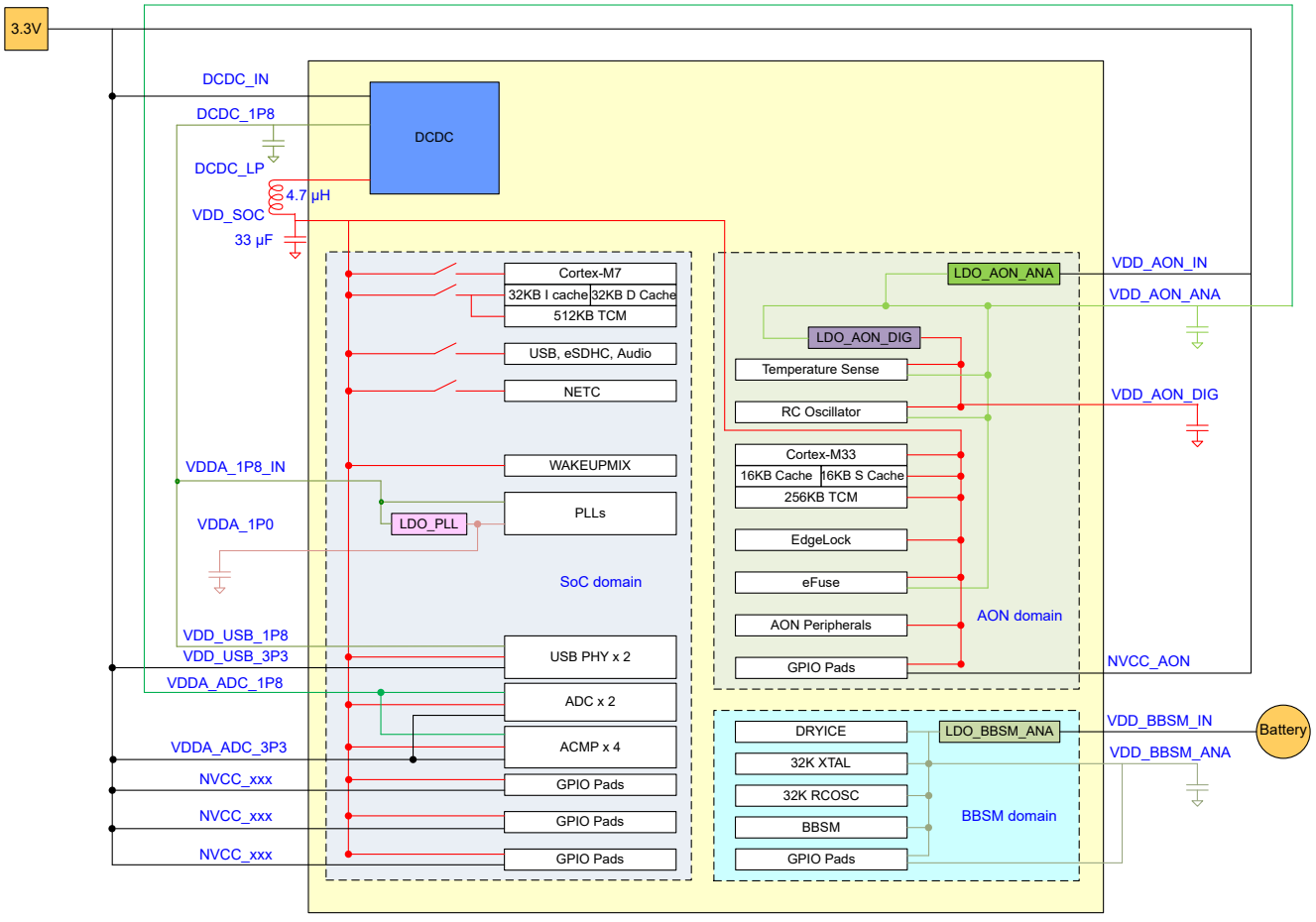


Figure 90. Power system overview

24.3.2.2 PMU Overview

The PMU has the following components integrated for power management:

- One DCDC to generate core power supply, with dynamic voltage scaling capability
- LDOs to generate power for internal logics
- Multiple Power Switches for sophisticated power mode management

24.3.2.3 PMU Functional Description

The PMU operations are detailed in the sections below.

24.3.2.3.1 PMU LDO Regulators

The PMU has four LDO regulators. Two LDOs are in the AON Domain covering the 1.8 V Analog source (LDO_AON_ANA) and 1.0 V Digital source (LDO_AON_DIG). One LDO covers the SoC PLLs in the SoC Domain, and one LDO in the BBSM domain, which is supplied by the VDD_BBSM_IN rail from the battery backed source.

24.3.2.3.2 PMU Well-Bias configuration

The well-bias configuration is shown below. The PWELL is an adaptive, quiescent consumption, positive voltage tracker, switched regulator. The PWELL regulator is biased as FBB. The PWELL regulator is connected to a LVT transistors.

- For low-voltage threshold (LVT), FBB is supported. It provides enhanced SoC core performance at a cost of higher leakage.

The VDD_SOC_IN and VDD_AON_ANA supplies need to be settled before the well-bias regulator is turned on (WB_EN) for FBB.

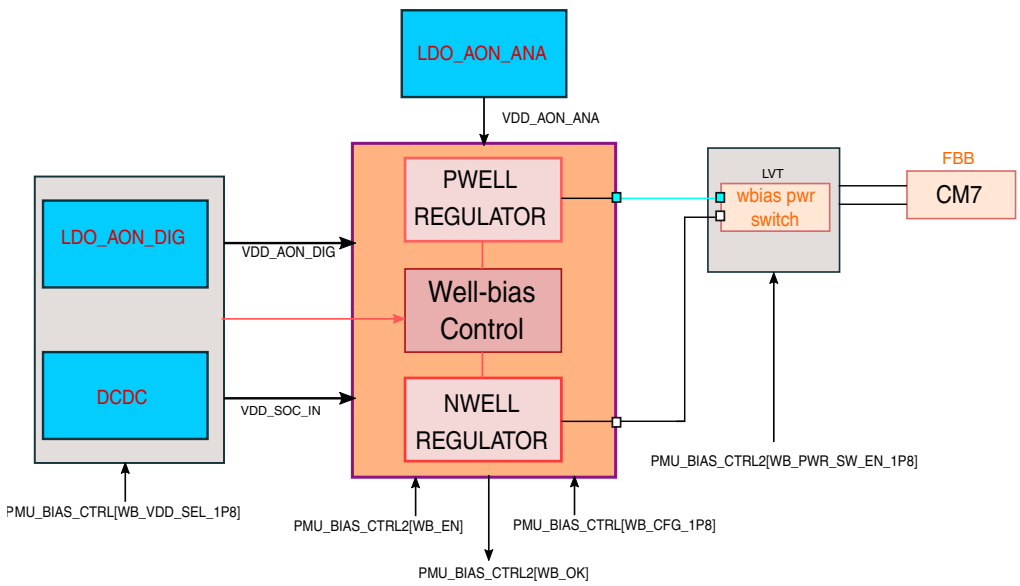


Figure 91. Well-Biasing Block Diagram

NOTE

The VDD selection (WB_VDD_SEL_1P8) needs to be done before enabling well-bias.

24.3.2.3.2.1 Body Biasing

Forward Body Biasing (FBB)is implemented to boost performance. The implementation of FBB is listed below:

- FBB is implemented in Cortex-M7 Platform in order to achieve higher performance

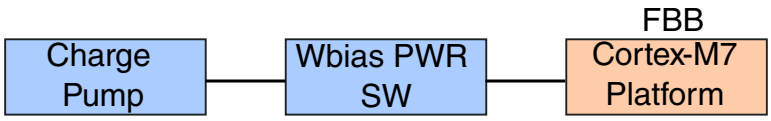


Figure 92. Body Biasing Diagram

24.3.2.3.3 PMU Control Mode

The PMU supports two control modes that can be configured by x_CONTROL_MODE fields. These two modes are Software control mode and Hardware control mode.

- The Software control mode enables LDOs, switches and bandgap to be fully configured by registers.
- The Hardware control mode enables the ON/OFF of the LDOs, LDO bypass switches and bandgap to be configured by GPC. The voltage of AON_DIG and operation modes of LDOs (lp_mode) are also configured by GPC. The other settings remain unchanged.

The following table shows the functions controlled by GPC:

Table 157. Functions controlled by GPC

IP	ON/OFF Control	Operation Mode	Tracking Mode	Bypass Mode (Switch ON/ OFF)	Standby Mode (Low Power Mode)	Voltage Control
LDO_PLL	Y	-	-	-	Y	-
AON_ANA	-	HP/LP	Y	Y	HP/LP	-
AON_DIG	-	HP/LP	Y	Y	HP/LP	Y
REFTOP (Bandgap)	Y	-	-	-	Y	-
FBB M7	Y	-	-	-	Y	-

Each time there is a change, there will be two steps of requests - voltage down and voltage up. The new target is compared with the current setting to check if the voltage change should happen at each step. When there is a change request, the 'done' signals will be generated until all the settings have been changed. When there is a sleep in request, the acknowledge signal will be held high until the set modules are put into low-power mode. When there is sleep out request, the acknowledge signal will held high until the modules are brought back from low power mode.

24.3.2.3.3.1 PMU ON/OFF Control Mode

The ON/OFF control for LDOs:

- When LDO_x_CONTROL_MODE = 0 (Software control mode), LDO_x_ENABLE controls the ON/OFF

Bandgap ON/OFF Control:

- When REF_CONTROL_MODE = 0 (Software control mode), REF_ENABLE controls the ON/OFF of the bandgap

NOTE

Software mode supports flexible control, but care and consideration should be followed in regards to analog dependencies and power structure sequences. In GPC mode, everything is handled precisely by hardware.

24.3.2.3.3.2 PMU Well Bias Enable Sequence

The following sequence should be followed to enable Well Bias:

- Set well bias setting (voltage, LVT)
  - PMU_BIAS_CTRL[1] is set to 1 (NWELL is configured to supply and LVT CORE, FBB)
  - PMU_BIAS_CTRL[12] is set to 1 to enable FBB correctly
  - All the other bits are set to 0
- Turn on CM7 FBB switch, and well bias (must be executed in the same step)
  - PMU_BIAS_CTRL2[WB_EN] is set to 1
  - PMU_BIAS_CTRL2[WB_PWR_SW_EN_1P8] is set to 1
  - All other bits set to 0
- Check the PMU_BIAS_CTRL2[WB_OK] bit to confirm Well Bias is stable

24.3.2.4 Clocks

The PMU doesn't have any relevant application clock sources.

24.3.2.5 PMU External Signals

See the supply contact assignments in the data sheet for more information.

24.3.2.6 PMU Memory Map and register definition

Please see the PMU Memory Map section in this ANADIG chapter for more information about the PMU registers.

24.3.3 Phase Locked Loops (PLLs) - Programming Guidelines

This chip contains the following Fractional and Integer PLLs:

- Fractional PLLs:
  - SYS_PLL1 (1G_PLL)
  - AUDIO_PLL
- Integer PLLs:
  - SYS_PLL2 (528_PLL)
  - SYS_PLL3 (480_PLL)
  - ARM_PLL

The PLLs can work in either GPC mode or Software mode (default). In GPC mode, GPC configures the PLLs. In Software mode, the users can configure all the registers.

The equations given below help to calculate the output frequency for PLLs based on the mode (Software or GPC) and type of PLL (Fractional or Integer).

- During Software mode (default):
  - For Fractional PLLs (SYS_PLL1, AUDIO_PLL):

- Output frequency = Fref*(CTRL0[DIV_SELECT]+(NUMERATOR[NUM]/DENOMINATOR[DENOM]))/ 2^CTRL0[POST_DIV_SEL]

— For Integer PLLs (SYS_PLL2):

- Output frequency = Fref * (MFI+MFN/MFD)

— For Integer PLLs (SYS_PLL3):

- Output frequency = Fref * div_sel

— For Integer PLLs (ARM_PLL):

- Output frequency = Fref * ARM_PLL_CTRL[DIV_SELECT]/2 * ARM_PLL_CTRL[POST_DIV_SELECT]

- During GPC mode:

— For Fractional PLLs (SYS_PLL1, AUDIO_PLL):

- Output frequency = Fref * (SYS_PLLx_DIV_SELECT[DIV_SELECT] + (SYS_PLLx_NUMERATOR[NUM]/ SYS_PLLx_DENOMINATOR[DENOM]))

— For Integer PLLs (SYS_PLL2):

- Output frequency = Fref * (MFI+MFN/MFD)

— For Integer PLLs (SYS_PLL3):

- Output frequency = Fref * div_sel

— For Integer PLLs (ARM_PLL):

- Output frequency = Fref * ARM_PLL_CTRL[DIV_SELECT]/2 * ARM_PLL_CTRL[POST_DIV_SELECT]

Fref is the PLL Reference Frequency sourced from 24 MHz.

NOTE

For SYS_PLL2, MFI is a fixed value of 22 and for SYS_PLL3, div_sel is a fix value of 20.

24.3.3.1 PLL - Spread Spectrum Parameters

The SYS_PLL1, SYS_PLL2, AUDIO_PLL and the related PFDs support spread spectrum (down spread). The equations given below help to calculate spread spectrum range, and modulation frequency based on the mode (Software or GPC) and type of PLL (Fractional or Integer).

- During Software mode (default):

— For Fractional PLLs (SYS_PLL1, AUDIO_PLL):

- Spread spectrum range = Fref * SPREAD_SPECTRUM[STOP]/DENOMINATOR[DENOM]
- Modulation frequency = Fref * SPREAD_SPECTRUM[STEP]/(2*SPREAD_SPECTRUM[STOP])

— For Integer PLL (SYS_PLL2):

- Spread spectrum range = Fref * SYS_PLL2_SS[STOP]/SYS_PLL2_MFD[MFD]
- Modulation frequency= Fref * SYS_PLL2_SS[STEP]/(SYS_PLL2_SS[STOP] *2)

- During GPC mode:

— For Fractional PLLs (SYS_PLL1, AUDIO_PLL):

- Spread spectrum range = Fref * SYS_PLLx_SS[STOP]/SYS_PLLx_DENOMINATOR[DENOM]
- Modulation frequency = Fref * SYS_PLLx_SS[STEP]/(2*SYS_PLLx_SS[STOP])

— For Integer PLL (SYS_PLL2):

- Spread spectrum range = Fref * SYS_PLL2_SS[STOP]/SYS_PLL2_MFD[MFD]
- Modulation frequency= Fref * SYS_PLL2_SS[STEP]/(SYS_PLL2_SS[STOP] *2)

Fref is the PLL Reference Frequency sourced from 24MHz.

The spread range indicates how much frequency the PLL will sweep down. For example, a spread range of 6MHz would mean that the PLL will sweep from 'Target_frequency' to 'Target_Frequency - 6MHz' and sweep back.

24.3.4 TMPSNS temperature trim value

The ANADIG TEMPSENSOR registers contain the temperature value at 25C. This is a read-only value that comes from the fuses.

Other TMPSNS information can be found in the TMPSNS chapter.

24.4 Memory Map and register definition

The memory map register definition for XTALOSC, LDOs, PLLs, TMPSNS (analog), and IPS Domain Slot controls are shown below.

24.4.1 RT1180_ANADIG_REGISTER register descriptions

24.4.1.1 LDO_BBSM memory map

ANADIG base address: 4448_0000h

Offset	Register	Width (In bits)	Access	Reset value
4740h	PMU_LDO_AON_ANA_REGISTER (PMU_LDO_AON_ANA)	32	RW	0000_0108h
4760h	PMU_LDO_AON_DIG_REGISTER (PMU_LDO_AON_DIG)	32	RW	0130_1C05h

24.4.1.2 MISC memory map

ANADIG base address: 4448_0000h

Offset	Register	Width (In bits)	Access	Reset value
4800h	Chip Silicon Version Register (MISC_DIFPROG)	32	R	0090_0200h

24.4.1.3 OSC memory map

ANADIG base address: 4448_0000h

Offset	Register	Width (In bits)	Access	Reset value
4310h	24MHz RCOSC Control Register (OSC_RC24M_CTRL)	32	RW	0079_01F2h
4320h	24MHz OSC Control Register (OSC_24M_CTRL)	32	RW	0000_0080h
4340h	400MHz RCOSC Control0 Register (OSC_400M_CTRL0)	32	RW	8000_0000h
4350h	400MHz RCOSC Control1 Register (OSC_400M_CTRL1)	32	RW	0000_0001h

24.4.1.4 PLL memory map

ANADIG base address: 4448_0000h

Offset	Register	Width (In bits)	Access	Reset value
4000h	ARM_PLL_CTRL_REGISTER (ARM_PLL_CTRL)	32	RW	4000_00A6h
4010h	SYS_PLL3_CTRL_REGISTER (SYS_PLL3_CTRL)	32	RW	4000_0003h
4020h	SYS_PLL3_UPDATE_REGISTER (SYS_PLL3_UPDATE)	32	RW	0000_0000h
4030h	SYS_PLL3_PFD_REGISTER (SYS_PLL3_PFD)	32	RW	8CA0_918Dh
4040h	SYS_PLL2_CTRL_REGISTER (SYS_PLL2_CTRL)	32	RW	4000_0000h
4050h	SYS_PLL2_UPDATE_REGISTER (SYS_PLL2_UPDATE)	32	RW	0000_0000h
4060h	SYS_PLL2_SS_REGISTER (SYS_PLL2_SS)	32	RW	0000_0000h
4070h	SYS_PLL2_PFD_REGISTER (SYS_PLL2_PFD)	32	RW	A098_909Bh
4080h	SYS_PLL2_MFN_REGISTER (SYS_PLL2_MFN)	32	RW	0000_0000h
4090h	SYS_PLL2_MFI_REGISTER (SYS_PLL2_MFI)	32	RW	0000_0016h
40A0h	SYS_PLL2_MFD_REGISTER (SYS_PLL2_MFD)	32	RW	0FFF_FFFFh
4100h	SYS_PLL1_CTRL_REGISTER (SYS_PLL1_CTRL)	32	RW	0000_4000h
4200h	PLL_AUDIO_CTRL_REGISTER (PLL_AUDIO_CTRL)	32	RW	0000_4000h

24.4.1.5 PMU memory map

ANADIG base address: 4448_0000h

Offset	Register	Width (In bits)	Access	Reset value
4600h	PMU_BIAS_CTRL_REGISTER (PMU_BIAS_CTRL)	32	RW	0000_8000h
4610h	PMU_BIAS_CTRL2_REGISTER (PMU_BIAS_CTRL2)	32	RW	0000_0000h
4640h	PMU_LDO_PLL_REGISTER (PMU_LDO_PLL)	32	RW	0000_0005h
4700h	PMU_POWER_DETECT_CTRL_REGISTER (PMU_POWER_DETECT_CTRL)	32	RW	0000_0000h
4710h	PMU_REF_CTRL_REGISTER (PMU_REF_CTRL)	32	RW	0000_0040h

24.4.1.6 TEMPSENSOR memory map

ANADIG base address: 4448_0000h

Offset	Register	Width (In bits)	Access	Reset value
4530h	TEMPSNS_OTP_TRIM_VALUE_REGISTER (TEMPSNS_OTP_TRIM_VALUE)	32	R	0000_0000h

24.4.1.7 ARM_PLL_CTRL_REGISTER (ARM_PLL_CTRL)

Offset

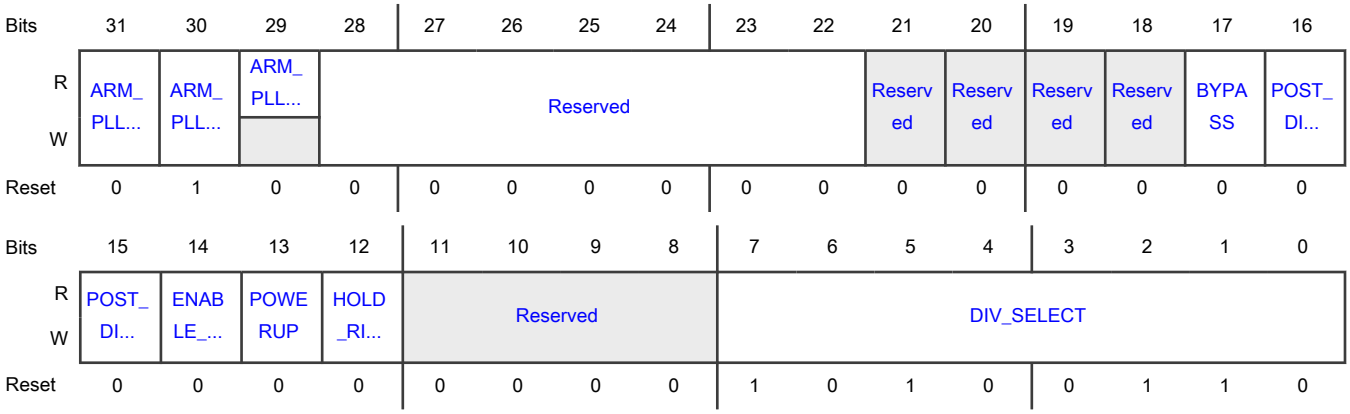
Register	Offset
ARM_PLL_CTRL	4000h

Function

ARM_PLL control register

The control register provides control for the ARM PLL.

Diagram



Fields

Field	Function
31 ARM_PLL_CONTROL_MODE	pll_arm_control_mode Enable mode 0b - Software Mode (Default) 1b - GPC Mode
30 ARM_PLL_GATE	ARM_PLL_GATE default value is 1'b1 0b - Clock is not gated

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is gated
29 ARM_PLL_STABLE	ARM_PLL_STABLE The stable indicate bit. Normally in Software mode, after the power up and enable sequence for arm pll, arm pll will not be in stable state immediately. This bit is used by software to monitor the locking status for arm pll. GPC mode do not need to take care of this bit. Hardware will handle it automatically. 0b - ARM PLL is not stable 1b - ARM PLL is stable
28-22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 BYPASS	Bypass the pll. Bypass the pll enable bit. This bit could be used to bypass the PLL output to the source of reference clock 0b - Function mode 1b - Bypass Mode
16-15 POST_DIV_SEL	POST_DIV_SEL 00b - post_div=2 01b - post_div=4 10b - post_div=8 11b - post_div=1
14 ENABLE_CLK	Enable the clock output. Please refer to 'PLL Enable Sequence' topic for more details on how to use this bit during the PLL enablement. 0b - Disable the clock 1b - Enable the clock

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 POWERUP	Power up the PLL Please refer to 'PLL Enable Sequence' topic for more details on how to use this bit during the PLL enablement. 0b - Power down the PLL 1b - Power Up the PLL
12 HOLD_RING_OFF	PLL Start up initialization This field should be set to 1 every time during PLL lock, and cleared after a certain delay. Refer to 'PLL enable Sequence' for details. 0b - Normal operation 1b - Initialize PLL start up
11-8 —	Reserved
7-0 DIV_SELECT	DIV_SELECT This field controls the pll loop divider. Valid range for divider value: 104-208. output_fre = fref * div_select/(2 * post_div)

24.4.1.8 SYS_PLL3_CTRL_REGISTER (SYS_PLL3_CTRL)

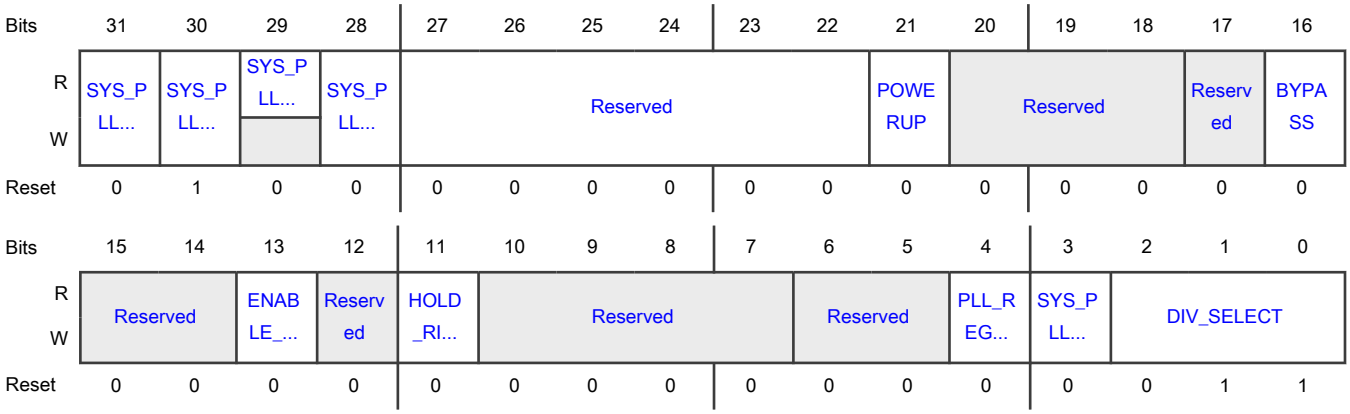
Offset

Register	Offset
SYS_PLL3_CTRL	4010h

Function

The control register provides control for the 480 PLL.

Diagram



Fields

Field	Function
31 SYS_PLL3_CONTROL_MODE	SYS_PLL3_control_mode Enable mode 0b - Software Mode (Default) 1b - GPC Mode
30 SYS_PLL3_GATE	SYS_PLL3_GATE default value is 1'b1 0b - Clock is not gated 1b - Clock is gated
29 SYS_PLL3_STABLE	SYS_PLL3_STABLE The stable indicate bit. Normally in Software mode, after the power up and enable sequence for SYS_PLL3, arm pll will not be in stable state immediately. This bit is used by software to monitor the locking status for SYS_PLL3. 0b - Not Stable 1b - Stable
28 SYS_PLL3_DIV2_CONTROL_MODE	SYS_PLL3_DIV2_CONTROL_MODE Enable mode 0b - Software Mode (Default) 1b - GPC Mode
27-22 —	Reserved
21 POWERUP	Power up the PLL

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Please refer to 'PLL Enable Sequence' topic for more details on how to use this bit during the PLL enablement. 0b - Power down the PLL 1b - Power Up the PLL
20-18 —	Reserved
17 —	Reserved
16 BYPASS	BYPASS Bypass the pll enable bit. This bit could be used to bypass the PLL output to the source of reference clock 0b - Function mode 1b - Bypass Mode
15-14 —	Reserved
13 ENABLE_CLK	Enable the clock output. Please refer to 'PLL Enable Sequence' topic for more details on how to use this bit during the PLL enablement. 0b - Disable the clock 1b - Enable the clock
12 —	Reserved
11 HOLD_RING_OFF	PLL Start up initialization This field should be set to 1 every time during PLL lock, and cleared after a certain delay. Refer to 'PLL enable Sequence' for details. 0b - Normal operation 1b - Initialize PLL start up
10-7 —	Reserved
6-5 —	Reserved
4	Enable Internal PLL Regulator

Table continues on the next page...

Table continued from the previous page...

Field	Function
PLL_REG_EN	See 'PLL Enable Sequence' topic for more details on how to use this bit during the PLL enablement.
3 SYS_PLL3_DIV2	SYS PLL3 DIV2 gate Enable SYS_PLL3_DIV2 clock which is sourced from SYS_PLL3 0 : disable 1 : enable
2-0 DIV_SELECT	DIV_SELECT This field controls the pll loop divider. Valid range for divider value: 54-108. output_fre = fref * div_select. 000b - div_select=130x1: div_select=150x2 001b - div_select=160x3: div_select=200x4 010b - div_select=220x5: div_select=250x6 011b - div_select=300x7: div_select=240

24.4.1.9 SYS_PLL3_UPDATE_REGISTER (SYS_PLL3_UPDATE)

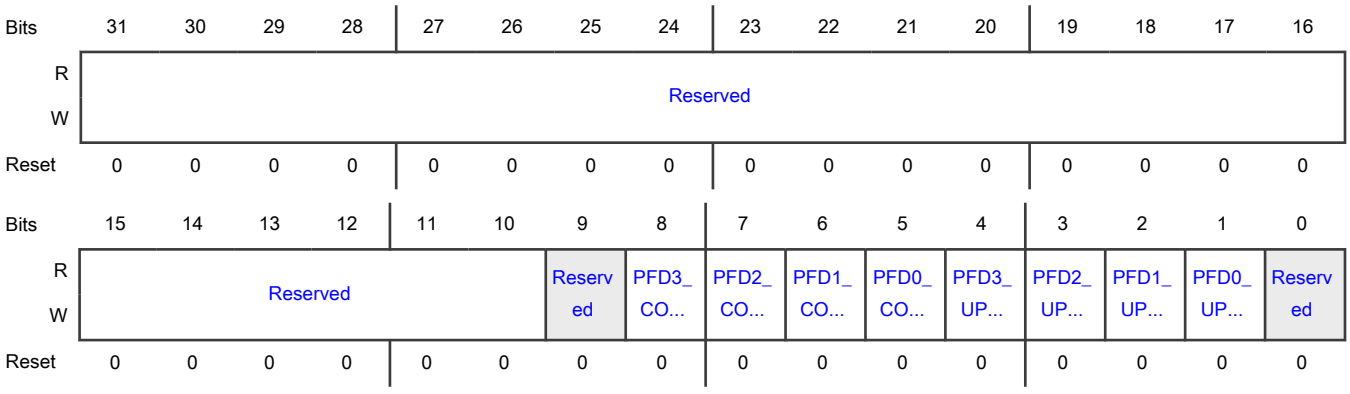
Offset

Register	Offset
SYS_PLL3_UPDATE	4020h

Function

The control register provides control for the 480 PLL.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 PFD3_CONTROL_MODE	pfd3_control_mode Enable mode 0b - Software Mode (Default) 1b - GPC Mode
7 PFD2_CONTROL_MODE	pdf2_control_mode Enable mode 0b - Software Mode (Default) 1b - GPC Mode
6 PFD1_CONTROL_MODE	pfd1_control_mode Enable mode 0b - Software Mode (Default) 1b - GPC Mode
5 PFD0_CONTROL_MODE	pfd0_control_mode Enable mode 0b - Software Mode (Default) 1b - GPC Mode
4 PFD3_UPDATE	PFD3_UPDATE PFD update This bit is used to update the pfd value. Toggle it will make the update function.
3 PFD2_UPDATE	PFD2_OVERRIDE PFD update This bit is used to update the pfd value. Toggle it will make the update function.
2 PFD1_UPDATE	PFD1_OVERRIDE PFD update This bit is used to update the pfd value. Toggle it will make the update function.
1 PFD0_UPDATE	PFD0_OVERRIDE PFD update

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit is used to update the pfd value. Toggle it will make the update function.
0 —	Reserved

24.4.1.10 SYS_PLL3_PFD_REGISTER (SYS_PLL3_PFD)

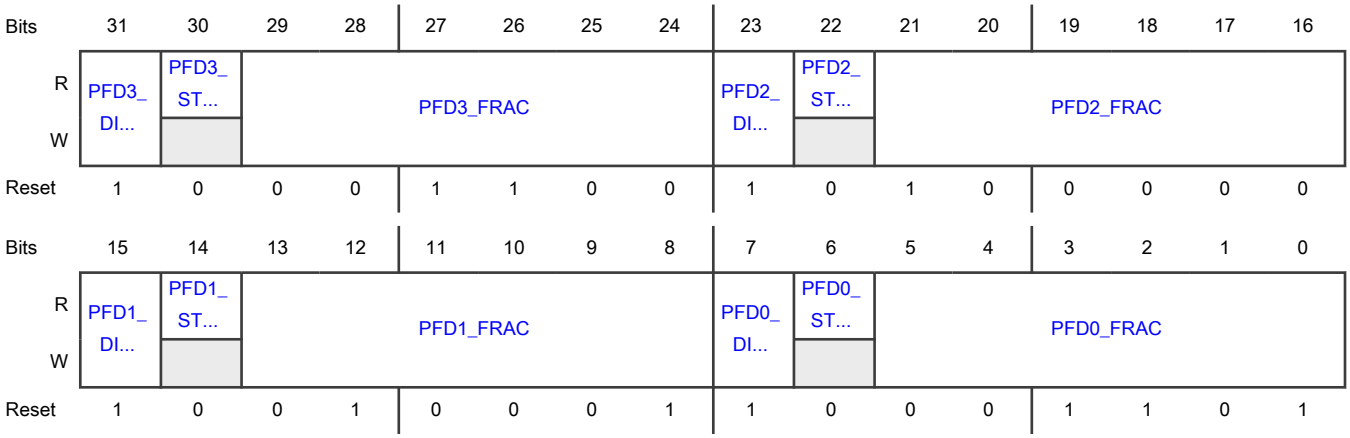
Offset

Register	Offset
SYS_PLL3_PFD	4030h

Function

This register contains the numerator of SYS PLL3 fractional loop divider.

Diagram



Fields

Field	Function
31 PFD3_DIV1_CLKGATE	PFD3_CLKGATE 0b - PFD3 fractional divider clock is enabled 1b - Fractional divider clock (reference PFD3) is off (power savings)
30 PFD3_STABLE	PFD3_STABLE This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit,

Table continues on the next page...

Table continued from the previous page...

Field	Function
	program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
29-24 PFD3_FRAC	PFD3_FRAC This field controls the fractional divide value. The resulting frequency shall be 480*18/PFD3_FRAC where PFD3_FRAC is in the range 13-35. **NOTE** The PFD value can be set to 12, however the lowest recommended setting for PFD is 13.
23 PFD2_DIV1_CLKGATE	PFD2_CLKGATE 0b - PFD2 fractional divider clock is enabled 1b - Fractional divider clock (reference PFD2) is off (power savings)
22 PFD2_STABLE	PFD2_STABLE This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
21-16 PFD2_FRAC	PFD2_FRAC This field controls the fractional divide value. The resulting frequency shall be 480*18/PFD2_FRAC where PFD2_FRAC is in the range 13-35.
15 PFD1_DIV1_CLKGATE	PFD1_CLKGATE 0b - PFD1 fractional divider clock is enabled 1b - Fractional divider clock (reference PFD1) is off (power savings)
14 PFD1_STABLE	PFD1_STABLE This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
13-8 PFD1_FRAC	PFD1_FRAC This field controls the fractional divide value. The resulting frequency shall be 480*18/PFD1_FRAC where PFD1_FRAC is in the range 13-35.
7 PFD0_DIV1_CLKGATE	PFD0_CLKGATE 0b - PFD0 fractional divider clock is enabled 1b - Fractional divider clock (reference ref_pfd0) is off (power savings
6	PFD0_STABLE

Table continues on the next page...

Table continued from the previous page...

Field	Function
PFD0_STABLE	This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
5-0 PFD0_FRAC	PFD0_FRAC This field controls the fractional divide value. The resulting frequency shall be 480*18/PFD0_FRAC where PFD0_FRAC is in the range 13-35.

24.4.1.11 SYS_PLL2_CTRL_REGISTER (SYS_PLL2_CTRL)

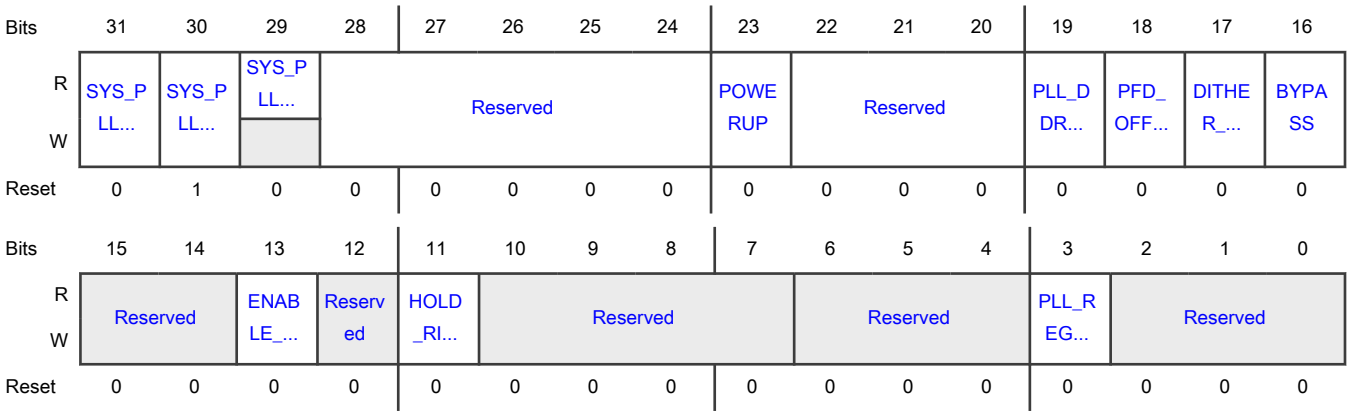
Offset

Register	Offset
SYS_PLL2_CTRL	4040h

Function

The control register provides control for the SYS_PLL2

Diagram



Fields

Field	Function
31 SYS_PLL2_CONTROL_MODE	SYS_PLL2_control_mode SYS_PLL2 has two mode to for enable. One is software mode, the other is GPC mode. 0b - Software Mode (Default) 1b - GPC Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 SYS_PLL2_GATE	SYS_PLL2_GATE default value is 1'b1 0b - Clock is not gated 1b - Clock is gated
29 SYS_PLL2_STABLE	SYS_PLL2_STABLE The stable indicate bit. Normally in Software mode, after the power up and enable sequence for SYS_PLL2, ARM_PLL will not be in stable state immediately. This bit is used by software to monitor the locking status for SYS_PLL2.
28-24 —	Reserved
23 POWERUP	Power up the PLL Please refer to 'PLL Enable Sequence' topic for more details on how to use this bit during the PLL enablement. 0b - Power down the PLL 1b - Power Up the PLL
22-20 —	Reserved
19 PLL_DDR_OVERRIDE	PLL_DDR_OVERRIDE The OVERRIDE bit allows the clock control module to automatically override portions of the register. PLL_DDR_OVERRIDE = 0x0: ENABLE_CLK bits and the POWERDOWN bit controlled by this register. PLL_DDR_OVERRIDE = 0x1: CCM based hardware bits override the ENABLE_CLK bits and the POWERDOWN bit in this register.
18 PFD_OFFSET_EN	PFD_OFFSET_EN Enables an offset in the phase frequency detector.
17 DITHER_ENABLE	DITHER_ENABLE Enables dither in the fractional modulator calculation. 0b - Disable Dither 1b - Enable Dither
16 BYPASS	Bypass the pll. Control bit to Bypass the pll to the reference clock 0b - Function mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Bypass Mode
15-14 —	Reserved
13 ENABLE_CLK	Enable the clock output. 0b - Disable the clock 1b - Enable the clock
12 —	Reserved
11 HOLD_RING_OFF	PLL Start up initialization This field should be set to 1 every time during PLL lock, and cleared after a certain delay. Refer to 'PLL enable Sequence' for details. 0b - Normal operation 1b - Initialize PLL start up
10-7 —	Reserved
6-4 —	Reserved
3 PLL_REG_EN	Enable Internal PLL Regulator Please refer to 'PLL Enable Sequence' topic for more details on how to use this bit during the PLL enablement. 0b - Disable 1b - Enable
2-0 —	Reserved

24.4.1.12 SYS_PLL2_UPDATE_REGISTER (SYS_PLL2_UPDATE)

Offset

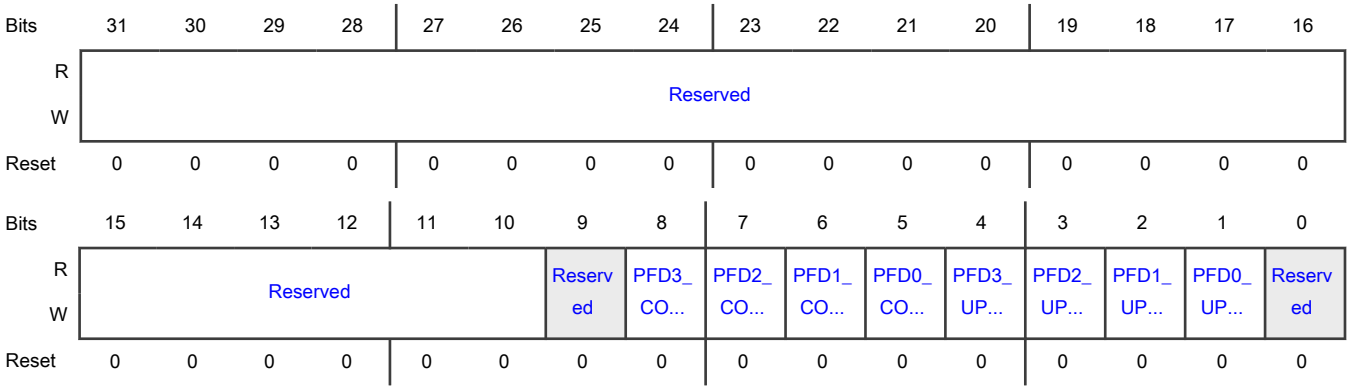
Register	Offset
SYS_PLL2_UPDATE	4050h

Function

SYS_PLL2 PFD control register

The control register provides control for the 528 PLL.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 PFD3_CONTROL_MODE	pfd3_control_mode Enable mode 0b - Software Mode (Default) 1b - GPC Mode
7 PFD2_CONTROL_MODE	pfd2_control_mode Enable mode 0b - Software Mode (Default) 1b - GPC Mode
6 PFD1_CONTROL_MODE	pfd1_control_mode Enable mode 0b - Software Mode (Default) 1b - GPC Mode
5 PFD0_CONTROL_MODE	pfd0_control_mode Enable mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Software Mode (Default) 1b - GPC Mode
4 PFD3_UPDATE	PFD3_UPDATE PFD update This bit is used to update the pfd value. Toggle it will make the update function.
3 PFD2_UPDATE	PFD2_UPDATE PFD update This bit is used to update the pfd value. Toggle it will make the update function.
2 PFD1_UPDATE	PFD1_UPDATE PFD update This bit is used to update the pfd value. Toggle it will make the update function.
1 PFD0_UPDATE	PFD0_UPDATE PFD update This bit is used to update the pfd value. Toggle it will make the update function.
0 —	Reserved

24.4.1.13 SYS_PLL2_SS_REGISTER (SYS_PLL2_SS)

Offset

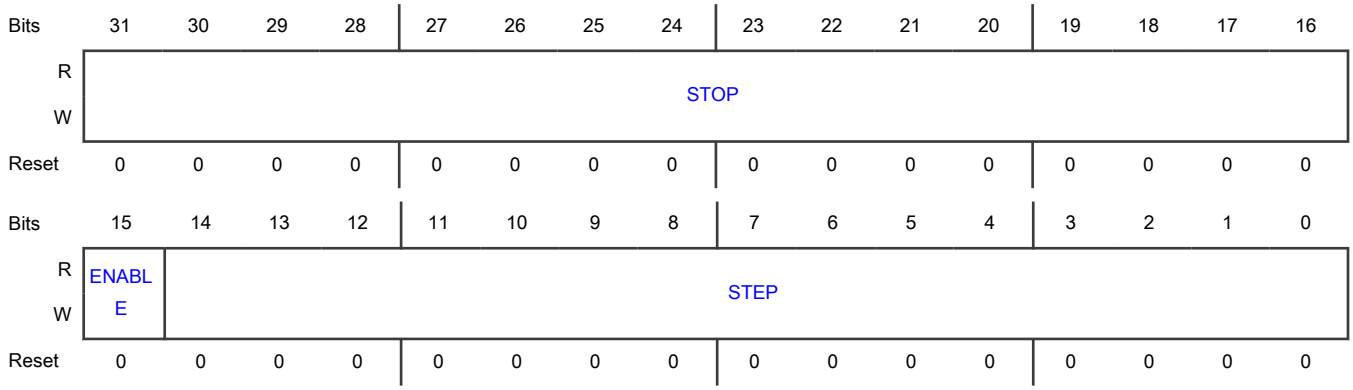
Register	Offset
SYS_PLL2_SS	4060h

Function

This register contains the SYS PLL2 spread spectrum controls

Refer to 'Spread Spectrum Parameters' topic for more details.

Diagram



Fields

Field	Function
31-16 STOP	STOP Frequency change = stop/MFD*24MHz.
15 ENABLE	ENABLE This bit enables the spread spectrum modulation 0b - Disable Spread Spectrum. In this setting, the values for STEP and STOP are ignored 1b - Enable Spread Spectrum
14-0 STEP	STEP The max frequency change = step/MFD*24MHz.

24.4.1.14 SYS_PLL2_PFD_REGISTER (SYS_PLL2_PFD)

Offset

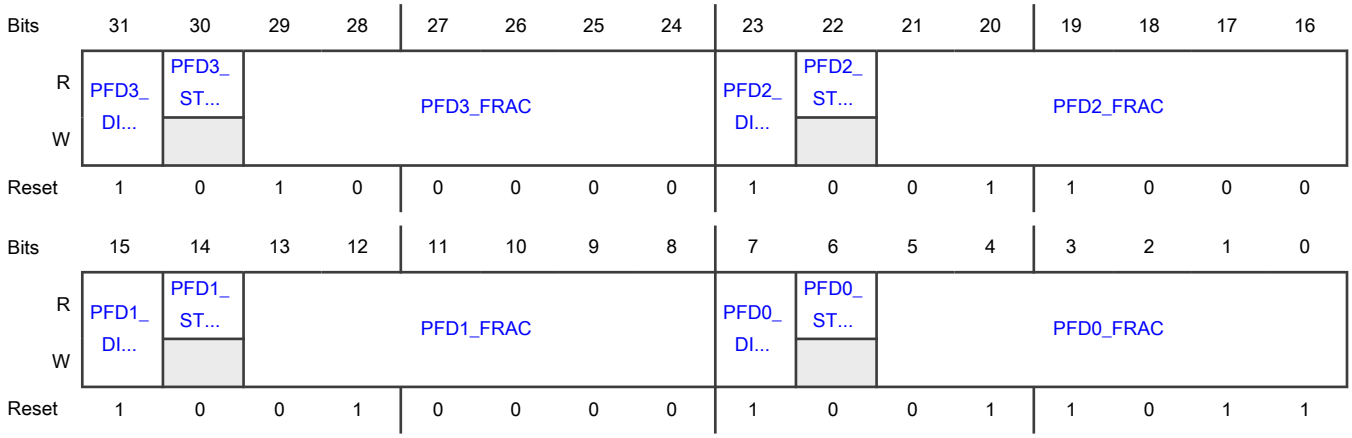
Register	Offset
SYS_PLL2_PFD	4070h

Function

SYS_PLL2 fractional loop divider

This register contains the numerator of DDR PLL fractional loop divider.

Diagram



Fields

Field	Function
31 PFD3_DIV1_CLKGATE	PFD3_CLKGATE 0b - PFD3 fractional divider clock is enabled. 1b - Fractional divider clock (reference PFD3) is off (power savings)
30 PFD3_STABLE	PFD3_STABLE This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
29-24 PFD3_FRAC	PFD3_FRAC This field controls the fractional divide value. The resulting frequency shall be 528*18/PFD3_FRAC where PFD3_FRAC is in the range 13-35.
23 PFD2_DIV1_CLKGATE	PFD2_CLKGATE 0b - PFD2 fractional divider clock is enabled. 1b - Fractional divider clock (reference PFD2) is off (power savings)
22 PFD2_STABLE	PFD2_STABLE This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
21-16 PFD2_FRAC	PFD2_FRAC This field controls the fractional divide value. The resulting frequency shall be 528*18/PFD2_FRAC where PFD2_FRAC is in the range 13-35.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 PFD1_DIV1_CLKGATE	PFD1_CLKGATE 0b - PFD1 fractional divider clock is enabled. 1b - Fractional divider clock (reference PFD1) is off (power savings)
14 PFD1_STABLE	PFD1_STABLE This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
13-8 PFD1_FRAC	PFD1_FRAC This field controls the fractional divide value. The resulting frequency shall be 528*18/PFD1_FRAC where PFD1_FRAC is in the range 13-35.
7 PFD0_DIV1_CLKGATE	PFD0_CLKGATE 0b - PFD0 fractional divider clock is enabled. 1b - Fractional divider clock (reference PFD0) is off (power savings)
6 PFD0_STABLE	PFD0_STABLE This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
5-0 PFD0_FRAC	PFD0_FRAC This field controls the fractional divide value. The resulting frequency shall be 528*18/PFD0_FRAC where PFD0_FRAC is in the range 13-35.

24.4.1.15 SYS_PLL2_MFN_REGISTER (SYS_PLL2_MFN)

Offset

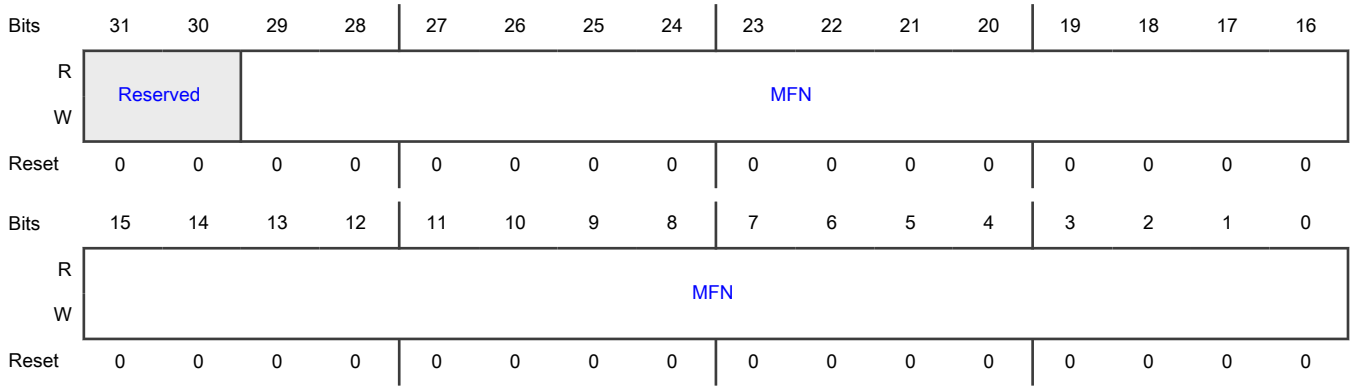
Register	Offset
SYS_PLL2_MFN	4080h

Function

SYS_PLL2_MFN PLL output frequency=Fref*(MFI+MFN/MFD)

Refer to Programming Guidelines for more details.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-0 MFN	MFN PLL output frequency=Fref*(MFI+MFN/MFD)

24.4.1.16 SYS_PLL2_MFI_REGISTER (SYS_PLL2_MFI)

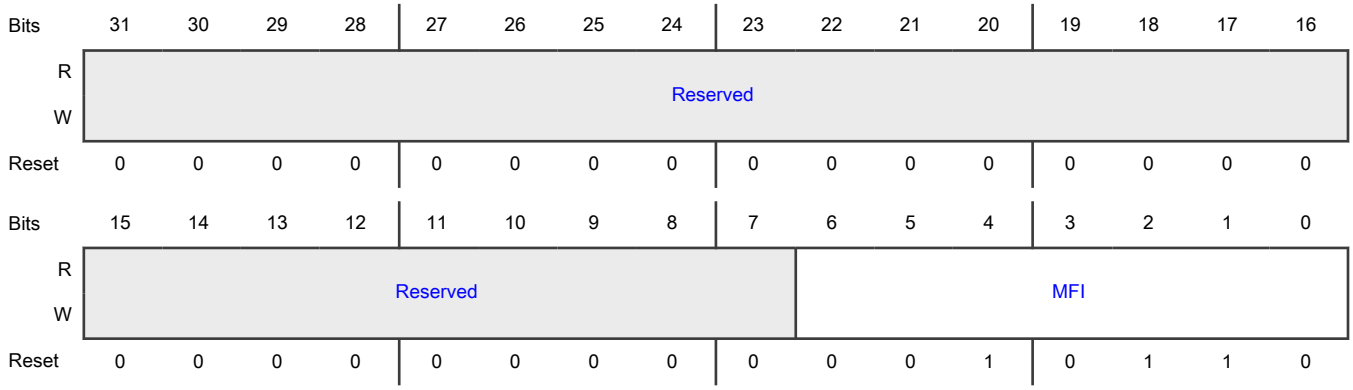
Offset

Register	Offset
SYS_PLL2_MFI	4090h

Function

SYS_PLL2_MFI PLL output frequency=Fref*(MFI+MFN/MFD)

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 MFI	MFI PLL output frequency=Fref*(MFI+MFN/MFD)

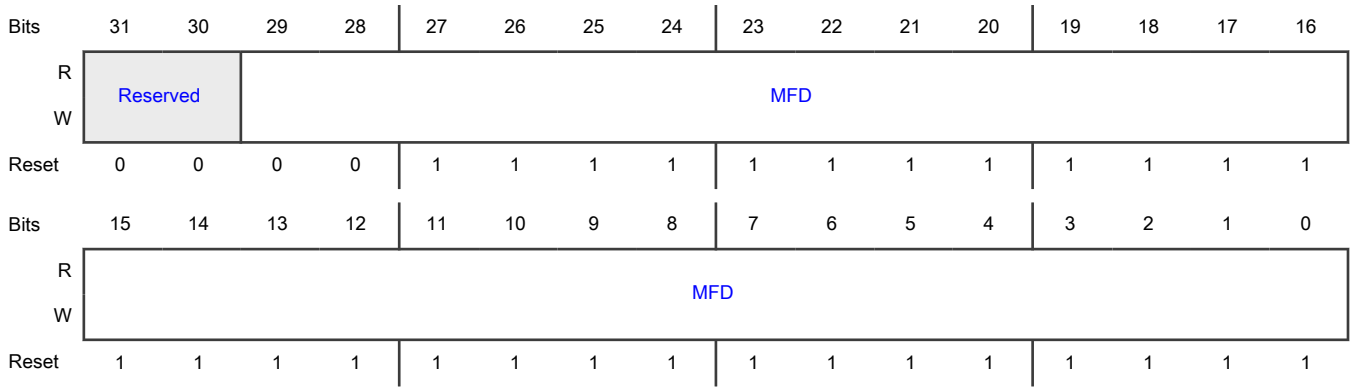
24.4.1.17 SYS_PLL2_MFD_REGISTER (SYS_PLL2_MFD)

Offset

Register	Offset
SYS_PLL2_MFD	40A0h

Function
Spread Spectrum Frequency Range is calculated as follows: Range = (STOP/MFD) * Fref

Diagram



Fields

Field	Function
31-30 —	Reserved
29-0 MFD	Denominator Refer to Programming Guidelines for more details.

24.4.1.18 SYS_PLL1_CTRL_REGISTER (SYS_PLL1_CTRL)

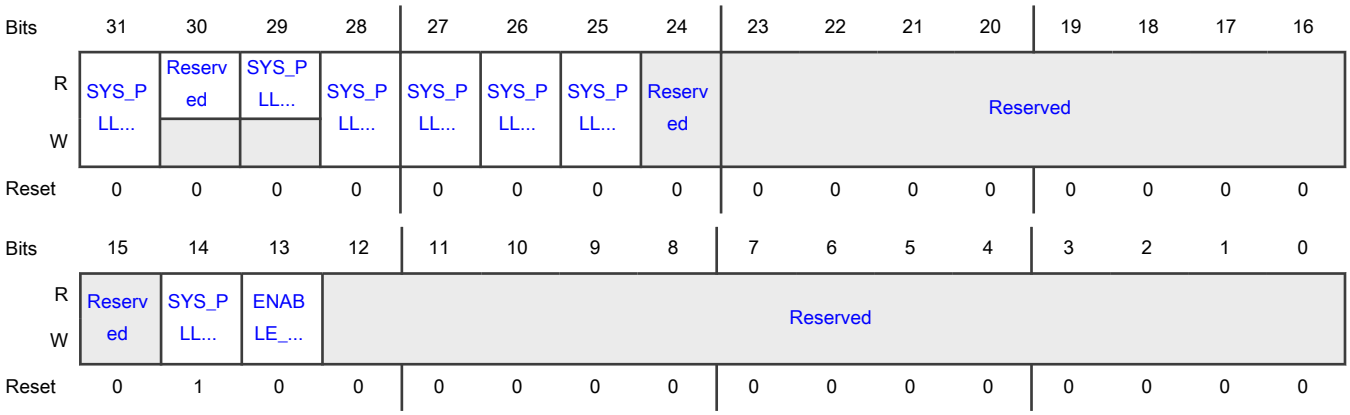
Offset

Register	Offset
SYS_PLL1_CTRL	4100h

Function

The control register provides control for the 1G PLL.

Diagram



Fields

Field	Function
31 SYS_PLL1_CONTROL_MODE	SYS_PLL1_CONTROL_MODE Enable mode 0b - Software Mode (Default) 1b - GPC Mode
30 —	Reserved
29 SYS_PLL1_STABLE	SYS_PLL1_STABLE Normally in Software mode, after the power up and enable sequence for pll, arm pll will not be in stable state immediately. This bit is used by software to monitor the locking status for pll. GPC mode do not need to take care of this bit. Hardware will handle it automatically. 0b - Not Stable 1b - Stable
28	SYS_PLL1_DIV2_CONTROL_MODE Enable mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYS_PLL1_DIV2_CONTROL_MODE	0b - Software Mode (Default) 1b - GPC Mode
27 SYS_PLL1_DIV5_CONTROL_MODE	SYS_PLL1_DIV5_CONTROL_MODE Enable mode 0b - Software Mode (Default) 1b - GPC Mode
26 SYS_PLL1_DIV5	SYS_PLL1_DIV5 SYS_PLL1_DIV5 The PLL 1g has the source which is DIV5 by the actual frequency of PLL 1G, this bit control of the enable of this divider. 0b - Disabled 1b - Enabled
25 SYS_PLL1_DIV2	SYS_PLL1_DIV2 The PLL 1g has the source which is DIV2 by the actual frequency of PLL 1G, this bit control of the enable of this divider. 0b - Disabled 1b - Enabled
24 —	Reserved
23-15 —	Reserved
14 SYS_PLL1_GATE	SYS_PLL1_GATE SYS_PLL1_GATE There is a gate in top level to gate the pll output for power saving purpose,this field it to control this gate. 0b - No gate 1b - Gate the output
13 ENABLE_CLK	ENABLE_CLK Enables the clock output. This field is used to align the software mode and GPC mode. The usage of this bit is to describe the last PLL mode(enable or disable) before entering GPC mode. 0b - Enable 1b - Disable
12-0 —	Reserved

24.4.1.19 PLL_AUDIO_CTRL_REGISTER (PLL_AUDIO_CTRL)

Offset

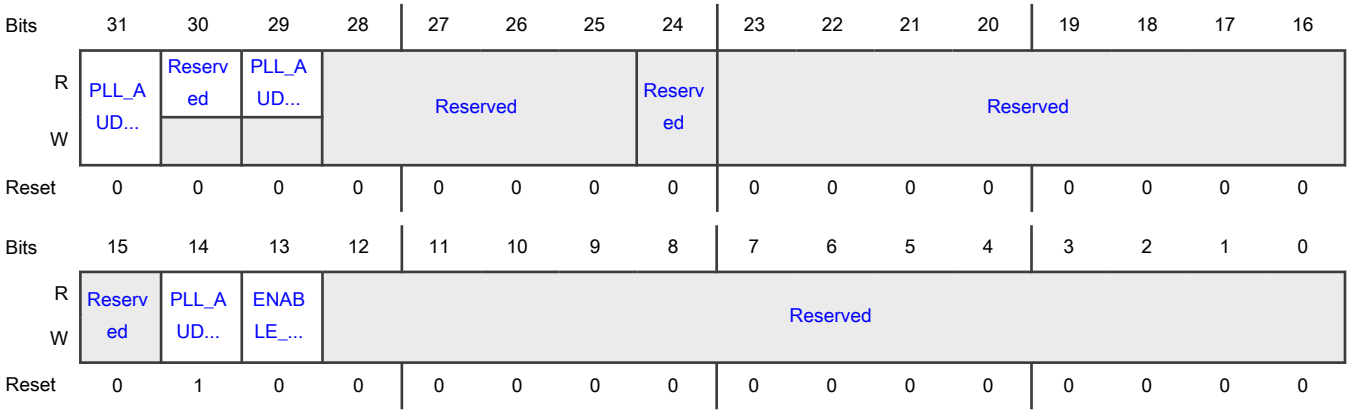
Register	Offset
PLL_AUDIO_CTRL	4200h

Function

PLL_AUDIO_control_register

The control register provides control for the AUDIO PLL.

Diagram



Fields

Field	Function
31 PLL_AUDIO_CONTROL_MODE	pll_audio_control_mode Enable mode 0b - Software Mode (Default) 1b - GPC Mode
30 —	Reserved
29 PLL_AUDIO_STABLE	PLL_AUDIO_STABLE Normally in Software mode, after the power up and enable sequence for pll, arm pll will not be in stable state immediately. This bit is used by software to monitor the locking status for pll. GPC mode do not need to take care of this bit. Hardware will handle it automatically.
28-25 —	Reserved
24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23-15 —	Reserved
14 PLL_AUDIO_GATE	PLL_AUDIO_GATE There is a gate in top level to gate the pll output for power saving purpose,this field it to control this gate. 0b - No gate 1b - Gate the output
13 ENABLE_CLK	ENABLE_CLK Enable the clock output. This field is used to align the software mode and GPC mode. The usage of this bit is to describe the last PLL mode(enable or disable) before entering GPC mode. 0b - Disable 1b - Enable
12-0 —	Reserved

24.4.1.20 24MHz RCOSC Control Register (OSC_RC24M_CTRL)

Offset

Register	Offset
OSC_RC24M_CTRL	4310h

Function

This register is used to control the 24MHz RC oscillator.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RC_24M_...	Reserved	Reserved				SOURCE_...	Reserved	Reserved							
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														TEN	Reserved
Reset	0	0	0	0	0	0	0	1	1	1	1	1	0	0	1	0

Fields

Field	Function
31 RC_24M_CONTROL_MODE	RCOSC Control Mode This field selects between GPC mode and Software mode 0b - Software mode (default) 1b - GPC mode
30 —	Reserved
29-26 —	Reserved
25 SOURCE_SEL_24M	source_sel_24M This is the source select for 24M. An option is provided to switch the osc_rc24m to the source of osc24M. 24m source sel. 0b - OSC_RC24M 1b - OSC24M
24 —	Reserved
23-2 —	Reserved
1 TEN	RC24M Enable This field powers up or powers down the RCOSC 0b - Power down 1b - Power up
0 —	Reserved

24.4.1.21 24MHz OSC Control Register (OSC_24M_CTRL)

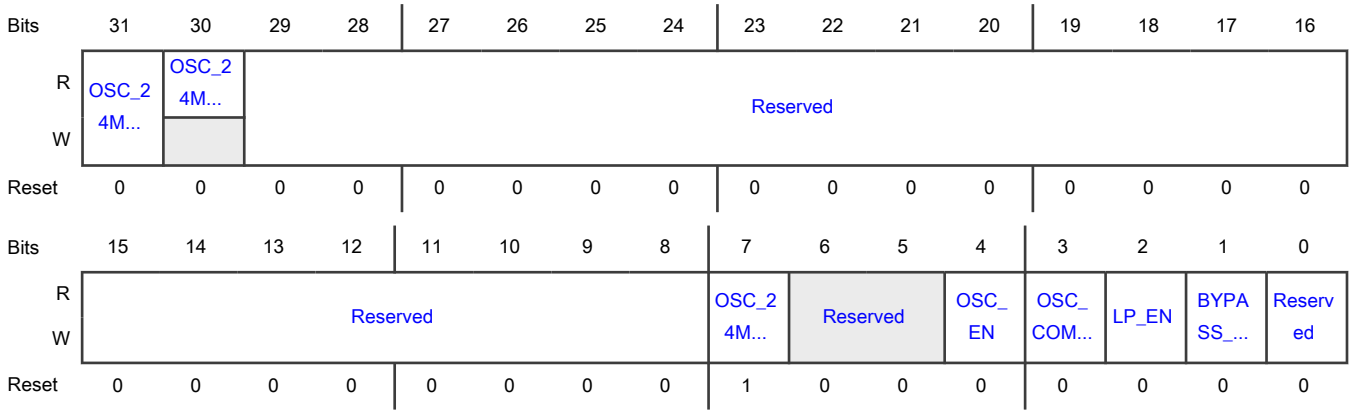
Offset

Register	Offset
OSC_24M_CTRL	4320h

Function

This register is used to control the 24MHz Oscillator.

Diagram



Fields

Field	Function
31 OSC_24M_CONTROL_MODE	24MHz OSC Control Mode This field selects between GPC mode and Software mode 0b - Software mode (default) 1b - GPC mode
30 OSC_24M_STABLE	24MHz OSC Stable This field indicates whether 24MHz OSC is stable (24MHz OSC will not be stable the time it is enabled) 0b - Not Stable 1b - Stable
29-8 —	Reserved
7 OSC_24M_GATE	24MHz OSC Gate Control This field gates the 24MHz OSC output for saving power 0b - Not Gated 1b - Gated
6-5 —	Reserved
4 OSC_EN	24MHz OSC Enable This field enables the 24MHz XTALOSC 0b - Disable 1b - Enable
3	24MHz OSC Comparator Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
OSC_COMP_MODE	This field selects the ac-coupling comparator operation-mode 0b - Single-ended mode (default) 1b - Differential mode (test mode)
2 LP_EN	24MHz OSC Low-Power Mode Enable 0b - High Gain mode (HP) 1b - Low-power mode (LP)
1 BYPASS_EN	24MHz OSC Bypass Enable Enable signal for external bypass clock (mostly connect to XTAL directly in SOC level). Users can use this field to enable an external source of XTAL. 0b - Disable 1b - Enable
0 —	Reserved

24.4.1.22 400MHz RCOSC Control0 Register (OSC_400M_CTRL0)

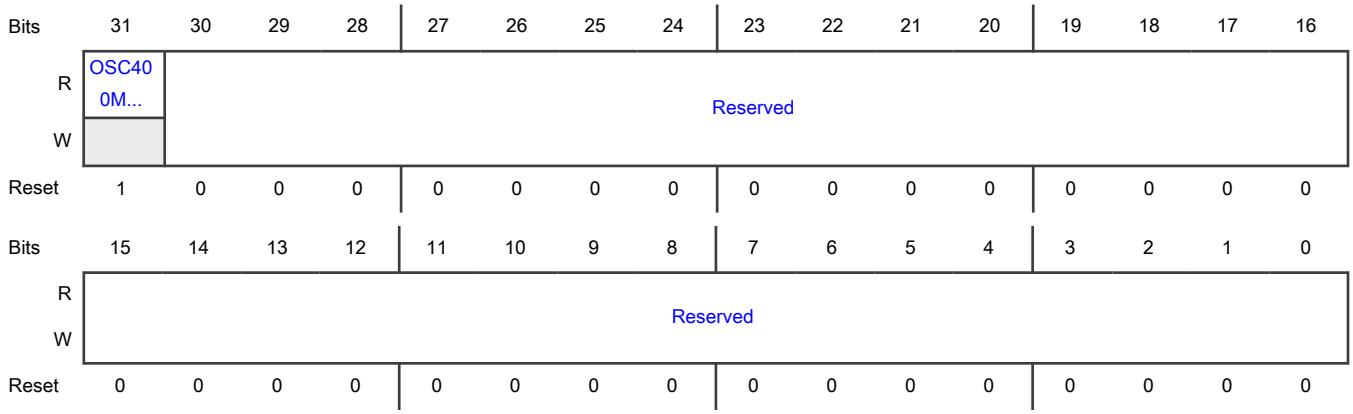
Offset

Register	Offset
OSC_400M_CTRL0	4340h

Function

Contains control bits for the 400MHz RCOSC.

Diagram



Fields

Field	Function
31 OSC400M_AI_BUSY	400MHz OSC AI BUSY This field is associated with AI busy monitor function in AI bus.
30-0 —	Reserved

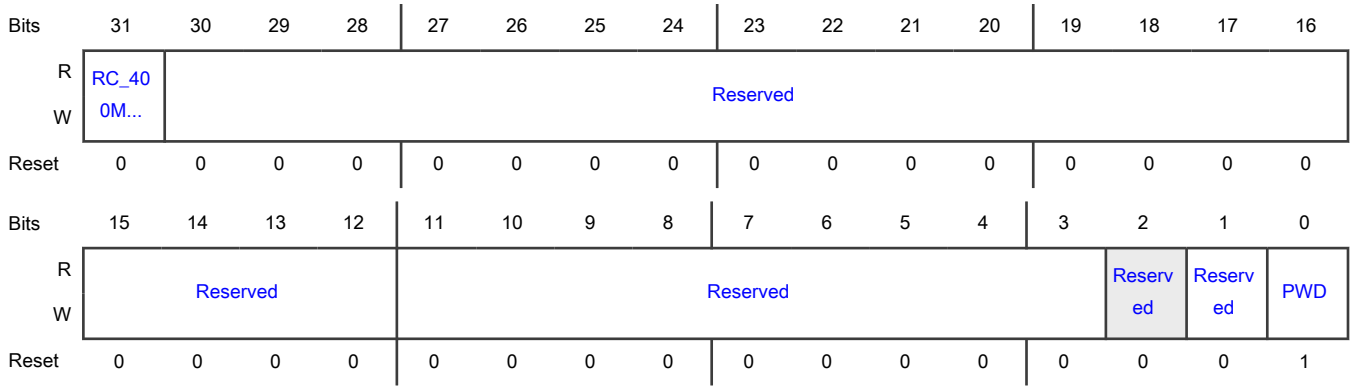
24.4.1.23 400MHz RCOSC Control1 Register (OSC_400M_CTRL1)

Offset

Register	Offset
OSC_400M_CTRL1	4350h

Function
This register provides the controls for clock gate, power down, and RC400M mode selects.

Diagram



Fields

Field	Function
31 RC_400M_CONTROL_MODE	400MHz RCOSC Control mode This field selects between GPC mode and Software mode 0b - Software mode (default) 1b - GPC mode
30-12 —	Reserved Always set to zero (0)

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-3 —	Reserved
2 —	Reserved
1 —	Reserved
0 PWD	Power down control for 400MHz RCOSC 400MHz RCOSC can be powered down directly with this bit. There is no need to follow AI bus sequence 0b - No Power down 1b - Power down

24.4.1.24 TEMPSNS_OTP_TRIM_VALUE_REGISTER (TEMPSNS_OTP_TRIM_VALUE)

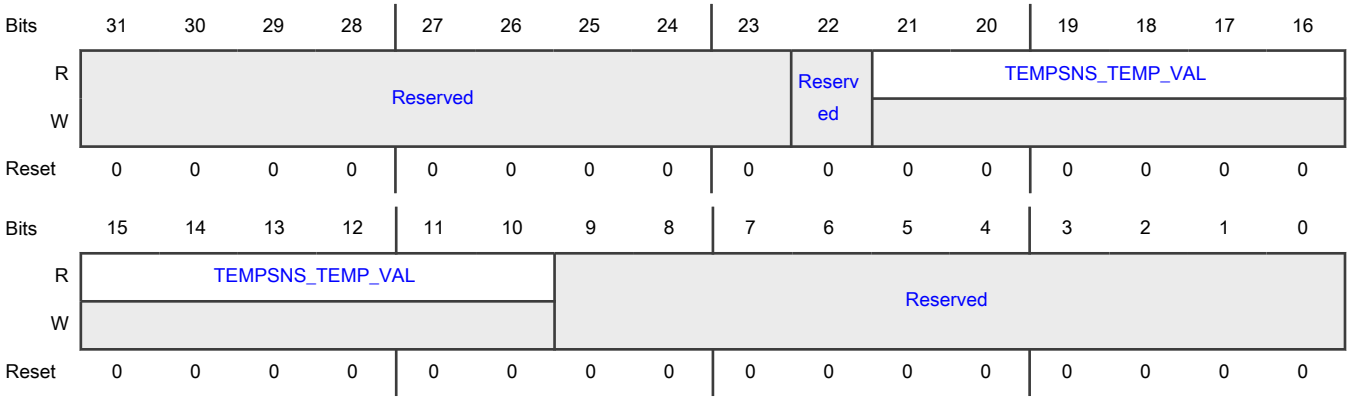
Offset

Register	Offset
TEMPSNS_OTP_TRIM_VALUE	4530h

Function

This is the control for TRIM bus control. For ANATOP, this is the source from software register inside ANATOP.
TEMPSNS_OTP_TRIM_VALUE

Diagram



Fields

Field	Function
31-23 —	Reserved
22 —	Reserved
21-10 TEMPSNS_TEMP_VAL	Temperature Value at 25C The value for this bit field comes from the fuses. See the TMPSNS chapter's Temperature Sensing section for the equation to calculate temperature value.
9-0 —	Reserved

24.4.1.25 PMU_BIAS_CTRL_REGISTER (PMU_BIAS_CTRL)

Offset

Register	Offset
PMU_BIAS_CTRL	4600h

Function

PMU_Well_BIAS_Control_Register

This register defines functions for wb bias.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WB_NW_LVL_1P8				WB_PW_LVL_1P8				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FBB_M7_...	WB_VDD_...	Reserved	WB_CFG_1P8												
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 WB_NW_LVL_1P8	wb_nw_lvl_1p8 If WB_EN=1，these bits set the N-well voltage level. Select the state level by setting wb_nw_lvl_1p8[3:0] before enabling the wbias. NW Output voltage selection. Valid range is 0000-1000. • 0000 NW Output voltage 0.5V • 0001 NW Output voltage 0.6V • 0010 NW Output voltage 0.7V • 0011 NW Output voltage 0.8V • 0100 NW Output voltage 0.9V • 0101 NW Output voltage 1.0V • 0110 NW Output voltage 1.1V • 0111 NW Output voltage 1.2V • 1000 NW Output voltage 1.3V
27-24 WB_PW_LVL_1P8	wb_pw_lvl_1p8 If WB_EN=1，these bits set the P-well voltage level. Select the state level by seting wb_pw_lvl_1p8[3:0] before enabling the wbias. PW Output voltage selection. Valid range is 0000-1000. • 0000 PW Output voltage -0.5V • 0001 PW Output voltage -0.6V • 0010 PW Output voltage -0.7V • 0011 PW Output voltage -0.8V • 0100 PW Output voltage -0.9V • 0101 PW Output voltage -1.0V • 0110 PW Output voltage -1.1V • 0111 PW Output voltage -1.2V • 1000 PW Output voltage -1.3V
23-16 —	Reserved
15 FBB_M7_STBY_EN	standby enable bit of fbb m7 Enables gpc standby mode.The default value is 1. 0b - FBB_M7 will be still on when gpc give standby request. After the mode is switched to gpc mode, keep this bit as it is. 1b - FBB_M7 will standby when gpc give standby request.
14	wb_vdd_sel_1p8 Well bias VDD selector 1P8. There are two power sources (AON_DIG_LDO or DCDC).

Table continues on the next page...

Table continued from the previous page...

Field	Function
WB_VDD_SEL_1P8	0b - VDD_LV1. VDD_LV1 supplies the power stage and NWELL sampler (DCDC) 1b - VDD_LV2. VDD_LV2 supplies the power stage and NWELL sampler (AON_DIG_LDO)
13 —	Reserved
12-0 WB_CFG_1P8	wb_cfg_1p8 Well Bias Configuration 1P8. For bit 0 : • 0 PWELL and NWELL is turned on when wb_en_1p8 is set; • 1 PWELL regulator is turned on only when wb_en_1p8 is set. NWELL is kept disabled. For bit 1 : Body bias • 0 NWELL is configured to supply and RVT CORE - RBB. • 1 NWELL is configured to supply and LVT CORE - FBB. For bit 2-4 : Select size of bias area • 000 Imax = 180uA; Areamax-RVT = 6.00mm2 at 125C • 001 Imax = 150uA; Areamax-RVT = 5.00mm2 at 125C • 010 Imax = 120uA; Areamax-RVT = 4.00mm2 at 125C • 011 Imax = 90uA; Areamax-RVT = 3.00mm2 at 125C • 100 Imax = 60uA; Areamax-RVT = 2.00mm2 at 125C • 101 Imax = 45uA; Areamax-RVT = 0.15mm2 at 125C • 110 Imax = 30uA; Areamax-RVT = 1.0mm2 at 125C • 111 Imax = 15uA; Areamax-RVT = 0.50mm2 at 125C For bit 5 : Adaptive function • 1 Adaptive frequency disabled. Frequency determined by wb_cfg_1p8[8:6]. • 0 Frequency change after each half cycle Minimum frequency determined by wb_cfg_1p8[8:6] For bit 6-8 : Oscillator bits • 000 typical frequency = osc_freq/128 • 001 typical frequency = osc_freq/64 • 010 typical frequency = osc_freq/32 • 011 typical frequency =osc_freq/16 • 100 typical frequency = osc_freq/8 • 110 typical frequency = osc_freq/2 • 111 typical frequency = oscillator Frequency (osc_freq)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For bit 9 : TRIM BUS. It is used to set an internal adaptive configuration. It configures the adaptive clock source. The options are: 1) the oscillator clock 2) clock synchronous with the Charge Pump clock. These 2 signals have the same frequency, but delayed one to other. It determines if the adaptive control signal changes the frequency in the same or in the next oscillator cycle. • 0 The option is the oscillator clock • 1 The option is clock synchronous with the Charge Pump clock For bit 10 - 11 : TRIM BUS • 00 no frequency reduction • 01 30% frequency reduction due to cap. increment. • 10 40% frequency reduction due to cap. increment. • 11 50% frequency reduction due to cap. increment. For bit 12 : • 1 Pull down option is enabled. • 0 Pull down option is disabled.

24.4.1.26 PMU_BIAS_CTRL2_REGISTER (PMU_BIAS_CTRL2)

Offset

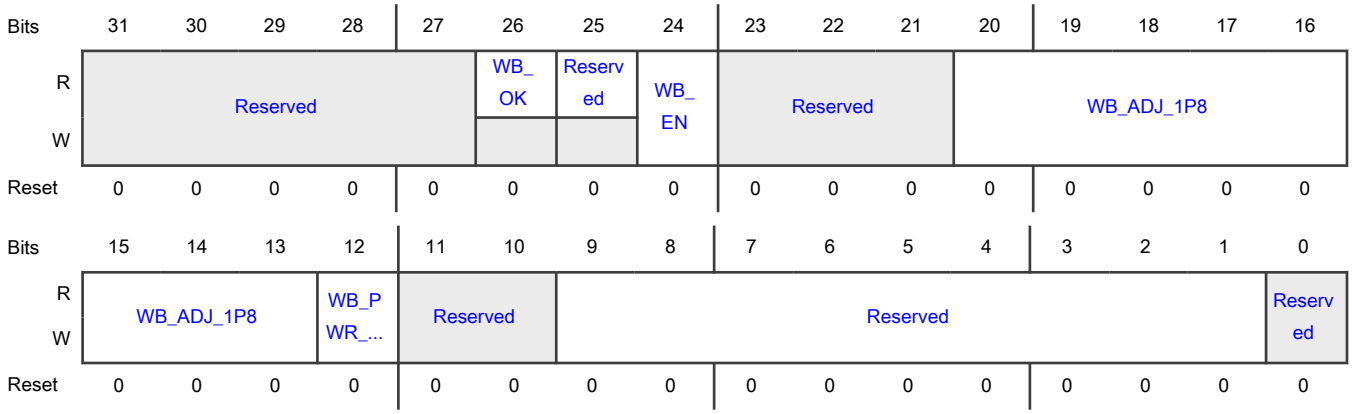
Register	Offset
PMU_BIAS_CTRL2	4610h

Function

Well Bias IP control register

This register defines the test mode function for wb bias.

Diagram



Fields

Field	Function
31-27 —	Reserved
26 WB_OK	Digital Output pin. Turn on/off acknowledge bit from regulator 0b - Regulator is unstable. 1b - Regulator is stable.
25 —	Reserved
24 WB_EN	wb_en Well bias enable 0b - disable 1b - Enable
23-21 —	Reserved
20-13 WB_ADJ_1P8	wb_adj_1p8 Well Bias Adjustment 1P8. The bit values detailed below is a representation of bits 0 - 3 : TRIM BUS, and bits 4 - 7. 0000_0000b - Cref= 0fF Cspl= 0fF DeltaC= 0fF 0000_0001b - Cref= 0fF Cspl= 30fF DeltaC= -30fF 0000_0010b - Cref= 0fF Cspl= 43fF DeltaC= -43fF 0000_0011b - Cref= 0fF Cspl= 62fF DeltaC=-62fF 0000_0100b - Cref= 0fF Cspl=105fF DeltaC=-105fF 0000_0101b - Cref= 30fF Cspl= 0fF DeltaC= 30fF 0000_0110b - Cref= 30fF Cspl= 43fF DeltaC= -12fF 0000_0111b - Cref= 30fF Cspl=105fF DeltaC= -75fF 0000_1000b - Cref= 43fF Cspl= 0fF DeltaC= 43fF 0000_1001b - Cref= 43fF Cspl= 30fF DeltaC= 13fF 0000_1010b - Cref= 43fF Cspl= 62fF DeltaC= -19fF 0000_1011b - Cref= 62fF Cspl= 0fF DeltaC= 62fF 0000_1100b - Cref= 62fF Cspl= 43fF DeltaC= 19fF 0000_1101b - Cref=105fF Cspl= 0fF DeltaC= 105fF 0000_1110b - Cref=105fF Cspl=30fF DeltaC= 75fF

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0000_1111b - Cref=0fF Cspl=0fF DeltaC= 0fF
12 WB_PWR_SW_EN_1P8	MODSEL_wb_tst_md_1p8 Power switch enable. This is a PWR_SW_EN, that is a module selection. Setting these bits will connect the NWELL/PWELL to back-biasing power supplies (0 - not connected, 1 - connected). Each bit represents one configuration. represents FBB M7 0b - WELL connected to no back-biasing power supplies 1b - WELL connected to back biasing generators.
11-10 —	Reserved
9-1 —	Reserved
0 —	Reserved

24.4.1.27 PMU_LDO_PLL_REGISTER (PMU_LDO_PLL)

Offset

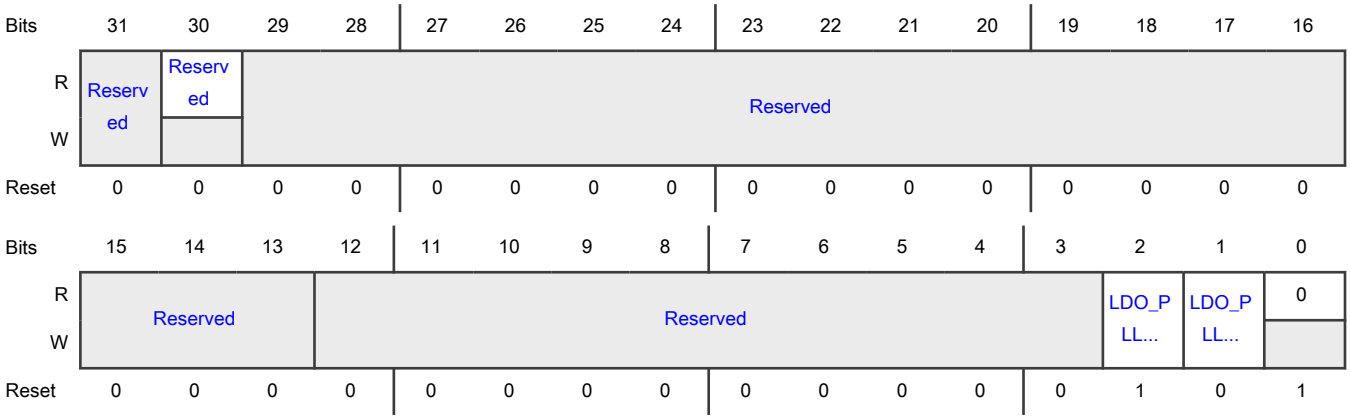
Register	Offset
PMU_LDO_PLL	4640h

Function

PMU_LDO_PLL_Regulator_Control_Register

This register defines the control and status bits for PLL regulator. This regulator is designed to power the digital portions of the analog cells

Diagram



Fields

Field	Function
31 —	Reserved
30 —	Reserved
29-16 —	Reserved
15-13 —	Reserved
12-3 —	Reserved
2 LDO_PLL_STBY_EN	standby enable bit of ldopll Enables gpc standby mode. The default value is 1. 0b - phy_ldo will be still on when gpc gives standby request. After the mode is switched to gpc mode, keep this bit as it is. 1b - phy_ldo will standby when gpc give standby request
1 LDO_PLL_CONTROL_MODE	LDO_PLL_CONTROL_MODE LDO_PLL has two mode to for enable. One is software mode, the other is GPC mode. LDO_PLL_CONTROL_MODE select 0b - SW Control. Software control mode 1b - HW Control. Hardware / GPC control mode
0 —	Reserved

24.4.1.28 PMU_POWER_DETECT_CTRL_REGISTER (PMU_POWER_DETECT_CTRL)

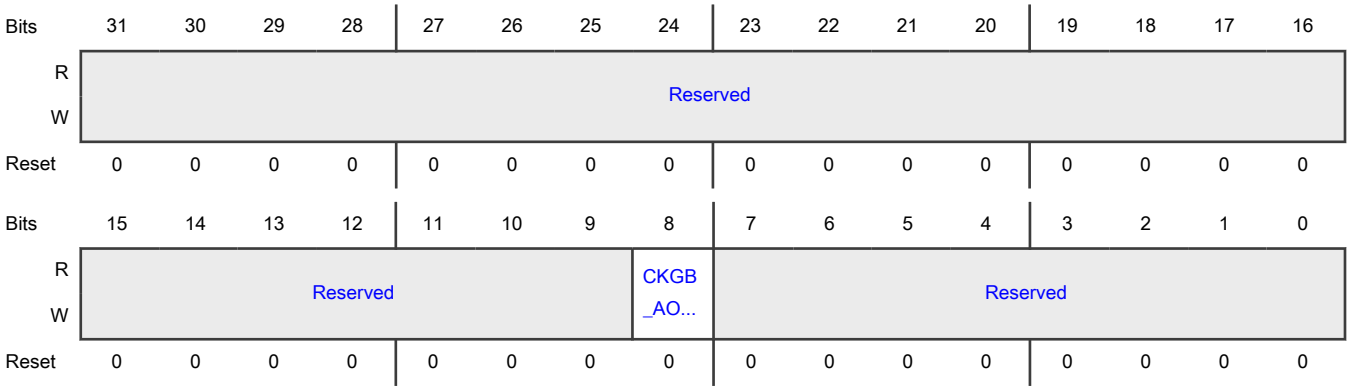
Offset

Register	Offset
PMU_POWER_DETECT_CTRL	4700h

Function

This register will be deleted in next release except ckgb bit

Diagram



Fields

Field	Function
31-16 —	Reserved
15-9 —	Reserved
8 CKGB_AON1P0	ckgb_aon1p0 Controls PHY LDO isolation. This bit is used during PHY LDO ON/OFF sequence. This bit is for software usage. In GPC mode, it will automatically be handled by hardware. 0b - To disable lpsr_1p0, ckgb need to be disabled first. 1b - After lpsr_1p0 start up, wait for 1ms and then set ckgb bit.
7-0 —	Reserved

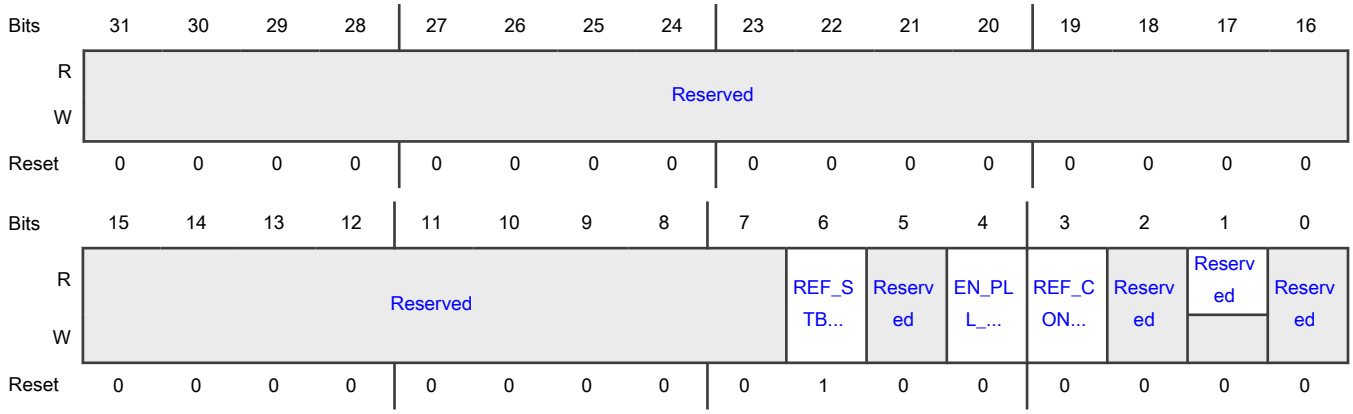
24.4.1.29 PMU_REF_CTRL_REGISTER (PMU_REF_CTRL)

Offset

Register	Offset
PMU_REF_CTRL	4710h

Function
Anadig_Reference_Analog_Control_and_Status_Control_Register

Diagram



Fields

Field	Function
31-7 —	Reserved
6 REF_STBY_EN	standby enable bit of reftop default value is 1. The usage of this bit is for gpc standby mode. 1: bandgap will standby when gpc give standby request; 0:bandgap will still on when gpc give standby request. After the mode switched to gpc mode, keep this bit as it is should be fine.
5 —	Reserved Always set to zero (0).
4 EN_PLL_VOL_REF_BUFFER	en_pll_vol_ref_buffer This control bit enables or disables the reference voltage for the PLLs.
3 REF_CONTROL_MODE	REF_CONTROL_MODE SW/HW control mode for reftop 0b - SW Control. Software control mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - HW Control. Hardware / GPC control mode
2 —	Reserved
1 —	Reserved
0 —	Reserved

24.4.1.30 PMU_LDO_AON_ANA_REGISTER (PMU_LDO_AON_ANA)

Offset

Register	Offset
PMU_LDO_AON_ANA	4740h

Function

PMU_LDO_AON_ANA_Regulator_Control_Register

This register defines the control and status bits for LDO_AON_ANA regulator. This regulator is designed to power the digital portions of the analog cells

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved											PULL_DO...	TRACK_M...	Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	Reserved		Reserved				ALWAYS_...	Reserved	STANDBY...	Reserved	Reserved	PULL_DO...	REG_DIS...	Reserved	REG_LP_...
Reset	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0

Fields

Field	Function
31-21	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
20 PULL_DOWN_20UA_EN	pull_down_20ua_en Enable or disable 20uA loading to prevent the overshoot. 0b - Disable 20uA loading 1b - Enable 20uA loading
19 TRACK_MODE_EN	Track Mode Enable This bit lets the power switch enter a preparation stage that allows power supply switching between LDO and DCDC. This bit should be set before the power supply switch and cleared after power switch is complete. This bit is used during software mode. 0b - Normal use 1b - Switch preparation
18-16 —	Reserved
15 —	Reserved
14-13 —	Reserved
12-9 —	Reserved
8 ALWAYS_4MA_PULLDOWN_EN	always_4ma_pulldown_en High: enable 4mA loading to prevent the big voltage drop when a sharp loading coming.It's recommend to set this bit to high under the reset, and set it to low under the normal use case.
7 —	Reserved
6 STANDBY_EN	standby_en This is the standby mode for LDO_AON_ANA 0b - Standby mode disable 1b - Standby mode enable
5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 —	Reserved
3 PULL_DOWN_2MA_EN	pull_down_2ma_en Enable or disable 2 mA loading to prevent the overshoot. 0b - Disable 1b - Enable
2 REG_DISABLE	reg_disable Enable or disable the output of "vreg_1p8" 0b - Enable 1b - Disable
1 —	Reserved
0 REG_LP_EN	reg_lp_en Enable or disable the low power mode of the ldo. 0b - Enable 1b - Disable

24.4.1.31 PMU_LDO_AON_DIG_REGISTER (PMU_LDO_AON_DIG)

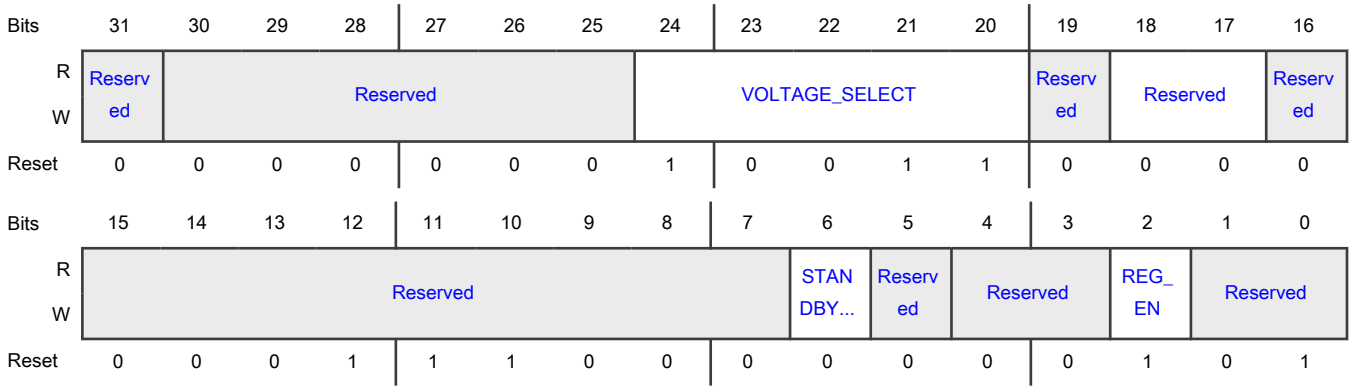
Offset

Register	Offset
PMU_LDO_AON_DIG	4760h

Function

PMU_LDO_AON_DIG_Regulator_Control_Register

Diagram



Fields

Field	Function
31 —	Reserved
30-25 —	Reserved
24-20 VOLTAGE_SELECT	VOLTAGE_SELECT VOLTAGE SELECT for AON_DIG For aon_dig voltage switch to different voltage In SW mode, software is responsible to take care of the stepping time. In HW mode, hardware is responsible to take care of the stepping time. During the mode transition, in case the aon_dig voltage is out of table, software is responsible to take care of the stepping time (i.e. mode transist), software calculates the steps difference multiply the stepping requirement, after the voltage is stable. GPC mode transition is allowed. The values below are the stable voltage with the range in parentheses (e.g. Stable Voltage (Range)V). 0_0000b - Stable Voltage (range). 0.631 (+0.039)V 0_0001b - Stable Voltage (range). 0.65 (+0.041)V 0_0010b - Stable Voltage (range). 0.67 (+0.041)V 0_0011b - Stable Voltage (range). 0.689 (+0.043)V 0_0100b - Stable Voltage (range). 0.709 (+0.044)V 0_0101b - Stable Voltage (range). 0.728 (+0.045)V 0_0110b - Stable Voltage (range). 0.748 (+0.046)V 0_0111b - Stable Voltage (range). 0.767 (+0.047)V 0_1000b - Stable Voltage (range). 0.786 (+0.049)V 0_1001b - Stable Voltage (range). 0.806 (+0.05)V

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0_1010b - Stable Voltage (range). 0.825 (+0.051)V 0_1011b - Stable Voltage (range). 0.845 (+0.052)V 0_1100b - Stable Voltage (range). 0.864 (+0.054)V 0_1101b - Stable Voltage (range). 0.883 (+0.055)V 0_1110b - Stable Voltage (range). 0.903 (+0.056)V 0_1111b - Stable Voltage (range). 0.922 (+0.057)V 1_0000b - Stable Voltage (range). 0.942 (+0.058)V 1_0001b - Stable Voltage (range). 0.961 (+0.06)V 1_0010b - Stable Voltage (range). 0.981 (+0.06)V 1_0011b - Stable Voltage (range). 1 (+0.062)V 1_0100b - Stable Voltage (range). 1.019 (+0.063)V 1_0101b - Stable Voltage (range). 1.039 (+0.064)V 1_0110b - Stable Voltage (range). 1.058 (+0.066)V 1_0111b - Stable Voltage (range). 1.078 (+0.066)V 1_1000b - Stable Voltage (range). 1.097 (+0.068)V 1_1001b - Stable Voltage (range). 1.117 (+0.069)V 1_1010b - Stable Voltage (range). 1.136 (+0.07)V 1_1011b - Stable Voltage (range). 1.155 (+0.072)V 1_1100b - Stable Voltage (range). 1.175 (+0.072)V 1_1101b - Stable Voltage (range). 1.194 (+0.074)V 1_1110b - Stable Voltage (range). 1.214 (+0.075)V 1_1111b - Stable Voltage (range). 1.233 (+0.076)V
19 —	Reserved
18-17 —	Reserved
16-7 —	Reserved
6 STANDBY_EN	standby_en Enable or disable to enter Standby mode for aon_dig. 0b - Standby disable 1b - Standby enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 —	Reserved
4-3 —	Reserved
2 REG_EN	ENABLE_ILIMIT Control bit to enable the current-limit circuitry in the regulator. 0b - LDO_AON_DIG disable 1b - LDO_AON_DIG enable
1-0 —	Reserved

24.4.1.32 Chip Silicon Version Register (MISC_DIFPROG)

Offset

Register	Offset
MISC_DIFPROG	4800h

Function

Chip Silicon Version Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CHIPID															
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHIPID															
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CHIPID	Chip ID

24.4.2 IPS Domain register descriptions

24.4.2.1 IPS_DOMAIN memory map

ANADIG_SLOTS base address: 4448_0000h

Offset	Register	Width (In bits)	Access	Reset value
4C00h - 4C88h	Slot Control Register (SLOT0_CTRL - SLOT34_CTRL)	32	RW	0000_0002h

24.4.2.2 Slot Control Register (SLOT0_CTRL - SLOT34_CTRL)

Offset

For n = 0 to 34:

Register	Offset
SLOTn_CTRL	4C00h + (n × 4h)

Function

Table 158. Slot Assignments

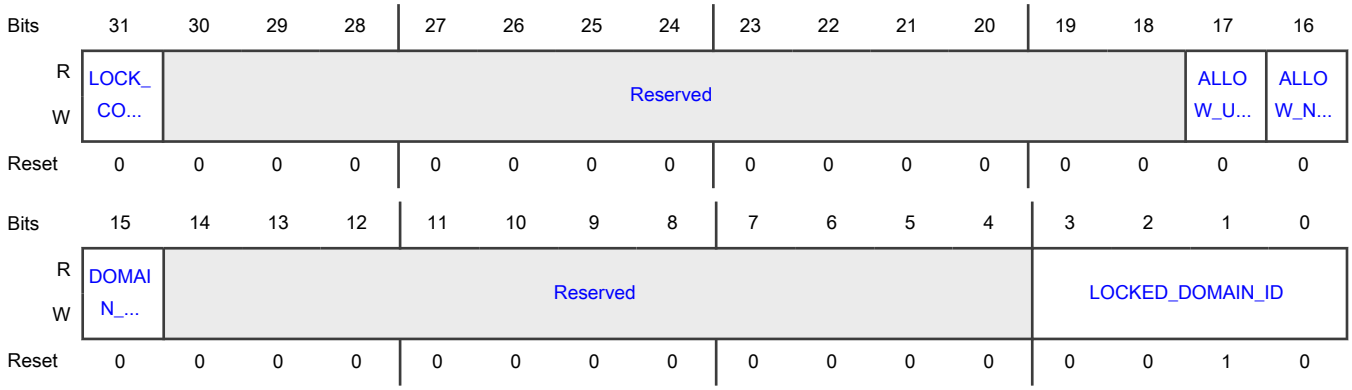
Slot Number	Assignment
0	PLL_ARM
1	SYS_PLL3 (480 PLL)
2	SYS_PLL2 (528 PLL)
3	SYS_PLL1 (1G PLL)
4	AI_1G
5	PLL_AUDIO
6	AI_AUDIO
7	ANATOP_CTRL
8	48M
9	24M

Table continues on the next page...

Table 158. Slot Assignments (continued)

Slot Number	Assignment
10	400M
11	AI_400M
12	OSC TRIM
13	TEMPSENSOR
14	ANAMUX
15	PWRDET
16	TEMPSENSOR_OTP
17	PWR_OTP
18	AI_TMPSNS
19	BIAS
20	FBB_M7_CFG
21	WB_OTP
22	PLL_LDO
23	AI_PLL_LDO
24	CKGB
25	BANDGAP
26	AON_ANA_OTP
27	BANDGAP_OTP
28	AON_ANA
29	AON_DIG
30	Reserved
31	AI_BANDGAP
32	MISC
33	LVDS
34	SPARE

Diagram



Fields

Field	Function
31 LOCK_CONTROL	Lock control of this slot 0b - Do not lock the control register of this slot 1b - Lock the control register of this slot
30-18 —	Reserved
17 ALLOW_USER	Allow user write access to this domain control register or domain register 0b - Do not allow user write access 1b - Allow user write access
16 ALLOW_NONSECURE	Allow non-secure write access to this domain control register or domain register 0b - Do not allow non-secure write access 1b - Allow non-secure write access
15 DOMAIN_LOCK	Lock domain ID of this slot 0b - Do not lock the domain ID 1b - Lock the domain ID
14-4 —	Reserved
3-0 LOCKED_DOMAIN_ID	Domain ID of the slot to be locked Domain id3~0 maps to bit3~0. It indicates whether related domain can write to domain register or not. The slot can be accessed only when ips_domain_id is same with the value of these bits

24.5 About this module

24.5.1 Block diagram

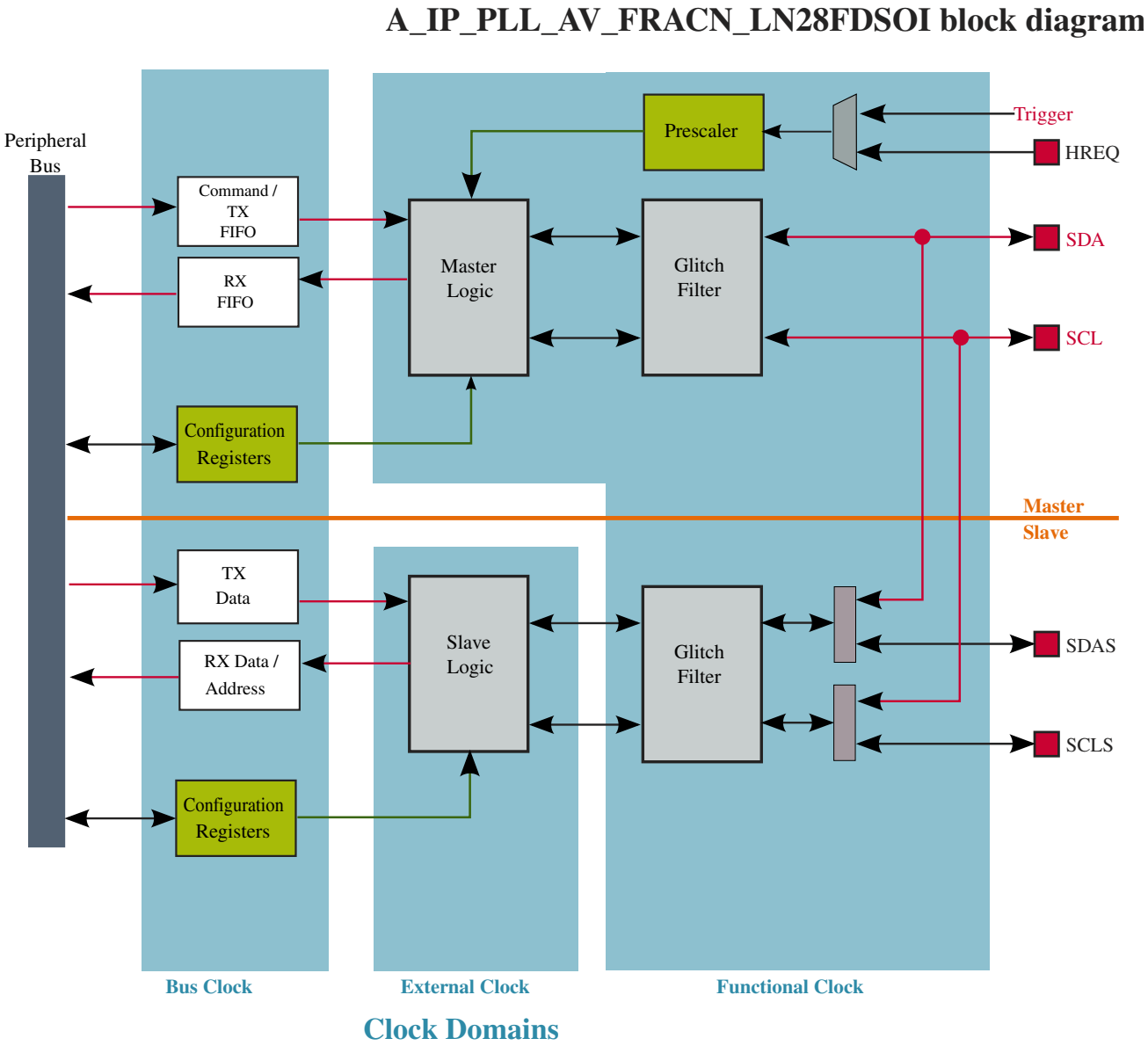


Figure 93. Fractional PLL block diagram

24.6 External signals

This table describes the module's external signals.

Signal	I/O	Description
ATRIG0	I	CFIFO 0 Auxiliary Trigger–Used only for CFIFO 0 when it's in one of the single-scan or continuous-scan edge hardware trigger modes and streaming is enabled (EQADC_CFCR0_1[STRME0] is 1b). Receives auxiliary triggers that assert the EN_TRIG0 and ADVANCE0 flags to control the streaming operation of CFIFO 0.
EWM_out_B	O	Reset Out–Gates an external circuit (application specific) that controls critical safety functions. Remains deasserted when the EWM is being regularly serviced by the CPU within the programmable service window, indicating that the application code is executed as expected.
INA_ADC1_0	I	Integration-Assigned Single-Ended ADC 1 Analog Input 0–Supplies a continuously variable voltage directly to the associated ADC for single-ended conversions to discrete positive digital values
TRIG1[7:0]	I/O	CFIFO 1 Standard Trigger–Receives triggers that initiate transfers from the associated CFIFO to the ADC command buffers. The EQADC filters this signal to prevent interpreting short-lived, noise-related state changes as valid state changes. You can change the digital filter length for all standard triggers by changing the value of EQADC_ETDFR[DFL].
VDDA	I/O	Analog Positive Power Supply–Supplies power to the integrated ADCs.

24.7 Memory Map and register definition

This section includes the Fractional PLL module memory map and detailed descriptions of all registers.

24.7.1 Fractional PLL register descriptions

24.7.1.1 FRACTIONAL_PLL memory map

AUDIO_PLL base address: 4448_4280h

ETHERNET_PLL base address: 4448_4180h

Offset	Register	Width (In bits)	Access	Reset value
0h	Fractional PLL Control Register (CTRL0)	32	RW	0000_0000h
4h	Fractional PLL Control Register (CTRL0_SET)	32	RW	0000_0000h
8h	Fractional PLL Control Register (CTRL0_CLR)	32	RW	0000_0000h
Ch	Fractional PLL Control Register (CTRL0_TOG)	32	RW	0000_0000h
10h	Fractional PLL Spread Spectrum Control Register (SPREAD_SPECTRUM)	32	RW	0000_0000h
14h	Fractional PLL Spread Spectrum Control Register (SPREAD_SPECTRUM_SET)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18h	Fractional PLL Spread Spectrum Control Register (SPREAD_SPECTRUM_CLR)	32	RW	0000_0000h
1Ch	Fractional PLL Spread Spectrum Control Register (SPREAD_SPECTRUM_TOG)	32	RW	0000_0000h
20h	Fractional PLL Numerator Control Register (NUMERATOR)	32	RW	0000_0000h
24h	Fractional PLL Numerator Control Register (NUMERATOR_SET)	32	RW	0000_0000h
28h	Fractional PLL Numerator Control Register (NUMERATOR_CLR)	32	RW	0000_0000h
2Ch	Fractional PLL Numerator Control Register (NUMERATOR_TOG)	32	RW	0000_0000h
30h	Fractional PLL Denominator Control Register (DENOMINATOR)	32	RW	0000_0000h
34h	Fractional PLL Denominator Control Register (DENOMINATOR_SET)	32	RW	0000_0000h
38h	Fractional PLL Denominator Control Register (DENOMINATOR_CLR)	32	RW	0000_0000h
3Ch	Fractional PLL Denominator Control Register (DENOMINATOR_TOG)	32	RW	0000_0000h

24.7.1.2 Fractional PLL Control Register (CTRL0)

Offset

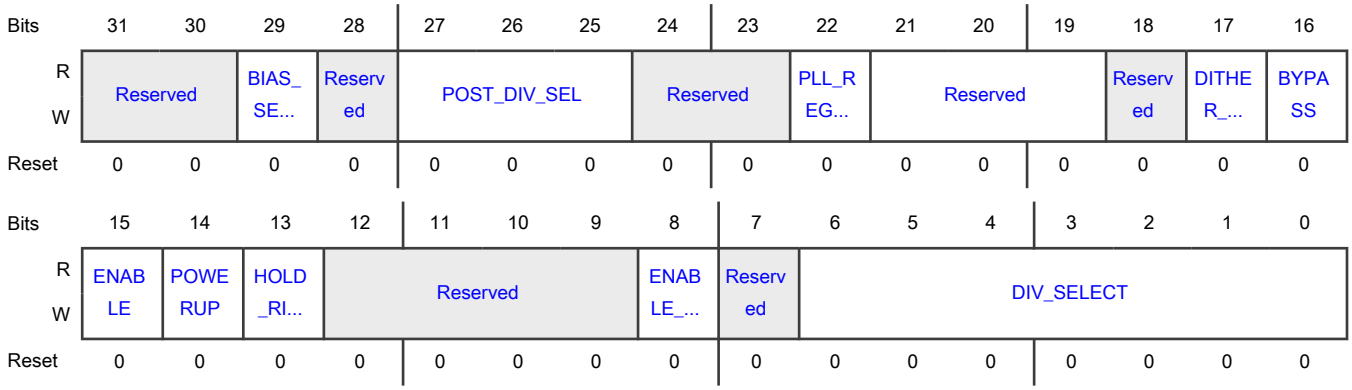
This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
CTRL0	0h	Fractional PLL Control Register
CTRL0_SET	4h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL0 is 1
CTRL0_CLR	8h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL0 is 0
CTRL0_TOG	Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in CTRL0

Function

This register contains fractional pll control registers.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 BIAS_SELECT	BIAS_SELECT Selects which input bias current is expected by the PLL. 0b - Used in SoCs with a bias current of 10uA 1b - Used in SoCs with a bias current of 2uA
28 —	Reserved
27-25 POST_DIV_SEL	Post Divide Select Fractional PLL Post Divider Select (glitchless clock mux) 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32
24-23 —	Reserved
22 PLL_REG_EN	PLL_REG_EN Enables the internal PLL regulator. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-19 —	Reserved
18 —	Reserved
17 DITHER_EN	DITHER_EN Enables dither in the fractional modulator calculation. 0b - Disable Dither 1b - Enable Dither
16 BYPASS	BYPASS Byapss the PLL. Output is the PLL reference clock source 0b - No Bypass 1b - Bypass the PLL
15 ENABLE	ENABLE 0b - Disable the clock output 1b - Enable the clock output
14 POWERUP	POWERUP 0b - Power down the PLL 1b - Power Up the PLL
13 HOLD_RING_OFF	PLL Start up initialization This field should be set to 1 every time during PLL lock, and cleared after a certain delay. 0b - Normal operation 1b - Initialize PLL start up
12-9 —	Reserved
8 ENABLE_ALT	ENABLE_ALT 0b - Disable the alternate clock output 1b - Enable the alternate clock output which is the output of the post_divider, and cannot be bypassed
7 —	Reserved
6-0	DIV_SELECT

Table continues on the next page...

Table continued from the previous page...

Field	Function
DIV_SELECT	This field controls the pll loop divider. Valid range for DIV_SELECT divider value: 27-54 decimal.

24.7.1.3 Fractional PLL Spread Spectrum Control Register (SPREAD_SPECTRUM)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
SPREAD_SPECTRUM	10h	Fractional PLL Spread Spectrum Control Register
SPREAD_SPECTRUM_SET	14h	Writing 1 to a bit in this register ensures that the corresponding bit in SPREAD_SPECTRUM is 1
SPREAD_SPECTRUM_CLR	18h	Writing 1 to a bit in this register ensures that the corresponding bit in SPREAD_SPECTRUM is 0
SPREAD_SPECTRUM_TOG	1Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in SPREAD_SPECTRUM

Function

This register contains fractional pll spread spectrum control registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STOP															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ENABLE	STEP														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 STOP	Stop Spread Spectrum Stop Register. Frequency change = step/MFD*24MHz.
15	Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENABLE	This bit enables the spread spectrum modulation. 0b - Disable 1b - Enable
14-0 STEP	Step Spread Spectrum Step The max frequency change = stop/MFD*24MHz.

24.7.1.4 Fractional PLL Numerator Control Register (NUMERATOR)

Offset

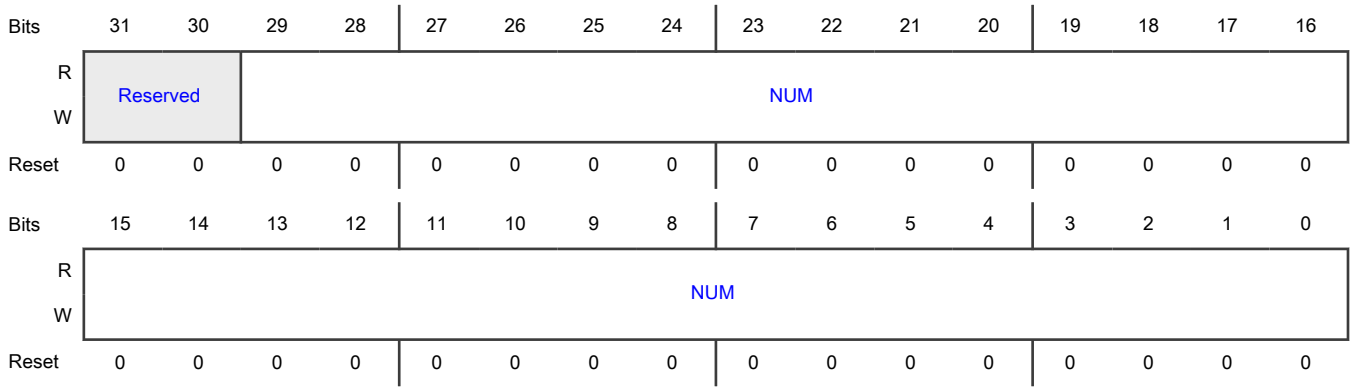
This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
NUMERATOR	20h	Fractional PLL Numerator Control Register
NUMERATOR_SET	24h	Writing 1 to a bit in this register ensures that the corresponding bit in NUMERATOR is 1
NUMERATOR_CLR	28h	Writing 1 to a bit in this register ensures that the corresponding bit in NUMERATOR is 0
NUMERATOR_TOG	2Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in NUMERATOR

Function

This register contains fractional pll numerator control registers.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-0 NUM	Numerator Numerator of PLL Fractional Loop Divider. 2's complement .

24.7.1.5 Fractional PLL Denominator Control Register (DENOMINATOR)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
DENOMINATOR	30h	Fractional PLL Denominator Control Register
DENOMINATOR_SET	34h	Writing 1 to a bit in this register ensures that the corresponding bit in DENOMINATOR is 1
DENOMINATOR_CLR	38h	Writing 1 to a bit in this register ensures that the corresponding bit in DENOMINATOR is 0
DENOMINATOR_TOG	3Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in DENOMINATOR

Function

This register contains fractional pll denominator control registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		DENOM													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DENOM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-0 DENOM	Denominator Denominator of Fractional Loop Divider. Unsigned number.

24.8 register descriptions

24.8.1 PHY_LDO memory map

PHY_LDO base address: 4448_4680h

Offset	Register	Width (In bits)	Access	Reset value
0h	Analog Control Register CTRL0 (CTRL0)	32	RW	0000_0000h
4h	Analog Control Register CTRL0 (CTRL0_SET)	32	RW	0000_0000h
8h	Analog Control Register CTRL0 (CTRL0_CLR)	32	RW	0000_0000h
Ch	Analog Control Register CTRL0 (CTRL0_TOG)	32	RW	0000_0000h
50h	Analog Status Register STAT0 (STAT0)	32	R	0000_0000h
54h	Analog Status Register STAT0 (STAT0_SET)	32	R	0000_0000h
58h	Analog Status Register STAT0 (STAT0_CLR)	32	R	0000_0000h
5Ch	Analog Status Register STAT0 (STAT0_TOG)	32	R	0000_0000h

24.8.2 Analog Control Register CTRL0 (CTRL0)

Offset

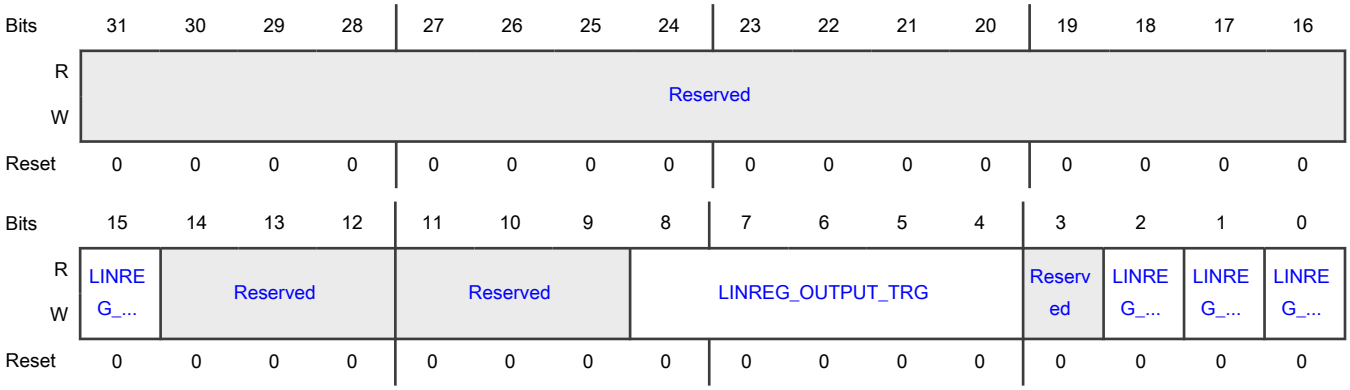
This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
CTRL0	0h	Analog Control Register CTRL0
CTRL0_SET	4h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL0 is 1
CTRL0_CLR	8h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL0 is 0
CTRL0_TOG	Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in CTRL0

Function

This register contains analog control bits.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 LINREG_PHY_ISO_B	Isolation control for attached PHY load This control bit is to be used by the system controller to isolate the attached PHY load when the LinReg is powered down. During a power-up event of the regulator it is expected that this control signal is set high at least 100us after the main regulator is enabled. During a power-down event of the regulator it is expected that this control signal is set low before the main regulator is disabled/power-down.
14-12 —	Reserved
11-9 —	Reserved
8-4 LINREG_OUTPUT_TRG	LinReg output voltage target setting LinReg output voltage target setting. The nominal voltage step per code is 25mV. Setting the output voltage beyond the technology reliability limit is not recommended.
3 —	Reserved
2 LINREG_ILIMIT_EN	LinReg current-limit enable LinReg current-limit enable. Setting this bit will enable the current-limiter in the regulator.
1	LinReg power-up load disable LinReg power-up load disable control bit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LINREG_PWR UPLOAD_DIS	0b - Internal pull-down enabled 1b - Internal pull-down disabled
0 LINREG_EN	LinrReg master enable LinReg master enable. Setting this bit will enable the regulator.

24.8.3 Analog Status Register STAT0 (STAT0)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
STAT0	50h	Analog Status Register STAT0
STAT0_SET	54h	Writing 1 to a bit in this register ensures that the corresponding bit in STAT0 is 1
STAT0_CLR	58h	Writing 1 to a bit in this register ensures that the corresponding bit in STAT0 is 0
STAT0_TOG	5Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in STAT0

Function

This register contains analog status bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												LINREG_STAT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 LINREG_STAT	LinReg Status Bits

24.9 register descriptions

24.9.1 OSC memory map

OSC_RC_400M base address: 4448_4380h

Offset	Register	Width (In bits)	Access	Reset value
0h	Analog Control Register CTRL0 (CTRL0)	32	RW	0000_0000h
4h	Analog Control Register CTRL0 (CTRL0_SET)	32	RW	0000_0000h
8h	Analog Control Register CTRL0 (CTRL0_CLR)	32	RW	0000_0000h
Ch	Analog Control Register CTRL0 (CTRL0_TOG)	32	RW	0000_0000h
10h	Analog Control Register CTRL1 (CTRL1)	32	RW	0000_0000h
14h	Analog Control Register CTRL1 (CTRL1_SET)	32	RW	0000_0000h
18h	Analog Control Register CTRL1 (CTRL1_CLR)	32	RW	0000_0000h
1Ch	Analog Control Register CTRL1 (CTRL1_TOG)	32	RW	0000_0000h
20h	Analog Control Register CTRL2 (CTRL2)	32	RW	0000_0000h
24h	Analog Control Register CTRL2 (CTRL2_SET)	32	RW	0000_0000h
28h	Analog Control Register CTRL2 (CTRL2_CLR)	32	RW	0000_0000h
2Ch	Analog Control Register CTRL2 (CTRL2_TOG)	32	RW	0000_0000h
30h	Analog Control Register CTRL3 (CTRL3)	32	RW	0000_0000h
34h	Analog Control Register CTRL3 (CTRL3_SET)	32	RW	0000_0000h
38h	Analog Control Register CTRL3 (CTRL3_CLR)	32	RW	0000_0000h
3Ch	Analog Control Register CTRL3 (CTRL3_TOG)	32	RW	0000_0000h
50h	Analog Status Register STAT0 (STAT0)	32	R	0000_0000h
60h	Analog Status Register STAT1 (STAT1)	32	R	0000_0000h
70h	Analog Status Register STAT2 (STAT2)	32	R	0000_0000h

24.9.2 Analog Control Register CTRL0 (CTRL0)

Offset

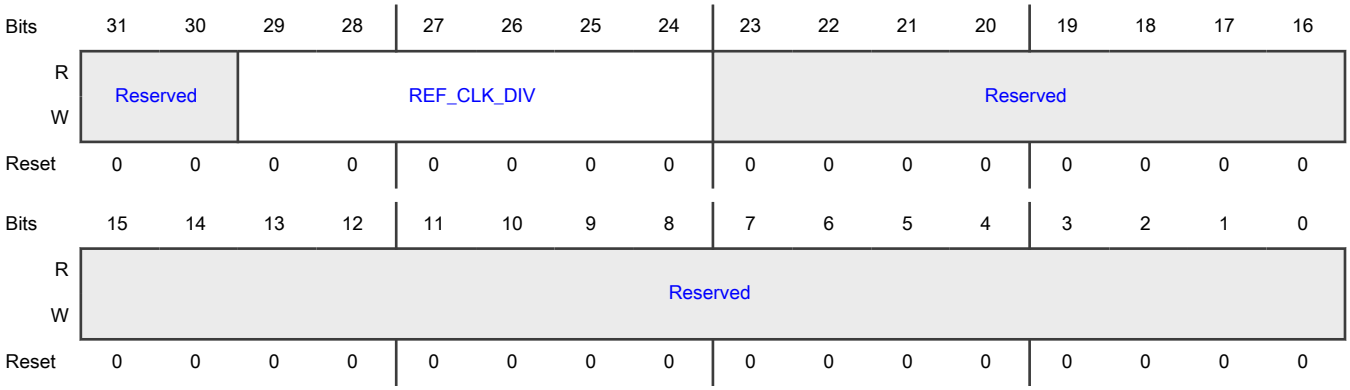
This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
CTRL0	0h	Analog Control Register CTRL0
CTRL0_SET	4h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL0 is 1
CTRL0_CLR	8h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL0 is 0
CTRL0_TOG	Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in CTRL0

Function

This register contains analog control bits.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 REF_CLK_DIV	Divide value for ref_clk to generate slow_clk. Divide value for ref_clk to generate slow_clk. 0: divide by 1 1-63: corresponding division by 1-63.
23-0 —	Reserved

24.9.3 Analog Control Register CTRL1 (CTRL1)

Offset

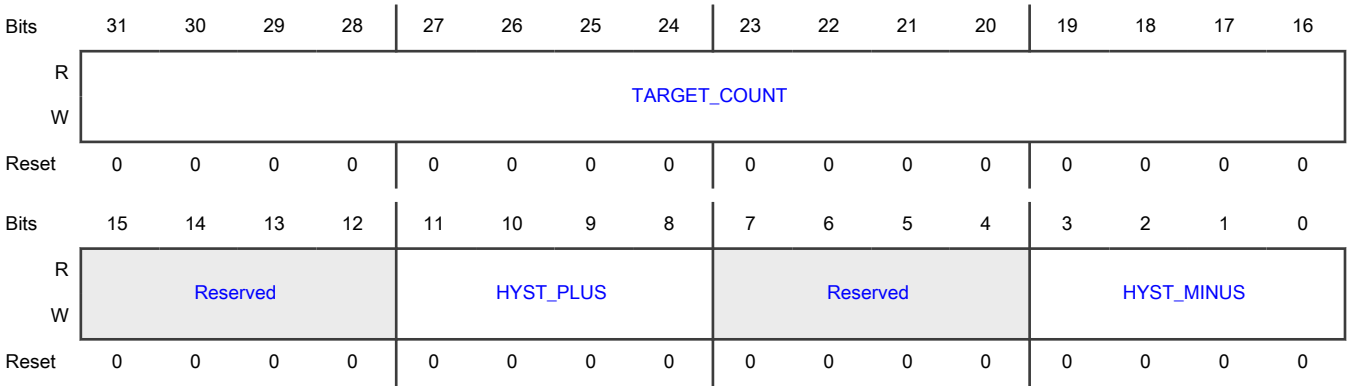
This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
CTRL1	10h	Analog Control Register CTRL1
CTRL1_SET	14h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL1 is 1
CTRL1_CLR	18h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL1 is 0
CTRL1_TOG	1Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in CTRL1

Function

This register contains analog control bits.

Diagram



Fields

Field	Function
31-16 TARGET_COUNT	Target count for the fast clock. Desired target for the fast clock(osc_out_400m). Value in no. of clock cycles of the fast_clk(osc_out_400m) per divided ref_clk.
15-12 —	Reserved
11-8 HYST_PLUS	Positive hysteresis value for the tuned clock. Positive hysteresis value for the tuned clock. Set this value after the clock is tuned. Value in no. of clock cycles of the fast clock(osc_out_400m).
7-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3-0 HYST_MINUS	Negative hysteresis value for the tuned clock. Negative hysteresis value for the tuned clock. Set this value after the clock is tuned. Value in no. of clock cycles of the fast clock(osc_out_400m).

24.9.4 Analog Control Register CTRL2 (CTRL2)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
CTRL2	20h	Analog Control Register CTRL2
CTRL2_SET	24h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL2 is 1
CTRL2_CLR	28h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL2 is 0
CTRL2_TOG	2Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in CTRL2

Function

This register contains analog control bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OSC_TUNE_VAL								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	TUNE_ST...	Reserved	TUNE_EN	Reserved	TUNE_BYP	Reserved	TUNE_INV	Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24	Program the oscillator frequency.

Table continues on the next page...

Table continued from the previous page...

Field	Function
OSC_TUNE_VAL	These bits determine the frequency of oscillator when tuning is not enabled. Frequency varies with PVT. 0 - Lowest frequency. 255 - Highest frequency.
23-15 —	Reserved
14 TUNE_START	Start/Stop tuning. 0: Stop tuning and reset the tuning logic. Oscillator runs using programmed OSC_TUNE_VAL. 1: Start tuning.
13 —	Reserved
12 TUNE_EN	Freeze/Unfreeze the tuning value. 0: Freezes the tuning at the current tuned value. Oscillator runs at the frozen tuning value. 1: Unfreezes and continues the tuning operation.
11 —	Reserved
10 TUNE_BYP	Bypass the tuning logic 0: Use the output of tuning logic to run the oscillator. 1: Bypass the tuning logic and use the programmed OSC_TUNE_VAL to run the oscillator.
9 —	Reserved
8 TUNE_INV	Inverse tuning direction. 0: Normal tuning. 1: Inverse the direction of tuning. For DEBUG only.
7-0 —	Reserved

24.9.5 Analog Control Register CTRL3 (CTRL3)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
CTRL3	30h	Analog Control Register CTRL3
CTRL3_SET	34h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL3 is 1

Table continues on the next page...

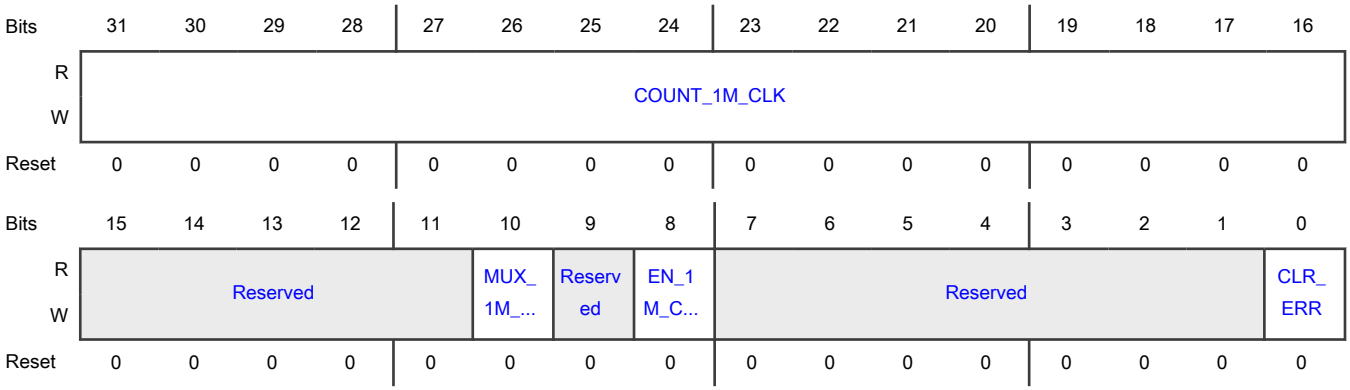
Table continued from the previous page...

Register	Offset	Description
CTRL3_CLR	38h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL3 is 0
CTRL3_TOG	3Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in CTRL3

Function

This register contains analog control bits.

Diagram



Fields

Field	Function
31-16 COUNT_1M_CLK	Count for the locked clk_1m_out. Desired target for the locked clk_1m_out. Value in no. of clock cycles of the fast_clk(osc_out_400m) per divided ref_clk. This value should be 4 to 8 counts less than TARGET_COUNT-HYST_MINUS.
15-11 —	Reserved
10 MUX_1M_CLK	Select free/locked 1MHz output 0: Select free-running 1MHz to be put out on clk_1m_out. 1: Select locked 1MHz to be put out on clk_1m_out.
9 —	Reserved
8 EN_1M_CLK	1: Disable clk_1m_out. 0: Enables the output clk_1m_out. Though the name is EN_1M_CLK, it's funtion is inverted in analog to make this clock enabled by default. 1: Disable the output clk_1m_out.
7-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 CLR_ERR	Clear the error flag CLK1M_ERR 0b - Normal operation 1b - Clears the error flag CLK1M_ERR in status register STAT0.

24.9.6 Analog Status Register STAT0 (STAT0)

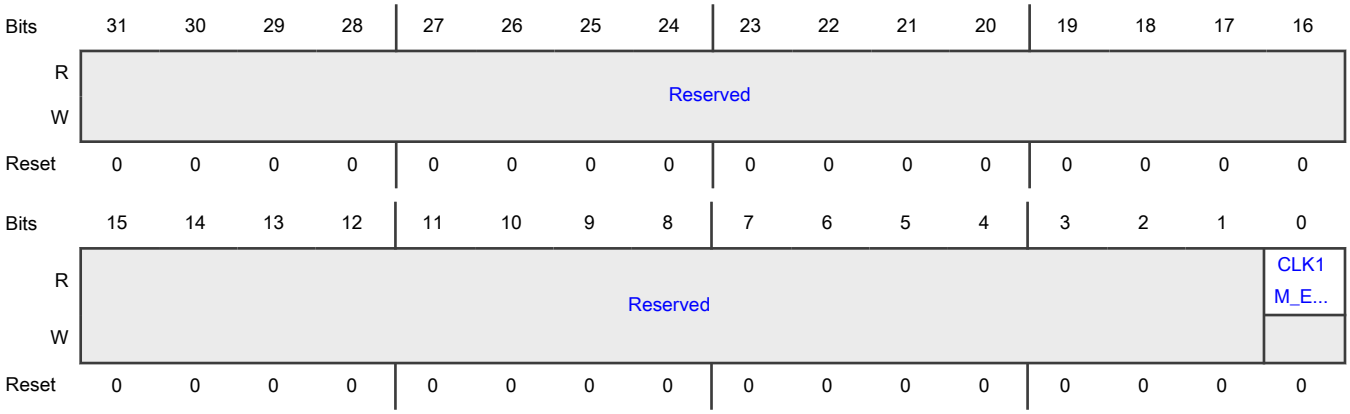
Offset

Register	Offset
STAT0	50h

Function

This register contains analog status bits.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CLK1M_ERR	Error flag for clk_1m_locked. Flag indicates that the count_1m count wasn't reached within one divided ref_clk period. 0b - No error. 1b - Indicates error.

24.9.7 Analog Status Register STAT1 (STAT1)

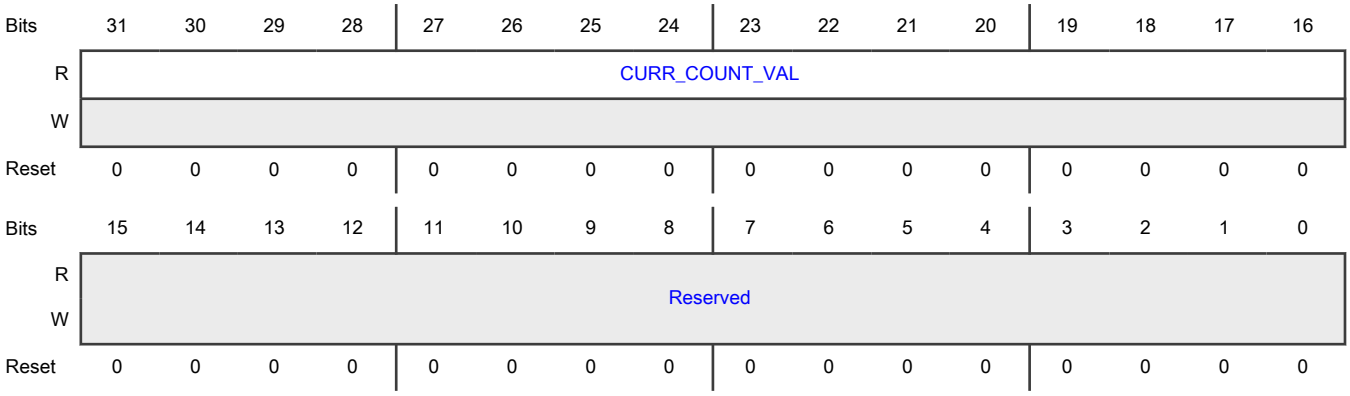
Offset

Register	Offset
STAT1	60h

Function

This register contains analog status bits.

Diagram



Fields

Field	Function
31-16 CURR_COUNT_VAL	Current count for the fast clock. Current count during the tuning process. This value converges to TARGET_COUNT as tuning progresses.
15-0 —	Reserved

24.9.8 Analog Status Register STAT2 (STAT2)

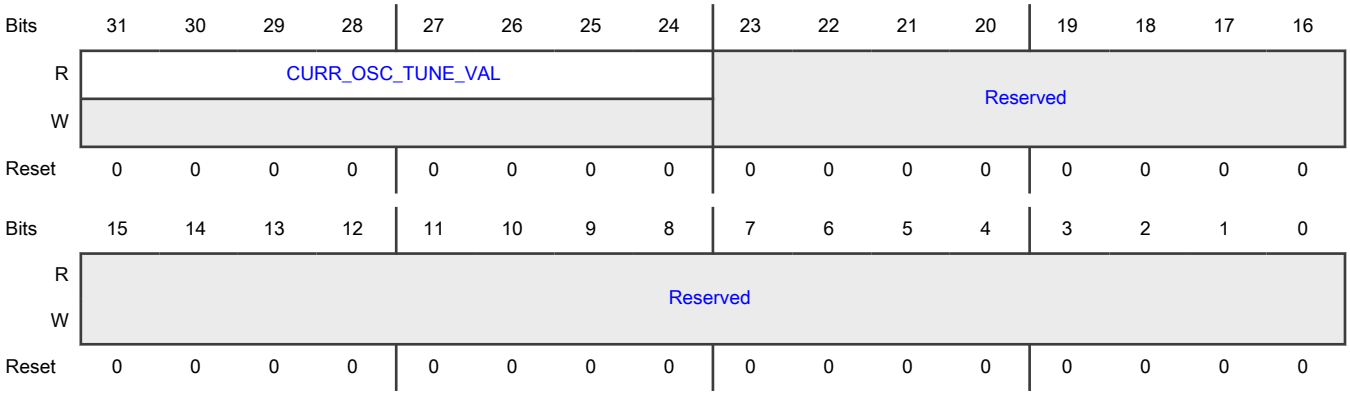
Offset

Register	Offset
STAT2	70h

Function

This register contains analog status bits.

Diagram



Fields

Field	Function
31-24 CURR_OSC_TUNE_VAL	Current tuning value used by oscillator. Current OSC_TUNE_VAL that's being used by the oscillator. This value changes as the tuning progresses.
23-0 —	Reserved

Chapter 25
Battery-Backed Non-Secure Module (BBNSM)

25.1 Chip-specific BBNSM Information

Table 159. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

To know BBNSM's interrupt request assignment at chip level, see the table "System interrupt summary" in the chapter "Interrupt, DMA Events, and XBAR Assignments".

WAKEUP pin is a pin connected to below input of BBNSM:

Table 160. Inputs to BBNSM

wakeup_request	Input	Request for system wakeup

NOTE

BBNSM_PAD_CTRL[PAD_CTRLn] detail descriptions:

- BBNSM PAD_CTRL[2]: POR_B pull enable, 0 – enable, 1 – disable, default - enabled
- BBNSM PAD_CTRL[3]: POR_B pull select, 0 – pull up, 1 – pull down, default – pull up
- BBNSM PAD_CTRL[4]: ONOFF pull enable, 0 – enable, 1 – disable, default - enabled
- BBNSM PAD_CTRL[5]: ONOFF pull select, 0 – pull up, 1 – pull down, default – pull up
- BBNSM PAD_CTRL[6]: WAKEUP pull enable, 0 – enable, 1 – disable, default - enabled
- BBNSM PAD_CTRL[7]: WAKEUP pull select, 0 – pull down, 1 – pull up, default – pull down
- BBNSM PAD_CTRL[0]: TESTMODE pull enable, 0 – enable, 1 – disable, default - enabled
- BBNSM PAD_CTRL[1]: TESTMODE pull select, 0 – pull down, 1 – pull up, default – pull down

25.2 Overview

BBNSM serves as non-volatile logic and storage for the system.

25.2.1 Block diagram

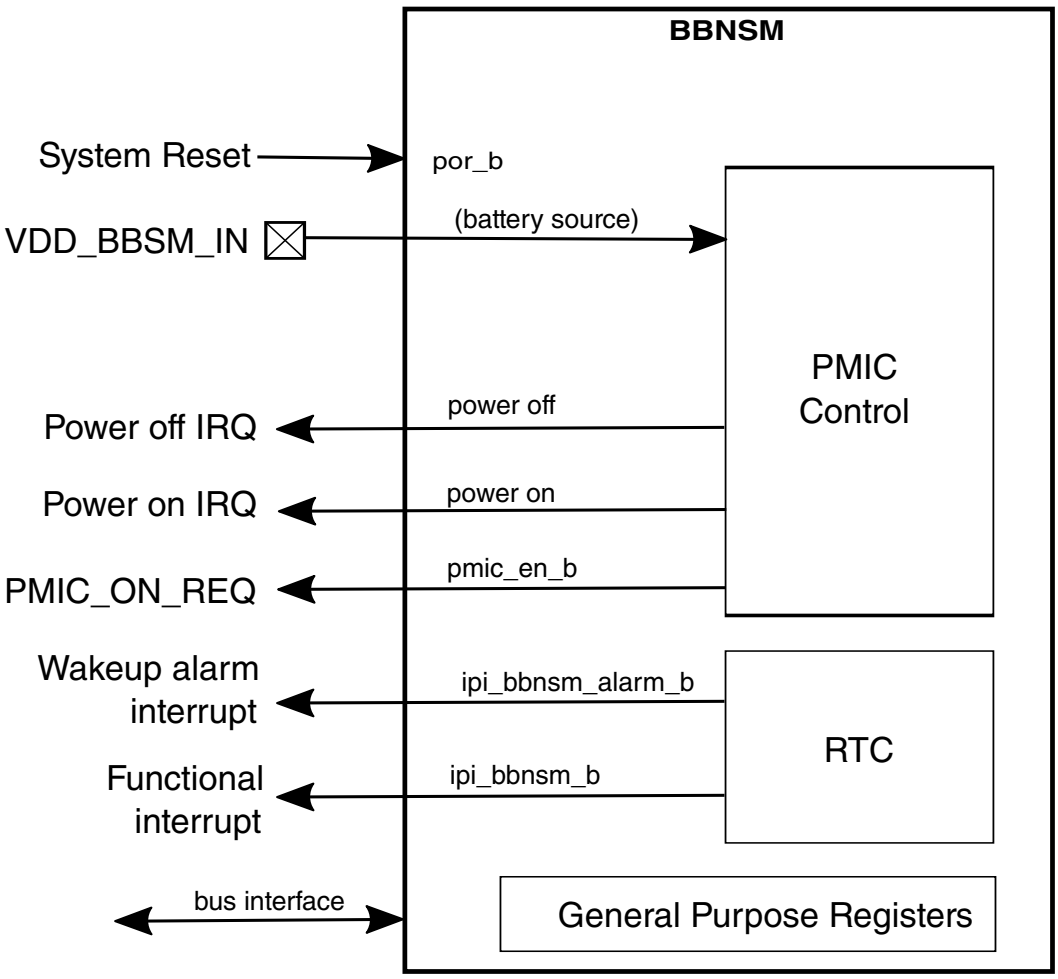


Figure 94. BBNSM block diagram

25.2.2 Features

BBNSM implements the following features:

Table 161. BBNSM feature list

Feature	Description
Non-volatile storage and logic (PMIC logic)	• If the BBNSM power input is connected to an uninterrupted power supply: — BBNSM register values are maintained even when the main SoC is powered off — BBNSM logic continues to operate even when the main SoC is powered off

Table continues on the next page...

Table 161. BBNSM feature list (continued)

Feature	Description
General-purpose register	• The general-purpose register is available to software to store data. It is 8 registers of 32 bits each by default.
Real-time counter (RTC)	• Triggers an interrupt if the RTC reaches its maximum value. • A time alarm can be programmed to generate an interrupt at a specific time.
On/Off Control	• Allows connectivity to a Power Management Integrated Circuit (PMIC) or other voltage regulator devices. • Configurable debounce support for button presses.

25.3 Functional description

The following sections describe functional details of the BBNSM module.

25.3.1 Real-Time Counter (RTC)

BBNSM implements a real-time counter that:

- Retains its state and continues counting even when the main chip is powered down. The BBNSM power input is connected to an uninterrupted power supply
- Generates an interrupt when it reaches the maximum value of all ones. It will then rollover and continue to count
- Generates a functional interrupt request at a particular time by setting the desired time in the Time Alarm register and setting BBNSM_CTRL[TA_EN]
- Can be written at any time
- Can be calibrated so that its count frequency tracks some other time source
- Can be configured to capture the time of a security violation

25.3.1.1 Time Counter Calibration

The real-time counter accuracy may suffer from a drift in the clock, which is used to increment the RTC register. To compensate for this drift, a clock calibration mechanism can adjust the RTC value. It is up to the system processor to decide whether calibration is required or not. If RTC correction is required, enable the mechanism and set the calibration value in the control register. The calibration value (BBNSM_CTRL[CAL_VAL]) includes the sign bit, which is implemented in twos' complement.

If the calibration mechanism is enabled, the calibration value is added or subtracted from the RTC on a periodic basis, once per 32768 cycles of the RTC clock.

The following table shows the available correction range.

Table 162. Real-time counter calibration settings

Calibration setting (BBNSM_CTRL[CAL_VAL])	Correction in counts per 32768 cycles of the counter clock
01111	+15
...	...
00010	+2

Table continues on the next page...

Table 162. Real-time counter calibration settings (continued)

Calibration setting (BBNSM_CTRL[CAL_VAL])	Correction in counts per 32768 cycles of the counter clock
00001	+1
00000	0
11111	-1
11110	-2
...	...
10001	-15
10000	-16

25.3.1.2 Time Alarm

A time alarm is generated once the most significant 32 bits of the real-time counter match the value in the Time Alarm register. The time alarm can generate interrupts to alert the host processor and can wake up the host processor from one of its low-power modes. This alarm can also wake up the entire system in the power-down mode by asserting the wake-up external output signal.

25.3.2 On/Off Control

The on/off logic inside the BBNSM allows for connecting directly to a PMIC or other voltage regulator device. The module has a button input signal and a wakeup request input signal. It also has two interrupts (power off and power on) and an active-low PMIC enable (pmic_en_b) output. The module can function in two different modes of operation (Dumb PMIC mode and Smart PMIC mode).

25.3.2.1 Smart PMIC Mode

In Smart PMIC mode, BBNSM can generate interrupt outputs (power off and power on) like in Dumb PMIC mode. However, the pmic_en_b signal is a pulse and active-high signal in Smart PMIC mode instead of a level and active-low signal in Dumb PMIC mode. The only configuration option available for this mode is the debounce configuration, which is used for the power off and power on interrupts in the same way as Dumb PMIC mode.

25.3.2.1.1 Smart PMIC Enable Control

The active-high pmic_en_b output asserts if either condition occurs:

- the button input signal asserts
- the wakeup alarm triggers when system power has failed (isolation enable is asserted)

25.3.2.2 Dumb PMIC Mode

In Dumb PMIC mode, BBNSM can generate interrupt outputs (power off and power on) and control the active-low PMIC enable output (pmic_en_b).

25.3.2.2.1 Dumb PMIC Enable Control

The pmic_en_b signal is active (ON state) at reset. In the ON state, the pmic_en_b is deactivated (to OFF state) if either condition occurs:

- the button input signal asserts longer than the BBNSM_CTRL[BTN_TIMEOUT] configuration
- the BBNSM_CTRL[TOSP] bit is set

In the OFF state, the pmic_en_b is activated (to ON state) if either condition occurs:

- the button input signal asserts longer than the BBNSM_CTRL[TURN_ON_TIME] configuration
- the external wakeup request signal asserts
- the wakeup alarm triggers from programming the time alarm and enabling the real-time counter

The behavior of the pmic_en_b signal is described in the state machine below.

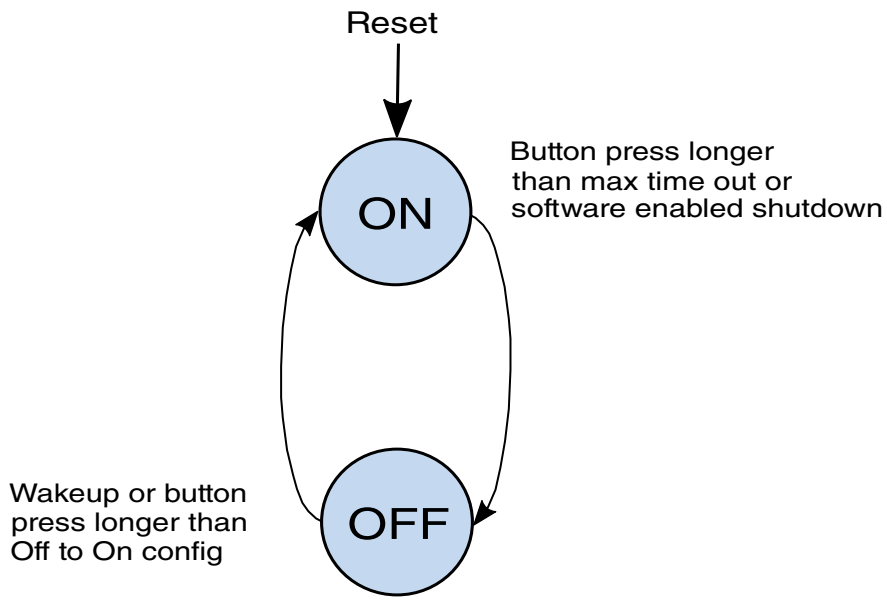


Figure 95. PMIC Enable State Machine (Dumb PMIC Mode)

25.3.3 Clocking

Table 163. BBNSM Clock

Clock	Description
32kHZ RTC clock	Clock is used by BBNSM real-time counter. This clock does not need to be synchronous with other clocks

25.3.4 Reset

Table 164. BBNSM Reset

Reset	Description
por_b	Power-on reset

25.3.5 Interrupts and Alarms

BBNSM provides the following interrupts and alarms:

Table 165. Interrupts and alarms summary

Interrupt/Violation	Event	Default configuration[1]	Configuration options
Functional interrupt (ipi_bbnsm_b)	RTC rollover	Disable	Enable/disable
Wakeup alarm interrupt (ipi_bbnsm_alarm_b)	RTC time alarm	Disable	Enable/disable
Power off interrupt	Button input signal	50 msec debounce	Debounce time
Power on interrupt	Button input signal	50 msec debounce	Debounce time

1. Default configuration refers to the state after reset.

25.3.5.1 PMIC Interrupts

The interrupts behave identically in Smart and Dumb PMIC modes.

The power on asserts if:

- system power has failed (isolation enable is asserted), **and**
- the button input signal asserts longer than the BBNSM_CTRL[DEBOUNCE] configuration

The power off asserts if:

- system power has not failed (isolation enable is not asserted), **and**
- the button input signal asserts longer than the BBNSM_CTRL[DEBOUNCE] configuration

25.4 External Signals

The table below lists the external signals to the BBNSM.

Table 166. BBNSM external signals

Signal	Description	Direction
ONOFF (btn)	Active-high external on/off button	Input
PMIC_ON_REQ (pmic_en_b)	PMIC on request signal. Active-low supply enable in dumb PMIC mode, active-high pulse in smart PMIC mode	Output

25.5 Initialization

There are no initialization steps.

25.6 Memory Map and register definition

This section includes the BBNSM module memory map and detailed descriptions of all registers.

25.6.1 BBNSM register descriptions

25.6.1.1 BBNSM memory map

BBNSM base address: 4444_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	BBNSM Version ID Register (BBNSM_VID)	32	R	0001_013Eh
4h	BBNSM Features Register (BBNSM_FEATURES)	32	R	0000_0020h
8h	BBNSM Control Register (BBNSM_CTRL)	32	RW	0100_0005h
10h	BBNSM Interrupt Enable Register (BBNSM_INT_EN)	32	RW	0000_0005h
14h	BBNSM Events Register (BBNSM_EVENTS)	32	RW	0000_0005h
24h	BBNSM External Pad Control Register (BBNSM_PAD_CTRL)	32	RW	0000_0000h
40h	BBNSM Real-Time Counter LS Register (BBNSM_RTC_LS)	32	RW	0000_0000h
44h	BBNSM Real-Time Counter MS Register (BBNSM_RTC_MS)	32	RW	0000_0000h
50h	BBNSM Time Alarm Register (BBNSM_TA)	32	RW	0000_0000h
300h - 31Ch	General Purpose Register Word a (GPR0 - GPR7)	32	RW	0000_0000h

25.6.1.2 BBNSM Version ID Register (BBNSM_VID)

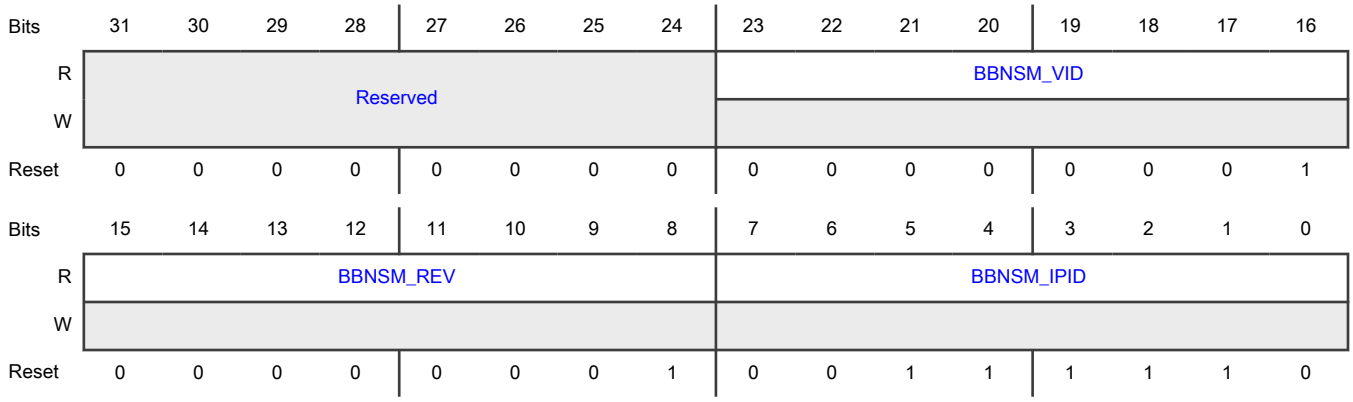
Offset

Register	Offset
BBNSM_VID	0h

Function

The BBNSM Version ID register can be used by software to differentiate between different versions of the BBNSM and determine what options are implemented in this version of BBNSM. The bit assignments of this register appear below.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 BBNSM_VID	BBNSM Version ID Version identifier or major version ID of the BBNSM.
15-8 BBNSM_REV	BBNSM Revision Revision number or minor version ID of the BBNSM.
7-0 BBNSM_IPID	BBNSM IP ID Identifier for the Battery-Backed Non-Secure Module.

25.6.1.3 BBNSM Features Register (BBNSM_FEATURES)

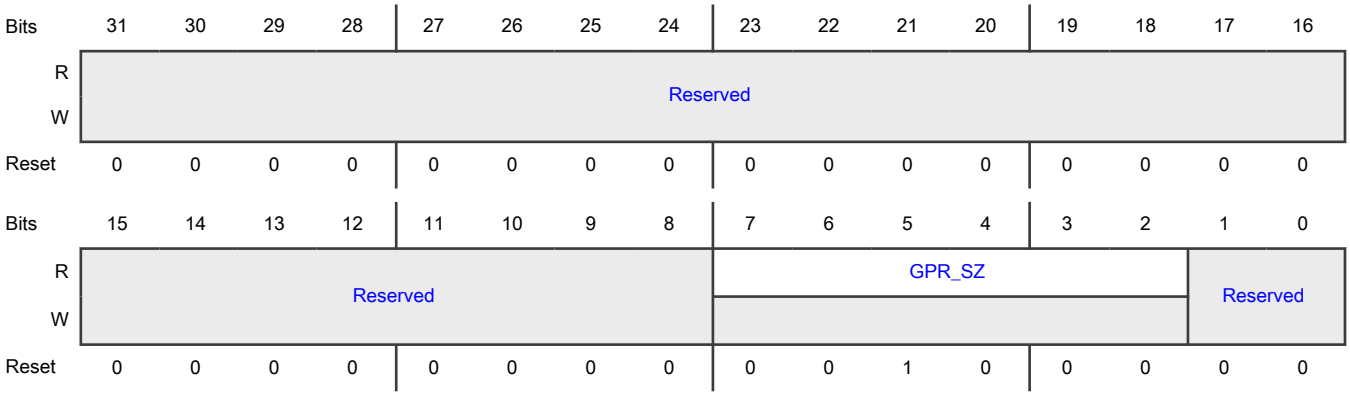
Offset

Register	Offset
BBNSM_FEATURES	4h

Function

The BBNSM Features register indicates which optional features are implemented in this version of BBNSM. The bit assignments of this register appear below.

Diagram



Fields

Field	Function
31-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7-2 GPR_SZ	GPR Register Array Size BBNSM implements a register array of 32-bit words that ELKe firmware can use to retain the GPR values across ELKe power cycles. 00_0000b - This version of BBNSM does not implement a general-purpose register array. All other values - The number of 32-bit words implemented in the general-purpose register array.
1-0 —	Reserved

25.6.1.4 BBNSM Control Register (BBNSM_CTRL)

Offset

Register	Offset
BBNSM_CTRL	8h

Function

The BBNSM Control register is used to enable features such as the real-time counter and the time alarm. It can also be used to calibrate the RTC or control the on/off logic for connecting to a PMIC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				Reserved		TOSP	DP_EN	PK_OVR	PK_EN	TURN_ON_TIME		DEBOUNCE		BTN_TIMEOUT	
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			CAL_VAL					Reserved			CAL_EN	TA_EN		RTC_EN	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-29 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-26 —	Reserved
25 TOSP	Turn Off System Power When set, a signal is sent to the PMIC to turn off the system power. This bit will clear once power is off. This bit is only valid when Dumb PMIC is enabled. 0b - Leave system power on. 1b - Turn off system power when Dumb PMIC is enabled.
24 DP_EN	Dumb PMIC Enable When set, software can control the system power. When cleared, the system requires a Smart PMIC to automatically turn power off. 0b - Smart PMIC is enabled. 1b - Dumb PMIC is enabled.
23 PK_OVR	PMIC On Request Override The value written to PK_OVR will be asserted on output signal bbnsm_pk_override. That signal is used to override the IOMUX control for the PMIC I/O pad. 0b - PMIC On Request Override is disabled. 1b - PMIC On Request Override is enabled.
22 PK_EN	PMIC On Request Enable The value written to PK_EN will be asserted on output signal bbnsm_pk_en. That signal is used to turn off the pullup/pulldown circuitry in the PMIC I/O pad. 0b - PMIC On Request is disabled. 1b - PMIC On Request is enabled.
21-20 TURN_ON_TIME	Turn-On Time Amount of time that button is pressed before pmic_en_b is asserted to turn on the SoC power. 00b - 500 milliseconds. 01b - 50 milliseconds. 10b - 100 milliseconds. 11b - 0 milliseconds.
19-18 DEBOUNCE	Debounce Time Amount of debounce time for the BTN input signal. 00b - 50 milliseconds. 01b - 100 milliseconds.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - 500 milliseconds. 11b - 0 milliseconds.
17-16 BTN_TIMEOUT	Button Press Timeout Time it takes to hold the button to request a power down. 00b - 5 seconds. 01b - 10 seconds. 10b - 15 seconds. 11b - Timeout disabled. Long button presses will not request a power down.
15-13 —	Reserved
12-8 CAL_VAL	Calibration Value Defines signed calibration value for the RTC. This field can be programmed only when RTC calibration is disabled. This is a 5-bit twos' complement value. Hence, the allowable calibration values are in the range from -16 to +15 counts per 32768 ticks of the counter clock. 0_0000b - +0 counts per each 32768 ticks of the counter clock. 0_0001b - +1 counts per each 32768 ticks of the counter clock. 0_0010b - +2 counts per each 32768 ticks of the counter clock. 0_1111b - +15 counts per each 32768 ticks of the counter clock. 1_0000b - -16 counts per each 32768 ticks of the counter clock. 1_0001b - -15 counts per each 32768 ticks of the counter clock. 1_1110b - -2 counts per each 32768 ticks of the counter clock. 1_1111b - -1 counts per each 32768 ticks of the counter clock.
7-5 —	Reserved
4 CAL_EN	Calibration Enable Enables the RTC calibration mechanism when set. This bit cannot be changed once set. 0b - RTC Time calibration is disabled. 1b - RTC Time calibration is enabled.
3-2 TA_EN	Time Alarm Enable 01b - Disable the time alarm. 10b - Enable the time alarm. A time alarm event occurs if the value in the real-time counter register is equal to the value in the time alarm register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1-0 RTC_EN	Real-Time Counter Enable 01b - Disable the real-time counter. 10b - Enable the real-time counter.

25.6.1.5 BBNSM Interrupt Enable Register (BBNSM_INT_EN)

Offset

Register	Offset
BBNSM_INT_EN	10h

Function

The BBNSM Interrupt Enable Register is used to enable interrupt generation when a status event has occurred in BBNSM. Interrupts supported include RTC rollover and time alarm. The bit assignments of this register appear below.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											Reserved	TA_INT_EN		RTC_INT_EN	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-5 —	Reserved
4 —	Reserved
3-2 TA_INT_EN	Time Alarm Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE To use TA_INT_EN, enable the RTC_INT_EN. 01b - Do not issue an interrupt when RTC has reached alarm time. The interrupt is cleared when this value is written. 10b - Issue an interrupt when RTC has reached alarm time.
1-0 RTC_INT_EN	Real-Time Counter Rollover Interrupt Enable 01b - Do not issue an interrupt when RTC has rolled over. The interrupt is cleared when this value is written. 10b - Issue an interrupt when RTC has rolled over.

25.6.1.6 BBNSM Events Register (BBNSM_EVENTS)

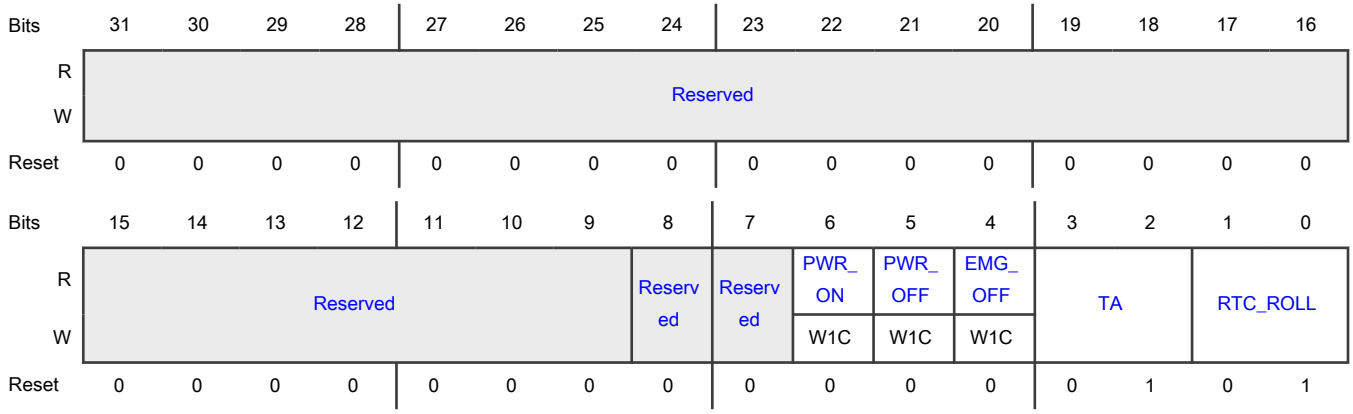
Offset

Register	Offset
BBNSM_EVENTS	14h

Function

The BBNSM Events Register indicates whether any features internal to BBNSM is reporting an event. The bit assignments of this register appear below.

Diagram



Fields

Field	Function
31-9	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
8 —	Reserved
7 —	Reserved
6 PWR_ON	Set Power On Event This bit is set when the power on interrupt is triggered, which happens when the power button is pressed longer than the configured debounce time (when system power is off). 0b - The power on interrupt has not been requested. 1b - The power on interrupt has been requested.
5 PWR_OFF	Set Power Off Event This bit is set when the power off interrupt is triggered, which happens when the power button is pressed longer than the configured debounce time (when system power is on). 0b - The power off interrupt has not been requested. 1b - The power off interrupt has been requested.
4 EMG_OFF	Emergency Off Event This bit is set when the TOSP bit in BBNSM_CTRL is written or when the power button is pressed longer than the configured button timeout time. 0b - An emergency power off has not been requested. 1b - An emergency power off has been requested.
3-2 TA	Time Alarm Event TA is only cleared when 0x10 is written. 01b - The real-time counter has not reached the alarm time. 10b - The real-time counter has reached the alarm time.
1-0 RTC_ROLL	Real-Time Counter Rollover Event RTC_ROLL is only cleared when 0x10 is written. 01b - The real-time counter has not rolled over. 10b - The real-time counter has rolled over.

25.6.1.7 BBNSM External Pad Control Register (BBNSM_PAD_CTRL)

Offset

Register	Offset
BBNSM_PAD_CTRL	24h

Function

The BBNSM External Pad Control Register allows configuration and control of I/O pads. The bit assignments of this register appear below.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...	PAD_CTR...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 PAD_CTRLn	Control I/O Pads General purpose outputs for configuring and controlling I/O pads. 0b - Deasserts bit n in bbnsm_pad_ctrl[n] 1b - Assert bit n in bbnsm_pad_ctrl[n]

25.6.1.8 BBNSM Real-Time Counter LS Register (BBNSM_RTC_LS)

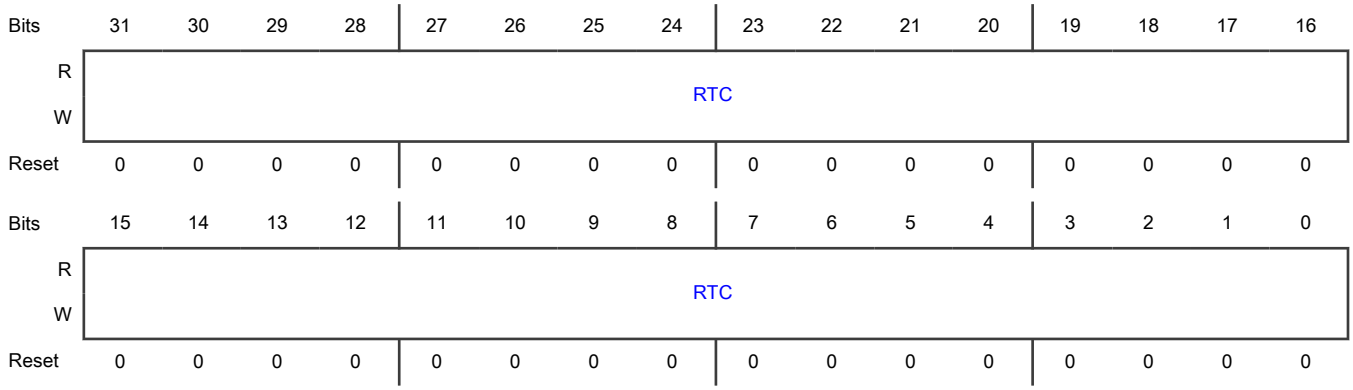
Offset

Register	Offset
BBNSM_RTC_LS	40h

Function

The BBNSM Real-Time Counter LSB register contains the 32 least-significant bits of the real-time counter.

Diagram



Fields

Field	Function
31-0 RTC	Real-time Counter The least-significant 32 bits of the RTC.

25.6.1.9 BBNSM Real-Time Counter MS Register (BBNSM_RTC_MS)

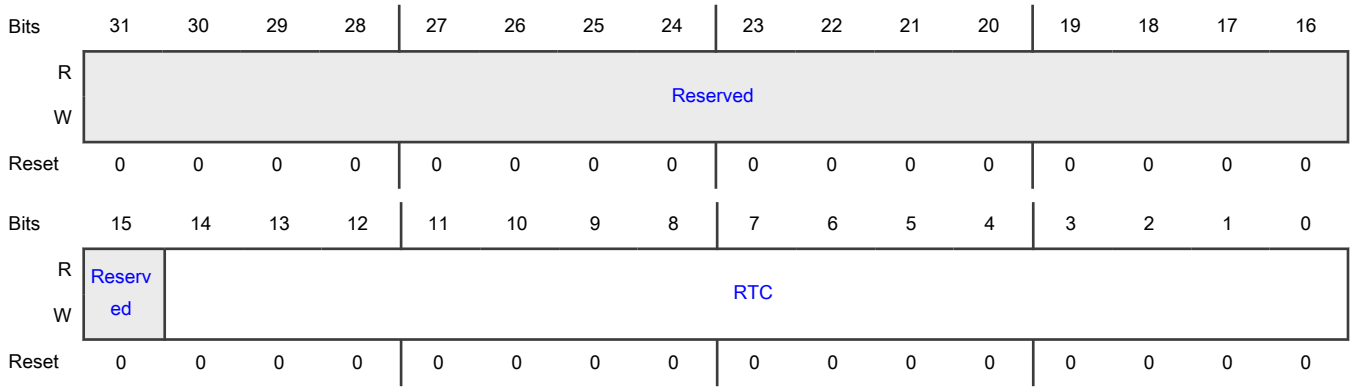
Offset

Register	Offset
BBNSM_RTC_MS	44h

Function

The BBNSM Real-Time Counter MS register contains the 15 most-significant bits of the real-time counter.

Diagram



Fields

Field	Function
31-15 —	Reserved
14-0 RTC	Real-Time Counter The most-significant 15 bits of the RTC.

25.6.1.10 BBNSM Time Alarm Register (BBNSM_TA)

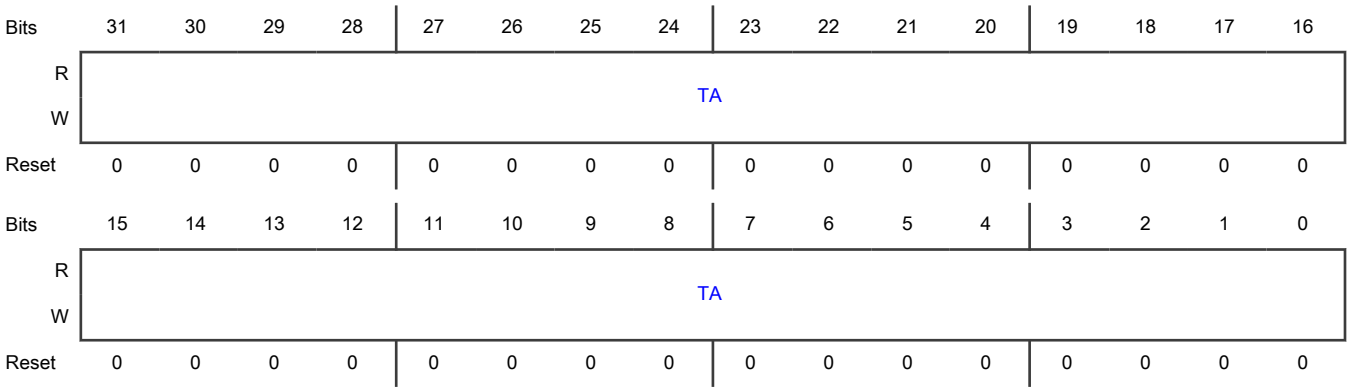
Offset

Register	Offset
BBNSM_TA	50h

Function

The BBNSM Time Alarm register contains the 32-bit value of the time alarm.

Diagram



Fields

Field	Function
31-0 TA	Time Alarm Value The 32 bits of the time alarm value.

25.6.1.11 General Purpose Register Word a (GPR0 - GPR7)

Offset

For a = 0 to 7:

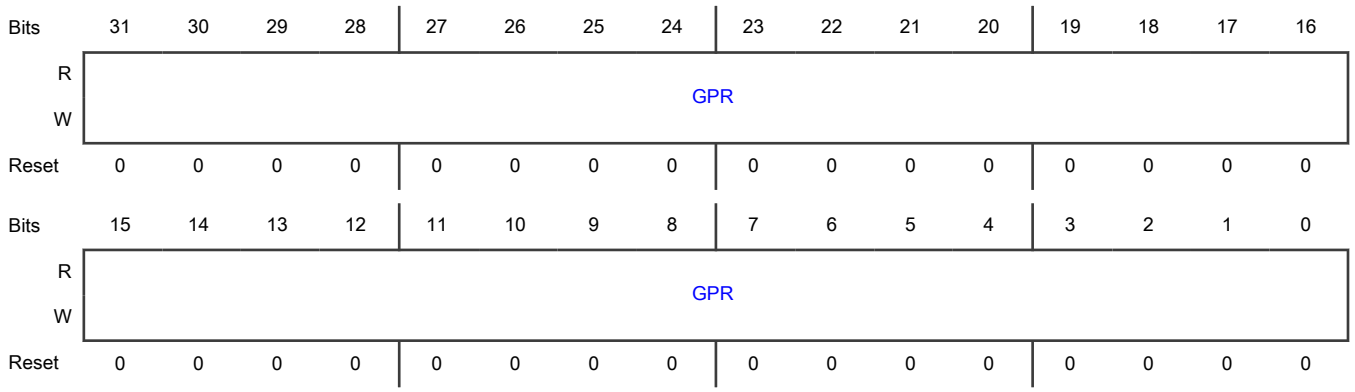
Register	Offset
GPRa	300h + (a × 4h)

Function

The GPR register array holds data whose value is retained across power cycles of the SoC. The words of the array can be used to store general purpose data.

It is up to ELKe firmware to determine how the space in the array is allocated.

Diagram



Fields

Field	Function
31-0 GPR	32 bits of the GPR.

Chapter 26
Fusemap

26.1 Overview

This section details the various fuses and their associated configuration settings. This chapter also describes various modes and the selection of the required boot devices.

NOTE

The fuses marked as “Reserved” are reserved for NXP internal (and future) use only, or are security related. Customers must not attempt to burn these, because the IC behavior may be unpredictable. The reserved fuses can be read as either '0' or '1'. Security-related fuses are described in the Security Reference Manual (SRM) for this device.

26.1.1 Lock bits

There are three bits per lock, read, write, and overwrite.

The write lock bits are used by the controller to control writing/programming of the fuse bits and the shadow registers (which hold the value read from the fuses). Each word can be write locked independently, which means that it can no longer be programmed. If the word is write locked, only read is permitted and the overwriting of the shadow registers is not permitted. If the word is not locked, then it can be read sensed and programmed and the shadow registers can be overwritten (to allow emulation of programming without actually blowing fuses). The supplemental array of fuses are always override locked.

Read lock bits are a specific set of fuses that give you the ability to block bus access to specific words. Read locks are on parts of the array consisting of multiple words. The bits for these are stored in the supplementary array section in the last word.

Table 167. Lock Fuses

Fuse Word Index	Fuse Bit Index	# of fuses	Fuse Lock	Function	Setting	Used By
0	0	3	BOOT_CFG_LOCK	Boot Configuration Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
0	12	3	LOCK_CFG_LOCK	Lock for Fuse Lock bits	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register)	ELE

Table continues on the next page...

Table 167. Lock Fuses (continued)

Fuse Word Index	Fuse Bit Index	# of fuses	Fuse Lock	Function	Setting	Used By
					xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	
1	35	3	COM_DEVICE_ID0_LOCK	Communication Device ID0 Lock - Ethernet	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
1	38	3	COM_DEVICE_ID1_LOCK	Communication Device ID1 Lock - USB, Ethernet	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
8	268	3	GPR0_LOCK	General Purpose 0 Fuse Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register)	ELE

Table continues on the next page...

Table 167. Lock Fuses (continued)

Fuse Word Index	Fuse Bit Index	# of fuses	Fuse Lock	Function	Setting	Used By
					xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	
9	288	3	GPR1_LOCK	General Purpose 1 Fuse Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
9	291	3	GPR2_LOCK	General Purpose 2 Fuse Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
9	294	3	GPR3_LOCK	General Purpose 3 Fuse Lock / A-Core ROM Patch Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register)	ELE

Table continues on the next page...

Table 167. Lock Fuses (continued)

Fuse Word Index	Fuse Bit Index	# of fuses	Fuse Lock	Function	Setting	Used By
					xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	
9	297	3	GPR4_LOCK	General Purpose 4 Fuse Lock / A-Core ROM Patch Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
9	300	3	GPR5_LOCK	General Purpose 5 Fuse Lock / A-Core ROM Patch Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
10	320	3	GPR6_LOCK	General Purpose 6 Fuse Lock / A-Core ROM Patch Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register)	ELE

Table continues on the next page...

Table 167. Lock Fuses (continued)

Fuse Word Index	Fuse Bit Index	# of fuses	Fuse Lock	Function	Setting	Used By
					xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	
10	323	3	GPR7_LOCK	General Purpose 7 Fuse Lock / A-Core ROM Patch Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
10	326	3	GPR8_LOCK	General Purpose 8 Fuse Lock / A-Core ROM Patch Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
10	329	3	GPR9_LOCK	General Purpose 9 Fuse Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register)	ELE

Table continues on the next page...

Table 167. Lock Fuses (continued)

Fuse Word Index	Fuse Bit Index	# of fuses	Fuse Lock	Function	Setting	Used By
					xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	
10	332	3	GPR10_LOCK	General Purpose 10 Fuse Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
11	352	3	GPR11_LOCK	General Purpose 11 Fuse Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
11	355	3	GPR12_LOCK	General Purpose 12 Fuse Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register)	ELE

Table continues on the next page...

Table 167. Lock Fuses (continued)

Fuse Word Index	Fuse Bit Index	# of fuses	Fuse Lock	Function	Setting	Used By
					xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	
11	358	3	GPR13_LOCK	General Purpose 13 Fuse Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
11	361	3	GPR14_LOCK	General Purpose 14 Fuse Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register) xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	ELE
11	364	3	GPR15_LOCK	General Purpose 15 Fuse Lock	000 - Unlocked (fuses can be read, sensed, burned or over-ridden in the corresponding OTP shadow register)	ELE

Table continues on the next page...

Table 167. Lock Fuses (continued)

Fuse Word Index	Fuse Bit Index	# of fuses	Fuse Lock	Function	Setting	Used By
					xx1 - WP (Write Protect, the controlled field can not be burned) x1x - OP (Over-ride Protect, the controlled field shadow registers can not be over-written) 1xx - RP (Read Protect, the controlled field can be sensed only, but can't be read from shadow registers)	

26.2 Boot Fusemap

This section describes the various modes and the selection of the required boot devices. A separate map is given for each and every boot device. The device select is specified by the BOOT_CFG0[2:0] fuses.

Table 168. Boot device select

Boot Device	BOOT_CFG0[2]	BOOT_CFG0[1]	BOOT_CFG0[0]
FlexSPI (Serial NOR)	0	0	0
FlexSPI (Serial NAND)	0	0	1
MMC/eMMC	0	1	0
SD	0	1	1
SEMC SLC (NAND)	1	0	0
Serial Boot	1	0	1

NOTE

The fuses marked as “Reserved” are reserved for NXP internal (and future) use only. Customers **must not** attempt to burn these, because the IC behavior may be unpredictable. The reserved fuses can be read as either '0' or '1'. Security-related fuses are described in the Security Reference Manual (SRM) for this device.

The Fuse Word Index and Fuse Bit Index mapping for the BOOT_CFGn fuses is provided in the following table.

The following table applies for all boot devices.

Table 169. Configuration fuses

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
24 [775:768]	FORCE_BT_FROM_FUSE	BT_FUSE_SEL 0 - BOOT_MO	DIS_SDMMC_MFG_BOOT 0 - Enable	RECOVERY_BOOT_EN	Reserved	BOOT_MODE_FROM_FUSE 0 - FlexSPI NOR 1 - FlexSPI NAND		

Table continues on the next page...

Table 169. Configuration fuses (continued)

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
	0 - Boot mode determined by BOOT_MODE pins 1 - Boot mode determined by BOOT_MODE_FROM_FUSE	DE_FROM_FUSE is invalid 1 - BOOT_MODE_FROM_FUSE is valid	1 - Disable	0 - Recovery boot is disabled 1 - Recovery boot is enabled		2 - eMMC 3 - SD 4 - SEMC SLC raw NAND 5 - Serial Boot		
24 [783:776]	DIS_SERIAL_DWNLD 0 - Allow Serial downloader mode 1 - Disable Serial Downloader mode	DIS_BT_MODE_API 0 - Allow boot from ROM API call 1 - Disallow boot from ROM API call	DIS_INFINITE_LOOP 0 - Infinite loop is allowed 1 - Infinite Loop is forbidden	DIS_USB_HID_DWNLD 0 - Enable USB in Serial Downloader 1 - Disable USB in Serial Downloader	DIS_SPI_DWNLD 0 - Enable SPI in Serial Downloader 1 - Disable SPI in Serial Downloader	DIS_UART_DWNLD 0 - Enable UART in Serial Downloader 1 - Disable UART in Serial Downloader	Reserved	
24 [791:784]	LPSPI_INTERNAL_PULL	LPSPI_DUMMY_BYTES 0 - Default (4 cycles for dual/6 cycles for quad) n (non-zero) - 4n cycles for dual/2n cycles for quad			LPSPI_SPEED 0 - 30MHz 1 - 10MHz 2 - 30MHz Dual Read 3 - 30MHz Quad Read		LPSPI_PORT_SEL 0 - LPSPI1 1 - LPSPI2 2 - LPSPI3 3 - LPSPI4	
24 [799:792]	BOOT_FAIL_IND_EN	Reserved			BOOT_FAIL_IND_PIN n - GPIO4_IOn/GPIO_AD_n selected			

Table 170. FlexSPI (Serial NOR) boot fusemap

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
25 [807:800]	xSPI_NOR_CONNECTION_SEL	xSPI_NOR_FREQ 0 - 100MHz 1 - 120MHz 2 - 133MHz			xSPI_NOR_HOLD_TIME 0 - 500us 1 - 1ms 2 - 3ms		xSPI_NOR_PROBE_TYPE (Non-0 only when xSPI_NOR_TYPE is 0)	

Table continues on the next page...

Table 170. FlexSPI (Serial NOR) boot fusemap (continued)

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
	0 - Connected via PORTA CS0 1 - Connected via PORTB CS0	3 - 166MHz 4 - 200MHz 5 - 80MHz 6 - 60MHz 7 - 50MHz			3 - 10ms		0 - JESD216 1 - MXIC Octal 2 - Micron Octal 3 - Adesto Octal	
25 [815:808]	Reserved		xSPI_NOR_RESET_TYPE (Non-0 only when xSPI_NOR_TYPE is 0) 0 - Reset sequence is specified in FLASH_STATE_CTX Register 1 - Reset FLASH by Reset Pin before access 2 - Reset FLASH by JEDEC Hardware Reset flow 3 - Typical Reset Sequence (0x66, 0x99)		RST_REQUIRED Reset required after boot failure. Set if xSPI_NOR_ENCRYPT_XIP_EN is 1	xSPI_NOR_TYPE 0 - Boot with default 0x03 Read 1 - Reserved 2 - HyperFLASH 1V8 3 - HyperFLASH 3V0 4 - MXIC Octal Read 5 - Micron Octal DDR Read Others - Reserved		
25 [823:816]	Reserved			DIS_xSPI_NOR_FCB 0 - Enable FLASH CFG Block 1 - Disable FLASH CFG Block	XSPI_NOR_FCB_OFFSET 0 - 0x400 1 - 0x600 2 - 0xA00 3 - 0xC00		XSPI_NOR_ENCRYPT_ENG 0 - OTFAD 1 - IEE	xSPI_NOR_ENCRYPT_XIP_EN 0 - Encrypted XIP is disabled 1 - Encrypted XIP is enabled
25 [831:824]	Reserved							

Table 171. FlexSPI boot configuration fusemap

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
26 [839:832]	FLEXSPI_INSTANCE 0 - FLEXSPI1 1 - FLEXSPI2	FLEXSPI_PIN_GROUP_SEL 0 - Primary group 1 - Secondary group	xSPI_LOOPBACK_DQS_PIN_SEL 0 - Secondary(dummy) DQS pin 1 - Standard DQS pin	FLEXSPI_RESET_PIN_SEL Select the Reset Pin 0 - Reset Pin 0 1 - Reset Pin 1	FLEXSPI_PAD_OVERRIDE_EN 0 - Use default pad setting 1 - Use the pad setting specified by FLEXSPI_PAD_SETTING_OVERRIDE FUSE field	FLEXSPI_PAD_SETTING_OVERRIDE 0 - Slew rate , 0: slow slew rate, 1: fast slew rate bit 1 - Drive Strength, 0: normal driver, 1 : high driver bit 2 - Pull selection, 0: pull off, 1 - Pull on		
26 [847:840]	Reserved			FLEXSPI_DELAY_CELL_NUM 0 - Use the Delay cell value calculated by ROM Others - Use the Delay cell value in terms of 100ps				
26 [863:848]	Reserved							

Table 172. FlexSPI (Serial NAND) boot fusemap

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
27 [871:864]	xSPI_NAND_HOLD_TIME Hold Time before issuing Command to FLASH 0 - Wait until FLASH is idle by polling status register in the FLASH 1 - 500us 2 - 1ms 3 - 3ms		xSPI_NAND_SEARCH_STRIDE 0 - 64 pages 1 - 128 pages 2 - 256 pages 3 - 32 pages		xSPI_NAND_CS_INTERVAL CS de-asserted interval between two commands 0 - 100ns 1 - 200ns 2 - 400ns 3 - 50ns		xSPI_NAND_COL_ADDR_WIDTH 0 - 12 bits 1 - 13bits	xSPI_NAND_SEARCH_COUNT 0 - 1 1 -2
27 [879:872]	Reserved			xSPI_NAND_CONNECTION_SEL 0 - Connected via PORTA CS0	xSPI_NAND_RESET_TYPE 0 - No reset 1 - Reset the FLASH by 0xFF		Reserved	

Table continues on the next page...

Table 172. FlexSPI (Serial NAND) boot fusemap (continued)

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
				1 - Connected via PORB CS0	2 - Reset the FLASH by 0x66, 0x99 3 - Reset the FLASH by dedicated Reset Pin			
27 [887:880]	Reserved					xSPI_NAND_FREQ 0 - 80MHz / 1 - 100MHz / 2 - 120MHz / 3 - 133MHz / 4 - 166MHz / 5 - 200MHz 6 - 60MHz/ 7 - 30MHz		
27 [895:888]	Reserved							

Table 173. MMC/eMMC boot fusemap

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
28 [903:896]	PWR_STABLE_CYCLE 0 - 5ms 1 - 2.5ms	PWR_CYCLE_EN 0 - No power cycle 1 - Enabled via SD_RST	PWR_CYCLE 0 - 20ms 1 - 10ms 2 - 5ms 3 - 2.5ms		USDHC2_RST_POLARITY 0 - Reset active low 1 - Reset active high	USDHC1_RST_POLARITY 0 - Reset active low 1 - Reset active high	SION_DISABLE 0 - Enable 1 - Disable	USDHC_PORT 0 - USDHC1 1 - USDHC2
28 [911:904]	MMC_FAST_BOOT_ACK_EN 0 - Disabled 1 - Enabled	MMC_FAST_BOOT_EN 0 - Normal boot 1 - Fast boot	MMC_BUS_WIDTH 0 - 4-bit 1 - 8-bit 2 - 4-bit DDR 3 - 8-bit DDR		USDHC2_VSEL 0 - 3v3 1 - 1v8	USDHC1_VSEL 0 - 3v3 1 - 1v8	NO_PREIDLE_STATE 0 - Issue pre-idle command 1 - Do not issue	MMC_SPEED 0 - Normal 1 - High speed
28 [927:912]	Reserved							

Table 174. SEMC (NAND) boot fusemap

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
29 [935:928]	RDY_CHECK_TYPE	RDY_POLARITY	ACCESS_CMD 0 - IP read	EDO_MODE	SEMC_ECC_MODE 0 - BCH4	ECC_SEL 0 - NAND device ECC	PORT_WIDTH 0 - 8-bit	SEARCH_COUNT 0 - 1

Table continues on the next page...

Table 174. SEMC (NAND) boot fusemap (continued)

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
	0 - Via status register 1 - Via R/B# pin	0 - High active 1 - Low active	1 - AHB read	0 - EDO mode 1 - Non-EDO mode	1 - BCH8	1 - SEMC built-in ECC	1 - 16-bit	1 - 2
29 [943:936]	Reserved	PCS_SEL 000 - CSX0 001 - CSX1 010 - CSX2 011 - CSX3 100 - A8 Others - CSX0			Reserved	TIMING_MODE 000 - Mode0: 10MHz 001 - Mode1: 20MHz 010 - Mode2: 28MHz 011 - Mode3: 33MHz 100 - Mode4: 40MHz 101 - Mode5: 50MHz 11x - Fastest Mode		
29 [951:944]	STATUS_CMD_TYPE (Applicable only for non-ONFI devices) 0 - Common (0x70) 1 - Enhanced (0x78)	ROW_COL_ADDR_MODE (Applicable only for non-ONFI devices) 00x - 5 bytes (CA2+RA3) 010 - 4 bytes (CA2+RA2) 011 - 3 bytes (CA2+RA1) 10x - 4 bytes (CA1+RA3) 110 - 3 bytes (CA1+RA2) 111 - 2 bytes (CA1+RA1)			BOOT_SEARCH_STRIDE: Search Stride for FCB and DBBT Search strides in terms of page 0000 - 64 other: Value = 2^(BOOT_SEARCH_STRIDE)			
29 [959:952]	ONFI_COMPLIANCE_SEL 0 - ONFI 1.0 1 - Non-ONFI	Pages In Block: 000 - 128 001 - 8 010 - 16 011 - 32 100 - 64 101 - 256 110 - 512 111 - 1024			DEV_ECC_INIT_STATUS (Applicable only for non-ONFI devices) 0 - Enabled 1 - Disabled	COL_ADDRESS_WIDTH: 000 - 12bits 001 - 09bits 010 - 10bits 011 - 11bits 100 - 13bits 101 - 14bits 110 - 15bits 111 - 16bits		

Table 175. SD boot fusemap

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
30 [967:960]	PWR_STABLE_CYCLE 0 - 5ms 1 - 2.5ms	PWR_CYCLE_EN 0 - No power cycle 1 - Enabled via SD_RST	PWR_CYCLE 0 - 20ms 1 - 10ms 2 - 5ms 3 - 2.5ms		USDHC2_RST_POLARITY 0 - Reset active low 1 - Reset active high	USDHC1_RST_POLARITY 0 - Reset active low 1 - Reset active high	SION_DISABLE 0 - Enable 1 - Disable	USDHC_PORT 0 - USDHC1 1 - USDHC2
30 [975:968]	Reserved					SD_BUS_WIDTH 0 - 1-bit 1 - 4-bit	SD_SPEED 0 - Normal (SDR12) 1 - High (SDR25) 2 - SDR50 3 - SDR104	
30 [983:976]	DLL_DELAY_CALIB_STEP 00 - 1 01 - 2 10 - 4 11 - 6		Reserved					
30 [991:984]	DLL_DELAY_OVERRIDE 0 - DLL Slave Mode 1 - DLL Override Mode	DLL_DELAY_CALIB_START DLL tuning start						

Table 176. Boot parameter fusemap

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
31 [999:992]	2ND_IMG_SIZE_UNIT 0 - 256KB 1 - 1MB 2 - 4MB 3 - 16MB		2ND_IMG_OFFSET 0 - No 2nd physical image present Otherwise -2nd physical image located at 2ND_IMG_SIZE_UNIT * 2ND_IMG_OFFSET					

Table continues on the next page...

Table 176. Boot parameter fusemap (continued)

Fuse Word Index [Fuse Bit Index]	7	6	5	4	3	2	1	0
31 [1007:1000]	Reserved				WDOG_TIMEOUT_SEL 0 - 128s 1 - 64s 2 - 32s 3 - 16s 4 - 8s Others - 4s			WDOG_EN 0 - Disable all WDOGs during boot 1 - Keep WDOG1 enabled during boot
31 [1015:1008]	Reserved			XMC_CRC32_CHECK_EN 0 - Disabled 1 - Enabled	BOOT_FREQ 0 - 200MHz derived from RC400M 1 - 240MHz derived from PLL480MHz	Release_M7_RST_STAT 0 - RST assert 1 - RST release (Required to access M7 TCM)	POR_PRELOAD_CM7_TCM_ECC 0 - Not load 1 - load	POR_PRELOAD_CM33_TCM_ECC 0 - Not load 1 - load
31 [1023:1016]	Reserved	INVERT_CM33_PRELOAD (impacts POR_PRELOAD_CM33_TCM_ECC) 0 - no invert 1 - invert	Reserved					

26.3 Fusemap Descriptions Table

This section provides the chip fusemap descriptions.

NOTE

All fuse programming operations go through ELE.

NOTE

Also see the EdgeLock Secure Enclave (ELE) API Reference Guide.

The following two mechanism are used by the fuse block to ensure that its values are correct:

- Redundancy (RED) mode: In this case only 16 of the 32 bits are available (bits 15:0 are automatically copied to bits 31:16) and checked on read. Each pair of bits can be programmed at different times. This allows for multiple programming operations on a word, so bits can be configured at different times.
- ECC mode: The ECC is generated for the whole word. The whole word must be setup at one time and individual bits cannot be modified independently.

Table 177. Fusemap Descriptions

Fuse Word Index	Fuse Bit Index	# of fuses	Fuse Name	Function	Comments	Read Locked by	ECC/RED
24	768	384	BOOT_CFG	Boot Configuration	See Boot Fuses	BOOT_CFG_LOCK	ECC
312	9984	16	USB_VID	USB Vendor ID	-	COM_DEVICE_ID0_LOCK	-
312	10000	16	USB1_PID	USB0 Product ID	-	COM_DEVICE_ID0_LOCK	-
313	10016	16	USB2_PID	USB1 Product ID	-	COM_DEVICE_ID0_LOCK	-
315	10080	32	MAC1_ADDR_31_0	MAC_0 Address	Reserved for customer SW	COM_DEVICE_ID0_LOCK	-
316	10112	16	MAC1_ADDR_47_32	MAC_0 Address	Reserved for customer SW	COM_DEVICE_ID0_LOCK	-
316	10128	16	MAC2_ADDR_47_32	MAC_1 Address	Reserved for customer SW	COM_DEVICE_ID0_LOCK	-
317	10144	32	MAC2_ADDR_31_0	MAC_1 Address	Reserved for customer SW	COM_DEVICE_ID0_LOCK	-
318	10176	32	MAC3_ADDR_31_0	MAC_2 Address	Reserved for customer SW	COM_DEVICE_ID0_LOCK	-
319	10208	16	MAC3_ADDR_47_32	MAC_2 Address	Reserved for customer SW	COM_DEVICE_ID0_LOCK	-
319	10224	16	MAC4_ADDR_47_32	MAC_3 Address	Reserved for customer SW	COM_DEVICE_ID0_LOCK	-
320	10240	32	MAC4_ADDR_31_0	MAC_3 Address	Reserved for customer SW	COM_DEVICE_ID0_LOCK	-
321	10272	32	MAC5_ADDR_31_0	MAC_4 Address	Reserved for customer SW	COM_DEVICE_ID0_LOCK	-
322	10304	16	MAC5_ADDR_47_32	MAC_4 Address	Reserved for customer SW	COM_DEVICE_ID1_LOCK	-
322	10320	16	MAC6_ADDR_47_32	MAC_5 Address	Reserved for customer SW	COM_DEVICE_ID1_LOCK	-
323	10336	32	MAC6_ADDR_31_0	MAC_5 Address	Reserved for customer SW	COM_DEVICE_ID1_LOCK	-

Table continues on the next page...

Table 177. Fusemap Descriptions (continued)

Fuse Word Index	Fuse Bit Index	# of fuses	Fuse Name	Function	Comments	Read Locked by	ECC/RED
324	10368	32	MAC7_ADDR_31_0	MAC_6 Address	Reserved for customer SW	COM_DEVICE_ID1_LOCK	-
325	10400	16	MAC7_ADDR_47_32	MAC_6 Address	Reserved for customer SW	COM_DEVICE_ID1_LOCK	-
325	10416	16	MAC8_ADDR_47_32	MAC_7 Address	Reserved for customer SW	COM_DEVICE_ID1_LOCK	-
326	10432	32	MAC8_ADDR_31_0	MAC_7 Address	Reserved for customer SW	COM_DEVICE_ID1_LOCK	-
327	10464	1	MAC_ADDR_DISABLE	MAC ADDR Disable		COM_DEVICE_ID1_LOCK	-
327	10465	1	USB_ADDR_DISABLE	USB ADDR Disable		COM_DEVICE_ID1_LOCK	-
327	10466	2	CM33_TCM_CFG	CM33 TCM configuration	00 - REGULAR_TCM (128 kB Code TCM, 128 kB Sys TCM) 01 - DOUBLE_CODE_TCM (256 kB Code TCM) 10 - DOUBLE_SYS_TCM (256 kB Sys TCM) 11 - Reserved	COM_DEVICE_ID1_LOCK	-
327	10468	3	CM7_TCM_CFG	CM7 TCM configuration	000 - REGULAR_TCM (256 kB ITCM, 256 kB DTCM) 001 - DOUBLE_ITCM (512 kB ITCM) 010 - DOUBLE_DTCM (512 kB DTCM) 100 - HALF_ITCM (128 kB ITCM, 384 kB DTCM) 101 - HALF_DTCM (384 kB ITCM, 128 kB DTCM)	COM_DEVICE_ID1_LOCK	-
327	10471	2	FORCE_EWMOUT	EWM_FORCE_OUT	00 - No EWM_OUT output is forced to GPIO, user should use IOMUXC to	COM_DEVICE_ID1_LOCK	-

Table continues on the next page...

Table 177. Fusemap Descriptions (continued)

Fuse Word Index	Fuse Bit Index	# of fuses	Fuse Name	Function	Comments	Read Locked by	ECC/RED
					configure pinmux and select GPIO to output EWM_OUT 01 - EWM_OUT output is forced to GPIO_AON_27 (ALT7), GPIO_AON_27 IOMUXC control register is override and no longer works. 10 - EWM_OUT output is forced to GPIO_AD_12 (ALT7), GPIO_AD_12 IOMUXC control register is override and no longer works. 11 - EWM_OUT output is forced to GPIO_AD_29 (ALT7), GPIO_AD_29 IOMUXC control register is override and no longer works. Attention - SoC or IOMUXC must mask FORCE_EWMOUT in scan mode		

Chapter 27
System Reset Controller (SRC)

27.1 Chip-specific SRC information

Table 178. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

27.2 Overview

SRC is responsible for the generation of all the system reset signals and boot argument latching. Its main functions are as follows:

- Deals with all global system reset sources from other modules, and generates global system reset.
- Responsible for power gating of MIXs (slices) and their memory low power control. To make sure the chip's function while turning off some MIXs or memory power, SRC uses a fixed sequence to power off/on MIXs.

27.2.1 Block diagram

There are two major sub-modules in SRC according to its main functions, as shown in the block diagram.

- Global system reset control: collects all reset sources from other modules to generate global system reset to the whole chip.
- Six slices or MIXs: implements UPI handshake with GPC, and generates MIX Edgelock handshake, isolation, reset and power off signals, and MIX memory low power control signals, based on GPC's request.

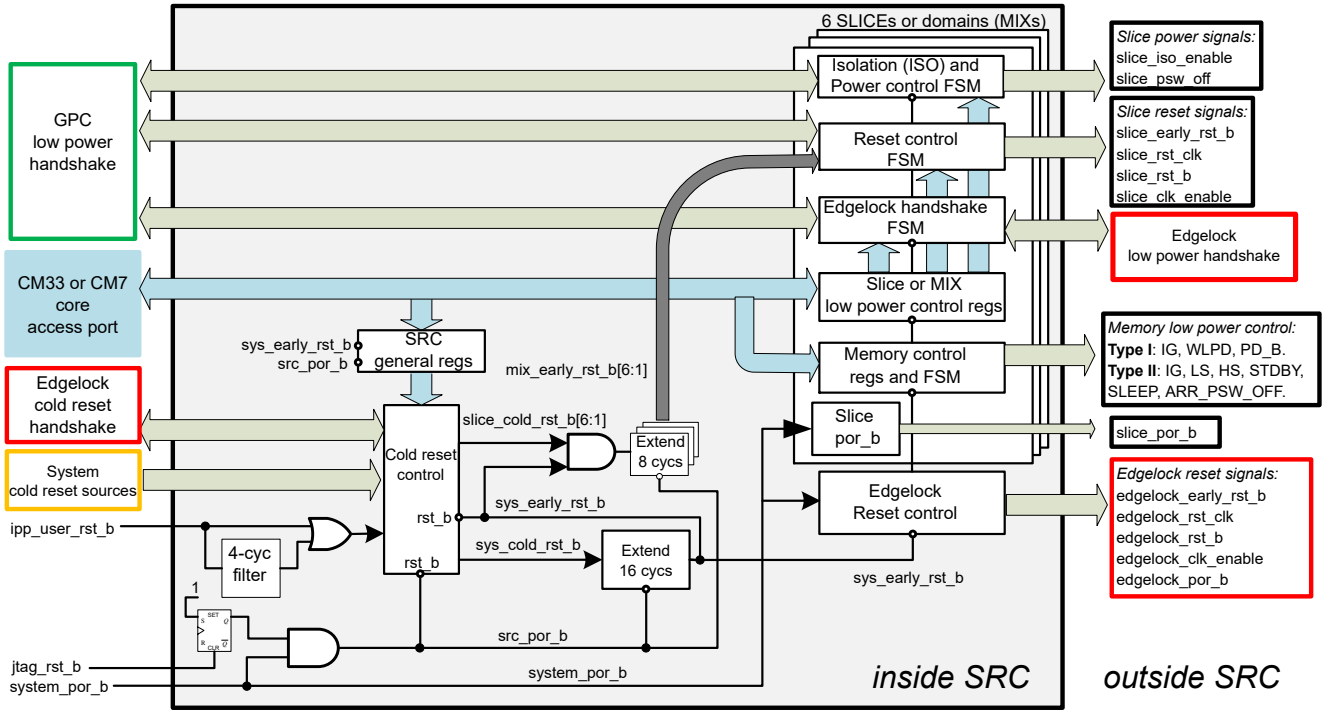


Figure 96. SRC block diagram

27.2.2 Features

SRC includes the following features:

- Receives and handles global system reset sources from other modules
- Saves the active reset source flag into the reset status register, in both DCDC and BBSM power domains
- Latches the IPP_BOOT_MODE pins from the chip top pads
- Automatic power domain control (MIXs power off/on), based on GPC's low power request
- Software power domain control (MIXs power off/on)
- Programmable registers to adjust low power sequence timing
- Programmable memory power domain control (MIXs memory power off/on and low power mode)

27.3 Functional description

The reset control determines the source and type of reset, and performs the necessary reset qualification and stretching sequences. Based on the type of reset, the reset logic generates the reset sequence for the entire chip.

The boot argument latching includes the IPP_BOOT_MODE pins latch, when the system comes out of reset.

27.3.1 Reset and Power Scheme

The chip presumes the following system reset and power schemes. See the power management chapter for more details on the specific power sources, controls, and domains.

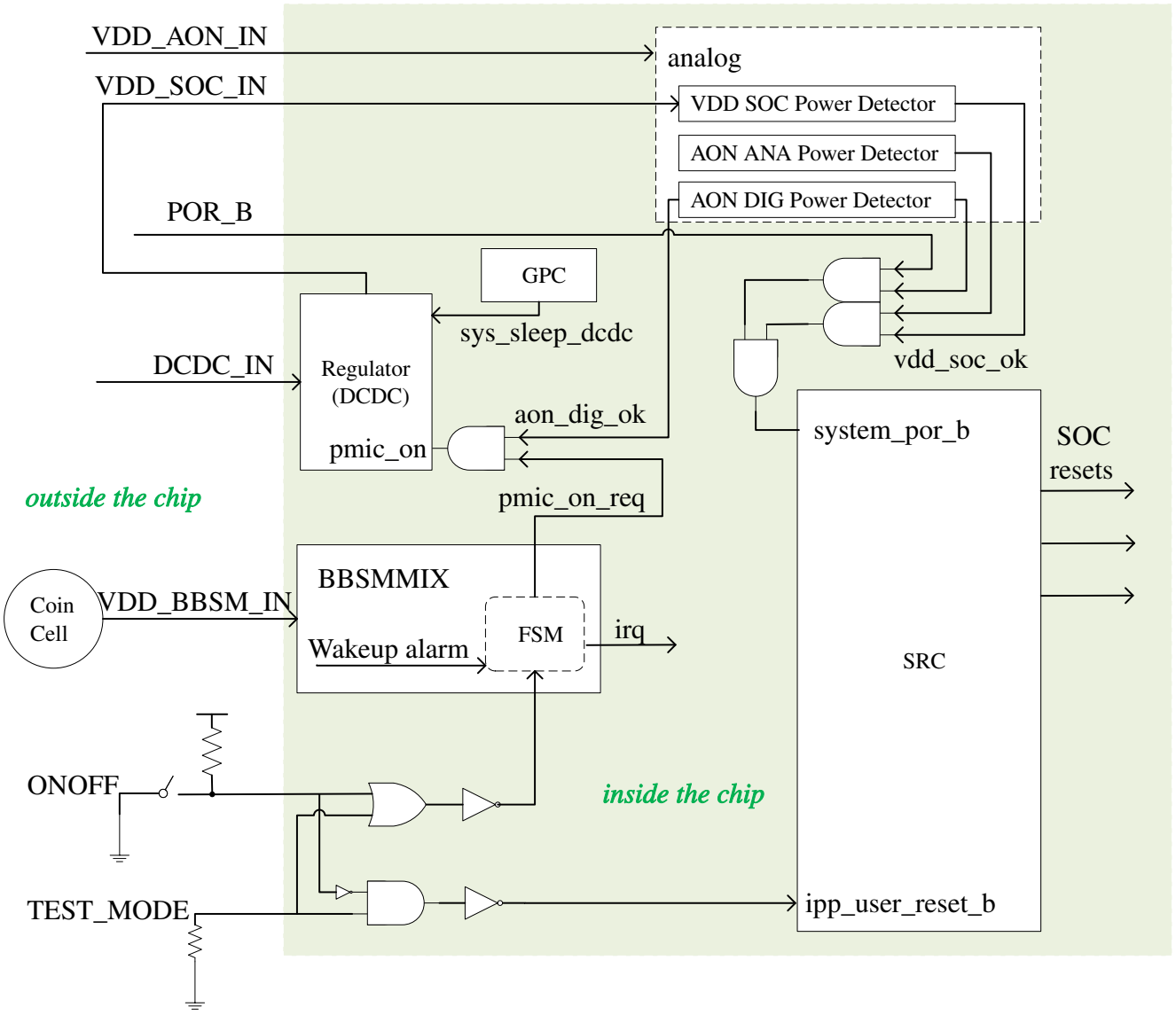


Figure 97. Chip Reset with on-chip DCDC

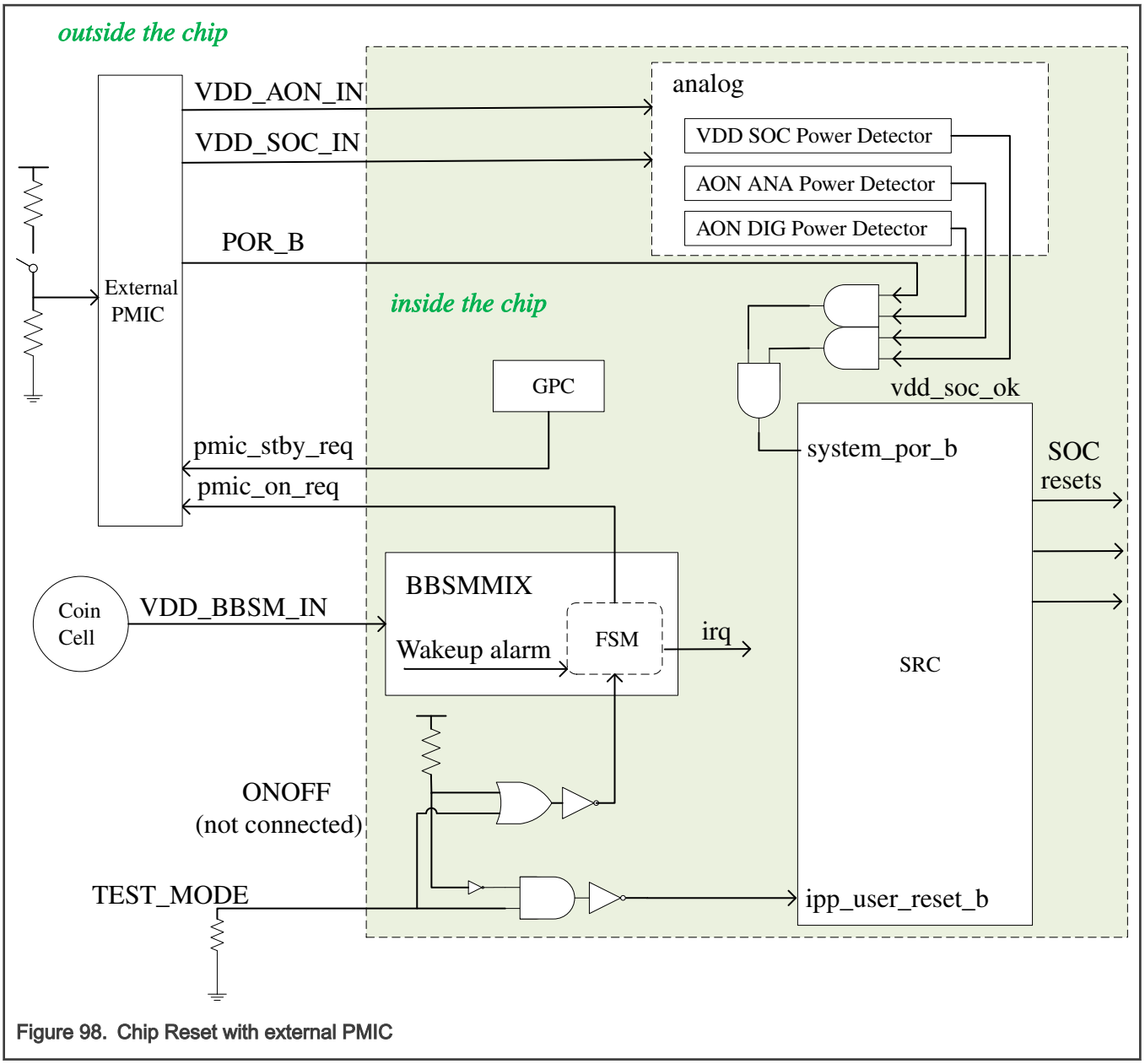


Figure 98. Chip Reset with external PMIC

27.3.2 Finite State Machine (FSM)

The FSM operation of SRC is illustrated below. The details of the states are explained in the table as well.

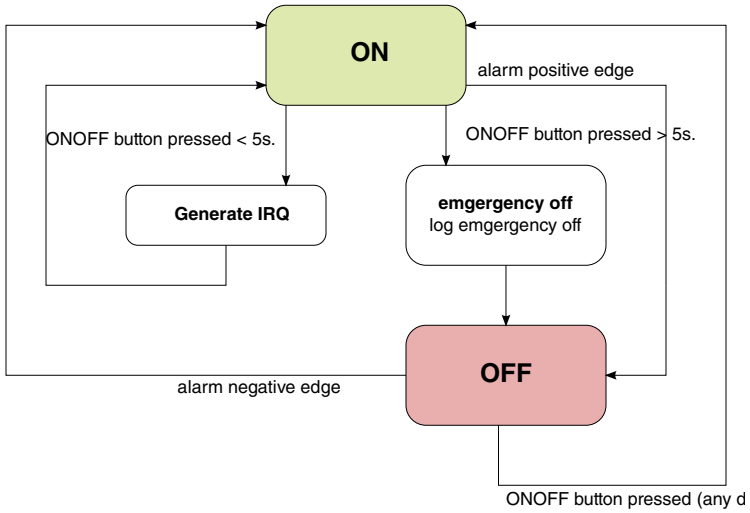


Figure 99. FSM Operation

The following table describes the ON-OFF state mechanism:

State	Configuration with external PMIC	Configuration with on-chip DCDC
ON, first time	1. BBSM is powered, PMIC_ON_REQ goes to logic '1'. 2. When PMIC button is pressed, PMIC powers on.	1. BBSM is powered, PMIC_ON_REQ goes to logic '1'. 2. VDD_AON_IN and DCDC_IN are powered. 3. Drive DCDC_PSWITCH to high, 1ms after DCDC_IN is powered. 4. DCDC regulator is enabled.
Normal ON to OFF, by button (PMIC or ONOFF)	1. PMIC button is pressed (short duration) on the external PMIC. 2. IRQ is sent to SoC from the external PMIC. 3. SoC is programing PMIC for power off when PMIC_STBY_REQ is asserted. 4. In low-power mode, the PMIC_STBY_REQ is asserted according to the GPC configuration. PMIC gates SoC supplies.	1. ONOFF button is pressed (short duration, less than 5s). 2. IRQ is sent to SoC from FSM. 3. Alarm timer is set by software routine and started. 4. Upon wakeup alarm assertion to logic '1', PMIC_ON_REQ goes to logic '0'. 5. DCDC regulator goes OFF.
Emergency On to OFF, by button (PMIC or ONOFF)	1. PMIC button is pressed long way on the external PMIC. 2. PMIC is powering off.	1. ONOFF button is pressed for longer than 5s on the SoC. 2. FSM validates ONOFF button pressed for 5s.

Table continues on the next page...

Table continued from the previous page...

State	Configuration with external PMIC	Configuration with on-chip DCDC
		3. Emergency power off is logged, PMIC_ON_REQ goes to logic '0'. 4. DCDC regulator goes OFF.
OFF to ON, by button (PMIC or ONOFF)	1. PMIC button is pressed on the external PMIC. 2. PMIC is powering on.	1. ONOFF button is pressed on the SoC. 2. PMIC_ON_REQ goes to logic '1'. 3. DCDC regulator powers ON.
OFF to ON, by timer alarm	1. Timer alarm in BBSM is programmed by software before SoC goes OFF. 2. SoC enters OFF mode. 3. Upon timer limit, wake up alarm goes to logic '0'. PMIC_ON_REQ goes to logic '1'. 4. PMIC receives assertion of PMIC_ON_REQ and wakes up.	1. Timer alarm in BBSM is programmed by software before SoC goes OFF. 2. SoC enters OFF mode. 3. Upon timer limit, wakeup alarm goes to logic '0'. PMIC_ON_REQ goes to logic '1'. 4. DCDC regulator is enabled by PMIC_ON_REQ=1.

27.3.3 Reset Control

This section describes the reset control of this device. Any of the following sources can cause a reset event:

- power-on reset (POR),
- global system reset request,
- low-power mode transition.

27.3.3.1 Reset behavior of POR

POR reset occurs when any of the following conditions is met. It resets the whole chip, except the BBSM domain, as shown in the "Power Architecture" diagram (see the "Clock and Power Overview" chapter).

- VDD_AON_IN is not supplied (**Note**: BBSM power supply must be earlier than VDD_AON_IN, or at least at the same time.)
- VDD_SOC_IN power signal not OK, indicated by the chip internal analog power detector
- LDO_AON_DIG output power signal not OK, indicated by the chip internal analog power detector
- LDO_AON_ANA output power signal not OK, indicated by the chip internal analog power detector
- Chip POR_B pad = 0
- Chip internal module JTAGC output signal jtag_reset_b = 0 (**Note**: this is not applicable in the user mode.)

NOTE

The boot argument latching is the IPP_BOOT_MODE value latching. The value of IPP_BOOT_MODE[5:3] is tied to 3'b100. And the value of IPP_BOOT_MODE[2:0] comes from the chip top pads GPIO_AON_02, GPIO_AON_01 and GPIO_AON_00. The value on chip top pads is latched to IPP_BOOT_MODE[2:0] when the chip comes out of POR reset one 32kHz clock cycle later. And the latched value cannot be changed by global system reset or slice reset.

27.3.3.2 Reset behavior of the global system reset request

As shown in the following table, global system reset sources can trigger system reset or not:

- Global system reset request can reset the whole chip, except BBSM domain, CM33 debug and trace, CM7 debug and trace, IOMUXC, reset control logic and GPR in SRC.
- Global system reset source can be masked by setting relative bits in SRC RESET MASK register. (By default, only Edgelock's global system reset source can trigger global system reset, while other reset sources are masked.)

Table 179. Global System Reset Sources

Reset sources	Description
ecat_rsto	Asserts when EtherCat outputs a reset request
dcdc_ovvt	Asserts when DCDC output power 1.0V (over 1.5V) or 1.8V (over 2.5V) voltage is too high
cm7_req_rst	Asserts when software writes SYSRESETREQ bit of register APPLICATION_INTERRUPT_CONTROL_REGISTER in CM7
cm7_lockup_rst	Asserts when CM7 lock up
cm33_req_rst	Asserts when software writes SYSRESETREQ bit of register APPLICATION_INTERRUPT_CONTROL_REGISTER in CM33
cm33_lockup_rst	Asserts when CM33 lock up
jtag_sw_b	Asserts when the debugger asserts software reset by JTAG
edgelock_rst	Asserts when Edgelock secure violation happens
tempsense_rst	Asserts when the Temper Sensor requests a reset
wdog*n*_rst	Asserts when the peripheral WDOG*n* timeouts

27.3.3.3 Reset behavior under the low-power mode transition

The reset step is one of the four steps in the low-power mode transition. After the MIX goes through power off and power on, the MIX is reset (all registers in the MIX also reset). See the following section "low-power mode transition", for more details.

NOTE

When CM7 Platform is reset, its debug and trace logic will not be reset. Only the POR reset can reset this debug and trace logic.

27.3.4 Low-power mode transition

Among the several MIXs, MEGA MIX, NETC MIX and CM7P MIX can be powered off. A fixed power off/on sequence is adopted in this chip.

The fixed power-off sequence is as follows:

1. Edgelock handshake for power-off
2. isolation asserted
3. reset asserted
4. power off MIXs or its memory

And the fixed power-on sequence is as follows:

1. power on MIXs or its memory
2. reset released
3. isolation released
4. Edgelock handshake for power-on

27.3.4.1 Control of MIX power-off

There are two ways to power off a MIX.

- GPC mode (by setting AUTHEN_CTRL[LPM_MODE] = 1)
- software (SW) mode (by setting AUTHEN_CTRL[LPM_MODE] = 0, If user fully knows how to control the power-off sequence)

NOTE

Do not change AUTHEN_CTRL[LPM_MODE] value, when the MIX is in power off/on sequence or is already in power off state.

In the GPC mode, SRC powers off the MIX under the control of GPC.

In the chip, each Core can be mapped to any of the 16 domains by TRDC. For example, CM33 Core can be mapped to both domain2 and domain15. In SRC, each MIX can be assigned to any of the 16 domains via AUTHEN_CTRL[WHITE_LIST] bits. When the bit is set, the domain's mapped Core's power state transition is considered as a low power trigger, else the transition is ignored. In each MIX, there is 16 LPM_SETTING_Dx in LPM_SETTING_1 and LPM_SETTING_2, via this software mode, configuring LPM_SETTING_Dx to select what Core power state triggers the MIX's power off/on request.

Because a MIX can be assigned to 16 domains at most, a unified arbitration rule is defined for each MIX.

- The MIX enters power-off sequence with the last active Core power state transition, if several cores (of MIX's assigned domains' mapping Cores) request the MIX to power off.
- The MIX enters power-on sequence with the first active Core power state transition, if any one core (of MIX's assigned domains' mapping Cores) requests the MIX to power on while the MIX is in power-off status.

For example, CM7P MIX can be assigned to the CM7 Core. After CM7 core requests MIX to power off/on, CM7P MIX enters power off/on sequence with CM7 Core power state transition. For comparison, MEGA MIX can be assigned to both CM33 and CM7 Cores. If only one Core requests the MIX to power off, MEGA MIX keeps in power on status. If CM33 requests the MIX to power off, and CM7 requests the MIX to power off later, then MEGAMIX enters power-off sequence with CM7 Core's power state transition. When MEGA MIX is already in power-off status, after any Core requests the MIX to power on, MEGA MIX enters power-on sequence with the Core's power state transition.

NOTE

If the MIX enters power-off/on sequence, it goes through all the four steps[1] and cannot be interrupted. So during the power-off sequence, any Core's power-on request has to be delayed until the completion of MIX's power-off sequence. During the power-on sequence, any Core's power-off request has to be delayed until the completion of MIX's power-on sequence.

27.3.4.2 MIX power-off and power-on sequence

After the initial power-up of the entire chip, all MIXs in the chip are powered on, unless the chip's CM7P MIX or NETC MIX is fused out. So each MIX must be firstly powered off before it can be powered on.

The power-off sequence of MEGA MIX, NETC MIX and CM7P MIX is as follows:

1. CPU WFI.
2. GPC sends the domain's power state change request and the Core's current low power state. SRC controls the MIX's low power sequence based on its AUTHEN_CTRL[LPM_MODE] , AUTHEN_CTRL[WHITE_LIST] and LPM_SETTING_x values.

[1] See the "Low-power mode transition" section.

- if GPC mode is adopted, then go to the next step.
- If SW mode is adopted, software can use the UPI status to check GPC's *req*. Also the software can use Slice software control to begin the whole low power sequence by one bit, or begin some or one step of the sequence.

3. GPC sends *stop clock req* to CCM, to halt the clock.
4. GPC sends *Edgelock handshake req* to SRC, to implement handshake with Edgelock.
5. GPC sends *ISO assert req* to SRC, to isolate the MIX intended to power off.
6. GPC sends *reset assert req* to SRC, to assert reset to the MIX intended to power off.
7. GPC sends *power off req* to SRC, to power off the intended MIX.

The power-on sequence of MEGA MIX, NETC MIX and CM7P MIX is as follows:

1. CPU gets interrupt to wake up from low power state.
2. GPC sends the domain's power state transition request and the Core's low power state of current domain. SRC controls the MIX's low power sequence based on its AUTHEN_CTRL[LPM_MODE] , AUTHEN_CTRL[WHITE_LIST] and LPM_SETTING_x values.
   - if GPC mode is adopted, then go to the next step.
   - if SW mode is adopted, software can use the UPI status to check GPC's *req*. Also the software can use Slice software control to begin the whole low power sequence by one bit, or begin some or one step of the sequence.
3. GPC sends *power on req* to SRC, to power on the intended MIX.
4. GPC sends *reset release req* to SRC, to release reset to the MIX intended to power on.
5. GPC sends *ISO release req* to SRC, to release isolation of the MIX intended to power on
6. GPC sends *Edgelock handshake req* to SRC, to implement handshake with Edgelock.
7. GPC sends *enable clock req* to CCM, to resume the clock.

Edgelock handshake: EDGELOCK_HDSK_CTRL and EDGELOCK_HDSK_CNT_CFG can be used to control the Edgelock handshake step. EDGELOCK_HDSK_CTRL[POFF_CNT_MODE] and EDGELOCK_HDSK_CTRL[PON_CNT_MODE] can control how *ack* affects the FSM of Edgelock handshake. FSM can wait *ack* or ignore *ack*, or use internal delay counter.

Isolation (ISO) and power switch (PSW): ISO_DLY_PRE can control the delay from ISO change request to the actual ISO change on the MIX. PSW_DLY_PRE_HF/LF can control the delay from power change request to the actual power change on the MIX. Also PSW_CTRL and PSW_CNT_CFG_HF/LF can control how *ack* affects the FSM of power change. FSM can wait *ack* or ignore *ack*, or use internal delay counter.

The following timing diagram shows an example in which a power-off sequence is seen, with UPI handshake to GPC.

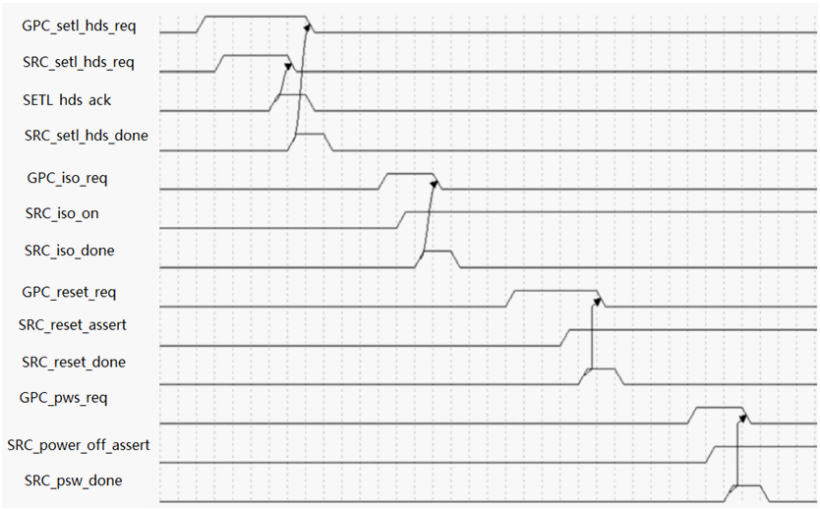


Figure 100. Power-off sequence, with UPI handshake to GPC

27.3.5 Memory low power control

Each MIX contains two kinds of memory type: MEM type I and MEM type II. Because their low power control signals are different, SRC has two types of controllers designed accordingly.

- For MEM type I, the signals (HS, LS, IG, STDBY, SLEEP and ARR_PDN) to control low power mode. [2]
- For MEM type II, the signals (IG, PD_B and WLPD) to control low power mode.

User can control the memory low power mode in either HW way or SW way. When the CM7P MIX enters power-off sequence, the memory in the MIX can change its memory power state automatically, which is the HW way (For MEGA MIX and NETC MIX, because its MIX and memory use the same power switch, its memory is powered off/on with the MIX's power off/on). Also there is a SW way to change memory power state. MEM_LP_TRIG_CTRL register selects either way.

There are two memory low power controllers in each MIX: one for cache and the other for local TCM. If there is no cache and local TCM in the MIX, then the former one controls MEM type I and the latter controls MEM type II.

MIX	Cache memory	TCM memory
AON MIX	MEM Type I	MEM Type II
WAKEUP MIX	MEM Type I	MEM Type II
MEGA MIX	MEM Type I	MEM Type II
NETC MIX	MEM Type I	MEM Type II
CM33P MIX	MEM Type I	MEM Type I
CM7P MIX	MEM Type I	MEM Type I with PSW

In HW way, set one of 8 levels of low power depth for each signal before the low power sequence, via the MEM_LP_CFG register, and let them take effect when SRC low power sequence starts. There are 8 low power levels, as shown in the following table.

In SW way, users can make their own strategy by changing values of the MPL_CTRL bits in MEM Type I and II. No matter a MIX enters power off/on sequence or a software trigger happens, the level value is always used to select one of 8 values in the MPL_CTRL bits in MEM Type I and II, and then to change the memory low power control signals accordingly.

[2] See the "Memory map and register definition" section for detailed descriptions of control signals.

Table 180. Reset value of memory low power control

Signal	Level7	Level6	Level5	Level4	Level3	Level2	Level1	Level0
LS	0	0	0	0	0	0	0	0
HS	0	0	0	0	0	0	**1**	0
IG	0	0	0	**1**	0	0	0	0
STDBY	0	0	0	0	0	0	0	0
SLEEP[1]	**1**	**1**	0	0	0	0	0	0
ARR_PDN[1]	**1**	**1**	0	0	0	0	0	0
IG	0	0	0	**1**	0	0	0	0
WLPD	0	0	0	0	0	**1**	**1**	0
PD_B	0	0	0	**1**	**1**	0	**1**	**1**

1. only for CM7 TCM

27.3.5.1 CM7 TCM content retention

The memory content usually gets lost after the memory is powered off. CM7 TCM memory can still be alive when CM7P MIX is powered off. So its memory content can be retained in this case.

In HW mode, change the value of MEM_LP_CFG[MLPL_HW_PDN_LMEM] to 0x0~0x5 (**not** the default value 0x7) in CM7PLATFORM_MIX_SLICE, because the value 0x6 or 0x7 means CM7 TCM memory is powered off with CM7P MIX power-off together.

In SW mode (if you want to control memory power by programing), it has to be **noted** there are two low power control signals: SLEEP and ARR_PDN. SLEEP is an isolation signal between the memory array power domain and the CM7P MIX power domain. SLEEP must be asserted before ARR_PDN changes to 1, and also kept asserted, until both power domains are powered on.

- In the memory array power-on and CM7P MIX power-off case, SLEEP signal can be asserted high automatically, and there is no need for software intervention.
- In the memory array power-off and CM7P MIX power-on case, SLEEP signal must be asserted ahead via setting MIF_MPL_SLEEP[SW_SLEEP] by software, and then assert memory array power off signal MIF_MPL_ARR_PDN[SW_ARR_PDN]. If later the software needs to power on CM7 TCM, it must clear MIF_MPL_ARR_PDN[SW_ARR_PDN] at first to power on the memory array, and then clear MIF_MPL_SLEEP[SW_SLEEP] to deassert the SLEEP signal.

27.3.6 Clocking

The following table describes the relevant SRC clock sources.

Table 181. SRC Clocks

Clock name	Description
ipg_clk	Peripheral clock
ipg_clk_s	Peripheral access clock. This clock is typically the same source as the ipg_clk.

NOTE

See the clock controller chapter for clock setting, configuration, and gating information.

27.4 External Signals

The following table describes the SRC external signals.

Table 182. External Signals

Signal	Description	Direction
POR_B	Power on reset input pin, active low.	I
IPP_BOOT_MODE[2:0] {GPIO_AON_02, GPIO_AON_01, GPIO_AON_00}	Boot configuration pins. The values are sampled during system reset and can be read in the SBMR1 register. See the boot chapter for specific boot configuration settings, and the pin mux chapter for pin assignments.	I

27.5 Initialization

27.5.1 Notes of power sequencing in POR

The following conditions must be met for power-up:

- VDD_BBSM_IN supply must be turned on before any other power supply, or be connected (shorted) with DCDC_IN supply.
- If a coin cell is used to power VDD_BBSM_IN, then ensure that it is connected before any other supply is switched on.
- When the internal DCDC is enabled, DCDC_IN and VDD_AON_IN must be supplied. And an external delay circuit is required to delay the DCDC_PSWITCH signal at least 1ms after DCDC_IN is stable.
- User can choose not using POR_B and just tie it to high. Else if it is expected to release the chip by POR_B signal, the POR_B input must be immediately asserted at power-up, and kept asserted until the last power rail reaches its working voltage.

Similarly, for power-down, the following conditions must be met:

- VDD_BBSM_IN supply must be turned off after any other power supply, or be connected (shorted) with DCDC_IN supply.
- If a coin cell is used to power VDD_BBSM_IN, then ensure that it is removed after any other supply is switched off.

27.6 Memory map and register definition

The SRC registers can be separated into three groups:

1. General registers, which control reset sources from other modules to generate global system reset or not.
2. MIX registers, which control MIX power off/on sequence.
3. MIX memory registers, which control MIXs memory low power functions.

NOTE

For SRC register access, the register AUTHEN_CTRL is used to control access permission. The general registers and MIX registers have their own AUTHEN_CTRL to control access permission, non-secure or secure access, and lock configuration for each MIX.

27.6.1 SRC General register descriptions

27.6.1.1 SRC general registers memory map

SRC_GENERAL_REG base address: 4446_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	Authentication Control (AUTHEN_CTRL)	32	RW	FFFF_0000h
10h	SRC Control Register (SCR)	32	RW	0000_0000h
14h	SRC Reset Trigger Mode Register (SRTMR)	32	RW	0000_0000h
18h	SRC Reset Mask Register (SRMASK)	32	RW	0000_3FBFh
40h	SRC Boot Mode Register 1 (SBMR1)	32	R	0000_0000h
44h	SRC Boot Mode Register 2 (SBMR2)	32	R	0800_0000h
4Ch	SRC Reset Status Register backup in BBSM domain (SRSR_BBSM)	32	RW	0000_0001h
50h	SRC Reset Status Register (SRSR)	32	RW	0000_0001h
54h - A0h	SRC General Purpose Register (GPR0 - GPR19)	32	RW	0000_0000h

27.6.1.2 Authentication Control (AUTHEN_CTRL)

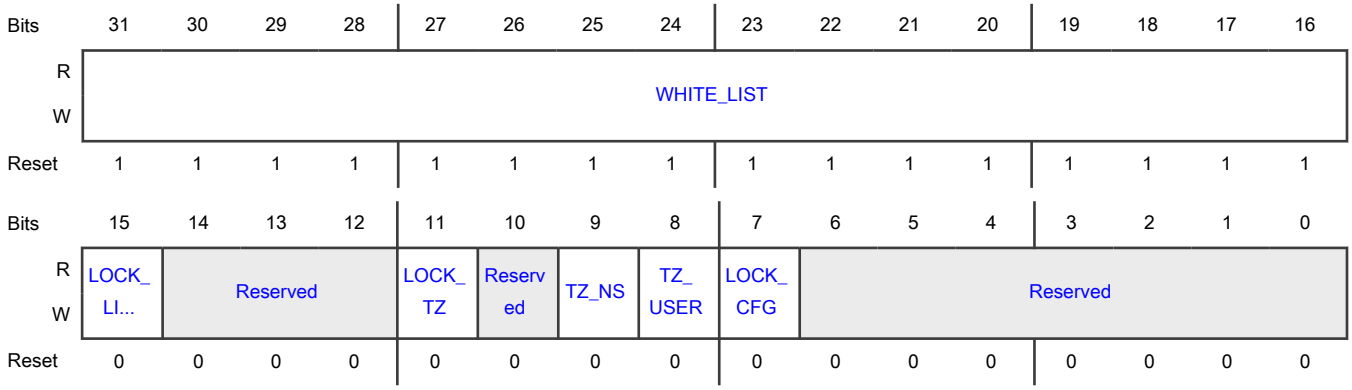
Offset

Register	Offset
AUTHEN_CTRL	4h

Function

Authentication Control. Reset by global system reset.

Diagram



Fields

Field	Function
31-16	Domain ID white list

Table continues on the next page...

Table continued from the previous page...

Field	Function
WHITE_LIST	Each master (Core or non-Core) can be assigned to any domains among domain0~domain15 by TRDC. Once its domains are assigned, then its accesses will have its unique domain ID. Each bit in this field represents for one domain. Bit16~bit31 represent for domain0~domain15 respectively. Only the domains whose corresponding bits are 1 can write the General registers. Write violation will fail without any exception or information. Read violation can still get read data. 0000_0000_0000_0001b - Core with domain ID=0 can write General registers. 0000_0000_0000_0010b - Core with domain ID=1 can write General registers. 0000_0000_0000_0100b - Core with domain ID=2 can write General registers. 0000_0000_0000_1000b - Core with domain ID=3 can write General registers. 0000_0000_0001_0000b - Core with domain ID=4 can write General registers. 0000_0000_0010_0000b - Core with domain ID=5 can write General registers. 0000_0000_0100_0000b - Core with domain ID=6 can write General registers. 0000_0000_1000_0000b - Core with domain ID=7 can write General registers. 0000_0001_0000_0000b - Core with domain ID=8 can write General registers. 0000_0010_0000_0000b - Core with domain ID=9 can write General registers. 0000_0100_0000_0000b - Core with domain ID=10 can write General registers. 0000_1000_0000_0000b - Core with domain ID=11 can write General registers. 0001_0000_0000_0000b - Core with domain ID=12 can write General registers. 0010_0000_0000_0000b - Core with domain ID=13 can write General registers. 0100_0000_0000_0000b - Core with domain ID=14 can write General registers. 1000_0000_0000_0000b - Core with domain ID=15 can write General registers.
15 LOCK_LIST	White list lock This field can lock itself and WHITE_LIST. 0b - WHITE_LIST value can be changed. 1b - LOCK_LIST and WHITE_LIST values cannot be changed.
14-12 —	Reserved
11 LOCK_TZ	Lock Trust Zone Non Secure(TZ_NS) and Trust Zone User(TZ_USER) bits This field can lock itself, TZ_NS and TZ_USER. 0b - TZ_NS and TZ_USER values can be changed. 1b - LOCK_TZ, TZ_NS and TZ_USER values cannot be changed.
10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 TZ_NS	Allow non-secure mode access Allows either secure mode or non-secure mode to write general registers. 0b - General registers can only be written in secure mode. 1b - General registers can be written either in secure mode or non-secure mode.
8 TZ_USER	Allow user mode write Allows either privilege mode or user mode to write general registers. 0b - General registers can only be written in privilege mode. 1b - General registers can be written either in privilege mode or user mode.
7 LOCK_CFG	Configuration lock This field can lock itself and some registers in the General registers: i.e. SRTMR and SRMASK. 0b - General registers are not locked. 1b - LOCK_CFG and registers in the list are locked.
6-0 —	Reserved

27.6.1.3 SRC Control Register (SCR)

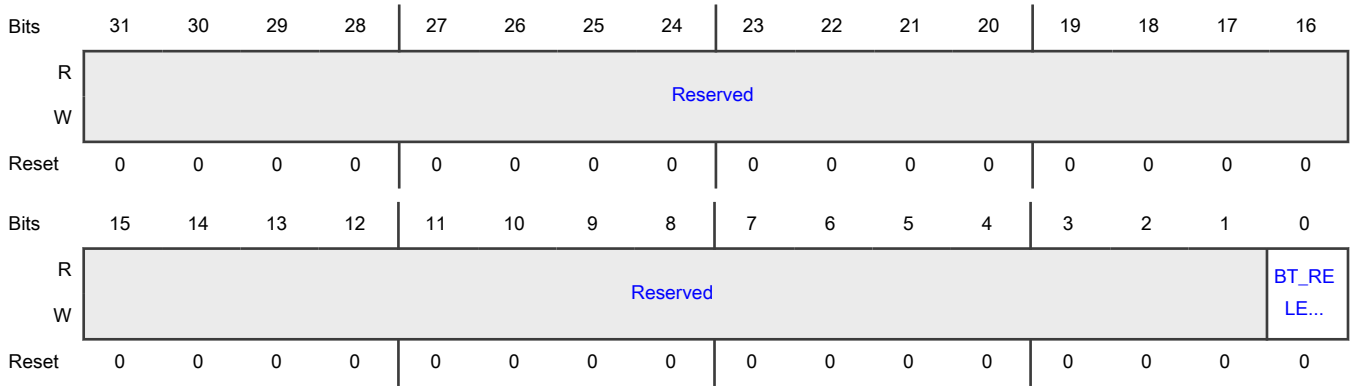
Offset

Register	Offset
SCR	10h

Function

Contains bits that control operation of the reset controller. Reset by global system reset.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 BT_RELEASE_M7	Boot release M7 0b - Holds M7 Core reset. 1b - Releases M7 Core reset and let it run. After this bit is set, it cannot be cleared by SW write.

27.6.1.4 SRC Reset Trigger Mode Register (SRTMR)

Offset

Register	Offset
SRTMR	14h

Function
Reset by POR not global system reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		ECAT_RS...	DCDC_OV...	CM7_LOC...	CM7_RES...	CM33_LO...	CM33_RE...	JTAG SW_...	EDGE LOC...	TEMP SEN...	WDOG5_T...	WDOG4_T...	WDOG3_T...	WDOG2_T...	WDOG1_T...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved
13 ECAT_RSTO_TRIG_MODE	ECAT reset output mode configuration, locked by LOCK_CFG field 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 DCDC_OVVT_TRIG_MODE	DCDC over voltage trigger mode configuration, locked by LOCK_CFG field 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
11 CM7_LOCKUP_TRIG_MODE	CM7 lockup trigger mode configuration, locked by LOCK_CFG field 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
10 CM7_RESET_TRIG_MODE	CM7 reset trigger mode configuration, locked by LOCK_CFG field 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
9 CM33_LOCKUP_TRIG_MODE	CM33 lockup trigger mode configuration, locked by LOCK_CFG field. 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
8 CM33_RESET_TRIG_MODE	CM33 reset trigger mode configuration, locked by LOCK_CFG field. 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
7 JTAGSW_TRIG_MODE	Jtagsw reset trigger mode configuration, locked by LOCK_CFG field 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
6 EDGELOCK_TRIG_MODE	Edgelock reset trigger mode configuration, locked by LOCK_CFG field 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
5 TEMPSENSE_TRIG_MODE	TempSense reset trigger mode configuration, locked by LOCK_CFG field 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
4 WDOG5_TRIG_MODE	Wdog5 reset trigger mode configuration, locked by LOCK_CFG field 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
3 WDOG4_TRIG_MODE	Wdog4 reset trigger mode configuration, locked by LOCK_CFG field 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
2	Wdog3 reset trigger mode configuration, locked by LOCK_CFG field

Table continues on the next page...

Table continued from the previous page...

Field	Function
WDOG3_TRIG_MODE	0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
1 WDOG2_TRIG_MODE	Wdog2 reset trigger mode configuration, locked by LOCK_CFG field 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.
0 WDOG1_TRIG_MODE	Wdog1 reset trigger mode configuration, locked by LOCK_CFG field 0b - Level-sensitive: System stays in reset until the reset source deasserts. 1b - Edge-sensitive: System resets once, even if the reset source remains asserted.

27.6.1.5 SRC Reset Mask Register (SRMASK)

Offset

Register	Offset
SRMASK	18h

Function

Reset by POR not global system reset

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		ECAT_RS...	DCDC_OV...	CM7_LOC...	CM7_RES...	CM33_LO...	CM33_RE...	JTAG SW_...	EDGE LOC...	TEMP SEN...	WDOG5_M...	WDOG4_M...	WDOG3_M...	WDOG2_M...	WDOG1_M...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		ECAT_RS...	DCDC_OV...	CM7_LOC...	CM7_RES...	CM33_LO...	CM33_RE...	JTAG SW_...	EDGE LOC...	TEMP SEN...	WDOG5_M...	WDOG4_M...	WDOG3_M...	WDOG2_M...	WDOG1_M...
Reset	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1

Fields

Field	Function
31-30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 ECAT_RSTO_MASK_LOCKED	Lock ECAT_RSTO_MASK Locks ECAT_RSTO_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - ECAT_RSTO_MASK's value can be changed. 1b - This bit and ECAT_RSTO_MASK's value cannot be changed.
28 DCDC_OVVT_MASK_LOCKED	Lock DCDC_OVVT_MASK Locks DCDC_OVVT_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - DCDC_OVVT_MASK's value can be changed. 1b - This bit and DCDC_OVVT_MASK's value cannot be changed.
27 CM7_LOCKUP_MASK_LOCKED	Lock CM7_LOCKUP_MASK Locks CM7_LOCKUP_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - CM7_LOCKUP_MASK's value can be changed. 1b - This bit and CM7_LOCKUP_MASK's value cannot be changed.
26 CM7_RESET_MASK_LOCKED	Lock CM7 reset mask bit Locks CM7_RESET_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - CM7_RESET_MASK's value can be changed. 1b - This bit and CM7_RESET_MASK's value cannot be changed.
25 CM33_LOCKUP_MASK_LOCKED	Lock CM33_LOCKUP_MASK Locks CM33_LOCKUP_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - CM33_LOCKUP_MASK's value can be changed. 1b - This bit and CM33_LOCKUP_MASK's value cannot be changed.
24 CM33_RESET_MASK_LOCKED	Lock CM33_RESET_MASK Locks CM33_RESET_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - CM33_RESET_MASK's value can be changed. 1b - This bit and CM33_RESET_MASK's value cannot be changed.
23 JTAGSW_MASK_LOCKED	Lock JTAGSW_MASK Locks JTAGSW_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - JTAGSW_MASK's value can be changed. 1b - This bit and JTAGSW_MASK's value cannot be changed.
22 EDGELOCK_MASK_LOCKED	Lock EDGELOCK_MASK Locks EDGELOCK_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - EDGELOCK_MASK's value can be changed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - This bit and EDGELOCK_MASK's value cannot be changed.
21 TEMPSENSE_MASK_LOCKED	Lock TEMPSENSE_MASK Locks TEMPSENSE_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - TEMPSENSE_MASK's value can be changed. 1b - This bit and TEMPSENSE_MASK's value cannot be changed.
20 WDOG5_MASK_LOCKED	Lock WDOG5_MASK Locks WDOG5_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - This bit and WDOG5_MASK's value can be changed. 1b - This bit and WDOG5_MASK's value cannot be changed.
19 WDOG4_MASK_LOCKED	Lock WDOG4_MASK Locks WDOG4_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - This bit and WDOG4_MASK's value can be changed. 1b - This bit and WDOG4_MASK's value cannot be changed.
18 WDOG3_MASK_LOCKED	Lock WDOG3_MASK Locks WDOG3_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - This bit and WDOG3_MASK's value can be changed. 1b - This bit and WDOG3_MASK's value cannot be changed.
17 WDOG2_MASK_LOCKED	Lock WDOG2_MASK Locks WDOG2_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - This bit and WDOG2_MASK's value can be changed. 1b - This bit and WDOG2_MASK's value cannot be changed.
16 WDOG1_MASK_LOCKED	Lock WDOG1_MASK Locks WDOG1_MASK's value. This bit can be locked by itself or AUTHEN_CTRL[LOCK_CFG]. 0b - This bit and WDOG1_MASK's value can be changed. 1b - This bit and WDOG1_MASK's value cannot be changed.
15-14 —	Reserved
13 ECAT_RSTO_MASK	ECAT reset output mask Masks ECAT reset output source. This bit can be locked by ECAT_RSTO_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - The reset source is masked, cannot work
12 DCDC_OVVT_MASK	DCDC over voltage mask Masks DCDC over voltage reset source. This bit can be locked by DCDC_OVVT_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
11 CM7_LOCKUP_MASK	CM7 lockup reset mask Masks CM7 lockup reset source. This bit can be locked by CM7_LOCKUP_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
10 CM7_RESET_MASK	CM7 reset mask This bit can mask CM7 reset source. This bit can be locked by CM7_RESET_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
9 CM33_LOCKUP_MASK	CM33 lockup mask Masks CM33 lockup reset source. This bit can be locked by CM33_LOCKUP_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
8 CM33_RESET_MASK	CM33 reset mask Masks CM33 reset source. This bit can be locked by CM33_RESET_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
7 JTAGSW_MASK	JTAGSW reset mask Masks JTAG software reset source. This bit can be locked by JTAGSW_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
6	Edgelock reset mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
EDGELOCK_MASK	Masks Edgelock reset source. This bit can be locked by EDGELOCK_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
5 TEMPSENSE_MASK	TempSense reset mask Masks TempSense reset source. This bit can be locked by TEMPSENSE_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
4 WDOG5_MASK	WDOG5 reset mask Masks WDOG5 reset source. This bit can be locked by WDOG5_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
3 WDOG4_MASK	WDOG4 reset mask Masks WDOG4 reset source. This bit can be locked by WDOG4_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
2 WDOG3_MASK	WDOG3 reset mask Masks WDOG3 reset source. This bit can be locked by WDOG3_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
1 WDOG2_MASK	WDOG2 reset mask Masks WDOG2 reset source. This bit can be locked by WDOG2_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work
0 WDOG1_MASK	WDOG1 reset mask Masks WDOG1 reset source. This bit can be locked by WDOG1_MASK_LOCKED or AUTHEN_CTRL[LOCK_CFG]. 0b - The reset source can work 1b - The reset source is masked, cannot work

27.6.1.6 SRC Boot Mode Register 1 (SBMR1)

Offset

Register	Offset
SBMR1	40h

Function
Contains bits that reflect the status of Boot Mode Pins of the chip. Reset by POR not global system reset

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BOOT_CFG4								BOOT_CFG3							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BOOT_CFG2								BOOT_CFG1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 BOOT_CFG4	Reserved
23-16 BOOT_CFG3	Reserved
15-8 BOOT_CFG2	Reserved
7-0 BOOT_CFG1	Reserved

27.6.1.7 SRC Boot Mode Register 2 (SBMR2)

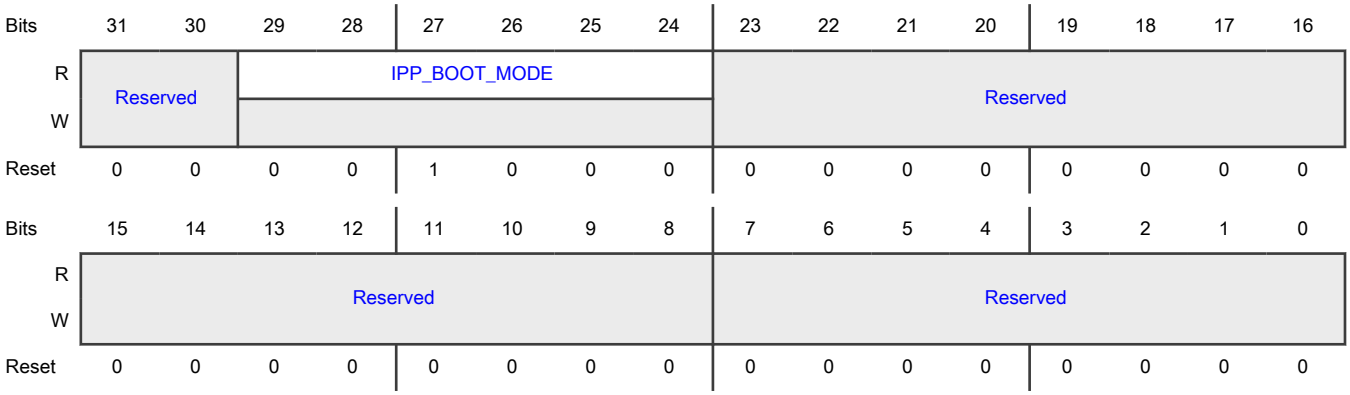
Offset

Register	Offset
SBMR2	44h

Function

Contains bits that reflect the status of Boot Mode Pins of the chip. Reset by POR not global system reset

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 IPP_BOOT_MODE	IPP_BOOT_MODE[5:4] reserved. Ties to 0. IPP_BOOT_MODE[3] ties to 1, which means CM33 boot mode selected. IPP_BOOT_MODE[2:0] value is latched from chip top PADS at 2nd 32KHz clock rising edge after POR reset. IPP_BOOT_MODE[2:0] selects boot devices. The map table for this chip is as below: 00_0000b - Boot from internal Fuses 00_0001b - Serial Downloader: USB1 or LPUART1 00_0010b - USDHC1 8-bit eMMC 5.1 00_0011b - USDHC2 4-bit SD 3.0 00_0100b - FlexSPI Serial NOR with SFDP (JESD-216) discoverable parameters 00_0101b - FlexSPI Serial NAND 2k page 00_0110b - FlexSPI Serial NAND 4k page 00_0111b - Test mode/Infinite loop mode
23-8 —	Reserved
7-0 —	Reserved

27.6.1.8 SRC Reset Status Register backup in BBSM domain (SRSR_BBSM)

Offset

Register	Offset
SRSR_BBSM	4Ch

Function

Records the source of the reset events for the chip. Writing zero does not have any effect. Writing one to the bit clears the corresponding bit to 0.

NOTE

This register is only reset by BBSM power-on reset, not POR or global system reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	ECAT_RS...	DCDC_OV...	CM7_LOC...	CM7_REQ...	CM33_LO...	CM33_RE...	JTAG_SW...	EDGELOC...	TEMPSEN...	WDOG5_R...	WDOG4_R...	WDOG3_R...	WDOG2_R...	WDOG1_R...	POR_RST
W		W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-15 —	Reserved
14 ECAT_RSTO	Indicates a reset has been caused by ECAT reset output 0b - Reset is not a result of the ECAT reset output. 1b - Reset is a result of the ECAT reset output.
13 DCDC_OVVT	Indicates a reset has been caused by DCDC over voltage 0b - Reset is not a result of the DCDC over voltage. 1b - Reset is a result of the DCDC over voltage.
12 CM7_LOCKUP	Indicates a reset has been caused by CM7 CPU 0b - Reset is not a result of the cm7 lockup.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Reset is a result of the cm7 lockup.
11 CM7_REQUEST	Indicates whether reset was the result of cm7 reset request 0b - Reset is not a result of cm7 reset request. 1b - Reset is a result of cm7 reset request.
10 CM33_LOCKUP	Indicates a reset has been caused by cm33 CPU lockup 0b - Reset is not a result of the cm33 lockup. 1b - Reset is a result of the cm33 lockup.
9 CM33_REQUEST	Indicates whether reset was the result of cm33 reset request 0b - Reset is not a result of cm33 reset request. 1b - Reset is a result of cm33 reset request.
8 JTAG_SW_RST	JTAG software reset. Indicates whether the reset was the result of software reset from JTAG. 0b - Reset is not a result of software reset from JTAG. 1b - Reset is a result of software reset from JTAG.
7 EDGELOCK_RESET_B	Indicates whether the reset was the result of the Edgelock's reset input. 0b - Reset is not a result of the Edgelock's reset event. 1b - Reset is a result of the Edgelock's reset event.
6 TEMPSENSE_RST_B	TempSensor software reset. Indicates whether the reset was the result of software reset from on-chip Temperature Sensor. 0b - Reset is not a result of software reset from Temperature Sensor. 1b - Reset is a result of software reset from Temperature Sensor.
5 WDOG5_RST_B	Time-out reset. Indicates whether the reset was the result of the watchdog5 time-out 0b - Reset is not a result of the watchdog time-out event. 1b - Reset is a result of the watchdog time-out event.
4 WDOG4_RST_B	Time-out reset. Indicates whether the reset was the result of the watchdog4 time-out 0b - Reset is not a result of the watchdog time-out event. 1b - Reset is a result of the watchdog time-out event.
3 WDOG3_RST_B	Time-out reset. Indicates whether the reset was the result of the watchdog3 time-out 0b - Reset is not a result of the watchdog3 time-out event. 1b - Reset is a result of the watchdog3 time-out event.
2 WDOG2_RST_B	Time-out reset. Indicates whether the reset was the result of the watchdog2 time-out event. 0b - Reset is not a result of the watchdog time-out event. 1b - Reset is a result of the watchdog time-out event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 WDOG1_RST_B	Time-out reset. Indicates whether the reset was the result of the watchdog1 time-out event. 0b - Reset is not a result of the watchdog time-out event. 1b - Reset is a result of the watchdog time-out event.
0 POR_RST	Indicates whether the reset was the result of power up or chip PAD POR_B. 0b - Reset is not a result of power up or chip PAD POR_B. 1b - Reset is a result of power up or chip PAD POR_B.

27.6.1.9 SRC Reset Status Register (SRSR)

Offset

Register	Offset
SRSR	50h

Function

Records the source of the reset events. Writing zero does not have any effect. Writing one clears the corresponding bit. This register is reset by POR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved									Reserved						IPP_POR...
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	ECAT_RS...	DCDC_OV...	CM7_LOC...	CM7_REQ...	CM33_LO...	CM33_RE...	JTAG_SW...	EDGELOC...	TEMPSEN...	WDOG5_R...	WDOG4_R...	WDOG3_R...	WDOG2_R...	WDOG1_R...	POR_RST
W		W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-23 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
22-17 —	Reserved
16 IPP_POR_B	Indicates whether the reset was the result of chip PAD POR_B. 0b - Reset is not a result of chip PAD POR_B. 1b - Reset is a result of chip PAD POR_B.
15 —	Reserved
14 ECAT_RSTO	Indicates a reset has been caused by ECAT reset output 0b - Reset is not a result of the ECAT reset output. 1b - Reset is a result of the ECAT reset output.
13 DCDC_OVVT	Indicates a reset has been caused by DCDC over voltage 0b - Reset is not a result of the DCDC over voltage. 1b - Reset is a result of the DCDC over voltage.
12 CM7_LOCKUP	Indicates a reset has been caused by CM7 CPU 0b - Reset is not a result of the cm7 lockup. 1b - Reset is a result of the cm7 lockup.
11 CM7_REQUEST	Indicates whether reset was the result of cm7 reset request 0b - Reset is not a result of cm7 reset request. 1b - Reset is a result of cm7 reset request.
10 CM33_LOCKUP	Indicates a reset has been caused by cm33 CPU lockup 0b - Reset is not a result of the cm33 lockup. 1b - Reset is a result of the cm33 lockup.
9 CM33_REQUEST	Indicates whether reset was the result of cm33 reset request 0b - Reset is not a result of cm33 reset request. 1b - Reset is a result of cm33 reset request.
8 JTAG_SW_RST	JTAG software reset. Indicates whether the reset was the result of software reset from JTAG. 0b - Reset is not a result of software reset from JTAG. 1b - Reset is a result of software reset from JTAG.
7 EDGELOCK_RESET_B	Indicates whether the reset was the result of the Edgelock's reset input. 0b - Reset is not a result of the Edgelock's reset event. 1b - Reset is a result of the Edgelock's reset event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 TEMPSENSE_RST_B	Temper Sensor software reset. Indicates whether the reset was the result of software reset from on-chip Temperature Sensor. 0b - Reset is not a result of software reset from Temperature Sensor. 1b - Reset is a result of software reset from Temperature Sensor.
5 WDOG5_RST_B	Time-out reset. Indicates whether the reset was the result of the watchdog5 time-out 0b - Reset is not a result of the watchdog time-out event. 1b - Reset is a result of the watchdog time-out event.
4 WDOG4_RST_B	Time-out reset. Indicates whether the reset was the result of the watchdog4 time-out 0b - Reset is not a result of the watchdog time-out event. 1b - Reset is a result of the watchdog time-out event.
3 WDOG3_RST_B	Time-out reset. Indicates whether the reset was the result of the watchdog3 time-out 0b - Reset is not a result of the watchdog3 time-out event. 1b - Reset is a result of the watchdog3 time-out event.
2 WDOG2_RST_B	Time-out reset. Indicates whether the reset was the result of the watchdog2 time-out event. 0b - Reset is not a result of the watchdog time-out event. 1b - Reset is a result of the watchdog time-out event.
1 WDOG1_RST_B	Time-out reset. Indicates whether the reset was the result of the watchdog1 time-out event. 0b - Reset is not a result of the watchdog time-out event. 1b - Reset is a result of the watchdog time-out event.
0 POR_RST	Indicates whether the reset was the result of POR. 0b - Reset is not a result of POR. 1b - Reset is a result of POR.

27.6.1.10 SRC General Purpose Register (GPR0 - GPR19)

Offset

For a = 0 to 19:

Register	Offset
GPRa	54h + (a × 4h)

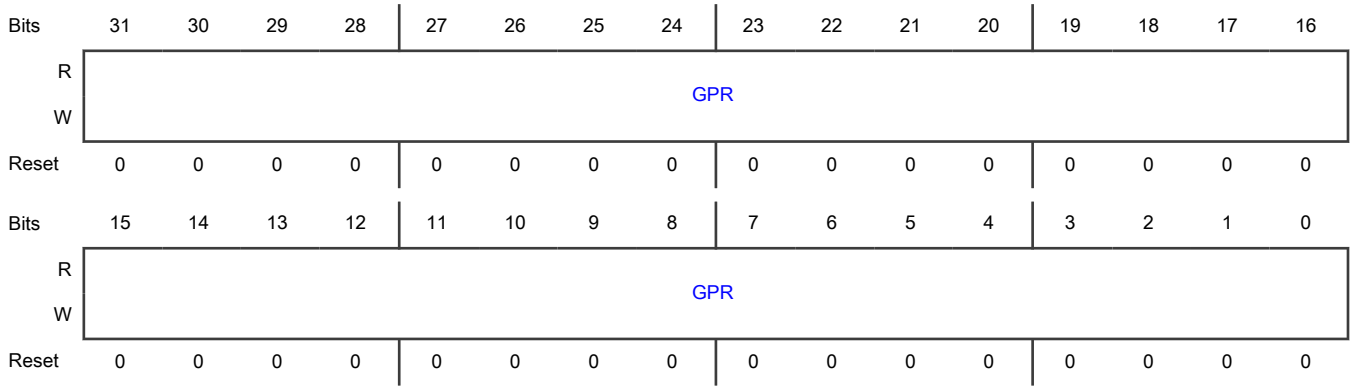
Function

The registers of GPR0, GPR1, GPR2, GPR3 and GPR9 are used by the ROM code and should not be used by user application. Reset by POR only, not by global system reset.

NOTE

The reset values of GPR0, GPR1, GPR2, GPR3 and GPR9 may not be logic 0, because ROM code runs first after the system reset.

Diagram



Fields

Field	Function
31-0 GPR	General Purpose Register. GPR0, GPR1, GPR2, GPR3 and GPR9 are used by the ROM code. User application should not use them.

27.6.2 SRC MIX SLICE register descriptions

27.6.2.1 SRC MIX SLICE Register memory map

AON_MIX_SLICE base address: 4446_0800h

CM7PLATFORM_MIX_SLICE base address: 4446_1C00h

CM33PLATFORM_MIX_SLICE base address: 4446_1800h

MEGA_MIX_SLICE base address: 4446_1000h

NETC_MIX_SLICE base address: 4446_1400h

WAKEUP_MIX_SLICE base address: 4446_0C00h

Offset	Register	Width (In bits)	Access	Reset value
4h	Authentication Control (AUTHEN_CTRL)	32	RW	FFFF_0000h
10h	SLICE Software Control (SLICE_SW_CTRL)	32	RW	0000_0000h
14h	Function Status (FUNC_STAT)	32	R	0000_0004h
20h	UPI Status 0 (UPI_STAT_0)	32	R	0000_0000h
24h	UPI Status 1 (UPI_STAT_1)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
30h	Low Power Mode Setting 0 (LPM_SETTING_0)	32	RW	0000_0003h
34h	Low Power Mode Setting 1 (LPM_SETTING_1)	32	RW	3333_3333h
38h	Low Power Mode Setting 2 (LPM_SETTING_2)	32	RW	3333_3333h
40h	Edgelock Handshake Control (EDGELOCK_HDSK_CTRL)	32	RW	8000_8000h
44h	Edgelock Handshake Status (EDGELOCK_HDSK_STAT)	32	R	0000_0001h
48h	Edgelock Handshake Counter Config (EDGELOCK_HDSK_CNT_CFG)	32	RW	2EE0_2EE0h
4Ch	Edgelock Handshake Counter Status (EDGELOCK_HDSK_CNT_STAT)	32	R	0000_0000h
50h	ISO Delay Pre control (ISO_DLY_PRE)	32	RW	0000_0000h
54h	PSW Delay Pre for HF (PSW_DLY_PRE_HF)	32	RW	0000_0000h
58h	PSW Delay Pre for LF (PSW_DLY_PRE_LF)	32	RW	0000_0000h
5Ch	PSW Control (PSW_CTRL)	32	RW	8000_8000h
60h	PSW Status (PSW_STAT)	32	R	0000_0000h
64h	PSW Counter Config for HF (PSW_CNT_CFG_HF)	32	RW	0000_0010h
68h	PSW Counter Status for HF (PSW_CNT_STAT_HF)	32	R	0000_0000h
6Ch	PSW Counter Config for LF (PSW_CNT_CFG_LF)	32	RW	0000_0010h
70h	PSW Counter Status for LF (PSW_CNT_STAT_LF)	32	R	0000_0000h
80h	Memory Low Power Level Trigger Control (MLPL_TRIG_CTRL)	32	RW	0000_0000h
84h	Memory Low Power Level Config (MLPL_CFG)	32	RW	3037_3037h
88h	Memory Low Power Level Status (MLPL_STAT)	32	R	0000_0033h

27.6.2.2 Authentication Control (AUTHEN_CTRL)

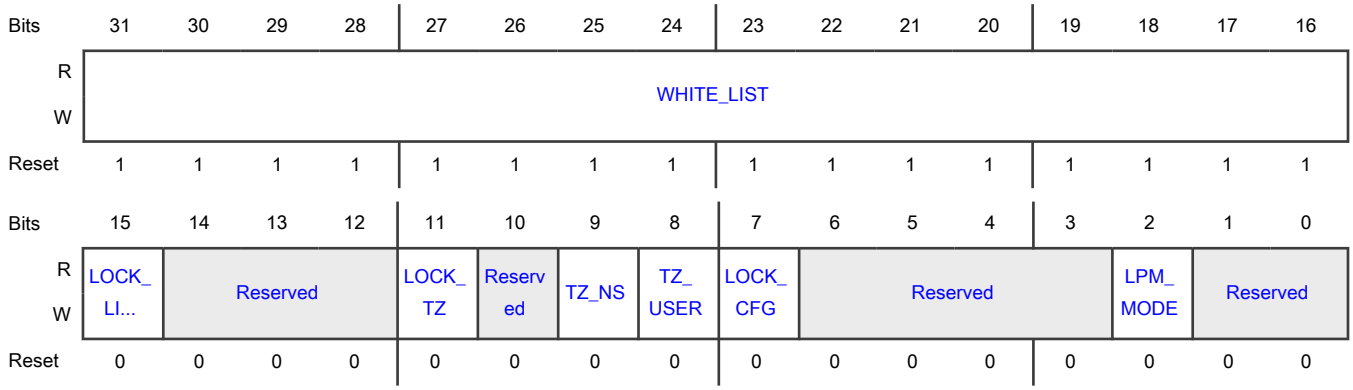
Offset

Register	Offset
AUTHEN_CTRL	4h

Function

Reset by global system reset.

Diagram



Fields

Field	Function
31-16 WHITE_LIST	Domain ID white list This field has functions as below: • Each master (Core or non-Core) can be assigned to any domains among domain0~domain15 by TRDC. Once its domains are assigned, then its accesses will have its unique domain ID. Each bit in this field represent for one domain. Bit16~bit31 represent for domain0~domain15 respectively. Only the domains whose corresponding bits are 1 can write the MIX SLICE registers and its MEM Type I/II registers. Write violation will fail without any exception or information. Read violation can still get read data. • When the bit is 1, LPM_SETTING value with corresponding domain ID will be used to compare with the relative Core's low power state bits. If Core's low power state value is not smaller than the LPM_SETTING value, then low power sequence will be triggered. When this bit is 0, relative Core's low power state value will be ignored, and the MIX SLICE will always be in normal run mode. 0000_0000_0000_0001b - Core with domain ID=0 can write SRC MIX SLICE registers. 0000_0000_0000_0010b - Core with domain ID=1 can write SRC MIX SLICE registers. 0000_0000_0000_0100b - Core with domain ID=2 can write SRC MIX SLICE registers. 0000_0000_0000_1000b - Core with domain ID=3 can write SRC MIX SLICE registers. 0000_0000_0001_0000b - Core with domain ID=4 can write SRC MIX SLICE registers. 0000_0000_0010_0000b - Core with domain ID=5 can write SRC MIX SLICE registers. 0000_0000_0100_0000b - Core with domain ID=6 can write SRC MIX SLICE registers. 0000_0000_1000_0000b - Core with domain ID=7 can write SRC MIX SLICE registers. 0000_0001_0000_0000b - Core with domain ID=8 can write SRC MIX SLICE registers. 0000_0010_0000_0000b - Core with domain ID=9 can write SRC MIX SLICE registers. 0000_0100_0000_0000b - Core with domain ID=10 can write SRC MIX SLICE registers. 0000_1000_0000_0000b - Core with domain ID=11 can write SRC MIX SLICE registers. 0001_0000_0000_0000b - Core with domain ID=12 can write SRC MIX SLICE registers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0010_0000_0000_0000b - Core with domain ID=13 can write SRC MIX SLICE registers. 0100_0000_0000_0000b - Core with domain ID=14 can write SRC MIX SLICE registers. 1000_0000_0000_0000b - Core with domain ID=15 can write SRC MIX SLICE registers.
15 LOCK_LIST	White list lock Can lock itself and WHITE_LIST. 0b - WHITE_LIST value can be changed. 1b - LOCK_LIST and WHITE_LIST value cannot be changed.
14-12 —	Reserved
11 LOCK_TZ	Lock TZ_NS and TZ_USER Can lock itself, TZ_NS and TZ_USER. 0b - TZ_NS and TZ_USER value can be changed. 1b - LOCK_TZ, TZ_NS and TZ_USER value cannot be changed.
10 —	Reserved
9 TZ_NS	Allow non-secure mode access Allows either secure mode or non-secure mode to write the MIX SLICE registers and its MEM Type I/II registers. 0b - The MIX SLICE registers and its MEM Type I/II registers can only be written in secure mode. 1b - The MIX SLICE registers and its MEM Type I/II registers can be written either in secure mode or non-secure mode.
8 TZ_USER	Allow user mode write Allows either privilege mode or user mode to write the MIX SLICE registers and its MEM Type I/II registers. 0b - The MIX SLICE registers and its MEM Type I/II registers can only be written in privilege mode. 1b - The MIX SLICE registers and its MEM Type I/II registers can be written either in privilege mode or user mode.
7 LOCK_CFG	Configuration lock Can lock itself and LPM_MODE fields. Also it can lock EDGELOCK_HDSK_CTRL, EDGELOCK_HDSK_CNT_CFG, ISO_DLY_PRE, PSW_DLY_PRE_HF, PSW_DLY_PRE_LF, PSW_CTRL, PSW_CNT_CFG_HF and PSW_CNT_CFG_LF registers. 0b - Registers are not locked.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Fields and registers in the list are locked.
6-3 —	Reserved
2 LPM_MODE	HW low power mode Controls whether low power works in Hardware(GPC) mode or Software mode. If it works in Software mode, all GPC's change state request will be acknowledged at once. This value can be locked by LOCK_CFG. 0b - Low power mode controlled by software. SLICE_SW_CTRL register can be updated. LPM_SETTING_0 register cannot be updated. 1b - Low power mode controlled by GPC. SLICE_SW_CTRL register cannot be updated. LPM_SETTING_0 register can be updated.
1-0 —	Reserved

27.6.2.3 SLICE Software Control (SLICE_SW_CTRL)

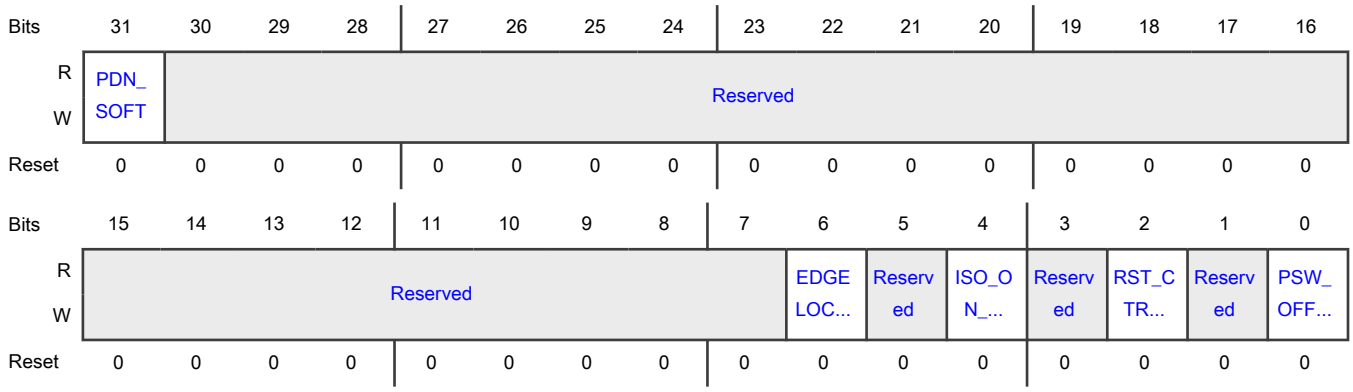
Offset

Register	Offset
SLICE_SW_CTRL	10h

Function

Controls the whole slice.

Diagram



Fields

Field	Function
31 PDN_SOFT	By flow sequence, Software power down sequence includes Edgelock handshake for power down, isolation on, reset assert, power off; Software power up sequence includes power up, reset deassert, isolation off, Edgelock handshake for power up. This bit can only be updated while LPM_MODE=0. 0b - Clear to 0 to trigger a power up sequence. 1b - Write 1 to trigger a power down sequence.
30-7 —	Reserved
6 EDGELOCK_HDSK_SOFT	Software Edgelock handshake control. This field can only be updated while LPM_MODE=0. 0b - Clear to 0 to trigger Edgelock handshake for power on. 1b - Write to 1 to trigger Edgelock handshake for power off.
5 —	Reserved
4 ISO_ON_SOFT	Software isolation on control. This field can only be updated while LPM_MODE=0. 0b - Clear to 0 to trigger isolation off 1b - Write 1 to trigger isolation on
3 —	Reserved
2 RST_CTRL_SOFT	Software reset control. This field can only be updated while LPM_MODE=0. **NOTE** Do not reset a target MIX slice if it has been in low power sequence, no matter it is in Edgelock, ISO, reset or Power off phase. 0b - Clear to 0 to trigger reset deassert 1b - Write 1 to trigger reset assert
1 —	Reserved
0 PSW_OFF_SOFT	Software power off control. This field can only be updated while LPM_MODE=0. 0b - Clear to 0 to trigger power switch on 1b - Write 1 to trigger power switch off

27.6.2.4 Function Status (FUNC_STAT)

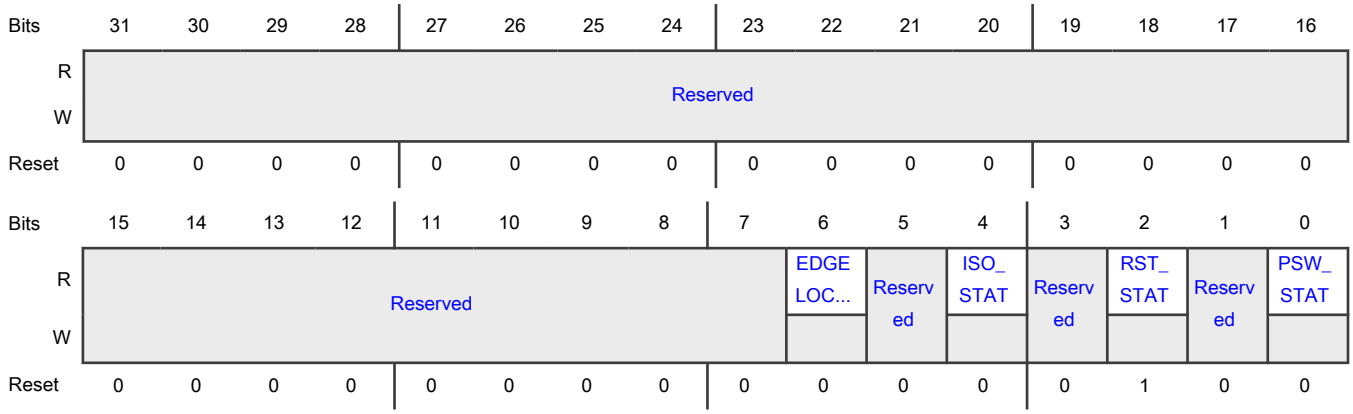
Offset

Register	Offset
FUNC_STAT	14h

Function

Function Status register

Diagram



Fields

Field	Function
31-16 —	Reserved
15-7 —	Reserved
6 EDGELOCK_HDSK_STAT	Edgelock handshake status 0b - no effect or power up handshake with Edgelock done 1b - power down handshake with Edgelock done
5 —	Reserved
4 ISO_STAT	isolation status 0b - isolation off 1b - isolation on
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 RST_STAT	reset status 0b - reset asserted 1b - reset released
1 —	Reserved
0 PSW_STAT	power switch status 0b - power switch on 1b - power switch off

27.6.2.5 UPI Status 0 (UPI_STAT_0)

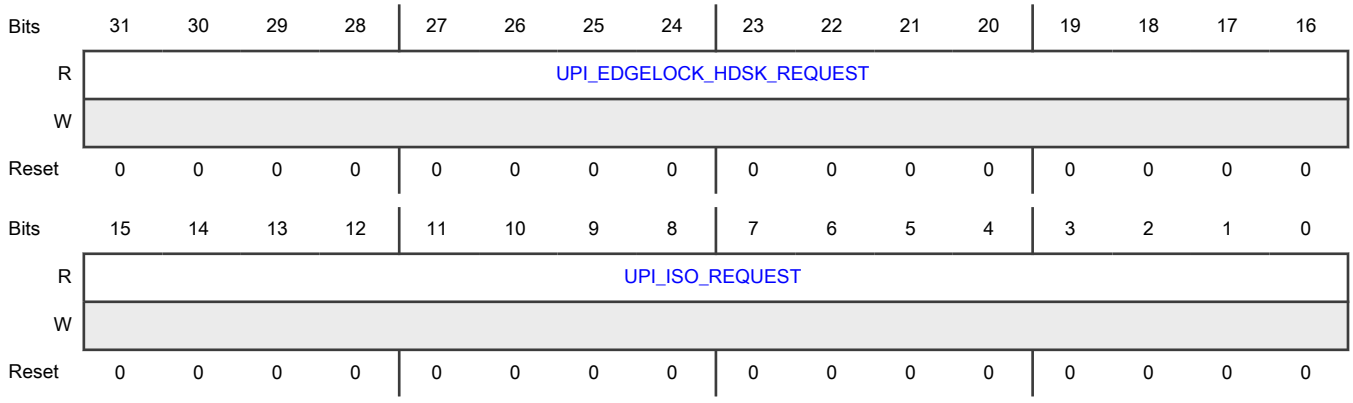
Offset

Register	Offset
UPI_STAT_0	20h

Function

UPI Status 0 register

Diagram



Fields

Field	Function
31-16	CPU mode transfer to trigger Edgelock handshake request of 16 domains

Table continues on the next page...

Table continued from the previous page...

Field	Function
UPI_EDGELOCK_HDSK_REQUEST	Each bit in this field represents for one domain. Bit16~bit31 represent for domain0~domain15 respectively.
15-0 UPI_ISO_REQUEST	CPU mode transfer to trigger isolation change request of 16 domains Each bit in this field represents for one domain. Bit0~bit15 represent for domain0~domain15 respectively.

27.6.2.6 UPI Status 1 (UPI_STAT_1)

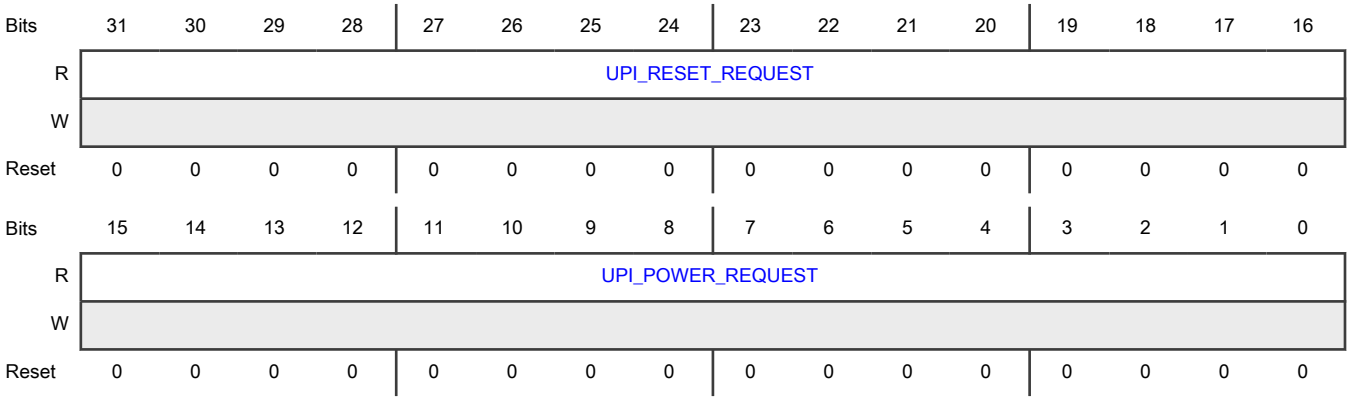
Offset

Register	Offset
UPI_STAT_1	24h

Function

UPI Status 1 register

Diagram



Fields

Field	Function
31-16 UPI_RESET_REQUEST	CPU mode transfer to trigger reset change request of 16 domains Each bit in this field represents for one domain. Bit16~bit31 represent for domain0~domain15 respectively.
15-0 UPI_POWER_REQUEST	CPU mode transfer to trigger power switch request of 16 domains Each bit in this field represents for one domain. Bit16~bit31 represent for domain0~domain15 respectively.

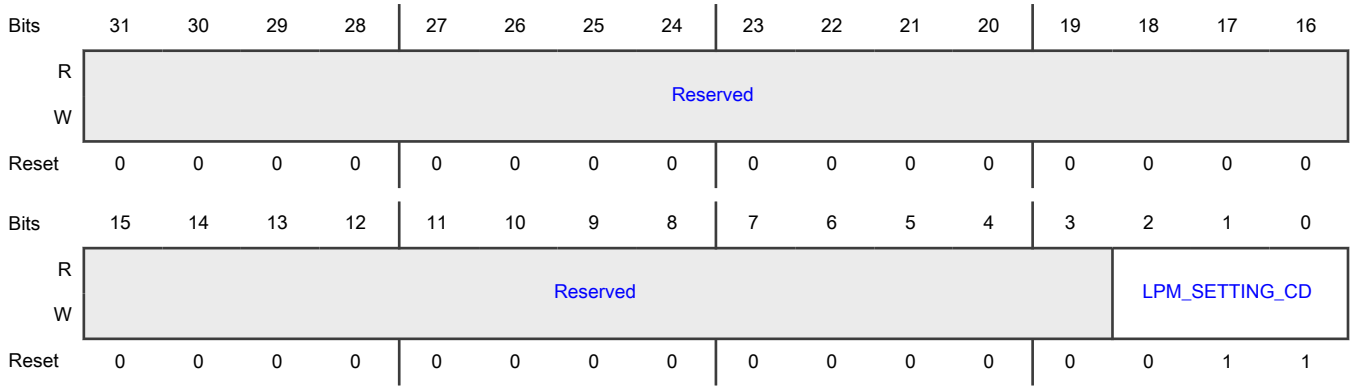
27.6.2.7 Low Power Mode Setting 0 (LPM_SETTING_0)

Offset

Register	Offset
LPM_SETTING_0	30h

Function
Low Power Mode Setting 0 register

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 LPM_SETTING_CD	LPM setting of current domain Can only be updated while MIX SLICE AUTHEN_CTRL[LPM_MODE]=1. When writing to this register, the data will update related domain register based on ips_domain_id.

27.6.2.8 Low Power Mode Setting 1 (LPM_SETTING_1)

Offset

Register	Offset
LPM_SETTING_1	34h

Function
Low Power Mode Setting 1 register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	LPM_SETTING_D7			Reserved	LPM_SETTING_D6			Reserved	LPM_SETTING_D5			Reserved	LPM_SETTING_D4		
Reset	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	LPM_SETTING_D3			Reserved	LPM_SETTING_D2			Reserved	LPM_SETTING_D1			Reserved	LPM_SETTING_D0		
Reset	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1

Fields

Field	Function
31 —	Reserved
30-28 LPM_SETTING_D7	LPM setting of domain 7 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
27 —	Reserved
26-24 LPM_SETTING_D6	LPM setting of domain 6 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
23 —	Reserved
22-20 LPM_SETTING_D5	LPM setting of domain 5 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
19 —	Reserved
18-16 LPM_SETTING_D4	LPM setting of domain 4 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
15 —	Reserved
14-12 LPM_SETTING_D3	LPM setting of domain 3 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
11 —	Reserved
10-8 LPM_SETTING_D2	LPM setting of domain 2 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
7 —	Reserved
6-4 LPM_SETTING_D1	LPM setting of domain 1 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
3 —	Reserved
2-0 LPM_SETTING_D0	LPM setting of domain 0 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on

27.6.2.9 Low Power Mode Setting 2 (LPM_SETTING_2)

Offset

Register	Offset
LPM_SETTING_2	38h

Function

Low Power Mode Setting 2 register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	LPM_SETTING_D15			Reserved	LPM_SETTING_D14			Reserved	LPM_SETTING_D13			Reserved	LPM_SETTING_D12		
Reset	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	LPM_SETTING_D11			Reserved	LPM_SETTING_D10			Reserved	LPM_SETTING_D9			Reserved	LPM_SETTING_D8		
Reset	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1

Fields

Field	Function
31 —	Reserved
30-28 LPM_SETTING_D15	LPM setting of domain 15 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
27 —	Reserved
26-24 LPM_SETTING_D14	LPM setting of domain 14 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
23 —	Reserved
22-20 LPM_SETTING_D13	LPM setting of domain 13 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
19 —	Reserved
18-16 LPM_SETTING_D12	LPM setting of domain 12 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 —	Reserved
14-12 LPM_SETTING_D11	LPM setting of domain 11 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
11 —	Reserved
10-8 LPM_SETTING_D10	LPM setting of domain 10 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
7 —	Reserved
6-4 LPM_SETTING_D9	LPM setting of domain 9 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on
3 —	Reserved
2-0 LPM_SETTING_D8	LPM setting of domain 8 000b - Not used. Do not write this value. 001b - power on when domain n is in RUN, off in WAIT/STOP/SUSPEND 010b - power on when domain n is in RUN/WAIT, off in STOP/SUSPEND 011b - power on when domain n is in RUN/WAIT/STOP, off in SUSPEND 100b-111b - power always on

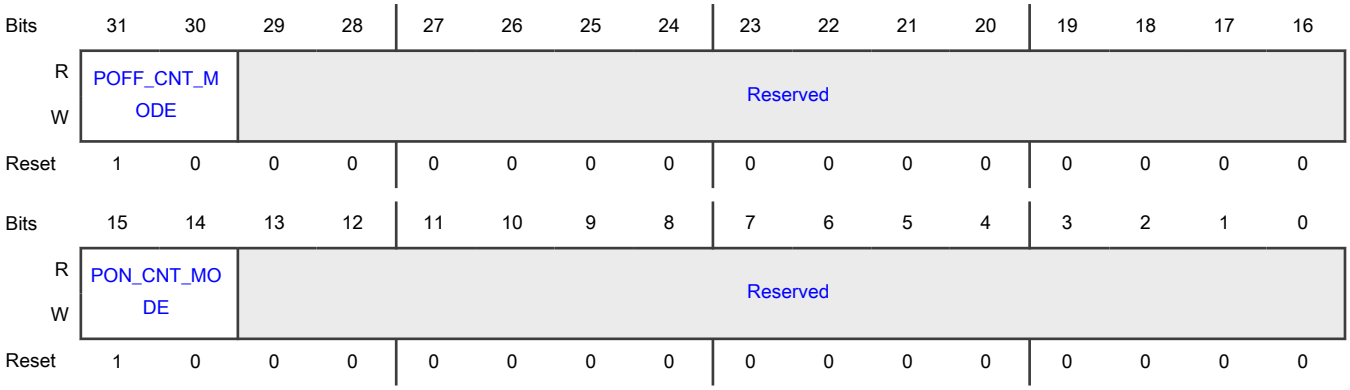
27.6.2.10 Edgelock Handshake Control (EDGELOCK_HDSK_CTRL)

Offset

Register	Offset
EDGELOCK_HDSK_CTRL	40h

Function
Edgelock Handshake Control register

Diagram



Fields

Field	Function
31-30 POFF_CNT_MODE	Configures the acknowledge counter for power down working mode, locked by LOCK_CFG field 00b - Not use counter, raise Edgelock handshake done to GPC once get Edgelock ack 01b - Delay after receiving Edgelock ack, delay cycle number is POFF_CNT_CFG 10b - Ignore Edgelock ack, raise Edgelock handshake done to GPC when counting to POFF_CNT_CFG value 11b - Time out mode, raise Edgelock handshake done to GPC when either Edgelock ack received or counting to POFF_CNT_CFG value
29-16 —	Reserved
15-14 PON_CNT_MODE	Configures the acknowledge counter for power up working mode, locked by LOCK_CFG field 00b - Not use counter, raise Edgelock handshake done to GPC once get Edgelock ack 01b - Delay after receiving Edgelock ack, delay cycle number is PON_CNT_CFG 10b - Ignore Edgelock ack, raise Edgelock handshake done to GPC when counting to PON_CNT_CFG value

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Time out mode, raise Edgelock handshake done to GPC when either Edgelock ack received or counting to PON_CNT_CFG value
13-0 —	Reserved

27.6.2.11 Edgelock Handshake Status (EDGELOCK_HDSK_STAT)

Offset

Register	Offset
EDGELOCK_HDSK_STAT	44h

Function
Edgelock handshake acknowledge status

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														POFF_BU...	POFF_ACK
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														PON_BUSY	PON_ACK
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-18 —	Reserved
17 POFF_BUSY	Busy requesting Edgelock handshake for power down 0b - Does not send Edgelock handshake for power down. 1b - Sends Edgelock handshake for power down.

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 POFF_ACK	Indicates this mix power down sequence has accepted Edgelock ack 0b - Does not get Edgelock ack for power down. 1b - Gets Edgelock ack for power down.
15-2 —	Reserved
1 PON_BUSY	Busy requesting Edgelock handshake for power up 0b - Does not send Edgelock handshake for power up. 1b - Sends Edgelock handshake for power up.
0 PON_ACK	Indicates this mix power up sequence has accepted Edgelock ack 0b - Does not get Edgelock ack for power up. 1b - Gets Edgelock ack for power up.

27.6.2.12 Edgelock Handshake Counter Config (EDGELOCK_HDSK_CNT_CFG)

Offset

Register	Offset
EDGELOCK_HDSK_CNT_CFG	48h

Function

Edgelock handshake acknowledge counter configuration

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	POFF_CNT_CFG															
Reset	0	0	1	0	1	1	1	0	1	1	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PON_CNT_CFG															
Reset	0	0	1	0	1	1	1	0	1	1	1	0	0	0	0	0

Fields

Field	Function
31-16 POFF_CNT_CFG	Edgelock handshake ack count for power down config: usage depends on CNT_MODE, locked by LOCK_CFG field
15-0 PON_CNT_CFG	Edgelock handshake ack count for power up config: usage depends on CNT_MODE, locked by LOCK_CFG field

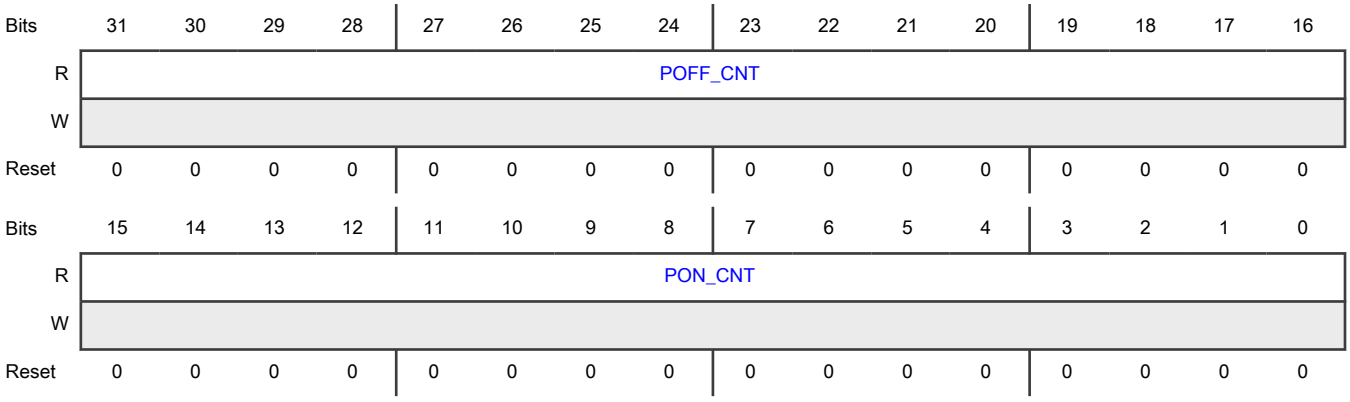
27.6.2.13 Edgelock Handshake Counter Status (EDGELOCK_HDSK_CNT_STAT)

Offset

Register	Offset
EDGELOCK_HDSK_CNT_STAT	4Ch

Function
Edgelock handshake acknowledge counter status

Diagram



Fields

Field	Function
31-16 POFF_CNT	Edgelock handshake for power down acknowledge count, record the delay from stat change to acknowledge received
15-0 PON_CNT	Edgelock handshake for power up acknowledge count, record the delay from stat change to acknowledge received

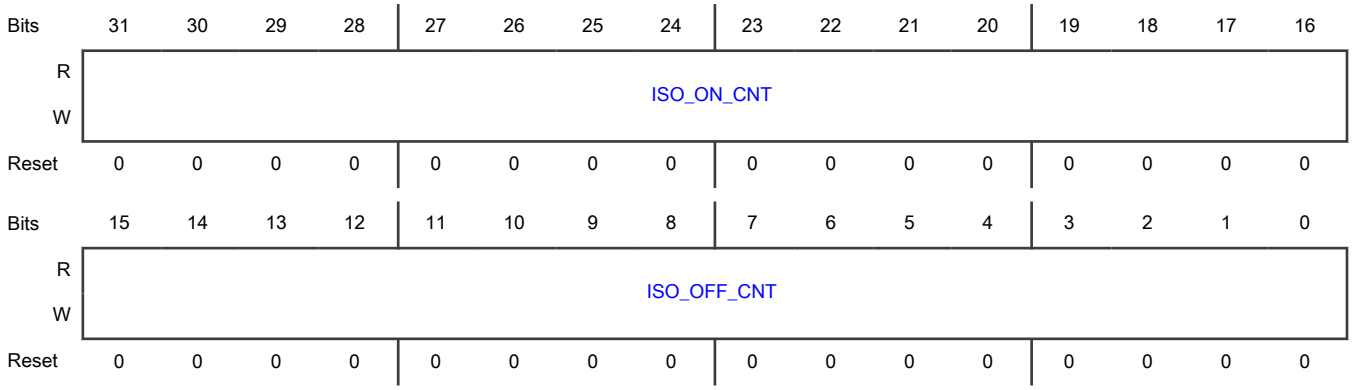
27.6.2.14 ISO Delay Pre control (ISO_DLY_PRE)

Offset

Register	Offset
ISO_DLY_PRE	50h

Function
ISO Delay Pre control register

Diagram



Fields

Field	Function
31-16 ISO_ON_CNT	Delay from receiving iso_on request to isolation enable, locked by LOCK_CFG field
15-0 ISO_OFF_CNT	Delay from receiving iso off request to isolation disable, locked by LOCK_CFG field

27.6.2.15 PSW Delay Pre for HF (PSW_DLY_PRE_HF)

Offset

Register	Offset
PSW_DLY_PRE_HF	54h

Function
Delay before real power switch for high fanout logic.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PSW_OFF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PSW_ON															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 PSW_OFF	Delay from receiving power off hf request to power switch shut off, locked by LOCK_CFG field
15-0 PSW_ON	Delay from receiving power on hf request to power switch turn on, locked by LOCK_CFG field

27.6.2.16 PSW Delay Pre for LF (PSW_DLY_PRE_LF)

Offset

Register	Offset
PSW_DLY_PRE_LF	58h

Function

Delay before real power switch for low fanout logic.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PSW_OFF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PSW_ON															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 PSW_OFF	Delay from receiving power off lf request to power switch shut off, locked by LOCK_CFG field
15-0 PSW_ON	Delay from receiving power on lf request to power switch turn on, locked by LOCK_CFG field

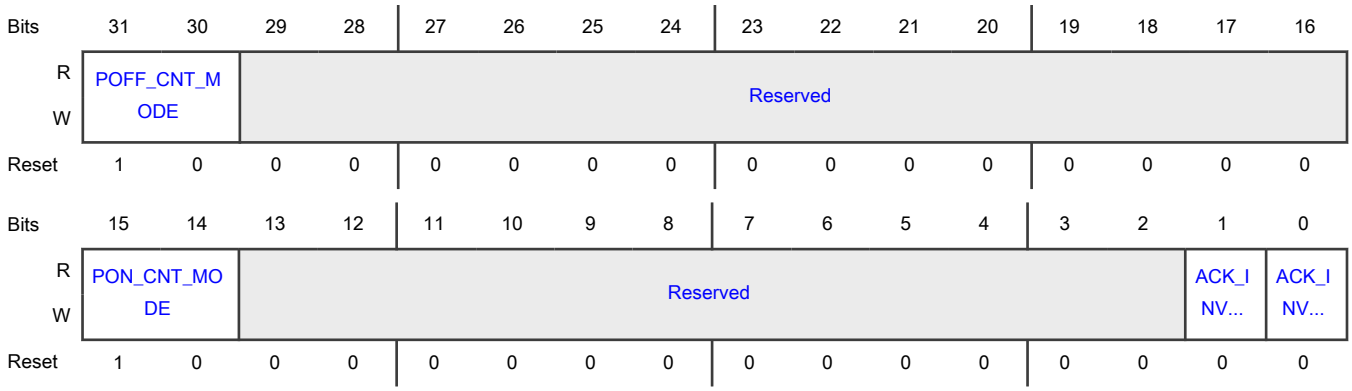
27.6.2.17 PSW Control (PSW_CTRL)

Offset

Register	Offset
PSW_CTRL	5Ch

Function
Power switch acknowledge control

Diagram



Fields

Field	Function
31-30 POFF_CNT_MODE	Configures the acknowledge counter for power down working mode, locked by LOCK_CFG field **NOTE** This field must be 10b. Do not change this field to other value. 00b - Not use counter, raise power_on/off done to GPC once get psw ack 01b - Delay after receiving psw ack, delay cycle number is POFF_CNT_CFG. When POFF_CNT_CFG value in HF/LF is different, bigger value will be used.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Ignore psw ack, raise power_on/off done to GPC when counting to POFF_CNT_CFG value. When POFF_CNT_CFG value in HF/LF is different, bigger value will be used. 11b - Time out mode, raise power_on/off done to GPC when either psw ack received or counting to POFF_CNT_CFG value. When POFF_CNT_CFG value in HF/LF is different, bigger value will be used.
29-16 —	Reserved
15-14 PON_CNT_MODE	Configure the acknowledge counter for power up working mode, locked by LOCK_CFG field **NOTE** This field must be 10b. Do not change this field to other value. 00b - Not use counter, raise power_on/off done to GPC once get psw ack 01b - Delay after receiving psw ack, delay cycle number is PON_CNT_CFG 10b - Ignore psw ack, raise power_on/off done to GPC when counting to PON_CNT_CFG value 11b - Time out mode, raise power_on/off done to GPC when either psw ack received or counting to PON_CNT_CFG value
13-2 —	Reserved
1 ACK_INVERT_HF	Acknowledge for high fanout value is inverted from power switch control, locked by LOCK_CFG field 0b - Use original signal as ack 1b - Use inverted original signal as ack
0 ACK_INVERT_LF	Acknowledge for low fanout value is inverted from power switch control, locked by LOCK_CFG field 0b - Use original signal as ack 1b - Use inverted original signal as ack

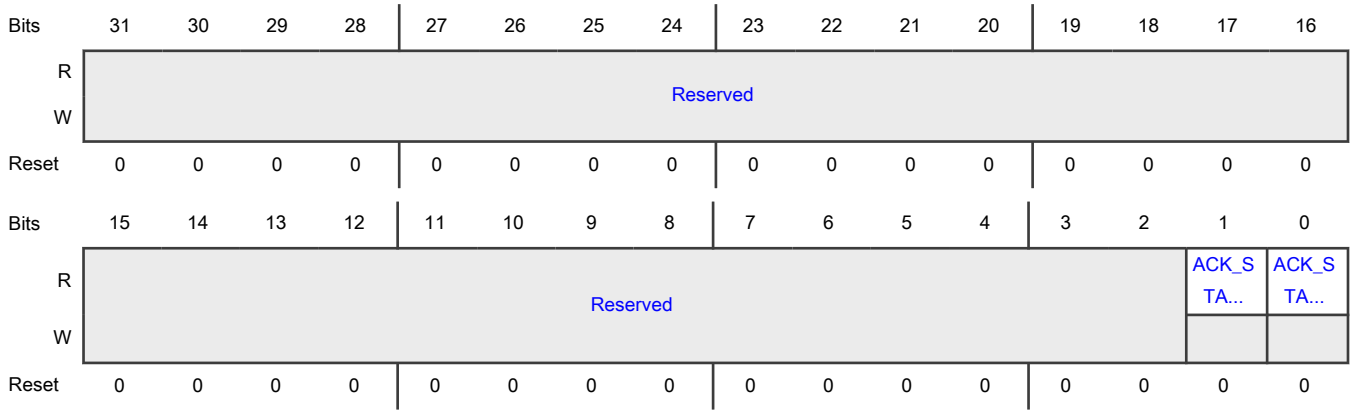
27.6.2.18 PSW Status (PSW_STAT)

Offset

Register	Offset
PSW_STAT	60h

Function
Power switch acknowledge status

Diagram



Fields

Field	Function
31-2 —	Reserved
1 ACK_STAT_HF	Power switch acknowledge status for high fanout 0b - Does not get ack for power switch 1b - Gets ack for power switch
0 ACK_STAT_LF	Power switch acknowledge status for low fanout 0b - Does not get ack for power switch 1b - Gets ack for power switch

27.6.2.19 PSW Counter Config for HF (PSW_CNT_CFG_HF)

Offset

Register	Offset
PSW_CNT_CFG_HF	64h

Function

Power switch acknowledge counter config for high fanout.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	POFF_CNT_CFG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PON_CNT_CFG															
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-16 POFF_CNT_CFG	PDN HF Count config: usage depends on CNT_MODE, locked by LOCK_CFG field
15-0 PON_CNT_CFG	PUP HF Count config: usage depends on CNT_MODE, locked by LOCK_CFG field

27.6.2.20 PSW Counter Status for HF (PSW_CNT_STAT_HF)

Offset

Register	Offset
PSW_CNT_STAT_HF	68h

Function

Power switch acknowledge counter status for high fanout.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	POFF_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PON_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 POFF_CNT	HF PSW acknowledge count for power down, record the delay from power switch change to acknowledge received
15-0 PON_CNT	HF PSW acknowledge count for power up, record the delay from power switch change to acknowledge received

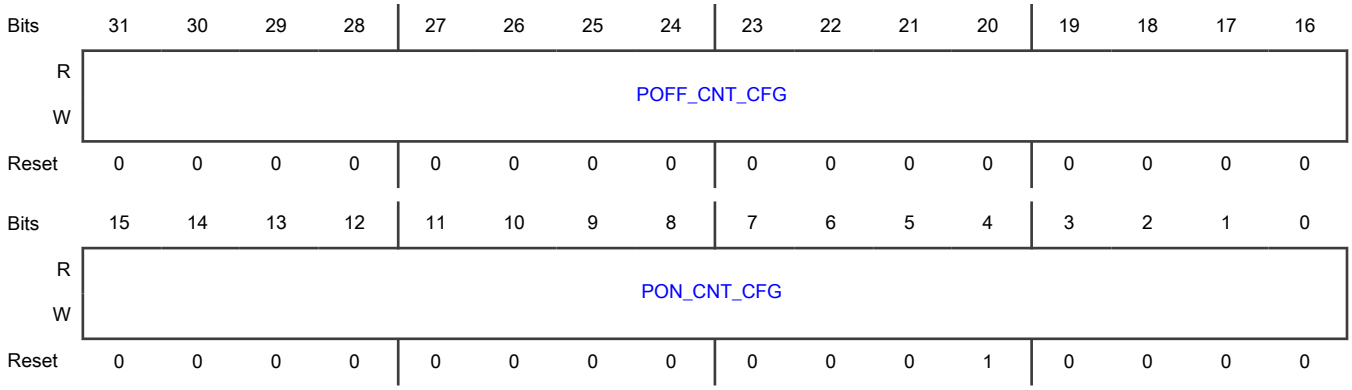
27.6.2.21 PSW Counter Config for LF (PSW_CNT_CFG_LF)

Offset

Register	Offset
PSW_CNT_CFG_LF	6Ch

Function
Power switch acknowledge counter config for low fanout.

Diagram



Fields

Field	Function
31-16 POFF_CNT_CFG	PDN LF Count config: usage depends on CNT_MODE, locked by LOCK_CFG field
15-0 PON_CNT_CFG	PUP LF Count config: usage depends on CNT_MODE, locked by LOCK_CFG field

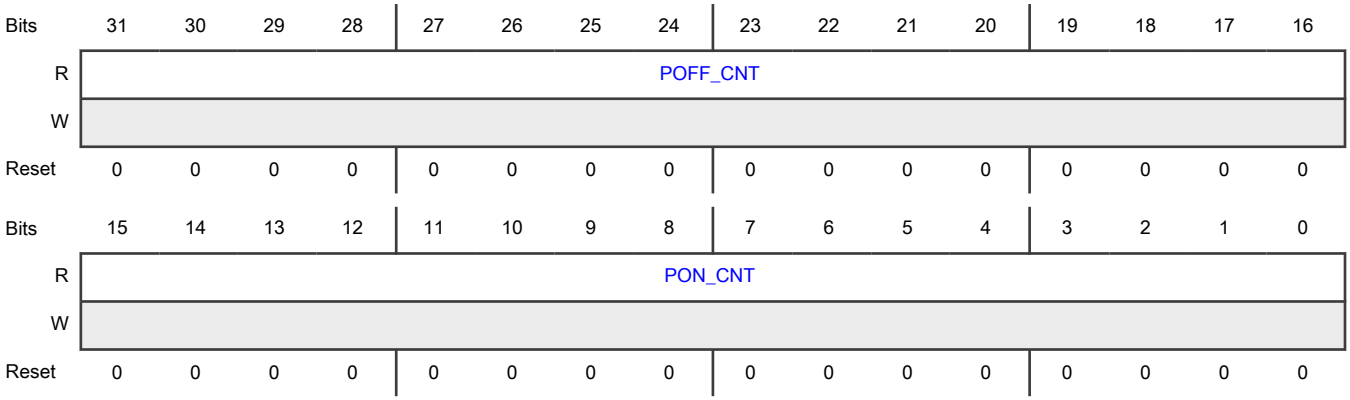
27.6.2.22 PSW Counter Status for LF (PSW_CNT_STAT_LF)

Offset

Register	Offset
PSW_CNT_STAT_LF	70h

Function
PSW acknowledge counter status for low fanout.

Diagram



Fields

Field	Function
31-16 POFF_CNT	LF PSW acknowledge count for power down, record the delay from power switch change to acknowledge received
15-0 PON_CNT	LF PSW acknowledge count for power up, record the delay from power switch change to acknowledge received

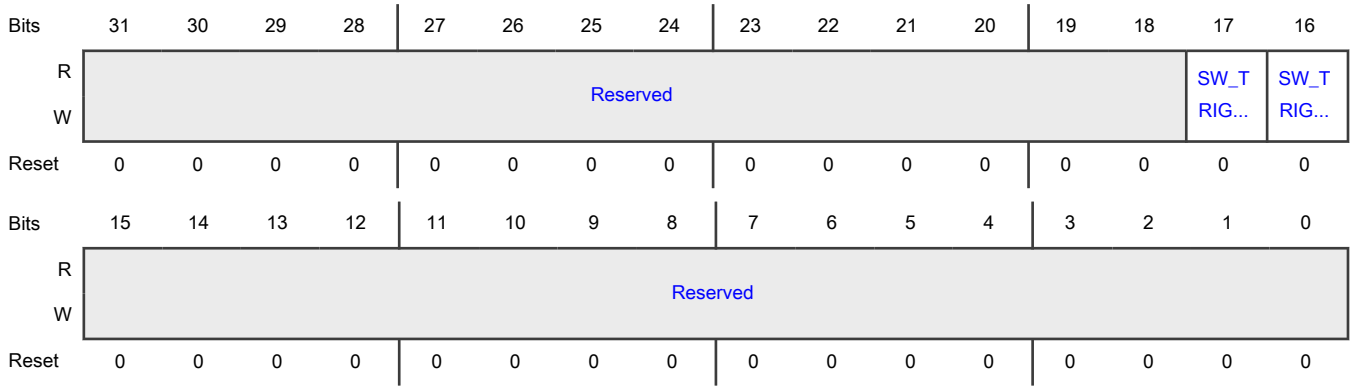
27.6.2.23 Memory Low Power Level Trigger Control (MLPL_TRIG_CTRL)

Offset

Register	Offset
MLPL_TRIG_CTRL	80h

Function
Memory Low Power Level Trigger Control register

Diagram



Fields

Field	Function
31-18 —	Reserved
17 SW_TRIG_CACHE_MLPL_TRANS	Software trigger CACHE memory (MEM Type II or MEM Type I's CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register) update low power level 0b - No trigger 1b - Software trigger CACHE memory update low power level
16 SW_TRIG_LMEM_MLPL_TRANS	Software trigger local memory (MEM Type I with PSW or MEM Type I except for CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register) update low power level 0b - No trigger 1b - Software trigger local memory update low power level
15-0 —	Reserved

27.6.2.24 Memory Low Power Level Config (MLPL_CFG)

Offset

Register	Offset
MLPL_CFG	84h

Function

Memory Low Power Level Config register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	MLPL_SW_CACHE			Reserved					MLPL_HW_PUP_CACHE			Reserved	MLPL_HW_PDN_CACHE		
Reset	0	0	1	1	0	0	0	0	0	0	1	1	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	MLPL_SW_LMEM			Reserved					MLPL_HW_PUP_LMEM			Reserved	MLPL_HW_PDN_LMEM		
Reset	0	0	1	1	0	0	0	0	0	0	1	1	0	1	1	1

Fields

Field	Function
31 —	Reserved
30-28 MLPL_SW_CACHE	Memory(MEM Type II or MEM Type I's CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register) low power level when sw trigger memory update power level
27-23 —	Reserved
22-20 MLPL_HW_PUP_CACHE	Memory(MEM Type II or MEM Type I's CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register) low power level when hw power up
19 —	Reserved
18-16 MLPL_HW_PDN_CACHE	Memory(MEM Type II or MEM Type I's CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register) low power level when hw power down
15 —	Reserved
14-12 MLPL_SW_LMEM	Memory(MEM Type I with PSW or MEM Type I except for CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register) low power level when sw trigger memory update power level
11-7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-4 MLPL_HW_PUP_LMEM	Memory(MEM Type I with PSW or MEM Type I except for CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register) low power level when hw power up
3 —	Reserved
2-0 MLPL_HW_PDN_LMEM	Memory(MEM Type I with PSW or MEM Type I except for CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register) low power level when hw power down

27.6.2.25 Memory Low Power Level Status (MLPL_STAT)

Offset

Register	Offset
MLPL_STAT	88h

Function
Memory Low Power Level Status register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								CACHE_B...	CURRENT_MLPL_CACHE			LMEM_BU...	CURRENT_MLPL_LMEM		
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1

Fields

Field	Function
31-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 CACHE_BUSY	Busy requesting low power level of memory (MEM Type II or MEM Type I's CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register)
6-4 CURRENT_MLPL_CACHE	Current memory (MEM Type II or MEM Type I's CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register) low power level
3 LMEM_BUSY	Busy requesting low power level of memory (MEM Type I with PSW or MEM Type I except for CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register)
2-0 CURRENT_MLPL_LMEM	Current memory (MEM Type I with PSW or MEM Type I except for CM33PLATFORM_CACHE or CM7PLATFORM_CACHE, depending on which MIX of the register) low power level

27.6.3 MEM Type I register descriptions

27.6.3.1 src_mif_s28spregh memory map

AON_MIF_S28SPREGH base address: 4446_2000h

CM7PLATFORM_CACHE base address: 4446_4800h

CM33PLATFORM_CACHE base address: 4446_4000h

CM33PLATFORM_TCM base address: 4446_4400h

MEGA_MIF_S28SPREGH base address: 4446_3000h

NETC_MIF_S28SPREGH base address: 4446_3800h

WAKEUP_MIF_S28SPREGH base address: 4446_2800h

Offset	Register	Width (In bits)	Access	Reset value
4h	MPC Control (MIF_CTRL)	32	RW	0000_0000h
8h	MIF Status (MIF_STAT)	32	R	0000_0000h
10h	MIF MLPL control of LS (MIF_MLPL_LS)	32	RW	0000_0000h
14h	MIF Delay of LS (MIF_DLY_LS)	32	RW	0000_0000h
20h	MIF MLPL control of HS (MIF_MLPL_HS)	32	RW	0000_0002h
24h	MIF Delay of HS (MIF_DLY_HS)	32	RW	0000_0000h
30h	MIF MLPL control of Input Gating (IG) (MIF_MLPL_IG)	32	RW	0000_0010h
34h	MIF Delay of IG (MIF_DLY_IG)	32	RW	0000_0000h
40h	MIF MLPL control of STDBY (MIF_MLPL_STDBY)	32	RW	0000_0000h
44h	MIF Delay of STDBY (MIF_DLY_STDBY)	32	RW	0000_0000h

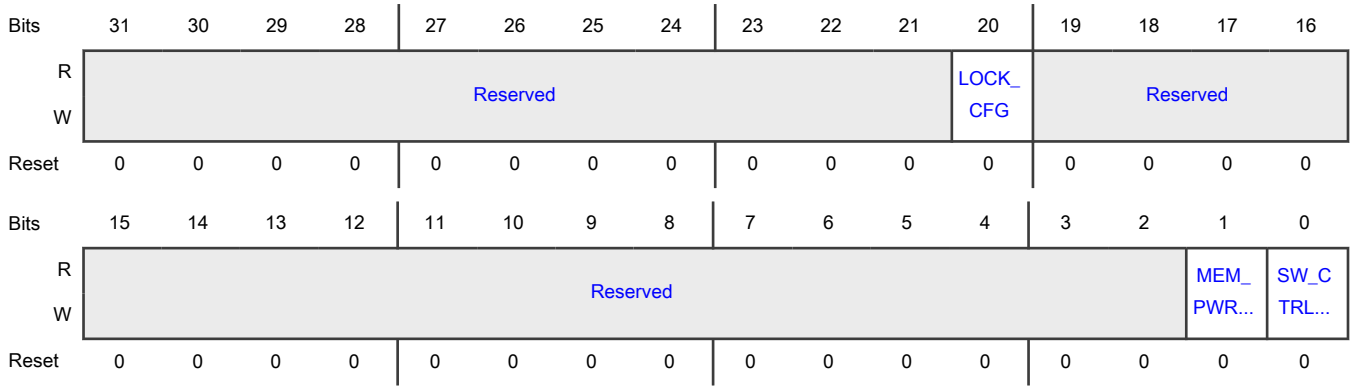
27.6.3.2 MPC Control (MIF_CTRL)

Offset

Register	Offset
MIF_CTRL	4h

Function
MPC Control register

Diagram



Fields

Field	Function
31-21 —	Reserved
20 LOCK_CFG	Configuration lock Locks itself and some fields of the registers in this module. If the field can be locked by LOCK_CFG, it will be shown in the field's description. Else the field cannot be locked by LOCK_CFG. 0b - The fields are not locked. 1b - The fields are locked.
19-2 —	Reserved
1 MEM_PWR_ST _EN	Memory power status will be considered when determining slice power status. This field should only be used for debug. So do not change it. 0b - Memory power status will not be considered when determining slice power status. 1b - Memory power status will be considered when determining slice power status.
0	Memory low power pins controlled by software (SW)

Table continues on the next page...

Table continued from the previous page...

Field	Function
SW_CTRL_PIN	Selects how to control such type of memory's low power control signals. They are IG, LS, HS and STDBY. Selecting SW mode means use SW_IG, SW_LS, SW_HS and SW_STDBY fields in each MIF MLPL control registers to control relative signals directly. Otherwise CURRENT_MLPL value will be used to select 1 value from 8 values in each MLPL_CTRL field to control the relative low power control signal. 0b - Uses CURRENT_MLPL field to select MLPL_CTRL to control low power signal. 1b - Uses SW_* field to control low power signal directly.

27.6.3.3 MIF Status (MIF_STAT)

Offset

Register	Offset
MIF_STAT	8h

Function

.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved									STDBY_S...	IG_STATE	HS_STATE	LS_STATE	MLPL_STATE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved
6 STDBY_STATE	Current state of STDBY 0b - STDBY is 0. 1b - STDBY is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 IG_STATE	Current state of IG 0b - IG is 0. 1b - IG is 1.
4 HS_STATE	Current state of HS 0b - HS is 0. 1b - HS is 1.
3 LS_STATE	Current state of LS 0b - LS is 0. 1b - LS is 1.
2-0 MLPL_STATE	Current state of CURRENT_MLPL

27.6.3.4 MIF MLPL control of LS (MIF_MLPL_LS)

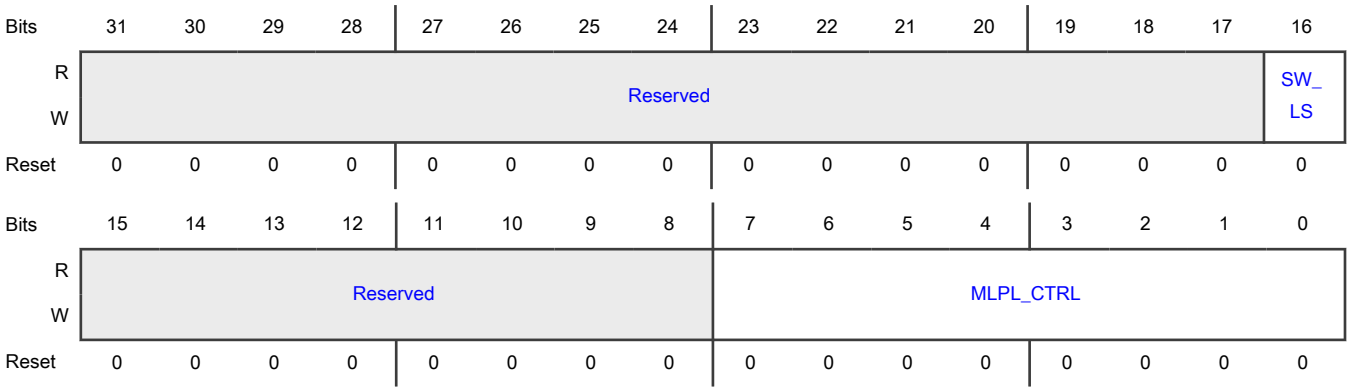
Offset

Register	Offset
MIF_MLPL_LS	10h

Function

Speed mode provides the flexibility to boost speed and gain dynamic power by controlling memory LS/HS pins depending on operating voltage. This option enables the user to operate memory in different speed modes. LS pin can be constraint to H only for low voltages, and LS = L at higher voltage will boost speed and reduce dynamic power. In conjunction with LS = L, HS = H with specific voltage constraints will further boost speed.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 SW_LS	Software control LS 0b - LS is 0. 1b - LS is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is logic 1 at the MLPL setting. This field is locked by LOCK_CFG field.

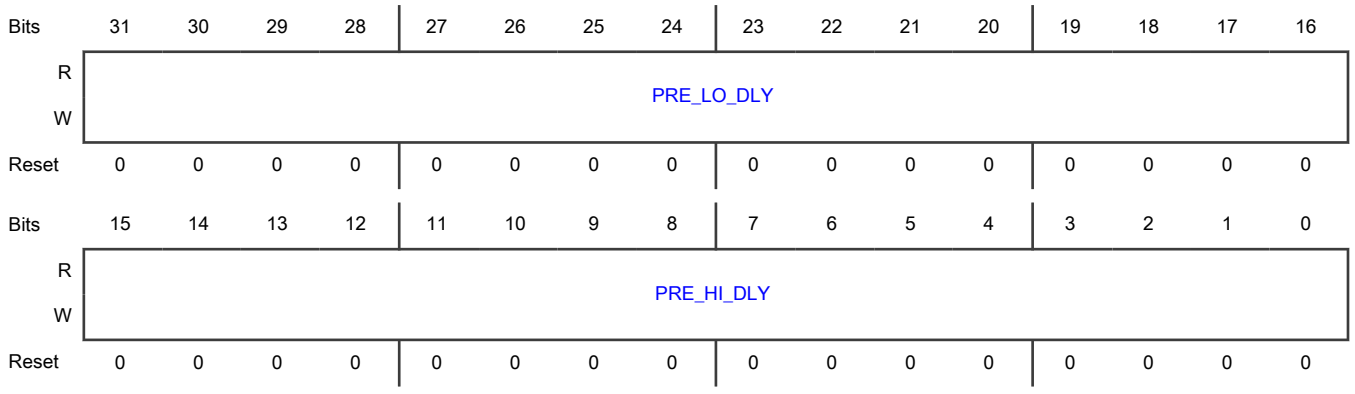
27.6.3.5 MIF Delay of LS (MIF_DLY_LS)

Offset

Register	Offset
MIF_DLY_LS	14h

Function
MIF Delay of LS register

Diagram



Fields

Field	Function
31-16	Delay before de-asserting signal to low, locked by LOCK_CFG field

Table continues on the next page...

Table continued from the previous page...

Field	Function
PRE_LO_DLY	This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the LS signal to low.
15-0 PRE_HI_DLY	Delay before asserting signal to high, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the LS signal to high.

27.6.3.6 MIF MLPL control of HS (MIF_MLPL_HS)

Offset

Register	Offset
MIF_MLPL_HS	20h

Function

Speed mode provides the flexibility to boost speed and gain dynamic power by controlling memory LS/HS pins depending on operating voltage. This option enables the user to operate memory in different speed modes. LS pin can be constraint to H only for low voltages, and LS = L at higher voltage will boost speed and reduce dynamic power. In conjunction with LS = L, HS = H with specific voltage constraints will further boost speed.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															SW_HS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								MLPL_CTRL							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-17 —	Reserved
16 SW_HS	software control HS 0b - HS is 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - HS is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is logic 1 at the MLPL setting. This field is locked by LOCK_CFG field.

27.6.3.7 MIF Delay of HS (MIF_DLY_HS)

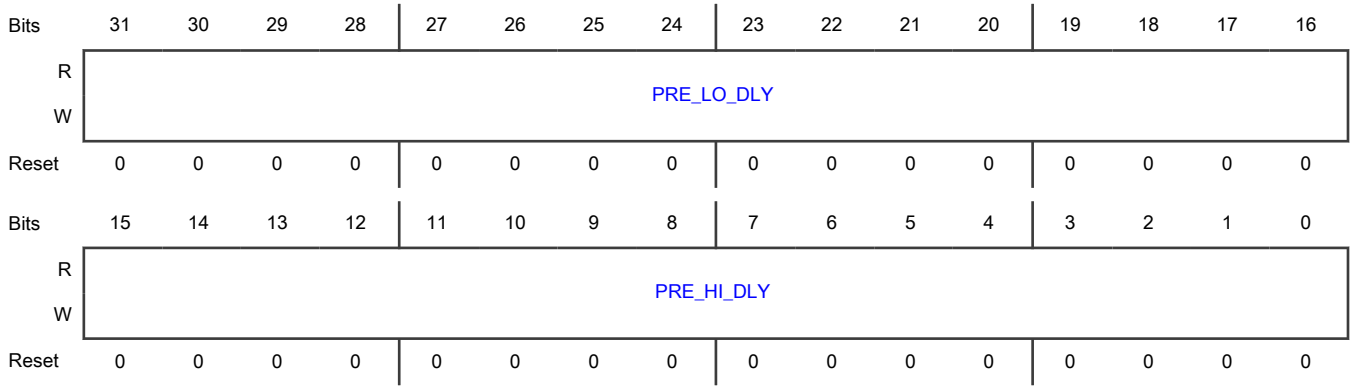
Offset

Register	Offset
MIF_DLY_HS	24h

Function

MIF Delay of HS register

Diagram



Fields

Field	Function
31-16 PRE_LO_DLY	Delay before de-asserting signal to low, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the HS signal to low.
15-0 PRE_HI_DLY	Delay before asserting signal to high, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the HS signal to high.

27.6.3.8 MIF MLPL control of Input Gating (IG) (MIF_MLPL_IG)

Offset

Register	Offset
MIF_MLPL_IG	30h

Function
IG gates all the input pins including clock of memory to save toggling power of the inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															SW_IG
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								MLPL_CTRL							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 SW_IG	Software control IG 0b - IG is 0. 1b - IG is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings 1 means the signal is 1 at the MLPL setting. This field is locked by LOCK_CFG field.

27.6.3.9 MIF Delay of IG (MIF_DLY_IG)

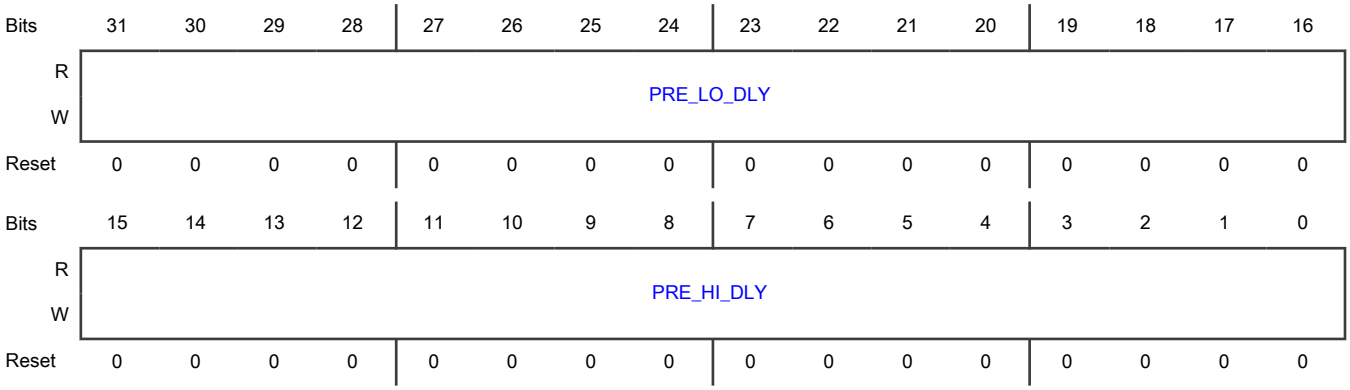
Offset

Register	Offset
MIF_DLY_IG	34h

Function

MIF Delay of IG register

Diagram



Fields

Field	Function
31-16 PRE_LO_DLY	Delay before de-asserting signal to low, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the IG signal to low.
15-0 PRE_HI_DLY	Delay before asserting signal to high, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the IG signal to high.

27.6.3.10 MIF MLPL control of STDBY (MIF_MLPL_STDBY)

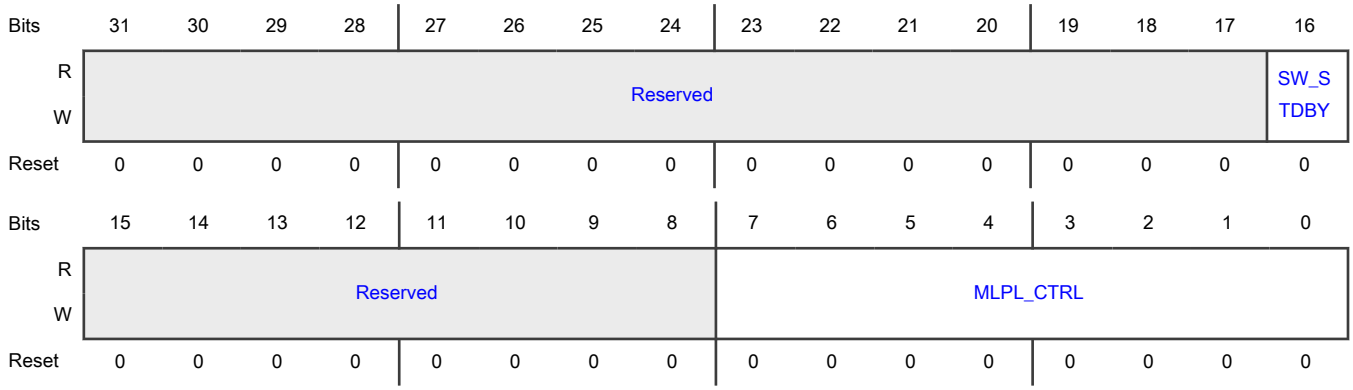
Offset

Register	Offset
MIF_MLPL_STDBY	40h

Function

To reduce the static power without changing the external supply, STDBY pin is provided. This pin gates the power supply to word-line final driver that is the leakiest part of the memory periphery. For the dual-rail memory, this mode reduces the static power of vddma supply. For large number of rows instances and consequently word-line final drivers, this mode can provide a good static power saving.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 SW_STDBY	Software control STDBY 0b - STDBY is 0. 1b - STDBY is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is logic 1 at the MLPL setting. This field is locked by LOCK_CFG field.

27.6.3.11 MIF Delay of STDBY (MIF_DLY_STDBY)

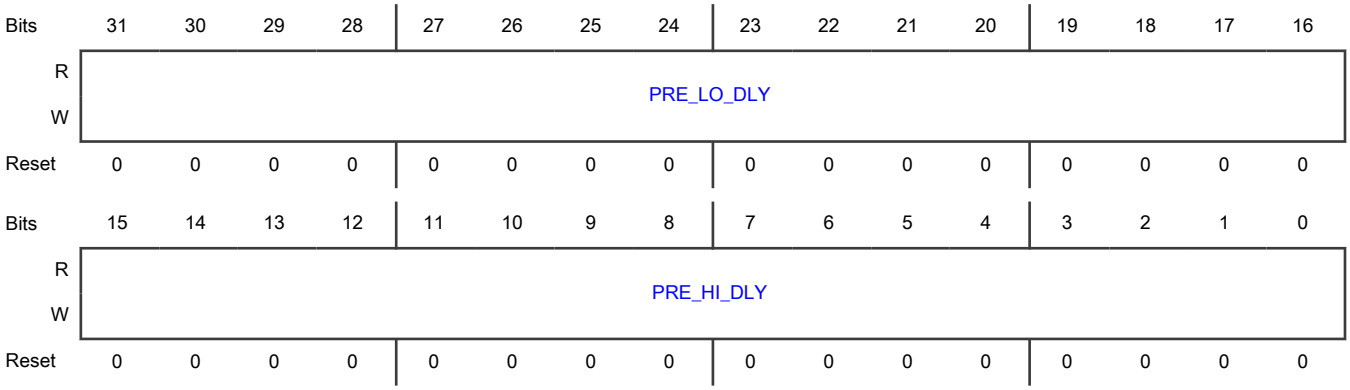
Offset

Register	Offset
MIF_DLY_STDBY	44h

Function

Controls the assertion delay of signal STDBY.

Diagram



Fields

Field	Function
31-16 PRE_LO_DLY	Delay before de-asserting signal to low, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the STDBY signal to low.
15-0 PRE_HI_DLY	Delay before asserting signal to high, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the STDBY signal to high.

27.6.4 MEM Type I with PSW register descriptions

27.6.4.1 src_mif_s28spregh_pswa memory map

CM7PLATFORM_TCM base address: 4446_4C00h

Offset	Register	Width (In bits)	Access	Reset value
4h	MPC Control (MIF_CTRL)	32	RW	0000_0000h
8h	MIF Status (MIF_STAT)	32	R	0000_0000h
10h	MIF MLPL control of LS (MIF_MLPL_LS)	32	RW	0000_0000h
14h	MIF Delay of LS (MIF_DLY_LS)	32	RW	0000_0000h
20h	MIF MLPL control of HS (MIF_MLPL_HS)	32	RW	0000_0002h
24h	MIF Delay of HS (MIF_DLY_HS)	32	RW	0000_0000h
30h	MIF MLPL control of Input Gating (IG) (MIF_MLPL_IG)	32	RW	0000_0010h
34h	MIF Delay of IG (MIF_DLY_IG)	32	RW	0000_0000h
40h	MIF MLPL control of STDBY (MIF_MLPL_STDBY)	32	RW	0000_0000h
44h	MIF Delay of STDBY (MIF_DLY_STDBY)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
50h	MIF MLPL control of SLEEP (MIF_MLPL_SLEEP)	32	RW	0000_00C0h
54h	MIF Delay of SLEEP (MIF_DLY_SLEEP)	32	RW	0002_0000h
60h	MIF MLPL control of array power down (MIF_MLPL_ARR_PDN)	32	RW	0000_00C0h
64h	MIF Delay of array high-fanout power switch (MIF_DLY_ARR_HF)	32	RW	0000_0002h

27.6.4.2 MPC Control (MIF_CTRL)

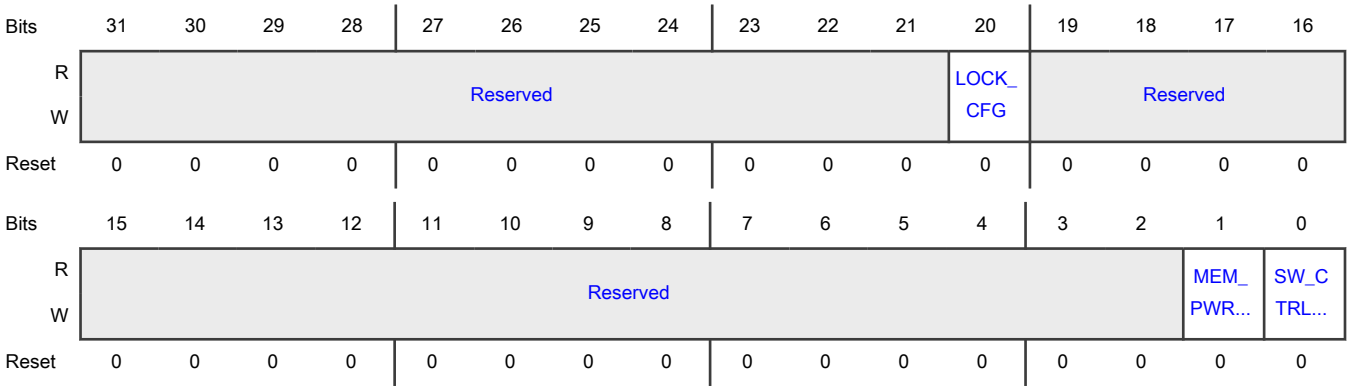
Offset

Register	Offset
MIF_CTRL	4h

Function

.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 LOCK_CFG	Configuration lock This field can lock itself and some fields of the registers in this module. If the field can be locked by LOCK_CFG, it will be shown in the field's description. Else the field cannot be locked by LOCK_CFG.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - The fields are not locked. 1b - The fields are locked.
19-2 —	Reserved
1 MEM_PWR_ST_EN	Memory power status will be considered when determining slice power status. This field should only be used for debug. So do not change it. 0b - Memory power status will not be considered when determining slice power status. 1b - Memory power status will be considered when determining slice power status.
0 SW_CTRL_PIN	Memory low power pins controlled by SW Selects how to control such type of memory's low power control signals. They are IG, LS, HS and STDBY. Selecting SW mode means use SW_IG, SW_LS, SW_HS and SW_STDBY fields in each MIF MLPL control registers to control relative signals directly. Otherwise CURRENT_MLPL value will be used to select 1 value from 8 values in each MLPL_CTRL fields to control the relative low power control signal. 0b - Use CURRENT_MLPL field to select MLPL_CTRL to control low power signal. 1b - Use SW_* field to control low power signal directly.

27.6.4.3 MIF Status (MIF_STAT)

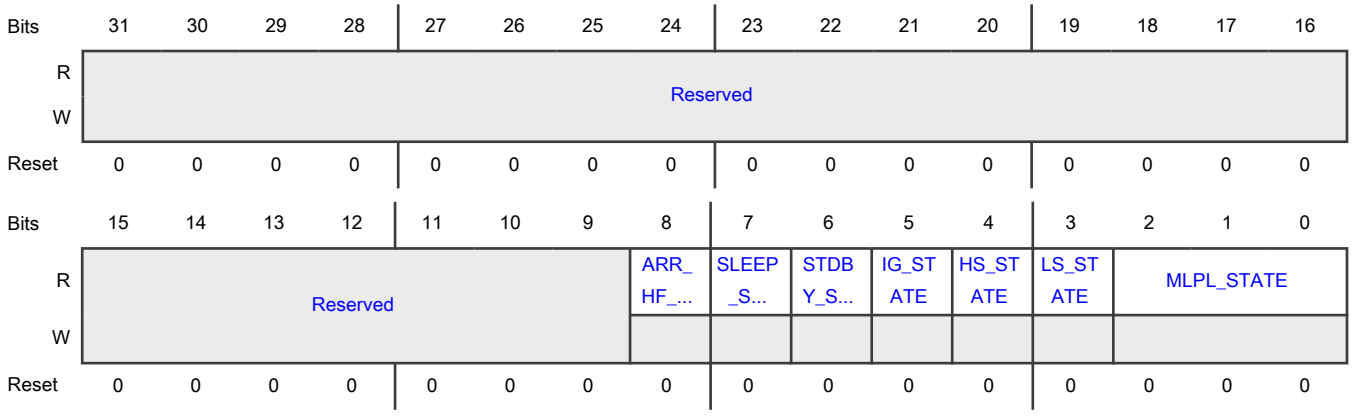
Offset

Register	Offset
MIF_STAT	8h

Function

.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 ARR_HF_STATE	ARR_HF_OFF status 0b - ARR_HS is 0. 1b - ARR_HS is 1.
7 SLEEP_STATE	SLEEP status 0b - SLEEP is 0. 1b - SLEEP is 1.
6 STDBY_STATE	Current state of STDBY 0b - STDBY is 0. 1b - STDBY is 1.
5 IG_STATE	Current state of IG 0b - IG is 0. 1b - IG is 1.
4 HS_STATE	Current state of HS 0b - HS is 0. 1b - HS is 1.
3 LS_STATE	Current state of LS 0b - LS is 0. 1b - LS is 1.
2-0 MLPL_STATE	Current state of CURRENT_MLPL

27.6.4.4 MIF MLPL control of LS (MIF_MLPL_LS)

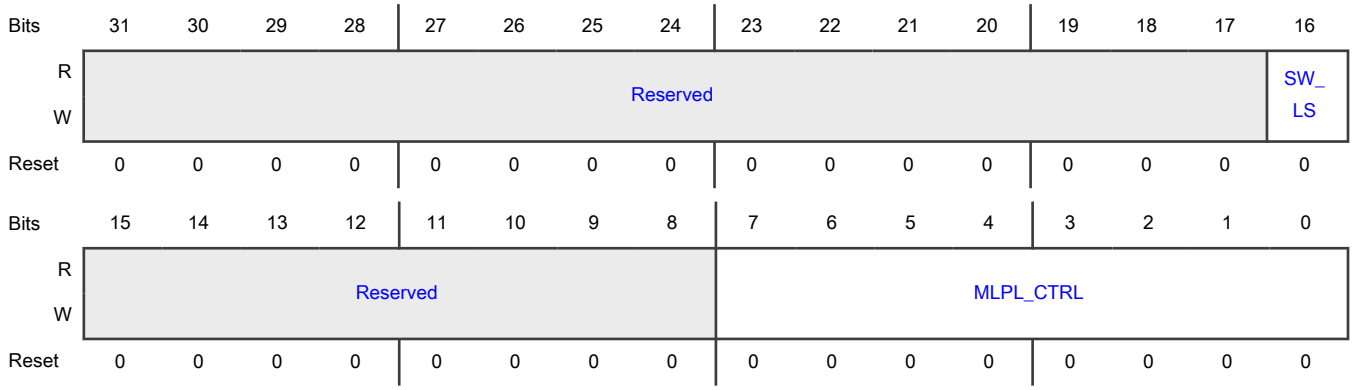
Offset

Register	Offset
MIF_MLPL_LS	10h

Function

Speed mode provides the flexibility to boost speed and gain dynamic power by controlling memory LS/HS pins depending on operating voltage. This option enables the user to operate memory in different speed modes. LS pin can be constraint to H only for low voltages and LS = L at higher voltage will boost speed and reduce dynamic power. In conjunction with LS = L, HS = H with specific voltage constraints will further boost speed.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 SW_LS	Software control LS 0b - LS is 0. 1b - LS is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is 1 at the MLPL setting. This field is locked by LOCK_CFG field.

27.6.4.5 MIF Delay of LS (MIF_DLY_LS)

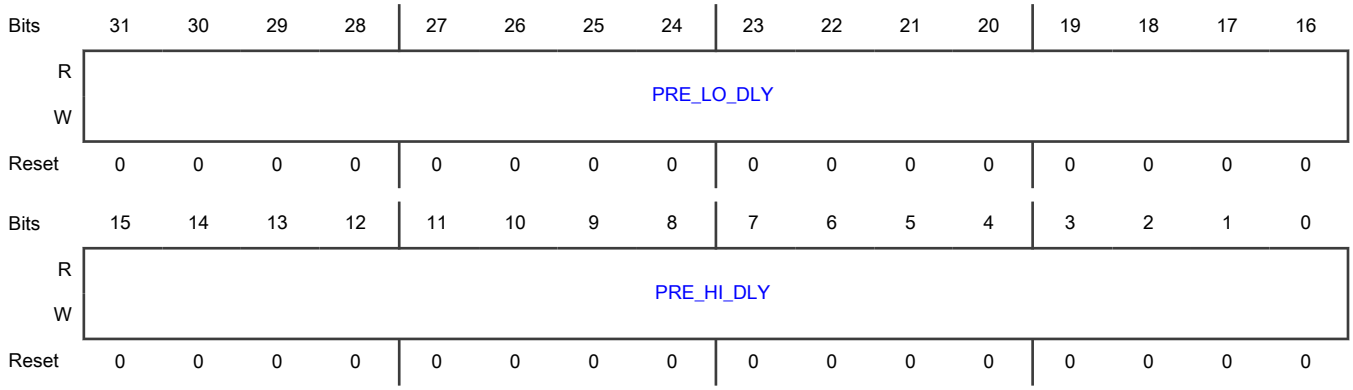
Offset

Register	Offset
MIF_DLY_LS	14h

Function

MIF Delay of LS register

Diagram



Fields

Field	Function
31-16 PRE_LO_DLY	Delay before de-asserting signal to low, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the LS signal to low.
15-0 PRE_HI_DLY	Delay before asserting signal to high, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the LS signal to high.

27.6.4.6 MIF MLPL control of HS (MIF_MLPL_HS)

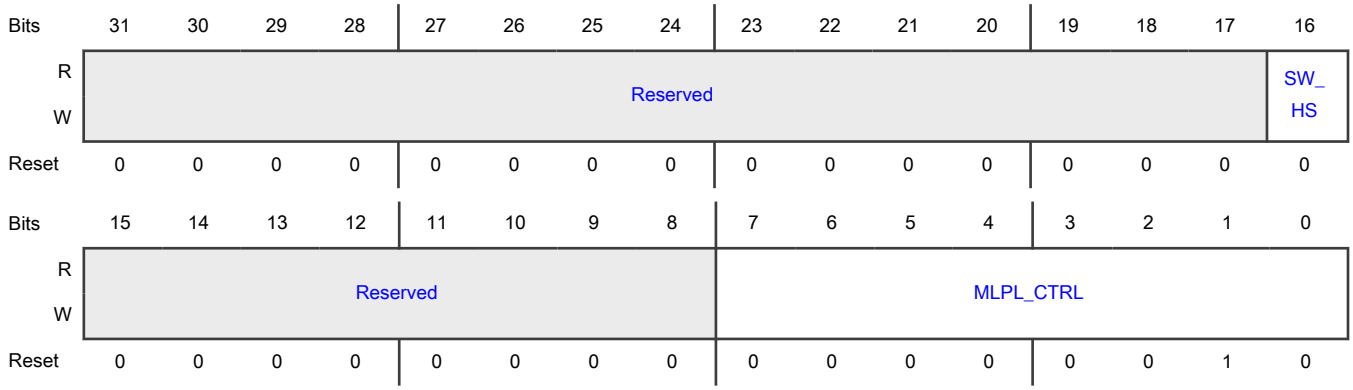
Offset

Register	Offset
MIF_MLPL_HS	20h

Function

Speed mode provides the flexibility to boost speed and gain dynamic power by controlling memory LS/HS pins depending on operating voltage. This option enables the user to operate memory in different speed modes. LS pin can be constraint to H only for low voltages and LS = L at higher voltage will boost speed and reduce dynamic power. In conjunction with LS = L, HS = H with specific voltage constraints will further boost speed.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 SW_HS	software control HS 0b - HS is 0. 1b - HS is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is 1 at the MLPL setting. This field is locked by LOCK_CFG field.

27.6.4.7 MIF Delay of HS (MIF_DLY_HS)

Offset

Register	Offset
MIF_DLY_HS	24h

Function

MIF Delay of HS register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PRE_LO_DLY															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PRE_HI_DLY															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 PRE_LO_DLY	Delay before de-asserting signal to low, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the HS signal to low.
15-0 PRE_HI_DLY	Delay before asserting signal to high, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the HS signal to high.

27.6.4.8 MIF MLPL control of Input Gating (IG) (MIF_MLPL_IG)

Offset

Register	Offset
MIF_MLPL_IG	30h

Function

IG gates all the input pins including clock of memory to save toggling power of the inputs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															SW_IG
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								MLPL_CTRL							
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 SW_IG	Software control IG 0b - IG is 0. 1b - IG is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is 1 at the MLPL setting. This field is locked by LOCK_CFG field.

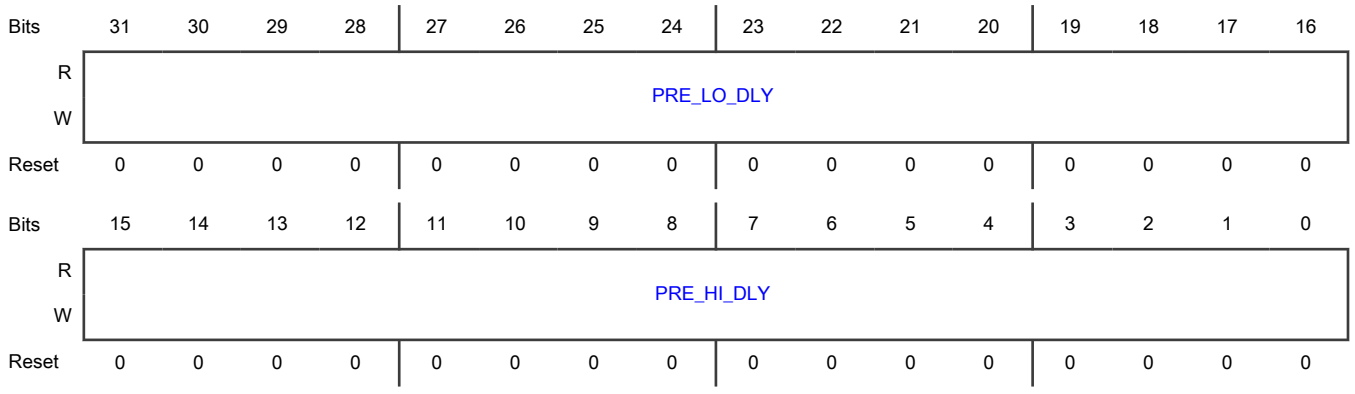
27.6.4.9 MIF Delay of IG (MIF_DLY_IG)

Offset

Register	Offset
MIF_DLY_IG	34h

Function
MIF Delay of IG register

Diagram



Fields

Field	Function
31-16	Delay before de-asserting signal to low, locked by LOCK_CFG field

Table continues on the next page...

Table continued from the previous page...

Field	Function
PRE_LO_DLY	This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the IG signal to low.
15-0 PRE_HI_DLY	Delay before asserting signal to high, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the IG signal to high.

27.6.4.10 MIF MLPL control of STDBY (MIF_MLPL_STDBY)

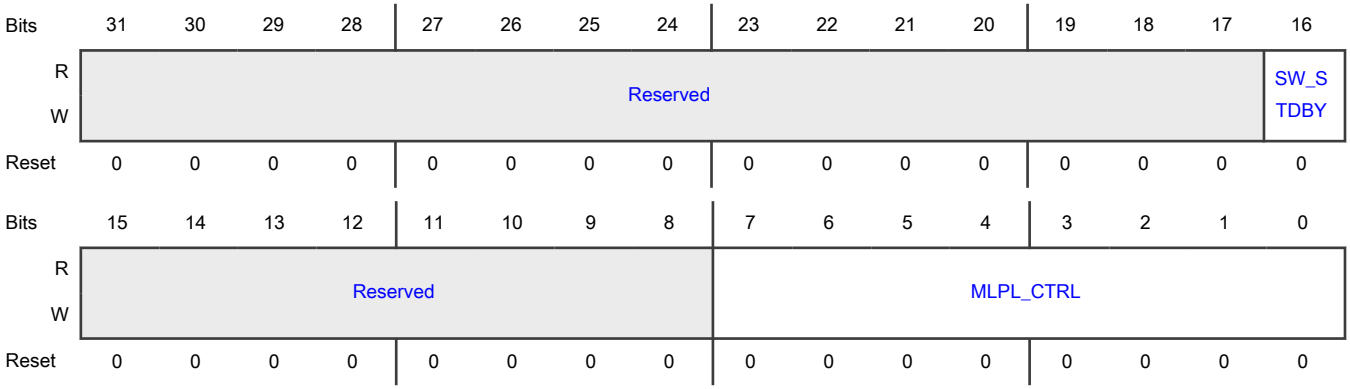
Offset

Register	Offset
MIF_MLPL_STDBY	40h

Function

To reduce the static power without changing the external supply, STDBY pin is provided. This pin gates the power supply to word-line final driver that is the leakiest part of the memory periphery. For the dual-rail memory, this mode reduces the static power of vddma supply. For large number of rows instances and consequently word-line final drivers, this mode can provide a good static power saving.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 SW_STDBY	Software control STDBY 0b - STDBY is 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - STDBY is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is 1 at the MLPL setting. This field is locked by LOCK_CFG field.

27.6.4.11 MIF Delay of STDBY (MIF_DLY_STDBY)

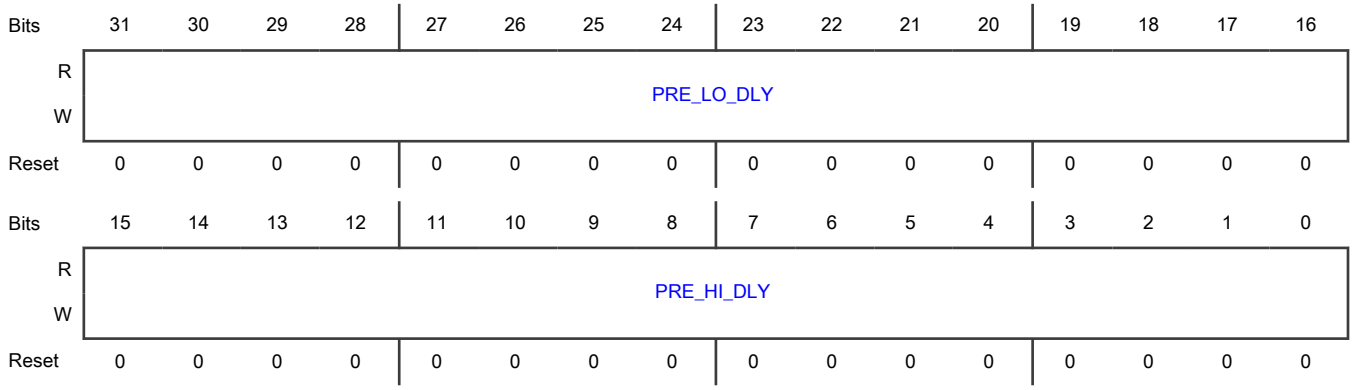
Offset

Register	Offset
MIF_DLY_STDBY	44h

Function

Controls the assertion delay of signal STDBY.

Diagram



Fields

Field	Function
31-16 PRE_LO_DLY	Delay before de-asserting signal to low, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the STDBY signal to low.
15-0 PRE_HI_DLY	Delay before asserting signal to high, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the STDBY signal to high.

27.6.4.12 MIF MLPL control of SLEEP (MIF_MLPL_SLEEP)

Offset

Register	Offset
MIF_MLPL_SLEEP	50h

Function

SLEEP changing to high must be earlier than ARR_HF changes to high. SLEEP changing to low must be later than ARR_HF changes to low.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															SW_SLEEP
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								MLPL_CTRL							
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 SW_SLEEP	Software control SLEEP When CM7 TCM memory needs to be powered down, SW must set this bit first, then set SW_ARR_PDN in MIF_MLPL_ARR_PDN. When CM7 TCM memory needs to be powered up, SW must clear this bit later then clear SW_ARR_PDN in MIF_MLPL_ARR_PDN. 0b - SLEEP is 0. 1b - SLEEP is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is 1 at the MLPL setting. This field is locked by LOCK_CFG field.

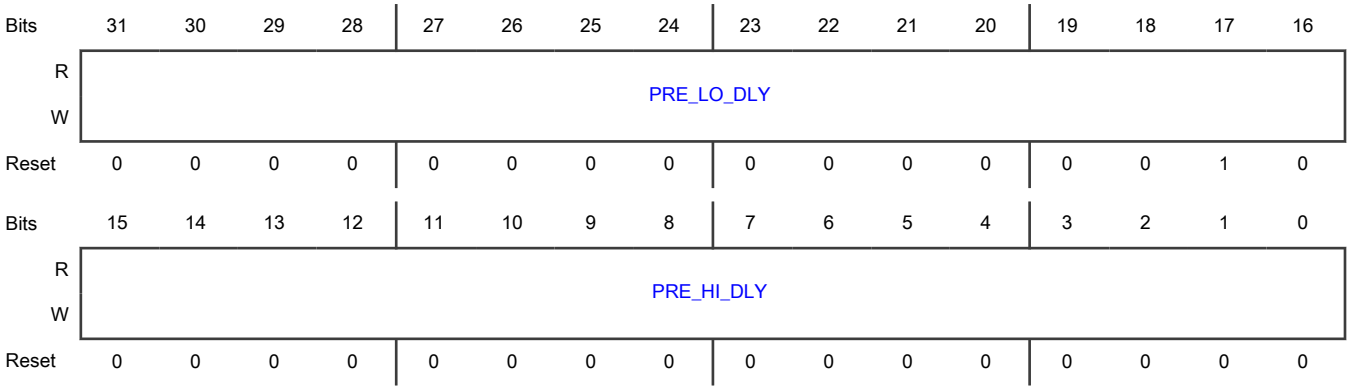
27.6.4.13 MIF Delay of SLEEP (MIF_DLY_SLEEP)

Offset

Register	Offset
MIF_DLY_SLEEP	54h

Function
MIF Delay of SLEEP register

Diagram



Fields

Field	Function
31-16 PRE_LO_DLY	Delay before de-asserting signal to low, locked by LOCK_CFG field. This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the SLEEP signal to low. SLEEP change to low must be later than ARR_HF changes to low. So delay counter must be bigger than PRE_LO_DLY in MIF_DLY_ARR_HF.
15-0 PRE_HI_DLY	Delay before asserting signal to high, locked by LOCK_CFG field. This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the SLEEP signal to high. SLEEP changes to high must be earlier than ARR_HF changes to high. So delay counter must be smaller than PRE_HI_DLY in MIF_DLY_ARR_HF.

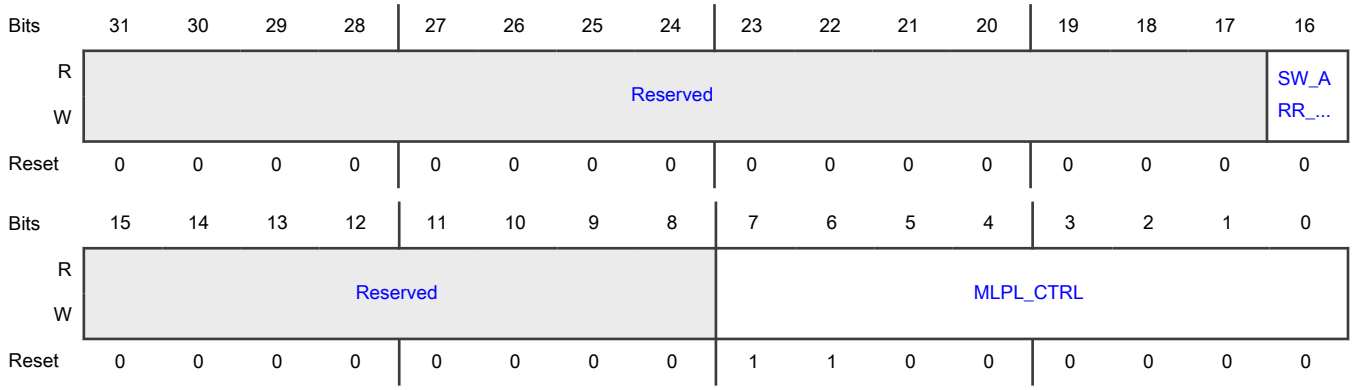
27.6.4.14 MIF MLPL control of array power down (MIF_MLPL_ARR_PDN)

Offset

Register	Offset
MIF_MLPL_ARR_PDN	60h

Function
1 means the memory array power off.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 SW_ARR_PDN	Software control arr pdn 0b - ARR_PDN is 0. 1b - ARR_PDN is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is 1 at the MLPL setting. This field is locked by LOCK_CFG field.

27.6.4.15 MIF Delay of array high-fanout power switch (MIF_DLY_ARR_HF)

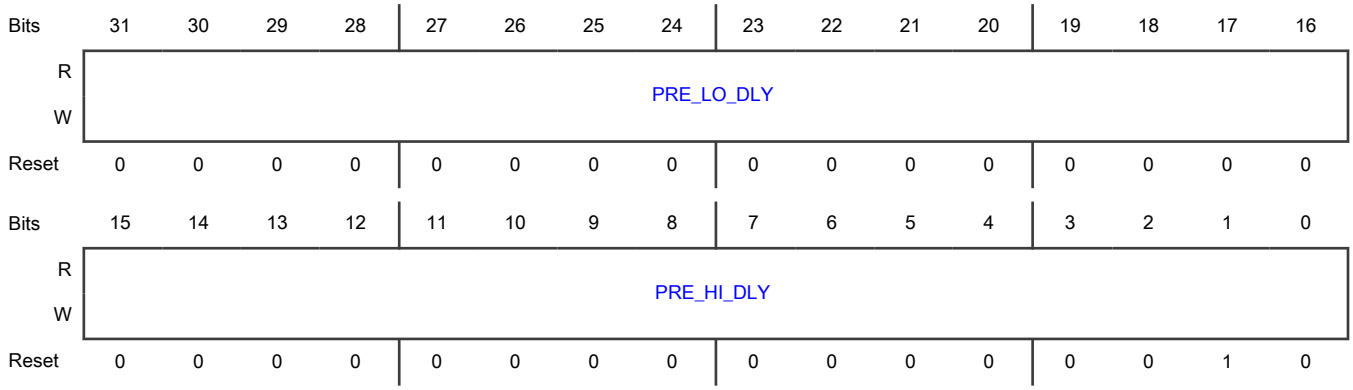
Offset

Register	Offset
MIF_DLY_ARR_HF	64h

Function

.

Diagram



Fields

Field	Function
31-16 PRE_LO_DLY	Delay before turn on the high-fanout power switch, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the ARR_PDN signal to low.
15-0 PRE_HI_DLY	Delay before turn off the high-fanout power switch, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the ARR_PDN signal to high.

27.6.5 MEM Type II register descriptions

27.6.5.1 src_mif_ln28fdsoi_spllram memory map

AON_MIF_LN28FDSOI_SPLLRAM base address: 4446_2400h

MEGA_MIF_LN28FDSOI_SPLLRAM base address: 4446_3400h

NETC_MIF_LN28FDSOI_SPLLRAM base address: 4446_3C00h

WAKEUP_MIF_LN28FDSOI_SPLLRAM base address: 4446_2C00h

Offset	Register	Width (In bits)	Access	Reset value
4h	MPC Control (MIF_CTRL)	32	RW	0000_0000h
8h	MIF Status (MIF_STAT)	32	R	0000_0020h
20h	MIF MLPL control of IG (MIF_MLPL_IG)	32	RW	0000_0010h
24h	MIF Delay of IG (MIF_DLY_IG)	32	RW	0000_0000h
30h	MIF MLPL control of WLPD (MIF_MLPL_WLPD)	32	RW	0000_0006h
34h	MIF Delay of WLPD (MIF_DLY_WLPD)	32	RW	0000_0000h
40h	MIF MLPL control of PD_B (MIF_MLPL_PD_B)	32	RW	0000_001Bh
44h	MIF Delay of PD_B (MIF_DLY_PD_B)	32	RW	0000_0000h

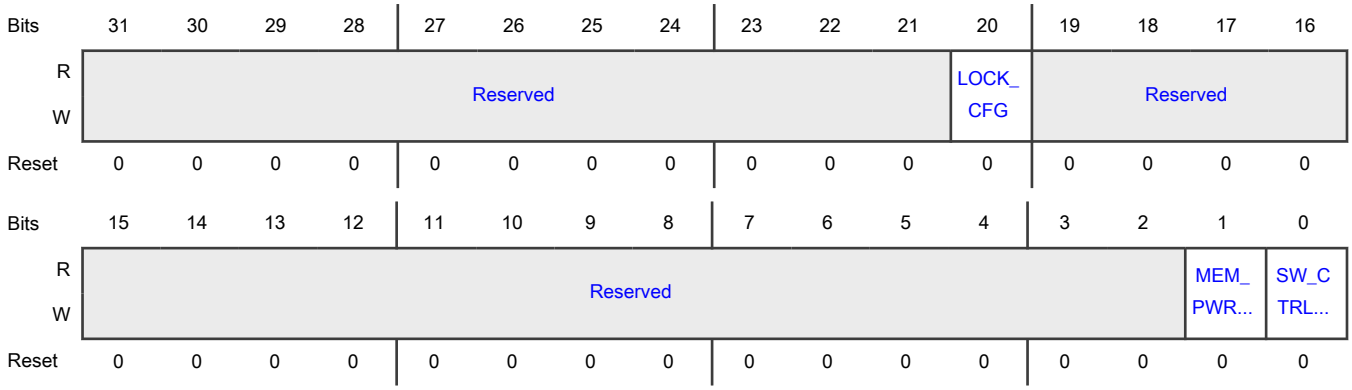
27.6.5.2 MPC Control (MIF_CTRL)

Offset

Register	Offset
MIF_CTRL	4h

Function
MPC Control register

Diagram



Fields

Field	Function
31-21 —	Reserved
20 LOCK_CFG	Configuration lock This field can lock itself and some fields of the registers in this module. If the field can be locked by LOCK_CFG, it will be shown in the field's description. Else the field cannot be locked by LOCK_CFG. 0b - The fields are not locked. 1b - The fields are locked.
19-2 —	Reserved
1 MEM_PWR_ST_EN	Memory power status will be considered when determining slice power status. This field should only be used for debug. So do not change it. 0b - Memory power status will not be considered when determining slice power status. 1b - Memory power status will be considered when determining slice power status.
0	Memory low power pins controlled by SW

Table continues on the next page...

Table continued from the previous page...

Field	Function
SW_CTRL_PIN	Selects how to control such type of memory's low power control signals. They are IG, WLPD and PD_B. Selecting SW mode means use SW_IG, SW_WLPD and SW_PD_B fields in each MIF_MLPL control registers to control relative signals directly. Otherwise CURRENT_MLPL value will be used to select 1 value from 8 values in each MLPL_CTRL fields to control the relative low power control signal. 0b - Use CURRENT_MLPL field to select MLPL_CTRL to control low power signal. 1b - Use SW_* field to control low power signal directly.

27.6.5.3 MIF Status (MIF_STAT)

Offset

Register	Offset
MIF_STAT	8h

Function

MIF Status register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										PD_B_ST...	WLPD_ST...	IG_STATE	MLPL_STATE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5 PD_B_STATE	Current state of PD_B 0b - PD_B is 0. 1b - PD_B is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 WLPD_STATE	Current state of WLPD 0b - WLPD is 0. 1b - WLPD is 1.
3 IG_STATE	Current state of IG 0b - IG is 0. 1b - IG is 1.
2-0 MLPL_STATE	Current state of CURRENT_MLPL

27.6.5.4 MIF MLPL control of IG (MIF_MLPL_IG)

Offset

Register	Offset
MIF_MLPL_IG	20h

Function

The IG signal will disable the input buffers for cen_b, wen, a, d, iwen. This eliminates any active power associated with these input pins toggling when the memory is de-selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															SW_IG
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								MLPL_CTRL							
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-17 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 SW_IG	Software control IG 0b - IG is 0. 1b - IG is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is 1 at the MLPL setting. This field is locked by LOCK_CFG field.

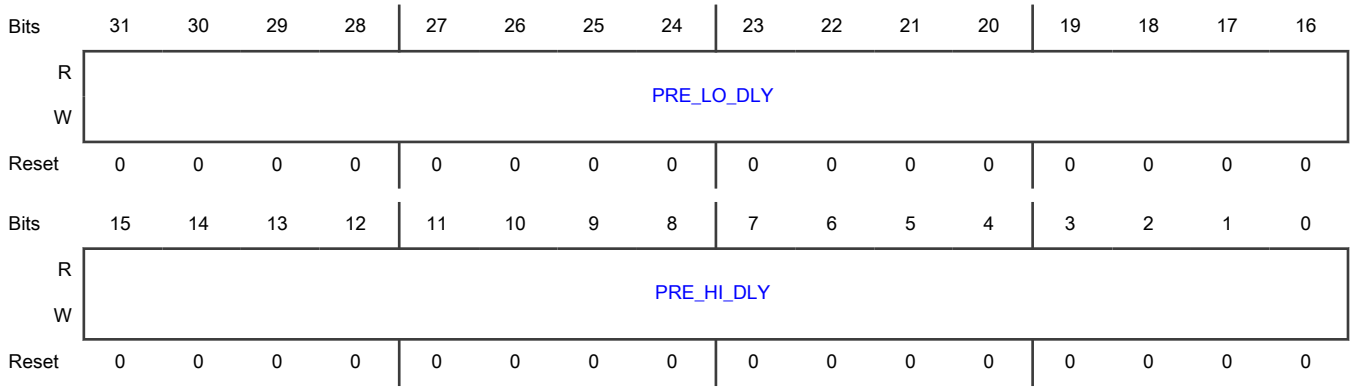
27.6.5.5 MIF Delay of IG (MIF_DLY_IG)

Offset

Register	Offset
MIF_DLY_IG	24h

Function
MIF Delay of IG register

Diagram



Fields

Field	Function
31-16 PRE_LO_DLY	Delay before de-asserting signal to low, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the IG signal to low.
15-0	Delay before asserting signal to high, locked by LOCK_CFG field

Table continues on the next page...

Table continued from the previous page...

Field	Function
PRE_HI_DLY	This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the IG signal to high.

27.6.5.6 MIF MLPL control of WLPD (MIF_MLPL_WLPD)

Offset

Register	Offset
MIF_MLPL_WLPD	30h

Function

The word-line power down feature will gate off the power supply to the word-line drivers. When asserted, wlpd=1, the word-line drivers will be power gated and the memory will be prevented from performing any normal functional mode operations. Assertion of wlpd enables leakage power savings on the vdda supply.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															SW_ WLPD
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								MLPL_CTRL							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-17 —	Reserved
16 SW_WLPD	Software control WLPD 0b - WLPD is 0. 1b - WLPD is 1.
15-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is 1 at the MLPL setting. This field is locked by LOCK_CFG field.

27.6.5.7 MIF Delay of WLPD (MIF_DLY_WLPD)

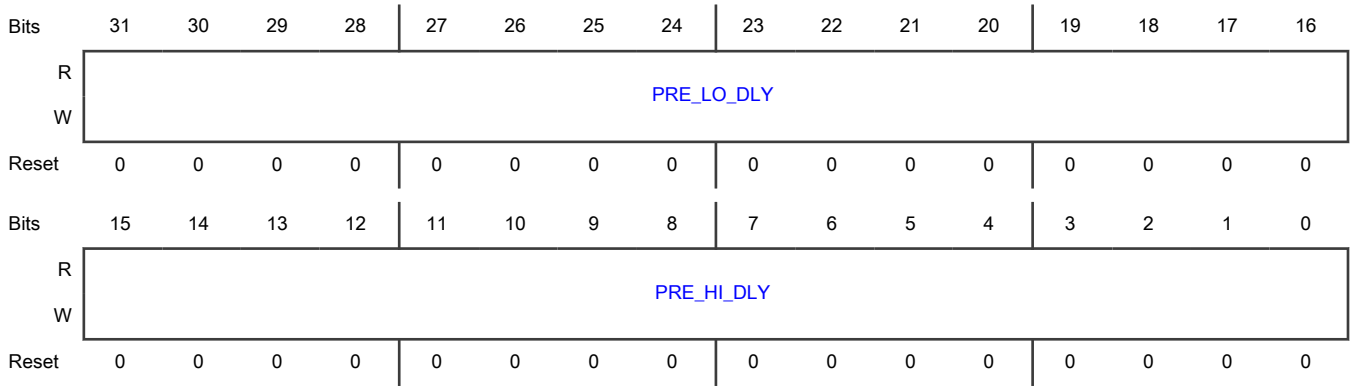
Offset

Register	Offset
MIF_DLY_WLPD	34h

Function

MIF Delay of WLPD register

Diagram



Fields

Field	Function
31-16 PRE_LO_DLY	Delay before de-asserting signal to low, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the WLPD signal to low.
15-0 PRE_HI_DLY	Delay before asserting signal to high, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the WLPD signal to high.

27.6.5.8 MIF MLPL control of PD_B (MIF_MLPL_PD_B)

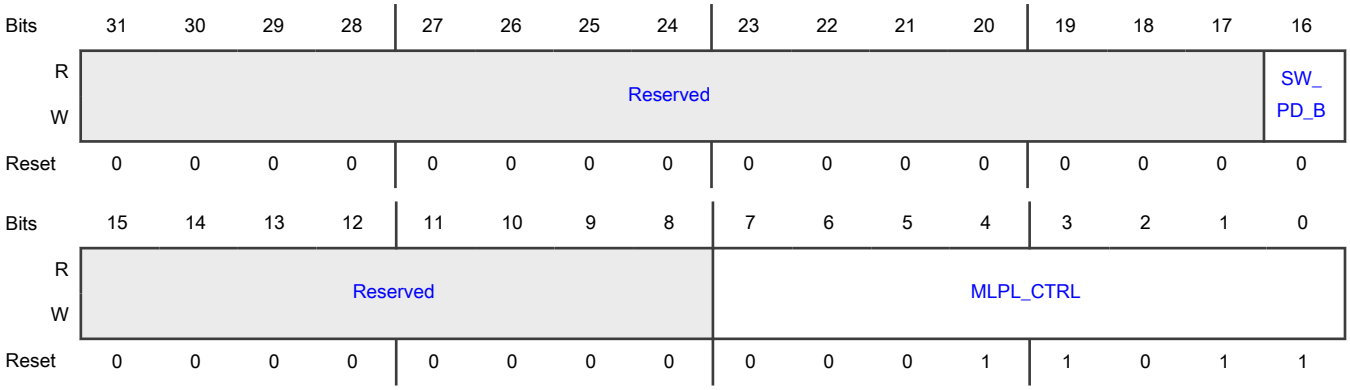
Offset

Register	Offset
MIF_MLPL_PD_B	40h

Function

In Power-down Mode, additional static power savings may be obtained by asserting the active-low power-down signal: pd_b(1'b0).The pd_b input has precedence over all other memory inputs.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 SW_PD_B	software control PD_B 0b - PD_B is 0. 1b - PD_B is 1.
15-8 —	Reserved
7-0 MLPL_CTRL	Signal behavior at 8 different MLPL settings Value 1 means the signal is 1 at the MLPL setting. This field is locked by LOCK_CFG field.

27.6.5.9 MIF Delay of PD_B (MIF_DLY_PD_B)

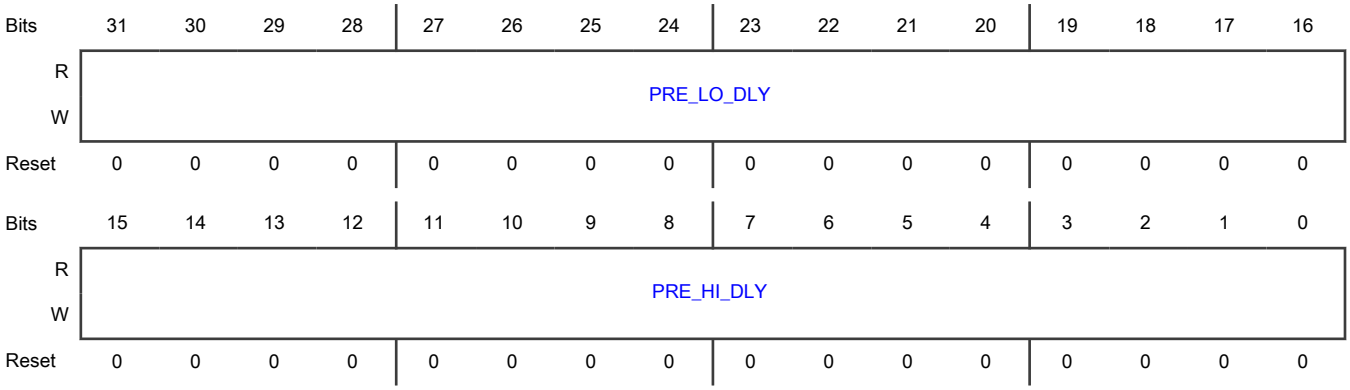
Offset

Register	Offset
MIF_DLY_PD_B	44h

Function

PD_B signal must be 0 before power off, and must be 1 after power on.

Diagram



Fields

Field	Function
31-16 PRE_LO_DLY	Delay before de-asserting signal to low, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the PD_B signal to low.
15-0 PRE_HI_DLY	Delay before asserting signal to high, locked by LOCK_CFG field This field is only valid when SW_CTRL_PIN is 0. It controls the delay time from SRC start MIX slice power switch to change the PD_B signal to high.

Chapter 28
External Memory Controllers

28.1 Overview

This section lists the external memory interfaces and controllers.

This chip has these external memory interfaces and controllers:

- Smart External Memory Controller (SEMC)
- eSD/eMMC/SDIO Interface
- Flexible Serial Peripheral Interface (FlexSPI)
- Flexible Serial Peripheral Interface Follower (FlexSPI_FLR)
- FPGA interface (SRAMC)

28.2 Smart External Memory Controller (SEMC)

The SEMC is a multi-standard memory controller optimized for both high-performance and low pin-count. It can support multiple external memories in the same application with shared address and data pins. The interface supported includes SDRAM, NOR Flash, SRAM and NAND Flash, as well as 8080 display interface.

The key features of the SEMC include:

- Support up to 12 memory regions up to 512 Mbit per region which can be configured to be memory space of SDRAM, NOR Flash, NAND Flash, SRAM or 8080 display frame buffer
- Chip Select (CS) signals
- Support Address Latch Enable (ALE)
- SDRAM interface
  - Supports both 8-bit, 16-bit, and 32-bit modes
  - Max 200 MHz
  - Up to 512Mb per each Chip Select (CS) and up to 4x CS
- NOR Flash and SRAM interface
  - Supports both 8-bit and 16-bit modes
  - Asynchronous and Synchronous mode
  - Address and Data Multiplexing (ADM)
  - Maximum 128Mb per each Chip Select (CS)
  - Support NOR Flash program via IPBus configuration
- 8080 display interface
  - Supports both 8/16/24-bit modes
  - Up to 100MHz
  - Without tearing effect support
- NAND Flash
  - 8/16-bit NAND FLASH
  - Asynchronous and Synchronous mode
  - Only supports device which is capable of cache read/write operations

  - Only supports page based operation
- Dynamic I/O sharing which enables multiple external memory device in parallel
- Multi-device access pattern scheduler
  - 8 entries scheduler
  - QoS priority, latency and efficiency adjustable arbitration scheme
- SDRAM access pattern optimization
  - 8 entries command reordering queue
  - QoS priority, latency and efficiency adjustable arbitration scheme

28.3 eMMC/eSD/SDIO

This chip has two Ultra Secured Digital Host Controller (uSDHC) modules for SD/eMMC interface. It provides the interface between the host system and the SD/SDIO/MMC cards.

The key features include:

- Support SD/SDIO standard, up to version 3.0
- Support MMC standard, up to version 5.1
- Support 3.3V and 1.8V operation.
- uSDHC1 supports maximum 4bit interface, uSDHC2 supports maximum 8bit interface

28.4 Flexible Serial Peripheral Interface (FlexSPI)

i.MX RT1180 has two FlexSPI host controllers - FlexSPI1 and FlexSPI2. Each FlexSPI host controller supports two SPI ports - A and B. Each port supports up two chip select outputs. Each port of FlexSPI1 supports Single/Dual/Quad/Octal mode data transfer (1/2/4/8 bidirectional data lines), while each port of FlexSPI2 supports Single/ Dual/Quad mode data transfer (1/2/4 bidirectional data lines).

The key features include:

- Support both SDR mode and DDR mode
- Support Individual/Parallel mode
- HyperBus device (HyperFlash/HyperRAM)
- Support On-The-Fly AES Decryption (OTFAD) with 4 keys

28.5 Flexible Serial Peripheral Interface Follower (FlexSPI_FLR)

The FlexSPI follower provides a FlexSPI interface, so that the i.MX RT1180 can act as a FlexSPI device. The FlexSPI follower module acts as a gasket between an external FlexSPI and the AXI bus. As AXI leader, the FlexSPI follower can access all memory mapped registers and memory in the i.MX RT1180.

The FlexSPI module provides basic hardware to process access control by high level software or protocol. By default, the FlexSPI AXI leader is disabled, external FlexSPI leader can access limited registers in the FlexSPI follower module. The FlexSPI follower module contains three kind of registers:

- GCR: General Control Register, read/write by IPS bus
  - Defines Command Set, Base Address, dummy cycles that necessary for FlexSPI transfer
  - Enable AXI leader
- CSR: Control and Status Register, read/write by IPS bus, read by FlexSPI leader via FlexSPI register read command
  - Indicate status of FlexSPI follower module, transfer fail, busy, AXI write in progress

  - Handshake, such as grant or reject FlexSPI leader access to RT1180 memory mapped register or SRAM
- GPR: General purpose Register, read by IPS bus, clean by Control and Status Register, read/write by FlexSPI leader via FlexSPI register read/write command
  - Can trigger interrupt once FlexSPI leader write GPR registers
  - 8 words in size
  - It depends on software or high level protocol to define the detailed format of SRAMCR0 and SRAMCR1, such as, but not limited to:
    - FlexSPI leader authentication
    - Define Base address that FlexSPI leader request to access
    - Define Command Set that FlexSPI leader should use

28.6 FPGA SRAM Interface (SRAMC)

This chip provides a memory mapped SRAMC to CM7 AHBP to achieve fast access:

- Support both A/D mux or non-A/D mux interfacing with FPGA
- Support Async SRAM timing
- WAIT signal is not supported
- 4 CS signal, each CS cover 128KB memory space, total 512KB
- Memory mapped, start from 0x43880000
- Support 8/16 mode

Chapter 29
Smart External Memory Controller (SEMC)

29.1 Chip-specific SEMC Information

Table 183. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

29.2 Overview

The SEMC is a multi-standard memory controller optimized for both high-performance and low pin-count. It can support multiple external memories in the same application with shared address and data pins. The interfaces supported include SDRAM, NOR Flash, SRAM, NAND Flash, and 8080 display interface.

29.2.1 Block Diagram

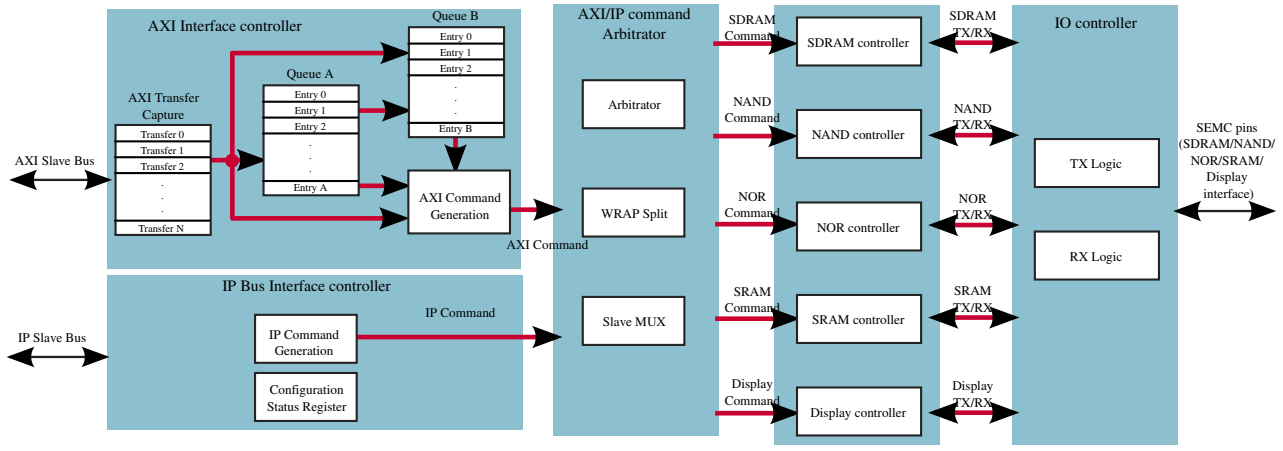


Figure 101. SEMC Block Diagram

29.2.2 Features

The SEMC includes the following features:

- Memory space:
  - Supports 12 memory regions:
    - Region #0 for SDRAM CS0 device
    - Region #1 for SDRAM CS1 device
    - Region #2 for SDRAM CS2 device
    - Region #3 for SDRAM CS3 device
    - Region #4 and #8 for NAND device (IP command and AXI command use different region to avoid large region address range on AXI interface)
    - Region #5 for NOR device
    - Region #6 for SRAM CS0 device
    - Region #7 for 8080 display frame buffer
    - Region #9 for SRAM CS1 device
    - Region #10 for SRAM CS2 device
    - Region #11 for SRAM CS3 device
  - Each region is configurable for base address, memory size and valid bit
- AXI slave interface
  - Supports 16 outstanding transfers (total including both reads and writes)
    - Supports 8 outstanding write transfers
    - Supports 16 outstanding read transfers
  - Multi-device access pattern optimization with Queue A
    - 8 entries command reordering in Queue A
    - Adjustable arbitration scheme based on QoS priority, latency and efficiency

  - SDRAM access pattern optimization with Queue B
    - 8 entries command reordering in Queue B
    - Adjustable arbitration scheme based on QoS priority, latency and efficiency
  - Memory-mapped AXI interface for read and write access
  - Supports 32/16/8 bit access
- IP Bus interface
  - Internal configuration register access
  - Trigger device access such as device initialization/configuration/read/write (IP command)
- SDRAM interface
  - Supports 8/16/32 bit modes
  - Supports 4 Chip Select (CS) and each CS up to 1024Mb
- NAND Flash interface
  - Supports 8/16 bit modes
  - Supports hardware ECC
  - Supports device which is capable of cache read/write operations
  - Supports page based read/write operation
  - Supports 32 bit transfer size for AXI write operation
  - Up to 512GB
- NOR Flash interface
  - Supports 8/16 bit modes
  - Supports ADMUX, AADM and Non-ADMUX modes
  - NOR Flash write access is supported only by IP command
  - Up to 4096Mb memory size

NOTE

- For 16 bit devices, up to 4096Mb memory size
- For 8 bit devices, up to 2048Mb memory size

- SRAM interface
  - Supports SRAM and Pseudo SRAM
  - Supports 8/16 bit modes
  - Supports ADMUX, AADM and Non-ADMUX modes
  - Up to 4 Chip Select (CS)
  - Up to 4096Mb memory size

NOTE

- For 16 bit devices, up to 4096Mb memory size
- For 8 bit devices, up to 2048Mb memory size

- 8080 display interface
  - Supports 8/16 bit modes
  - No tearing effect support

- Misc
  - Dynamic I/O sharing which enables multiple external memory device in parallel
  - Up to 7 Chip Select (CS) signals
  - Supports Address Latch Enable (ALE) for SRAM/NOR device

29.3 Functional Description

The following sections describe functional details of the SEMC module.

29.3.1 Operations

The following sections describe operations of the SEMC module.

29.3.1.1 Modes of Operation

The SEMC supports the following operation modes:

- Module Disable mode

  In module disable mode, the internal clock is gated for low power, but access to configuration and status registers is available.

  This mode is entered by setting MCR[MDIS], and exited by clearing MCR[MDIS]. Software should poll STS0[IDLE] to make sure there are no pending transactions before entering module disable mode.

- Stop mode

  When the system requires the SEMC to enter stop mode, the SEMC waits for all transactions to complete (STS0[IDLE]=1) and returns acknowledge handshake to system. The system can gate off the SEMC clock after the stop acknowledge handshake is returned. There is no clock gating within the SEMC module.

  The SEMC exits this mode immediately when the system stop mode request is negated.

- Normal mode

  Access to external devices and internal registers are available in normal mode. The SEMC clock is not gated internally.

29.3.1.2 Device Access by AXI Command

Devices can be accessed by AXI bus access directly. AXI command is the device access sequence triggered by AXI bus access. The device access sequence for AXI commands is described in the following sections.

NOTE

- The AXI bus memory map for each device is defined by Base Registers.
- There is no software configuration or polling needed for AXI command.
- AXI command is the first choice to access device memory. IP command is suitable for other access, such as device register/OTP space access, device initialization, etc.
- IP command can be used to access memory like AXI command, but it is not recommended if AXI command is available. IP command access is low efficiency.

29.3.1.2.1 Bus Master Control Registers (BMCRn) Configuration

BMCRn can be used to balance external memory access efficiency, urgency and latency based on the requirements of a specific application.

29.3.1.2.1.1 BMCR0 Register Configuration

For Queue A controlled by BMCR0, the arbitration logic adopts a weight based algorithm that assigns each command in the reordering queue with a final score.

The arbitration logic calculates a SCORE for each command based on the formula below, where the weighting factor for each parameter is controlled by the BMCR0 configuration. The command with the highest SCORE is served first.

SCORE = QOS*WQOS + AGE*WAGE/4 + WSH + WRWS

1. QOS stands for AxQOS of AXI bus, and WQOS is the weight factor of QOS.
2. AGE stands for the wait period for each command in queue, and WAGE is the weight factor of AGE.
3. WSH stands for weight of slave hit without read/write switch scenario.
4. WRWS stands for weight of slave hit with read/write switch scenario.

Recommend to set BMCR0 with 0x0 for applications that require restrict sequence of transactions, such as stack is allocated in SDRAM. The write and read transactions to the same address might be reordered if transactions in the queue is reordered, and it might crash the routine. WQOS can be non-zero value if needed, meanwhile WAGE must be non-zero value as well. In any case, WBR, WRWS, and WPH should be 0x0.

29.3.1.2.1.2 BMCR1 Register Configuration

For Queue B controlled by BMCR1, the arbitration logic adopts a weight based algorithm that assigns each command in the reordering queue with a final score.

The arbitration logic calculates SCORE for each SDRAM command based on the formula below. The command with the highest SCORE is served first.

SCORE = QOS*WQOS + AGE*WAGE/4 + WPH + WBR + WRWS

1. QOS stands for AxQOS of AXI bus, and WQOS is the weight factor of QOS.
2. AGE stands for the wait period for each command in queue, and WAGE is the weight factor of AGE.
3. WPH stands for weight of page hit scenario.
4. WBR stands for weight of bank rotation/switch scenario.
5. WRWS stands for weight of slave hit without read/write switch scenario.

Recommend to set BMCR1 with 0x0 for applications that require restrict sequence of transactions, such as stack is allocated in SDRAM. The write and read transactions to the same address might be reordered if transactions in the queue is reordered, and it might crash the routine. WQOS can be non-zero value if needed, meanwhile WAGE must be non-zero value as well. Anyway, WRWS, WBR and WPH should be 0x0.

29.3.1.3 Device Access by IP Command

Devices can be accessed (initialized/configured/read/written) by IP bus access also. IP command is the device access sequence triggered by IP bus access. The device access sequence for IP command is described in the following sections.

Device access can be triggered by IP command using the following steps:

1. Set IPCR0/IPCR1/IPCR2/IPTXDAT registers for device address, data size, write mask and write data.
2. Set IPCMD register with valid command code (IPCMD[CMD]) and command key (IPCMD[KEY]).
3. Wait for IP command done (INTR[IPCMDDONE] sets).
4. Read the device data from IPRXDAT register. This is for read access only.

NOTE

- If device address or command code setting is not valid, an IP command error interrupt is generated (INTR[IPCMDERR]) and no device access occurs.
- If another IP command is triggered when the previous IP command has not finished yet, an IP command error interrupt is generated.
- IPCR0/IPCR1/IPCR2/IPTXDAT registers are protected when an IP command is in progress. Any write access to these registers is ignored until the IP command completes.
- The IPCR0 slave address field must be configured even if the memory does not require an address for the selected command (such as WRITE ENABLE command). The SEMC determines the slave device by decoding the device address.
- The slave device is limited to NAND when IPCMD[KEY] is 0x5AA5.

29.3.1.4 SDRAM Controller Operations

This section describes the operations of SDRAM controller.

29.3.1.4.1 SDRAM Address Multiplexing

SDRAMs use a multiplexed address split into rows, columns, and banks. Figure 102 shows how the bits of the linear address sent to the SEMC are split up into the row, column, and bank addresses sent to the SDRAM memory device.

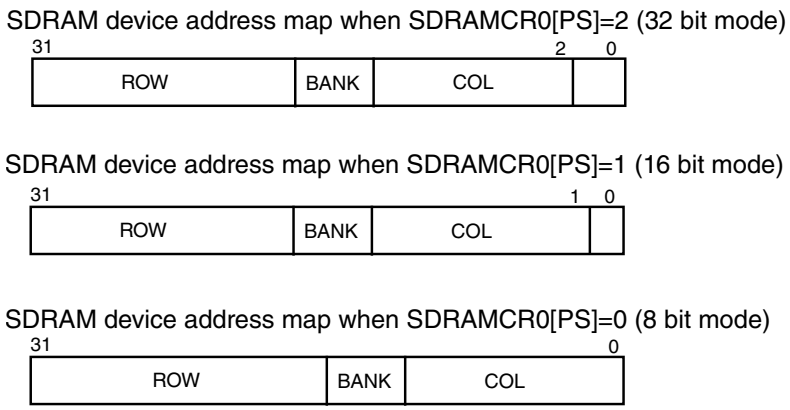


Figure 102. SDRAM Device Address Multiplexing

NOTE

- SDRAM address consists of row, bank and column address bits.
- Column address bits offset is bit 0 if SDRAMCR0[PS]=0, bit 1 if SDRAMCR0[PS]=1, bit 2 if SDRAMCR0[PS]=2. Column address bit width is determined by SDRAMCR0[COL] and SDRAMCR0[COL8].
- Bank address bits offset is determined by SDRAMCR0[PS], SDRAMCR0[COL] and SDRAMCR0[COL8]. Bank address bit width is determined by SDRAMCR0[BANK2].
- Row address bits offset is determined by SDRAMCR0[PS], SDRAMCR0[COL], SDRAMCR0[COL8] and SDRAMCR0[BANK2]. Row address bit width is determined by external SDRAM device.
- SDRAM device base address on SoC is configured by BR0, BR1, BR2 and BR3 registers.

29.3.1.4.2 SDRAM Access by IP Command

Table 184 lists the SDRAM access types that are triggered by IP commands.

Table 184. SDRAM IP Command

Command	Code	Comment
EXTENDED MODE REGISTER SET	0x5	Writes the extended mode register inside the SDRAM memory
DEEP POWER DOWN	0x6	Puts the SDRAM into deep power down mode
READ	0x8	Reads from the SDRAM memory
WRITE	0x9	Writes to the SDRAM memory
MODE REGISTER SET	0xA	Writes the mode register inside the SDRAM memory
ACTIVE	0xB	Sends a new bank and row address to the memory
AUTO REFRESH	0xC	Auto refresh
SELF REFRESH	0xD	Puts the SDRAM into low power data retention mode. The memory cannot be accessed while in self refresh mode.
PRECHARGE	0xE	Closes the currently active row for the specified bank
PRECHARGE ALL	0xF	Closes the active rows for all banks

29.3.1.4.2.1 READ

When an IP command is triggered with command code - READ, the SEMC will send a READ command to the SDRAM device.

Figure 103 shows the sequence for IP command - READ.

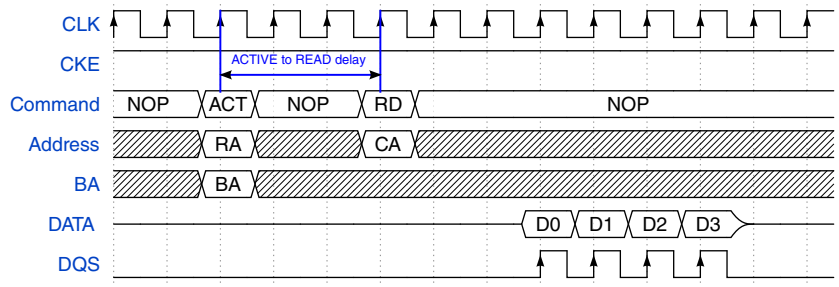


Figure 103. IP Command - SDRAM READ

NOTE

- It is supported but not recommended to send READ command with IP command trigger. The SEMC should read SDRAM device by AXI command.
- Read address is from IPCR0 register, read data is sent to IPRXDAT register.
- For this example:
  - ACTIVE to READ/WRITE delay is 3 (SDRAMCR1[ACT2RW]=2)

29.3.1.4.2.2 WRITE

When an IP command is triggered with command code - WRITE, the SEMC will send a WRITE command to the SDRAM device.

Figure 104 shows the sequence for IP command - WRITE.

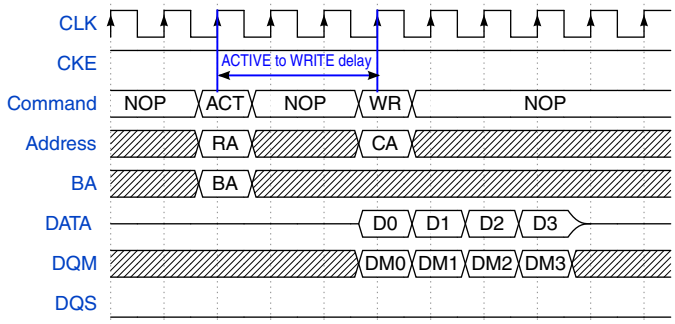


Figure 104. IP Command - SDRAM WRITE

NOTE

- It is supported but not recommended to send WRITE command with IP command trigger. The SEMC should write SDRAM device by AXI command.
- Write address is from IPCR0 register, write data is from IPTXDAT register, write mask is from IPCR2 register.
- For this example:
  - ACTIVE to READ/WRITE delay is 3 (SDRAMCR1[ACT2RW]=2)

29.3.1.4.2.3 MODE REGISTER SET

When an IP command is triggered with command code - MODE REGISTER SET, the SEMC will send a MODE REGISTER SET command to the SDRAM device.

Figure 105 shows the sequence for IP command - MODE REGISTER SET.

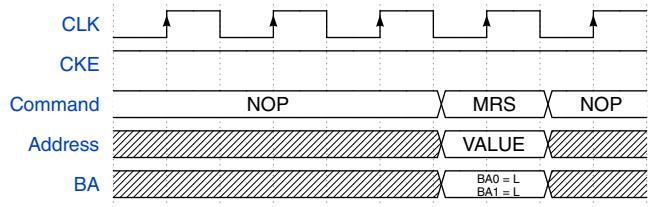


Figure 105. IP Command - SDRAM MODE REGISTER SET

NOTE

- MODE REGISTER SET command triggered by IP command should be used for SDRAM device initialization only.
- The "Value" of Address comes from IPTXDAT[12:0] register.

29.3.1.4.2.4 EXTENDED MODE REGISTER SET

When an IP command is triggered with command code - EXTENDED MODE REGISTER SET, the SEMC will send an EXTENDED MODE REGISTER SET command to the SDRAM device.

Figure 106 shows the sequence for IP command - EXTENDED MODE REGISTER SET.

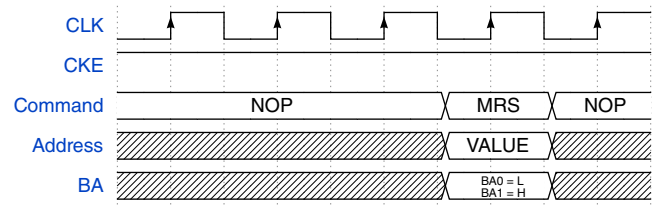


Figure 106. IP Command - EXTENDED SDRAM MODE REGISTER SET

NOTE

- EXTENDED MODE REGISTER SET command triggered by IP command should be used for SDRAM device initialization only.
- The "Value" of Address comes from IPTXDAT[12:0] register.

29.3.1.4.2.5 DEEP POWER DOWN

When an IP command is triggered with command code - DEEP POWER DOWN, the SEMC will send a DEEP POWER DOWN command to the SDRAM device.

Figure 107 shows the sequence for IP command - DEEP POWER DOWN.

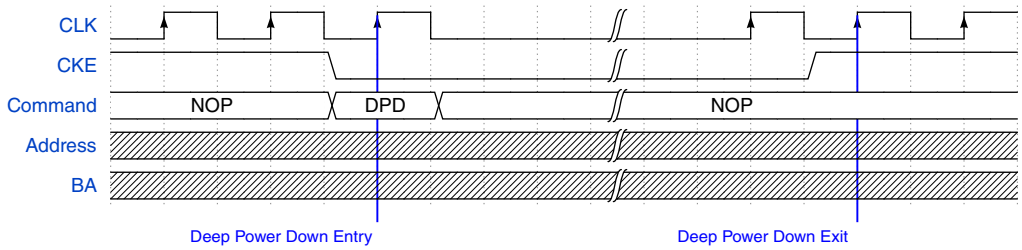


Figure 107. IP Command - DEEP POWER DOWN

NOTE

- The SEMC sends PRECHARGE ALL command to close any opened page on device before sending DEEP POWER DOWN command. There is no PRECHARGE ALL command if no pages on the selected device are open.
- Deep Power Mode is exited by any IP or AXI command.

29.3.1.4.2.6 ACTIVE

When an IP command is triggered with command code - ACTIVE, the SEMC will send an ACTIVE command to the SDRAM device.

Figure 108 shows the sequence for IP command - ACTIVE.

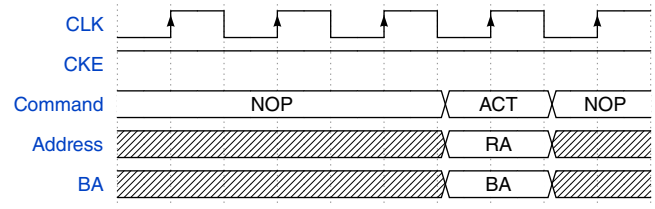


Figure 108. IP Command - SDRAM ACTIVE

NOTE

- It is supported but not recommended to send ACTIVE command with an IP command trigger. The SEMC can handle page management for READ/WRITE/AUTO REFRESH/SELF REFRESH etc.

29.3.1.4.2.7 AUTO REFRESH

When an IP command is triggered with command code - AUTO REFRESH, the SEMC will send an AUTO REFRESH command to the SDRAM device.

Figure 109 shows the sequence for IP command - AUTO REFRESH.

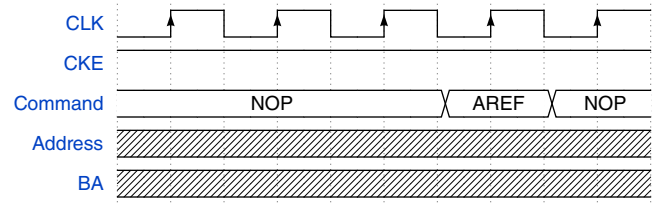


Figure 109. IP Command - SDRAM AUTO REFRESH

NOTE

- There is only one AUTO REFRESH command sent when triggered by IP command and only one device is accessed.
- The SEMC sends PRECHARGE ALL command to close any opened page on device before sending AUTO REFRESH command. There is no PRECHARGE ALL command if no pages on the selected device are open.
- AUTO REFRESH command triggered by IP command should be used for SDRAM device initialization only. For entering auto refresh during normal operation, set SDRAMCR3[REN] to 1 and the SEMC will send AUTO REFRESH command automatically.

29.3.1.4.2.8 SELF REFRESH

When a SELF REFRESH IP command is requested for any SDRAM device, the SEMC will send a SELF REFRESH command to all SDRAM devices. After the SELF REFRESH command is completed, all SDRAM devices should be in self refresh mode and the SEMC keeps this mode on SDRAM interface (SEMC_CKE=0 and SEMC_CLK=0).

The SEMC brings all SDRAM devices out of self refresh mode when there is any new IP command or AXI command triggered to an SDRAM device.

Figure 110 shows the sequence for self refresh entering and exiting by AXI command.

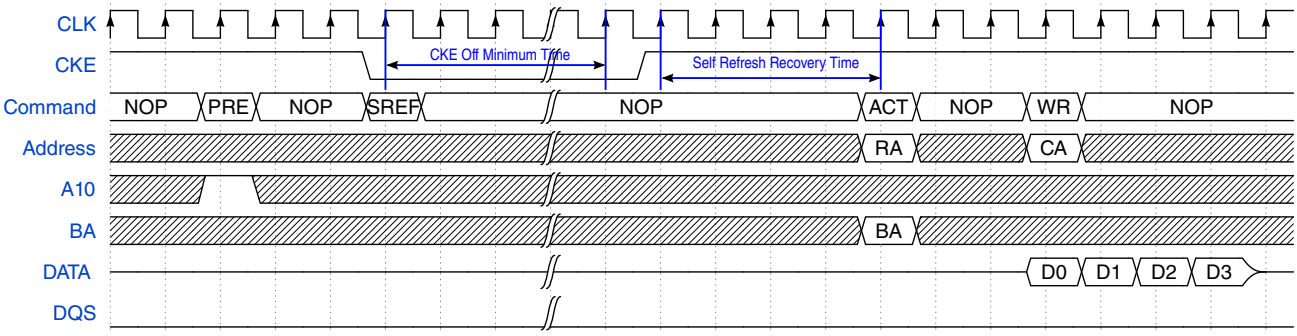


Figure 110. IP Command - SDRAM SELF REFRESH

NOTE

- The SEMC sends PRECHARGE ALL command to close any opened page on device before entering self refresh mode. There is no PRECHARGE ALL command if no page is opened at any device.
- SELF REFRESH command sends to all devices.
- It is not recommended to put device into self refresh mode by IP command triggering although it is supported. Recommend to put device into self refresh mode by the SEMC STOP mode instead.
- Disable the SEMC auto refresh enable bit (SDRAMCR3[REN]=0) before entering self refresh mode, otherwise the SEMC may bring devices out of self refresh mode.
- For this example:
  - Self refresh recovery time is 4 clock cycles (SDRAMCR2[SRRC]=3)
  - CKE off minimum time is 4 clock cycles (SDRAMCR1[CKEOFF]=3)
  - The SEMC brings SDRAM out of self refresh mode when an AXI write command is triggered.

29.3.1.4.2.9 PRECHARGE

When an IP command is triggered with command code - PRECHARGE, The SEMC will send a PRECHARGE command to the SDRAM device.

Figure 111 shows the sequence for IP command - PRECHARGE.

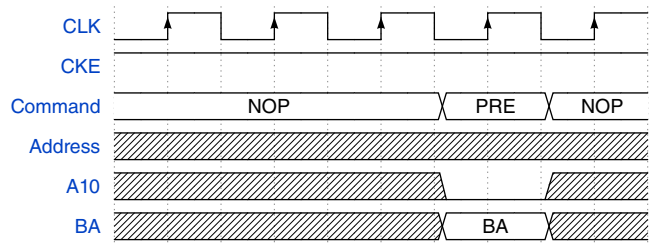


Figure 111. IP Command - SDRAM PRECHARGE

NOTE

- It is not recommended to send PRECHARGE command triggered by IP command although it is supported. The SEMC can handle page management for READ/WRITE/AUTO REFRESH/SELF REFRESH etc.

29.3.1.4.2.10 PRECHARGE ALL

When an IP command is triggered with command code - PRECHARGE ALL, the SEMC will send a PRECHARGE ALL command to the SDRAM device.

Figure 112 shows the sequence for IP command - PRECHARGE ALL.

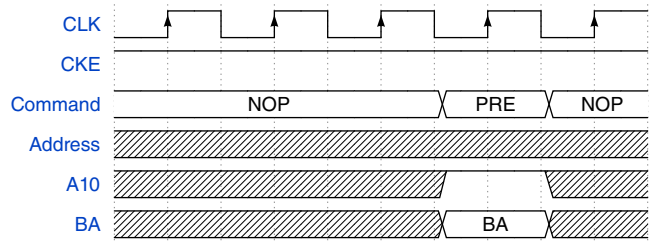


Figure 112. IP Command - SDRAM PRECHARGE ALL

NOTE

- It is not recommended to send PRECHARGE ALL command triggered by IP command although it is supported. The SEMC can handle page management for READ/WRITE/AUTO REFRESH/SELF REFRESH etc.
- PRECHARGE ALL command triggered by IP command should be used for SDRAM device initialization only.

29.3.1.4.2.11 Combination of IP Commands

Figure 113 shows a combination sequence of IP commands, that includes AUTO REFRESH, ACTIVE, WRITE and PRECHARGE IP commands.

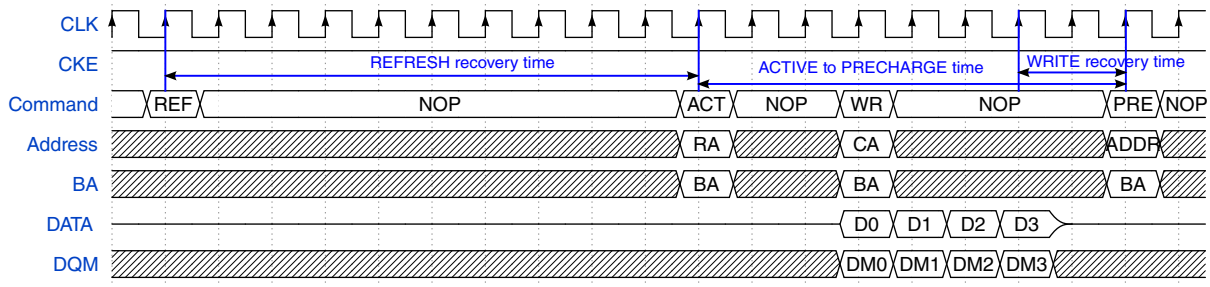


Figure 113. IP Command - AUTO REFRESH, ACTIVE, WRITE and PRECHARGE

NOTE

- For this example:
  - REFRESH recovery time is 10 clock cycles (SDRAMCR1[RFRC]=9)
  - WRITE recovery time is 2 clock cycles (SDRAMCR1[WRC]=1)
  - ACTIVE to PRECHARGE minimum time is 6 clock cycles (SDRAMCR1[ACT2PRE]=5). Current ACTIVE to PRECHARGE time is 8 clock cycles by the limitation of WRITE burst beats and recovery time.

29.3.1.4.3 SDRAM Device Access by AXI Command

SDRAM device read/write access can be triggered by AXI Command. For AXI read access, the SEMC will send READ commands to SDRAM device automatically and return data on AXI bus. For AXI write access, the SEMC will save AXI bus write data and send WRITE commands to the SDRAM device automatically.

29.3.1.4.3.1 Read Access

Figure 114 shows an SDRAM read triggered by an AXI command.

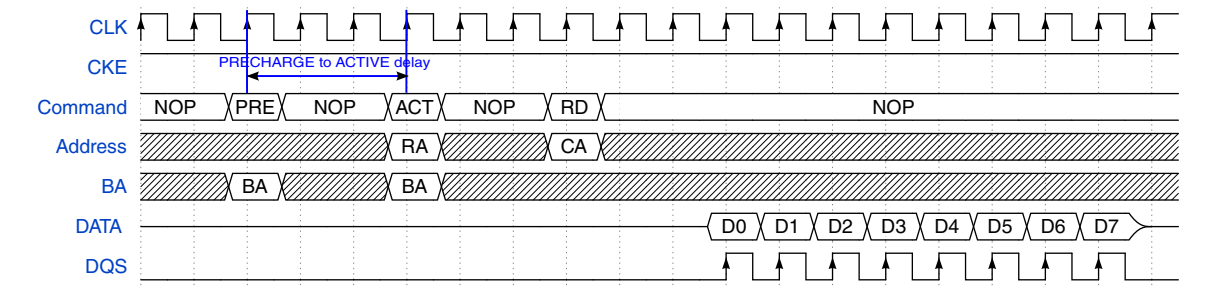


Figure 114. AXI Command - SDRAM READ

NOTE

- The SEMC opens/closes pages automatically for current AXI command.
- For this example:
  - CAS latency is 3 (SDRAMCR0[CL]=3)
  - Burst length is 8 (SDRAMCR0[BL]=3)
  - PRECHARGE to ACTIVE delay is 3 clock cycles (SDRAMCR1[PRE2ACT]=2)

29.3.1.4.3.2 Write Access

Figure 115 shows an SDRAM write triggered by an AXI command.

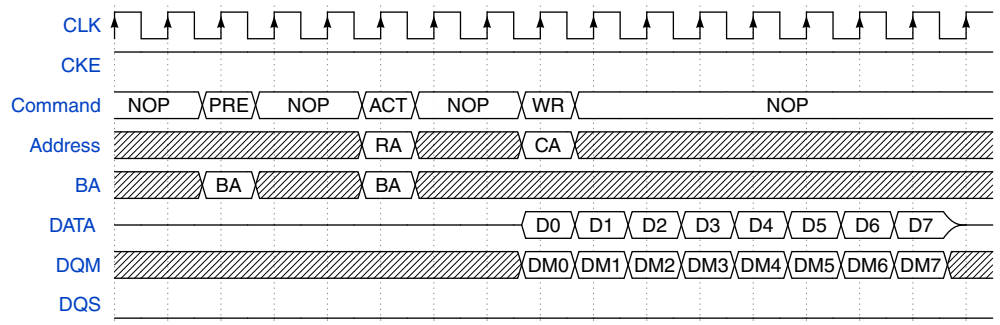


Figure 115. AXI Command - SDRAM WRITE

NOTE

- The SEMC opens/closes pages automatically for current AXI command.
- SDRAM device burst length is 8 in this example. The SEMC may send multiple WRITE commands if AXI burst size is larger than SDRAM burst length.

29.3.1.4.3.3 Pipelined Access

Figure 116 shows an SDRAM pipelined read access that is triggered by AXI bus read access.

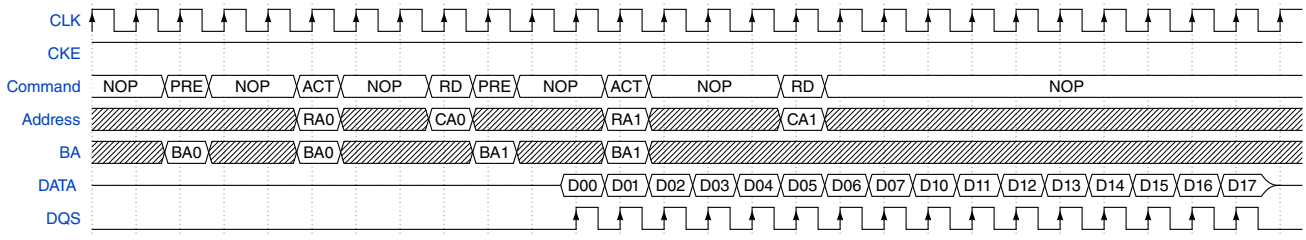


Figure 116. AXI Command - Pipelined SDRAM Read

NOTE

- The SEMC opens/closes pages automatically for next AXI command during SDRAM Read burst.
- Data D0x is from CA0, and data D1x is from CA1. This naming rule is also applied to the following SDRAM figures.

Figure 117 shows the SDRAM device pipelined write access that triggered by AXI bus write access.

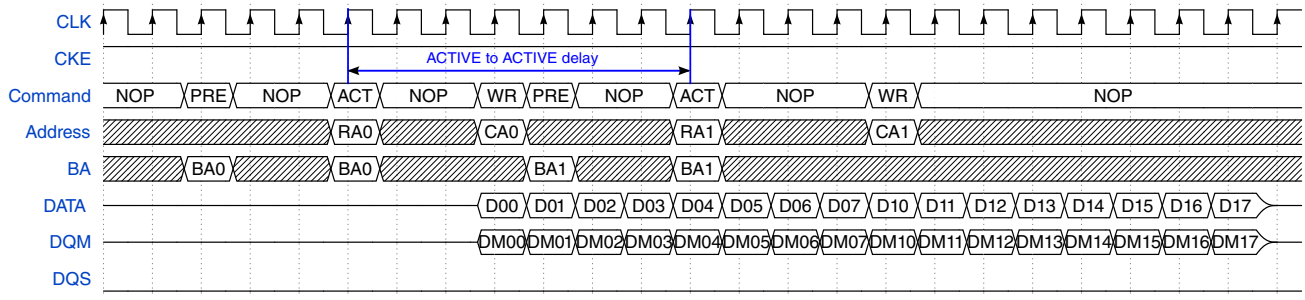


Figure 117. AXI Command - Pipelined SDRAM Write

NOTE

- The SEMC opens/closes pages automatically for next AXI command during SDRAM Write burst.

29.3.1.4.3.4 SDRAM Back-to-Back Access

This section describes the SDRAM back-to-back access.

29.3.1.4.3.4.1 Read-to-Read Access

Figure 118 shows SDRAM read-to-read access timing when CL=1.

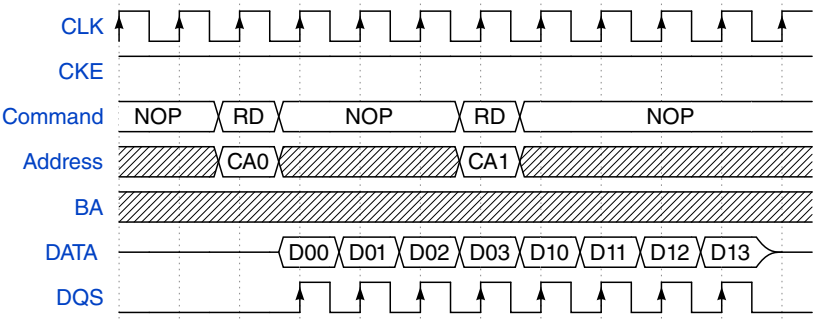


Figure 118. SDRAM Read-to-Read Access (CL=1)

Figure 119 shows SDRAM read-to-read access timing when CL=2.

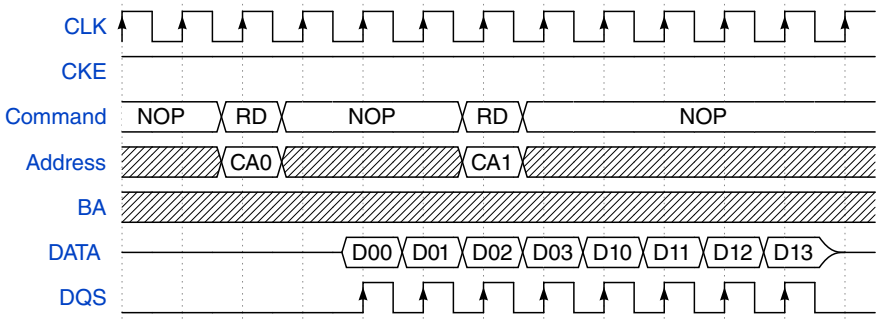


Figure 119. SDRAM Read-to-Read Access (CL=2)

Figure 120 shows SDRAM read-to-read access timing when CL=3.

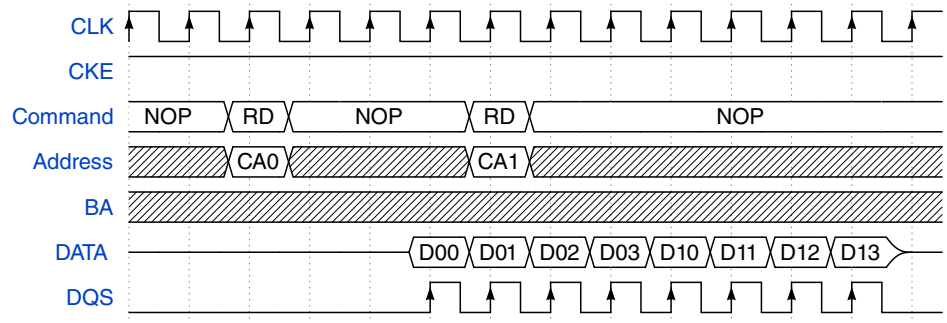


Figure 120. SDRAM Read-to-Read Access (CL=3)

29.3.1.4.3.4.2 Read-to-Write Access

Figure 121 shows SDRAM read-to-write access timing when CL=1.

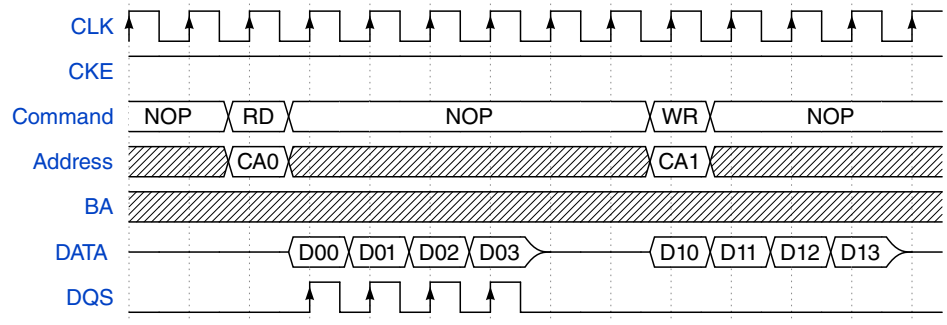


Figure 121. SDRAM Read-to-Write Access (CL=1)

Figure 122 shows SDRAM read-to-write access timing when CL=2.

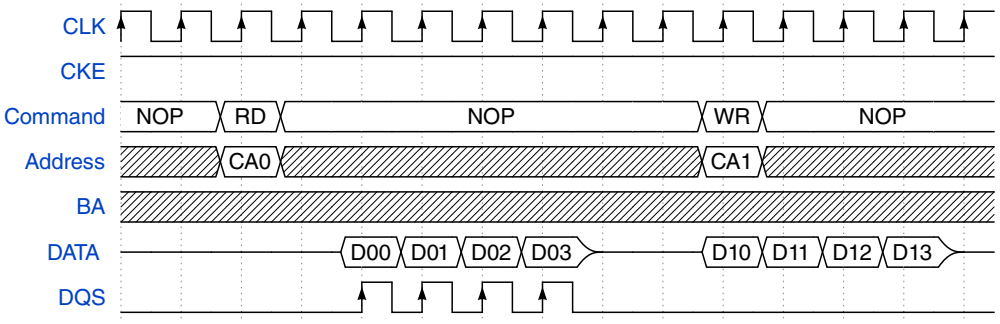


Figure 122. SDRAM Read-to-Write Access (CL=2)

Figure 123 shows SDRAM read-to-write access timing when CL=3.

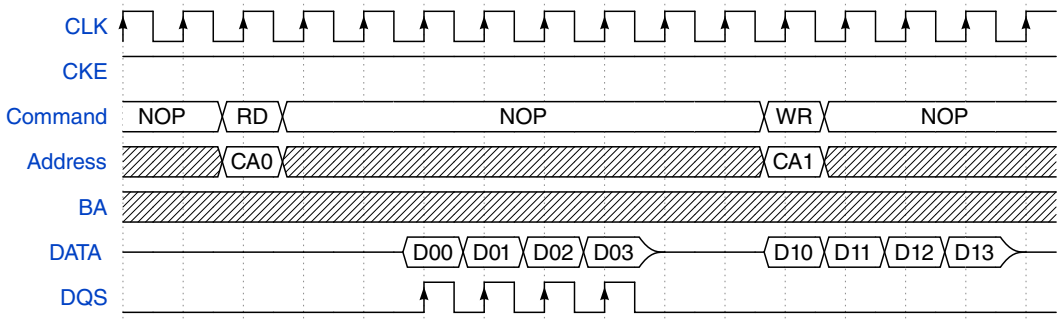


Figure 123. SDRAM Read-to-Write Access (CL=3)

29.3.1.4.3.4.3 Write-to-Write Access

Figure 124 shows SDRAM write-to-write access timing.

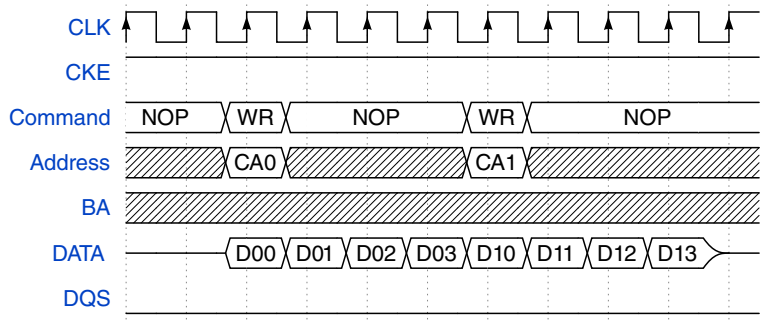


Figure 124. SDRAM Write-to-Write Access

29.3.1.4.3.4.4 Write-to-Read Access

Figure 125 shows SDRAM write-to-read access timing when CL=1.

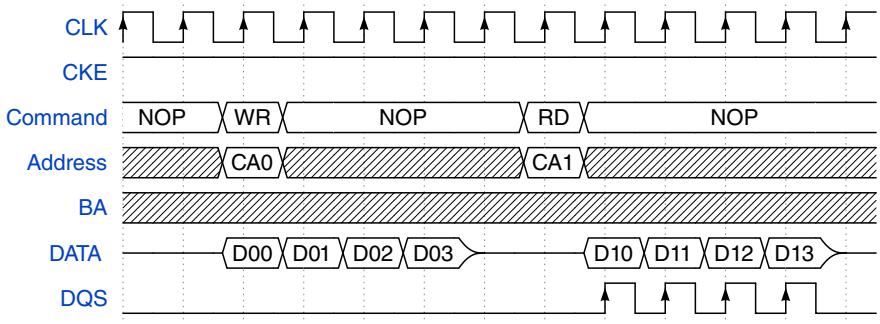


Figure 125. SDRAM Write-to-Read Access (CL=1)

Figure 126 shows SDRAM write-to-read access timing when CL=2.

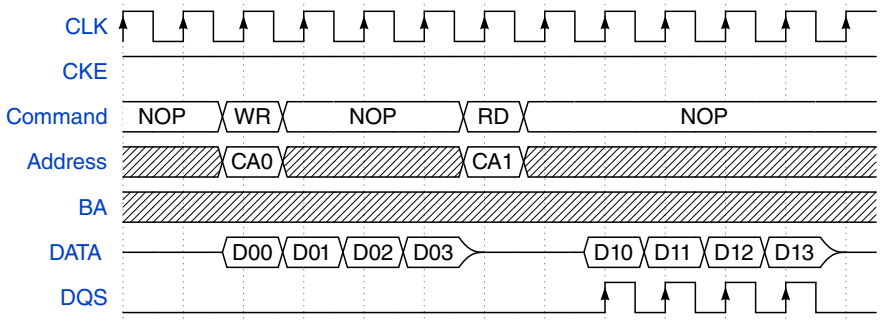


Figure 126. SDRAM Write-to-Read Access (CL=2)

Figure 127 shows SDRAM write-to-read access timing when CL=3.

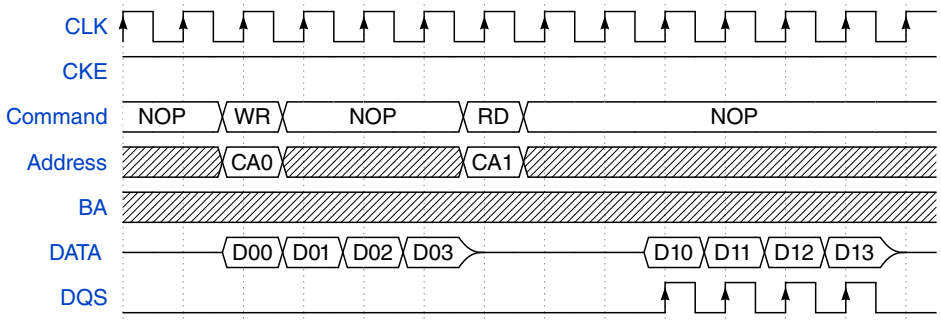


Figure 127. SDRAM Write-to-Read Access (CL=3)

29.3.1.4.4 Refresh Command - SDRAM AUTO REFRESH

When SDRAM refresh is enabled (SDRAMCR3[REN]=1), the SEMC will send an AUTO REFRESH command to all SDRAM devices whenever the refresh timer expires. Multiple AUTO REFRESH commands can be sent if refresh burst is not set to one (SDRAMCR3[REBL]!=0).

The refresh timer's count cycle is controlled by UT, RT and PRESCALE bits in SDRAMCR3.

Figure 128 shows the sequence of AUTO REFRESH Command.

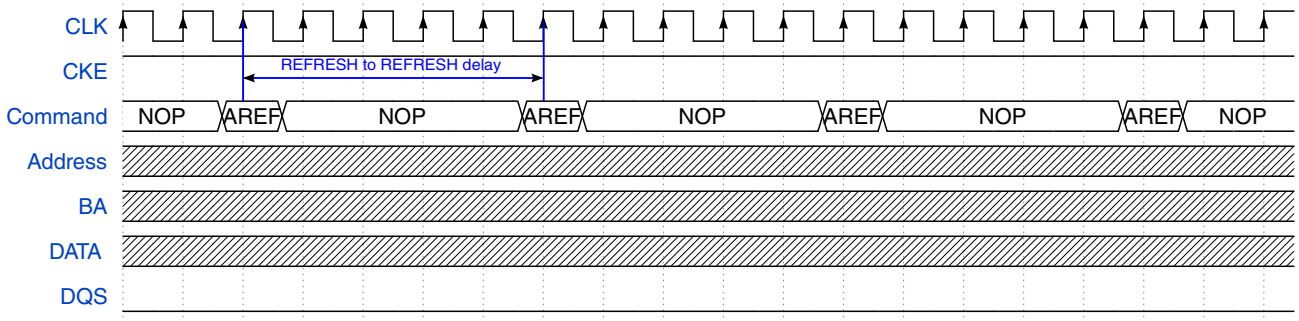


Figure 128. Refresh Command - SDRAM AUTO REFRESH

NOTE

- AUTO REFRESH command is sent to all devices.
- For this example:
  - Refresh to refresh delay is 5 clock cycles (SDRAMCR2[REF2REF]=4)
  - Refresh burst length is 4 (SDRAMCR3[REBL]=3)

29.3.1.4.5 SDRAM Low Power Feature

This section describes the low power feature that is implemented for SDRAM.

29.3.1.4.5.1 SDRAM Stop Mode

The SEMC puts SDRAM to self refresh mode automatically when the system enters STOP mode. The SEMC brings SDRAM out of self refresh mode when the system exits STOP mode.

Figure 129 shows the sequence of STOP mode entering and exiting.

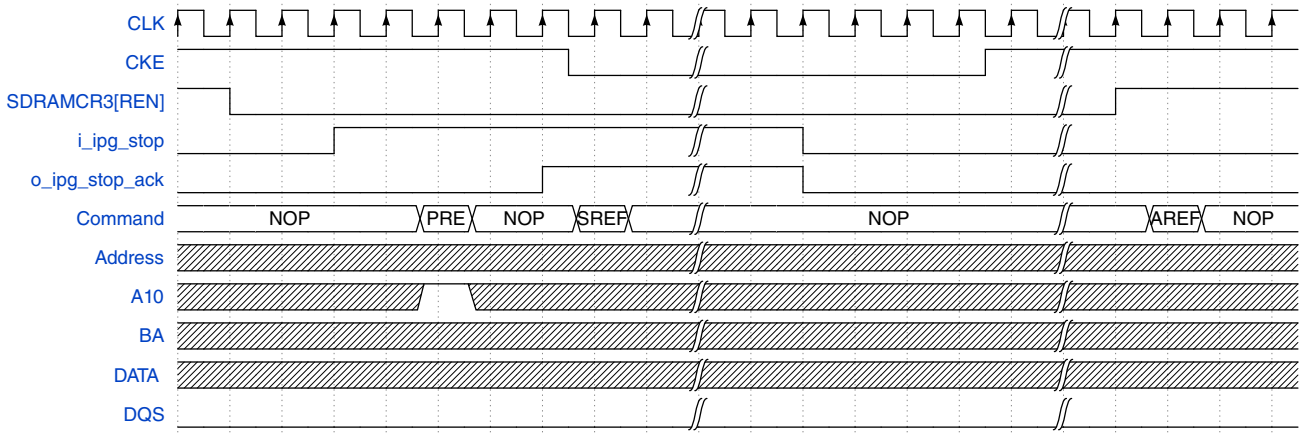


Figure 129. SDRAM Stop Mode Enter and Exit

NOTE

- The SEMC sends PRECHARGE ALL command to close any opened page on device before entering self refresh mode. There is no PRECHARGE ALL command if no page is opened at any device.
- SELF REFRESH command is sent to all devices.
- Disable the SEMC auto refresh bit (SDRAMCR3[REN]=0) before entering STOP mode, otherwise the SEMC may bring devices out of self refresh mode.

29.3.1.4.5.2 SDRAM Bus Idle

When there are no pending IP, AXI, or refresh commands, SDRAM interface will in the idle state. When the SDRAM interface bus idle timer expires, the SEMC closes all opened pages on all SDRAM devices. Refer to SDRAMCR2[ITO] for more information.

29.3.1.4.6 SDRAM Page Management

This section describes the SDRAM page management.

The SEMC will save row addresses for up to 8 opened pages: 4 pages for CS0/CS2 (4 banks), 4 pages for CS1/CS3 (4 banks). Device CS0 and CS2 pages are never opened at the same time. Before the SEMC opens pages on CS2, it closes all opened pages on CS0. This is similar for CS1/CS3.

The SEMC closes all pages on all CS by issuing a PRECHARGE ALL command in following cases:

1. SELF REFRESH triggered by IP command
2. SELF REFRESH triggered by stop mode entry
3. The idle time on AXI bus and SDRAM interface is longer than SDRAM idle timeout time (SDRAMCR2[ITO]).

NOTE

The SEMC will not send out PRECHARGE ALL command if there are no pages opened on any bank/CS.

The SEMC will close all pages on one CS in following cases:

1. PRECHARGE ALL command triggered by IP command
2. AUTO REFRESH command triggered by IP command
3. ACTIVE/WRITE/READ command triggered by IP command and CS miss
4. WRITE/READ command triggered by current AXI command and CS miss
5. WRITE/READ command triggered by next AXI command (not accepted yet), CS miss and not same CS as current AXI command

NOTE

One example for CS miss: current IP command access CS2 but there is page opened on CS0 already.

The SEMC will close one page on one CS in following cases:

1. PRECHARGE command triggered by IP command and this bank is opened
2. ACTIVE/READ/WRITE command triggered by IP command, this bank is opened and page missed.
3. WRITE/READ command triggered by current AXI command, this bank is opened and page missed.
4. WRITE/READ command triggered by next AXI command (not accepted yet), this bank is opened, page missed and different bank accessed with current AXI command.

NOTE

One example for page miss: current bank is opened but the opened page row address is different with current SDRAM command access.

The SEMC will open one page on one CS in following cases:

1. ACTIVE/READ/WRITE command triggered by IP command, this bank is not opened.
2. WRITE/READ command triggered by current AXI command, this bank is not opened.
3. WRITE/READ command triggered by next AXI command (not accepted yet), this bank is not opened and different bank accessed with current AXI command.

29.3.1.5 NAND Flash Controller Operations

This section describes the operations of NAND Flash controller.

29.3.1.5.1 NAND Flash Address Multiplexing

The NAND Flash controller uses a multiplexed address split into rows and columns. Figure 130 shows how the bits of the linear address sent to the SEMC are split up into the row and column addresses sent to the NAND Flash memory device.

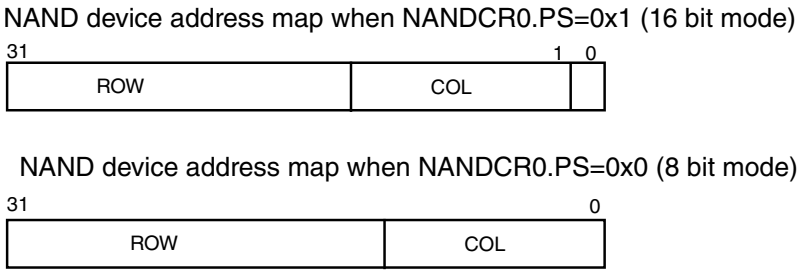


Figure 130. NAND Flash Address Multiplexing

NOTE

- NAND address bits consist of row and column address bits.
- Column address bits offset is bit 0 if NANDCR0[PS]=0, the offset is bit 1 if NANDCR0[PS]=1. Column address bit width is determined by NANDCR0[COL].
- Column bit number should cover all page size. For example, column bit number should be 14 if NAND Flash page size is 8192 + 448 (main + spare) bytes.
- Row address bits offset is determined by NANDCR0[PS] and NANDCR[COL].
- NAND device base address on SoC is configured by BR4 and BR8 registers.

29.3.1.5.2 NAND Flash Data Map

Table 185 lists the data map when NAND controller using NAND buffer.

Table 185. Buffer to Physical Mappings Examples

Flash page size (main+spare)	ECC Mode	Sector Number	Sector Size in Buffer	Sector Size in Flash	Mapping
512+16	0	1	528	528	Buffer_0[527:0]=Flash[527:0]
512+16	0	1	512	512	Buffer_0[511:0]=Flash[511:0]
512+16	1	1	512	520	Buffer_0[511:0]=Flash[511:0]
512+16	2	1	512	528	Buffer_0[511:0]=Flash[511:0]
2048+64	0	1	2112	2112	Buffer_0[2111:0]=Flash[2111:0]
2048+64	0	1	2048	2048	Buffer_0[2047:0]=Flash[2047:0]
2048+64	1	1	2048	2056	Buffer_0[2047:0]=Flash[2047:0]
2048+64	2	1	2048	2064	Buffer_0[2047:0]=Flash[2047:0]
2048+64	1	4	512	520	Buffer_0[511:0]=Flash[511:0] Buffer_1[511:0]=Flash[1031:520] Buffer_2[511:0]=Flash[1551:1040] Buffer_3[511:0]=Flash[2071:1560]
2048+64	2	4	512	528	Buffer_0[511:0]=Flash[511:0] Buffer_1[511:0]=Flash[1039:528] Buffer_2[511:0]=Flash[1567:1056] Buffer_3[511:0]=Flash[2095:1584]

NOTE

- Sector Size in Buffer stands for NANDCR0[SECTOR_SIZE]
- Sector Size in Flash = Sector Size in Buffer + ECC bytes
- If ECC Mode is 0, then ECC bytes is 0, and Sector Size in Flash = Sector Size in Buffer
- If ECC Mode is 1, then ECC bytes is 8, and Sector Size in Flash = Sector Size in Buffer + 8
- If ECC Mode is 2, then ECC bytes is 16, and Sector Size in Flash = Sector Size in Buffer + 16

29.3.1.5.3 NAND Flash Access Sequences

NAND Flash controller supports program and read operations by IPS and AXI bus. It has internal 2112 bytes NAND buffer to support encoding or decoding of ECC schemes. The NAND buffer must be enabled (NANDCR0[BUFEN]=1) if ECC function is required, otherwise it is not mandatory. Some program and read sequences are listed as below:

29.3.1.5.3.1 NAND Flash IPS Program Sequence

IPS program operation with buffer enabled (NANDCR0[BUFEN]=1):

1. Configure NANDCR0 register for ECC mode, sector number and size.
2. Configure IPCR0/IPCR1/IPCR2 registers for device address, data size and write mask.
3. Send Flash command "PROGRAM PAGE first command (80h)".
4. Configure NDBA register with proper NAND buffer start address, recommend 0.
5. Write data to NDBD register. IPCR1[DATSZ] should be 4 bytes, then the write pointer of buffer address increases 4 by each write action. It can not exceed (sector size)*(sector number).
6. Send IP command "NAND Buffer Write".
7. Repeat steps 4 - 6 if NAND device's main page space is more than 2048 bytes, otherwise go to step 8.
8. Send Flash command "PROGRAM PAGE second command (10h)".
9. Wait program sequence done.

IPS program operation with buffer disabled (NANDCR0[BUFEN]=0):

1. Configure IPCR0/IPCR1/IPCR2 registers for device address, data size and write mask.
2. Send Flash command "PROGRAM PAGE first command (80h)".
3. Write data to IPTXDAT register.
4. Send IP command "NAND Write".
5. Repeat steps 3 - 4 to write more data in the page, otherwise go to step 6.
6. Send Flash command "PROGRAM PAGE second command (10h)".
7. Wait program sequence done.

NOTE

- Be caution that erased data is not protected by ECC scheme.

29.3.1.5.3.2 NAND Flash IPS Read Sequence

IPS read operation with buffer enabled (NANDCR0[BUFEN]=1):

1. Configure NANDCR0 register for ECC mode, sector number and size.
2. Configure IPCR0/IPCR1/IPCR2 registers for device address, data size and write mask.

3. Send Flash command "PAGE READ".
4. Wait read data ready.
5. Send IP command "NAND Buffer Read".
6. Configure NDBA register with proper NAND buffer start address.
7. Read data from NDBD register. IPCR1[DATSZ] should be 4 bytes, then the read pointer of buffer address increases 4 by read action. It can not exceed (sector size)*(sector number).
8. Repeat steps 5 - 7 if NAND device's main page space is more than 2048 bytes.

IPS read operation with buffer disabled (NANDCR0[BUFEN]=0):

1. Configure IPCR0/IPCR1/IPCR2 registers for device address, data size and write mask.
2. Send Flash command "PAGE READ".
3. Wait read data ready.
4. Send IP command "NAND Read".
5. Read data from IPRXDAT register.
6. Repeat steps 4 - 5 to read more data in the page.

29.3.1.5.3.3 NAND Flash AXI Program Sequence

AXI program operation with buffer enabled (NANDCR0.BUFEN=1):

1. Configure NANDCR0 register for ECC mode, sector number and size.
2. Configure IPCR0/IPCR1/IPCR2 registers for device address, data size and write mask.
3. Send Flash command "PROGRAM PAGE first command (80h)".
4. Write data by AXI bus with valid address, the address offset can not exceed (sector size)*(sector number).
5. Send IP command "NAND Buffer Write".
6. Repeat steps 4 - 5 if NAND device's main page space is more than 2048 bytes, otherwise go to step 7.
7. Send Flash command "PROGRAM PAGE second command (10h)".
8. Wait program sequence done.

AXI program operation with buffer disabled (NANDCR0[BUFEN]=0):

1. Configure IPCR0/IPCR1/IPCR2 registers for device address, data size and write mask.
2. Send Flash command "PROGRAM PAGE first command (80h)".
3. Write data by AXI bus with valid address.
4. Repeat step 3 to write more data in the page, otherwise go to step 5.
5. Send Flash command "PROGRAM PAGE second command (10h)".
6. Wait program sequence done.

29.3.1.5.3.4 NAND Flash AXI Read Sequence

AXI read operation with buffer enabled (NANDCR0[BUFEN]=1):

1. Configure NANDCR0 register for ECC mode, sector number and size.
2. Configure IPCR0/IPCR1/IPCR2 registers for device address, data size and write mask.
3. Send Flash command "PAGE READ".

4. Wait read data ready.
5. Send IP command "NAND Buffer Read".
6. Read data by AXI bus with valid address, the read address can not exceed (sector size)*(sector number).
7. Repeat steps 5 - 6 if NAND device's main page space is more than 2048 bytes.

AXI read operation with buffer disabled (NANDCR0[BUFEN]=0):

1. Configure IPCR0/IPCR1/IPCR2 registers for device address, data size and write mask.
2. Send Flash command "PAGE READ".
3. Wait read data ready.
4. Send IP command "NAND Read".
5. Read data by AXI bus with valid address, the read address can not exceed page size.
6. Repeat step 5 to read more data.

29.3.1.5.4 NAND Flash Access Address Type

Table 186 lists the NAND address types that are triggered by IP commands.

Table 186. NAND Address Types

Address Type	IPCMD[7:4]
Five Bytes Row/Column Address	0x0
One Byte Column Address	0x1
Two Bytes Column Address	0x2
One Byte Row Address	0x3
Two Bytes Row Address	0x4
Three Bytes Row Address	0x5

For IP command, address type is determined by IPCMD register. For AXI command, address type is fixed to "Two Bytes Column Address" because AXI access is limited in opened NAND page.

29.3.1.5.4.1 Five Bytes Row/Column Address

Figure 131 shows the Five Bytes Row/Column Address type access timing.

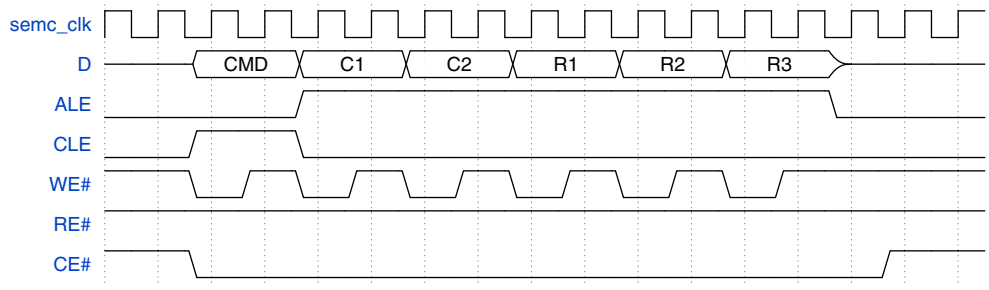


Figure 131. NAND Flash Access Address Type - Five Bytes Row/Column Address

There are some option bits to support different NAND address types in the Five Bytes Row/Column Address type. Table 187 lists all the combinations of address types.

Table 187. Address Bytes Order for five Bytes Row/Column Address Mode

NANDCR3[NDOPT3]	NANDCR3[NDOPT2]	NANDCR3[NDOPT1]	Address byte order
0	0	0	C1/C2/R1/R2/R3
0	0	1	C1/R1/R2/R3
0	1	0	C1/C2/R1/R2
0	1	1	C1/R1/R2
1	-	0	C1/C2/R1
1	-	1	C1/R1

29.3.1.5.4.2 One Byte Column Address

Figure 132 shows the One Byte Column Address type access timing.

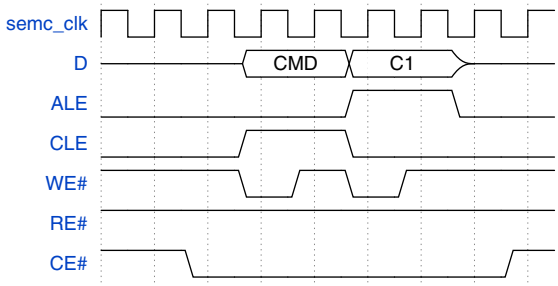


Figure 132. NAND Flash Access Address Type - One Byte Column Address

29.3.1.5.4.3 Two Bytes Column Address

Figure 133 shows the Two Bytes Column Address type access timing.

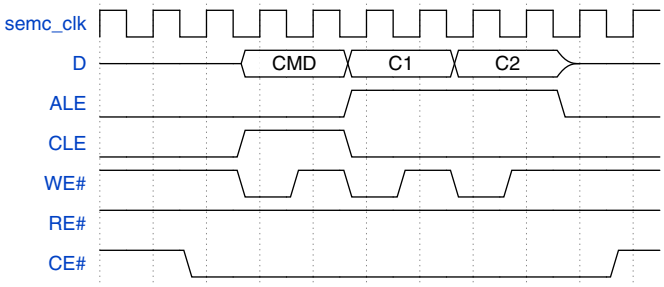


Figure 133. NAND Flash Access Address Type - Two Bytes Column Address

29.3.1.5.4.4 One Byte Row Address

Figure 134 shows the One Byte Row Address type access timing.

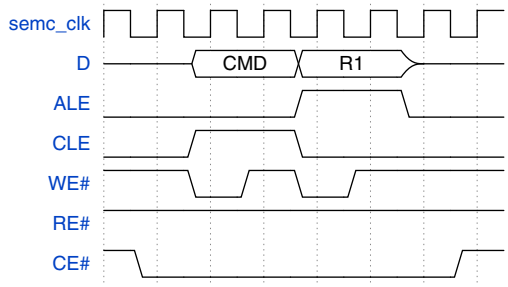


Figure 134. NAND Flash Access Address Type - One Byte Row Address

29.3.1.5.4.5 Two Bytes Row Address

Figure 135 shows the Two Bytes Row Address type access timing.

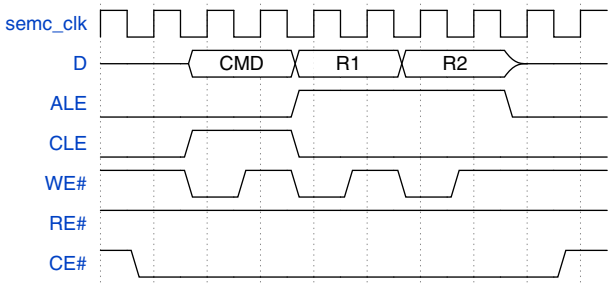


Figure 135. NAND Flash Access Address Type - Two Bytes Row Address

29.3.1.5.4.6 Three Bytes Row Address

Figure 136 shows the Three Bytes Row Address type access timing.

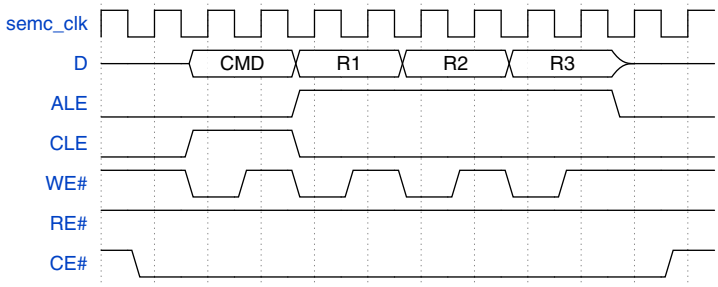


Figure 136. NAND Flash Access Address Type - Three Bytes Row Address

29.3.1.5.5 NAND Flash Access Command Type

NAND Flash access support following command types:

1. Command Phase
2. Command + Wait Phase
3. Command + Address Phase
4. Command + Address + Wait Phase

5. Command + Address + Read Phase
6. Command + Address + Write Phase (for IP command)
7. Command + Write Phase
8. Command + Read Phase
9. Write Phase
10. Read Phase
11. Command + Address + Command + Read Phase (for AXI command - Read)
12. Command + Address + Write Phase (for AXI command - Write)

29.3.1.5.5.1 Command Phase

Figure 137 shows the NAND Flash Command Phase access timing.

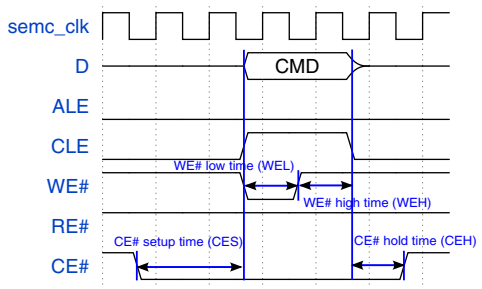


Figure 137. NAND Flash Access Command Type - Command Phase

NOTE

- semc_clk is the SEMC functional clock.
- For this example:
  - CE# setup time is 2 clock cycles (NANDCR1[CES]=1)
  - WE# low time is 1 clock cycle (NANDCR1[WEL]=0)
  - WE# high time is 1 clock cycle (NANDCR1[WEH]=0)
  - CE# hold time is 1 clock cycle (NANDCR1[CEH]=0)

29.3.1.5.5.2 Command + Wait Phase

Figure 138 shows NAND Flash Command + Wait Phase access timing

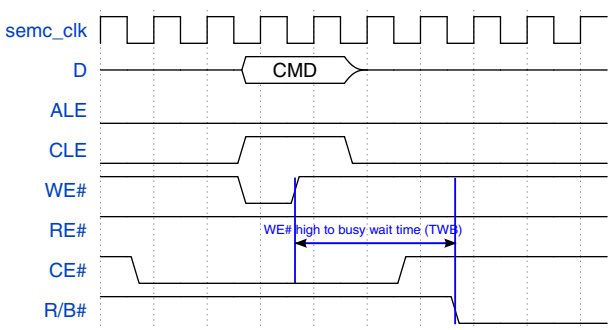


Figure 138. NAND Flash Access Command Type - Command + Wait Phase

NOTE

- For this example:
  - WE# high to busy time is 3 clock cycles (NANDCR1[TWB]=2)

29.3.1.5.5.3 Command + Address Phase

Figure 139 shows the NAND Flash Command + Address Phase access timing.

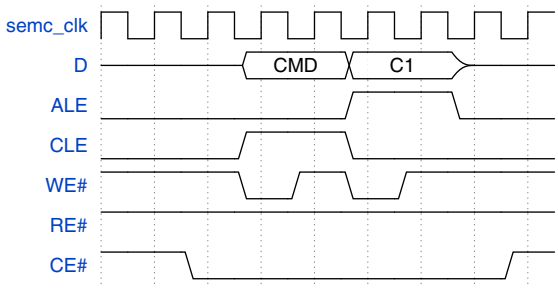


Figure 139. NAND Flash Access Command Type - Command + Address Phase

29.3.1.5.5.4 Command + Address + Wait Phase

Figure 140 shows NAND Flash Command + Address + Wait Phase access timing.

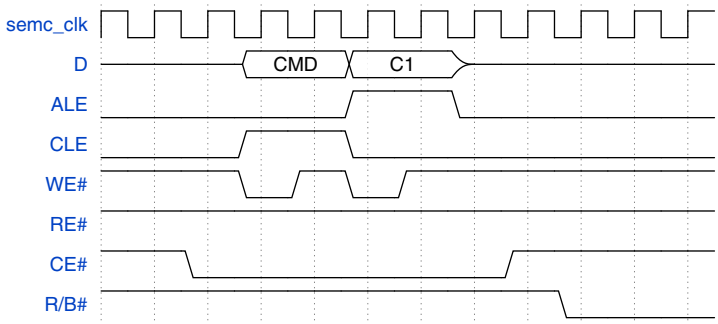


Figure 140. NAND Flash Access Command Type - Command + Address + Wait Phase

29.3.1.5.5.5 Command + Address + Read Phase

Figure 141 shows NAND Flash Command + Address + Read Phase access timing.

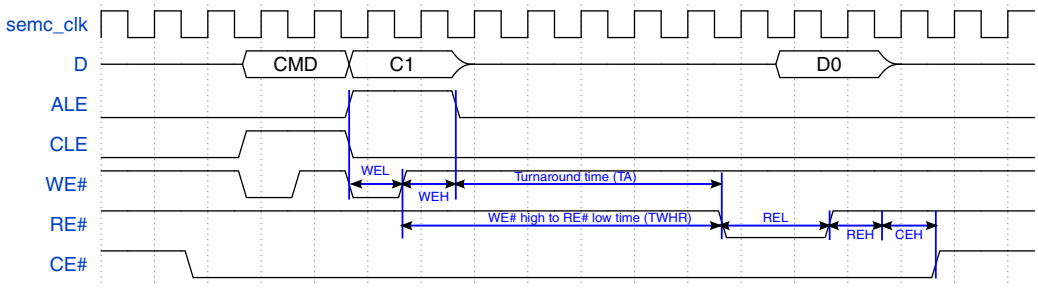


Figure 141. NAND Flash Access Command Type - Command + Address + Read Phase

NOTE

- For this example:
  - WE# low time is 1 clock cycle (NANDCR1[WEL]=0)
  - WE# high time is 1 clock cycle (NANDCR1[WEH]=0)
  - RE# low time is 2 clock cycles (NANDCR1[REL]=1)
  - RE# high time is 1 clock cycle (NANDCR1[REH]=0)
  - Turnaround time is 5 clock cycles (NANDCR1[TA]=4)
  - WE# high to RE# low time is 6 clock cycles (NANDCR2[TWHR]=5)
  - CE# hold time is 1 clock cycle (NANDCR1[CEH]=0)

29.3.1.5.5.6 Command + Address + Write Phase (for IP Command)

Figure 142 shows NAND Flash Command + Address + Write Phase access timing.

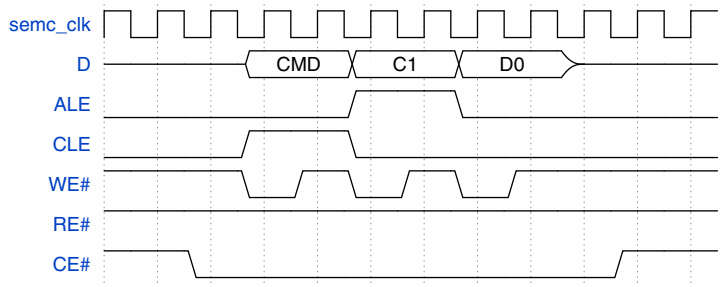


Figure 142. NAND Flash Access Command Type - Command + Address + Write Phase

29.3.1.5.5.7 Command + Read Phase

Figure 143 shows NAND Flash Command + Read Phase access timing.

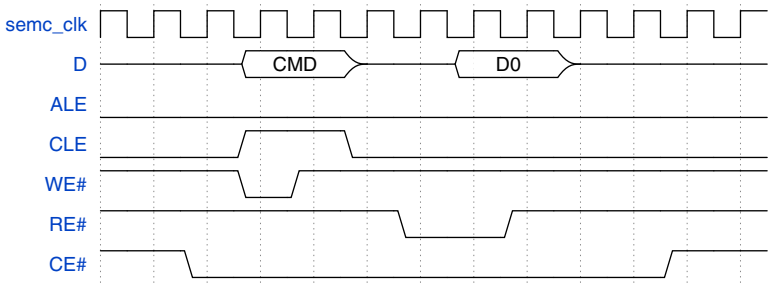


Figure 143. NAND Flash Access Command Type - Command + Read Phase

29.3.1.5.5.8 Command + Write Phase

Figure 144 shows NAND Flash Command + Write Phase access timing.

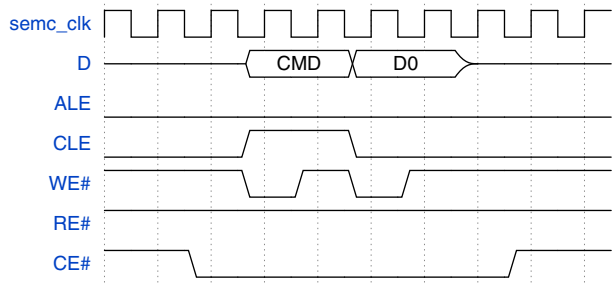


Figure 144. NAND Flash Access Command Type - Command + Write Phase

29.3.1.5.5.9 Read Phase

Figure 145 shows NAND Flash Read Phase access timing.

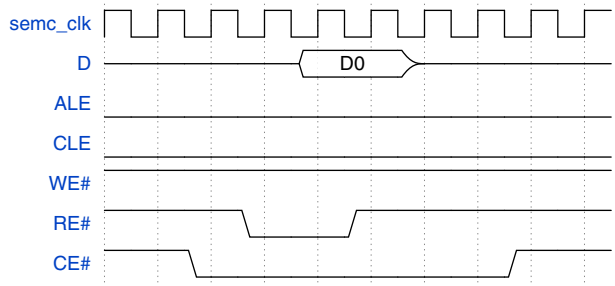


Figure 145. NAND Flash Access Command Type - Read Phase

29.3.1.5.5.10 Write Phase

Figure 146 shows the timing of NAND Flash Write Phase access.

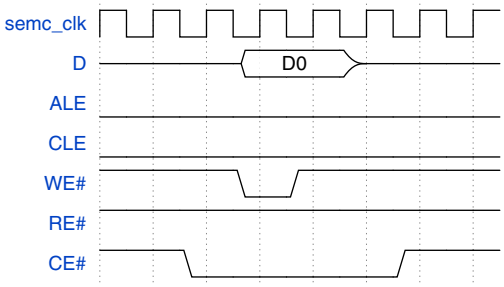


Figure 146. NAND Flash Access Command type - Write Phase

29.3.1.5.5.11 Command + Address + Command + Read Phase

Figure 147 shows the NAND Flash Command + Address + Command + Read Phase access timing.

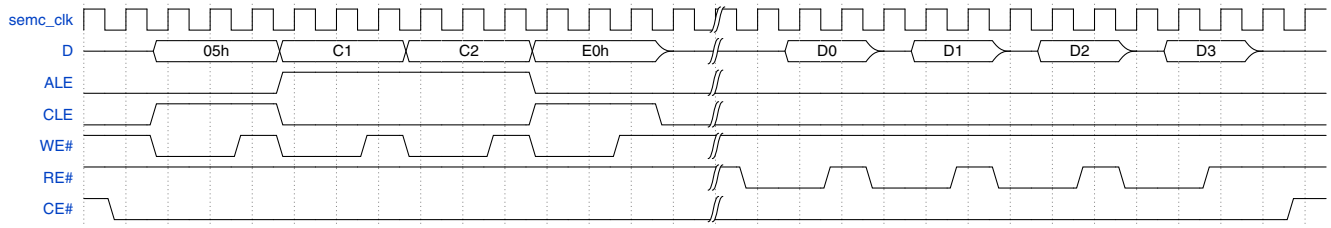


Figure 147. NAND Flash Access Command Type - Command + Address + Command + Read Phase

NOTE

- This command type is for AXI command only
- The command code in this command type is fixed value (Change Column Address) to support ONFI NAND device. The first byte command code is 05h, and second byte command code is E0h.
- The address type in this command type is fixed to two byte column address.

29.3.1.5.5.12 Command + Address + Write Phase (for AXI command)

Figure 148 shows the NAND Flash Command + Address + Write Phase access timing.

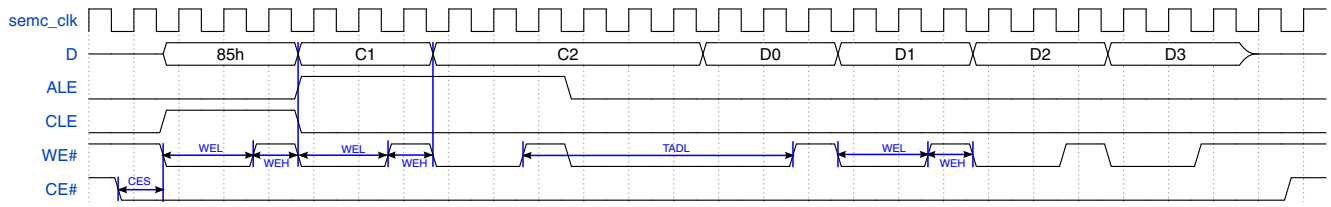


Figure 148. NAND Flash Access Command Type - Command + Address + Write Phase

NOTE

- This command type is for AXI command only
- The command code in this command type is fixed value (85h - Change Write Column Address) to support ONFI NAND device.
- The address type in this command type is fixed to two byte column address.
- For this example:
  - WE# low time is 2 clock cycles (NANDCR1[WEL]=1)
  - WE# high time is 1 clock cycle (NANDCR1[WEH]=0)
  - Address cycle to data loading time is 6 clock cycles (NANDCR2[TADL]=5)

29.3.1.5.6 NAND Flash Read Access in EDO and non-EDO Mode

NAND Flash support both EDO and non-EDO modes. Figure 149 shows the access timing of EDO mode.

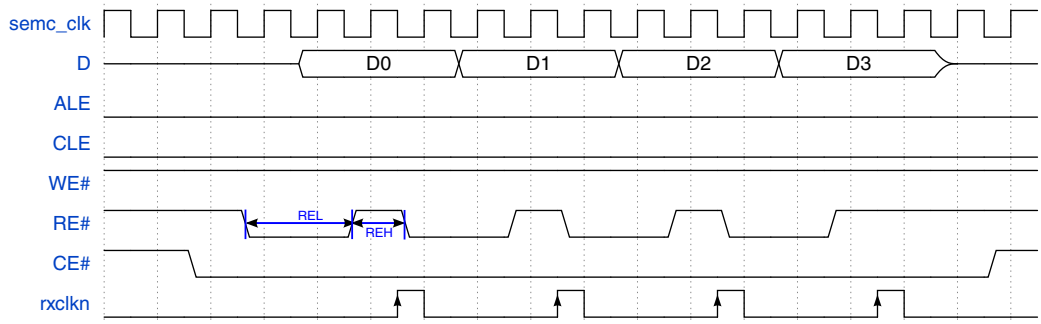


Figure 149. NAND Flash Read Access in EDO mode

NOTE

- rxclkn is the SEMC internal clock to sample read data. It is close to falling edge of RE#.
- For this example:
  - RE# low time is 2 clock cycles (NANDCR1[REL]=1)
  - RE# high time is 1 clock cycle (NANDCR1[REH]=0)

Figure 150 shows the access timing of non-EDO mode.

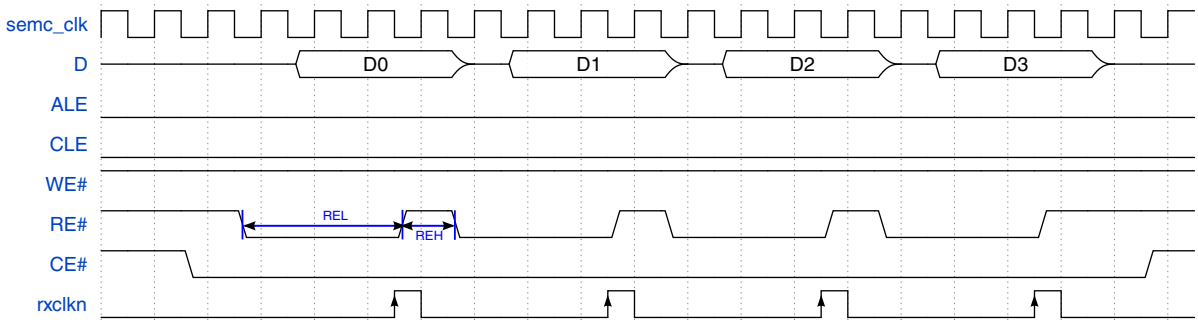


Figure 150. NAND Flash Read Access in non-EDO mode

NOTE

- rxclkn is the SEMC internal clock to sample read data. It is close to rising edge of RE#.
- For this example:
  - RE# low time is 3 clock cycles (NANDCR1[REL]=2)
  - RE# high time is 1 clock cycle (NANDCR1[REH]=0)

29.3.1.5.7 NAND Flash Access in SYNC Mode

NAND Flash supports SYNC mode, the following chapters describes the access timings.

29.3.1.5.7.1 NAND Command + Address + Command + Read Operation in SYNC Mode

Figure 151 shows NAND Command + Address + Command + Read operation in SYNC mode.

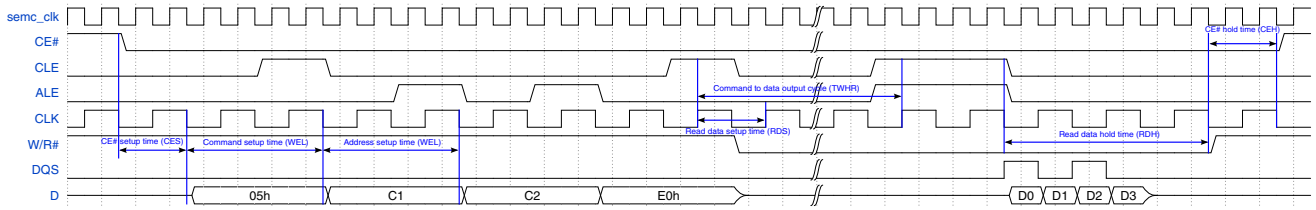


Figure 151. NAND Command + Address + Command + Read in SYNC Mode

NOTE

- semc_clk is the SEMC internal functional clock. CLK is the synchronous clock to latch command and address cycles. The frequency of CLK is half of semc_clk. For example, if semc_clk frequency is 200 MHz, the CLK frequency is 100 MHz.
- For this example:
  - CE# setup time is 2 clock cycles (NANDCR1[CES]=1)
  - Command/Address setup time is 4 clock cycles (NANDCR1[WEL]=2)
  - NANDCR3[RDS] and NANDCR2[TWHR] control the timing from command to data read phase. In general, NANDCR3[RDS] controls the minimum delay before data read phase. In this figure, NANDCR3[RDS]=0.
  - Read data hold time is 6 clock cycles (NANDCR3[RDH]=4)
  - CE# hold time is 2 clock cycles (NANDCR1[CEH]=1)

29.3.1.5.7.2 NAND Command + Address + Write Operation in SYNC Mode

Figure 152 shows NAND Command + Address + Write operation in SYNC mode.

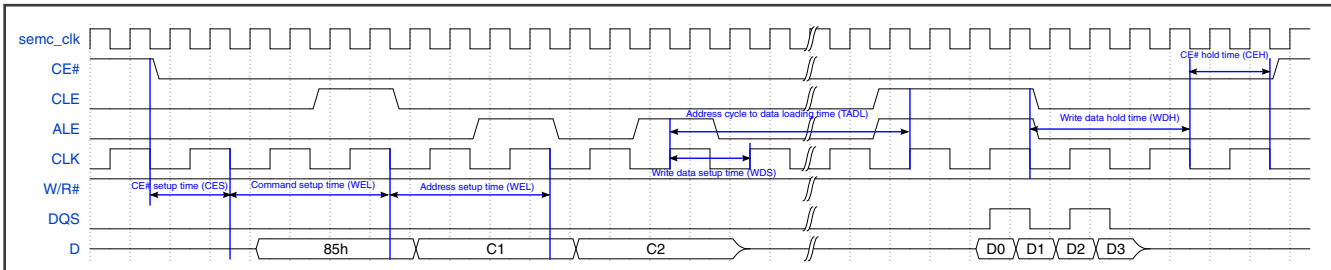


Figure 152. NAND Command + Address + Write in SYNC Mode

NOTE

- For this example:
  - CE# setup time is 2 clock cycles (NANDCR1[CES]=1)
  - Command/Address setup time is 4 clock cycles (NANDCR1[WEL]=2)
  - NANDCR3[WDS] and NANDCR2[TADL] control the timing from command to data write phase. In general, NANDCR3[WDS] controls the minimum delay before data write phase. In this figure, NANDCR3[WDS]=0.
  - Write data hold time is 4 clock cycles (NANDCR3[WDH]=2)
  - CE# hold time is 2 clock cycles (NANDCR1[CEH]=1)

29.3.1.6 NOR Flash Controller Operations

This section describes the operations of NOR Flash controller.

29.3.1.6.1 NOR Flash Address Multiplexing

NOR Flash use a multiplexed address split into rows and columns. Figure 153 shows how the bits of the linear address sent to the SEMC are split up into the row and column addresses sent to the NOR Flash memory device.

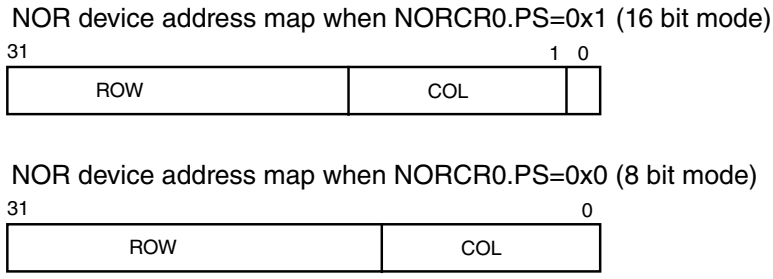


Figure 153. NOR Flash Address Multiplexing

NOTE

- NOR address bits consist of row and column address bits.
- Column address bits offset is bit 0 if NORCR0[PS]=0 and bit 1 if NORCR0[PS]=1. Column address bit width is determined by NORCR0[COL].
- Row address bits offset is determined by NORCR0[PS] and NORCR0[COL].
- NOR device base address on SoC is configured by BR5 register.
- Write data should be 16 bit aligned when connecting to 16 bit NOR device.

29.3.1.6.2 NOR Flash Read Operation in ASYNC Mode

NOR Flash controller supports following address modes, which setting by NORCR0[AM].

Figure 154 shows the ASYNC NOR Flash read in Non-ADMUX address mode.

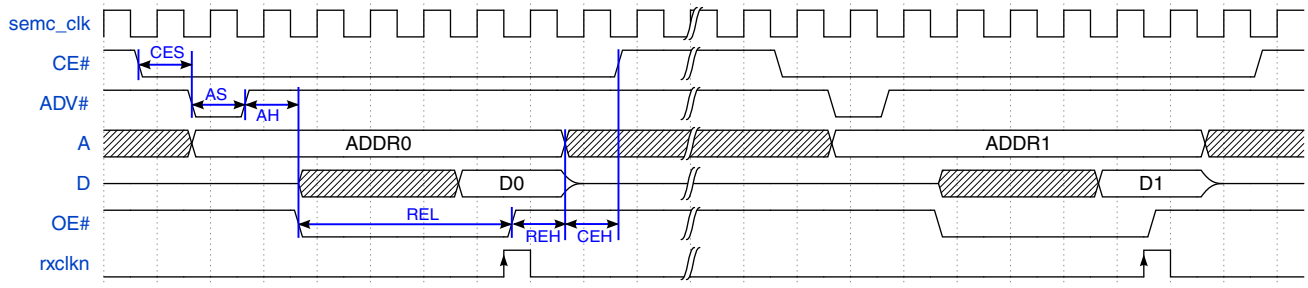


Figure 154. ASYNC NOR Flash Read in Non-ADMUX Address Mode

NOTE

- semc_clk and rxclkn are the internal signals. semc_clk is the SEMC functional clock. rxclkn is the clock to sample data from NOR Flash.
- For this example:
  - CE# setup time is 1 clock cycle (NORCR1[CES]=0)
  - Address setup time is 1 clock cycle (NORCR1[AS]=0)
  - Address hold time is 1 clock cycle (NORCR1[AH]=0)
  - OE# low time is 4 clock cycles (NORCR1[REL]=3)
  - OE# high time is 1 clock cycle (NORCR1[REH]=0)
  - CE# hold time is 1 clock cycle (NORCR1[CEH]=0)

Figure 155 shows the ASYNC NOR Flash read in ADMUX address mode.

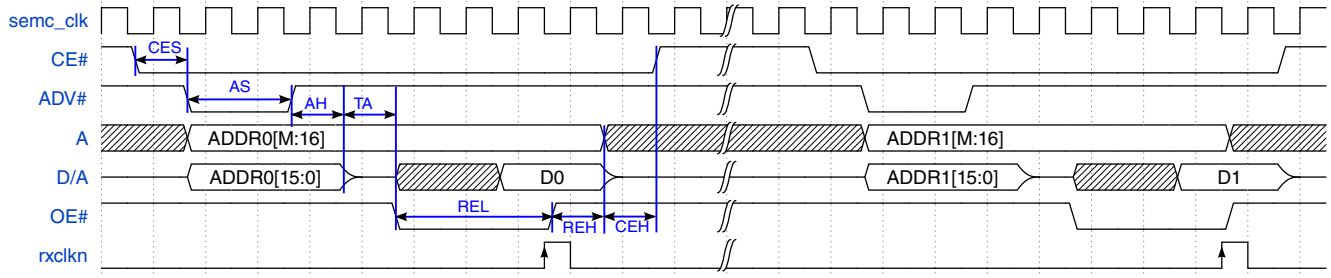


Figure 155. ASYNC NOR Flash Read in ADMUX Address Mode

NOTE

- semc_clk and rxclkn are the internal signals. semc_clk is the SEMC functional clock. rxclkn is the clock to sample data from NOR Flash.
- For this example:
  - CE# setup time is 1 clock cycle (NORCR1[CES]=0)
  - Address setup time is 2 clock cycles (NORCR1[AS]=1)
  - Address hold time is 1 clock cycle (NORCR1[AH]=0)
  - Turnaround time is 1 clock cycle (NORCR2[TA]=0)
  - OE# low time is 3 clock cycles (NORCR1[REL]=2)
  - OE# high time is 1 clock cycle (NORCR1[REH]=0)
  - CE# hold time is 1 clock cycle (NORCR1[CEH]=0)

Figure 156 shows the ASYNC NOR Flash read in AADM address mode.

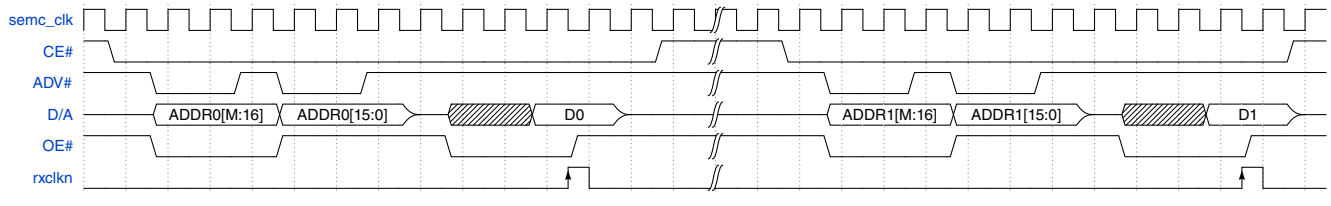


Figure 156. ASYNC NOR Flash Read in AADM Address Mode

29.3.1.6.3 NOR Flash Write Operation in ASYNC Mode

Figure 157 shows the ASYNC NOR Flash write in Non-ADMUX address mode.

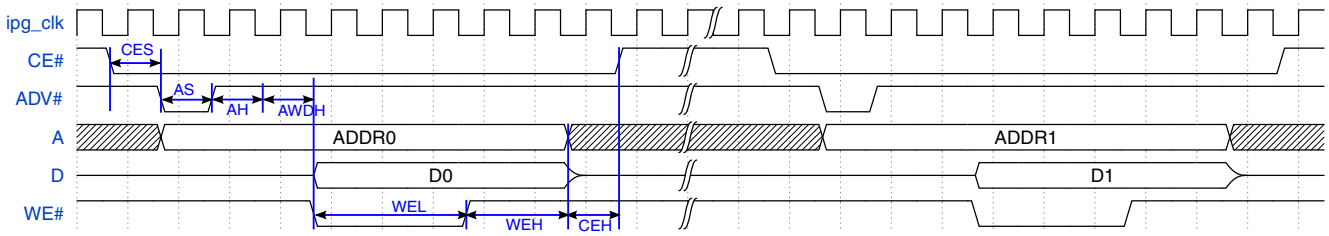


Figure 157. ASYNC NOR Flash Write in Non-ADMUX Address Mode

NOTE

- semc_clk is the SEMC functional clock.
- For this example:
  - CE# setup time is 1 clock cycle (NORCR1[CES]=0)
  - Address setup time is 1 clock cycle (NORCR1[AS]=0)
  - Address hold time is 1 clock cycle (NORCR1[AH]=0)
  - Address to write data hold time is 1 clock cycle (NORCR2[AWDH]=1)
  - WE# low time is 3 clock cycles (NORCR1[WEL]=2)
  - WE# high time is 2 clock cycles (NORCR1[WEH]=1)
  - CE# hold time is 1 clock cycle (NORCR1[CEH]=0)

Figure 158 shows the ASYNC NOR Flash write in ADMUX address mode.

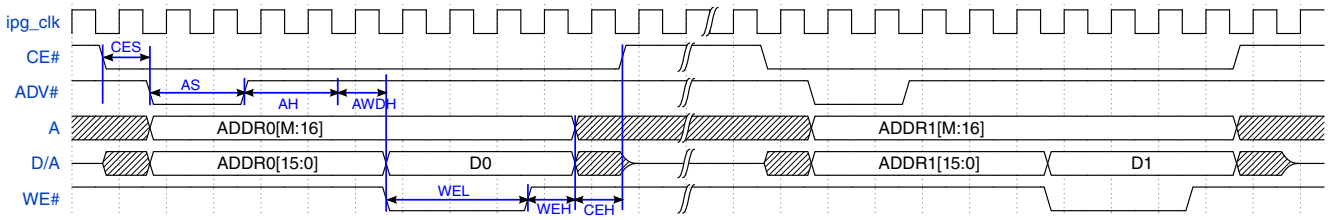


Figure 158. ASYNC NOR Flash Write in ADMUX Address Mode

NOTE

- semc_clk is the SEMC functional clock.
- For this example:
  - CE# setup time is 1 clock cycle (NORCR1[CES]=0)
  - Address setup time is 2 clock cycles (NORCR1[AS]=1)
  - Address hold time is 2 clock cycles (NORCR1[AH]=1)
  - Address to write data hold time is 1 clock cycle (NORCR2[AWDH]=1)
  - WE# low time is 3 clock cycles (NORCR1[WEL]=2)
  - WE# high time is 1 clock cycle (NORCR1[WEH]=0)
  - CE# hold time is 1 clock cycle (NORCR1[CEH]=0)

Figure 159 shows the ASYNC NOR Flash write in AADM address mode.

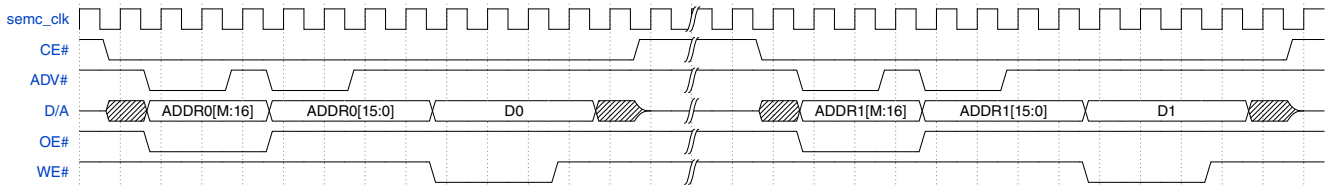


Figure 159. ASYNC NOR Flash Write in AADM Address Mode

29.3.1.6.4 NOR Flash Read Operation in SYNC Mode

Figure 160 shows the SYNC NOR Flash read in Non-ADMUX address mode.

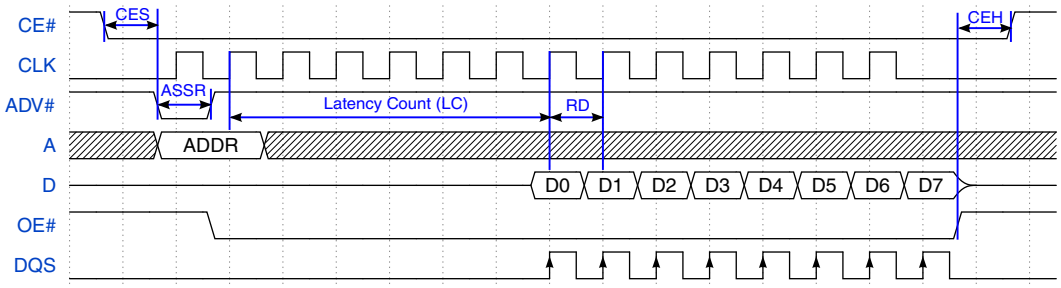


Figure 160. SYNC Flash Read in Non-ADMUX Address Mode

NOTE

- DQS is the clock to sample data from NOR Flash. It generally loops back from DQS pad.
- For this example:
  - CE# setup time is 1 clock cycle (NORCR1[CES]=0)
  - Address setup time is 1 clock cycle (NORCR3[ASSR]=0)
  - Latency count is 6 clock cycles (NORCR2[LC]=6)
  - Read time is 1 clock cycle (NORCR2[RD]=0)
  - CE# hold time is 1 clock cycle (NORCR1[CEH]=0)

Figure 161 shows the SYNC NOR Flash read in ADMUX address mode.

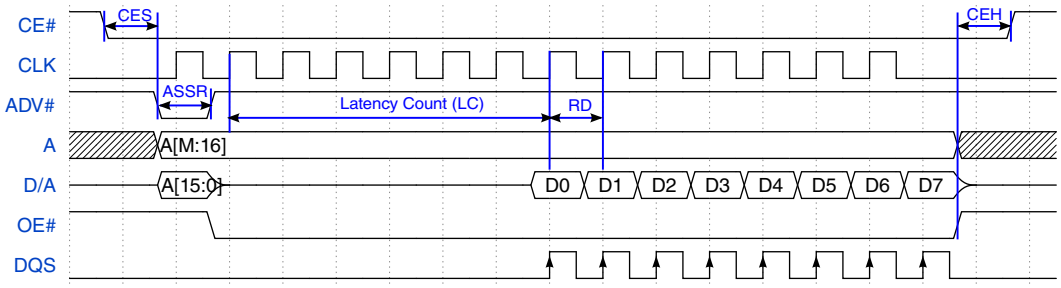


Figure 161. SYNC Flash Read in ADMUX Address Mode

NOTE

- DQS is the clock to sample data from NOR Flash. It generally loops back from DQS pad.
- For this example:
  - CE# setup time is 1 clock cycle (NORCR1[CES]=0)
  - Address setup time is 1 clock cycle (NORCR3[ASSR]=0)
  - Latency count is 6 clock cycles (NORCR2[LC]=6)
  - Read time is 1 clock cycle (NORCR2[RD]=0)
  - CE# hold time is 1 clock cycle (NORCR1[CEH]=0)

Figure 162 shows the SYNC NOR Flash read in AADM address mode.

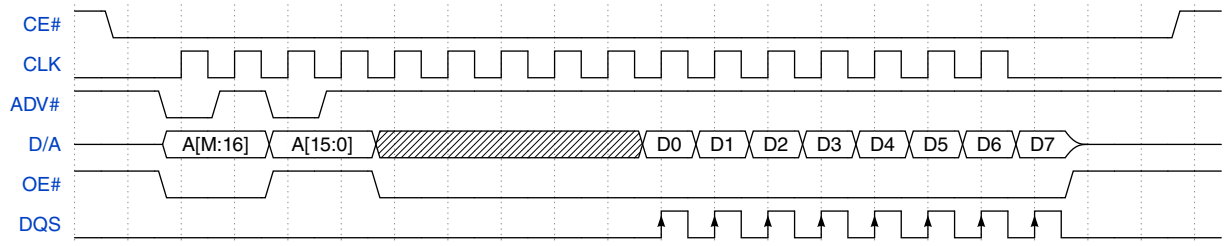


Figure 162. SYNC Flash Read in AADM Address Mode

29.3.1.6.5 NOR Flash Write Operation in SYNC Mode

Flash write operation in SYNC mode is same as asynchronous.

29.3.1.7 SRAM Controller Operations

This section describes the operations of SRAM controller.

29.3.1.7.1 SRAM Address Multiplexing

SRAM use a multiplexed address split into rows and columns. Figure 163 shows how the bits of the linear address sent to the SEMC are split up into the row and column addresses sent to the SRAM memory device.

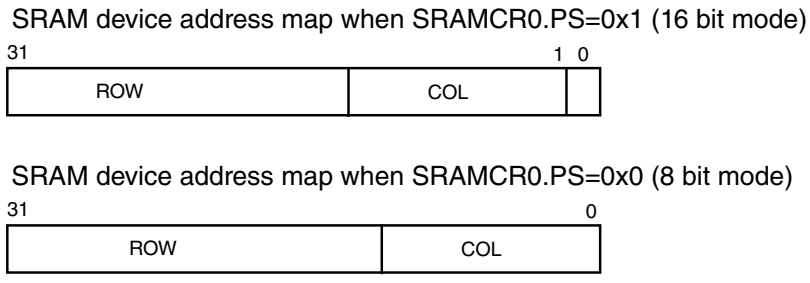


Figure 163. SRAM Address Multiplexing

NOTE

- SRAM address bits consist of row and column address bits.
- Column address bits offset is bit 0 if SRAMCR0[PS]=0, offset is bit 1 if SRAMCR0[PS]=1. Column address bit width is determined by SRAMCR0[COL].
- Row address bits offset is determined by SRAMCR0[PS] and SRAMCR0[COL].
- SRAM device base address on SoC is configured by BR6, BR9, BR10 and BR11 registers.

29.3.1.7.2 SRAM Read Operation in ASYNC Mode

Figure 164 shows the ASYNC SRAM read in Non-ADMUX address mode.

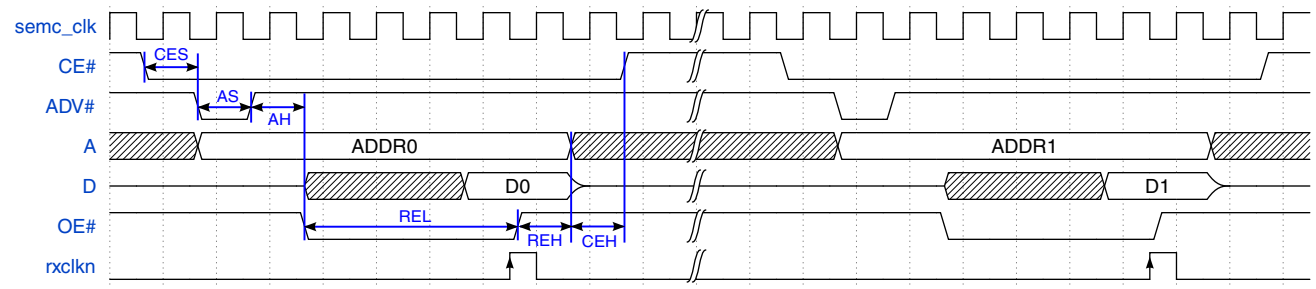


Figure 164. ASYNC SRAM Read in Non-ADMUX Address Mode

NOTE

- semc_clk and rxclkn are the internal signals. semc_clk is the SEMC functional clock. rxclkn is the clock to sample data from SRAM.
- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR1[CES]=0)
  - Address setup time is 1 clock cycle (SRAMCR1[AS]=0)
  - Address hold time is 1 clock cycle (SRAMCR1[AH]=0)
  - OE# low time is 4 clock cycles (SRAMCR1[REL]=3)
  - OE# high time is 1 clock cycle (SRAMCR1[REH]=0)
  - CE# hold time is 1 clock cycle (SRAMCR1[CEH]=0)

Figure 165 shows the ASYNC SRAM read in ADMUX address mode.

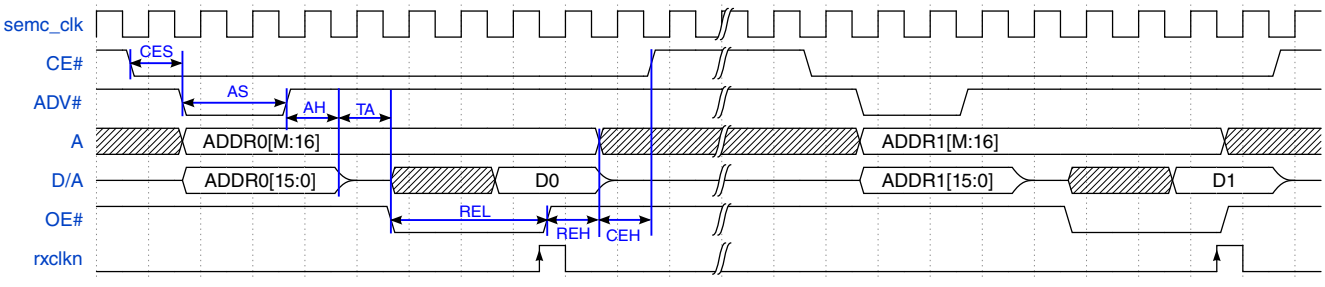


Figure 165. ASYNC SRAM Read in ADMUX Address Mode

NOTE

- semc_clk and rxclkn are the internal signals. semc_clk is the SEMC functional clock. rxclkn is the clock to sample data from SRAM.
- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR1[CES]=0)
  - Address setup time is 2 clock cycles (SRAMCR1[AS]=1)
  - Address hold time is 1 clock cycle (SRAMCR1[AH]=0)
  - Turnaround time is 1 clock cycle (SRAMCR2[TA]=0)
  - OE# low time is 3 clock cycles (SRAMCR1[REL]=2)
  - OE# high time is 1 clock cycle (SRAMCR1[REH]=0)
  - CE# hold time is 1 clock cycle (SRAMCR1[CEH]=0)

Figure 166 shows the ASYNC SRAM read in AADM address mode.

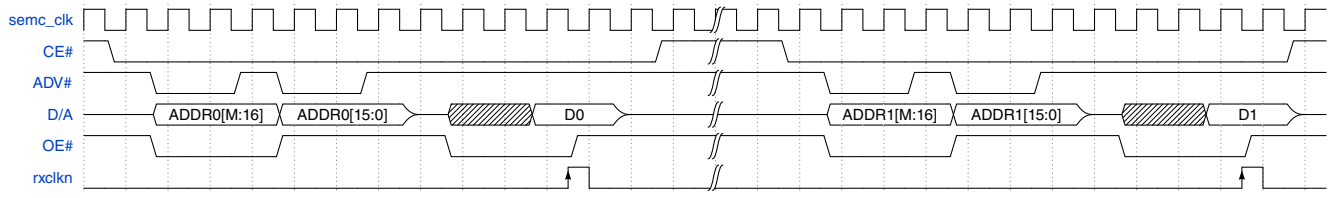


Figure 166. ASYNC SRAM Read in AADM Address Mode

29.3.1.7.3 SRAM Write Operation in ASYNC Mode

Figure 167 shows the ASYNC SRAM write in Non-ADMUX address mode.

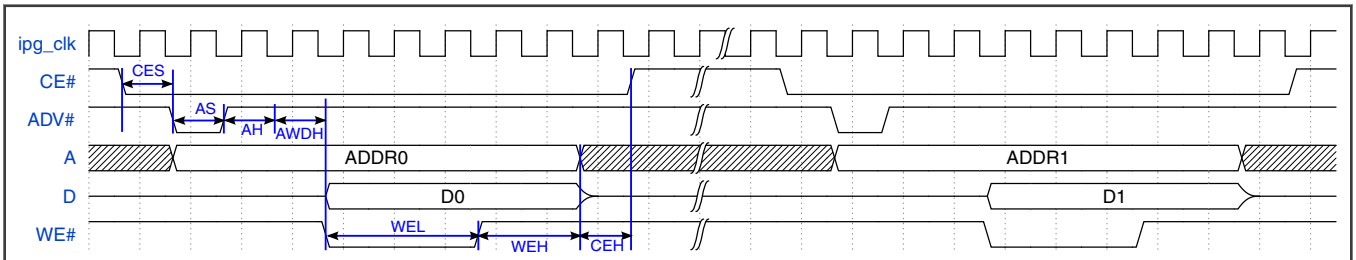


Figure 167. ASYNC SRAM Write in Non-ADMUX Address Mode

NOTE

- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR1[CES]=0)
  - Address setup time is 1 clock cycle (SRAMCR1[AS]=0)
  - Address hold time is 1 clock cycle (SRAMCR1[AH]=0)
  - Address to write data hold time is 1 clock cycle (SRAMCR2[AWDH]=1)
  - WE# low time is 3 clock cycles (SRAMCR1[WEL]=2)
  - WE# high time is 2 clock cycles (SRAMCR1[WEH]=1)
  - CE# hold time is 1 clock cycle (SRAMCR1[CEH]=0)

Figure 168 shows the ASYNC SRAM write in ADMUX address mode.

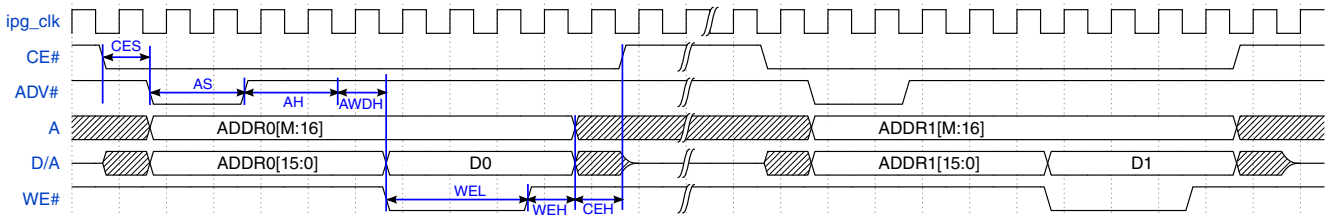


Figure 168. ASYNC SRAM Write in ADMUX Address Mode

NOTE

- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR1[CES]=0)
  - Address setup time is 2 clock cycles (SRAMCR1[AS]=1)
  - Address hold time is 2 clock cycles (SRAMCR1[AH]=1)
  - Address to write data hold time is 1 clock cycle (SRAMCR2[AWDH]=1)
  - WE# low time is 3 clock cycles (SRAMCR1[WEL]=2)
  - WE# high time is 1 clock cycle (SRAMCR1[WEH]=0)
  - CE# hold time is 1 clock cycle (SRAMCR1[CEH]=0)

Figure 169 shows the ASYNC SRAM write in AADM address mode.

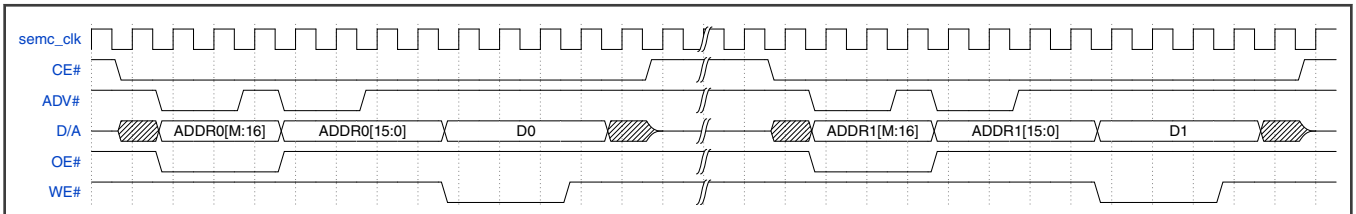


Figure 169. ASYNC SRAM Write in AADM Address Mode

29.3.1.7.4 SRAM Register Operation in ASYNC Mode

Figure 170 shows the ASYNC SRAM register access in ADMUX address mode.

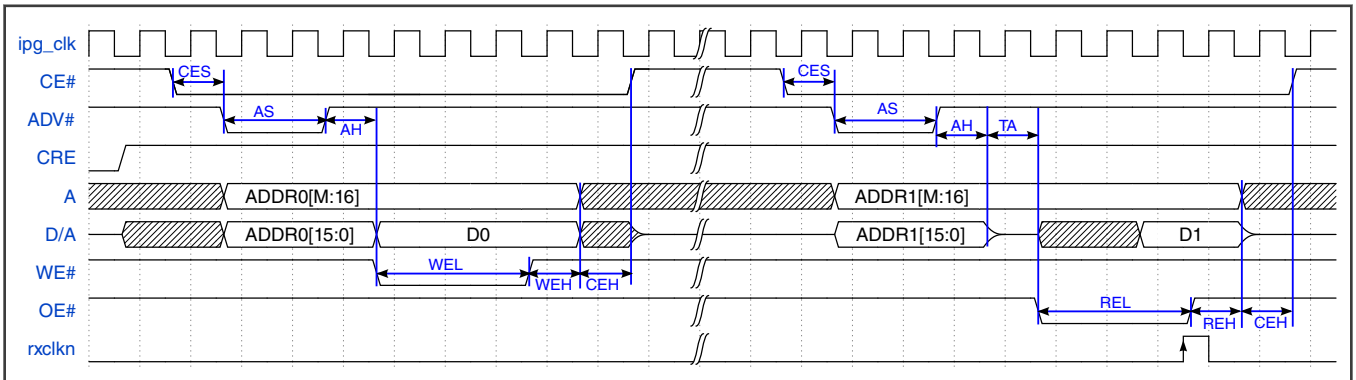


Figure 170. ASYNC SRAM Register Access in ADMUX Address Mode

NOTE

- For this example:
  - First operation is SRAM register write access. ADDR0[M:0] contains the value for writing to SRAM register. The value of D0 can be ignored.
  - Second operation is SRAM register read access. ADDR1[M:0] contains the value for choosing of SRAM register. D1 is the value that read from SRAM register.
  - The value of ADDR0/1 comes from IPCR0 register.
    - If BR6[BA]=0x98000 and IPCR0=0x9810883E, ADDR0/1 is 0x8441F when port size is 16bit mode. The address is divided by 2 in 16bit mode.
    - If BR6[BA]=0x98000 and IPCR0=0x9810883E, ADDR0/1 is 0x10883E when port size is 8bit mode.
  - CE# setup time is 1 clock cycle (SRAMCR1[CES]=0)
  - Address setup time is 2 clock cycles (SRAMCR1[AS]=1)
  - Address hold time is 1 clock cycle (SRAMCR1[AH]=0)
  - WE# low time is 3 clock cycles (SRAMCR1[WEL]=2)
  - WE# high time is 1 clock cycle (SRAMCR1[WEH]=0)
  - CE# hold time is 1 clock cycle (SRAMCR1[CEH]=0)
  - Turnaround time is 1 clock cycle (SRAMCR2[TA]=0)
  - OE# low time is 3 clock cycles (SRAMCR1[REL]=2)
  - OE# high time is 1 clock cycle (SRAMCR1[REH]=0)

29.3.1.7.5 SRAM Read Operation in ASYNC Mode with Wait Pin

SRAM read operation in ASYNC mode with wait pin is indicated in following diagrams:

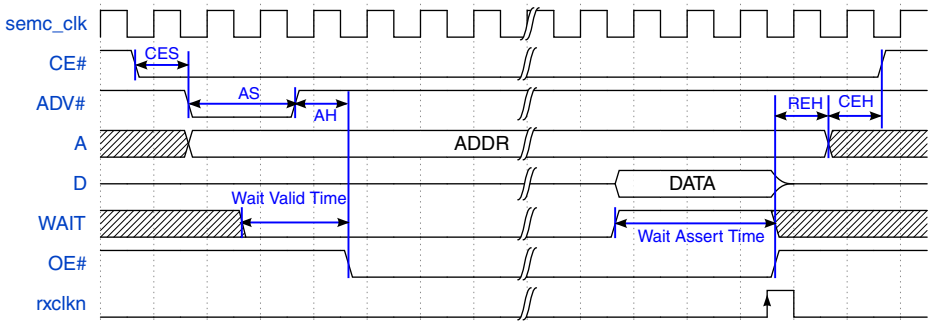


Figure 171. SRAM Read in ASYNC Mode (Non-ADMUX) with Wait Pin

NOTE

1. Wait Valid Time must be no less than 2 semc_clk cycles.
2. Wait and DATA must be valid till OE# high.
3. For this example:
   - SRAMCR0[WAITEN]=1
   - SRAMCR0[WAITSP]=1
   - MCR[WPOL0]=0

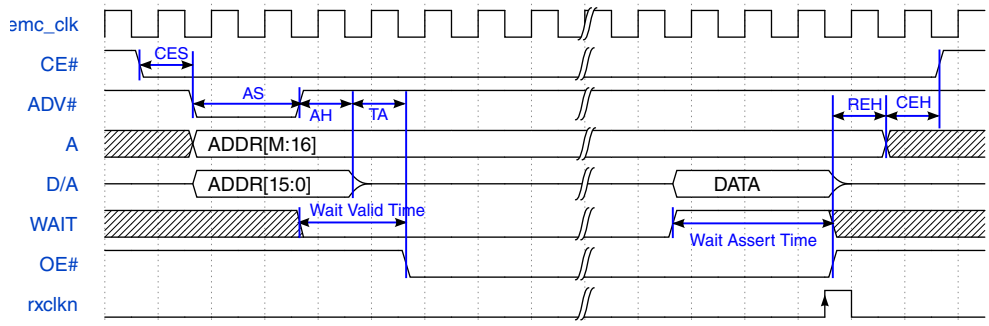


Figure 172. SRAM Read in ASYNC Mode (ADMUX) with Wait Pin

NOTE

1. Wait Valid Time must be no less than 2 semc_clk cycles.
2. Wait and DATA must be valid till OE# high.
3. For this example:
   - SRAMCR0[WAITEN]=1
   - SRAMCR0[WAITSP]=1
   - MCR[WPOL0]=0

29.3.1.7.6 SRAM Write Operation in ASYNC Mode with Wait Pin

SRAM write operation in ASYNC mode is indicated in following diagrams:

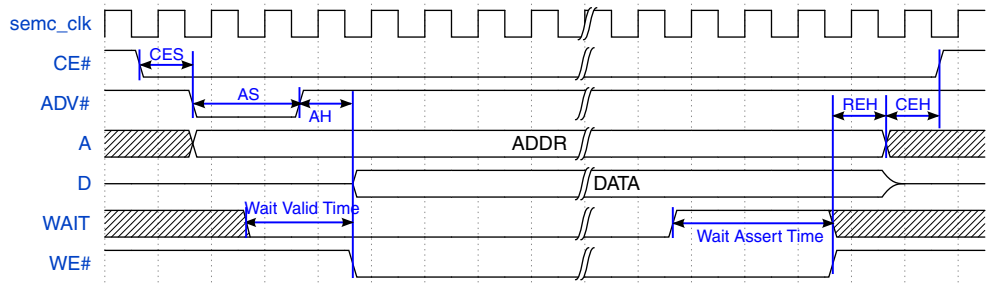


Figure 173. SRAM Write in ASYNC Mode (Non-ADMUX) with Wait Pin

NOTE

1. Wait Valid Time must be no less than 2 semc_clk cycles.
2. Wait must be valid till WE# high.
3. For this example:
   - SRAMCR0[WAITEN]=1
   - SRAMCR0[WAITSP]=1
   - MCR[WPOL0]=0

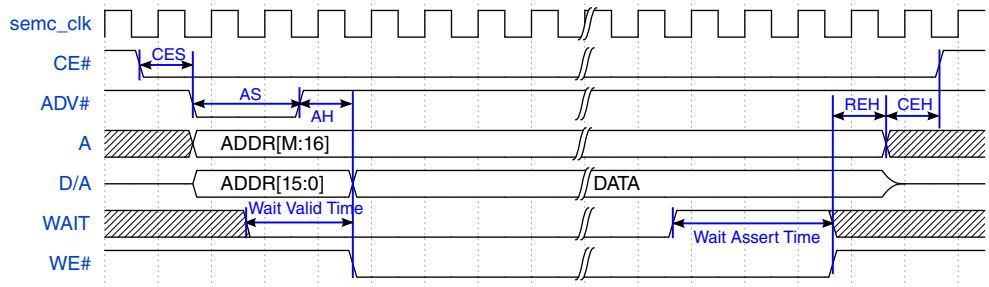


Figure 174. SRAM Write in ASYNC Mode (ADMUX) with Wait Pin

NOTE

1. Wait Valid Time must be no less than 2 semc_clk cycles.
2. Wait must be valid till WE# high.
3. For this example:
   - SRAMCR0[WAITEN]=1
   - SRAMCR0[WAITSP]=1
   - MCR[WPOL0]=0

29.3.1.7.7 SRAM Read Operation in SYNC Mode

Figure 175 shows the SYNC SRAM read in Non-ADMUX address mode.

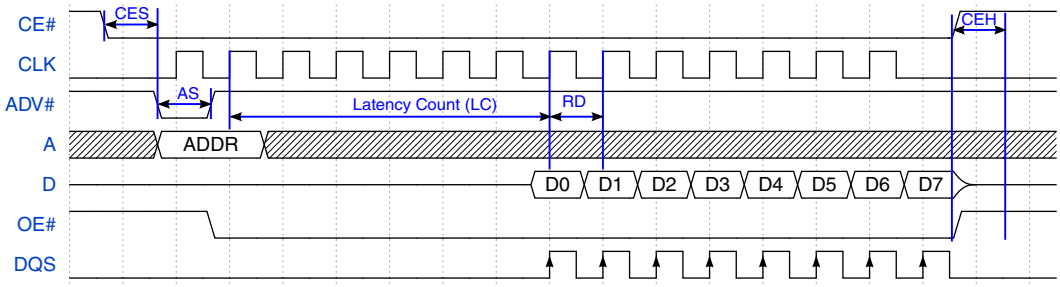


Figure 175. SYNC SRAM Read in Non-ADMUX Address Mode

NOTE

- DQS is the clock to sample data from SRAM. It generally loops back from DQS pad.
- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR1[CES]=0)
  - Address setup time is 1 clock cycle (SRAMCR1[AS]=0)
  - Latency count is 6 clock cycles (SRAMCR2[LC]=6)
  - Read time is 1 clock cycle (SRAMCR2[RD]=0)
  - CE# hold time is 1 clock cycle (SRAMCR1[CEH]=0)

Figure 176 shows the SYNC SRAM read in ADMUX address mode.

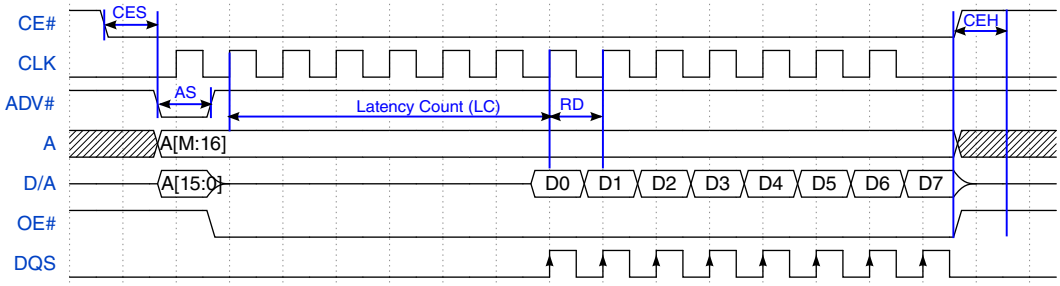


Figure 176. SYNC SRAM Read in ADMUX Address Mode

NOTE

- DQS is the clock to sample data from SRAM. It generally loops back from DQS pad.
- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR1[CES]=0)
  - Address setup time is 1 clock cycle (SRAMCR1[AS]=0)
  - Latency count is 6 clock cycles (SRAMCR2[LC]=6)
  - Read time is 1 clock cycle (SRAMCR2[RD]=0)
  - CE# hold time is 1 clock cycle (SRAMCR1[CEH]=0)

29.3.1.7.8 SRAM Write Operation in SYNC Mode

Figure 177 shows the SYNC SRAM write in Non-ADMUX address mode.

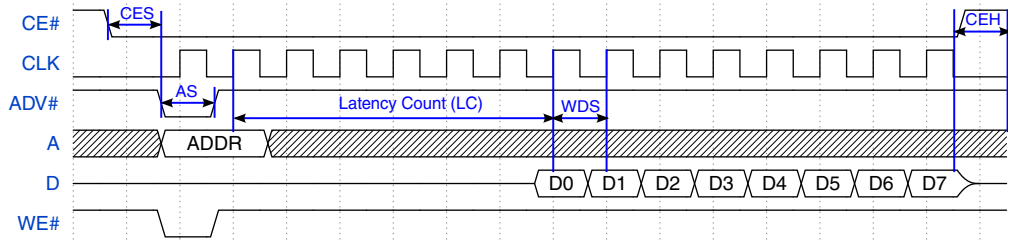


Figure 177. SRAM Write in SYNC Mode (Non-ADMUX)

NOTE

- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR1[CES]=0)
  - Address setup time is 1 clock cycle (SRAMCR1[AS]=0)
  - Latency count is 6 clock cycles (SRAMCR2[LC]=6)
  - Write data setup time is 1 clock cycle (SRAMCR2[WDS]=0)
  - CE# hold time is 1 clock cycle (SRAMCR1[CEH]=0)

Figure 178 shows the SYNC SRAM write in ADMUX address mode.

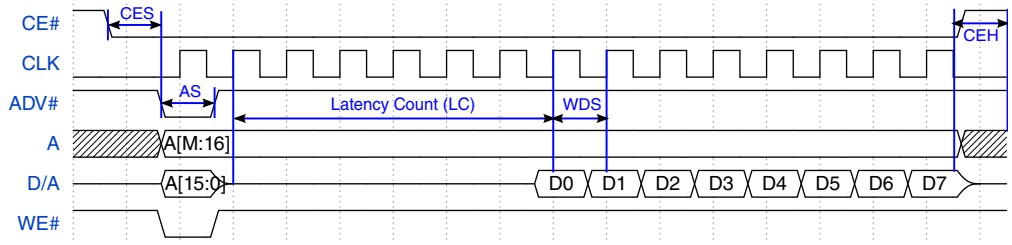


Figure 178. SRAM Write in SYNC Mode (ADMUX)

NOTE

- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR1[CES]=0)
  - Address setup time is 1 clock cycle (SRAMCR1[AS]=0)
  - Latency count is 6 clock cycles (SRAMCR2[LC]=6)
  - Write data setup time is 1 clock cycle (SRAMCR2[WDS]=0)
  - CE# hold time is 1 clock cycle (SRAMCR1[CEH]=0)

29.3.1.8 Display Bus Interface Controller Operations

This section describes the operations of DBI controller.

DCX signal is controlled by write address. Write access to DBI is handled as DBI COMMAND WRITE transaction if address is higher than 0x10000 (DCX low). Otherwise, it is handled as DBI DATA WRITE transaction (DCX high).

For example, if BR7[BA]=0x9C000:

- An AXI write to address 0x9C010000 is treated as DBI COMMAND WRITE transaction.
- An AXI write to address 0x9C000000 is treated as DBI DATA WRITE transaction.

Figure 179 shows DBI write operation.

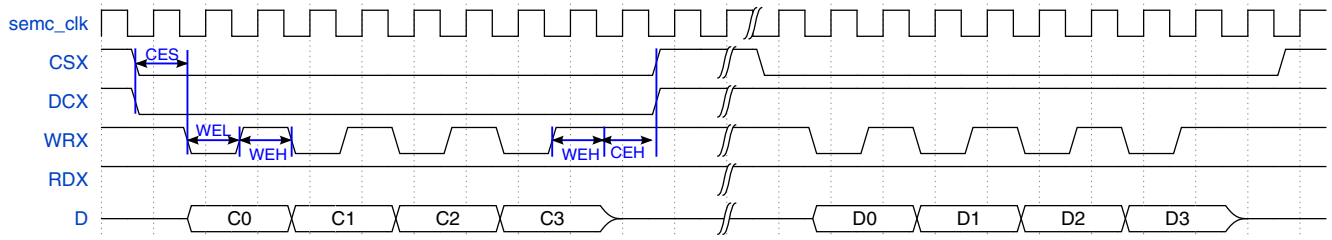


Figure 179. DBI Write Operation

NOTE

- DBI address is used to determine DBI COMMAND or DATA WRITE.
- DBI bus never send out address information for COMMAND and DATA WRITE.
- AXI WRAP write access to DBI is supported.
- DBI bus interface does not support WRITE mask feature.
- All write strobe in AXI write access to DBI must be valid. If AXI write strobe is invalid, the SEMC sends write data 0x00 instead of original write data.
- When DBI bus is 16 bit (DBICR0[PS]=1), AXI or IP command access start address to DBI bus must be 16 bit aligned. Otherwise the write data to DBI may be wrong. There is no bus error response or internal error status generated for this case.
- For this example:
  - CSX setup time is 1 clock cycle (DBICR1[CES]=0)
  - WRX low time is 1 clock cycle (DBICR1[WEL]=0)
  - WRX high time is 1 clock cycle (DBICR1[WEH]=0)
  - CSX hold time is 1 clock cycle (DBICR1[CEH]=0)

Figure 180 shows DBI read operation.

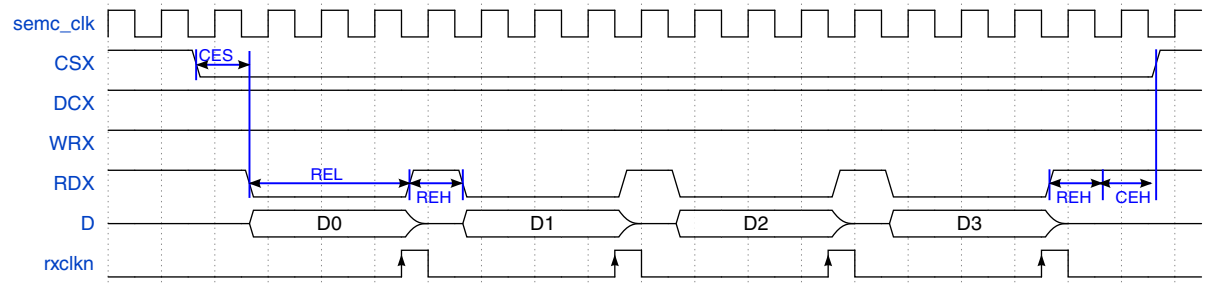


Figure 180. DBI Read Operation

NOTE

- Read access to DBI is always handled as DBI DATA READ transaction.
- DBI bus never sent out address information for DATA READ command.
- AXI WRAP read access to DBI is not supported except when the first beat address offset is zero in the burst.
- When DBI bus is 16 bit (DBICR0[PS]=1), AXI command or IP command access start address to DBI bus must be 16 bit aligned. Otherwise the read data is unknown. There is no bus error response or internal error status generated for this case.
- For this example:
  - CSX setup time is 1 clock cycle (DBICR1[CES]=0)
  - RDX low time is 3 clock cycles (DBICR1[REL]=2)
  - RDX high time is 1 clock cycle (DBICR1[REH]=0)
  - CSX hold time is 1 clock cycle (DBICR1[CEH]=0)

29.3.2 Clocks

There is one clock used for the SEMC AXI bus, IP bus, internal logic and external interface. It is shown as semc_clk in SEMC timing figures.

29.3.3 Reset

There is a hardware reset for the SEMC, which reset all the logic inside it. When set MCR[SWRST], there will be a software reset to reset all logic in the SEMC except configuration registers.

29.3.4 Interrupts

There is an interrupt output from the SEMC. It indicates the interrupt source that defined in INTR register. The interrupt source is enabled by INTEN register.

29.4 External Signals

Table 188 lists the SEMC module signals.

Table 188. SEMC Signals

Signal	Direction	Description
SEMC_DATA[31-00]	I/O	Data/Address Bits The SEMC has several controllers, and they share pins. SEMC_DATA works as data signal in SDRAM/NAND/8080 applications and works as data or address signal in NOR/SRAM applications. For detail pin mux information, refer to Pin Mux.
SEMC_DM[3-0]	O	Write Data Mask Bits
SEMC_ADDR[12-00]	O	Address Bits SEMC_ADDR08 can work as chip select pin in NAND/NOR/SRAM/ 8080 applications.
SEMC_BA[1-0]	O	Bank Address Bits SEMC_BA0 can work as WAIT in SRAM application. SEMC_BA1 can work as ALE/ADV#/DCX in NAND/NOR/SRAM/8080 applications.
SEMC_CAS	O	SDRAM Column Address Strobe
SEMC_RAS	O	SDRAM Row Address Strobe
SEMC_WE	O	SDRAM Write Enable
SEMC_CKE	O	SDRAM Clock enable
SEMC_CLK	O	SDRAM Clock
SEMC_CS0	O	SDRAM Chip Select 0
SEMC_CSX[3-0]	O	Configurable Chip Select bits
SEMC_RDY	I/O	Ready input pin for NAND It can also work as chip select pin in SDRAM/NOR/SRAM/ 8080 applications.
SEMC_DQS	I/O	DQS SEMC controller provides internal dummy read strobe for read data from SDRAM/NOR/SRAM. Higher read frequency can be achieved by looping back this dummy read strobe from pad. This pin can be floated

Table continues on the next page...

Table 188. SEMC Signals (continued)

Signal	Direction	Description
		or put some cap loads on board level to compensate DATA/CLK pins load. DCCR register can also be configured to compensate DATA/CLK timing delays.
SEMC_DQS4	I/O	DQS Data read strobe for NAND Flash.
SEMC_CLKX0	O	Configurable clock 0 for NOR/SRAM
SEMC_CLKX1	O	Configurable clock 1 for NOR/SRAM

29.4.1 Pin Mux

Table 189 lists the pin mux of the SEMC module.

Table 189. SEMC Pin Mux Overview

Signal	SDRAM	NAND	NOR (ADMUX 16bit)	SRAM (ADMUX 16bit)	DBI	Comment
SEMC_DATA00	D0	D0	D0/A0	D0/A0	D0	Hardware MUX The SEMC determines the pin function according to its internal state
SEMC_DATA01	D1	D1	D1/A1	D1/A1	D1	
SEMC_DATA02	D2	D2	D2/A2	D2/A2	D2	
SEMC_DATA03	D3	D3	D3/A3	D3/A3	D3	
SEMC_DATA04	D4	D4	D4/A4	D4/A4	D4	
SEMC_DATA05	D5	D5	D5/A5	D5/A5	D5	
SEMC_DATA06	D6	D6	D6/A6	D6/A6	D6	
SEMC_DATA07	D7	D7	D7/A7	D7/A7	D7	
SEMC_DATA08	D8	D8	D8/A8	D8/A8	D8	
SEMC_DATA09	D9	D9	D9/A9	D9/A9	D9	
SEMC_DATA10	D10	D10	D10/A10	D10/A10	D10	
SEMC_DATA11	D11	D11	D11/A11	D11/A11	D11	
SEMC_DATA12	D12	D12	D12/A12	D12/A12	D12	
SEMC_DATA13	D13	D13	D13/A13	D13/A13	D13	
SEMC_DATA14	D14	D14	D14/A14	D14/A14	D14	
SEMC_DATA15	D15	D15	D15/A15	D15/A15	D15	
SEMC_DATA16	D16	-	A24	A24	-	
SEMC_DATA17	D17	-	A25	A25	-	
SEMC_DATA18	D18	-	A26	A26	-	
SEMC_DATA19	D19	-	A27	A27	-	

Table continues on the next page...

Table 189. SEMC Pin Mux Overview (continued)

Signal	SDRAM	NAND	NOR (ADMUX 16bit)	SRAM (ADMUX 16bit)	DBI	Comment
SEMC_DATA20	D20	-	-	-	-	
SEMC_DATA21	D21	-	-	-	-	
SEMC_DATA22	D22	-	-	-	-	
SEMC_DATA23	D23	-	-	-	-	
SEMC_DATA24	D24	-	-	-	-	
SEMC_DATA25	D25	-	-	-	-	
SEMC_DATA26	D26	-	-	-	-	
SEMC_DATA27	D27	-	-	-	-	
SEMC_DATA28	D28	-	-	-	-	
SEMC_DATA29	D29	-	-	-	-	
SEMC_DATA30	D30	-	-	-	-	
SEMC_DATA31	D31	-	-	-	-	
SEMC_DM3	DQM3	-	-	-	-	
SEMC_DM2	DQM2	-	-	-	-	
SEMC_DM1	DQM1	-	-	UB#	-	
SEMC_DM0	DQM0	-	-	LB#	-	
SEMC_ADDR00	A0	-	A16	A16	-	
SEMC_ADDR01	A1	-	A17	A17	-	
SEMC_ADDR02	A2	-	A18	A18	-	
SEMC_ADDR03	A3	-	A19	A19	-	
SEMC_ADDR04	A4	-	A20	A20	-	
SEMC_ADDR05	A5	-	A21	A21	-	
SEMC_ADDR06	A6	-	A22	A22	-	
SEMC_ADDR07	A7	-	A23	A23	-	
SEMC_ADDR08	A8	CE#	CE#/A24	CE#/A24	CSX	Configured by IOCR[MUX_A8]
SEMC_ADDR09	A9	CLE	-	CRE	-	Hardware MUX The SEMC determines the pin function according to its internal state
SEMC_ADDR10	A10	-	-	-	-	
SEMC_ADDR11	A11	WE#	WE#	WE#	WRX	SEMC_ADDR11 works as CLK signal in NAND synchronous mode
SEMC_ADDR12	A12	RE#	OE#	OE#	RDX	SEMC_ADDR12 works as W/R# signal in NAND synchronous mode

Table continues on the next page...

Table 189. SEMC Pin Mux Overview (continued)

Signal	SDRAM	NAND	NOR (ADMUX 16bit)	SRAM (ADMUX 16bit)	DBI	Comment
SEMC_BA0	BA0	-	-	WAIT	-	Hardware MUX The SEMC determines the pin function according to its internal state
SEMC_BA1	BA1	ALE	ADV#	ADV#	DCX	
SEMC_CAS	CAS#	-	-	-	-	
SEMC_RAS	RAS#	-	-	-	-	
SEMC_WE	WE#	-	-	-	-	
SEMC_CKE	CKE	-	-	-	-	
SEMC_CLK	CLK	-	-	-	-	
SEMC_CS0	CS0	-	-	-	-	
SEMC_CSX0	CS	CE#	CE#	CE#	CSX	Configured by IOCR[MUX_CSX0]
SEMC_CSX1	CS	CE#	CE#	CE#	CSX	Configured by IOCR[MUX_CSX1]
SEMC_CSX2	CS	CE#	CE#	CE#	CSX	Configured by IOCR[MUX_CSX2]
SEMC_CSX3	CS	CE#	CE#	CE#	CSX	Configured by IOCR[MUX_CSX3]
SEMC_RDY	CS	R/B#	CE#	CE#	CSX	Configured by IOCR[MUX_RDY]
SEMC_DQS	DQS	-	DQS	DQS	-	Hardware MUX
SEMC_DQS4	-	DQS	-	-	-	Hardware MUX
SEMC_CLKX0	-	-	CLK	CLK	-	Configured by IOCR[MUX_CLKX0]
SEMC_CLKX1	-	-	CLK	CLK	-	Configured by IOCR[MUX_CLKX1]

NOTE

- IOCR configuration should be static. Dynamic remapping of pin muxing controlled by IOCR is not supported.
- NOR and SRAM pins are configured as 16 bit ADMUX mode in above table. It is the recommended configuration.

Table 190 lists the different configurations for NOR and SRAM pin mux of the SEMC module.

Table 190. SEMC Pin Mux for NOR and SRAM

Signal	NOR (Non-ADMUX)	NOR (ADMUX 8bit)	NOR (ADMUX 16bit)	NOR (AADM 16bit)	SRAM (Non-ADMUX)	SRAM (ADMUX 8bit)	SRAM (ADMUX 16bit)	SRAM (AADM 16bit)
SEMC_DATA00	D0	D0/A0	D0/A0	D0/A0	D0	D0/A0	D0/A0	D0/A0
SEMC_DATA01	D1	D1/A1	D1/A1	D1/A1	D1	D1/A1	D1/A1	D1/A1
SEMC_DATA02	D2	D2/A2	D2/A2	D2/A2	D2	D2/A2	D2/A2	D2/A2
SEMC_DATA03	D3	D3/A3	D3/A3	D3/A3	D3	D3/A3	D3/A3	D3/A3
SEMC_DATA04	D4	D4/A4	D4/A4	D4/A4	D4	D4/A4	D4/A4	D4/A4
SEMC_DATA05	D5	D5/A5	D5/A5	D5/A5	D5	D5/A5	D5/A5	D5/A5

Table continues on the next page...

Table 190. SEMC Pin Mux for NOR and SRAM (continued)

Signal	NOR (Non-ADMUX)	NOR (ADMUX 8bit)	NOR (ADMUX 16bit)	NOR (AADM 16bit)	SRAM (Non-ADMUX)	SRAM (ADMUX 8bit)	SRAM (ADMUX 16bit)	SRAM (AADM 16bit)
SEMC_DATA06	D6	D6/A6	D6/A6	D6/A6	D6	D6/A6	D6/A6	D6/A6
SEMC_DATA07	D7	D7/A7	D7/A7	D7/A7	D7	D7/A7	D7/A7	D7/A7
SEMC_DATA08	D8	-	D8/A8	D8/A8	D8	-	D8/A8	D8/A8
SEMC_DATA09	D9	-	D9/A9	D9/A9	D9	-	D9/A9	D9/A9
SEMC_DATA10	D10	-	D10/A10	D10/A10	D10	-	D10/A10	D10/A10
SEMC_DATA11	D11	-	D11/A11	D11/A11	D11	-	D11/A11	D11/A11
SEMC_DATA12	D12	-	D12/A12	D12/A12	D12	-	D12/A12	D12/A12
SEMC_DATA13	D13	-	D13/A13	D13/A13	D13	-	D13/A13	D13/A13
SEMC_DATA14	D14	-	D14/A14	D14/A14	D14	-	D14/A14	D14/A14
SEMC_DATA15	D15	-	D15/A15	D15/A15	D15	-	D15/A15	D15/A15
SEMC_DATA16	A8	A16	A24	-	A8	A16	A24	-
SEMC_DATA17	A9	A17	A25	-	A9	A17	A25	-
SEMC_DATA18	A10	A18	A26	-	A10	A18	A26	-
SEMC_DATA19	A11	A19	A27	-	A11	A19	A27	-
SEMC_DATA20	A12	A20	-	-	A12	A20	-	-
SEMC_DATA21	A13	A21	-	-	A13	A21	-	-
SEMC_DATA22	A14	A22	-	-	A14	A22	-	-
SEMC_DATA23	A15	A23	-	-	A15	A23	-	-
SEMC_DATA24	A16	A24	-	-	A16	A24	-	-
SEMC_DATA25	A17	A25	-	-	A17	A25	-	-
SEMC_DATA26	A18	A26	-	-	A18	A26	-	-
SEMC_DATA27	A19	A27	-	-	A19	A27	-	-
SEMC_DATA28	A20	-	-	-	A20	-	-	-
SEMC_DATA29	A21	-	-	-	A21	-	-	-
SEMC_DATA30	A22	-	-	-	A22	-	-	-
SEMC_DATA31	A23	-	-	-	A23	-	-	-
SEMC_DM3	-	-	-	-	-	-	-	-
SEMC_DM2	-	-	-	-	-	-	-	-
SEMC_DM1	-	-	-	-	UB#	-	UB#	UB#
SEMC_DM0	-	-	-	-	LB#	-	LB#	LB#
SEMC_ADDR00	A0	A8	A16	-	A0	A8	A16	-

Table continues on the next page...

Table 190. SEMC Pin Mux for NOR and SRAM (continued)

Signal	NOR (Non-ADMUX)	NOR (ADMUX 8bit)	NOR (ADMUX 16bit)	NOR (AADM 16bit)	SRAM (Non-ADMUX)	SRAM (ADMUX 8bit)	SRAM (ADMUX 16bit)	SRAM (AADM 16bit)
SEMC_ADDR01	A1	A9	A17	-	A1	A9	A17	-
SEMC_ADDR02	A2	A10	A18	-	A2	A10	A18	-
SEMC_ADDR03	A3	A11	A19	-	A3	A11	A19	-
SEMC_ADDR04	A4	A12	A20	-	A4	A12	A20	-
SEMC_ADDR05	A5	A13	A21	-	A5	A13	A21	-
SEMC_ADDR06	A6	A14	A22	-	A6	A14	A22	-
SEMC_ADDR07	A7	A15	A23	-	A7	A15	A23	-
SEMC_ADDR08	CE#	CE#	CE#	CE#	CE#	CE#	CE#	CE#
SEMC_ADDR09	-	-	-	-	CRE	CRE	CRE	CRE
SEMC_ADDR10	-	-	-	-	-	-	-	-
SEMC_ADDR11	WE#	WE#	WE#	WE#	WE#	WE#	WE#	WE#
SEMC_ADDR12	OE#	OE#	OE#	OE#	OE#	OE#	OE#	OE#
SEMC_BA0	-	-	-	-	WAIT	WAIT	WAIT	WAIT
SEMC_BA1	ADV#	ADV#	ADV#	ADV#	ADV#	ADV#	ADV#	ADV#
SEMC_CAS	-	-	-	-	-	-	-	-
SEMC_RAS	-	-	-	-	-	-	-	-
SEMC_WE	-	-	-	-	-	-	-	-
SEMC_CKE	-	-	-	-	-	-	-	-
SEMC_CLK	-	-	-	-	-	-	-	-
SEMC_CS0	-	-	-	-	-	-	-	-
SEMC_CSX0	CE#/A24	CE#	CE#	CE#	CE#/A24	CE#	CE#	CE#
SEMC_CSX1	CE#/A25	CE#	CE#	CE#	CE#/A25	CE#	CE#	CE#
SEMC_CSX2	CE#/A26	CE#	CE#	CE#	CE#/A26	CE#	CE#	CE#
SEMC_CSX3	CE#/A27	CE#	CE#	CE#	CE#/A27	CE#	CE#	CE#
SEMC_RDY	CE#/A27	CE#	CE#	CE#	CE#/A27	CE#	CE#	CE#
SEMC_DQS	DQS	DQS	DQS	DQS	DQS	DQS	DQS	DQS
SEMC_DQS4	-	-	-	-	-	-	-	-
SEMC_CLKX0	CLK	CLK	CLK	CLK	CLK	CLK	CLK	CLK
SEMC_CLKX1	CLK	CLK	CLK	CLK	CLK	CLK	CLK	CLK

29.5 Initialization

To initialize the block:

- Ensure the clock for the SEMC is enabled.
- Configure corresponding pin MUX for the SEMC external signals.
- Configure registers of the SEMC properly, according to the applications.
- Configure the Module Control Register and clear MCR[MDIS] bit to enable the SEMC module.
- Reset SEMC optionally by setting the MCR[SWRST] bit.

29.6 Application Information

This section describes applications supported by the SEMC module.

29.6.1 SDRAM Application

This section provides the example sequences for SDRAM device.

- Configure MCR[DQSMD] bit to select the read clock source. Suggest to set it with 0x1 to reach high clock frequency.
- Configure IOCR register to choose chip select pins.
- Configure BMCR0/1 registers to adjust AXI bus access efficiency scheme.
- Configure Base Register 0/1/2/3 with base address, memory size and valid information.
- Configure INTEN and INTR registers if need to generate interrupt.
- Configure SDRAM Control Registers with valid settings that matches SDRAM device.
- Initialize SDRAM device by IP command registers (IPCR0/1/2, IPCMD and IPTXDAT).
- Write and read operations can be triggered by AXI or IP command. However, AXI command is suggested.

29.6.2 NAND Application

This section provides the example sequences for NAND device.

- Configure IOCR register to choose chip select pins.
- Configure BMCR0/1 registers to adjust AXI bus access efficiency scheme.
- Configure Base Register 4/8 with base address, memory size and valid information.
- Configure INTEN and INTR registers if need to generate interrupt.
- Configure NAND Control Registers with valid settings that matches NAND device.
- Configure DLL Control Register for synchronous mode.
- Initialize NAND device by IP command registers (IPCR0/1/2, IPCMD and IPTXDAT).
- For programming operation, need to open page by IP command. Then you can write data to the NAND buffer by IP or AXI bus.
- For read operation, need to open page by IP command. Then you can read data from the NAND buffer by IP or AXI bus.
- For erase operation, you should do it by IP command.

29.6.3 NOR Application

This section provides the example sequences for NOR device.

- Configure MCR[DQSMD] bit to select the read clock source for synchronous mode. Suggest to set it with 0x1 to reach high clock frequency.
- Configure IOCR register to choose chip select pins.

- Configure BMCR0/1 registers to adjust AXI bus access efficiency scheme.
- Configure Base Register 5 with base address, memory size and valid information.
- Configure INTEN and INTR registers if need to generate interrupt.
- Configure NOR Control Registers with valid settings that matches NOR device.
- Initialize NOR device by IP command registers (IPCR0/1/2, IPCMD and IPTXDAT) if needed.
- Programming operation can be triggered by IP command only.
- Read operation can be triggered by AXI or IP command.

29.6.4 SRAM Application

This section provides the example sequences for SRAM device.

- Configure MCR[DQSMD] bit to select the read clock source for synchronous mode. Suggest to set it with 0x1 to reach high clock frequency.
- Configure IOCR register to choose chip select pins.
- Configure BMCR0/1 registers to adjust AXI bus access efficiency scheme.
- Configure Base Register 6/9/10/11 with base address, memory size and valid information.
- Configure INTEN and INTR registers if need to generate interrupt.
- Configure SRAM Control Registers with valid settings that matches SRAM device.
- Initialize SRAM device by IP command registers (IPCR0/1/2, IPCMD and IPTXDAT) if needed.
- Write and read operations can be triggered by AXI or IP command.

29.6.5 DBI Application

This section provides the example sequences for Display Bus Interface.

- Configure IOCR register to choose chip select pins.
- Configure BMCR0/1 registers to adjust AXI bus access efficiency scheme.
- Configure Base Register 7 with base address, memory size and valid information.
- Configure INTEN and INTR registers if need to generate interrupt.
- Configure DBI Control Registers with valid settings that matches DBI device.
- Write and read operations can be triggered by AXI or IP command.

29.7 Memory Map and Register Definition

This section includes the SEMC module memory map and detailed descriptions of all registers.

29.7.1 SEMC register descriptions

29.7.1.1 SEMC memory map

SEMC base address: 4291_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Control Register (MCR)	32	RW	1000_0002h
4h	IO MUX Control Register (IOCR)	32	RW	0000_0000h
8h	Bus (AXI) Master Control Register 0 (BMCR0)	32	RW	0000_0000h
Ch	Bus (AXI) Master Control Register 1 (BMCR1)	32	RW	0000_0000h
10h - 30h	Base Register n (BR0 - BR8)	32	RW	0000_0000h
34h	DLL Control Register (DLLCR)	32	RW	0000_0100h
38h	Interrupt Enable Register (INTEN)	32	RW	0000_0000h
3Ch	Interrupt Register (INTR)	32	RW	0000_0000h
40h	SDRAM Control Register 0 (SDRAMCR0)	32	RW	0000_0C26h
44h	SDRAM Control Register 1 (SDRAMCR1)	32	RW	0099_4934h
48h	SDRAM Control Register 2 (SDRAMCR2)	32	RW	8000_0EEEh
4Ch	SDRAM Control Register 3 (SDRAMCR3)	32	RW	4080_8000h
50h	NAND Control Register 0 (NANDCR0)	32	RW	0000_0000h
54h	NAND Control Register 1 (NANDCR1)	32	RW	0000_0000h
58h	NAND Control Register 2 (NANDCR2)	32	RW	0001_0410h
5Ch	NAND Control Register 3 (NANDCR3)	32	RW	0000_0000h
60h	NOR Control Register 0 (NORCR0)	32	RW	0000_0000h
64h	NOR Control Register 1 (NORCR1)	32	RW	0000_0000h
68h	NOR Control Register 2 (NORCR2)	32	RW	0000_0000h
6Ch	NOR Control Register 3 (NORCR3)	32	RW	0000_0000h
70h	SRAM Control Register 0 (SRAMCR0)	32	RW	0000_0000h
74h	SRAM Control Register 1 (SRAMCR1)	32	RW	0000_0000h
78h	SRAM Control Register 2 (SRAMCR2)	32	RW	0000_0000h
7Ch	SRAM Control Register 3 (SRAMCR3)	32	RW	0000_0000h
80h	DBI-B Control Register 0 (DBICR0)	32	RW	0000_0000h
84h	DBI-B Control Register 1 (DBICR1)	32	RW	0000_0000h
88h	DBI-B Control Register 2 (DBICR2)	32	RW	0000_0000h
90h	IP Command Control Register 0 (IPCR0)	32	RW	0000_0000h
94h	IP Command Control Register 1 (IPCR1)	32	RW	0000_0000h
98h	IP Command Control Register 2 (IPCR2)	32	RW	0000_0000h
9Ch	IP Command Register (IPCMD)	32	RW	0000_0000h
A0h	TX DATA Register (IPTXDAT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

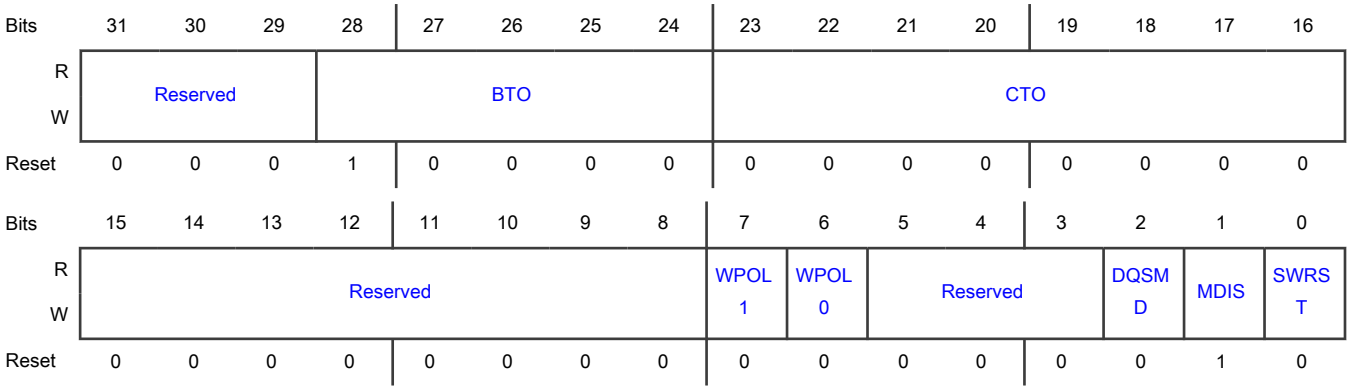
Offset	Register	Width (In bits)	Access	Reset value
B0h	RX DATA Register (IPRXDAT)	32	R	0000_0000h
C0h	Status Register 0 (STS0)	32	R	0000_0001h
C4h	Status Register 1 (STS1)	32	R	0000_0000h
C8h	Status Register 2 (STS2)	32	R	0000_0000h
CCh	Status Register 3 (STS3)	32	R	0000_0000h
D0h	Status Register 4 (STS4)	32	R	0000_0000h
D4h	Status Register 5 (STS5)	32	R	0000_0000h
D8h	Status Register 6 (STS6)	32	R	0000_0000h
DCh	Status Register 7 (STS7)	32	R	0000_0000h
E0h	Status Register 8 (STS8)	32	R	0000_0000h
E4h	Status Register 9 (STS9)	32	R	0000_0000h
E8h	Status Register 10 (STS10)	32	R	0000_0000h
ECh	Status Register 11 (STS11)	32	R	0000_0000h
F0h	Status Register 12 (STS12)	32	R	0000_0000h
F4h	Status Register 13 (STS13)	32	R	0000_0100h
F8h	Status Register 14 (STS14)	32	R	0000_0000h
FCh	Status Register 15 (STS15)	32	R	0000_0000h
100h	Base Register 9 (BR9)	32	RW	A000_0018h
104h	Base Register 10 (BR10)	32	RW	A400_0018h
108h	Base Register 11 (BR11)	32	RW	A800_0018h
120h	SRAM Control Register 4 (SRAMCR4)	32	RW	0000_0000h
124h	SRAM Control Register 5 (SRAMCR5)	32	RW	0000_0000h
128h	SRAM Control Register 6 (SRAMCR6)	32	RW	0000_0000h
140h	NAND Buffer DATA Register (NDBD)	32	RW	0000_0000h
144h	NAND Buffer Address Register (NDBA)	32	RW	0000_0000h
150h	Delay Chain Control Register (DCCR)	32	RW	0000_0000h
154h	SDRAM Prefetch Control Register (SDRAMPCR)	32	RW	0000_0000h

29.7.1.2 Module Control Register (MCR)

Offset

Register	Offset
MCR	0h

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 BTO	Bus timeout cycles The SEMC monitors AXI transactions in the queue. When it exceeds the timeout cycles, an AXIBUSERR interrupt is generated. The AXI Bus timeout is $255*2^{BTO}$ clock cycles. 0_0000b - 255*1 0_0001b - 255*2 1_1111b - 255*2^31
23-16 CTO	Command Execution timeout cycles When Command Execution time exceeds the timeout cycles, an IPCMDERR or AXICMDERR interrupt is generated. When CTO is set to zero, timeout cycles is 256*1024 cycle. Otherwise timeout cycles is CTO*1024 cycle.
15-8 —	Reserved
7 WPOL1	R/B# polarity for NAND device This bit configures the expected polarity of R/B# signal from NAND device.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - R/B# polarity is not changed. 1b - R/B# polarity is inverted.
6 WPOL0	WAIT/RDY polarity for SRAM/NOR This bit configures the expected polarity of WAIT/RDY signal from SRAM and NOR devices. 0b - WAIT/RDY polarity is not changed. 1b - WAIT/RDY polarity is inverted.
5-3 —	Reserved
2 DQSMD	DQS (read strobe) mode DQSMD controls the read clock source of SDRAM, NOR and SRAM in synchronous mode. When it set, the read clock is loopbacked from DQS pad, it benefits the read path timing, and can get high read clock frequency. Delay Chain Control Register (DCCR) register can slightly adjust the DQS timing if DQS is faster than read data in sample phase. 0b - Dummy read strobe loopbacked internally 1b - Dummy read strobe loopbacked from DQS pad
1 MDIS	Module Disable When this bit set, the SEMC internal clock is stopped, and no function is available except register access. 0b - Module enabled 1b - Module disabled
0 SWRST	Software Reset Reset all internal logic in the SEMC except configuration registers. It is an auto clear bit. 0b - No reset 1b - Reset

29.7.1.3 IO MUX Control Register (IOCR)

Offset

Register	Offset
IOCR	4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		CLKX1_AO	CLKX0_AO	MUX_CLKX1		MUX_CLKX0		MUX_RDY				MUX_CSX3			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MUX_CSX2				MUX_CSX1				MUX_CSX0				MUX_A8			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 CLKX1_AO	SEMC_CLKX1 Always On This field controls the clock gating of SEMC_CLKX1 signal. 0b - SEMC_CLKX1 is controlled by MUX_CLKX1 1b - SEMC_CLKX1 is always on
28 CLKX0_AO	SEMC_CLKX0 Always On This field controls the clock gating of SEMC_CLKX0 signal. 0b - SEMC_CLKX0 is controlled by MUX_CLKX0 1b - SEMC_CLKX0 is always on
27-26 MUX_CLKX1	SEMC_CLKX1 function selection This field controls the clock source of SEMC_CLKX1 signal. 00b - Keep low 01b - NOR clock 10b - SRAM clock 11b - NOR and SRAM clock, suitable for Multi-Chip Product package
25-24 MUX_CLKX0	SEMC_CLKX0 function selection This field controls the clock source of SEMC_CLKX0 signal. 00b - Keep low 01b - NOR clock 10b - SRAM clock 11b - NOR and SRAM clock, suitable for Multi-Chip Product package

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-20 MUX_RDY	SEMC_RDY function selection This field controls the function of SEMC_RDY signal. 0000b - NAND R/B# input 0001b - SDRAM CS1 0010b - SDRAM CS2 0011b - SDRAM CS3 0100b - NOR/SRAM Address bit 27 (A27) in Non-ADMUX mode 0101b - NOR CE# 0110b - SRAM CE# 0 0111b - DBI CSX 1000b - SRAM CE# 1 1001b - SRAM CE# 2 1010b - SRAM CE# 3 1011b-1111b - NOR/SRAM Address bit 27 in Non-ADMUX mode
19-16 MUX_CSX3	SEMC_CSX3 output selection This field controls the function of SEMC_CSX3 signal. 0000b - NOR/SRAM Address bit 27 (A27) in Non-ADMUX mode 0001b - SDRAM CS1 0010b - SDRAM CS2 0011b - SDRAM CS3 0100b - NAND CE# 0101b - NOR CE# 0110b - SRAM CE# 0 0111b - DBI CSX 1000b - SRAM CE# 1 1001b - SRAM CE# 2 1010b - SRAM CE# 3 1011b-1111b - NOR/SRAM Address bit 27 (A27) in Non-ADMUX mode
15-12 MUX_CSX2	SEMC_CSX2 output selection This field controls the function of SEMC_CSX2 signal. 0000b - NOR/SRAM Address bit 26 (A26) in Non-ADMUX mode 0001b - SDRAM CS1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0010b - SDRAM CS2 0011b - SDRAM CS3 0100b - NAND CE# 0101b - NOR CE# 0110b - SRAM CE# 0 0111b - DBI CSX 1000b - SRAM CE# 1 1001b - SRAM CE# 2 1010b - SRAM CE# 3 1011b-1111b - NOR/SRAM Address bit 26 (A26) in Non-ADMUX mode
11-8 MUX_CSX1	SEMC_CSX1 output selection This field controls the function of SEMC_CSX1 signal. 0000b - NOR/SRAM Address bit 25 (A25) in Non-ADMUX mode 0001b - SDRAM CS1 0010b - SDRAM CS2 0011b - SDRAM CS3 0100b - NAND CE# 0101b - NOR CE# 0110b - SRAM CE# 0 0111b - DBI CSX 1000b - SRAM CE# 1 1001b - SRAM CE# 2 1010b - SRAM CE# 3 1011b-1111b - NOR/SRAM Address bit 25 (A25) in Non-ADMUX mode
7-4 MUX_CSX0	SEMC_CSX0 output selection This field controls the function of SEMC_CSX0 signal. 0000b - NOR/SRAM Address bit 24 (A24) in Non-ADMUX mode 0001b - SDRAM CS1 0010b - SDRAM CS2 0011b - SDRAM CS3 0100b - NAND CE# 0101b - NOR CE#

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0110b - SRAM CE# 0 0111b - DBI CSX 1000b - SRAM CE# 1 1001b - SRAM CE# 2 1010b - SRAM CE# 3 1011b-1111b - NOR/SRAM Address bit 24 (A24) in Non-ADMUX mode
3-0 MUX_A8	SEMC_ADDR08 output selection This field controls the function of SEMC_ADDR08 signal. 0000b-0011b - SDRAM Address bit 8 (A8) or NOR/SRAM Address bit 24 (A24) in ADMUX 16bit mode 0100b - NAND CE# 0101b - NOR CE# 0110b - SRAM CE# 0 0111b - DBI CSX 1000b - SRAM CE# 1 1001b - SRAM CE# 2 1010b - SRAM CE# 3 1011b-1111b - SDRAM Address bit 8 (A8) or NOR/SRAM Address bit 24 (A24) in ADMUX 16bit mode

29.7.1.4 Bus (AXI) Master Control Register 0 (BMCR0)

Offset

Register	Offset
BMCR0	8h

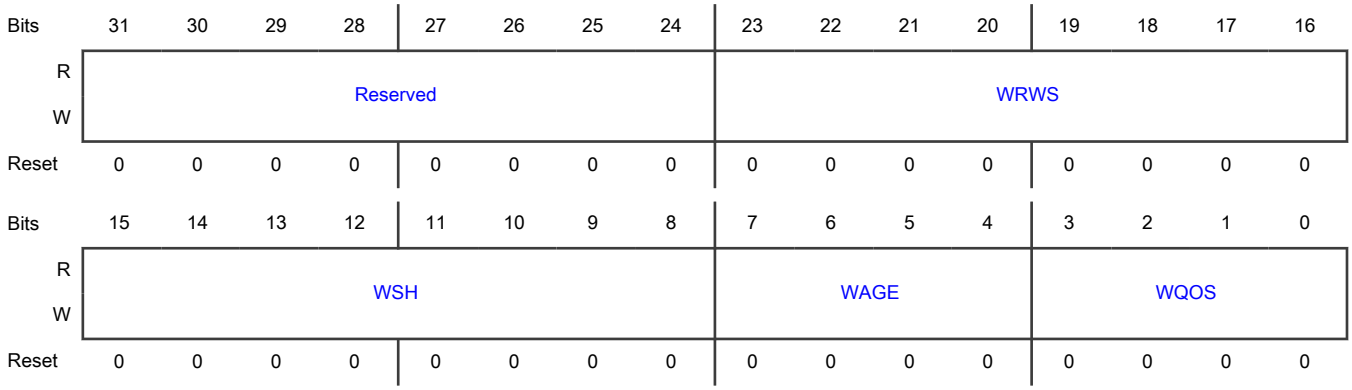
Function

Queue A weight settings for AXI bus access to SDRAM, NAND, NOR, SRAM and DBI slaves.

NOTE

Please refer to BMCR0 Register Configuration

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 WRWS	Weight of slave hit with Read/Write Switch This weight score is valid when the queue command's slave is same as current executing command with read/write operation switch.
15-8 WSH	Weight of Slave Hit without read/write switch This weight score is valid when queue command's slave is the same as current executing command without read/write operation switch.
7-4 WAGE	Weight of AGE Each command in queue has an age score to indicate its wait period. The age score is multiplied by WAGE to get the weight score. Recommend to use non-zero value.
3-0 WQOS	Weight of QOS AXI bus access has AxQOS signal set, which is used as a priority indicator for the associated write or read transaction. A higher value indicates a higher priority transaction. AxQOS is multiplied by WQOS to get the weight score.

29.7.1.5 Bus (AXI) Master Control Register 1 (BMCR1)

Offset

Register	Offset
BMCR1	Ch

Function

Queue B weight settings for AXI bus access to SDRAM slave.

NOTE

Please refer to BMCR1 Register Configuration

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WBR								WRWS							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WPH								WAGE				WQOS			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 WBR	Weight of Bank Rotation This weight score is valid when queue command's bank is not same as current executing command.
23-16 WRWS	Weight of slave hit without Read/Write Switch This weight score is valid when queue command's read/write operation is same as current executing command.
15-8 WPH	Weight of Page Hit This weight score is valid when queue command's page hits current executing command.
7-4 WAGE	Weight of AGE Each command in queue has an age signal to indicate its wait period. It is multiplied by WAGE to get the weight score. Recommend to use non-zero value.
3-0 WQOS	Weight of QOS AXI bus access has AxQOS signal set, which is used as a priority indicator for the associated write or read transaction. A higher value indicates a higher priority transaction. AxQOS is multiplied by WQOS to get the weight score.

29.7.1.6 Base Register n (BR0 - BR8)

Offset

For n = 0 to 8:

Register	Offset
BRn	10h + (n × 4h)

Function

Table 191 shows the detail information about the Base Registers.

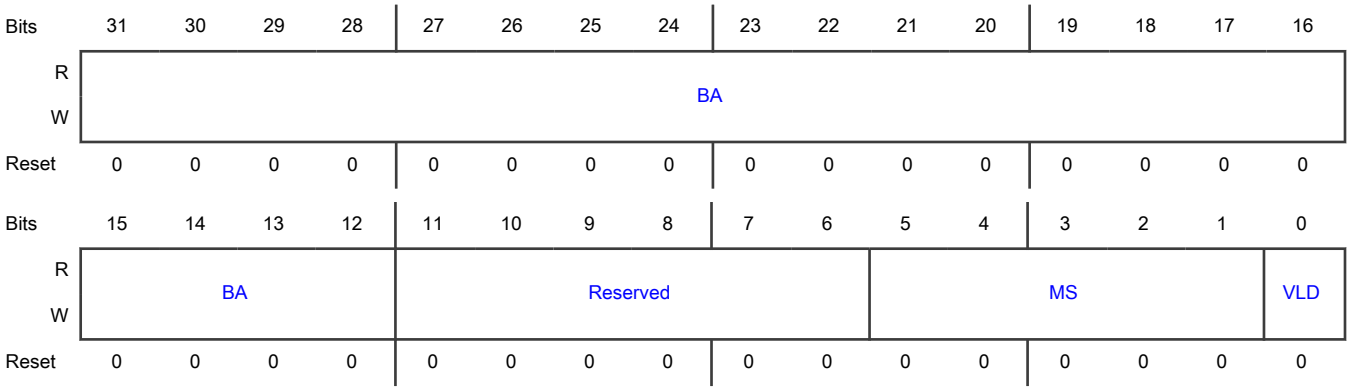
Table 191. Base Register Information

Register	Reset Value	Purpose	Comments
BR0	0x8000001D	For SDRAM CS0	For AXI and IP commands
BR1	0x8400001C	For SDRAM CS1	For AXI and IP commands
BR2	0x8800001C	For SDRAM CS2	For AXI and IP commands
BR3	0x8C00001C	For SDRAM CS3	For AXI and IP commands
BR4	0x9E00001A	For NAND	For AXI command only
BR5	0x90000018	For NOR	For AXI and IP commands
BR6	0x98000018	For SRAM CS0	For AXI and IP commands
BR7	0x9C00001A	For DBI-B	For AXI and IP commands
BR8	0x00000026	For NAND	For IP command only

NOTE

Base Register 9 (BR9), Base Register 10 (BR10) and Base Register 11 (BR11) show Base Registers for other SRAM devices.

Diagram



Fields

Field	Function
31-12 BA	Base Address This field determines high position 20 bits of SoC level base address. The low position 12 bits of the address are all zero. The base address must be within the SEMC range defined for the device memory map.
11-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-1 MS	Memory size It defines the size of the memory. Memory size must be aligned with base address. Memory regions should be configured to avoid overlapping areas. Overlapping regions will cause bus errors. 0_0000b - 4KB 0_0001b - 8KB 0_0010b - 16KB 0_0011b - 32KB 0_0100b - 64KB 0_0101b - 128KB 0_0110b - 256KB 0_0111b - 512KB 0_1000b - 1MB 0_1001b - 2MB 0_1010b - 4MB 0_1011b - 8MB 0_1100b - 16MB 0_1101b - 32MB 0_1110b - 64MB 0_1111b - 128MB 1_0000b - 256MB 1_0001b - 512MB 1_0010b - 1GB 1_0011b - 2GB 1_0100b-1_1111b - 4GB
0 VLD	Valid This bit controls whether the memory is valid or not. 0b - The memory is invalid, can not be accessed. 1b - The memory is valid, can be accessed.

29.7.1.7 DLL Control Register (DLLCR)

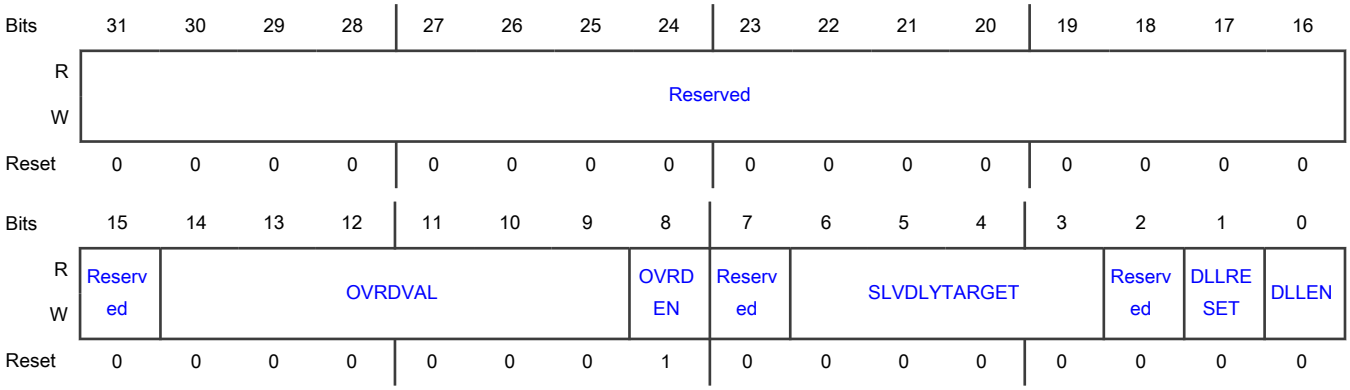
Offset

Register	Offset
DLLCR	34h

Function

This register provides the configuration fields for sample clock from DLL. It is for NAND only.

Diagram



Fields

Field	Function
31-15 —	Reserved
14-9 OVRDVAL	Override Value When OVRDEN is set, the delay cell number is OVRDVAL+1 for slave clock delay line.
8 OVRDEN	Override Enable Slave clock delay line delay cell number selection override enable. 0b - The delay cell number is not overridden. 1b - The delay cell number is overridden.
7 —	Reserved
6-3 SLVDLYTARGET	Delay Target for Slave The delay target for slave delay line is: ((SLVDLYTARGET+1) * 1/32 * clock cycle of reference clock (ipgclock).
2 —	Reserved
1 DLLRESET	DLL Reset Software can force a reset on DLL by setting this field to 0x1. This causes the DLL to lose lock and re-calibrate to detect an ref_clock half period phase shift. The reset action is edge triggered, so software need to clear this bit after setting this bit (no delay limitation). 0b - DLL is not reset.

Table continues on the next page...

Table continued from the previous page...

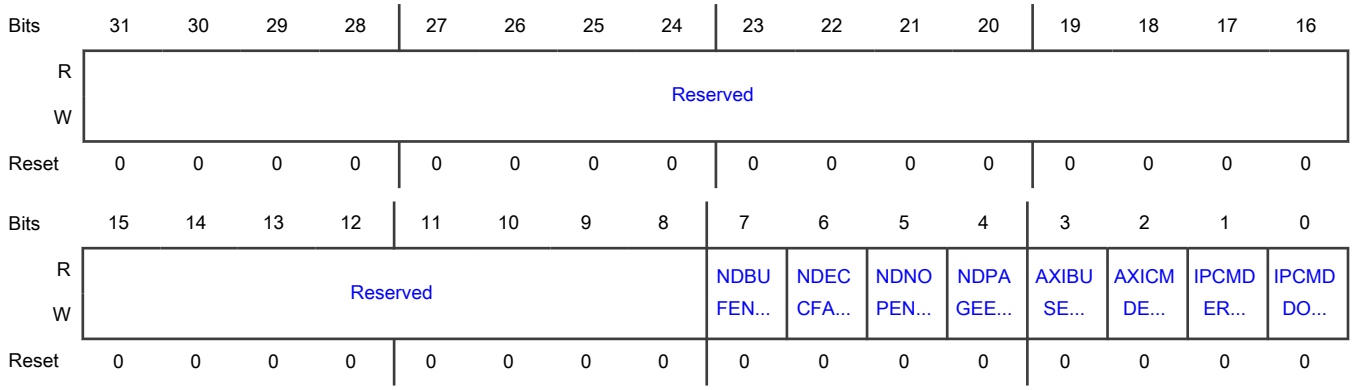
Field	Function
	1b - DLL is reset.
0 DLLEN	DLL calibration enable When this bit is cleared, DLL calibration is disabled and the delay cell number in slave delay line is always 1. Please note that SLV delay line is overridden when OVRDEN bit is set and this bit field setting is ignored. 0b - DLL calibration is disabled. 1b - DLL calibration is enabled.

29.7.1.8 Interrupt Enable Register (INTEN)

Offset

Register	Offset
INTEN	38h

Diagram



Fields

Field	Function
31-8 —	Reserved
7 NDBUFENDEN	NAND buffer end interrupt enable This bit enable/disable the NDBUFEND interrupt generation. 0b - Interrupt is disabled 1b - Interrupt is enabled

Table continues on the next page...

Table continued from the previous page...

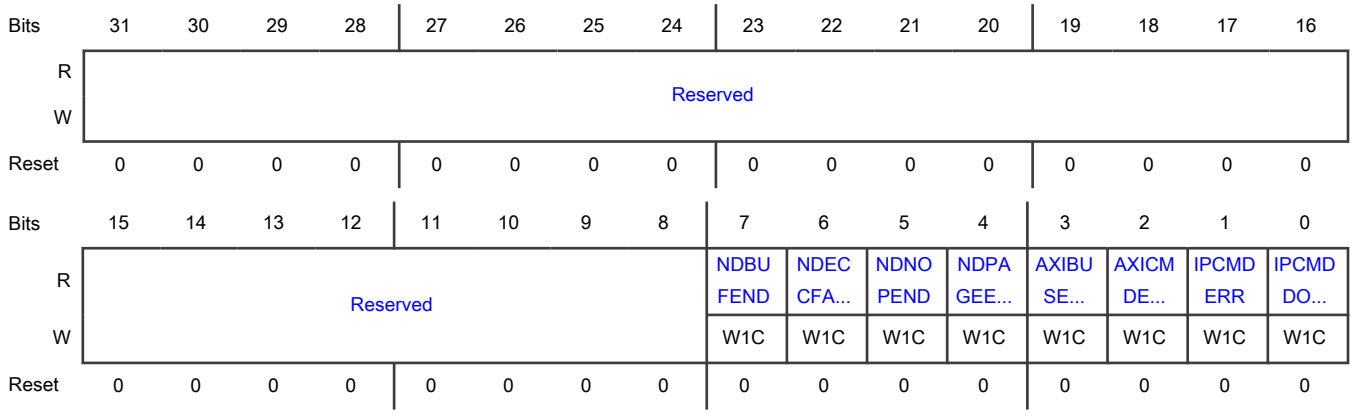
Field	Function
6 NDECCFAILEN	NAND ECC fail interrupt enable This bit enable/disable the NDECCFAIL interrupt generation. 0b - Interrupt is disabled 1b - Interrupt is enabled
5 NDNOPENDEN	NAND no pending AXI access interrupt enable This bit enable/disable the NDNOPEND interrupt generation. 0b - Interrupt is disabled 1b - Interrupt is enabled
4 NDPAGEENDEN	NAND page end interrupt enable This bit enable/disable the NDPAGEEND interrupt generation. 0b - Interrupt is disabled 1b - Interrupt is enabled
3 AXIBUSERREN	AXI bus error interrupt enable 0b - Interrupt is disabled 1b - Interrupt is enabled
2 AXICMDERREN	AXI command error interrupt enable 0b - Interrupt is disabled 1b - Interrupt is enabled
1 IPCMDERREN	IP command error interrupt enable 0b - Interrupt is disabled 1b - Interrupt is enabled
0 IPCMDDONEEN	IP command done interrupt enable 0b - Interrupt is disabled 1b - Interrupt is enabled

29.7.1.9 Interrupt Register (INTR)

Offset

Register	Offset
INTR	3Ch

Diagram



Fields

Field	Function
31-8 —	Reserved
7 NDBUFEND	NAND buffer end interrupt This interrupt is generated when the last valid address of NAND buffer is written by AXI command. The last valid address is related with NAND Control register's SECTOR_SIZE and SECTOR_NUM settings. The trigger address is SECTOR_SIZE*(sector numbers)-1. 0b - Last valid address of NAND buffer is not written by AXI command. 1b - Last valid address of NAND buffer is written by AXI command.
6 NDECCFAIL	NAND ECC fail interrupt This interrupt is generated when NAND ECC data correction failed. 0b - NAND ECC data correction pass. 1b - NAND ECC data correction fail.
5 NDNOPEND	NAND no pending AXI write transaction interrupt This interrupt is generated when all pending AXI write transactions to NAND are finished. 0b - At least one NAND AXI write transaction is pending or no NAND write transaction is sent to the queue. 1b - All NAND AXI write pending transactions are finished.
4 NDPAGEEND	NAND page end interrupt This interrupt is generated when the last address of one page in NAND device is written by AXI command. 0b - The last address of main space in the NAND is not written by AXI command. 1b - The last address of main space in the NAND is written by AXI command.
3	AXI bus error interrupt

Table continues on the next page...

Table continued from the previous page...

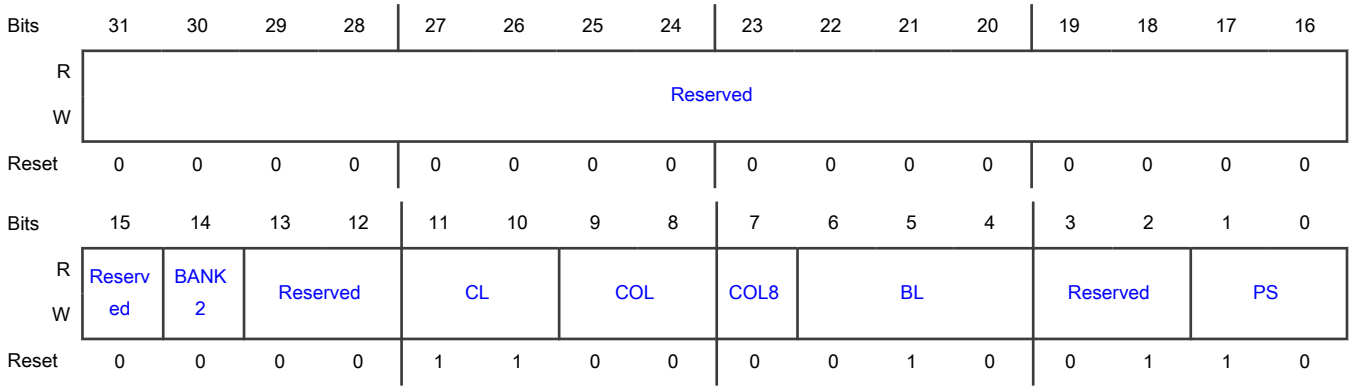
Field	Function
AXIBUSERR	AXI Bus error interrupt is generated in following cases: • AXI address is invalid • AXI write to NOR Flash • AXI 8-bit write to 16-bit NAND Flash • AXI 8-bit or 16-bit WRAP write/read 0b - No AXI bus error. 1b - AXI bus error occurs.
2 AXICMDERR	AXI command error interrupt AXI command error interrupt is generated when AXI command execution times out. 0b - No AXI command error. 1b - AXI command error occurs.
1 IPCMDERR	IP command error done interrupt IP command error interrupt is generated in following cases: • IP Command Address target invalid device space • IP Command Code unsupported • IP Command triggered when previous command not finished yet • IP Command Execution timeout 0b - No IP command error. 1b - IP command error occurs.
0 IPCMDDONE	IP command normal done interrupt It monitors whether IP command is done or not. 0b - IP command is not done. 1b - IP command is done.

29.7.1.10 SDRAM Control Register 0 (SDRAMCR0)

Offset

Register	Offset
SDRAMCR0	40h

Diagram



Fields

Field	Function
31-15 —	Reserved
14 BANK2	2 Bank selection bit This bit configures the bank numbers. 0b - SDRAM device has 4 banks. 1b - SDRAM device has 2 banks.
13-12 —	Reserved
11-10 CL	CAS Latency The CAS latency (CL) is the delay, in clock cycles, between the registration of a READ command and the availability of the output data. 00b - 1 01b - 1 10b - 2 11b - 3
9-8 COL	Column address bit number 00b - 12 01b - 11 10b - 10 11b - 9
7 COL8	Column 8 selection This bit works with COL field to determine the column address bit number.

Table continues on the next page...

Table continued from the previous page...

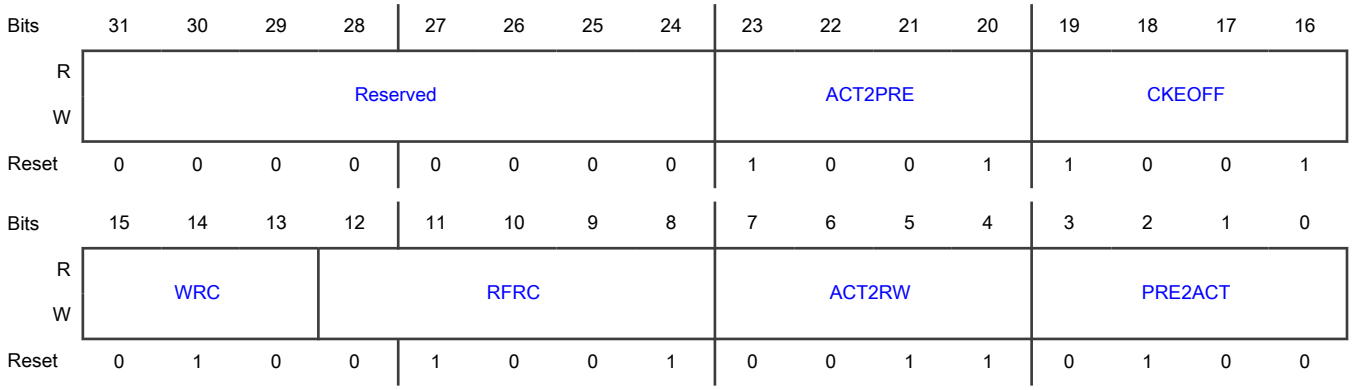
Field	Function
	0b - Column address bit number is decided by COL field. 1b - Column address bit number is 8. COL field is ignored.
6-4 BL	Burst Length The burst length determines the number of column locations that can be accessed for a given READ or WRITE command. 000b - 1 001b - 2 010b - 4 011b - 8 100b - 8 101b - 8 110b - 8 111b - 8
3-2 —	Reserved
1-0 PS	Port Size The port size must be aligned with SDRAM data width. 00b - 8bit 01b - 16bit 10b - 32bit 11b - Reserved

29.7.1.11 SDRAM Control Register 1 (SDRAMCR1)

Offset

Register	Offset
SDRAMCR1	44h

Diagram



Fields

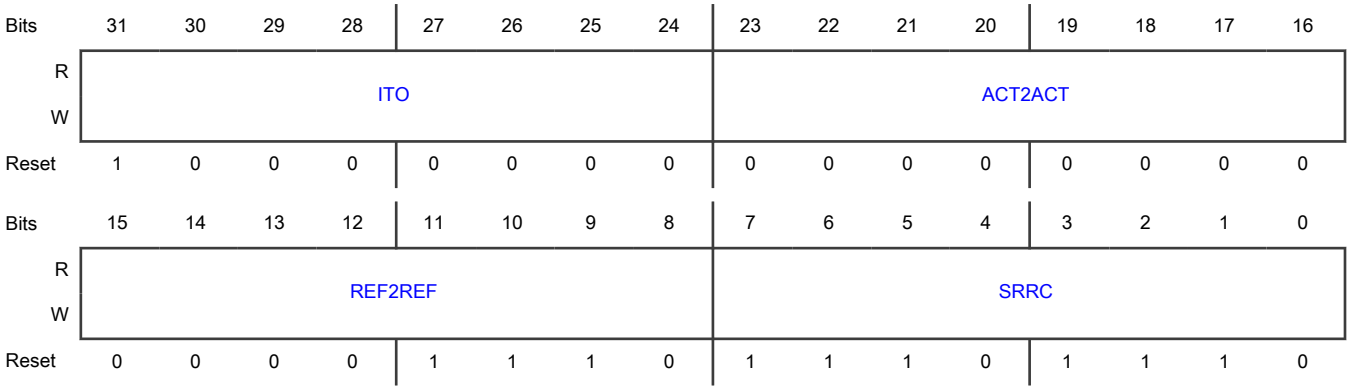
Field	Function
31-24 —	Reserved
23-20 ACT2PRE	ACTIVE to PRECHARGE minimum time The SEMC promises ACT2PRE+1 clock cycles delay between ACTIVE command to PRECHARGE/ PRECHARGE ALL command. It helps to meet tRAS timing requirement by SDRAM device.
19-16 CKEOFF	CKE off minimum time The SEMC promises at least CKEOFF+1 clock cycles in self refresh mode. It helps to meet the minimum period request to remain in self refresh mode.
15-13 WRC	WRITE recovery time The SEMC promises WRC+1 clock cycles delay between WRITE command to PRECHARGE/ PRECHARGE ALL command. It helps to meet tWR timing requirement by SDRAM device.
12-8 RFRC	REFRESH recovery time The SEMC promises RFRC+1 clock cycles delay between AUTO REFRESH command to ACTIVE command. It helps to meet tRFC timing requirement by SDRAM device.
7-4 ACT2RW	ACTIVE to READ/WRITE delay The SEMC promises ACT2RW+1 clock cycles delay between ACTIVE command to READ/WRITE command. It helps to meet tRCD timing requirement by SDRAM device.
3-0 PRE2ACT	PRECHARGE to ACTIVE/REFRESH command wait time The SEMC promises PRE2ACT+1 clock cycles delay between PRECHARGE/PRECHARGE ALL command to ACTIVE/REFRESH command. It helps to meet tRP timing requirement by SDRAM device.

29.7.1.12 SDRAM Control Register 2 (SDRAMCR2)

Offset

Register	Offset
SDRAMCR2	48h

Diagram



Fields

Field	Function
31-24 ITO	SDRAM idle timeout The SEMC closes all opened pages if the SDRAM idle time lasts more than idle timeout period. SDRAM is considered idle when there is no AXI Bus transfer and no SDRAM command pending. 0000_0000b - IDLE timeout period is 256*Prescale period. 0000_0001b-1111_1111b - IDLE timeout period is ITO*Prescale period.
23-16 ACT2ACT	ACTIVE to ACTIVE delay The SEMC promises ACT2ACT+1 clock cycles delay between ACTIVE command to ACTIVE command. It helps to meet tRRD timing requirement by SDRAM device.
15-8 REF2REF	REFRESH to REFRESH delay The SEMC promises REF2REF+1 clock cycles delay between REFRESH command to REFRESH command. It helps to meet tRFC timing requirement by SDRAM device.
7-0 SRRC	SELF REFRESH recovery time The SEMC promises SRRC+1 clock cycles delay between SELF REFRESH command to any subsequent command. SDRAM device may have limitation on self refresh recovery time. It help to meet the wait time from exiting self refresh mode to sending new command to device.

29.7.1.13 SDRAM Control Register 3 (SDRAMCR3)

Offset

Register	Offset
SDRAMCR3	4Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	UT								RT							
W																
Reset	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRESCALE								Reserved				REBL			REN
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 UT	Urgent refresh threshold Urgent refresh threshold is used to handle urgent refresh request. Urgent refresh timer starts to count when normal refresh request is asserted but not handled by the controller yet. When the urgent refresh timer count up to the threshold, urgent refresh request is sent to the controller. The priority of urgent refresh is higher than any pending AXI or IP command to SDRAM, so it can be handled by the controller at the first time. **NOTE** Refresh request is considered as urgent refresh request when urgent threshold is no less than refresh period. 0000_0000b - 256*(Prescaler period) 0000_0001b-1111_1111b - UT*(Prescaler period)
23-16 RT	Refresh timer period Refresh timer period is the threshold for SDRAM normal refresh request. The normal refresh request is sent to the controller when the refresh timer count up to it. The priority of normal refresh request is lower than any pending AXI or IP command to SDRAM. 0000_0000b - (256+1)*(Prescaler period) 0000_0001b-1111_1111b - (RT+1)*(Prescaler period)
15-8 PRESCALE	Prescaler period Prescaler period is used as a clock cycle count unit for refresh and bus idle timer period.

Table continues on the next page...

Table continued from the previous page...

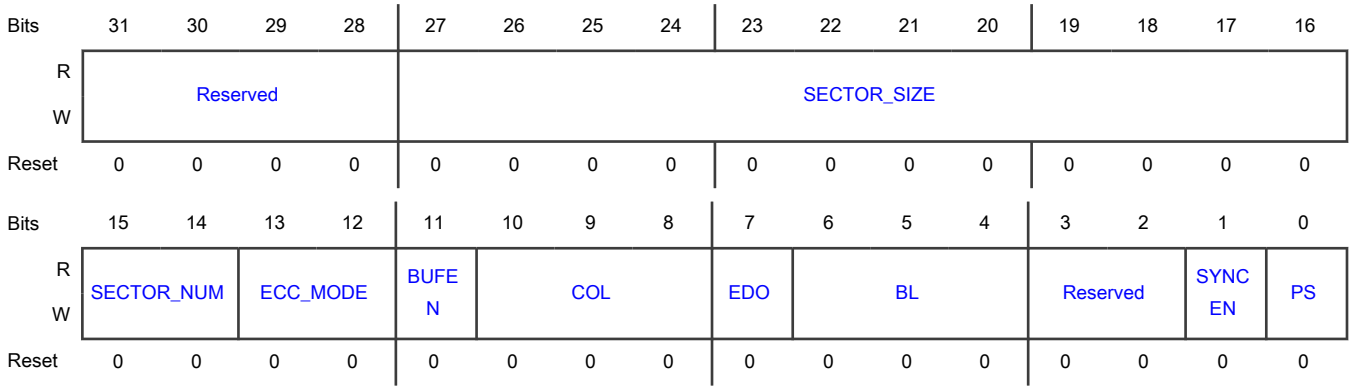
Field	Function
	0000_0000b - (256*16+1) clock cycles 0000_0001b-1111_1111b - (PRESCALE*16+1) clock cycles
7-4 —	Reserved
3-1 REBL	Refresh burst length The SEMC can send multiple AUTO REFRESH commands in one burst when REBL is set to non-zero. The number of AUTO REFRESH command in one burst is listed below. 000b - 1 001b - 2 010b - 3 011b - 4 100b - 5 101b - 6 110b - 7 111b - 8
0 REN	Refresh enable The SEMC sends AUTO REFRESH command automatically when REN bit is set. The AUTO REFRESH command interval is decided by UT, RT and PRESCALE bits. 0b - The SEMC does not send AUTO REFRESH command automatically 1b - The SEMC sends AUTO REFRESH command automatically

29.7.1.14 NAND Control Register 0 (NANDCR0)

Offset

Register	Offset
NANDCR0	50h

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 SECTOR_SIZE	Size in bytes of one elementary unit of ECC correction. Sector size includes data bits only. Sector size can not greater than (buffer size)/SECTOR_NUM, and it should be 4 bytes aligned. NAND buffer size is 2112 bytes. **NOTE** It refer to "Sector Size in Buffer" of NAND Flash Data Map
15-14 SECTOR_NUM	Sector numbers in NAND buffer 00b - There is 1 sector in buffer 01b - There are 2 sectors in buffer 10b - There are 4 sectors in buffer 11b - Reserved
13-12 ECC_MODE	ECC mode selection 00b - No correction, ECC bypass 01b - 4-error correction (8 ECC bytes) 10b - 8-error correction (16 ECC bytes) 11b - Reserved
11 BUFEN	NAND buffer enable for AXI access 0b - AXI access to NAND device directly 1b - AXI access through NAND buffer. It must be enabled for error correction schemes.
10-8 COL	Column address bit number 000b - 16

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - 15 010b - 14 011b - 13 100b - 12 101b - 11 110b - 10 111b - 9
7 EDO	EDO mode enabled It defines the read access timing in asynchronous mode. 0b - EDO mode disabled 1b - EDO mode enabled
6-4 BL	Burst Length The burst length determines the number of column locations that can be accessed for a given READ or WRITE command. 000b - 1 001b - 2 010b - 4 011b - 8 100b - 16 101b - 32 110b - 64 111b - 64
3-2 —	Reserved
1 SYNCEN	Synchronous Mode Enable It defines the access mode. 0b - Asynchronous mode is enabled. 1b - Synchronous mode is enabled.
0 PS	Port Size The port size must be aligned with NAND Flash data width. 0b - 8bit 1b - 16bit

29.7.1.15 NAND Control Register 1 (NANDCR1)

Offset

Register	Offset
NANDCR1	54h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CEITV				TA				REH				REL			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WEH				WEL				CEH				CES			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 CEITV	CE# interval time CE# interval minimum time is CEITV+1 clock cycles.
27-24 TA	Turnaround time Turnaround time is TA+1 clock cycles. Both the SEMC and Device do not drive Data Lines to avoid bus confliction. This field applied to ASYNC mode only.
23-20 REH	RE# high time RE# high time is REH+1 clock cycles. This field applied to ASYNC mode only.
19-16 REL	RE# low time RE# low time is REL+1 clock cycles. This field applied to ASYNC mode only.
15-12 WEH	WE# high time WE# high time is WEH+1 clock cycles in ASYNC mode. It is used as command and address hold time in SYNC mode. WEH must be even number in SYNC mode, and command/address hold time is WEH clock cycles.
11-8 WEL	WE# low time WE# low time is WEL+1 clock cycles in ASYNC mode. It is used as command and address setup time in SYNC mode. WEL must be even number in SYNC mode, and command/address setup time is WEL+2 clock cycles.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 CEH	CE# hold time CE# hold minimum time is CEH+1 clock cycles. CEH must be odd number in SYNC mode. CE# hold time may be larger than this time to wait all read data sampled.
3-0 CES	CE# setup time CE# setup time is CES+1 clock cycles. CES must be odd number in SYNC mode.

29.7.1.16 NAND Control Register 2 (NANDCR2)

Offset

Register	Offset
NANDCR2	58h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				TWB				TRR						TADL	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TADL				TRHW				TWHR							
Reset	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 TWB	WE# high to busy time WE# high to busy time is TWB+1 clock cycles
23-18 TRR	Ready to RE# low time Ready to RE# low minimum time is TRR+1 clock cycles. Actual time may be 1 or 2 clock cycles more than this because of synchronizer logic. This field applied to ASYNC mode only.
17-12	Address cycle to data loading time

Table continues on the next page...

Table continued from the previous page...

Field	Function
TADL	Address to data loading time is TADL+1 clock cycles. Timing for TADL begins in the address cycle on the final rising edge of WE#, and ends with the first rising edge of WE# for data input.
11-6 TRHW	RE# high to WE# low time RE# high to WE# low time is TRHW+1 clock cycles
5-0 TWHR	WE# high to RE# low time WE# high to RE# low time is TWHR+1 clock cycles in ASYNC mode. TWHR represents command, address or data input cycle to data output clock cycles in SYNC mode. The delay is TWHR+2 clock cycles and TWHR must be even.

29.7.1.17 NAND Control Register 3 (NANDCR3)

Offset

Register	Offset
NANDCR3	5Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WDH				WDS				RDH				RDS			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												CLE	NDOP T3	NDOP T2	NDOP T1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 WDH	Write Data Hold time Write data hold time is WDH+2 clock cycles, and WDH must be even number. This field applied to SYNC mode only.
27-24 WDS	Write Data Setup time Write data setup time is WDS+2 clock cycles, and WDS must be even number. This field applied to SYNC mode only.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-20 RDH	Read Data Hold time Read data hold time is RDH+2 clock cycles, and RDH must be even number. This field applied to SYNC mode only.
19-16 RDS	Read Data Setup time Read data setup time is RDS+2 clock cycles, and RDS must be even number. It provides the minimum timing control before data output phase. This field applied to SYNC mode only.
15-4 —	Reserved
3 CLE	NAND CLE Option This bit is intended to add a CLE valid clock cycle before WE# valid clock cycle for ASYNC mode.
2 NDOPT3	NAND option bit 3 This bit is intended to bypass row address byte #1 for address mode 0x0 - Column and Row address(5 Byte-CA0/CA1/RA0/RA1/RA2).
1 NDOPT2	NAND option bit 2 This bit is intended to bypass row address byte #2 for address mode 0x0 - Column and Row address(5 Byte-CA0/CA1/RA0/RA1/RA2).
0 NDOPT1	NAND option bit 1 This bit is intended to bypass column address byte #1 for address mode 0x0 - Column and Row address(5 Byte-CA0/CA1/RA0/RA1/RA2). **NOTE** Please refer to NAND Flash Access Address Type

29.7.1.18 NOR Control Register 0 (NORCR0)

Offset

Register	Offset
NORCR0	60h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COL				ADVH	ADVP	AM		Reserved	BL			Reserved		SYNC EN	PS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-12 COL	Column Address bit width 0000b - 12 Bits 0001b - 11 Bits 0010b - 10 Bits 0011b - 9 Bits 0100b - 8 Bits 0101b - 7 Bits 0110b - 6 Bits 0111b - 5 Bits 1000b - 4 Bits 1001b - 3 Bits 1010b - 2 Bits 1011b - 12 Bits 1100b - 12 Bits 1101b - 12 Bits 1110b - 12 Bits 1111b - 12 Bits
11 ADVH	ADV# level control during address hold state 0b - ADV# is high during address hold state. 1b - ADV# is low during address hold state.
10	ADV# Polarity

Table continues on the next page...

Table continued from the previous page...

Field	Function
ADVP	0b - ADV# is active low. 1b - ADV# is active high.
9-8 AM	Address Mode It defines the address mode. 00b - Address/Data MUX mode (ADMUX) 01b - Advanced Address/Data MUX mode (AADM) 10b - Address/Data non-MUX mode (Non-ADMUX) 11b - Address/Data non-MUX mode (Non-ADMUX)
7 —	Reserved
6-4 BL	Burst Length The burst length determines the number of column locations that can be accessed for a given READ or WRITE command. 000b - 1 001b - 2 010b - 4 011b - 8 100b - 16 101b - 32 110b - 64 111b - 64
3-2 —	Reserved
1 SYNCEN	Synchronous Mode Enable It defines the access mode. 0b - Asynchronous mode is enabled. 1b - Synchronous mode is enabled. Only fixed latency mode is supported.
0 PS	Port Size The port size must be aligned with NOR Flash data width. 0b - 8bit 1b - 16bit

29.7.1.19 NOR Control Register 1 (NORCR1)

Offset

Register	Offset
NORCR1	64h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	REH				REL				WEH				WEL			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	AH				AS				CEH				CES			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 REH	RE high time RE high time is REH+1 clock cycles for ASYNC mode.
27-24 REL	RE low time RE low time is REL+1 clock cycles for ASYNC mode.
23-20 WEH	WE high time WE high time is WEH+1 clock cycles for ASYNC mode.
19-16 WEL	WE low time WE low time is WEL+1 clock cycles for ASYNC mode.
15-12 AH	Address hold time Address hold time is AH+1 clock cycles for ASYNC mode.
11-8 AS	Address setup time Address setup time is AS+1 clock cycles
7-4 CEH	CE hold time CE Hold time is CEH+1 clock cycles

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field determines the minimum hold time. Actual hold time may be longer in order to wait all read data sampled.
3-0 CES	CE setup time CE setup time is CES+1 clock cycles.

29.7.1.20 NOR Control Register 2 (NORCR2)

Offset

Register	Offset
NORCR2	68h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RDH				CEITV				RD				LC			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	AWDH				TA				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 RDH	Read hold time Read hold time is RDH clock cycles. This field applied to SYNC mode only.
27-24 CEITV	CE# interval time CE# interval minimum time is CEITV+1 clock cycles.
23-20 RD	Read time Read time is RD+1 clock cycles. This field applied to SYNC mode only.
19-16 LC	Latency count Latency count is LC clock cycles. This field applied to SYNC mode only.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 AWDH	Address to write data hold time Address to write data hold time is AWDH clock cycles. This field applied to ASYNC mode only.
11-8 TA	Turnaround time Turnaround time is TA+1 clock cycles. Both the SEMC and Device do not drive Data Lines to avoid bus confliction. This field applied to ASYNC mode only.
7-0 —	Reserved

29.7.1.21 NOR Control Register 3 (NORCR3)

Offset

Register	Offset
NORCR3	6Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								AHSR				ASSR			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-4 AHSR	Address hold time for SYNC read Address hold time is AHSR clock cycles for SYNC read.
3-0 ASSR	Address setup time for SYNC read Address setup time ASSR+1 clock cycles for SYNC read.

29.7.1.22 SRAM Control Register 0 (SRAMCR0)

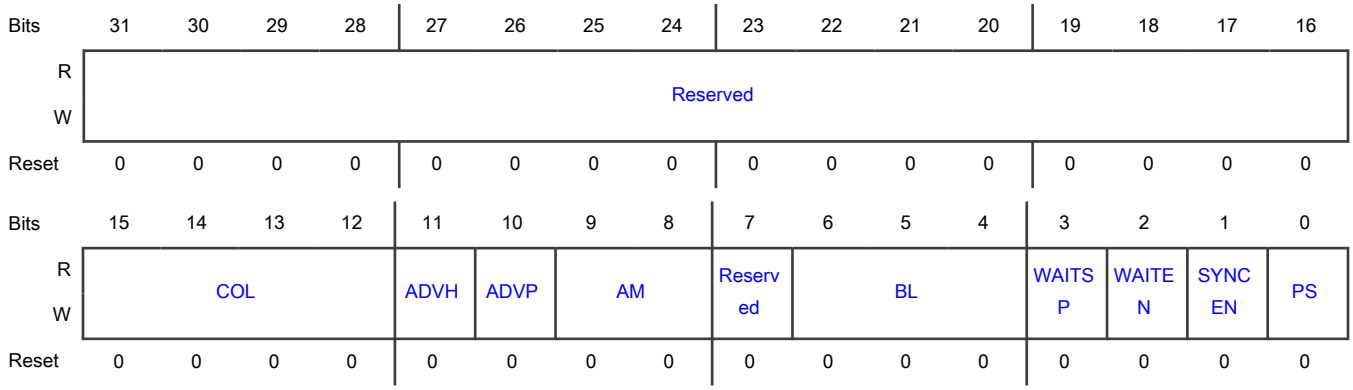
Offset

Register	Offset
SRAMCR0	70h

Function

This register configures SRAM device 0

Diagram



Fields

Field	Function
31-16 —	Reserved
15-12 COL	Column Address bit width 0000b - 12 Bits 0001b - 11 Bits 0010b - 10 Bits 0011b - 9 Bits 0100b - 8 Bits 0101b - 7 Bits 0110b - 6 Bits 0111b - 5 Bits 1000b - 4 Bits 1001b - 3 Bits 1010b - 2 Bits

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1011b - 12 Bits 1100b - 12 Bits 1101b - 12 Bits 1110b - 12 Bits 1111b - 12 Bits
11 ADVH	ADV# level control during address hold state 0b - ADV# is high during address hold state. 1b - ADV# is low during address hold state.
10 ADVP	ADV# polarity 0b - ADV# is active low. 1b - ADV# is active high.
9-8 AM	Address Mode It defines the address mode. 00b - Address/Data MUX mode (ADMUX) 01b - Advanced Address/Data MUX mode (AADM) 10b - Address/Data non-MUX mode (Non-ADMUX) 11b - Address/Data non-MUX mode (Non-ADMUX)
7 —	Reserved
6-4 BL	Burst Length The burst length determines the number of column locations that can be accessed for a given READ or WRITE command. 000b - 1 001b - 2 010b - 4 011b - 8 100b - 16 101b - 32 110b - 64 111b - 64
3 WAITSP	Wait Sample Wait pin is sampled by internal clock if this bit is set. It is ignored when WAITEN bit clear. It should be set for ASYNC mode when WAITEN set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Wait pin is directly used by the SEMC. 1b - Wait pin is sampled by internal clock before it is used.
2 WAITEN	Wait Enable The SEMC monitors wait pin if this bit set. It is for ASYNC mode only. 0b - The SEMC does not monitor wait pin. 1b - The SEMC monitors wait pin. The SEMC does not transfer/receive data when wait pin is asserted.
1 SYNCEN	Synchronous Mode Enable It defines the access mode. 0b - Asynchronous mode is enabled. 1b - Synchronous mode is enabled. Only fixed latency mode is supported.
0 PS	Port Size The port size must be aligned with SRAM data width. 0b - 8bit 1b - 16bit

29.7.1.23 SRAM Control Register 1 (SRAMCR1)

Offset

Register	Offset
SRAMCR1	74h

Function

This register configures SRAM device 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	REH				REL				WEH				WEL			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	AH				AS				CEH				CES			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 REH	RE high time RE high time is REH+1 clock cycles. This field applied to ASYNC mode only.
27-24 REL	RE low time RE low time is REL+1 clock cycles. This field applied to ASYNC mode only.
23-20 WEH	WE high time WE high time is WEH+1 clock cycles. This field applied to ASYNC mode only.
19-16 WEL	WE low time WE low time is WEL+1 clock cycles. This field applied to ASYNC mode only.
15-12 AH	Address hold time Address hold time is AH+1 clock cycles for ASYNC mode. Address hold time is AH clock cycles for SYNC mode.
11-8 AS	Address setup time Address setup time is AS+1 clock cycles
7-4 CEH	CE hold time CE Hold time is CEH+1 clock cycles This field determines the minimum hold time. Actual hold time may be larger in order to wait all read data sampled.
3-0 CES	CE setup time CE setup time is CES+1 clock cycles.

29.7.1.24 SRAM Control Register 2 (SRAMCR2)

Offset

Register	Offset
SRAMCR2	78h

Function
This register configures SRAM device 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RDH				CEITV				RD				LC			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	AWDH				TA				WDH				WDS			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 RDH	Read hold time Read hold time is RDH clock cycles. This field applied to SYNC mode only.
27-24 CEITV	CE# interval time CE# interval minimum time is CEITV+1 clock cycles.
23-20 RD	Read time Read time is RD+1 clock cycles. This field applied to SYNC mode only.
19-16 LC	Latency count Latency count is LC clock cycles. This field applied to SYNC mode only.
15-12 AWDH	Address to write data hold time Address to write data hold time is AWDH clock cycles. This field applied to ASYNC mode only.
11-8 TA	Turnaround time Turnaround time is TA+1 clock cycles. Both the SEMC and Device do not drive Data Lines to avoid bus confliction. This field applied to ASYNC mode only.
7-4 WDH	Write Data hold time Write data hold time is WDH clock cycles. This field applied to SYNC mode only.
3-0 WDS	Write Data setup time Write data setup time is WDS+1 clock cycles. This field applied to SYNC mode only.

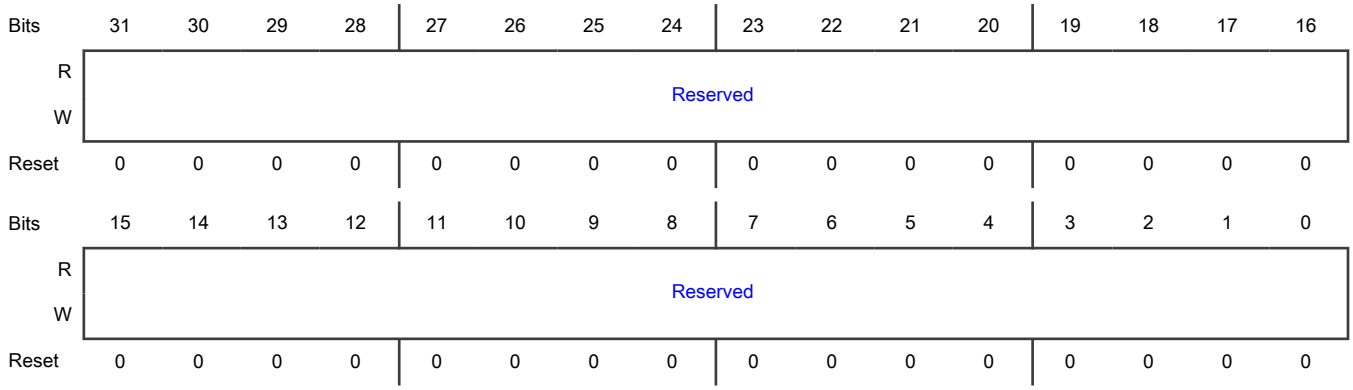
29.7.1.25 SRAM Control Register 3 (SRAMCR3)

Offset

Register	Offset
SRAMCR3	7Ch

Function
This register configures SRAM device 0

Diagram



Fields

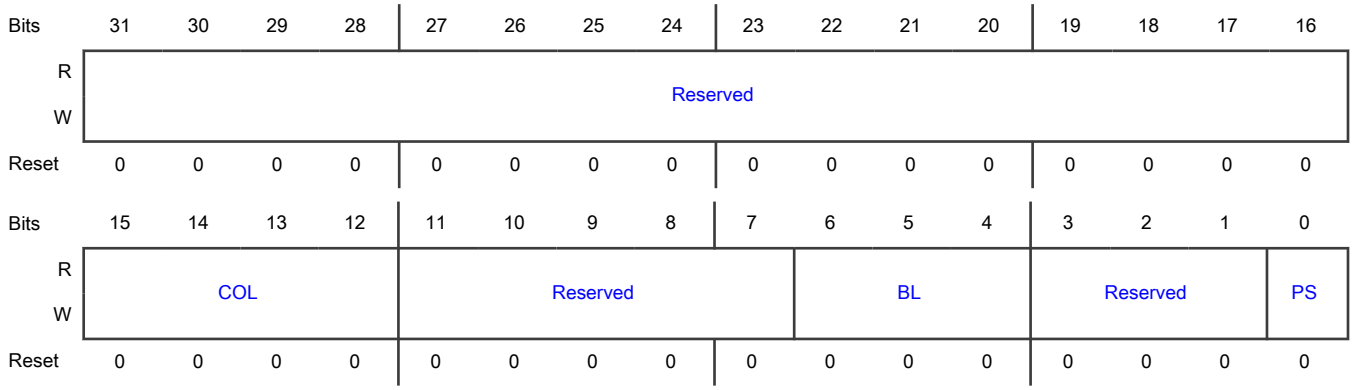
Field	Function
31-0 —	Reserved

29.7.1.26 DBI-B Control Register 0 (DBICR0)

Offset

Register	Offset
DBICR0	80h

Diagram



Fields

Field	Function
31-16 —	Reserved
15-12 COL	Column Address bit width 0000b - 12 Bits 0001b - 11 Bits 0010b - 10 Bits 0011b - 9 Bits 0100b - 8 Bits 0101b - 7 Bits 0110b - 6 Bits 0111b - 5 Bits 1000b - 4 Bits 1001b - 3 Bits 1010b - 2 Bits 1011b - 12 Bits 1100b - 12 Bits 1101b - 12 Bits 1110b - 12 Bits 1111b - 12 Bits
11-7 —	Reserved
6-4 BL	Burst Length

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The burst length determines the number of column locations that can be accessed for a given READ or WRITE command. 000b - 1 001b - 2 010b - 4 011b - 8 100b - 16 101b - 32 110b - 64 111b - 64
3-1 —	Reserved
0 PS	Port Size The port size must be aligned with DBI data width. 0b - 8bit 1b - 16bit

29.7.1.27 DBI-B Control Register 1 (DBICR1)

Offset

Register	Offset
DBICR1	84h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	REH							Reserved	REL						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WEH				WEL				CEH				CES			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-24 REH	RDX High Time RDX High time is REH+1 clock cycles
23 —	Reserved
22-16 REL	RDX Low Time RDX Low time is REL+1 clock cycles
15-12 WEH	WRX High Time WRX High time is WEH+1 clock cycles
11-8 WEL	WRX Low Time WRX Low time is WEL+1 clock cycles
7-4 CEH	CSX Hold Time CSX Hold minimum time is CEH+1 clock cycles. This field determines the minimum hold time. Actual hold time may be larger in order to wait all read data sampled.
3-0 CES	CSX Setup Time CSX Setup time is CES+1 clock cycles

29.7.1.28 DBI-B Control Register 2 (DBICR2)

Offset

Register	Offset
DBICR2	88h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												CEITV			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 CEITV	CSX interval time CSX interval minimum time is CEITV+1 clock cycles.

29.7.1.29 IP Command Control Register 0 (IPCR0)

Offset

Register	Offset
IPCR0	90h

Function

Slave address

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

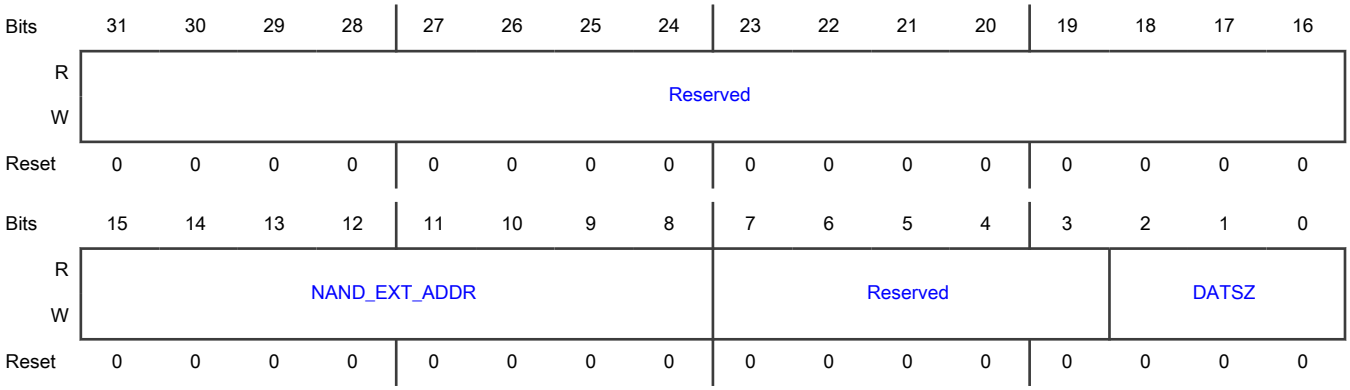
Field	Function
31-0 SA	Slave address The slave address is used for any command from IP bus.

29.7.1.30 IP Command Control Register 1 (IPCR1)

Offset

Register	Offset
IPCR1	94h

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 NAND_EXT_ADDR	NAND Extended Address NAND IP command address is extended to 40-bit when IPCMD's KEY is 0x5AA5. High 8-bit is from NAND_EXT_ADDR, and low 32-bit is from IPCR0.
7-3 —	Reserved
2-0 DATSZ	Data Size in Byte It controls the data size of any write/read command. When IP command is not a write/read operation, DATSZ field can be ignored.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - 4 001b - 1 010b - 2 011b - 3 100b - 4 101b - 4 110b - 4 111b - 4

29.7.1.31 IP Command Control Register 2 (IPCR2)

Offset

Register	Offset
IPCR2	98h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												BM3	BM2	BM1	BM0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3 BM3	Byte Mask for Byte 3 (IPTXDAT bit 31:24) 0b - Byte is unmasked 1b - Byte is masked

Table continues on the next page...

Table continued from the previous page...

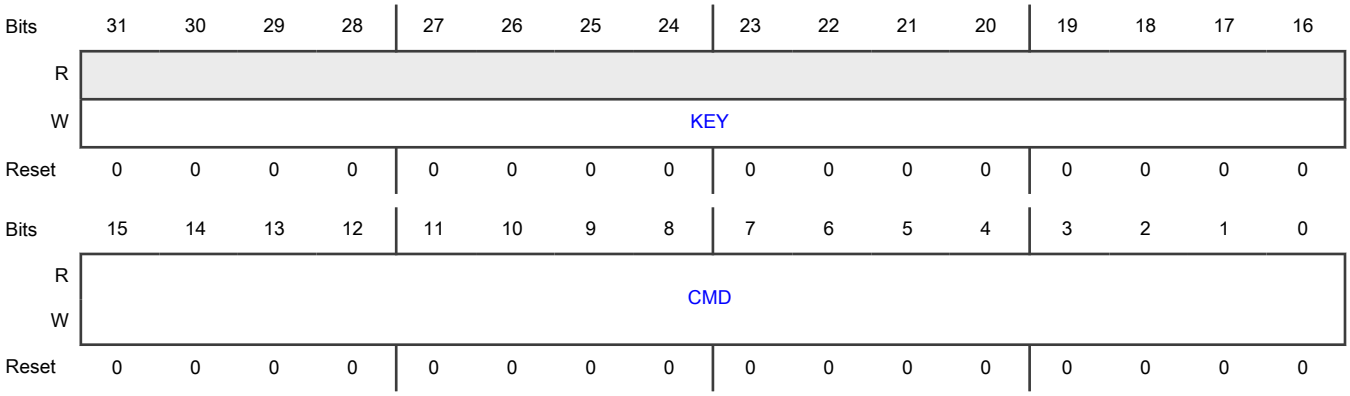
Field	Function
2 BM2	Byte Mask for Byte 2 (IPTXDAT bit 23:16) 0b - Byte is unmasked 1b - Byte is masked
1 BM1	Byte Mask for Byte 1 (IPTXDAT bit 15:8) 0b - Byte is unmasked 1b - Byte is masked
0 BM0	Byte Mask for Byte 0 (IPTXDAT bit 7:0) 0b - Byte is unmasked 1b - Byte is masked

29.7.1.32 IP Command Register (IPCMD)

Offset

Register	Offset
IPCMD	9Ch

Diagram



Fields

Field	Function
31-16 KEY	This field should be written with 0xA55A when trigging an IP command for all device types. The memory device is selected by the settings of IPCR0 and Base Registers. This field should be written with 0x5AA5 when trigging an IP command for NAND only. The address is extended to 40-bit with the use of NAND_EXT_ADDR. NAND IP base address is from 0x0 and memory size is not limited by BR8[MS].

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 CMD	SDRAM Commands: • 0x5: Extended Mode Register Set • 0x6: Deep Power Down • 0x8: Read • 0x9: Write • 0xA: Mode Register Set • 0xB: Active • 0xC: Auto Refresh • 0xD: Self Refresh • 0xE: Precharge • 0xF: Precharge All • Others: Reserved **NOTE** Self Refresh is sent to all SDRAM devices because they share the same SEMC_CLK pin. NAND Commands: Bit 15-8 (Command Code) Bit 7-4 (Address mode): • 0x0: Column and Row address(5 Byte-CA0/CA1/RA0/RA1/RA2) • 0x1: Column address only (1 Byte-CA0) • 0x2: Column address only (2 Byte-CA0/CA1) • 0x3: Row address only (1 Byte-RA0) • 0x4: Row address only (2 Byte-RA0/RA1) • 0x5: Row address only (3 Byte-RA0/RA1/RA2) • Others: Reserved Bit 3-0 (Command mode): • 0x0: Command(0x05)-Address-Command-Read (Reserved for AXI Read) • 0x1: Command(0x85)-Address-Write (Reserved for AXI Write) • 0x2: Command • 0x3: Command-Hold • 0x4: Command-Address • 0x5: Command-Address-Hold • 0x6: Command-Address-Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0x7: Command-Address-Write • 0x8: Command-Read • 0x9: Command-Write • 0xA: Read • 0xB: Write • 0xC: Buffer Read • 0xD: Buffer Write • Others: Reserved NOR Commands: • 0x2: Read • 0x3: Write • Others: Reserved SRAM Commands: • 0x2: Memory Array Read • 0x3: Memory Array Write • 0x4: Memory Register Read • 0x5: Memory Register Write • Others: Reserved DBI_B Commands: • 0x2: Read • 0x3: Write • Others: Reserved

29.7.1.33 TX DATA Register (IPTXDAT)

Offset

Register	Offset
IPTXDAT	A0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DAT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DAT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DAT	Data value to use for an IP write command. Bytes to use are determined based on IPCR2 configuration.

29.7.1.34 RX DATA Register (IPRXDAT)

Offset

Register	Offset
IPRXDAT	B0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DAT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DAT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

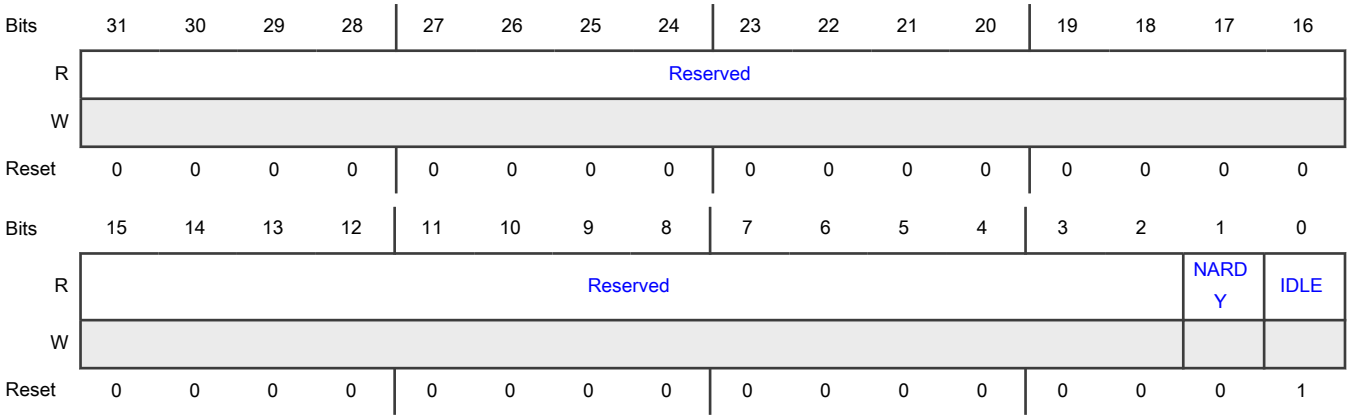
Field	Function
31-0 DAT	Data returned by device for an IP read command.

29.7.1.35 Status Register 0 (STS0)

Offset

Register	Offset
STS0	C0h

Diagram



Fields

Field	Function
31-2 —	Reserved
1 NARDY	Indicating NAND device Ready/WAIT# pin level. 0b - NAND device is not ready 1b - NAND device is ready
0 IDLE	Indicating whether the SEMC is in idle state. When IDLE=1, the SEMC is in idle state. There is no pending AXI command in internal queue and no pending device access.

29.7.1.36 Status Register 1 (STS1)

Offset

Register	Offset
STS1	C4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

29.7.1.37 Status Register 2 (STS2)

Offset

Register	Offset
STS2	C8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												NDWR PEND	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

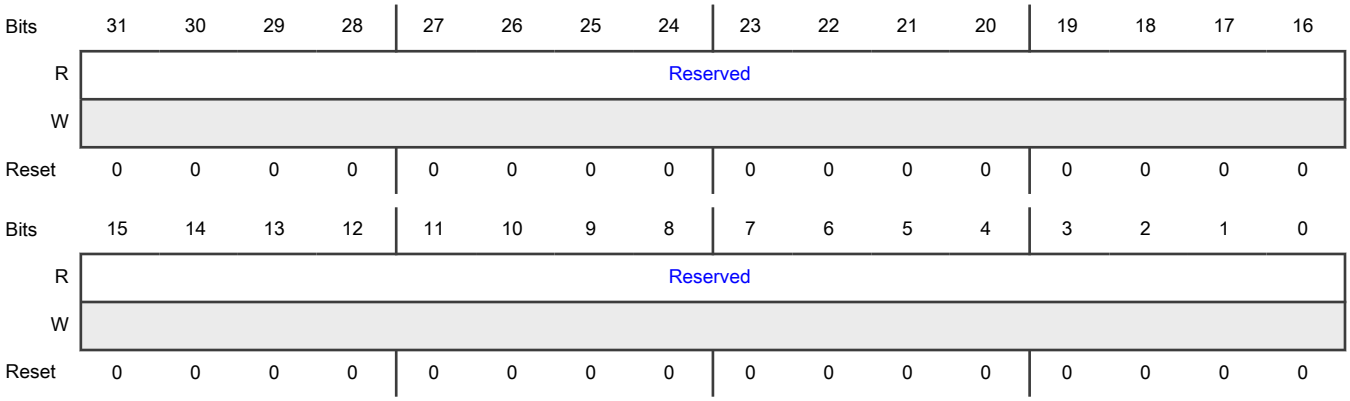
Field	Function
31-4 —	Reserved
3 NDWRPEND	This field indicating whether there is pending AXI command (write) to NAND device. 0b - No pending 1b - Pending
2-0 —	Reserved

29.7.1.38 Status Register 3 (STS3)

Offset

Register	Offset
STS3	CCh

Diagram



Fields

Field	Function
31-0 —	Reserved

29.7.1.39 Status Register 4 (STS4)

Offset

Register	Offset
STS4	D0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

29.7.1.40 Status Register 5 (STS5)

Offset

Register	Offset
STS5	D4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

29.7.1.41 Status Register 6 (STS6)

Offset

Register	Offset
STS6	D8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

29.7.1.42 Status Register 7 (STS7)

Offset

Register	Offset
STS7	DCh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

29.7.1.43 Status Register 8 (STS8)

Offset

Register	Offset
STS8	E0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

29.7.1.44 Status Register 9 (STS9)

Offset

Register	Offset
STS9	E4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

29.7.1.45 Status Register 10 (STS10)

Offset

Register	Offset
STS10	E8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

29.7.1.46 Status Register 11 (STS11)

Offset

Register	Offset
STS11	ECh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

29.7.1.47 Status Register 12 (STS12)

Offset

Register	Offset
STS12	F0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NDADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NDADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 NDADDR	This field indicating the last write address (AXI command) to NAND device (without base address in SEMC_BR4).

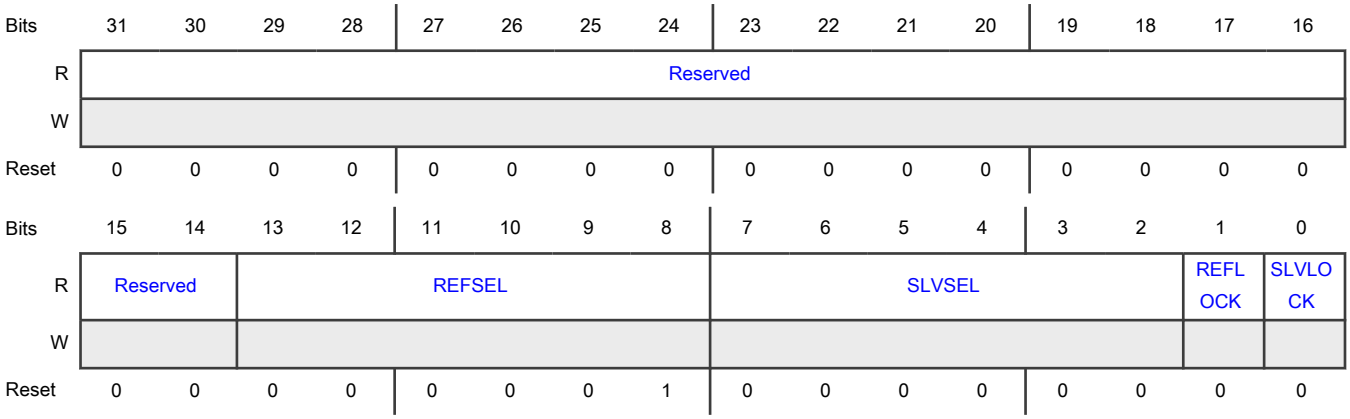
29.7.1.48 Status Register 13 (STS13)

Offset

Register	Offset
STS13	F4h

Function
This register indicates the status of sample clock DLLs for synchronous NAND read access.

Diagram



Fields

Field	Function
31-14 —	Reserved
13-8 REFSEL	Sample clock reference delay line delay cell number selection.
7-2 SLVSEL	Sample clock slave delay line delay cell number selection.
1 REFLOCK	Sample clock reference delay line locked. 0b - Reference delay line is not locked. 1b - Reference delay line is locked.
0 SLVLOCK	Sample clock slave delay line locked. 0b - Slave delay line is not locked. 1b - Slave delay line is locked.

29.7.1.49 Status Register 14 (STS14)

Offset

Register	Offset
STS14	F8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

29.7.1.50 Status Register 15 (STS15)

Offset

Register	Offset
STS15	FCh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 —	Reserved

29.7.1.51 Base Register 9 (BR9)

Offset

Register	Offset
BR9	100h

Function

This is the address base register for SRAM device 1. It is applied to both AXI and IP commands.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BA															
W																
Reset	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BA				Reserved				MS				VLD			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

Fields

Field	Function
31-12 BA	Base Address This field determines high position 20 bits of SoC level Base Address. The low position 12 bits of the address are all zero. The base address must be within the SEMC range defined for the device memory map.
11-6 —	Reserved
5-1 MS	Memory size It defines the size of the memory. Memory size must be aligned with base address. Memory regions should be configured to avoid overlapping areas. Overlapping regions will cause bus errors. 0_0000b - 4KB 0_0001b - 8KB 0_0010b - 16KB 0_0011b - 32KB 0_0100b - 64KB 0_0101b - 128KB 0_0110b - 256KB 0_0111b - 512KB 0_1000b - 1MB 0_1001b - 2MB 0_1010b - 4MB 0_1011b - 8MB 0_1100b - 16MB 0_1101b - 32MB 0_1110b - 64MB 0_1111b - 128MB 1_0000b - 256MB 1_0001b - 512MB 1_0010b - 1GB 1_0011b - 2GB 1_0100b-1_1111b - 4GB
0 VLD	Valid This bit controls whether the memory is valid or not. 0b - The memory is invalid, can not be accessed. 1b - The memory is valid, can be accessed.

29.7.1.52 Base Register 10 (BR10)

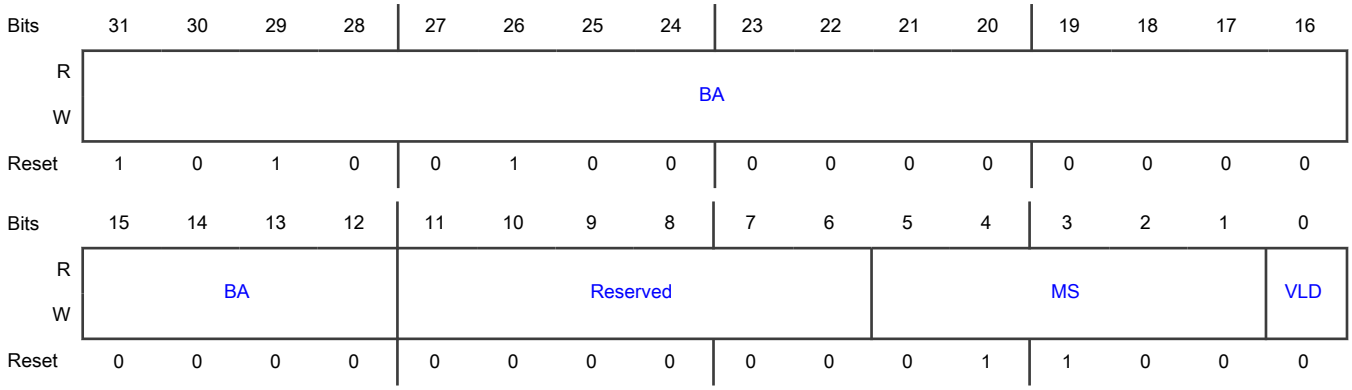
Offset

Register	Offset
BR10	104h

Function

This is the address base register for SRAM device 2. It is applied to both AXI and IP commands.

Diagram



Fields

Field	Function
31-12 BA	Base Address This field determines high position 20 bits of SoC level Base Address. The low position 12 bits of the address are all zero. The base address must be within the SEMC range defined for the device memory map.
11-6 —	Reserved
5-1 MS	Memory size It defines the size of the memory. Memory size must be aligned with base address. Memory regions should be configured to avoid overlapping areas. Overlapping regions will cause bus errors. 0_0000b - 4KB 0_0001b - 8KB 0_0010b - 16KB 0_0011b - 32KB 0_0100b - 64KB 0_0101b - 128KB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0_0110b - 256KB 0_0111b - 512KB 0_1000b - 1MB 0_1001b - 2MB 0_1010b - 4MB 0_1011b - 8MB 0_1100b - 16MB 0_1101b - 32MB 0_1110b - 64MB 0_1111b - 128MB 1_0000b - 256MB 1_0001b - 512MB 1_0010b - 1GB 1_0011b - 2GB 1_0100b-1_1111b - 4GB
0 VLD	Valid This bit controls whether the memory is valid or not. 0b - The memory is invalid, can not be accessed. 1b - The memory is valid, can be accessed.

29.7.1.53 Base Register 11 (BR11)

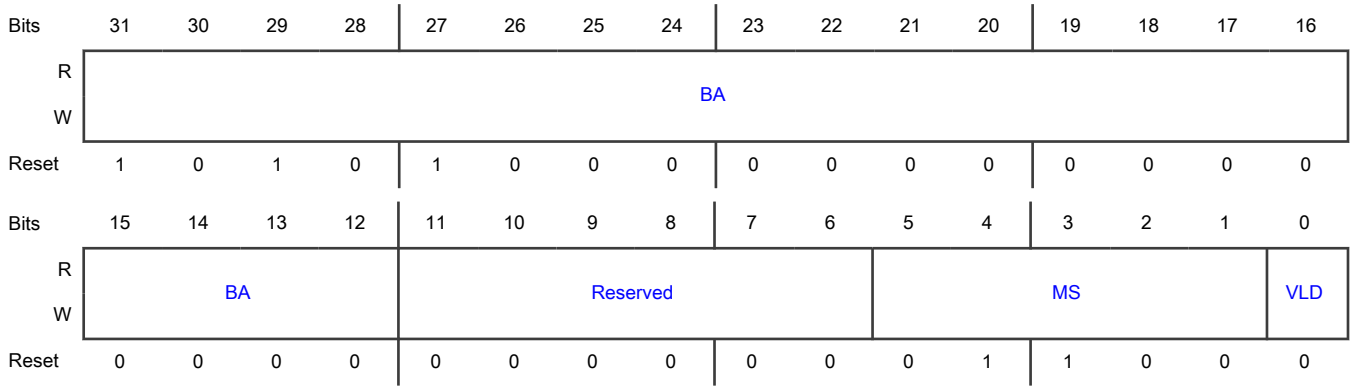
Offset

Register	Offset
BR11	108h

Function

This is the address base register for SRAM device 3. It is applied to both AXI and IP commands.

Diagram



Fields

Field	Function
31-12 BA	Base Address This field determines high position 20 bits of SoC level Base Address. The low position 12 bits of the address are all zero. The base address must be within the SEMC range defined for the device memory map.
11-6 —	Reserved
5-1 MS	Memory size It defines the size of the memory. Memory size must be aligned with base address. Memory regions should be configured to avoid overlapping areas. Overlapping regions will cause bus errors. 0_0000b - 4KB 0_0001b - 8KB 0_0010b - 16KB 0_0011b - 32KB 0_0100b - 64KB 0_0101b - 128KB 0_0110b - 256KB 0_0111b - 512KB 0_1000b - 1MB 0_1001b - 2MB 0_1010b - 4MB 0_1011b - 8MB 0_1100b - 16MB 0_1101b - 32MB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0_1110b - 64MB 0_1111b - 128MB 1_0000b - 256MB 1_0001b - 512MB 1_0010b - 1GB 1_0011b - 2GB 1_0100b-1_1111b - 4GB
0 VLD	Valid This bit controls whether the memory is valid or not. 0b - The memory is invalid, can not be accessed. 1b - The memory is valid, can be accessed.

29.7.1.54 SRAM Control Register 4 (SRAMCR4)

Offset

Register	Offset
SRAMCR4	120h

Function

This register configures SRAM device 1 - 3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COL				ADVH	ADVP	AM		Reserved	BL			WAITSP	WAITEN	SYNCEN	PS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-12 COL	Column Address bit width 0000b - 12 Bits 0001b - 11 Bits 0010b - 10 Bits 0011b - 9 Bits 0100b - 8 Bits 0101b - 7 Bits 0110b - 6 Bits 0111b - 5 Bits 1000b - 4 Bits 1001b - 3 Bits 1010b - 2 Bits 1011b - 12 Bits 1100b - 12 Bits 1101b - 12 Bits 1110b - 12 Bits 1111b - 12 Bits
11 ADVH	ADV# level control during address hold state 0b - ADV# is high during address hold state. 1b - ADV# is low during address hold state.
10 ADVP	ADV# polarity 0b - ADV# is active low. 1b - ADV# is active high.
9-8 AM	Address Mode It defines the address mode. 00b - Address/Data MUX mode (ADMUX) 01b - Advanced Address/Data MUX mode (AADM) 10b - Address/Data non-MUX mode (Non-ADMUX) 11b - Address/Data non-MUX mode (Non-ADMUX)
7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6-4 BL	Burst Length The burst length determines the number of column locations that can be accessed for a given READ or WRITE command. 000b - 1 001b - 2 010b - 4 011b - 8 100b - 16 101b - 32 110b - 64 111b - 64
3 WAITSP	Wait Sample Wait pin is sampled by internal clock if this bit is set. It is ignored when WAITEN bit clear. It should be set for ASYNC mode. 0b - Wait pin is directly used by the SEMC. 1b - Wait pin is sampled by internal clock before it is used.
2 WAITEN	Wait Enable The SEMC monitors wait pin if this bit set. It is for ASYNC mode only. 0b - The SEMC does not monitor wait pin. 1b - The SEMC monitors wait pin. The SEMC does not transfer/receive data when wait pin is asserted.
1 SYNCEN	Synchronous Mode Enable It defines the access mode. 0b - Asynchronous mode is enabled. 1b - Synchronous mode is enabled. Only fixed latency mode is supported.
0 PS	Port Size The port size must be aligned with SRAM data width. 0b - 8bit 1b - 16bit

29.7.1.55 SRAM Control Register 5 (SRAMCR5)

Offset

Register	Offset
SRAMCR5	124h

Function

This register configures SRAM device 1 - 3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	REH				REL				WEH				WEL			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	AH				AS				CEH				CES			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 REH	RE high time RE high time is REH+1 clock cycles. This field applied to ASYNC mode only.
27-24 REL	RE low time RE low time is REL+1 clock cycles. This field applied to ASYNC mode only.
23-20 WEH	WE high time WE high time is WEH+1 clock cycles. This field applied to ASYNC mode only.
19-16 WEL	WE low time WE low time is WEL+1 clock cycles. This field applied to ASYNC mode only.
15-12 AH	Address hold time Address hold time is AH+1 clock cycles for ASYNC mode. Address hold time is AH clock cycles for SYNC mode.
11-8 AS	Address setup time Address setup time is AS+1 clock cycles

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 CEH	CE hold time CE Hold time is CEH+1 clock cycles This field determines the minimum hold time. Actual hold time may be larger in order to wait all read data sampled.
3-0 CES	CE setup time CE setup time is CES+1 clock cycles.

29.7.1.56 SRAM Control Register 6 (SRAMCR6)

Offset

Register	Offset
SRAMCR6	128h

Function

This register configures SRAM device 1 - 3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RDH				CEITV				RD				LC			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	AWDH				TA				WDH				WDS			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 RDH	Read hold time Read hold time is RDH clock cycles. This field applied to SYNC mode only.
27-24 CEITV	CE# interval time CE# interval minimum time is CEITV+1 cycles.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-20 RD	Read time Read time is RD+1 clock cycles. This field applied to SYNC mode only.
19-16 LC	Latency count Latency count is LC clock cycles. This field applied to SYNC mode only.
15-12 AWDH	Address to write data hold time Address to write data hold time is AWDH clock cycles. This field applied to ASYNC mode only.
11-8 TA	Turnaround time Turnaround time is TA+1 clock cycles. Both the SEMC and Device do not drive Data Lines to avoid bus confliction. This field applied to ASYNC mode only.
7-4 WDH	Write Data hold time Write data hold time is WDH clock cycles. This field applied to SYNC mode only.
3-0 WDS	Write Data setup time Write data setup time is WDS+1 clock cycles. This field applied to SYNC mode only.

29.7.1.57 NAND Buffer DATA Register (NDBD)

Offset

Register	Offset
NDBD	140h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DAT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DAT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

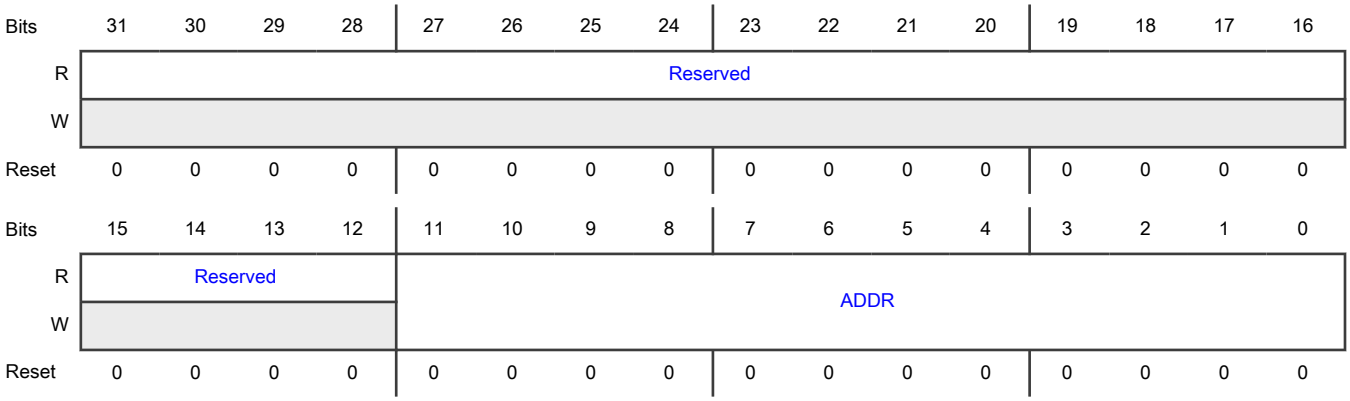
Field	Function
31-0 DAT	NAND Buffer data. It is used for program or read operation from IPS bus.

29.7.1.58 NAND Buffer Address Register (NDBA)

Offset

Register	Offset
NDBA	144h

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 ADDR	NAND Buffer address. It is used for program or read operation from IPS bus. It should be configured to proper value before access to NDBD register.

29.7.1.59 Delay Chain Control Register (DCCR)

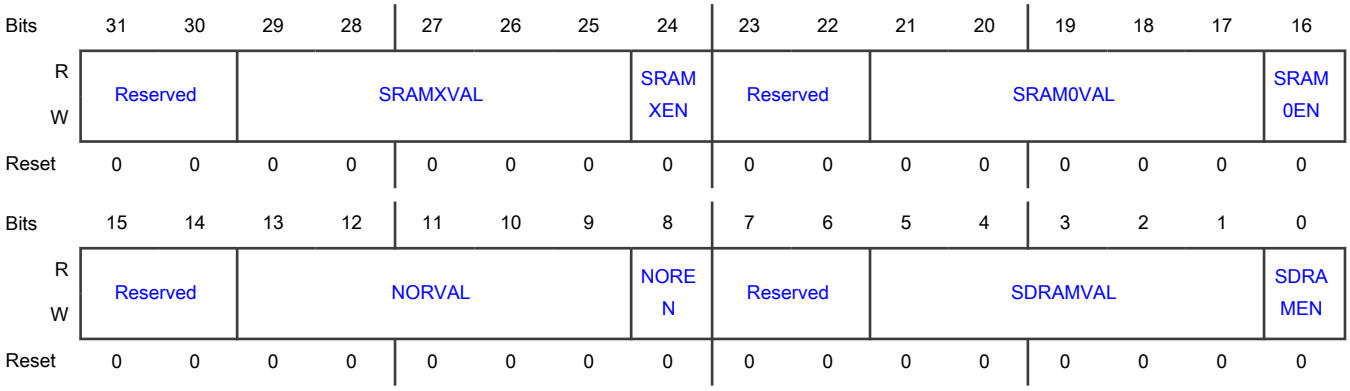
Offset

Register	Offset
DCCR	150h

Function

This register provides the configuration fields for sample clock DQS in synchronous mode. It adds delays on DQS clock to compensate timings while DQS is faster than read data. It is for synchronous SDRAM, SRAM and NOR read access.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-25 SRAMXVAL	Clock delay line delay cell number selection value for SRAM device 1-3. When SRAMXEN is set, the delay cell number in delay chain is SRAMXVAL+1.
24 SRAMXEN	Delay chain insertion enable for SRAM device 1-3. 0b - Delay chain is not inserted. 1b - Delay chain is inserted.
23-22 —	Reserved
21-17 SRAM0VAL	Clock delay line delay cell number selection value for SRAM device 0. When SRAM0EN is set, the delay cell number in delay chain is SRAM0VAL+1.
16 SRAM0EN	Delay chain insertion enable for SRAM device 0. 0b - Delay chain is not inserted. 1b - Delay chain is inserted.
15-14 —	Reserved
13-9 NORVAL	Clock delay line delay cell number selection value for NOR device. When NOREN is set, the delay cell number in delay chain is NORVAL+1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 NOREN	Delay chain insertion enable for NOR device. 0b - Delay chain is not inserted. 1b - Delay chain is inserted.
7-6 —	Reserved
5-1 SDRAMVAL	Clock delay line delay cell number selection value for SDRAM device. When SDRAMEN is set, the delay cell number in delay chain is SDRAMVAL+1.
0 SDRAMEN	Delay chain insertion enable for SRAM device. 0b - Delay chain is not inserted. 1b - Delay chain is inserted.

29.7.1.60 SDRAM Prefetch Control Register (SDRAMPCR)

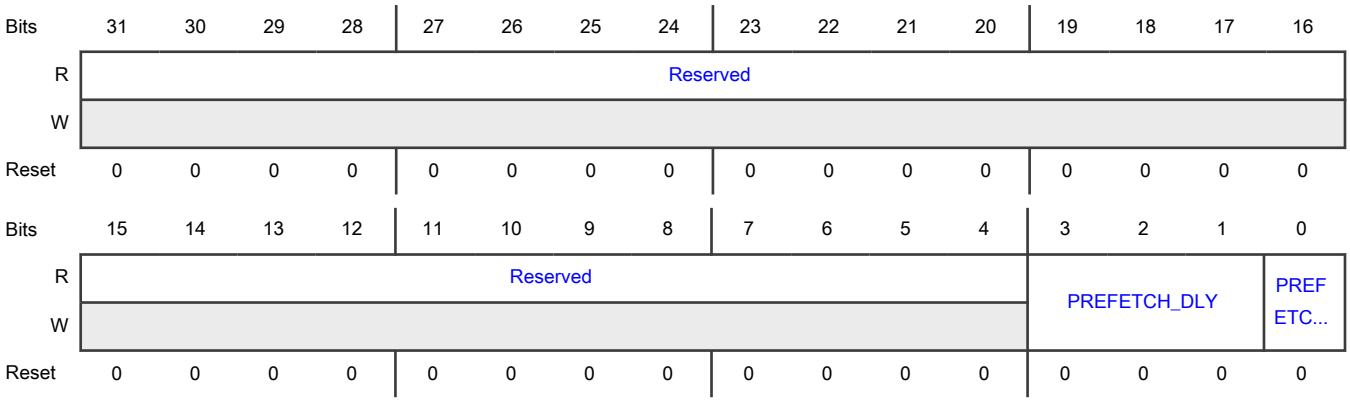
Offset

Register	Offset
SDRAMPCR	154h

Function

This register provides the control fields for SDRAM prefetch function.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-1 PREFETCH_DLY	SDRAM prefetch delay cycle. This field defines the delay cycles from SDRAM read action to the triggered prefetch action.
0 PREFETCH_EN	SDRAM prefetch enable. When set, SDRAM prefetch function is enabled. 0b - SDRAM prefetch function is disabled. 1b - SDRAM prefetch function is enabled.

Chapter 30
AHB SRAM Controller (SRAMC)

30.1 Chip-specific SRAMC Information

Table 192. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

30.1.1 Address space for SRAMC CS0-CS3

The following memory address spaces are used for SRAMC CS0-CS3:

- Region from 4388_0000 (NS) to 4389_FFFF (NS) for SRAMC CS0
- Region from 438A_0000 (NS) to 438B_FFFF (NS) for SRAMC CS1 device
- Region from 438C_0000 (NS) to 438D_FFFF (NS) for SRAMC CS2 device
- Region from 438E_0000 (NS) to 438F_FFFF (NS) for SRAMC CS3 device

30.2 Overview

The AHB SRAM Controller is a memory controller for asynchronous SRAM or SRAM like FPGA.

30.2.1 Block diagram

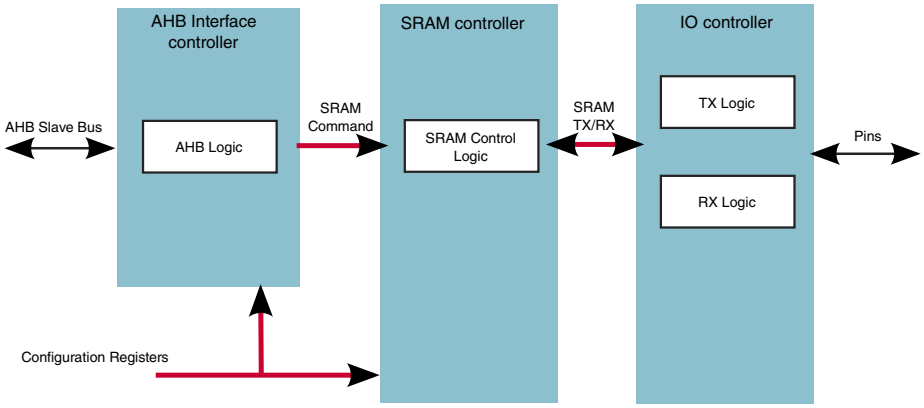


Figure 181. AHB SRAM Controller block diagram

30.2.2 Features

The AHB SRAM Controller includes the following features:

- Supports Asynchronous SRAM
- Supports 8/16 bit modes
- Supports ADMUX and Non-ADMUX modes
- Up to 4 Chip Select (CS)
- Up to 128 KB memory size each CS

30.3 Functional description

The following sections describe functional details of the AHB SRAM Controller module.

It support two address mux modes:

- ADMUX mode: Address/Data Multiplex mode
- Non-ADMUX mode: Address/Data Non-Multiplex mode

30.3.1 Operations

The following sections describe operations of the AHB SRAM Controller module.

30.3.1.1 SRAM Read Operation in ASYNC Mode

Figure 182, Figure 183 and Figure 184 show the ASYNC SRAM read in Non-ADMUX address mode.

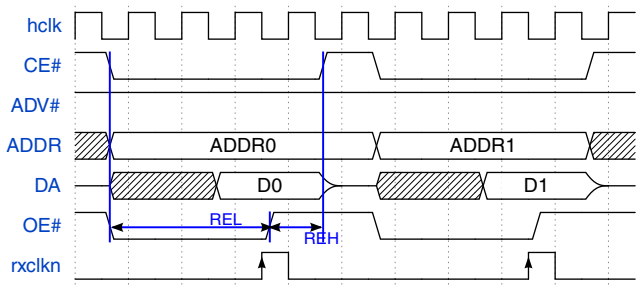


Figure 182. ASYNC SRAM Read in Non-ADMUX Address Mode

NOTE

- hclk is the AHB SRAM Controller functional clock. rxclk is the clock to sample data from SRAM.
- For this example:
  - CE# setup time is 0 clock cycle (SRAMCR0[CES]=0)
  - CE# hold time is 0 clock cycle (SRAMCR0[CEH]=0)
  - Address setup time is 0 clock cycle (SRAMCR0[AS]=0)
  - Address hold time is 0 clock cycle (SRAMCR0[AH]=0)
  - OE# low time is 3 clock cycles (SRAMCR1[REL]=2)
  - OE# high time is 1 clock cycle (SRAMCR1[REH]=0)
  - Prescaler timer's time granularity is 1 clock cycle (SRAMCR1[PRE]=0)

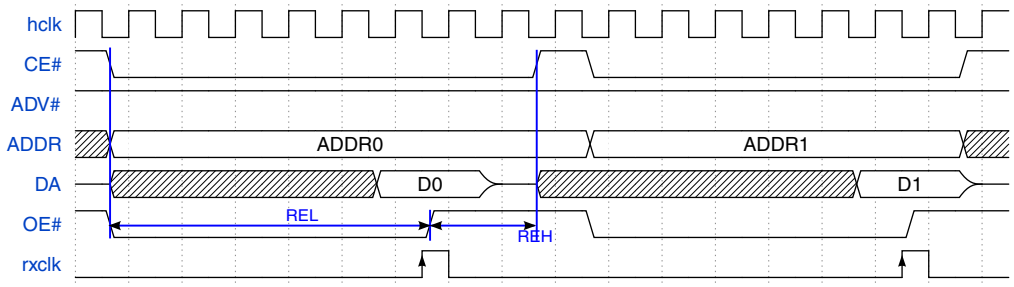


Figure 183. ASYNC SRAM Read in Non-ADMUX Address Mode

NOTE

- For this example:
  - CE# setup time is 0 clock cycle (SRAMCR0[CES]=0)
  - CE# hold time is 0 clock cycle (SRAMCR0[CEH]=0)
  - Address setup time is 0 clock cycle (SRAMCR0[AS]=0)
  - Address hold time is 0 clock cycle (SRAMCR0[AH]=0)
  - OE# low time is 6 clock cycles (SRAMCR1[REL]=2)
  - OE# high time is 2 clock cycle (SRAMCR1[REH]=0)
  - Prescaler timer's time granularity is 2 clock cycles (SRAMCR1[PRE]=1)

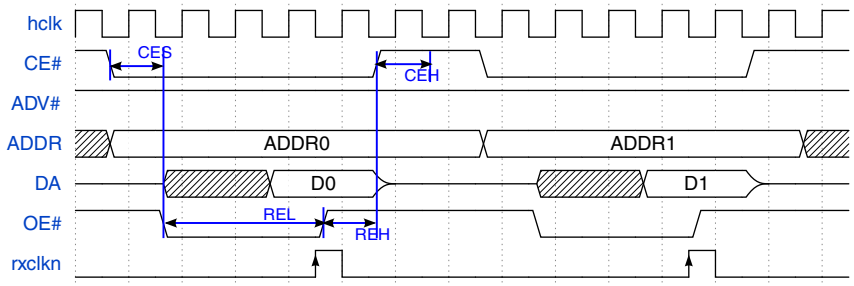


Figure 184. ASYNC SRAM Read in Non-ADMUX Address Mode

NOTE

- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR0[CES]=1)
  - CE# hold time is 1 clock cycle (SRAMCR0[CEH]=1)
  - Address setup time is 0 clock cycle (SRAMCR0[AS]=0)
  - Address hold time is 0 clock cycle (SRAMCR0[AH]=0)
  - OE# low time is 3 clock cycles (SRAMCR1[REL]=2)
  - OE# high time is 1 clock cycle (SRAMCR1[REH]=0)
  - Prescaler timer's time granularity is 1 clock cycle (SRAMCR1[PRE]=0)

Figure 185 and Figure 186 show the ASYNC SRAM read in ADMUX address mode.

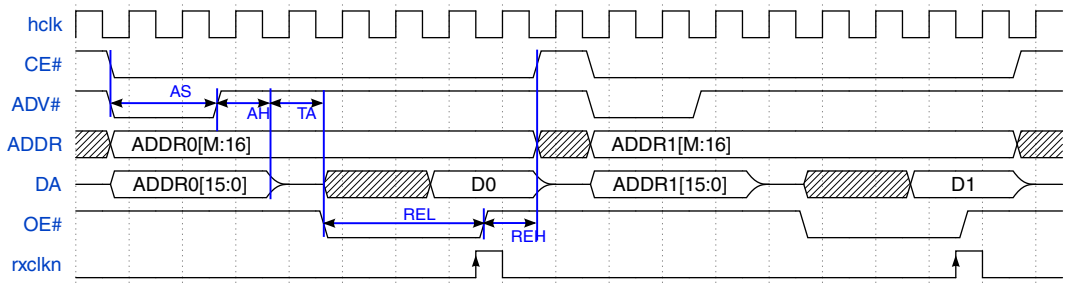


Figure 185. ASYNC SRAM Read in ADMUX Address Mode

NOTE

- For this example:
  - CE# setup time is 0 clock cycle (SRAMCR0[CES]=0)
  - CE# hold time is 0 clock cycle (SRAMCR0[CEH]=0)
  - Address setup time is 2 clock cycles (SRAMCR0[AS]=2)
  - Address hold time is 1 clock cycle (SRAMCR0[AH]=0)
  - Turnaround time is 1 clock cycle (SRAMCR0[TA]=0)
  - OE# low time is 3 clock cycles (SRAMCR1[REL]=2)
  - OE# high time is 1 clock cycle (SRAMCR1[REH]=0)
  - Prescaler timer's time granularity is 1 clock cycle (SRAMCR1[PRE]=0)

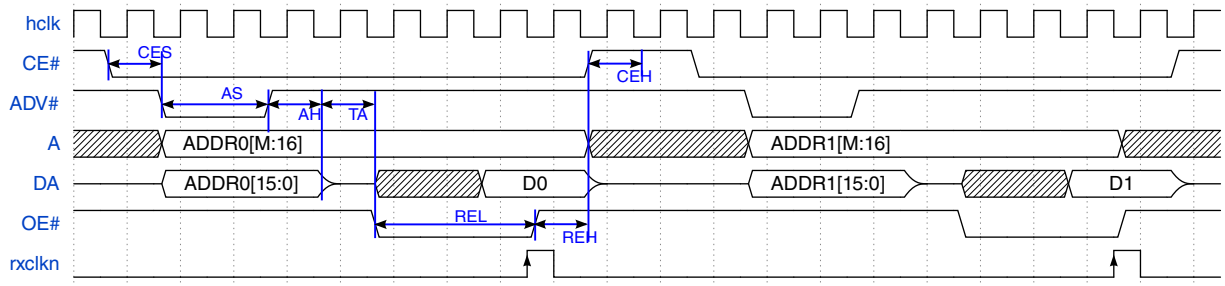


Figure 186. ASYNC SRAM Read in ADMUX Address Mode

NOTE

- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR0[CES]=1)
  - CE# hold time is 1 clock cycle (SRAMCR0[CEH]=1)
  - Address setup time is 2 clock cycles (SRAMCR0[AS]=2)
  - Address hold time is 1 clock cycle (SRAMCR0[AH]=0)
  - Turnaround time is 1 clock cycle (SRAMCR0[TA]=0)
  - OE# low time is 3 clock cycles (SRAMCR1[REL]=2)
  - OE# high time is 1 clock cycle (SRAMCR1[REH]=0)
  - Prescaler timer's time granularity is 1 clock cycle (SRAMCR1[PRE]=0)

30.3.1.2 SRAM Write Operation in ASYNC Mode

Figure 187, Figure 188 and Figure 189 show the ASYNC SRAM write in Non-ADMUX address mode.

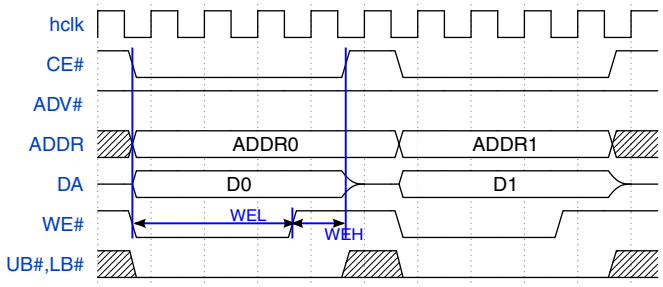


Figure 187. ASYNC SRAM Write in Non-ADMUX Address Mode

NOTE

- For this example:
  - CE# setup time is 0 clock cycle (SRAMCR0[CES]=0)
  - CE# hold time is 0 clock cycle (SRAMCR0[CEH]=0)
  - Address setup time is 0 clock cycle (SRAMCR0[AS]=0)
  - Address hold time is 0 clock cycle (SRAMCR0[AH]=0)
  - WE# low time is 3 clock cycles (SRAMCR1[WEL]=2)
  - WE# high time is 1 clock cycles (SRAMCR1[WEH]=0)
  - Prescaler timer's time granularity is 1 clock cycle (SRAMCR1[PRE]=0)

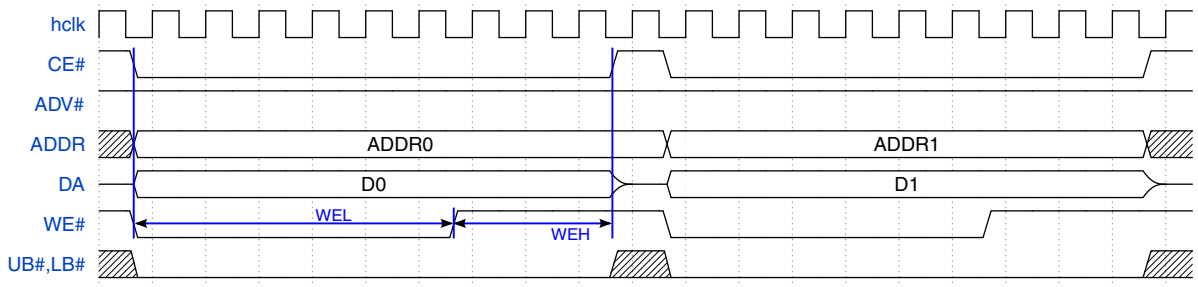


Figure 188. ASYNC SRAM Write in Non-ADMUX Address Mode

NOTE

- For this example:
  - CE# setup time is 0 clock cycle (SRAMCR0[CES]=0)
  - CE# hold time is 0 clock cycle (SRAMCR0[CEH]=0)
  - Address setup time is 0 clock cycle (SRAMCR0[AS]=0)
  - Address hold time is 0 clock cycle (SRAMCR0[AH]=0)
  - WE# low time is 6 clock cycles (SRAMCR1[WEL]=1)
  - WE# high time is 3 clock cycles (SRAMCR1[WEH]=0)
  - Prescaler timer's time granularity is 3 clock cycles (SRAMCR1[PRE]=2)

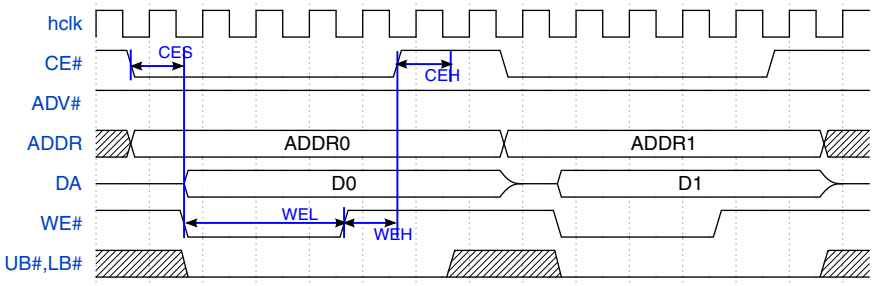


Figure 189. ASYNC SRAM Write in Non-ADMUX Address Mode

NOTE

- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR0[CES]=1)
  - CE# hold time is 1 clock cycle (SRAMCR0[CEH]=1)
  - Address setup time is 0 clock cycle (SRAMCR0[AS]=0)
  - Address hold time is 0 clock cycle (SRAMCR0[AH]=0)
  - WE# low time is 3 clock cycles (SRAMCR1[WEL]=2)
  - WE# high time is 1 clock cycles (SRAMCR1[WEH]=0)
  - Prescaler timer's time granularity is 1 clock cycle (SRAMCR1[PRE]=0)

Figure 190 and Figure 191 show the ASYNC SRAM write in ADMUX address mode.

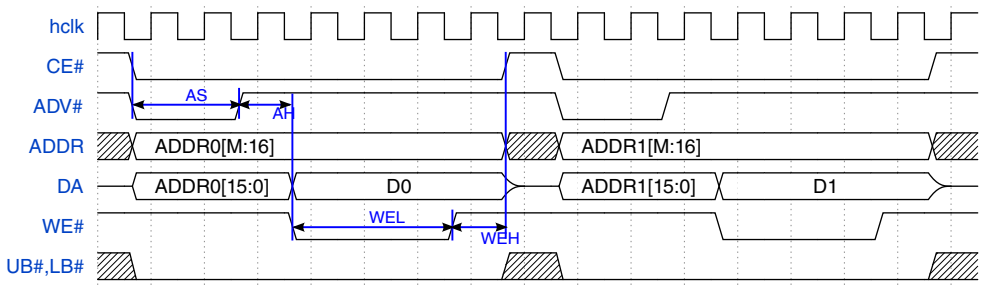


Figure 190. ASYNC SRAM Write in ADMUX Address Mode

NOTE

- For this example:
  - CE# setup time is 0 clock cycle (SRAMCR0[CES]=0)
  - CE# hold time is 0 clock cycle (SRAMCR0[CEH]=0)
  - Address setup time is 2 clock cycles (SRAMCR0[AS]=2)
  - Address hold time is 1 clock cycles (SRAMCR0[AH]=0)
  - WE# low time is 3 clock cycles (SRAMCR1[WEL]=2)
  - WE# high time is 1 clock cycle (SRAMCR1[WEH]=0)
  - Prescaler timer's time granularity is 1 clock cycle (SRAMCR1[PRE]=0)

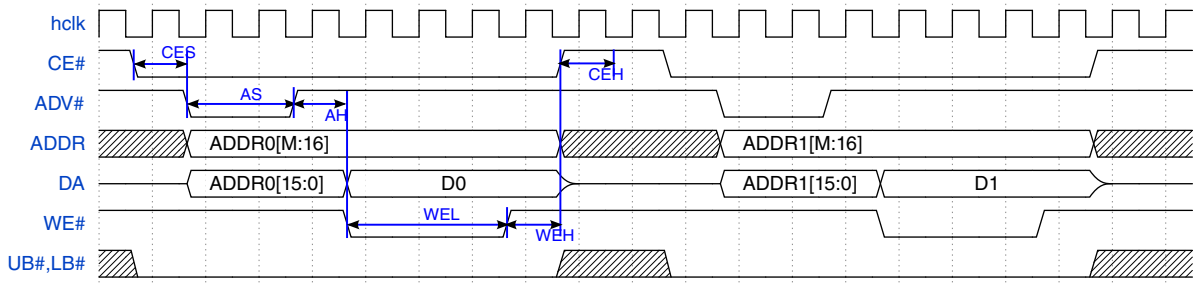


Figure 191. ASYNC SRAM Write in ADMUX Address Mode

NOTE

- For this example:
  - CE# setup time is 1 clock cycle (SRAMCR0[CES]=1)
  - CE# hold time is 1 clock cycle (SRAMCR0[CEH]=1)
  - Address setup time is 2 clock cycles (SRAMCR0[AS]=2)
  - Address hold time is 1 clock cycles (SRAMCR0[AH]=0)
  - WE# low time is 3 clock cycles (SRAMCR1[WEL]=2)
  - WE# high time is 1 clock cycle (SRAMCR1[WEH]=0)
  - Prescaler timer's time granularity is 1 clock cycle (SRAMCR1[PRE]=0)

30.3.2 Clocks

hclk is used for the functional clock for AHB SRAM Controller's AXI bus, internal logic and external interface. rxclkn is used to sample data from external SRAM device, which is generated from hclk.

30.3.3 Reset

There is a hardware reset for the AHB SRAM Controller, which reset all the logic inside it.

30.3.4 Interrupts

There is no interrupt output from the AHB SRAM Controller.

30.4 External signals

Table 193 lists the AHB SRAM Controller module signals.

Table 193. AHB SRAM Controller signals

Signal	Direction	Description
SRAMC_DA[15-00]	I/O	Data/Address Bits SRAMC_DATA works as data or address signal in applications..
SRAMC_ADDR[16-00]	O	Address Bits
SRAMC_CEB[3-0]	O	Chip Enable bits
SRAMC_OEB	O	Read Enable
SRAMC_WEB	O	Write Enable
SRAMC_ADVB	O	Address Enable
SRAMC_UBB	O	Upper Byte Control
SRAMC_LBB	O	Lower Byte Control

30.5 Initialization

To initialize the block:

- Ensure the clock for the AHB SRAM Controller is enabled.
- Configure corresponding pin MUX for the AHB SRAM Controller external signals.

- Configure registers of the AHB SRAM Controller properly, according to the applications.

30.6 Application information

This section describes applications supported by the AHB SRAM Controller module.

30.6.1 SRAM Application

This section provides the example sequences for SRAM device.

- Configure SRAM Control Registers with valid settings that matches SRAM device.
- Write and read operations can be triggered by AXI access.

30.7 Memory map and register definition

The AHB SRAM Controller registers are defined in chapter "Block Control - Wake-up Domain(BLK_CTRL_WAKEUP)" memory map, as registers SRAMCR0 and SRAMCR1.

Chapter 31
Flexible Serial Peripheral Interface (FlexSPI)

31.1 Chip-specific FlexSPI information

Table 194. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

Table 195. FlexSPI memory details

Start address	End address	Size (KB)	Description
2800_0000 (NS) 3800_0000 (S)	2FFF_FFFF (NS) 3FFF_FFFF (S)	131072	FlexSPI1
0400_0000 (NS)1400_0000 (S)	07FF_FFFF (NS)17FF_FFFF (S)	65536	FlexSPI2

31.1.1 AHB Memory Map Address Ranges

The table below shows the address range calculations for the various instances of FlexSPI:

Table 196. Memory Map Address Ranges

Description	FlexSPI1	FlexSPI2
Address range is mapped for AHB read access to IP RX FIFO	0x4742_0000 – 0x4742_00FF	0x4DE0_0000 – 0x4DE0_00FF
Address range is mapped for AHB write access to IP TX FIFO	0x4743_0000 – 0x4743_03FF	0x4DE1_0000 – 0x4DE1_03FF

Table continues on the next page...

Table 196. Memory Map Address Ranges (continued)

Description	FlexSPI1	FlexSPI2
AHB Memory Map for Serial Flash memory access	0x2800_0000	0x0400_0000
Total AHB RX Buffer size	4 KBytes	4 KBytes
AHB TX Buffer size	64 Bytes	64 Bytes
IP Bus memory space access	0x4742_0000 – 0x4742_00FF	0x4DE0_0000 – 0x4DE0_00FF
AHB bus memory space access	0x2800_0000 – 0x2FFF_FFFF	0x0400_0000 – 0x07FF_FFFF

31.1.2 Controller ID allocation

Table 197. Controller IDs

Module	Controller ID
NETC	0000
USB1, USB2	0001
FlexSPI Follower	0010
Cortex-M33	1000
Cortex-M7	1001
Edgelock	1010
eDMA3(AHB)	1011
eDMA4(AXI)	1100
uSDHC1, uSDHC2, and others	1111

31.1.3 FlexSPI connection modes

The table below shows details about the Parallel, Individual, and Combination modes for Flash memory.

Table 198. Relation between Modes and Bus size for FlexSPI instances

Mode	Instance	Effective Bus Size	Data Signals	Function	Devices Attached to Bus	Chip Selects
INDIVIDUAL	FLEXSPI1	4bit	A_DATA0- A_DATA3	Single QSPI chip support(1b,2b,4b)	Up to 2	A_SS0_B, A_SS1_B
INDIVIDUAL	FLEXSPI1	4bit	B_DATA0- B_DATA3	Single QSPI chip support(1b,2b,4b)	Up to 2	B_SS0_B, B_SS1_B
INDIVIDUAL	FLEXSPI1	8bit	A_DATA0- A_DATA7	Single Octal/	Up to 2	A_SS0_B, A_SS1_B

Table continues on the next page...

Table 198. Relation between Modes and Bus size for FlexSPI instances (continued)

Mode	Instance	Effective Bus Size	Data Signals	Function	Devices Attached to Bus	Chip Selects
				Hyperbus/ Xccela chip support(1x8b)		
INDIVIDUAL	FLEXSPI1	8bit	B_DATA0- B_DATA7	Single Octal/ Hyperbus/ Xccela chip support(1x8b)	Up to 2	B_SS0_B, B_SS1_B
PARALLEL	FLEXSPI1	8bit	A_DATA0- A_DATA3 | B_DATA0- B_DATA3	Two QSPI chips in parallel(2x4b)	2 parallel chips x2 = up to 4	(A_SS0_B | B_SS0_B), (A_SS1_B | B_SS1_B)
PARALLEL	FLEXSPI1	16bit	A_DATA0- A_DATA7 | B_DATA0- B_DATA7	Two Octal/ Hyperbus/ Xccela chips in parallel(2x8b)	2 parallel chips x2 = up to 4	(A_SS0_B | B_SS0_B), (A_SS1_B | B_SS1_B)
INDIVIDUAL	FLEXSPI2	4bit	A_DATA0- A_DATA3	Single QSPI chip support(1b,2b,4b)	Up to 2	A_SS0_B, A_SS1_B
INDIVIDUAL	FLEXSPI2	4bit	B_DATA0- B_DATA3	Single QSPI chip support(1b,2b,4b)	Up to 2	B_SS0_B, B_SS1_B
COMBINE	FLEXSPI2	8bit	A_DATA0- A_DATA3 | B_DATA0- B_DATA3	Single Octal/ Hyperbus/ Xccela chip support(1x8b)	Up to 2	A_SS0_B, A_SS1_B
PARALLEL	FLEXSPI2	8bit	A_DATA0- A_DATA3 | B_DATA0- B_DATA3	Two QSPI chips in parallel(2x4b)	2 parallel chips x2 = up to 4	(A_SS0_B | B_SS0_B), (A_SS1_B | B_SS1_B)

31.2 Overview

FlexSPI supports two SPI channels and up to four external devices. Each channel supports Single/Dual/Quad/Octal mode data transfer (1/2/4/8 bidirectional data lines).

The FlexSPI configuration depends on the chip configuration. See the system-level section for boot information and pinmux for chip-specific information regarding the modes and number of devices supported.

FlexSPI supports communication with both serial flash memory and serial RAM devices. While many of the descriptions, registers, and fields specifically reference flash memory, almost all information can also be applied to serial RAM. Flash memory is used as an example in the tables and figures in this chapter.

NOTE

Terminology in this chapter has been updated to align with JEDEC standard *Expanded Serial Peripheral Interface (xSPI) for Non Volatile Memory Devices, Version 1.0*.

Table 199. Updated terms

Updated term	Deprecated term
Controller	Master
Target	Slave

31.2.1 Block diagram

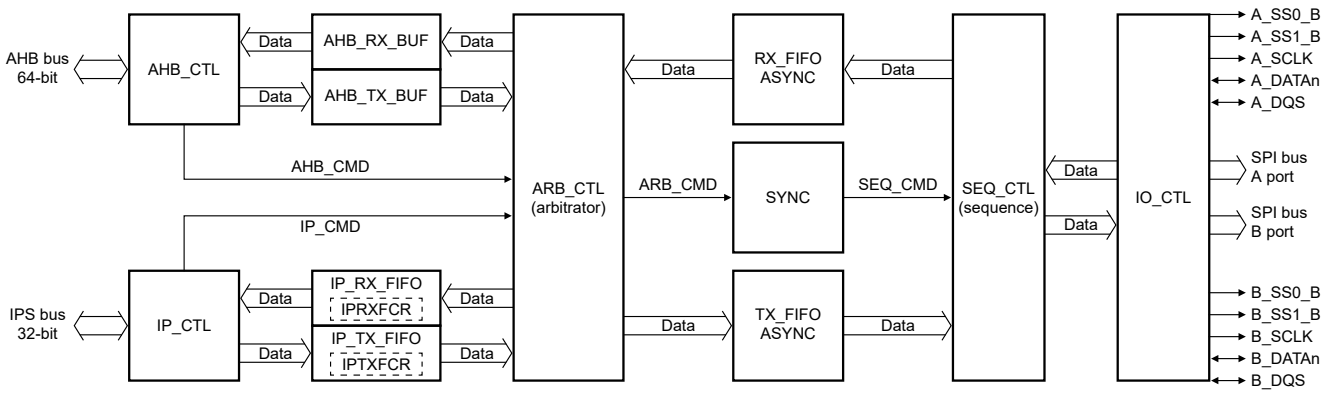


Figure 192. Block diagram

31.2.2 Features

FlexSPI supports:

- Flexible sequence engine (lookup table) to support various vendor devices
  - Serial NOR flash memory and other devices with SPI protocol similar to serial NOR flash memory
  - Serial NAND flash memory
  - HyperBus devices (HyperFlash/HyperRAM)
  - XCCELA devices
  - FPGA devices
- Flash memory access modes
  - Single, Dual, Quad, Octal mode
  - Single Data Transfer Rate (SDR) and Double Data Transfer Rate (DDR) mode
  - Individual/Parallel mode
- Sampling clock mode
  - Internal dummy read strobe looped back internally
  - Internal dummy read strobe looped back from pad

- Flash-memory-provided read strobe

- Memory mapped read and write access by AHB bus
  - AHB receive buffer implemented to reduce read latency. See the chip-specific section for AHB buffer size.
  - 16 AHB controllers with programmable priority for read access from each
  - 8 flexible and configurable buffers in AHB receive buffer
  - AHB transmit buffer implemented to buffer all write data from one AHB burst. See the chip-specific section for AHB buffer size.

NOTE

All AHB controllers share this AHB transmit buffer. There is no AHB controller number limitation for write access.

- Software-triggered flash memory read and write access by IP bus
  - IP receive FIFO implemented to buffer all read data from external devices. FIFO size: 256 bytes
  - IP transmit FIFO implemented to buffer all write data to external devices. FIFO size: 1024 bytes
  - DMA support to read IP receive FIFO
  - DMA support to fill IP transmit FIFO
  - SCLK stops when IP receive FIFO is full during reading flash memory data
  - SCLK stops when IP transmit FIFO is empty during writing flash memory data

31.3 Functional description

31.3.1 Flash connection

There are two FlexSPI interface ports (port A and port B). Each port supports up to two flash devices by providing two chip select outputs.

NOTE

FlexSPI configuration depends on the chip configuration. See the chip-specific FlexSPI information regarding the number of devices supported.

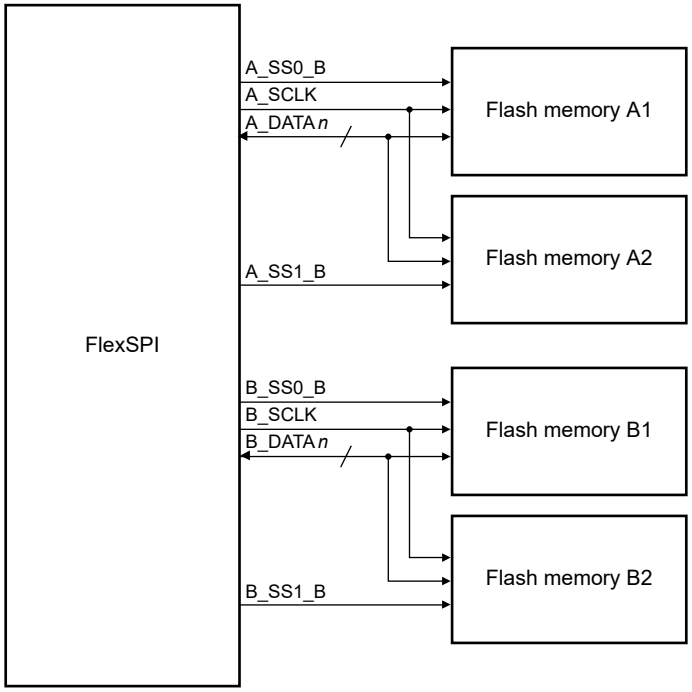


Figure 193. FlexSPI connection diagram with four devices.

NOTE

- Flash A1 and Flash A2 can be two flash chip packages or two flash dies in the same package. There is no difference to FlexSPI. This statement is also true for Flash B1 and Flash B2.
- Flash A1 and Flash B1 can be accessed in parallel, using parallel mode. FlexSPI merges or splits the flash read and program data automatically. This statement is also true for Flash A2 and Flash B2.
- In parallel mode, Flash A1 and Flash A2 cannot be accessed at the same time. This statement is also true for Flash B1 and Flash B2.
- In individual mode, flash devices cannot be accessed at the same time. But these four devices can be accessed separately.

There is a combination mode (MCR0[COMBINATIONEN] = 1) used to provide octal flash memory support by combining Port A (A_DATA[3:0]) and Port B (B_DATA[3:0]) together. Normal mode can also support Octal mode by using all eight data lines from Port A or Port B. Figure 194 shows the connection diagram for combination mode.

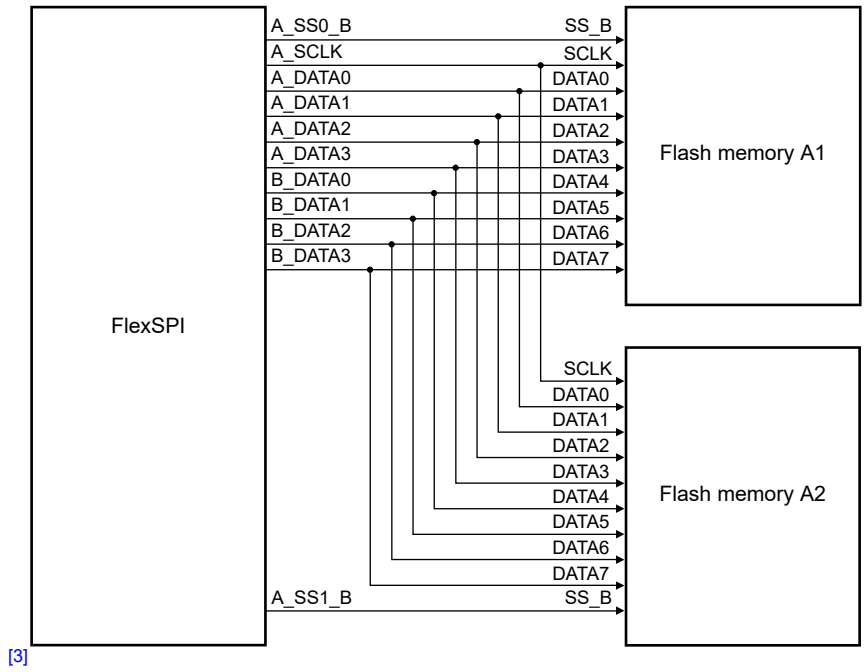


Figure 194. FlexSPI connection diagram for combination mode

31.3.2 Flash memory access mode

31.3.2.1 SPI clock mode

FlexSPI only supports SPI clock mode 0: clock polarity (CPOL) = 0 and clock phase (CPHA) = 0. When the SPI bus is idle, SCLK stays at a logic low state.

31.3.2.2 Individual mode and Parallel mode

In individual mode, read and write data are received and transmitted on port A or port B.

In parallel mode, read and write data are received and transmitted on port A and port B in parallel. FlexSPI merges or splits the read and program data automatically (READ/WRITE instruction). Only read and program data is merged or split (READ/WRITE instruction). For other instructions (such as Command, Address, Mode, and Data size), the same associated information is transmitted to the devices connected to port A and port B. For more details, see Executing instructions on SPI interface.

IPCR1[IPAREN] (for IP command) or AHBCR[APAREN] (for AHB command) statistically determines Individual and Parallel mode.

NOTE

FlexSPI does not support 16-bit read and write for the SPI interface.

31.3.2.3 SDR mode and DDR mode

In SDR mode, flash memory receives data on the rising edge of SCLK and transmits data on the falling edge of SCLK.

In DDR mode, flash memory receives and transmits data on both the rising and falling edges of SCLK.

[3] Parallel mode access is not available in combination mode.

The instruction (opcode) in the LUT sequence dynamically determines SDR and DDR modes. There is no static configuration register field setting for SDR and DDR modes. See FlexSPI input timing and FlexSPI output timing for details about input and output timing.

31.3.2.4 Single, Dual, Quad, and Octal mode

In Single mode, FlexSPI transmits and receives data on one data pin (DATA0 for transmitting, DATA1 for receiving).

In Dual mode, FlexSPI transmits and receives data on two data pins (DATA0–DATA1 for both transmitting and receiving).

In Quad mode, FlexSPI transmits and receives data on four data pins (DATA0–DATA3 for both transmitting and receiving).

In Octal mode, FlexSPI transmits and receives data on eight data pins (DATA0–DATA7 for both transmitting and receiving).

Instruction (num_pads) in the LUT sequence dynamically determine Single, Dual, Quad, and Octal modes. There is no static configuration register field setting for Single, Dual, Quad, and Octal modes.

31.3.3 Modes of operation

FlexSPI operates with these modes.

Table 200. Operating modes

Mode	Description
Module Disable	This mode is a low-power mode for FlexSPI. In Module Disable mode, the AHB clock and serial clock domains are gated off internally, but the IPS bus clock domain is not gated off. Read and write access to the control and status register are available, but the LUT, IP receive FIFO, and IP transmit FIFO cannot be accessed. Serial memory access is also not available. Write 1 to MCR0[MDIS] to enter this mode. Write 0 to MCR0[MDIS] to exit this mode.
Doze	This mode is a low-power mode for the chip. When the chip requires FlexSPI to enter Doze mode and MCR0[DOZEEN] = 1, FlexSPI enters Doze mode after all transactions are completed (STS0[ARBIDLE] = 1). In Doze mode, the AHB clock and serial clock domains are gated off internally, but the IPS bus clock is not gated off. Read and write access to the control and status register are available, but the LUT, IP receive FIFO, the IP transmit FIFO cannot be accessed. Serial memory access is also not available. This mode is entered via system request, and exited by deasserting this system request.
Stop	This mode is a low-power mode for the chip. When the chip requires the FlexSPI to enter Stop mode, FlexSPI waits for all transactions to complete (STS0[ARBIDLE] = 1) and return ACK handshake to the system. After the ACK handshake is returned, FlexSPI gates off the AHB clock and serial clock domains internally. The system can gate off the AHB bus clock, IPS bus clock, and serial clock at the system level. This mode is entered via system request. This mode is exited by deasserting this system request, and the ACK handshake message is also deasserted immediately.
Normal	No clock is gated off internally. Normal register access and serial memory access are available.

31.3.4 AHB access memory map

FlexSPI allocates AHB memory space for each memory device starting from the FlexSPI region base address of the system memory map. The FLSHxCR0[FLSHSZ] field determines the amount of memory allocated for each memory device in KB, Fx_SIZE. The memory is then allocated sequentially and contiguously.

NOTE

See the chip-specific section for the maximum flash size supported.

When MCR2[SAMEDEVICEEN] = 1:

- FA1_SIZE = FLSHA1CR0[FLSHSZ] × 1 KB
- FA2_SIZE = FLSHA1CR0[FLSHSZ] × 1 KB
- FB1_SIZE = FLSHA1CR0[FLSHSZ] × 1 KB
- FB2_SIZE = FLSHA1CR0[FLSHSZ] × 1 KB

When MCR2[SAMEDEVICEEN] = 0:

- FA1_SIZE = FLSHSZ × 1 KB
- FA2_SIZE = FLSHA2CR0[FLSHSZ] × 1 KB
- FB1_SIZE = FLSHB1CR0[FLSHSZ] × 1 KB
- FB2_SIZE = FLSHB2CR0[FLSHSZ] × 1 KB

Flash B1 and Flash B2 size settings are ignored in parallel mode (FLSHB1CR0[FLSHSZ], FLSHB2CR0[FLSHSZ]). To execute in parallel mode, Flash B1 should be the same device as Flash A1, and Flash B2 should be the same device as Flash A2.

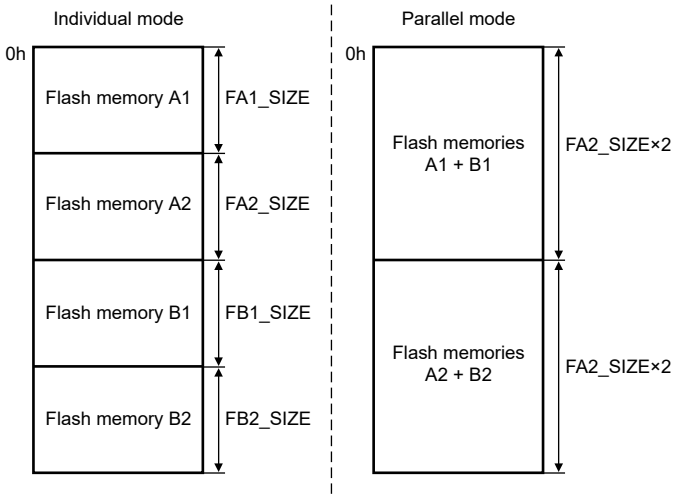


Figure 195. AHB access memory map in individual and parallel mode

AHB access memory map in individual mode:

- Flash A1 address range: 0000_0000h to FA1_SIZE
- Flash A2 address range: FA1_SIZE to (FA1_SIZE + FA2_SIZE)
- Flash B1 address range: (FA1_SIZE + FA2_SIZE) to (FA1_SIZE + FA2_SIZE + FB1_SIZE)
- Flash B2 address range: (FA1_SIZE + FA2_SIZE + FB1_SIZE) to (FA1_SIZE + FA2_SIZE + FB1_SIZE + FB2_SIZE)

AHB access memory map in parallel mode:

- Flash A1+B1 address range: 0000_0000h to FA1_SIZE × 2
- Flash A2+B2 address range: FA1_SIZE × 2 to (FA1_SIZE × 2 + FA2_SIZE × 2)

NOTE

The address used in Figure 195 and equations in this topic is the address presented to the memory. The base address for FlexSPI in the system memory map has already been removed, so it not used or shown here.

31.3.4.1 Flash address sent to flash memory devices

The AHB address (AHB command) or IPCR0[SFAR] (IP command) determines the flash memory access start address. See Flash memory access via AHB command and Flash memory access via IP command for more details.

For AHB commands, the FlexSPI controller removes the flash memory base address automatically when sending flash addresses to flash devices. The flash address is sent to devices in two parts: row address and column address. For flash devices that do not support the column address, set FLSHxCR1[CAS] to 0. This setting causes all flash address bits to be sent to flash devices as row addresses.

For word-addressable flash devices, the last bit of the address is not needed, because flash memory is read and programmed in terms of two bytes. For parallel mode, Flash A1 and Flash B1 (or Flash A2 and Flash B2) are accessed in parallel. As a result, the flash memory address sent to flash devices should be divided by 2. Table 201 indicates the relationship of row address, column address, and flash address (FA).

Table 201. Flash address, row address, and column address

Parallel mode	Word-addressable	Row address	Column address	Comment
0	0	FA[31:CAS]	FA[CAS-1:0]	There is no limitation on FA and data size alignment.
0	1	FA[31:CAS+1]	FA[CAS:1]	FA and data size should be two-byte aligned.
1	0	FA[31:CAS+1]	FA[CAS:1]	FA and data size should be two-byte aligned.
1	1	FA[31:CAS+2]	FA[CAS+1:2]	FA and data size should be four-byte aligned.

NOTE

- FA is the flash address without base address.
- If the bit number of the row or column flash address is more than the valid value, the high position bits are supplemented with zero. See Programmable sequence engine for details about row or column address instructions.

When parallel mode is enabled or word-addressable flash memory is used, there are limitations on the flash memory start address and data size. You can address this requirement by aligning the AHB bus access address (for AHB command) or IP command address IPCR0[SFAR] (for IP command) in software. There are two ways to avoid these limitations in the specified case.

1. For AHB Read Command only:

   When AHBCR[READADDROPT] and AHBCR[PREFETCHEN] are both 1, or OTFAD is enabled, FlexSPI guarantees the start address and data size for the flash memory access are eight-byte aligned by hardware.

   When AHBCR[READADDROPT] is 1, or OTFAD is enabled, FlexSPI fetches redundant data to guarantee the flash memory start address is eight-byte aligned. When prefetch is enabled (AHBCR[PREFETCHEN] is 1), the size of the eight-byte-aligned AHB receive buffer determines the flash memory read data size.

2. For AHB Write Command and Individual Mode only:

   By default, FlexSPI guarantees the flash memory write access start address and data size are two-byte aligned when using DQS as write mask (FLSHCR4[WMENB or WMENA] = 1).

   This feature is not applied in parallel mode and must be used if the external device supports write mask feature.

   For example, for odd start address:

- To write 6 bytes, FlexSPI sends 8 bytes, the first and last byte are both masked.
- To write 7 bytes, FlexSPI sends 8 bytes, only the first byte is masked.
- To write 8 bytes, FlexSPI sends 10 bytes, the first and last byte are both masked.

For even start address:

- To write 6 bytes, FlexSPI sends 6 bytes, no bytes are masked.
- To write 7 bytes, FlexSPI sends 8 bytes, last byte is masked.
- To write 8 bytes, FlexSPI sends 8 bytes, no bytes are masked.

31.3.5 Lookup table (LUT)

The LUT is an internal memory that preserves a number of preprogrammed sequences. Each sequence consists of up to eight instructions which are executed sequentially. When an IP command or an AHB command triggers a flash memory access, the FlexSPI controller:

1. Fetches the sequence from LUT (sequence index or number).
2. Executes the flash memory access to generate a valid flash transaction on the SPI interface.

Figure 196 shows the LUT structures, sequences, and instructions.

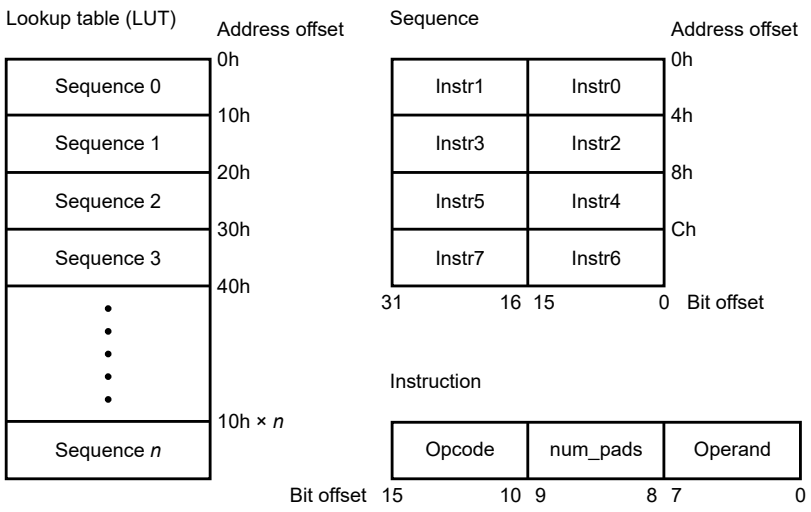


Figure 196. LUT structure, sequences, and instructions

NOTE

If the number of instructions needed for a flash transaction is less than eight, the STOP instruction (code 0000_0000h) must be programmed for unneeded instructions.

For IP and AHB write commands, the FlexSPI controller always executes from instruction pointer 0. For AHB read commands, the FlexSPI controller executes from a saved instruction start pointer. The FlexSPI controller saves the instruction start pointers separately for each flash device. All these saved instruction pointers are zero before the JMP_ON_CS instruction is executed. When the JMP_ON_CS instruction is executed, the operand in the JMP_ON_CS instruction is saved as instruction start pointer. See Execute-In-Place (XIP) enhanced mode.

The reset value of the LUT is unknown because it is implemented as internal memory. The LUT must be programmed according to the connected device. To protect its contents during a code runover, the LUT can be locked or unlocked to prevent unwanted changes after the LUT has been configured. The key to lock or unlock the LUT is **5AF05AF0h**.

To lock the LUT:

1. Write the key (5AF05AF0h) to LUT Key (LUTKEY).
2. Write 1 to LUTCR[LOCK] and write 0 to LUTCR[UNLOCK]. When there is another register write access to FlexSPI between these two write accesses, this LUT is not successfully locked.

To unlock the LUT:

1. Write the key (5AF05AF0h) to the LUT Key Register LUT Key (LUTKEY).
2. Write 0 to LUTCR[LOCK] and write 1 to LUTCR[UNLOCK]. When there is another register write access to FlexSPI between these two write accesses, this LUT is not successfully locked.

The lock status of the LUT can be read from LUTCR[LOCK] and LUTCR[UNLOCK].

31.3.6 Programmable sequence engine

The FlexSPI controller implements a programmable sequence engine that executes the sequence from the LUT. The FlexSPI controller executes the instructions sequentially, and generates flash transactions on the SPI interface accordingly. Table 202 is a complete list of the supported instructions.

Table 202. Instruction set

Name	Opcode	num_pads	Action on SPI interface	Transmit data	Bits/bytes/cycle number
CMD_SDR	00_0001h	00h - one pad (Single mode) 01h - two pads (Dual mode) 02h - four pads (Quad mode) 03h - eight pads (Octal mode)	Transmit command code to flash memory	Command code: Operand[7:0]	Bit number: 8
CMD_DDR	00_0021h				
RADDR_SDR	00_0002h		Transmit row address to flash memory See Flash address sent to flash memory devices	Row_Address[31:0] Row_Address comes from AHB bus (AHB command) or IPCR0[SFAR] (IP command).	Bit number: operand[7:0] Value of operand determines number of bits sent in Row_Address.
RADDR_DDR	00_0022h				
CADDR_SDR	00_0003h		Transmit column address to flash memory. See Flash address sent to flash memory devices.	Column_Address[31:0] Column_Address comes from AHB bus (AHB command) or IPCR0[SFAR] (IP command).	Bit number: operand[7:0] Value of operand determines number of bits sent in Column_Address.
CADDR_DDR	00_0023h				
MODE1_SDR	00_0004h		Transmit mode bits to flash memory	Mode bits: Operand[0]	Bit number: 1
MODE1_DDR	00_0024h				
MODE2_SDR	00_0005h			Mode bits: Operand[1:0]	Bit number: 2
MODE2_DDR	00_0025h				
MODE4_SDR	00_0006h			Mode bits: Operand[3:0]	Bit number: 4
MODE4_DDR	00_0026h				

Table continues on the next page...

Table 202. Instruction set (continued)

Name	Opcode	num_pads	Action on SPI interface	Transmit data	Bits/bytes/cycle number
MODE8_SDR	00_0007h			Mode bits: Operand[7:0]	Bit number: 8
MODE8_DDR	00_0027h				
WRITE_SDR	00_0008h		Transmit program data to flash device	Program data in IP_TX_FIFO or AHB_TX_BUF	AHB burst size and burst type (AHB command) or IPCR1[IDATSZ] (IP command) determines byte number (data size). For details about flash read or program data size, see Flash memory access via AHB command and Flash memory access via IP command.
WRITE_DDR	00_0028h				
READ_SDR	00_0009h		Receive read data from flash device Read data is put into AHB_RX_BUF or IP_RX_FIFO.	-	
READ_DDR	00_0029h				
DATSZ_SDR	00_000Bh		Transmit read or program data size (byte number) to flash device	Internal logic Read or program data size for current command sequence	Bit number: operand[7:0] Never set operand to zero or greater than 64 for DATSZ instruction.
DATSZ_DDR	00_002Bh				
DUMMY_SDR	00_000Ch		Leave data lines undriven by FlexSPI controller. Turnaround cycles are provided from host driving to device driving. num_pads determines number of pads in input mode.	-	Dummy cycle number (in serial root clock): Operand[7:0] Dummy cycle (N), described in data sheet of flash device, is in number of SCLK cycles. This number may be configurable. In SDR mode, SCLK cycle is same as serial root clock. Operand value must be set to N. In DDR mode, SCLK cycle is double serial root clock cycle. Operand value must be set to 2N, 2N-1 or 2N+1 depending on definition of dummy in data sheet of flash device. See Flash memory access sequence examples and dummy cycle definition in data sheet of external memory.
DUMMY_DDR	00_002Ch				
DUMMY_RWDS_SDR	00_000Dh		Similar to DUMMY_SDR/ DUMMY_DDR instruction.	-	For read command, dummy cycle number (in serial root clock):
DUMMY_RWDS_DDR	00_002Dh				

Table continues on the next page...

Table 202. Instruction set (continued)

Name	Opcode	num_pads	Action on SPI interface	Transmit data	Bits/bytes/cycle number
			Difference is in dummy cycle number. DQS pin is called RWDS in HyperBus specification. See Dummy instruction for details. **Set operand to "Latency count" for HyperBus devices.**		(operand[7:0] × 4 - 1) if RWDS (DQS pin) is high; (operand[7:0] × 2 - 1) if RWDS (DQS pin) is low; For write command, dummy cycle number (in serial root clock): (operand[7:0] × 4 - 2) if RWDS (DQS pin) is high; (operand[7:0] × 2 - 2) if RWDS (DQS pin) is low;
JMP_ON_CS	00_001Fh	Num_pads setting ignored.	Stop execution, deassert CS and save operand[7:0] as instruction start pointer for next sequence. Normally this instruction is used to support Execute-In-Place enhanced mode. See Execute-In-Place (XIP) enhanced mode. This instruction is only allowed for AHB read commands. When using this instruction in IP command or AHB write command, interrupt status flag set (INTR[IPCMDERR] or INTR[AHBCMDERR]).	-	No transaction on SPI interface.
STOP	00_0000h		Stop execution and deassert CS. Next command sequence (to same	-	

Table continues on the next page...

Table 202. Instruction set (continued)

Name	Opcode	num_pads	Action on SPI interface	Transmit data	Bits/bytes/cycle number
			flash device) starts from instruction pointer 0.		

The programmable sequence engine allows configuration of the LUT according to the connected external serial device. The flexible LUT structure easily adapts to new command or protocol changes from different vendors.

The DDR sequence is a flash memory access sequence that contains DDR instructions other than DUMMY_DDR, and may contain SDR instructions. The output and input timing on FlexSPI differs for SDR and DDR sequences. When included as part of a DDR sequence, SDR instructions execute differently from SDR instructions in an SDR sequence. See FlexSPI input timing and FlexSPI output timing.

31.3.6.1 Executing instructions on SPI interface

This section describes the execution of instructions on the SPI interface. For all instructions that receive bits from or transmit bits to flash devices:

- The bit order in one byte is higher on DATA7 than DATA0.
- The bit order is higher on port B than port A.

This order is shown in Figure 197.

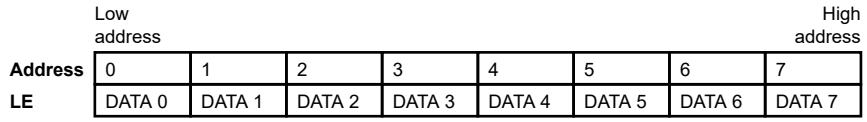


Figure 197. Bus bit order

Table 203. SPI instructions

Instructions	Description
Command (CMD_SDR, CMD_DDR)	Normally used to transmit a command code to the external device. The command code is the 8-bit operand in the instructions. Command code is sent to both port A and port B in parallel mode. See Flash memory access sequence examples.
Address (RADDR_SDR, RADDR_DDR, CADDR_SDR, CADDR_DDR)	Normally used to send the flash memory access start address (row address or column address) to an external device. FlexSPI determines the bits of the row address or column address according to the AHB access address or IP command address. See Flash address sent to flash memory devices. The operand value in the instruction code represents the number of address bits to send. The bits in the row address or column address are sent to both port A and port B in parallel mode. For example, when NUM_PADSn = 8, a memory reading the flash memory address on four SCK edges requires the sum of bits in CADDR and RADDR instructions to be 32. Otherwise, FlexSPI does not generate four SCK edges of address phase. See Flash memory access sequence examples.
Mode	Normally used to send mode bits to external devices. Mode bits are the lower bits of an operand. For example, the bit number is 1 for MODE1_SDR and MODE1_DDR, 2 for MODE2_SDR

Table continues on the next page...

Table 203. SPI instructions (continued)

Instructions	Description
(MODEx_SDR, MODEx_DDR)	and MODE2_DDR, 4 for MODE4_SDR and MODE4_DDR, and 8 for MODE8_SDR and MODE8_DDR. The pad number should not be greater than the mode bit number. For example, you cannot set NUM_PADSn to 11b (Octal mode) for MODE4_* instructions. Mode bits are sent to both port A and port B in parallel mode. See Flash memory access sequence examples for more details.
Data Size (DATSZ_SDR, DATSZ_DDR)	Used to send the size of program data or read data (byte number) to external devices. This instruction is normally used in FPGA applications where the memory space in external device acts like a FIFO. The external device requires data size information to determine how much data are popped from or pushed into the internal FIFO. The operation value in the data size instructions is the bit number. Data size is sent to both port A and port B in parallel mode. See Flash memory access sequence examples.
Write (WRITE_SDR, WRITE_DDR)	Normally used to send program data to external device. Programming data is fetched from IP_TX_FIFO (IP Command) or AHB_TX_Buffer (AHB command). For details about flash program data size, see Flash memory access via AHB command and Flash memory access via IP command. The byte order for program data is always from low to high. Odd bytes are sent on port A and even bytes are sent on port B in parallel mode. See Flash memory access sequence examples for more details.
Read (READ_SDR, READ_DDR)	Normally used to receive flash memory data from external devices. Received data is put into IP_RX_FIFO (IP Command) or AHB_RX_Buffer (AHB command). For information about flash memory read data size, see Flash memory access via AHB command and Flash memory access via IP command. The byte order for reading data is always from low to high. Odd bytes are received from port A and even bytes are received from port B in parallel mode. See Flash memory access sequence examples.
Dummy (DUMMY_SDR, DUMMY_DDR, DUMMY_RWDS_SDR, DUMMY_RWDS_DDR)	Used to provide turnaround cycles on the SPI interface. During dummy instruction, the FlexSPI controller and external devices do not drive the SPI interface. See Programmable sequence engine for details about dummy cycle number. DUMMY_RWDS_DDR can be used for a HyperBus device that uses RWDS pin to indicate whether extra latency is needed. DUMMY_RWDS_SDR is reserved. The FlexSPI controller checks the DQS pin input level at the fourth cycle after SCLK output toggling is enabled. The DQS pin is called RWDS in HyperBus specification. See Flash memory access sequence examples. **NOTE** FlexSPI releases the bus after at least one cycle. To avoid data contention, the NUM_PADSn value for the dummy commands should be configured to match the number of data lines used for the external memory.

31.3.6.2 Flash memory access sequence examples

Diagrams below all assume MCR0[SERCLKDIV] = 0.

Figure 198 shows an example SDR single I/O read sequence (Cypress Serial Nor Flash S25FS512S) in individual mode.

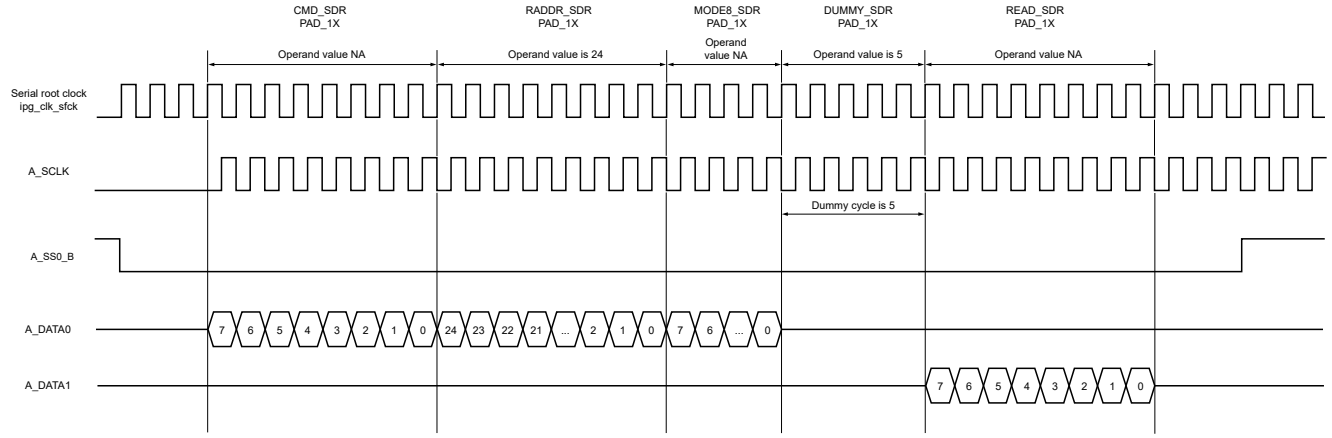


Figure 198. Flash memory access sequence example (SDR single I/O read sequence)

NOTE

- FlexSPI dummy instruction starts and ends at the rising edge of serial root clock.
- Device dummy cycle starts and ends at the falling edge of SCLK (on Cypress S25FS512S data sheet).

Figure 199 shows an example SDR quad I/O read sequence (Cypress Serial Nor Flash S25FS512S) in individual mode.

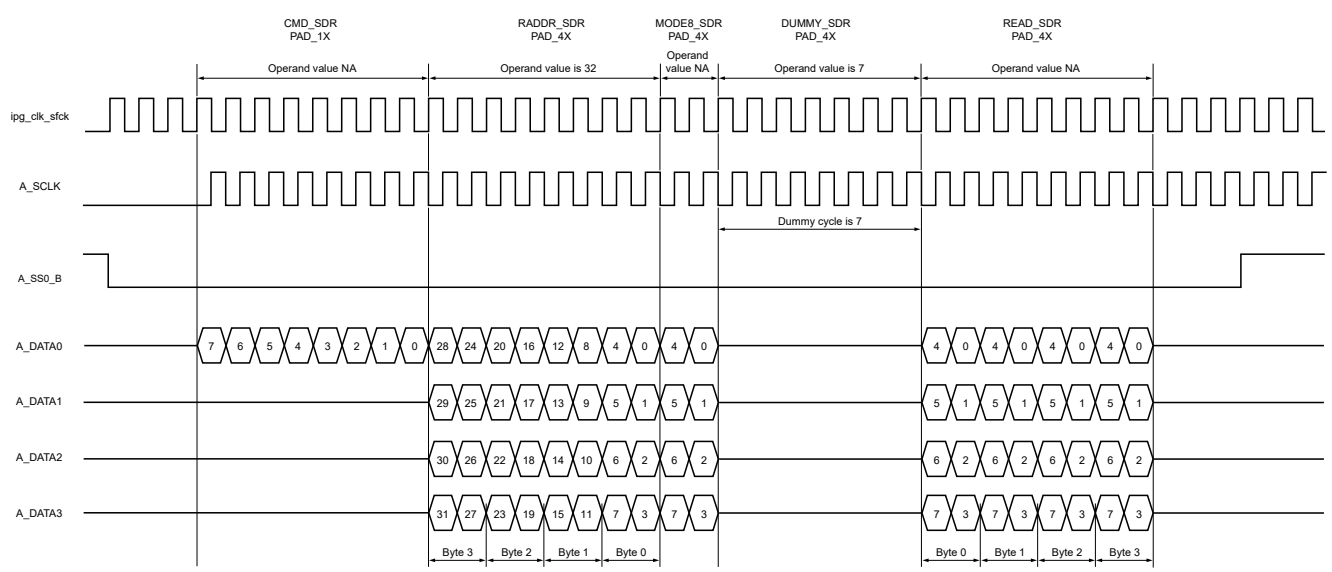


Figure 199. Flash memory access sequence example (SDR quad I/O read sequence)

NOTE

- FlexSPI dummy instruction starts and ends at the rising edge of serial root clock.
- Device dummy cycle starts and ends at the falling edge of SCLK (on Cypress S25FS512S data sheet).

Figure 200 shows an example DDR quad I/O read sequence (Cypress Serial Nor Flash S25FS512S) in parallel mode.

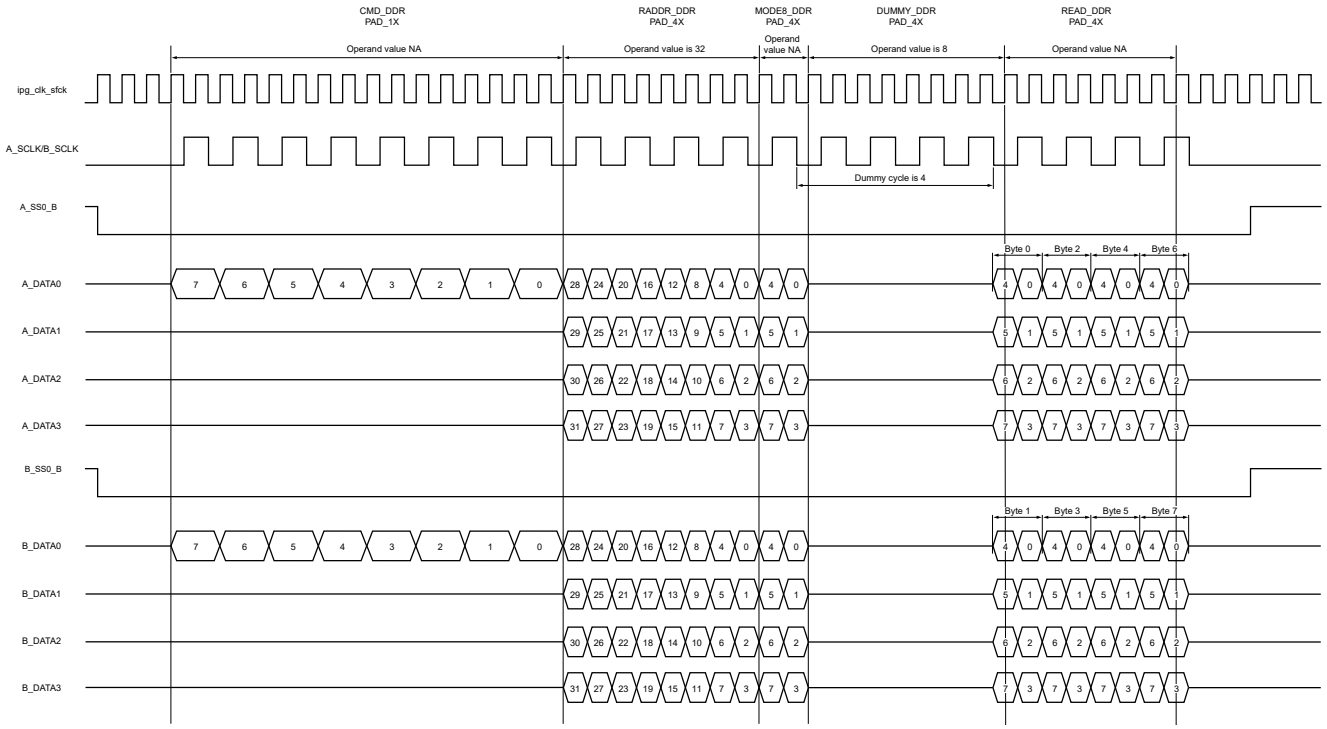


Figure 200. Flash memory access sequence example (DDR quad I/O read sequence)

Figure 201 shows an example HyperBus device read transaction (single latency count) in individual mode.

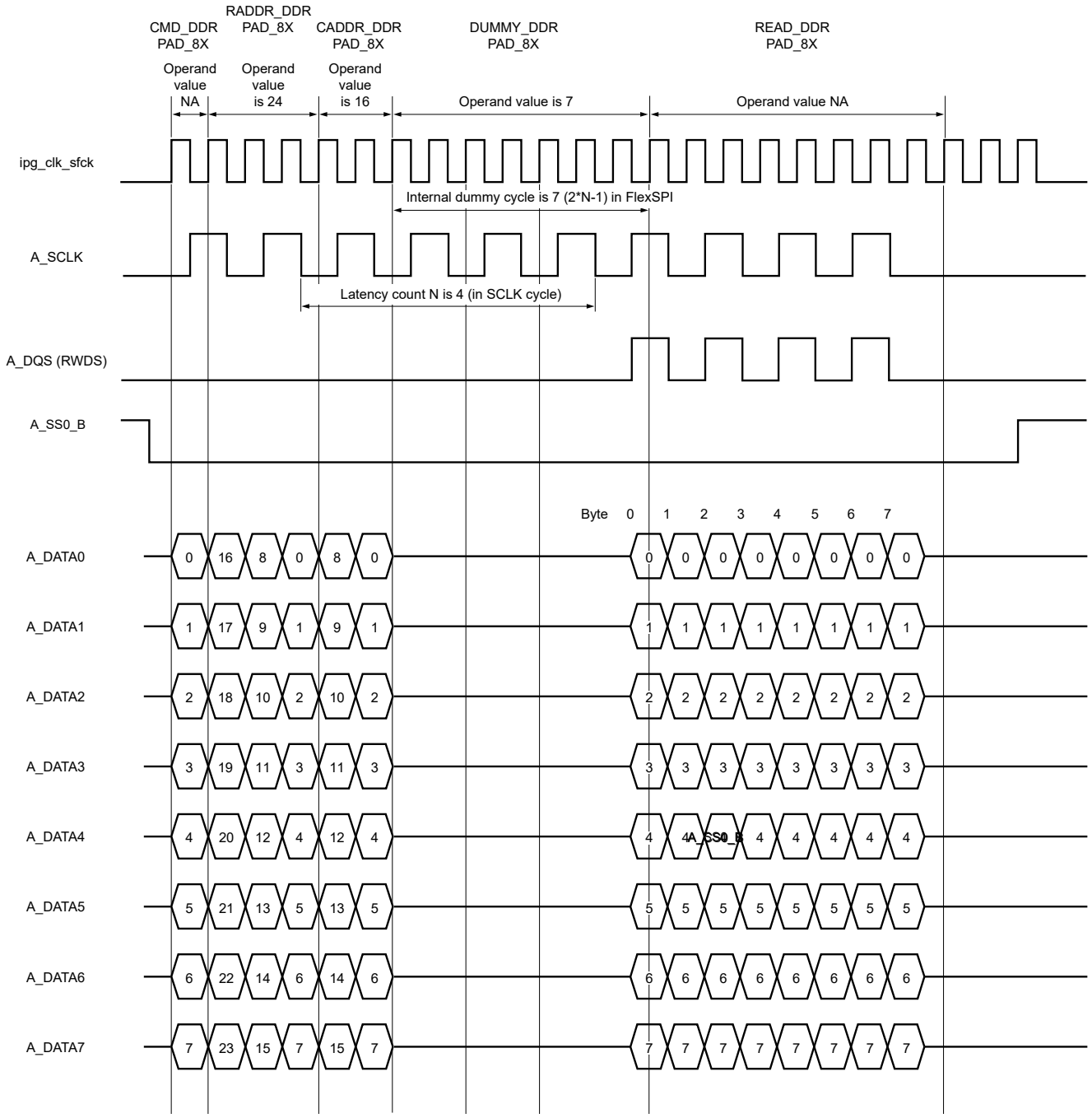


Figure 201. HyperBus device read transaction with single latency count

NOTE

FlexSPI continues to drive the clock until it has latched all of the read data it is expecting. On-chip delay for the data to reach FlexSPI can increase the number of SCK clock pulses.

Figure 202 shows an example HyperBus device read transaction (additional latency count) in individual mode.

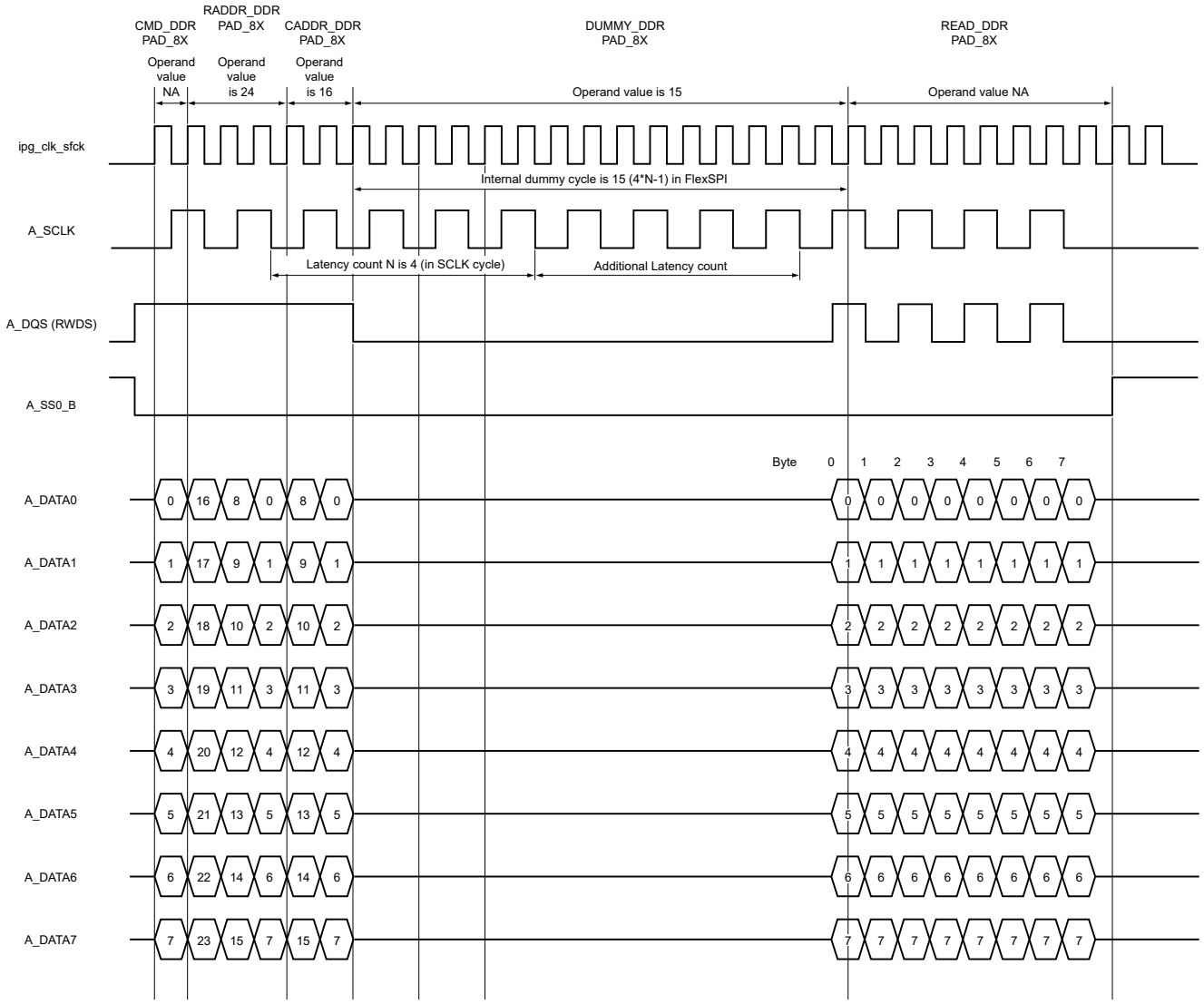


Figure 202. HyperBus device read transaction with additional latency count

Figure 203 shows an example HyperBus device write transaction (single latency count) in individual mode.

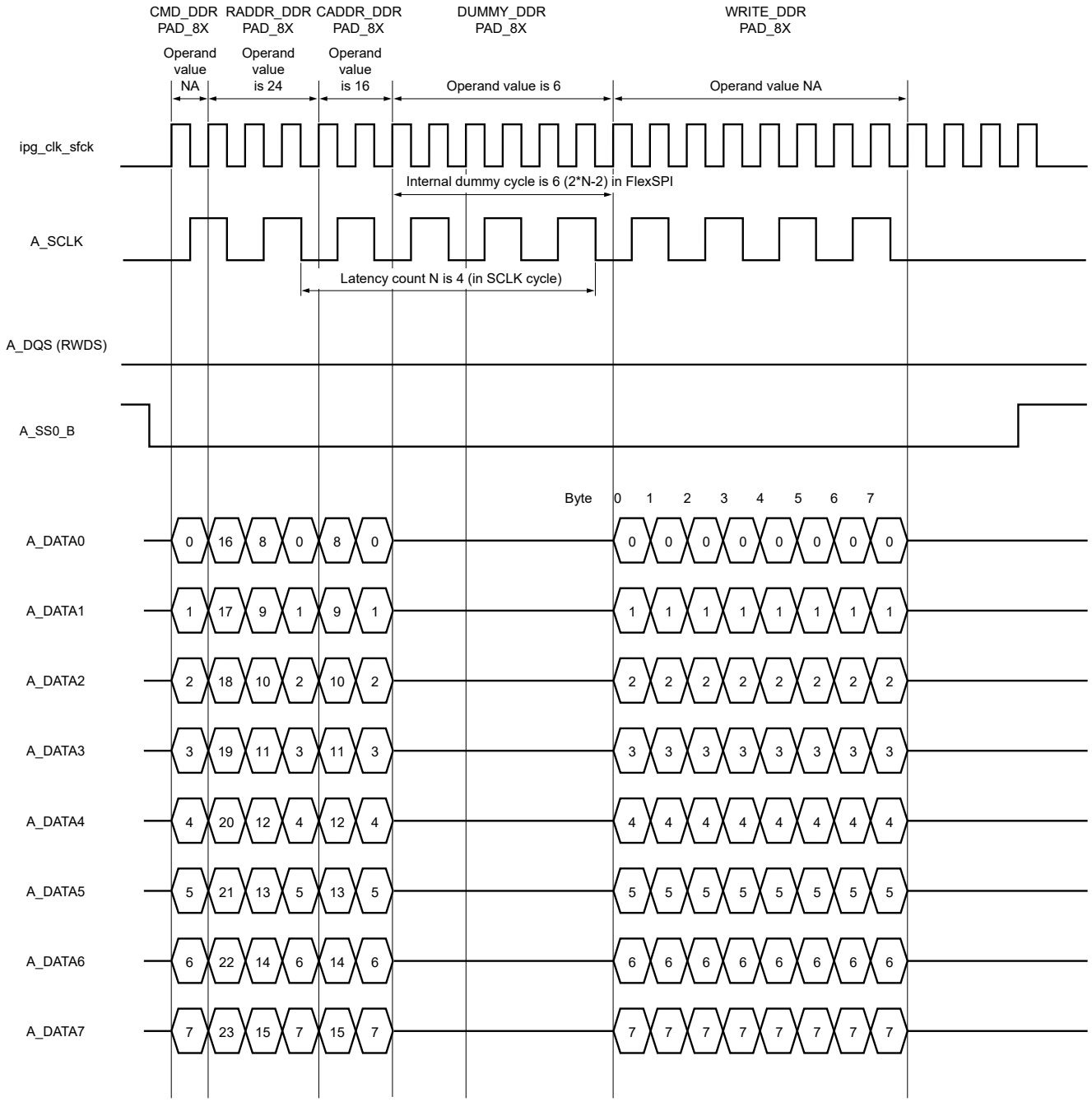


Figure 203. HyperBus device write transaction with single latency count

Figure 204 shows an example HyperBus device write transaction (additional latency count) in individual mode.

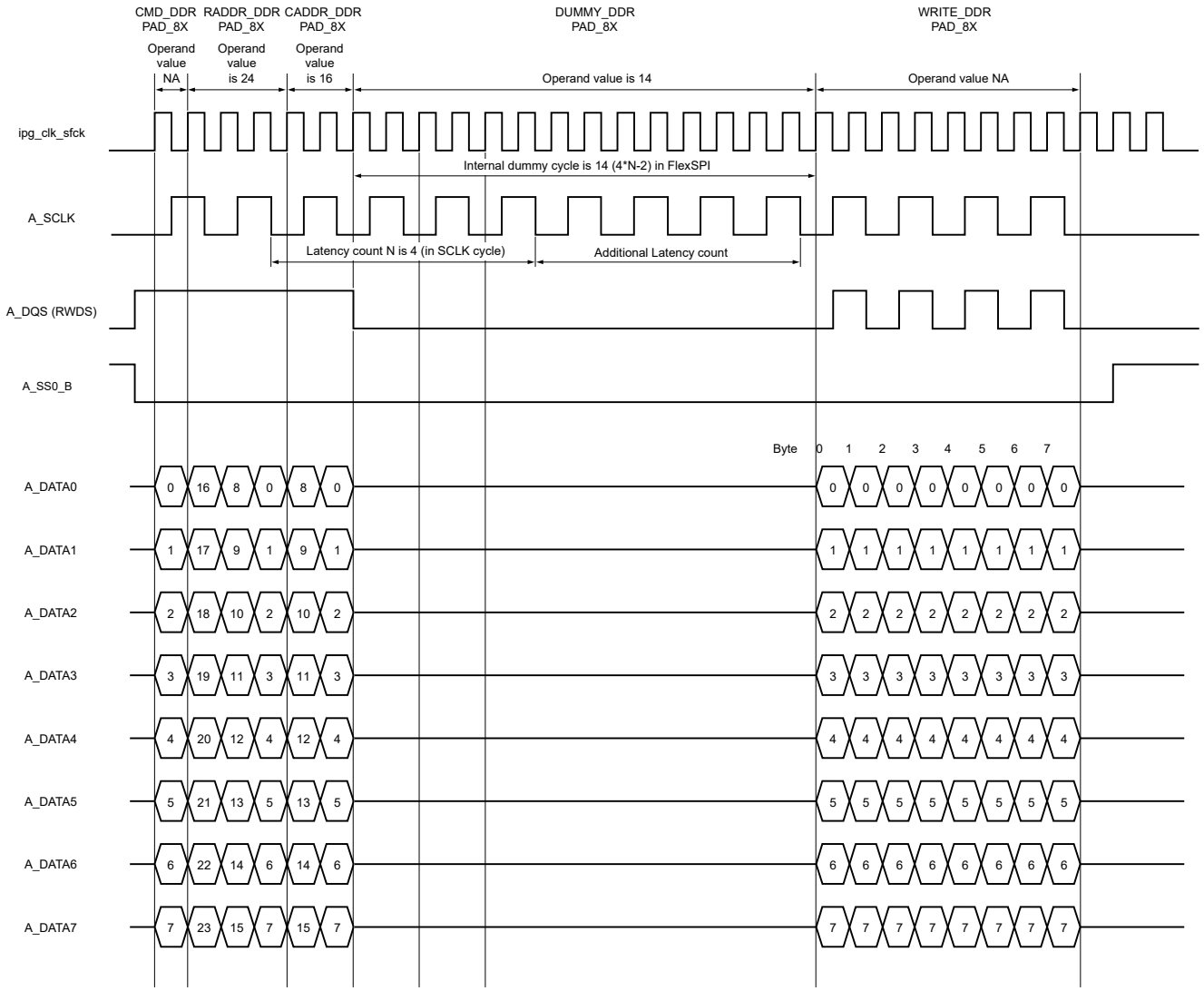


Figure 204. HyperBus device write transaction with additional latency count

31.3.7 Clocking

This section describes FlexSPI clocks and special clocking requirements.

Table 204. Clock usage

Clock name	Description	Comment
serial clock root (ipg_clk_sfck)	Root clock for Serial domain	FlexSPI core clock, used to generate the serial clock output. The clock is optionally divided using MCR0[SERCLKDIV] to obtain the SCLK.

Table continues on the next page...

Table 204. Clock usage (continued)

Clock name	Description	Comment
		Its frequency is same as SPI SCK frequency in SDR mode if MCR0[SERCLKDIV] = 0.
ahb clock (hclk)	AHB bus clock	hclk frequency is higher than 1/4 of serial root clock, so internal async FIFO is never full. This clock is used for internal logic in FlexSPI.
ipg clock (ipg_clk)	IPS bus clock	Register access clock. ipg_clk must be equal to hclk or an integer division of hclk. For example, ipg_clk = hclk/2.
SCLK	FlexSPI serial clock. Output clock on SCLK pin	Half clock frequency of serial clock root in DDR mode, and same frequency as serial clock root in SDR mode. Clock output toggles during the entire flash memory access sequence.
DQS_OUT	Dummy Read Strobe output	Same frequency as SCLK. Clock output toggles during READ and LEARN instructions. The DQS ensures that the data can be strobed correctly. It is only used for data strobe, instead of command or address strobe. The data includes useful read data and data learning patterns.
DQS_IN	Sample clock for receive data	Same frequency as SCLK. Sample clock comes from looped-back dummy read strobe, looped-back SCLK, or flash-memory-provided read strobe.

31.3.8 Interrupts

Table 205 describes all interrupts that FlexSPI generates.

Table 205. Flash memory address, row address, and column address

Interrupt	Enable	Condition
IP command done	INTEN[IPCMDDONEEN]	An IP command is finished.
IP command grant error	INTEN[IPCMDGEEN]	An IP command grant timeout occurs (bus not granted after MCR0[IPGRANTWAIT] × 1024 AHB clock cycles). See Overview of error categories, flags, and triggered sources.
AHB command grant error	INTEN[AHBCMDGEEN]	An AHB command grant timeout occurs (bus not granted after MCR0[AHBGRANTWAIT] × 1024 AHB

Table continues on the next page...

Table 205. Flash memory address, row address, and column address (continued)

		clock cycles). See Overview of error categories, flags, and triggered sources.
IP command error	INTEN[IPCMDERREN]	A command check error or a command execution error for an IP command occurs. See Overview of error categories, flags, and triggered sources.
AHB command error	INTEN[AHBCMDERREN]	A command check error or a command execution error for AHB command occurs. See Overview of error categories, flags, and triggered sources.
IP_RX_FIFO watermark available	INTEN[IPRXWAEN]	The fill level of IP_RX_FIFO reaches the watermark level (IPRXFCR[RXWMRK]).
IP_TX_FIFO watermark empty	INTEN[IPTXWEEN]	The empty level of IP_TX_FIFO reaches the watermark level (IPTXFCR[TXWMRK]).
Sequence execution timeout	INTEN[SEQTIMEOUTEN]	A sequence execution time exceeds the timeout wait time (MCR1[SEQWAIT]). For example, the following flash memory read command sequence lasts about 800_0000h cycles (ipg_clk_sfck). If SEQWAIT is set to FFFFh, a sequence timeout interrupt is generated. • IP command triggers timeout. • Flash memory read data size is 100_0000h bytes. • Flash memory accesses in single mode and SDR mode.
AHB bus error	INTEN[AHBBUSERROREN]	AHB bus response timeout occurs, or AHB read access occurs while OTFAD is enabled and FlexSPI is still fetching the key blob. For example, AHB read sequence is not configured properly in LUT (such as without a READ instruction). No data is read from the external device. As a result, FlexSPI cannot reach the read data in AHB_RX_Buffers for the AHB read command.
Write command stops SCLK	INTEN[SCKSTOPBYWREN]	IP_TX_FIFO is empty while executing a write command sequence. FlexSPI stops SCLK output clock toggling and waits for write data filling.
Read command stops SCLK	INTEN[SCKSTOPBYRDEN]	IP_RX_FIFO is full while executing a read command sequence. FlexSPI stops SCLK output clock toggling

Table continues on the next page...

Table 205. Flash memory address, row address, and column address (continued)

		and waits to read the data from IP_RX_FIFO.
Key done	INTEN[KEYDONEEN]	OTFAD is enabled and key blob processing is enabled, key blob processing is done, and no error is detected.
Key error	INTEN[KEYERROREN]	OTFAD is enabled and key blob processing is enabled, and an error occurs during key blob processing. Once this interrupt is set, only reset can clear it.

31.3.9 On-The-Fly AES Decryption (OTFAD) support

After power-up, if OTFAD and key fetches are enabled in the system, FlexSPI automatically starts fetching the key blobs from location 0 of flash A1. The key fetches continue until OTFAD receives and processes all key blobs. During the fetch operation, any AHB access causes an error response, and INTR[AHBBUSERROR] is set. No IPS access is allowed during the fetching of key blobs. INTR[KEYDONE] or INTR[KEYERROR] represents the key blob processing result.

After OTFAD processes all key blobs, the OTFAD engine decrypts any reads from FlexSPI and returns them on the AHB bus. When OTFAD is enabled, FlexSPI always enables prefetch; only AHBCR[READSZALIGN] can disable the prefetch. The Read Resume function is also disabled. All read access addresses are eight-byte-aligned automatically, regardless the setting of AHBCR[READADDROPT]. If OTFAD is enabled, the prefetch never crosses the 1 KB boundary.

31.3.10 Flash memory access via IP command

Follow these steps to trigger a flash memory access via IP command.

1. If this command is a programing command, fill the IP transmit FIFO with programming data (for instance, programming flash data and flash memory status registers.)
2. Write the flash memory access start address to IPCR0[SFAR].
3. Write the read or program data size to IPCR1[IDATSZ]. Write the sequence index to IPCR1[ISEQID]. Write the sequence number to IPCR1[ISEQNUM].
4. Write 1 to IPCMD[TRG] to trigger a flash memory access command.
5. Wait for the INTR[IPCMDDONE] flag to set or for the IP command done interrupt to fire, indicating the command has completed on the FlexSPI interface.

NOTE

- The IP transmit FIFO can be filled before or after writing the IPCR0, IPCR1, and IPCMD registers. If the SFM command starts with the IP transmit FIFO empty, FlexSPI stops SCLK toggling to wait for transmit data ready automatically.
- The IPCMD register must be written after writing the IPCR0 and IPCR1 registers.
- One IP command can issue up to eight command sequences.
- You cannot issue another IP command before the previous IP command is finished. Performing this action results in unknown behavior.

If this command is a read command, all read data from flash memory is put into the IP receive FIFO. Software must read data from the IP receive FIFO via AHB bus or IP bus. When the IP receive FIFO is full and the flash device still contains data to read, FlexSPI stops SCLK toggling until the FIFO has free space. See SCLK stop.

A triggered serial flash command contains:

- A flash memory access start address. IPCR0[SFAR] determines this address.
- A flash chip select. The flash memory access address and flash memory size setting (FLSHxCR0[FLSHSZ]) determine the chip select.
- A flash command sequence index and sequence number. FlexSPI sequentially executes the sequences indexed from IPCR1[ISEQID] to (IPCR1[ISEQID] + IPCR1[ISEQNUM]) in LUT.
- Flash individual/parallel memory access mode:

  Determined by IPCR1[IPAREN].
- The flash memory read or program data size, in bytes.
  - If the value of IPCR1[IDATSZ] is nonzero, the data size is IPCR1[IDATSZ].
  - If the value of IPCR1[IDATSZ] is zero, the operand value in the read or write instruction determines the data size.

NOTE

- Software must ensure that the last sequence index never exceeds the LUT sequence number. That is, IPCR1[ISEQID] + IPCR1[ISEQNUM] must be less than 32.
- For sequence numbers greater than one, the data size is applied to every command sequence.
- If there is no write or read instruction in the command sequence, the data size is ignored.

The IP command request is sent to the arbitrator after software triggers it. It is not executed on the FlexSPI interface until the arbitrator grants it. See Command arbitration.

31.3.10.1 Reading data from IP receive FIFO

FlexSPI puts read data from the external device into the IP receive FIFO for IP commands.

The data stored in the IP receive FIFO can be read using IPS and register or system and AHB bus transactions:

- Via IP bus
- Via AHB bus

If MCR0[ARDFEN] = 1, only the AHB bus can read the read data in the IP receive FIFO. Read access to the IP receive FIFO using IP bus always returns with data zero, and no bus error occurs.

If MCR0[ARDFEN] = 0, only the IPS bus can read the read data in the IP receive FIFO. Read access to the IP receive FIFO using the AHB bus triggers a bus error.

FlexSPI pushes read data into the IP receive FIFO in sets of 64 bits each time it receives 64 bits of data from an external device. When the number of read data bits is not 64-bit-aligned, FlexSPI pushes additional zero bits into the IP receive FIFO for the last push.

The processor or DMA can read the IP receive FIFO.

31.3.10.1.1 Reading via processor

To read data from the IP receive FIFO via processor, configure the items below.

- Write 0 to IPRXFCR[RXDMAEN].
- Write the watermark level to IPRXFCR[RXWMRK]. The watermark level is (IPRXFCR[RXWMRK] + 1) × 8 bytes.
- Write 1 to INTEN[IPRXWAEN] to enable the IP receive FIFO watermark available interrupt (optional).

The processor must poll INTR[IPRXWA] or wait for the IP receive FIFO watermark available interrupt before reading data from the FIFO. Waiting ensures that a watermark level amount of data is filled in the IP receive FIFO before reading.

After reading a watermark level amount of data from the FIFO, software must set the INTR[IPRXWA] flag to pop that data from the FIFO.

Figure 205 shows the reading flow from the IP receive FIFO via processor.

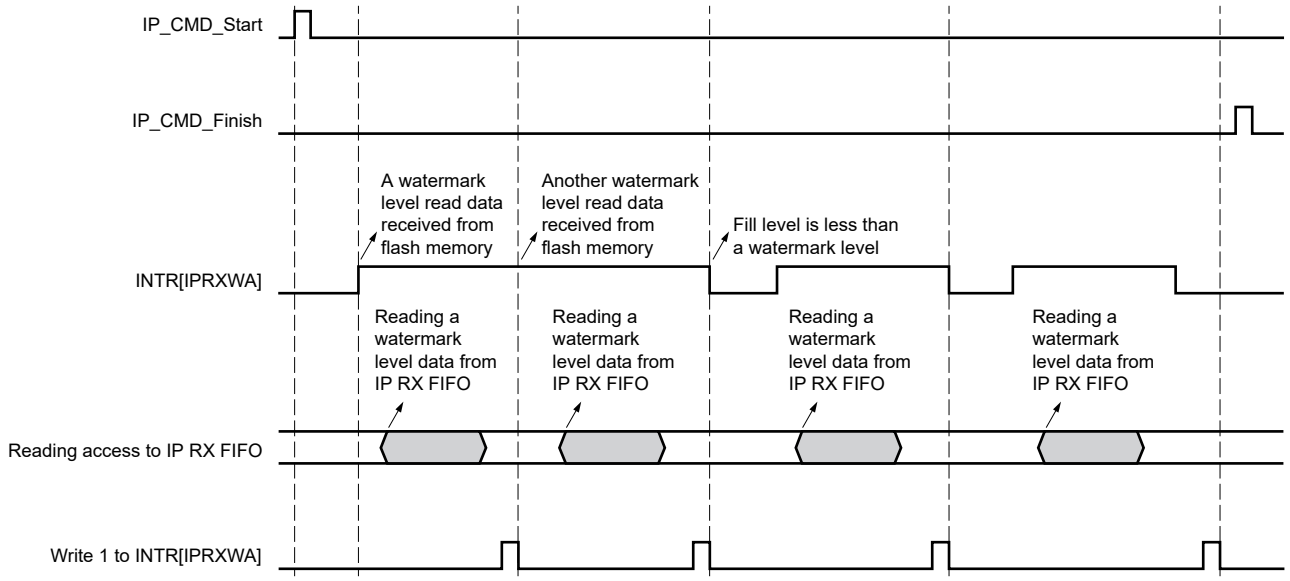


Figure 205. Reading IP receive FIFO via processor

NOTE

- The processor must read a watermark level of data from the IP receive FIFO before setting the INTR[IPRXWA] flag.
- The total flash read or program data size is not required to be a multiple of the watermark level. When the reading data size from the FIFO is less than a watermark level for the last time, software polls IPRXFSTS[FILL], not INTR[IPRXWA]. After all data from the FIFO are copied and all command sequences to flash memory are finished (INTR[IPCMDDONE] = 1), software sets IPRXFCR[CLRIPRXF] to clear the FIFO. If the FIFO is not cleared, the reading data will be incorrect for the next reading command to flash memory.
- Setting the INTR[IPRXWA] flag pops the IP receive FIFO data. Each read access to the FIFO does not pop the data.

31.3.10.1.2 Reading via DMA

To read data from the IP receive FIFO via DMA, configure the items below.

- Write 1 to IPRXFCR[RXDMAEN].
- Write the watermark level to IPRXFCR[RXWMRK]. The watermark level is (IPRXFCR[RXWMRK] + 1) × 8 bytes.
- Configure the DMA transfer minor loop size to match the watermark level.

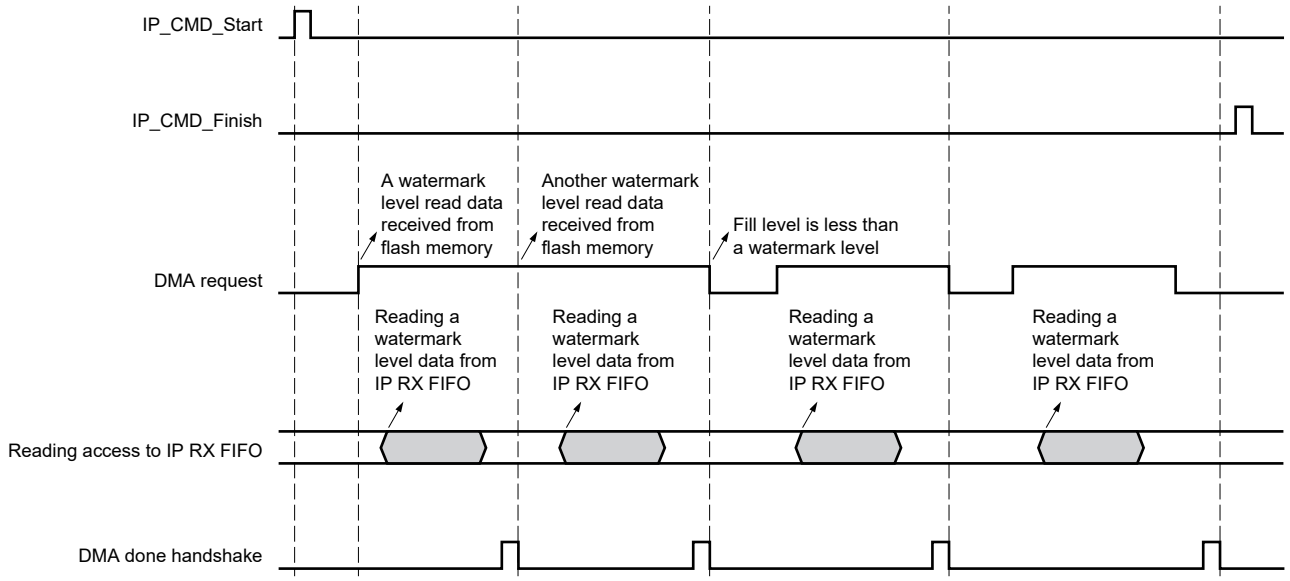


Figure 206. Reading from IP receive FIFO via DMA

NOTE

- When the fill level of the FIFO meets or exceeds the watermark level, a DMA request is generated. This request is level-valid, not pulse-valid.
- DMA reads a watermark level amount of data from the IP receive FIFO each time. (Configure the minor loop size to match the watermark level).
- DMA must return a done handshake (pulse valid) to FlexSPI each time it finishes reading a watermark level amount of data.
- The DMA done handshake (not each read access) pops the IP receive FIFO data.
- The total read and write data size (major loop size) must be a multiple of the watermark level. The DMA does not know when the data is ready for the last reading. When the data size is not a multiple of the watermark level, it is too complex for the DMA driver to poll IPRXFSTS[FILL].

31.3.10.2 Writing data to IP transmit FIFO

The programming data is put into the IP transmit FIFO, then FlexSPI transmits it to the flash memory. IPS or AHB access can write it:

- Via IP bus
- Via AHB bus

To write the programming data into the IP transmit FIFO via AHB bus, write 1 to MCR0[ATDFEN]. The IP bus write access to the FIFO is ignored, but no bus error occurs.

To write the programming data into the IP transmit FIFO via IP bus, write 0 to MCR0[ATDFEN]. The AHB bus write access to the FIFO returns a bus error.

When FlexSPI fetches data to transmit, 64 bits of data from the IP transmit FIFO are popped. If the programming data size is not a multiple of 64 bits, the number of last popped valid bits is less than 64 bits. No error occurs; these invalid bits are not transmitted to the flash memory.

The processor or DMA can fill the IP transmit FIFO with programming data.

31.3.10.2.1 Writing via processor

To write data into the IP transmit FIFO via processor, configure the items below.

- Write 0 to IPTXFCR[TXDMAEN].
- Write the watermark level to IPTXFCR[TXWMRK]. The watermark level is (IPTXFCR[TXWMRK] + 1) × 8 bytes.
- Write 1 to INTEN[IPTXWEEN] to enable the IP transmit FIFO watermark available interrupt (optional).

The processor must poll INTR[IPTXWE] or wait for the IP transmit FIFO watermark empty interrupt before writing data to the FIFO. Waiting ensures that a watermark level amount of data is available in the IP transmit FIFO before writing.

After writing a watermark level amount of data to the FIFO, software must set the INTR[IPTXWE] flag to push that data into the FIFO. (The write pointer is incremented.)

NOTE

Setting the INTR[IPTXWE] flag pushes the IP transmit FIFO data. Each write access to the FIFO does not push the data.

Figure 207 shows the writing flow to the IP transmit FIFO via processor.

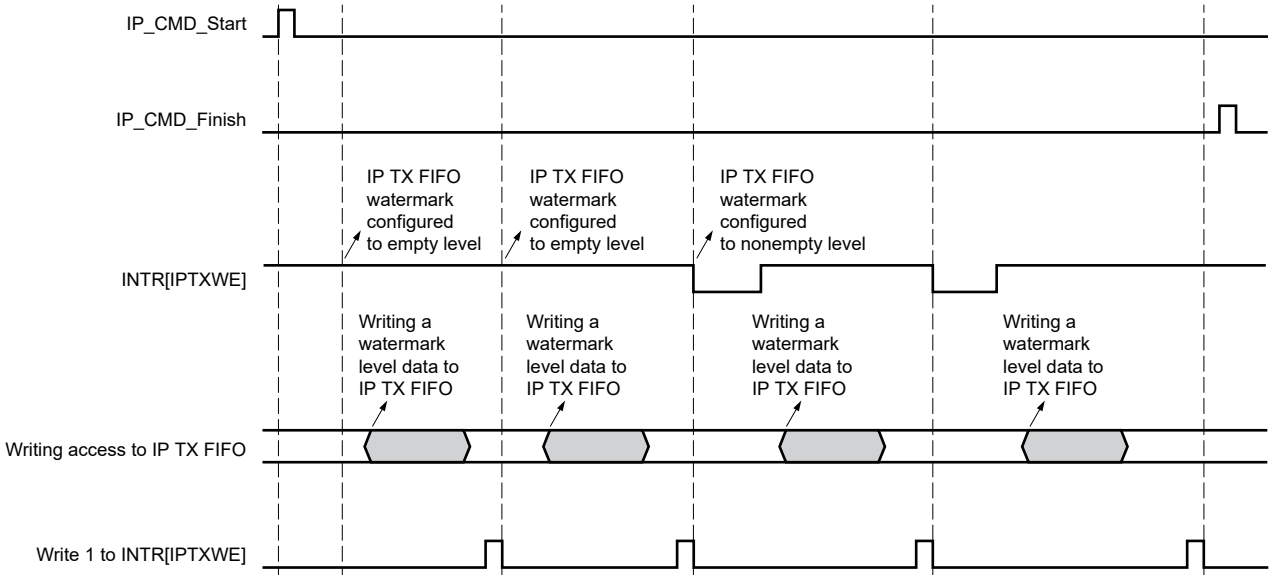


Figure 207. Writing to IP transmit FIFO via processor

NOTE

- The processor must write a watermark level amount of data into the IP transmit FIFO each time.
- The total flash write data size is not required to be a multiple of the watermark level. In this case, the writing data size to the FIFO is less than a watermark level for the last time. After all data to the FIFO are written and all command sequences to flash memory are finished (INTR[IPCMDDONE] = 1), the processor sets IPTXFCR[CLRIPTXF] to clear the FIFO. If the FIFO is not cleared, the programming data will be incorrect for the next programming command to flash memory.

31.3.10.2.2 Writing via DMA

To write data into the IP transmit FIFO via DMA, configure the items below.

- Write 1 to IPTXFCR[TXDMAEN].
- Write the watermark level to IPTXFCR[TXWMRK]. The watermark level is (IPTXFCR[TXWMRK] + 1) × 8 bytes.

- Configure the DMA transfer minor loop size to match the watermark level.

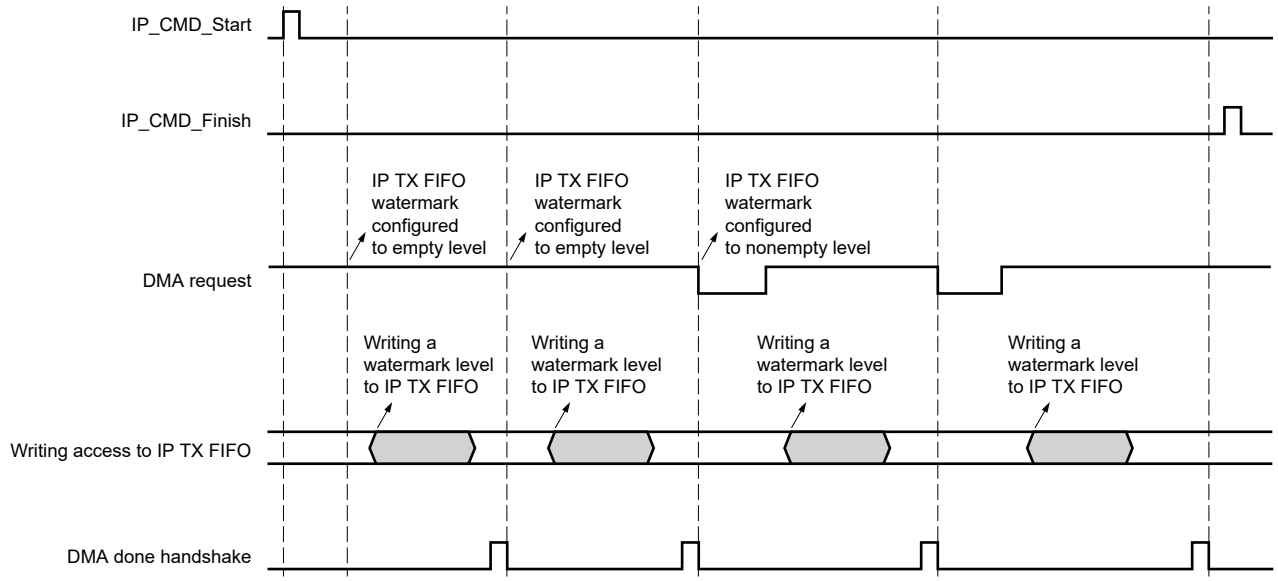


Figure 208. Writing into IP transfer FIFO via DMA

NOTE

- When there is empty space in the IP transmit FIFO greater than the watermark level, a DMA request is generated. This request is level-valid, not pulse-valid.
- DMA writes a watermark level amount of data into the FIFO each time. (Configure the minor loop size to match the watermark level.
- DMA returns a done handshake (pulse valid) to FlexSPI each time it finishes writing a watermark level amount of data.
- The DMA done handshake (not each write access) pushes the IP transmit FIFO data.
- The total program data size (major loop size) is not required to be a multiple of the watermark level. After writing all data into the IP transmit FIFO and all command sequences to flash memory are finished (INTR[IPCMDDONE] = 1), write 1 to IPTXFCR[CLRIPTXF] to clear the FIFO. Failure to clear the FIFO results in incorrect programming data for the next programming command to flash memory.

31.3.11 Flash memory access via AHB command

Flash memory can be accessed via AHB bus directly on the AHB address space. This address space is mapped to serial flash memory in FlexSPI. AHB bus accesses to this address space trigger flash memory access command sequences as needed.

For AHB read access to serial flash memory, FlexSPI fetches data from flash memory into the AHB receive buffer, then returns the data on the AHB bus. For AHB write access to serial flash memory, FlexSPI buffers AHB bus write data into the AHB transmit buffer, then transmits the data to serial flash memory.

No software configuration or polling is needed for AHB commands, except for FlexSPI initialization. As with a normal AHB target, the AHB controller accesses the external flash device transparently.

AHB commands are normally used to access serial flash memory space. IP commands must be used to access the control and status registers or other spaces, such as a one-time programmable (OTP) space in an external flash device.

The AHB commands for read and write access are discussed in detail below.

31.3.11.1 AHB write access to flash memory

For AHB write access to flash memory, FlexSPI buffers the write data from the AHB bus into the internal AHB transmit buffer. FlexSPI then transmits the data to flash memory. FlexSPI only buffers write data for one AHB burst. Figure 209 shows the hardware operation in response to an AHB write access to flash memory.

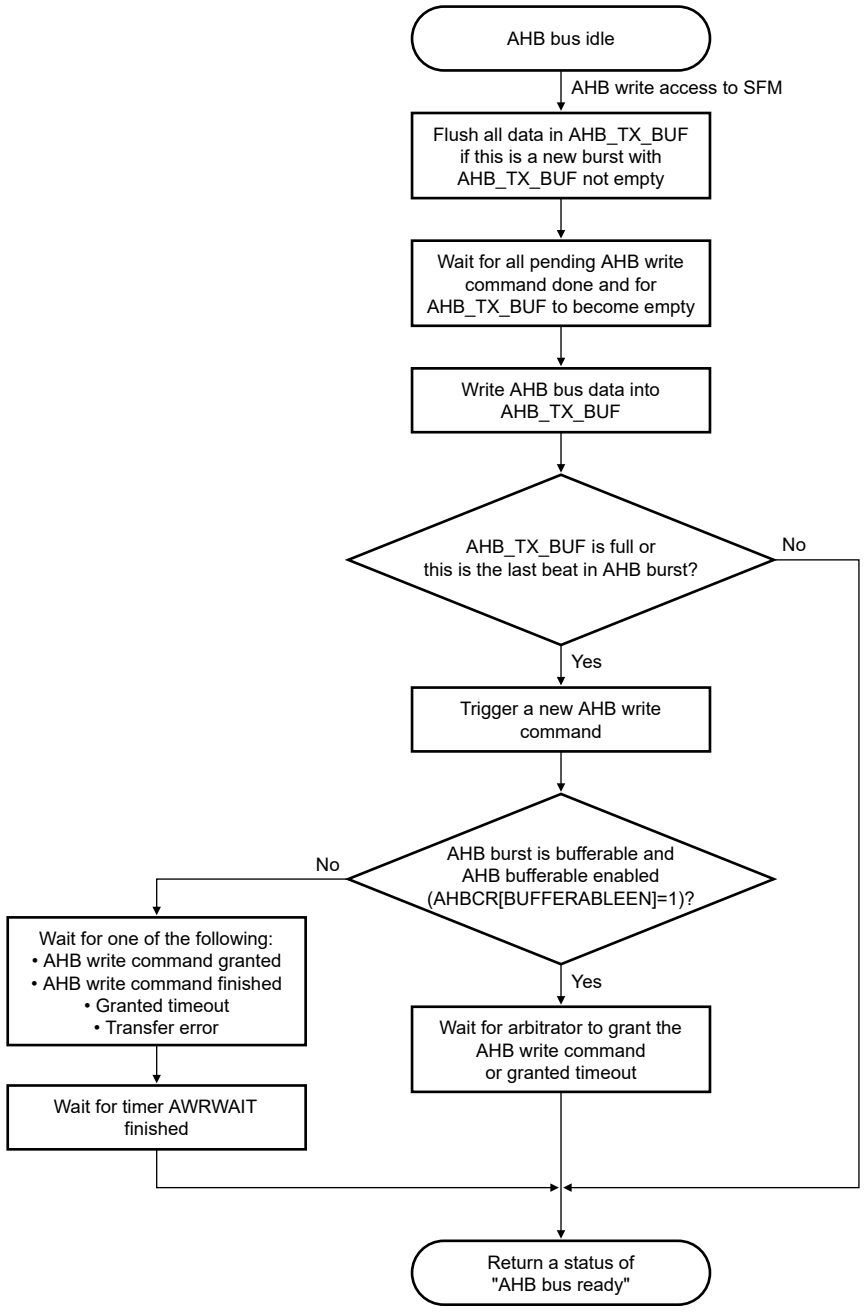


Figure 209. Hardware operation in response to AHB write access to flash memory

NOTE

1. If previous AHB burst is INCR write burst, AHB transmit buffer may not be empty when AHB bus is idle or new burst arrives. A new AHB write command is triggered to flush all data in AHB transmit buffer to external flash device. The new burst is held until this AHB write command finishes.
2. If the current access is bufferable, when the AHB write command is triggered, AHB ready is returned after the write command is granted. If the current access is non-bufferable, AHB ready is returned after the write command finishes.

FlexSPI triggers a new AHB write command in these cases:

- This beat is the last one in the current AHB burst, for any burst type except INCR.
- AHB transmit buffer is full after the current beat data is buffered.
- AHB bus becomes idle or a new burst arrives with the AHB transmit buffer not empty.

Table 206 describes the details of a triggered AHB write command.

Table 206. Triggered AHB write command

Item	Description
Flash memory access start address	Determined by AHB burst address. FlexSPI records the start address for the data in AHB transmit buffer, which is used as flash memory access start address.
Flash memory chip select	FlexSPI uses the settings of the flash memory access start address and flash memory size to determine the chip selection.
Flash memory command sequence index	Determined by FLSHxCR2[AWRSEQID].
Flash memory command sequence number	Determined by FLSHxCR2[AWRSEQNUM]. If AWRSEQNUM is not zero, multiple flash memory access command sequences are triggered every time for an AHB write command. The sequences indexed from AWRSEQID to (AWRSEQID + AWRSEQNUM) in LUT are executed sequentially.
Flash memory access mode (Individual/ Parallel)	Determined by AHBCR[APAREN].
Flash memory data size	Determined by byte number of buffered data in AHB transmit buffer.

The following examples indicate internal logic for AHB write access to flash memory, where AHB_TX_BUF is 64 bytes :

- AHB INCR 64-bit bufferable burst with address sequence 8h, 10h, 18h, 20h, ..., 50h (10 beats total).

  Two AHB write commands are triggered: the first command with flash memory start address 8h and data size 40h; the second command with flash memory start address 48h and data size 10h. See Figure 210.

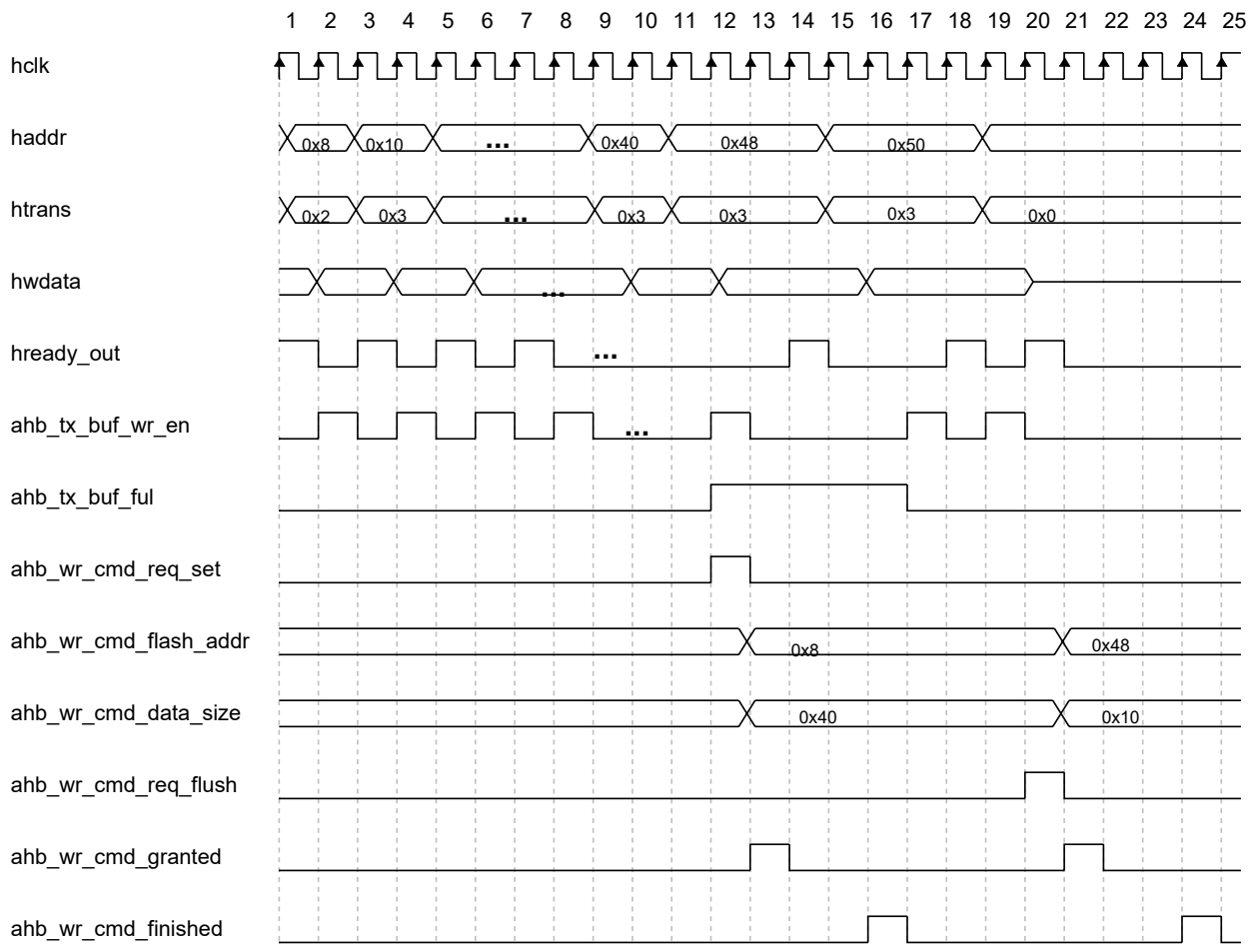


Figure 210. AHB write access (INCR, 64-bit, bufferable)

- AHB WRAP8 64-bit bufferable burst with address sequence 28h, 30h, 38h, 0h, 8h, 10h, 18h, 20h.

  One AHB write command is triggered with flash memory start address 0h and data size 40h. See Figure 211.

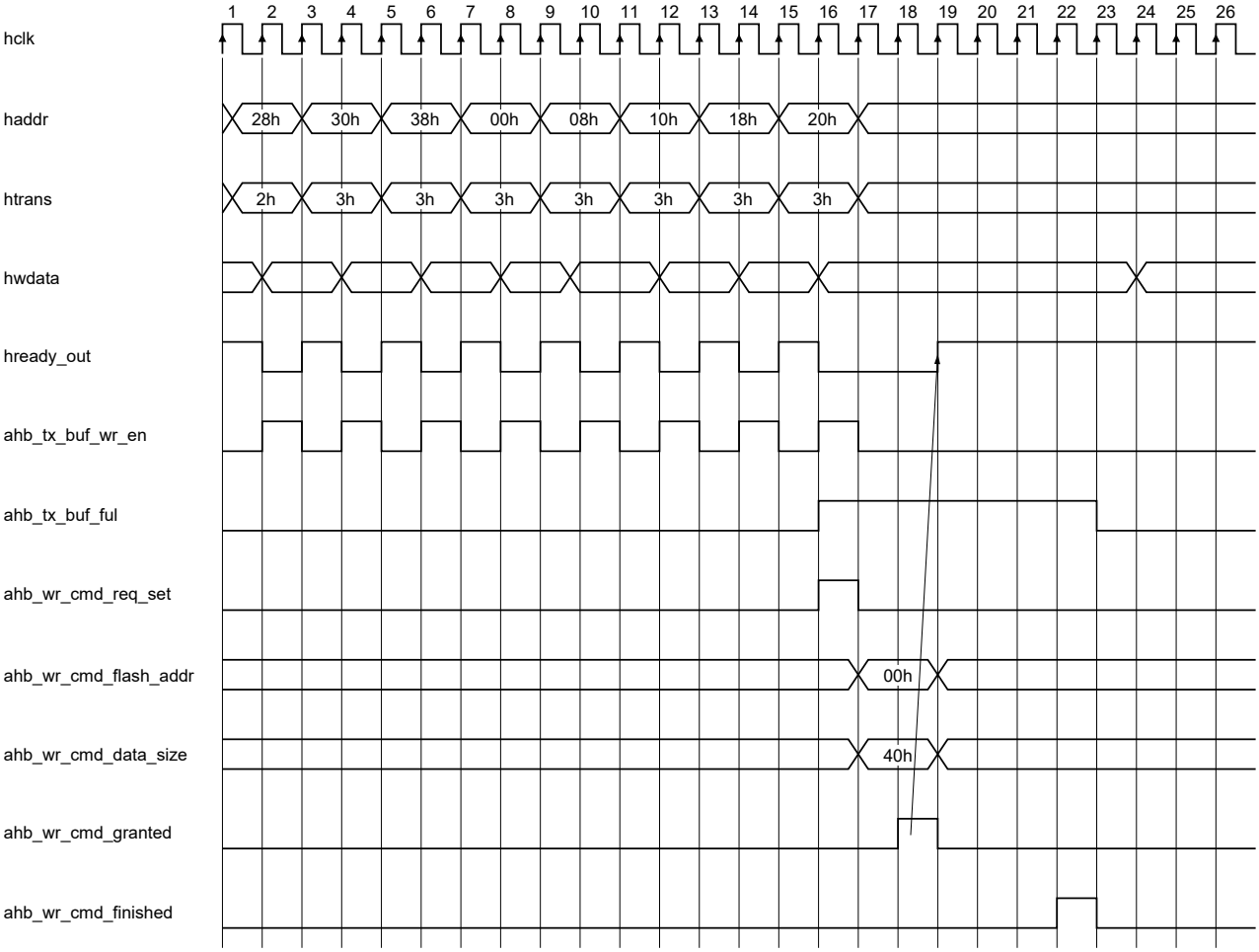


Figure 211. AHB write access (WRAP8, 64-bit, bufferable)

- AHB WRAP8 64-bit non-bufferable burst with address sequence 28h, 30h, 38h, 0h, 8h, 10h, 18h, 20h.

  One AHB write command is triggered with flash memory start address 0h and data size 40h.

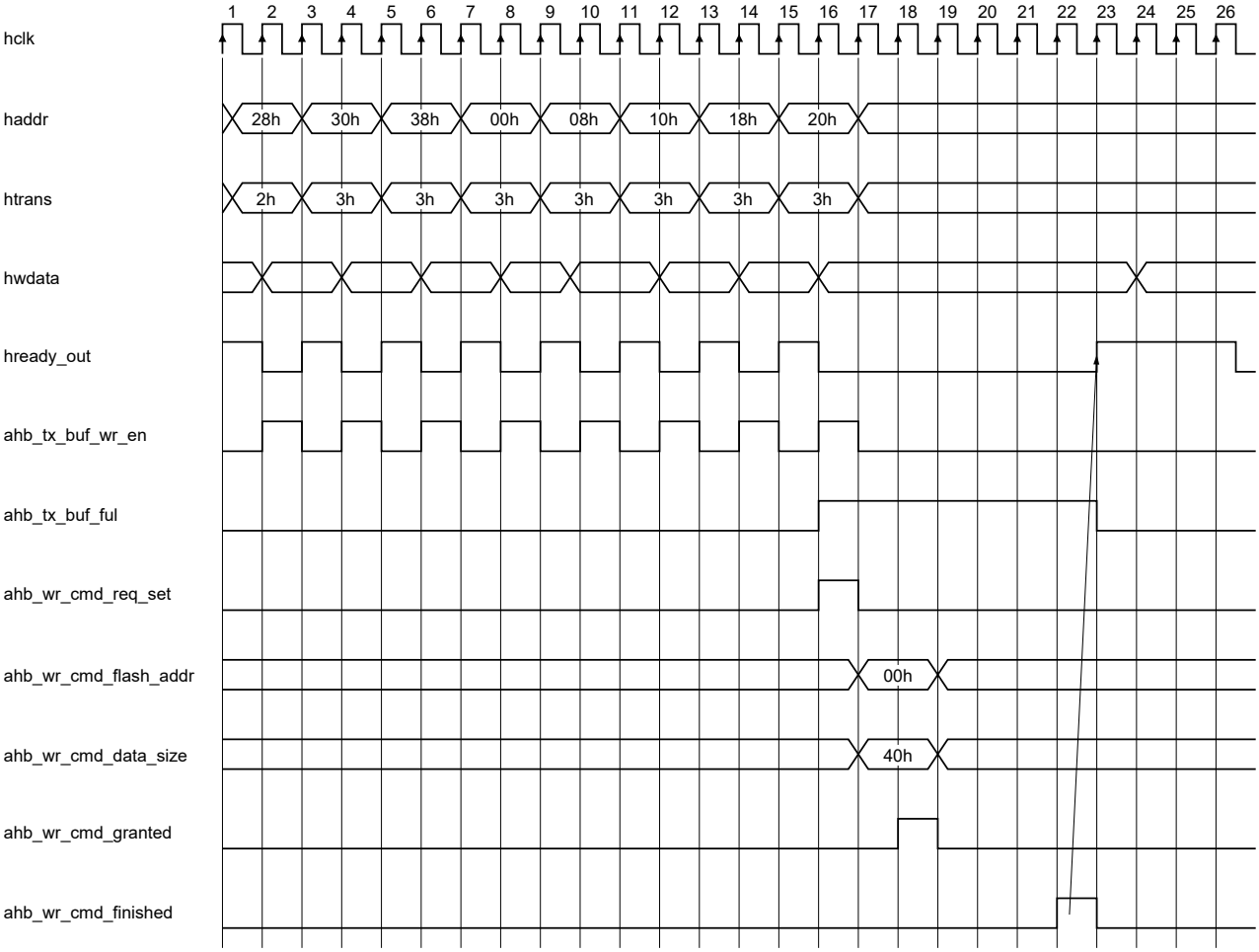


Figure 212. AHB write access (WRAP8, 64-bit, non-bufferable)

NOTE

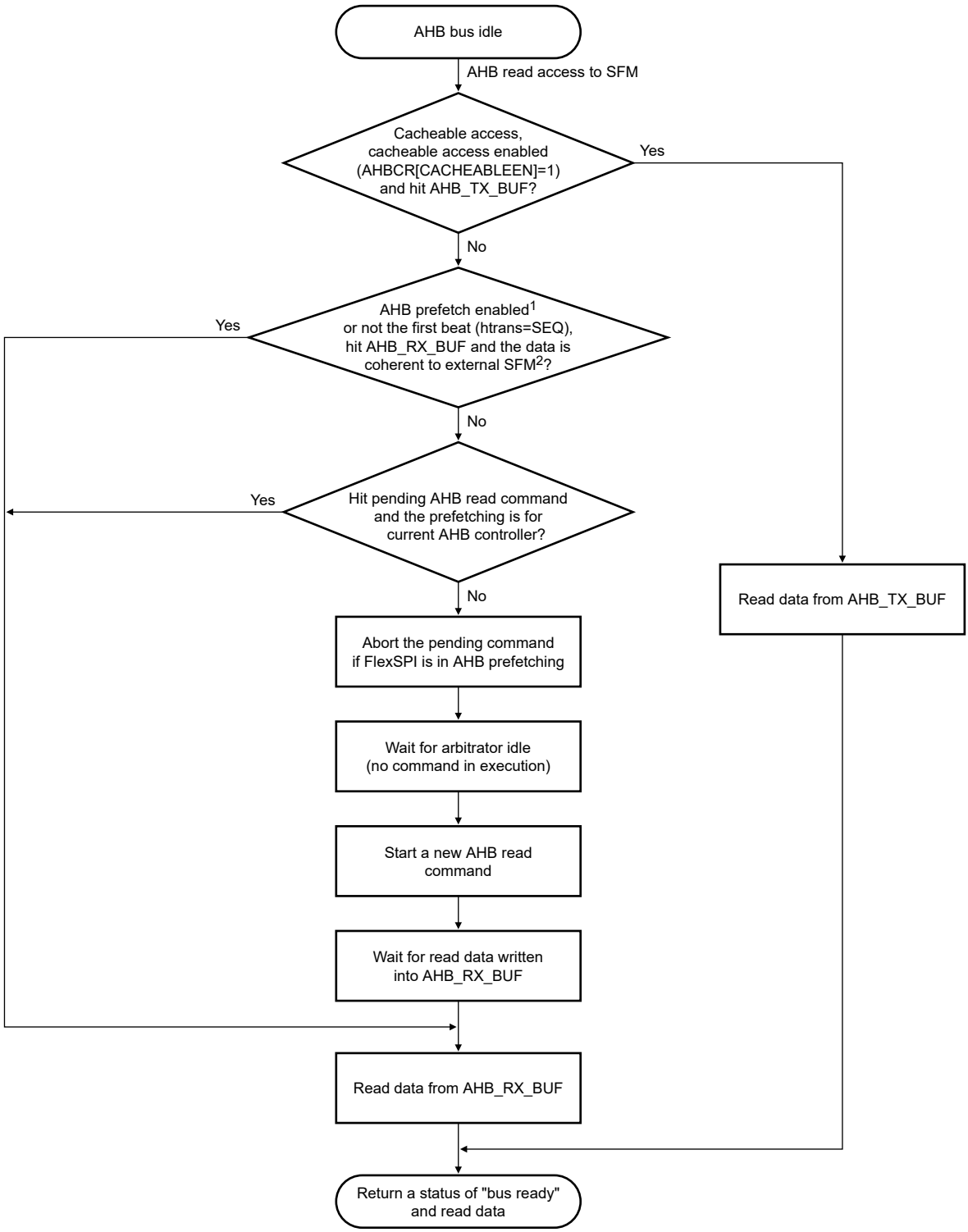
If the burst data size (in bytes) is greater than the AHB transmit buffer size, wrapper burst is not supported. For example, if AHB_TX_BUF is 64 bytes, AHB WRAP16 (16 × 64 bits = 128 bytes) write burst access is not supported.

31.3.11.2 AHB read access to flash memory

For AHB read access to flash memory, FlexSPI checks the burst access type and register setting to determine whether to access these locations:

- The AHB transmit buffer
- The AHB receive buffer
- A pending AHB read command

If all of these options are missed, FlexSPI triggers a new AHB read command to fetch data from flash memory. Figure 213 shows the hardware operation in response to AHB read access to flash memory.



Notes:
1) AHB prefetch is enabled when both AHBCR[PREFETCHEN]=1 and AHBRXBUFxCR0[PREFETCHEN]=1 (x is the AHB RX buffer ID for the current AHB controller) or OTFAD is enabled.
2) The AHB RX buffer holds the data read from flash memory. This data may become incoherent if AHB writes to the same flash-memory address.

Figure 213. Hardware operation in response to AHB read access to flash memory

Table 207 describes the details of a triggered AHB read command.

Table 207. Triggered AHB read command

Item	Description
Flash memory access start address and data size	Determined by AHB address, burst type, and burst size. FlexSPI fetches read data from the start address for the current burst or beat. See Table 208 for details.
Flash memory chip select	FlexSPI uses the settings of the flash memory access start address and flash memory size to determine the chip select.
Flash memory command sequence index	Determined by FLSHxCR2[AWRSEQID].
Flash memory command sequence number	Determined by FLSHxCR2[AWRSEQNUM]. If AWRSEQNUM is not zero, multiple flash memory access command sequences are triggered every time for an AHB read command. The sequences indexed from AWRSEQID to (AWRSEQID + AWRSEQNUM) in LUT are executed sequentially.
Flash memory access mode (Individual/ Parallel)	Determined by AHBCR[APAREN].

NOTE

- FlexSPI automatically determines which ARDSEQNUM and ARDSEQID fields that the flash device uses as sequence ID for chip selection. See MCR2[SAMEDEVICEEN].
- FlexSPI determines which AHB receive buffer to use for the current AHB read access by controller ID. For details about the AHB receive buffer ID and AHB controller ID mapping, see AHB receive buffer management.
- You cannot allocate the size of the AHB receive buffer to be less than the AHB Burst size. This allocation results in unknown behavior.

Table 208. AHB read command flash memory start address and data size

Prefetch enable	Cross flash boundary	Burst type	Flash memory start address [26:0]	Data size
0	Never cross flash boundary, because AHB burst never crosses 1 KB boundary.	SINGLE, INCR4, INCR8, INCR16	hbeat_start_address	(hburst_end_address-hbeat_start_address)
		INCR	hbeat_start_address	byte size of current beat For INCR burst with prefetch disabled, each beat is handled same as SINGLE burst.
		WRAP4, WRAP8, WRAP16	hburst_start_address	(hburst_end_address-hburst_start_address)
1	No	INCR, SINGLE, INCR4, INCR8, INCR16	hbeat_start_address	ahb_rx_buf_sz
	No	WRAP4, WRAP8, WRAP16	hburst_start_address	ahb_rx_buf_sz

Table continues on the next page...

Table 208. AHB read command flash memory start address and data size (continued)

Prefetch enable	Cross flash boundary	Burst type	Flash memory start address [26:0]	Data size
	Yes	INCR, SINGLE, INCR4, INCR8, INCR16	hbeat_start_address	(flash_top_address-hbeat_start_address)
	Yes	WRAP4, WRAP8, WRAP16	hburst_start_address	(flash_top_address-hburst_start_address)

NOTE

- hbeat_start_address is HADDR input from the AHB controller for current beat.
- hburst_start_address (for example 00h) is the lowest address for current burst. hburst_end_address (for example 10h) is the highest address in current burst plus 1. For example, WRAP4 burst with HADDR 8h, Ch, 0h, 4h.
- ahb_rx_buf_sz is the buffer size in bytes of AHB receive buffer that the current AHB controller uses.
- flash_top_address is the top address of currently accessed flash memory.

31.3.11.3 AHB receive buffer management

There are 8 buffers (Buffer 0–7) in the AHB receive buffer. These buffers are transparent to AHB controllers. FlexSPI fetches flash memory data and returns it on the AHB bus automatically. No status register polling is needed for AHB read access to Serial Flash Memory (SFM).

See the chip-specific section for AHB buffer size.Use AHBRXBUF0CR0[BUFSZ]–AHBRXBUFCR7[BUFSZ] to configure the buffer size of each AHB receive buffer. The buffer size for Buffer 0 to Buffer 7 could be set to 0. If the buffer size is set to 0, FlexSPI ignores the related setting of AHBRXBUFxCR0[MSTRID]. Buffer 7 is used for all AHB controllers not assigned to Buffer 0–6. FlexSPI ignores the Buffer 7 size setting field (AHBRXBUF7CR0[BUFSZ]). Its buffer size is: (AHB receive buffer total size) - (the sum of the sizes of Buffer 0–6).

When an AHB read access to serial flash memory occurs, FlexSPI determines which buffer to use via this method:

1. If the controller ID equals AHBRXBUF0CR0[MSTRID] and AHBRXBUF0CR0[BUFSZ] is not zero, Buffer 0 is used.
2. If the controller ID equals AHBRXBUF1CR0[MSTRID] and AHBRXBUF1CR0[BUFSZ] is not zero, Buffer 1 is used.
3. If the controller ID equals AHBRXBUF2CR0[MSTRID] and AHBRXBUF2CR0[BUFSZ] is not zero, Buffer 2 is used.
4. If the controller ID equals AHBRXBUF3CR0[MSTRID] and AHBRXBUF3CR0[BUFSZ] is not zero, Buffer 3 is used.
5. If the controller ID equals AHBRXBUF4CR0[MSTRID] and AHBRXBUF4CR0[BUFSZ] is not zero, Buffer 4 is used.
6. If the controller ID equals AHBRXBUF5CR0[MSTRID] and AHBRXBUF5CR0[BUFSZ] is not zero, Buffer 5 is used.
7. If the controller ID equals AHBRXBUF6CR0[MSTRID] and AHBRXBUF6CR0[BUFSZ] is not zero, Buffer 6 is used.
8. If all the above cases are not met, Buffer 7 is used.

NOTE

- Software must ensure that the size of each buffer is not less than the maximum burst size of an AHB read access from the AHB controller using this buffer. Otherwise, the behavior is undefined. The minimum buffer size request depends on the chip implementation. For example, if the capability of the chip to send out a maximum burst is 64 bytes, the minimum AHB receive buffer is 64 bytes.
- Assigning multiple buffers to a single AHB controller is not supported, unless the prefetch buffer enhancement feature is being used.
- When AHB read prefetch is enabled (AHBCR[PREFETCHEN] = 1), the size of the buffer determines the prefetch data size. If it does not cross the flash boundary, FlexSPI fetches data from the external flash device with that buffer size.
- The priority setting of AHB controller (AHBRXBUFxCR0[PRIORITY] = 1) is used only for suspending control of AHB prefetching. See Command abort and suspend.

31.3.12 Command arbitration

There are four flash memory access command sources:

1. AHB command, that AHB write access to serial flash memory space triggers
2. AHB command, that AHB read access to serial flash memory space triggers
3. IP command, that writing IPCMD[TRG] triggers
4. Suspended command, an AHB read prefetch sequence which is suspended

An AHB bus access never triggers a write command and a read command at the same time.

The AHB prefetch sequence is an AHB command sequence triggered via AHB read access when AHB prefetch is enabled. After all read data is fetched for the current AHB read burst, FlexSPI prefetches more data to reduce read latency for the next AHB read access. An AHB command for a read operation is never aborted while fetching read data for the current AHB read burst. However, a new IP command or AHB command request while prefetching data (not for the current read burst) can abort an AHB read command.

When the arbitrator is idle (STS0[ARBIDLE] = 1), the granted priority of these command sources is:

1. AHB command (read or write)
2. IP command
3. Suspended command

NOTE

When the arbitrator is idle and there is no AHB or IP command request, a suspended command is not granted immediately. The arbitrator waits MCR2[RESUMEWAIT] AHB clock cycle idle states before resuming the suspended command to avoid AHB prefetch sequence being resumed and suspended frequently.

If the arbitrator is busy executing AHB or IP commands (not suspended commands), no new command requests are granted. If the grant times out, an AHB or IP command granted error occurs.

If a new AHB or IP command request comes while arbitrator is executing AHB read prefetch sequence, AHB read prefetch sequence is aborted (but not immediately). Arbitrator grants the AHB or IP command request after AHB read prefetch sequence is aborted on the FlexSPI interface and saved all internal data pointers.

31.3.12.1 Command abort and suspend

Table 209 describes the command abort and suspend mechanism.

Table 209. Command abort and suspend

Action	Description
Aborting a command	As mentioned above, if a new AHB or IP command request arrives, an AHB read prefetch sequence can be aborted.
Suspending a command	When an AHB read prefetch sequence is aborted on the FlexSPI interface, FlexSPI saves this suspended sequence in the cases listed below. FlexSPI resumes this sequence after the arbitrator has been idle for enough time. • There is no valid suspended command (AHBSPNDSTS[ACTIVE] = 0). This condition can occur when no command is suspended or when the suspended command is resumed. • The priority of aborted AHB read prefetch sequence is higher than the active suspend sequence. **NOTE** FlexSPI ignores and never resumes the original suspended sequence.
Activating a suspended command	The suspended command (internal status) becomes active when an AHB prefetch command is aborted and suspended. It becomes inactive in the following cases: • The arbitrator resumes the suspended command. • The AHB controller triggers a new AHB read command request using the same AHB receive buffer (buffer ID).

Figure 214 shows an example of the flow of command abort, suspend, or resume.

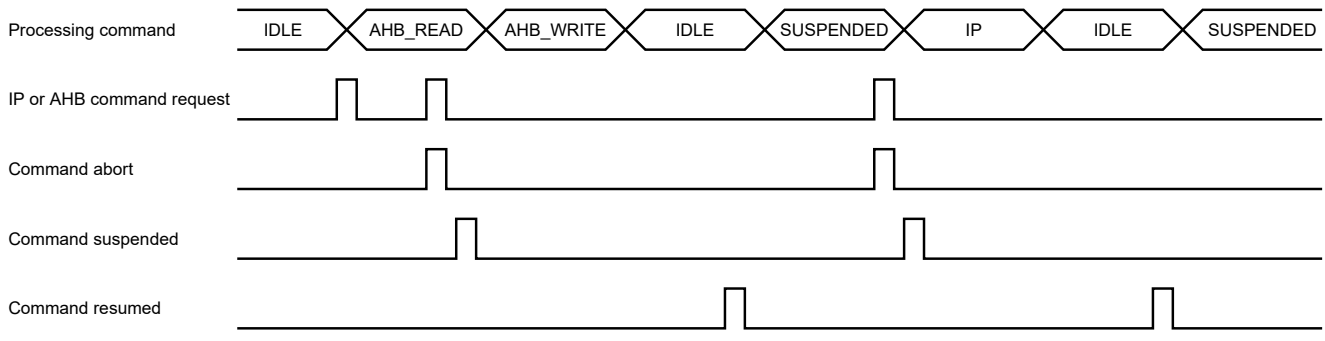


Figure 214. Command instruction execution on FlexSPI interface

NOTE

- The AHB prefetch sequence is aborted, suspended, or resumed twice in Figure 214. A new AHB write command request aborts the sequence the first time, and a new IP command request aborts the sequence the second time.
- The suspended sequence is resumed after a wait time of idle state. If there are frequent IP or AHB command requests and the wait time is not set reasonably, the suspend and resume activities may be triggered frequently.

31.3.13 SCLK stop

FlexSPI stops SCLK output toggling:

- When programming data is not ready for a programming command sequence.
- When an internal FIFO has no space to receive data for reading command sequence.

Certain devices may not support the stopping of SCLK during a command sequence (chip select is valid). SCLK stopping can be avoided via the methods described below.

- For flash memory reading that an IP command triggers:
  - Never trigger a read command with data size larger than IP receive FIFO size.
  - Internal asynchronous FIFO for flash memory reading should never be full.

    FlexSPI pops 64 bits of data per AHB clock cycle from this FIFO, and receives data from FlexSPI interface on the serial root clock. The flash memory access mode (Single, Dual, Quad, or Octal mode and Individual or Parallel mode) determines the receiving speed. For example, in Octal mode and Parallel mode, FlexSPI receives 16 bits per serial root clock cycle. If the AHB clock frequency is faster than 1/4 of the serial root clock, this asynchronous FIFO is never full.
- For flash memory programming that an IP command triggers:
  - Never trigger a program command with data size larger than the IP transmit FIFO size. Write all programming data into the IP transmit FIFO before triggering the IP command.
  - The internal asynchronous FIFO for flash memory programming must never be empty.

    FlexSPI fetches 64 bits of programming data per AHB clock cycle into this FIFO, and transmits data to the FlexSPI interface on the serial root clock. The flash memory access mode (Single, Dual, Quad, or Octal mode and Individual or Parallel mode) determines the transmitting speed. For example, in Octal mode and Parallel mode, FlexSPI transmits 16 bits per serial root clock cycle. If the AHB clock frequency is faster than 1/4 of the serial root clock, this asynchronous FIFO is never empty.
- For flash memory reading or programming that an AHB command triggers:
  - The internal asynchronous FIFO for flash memory reading and programming must never be full or empty. The frequency ratio limitation is same as it is for flash memory reading or flash memory programming that an IP command triggers.

NOTE

- FlexSPI never triggers an AHB read command with data size larger than the internal AHB receive buffer size.
- FlexSPI never triggers an AHB program command with data size larger than the internal AHB transmit buffer size.
- All programming data is buffered into the AHB transmit buffer before triggering an AHB program command in FlexSPI.

31.3.14 FlexSPI output timing

31.3.14.1 Output timing between data and SCLK

This section describes the output timing relationship of data (on A_DATA and B_DATA) and SCLK. There are three cases.

- **SDR instruction in SDR sequence**

  An SDR sequence contains only SDR instructions. In this case, all data bits last one serial root clock cycle on the FlexSPI interface. Figure 215 shows the relationship between the serial root clock, data, and SCLK:

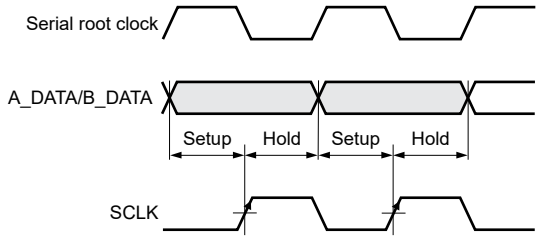


Figure 215. SDR instruction in SDR sequence

- **SDR instructions in a DDR sequence**

  A DDR sequence is a flash memory access command sequence that contains DDR instructions other than DUMMY_DDR. It may contain SDR instructions. If there are SDR instructions in a DDR sequence, all data bits last for two serial root clock cycles on the FlexSPI interface. Figure 216 shows the relationships between the serial root clock, data, and SCLK.

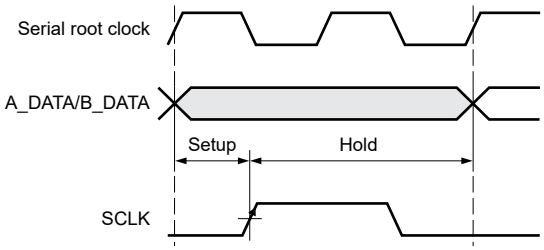


Figure 216. SDR instruction in DDR sequence

- **DDR instructions in a DDR sequence**

  For DDR instructions in a DDR sequence, all data bits last one serial root clock cycle on the FlexSPI interface. Figure 217 shows the relationships between the serial root clock, data, and SCLK.

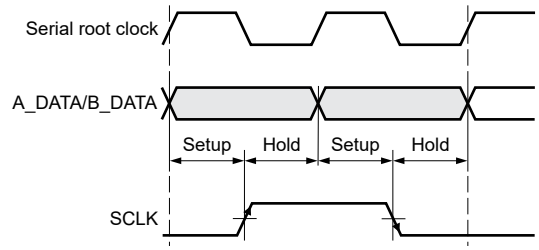


Figure 217. DDR instruction in DDR sequence

31.3.14.2 Output timing between chip select and SCLK

This section describes the output timing relationship between chip select (on A_SS0_B , A_SS1_B, B_SS0_B, and B_SS1_B) and SCLK. The timing relationship for the SDR sequence differs a little from the timing relationship for the DDR sequence.

- **Chip select timing in SDR sequence**

  For an SDR sequence, the delay from chip select assertion to the SCLK rising edge is (FLSHxCR1[TCSS] + 0.5) cycles of the serial root clock. The delay from SCLK falling edge to chip select deassertion is FLSHxCR1[TCSH] cycles of serial root clock. Figure 218 shows the timing relationship between the chip select and SCLK.

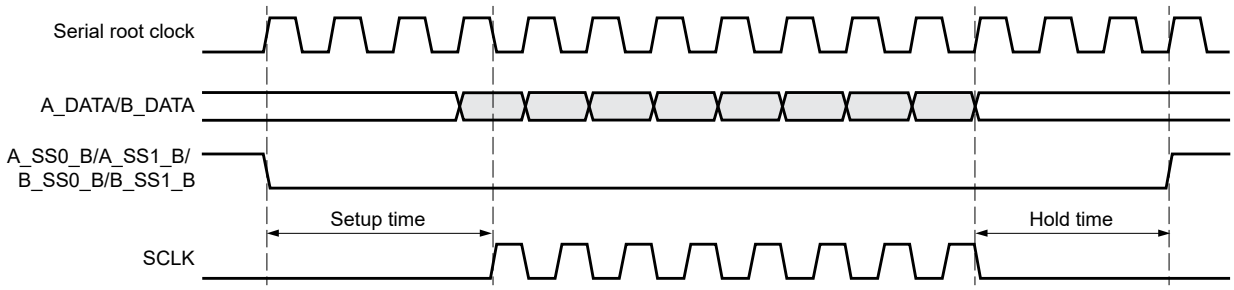


Figure 218. Chip select output timing for SDR sequence

- **Chip select timing in DDR sequence**

  For a DDR sequence, the delay from chip select assertion to the SCLK rising edge is **(FLSHxCR1[TCSS] + 0.5)** cycles of the serial root clock. The delay from the SCLK falling edge to chip select deassertion is **(FLSHxCR1[TCSH] + 0.5)** cycles of serial root clock.

  NOTE

  When AHB receive prefetch is enabled, and prefetch can be aborted, configure TCSH to 1 or greater. Configuring TCSH in this way guarantees a positive chip select hold time after the SCLK falling edge.

  Figure 219 shows the timing relationship between chip select and SCLK.

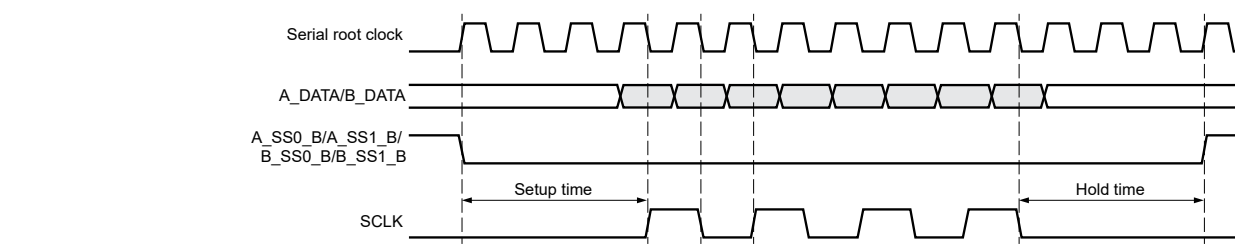


Figure 219. Chip select output timing for DDR sequence

For devices such as an FPGA device, a minimum chip select negation time (time between two chip select assertions) may be required. If the value of FLSHxCR1[CSINTERVAL] is nonzero, FlexSPI ensures a delay between chip select valid assertions. The delay time is CSINTERVAL × 256 cycles of serial root clock for an SDR or DDR sequence. If the external device has no limitation, configure the value of this field to zero. Figure 220 shows the timing of the chip select interval.

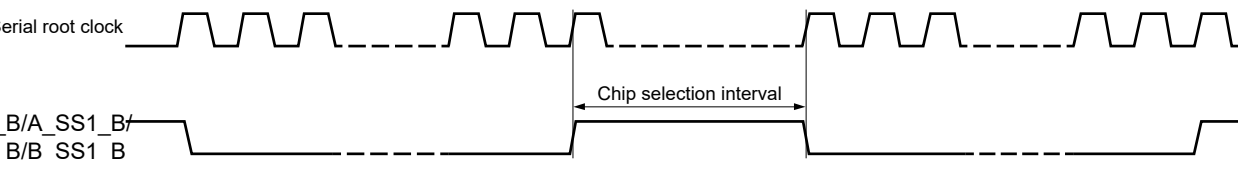


Figure 220. Chip select valid interval

31.3.15 FlexSPI input timing

This section describes the input timing of FlexSPI.

31.3.15.1 Receiving block

This section describes the receiving block in FlexSPI.

FlexSPI decodes instruction code to determine SDR mode or DDR mode. It also determines whether both the rising and falling edges are used for flash data sampling. FlexSPI samples the data line for learn instructions and read instructions.

MCR0[RXCLKSRC] selects the clock source from these four sources for the sampling clock:

- Internal dummy read strobe and looped back internally
- Internal dummy read strobe and looped back from DQS pad
- SCK output and looped back from SCK pad
- Flash-memory-provided read strobe

For details about DLL configuration, see DLL configuration for sampling.

Figure 221 shows the sampling block in FlexSPI.

FlexSPI uses both rising and falling edges to sample data from flash_A and flash_B. Two muxes with four clock source inputs generate the signals ipp_ind_io_fa, ipp_ind_io_fb. See Receive clock source features. ipp_ind_io_fa_int and ipp_ind_io_fb_int are reversed from ipp_ind_io_fa and ipp_ind_io_fb. Both sampling flip-flops are used when FlexSPI is in DDR mode. Only the falling edge sampling flip-flop is used in SDR mode.

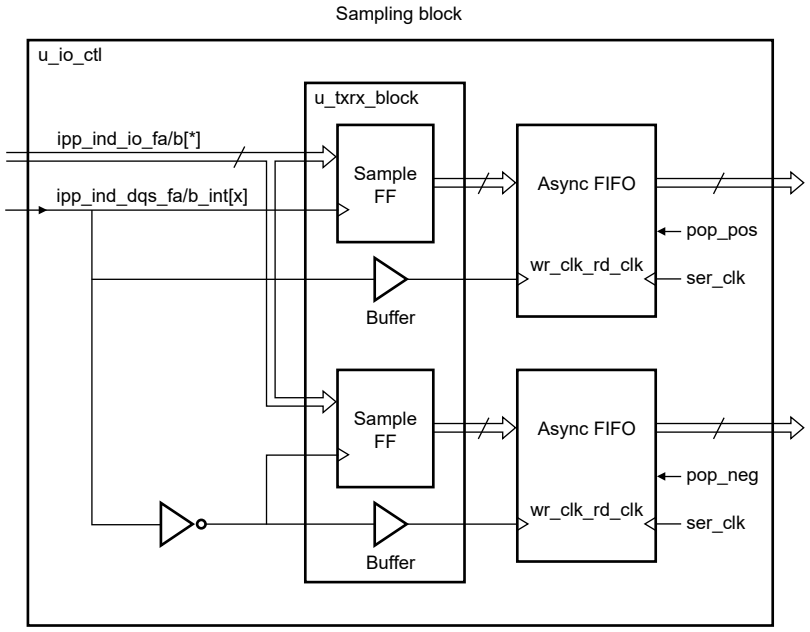


Figure 221. Sampling block in FlexSPI

There are 1 clock phases for sampling data. An internal delay chain delays the clock phases.

FlexSPI does not support read operations with a zero-turnaround cycle for direction switching on I/O pins. This read operation often occurs in QuadSPI configuration register reads. In Figure 222 and Figure 223, SIO[3:0] is an output from FlexSPI in the CMD phase, and is an input to FlexSPI in the DATA phase. No time remains for FlexSPI to switch the direction. This kind of read is not supported. FlexSPI and flash memory drive the I/O pins at the same time, which may cause a problem. To avoid this behavior on I/O pins, using SPI mode or reading with dummy cycles is required.

NOTE

The cycle numbers are not accurate in the examples below. See the device spec for accurate cycle numbers.

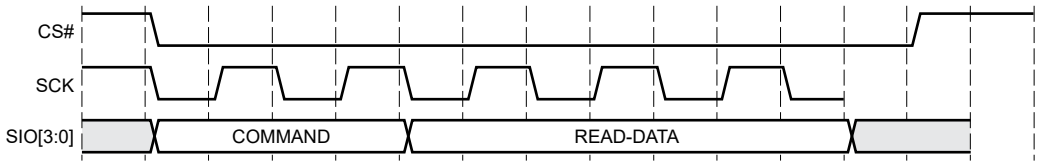


Figure 222. Unsupported read behavior

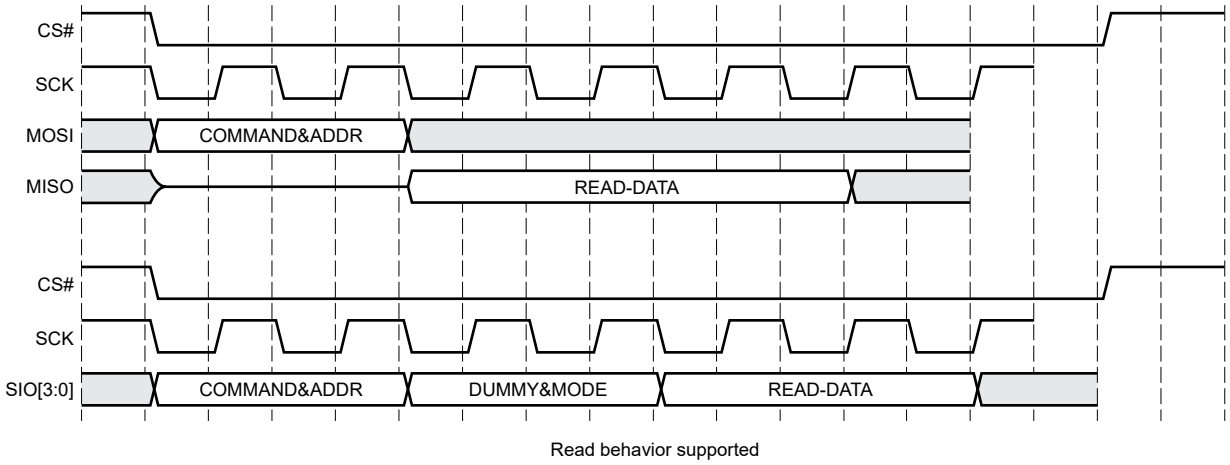


Figure 223. Supported read behavior

31.3.15.2 Receive clock source features

FlexSPI uses a read strobe to sample incoming data. There are several options for generating the read clock strobe that MCR0[RXCLKSRC] selects. Using the receive clock source method has implications for the board connections and can affect the maximum frequency that the interface supports. See the device data sheet for details on the maximum frequency supported for each RXCLKSRC option.

Table 210. RXCLKSRC options

MCR0[RXCLKSRC]	Description	Board connection	Max Frequency
0	Internal dummy read strobe and internal loopback	DQS pin is not used. The DQS pin can be configured for an alternate signal function.	Lowest
1	Internal dummy read strobe and loopback from DQS pad	FlexSPI uses the DQS pin, and it must be configured for its FlexSPI function. The internally generated read strobe is sent to the DQS pin and is sampled at the pin to match more closely the data pin timings. The DQS pin is typically left floating. External capacitance can be added to adjust timing.	Medium
3	Flash-memory-provided read strobe	DQS pin is connected to the DQS or RWDS signal that the memory provides.	Highest

31.3.15.3 Input timing for sampling with dummy read strobe

This section describes the input timing when sampling with the internal dummy read strobe (MCR0[RXCLKSRC] = 0 or 1). The timing for sampling with an internal dummy read strobe loopback is very similar to the timing for loopback from pad. Sampling with a dummy read strobe loopback from the DQS pad can achieve a higher read frequency. It compensates for the delay of the SCLK output path and data pin input path.

The input timing is different for SDR mode and DDR mode.

- **Input timing for sampling with dummy read strobe in SDR mode**

  For SDR read or learn instructions, FlexSPI samples input data pins with the falling edge of the dummy read strobe. Figure 224 shows the input timing for sampling with dummy read strobe in SDR mode.

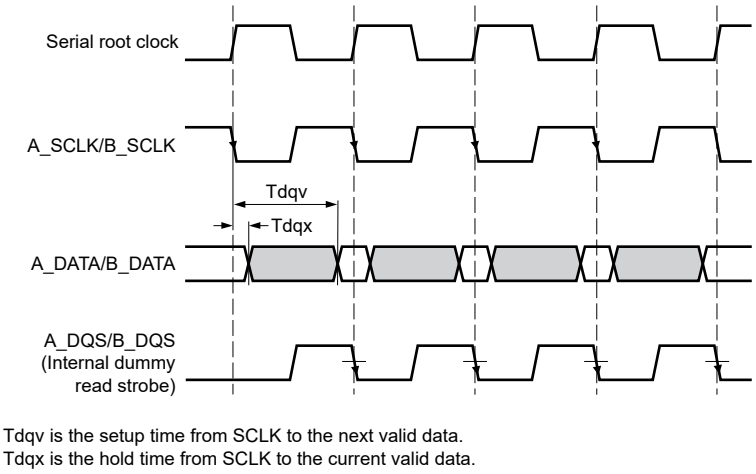


Figure 224. Input timing for sampling with dummy read strobe in SDR mode

- **Input timing for sampling with dummy read strobe in DDR mode**

  For DDR read or learn instructions, FlexSPI samples input data pins on rising and falling edges of the dummy read strobe. Figure 225 shows the input timing for sampling with dummy read strobe in DDR mode.

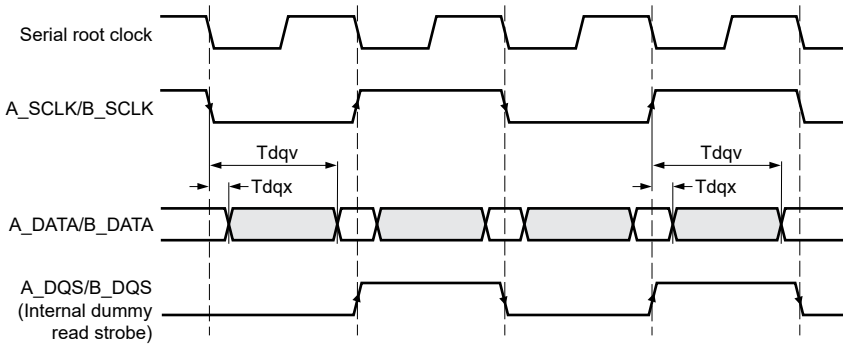


Figure 225. Input timing for sampling with dummy read strobe in DDR mode

31.3.15.4 Input timing for sampling with flash-memory-provided read strobe

This section describes the input timing when sampling with flash-memory-provided read strobe (MCR0[RXCLKSRC] = 3). The input timing is different for SDR mode and DDR mode.

NOTE

There are no known devices that provide a read strobe and support SDR mode operation.

- **Flash-memory-provided read strobe with SCLK**

  Certain flash devices provide both read data and read strobes with SCLK. Then the read strobe edge is aligned with the read data change. The FlexSPI controller delays the read strobe for one half cycle of the serial root clock (with DLL), then samples read data with the delayed strobe. See DLL configuration for sampling. Figure 226 and Figure 227 show the input timing for sampling with flash memory read strobe in SDR mode and DDR mode.

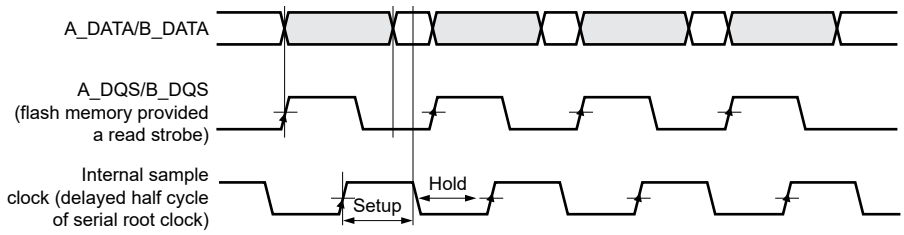


Figure 226. Input timing 2 for flash-memory-provided read strobe in SDR mode

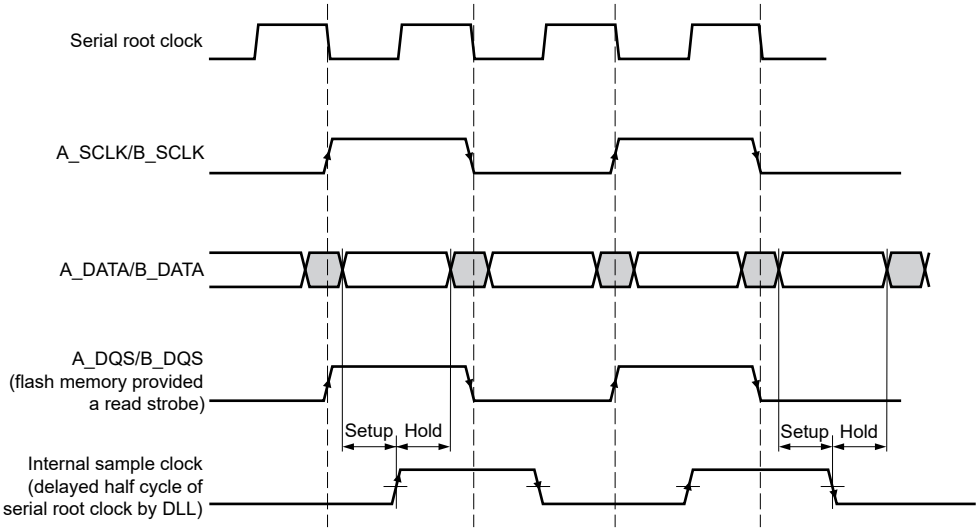


Figure 227. Input timing 2 for flash-memory-provided read strobe in DDR mode

31.3.15.5 DLL configuration for sampling

For the four sampling clock sources described previously, the input timing differs depending on the DLLxCR register configuration, according to the sampling clock source mode. The DLL is a delay-line chain that can be set to a fixed number of delay cells. It can also be auto-adjusted to lock on a certain phase delay to the reference clock.

- DLLxCR must be set to 0000_0100h (one fixed delay cell in DLL delay chain) in the following cases:
  - Sampling data with dummy read strobe looped back internally (MCR0[RXCLKSRC] = 0h)
  - Sampling data with dummy read strobe looped back from DQS pad (MCR0[RXCLKSRC] = 1h)
- When data is sampled with a flash-memory-provided read strobe (MCR0[RXCLKSRC] = 3h) and flash memory provides the read strobe with SCLK, DLL must be configured as follows (when serial root clock is over 100 Mhz. These settings ensure a lock of the reference clock (serial root clock).
  - DLLxCR[SLVDLYTARGET] is up to flash parameters used.
  - DLLxCR[REFPHASEGAP] = 2h

- DLLxCR[DLLEN] = 1h
- DLLxCR[OVRDEN] = 0h
- Other fields in DLLxCR must be kept at their reset values (all zeroes).

After the register settings above are completed, set the DLL with following steps.

1. Reset the DLL(DLLxCR[DLLRESET] or MCR0[SWRESET]).
2. Wait for STS2[xREFLOCK] and STS2[xSLVLOCK] being asserted.
3. Wait for equal or more than 512 serial root clock cycles.
4. Start read or write operation.

NOTE

If the serial root clock is slower than 100 MHz, DLL cannot lock on a half cycle of serial root clock. The delay cell number is limited in the delay chain. DLL must be configured as follows instead:

- DLLxCR[OVRDEN] = 1
- DLLxCR[OVRDVAL] = N

  Each delay cell in DLL is about 75 ps to 225 ps. The delay of DLL delay chain is (N × Delay_cell_delay). N is set based on the maximum DDR frequency (N ≤ equation). This value is calculated based on the MAX_DDR_FREQ parameter. N = 21. This value is a recommended value. If a failure occurs, the value might require adjustment in a real application.

  You can calculate the delay of one delay cell and obtain the value of N. If the serial root clock is lower than 100 MHz, the delay chain can't lock with DLLxCR[SLVDLYTARGET] = 0xF (half cycle of root clock delay). You need to set DLLxCR[SLVDLYTARGET] as 0xE (15/32 cycle delay) or less. When DLL is locked (read STS2[AREFLOCK] and STS2[ASLVLOCK]), the delay cell number = STS2[ASLVSEL] +1 and the delay value (one delay cell) = (total delay)/(delay cell number). Based on delay value, the N can be easily calculated.
- Other fields in DLLxCR must be kept at their reset values (all zeroes).

31.3.16 Execute-In-Place (XIP) enhanced mode

FlexSPI always supports XIP, regardless of whether external devices provide XIP enhanced mode. To support XIP, put program codes on an external device, then read or execute these codes directly via AHB read access to serial flash memory space. There is no configuration or status polling needed during AHB read access to the memory of the external device. The AHB receive buffer is fully transparent to software.

Certain devices provide XIP enhanced mode to improve code execution. In this mode, the command code is only sent in the first read instruction. It saves many cycles for command instructions and improves code execution. Devices enter or exit XIP enhanced mode via a special sequence that external devices specify. See the external device data sheet for more details.

Normally, a device enters XIP enhanced mode via this sequence:

1. Enable XIP enhanced mode in external flash device via IP command.
2. Send the first read sequence to external flash device with correct mode bits. Command code is included in this read sequence.
3. Send the subsequent read sequences to external flash device with correct mode bits. Command code is not included in these read sequences. Mode bits must be sent according to the flash specified; otherwise, the flash exits XIP enhanced mode.

The instruction JMP_ON_CS in FlexSPI must be used to support XIP enhanced mode in external devices. This instruction is only allowed in the AHB read command; otherwise, the instruction generates an IPCMDERR or AHBCDMERR interrupt (when the interrupt is enabled). When external devices do not support XIP enhanced mode, JMP_ON_CS must never be used. To support XIP enhanced mode:

- The first instruction in the read sequence must be the command instruction.
- The last valid instruction must be JMP_ON_CS (with operand 1h).

For the first AHB read command triggered, FlexSPI executes the instructions from the instruction pointer 0 in the sequence (which is the command instruction). After this sequence is executed, FlexSPI saves the operand in the JMP_ON_CS instruction as the start pointer for the next command to current device internally. For the next AHB read command triggered, FlexSPI executes from the instruction pointer 1h, bypassing the command instruction.

After XIP enhanced mode is started, it is possible to access flash via other commands like program commands. You must, however, use a special sequence to exit enhanced XIP mode before you can run a different command. Figure 228 indicates XIP operation with Flash XIP enhanced mode.

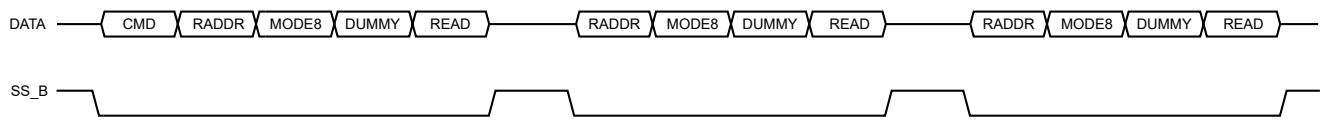


Figure 228. XIP Enhanced Mode Operation

31.4 External signals

This section provides the external signal information for the FlexSPI module.

Table 211. External Signal List

Signal name	Function	Direction	Description
A_SS0_B	Peripheral Chip Select Flash A1	O	Chip select for the serial flash device A1
A_SS1_B	Peripheral Chip Select Flash A2	O	Chip select for the serial flash device A2
B_SS0_B	Peripheral Chip Select Flash B1	O	Chip select for the serial flash device B1
B_SS1_B	Peripheral Chip Select Flash B2	O	Chip select for the serial flash device B2
A_SCLK	Serial Clock Flash A	O	Serial clock output to the serial flash device A. This clock runs at half the frequency of serial clock root in DDR mode, and at the same frequency as serial clock root in SDR mode. Clock output toggles during the entire flash memory access sequence.
B_SCLK	Serial Clock Flash B	O	Serial clock output to the serial flash device B. This clock runs at half the frequency of serial clock root in DDR mode, and at the same frequency as serial clock root in SDR mode. Clock output toggles during the entire flash memory access sequence.

Table continues on the next page...

Table 211. External Signal List (continued)

Signal name	Function	Direction	Description
			NOTE B_SCLK may be used as the differential clock output of A_SCLK by writing 1 to MCR2[SCKBDIFFOPT].
A_DATAn	Serial I/O Flash A	I/O	Data I/O lines to and from the serial flash device A
B_DATAn	Serial I/O Flash B	I/O	Data I/O lines to and from the serial flash device B
A_DQS	Data Strobe Signal Flash A	I/O	Data strobe signal for port A. This signal has three functions: • Driven with read strobe by external device. Some flash devices provide the read strobe signal together with read data. In this case, if the external device drives this pad only when reading flash memory data, this pad may require a **pull-down** resistor. • Driven with latency information by external device. Some devices use this pin to indicate the dummy cycles needed (before program or read data transfer) such as HyperRAM or HyperFlash. • Loopback dummy read strobe. The FlexSPI controller provides internal dummy read strobe for flash memory read data. Higher read frequency can be achieved by looping back this dummy read strobe from the pad. This pin can be floated or have some capacitive load added at the board level to compensate for load on DATA or SCLK pins.
B_DQS	Data Strobe Signal Flash B	I/O	Similar to A_DQS

31.5 Initialization

The FlexSPI controller initialization sequence is:

1. Enable controller clocks (AHB clock, IP bus clock, or serial root clock) at system level.
2. To enter module stop mode, write 1 to MCR0[MDIS].
3. Configure module control registers: MCR0, MCR1, and MCR2. Do not change MCR0[MDIS].
4. When AHB commands are used, configure AHB Bus Control (AHBCR) and AHB Receive Buffer Control (AHBRXBUFxCR0) registers.
5. Configure flash control registers (FLSHxCR0, FLSHxCR1, or FLSHxCR2) according to external device type.
6. Configure DLL control register (DLLxCR) according to sample clock source selection.

7. To exit module stop mode, write 0 to MCR0[MDIS].
8. Configure LUT as needed for AHB command or IP command.
9. Optionally, to reset controller, write 1 to MCR0[SWRESET].

External devices must be configured via IP command normally after initialization. For example, the WRITE STATUS command performs device configuration for most serial NOR flash memory.

31.6 Application information

This section describes applications that FlexSPI supports.

31.6.1 Application on serial NOR flash device

This section provides the example LUT instruction sequences for serial NOR flash device (Cypress Flash S25FS128S).

31.6.1.1 Write enable command

Table 212 shows the WRITE ENABLE command sequence.

Table 212. WRITE ENABLE command

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h	06h	Command name: WREN
1-7	STOP (00h)	0h	00h	

31.6.1.2 Write registers command

Table 213 shows the Write Registers command sequence.

Table 213. Write Registers command

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	01h	Command name: WRR
1	WRITE_SDR	0h (Single)	1h or 2h	Data size is 1 byte or 2 bytes. Byte 0 is write data for Status Register 1; Byte 1 is write data for Configuration Register 1. IPCR1[IDATSZ] can override this value.
2-7	STOP (00h)	0h	00h	

31.6.1.3 Page Program command

Table 214 shows Page Program command sequence.

Table 214. PAGE PROGRAM command (Cypress serial NOR flash)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	02h or 12h	Command name: PP or 4PP

Table continues on the next page...

Table 214. PAGE PROGRAM command (Cypress serial NOR flash) (continued)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
				PP is 3-byte address mode. 4PP is 4-byte address mode.
1	ADDR_SDR	0h (Single)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	WRITE_SDR	0h (Single)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default programming data size. This value is ignored for AHB command.
3-7	STOP (00h)	0h	00h	

Table 215 shows Page Program command sequence (QPI mode).

Table 215. PAGE PROGRAM (QPI mode) command (Cypress serial NOR flash)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	2h (Quad)	02h or 12h	Command name: PP or 4PP PP is 3-byte address mode. 4PP is 4-byte address mode.
1	ADDR_SDR	2h (Quad)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	WRITE_SDR	2h (Quad)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default programming data size. This value is ignored for AHB command.
3-7	STOP (00h)	0h	00h	

31.6.1.4 Read Status 1 command

Table 216 shows READ STATUS 1 command sequence.

Table 216. READ STATUS 1 command

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	05h	Command name: RDSR1
1	READ_SDR	0h (Single)	1h	1 byte for Status register 1
2-7	STOP (00h)	0h	00h	

31.6.1.5 Read command

Table 217 shows READ command sequence.

Table 217. READ command

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	03h or 13h	Command name: READ or 4READ READ is 3-byte address mode. 4READ is 4-byte address mode.
1	ADDR_SDR	0h (Single)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	READ_SDR	0h (Single)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size. This value is ignored for AHB command.
3-7	STOP (00h)	0h	00h	

31.6.1.6 Fast Read command

Table 218 shows Fast Read command sequence. This table shows Cypress SPI Configuration Register bits CR2V[7] = 0, CR2V[3:0] = 8h.

Table 218. FAST_READ command

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	0Bh or 0Ch	Command name: FAST_READ or 4FAST_READ FAST_READ is 3-byte address mode. 4FAST_READ is 4-byte address mode.
1	ADDR_SDR	0h (Single)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	DUMMY_SDR	0h (Single)	08h	Dummy cycle is 8 (in serial root clock) as CR2V[3:0] = 8h.
3	READ_SDR	0h (Single)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size. This value is ignored for AHB command.
4-7	STOP (00h)	0h	00h	

31.6.1.7 Dual IO Fast Read command

Table 219 shows Dual IO FAST_READ command sequence. This table shows Cypress SPI Configuration Register bits CR2V[7] = 0, CR2V[3:0] = 8h, Continuous Read mode.

Table 219. Dual IO FAST_READ command (Continuous Read mode)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	BBh or BCh	Command name: DIOR or 4DIOR DIOR is 3-byte address mode. 4DIOR is 4-byte address mode.
1	ADDR_SDR	1h (Dual)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	MODE8_SDR	1h (Dual)	Axh	Enter continuous read mode or remain in continuous read mode.
3	DUMMY_SDR	1h (Dual)	08h	Dummy cycle is 8 (in serial root clock) as CR2V[3:0] = 8h.
4	READ_SDR	1h (Dual)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size. This value is ignored for AHB command.
5	JMP_ON_CS	0h (or Don't care)	01h	CMD instruction is bypassed after the first read access.
6-7	STOP (00h)	0h	00h	

Table 220 shows Dual IO FAST_READ command sequence. This table shows Cypress SPI Configuration Register bits CR2V[7] = 0, CR2V[3:0] = 8h, Noncontinuous Read mode.

Table 220. Dual IO FAST_READ command (Noncontinuous Read mode)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	BBh or BCh	Command name: DIOR or 4DIOR DIOR is 3-byte address mode. 4DIOR is 4-byte address mode.
1	ADDR_SDR	1h (Dual)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	MODE8_SDR	1h (Dual)	Any value other than Axh	Exit continuous read mode or remain in noncontinuous read mode.
3	DUMMY_SDR	1h (Dual)	08h	Dummy cycle is 8 (in serial root clock) as CR2V[3:0] = 8h.
4	READ_SDR	1h (Dual)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size. This value is ignored for AHB command.
5-7	STOP (00h)	0h	00h	

31.6.1.8 Quad IO Fast Read command

Table 221 shows Quad IO FAST_READ command sequence. This table shows Cypress SPI Configuration Register bits CR2V[7] = 0, CR2V[6] = 0, CR2V[3:0] = 8h, Non-QPI mode, Noncontinuous read mode.

Table 221. Quad IO FAST_READ command (Non-QPI mode, Noncontinuous read mode)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	EBh or ECh	Command name: QIOR or 4QIOR QIOR is 3-byte address mode. 4QIOR is 4-byte address mode.
1	ADDR_SDR	2h (Quad)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	MODE8_SDR	2h (Quad)	Any value other than Axh	Exit continuous read mode or remain in noncontinuous read mode.
3	DUMMY_SDR	2h (Quad)	08h	Dummy cycle is 8 (in serial root clock) as CR2V[3:0] = 8h.
4	READ_SDR	2h (Quad)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size. This value is ignored for AHB command.
5-7	STOP (00h)	0h	00h	

Table 222 shows Quad IO FAST_READ command sequence. This table shows Cypress SPI Configuration Register bits CR2V[7] = 0, CR2V[6] = 0, CR2V[3:0] = 8h, Non-QPI mode, Continuous read mode.

Table 222. Quad IO FAST_READ command (Non-QPI mode, Continuous read mode)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	EBh or ECh	Command name: QIOR or 4QIOR QIOR is 3-byte address mode. 4QIOR is 4-byte address mode.
1	ADDR_SDR	2h (Quad)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	MODE8_SDR	2h (Quad)	A0h	Enter continuous read mode or remain in continuous read mode.
3	DUMMY_SDR	2h (Quad)	08h	Dummy cycle is 8 (in serial root clock) as CR2V[3:0] = 8h.
4	READ_SDR	2h (Quad)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size. This value is ignored for AHB command.

Table continues on the next page...

Table 222. Quad IO FAST_READ command (Non-QPI mode, Continuous read mode) (continued)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
5	JMP_ON_CS	0h (or Don't care)	01h	CMD instruction is bypassed after the first read access.
6-7	STOP (00h)	0h	00h	

Table 223 shows Quad IO FAST_READ command sequence. This table shows Cypress SPI Configuration Register bits CR2V[7] = 0, CR2V[6] = 1, CR2V[3:0] = 8h, QPI mode, Continuous read mode.

Table 223. Quad IO FAST_READ command (QPI mode, Continuous read mode)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	2h (Quad)	EBh or ECh	Command name: QIOR or 4QIOR QIOR is 3-byte address mode. 4QIOR is 4-byte address mode.
1	ADDR_SDR	2h (Quad)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	MODE8_SDR	2h (Quad)	A0h	Enter continuous read mode or remain in continuous read mode.
3	DUMMY_SDR	2h (Quad)	08h	Dummy cycle is 8 (in serial root clock) as CR2V[3:0] = 8h.
4	READ_SDR	2h (Quad)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size. This value is ignored for AHB command.
5	JMP_ON_CS	0h (or Don't care)	01h	CMD instruction is bypassed after the first read access.
6-7	STOP (00h)	0h	00h	

31.6.1.9 DDR Quad IO Fast Read command

Table 224 shows DDR Quad IO FAST_READ command sequence. This table shows Cypress SPI Configuration Register bits CR2V[7] = 0, CR2V[6] = 0, CR2V[3:0] = 8h, Non-QPI mode, Non-continuous read mode.

Table 224. DDR Quad IO FAST_READ command (Non-QPI mode, Noncontinuous read mode)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	EDh or EEh	Command name: QIOR_DDR or 4QIOR_DDR QIOR_DDR is 3-byte address mode. 4QIOR_DDR is 4-byte address mode.

Table continues on the next page...

Table 224. DDR Quad IO FAST_READ command (Non-QPI mode, Noncontinuous read mode) (continued)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
1	ADDR_DDR	2h (Quad)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	MODE8_DDR	2h (Quad)	Any value other than Axh	Exit continuous read mode or keep in noncontinuous read mode.
3	DUMMY_DDR	2h (Quad)	08h	Dummy cycle is 8 (in serial root clock) as CR2V[3:0] = 8h.
4	LEARN_DDR	2h (Quad)	1h	DLP is 8 bits.
5	READ_DDR	2h (Quad)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size. This value is ignored for AHB command.
6-7	STOP (00h)	0h	00h	

Table 225 shows DDR Quad IO FAST_READ command sequence. This table shows Cypress SPI Configuration register bits CR2V[7] = 0, CR2V[6] = 0, CR2V[3:0] = 8h, Non-QPI mode, Continuous read mode.

Table 225. DDR Quad IO FAST_READ command (Non-QPI mode, Continuous read mode)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	EDh or EEh	Command name: QIOR_DDR or 4QIOR_DDR QIOR_DDR is 3-byte address mode. 4QIOR_DDR is 4-byte address mode.
1	ADDR_DDR	2h (Quad)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	MODE8_DDR	2h (Quad)	A0h	Enter continuous read mode or remain in continuous read mode.
3	DUMMY_DDR	2h (Quad)	08h	Dummy cycle is 8 (in serial root clock) as CR2V[3:0] = 8h.
4	LEARN_DDR	2h (Quad)	1h	DLP is 8 bits.
5	READ_DDR	2h (Quad)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size. This value is ignored for AHB command.
6	JMP_ON_CS	0h (or Don't care)	01h	CMD instruction is bypassed after the first read access.
7	STOP (00h)	0h	00h	

Table 226 shows DDR Quad IO FAST_READ command sequence. This table shows Cypress SPI Configuration Register bits CR2V[7] = 0, CR2V[6] = 1, CR2V[3:0] = 8h, QPI mode, Continuous read mode.

Table 226. DDR Quad IO FAST_READ command (QPI mode, Continuous read mode)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	2h (Quad)	EDh or EEh	Command name: QIOR_DDR or 4QIOR_DDR QIOR_DDR is 3-byte address mode. 4QIOR_DDR is 4-byte address mode.
1	ADDR_DDR	2h (Quad)	18h or 20h	Address bit number (operand value) should be 24 in 3-byte address mode and 32 in 4-byte address mode.
2	MODE8_DDR	2h (Quad)	A0h	Enter continuous read mode or remain in continuous read mode.
3	DUMMY_DDR	2h (Quad)	08h	Dummy cycle is 8 (in serial root clock) as CR2V[3:0] = 8h.
4	LEARN_DDR	2h (Quad)	1h	DLP is 8 bits.
5	READ_DDR	2h (Quad)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size. This value is ignored for AHB command.
6	JMP_ON_CS	0h (or Don't care)	01h	CMD instruction is bypassed after the first read access.
7	STOP (00h)	0h	00h	

31.6.2 Application on HyperBus device

This section provides the example LUT instruction sequences for HyperBus device (Cypress RPC flash, HyperRam, or HyperFlash).

31.6.2.1 HyperFlash

This section provides example sequences for HyperFlash devices (Cypress S26KS series).

Table 227 shows the read status command sequence.

Table 227. Read status command

Seq num	Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0 (Write - Addr = 555h, Data = 70h)	0	CMD_DDR	3h (Octal)	20h	CA bit 47: (R/W# = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	CMD_DDR	3h (Octal)	00h	Row Address: 0000AAh (24 bit)
	2	CMD_DDR	3h (Octal)	00h	
	3	CMD_DDR	3h (Octal)	AAh	

Table continues on the next page...

Table 227. Read status command (continued)

Seq num	Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
	4	CMD_DDR	3h (Octal)	00h	Column Address: 05h (13 zero bits + 3 valid bits)
	5	CMD_DDR	3h (Octal)	05h	
	6	CMD_DDR	3h (Octal)	00h	Write Data: 0070h
	7	CMD_DDR	3h (Octal)	70h	
1 (Read - Addr = xxx, Data = Status register data)	0	CMD_DDR	3h (Octal)	A0h	CA bit 47: (R/W#) = 1h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	RADDR_DDR	3h (Octal)	18h	Row Address: 24 bits
	2	CADDR_DDR	3h (Octal)	10h	Column Address: (13 zero bits + 3 valid bits)
	3	DUMMY_RWDS_DDR	3h (Octal)	0Bh	When latency count = 11
	4	READ_DDR	0h (Octal)	4h	4-Byte read
	5-7	STOP (00h)	0h	00h	

Table 228 shows the read (memory) command sequence.

Table 228. Read (memory) command

Seq num	Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0 (Read - Addr = xxx, Data = memory data)	0	CMD_DDR	3h (Octal)	A0h	CA bit 47: (R/W#) = 1h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	RADDR_DDR	3h (Octal)	18h	Row Address: 24 bits
	2	CADDR_DDR	3h (Octal)	10h	Column Address: (13 zero bits + 3 valid bits)
	3	DUMMY_RWDS_DDR	3h (Octal)	0Bh	When latency count = 11
	4	READ_DDR	3h (Octal)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size. This value is ignored for AHB command.
	5-7	STOP (00h)	0h	00h	

Table 229 shows the word program command sequence.

Table 229. Word program command

Seq num	Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0 (Write - Addr = 555h, Data = AAh)	0	CMD_DDR	3h (Octal)	20h	CA bit 47: (R/W#) = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	CMD_DDR	3h (Octal)	00h	Row Address: 0000AAh (24 bit)
	2	CMD_DDR	3h (Octal)	00h	
	3	CMD_DDR	3h (Octal)	AAh	
	4	CMD_DDR	3h (Octal)	00h	Column Address: 05h (13 zero bits + 3 valid bits)
	5	CMD_DDR	3h (Octal)	05h	
	6	CMD_DDR	3h (Octal)	00h	Write Data: 00AAh
	7	CMD_DDR	3h (Octal)	AAh	
1 (Write - Addr = 2AAh, Data = 55h)	0	CMD_DDR	3h (Octal)	20h	CA bit 47: (R/W#) = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	CMD_DDR	3h (Octal)	00h	Row Address: 000055h (24 bit)
	2	CMD_DDR	3h (Octal)	00h	
	3	CMD_DDR	3h (Octal)	55h	
	4	CMD_DDR	3h (Octal)	00h	Column Address: 02h (13 zero bits + 3 valid bits)
	5	CMD_DDR	3h (Octal)	02h	
	6	CMD_DDR	3h (Octal)	00h	Write Data: 0055h
	7	CMD_DDR	3h (Octal)	55h	
2 (Write - Addr = 555h, Data = A0h)	0	CMD_DDR	3h (Octal)	20h	CA bit 47: (R/W#) = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	CMD_DDR	3h (Octal)	00h	Row Address: 0000AAh (24 bit)
	2	CMD_DDR	3h (Octal)	00h	
	3	CMD_DDR	3h (Octal)	AAh	
	4	CMD_DDR	3h (Octal)	00h	Column Address: 05h (13 zero bits + 3 valid bits)
	5	CMD_DDR	3h (Octal)	55h	
	6	CMD_DDR	3h (Octal)	00h	Write Data: 00A0h

Table continues on the next page...

Table 229. Word program command (continued)

Seq num	Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
	7	CMD_DDR	3h (Octal)	A0h	
3 (Word Program)	0	CMD_DDR	3h (Octal)	20h	CA bit 47: (R/W#) = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	RADDR_DDR	3h (Octal)	18h	Row Address: 24 bits
	2	CADDR_DDR	3h (Octal)	10h	Column Address: (13 zero bits + 3 valid bits)
	3	WRITE_DDR	3h (Octal)	02h	2-Byte written data
	4-7	STOP (0h)	0h	00h	

Table 230 shows Write-to-Buffer and Program-Buffer-to-Flash command sequence.

Table 230. Write-to-Buffer and Program-Buffer-to-Flash command

Seq num	Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0 (Write - Addr = 555h, Data = AAh)	0	CMD_DDR	3h (Octal)	20h	CA bit 47: (R/W#) = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	CMD_DDR	3h (Octal)	00h	Row Address: 0000AAh (24 bit)
	2	CMD_DDR	3h (Octal)	00h	
	3	CMD_DDR	3h (Octal)	AAh	
	4	CMD_DDR	3h (Octal)	00h	Column Address: 05h (13 zero bits + 3 valid bits)
	5	CMD_DDR	3h (Octal)	05h	
	6	CMD_DDR	3h (Octal)	00h	Write Data: 00AAh
	7	CMD_DDR	3h (Octal)	AAh	
1 (Write - Addr = 2AAh, Data = 55h)	0	CMD_DDR	0h (Octal)	20h	CA bit 47: (R/W#) = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	CMD_DDR	3h (Octal)	00h	Row Address: 000055h (24 bit)
	2	CMD_DDR	3h (Octal)	00h	
	3	CMD_DDR	3h (Octal)	55h	

Table continues on the next page...

Table 230. Write-to-Buffer and Program-Buffer-to-Flash command (continued)

Seq num	Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
	4	CMD_DDR	3h (Octal)	00h	Column Address: 02h (13 zero bits + 3 valid bits)
	5	CMD_DDR	3h (Octal)	02h	
	6	CMD_DDR	3h (Octal)	00h	Write Data: 0055h
	7	CMD_DDR	3h (Octal)	55h	
2 (Write - Addr = SA, Data = 25h)	0	CMD_DDR	3h (Octal)	20h	CA bit 47: (R/W#) = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	RADDR_DDR	3h (Octal)	18h	Row Address: SA (24 bit) SA is sector address. Write SA to IPCR0[SFAR].
	2	CADDR_DDR	3h (Octal)	10h	Column Address: 13 zero bits + 3 valid bits
	3	CMD_DDR	3h (Octal)	00h	Write Data: 0025h
	4	CMD_DDR	3h (Octal)	25h	
2 (Write - Addr = SA, Data = WC)	0	CMD_DDR	3h (Octal)	20h	CA bit 47: (R/W#) = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	RADDR_DDR	3h (Octal)	18h	Row Address: SA (24 bit) SA is sector address. Write SA to IPCR0[SFAR].
	2	CADDR_DDR	3h (Octal)	10h	Column Address: 13 zero bits + 3 valid bits
	3	CMD_DDR	3h (Octal)	WC	Write Data: WC WC is word count.
	4	CMD_DDR	3h (Octal)		
3 - N (Write - Addr = WBL, Data = PD) N is the word count + 2	0	CMD_DDR	3h (Octal)	20h	CA bit 47: (R/W#) = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	RADDR_DDR	3h (Octal)	18h	Row Address: WBL (24 bit) WBL is write buffer location. Write WBL to IPCR0[SFAR].

Table continues on the next page...

Table 230. Write-to-Buffer and Program-Buffer-to-Flash command (continued)

Seq num	Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
	2	CADDR_DDR	3h (Octal)	10h	Column Address: 13 zero bits + 3 valid bits
	3	WRITE_DDR	3h (Octal)	02h	2-Byte write data
	4-7	STOP (0h)	0h	00h	
N + 1 (Write - Addr = SA, Data = 29) Program Buffer to Flash	0	CMD_DDR	3h (Octal)	20h	CA bit 47: (R/W#) = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
	1	RADDR_DDR	3h (Octal)	18h	Row Address: SA (24 bit) SA is sector address. Write SA to IPCR0[SFAR].
	2	CADDR_DDR	3h (Octal)	10h	Column Address: 13 zero bits + 3 valid bits
	3	CMD_DDR	3h (Octal)	00h	Write Data: 29h
	4	CMD_DDR	3h (Octal)	29h	
	5-7	STOP (0h)	0h	00h	

31.6.2.2 HyperRAM

This section provides example sequences for HyperRAM (Cypress S27KL series).

The Read (memory) command sequence is same as HyperFlash. Table 231 shows the Write (memory) command sequence.

Table 231. Write (memory) command

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_DDR	3h (Octal)	20h	CA bit 47: (R/W#) = 0h CA bit 46: (Target) = 0h CA bit 45: (Burst Type) = 1h CA bit 44-40: All reserved = 0h
1	RADDR_DDR	3h (Octal)	18h	Row Address: 24 bits
2	CADDR_DDR	3h (Octal)	10h	Column Address: (13 zero bits + 3 valid bits)
3	DUMMY_RWDS_DDR	3h (Octal)	0Bh	When latency count = 11
4	WRITE_DDR	3h (Octal)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default writing

Table continues on the next page...

Table 231. Write (memory) command (continued)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
				data size. This value is ignored for AHB command.
5-7	STOP (00h)	0h	00h	

31.6.3 Application on Serial NAND flash device

This section provides example LUT instruction sequences for serial NAND flash device (Micron Flash MT29 series). The operation of serial NAND flash is similar to serial NOR flash.

The read operation sequence is:

- Page read. Transfer the data from the NAND flash array to the cache register.
- Get feature to read the status.
- Random data read

Table 232 shows the page read command sequence.

Table 232. Page Read command

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	1h (Single)	13h	Command code: 13h
1	RADDR_SDR	1h (Single)	18h	Row address: 24 bit
2-7	STOP (0h)	0h	00h	

Table 233 shows Get Feature command sequence.

Table 233. Get Feature command

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	0h (Single)	0Fh	Command code: 0Fh
1	CMD_SDR	1h (Single)	C0h	Status register address (C0h)
2	READ_SDR	0x1 (Single)	02h	2 bytes read data
3-7	STOP (0h)	0h	00h	

Table 234 shows Random Data Read command sequence.

Table 234. Read from Cache x4 command

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	1h (Single)	6Bh	Command code: 6Bh
1	MODE4_SDR	1h (Single)	0h or 1h	If plane selection is one, software should decode the flash address and write 1h to mode bits. If plane selection

Table continues on the next page...

Table 234. Read from Cache x4 command (continued)

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
				is zero, software should write 0h to mode bits. Plane selection bit is 18th bit of flash address. If NAND flash size is less than 4 Gbit, plane selection is always zero.
2	CADDR_SDR	1h (Single)	0Ch	Column address: 12 bit
3	DUMMY_SDR	2h (Quad)	08h	Dummy cycle number: 8 (serial root clock)
4	READ_SDR	2h (Quad)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default reading data size.
5-7	STOP (0x0)	0x0	0x00	

The program operation sequence is:

- Write enable
- Program Load. Transfer the write data to the cache register.
- Program Execute
- Get Feature to read the status.

Table 235 shows Program Load command sequence.

Table 235. Program Load command

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	1h (Single)	02h	Command code: 02h
1	MODE4_SDR	1h (Single)	0h or 1h	If plane selection is one, software should decode the flash address and write 1h to mode bits. If plane selection is zero, software should write 0h to mode bits. Plane selection bit is 18th bit of flash address. If NAND flash size is less than 4 Gbit, plane selection is always zero.
2	CADDR_SDR	1h (Single)	0Ch	Column address: 12 bit
3	WRITE_SDR	1h (Single)	Any nonzero value	If IPCR1[IDATSZ] is zero, this operand value can be used as default writing data size.
4-7	STOP (0h)	0h	00h	

Table 236 shows Program Execute command sequence.

Table 236. Program Execute command

Instruction number	Instruction opcode[5:0]	Instruction num_pads[1:0]	Instruction operand[7:0]	Comment
0	CMD_SDR	1h (Single)	10h	Command code: 10h
1	RADDR_SDR	1h (Single)	18h	Row Address: 24 bit
2-7	STOP (0h)	0h	00h	

31.6.4 Application on FPGA device

An FPGA device can be accessed via both AHB and IP commands. All AHB accesses to FPGA are transparent to the software driver (no software intervention). An FPGA device may have some special requirements.

Table 237. Special requirements for FPGA devices

Condition	How to manage the condition
Device type may be different on A1, A2, B1, or B2.	Write 0 to MCR2[SAMEDEVICEEN] and configure the FLSHxCR0 and FLSHxCR1 registers separately for up to four external devices.
Device requires different wait cycle for programming.	The AHB write wait cycle number can be set separately for these four external devices (via FLSHxCR2[AWRWAIT]). Software can configure the sequences in LUT with different dummy instructions (operand determines dummy cycle). FlexSPI holds the AHB bus ready for this wait time; AHB bus performance may decrease when this wait time is very long.
Device requires different wait cycle for reading.	The AHB read sequence index and sequence number can be set separately for these four external devices (via FLSHxCR2[ARDSEQID] and FLSHxCR2[ARDSEQNUM]). Software can configure the sequences in LUT with different dummy instructions (operand determines dummy cycle).
Device may be sensitive to read instruction clock cycle number.	If its internal memory is implemented similar to a FIFO, the device is sensitive to the read instruction clock cycle number. Software can send the data size information to the external device via DATSZ instruction. The FPGA device decodes the data size information and determines how many data bytes must be popped.
Device may require interval time between chip select valid.	This interval can be managed via FLSHxCR1[CSINTERVAL].
Device may use SCLK as reference clock for its internal PLL.	SCLK must be free-running and the clock frequency must be stable. To achieve this configuration, write 1 to MCR0[SCKFREERUNEN] and only use SDR sequences.

31.6.5 Overview of error categories, flags, and triggered sources

Table 238 provides an overview of the categories, flags, and triggered sources of errors in FlexSPI.

Table 238. Error category, triggered sources, and flags

Error Category	Triggered Sources	Description	Error Flags
Command grant error	AHB write command	Command grant timeout	INTR[AHBCMDGE] is set AHB bus error response

Table continues on the next page...

Table 238. Error category, triggered sources, and flags (continued)

Error Category	Triggered Sources	Description	Error Flags
	AHB read command		INTR[AHBCMDGE] is set AHB bus error response
	IP command		INTR[IPCMDGE] is set
Command check error	AHB write command	• AHB write command with JMP_ON_CS instruction used in the sequence • Unknown instruction opcode in the sequence • Instruction DUMMY_SDR or DUMMY_RWDS_SDR used in DDR sequence • Instruction DUMMY_DDR or DUMMY_RWDS_DDR used in SDR sequence	INTR[AHBCMDERR] is set Command is not executed when an error is detected in command check
	AHB read command	• Unknown instruction opcode in the sequence • Instruction DUMMY_SDR or DUMMY_RWDS_SDR used in DDR sequence • Instruction DUMMY_DDR or DUMMY_RWDS_DDR used in SDR sequence	INTR[AHBCMDERR] is set Command is not executed when an error is detected in command check
	IP command	• IP command with JMP_ON_CS instruction used in the sequence • Unknown instruction opcode in the sequence • Instruction DUMMY_SDR or DUMMY_RWDS_SDR used in DDR sequence • Instruction DUMMY_DDR or DUMMY_RWDS_DDR used in SDR sequence • Flash boundary across	INTR[IPCMDERR] is set Command is not executed when an error is detected in command check
Command execution error	AHB write command	Command timeout during execution	INTR[AHBCMDERR] is set INTR[SEQTIMEOUT] is set An AHB bus error response

Table continues on the next page...

Table 238. Error category, triggered sources, and flags (continued)

Error Category	Triggered Sources	Description	Error Flags
			occurs, except in following cases: • Flush triggers AHB write command (INCR burst ended with AHB_TX_BUF not empty) • AHB bufferable write access and bufferable enabled (AHBCR[BUFFERABLEEN] = 1)
	AHB read command		INTR[AHBCMDERR] is set INTR[SEQTIMEOUT] is set AHB bus error response
	IP command		INTR[IPCMDERR] is set INTR[SEQTIMEOUT] is set
AHB bus timeout	AHB write command	AHB bus timeout (no bus ready return)	AHB bus error response
	AHB read command		

31.7 Memory map and register definition

This section includes the FlexSPI module memory map and detailed descriptions of all registers.

31.7.1 Register access

All registers can be accessed with 8-bit, 16-bit, and 32-bit width operations. Never change the values of reserved fields in control registers. Changing the values of reserved fields may impact the normal functioning of the controller.

NOTE

When FlexSPI can access sensitive memory contents, the FlexSPI controller should protect those contents using resource isolation such as XRDC or TrustZone. Ensure that only trusted software is allowed to access the FlexSPI controller register interface.

31.7.2 FlexSPI register descriptions

31.7.2.1 FlexSPI memory map

FLEXSPI1 base address: 425E_0000h

FLEXSPI2 base address: 445E_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Control 0 (MCR0)	32	RW	FFFF_80C2h
4h	Module Control 1 (MCR1)	32	RW	FFFF_FFFFh
8h	Module Control 2 (MCR2)	32	RW	2000_81F7h
Ch	AHB Bus Control (AHBCR)	32	RW	0000_0018h
10h	Interrupt Enable (INTEN)	32	RW	0000_0000h
14h	Interrupt (INTR)	32	RW	0000_0000h
18h	LUT Key (LUTKEY)	32	RW	5AF0_5AF0h
1Ch	LUT Control (LUTCR)	32	RW	0000_0002h
20h	AHB Receive Buffer 0 Control 0 (AHBRXBUF0CR0)	32	RW	8000_0040h
24h	AHB Receive Buffer 1 Control 0 (AHBRXBUF1CR0)	32	RW	8001_0040h
28h	AHB Receive Buffer 2 Control 0 (AHBRXBUF2CR0)	32	RW	8002_0040h
2Ch	AHB Receive Buffer 3 Control 0 (AHBRXBUF3CR0)	32	RW	8003_0040h
30h	AHB Receive Buffer 4 Control 0 (AHBRXBUF4CR0)	32	RW	8004_0040h
34h	AHB Receive Buffer 5 Control 0 (AHBRXBUF5CR0)	32	RW	8005_0040h
38h	AHB Receive Buffer 6 Control 0 (AHBRXBUF6CR0)	32	RW	8006_0040h
3Ch	AHB Receive Buffer 7 Control 0 (AHBRXBUF7CR0)	32	RW	8007_0040h
60h	Flash Control 0 (FLSHA1CR0)	32	RW	0000_0000h
64h	Flash Control 0 (FLSHA2CR0)	32	RW	0000_0000h
68h	Flash Control 0 (FLSHB1CR0)	32	RW	0000_0000h
6Ch	Flash Control 0 (FLSHB2CR0)	32	RW	0000_0000h
70h	Flash Control 1 (FLSHA1CR1)	32	RW	0000_0063h
74h	Flash Control 1 (FLSHA2CR1)	32	RW	0000_0063h
78h	Flash Control 1 (FLSHB1CR1)	32	RW	0000_0063h
7Ch	Flash Control 1 (FLSHB2CR1)	32	RW	0000_0063h
80h	Flash Control 2 (FLSHA1CR2)	32	RW	0000_0000h
84h	Flash Control 2 (FLSHA2CR2)	32	RW	0000_0000h
88h	Flash Control 2 (FLSHB1CR2)	32	RW	0000_0000h
8Ch	Flash Control 2 (FLSHB2CR2)	32	RW	0000_0000h
94h	Flash Control 4 (FLSHCR4)	32	RW	0000_00C3h
A0h	IP Control 0 (IPCR0)	32	RW	0000_0000h
A4h	IP Control 1 (IPCR1)	32	RW	0000_0000h
B0h	IP Command (IPCMD)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B8h	IP Receive FIFO Control (IPRXFCR)	32	RW	0000_0000h
BCh	IP Transmit FIFO Control (IPTXFCR)	32	RW	0000_0000h
C0h	DLL Control 0 (DLLACR)	32	RW	0000_0100h
C4h	DLL Control 0 (DLLBCR)	32	RW	0000_0100h
E0h	Status 0 (STS0)	32	R	0000_0002h
E4h	Status 1 (STS1)	32	R	0000_0000h
E8h	Status 2 (STS2)	32	R	0100_0100h
ECh	AHB Suspend Status (AHBSPNDSTS)	32	R	0000_0000h
F0h	IP Receive FIFO Status (IPRXFSTS)	32	R	0000_0000h
F4h	IP Transmit FIFO Status (IPTXFSTS)	32	R	0000_0000h
100h - 17Ch	IP Receive FIFO Data a (RFDR0 - RFDR31)	32	R	0000_0000h
180h - 1FCh	IP TX FIFO Data a (TFDR0 - TFDR31)	32	W	0000_0000h
200h - 3FCh	Lookup Table a (LUT0 - LUT127)	32	RW	See section
440h	Receive Buffer Start Address of Region 0 (AHBBUFREGIONSTART0)	32	RW	0000_0000h
444h	Receive Buffer Region 0 End Address (AHBBUFREGIONEND0)	32	RW	0000_0000h
448h	Receive Buffer Start Address of Region 1 (AHBBUFREGIONSTART1)	32	RW	0000_0000h
44Ch	Receive Buffer Region 1 End Address (AHBBUFREGIONEND1)	32	RW	0000_0000h
450h	Receive Buffer Start Address of Region 2 (AHBBUFREGIONSTART2)	32	RW	0000_0000h
454h	Receive Buffer Region 2 End Address (AHBBUFREGIONEND2)	32	RW	0000_0000h
458h	Receive Buffer Start Address of Region 3 (AHBBUFREGIONSTART3)	32	RW	0000_0000h
45Ch	Receive Buffer Region 3 End Address (AHBBUFREGIONEND3)	32	RW	0000_0000h

31.7.2.2 Module Control 0 (MCR0)

Offset

Register	Offset
MCR0	0h

Function

Controls basic functions of FlexSPI module

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	AHBGRANTWAIT								IPGRANTWAIT							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	SCKFREE...	COMBINA...	DOZEEN	HSEN	SERCLKDIV			ATDFEN	ARDFEN	RXCLKSRC		Reserved		MDIS	SWRESET
Reset	1	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0

Fields

Field	Function
31-24 AHBGRANTWAIT	Timeouts Wait Cycle for AHB command Grant Sets duration of timeout wait cycle for AHB commands. If the arbitrator does not grant the AHB-triggered command, it times out after (AHBGRANTTIMEOUT × 1024) AHB clock cycles. When the pending command sequence is IP-triggered and the read or write data size is too large, this grant timeout may occur. When an AHB command grant timeout occurs, an INTR[AHBCMDGE] interrupt is generated. When INTEN[AHBCMDGEEN] = 1, the arbitrator ignores AHB commands. **NOTE** This field is for debugging only. Do not change the value from its default.
23-16 IPGRANTWAIT	Timeout Wait Cycle for IP Command Grant Sets duration of timeout wait cycle for IP commands. If the arbitrator does not grant the IP-triggered command, it times out after (IPGRANTTIMEOUT × 1024) AHB clock cycles. When the pending command sequence is AHB-triggered and the read or write data size is too large, this grant timeout may occur. When IP command grant timeout occurs, an INTR[IPCMDGE] interrupt is generated. When INTEN[IPCMDGEEN] = 1, the arbitrator ignores IP commands. **NOTE** This field is for debugging only. Do not change the value from its default.
15 —	Reserved
14 SCKFREERUNEN	SCLK Free-running Enable Enables free-running SCLK output. For FPGA applications, the external device may use SCLK as a reference clock to its internal PLL. 0b - Disable 1b - Enable
13	Combination Mode Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
COMBINATIONEN	Enables combination mode, which supports flash Octal mode access. When Port A and Port B data are four bits wide, this mode combines Port A and B Data pins (A_DATA[3:0] and B_DATA[3:0]). When Port A and Port B data are eight bits wide, combination mode is not supported; write 0 to this field in this case. 0b - Disable 1b - Enable
12 DOZEEN	Doze Mode Enable Enables Doze mode. When enabled, AHB clock and serial clock are gated off when there is a Doze-mode request from the system. 0b - Disable 1b - Enable
11 HSEN	Half Speed Serial Flash Memory Access Enable Enables the clock divider to provide a half-speed clock to external serial flash devices (A_SCLK/ B_SCLK) for all commands in SDR and DDR mode. Write 1 to MCR0[MDIS] before changing the value of this field. Failure to do so may cause issues in the internal logic or state machine. 0b - Disable 1b - Enable
10-8 SERCLKDIV	Serial Root Clock Divider Sets divider value for serial root clock. The serial root clock can be divided inside FlexSPI . See Clocking. **NOTE** Do not modify this field while FlexSPI is active (MCR0[MDIS] = 0). 000b - Divided by 1 001b - Divided by 2 010b - Divided by 3 011b - Divided by 4 100b - Divided by 5 101b - Divided by 6 110b - Divided by 7 111b - Divided by 8
7 ATDFEN	AHB Write Access to IP Transmit FIFO Enable Enables AHB write access to IP transmit FIFO. See Writing data to IP transmit FIFO. 0b - AHB write access disabled. IP bus writes to IP transmit FIFO. AHB bus write access to IP transmit FIFO memory space produces bus error.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - AHB write access enabled. AHB bus writes to IP transmit FIFO. IP Bus write access to IP transmit FIFO memory space is ignored and causes no bus error.
6 ARDFEN	AHB Read Access to IP Receive FIFO Enable Enables AHB read access to IP receive FIFO. See Reading data from IP receive FIFO. 0b - AHB read access disabled. IP bus reads IP receive FIFO. AHB Bus read access to IP receive FIFO memory space produces bus error. 1b - AHB read access enabled. AHB bus reads IP receive FIFO. IP Bus read access to IP receive FIFO memory space returns data zero and causes no bus error.
5-4 RXCLKSRC	Sample Clock Source for Flash Reading Selects sampling clock source for flash memory reading. See Receive clock source features. 00b - Dummy Read strobe that FlexSPI generates, looped back internally 01b - Dummy Read strobe that FlexSPI generates, looped back from DQS pad 10b - Reserved 11b - Flash-memory-provided read strobe and input from DQS pad
3-2 —	Reserved
1 MDIS	Module Disable Disables FlexSPI module. When the module is disabled, AHB and serial clock are gated off internally to save power. Only register access is allowed, except for the LUT, IP receive FIFO, and IP transmit FIFO. 0b - No impact 1b - Module disable
0 SWRESET	Software Reset Resets the internal FlexSPI state machine and aborts any transactions in progress. Hardware automatically writes 0 to this field after software reset is done. Configuration registers are not reset. 0b - No impact 1b - Software reset

31.7.2.3 Module Control 1 (MCR1)

Offset

Register	Offset
MCR1	4h

Function

Controls basic functions of FlexSPI module

Diagram

Fields

Field	Function
31-16 SEQWAIT	Command Sequence Wait Sets wait time for command sequence. Command sequence execution times out and aborts after (SEQWAIT × 1024) serial root clock cycles. When this timeout occurs, if the interrupt is enabled (INTEN[SEQTIMEOUTEN] = 1), an INTR[SEQTIMEOUT] interrupt is generated. Also, the arbitrator ignores AHB commands. **NOTE** You cannot write 0 to this field.
15-0 AHBBUSWAIT	AHB Bus Wait Sets wait time for AHB bus. When data is not received from or transmitted to serial flash memory space after (AHBBUSWAIT × 1024) AHB clock cycles, the read or write access times out. When this timeout occurs, the AHB bus receives an error response, and an interrupt is generated (INTEN[AHBBUSERROREN]). When INTR[AHBBUSERROREN] = 1, the arbitrator ignores AHB commands. **NOTE** You cannot write 0 to this field.

31.7.2.4 Module Control 2 (MCR2)

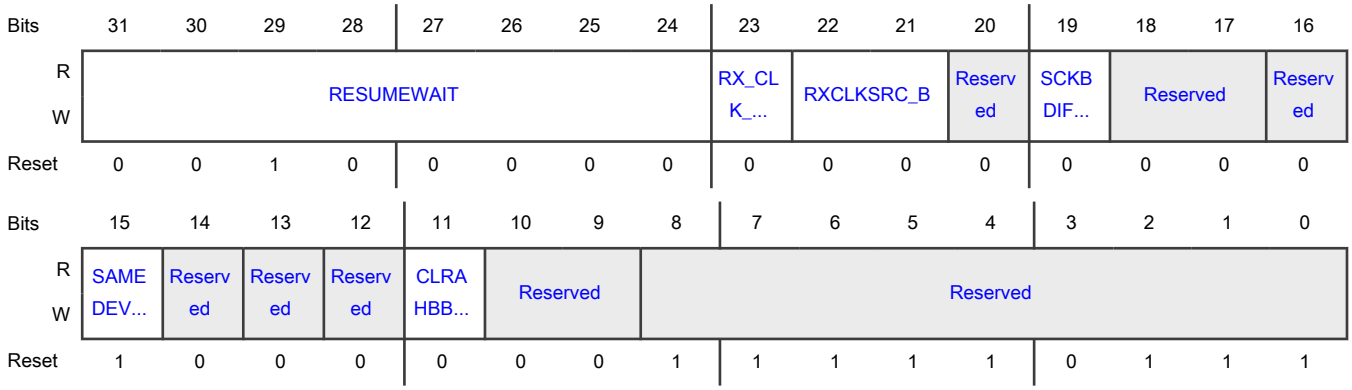
Offset

Register	Offset
MCR2	8h

Function

Controls basic functions of FlexSPI module.

Diagram



Fields

Field	Function
31-24 RESUMEWAIT	Resume Wait Duration Determines duration (in AHB clock cycles) to remain in idle state before suspended command sequence is resumed. See Command abort and suspend.
23 RX_CLK_SRC_DIFF	Sample Clock Source Different Selects whether to use the same clock source or a different clock source for Port A and Port B. 0b - Use MCR0[RXCLKSRC] for Port A and Port B. MCR2[RXCLKSRC_B] is ignored and MCR0[RXCLKSRC] selects the Sample Clock source for Flash Reading of both ports A and B. 1b - Use MCR0[RXCLKSRC] for Port A, and MCR2[RXCLKSRC_B] for Port B. MCR0[RXCLKSRC] selects the Sample Clock source for Flash Reading of port A (A_SCLK) and MCR2[RXCLKSRC_B] selects the Sample Clock source for Flash Reading of port B (B_SCLK).
22-21 RXCLKSRC_B	Port B Receiver Clock Source Selects Port B sample clock source selection for flash memory reading. 00b - Dummy read strobe that FlexSPI generates, looped back internally. 01b - Dummy read strobe that FlexSPI generates, looped back from DQS pad. 10b - SCLK output clock and looped back from SCLK padReserved 11b - Flash-memory-provided read strobe and input from DQS pad
20 —	Reserved
19 SCKBDIFFOPT	SCLK Port B Differential Output Controls whether to use B_SCLK pad as the Port B SCLK output. Alternatively, B_SCLK pad can be used as A_SCLK differential clock output (inverted clock to A_SCLK). Before changing the value of this field, write 1 to MCR0[MDIS]. After changing the value of this field, write 1 to MCR0[SWRESET].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Use B_SCLK pad as port B SCLK clock output. Port B flash memory access is available. 1b - Use B_SCLK pad as port A SCLK inverted clock output (Differential clock to A_SCLK). Port B flash memory access is not available.
18-17 —	Reserved
16 —	Reserved
15 SAMEDEVICEEN	Same Device Enable Sets all external devices (A1, A2, B1, and B2) to be the same devices (in type and in size). 0b - In Individual mode, FLSHA1CRx and FLSHA2CRx, FLSHB1CRx and FLSHB2CRx settings are applied to Flash A1, A2, B1, B2 separately. In Parallel mode, FLSHA1CRx register setting is applied to Flash A1 and B1, FLSHA2CRx register setting is applied to Flash A2 and B2. FLSHB1CRx and FLSHB2CRx register settings are ignored. 1b - FLSHA1CR0, FLSHA1CR1, and FLSHA1CR2 register settings are applied to Flash A1, A2, B1, B2. FLSHA2CRx, FLSHB1CRx, and FLSHB2CRx settings are ignored.
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 CLRAHBBUFOPT	Clear AHB Buffer Determines whether AHB receive and transmit buffers are cleared automatically when FlexSPI returns Stop mode ACK. If an AHB receive buffer or transmit buffer will be powered off in Stop mode, software should write 1 to this field. Otherwise, an AHB read access after exiting Stop mode may hit either buffer, but their data entries are invalid. 0b - Not cleared automatically 1b - Cleared automatically
10-9 —	Reserved
8-0 —	Reserved

31.7.2.5 AHB Bus Control (AHBCR)

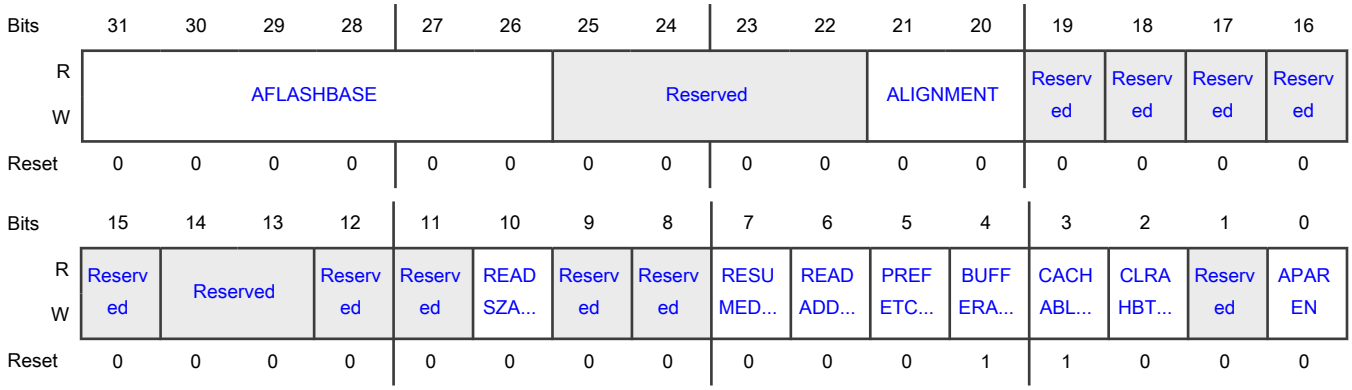
Offset

Register	Offset
AHBCR	Ch

Function

Controls AHB bus interface functions.

Diagram



Fields

Field	Function
31-26 AFLASHBASE	AHB Memory-Mapped Flash Base Address **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
FLEXSPI1	AHBCR[31–27]	AHBCR[26]
FLEXSPI2	AHBCR	—

25-22 —	Reserved
21-20 ALIGNMENT	AHB Boundary Alignment Configures the size of AHB read and write boundary. All accesses that cross the boundary are divided into smaller sub accesses.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - No limit 01b - 1 KB 10b - 512 bytes 11b - 256 bytes
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14-13 —	Reserved
12 —	Reserved
11 —	Reserved
10 READSZALIGN	AHB Read Size Alignment Configures how AHB read size is determined. See Table 208. **NOTE** When 1, data prefetching is disabled regardless of other field settings such as PREFETCH_EN and OTFAD_EN. Read size is rounded up to the nearest eight-byte multiple. For example, if current AHB read size is 12 bytes, it is aligned to 16 bytes. When the fetch size must be eight-byte aligned but no speculative fetching is wanted, write 1 to this field. 0b - Register settings such as PREFETCH_EN and OTFAD_EN determine AHB read size. 1b - AHB read size to up size to 8 bytes aligned, no prefetching
9	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
8 —	Reserved
7 RESUMEDISABLE	AHB Read Resume Disable Disables the resumption of AHB read prefetch. **NOTE** When OTFAD is enabled, this field does not affect the resume function; AHB read resume is always disabled. When OTFAD is disabled, suspended AHB read prefetch resumes based on the values of this field and MCR2[RESUMEWAIT]. 0b - Suspended AHB read prefetch resumes when AHB is IDLE. 1b - Suspended AHB read prefetch does not resume once aborted.
6 READADDROPT	AHB Read Address Option Removes AHB burst start address alignment limitation. When the FlexSPI controller is used for FPGA application, this field may be required for FlexSPI to fetch the exact byte number as an AHB burst. In this case, FPGA device should be non-word-addressable and this field must be 0. **NOTE** When OTFAD is enabled, FlexSPI functions as it does when READADDROPT = 1 regardless the value of this field. 0b - AHB read burst start address alignment is limited when flash memory is accessed in parallel mode or flash is word-addressable. 1b - AHB read burst start address alignment is not limited. FlexSPI fetches more data than the AHB burst requires for address alignment.
5 PREFETCHEN	AHB Read Prefetch Enable Enables AHB read prefetch. When enabled, FlexSPI fetches more flash read data than the current AHB burst requires. This action reduces the read latency for next AHB read access. **NOTE** When OTFAD is enabled, the AHB prefetch function is enabled regardless of the value of this field. **NOTE** AHB read prefetch is enabled only when both this field and AHBRXBUFnCR0[PREFETCHEN] are 1. 0b - Disable 1b - Enable
4	Bufferable Write Access Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
BUFFERABLEEN	Enables AHB bus bufferable write access. This field affects the last beat of an AHB write access. See AHB write access to flash memory. 0b - Disabled. For all AHB write accesses (bufferable or nonbufferable), FlexSPI returns AHB Bus Ready after transmitting all data and finishing command. 1b - Enabled. For AHB bufferable write access, FlexSPI returns AHB Bus Ready when the arbitrator grants the AHB command. FlexSPI does not wait for the AHB command to finish.
3 CACHABLEEN	Cacheable Read Access Enable Enables AHB bus cacheable read access. 0b - Disabled. When an AHB bus cacheable read access occurs, FlexSPI does not check whether it hit the AHB transmit buffer. 1b - Enabled. When an AHB bus cacheable read access occurs, FlexSPI first checks whether the access hit the AHB transmit buffer.
2 CLRAHBTXBUF	Clear AHB Transmit Buffer Clears status and pointers of AHB transmit buffer. When the clear operation completes, this field clears automatically, so it always reads as 0. 0b - No impact. 1b - Enable clear operation.
1 —	Reserved
0 APAREN	AHB Parallel Mode Enable Enables Parallel mode for AHB-triggered read and write commands. 0b - Flash is accessed in Individual mode. 1b - Flash is accessed in Parallel mode.

31.7.2.6 Interrupt Enable (INTEN)

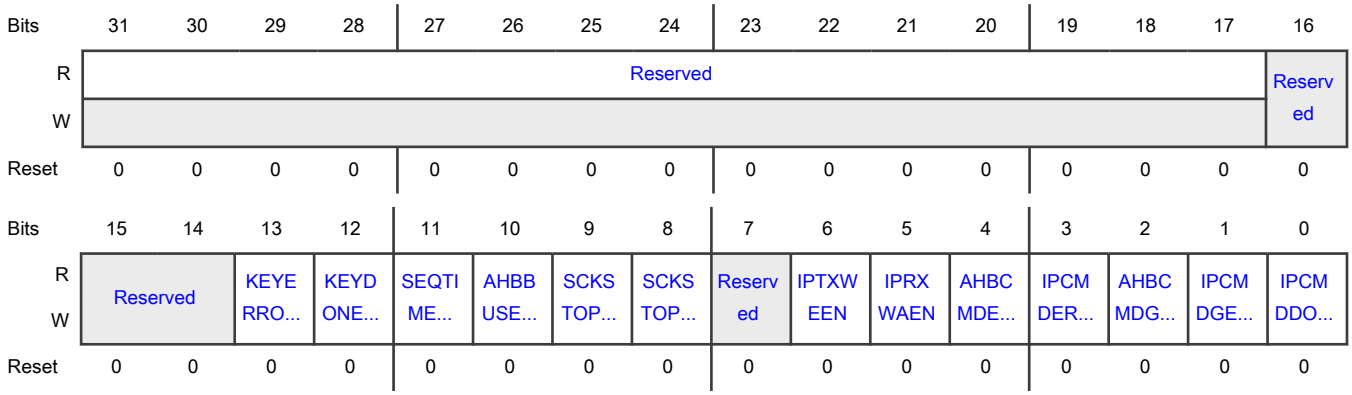
Offset

Register	Offset
INTEN	10h

Function

Includes AHB error response, IPS error response, and ECC error interrupt enables. See Interrupts.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 —	Reserved
15-14 —	Reserved
13 KEYERROREN	OTFAD Key Blob Processing Error Interrupt Enable 0b - Disable interrupt or no impact 1b - Enable interrupt
12 KEYDONEEN	OTFAD Key Blob Processing Done Interrupt Enable 0b - Disable interrupt or no impact 1b - Enable interrupt
11 SEQTIMEOUTEN	Sequence execution Timeout Interrupt Enable 0b - Disable interrupt or no impact 1b - Enable interrupt
10 AHBBUSERROREN	AHB Bus Error Interrupt Enable 0b - Disable interrupt or no impact 1b - Enable interrupt
9 SCKSTOPBYWREN	SCLK Stopped By Write Interrupt Enable Enables interrupt that indicates SCLK is stopped during command sequence because asynchronous transmit FIFO is empty.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable interrupt or no impact 1b - Enable interrupt
8 SCKSTOPBYRDEN	SCLK Stopped By Read Interrupt Enable Enables interrupt that indicates SCLK is stopped during command sequence because asynchronous receive FIFO is full. 0b - Disable interrupt or no impact 1b - Enable interrupt
7 —	Reserved
6 IPTXWEEN	IP Transmit FIFO Watermark Empty Interrupt Enable Enables interrupt that indicates IP transmit FIFO contains more empty space than watermark level. 0b - Disable interrupt or no impact 1b - Enable interrupt
5 IPRXWAEN	IP Receive FIFO Watermark Available Interrupt Enable Enables interrupt that indicates IP receive FIFO contains more valid data than the watermark level. 0b - Disable interrupt or no impact 1b - Enable interrupt
4 AHBCMDERREN	AHB-Triggered Command Sequences Error Detected Interrupt Enable 0b - Disable interrupt or no impact 1b - Enable interrupt
3 IPCMDERREN	IP-Triggered Command Sequences Error Detected Interrupt Enable 0b - Disable interrupt or no impact 1b - Enable interrupt
2 AHBCMDGEEN	AHB-Triggered Command Sequences Grant Timeout Interrupt Enable. 0b - Disable interrupt or no impact 1b - Enable interrupt
1 IPCMDGEEN	IP-Triggered Command Sequences Grant Timeout Interrupt Enable 0b - Disable interrupt or no impact 1b - Enable interrupt
0 IPCMDDONEEN	IP-Triggered Command Sequences Execution Finished Interrupt Enable 0b - Disable interrupt or no impact

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable interrupt

31.7.2.7 Interrupt (INTR)

Offset

Register	Offset
INTR	14h

Function

Includes AHB error response, IPS error response, and ECC error interrupts. See Interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		KEYERROR	KEYDONE	SEQTIME...	AHBBUSE...	SCKSTOP...	SCKSTOP...	Reserved	IPTXWE	IPRXWA	AHBCMDE...	IPCMDERR	AHBCMDGE	IPCMDGE	IPCMDDO...
W				W1C	W1C	W1C	W1C	W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 —	Reserved
15-14 —	Reserved
13 KEYERROR	OTFAD Key Blob Processing Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag
12 KEYDONE	OTFAD key blob processing done interrupt.
11 SEQTIMEOUT	Sequence Execution Timeout **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag
10 AHBBUSERROR	AHB Bus Error Generated upon an AHB bus timeout or an AHB bus illegal access to flash during OTFAD key blob processing. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag
9	SCLK Stopped Due To Empty Transmit FIFO

Table continues on the next page...

Table continued from the previous page...

Field	Function
SCKSTOPBYWR	Generated when SCLK is stopped during command sequence because the asynchronous transmit FIFO is empty. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag
8 SCKSTOPBYRD	SCLK Stopped Due To Full Receive FIFO Generated when SCLK is stopped during command sequence because the asynchronous receive FIFO is full. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag
7 —	Reserved
6 IPTXWE	IP Transmit FIFO Watermark Empty Generated when the IP transmit FIFO contains more empty space than the watermark level. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 IPRXWA	IP Receive FIFO Watermark Available Generated when the IP receive FIFO contains more valid data than the watermark level. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag
4 AHBCMDERR	AHB-Triggered Command Sequences Error Generated upon an AHB-triggered command sequence error. When an error is detected for an AHB command, the command is ignored and not executed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag
3 IPCMDERR	IP-Triggered Command Sequences Error Generated upon an IP-triggered command sequence error. When an error is detected for an IP command, the command is ignored and not executed. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag
2	AHB-Triggered Command Sequences Grant Timeout

Table continues on the next page...

Table continued from the previous page...

Field	Function
AHBCMDGE	**NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag
1 IPCMDGE	IP-Triggered Command Sequences Grant Timeout **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag
0 IPCMDDONE	IP-Triggered Command Sequences Execution Finished Generated upon completion of an IP-triggered command sequence. Also generated when IPCMDGE or IPCMDERR interrupt is generated **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt condition has not occurred 1b - Interrupt condition has occurred When writing 0b - No effect 1b - Clear the flag

31.7.2.8 LUT Key (LUTKEY)

Offset

Register	Offset
LUTKEY	18h

Function

Contains the key to lock and unlock LUT. See Lookup table (LUT).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	KEY															
Reset	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	KEY															
Reset	0	1	0	1	1	0	1	0	1	1	1	1	0	0	0	0

Fields

Field	Function
31-0 KEY	LUT Key Contains key to lock or unlock LUT. The key is 5AF05AF0h. Read value is always 5AF05AF0h.

31.7.2.9 LUT Control (LUTCR)

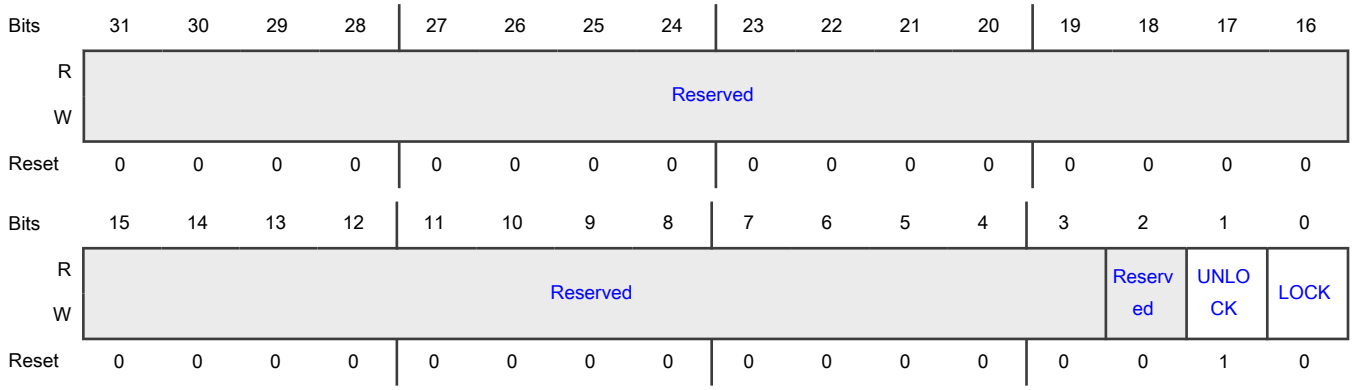
Offset

Register	Offset
LUTCR	1Ch

Function

Used with LUTKEY register to lock or unlock LUT. For the lock or unlock operation to be successful, this register must be written immediately after writing 5AF05AF0h to the LUTKEY register. See Lookup table (LUT) for details on locking and unlocking LUT. You cannot write 00 or 11 to the LOCK and UNLOCK fields.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 —	Reserved
1 UNLOCK	Unlock LUT 0b - LUT is locked (LUTCR[LOCK] must be 1) 1b - LUT is unlocked and can be written
0 LOCK	Lock LUT 0b - LUT is unlocked (LUTCR[UNLOCK] must be 1) 1b - LUT is locked and cannot be written

31.7.2.10 AHB Receive Buffer n Control 0 (AHBRXBUF0CR0 - AHBRXBUF7CR0)

Offset

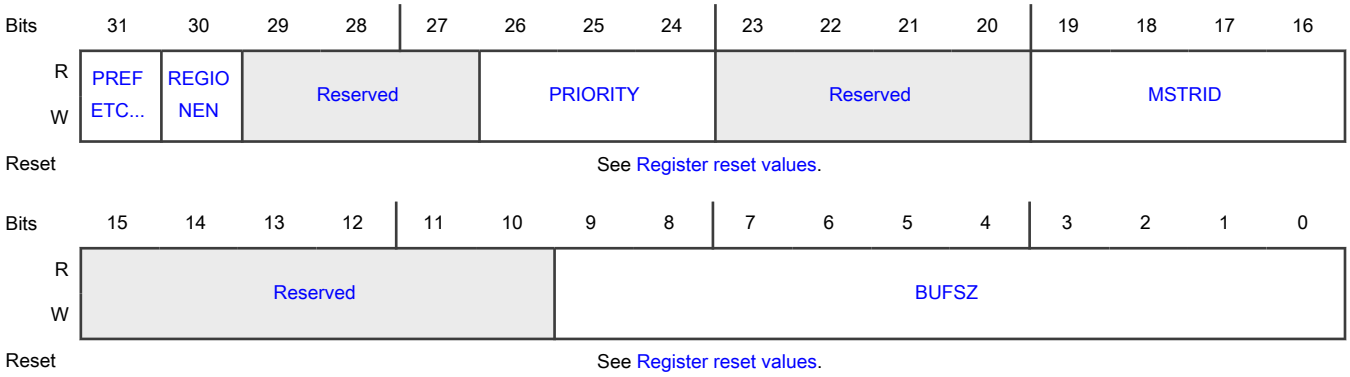
For n = 0 to 7:

Register	Offset
AHBRXBUFnCR0	20h + (n × 4h)

Function

Stores the read data from the SPI interface.

Diagram



Register reset values

Register	Reset value
AHBRXBUF0CR0	FLEXSPI1,FLEXSPI2: 8000_0040h
AHBRXBUF1CR0	FLEXSPI1,FLEXSPI2: 8001_0040h
AHBRXBUF2CR0	FLEXSPI1,FLEXSPI2: 8002_0040h
AHBRXBUF3CR0	FLEXSPI1,FLEXSPI2: 8003_0040h
AHBRXBUF4CR0	FLEXSPI1,FLEXSPI2: 8004_0040h
AHBRXBUF5CR0	FLEXSPI1,FLEXSPI2: 8005_0040h
AHBRXBUF6CR0	FLEXSPI1,FLEXSPI2: 8006_0040h
AHBRXBUF7CR0	FLEXSPI1,FLEXSPI2: 8007_0040h

Fields

Field	Function
31 PREFETCHEN	AHB Read Prefetch Enable Enables AHB read prefetch for the controller corresponding to the current AHB receive buffer. The prefetch feature is disabled when AHBCR[PREFETCHEN] is 0. You can use this field to enable or disable prefetch separately for each controller. 0b - Disabled 1b - Enabled when AHBCR[PREFETCHEN] is enabled.
30 REGIONEN	AHB Receive Buffer Address Region Enable Enable prefetch buffer enhancement. **NOTE** Only AHBRXBUF 0-3 have this function. 0b - Disabled. The buffer hit is based on the value of MSTRID only.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled. The buffer hit is based on the value of MSTRID and the address within AHBBUFREGIONSTARTn and AHBREGIONENDn.
29-27 —	Reserved
26-24 PRIORITY	AHB Controller Read Priority Configure the priority for the AHB Controller Read to which this AHB receive buffer is assigned. 7 is the highest priority, 0 the lowest. See Command abort and suspend.
23-20 —	Reserved
19-16 MSTRID	AHB Controller ID Configures the ID of the AHB controller to which this AHB receive buffer is assigned. See AHB receive buffer management.
15-10 —	Reserved
9-0 BUFSZ	AHB Receive Buffer Size Configures the size of the AHB receive buffer in multiples of 64 bits. See AHB receive buffer management.

31.7.2.11 Flash Control 0 (FLSHA1CR0 - FLSHB2CR0)

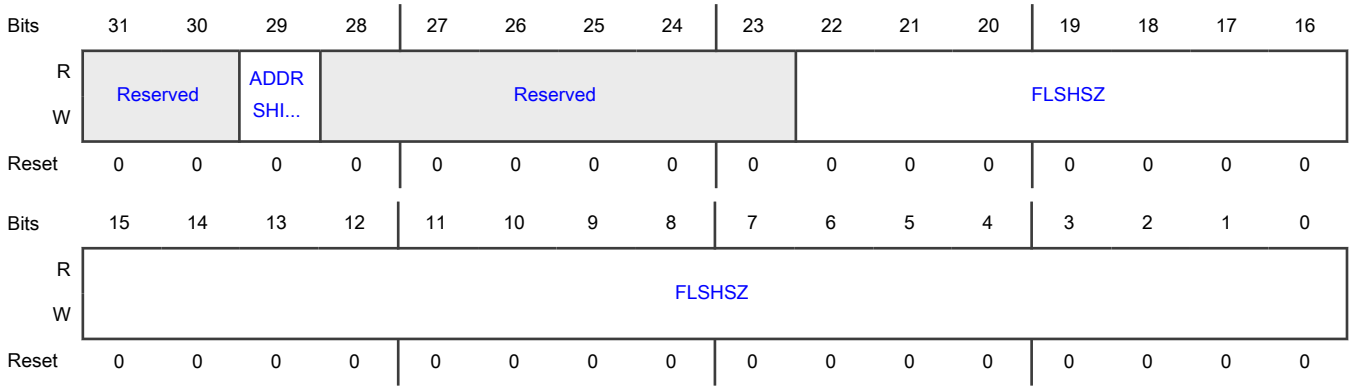
Offset

Register	Offset
FLSHA1CR0	60h
FLSHA2CR0	64h
FLSHB1CR0	68h
FLSHB2CR0	6Ch

Function

Contains flash memory size setting. FlexSPI determines which device is accessed (Chip Select) via this register.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 ADDRSHIFT	AHB Address Shift Function control Used to left-shift five bits for the flash address to support ISSI quad PSRAM. 0b - Disabled 1b - Enabled
28-23 —	Reserved
22-0 FLSHSZ	Flash Size in KB Configures the maximum flash memory size. See the chip-specific section for the maximum flash size supported. When the value of this field is greater than the maximum size, the device flash size is interpreted as the maximum. The maximum flash size is the largest flash size supported for a single device. It is also the maximum total flash size supported for all devices (up to four). If the total flash size is larger than the maximum size, only the maximum flash size address space is accessible. For IPS access, The maximum flash size supported for each device is 4 GB. When the value of this field is greater than 400000h, the device flash size is taken as 4 GB. The max total flash size supported (for all 4 devices) is also 4 GB. If the total flash size is larger than 4 GB, only 4 GB of address space is accessible.

31.7.2.12 Flash Control 1 (FLSHA1CR1 - FLSHB2CR1)

Offset

Register	Offset
FLSHA1CR1	70h

Table continues on the next page...

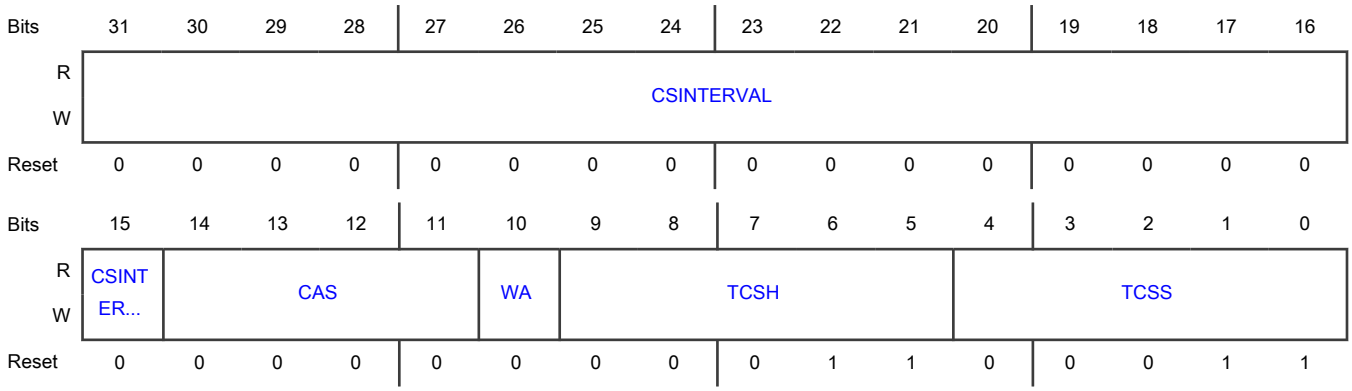
Table continued from the previous page...

Register	Offset
FLSHA2CR1	74h
FLSHB1CR1	78h
FLSHB2CR1	7Ch

Function

Contains settings for flash device-specific timings and flash internal address space.

Diagram



Fields

Field	Function
31-16 CSINTERVAL	Chip Select Interval Configures the minimum interval between flash device chip select deassertion and chip select assertion. If the external flash device has a limitation on the interval between command sequences, configure this field accordingly. If there is no limitation, write 0h to this field. When CSINTERVALUNIT = 0, the chip select invalid interval is: CSINTERVAL × 1 serial clock cycle; When CSINTERVALUNIT = 1, the chip select invalid interval is: CSINTERVAL × 256 serial clock cycles. **NOTE** The minimum chip select interval is 2 cycles, even when the value of CSINTERVAL is less than 2.
15 CSINTERVALUNIT	Chip Select Interval Unit Configures the interval unit for chip select. 0b - 1 serial clock cycle 1b - 256 serial clock cycles
14-11 CAS	Column Address Size

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When external flash memory has a separate address field for rows and columns, this field configures the flash column address bit width. FlexSPI automatically splits a flash-mapped address into row address and column address according to the values of this field and the WA field. When the external flash memory does not support column address, write 0 to this field. FlexSPI transmits all flash address bits as Row address. For flash address mapping, see Flash address sent to flash memory devices.
10 WA	Word-Addressable Configures whether external flash memory is word-addressable or byte-addressable. If flash memory is word-addressable, it should be accessed in multiples of 16 bits. Currently, FlexSPI does not transmit flash address bit 0 to external flash memory. For flash address mapping, see Flash address sent to flash memory devices. 0b - Byte-addressable 1b - Word-addressable
9-5 TCSH	Serial Flash CS Hold Time Used to meet flash TCSH timing requirement. Serial flash CS Hold time that FlexSPI promises is: (TCSH + 1/2) serial clock cycles for DDR mode, and TCSH serial clock cycles for SDR mode. See Output timing between chip select and SCLK.
4-0 TCSS	Serial Flash CS Setup Time Used to meet flash TCSS timing requirement. Serial flash CS Setup time that FlexSPI promises is: (TCSS + 1/2) serial root clock cycles for SDR and DDR mode. See Output timing between chip select and SCLK.

31.7.2.13 Flash Control 2 (FLSHA1CR2 - FLSHB2CR2)

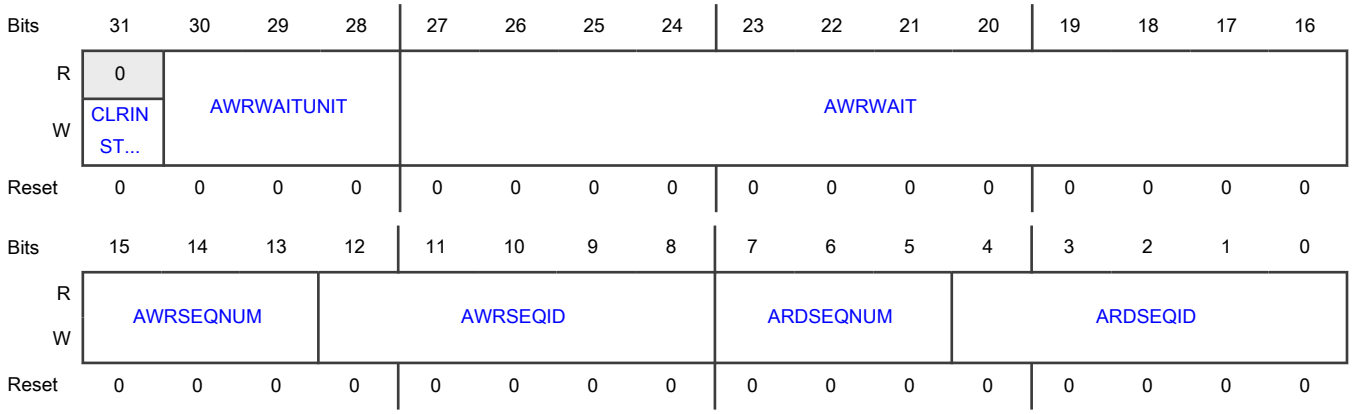
Offset

Register	Offset
FLSHA1CR2	80h
FLSHA2CR2	84h
FLSHB1CR2	88h
FLSHB2CR2	8Ch

Function

Contains fields to configure AHB bus access. If the four external devices are different types, AHB read and write commands may use different command sequences; AHB bus ready wait time may also differ.

Diagram



Fields

Field	Function
31 CLRINSTRPTR	Clear Instruction Pointer Clears the instruction pointer, which is the pointer that JMP_ON_CS saves internally. See Programmable sequence engine. This field is used for AHB read access to external flash memory supporting Execute-In-Place (XIP) mode.
30-28 AWRWAITUNIT	AWRWAIT Unit Configures the unit of AHB write wait time, as the value of AWRWAIT determines, in terms of AHB clock cycles. 000b - 2 001b - 8 010b - 32 011b - 128 100b - 512 101b - 2048 110b - 8192 111b - 32768
27-16 AWRWAIT	AHB Write Wait Configures AHB write wait time, with the AWRWAITUNIT field. Certain devices (such as FPGA) require time to write data into internal memory after the command sequences finished on the FlexSPI interface. If another read command sequence arrives before the current programming finishes internally, the read data may be wrong. This field is used to hold the AHB bus ready for AHB write access, waiting until the programming is finished in the external device. This hold ensures that no AHB read command is triggered before the programming finishes in the external device. The wait cycle between an AHB-triggered command sequence finishing on FlexSPI and the AHB return bus being ready is: AWRWAIT × AWRWAITUNIT.
15-13	Sequence Number for AHB Write-Triggered Command

Table continues on the next page...

Table continued from the previous page...

Field	Function
AWRSEQNUM	Configures the sequence number of an AHB read-triggered command. For certain flash devices (for example, HyperFlash, HyperRam, and Serial NAND flash), a flash programming access is done via several command sequences. An AHB write command triggers (AWRSEQNUM + 1) command sequences to external flash memory each time. FlexSPI executes the sequences in LUT incrementally. **NOTE** • Software should ensure that the last sequence index never exceeds LUT sequence numbers: AWRSEQID + AWRSEQNUM < 32 • Software must ensure that the AWRSEQNUM and LUT fields are configured correctly according to the external device specification. FlexSPI does not check the sequence; it executes the sequences one by one.
12-8 AWRSEQID	Sequence Index for AHB Write-Triggered Command
7-5 ARDSEQNUM	Sequence Number for AHB Read-Triggered Command Configures the sequence number of an AHB read-triggered command in the lookup table. For certain flash devices (for example, HyperFlash, HyperRam, and Serial NAND flash), a flash read access is done via several command sequences. An AHB read command triggers (ARDSEQNUM + 1) command sequences to external flash memory each time. FlexSPI executes the sequences in LUT incrementally. **NOTE** • Software should ensure that the last sequence index never exceeds LUT sequence numbers: ARDSEQID + ARDSEQNUM ≤ 32. • Software must ensure that the ARDSEQNUM and LUT fields are configured correctly according to the external device specification. FlexSPI does not check the sequence; it executes the sequences one by one.
4-0 ARDSEQID	Sequence Index for AHB Read-Triggered Command in LUT

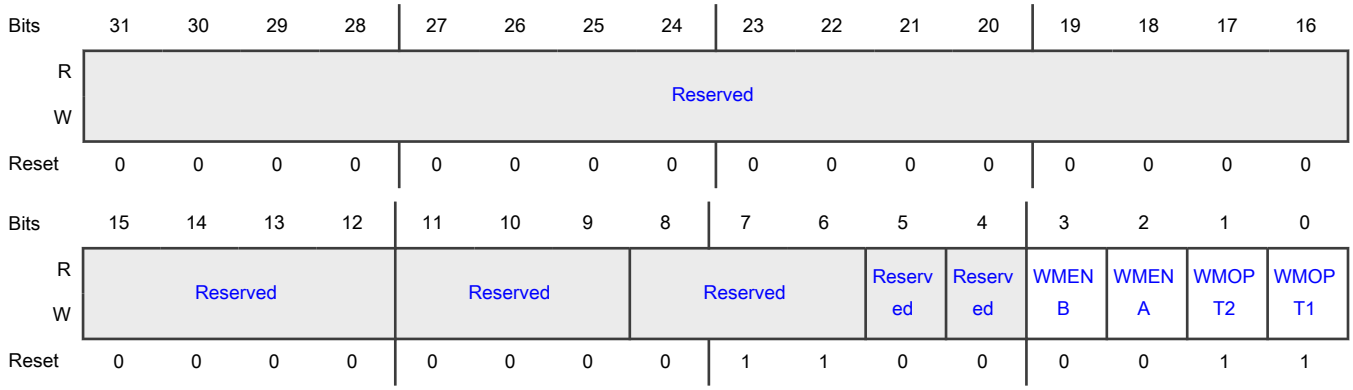
31.7.2.14 Flash Control 4 (FLSHCR4)

Offset

Register	Offset
FLSHCR4	94h

Function
Provides configuration for all external devices.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-9 —	Reserved
8-6 —	Reserved
5 —	Reserved
4 —	Reserved
3 WMENB	Write Mask Enable for Port B Enables write mask for flash device on port B. 0b - Disabled. When writing to external device, DQS(RWDS) pin is not driven. 1b - Enabled. When writing to external device, FlexSPI drives DQS(RWDS) pin as write mask output.
2 WMENA	Write Mask Enable for Port A Enables write mask for flash device on port A. 0b - Disabled. When writing to external device, DQS(RWDS) pin is not driven. 1b - Enabled. When writing to external device, FlexSPI drives DQS(RWDS) pin as write mask output.
1 WMOPT2	Write Mask Option 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When using AP memory, used to remove AHB or IP write burst minimum length limitation. When using this field, FLSHCR4[WMOPT1] should be 1. 0b - When writing to an external device, DQS pin is used as write mask. When flash memory is accessed in individual mode, AHB or IP write burst length is not limited. 1b - When writing to an external device, DQS pin is not used as write mask. When flash memory is accessed in individual mode, AHB or IP write burst length is limited. The minimum write burst length should be 4.
0 WMOPT1	Write Mask Option 1 Used to remove AHB and IP write burst start address alignment limitation. 0b - When writing to an external device, DQS pin is used as write mask. When flash memory is accessed in individual mode, AHB or IP write burst start address alignment is not limited. 1b - When writing to an external device, DQS pin is not used as write mask. When flash memory is accessed in individual mode, AHB or IP write burst start address alignment is limited.

31.7.2.15 IP Control 0 (IPCR0)

Offset

Register	Offset
IPCR0	A0h

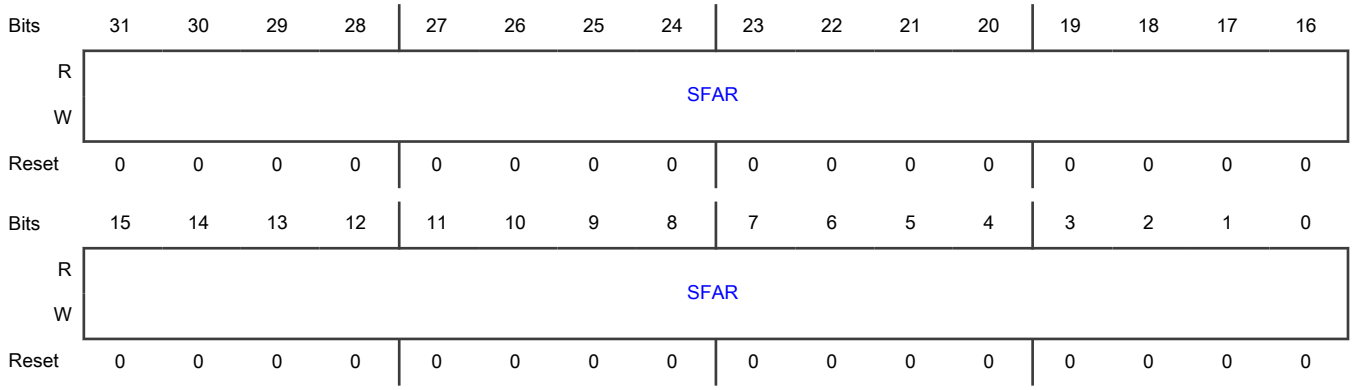
Function

Provides all configuration required for IP commands. Provides the start address for the flash device, instead of the chip address, to be accessed for IP command. FlexSPI determines the chip select automatically according to this start address.

NOTE

- You cannot issue an IP command that crosses flash device boundaries. If you do so, it generates an IPCMDERR interrupt.
- Configure this register before an IP command is triggered.
- Do not change the values in this register while an IP command is in progress.

Diagram



Fields

Field	Function
31-0 SFAR	Serial Flash Address Configures the serial flash address for IP commands. The address should be the address of the flash device without the base address.

31.7.2.16 IP Control 1 (IPCR1)

Offset

Register	Offset
IPCR1	A4h

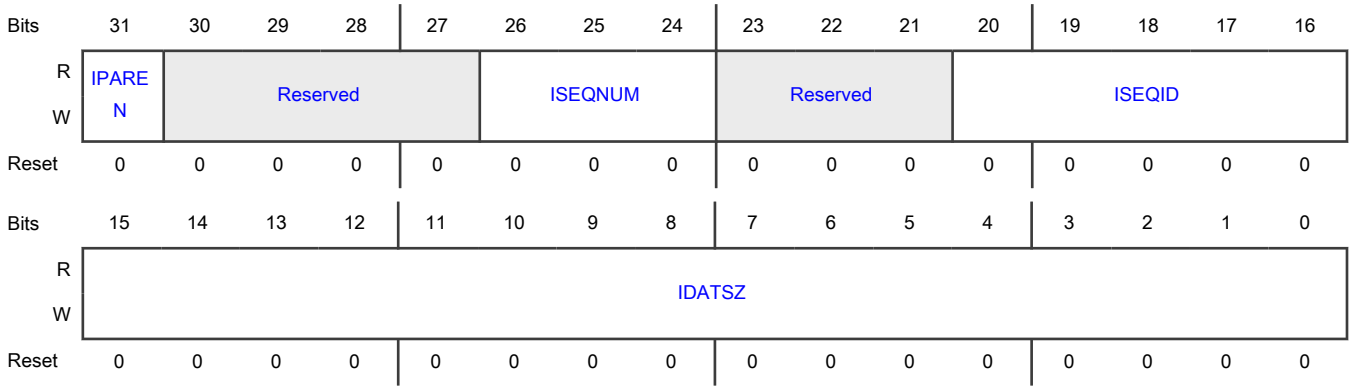
Function

Provides all configuration required for IP commands. Provides the flash read and program data size, sequence index in LUT, sequence number and individual/parallel mode settings for IP commands.

NOTE

- Configure this register before an IP command is triggered.
- Do not change the values in this register while an IP command is in progress.

Diagram



Fields

Field	Function
31 IPAREN	Parallel Mode Enable for IP Commands 0b - Disabled. Flash memory is accessed in Individual mode. 1b - Enabled. Flash memory is accessed in Parallel mode.
30-27 —	Reserved
26-24 ISEQNUM	Sequence Number for IP command: ISEQNUM+1.
23-21 —	Reserved
20-16 ISEQID	Sequence Index in LUT for IP command.
15-0 IDATSZ	Flash Read/Program Data Size (in bytes) for IP command.

31.7.2.17 IP Command (IPCMD)

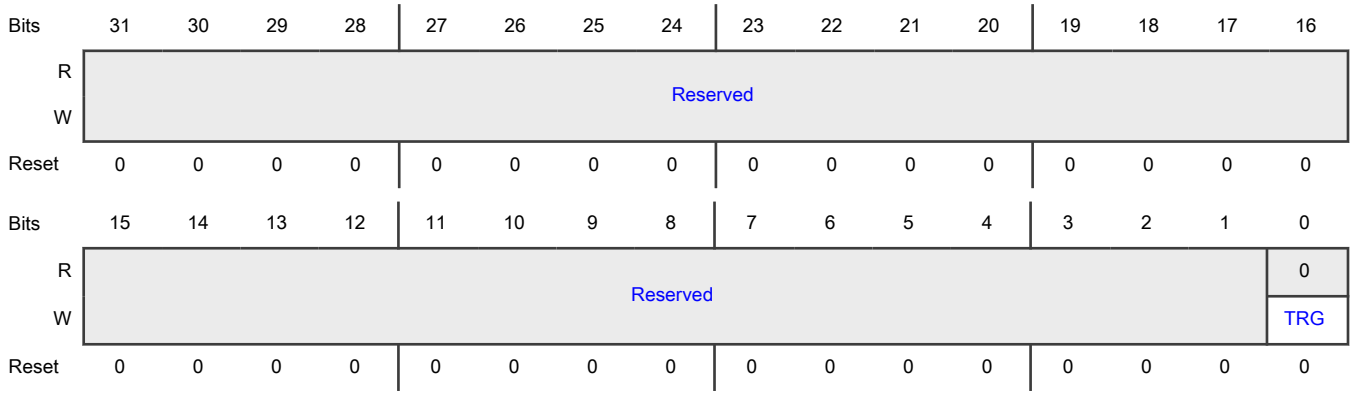
Offset

Register	Offset
IPCMD	B0h

Function

Used to trigger an IP command to access external flash device. When the arbitrator grants the IP command, the command is executed on the FlexSPI interface.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 TRG	Command Trigger Triggers an IP command. This field clears automatically after it has been written and always reads as 0. **NOTE** You cannot trigger another IP command before the previous IP command finishes on the FlexSPI interface. Software must poll INTR[IPCMDDONE] or wait for this interrupt. 0b - No action 1b - Start the IP command that the IPCR0 and IPCR1 registers define.

31.7.2.18 IP Receive FIFO Control (IPRXFCR)

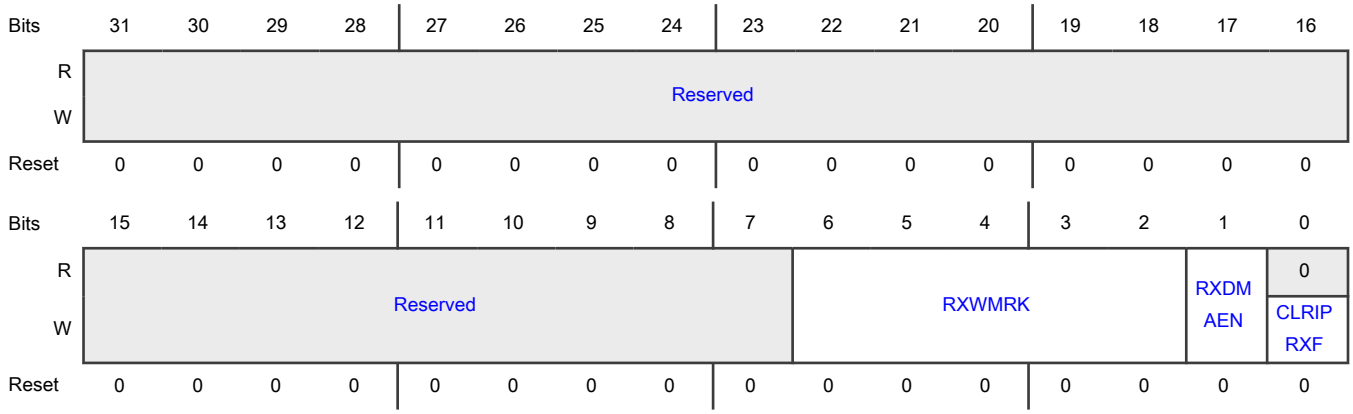
Offset

Register	Offset
IPRXFCR	B8h

Function

Provides the configuration fields for IP receive FIFO management.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-2 RXWMRK	IP Receive FIFO Watermark Level Configures the watermark level for the IP receive FIFO. The watermark level is (RXWMRK + 1) × 64 bits. The INTR[IPRXWA] interrupt is set when FlexSPI fills the IP receive FIFO ≥ the watermark level. A DMA request occurs when the fill level is ≥ the watermark level and IPRXFCR[RXDMAEN] = 1. When the fill level is ≥ the watermark level and INTEN[IPRXWAEN] = 1, an INTR[IPRXWA] interrupt is generated. **NOTE** After writing 1 to INTR[IPRXWA] to clear it, the read address should be rolled back to the start address (memory mapped). If the IP bus reads the IP receive FIFO, the read address should roll back to RFDR0. If the AHB bus reads the IP receive FIFO, the read address should roll back to ARDF_BASE.
1 RXDMAEN	IP Receive FIFO Reading by DMA Enable 0b - Disabled. The processor reads the FIFO. 1b - Enabled. DMA reads the FIFO.
0 CLRIPRXF	Clear IP Receive FIFO Clears all valid data entries in IP receive FIFO. Resets the read and write pointers in IP receive FIFO. 0b - No function 1b - A clock cycle pulse clears all valid data entries in IP receive FIFO.

31.7.2.19 IP Transmit FIFO Control (IPTXFCR)

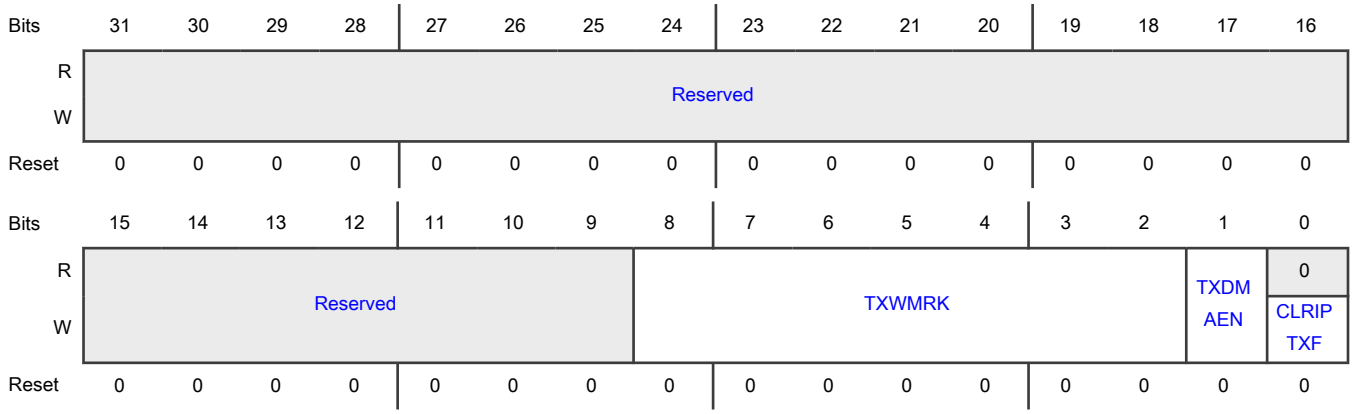
Offset

Register	Offset
IPTXFCR	BCh

Function

Provides the configuration fields for IP transmit FIFO management.

Diagram



Fields

Field	Function
31-9 —	Reserved
8-2 TXWMRK	Transmit Watermark Level Sets the transmit watermark level. The watermark level is (TXWMRK + 1) × 64 bits. INTR[IPTXWE] is set when FlexSPI empties the IP transmit FIFO ≥ the watermark level. When the empty level ≥ the watermark level and IPTXFCR[TXDMAEN] = 1, a DMA request occurs. When the empty level ≥ the watermark level and INTEN[IPTXWEEN] = 1, an INTR[IPTXWE] interrupt is generated. **NOTE** • The watermark level should not be larger than the write window. • The watermark level should not be larger than the IP transmit FIFO size. • The write address to the IP receive FIFO should roll back to the start address of the write window. After pushing the IP transmit fifo, you should write 1 to INTR[IPTXWE] to clear it, which will perform the rollback.
1 TXDMAEN	Transmit FIFO DMA Enable Selects whether DMA or processor fills IP transmit FIFO.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Processor 1b - DMA
0 CLRIPTXF	Clear IP Transmit FIFO Clears all valid data entries in IP transmit FIFO. The read and write pointers in IP transmit FIFO are reset. 0b - No function 1b - A clock cycle pulse clears all valid data entries in the IP transmit FIFO.

31.7.2.20 DLL Control 0 (DLLACR - DLLBCR)

Offset

Register	Offset
DLLACR	C0h
DLLBCR	C4h

Function
Configures Flash A and B sample clock DLL.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved												Reserved			REFPHAS...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	REFPHAS...	OVRDVAL						OVRDEN	Reserved	SLVDLYTARGET			Reserved		DLLRESET	DLLEN
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19-17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16-15 REFPHASEGAP	Reference Clock Delay Line Phase Adjust Gap. REFPHASEGAP setting of 2h is recommended if DLLEN is set.
14-9 OVRDVAL	Target Clock Delay Line Override Value Configures the override value for the target clock line delay cell number. When OVRDEN = 1, the delay cell number in DLL is OVRDVAL + 1. See DLL configuration for sampling.
8 OVRDEN	Target Clock Delay Line Override Value Enable Enables the override value for the target clock line delay cell number. 0b - Disable 1b - Enable
7 —	Reserved
6-3 SLVDLYTARGET	Target Delay Line Configures the target delay line for the target. The delay target for target delay line is: ((SLVDLYTARGET+1) × 1/32 × clock cycle of reference clock (serial root clock). If serial root clock is ≥ 100 MHz, DLLEN is set to 1, OVRDEN is set to 0, then SLVDLYTARGET setting is up to flash parameters used.
2 —	Reserved
1 DLLRESET	DLL reset Forces a DLL reset. The forced reset causes the DLL to lose lock and recalibrate to detect a ref_clock half-period phase shift. The reset action is edge-triggered, so software must write 0 to this field after writing 1 to it (no delay limitation). 0b - No function 1b - Force DLL reset.
0 DLLEN	DLL Calibration Enable Enables DLL calibration. When DLL calibration is disabled, the delay cell number in the target delay line is always 1. When DLLxCR[OVRDEN] = 1, the target delay line is overridden and this field is ignored. 0b - Disable 1b - Enable

31.7.2.21 Status 0 (STS0)

Offset

Register	Offset
STS0	E0h

Function

Indicates the status of the internal state machine.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				Reserved				Reserved				ARBCMDSRC		ARBIDLE	SEQIDLE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-2 ARBCMDSRC	ARB Command Source Indicates the trigger source of current command sequence that the arbitrator has granted. When ARB_CTL is not busy (STS0[ARBIDLE] = 1), the value of this field does not matter. 00b - Trigger source is AHB read command. 01b - Trigger source is AHB write command. 10b - Trigger source is IP command (by writing 1 to IPCMD[TRG]). 11b - Trigger source is a suspended command that has resumed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 ARBIDLE	ARB_CTL State Machine Idle Indicates whether the state machine in ARB_CTL is idle, or a command sequence that the arbitrator granted has not finished on the FlexSPI interface. When idle, no transaction is occurring on the FlexSPI interface (STS0[SEQIDLE] = 1). When waiting for the FlexSPI controller to become idle, poll this field instead of STS0[SEQIDLE]. 0b - Not idle 1b - Idle
0 SEQIDLE	SEQ_CTL State Machine Idle Indicates whether the state machine in SEQ_CTL is idle, or a command sequence is executing on the FlexSPI interface. 0b - Not idle 1b - Idle

31.7.2.22 Status 1 (STS1)

Offset

Register	Offset
STS1	E4h

Function
Indicates the error code of the AHB or IPS interface error response.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				IPCMDERRCODE				Reserved				IPCMDERRID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				AHBCMDERRCODE				Reserved				AHBCMDERRID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 IPCMDERRCODE	IP Command Error Code When an IP command Error is detected, indicates the error code. This field returns to zero when INTR[IPCMDERR] is cleared. 0000b - No error 0010b - IP command with JMP_ON_CS instruction used in the sequence 0011b - Unknown instruction opcode in the sequence 0100b - DUMMY_SDR or DUMMY_RWDS_SDR instruction used in DDR sequence 0101b - DUMMY_DDR or DUMMY_RWDS_DDR instruction used in SDR sequence 0110b - Flash memory access start address exceeds entire flash address range (A1, A2, B1, and B2) 1110b - Sequence execution timeout 1111b - Flash boundary crossed
23-21 —	Reserved
20-16 IPCMDERRID	IP Command Error ID When an IP command error is detected, indicates the sequence index. This field returns to zero when INTR[IPCMDERR] is cleared.
15-12 —	Reserved
11-8 AHBCMDERRCODE	AHB Command Error Code Contains the error code when an AHB command error is detected. This field returns to zero when INTR[AHBCMDERR] is cleared. 0000b - No error 0010b - AHB Write command with JMP_ON_CS instruction used in the sequence 0011b - Unknown instruction opcode in the sequence 0100b - DUMMY_SDR or DUMMY_RWDS_SDR instruction used in DDR sequence 0101b - DUMMY_DDR or DUMMY_RWDS_DDR instruction used in SDR sequence 1110b - Sequence execution timeout
7-5 —	Reserved
4-0	AHB Command Error ID

Table continues on the next page...

Table continued from the previous page...

Field	Function
AHBCMDERRID	When an AHB command error is detected, indicates the sequence index. This field returns to zero when INTR[AHBCMDERR] is cleared.

31.7.2.23 Status 2 (STS2)

Offset

Register	Offset
STS2	E8h

Function

Indicates the status of Flash A and B sample clock DLLs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				BREFSEL				BSLVSEL						BREFLOCK	BSLVLOCK
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				AREFSEL				ASLVSEL						AREFLOCK	ASLVLOCK
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 BREFSEL	Flash B Sample Clock Reference Delay Line Delay Cell Number Indicates the sample clock reference delay line delay cell number for Flash B. There are 0-63 phases. • 000001b - Indicates lock phase 0. • 100000b - Indicates lock phase 63.
23-18 BSLVSEL	Flash B Sample Clock Target Delay Line Delay Cell Number Indicates the sample clock target delay line delay cell number for Flash B. There are 0-63 phases.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 000001b - Indicates lock phase 0. • 100000b - Indicates lock phase 63.
17 BREFLOCK	Flash B Sample Clock Reference Delay Line Locked 0b - Not locked 1b - Locked
16 BSLVLOCK	Flash B Sample Target Reference Delay Line Locked 0b - Not locked 1b - Locked
15-14 —	Reserved
13-8 AREFSEL	Flash A Sample Clock Reference Delay Line Delay Cell Number Indicates the sample clock reference delay line delay cell number for Flash A. There are 0-63 phases. • 000001b - Indicates lock phase 0. • 100000b - Indicates lock phase 63.
7-2 ASLVSEL	Flash A Sample Clock Target Delay Line Delay Cell Number Indicates the sample clock target delay line delay cell number for Flash A. There are 0-63 phases. • 000001b - Indicates lock phase 0. • 100000b - Indicates lock phase 63.
1 AREFLOCK	Flash A Sample Clock Reference Delay Line Locked 0b - Not locked 1b - Locked
0 ASLVLOCK	Flash A Sample Target Delay Line Locked 0b - Not locked 1b - Locked

31.7.2.24 AHB Suspend Status (AHBSPNDSTS)

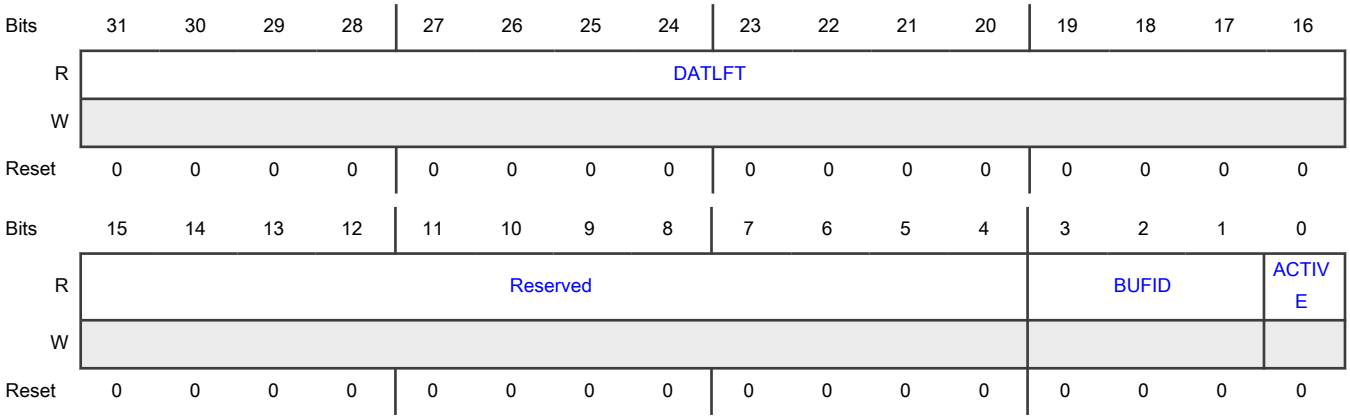
Offset

Register	Offset
AHBSPNDSTS	ECh

Function

Indicates the status of suspended AHB read prefetch command sequence. When an IP or AHB command is triggered while the arbitrator processes an AHB read sequence (prefetching additional data not for current AHB burst), the prefetch sequence is suspended. When there are no longer transactions on FlexSPI, the prefetch sequence may be resumed. FlexSPI saves only one AHB prefetch sequence. When a new prefetch sequence is suspended with an active sequence already suspended, the previous suspended sequence is removed and never resumed. See Command abort and suspend.

Diagram



Fields

Field	Function
31-16 DATLFT	Data Left Contains the data size remaining (in bytes) for a suspended command sequence.
15-4 —	Reserved
3-1 BUFID	AHB Receive Buffer ID for Suspended Command Sequence
0 ACTIVE	Active AHB Read Prefetch Suspended Indicates whether an AHB read prefetch command sequence has been suspended. 0b - No suspended AHB read prefetch command. 1b - An AHB read prefetch command sequence has been suspended.

31.7.2.25 IP Receive FIFO Status (IPRXFSTS)

Offset

Register	Offset
IPRXFSTS	F0h

Function

Indicates the status of IP receive FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RDCNTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								FILL							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 RDCNTR	Read Data Counter Contains counter for total data read. The data read is the value of this field × 64 bits.
15-8 —	Reserved
7-0 FILL	Fill Level of IP Receive FIFO Indicates how full the IP receive FIFO is. The fill level is the value of this field × 64 bits.

31.7.2.26 IP Transmit FIFO Status (IPTXFSTS)

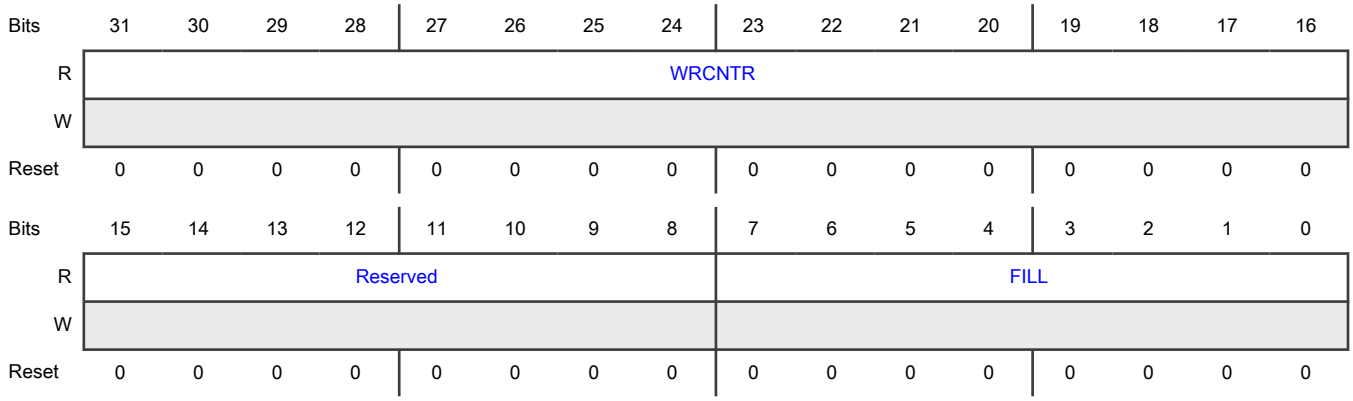
Offset

Register	Offset
IPTXFSTS	F4h

Function

Indicates the status of IP transmit FIFO.

Diagram



Fields

Field	Function
31-16 WRCNTR	Write Data Counter Contains counter for total data written. The data written is the value of this field × 64 bits.
15-8 —	Reserved
7-0 FILL	Fill Level of IP Transmit FIFO Indicates how full the IP transmit FIFO is. The fill level is the value of this field × 64 bits.

31.7.2.27 IP Receive FIFO Data a (RFDR0 - RFDR31)

Offset

For a = 0 to 31:

Register	Offset
RFDRa	100h + (a × 4h)

Function

Provides read access to IP receive FIFO via IPS bus. The read value is unknown for read access to invalid entries in the IP receive FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXDATA	Receive Data Contains receive data. See Reading data from IP receive FIFO.

31.7.2.28 IP TX FIFO Data a (TFDR0 - TFDR31)

Offset

For a = 0 to 31:

Register	Offset
TFDRa	180h + (a × 4h)

Function

Provides write access to IP transmit FIFO via IPS bus.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	TXDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	TXDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXDATA	Transmit Data Contains transmit data. See Writing data to IP transmit FIFO.

31.7.2.29 Lookup Table a (LUT0 - LUT127)

Offset

For a = 0 to 127:

Register	Offset
LUTa	200h + (a × 4h)

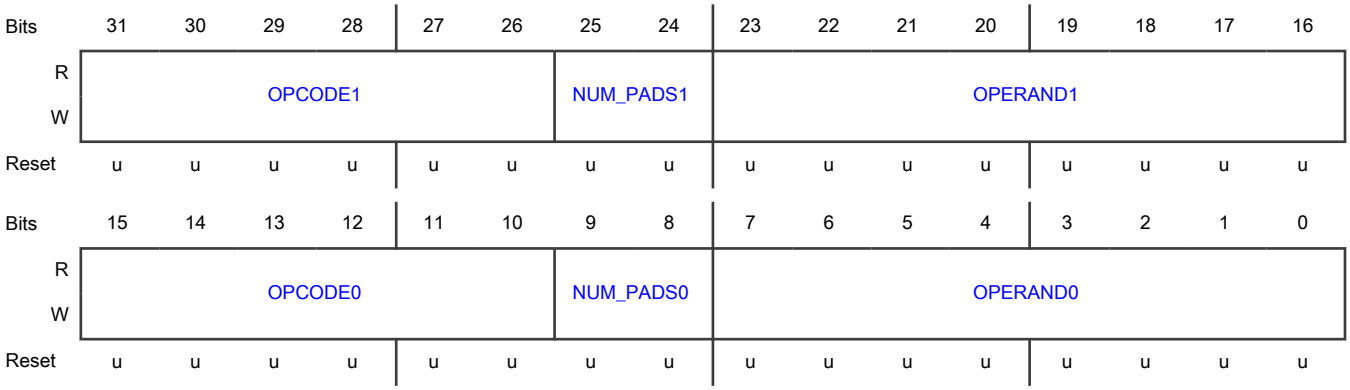
Function

Contains a lookup table for command sequences. Software should set the sequence index before triggering an IP command or AHB command. FlexSPI fetches the command sequence from LUT when an IP or AHB command is triggered. There are 32 command sequences in LUT. See Lookup table (LUT) and Programmable sequence engine.

NOTE

LUT is implemented as memory, so the reset value is unknown.

Diagram



Fields

Field	Function
31-26 OPCODE1	OPCODE1
25-24 NUM_PADS1	NUM_PADS1

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 OPERAND1	OPERAND1
15-10 OPCODE0	OPCODE
9-8 NUM_PADS0	NUM_PADS0
7-0 OPERAND0	OPERAND0

31.7.2.30 Receive Buffer Start Address of Region a (AHBBUFREGIONSTART0 - AHBBUFREGIONSTART3)

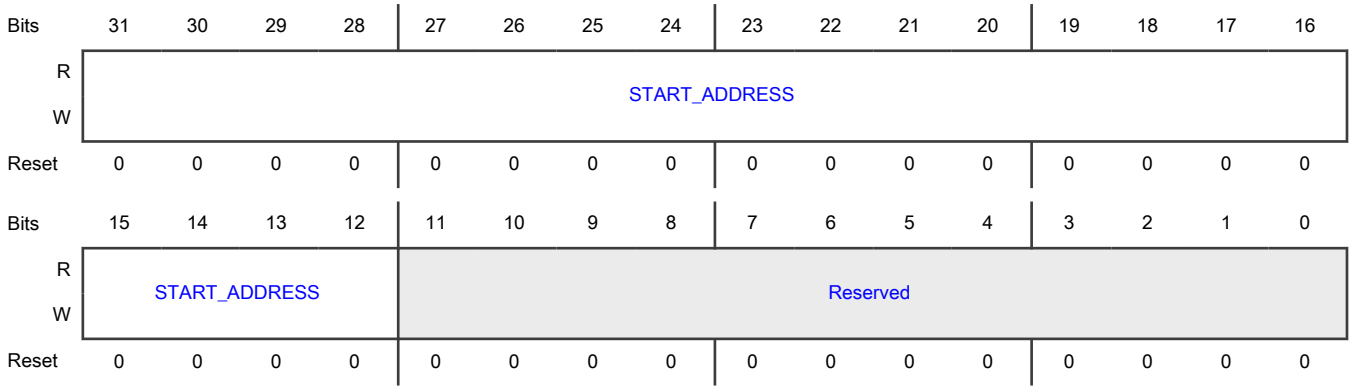
Offset

Register	Offset
AHBBUFREGIONSTART0	440h
AHBBUFREGIONSTART1	448h
AHBBUFREGIONSTART2	450h
AHBBUFREGIONSTART3	458h

Function

Indicates the start address of AHB receive buffer address region.

Diagram



Fields

Field	Function
31-12 START_ADDRESS	Start Address of Prefetch Sub-Buffer Region Indicates start address. The start address must be at least four-KB aligned. The address used here is the AHB address that is compared with the system bus address to determine if an AHB read hits in the region.
11-0 —	Reserved

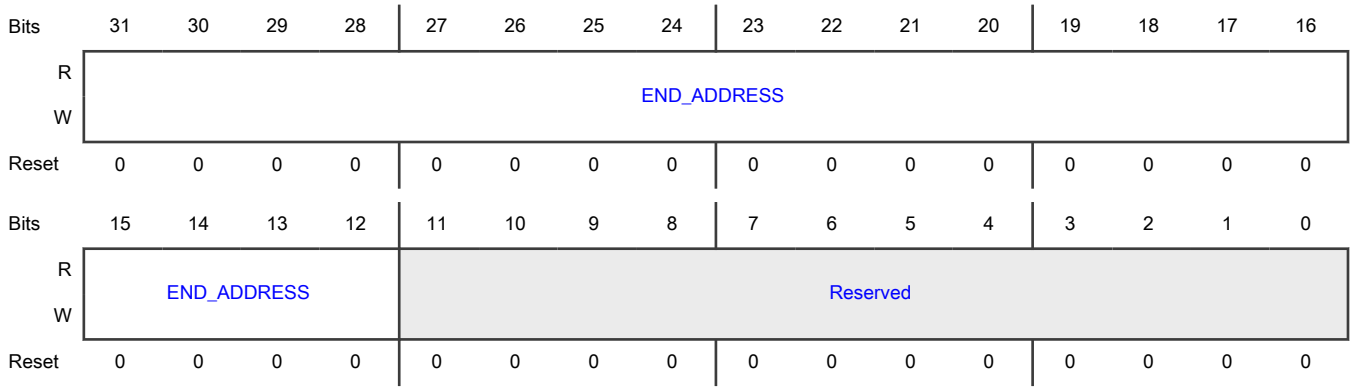
31.7.2.31 Receive Buffer Region a End Address (AHBBUFREGIONEND0 - AHBBUFREGIONEND3)

Offset

Register	Offset
AHBBUFREGIONEND0	444h
AHBBUFREGIONEND1	44Ch
AHBBUFREGIONEND2	454h
AHBBUFREGIONEND3	45Ch

Function
Indicates the end address of AHB receive buffer address region.

Diagram



Fields

Field	Function
31-12	End Address of Prefetch Sub-Buffer Region

Table continues on the next page...

Table continued from the previous page...

Field	Function
END_ADDRESS	Indicates end address. The end address must be at least four-KB aligned. The address used here is the AHB address that is compared with the system bus address to determine if an AHB read hits in the region.
11-0 —	Reserved

31.8 AHB memory map definition

This section describes FlexSPI module AHB memory map in detail.

31.8.1 AHB memory map for serial flash memory and RAM access

AHB read and write access for serial flash and RAM memory are mapped to a specific address range. See the system memory map for specific address ranges supported.

AHB bus features supported for serial flash and RAM memory reading:

- Cacheable and non-cacheable access
- Prefetch enable and disable
- Burst size: 8, 16, 32, 64 bits
- All burst types: types: SINGLE, INCR, WRAP4, INCR4, WRAP8, INCR8, WRAP16, INCR16

AHB bus features for Serial RAM writing:

- Bufferable and Non-Bufferable access
- Burst size: 8, 16, 32, 64 bits
- All burst types: SINGLE, INCR, WRAP4, INCR4, WRAP8, INCR8, WRAP16, INCR16

See Flash memory access via AHB command for details about AHB access to serial flash memory and RAM.

31.8.2 AHB Memory Map for IP RX FIFO read access

See chip-specific section for the address range mapped for AHB read access.

AHB Bus feature supported for IP RX FIFO reading:

- Burst size: 8/16/32/64 bits
- All burst type: SINGLE/INCR/WRAP4/INCR4/WRAP8/INCR8/WRAP16/INCR16

Refer Reading data from IP receive FIFO for more details about IP RX FIFO reading.

31.8.3 AHB Memory Map for IP TX FIFO write access

See chip-specific section for the address range mapped for AHB write access.

AHB Bus feature supported for IP TX FIFO writing:

- Burst size: 8/16/32/64 bits
- All burst type: SINGLE/INCR/WRAP4/INCR4/WRAP8/INCR8/WRAP16/INCR16

Refer Writing data to IP transmit FIFO for more details about IP TX FIFO filling.

Chapter 32
On-The-Fly AES Decryption (OTFAD)

32.1 Chip-specific OTFAD Information

Table 239. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

32.2 Overview

One unmistakable trend in embedded processor design is the increasing need for hardware support of cryptographic calculations required for system security. In particular, there are emerging customer requirements to protect application code and data stored in external memories in an encrypted form and have the embedded processor provide hardware support for "on-the-fly" decryption. As the name suggests, the external memory image of the code and data is always in an encrypted format, and, in response to processor references of the memory space, the memory image is decrypted on the fly, returning the original values to the requesting bus master.

In these applications, the standard Advanced Encryption Standard (AES) [1] is the preferred cryptographic symmetric key block cipher algorithm. The AES function operates on 128-bit (16 byte) data blocks with either 128-, 192- or 256-bit secret keys. For this form of code and data encryption, the AES-128 mode, that is, the use of a 128-bit key is deemed sufficient.

The AES algorithm specifies a variable number of "rounds" be performed in the cryptographic calculation, based on the size of the key. For a 128-bit key, the data is processed through a series of calculations requiring 10 rounds to complete. Each round performs four different data transformations: 1) byte substitution using a substitution table (S-box), 2) shifting rows of the state array by different offsets, 3) mixing the data within each column of the state array, and 4) adding a round key to the state.

Many typical AES hardware implementations perform a single round operation in 1-2 machine cycles, but the performance requirements associated with on-the-fly decryption dictate a far more aggressive implementation.

The on-the-fly decryption engine is combined with the QuadSPI external flash memory controller. The QuadSPI memory controller supports glueless external connections to a wide variety of industry standard external "SPI" flashes. These memories were initially developed as very low pin count alternatives to traditional NOR flash memories with their wide parallel data interfaces. Over time, the complexity of the SPI flash memories has increased (and this trend is continuing) and now, two bank, double data rate interfaces can supply 16 bits of flash data every cycle once the address and command have been sent and the basic access

has been performed. For a 100 MHz DDR Quad SPI external interface, the access time for a 4-beat, 64-bit, 32-byte burst fetch might generate an 18-4-4-4 response at the microprocessor pins. Given these access times, the latest generation of Quad SPI external flash devices provides a cost effective non-volatile memory solution that supports eXecute-in-Place (XiP) capabilities. Stated differently, the combined access speed of these external flash memories, coupled with internal processor-local cache memories, allow the application code to be executed directly from the external memory without the need to copy the code into another (faster) memory. This capability to execute directly from external flash without copying the code to another memory allows an easy migration from MCU to MPU configurations.

The resulting OTFAD engine adds ***zero cycles of additional latency*** to decrypt the code and data fetched from the external flash memory. The combination of the support for the CTR-AES128 mode plus an aggressive, pipelined implementation of the OTFAD performing up to 3 rounds in a single machine cycle allowing the engine to keep pace with the fastest data arrival rates. As a result, the OTFAD engine provides superior cryptographic decryption capabilities without compromising system performance for those embedded processor applications requiring this class of enhanced security.

32.2.1 Block diagram

Figure 229 shows the connection topology of the OTFAD and its relationship to the QuadSPI and its AHB RAM buffer.

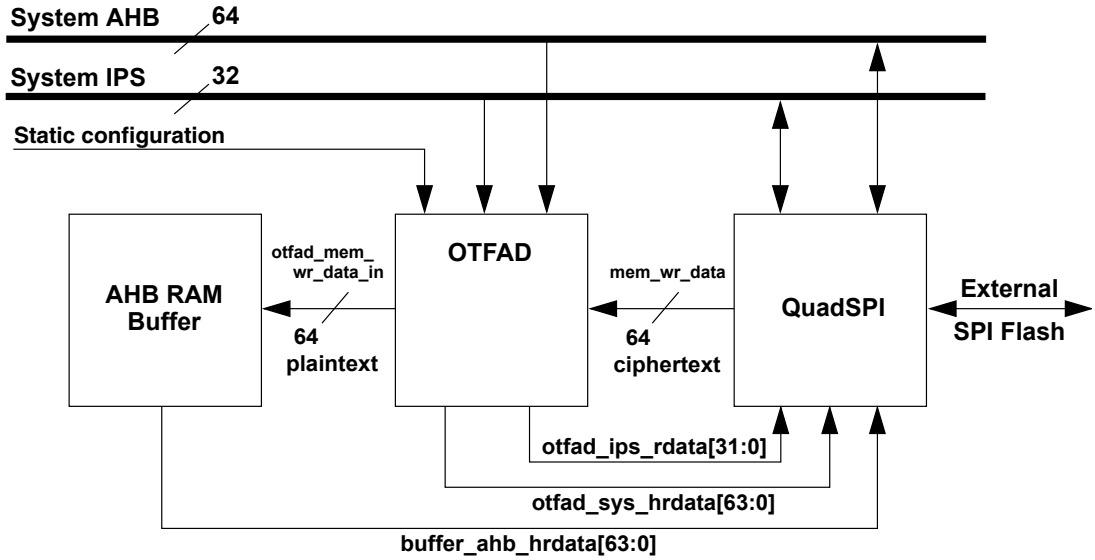


Figure 229. OTFAD and QuadSPI connection topology

32.2.2 Features

The key features of the OTFAD include:

- AES-128 Counter Mode On-the-Fly Decryption
  - 128-bit key and 128-bit data block sizes
    - 128-bit counter includes 64 bits of initialization vector + the 32-bit system address
  - Adds zero cycles of incremental latency for decryption when used with QuadSPI
  - Receives 64-bit encrypted data from QuadSPI, calculates decrypted data which is sent to AHB RAM buffer and bypassed back to system AHB read data bus
- Hardware support for 4 independent decryption segments, known as memory “contexts”
  - Each context has a unique 128-bit key, 64-bit counter and 64-bit memory region descriptor
- Hardware support for unwrapping "key blobs"

  - Key blob data structure prepended to the external flash memory image
  - Implements key blob unwrapping as defined by RFC3394 [3]
  - Each key blob data structure includes all the context registers
  - Initiated under software control after system reset is negated, if properly enabled
  - Unwraps the key blob data structure and loads the contents into the OTFAD registers
- Functionally acts as a slave submodule to the QuadSPI
  - Logically connected between the QuadSPI and its AHB RAM buffer
  - Private 64-bit data buses for encrypted (ciphertext) and decrypted (plaintext) data
- Hardware microarchitecture
  - Heavily pipelined AES engine, optimized for encryption, performing 3 rounds per cycle
  - 64-bit AHB connections for easy integration to system bus fabric and QuadSPI
  - Data storage for two 128-bit encrypted counters and three 64-bit decrypted data buffers
  - Optimized for {32,64}-bit WRAP4 bursts (CPU cache miss fetch size, typical DMA fetch size)

32.2.3 References and terminology

These documents provide specific references to the cryptographic functions supported by the OTFAD:

1. FIPS Publication 197, "Advanced Encryption Standard (AES)", U.S. Department of Commerce, National Institute of Standards and Technology (NIST), November 26, 2001.
2. NIST Special Publication 800-38A, "Recommendation for Block Cipher Modes of Operation: Methods and Techniques", U.S. Department of Commerce, National Institute of Standards and Technology (NIST), December 2001.
3. http://www.ietf.org/rfc/rfc3394.txt, "Advanced Encryption Standard (AES) Key Wrap Algorithm (RFC3394)", J. Schaad, R. Housley, September 2002.

There are numerous names and phrases that are commonly used in describing cryptographic functions. The following table provides a summary of the terms used in describing the OTFAD. It was mostly extracted from Section 4.1, "Definitions and Abbreviations" of [2].

Table 240. Cryptographic terms

Term	Definition
Block Cipher	A family of functions and their inverse functions that is parameterized by cryptographic keys; the functions map bit strings of a fixed length to bit strings of the same length.
Block Size	The number of bits in an input (or output) block of the block cipher. For OTFAD, this is 128 bits (16 bytes).
Ciphertext	Encrypted data.
CTR	Counter.
Cryptographic Key	A parameter used in the block cipher algorithm that determines the forward cipher operation and the inverse cipher operation.
Data Block (Block)	A sequence of bits whose length is the block size of the block cipher.

Table continues on the next page...

Table 240. Cryptographic terms (continued)

Term	Definition
Decryption (Deciphering)	The process of a confidentiality mode that transforms encrypted data into the original usable data.
ECB	Electronic Codebook.
Encrypted Counter (ECTR)	An output in the CTR-AES decryption process, generated by encrypting the unique counter value. Exclusive-OR'ed with the ciphertext to produce plaintext.
Encryption (Enciphering)	The process of a confidentiality mode that transforms usable data into an unreadable form.
Forward Cipher Function (Forward Cipher Operation)	One of the two functions of the block cipher algorithm that is selected by the cryptographic key.
Initialization Vector (IV)	A data block that some modes of operation require as an additional initial input.
Input Block	A data block that is an input to either the forward cipher function or the inverse cipher function of the block cipher algorithm.
Inverse Cipher Function (Inverse Cipher Operation)	The function that reverses the transformation of the forward cipher function when the same cryptographic key is used.
Key Encryption Key (KEK)	Cryptographic key used to wrap and unwrap key data.
Key Wrap	Key Wrap constructions are a class of symmetric encryption algorithms designed to encapsulate (encrypt) cryptographic key material. The constructions are typically built from standard primitives such as block ciphers and cryptographic hash functions.
Least Significant Bit(s)	The right-most bit(s) of a bit string.
Mode of Operation (Mode)	An algorithm for the cryptographic transformation of data that features a symmetric key block cipher algorithm.
Most Significant Bit(s)	The left-most bit(s) of a bit string.
Nonce	A value that is used only once.
OTP key, otp_key	One-time programmable key
Output Block	A data block that is an output of either the forward cipher function or the inverse cipher function of the block cipher algorithm.
Plaintext	Usable data that is formatted as input to a mode.

Table continues on the next page...

Table 240. Cryptographic terms (continued)

Term	Definition
RFC3394	The Advanced Encryption Standard (AES) Key Wrap Algorithm. Specifically defined to implement a key wrapping and unwrapping algorithm and originally intended for use by the Internet community.

32.3 Functional description

This section provides more details on the operation and implementation of the OTFAD.

32.3.1 Modes of operation

The OTFAD supports modes of operation as defined by input configuration (see Static configuration signals) and control signals. The operating mode is signaled in the SR and includes:

1. Logically Disabled Mode (LDM)

   This mode is entered immediately after reset by the assertion of a static configuration input signal, which is treated as the highest priority configuration signal. While in this mode, the OTFAD is logically disabled. Input data from the QuadSPI is simply passed through to the AHB RAM buffer and no data decryptions are performed. The OTFAD's programming model is inaccessible and all attempted accesses are error terminated. If the static configuration input signal is asserted, the OTFAD is always in this mode.

2. Normal Mode (NM)

   This mode is defined when the OTFAD is not in the logically disabled mode. In this mode, the OTFAD functions normally and is capable of performing data decryptions and/or data bypasses as it responds strictly to the memory transactions on its connected system buses. It should be noted the OTFAD operates in this mode regardless of the state of CR[GE]. The assertion of the global enable is required when the OTFAD is to perform data decryptions; if CR[GE] = 0, then the OTFAD simply bypasses all data fetched by the QuadSPI.

   The OTFAD's operating mode is encoded and visible in the SR[MODE] field. For more information, see OTFAD operating modes.

   In normal mode, as a memory-mapped device located in the core platform’s clock domain, it responds based on the memory addresses of its connected system buses. The slave peripheral bus may be used to access the OTFAD’s programming model to configure the module’s operation.

32.3.2 CTR-AES128 mode basics

When processing data fetched from an external memory, the OTFAD implements the CTR-AES128 mode for on-the-fly decryption.

The OTFAD engine implements a block cipher mode of operation, specifically supporting counter mode (CTR). The CTR mode "provides a confidentiality mode that features the application of the forward cipher (encryption) to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa." [2]

The CTR-AES128 mode was selected for two reasons: 1) it produces a cryptographically stronger solution than the simple Electronic Cookbook (ECB) mode since each block in the sequence is different than every other block, even for the same input data, and 2) the definition of CTR mode allows the 128-bit counter values to include the system memory address of the referenced flash block and to be computed in advance of their actual usage to convert the ciphertext back into plaintext for the requesting bus master device in the microprocessor - typically the CPU itself.

As a result, the OTFAD module adds zero cycles of additional latency to decrypt the code and data fetched from the external flash memory. The combined support for the CTR-AES128 mode plus an aggressive, pipelined implementation of the OTFAD performing up to 3 rounds in a single machine cycle allows the engine to keep pace with the fastest data arrival rates.

The OTFAD module performs the decryption operation within the overall CTR-AES128 flow, as shown in Figure 230.

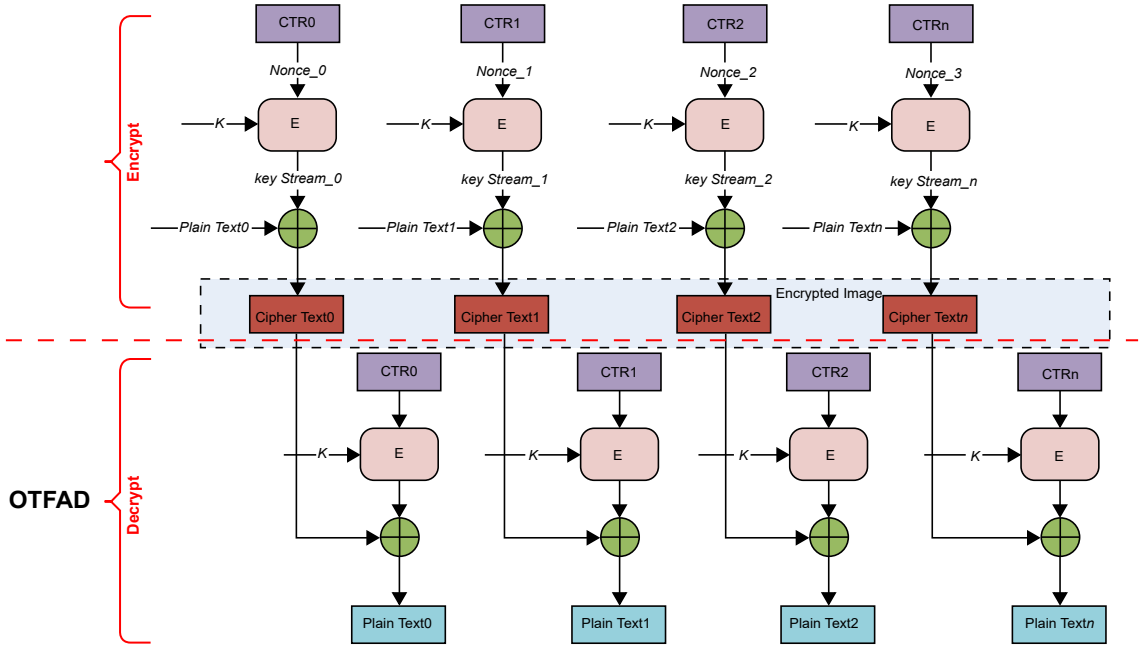


Figure 230. CTR-AES128 encrypt and decrypt overview

As shown in Figure 230, CTR-AES128 scheme combines a unique 128-bit counter (CTR*n*) with a 128-bit key (K) to encrypt (Ciphertext*n*) or decrypt (Plaintext*n*) each 128 bit data block.

Authorized software executes the "encrypt" portion to create the ciphertext image stored in the external memory, while the OTFAD module, in conjunction with the QuadSPI external memory controller, executes the "decrypt" portion.

Consider these two basic operations in more detail.

For encryption, the 0-modulo-16 byte system address is combined with 96 bits of counter information defined in the memory context registers to produce a unique 128-bit CTRn value for each data block. Specifically, the OTFAD's implementation defines the 128-bit CTR*n* for memory context "x" as the concatenation of 4 values:

```
CTRn_x[127-0] = {CTR_W0_x[C0...C3],                      // 32 bits of pre-programmed CTR

                 CTR_W1_x[C4...C7],                      // another 32 bits of CTR

                 CTR_W0_x[C0...C3] ^ CTR_W1_x[C4...C7],  // exclusive-OR of CTR values

                 systemAddress[31-4], 0000b}             // 0-modulo-16 system address
```

NOTE

The addresses used by both (software) encryption and OTFAD decryption are the physical system addresses associated with the QuadSPI, never the code alias addresses.

Software uses this CTR*n* value as the data input to the AES128 encryption process (E) to produce the Key_Stream_*n* output. The final step in encryption is to exclusive-OR the Key_Stream_n value with the data block's plaintext to create the 128-bit block of Ciphertextn which is stored in the external memory. This process is then repeated for each 16-byte data block of the code and data image needing encryption.

The OTFAD's decryption process begins by inputting the same CTRn value and same key K to the AES128 encryption process (E) to generate a 128-bit encrypted counter (ECTR*n*, not explicitly labeled in Figure 230). The ECTR*n* is exclusive-OR'ed with the Ciphertext*n* to produce the Plaintext*n*. The 128-bit key (K) is defined in the KEY_W[0-3]_x registers for context "x".

Since the encrypted counter value is *not* dependent on the ciphertext, it can be pre-calculated in response to a system bus access request such that the OTFAD only needs to perform the exclusive-OR once the ciphertext has been fetched from the external memory. This is the key concept in allowing the OTFAD to add zero incremental cycles of fetch latency.

For additional details on the OTFAD's processing during a 32-byte burst read, see Typical OTFAD CTR-AES128 decryption operation.

32.3.3 Microarchitecture overview

This section provides additional details on the hardware implementation. The following subsections provide details on the hardware decryption operations, and key blob processing. Refer to the block diagram of Figure 229.

32.3.3.1 Context determination

The first step in a CTR-AES128 decryption is the determination of the external memory context.

The OTFAD supports four memory contexts where each has its own 128-bit key, 64-bits of counter and a 64-bit memory region descriptor. The region descriptor includes start and end addresses plus three bits of control and configuration. See AES Region Descriptor Word0 (CTX0_RGD_W0 - CTX3_RGD_W0) and AES Region Descriptor Word1 (CTX0_RGD_W1 - CTX3_RGD_W1) for details.

NOTE

Always use an address range within the physical address space assigned to the QuadSPI module when defining an external memory context in a region descriptor, even if the system application uses the aliased area of the QuadSPI module.

The system address (SYSADDR) is compared to the start and end addresses of each context to determine a region "hit". To determine if the current reference hits in a given context, two magnitude comparators are used in conjunction with the region's start and end addresses. The boolean equation for this portion of the hit determination is defined as:

```
context_hit_n
      = ((addr[31:10] >= rgd_w0_n[srtaddr]) & (addr[31:10] <= rgd_w1_n[endaddr]))
      & rgd_w1_n[vld]
```

where `addr[*]` is the current system reference address, `rgd_w0_n[srtaddr]` and `rgd_w1_n[endaddr]` are the start and end addresses, and `rgd_w1_n[vld]` is the valid bit, all from context region descriptor n.

There are no hardware checks to verify that `rgd_w1_n[endaddr] ≥ rgd_w0_n[srtaddr]`, and it is software's responsibility to load appropriate values into these fields of the context region descriptor.

Recall the context memory regions are defined as modulo-1024 byte values to match the AMBA-AHB protocol requirement that no bursting transfer cross that boundary. The result is that each AHB transfer, regardless of access size or length, is limited to a single context for purposes of AES decryption.

Since each context has unique key and counter values, the OTFAD does not support any form of memory region overlap. For system accesses that hit in multiple contexts or no contexts, the fetched data is simply bypassed.

For accesses that hit in a single context, there is a separate register bit (RGD_W1_n[ADE]) that enables AES decryption or a simple bypass of the fetched data. Based on the hit determination and the state of the ADE indicator, the appropriate 128-bit key and 64 bit counter are selected from the context registers.

32.3.3.2 AES engine and encrypted counter generation

Once the appropriate context has been determined and the corresponding key and counter values selected, the OTFAD proceeds with the generation of the encrypted counters. Since the dominant QuadSPI system bus reference is a 64-bit WRAP4 burst transfer in response to a cache miss, the OTFAD is optimized for this transaction type. The subsequent discussion is described in terms of the 64-bit WRAP4 transfer.

As the 64-bit WRAP4 accesses 32 bytes of data, the OTFAD calculates two 128-bit encrypted counters since there are two AES128 data blocks referenced by this transaction. Since external flash SPI memories typically do not support any type of

address wrapping, the QuadSPI forces the external fetches to begin at the corresponding 0-modulo-size address. Accordingly, the OTFAD calculates the (first) encrypted counter associated with system address 0-modulo-32, and then the (second) address 16-modulo-32. The two encrypted counter calculations are performed in the AES Engine as the QuadSPI is making the external flash fetches.

As each encrypted counter is generated, the 128-bit value is stored in one of the two data registers connected to the AES Engine output.

Recall the OTFAD's AES Engine is heavily pipelined, performing up to 3 rounds per cycle, so the encryption speed (4 cycles total) matches the fastest data arrival rate.

Each AES round performs 4 different data transformations:

- Byte substitution
- State array row shift
- State array column mix
- Round key addition

Recall the QuadSPI initiated the external fetch at a 0-modulo-32 address, so as the encrypted data is fetched and presented to the OTFAD, it uses the first encrypted counter to perform the required exclusive-OR operation to generate the corresponding plaintext. The generated plaintext is output to the QuadSPI's AHB RAM buffer to be written into that memory (which operates as a small cache for QuadSPI accesses). See Figure 229. This process continues as each of the four fetches of the 64-bit WRAP4 burst is presented to the OTFAD, decrypted and then sent to the AHB RAM buffer.

32.3.3.3 Decrypted data buffer operation

Although the QuadSPI initiated its external fetches from a 0-modulo-32 address, the system bus transaction does not have any restrictions about its starting address and can begin at any of the 64-bit addresses. So, as the sequential fetched data is decrypted by the OTFAD, the resulting plaintext is also loaded into three 64-bit decrypted data buffers.

Depending on the original address of the system bus reference, the OTFAD either decrypts and outputs data to *both* the AHB RAM buffer and the system bus, or registers the decrypted data so it can be sourced later after the system bus address "wraps". Consider the following two examples: first, examine the sequencing on a 64-bit WRAP4 with an original address of 0x00, and a second example with a starting address of 0x18.

In both examples, the QuadSPI's fetch addresses are {0x00, 0x08, 0x10, 0x18}. In the first example, the system bus addresses are {0x00, 0x08, 0x10, 0x18}, while in the second example, the sequence is {0x18, 0x00, 0x08, 0x10}. See Table 241 for a description of the operations for these two examples.

Table 241. OTFAD and decrypted data buffer operation

QuadSPI fetch	SystemBus fetch	Description
Example 1		**64-bit WRAP4, miss @ 0×00**
0×00	0×00	OTFAD decrypts data and simultaneously sends to AHB RAM Buffer and SystemBus
0×08	0×08	OTFAD decrypts data and simultaneously sends to AHB RAM Buffer and SystemBus
0×10	0×10	OTFAD decrypts data and simultaneously sends to AHB RAM Buffer and SystemBus
0×18	0×18	OTFAD decrypts data and simultaneously sends to AHB RAM Buffer and SystemBus
Example 2		**64-bit WRAP4, miss @ 0×18**

Table continues on the next page...

Table 241. OTFAD and decrypted data buffer operation (continued)

QuadSPI fetch	SystemBus fetch	Description
0×00	0×18	OTFAD decrypts data, sends to AHB RAM Buffer and loads into Decrypted Data Buffer 0, SystemBus read is stalled
0×08	0×18	OTFAD decrypts data, sends to AHB RAM Buffer and loads into Decrypted Data Buffer 1, SystemBus read is stalled
0×10	0×18	OTFAD decrypts data, sends to AHB RAM Buffer and loads into Decrypted Data Buffer 2, SystemBus read is stalled
0×18	0×18	OTFAD decrypts data and simultaneously sends to AHB RAM Buffer and SystemBus
	0×00	OTFAD sources data from decrypted data buffer 0
	0×08	OTFAD sources data from decrypted data buffer 1
	0×10	OTFAD sources data from decrypted data buffer 2

The resulting behavior from the combined QuadSPI + OTFAD provides the best system performance. For longer burst sizes, for example, 64-bit WRAP8 and 64-bit WRAP16, the implemented number of decrypted data buffers is not sufficient to temporary store the entire burst. In these cases, the three OTFAD buffers are used in conjunction with "hits" from the AHB RAM Buffer to source the decrypted data back to the system bus.

A simplified (partial) timing diagram for the 64-bit WRAP4 burst of Example 1 is shown in Figure 231.

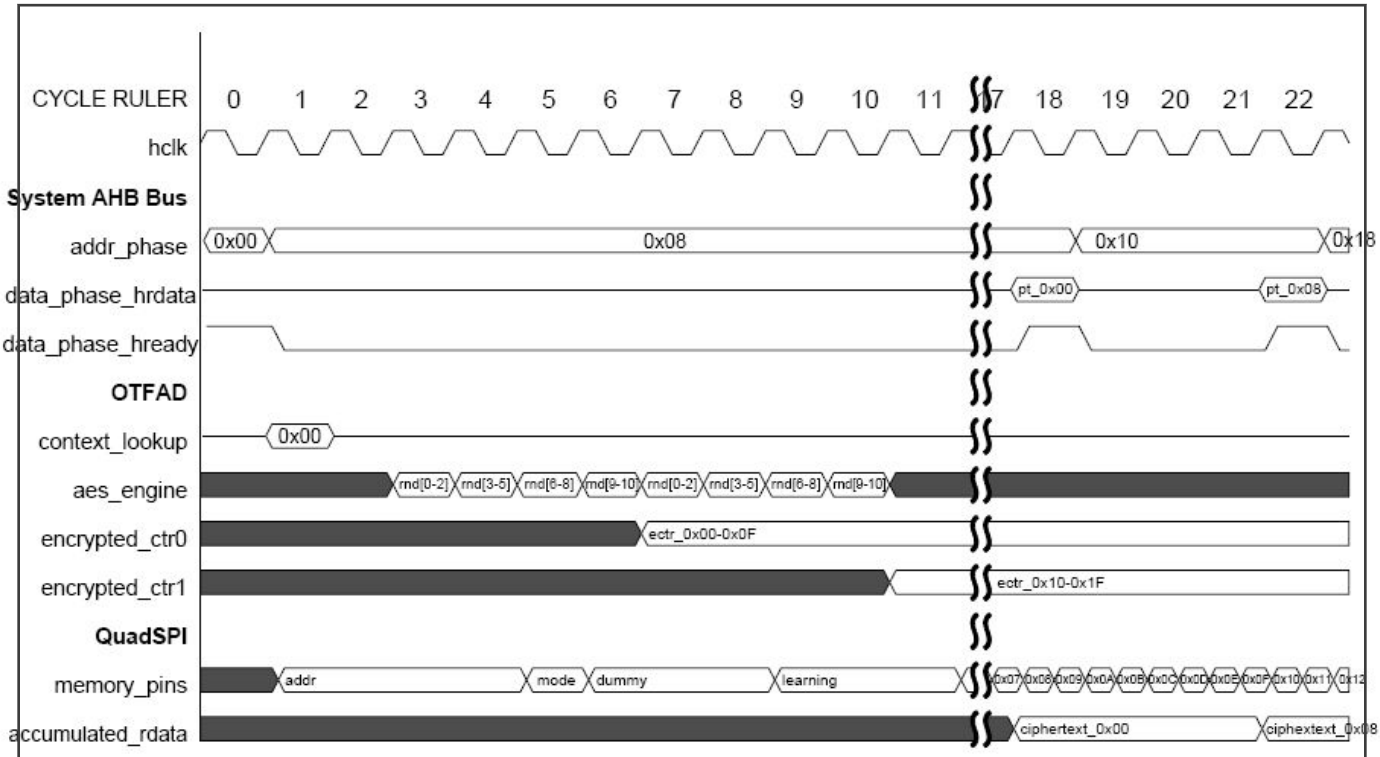


Figure 231. OTFAD 64-bit WRAP4 burst timing diagram

In Figure 231, the system AHB read data bus "pt_<burst_offset>" is meant to refer to decrypted plaintext data. The zero cycle incremental delay to perform the required CTR-AES128 decryption is visible in cycles 18 (the doubleword at burst offset 0×00) and 22 (doubleword at burst offset 0×08).

32.3.4 Clocking

This module has no special clocking considerations.

32.3.5 Interrupts

This module has no interrupts.

32.4 External signals

The OTFAD module does not directly support any external interfaces.

The internal interfaces include the address and attribute input signals from a standard 64-bit AMBA-AHB system bus and a 32-bit IPS slave peripheral bus for all programming model accesses, as shown in Figure 229.

The OTFAD receives the IPS input signals and generates a read data output which is sent to the QuadSPI where it is combined with local signals to form an IPS read data output as well as bus termination control signals. There are also three private 64-bit data buses: one input from the QuadSPI typically providing ciphertext and two outputs containing plaintext data after decryption - one sent to the AHB RAM buffer and another sent back to the QuadSPI for forwarding onto the system AHB read data bus. There is also a 64-bit read data bus from the AHB RAM buffer back to the QuadSPI for servicing system bus accesses which "hit" in the buffer.

32.4.1 Static configuration signals

The specific value settings of the static configuration signals are shown in Table 242.

Table 242. Static configuration signal settings

Configuration Signal	Description	Value
otfad_enable	Enable OTFAD decryption	connected
key_blob_enable	Enable key blob unwrap	connected
enb_otfad_kek_scramble	Enable KEK scramble during key blob processing	connected
restrict_otfad_ips	Restrict access to CR, SR	connected

32.5 Initialization

32.5.1 OTFAD configuration details

The operating configuration of the OTFAD is controlled by a number of static input signals (see Static configuration signals) plus programmable bits in the Control Register. All of the basic modes are discussed in Modes of operation and OTFAD operating modes. This section focuses on the configuration capabilities associated with Normal mode (SR[MODE] = 00).

There are three static input signals and three associated CR bits that define various configuration features of key blob processing. In all cases, the static input signal and the associated CR bit are logically summed (OR'd) together to form an “effective” configuration bit. The combined control is visible via a slave peripheral bus CR read. For a hierarchical view of increasing normal mode capabilities, see Table 243.

Table 243. OTFAD Normal mode configurations

otfad_enable | CR[GE]	key_blob_enable | CR[KBPE]	enb_otfad_kek_scramble | CR[KBSE]	Description
0	—	—	1) OTFAD operates with all data decryption disabled; QuadSPI fetched data is simply bypassed.
1	0	—	2) Basic OTFAD processing is enabled; decryptions are enabled based on context n “hit” and RGD_W1_n[ADE, VLD].
1	1	0	3) Same as #2, plus key blob unwrapping is enabled; KEK = otp_key[127:0].
1	1	1	4) Same as #3 but with KEK scrambling enabled; KEK = f(otp_key[127:0],key_scramble[31:0] key_scramble_align[7:0]).

In addition to the static configuration inputs and associated CR bits shown in Table 243 , there is one additional set: a static configuration input and CR[RRAE]. When either the configuration input signal or CR[RRAE] is asserted, only the CR and SR registers can be accessed; attempted accesses of all other registers are treated as read-as-zero, write-ignore.

For all configuration bits, the intended use case is to layer increasing OTFAD functionality into the system, first using the programmable CR indicators, and then, after the behavior has been fully debugged, to program the corresponding static input signals to define a “permanent” configuration. In the limit, the OTFAD fully initializes all four contexts via its key blob processing and requires no other system programming or interaction.

32.5.2 Key blob processing

32.5.2.1 Overview

Key blob processing is a mechanism where a preloaded data structure, wrapped using the RFC3394 standard, is fetched from the external flash memory after a system reset event (if properly enabled and initiated by the assertion of CSR[SKBP]), unwrapped by the OTFAD hardware and automatically loaded into the four memory context programming model registers. In this manner, the sensitive context data, for example, the 128-bit key and 64 bits of the counter, is unwrapped and handled entirely within the OTFAD.

32.5.2.2 Data structure

Each key blob data structure contains a 128-bit key, 64 bits of counter, a 64-bit memory region descriptor per context. See Figure 232. A C language description of the key blob data structure is presented in Figure 233.

SPI flash base + 64*n offset, n = 0-3

	31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10	9	8	7	6	5	4	3	2	1	0
0x00	KEY[A03...A00]															
0x04	KEY[A07...A04]															
0x08	KEY[A11...A08]															
0x0C	KEY[A15...A12]															
0x10	CTR[C03...C00]															
0x14	CTR[C07...C04]															
0x18	SRTADDR[31 – 10]						0	0	0	0	0	0	0	0	0	0
0x1C	ENDADDR[31 – 10]						1	1	1	1	1	1	1	RO	ADE	VLD
0x20	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0	0	0	0	0	0	0	0	0	0	0
0x24	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0	0	0	0	0	0	0	0	0	0	0
0x28	Used for RFC3394 Wrap Expanded Data															
0x2C	Used for RFC3394 Wrap Expanded Data															

Figure 232. OTFAD context "n" key blob data structure

```
union {

     unsigned char      key[16];

     unsigned char      ctr[8];

     unsigned int       srtaddr;

     unsigned int       endaddr;

     unsigned int       zero_fill;

     unsigned int       zero_fill;

     unsigned char      expanded_wrap_data[8];

     unsigned char      unused_filler[16]

}    otfad_keyblob[4];
```

Figure 233. OTFAD context "n" key blob data structure

32.5.2.3 RFC3394 wrap & unwrap specification overview

The specification “is intended to satisfy the NIST key wrap requirement to: design a cryptographic algorithm called a key wrap that uses the Advanced Encryption Standard (AES) as a primitive to securely encrypt plaintext key(s) with any associated integrity information and data, such that the combination could be longer than the width of the AES block size (128 bits).” [3]

To clarify terminology, be aware that throughout the specification, “any data being wrapped will be referred to as the key data. It makes no difference to the algorithm whether the data being wrapped is a key; in fact there is often good reason to include other data with the key.”

The key used to perform the wrap and unwrap operations is known as the key encryption key (KEK). For the OTFAD, the KEK is a 128-bit key, derived from multiple input signals including a 128-bit one-time programmable key (OTP key), a 32-bit key scramble value and an 8-bit key scramble align control field.

The key wrap and unwrap functions are defined to operate on blocks of 64-bit data with the algorithm being specified in terms of “n” 64-bit data blocks. For the OTFAD implementation, n = 5 as this corresponds to the 40 byte key blob data structure. When executing a wrap operation, the algorithm performs 6 * n AES cipher operations; for the OTFAD implementation, this corresponds to 30 unique AES encryptions plus the generation of an additional 64 bits of wrapped data. This is labeled as Extended Wrap Data and `expanded_wrap_data` in Figure 232 and Figure 233.

When the OTFAD hardware unwraps a key blob, it performs 30 AES inverse cipher operations to recover the original data. Additionally, the wrap implementation uses a 64-bit initialization vector (IV) defined as 0xa6a6a6a6_a6a6a6a6. At the conclusion of the OTFAD's unwrapping, the key blob hardware performs a data integrity check to validate the correct IV was recovered. If the IV integrity check miscompares, then the OTFAD asserts the SR[KBERR] indicator. In the event of this type of error, the contents of the programming model registers of the context being unwrapped are cleared, leaving an invalid context, that is, RGD_W1_n[0] = RGD_W1_n[VLD] = 0.

32.5.2.4 Key blob KEK details

Recall the 128-bit key used in the key blob wrap and unwrap is defined as the key encryption key (KEK). The OTFAD implementation generates the KEK under control of four configuration input signals.

With the scrambled KEK as the output, the 32-bit scrambled key and 8-bit key alignment control provide an optional capability to further randomize the KEK value used for each context. The C language representation of this scrambling is shown in Figure 234.

```
void do_kek_scramble   (unsigned char otp_key[],
                        unsigned char key_scramble[],
                        unsigned int key_scramble_align,
                        unsigned int enb_otfad_kek_scramble,
                        unsigned int context_select,
                        unsigned char scrambled_kek[])
{
    // otp_key[16], key_scramble[4], scrambled_kek[16] are unsigned char arrays

    unsigned int i,j,k;             // local variables
    // copy otp_key[] input to scrambled_kek[] output
    for (i = 0; i < 16; i++) {
        scrambled_kek[i] = otp_key[i];
}
if (enb_otfad_kek_scramble == 1) {
    // retrieve the 2-bit align select from the 8-bit key_scramble_align
    // context_0_select = key_scramble_align[1:0]
    // context_1_select = key_scramble_align[3:2]
    // context_2_select = key_scramble_align[5:4]
    // context_3_select = key_scramble_align[7:6]
    j = 2 * context_select;
    k = (key_scramble_align & (3 << j)) >> j;

    // XOR 4-byte key_scramble[] into appropriate 4-bytes of scrambled_kek[] output
    for (i = 0; i < 4; i++) {
          scrambled_kek[4*k + i] ^= key_scramble[i];
    }
  }
}
```

Figure 234. Language KEK scramble function

32.5.2.5 Key blob wrap

For consistency in data handling as described in Data organization and endianness considerations, the following C code implements the required key blob wrap function using unsigned char input and output data arrays.

The plaintext view of the context's programming model registers is specified as a 40-element byte array. As previously discussed, the number of 64-bit data blocks (N) is 5. The key_wrap function receives the expanded KEK and the input byte array (`plaintext[*]`) as input data and generates the 48-byte wrapped output (`wrapped_ciphertext[*]`). See Figure 235 for the C code. The pseudo-code comments and internal variable names are taken from the RFC3394 spec.

```
extern unsigned char     iv[8];              //  64-bit initialization vector
extern unsigned char     wrapped_ciphertext[8*(N+1)];
extern unsigned char     unwrapped_plaintext[8*(N+1)];

unsigned char               a[8];               //  64-bit integrity check register
unsigned char               b[16];             // 128-bit temp register
unsigned char               r[8*(N+1)];     //   8-bit array of 64-bit registers

//************************************************************************
//************************************************************************

void do_aes128_key_wrap(unsigned char plaintext[],
                                  unsigned char wrapped_ciphertext[],
                                  unsigned int expanded_kek[])
{
     unsigned int           i,j;                  // loop counters
     unsigned char        in[16];             // 128-bit temporary plaintext input vector

     // step 1: initialize the byte-sized data variables
     // set A = IV
     // for i = 1 to n
     //      R[i] = P[i]

     a[0]      = iv[0];
     a[1]      = iv[1];
     a[2]      = iv[2];
     a[3]      = iv[3];
     a[4]      = iv[4];
     a[5]      = iv[5];
     a[6]      = iv[6];
     a[7]      = iv[7];

     for (i = 1; i <= N; i++) {
          r[8*i+ 0] = plaintext[8*(i-1)+ 0];
          r[8*i+ 1] = plaintext[8*(i-1)+ 1];
          r[8*i+ 2] = plaintext[8*(i-1)+ 2];
          r[8*i+ 3] = plaintext[8*(i-1)+ 3];
          r[8*i+ 4] = plaintext[8*(i-1)+ 4];
          r[8*i+ 5] = plaintext[8*(i-1)+ 5];
          r[8*i+ 6] = plaintext[8*(i-1)+ 6];
          r[8*i+ 7] = plaintext[8*(i-1)+ 7];
     }

     // step 2: calculate intermediate values
     // for j = 0 to 5
     //      for i = 1 to n
     //           B = AES(K, A | R[i])
     //           A = MSB(64, B) ^ (n*j)+i
```

```
//           R[i] = LSB(64, B)

for (j = 0; j <= 5; j++) {
     for (i = 1; i <= N; i++) {
          in[0]      = a[0];
          in[1]      = a[1];
          in[2]      = a[2];
          in[3]      = a[3];
          in[4]      = a[4];
          in[5]      = a[5];
          in[6]      = a[6];
          in[7]      = a[7];

          in[8]     = r[8*i+ 0];
          in[9]     = r[8*i+ 1];
          in[10]    = r[8*i+ 2];
          in[11]    = r[8*i+ 3];
          in[12]    = r[8*i+ 4];
          in[13]    = r[8*i+ 5];
          in[14]    = r[8*i+ 6];
          in[15]    = r[8*i+ 7];

          Cipher(in, expanded_kek, 10, b); // perform aes128 encryption

          a[0] = b[0];
          a[1] = b[1];
          a[2] = b[2];
          a[3] = b[3];
          a[4] = b[4];
          a[5] = b[5];
          a[6] = b[6];
          a[7] = b[7] ^ ((N*j)+i);

          r[8*i+ 0]= b[8];
          r[8*i+ 1]= b[9];
          r[8*i+ 2]= b[10];
          r[8*i+ 3]= b[11];
          r[8*i+ 4]= b[12];
          r[8*i+ 5]= b[13];
          r[8*i+ 6]= b[14];
          r[8*i+ 7]= b[15];
     } // end for (i)
} // end for (j)

// step 3: output the results
// set C[0] = A
// for i = 1 to n
// C[i] = R[i]

wrapped_ciphertext[0] = a[0];
wrapped_ciphertext[1] = a[1];
wrapped_ciphertext[2] = a[2];
wrapped_ciphertext[3] = a[3];
wrapped_ciphertext[4] = a[4];
wrapped_ciphertext[5] = a[5];
wrapped_ciphertext[6] = a[6];
wrapped_ciphertext[7] = a[7];

for (i = 1; i <= N; i++) {
     wrapped_ciphertext[8*i+ 0] = r[8*i+ 0];
```

```
            wrapped_ciphertext[8*i+ 1] = r[8*i+ 1];
            wrapped_ciphertext[8*i+ 2] = r[8*i+ 2];
            wrapped_ciphertext[8*i+ 3] = r[8*i+ 3];
            wrapped_ciphertext[8*i+ 4] = r[8*i+ 4];
            wrapped_ciphertext[8*i+ 5] = r[8*i+ 5];
            wrapped_ciphertext[8*i+ 6] = r[8*i+ 6];
            wrapped_ciphertext[8*i+ 7] = r[8*i+ 7];
        }
}
```

Figure 235. C code for OTFAD key blob wrap function (N = 5)

This C key blob wrap implementation has been verified using the (N = 2) “golden” data vectors included in the RFC3394 spec.

As a specific OTFAD example, consider the following key wrap. The data values are described as 32-bit values:

```
kek_w0                                  = 0x33221100
kek_w1                                  = 0x77665544
kek_w2                                  = 0xBBAA9988
kek_w3                                  = 0xFFEEDDCC

// context registers as plaintext
key_w0                                  = 0x03020100          // context key
key_w1                                  = 0x07060504
key_w2                                  = 0x0B0A0908
key_w3                                  = 0x0F0E0D0C
ctr_w0                                  = 0x67452301          // upper 64 bits of counter
ctr_w1                                  = 0xEFCDAB89
rgd_w0                                  = 0x68000000          // srtaddr
rgd_w1                                  = 0x6800FFFB          // endaddr + ade + vld
fill_w0                                 = 0x00000000          // must be all zeroes (always)
fill_w1                                 = 0x00000000          // must be all zeroes (always)
```

Recall the key_wrap function views the entire plaintext data structure as a 40-element *byte array*. For the data set, this means the data is input to the wrap function as:

```
unsigned char plaintext[]   ={0x00, 0x01, 0x02, 0x03,           // key_w0
                              0x04, 0x05, 0x06, 0x07,           // key_w1
                              0x08, 0x09, 0x0A, 0x0B,           // key_w2
                              0x0C, 0x0D, 0x0E, 0x0F,           // key_w3
                              0x01, 0x23, 0x45, 0x67,           // ctr_w0
                              0x89, 0xAB, 0xCD, 0xEF,           // ctr_w1
                              0x00, 0x00, 0x00, 0x68,           // rgd_w0
                              0xFB, 0xFF, 0x00, 0x68,           // rgd_w1
                              0x00, 0x00, 0x00, 0x00,           // fill_w0
                              0x00, 0x00, 0x00, 0x00,           // fill_w1
```

Execution of the key_wrap function generates the following 32-bit view of the 48 bytes of wrapped ciphertext:

```
wrapped_ciphertext_w0           = 0x291F461E
wrapped_ciphertext_w1           = 0x57EF591D
wrapped_ciphertext_w2           = 0x47FB5D63
wrapped_ciphertext_w3           = 0x5497F296
wrapped_ciphertext_w4           = 0xB2571B8C
```

```
wrapped_ciphertext_w5           = 0x57F5B22B
wrapped_ciphertext_w6           = 0x39D320ED
wrapped_ciphertext_w7           = 0xAA7C0A81
wrapped_ciphertext_w8           = 0x567F9FB8
wrapped_ciphertext_w9           = 0x91F32684
wrapped_ciphertext_w10          = 0xDD5E431E
wrapped_ciphertext_w11          = 0xC96FA1FF
```

The four wrapped key blobs are programmed into the external flash memory at the following addresses:

```
Context 0 key blob @ SPI_BASE_ADDRESS + 0x00
Context 1 key blob @ SPI_BASE_ADDRESS + 0x40
Context 2 key blob @ SPI_BASE_ADDRESS + 0x80
Context 3 key blob @ SPI_BASE_ADDRESS + 0xC0
```

If key blob processing is initiated by the assertion of CR[SKBP], the OTFAD fetches each key blob, unwraps it, loads the unwrapped plaintext into the appropriate context registers. *This process is repeated for all four key blobs.*

NOTE

The processing of all four key blobs requires that each plaintext data structure must be programmed with “valid data”. For unused key blobs, an all-zero plaintext data structure defines a valid, but disabled, context.

32.5.2.6 Key blob unwrap

After a system reset event and the assertion of CR[SKBP], if key blob processing is enabled, the QuadSPI fetches the context 0 key blob from the external flash memory, sends the data to the OTFAD which automatically performs the required key blob unwrap function and loads the recovered plaintext data into the context 0 programming model registers. When the context 0 unwrap is completed, the OTFAD continues with the unwrap for context 1, then context 2 and finally context 3. As previously noted, OTFAD always processes key blob data for all four contexts when enabled. Software is responsible for creating and programming the external memory with a valid key wrap for all four contexts, regardless of their actual usage.

The OTFAD’s key blob processing logic provides a hardware implementation of the unwrap process. This section presents a C function of the comparable software implementation. The unwrap operation performs 6 * n AES inverse cipher decryptions (a total of 30 decryptions) to recover the original plaintext data structure.

The input data for the unwrap is specified as a 48-element byte array. Recall from the previous description, the key wrap process generates an “extra” 8 bytes of wrapped_ciphertext output. The number of 64-bit data blocks (N = n) remains as 5. The key_unwrap function receives the expanded KEK and the wrapped data byte array (wrapped_ciphertext[*]) as input data and recovers the original 40-byte unwrapped plaintext data structure (unwrapped_plaintext[*]). See Figure 236 for the C code. As before, the pseudo-code comments and internal variable names are taken from the RFC3394 spec.

```
extern unsigned char      iv[8];            // 64-bit initialization vector
extern unsigned char      wrapped_ciphertext[8*(N+1)];
extern unsigned char      unwrapped_plaintext[8*(N+1)];

unsigned char               a[8];             // 64-bit integrity check register
unsigned char               b[16];           // 128-bit temp register
unsigned char               r[8*(N+1)];    // 8-bit array of 64-bit registers

//************************************************************************
//************************************************************************

unsigned int do_aes128_key_unwrap(unsigned char wrapped_ciphertext[],
                                  unsigned char unwrapped_plaintext[],
                                  unsigned int expanded_kek[])
{
```

```
     signed int             i,j;           // loop counters
     unsigned char      in[16];           // 128-bit temporary ciphertext input vector

     // step 1: initialize variables
     // set A = C[0]
     // for i = 1 to n
     // R[i] = C[i]
     a[0]      = wrapped_ciphertext[0];
     a[1]      = wrapped_ciphertext[1];
     a[2]      = wrapped_ciphertext[2];
     a[3]      = wrapped_ciphertext[3];
     a[4]      = wrapped_ciphertext[4];
     a[5]      = wrapped_ciphertext[5];
     a[6]      = wrapped_ciphertext[6];
     a[7]      = wrapped_ciphertext[7];

     for (i = 1; i <= N; i++) {
          r[8*i+ 0] = wrapped_ciphertext[8*i+ 0];
          r[8*i+ 1] = wrapped_ciphertext[8*i+ 1];
          r[8*i+ 2] = wrapped_ciphertext[8*i+ 2];
          r[8*i+ 3] = wrapped_ciphertext[8*i+ 3];
          r[8*i+ 4] = wrapped_ciphertext[8*i+ 4];
          r[8*i+ 5] = wrapped_ciphertext[8*i+ 5];
          r[8*i+ 6] = wrapped_ciphertext[8*i+ 6];
          r[8*i+ 7] = wrapped_ciphertext[8*i+ 7];
     }

     // step 2: calculate intermediate values
     // for j = 5 to 0
     // for i = n to 1
     // B = AES-1(K, (A ^ (n*j+i) | R[i])
     // A = MSB(64, B)
     // R[i] = LSB(64, B)

     for (j = 5; j >= 0; j--) {
          for (i = N; i >= 1; i--) {
               in[0]      = a[0];
               in[1]      = a[1];
               in[2]      = a[2];
               in[3]      = a[3];
               in[4]      = a[4];
               in[5]      = a[5];
               in[6]      = a[6];
               in[7]      = a[7] ^ ((N*j)+i);

               in[8]      = r[8*i+ 0];
               in[9]      = r[8*i+ 1];
               in[10]     = r[8*i+ 2];
               in[11]     = r[8*i+ 3];
               in[12]     = r[8*i+ 4];
               in[13]     = r[8*i+ 5];
               in[14]     = r[8*i+ 6];
               in[15]     = r[8*i+ 7];

InvCipher(in, expanded_kek, 10, b); // perform aes128 decryption

               a[0]      = b[0];
               a[1]      = b[1];
               a[2]      = b[2];
```

```
            a[3]      = b[3];
            a[4]      = b[4];
            a[5]      = b[5];
            a[6]      = b[6];
            a[7]      = b[7];

            r[8*i+ 0]= b[8];
            r[8*i+ 1]= b[9];
            r[8*i+ 2]= b[10];
            r[8*i+ 3]= b[11];
            r[8*i+ 4]= b[12];
            r[8*i+ 5]= b[13];
            r[8*i+ 6]= b[14];
            r[8*i+ 7]= b[15];
       } // end for (i)
  } // end for (j)

  // step 3: output the results
  // if A == IV
  //      then
  //           for i = 1 to n
  //                P[i] = R[i]
  //      else
  //           return an error

  unwrapped_plaintext[0] = a[0];
  unwrapped_plaintext[1] = a[1];
  unwrapped_plaintext[2] = a[2];
  unwrapped_plaintext[3] = a[3];
  unwrapped_plaintext[4] = a[4];
  unwrapped_plaintext[5] = a[5];
  unwrapped_plaintext[6] = a[6];
  unwrapped_plaintext[7] = a[7];

  for (i = 1; i <= N; i++) {
       unwrapped_plaintext[8*i+ 0] = r[8*i+ 0];
       unwrapped_plaintext[8*i+ 1] = r[8*i+ 1];
       unwrapped_plaintext[8*i+ 2] = r[8*i+ 2];
       unwrapped_plaintext[8*i+ 3] = r[8*i+ 3];
       unwrapped_plaintext[8*i+ 4] = r[8*i+ 4];
       unwrapped_plaintext[8*i+ 5] = r[8*i+ 5];
       unwrapped_plaintext[8*i+ 6] = r[8*i+ 6];
       unwrapped_plaintext[8*i+ 7] = r[8*i+ 7];
  }

  if ((unwrapped_plaintext[0] == iv[0]) &&
       (unwrapped_plaintext[1] == iv[1]) &&
       (unwrapped_plaintext[2] == iv[2]) &&
       (unwrapped_plaintext[3] == iv[3]) &&
       (unwrapped_plaintext[4] == iv[4]) &&
       (unwrapped_plaintext[5] == iv[5]) &&
       (unwrapped_plaintext[6] == iv[6]) &&
       (unwrapped_plaintext[7] == iv[7]))
       return 0;                                    // error-free exit
  else
       return -1;                                    // error exit
```

```
}
```

Figure 236. C Code for OTFAD Key Blob Unwrap Function (N = 5)

32.5.2.7 Key blob error detection and reporting

At the conclusion of the key unwrap, there is a data integrity check performed to validate the original plaintext data structure has been correctly recovered. As shown in the last section of code in Figure 236, eight bytes of the unwrapped_plaintext[*] are compared with the initialization vector (iv[*]). The OTFAD initialization vector uses the default value specified in RFC3394, the static constant 0xa6a6a6a6_a6a6a6a6.

If the IV integrity check miscompares, then the OTFAD asserts the SR[KBERR] indicator. This indicator is typically configured as an interrupt request to signal the device of the detected error. Additionally, the appropriate bits in SR[CTXIE] and SR[CTXER] are set. The assertion of both SR[CTXIEn] and SR[CTXERn] indicates an integrity check error for context n.

When this error is detected, the contents of the programming model registers of the specific context being unwrapped are cleared, leaving an invalid context, that is, RGD_W1_n[0] = RGD_W1_n[VLD] = 0.

32.6 Application information

32.6.1 OTFAD operating modes

As discussed in Modes of operation, the OTFAD supports two modes of operation as defined by input configuration and control signals. The mode states and their transitions are shown in Figure 237. The states include Normal Mode (00) and Logically Disabled Mode (11). The operating mode state information is visible in the SR[MODE] field.

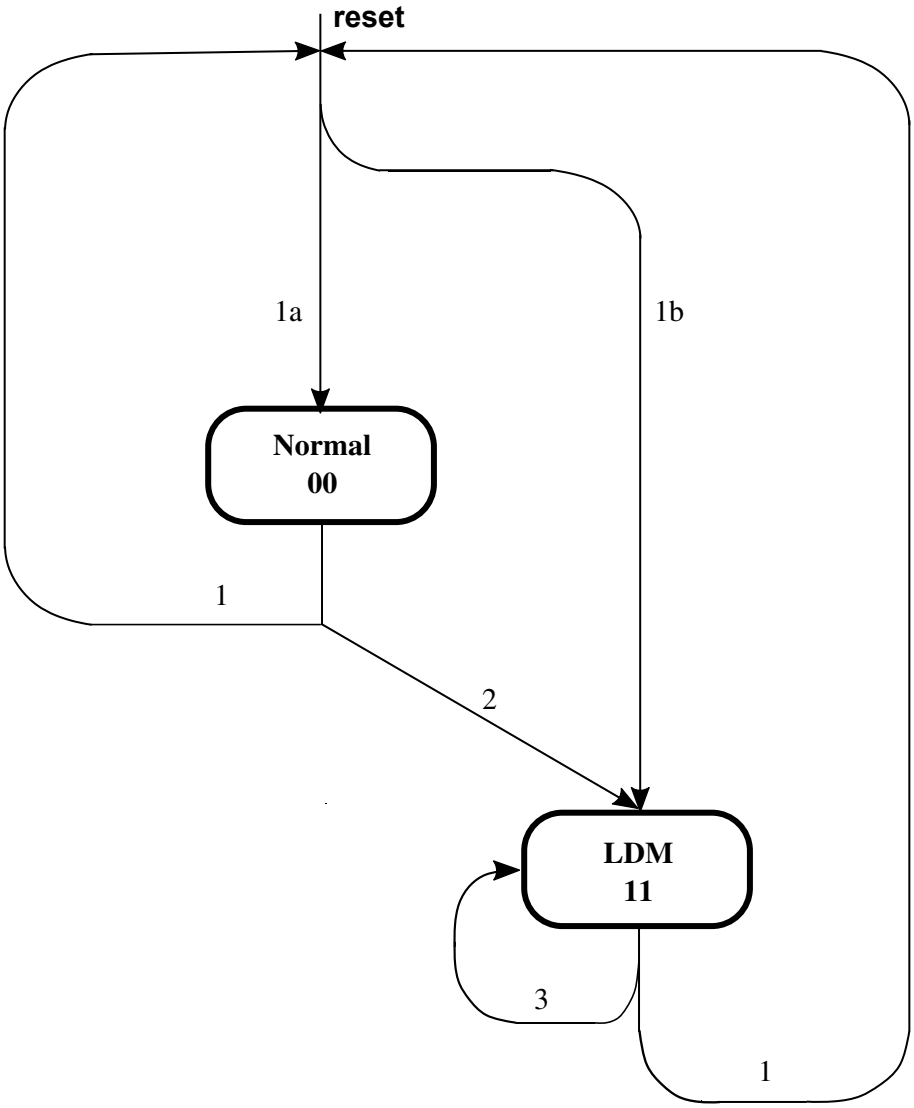


Figure 237. OTFAD Operating Mode States and Transitions

The state transitions are defined in Table 244.

Table 244. OTFAD mode state transitions

Current state	Next state	Transition	Description
–	Normal, LDM	1 -> 1a, 1b	Any reset event initializes the OTFAD mode state.
–	Normal	1a	Any reset, combined with a negated otfad_disable_override configuration input signal, unconditionally forces OTFAD into the Normal state.
–	LDM	1b	Any reset, combined with an asserted otfad_disable_override configuration input signal, unconditionally forces OTFAD into the LDM state.

Table continues on the next page...

Table 244. OTFAD mode state transitions (continued)

Current state	Next state	Transition	Description
Normal	LDM	2	A transition from Normal to LDM is made when the boolean expression (cr[fldm] & ~reset) is true.
LDM	LDM	3	The LDM is retained while reset is not asserted.

32.6.2 Typical OTFAD CTR-AES128 decryption operation

The dominant use case in normal mode is expected to be CTR-AES128 decryption operations performed in response to a 64-bit WRAP4 burst read in response to a processor cache miss. The following C code is a simplified representation of the OTFAD processing associated with this use case. The ciphertext fetched by the QuadSPI is the `spi_ciphertext[32]` byte array, and the resulting decrypted data is `otfad_plaintext[32]`. See Figure 238.

```
void KeyExpansion (unsigned char ckey[], int nbits, unsigned int w[]);
void Cipher (unsigned char cin[], unsigned int w[], int nr, unsigned char cout[]);

unsigned char                          spi_ciphertext[32];
unsigned char                          otfad_plaintext[32];

union context_regs {
     volatile unsigned char      key[16];
     volatile unsigned char      ctr[8];
     volatile unsigned int      rgd[2];
     volatile unsigned int      unused_filler[8]
     } otfad_context[4];

void otfad_32byte_processing (unsigned int sys_addr,
                                        unsigned char spi_ciphertext[],
                                        unsigned char otfad_plaintext[])

{
     unsigned int      i;
     unsigned int      ctx_hit;
     unsigned int      combined_hits;
     unsigned int      enable_decryption;
     unsigned int      w[44];           // expanded key
     unsigned char      encrypted_ctr[2][16];
     unsigned char      temp_ctr[16];
     unsigned int      local_addr;
     unsigned int      hit_addr;

     local_addr &= 0xfffffff0;      // force byte addr to 0-mod-16 (aes size)
     hit_addr = local_addr & 0xfffffc00;      // force hit addr to 0-mod-1K

     // context determination
     combined_hits = 0;           // 4-bit hit indicator: ctx{3,2,1,0}
     for (i = 0; i < 4; i++) {
          ctx_hit = ((hit_addr >= otfad_context[i].rgd[0])      // srtaddr
               & (hit_addr <= otfad_context[i].rgd[1]))      // endaddr
               & (otfad_context[i].rgd[i] & 1);           // vld
          combined_hits += (ctx_hit << i);
     }
     // check combined_hits to determine if decryption is enabled
     enable_decryption = 0;
```

```
switch (combined_hits) {
     case 1:                    // ctx0 hit
          ctx_hit = 0;
          if ((otfad_context[0].rgd[1] & 2) != 0) {      // ade
               enable_decryption = 1;
          }
          break;
     case 2:                    // ctx1 hit
          ctx_hit = 1;
          if ((otfad_context[1].rgd[1] & 2) != 0) {      // ade
               enable_decryption = 1;
          }
          break;
     case 4:                    // ctx2 hit
          ctx_hit = 2;
          if ((otfad_context[2].rgd[1] & 2) != 0) {      // ade
               enable_decryption = 1;
          }
          break;
     case 8:                    // ctx3 hit
          ctx_hit = 3;
          if ((otfad_context[3].rgd[1] & 2) != 0) {      // ade
               enable_decryption = 1;
          }
          break;
     default:                    // none or multiple ctx hits
          ctx_hit = 0;
               enable_decryption = 0;           // no decryption, bypass data
     break;
     }

// clear encrypted counters
for (i = 0; i < 16; i++) {
     encrypted_ctr[0][i] = 0;
     encrypted_ctr[1][i] = 0;
}

// if decryption is enabled, calculate two encrypted counter values
if (enable_decryption == 1) {
     KeyExpansion (&otfad_context[ctx_hit].key[0], 128, w);
     // encrypt the counter for the 1st 16 bytes
          // step 1: generate the 128-bit counter
          // step 1a: first 8 bytes are taken directly from context.ctr[]
          for (i = 0; i < 8; i++) {
               temp_ctr[i] = otfad_context[ctx_hit].ctr[i];
          }
          // step 1b: next 4 bytes are xor of ctr[0-3] ^ ctr[4-7]
          temp_ctr[8] = temp_ctr[0] ^ temp_ctr[4];
          temp_ctr[9] = temp_ctr[1] ^ temp_ctr[5];
          temp_ctr[10] = temp_ctr[2] ^ temp_ctr[6];
          temp_ctr[11] = temp_ctr[3] ^ temp_ctr[7];
          // step 1c: final 4 bytes are local_addr
          temp_ctr[12] = (local_addr & 0xff000000) >> 24;
          temp_ctr[13] = (local_addr & 0x00ff0000) >> 16;
          temp_ctr[14] = (local_addr & 0x0000ff00) >> 8;
          temp_ctr[15] = (local_addr & 0x000000ff);

     // step 2: perform the 1st counter encryption
     Cipher (&temp_ctr[0], w, 10, &encrypted_ctr[0][0]);
```

```
          // encrypt the counter for the 2nd 16 bytes
               // step 1a: increment the local_addr by 16
               local_addr += 16;
               // step 1b: final 4 bytes of counter are new local_addr
               temp_ctr[12] = (local_addr & 0xff000000) >> 24;
               temp_ctr[13] = (local_addr & 0x00ff0000) >> 16;
               temp_ctr[14] = (local_addr & 0x0000ff00) >> 8;
               temp_ctr[15] = (local_addr & 0x000000ff);

          // step 2: perform the 2nd counter encryption
          Cipher (&temp_ctr[0], w, 10, &encrypted_ctr[1][0]);
          }

          // generate plaintext data = f(encrypted_ctr[][], spi_ciphertext[])
          // for bypass operations, the encrypted_ctr[][] = 0, so plain = cipher
          // since spi_ciphertext arrives 8 bytes at a time, code does the same...
          for (i = 0; i < 8; i++)
               otfad_plaintext[i]      = encrypted_ctr[0][i]      ^ spi_ciphertext[i];
          for (i = 8; i < 16; i++)
               otfad_plaintext[i]      = encrypted_ctr[0][i]      ^ spi_ciphertext[i];
          for (i = 16; i < 24; i++)
               otfad_plaintext[i]      = encrypted_ctr[1][i-16]      ^ spi_ciphertext[i];
          for (i = 24; i < 32; i++)
               otfad_plaintext[i]      = encrypted_ctr[1][i-16]      ^ spi_ciphertext[i];
}
```

Figure 238. Basic OTFAD 32-byte decryption processing

32.7 Register descriptions and data organization

The registers of the OTFAD programming model can only be accessed while in secure privileged mode. Unless noted otherwise, the programming model registers can be accessed via 8-, 16- or 32-bit reads and 32-bit write references. Attempted accesses in a different operating mode, using unsupported write data sizes, writes to read-only resources, or to reserved spaces are terminated with an error unless noted otherwise. See also Data organization and endianness considerations.

The programming model associated with each memory context follows the same data structure as the key blobs stored in the external flash.

In the registers named *CTXn_<regname>*, the "n" is the context number.

Note that in a fully secure operating environment, the *CTXn_<regname>* registers are loaded by the OTFAD hardware during key blob processing. Software access in this environment may be considerably reduced based on the state of the CR[RRAE] and static chip configuration inputs (see Static configuration signals). For the affected *CTXn_<regname>* registers, their access values are shown as "RW" in this section; however, these registers are treated as read-as-zero/write-ignore (RAZ/WI) if operating in the restricted register access mode (CR[RRAE] = 1 and SR[RRAM] = 1). See the individual register descriptions for more information.

32.7.1 OTFAD register descriptions

32.7.1.1 OTFAD memory map

OTFAD1 base address: 425E_0000h

OTFAD2 base address: 445E_0000h

Offset	Register	Width (In bits)	Access	Reset value
C00h	Control Register (CR)	32	RW	0000_0000h
C04h	Status Register (SR)	32	RW	0000_0040h
D00h	AES Key Word (CTX0_KEY0)	32	RW	0000_0000h
D04h	AES Key Word (CTX0_KEY1)	32	RW	0000_0000h
D08h	AES Key Word (CTX0_KEY2)	32	RW	0000_0000h
D0Ch	AES Key Word (CTX0_KEY3)	32	RW	0000_0000h
D10h	AES Counter Word (CTX0_CTR0)	32	RW	0000_0000h
D14h	AES Counter Word (CTX0_CTR1)	32	RW	0000_0000h
D18h	AES Region Descriptor Word0 (CTX0_RGD_W0)	32	RW	0000_0000h
D1Ch	AES Region Descriptor Word1 (CTX0_RGD_W1)	32	RW	0000_03F8h
D40h	AES Key Word (CTX1_KEY0)	32	RW	0000_0000h
D44h	AES Key Word (CTX1_KEY1)	32	RW	0000_0000h
D48h	AES Key Word (CTX1_KEY2)	32	RW	0000_0000h
D4Ch	AES Key Word (CTX1_KEY3)	32	RW	0000_0000h
D50h	AES Counter Word (CTX1_CTR0)	32	RW	0000_0000h
D54h	AES Counter Word (CTX1_CTR1)	32	RW	0000_0000h
D58h	AES Region Descriptor Word0 (CTX1_RGD_W0)	32	RW	0000_0000h
D5Ch	AES Region Descriptor Word1 (CTX1_RGD_W1)	32	RW	0000_03F8h
D80h	AES Key Word (CTX2_KEY0)	32	RW	0000_0000h
D84h	AES Key Word (CTX2_KEY1)	32	RW	0000_0000h
D88h	AES Key Word (CTX2_KEY2)	32	RW	0000_0000h
D8Ch	AES Key Word (CTX2_KEY3)	32	RW	0000_0000h
D90h	AES Counter Word (CTX2_CTR0)	32	RW	0000_0000h
D94h	AES Counter Word (CTX2_CTR1)	32	RW	0000_0000h
D98h	AES Region Descriptor Word0 (CTX2_RGD_W0)	32	RW	0000_0000h
D9Ch	AES Region Descriptor Word1 (CTX2_RGD_W1)	32	RW	0000_03F8h
DC0h	AES Key Word (CTX3_KEY0)	32	RW	0000_0000h
DC4h	AES Key Word (CTX3_KEY1)	32	RW	0000_0000h
DC8h	AES Key Word (CTX3_KEY2)	32	RW	0000_0000h
DCCh	AES Key Word (CTX3_KEY3)	32	RW	0000_0000h
DD0h	AES Counter Word (CTX3_CTR0)	32	RW	0000_0000h
DD4h	AES Counter Word (CTX3_CTR1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
DD8h	AES Region Descriptor Word0 (CTX3_RGD_W0)	32	RW	0000_0000h
DDCh	AES Region Descriptor Word1 (CTX3_RGD_W1)	32	RW	0000_03F8h

32.7.1.2 Control Register (CR)

Offset

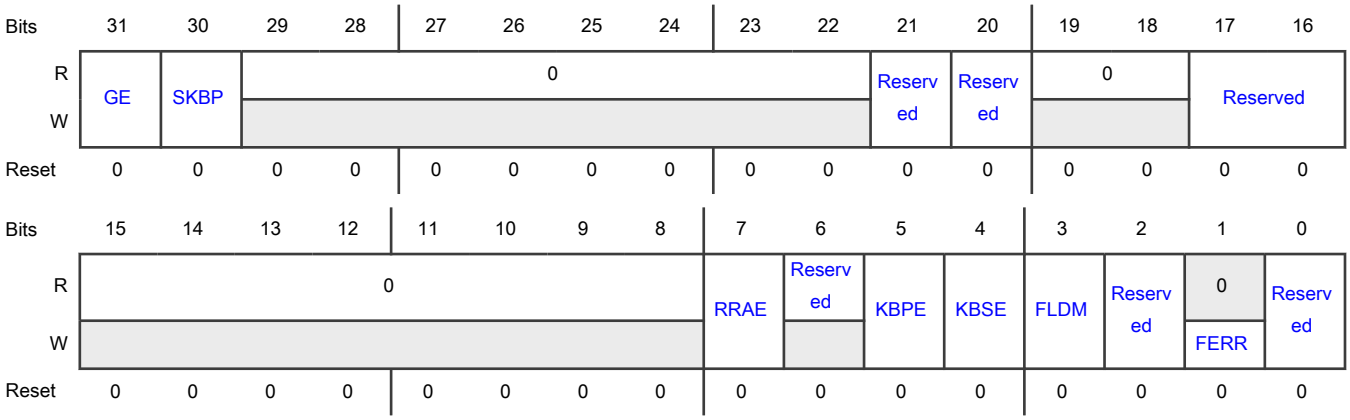
Register	Offset
CR	C00h

Function

This register defines the operating configuration of the OTFAD including a global enable indicator.

There are 5 register fields in the CR that control the OTFAD configuration, especially related to key blob processing after the negation of system reset. These fields (GE, RRAE, KBCE, KBPE, KBSE) are generated based on both the contents of the CR plus static input configuration signals. For all 5 bits, the contents of the CR flag is logically summed (OR'd) with the static input configuration signals to form the actual control used by the OTFAD. Reads of the CR return the logical summation of these configuration controls. Entry into the LDM mode clears all 5 indicators. The initiation of properly-enabled key blob processing is enabled by the assertion of the SKBP flag. For more details, see OTFAD configuration details.

Diagram



Fields

Field	Function
31 GE	Global OTFAD Enable This field enables the OTFAD operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The global enabled/disabled state of the OTFAD module is based on the logical OR of the value written to GE and the value of the static input *otfad_enable*. For example, if *otfad_enable* = 1, the OTFAD module is always enabled. Reading GE returns the global enabled/disabled state of the OTFAD module. It is cleared by entry into the LDM mode. 0b - OTFAD has decryption disabled. All data fetched by the QuadSPI bypasses OTFAD processing. 1b - OTFAD has decryption enabled, and processes data fetched by the QuadSPI as defined by the hardware configuration.
30 SKBP	Start key blob processing When set and key blob processing enabled (as signaled by CR[KBPE] asserted), the OTFAD initiates key blob processing. The simultaneous assertion of SKBP and KBPE initiates key blob processing. **NOTE** When the static configuration input key_blob_enable is set on a given device, key blob processing is always enabled. To initiate key blob processing in this case, software needs to set SKBP only once after a hard reset. 0b - Key blob processing is not initiated. 1b - Properly-enabled key blob processing is initiated.
29-22 —	Reserved
21 —	Reserved **NOTE** When writing to this register, software must write 0 to this field. This field always reads as zero.
20 —	Reserved **NOTE** When accessing this register, software must write 0 to this field and ignore read values.
19-18 —	Reserved
17-16 —	Reserved **NOTE** When accessing this register, software must write 0s to this field and ignore read values.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 —	Reserved
7 RRAE	Restricted Register Access Enable If this bit is asserted, only the CR, SR and optional MDPC registers can be accessed; attempted accesses of all other registers are treated as RAZ/WI (read-as-zero/write-ignore). This indicator is sticky; once set, it remains asserted until the next system reset, when it is cleared. It is also asserted by entry into the LDM mode. The value of this field is reflected in the SR[RRAM] flag. 0b - Register access is fully enabled. The OTFAD programming model registers can be accessed “normally”. 1b - Register access is restricted and only the CR, SR and optional MDPC registers can be accessed; others are treated as RAZ/WI.
6 —	Reserved
5 KBPE	Key Blob Processing Enable This field determines if key blob processing is enabled after a system reset. The assertion of CR[SKBP] determines when key blob processing is initiated. It is cleared by entry into the LDM mode. The combined {GE,KBPE,KBSE} control fields define the operating configuration of the OTFAD as: • If {GE,KBPE,KBSE} = 0--, OTFAD has decryption disabled • If {GE,KBPE,KBSE} = 10-, OTFAD has decryption enabled, but no key blob processing • If {GE,KBPE,KBSE} = 110, OTFAD has decryption enabled, key blob processing is enabled, no KEK scrambling • If {GE,KBPE,KBSE} = 111, OTFAD has decryption enabled, key blob processing is enabled with KEK scrambling enabled 0b - Key blob processing is disabled. 1b - Key blob processing is enabled.
4 KBSE	Key Blob Scramble Enable This field determines if key blob KEK scrambling is enabled after a system reset. It is cleared by entry into the LDM mode. The combined {GE,KBPE,KBSE} control bits define the operating configuration of the OTFAD as: • If {GE,KBPE,KBSE} = 0--, OTFAD has decryption disabled • If {GE,KBPE,KBSE} = 10-, OTFAD has decryption enabled, but no key blob processing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If {GE,KBPE,KBSE} = 110, OTFAD has decryption enabled, key blob processing is enabled, no KEK scrambling • If {GE,KBPE,KBSE} = 111, OTFAD has decryption enabled, key blob processing is enabled with KEK scrambling enabled 0b - Key blob KEK scrambling is disabled. 1b - Key blob KEK scrambling is enabled.
3 FLDM	Force Logically Disabled Mode This field is intended to provide a mechanism for software testing of entry into the Logically Disabled Mode (LDM). This indicator is sticky; once set, it remains asserted until the next system reset, when it is cleared. 0b - No effect on the operating mode. 1b - Force entry into LDM after a write with this data bit set. SR[MODE] signals the operating mode.
2 —	Reserved Reserved. Writes to this field should be '0' only.
1 FERR	Force Error A write that sets this field forces the OTFAD's key blob error flag (SR[KBERR]) to be asserted. It is intended to provide a mechanism for software testing of a key blob error handling routine. This field always reads as zero. 0b - No effect on the SR[KBERE] indicator. 1b - SR[KBERR] is immediately set after a write with this data bit set.
0 —	Reserved **NOTE** When writing to this register, software must write 0 for this bit.

32.7.1.3 Status Register (SR)

Offset

Register	Offset
SR	C04h

Function
This register provides OTFAD status information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	KBD	KBPE	GEM	RRAM	HRL				0				CTXIE3	CTXIE2	CTXIE1	CTXIE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				CTXER3	CTXER2	CTXER1	CTXER0	NCTX				MODE		MDPCP	KBERR
W																W1C
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Fields

Field	Function
31 KBD	Key Blob Processing Done This field signals the OTFAD has completed key blob processing. This indicator is sticky, that is, once set, it remains set until the next system reset. This field and KBPE can be used for polling purposes to indicate the OTFAD has completed key blob processing. The status of any errors detected during key blob processing is reflected in SR[KBERR] and does not affect the KBD indicator. 0b - Key blob processing was not enabled, or is not complete. 1b - Key blob processing was enabled and is complete.
30 KBPE	Key Blob Processing Enable This field signals if the OTFAD is configured to perform key blob processing. It is a read-only version of the CR[KBPE] indicator. It is included here so it and KBD can easily be used for polling purposes to indicate the OTFAD has completed key blob processing. The combined 2-bit field {KBD,KBPE} signals the key blob processing state: • If {KBD,KBPE} = 00, key blob processing is disabled • If {KBD,KBPE} = 01, key blob processing is enabled, but not yet complete • If {KBD,KBPE} = 11, key blob processing is enabled and complete 0b - Key blob processing is not enabled. 1b - Key blob processing is enabled.
29 GEM	Global Enable Mode This indicator signals the global enabled/disabled state of the OTFAD. This flag is identical to the read value of CR[GE]. 0b - OTFAD is disabled. All data fetched by the QuadSPI bypasses OTFAD processing. 1b - OTFAD is enabled, and processes data fetched by the QuadSPI as defined by the hardware configuration.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 RRAM	Restricted Register Access Mode This indicator signals that accesses to the slave peripheral bus are operating in a restricted mode, where only the CR, SR and optional MDPC registers can be referenced. This access mode can be defined by the assertion of CR[RRAE] = 1. Attempted accesses to other registers are treated as RAZ/WI. This flag is identical to the read value of CR[RRAE]. 0b - Register access is fully enabled. The OTFAD programming model registers can be accessed “normally”. 1b - Register access is restricted and only the CR, SR and optional MDPC registers can be accessed; others are treated as RAZ/WI.
27-24 HRL	Hardware Revision Level This field reads as zero.
23-20 —	Reserved
19-16 CTXIEn	Context Integrity Error CTXIE*n* signals an integrity error was detected in the corresponding context *n* during key blob processing as signaled by SR[KBERR] = 1. The integrity check specifically refers to the final state after the key unwrapping completes. The assertion of CTXIE*n* always means the comparable CTXER*n* flag is also set. This field is cleared if SR[KBERR] was set in response to a write asserting CR[FERR]. It is also cleared when SR[KBERR] is cleared. 0b - No key blob integrity error was detected for context “n”. 1b - A key blob integrity error was detected in context “n”.
15-12 —	Reserved
11-8 CTXERn	Context Error CTXER*n* signals an error was detected in the corresponding context *n* during key blob processing as signaled by SR[KBERR] = 1. The error condition from the combined CTXER and CTXIE indicators is defined as: • If CTXER*n* = 0 and CTXIE*n* = -, then no error • If CTXER*n* = 1 and CTXIE*n* = 0, then no error • If CTXER*n* = 1 and CTXIE*n* = 1, then integrity error This field is cleared if SR[KBERR] was set in response to a write asserting CR[FERR]. It is also cleared when SR[KBERR] is cleared. 0b - No key blob error was detected for context “n”. 1b - A key blob integrity error might have been detected in context “n”.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 NCTX	Number of Contexts This field signals the number of implemented hardware contexts. It reads as 4.
3-2 MODE	Operating Mode This field specifies the OTFAD's operating mode. Input configuration and control signals force entry from normal mode (NRM) into a special operating mode (LDM) . See Modes of operation for more details. 00b - Operating in Normal mode (NRM) 01b - Unused (reserved) 10b - Unused (reserved) 11b - Operating in Logically Disabled Mode (LDM)
1 MDPCP	MDPC Present This field signals the presence of the Multi-Dimensional Parity Checker. For this device, it always has the value 0.
0 KBERR	Key Blob Error This field signals that one or more errors were detected during key blob processing. SR[CTXER] provides the details on which contexts detected errors. This indicator can also be set by writing CR[FERR] = 1 (for testing purposes). 0b - No key blob error detected. 1b - One or more key blob errors has been detected.

32.7.1.4 AES Key Word (CTX0_KEY0 - CTX3_KEY3)

Offset

For n = 0 to 3; m = 0 to 3:

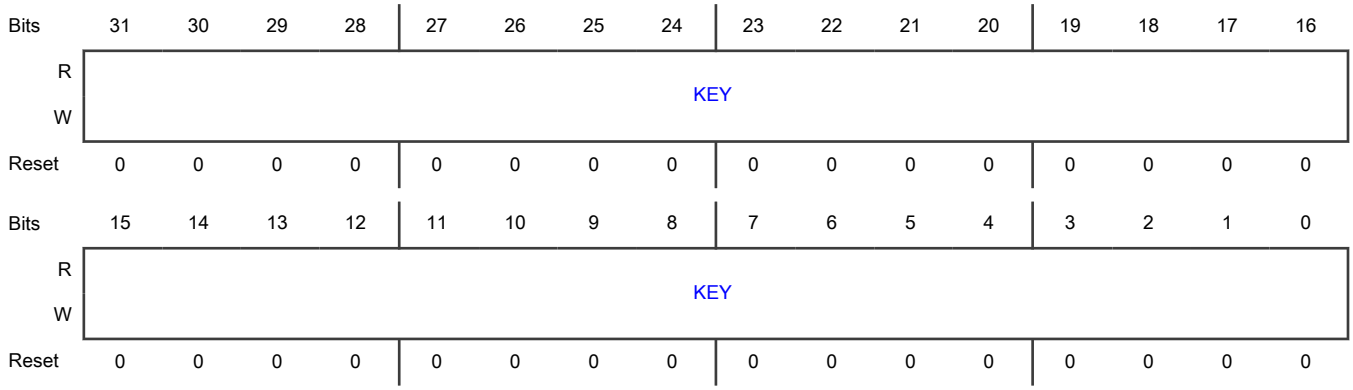
Register	Offset
CTXn_KEYm	D00h + (n × 40h) + (m × 4h)

Function

The CTXn_KEYm registers provide a 2-dimensional data structure for local OTFAD storage of the 128-bit key for context "n". There are four consecutive memory-mapped register words containing the key used for the AES calculations associated with the given context. The programming model view of the CTXn_KEYm registers is a little-endian 16-element byte data array, while the 128-bit key is defined as the concatenation of {A0, A1, A2,..., A14, A15}.

If SR[RRAM] = 1 indicating restricted register access mode, access to these registers is treated as read-as-zero, write-ignored (RAZ/WI).

Diagram



Fields

Field	Function
31-0 KEY	AES Key The key is typically loaded as the corresponding key blob is unwrapped; alternatively, if enabled, it can be written using the slave peripheral bus. The four consecutive little-endian memory-mapped registers provide 128 bits of key storage. Word0: KEY[31:0][A03, A02, A01, A00] Word1: KEY[31:0][A07, A06, A05, A04] Word2: KEY[31:0][A11, A10, A09, A08] Word3: KEY[31:0][A15, A14, A13, A12]

32.7.1.5 AES Counter Word (CTX0_CTR0 - CTX3_CTR1)

Offset

For n = 0 to 3; m = 0 to 1:

Register	Offset
CTXn_CTRm	D10h + (n × 40h) + (m × 4h)

Function

The CTXn_CTRm registers provide a 2-dimensional data structure for local OTFAD storage of 64 bits of the counter value for context “n”. There are two consecutive memory-mapped register words defining the upper 96 bits of the counter used for the AES calculations associated with the given context.

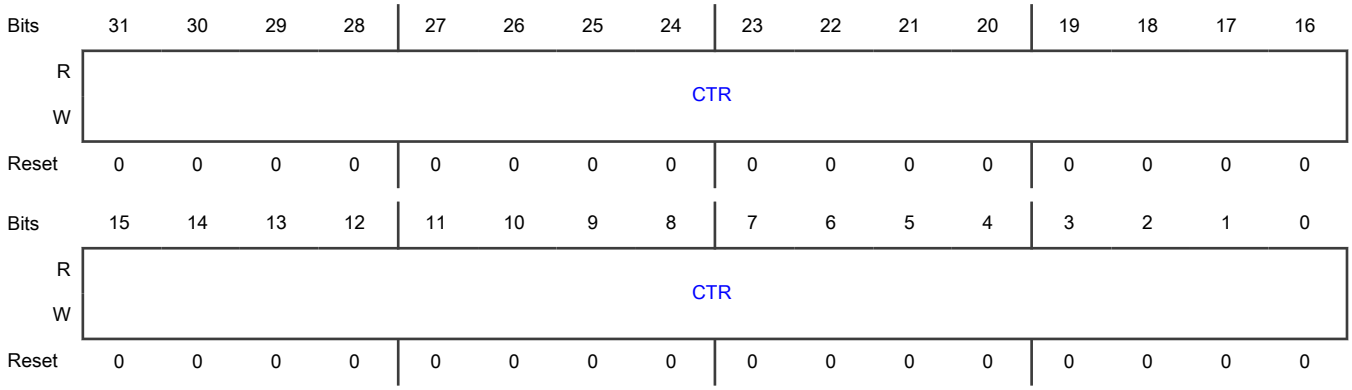
The programming model view of the CTXn_CTRm registers is a little-endian 8-element byte data array.

The entire 128-bit counter value is defined as the concatenation of four 32-bit values:

CTR[127–0] = {CTR0[C0...C3], CTR1[C4...C7], CTR0[C0...C3] ^ CTR1[C4...C7], systemAddress[31–4], 0h}

If SR[RRAM] = 1 indicating restricted register access mode, access to these registers is treated as read-as-zero, write-ignored (RAZ/WI).

Diagram



Fields

Field	Function
31-0 CTR	AES Counter The upper 64 bits of the counter are typically loaded as the corresponding key blob is unwrapped; alternatively, if enabled, it can be written using the slave peripheral bus. The two consecutive memory-mapped registers directly provide the upper 64 bits of counter storage. Word0: CTR[31:0][C3, C2, C1, C0] Word1: CTR[31:0][C7, C6, C5, C4] The third 32-bit portion of the CTR is formed by exclusive-or'ing the upper 64 bits of the counter as two 32-bit values, while the least-significant portion of the counter is the 32-bit 0-modulo-16 byte system address of the external flash memory. CTR[C0...C15] = {CTR[C0...C7], CTR[C0...C3] ^ CTR[C4...C7], systemAddress[31–4], 0h}

32.7.1.6 AES Region Descriptor Word0 (CTX0_RGD_W0 - CTX3_RGD_W0)

Offset

Register	Offset
CTX0_RGD_W0	D18h
CTX1_RGD_W0	D58h
CTX2_RGD_W0	D98h
CTX3_RGD_W0	DD8h

Function

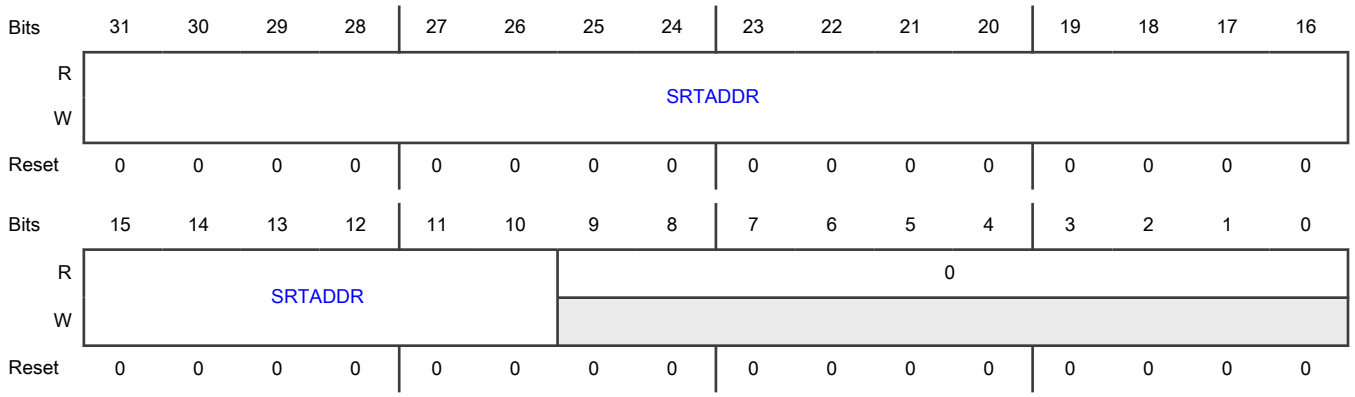
The CTXn_RGD_W0 registers provide a 2-dimensional data structure for local OTFAD storage of 64 bits of memory region descriptor for context “n”. There are two consecutive memory-mapped register words defining the starting and ending addresses for the external flash memory region associated with the given context.

Each memory region context defines a specified section of external flash memory associated with the given context and its associated 128-bit key and 128-bit counter values.

The context memory regions are defined as modulo-1024 bytes to match the AMBA-AHB protocol requirement that no bursting transfer cross that boundary. As a result, no AHB command, either single transfer, or any type of wrapping or incrementing burst, can cross any 1 KB boundary. This means that any AHB transfer is limited to a single context for purposes of AES decryption.

If SR[RRAM] = 1 indicating restricted register access mode, access to these registers is treated as read-as-zero, write-ignored (RAZ/WI).

Diagram



Fields

Field	Function
31-10 SRTADDR	Start Address This field defines the most significant bits of the 0-modulo-1024 byte start address of the memory region for context n.
9-0 —	Reserved

32.7.1.7 AES Region Descriptor Word1 (CTX0_RGD_W1 - CTX3_RGD_W1)

Offset

Register	Offset
CTX0_RGD_W1	D1Ch
CTX1_RGD_W1	D5Ch
CTX2_RGD_W1	D9Ch
CTX3_RGD_W1	DDCh

Function

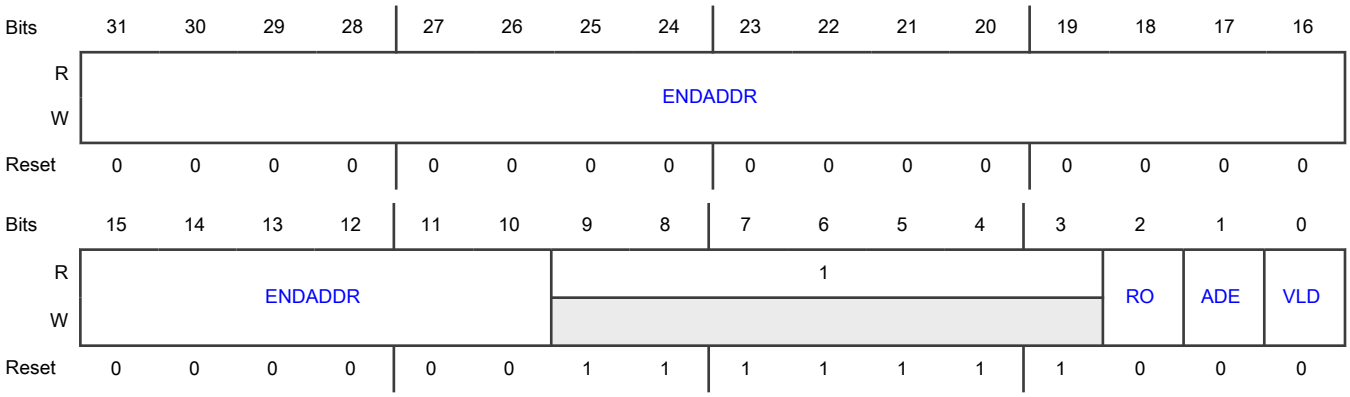
The CTXn_RGD_W1 registers provide a 2-dimensional data structure for local OTFAD storage of 64 bits of memory region descriptor for context “n”. There are two consecutive memory-mapped register words defining the starting and ending addresses for the external flash memory region associated with the given context.

Each memory region context defines a specified section of external flash memory associated with the given context and its associated 128-bit key and 128-bit counter values.

The context memory regions are defined as modulo-1024 bytes to match the AMBA-AHB protocol requirement that no bursting transfer cross that boundary. As a result, no AHB command, either single transfer, or any type of wrapping or incrementing burst, can cross any 1 KB boundary. This means that any AHB transfer is limited to a single context for purposes of AES decryption.

If SR[RRAM] = 1 indicating restricted register access mode, access to these registers is treated as read-as-zero, write-ignored (RAZ/WI).

Diagram



Fields

Field	Function
31-10 ENDADDR	End Address This field defines the most significant bits of the 1023-modulo-1024 byte end address of the memory region for context n.
9-3 —	Reserved
2 RO	Read-Only This field signals that the entire set of context registers (CTXn_KEY[0-3], CTXn_CTR[0-1], CTXn_RGD_W[0-1] are read-only and cannot be modified. This field is sticky and remains asserted until the next system reset. SR[RRAM] provides another level of register access control and is independent of the RO indicator. 0b - The context registers can be accessed normally (as defined by SR[RRAM]). 1b - The context registers are read-only and accesses may be further restricted based on SR[RRAM].
1 ADE	AES Decryption Enable. For accesses hitting in a valid context, this bit indicates if the fetched data is to be decrypted or simply bypassed. 0b - Bypass the fetched data. 1b - Perform the CTR-AES128 mode decryption on the fetched data.
0	Valid

Table continues on the next page...

Table continued from the previous page...

Field	Function
VLD	This field signals if the context is valid or not. 0b - Context is invalid. 1b - Context is valid.

32.7.2 Data organization and endianness considerations

OTFAD implements a little-endian memory organization. This affects the data organization of its programming model registers as well as how it handles memory data.

The OTFAD module follows the standard AES definition on data organization and mapping into the bit-oriented cryptographic algorithms. To review these definitions, recall the following:

"The basic unit for processing the AES algorithm is a byte, a sequence of eight bits treated as single entity. The input, output and Cipher Key bit sequences... are processed as arrays of bytes that are formed by dividing these sequences into groups of eight contiguous bits to form arrays of bytes" [1, pp. 8-9].

Further, if an input, output or key is denoted by the AES standard convention as "a", the bytes in the resulting array can be described as a[*n*], where 0 < *n* < 16. The mapping of this byte array data into the required 128-bit sequence produces:

```
data[127:0] = {a[0], a[1], a[2],..., a[14], a[15]}
```

That is, the 128-bit data is formed as the concatenation of the 0th through the 15th byte array element. The OTFAD hardware performs all the required byte "swapping" to convert the programming model view of the keys and counters, etc. from the little endian 32-bit view into the organization required by the AES standard. As an example, consider the mapping of the 128-bit key from its 32-bit programming model registers for the OTFAD AES engine. Let the little-endian memory image of the 128-bit key be defined as a byte-sized, sixteen element array, a[16]. Note A[16] is intended to be an alternate, but equivalent, description:

```
AES Key Word 0[31:0] = {a[ 3], a[ 2], a[ 1], a[ 0]}

AES Key Word 1[31:0] = {a[ 7], a[ 6], a[ 5], a[ 4]}

AES Key Word 2[31:0] = {a[11], a[10], a[ 9], a[ 8]}

AES Key Word 3[31:0] = {a[15], a[14], a[13], a[12]}
```

This data is converted by the OTFAD into a 128-bit key of the form:

```
AES Key[127:0] = {a[0], a[1], a[2], a[3],..., a[12], a[13], a[14], a[15]}
```

Next, consider the 64-bit counter as an eight element byte array, c[8]:

```
AES Counter Word 0[31:0] = {c[ 3], c[ 2], c[ 1], c[ 0]}

AES Counter Word 1[31:0] = {c[ 7], c[ 6], c[ 5], c[ 4]}
```

Before being used, this data is reorganized into the required 128-bit counter value of the form:

```
AES Counter[127:0]  = {c[ 0], c[ 1], c[ 2], c[ 3],  // bits[127:96]

                       c[ 4], c[ 5], c[ 6], c[ 7],  // bits[ 95:64]

                      {c[ 0], c[ 1], c[ 2], c[ 3]}
```

```
                    ^ {c[ 4], c[ 5], c[ 6], c[ 7]}, // bits[ 63:32]

                      sysAddr[31:4], 4'b0000}      // bits[ 31: 0]
```

To summarize, the OTFAD programming model registers are organized as little-endian byte-size data arrays and the hardware automatically reorganizes the register data into the 128-bit formats required by AES.

Note that since OTFAD is a little-endian design, it processes each 64 bits of external memory data fetched by the QuadSPI using the following mapping for data byte array:

```
mem_wr_data[63:0]

= {data[i+7],data[i+6],data[i+5],data[i+4],data[i+3],data[i+2],data[i+1],data[i+0]}
```

Chapter 33
Ultra Secured Digital Host Controller (uSDHC)

33.1 Chip-specific uSDHC Information

Table 245. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

There are two instances of uSDHC instances. The suggested frequency for uSDHC1 is 200 MHz with boot support for SD/SDIO 3.0 compliance with 200 MHz SDR signaling to support up to 100 MB/s. The suggested frequency for uSDHC2 is 400 MHz with boot support for eMMC 5.1 compliance with HS400 DDR signaling to support up to 400 MB/s.

33.2 Overview

uSDHC provides the interface between the host system and the eMMC, SD card, and SDIO as shown in Figure 240. The module acts as a bridge, passing host bus transactions to the eMMC, SD card, and SDIO by sending commands and performing data accesses to/from the cards. It handles the SD card/SDIO/eMMC protocols at the transmission level.

33.2.1 Block diagram

The following figure illustrates the block diagram for uSDHC. Dual port SRAM is FIFO for write/read. For more information, see Data buffer.

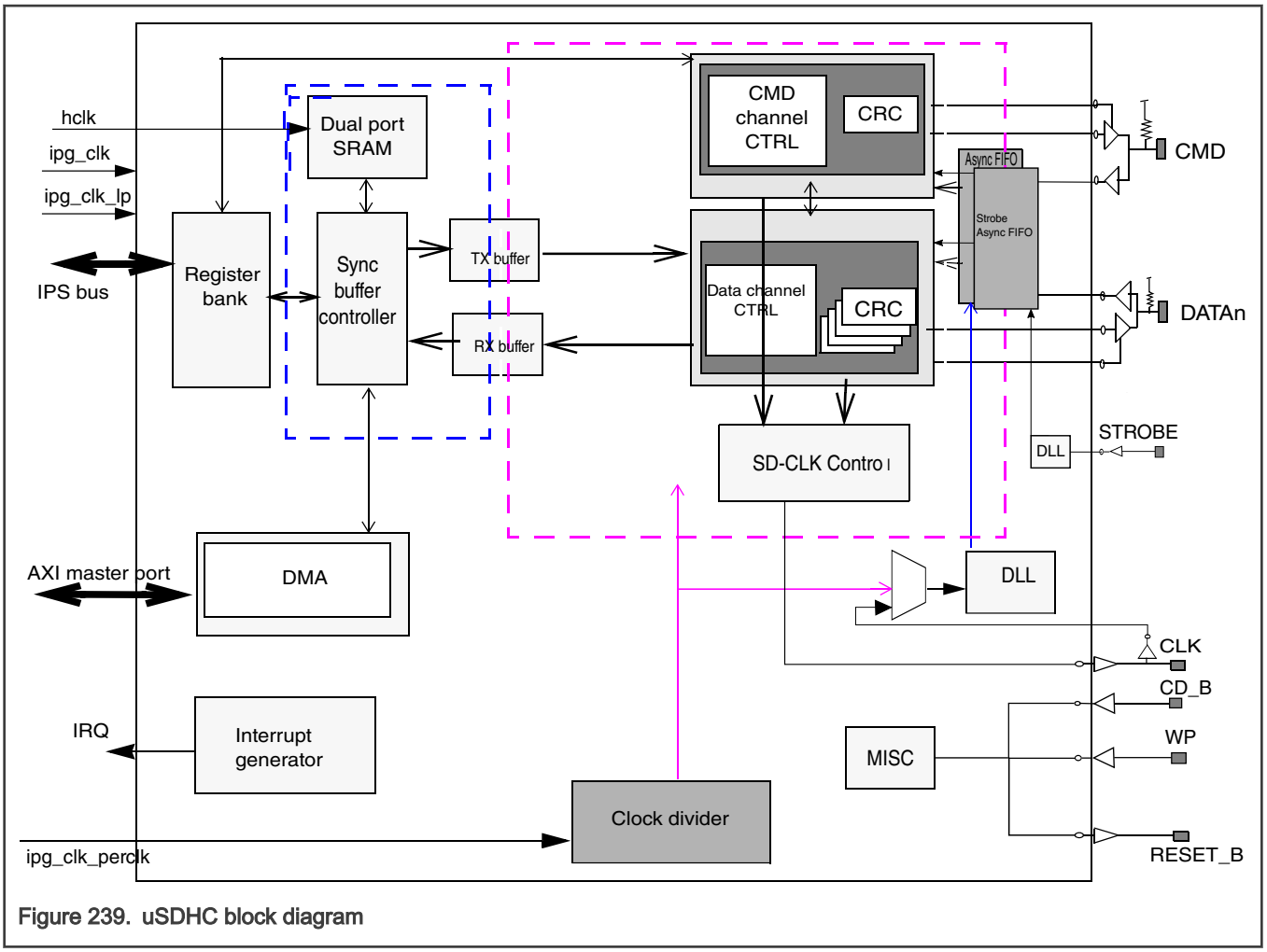


Figure 239. uSDHC block diagram

The figure below shows the System connection of uSDHC:

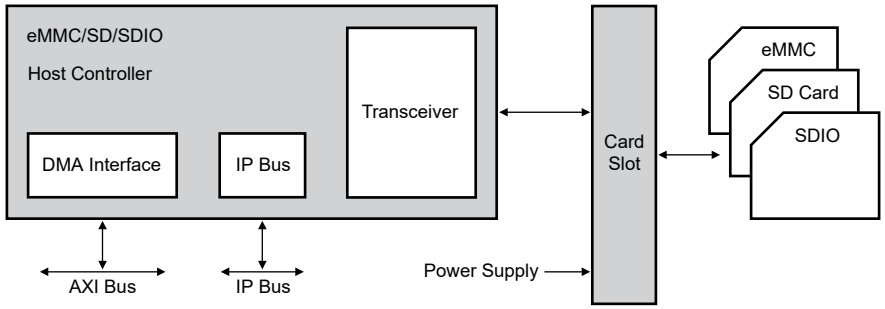


Figure 240. System connection of uSDHC

The following are brief descriptions of the cards supported by uSDHC:

- The Embedded MultiMediaCard (eMMC) is a universal low-cost data storage and communication media designed to cover a wide array of applications including mobile video and gaming. Previous eMMC were based on a 7-pin serial bus with a single data pin, while the new high speed eMMC communication is based on an advanced 11-pin serial bus designed to operate in the low-voltage range.
- The Secure Digital Card (SD) is an evolution of the old eMMC technology. It is specifically designed to meet the security, capacity, performance, and environmental requirements inherent in newly emerging audio and video consumer electronic

devices. The physical form factor, pin assignment, and data transfer protocol are backward compatible with the old eMMC (with some additions).

- Under the SD protocol, system connection can be categorized into memory card, I/O card, and combo card, which have both memory and I/O functions. The memory card invokes a copyright protection mechanism that complies with the security of the SDMI (Secure Digital Music Initiative) standard. The I/O card, which is also known as SDIO, provides high-speed data I/O with low-power consumption for mobile electronic devices.

33.2.2 Features

The list given below shows the features of the uSDHC module:

- Conforms to the SD Host Controller Standard Specification version 2.0/3.0
- Compatible with the eMMC System Specification version 4.2/4.3/4.4/4.41/4.5/5.0/5.1
- Compatible with the SD Memory Card Specification version 3.0 and supports the Extended Capacity SD Memory Card
- Compatible with the SDIO Specification version 2.0/3.0
- Designed to work with SD Memory, miniSD Memory, SDIO, miniSDIO, SD Combo, eMMC, MMC plus, and MMC RS cards
- For card bus clock frequency: See the product Data Sheet for the clock frequency of each supported mode.
- Supports 1-bit/4-bit SD and SDIO modes, and 1-bit/4-bit/8-bit eMMC modes
  - Up to 832 Mbps of data transfer for SDIO using four parallel data lines in the Single Data Rate (SDR) mode
  - Up to 400 Mbps of data transfer for SDIO using four parallel data lines in the Dual Data Rate (DDR) mode
  - Up to 832 Mbps of data transfer for SDXC cards using four parallel data lines in the Single Data Rate (SDR) mode
  - Up to 400 Mbps of data transfer for SDXC card using four parallel data lines in the Dual Data Rate (DDR) mode
  - Up to 1600 Mbps of data transfer for eMMC using eight parallel data lines in the Single Data Rate (SDR) mode
  - Up to 3200 Mbps of data transfer for eMMC using eight parallel data lines in the Dual Data Rate (DDR) mode
- Supports single block/multi-block read and write
- Supports block sizes of 1 ~ 4096 bytes
- Supports the write protection switch for write operations
- Supports both synchronous and asynchronous abort
- Supports pause during the data transfer at block gap
- Supports SDIO Read Wait and Suspend Resume operations
- Supports Auto CMD12 for multi-block transfer
- Host can initiate non-data transfer command while data transfer is in progress
- Allows cards to interrupt the host in 1-bit and 4-bit SDIO modes; also supports interrupt period
- Embodies two fully configurable 256x32-bit FIFO for read/write data
- Supports internal DMA capabilities
- Supports voltage selection by configuring vendor-specific register bit
- Supports Advanced DMA to perform linked memory access
- Supports Command Queue mechanism

NOTE

This block can support all the above-listed speed mode and maximum data throughput. However, these may be specific to the device. See the corresponding chip-specific information or the device data sheet for accurate details.

33.3 Functional description

The following sections provide a brief functional description of the major system blocks, including the data buffer, DMA AXI interface, register bank as well as IP Bus interface, dual-port memory wrapper, data/command controller, clock and reset manager, and clock generator.

33.3.1 Modes and operations

33.3.1.1 Data transfer modes

The uSDHC module can select the following modes for data transfer:

- SD 1-bit
- SD 4-bit
- eMMC 1-bit
- eMMC 4-bit
- eMMC 8-bit
- Identification mode (up to 400 kHz)
- eMMC full-speed mode (up to 26 MHz)
- eMMC high-speed mode (up to 52 MHz)
- eMMC HS200 mode (up to 200 MHz)
- eMMC HS400 mode (200 MHz both edges)
- eMMC DDR mode (52 MHz both edges)
- SD card or SDIO full-speed mode (up to 25 MHz)
- SD card or SDIO high-speed mode (up to 50 MHz)
- SD card or SDIO UHS-I mode (up to 208 or 100 MHz in SDR mode, up to 50 MHz in the DDR mode)

NOTE

This block can support all the above listed speed mode and maximum clock frequency. However, these may be specific to the device. See the corresponding chip-specific information or the device data sheet for accurate details.

33.3.2 Data buffer

The uSDHC module uses one configurable data buffer to transfer data between the system bus (IP bus or advanced extensible interface (AXI) bus) and the SD card in an optimized manner, maximizing throughput between the two clock domains (IP peripheral clock and the master clock). The buffer is used as a temporary storage for transferring data between the host system and the card. The watermark levels for read and write are both configurable and can range between 1 to 128 words. When using internal DMA mode, The watermark level for read should be configured to 16 and watermark level for write should be configured to integral multiple of 16 to get max AXI bus throughput. The next figure provides the uSDHC buffer scheme as buffer control and buffer RAM wrapper.

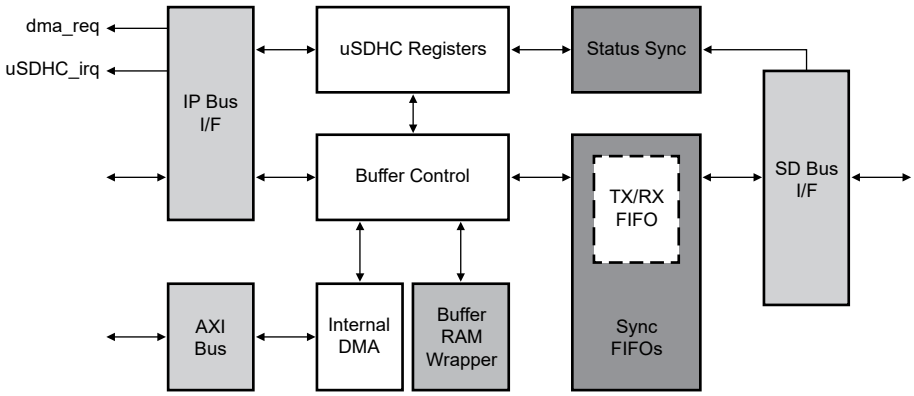


Figure 241. uSDHC buffer scheme

Here are 2 transfer modes to access the data buffer:

- CPU polling mode:
  - For a host-read operation, when the number of words received in the buffer meets or exceeds the RD_WML watermark value, by polling the BRR bit, the host driver can read the Buffer Data Port register to fetch the amount of words set in the RD_WML register from the buffer. The write operation is similar. For more information on the process of writing operation, see Write operation sequence.
- Internal DMA mode (includes simple and advanced DMA accesses):
  - The internal DMA access, either by simple or advanced DMA, is over the AXI bus.

For a read operation, when there are more words in the buffer than the amount set in WTMK_LVL[RD_WML], the internal DMA starts fetching data over the AXI bus.

The Write operation functions in a similar manner—sequential and contiguous access is necessary to ensure that the pointer address value is correct. Random or skipped access is not possible. The byte order, by reset, is little endian mode. The actual byte order is swapped inside the buffer, according to the endian mode configured by software (see the following figures). For a host write operation, the byte order is swapped after the data is fetched from the buffer and ready to send to the SD Bus. For a host read operation, the byte order is swapped before the data is stored in the buffer.

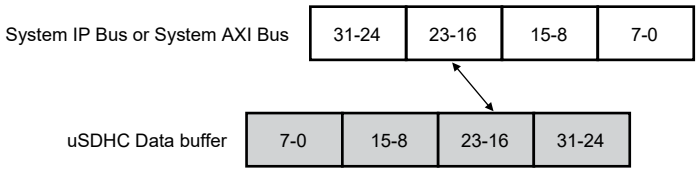


Figure 242. Data swap between system bus and uSDHC data buffer in the byte little endian mode

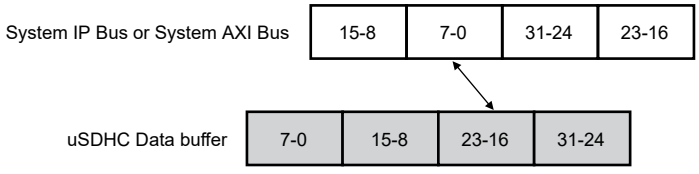


Figure 243. Data swap between system bus and uSDHC data buffer in half word big endian mode

33.3.2.1 Write operation sequence

There are 2 ways to write data into the buffer when the user transfers data to the card:

- Processor core polling through the BWR bit in the Interrupt Status register (interrupt or polling)
- Internal DMA

When the internal DMA is not used, the DMAEN bit in the Transfer Type register is not set when the command is sent, uSDHC asserts a DMA request when the amount of buffer space exceeds the value set in WTMK_LVL[WR_WML] and is ready for receiving new data. At the same time, uSDHC sets the BWR bit. The buffer write ready interrupt is generated if it is enabled by software.

When internal DMA is used, uSDHC does not inform the system before all the required number of bytes are transferred (if no error is encountered). When an error occurs during the data transfer, uSDHC aborts the data transfer and abandons the current block. The host driver should read the contents of the DMA System Address register to obtain the starting address of the abandoned data block. If the current data transfer is in multi-block mode, uSDHC does not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The host driver sends CMD12 in this scenario and restarts the write operation from that address. It is recommended that a software reset for Data be applied before the transfer is restarted.

The uSDHC module does not start data transmission until the buffer has been filled with the number of words set in the WR_WML register. If the buffer is empty and the host system does not write data in time, uSDHC stops the CLK to avoid the data buffer underrun situation.

33.3.2.2 Read operation sequence

There are 2 ways to read data from the buffer when the data is received from the card:

- Processor core polling through the BRR bit in the Interrupt Status register (interrupt or polling)
- Internal DMA

When internal DMA is not used (DMAEN bit in Transfer Type register is not set when the command is sent), uSDHC asserts a DMA request when the amount of data exceeds the value set in the RD_WML register, which is available and ready for system fetching data. At the same time, uSDHC sets the BRR bit. The buffer read ready interrupt is generated if it is enabled by the software.

When internal DMA is used, uSDHC does not inform the system before all the required number of bytes are transferred (if no error is encountered). When an error occurs during the data transfer, uSDHC aborts the data transfer and abandons the current block. The host driver should read the content of the DMA System Address register to get the starting address of the abandoned data block. If the current data transfer is in multi-block mode, uSDHC does not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The host driver sends CMD12 in this scenario and restarts the read operation from that address. It is recommended that a software reset (in register RSTD) for data be applied before the transfer is restarted.

For any read transfer mode, uSDHC does not start data transmission until the number of words set in the RD_WML register are in buffer. If the buffer is full and the host system does not read data in time, uSDHC stops the CLK to avoid the data buffer overrun situation.

33.3.2.3 Data buffer and block size

In the uSDHC module, the data buffer can hold up to 256 words (32-bit) and the watermark levels for write and read can be configured accordingly. The user needs to know the buffer size for the buffer operation during a data transfer to utilize it in the most optimized way. The host driver may configure the value according to the system and requirement.

During a multi-block data transfer, the block length can be set to any value between 1 and 4096 bytes, satisfying the requirements of the external card. The only restriction is from the external card, which can be limited in size or support of a partial block access (which is not the integer times of 512 bytes). That means, the largest block size is 512 bytes.

As uSDHC treats each block individually, for block sizes which are not multiples of four (not word-aligned), stuffed bytes are required at the end of each block. For example, if the block size is 7 bytes and there are 12 blocks to write, the software side must write twice for each block. For each block, the ending byte is abandoned by uSDHC because it only sends 7 bytes to the card and picks data from the following system write, resulting in 24 beats of write access in total.

33.3.2.4 Dividing large data transfer

This SDIO command CMD53 definition limits the maximum data size of data transfers according to the following formula:

Max data size = Block size x Block count

The length of a multiple block transfer needs to be in multiples of block size. If the total data length cannot be divided evenly into a multiple of the block size, then there are two options to transfer the data. These two options depend on the function and the card design. Option 1 is for the host driver to split the transaction. The remainder of the block size data is then transferred by using a single block command at the end. Option 2 is to add dummy data in the last block to fill the block size. For option 2, the card must manage the removal of the dummy data. Only for write, uSDHC sends data to the card.

See Figure 244 for an example showing the division of large data transfers, assuming a kind of WLAN SDIO that only supports a block size of up to 64 bytes. Although uSDHC supports a block size of up to 4096 bytes, SDIO can only accept a block size of less than 64 bytes, so the data must be divided (see the example below).

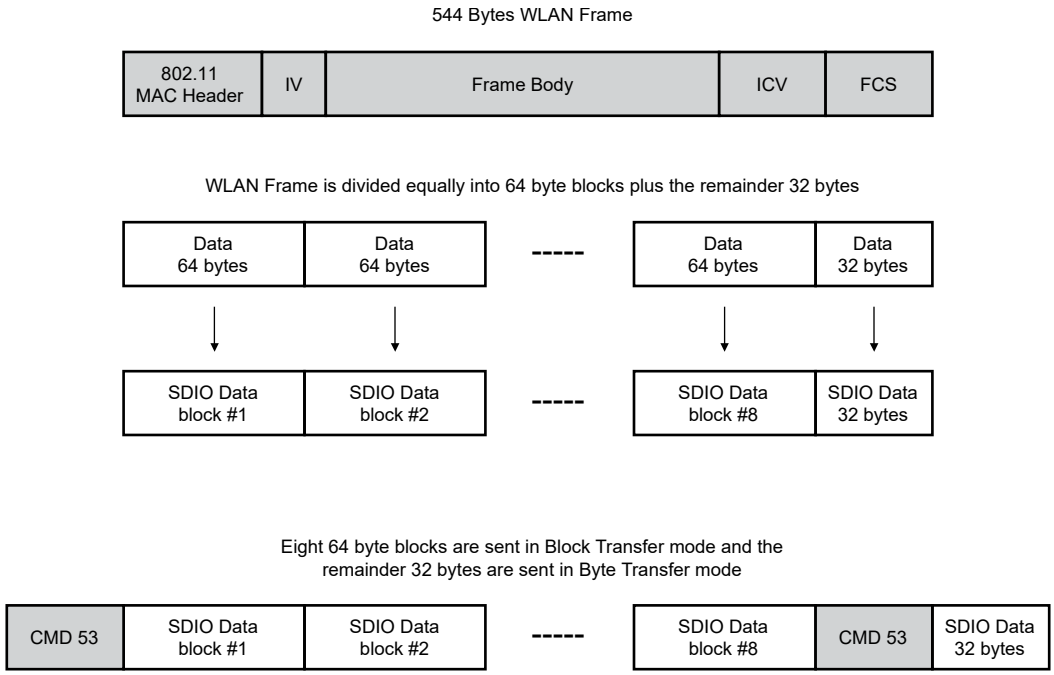


Figure 244. Example for dividing large data transfers

33.3.3 DMA AXI interface

The internal DMA implements a DMA engine and the AXI master. When the internal DMA is enabled, but the BWR and BRR bits are set if the BWRSEN and BRRSEN bits have been set in the Interrupt Status Enable register. See Figure 245 for an illustration of the DMA AXI interface block.

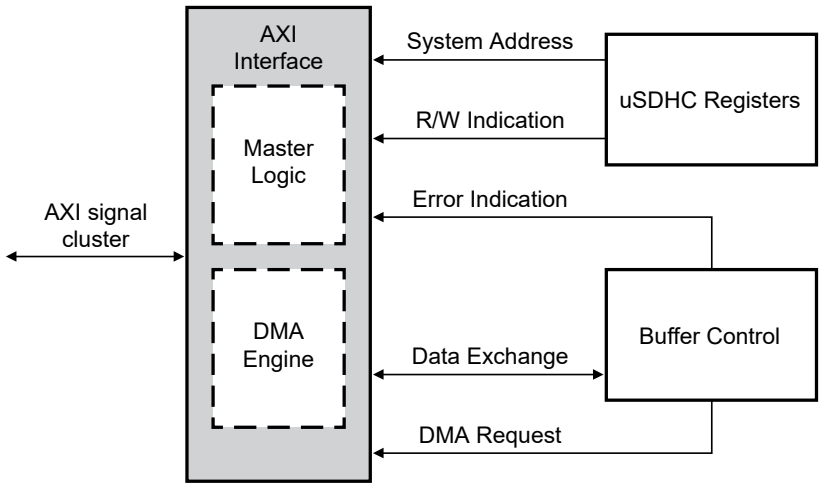


Figure 245. DMA AXI interface block

33.3.3.1 Internal DMA request

If the watermark level requirement is met in data transfer or if the last data of current block is ready in the data buffer, and the Internal DMA is enabled, the data buffer block sends a DMA request to AXI interface.

The delay in response from the internal DMA engine depends on the system AXI bus loading and the priority assigned to uSDHC. The DMA engine does not respond to the request during its burst transfer, but is ready to serve as soon as the burst is over. The data buffer de-asserts the request if the data buffer space (for write) or bytes in data buffer is smaller than the watermark level. Upon access to the buffer by internal DMA, the data buffer updates its internal buffer pointer, and when the watermark level is satisfied or the last data of the current block is ready in the data buffer, another DMA request is sent.

The data transfer is in the block unit, and the subsequent watermark level is always automatically set as the remaining number of words. For instance, for a multi block data read with each block size of 31 bytes, and the watermark level set to six words (24 bytes). After the first burst transfer, if there are greater than or equal to two words in the buffer, which might contain some data of the next block), another DMA request is sent. This is because the remaining number of words to transfer for the current block is ceiling((31 - 6 * 4) / 4) = 2. The uSDHC module reads two words out of the buffer, with seven valid bytes and one stuffed byte.

33.3.3.2 AXI master interface

It is possible that the internal AXI DMA engine could fail during the data transfer. Upon detection of an AXI bus error during DMA transfer, the DMA engine stops the transfer and goes to the idle state. At that point, the internal data buffer stops receiving incoming data and sending out data. The DMAE bit in the Interrupt Status register is generated to host CPU to report a bus error condition.

After the DMAE interrupt is received, the software sends a CMD12 to abort the current transfer and read the DS_ADDR bits of the DMA System Address register to get the starting address of the corrupted block. After the DMA error is fixed, the software should apply a data reset and restart the transfer from this address to recover the corrupted block.

33.3.3.3 ADMA engine

In the SD host controller standard, a new DMA transfer algorithm called the Advanced DMA (ADMA) is defined. For simple DMA, after the page boundary is reached, a DMA interrupt is generated and the new system address is programmed by the host driver. The ADMA defines the programmable descriptor table in the system memory. The host driver can calculate the system address at the page boundary and program the descriptor table before executing ADMA. It reduces the frequency of interrupts to the host system. Therefore, higher speed DMA transfers could be realized because the host MCU intervention is not be needed during long DMA-based data transfers.

There are two types of ADMA in host controller: ADMA1 and ADMA2. ADMA1 can support data transfer of 4KB aligned data in system memory, and ADMA2 eliminates the restriction so that the data of any location and any size can be transferred in system memory. Their formats of Descriptor Table are different.

ADMA can recognize all kinds of descriptors defined in SD host controller Standard, and if the "End" flag is detected in the descriptor, ADMA stops after this descriptor is processed.

33.3.3.3.1 ADMA concept and descriptor format

ADMA1 includes the following descriptors:

- Valid/invalid descriptor
- Nop descriptor
- Set data length descriptor
- Set data address descriptor
- Link descriptor
- Interrupt flag and end flag in descriptor

ADMA2 includes the following descriptors:

- Valid/invalid descriptor
- Nop descriptor
- Rsv descriptor (new in ADMA2)
- Set data length and address descriptor
- Link descriptor
- Interrupt flag and end flag in descriptor

ADMA starts read/write operation after it reaches the tran state, using the data length and data address analyzed from most recent descriptor(s).

For ADMA1, the valid data length descriptor is the last set type descriptor before the tran type descriptor. Every tran type descriptor triggers a transfer, and the transfer data length is extracted from the most recent set type descriptor. If there is no set type descriptor after the previous Trans descriptor, the data length is the value for previous transfer, or 0 if no set descriptor is ever met.

For ADMA2, the tran type descriptor contains both data length and transfer data address, so the tran type descriptor itself can start a data transfer.

See Figure 246 for the format of the descriptor table for ADMA1.

Address/ Page Field		Address/ Page Field		Attribute Field					
31	12	11	6	5	4	3	2	1	0
Address or Data Length		000000		Act 2	Act 1	0	Int	End	Valid

Act 2	Act1	Symbol	Comment	31- 28	27- 12
0	0	Nop	No Operation	Don't Care	
0	1	Set	Set Data Length	0000	Data Length
1	0	Tran	Transfer Data	Data Address	
1	1	Link	Link Descriptor	Descriptor Address	

Valid	Valid = 1 indicates this line of descriptor is effective. If Valid = 0 generate ADMA Error Interrupt and stop ADMA.
End	End = 1 indicates current descriptor is the ending one.
Int	Int = 1 generates DMA Interrupt when this descriptor is processed.

Figure 246. Format of the ADMA1 descriptor table

See Figure 247 for the concept and access method of the descriptor table for ADMA1.

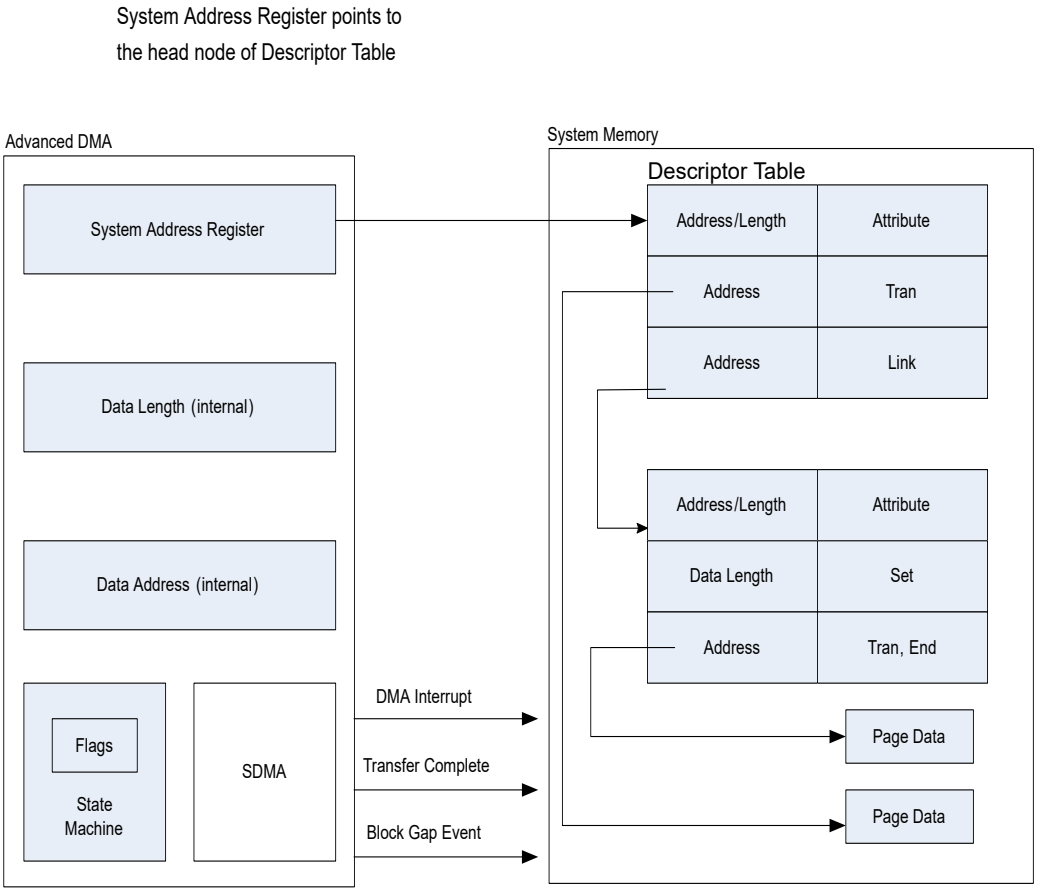


Figure 247. Concept and access method of the ADMA1 descriptor table

The figure below explains the ADMA2 format. ADMA2 deals with the lower 32-bit first, and then the higher 32-bit. If the 'Valid' flag of descriptor is 0, it ignores the higher 32-bit. The Address field should be set to word aligned (lower 2-bit is always set to 0). Data length is in byte unit.

Address Field		Length		Reserved		Attribute Field					
63	32	31	16	15	06	05	04	03	02	01	00
32-bit Address		16-bit length		0000000000		Act 2	Act 1	0	Int	End	Valid

Act 2	Act1	Symbol	Comment	Operation
0	0	Nop	No Operation	Don't Care
0	1	Rsv	Reserved	Same as Nop. Read this line and go to next one
1	0	Tran	Transfer Data	Transfer data with address and length set in this descriptor line
1	1	Link	Link Descriptor	Link to another descriptor

Valid	Valid = 1 indicates this line of descriptor is effective. If Valid = 0 generate ADMA Error Interrupt and stop ADMA.
End	End = 1 indicates current descriptor is the ending one.
Int	Int = 1 generates DMA Interrupt when this descriptor is processed.

Figure 248. Format of the ADMA2 descriptor table

See Figure 249 for the concept and access method of the descriptor table for ADMA2.

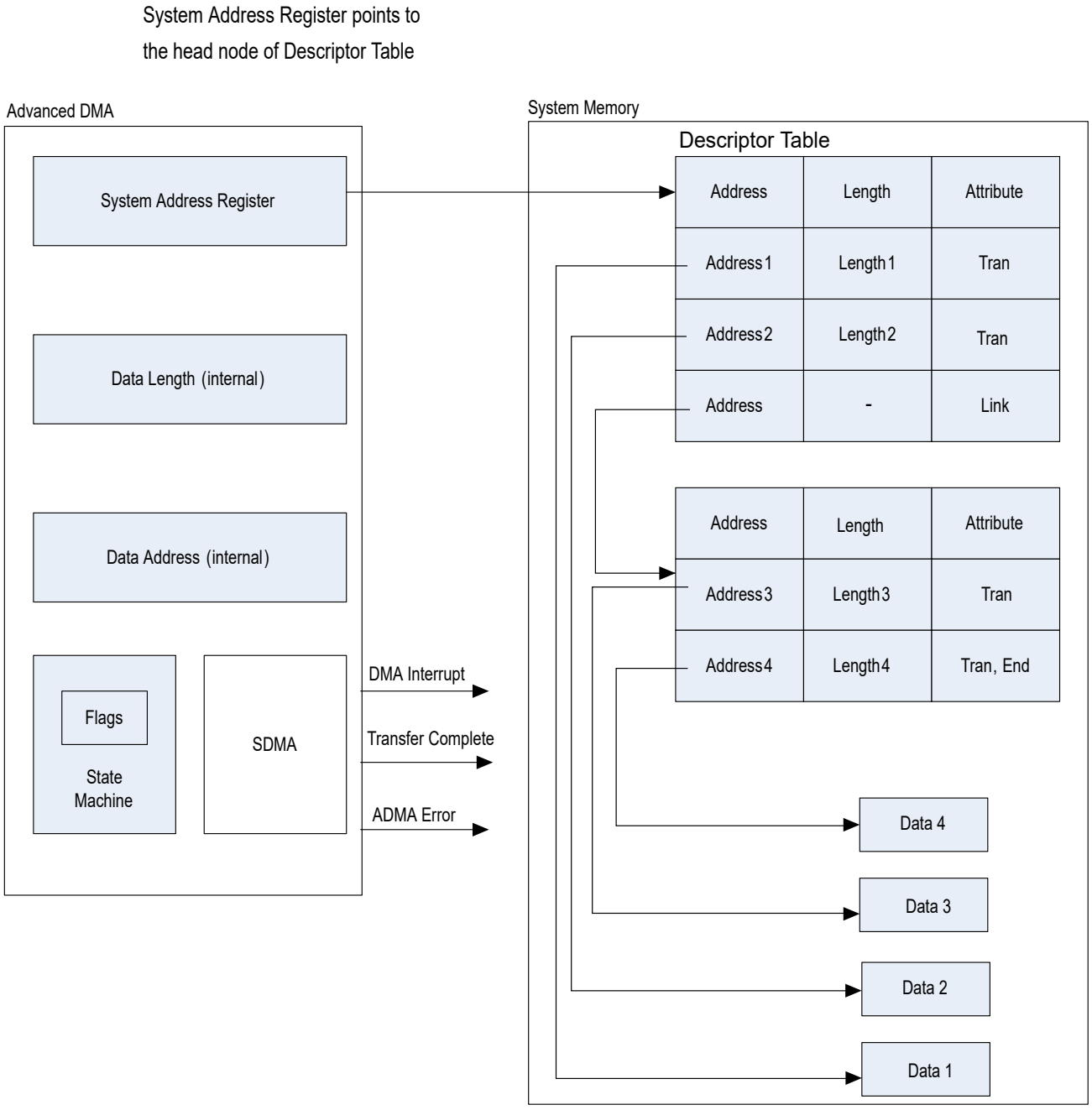


Figure 249. Concept and access method of the ADMA2 descriptor table

33.3.3.3.2 ADMA interrupt

If the interrupt flag descriptor is set, ADMA generates an interrupt according to various types of descriptors.

For ADMA1:

- Set type of descriptor: interrupt is generated when data length is set.
- Tran type descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.

- Nop type of descriptor: interrupt is generated just after this descriptor is fetched.

For ADMA2:

- Tran type of descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.
- Nop/Rsv type of descriptor: interrupt is generated just after this descriptor is fetched.

33.3.3.3.3 ADMA error

The ADMA stops whenever an error is encountered. These errors include:

- Fetching descriptor error
- AXI response error
- Data length mismatch error

An ADMA descriptor error is generated when it fails to detect a "valid" flag in the descriptor. If an ADMA descriptor error occurs, the interrupt is not generated even if the "Interrupt" flag of this descriptor is set.

When BLKCNTEN bit is set, data length set in register BLK_ATT must be equal to the whole data length set in descriptor, otherwise data length mismatch error is generated.

When BLKCNTEN bit is not set, then the whole data length set in descriptor is a multiple of block lengths; otherwise, when data set in the descriptor nodes are not performed at block boundaries, then data mismatch errors occur.

33.3.4 Register bank with IP bus interface

Register accesses through the IP bus interface are on the register bank. See Figure 250 for the block diagram.

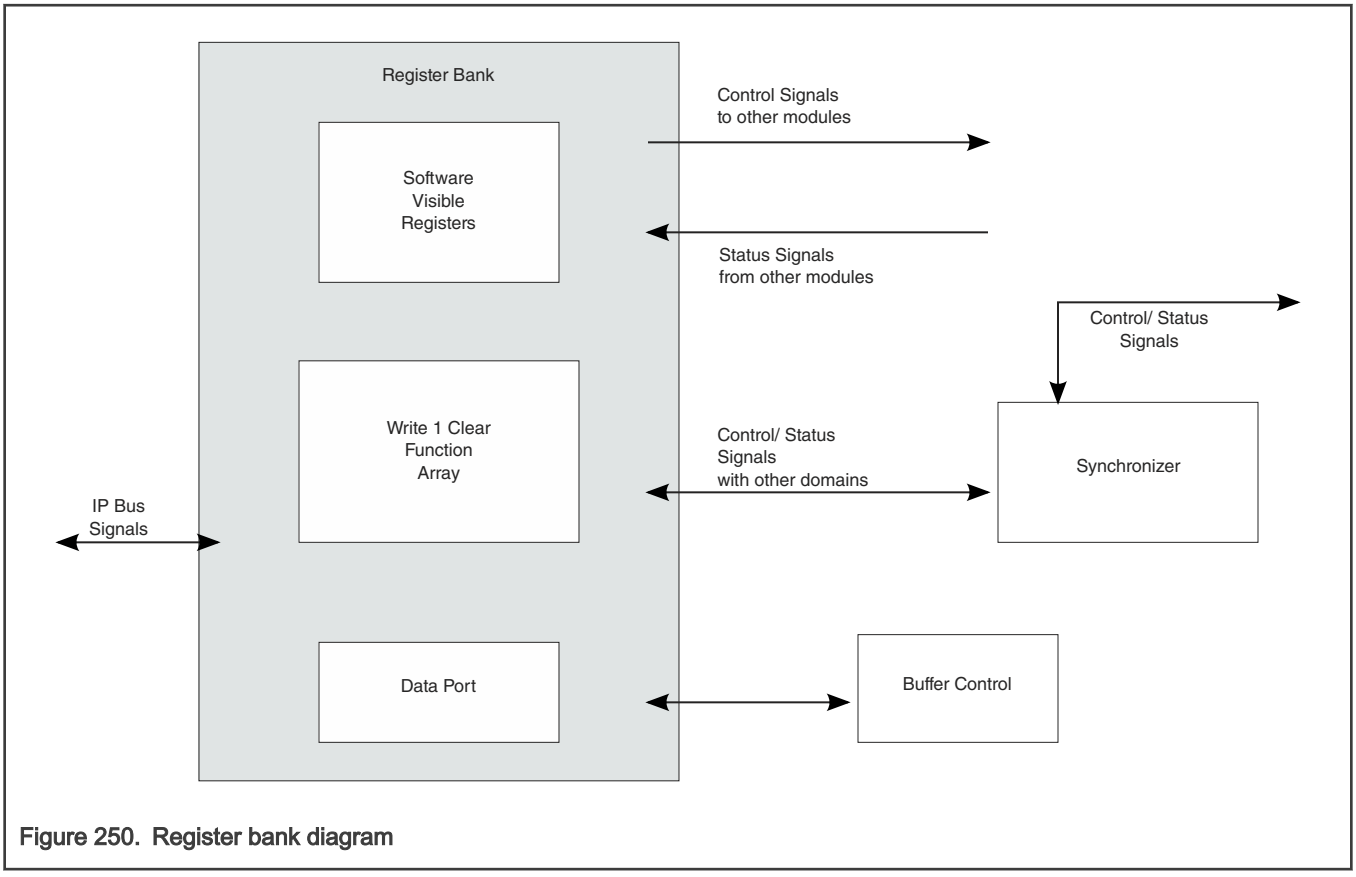


Figure 250. Register bank diagram

Only a 32-bit access is allowed, and no partial read/write is supported; therefore, all accesses are word aligned.

Access to an unimplemented address within the register memory map space does not generate a transfer error.

33.3.4.1 SD protocol unit

The SD protocol unit deals with all SD protocol affairs.

The SD protocol unit performs the following functions:

- Acts as the bridge between the internal buffer and the SD bus
- Sends the command data as well as its argument serially
- Stores the serial response bit stream into corresponding registers
- Detects the bus state on the CMD/DAT lines
- Monitors the interrupt from SDIO
- Asserts the read wait signal
- Gates off the SD clock when buffer is announcing danger status
- Detects the write protect state

The SD protocol unit consists of four sub-modules:

- SD control misc
- Command control
- Data control
- Clock control

33.3.4.2 SD control miscellaneous

In the SD control miscellaneous unit:

- The card detection (including CD_B and DATA3 for card detection) and card interrupt are implemented.
- The module monitors the signal level on all the eight data lines and the command lines. It directly routes the level values into the register bank.
- The module also detects the Write Protect (WP) line. If WP is active, writes to memory and combo cards are ignored.

33.3.4.3 SD clock control

If the internal data buffer is near full (for read) or near empty (for write), the SD clock must be gated off to avoid buffer over/under-run. This module asserts the gate of the output SD clock to shut the clock off. After the buffer has space (for read) or has data (for write), the clock gate of this module opens, and the SD clock is active again.

33.3.4.4 Command control

The Command Control module deals with the transactions on the CMD line.

See Figure 251 for an illustration of the structure for the Command CRC shift register.

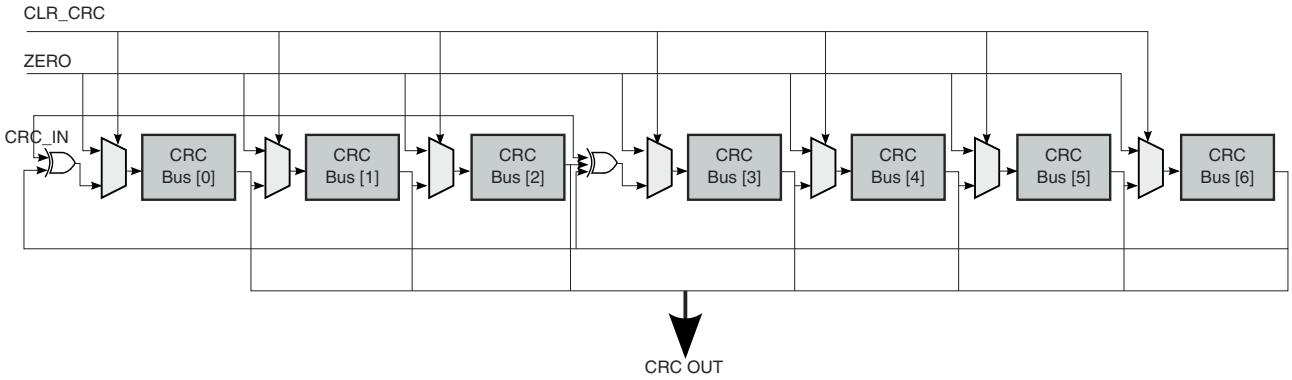


Figure 251. Command CRC shift register

The CRC polynomials for the CMD are as follows:

```
Generator polynomial: G(x) = x^7 + x^3 + 1
M(x) = (first bit) * x^n + (second bit) * x^n-1 +...+ (last bit) * x^0
CRC[6:0] = Remainder [(M(x) * x^7) / G(x)]
```

33.3.4.5 Data control

The data agent deals with the transactions on the eight data lines. Moreover, this module also detects the busy state on the DATA0 line and generates the read wait state by the request from the transceiver.

The CRC polynomials for the data are as follows:

```
Generator polynomial: G(x) = x^16 + x^12 + x^5 +1

M(x) = (first bit) * x^n + (second bit) * x^n-1 +...+ (last bit) * x^0

CRC[15:0] = Remainder [(M(x) * x^16) / G(x)]
```

33.3.5 Card insertion and removal detection

The uSDHC module uses either the DATA3 pin or the CD_B pin to detect card insertion or removal. It is controlled by SoC pad or other logic. When there is no card on the eMMC/SD bus, the DATA3 is pulled to a low voltage level by default.

When any card is inserted to or removed from the socket, uSDHC detects the logic value changes on the DATA3 pin and generates an interrupt. When the DATA3 pin is not used for card detection (for example, it is implemented in GPIO), the CD_B pin must be connected for card detection. Whether DATA3 is configured for card detection or not, the CD_B pin is always a reference for card detection. Whether the DATA3 pin or the CD_B pin is used to detect card insertion, uSDHC sends an interrupt (if enabled) to inform the Host system that a card is inserted.

33.3.6 Power management and wakeup events

When there is no operation between uSDHC and the card through the SD bus, the user can completely disable the peripheral clock and base clock in the chip-level clock control module to save power. When the user needs to use uSDHC to communicate with the card, it can enable the clock and start the operation.

In some circumstances, when the clocks to uSDHC are disabled, for instance, when the system is in low-power mode, there are some events for which the user needs to enable the clock and handle the event. These events are called wakeup interrupts. The uSDHC module can generate these interrupts even when there are no clocks enabled. The three interrupts that can be used as wakeup events are these:

- Card removal interrupt
- Card insertion interrupt
- Interrupt from SDIO (not available in multiple block data transfers)

These three wakeup events (or wakeup interrupts) can also be used to wakeup the system from low-power modes.

NOTE

To make the interrupt a wakeup event, when all the clocks to uSDHC are disabled or when the entire system is in low power mode, the corresponding wakeup enabled bit needs to be set. Refer to Protocol Control (PROT_CTRL) for more information on the uSDHC Protocol Control register.

33.3.6.1 Setting wakeup events

For uSDHC to respond to a wakeup event, the software must set the respective wakeup enable bit before the CPU enters the sleep mode. Before the software disables the host clock, it should ensure that all the following conditions have been met:

- No read or write transfer is active
- Data and command lines are not active
- No interrupts are pending
- Internal data buffer is empty

33.3.7 eMMC fast boot

The Embedded MultiMediaCard (eMMC4.3 or later) specification adds a fast boot feature that requires hardware support. There are two types of fast boot modes in the eMMC4.3 or later specification: boot operation and alternative boot operation. Each type also has with-acknowledge and without-acknowledge modes.

In the boot operation mode, the master (eMultiMediaCard host) can read boot data from the slave (eMMC device) by keeping CMD line low after power-on, or sending CMD0 with argument + 0xFFFFFFFA (optional for slave), before issuing CMD1.

NOTE

For the eMMC4.3 setting, please see the eMMC4.3 specification.

33.3.7.1 Boot operation

If the CMD line is held low for 74 clock cycles and more after power-up before the first command is issued, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within one second after the CMD line goes low, the slave starts to send the first boot data to the master on the DATA line(s). The master must keep the CMD line low to read all of the boot data.

NOTE

For the purposes of this documentation, fast boot is called "normal fast boot mode".

If boot acknowledge is enabled, the slave has to send acknowledge pattern '010' to the master within 50ms after the CMD line goes low. If boot acknowledge is disabled, the slave does not send out acknowledge pattern '010'.

The master can terminate the boot mode with the CMD line high.

The boot operation is terminated when all the contents of the enabled boot data are sent to the master. After the boot operation is executed, the slave gets ready for the CMD1 operation and the master needs to start a normal eMMC initialization sequence by sending CMD1.

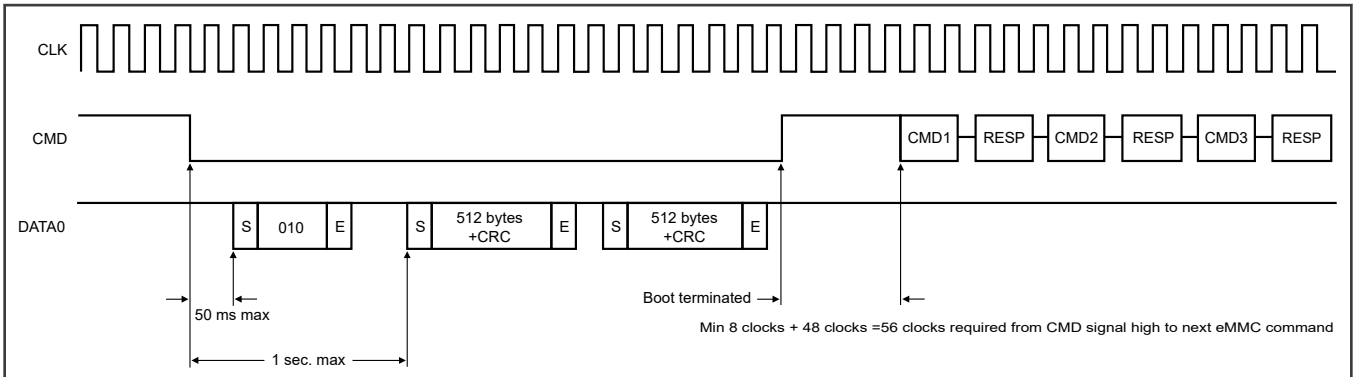


Figure 252. Embedded MultiMediaCard state diagram (normal boot mode)

33.3.7.2 Alternative boot operation

This boot function is optional for the device. If bit 0 in the extended CSD byte[228] is set to '1', the device supports the alternative boot operation.

After power-up, if the host issues CMD0 with the argument of 0xFFFFFFFA after 74 clock cycles, before CMD1 is issued or the CMD line goes low, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within one second after CMD0 with the argument of 0xFFFFFFFA is issued, the slave starts to send the first boot data to the master on the DATA line(s).

If boot acknowledge is enabled, the slave must send the acknowledge pattern '010' to the master within 50ms after the CMD0 with the argument of 0xFFFFFFFA is received. If boot acknowledge is disabled, the slave does not send out acknowledge pattern '010'.

The master can terminate the boot mode by issuing CMD0 (Reset).

Boot operation is terminated when all the contents of the enabled boot data are sent to the master. After the boot operation is executed, the slave gets ready for the CMD1 operation and the master needs to start a normal eMMC initialization sequence by sending CMD1.

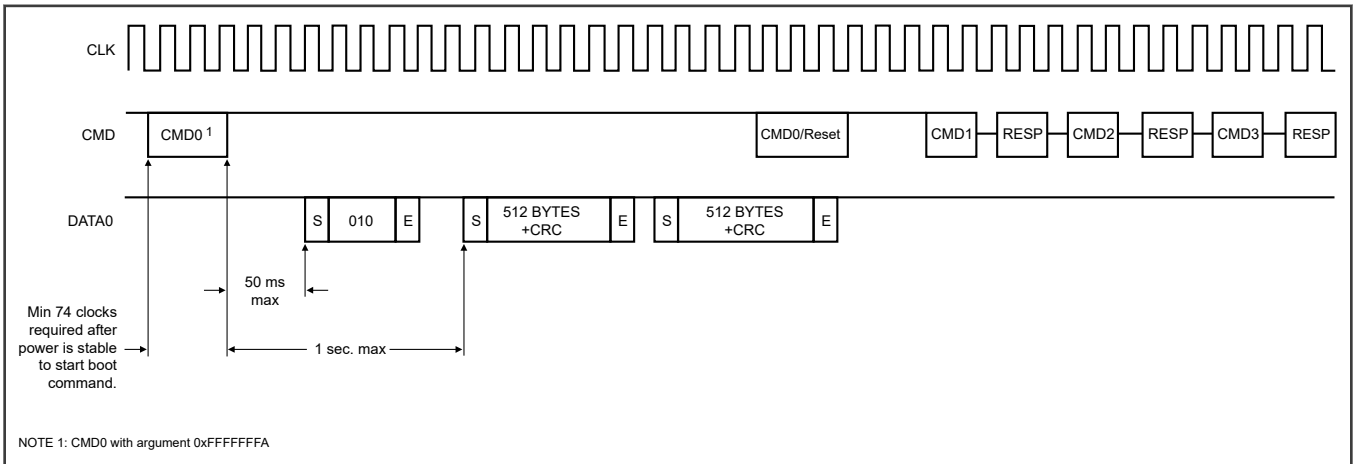


Figure 253. Embedded MultiMediaCard state diagram (alternative boot mode)

33.3.8 Commands for SD card, SDIO, and eMMC

A table containing the list of commands for the eMMC/SD card/SDIO is provided here.

Refer to the corresponding specifications for more details about the command information.

There are four kinds of commands defined to control the SD card, SDIO, and eMMC:

- Broadcast commands (bc), no response
- Broadcast commands with response (bcr), response from all cards simultaneously
- Addressed (point-to-point) commands (ac), no data transfer on the DATA
- Addressed (point-to-point) data transfer commands (adtc)

Response: A response is a token that is sent from the card to the host as an answer to a previously received command. A response is transferred serially on the CMD line.

Table 246. Commands for eMMC/SD card/SDIO

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	Resets all eMMC and SD memory cards to idle state.
CMD1	bcr	[31:0] OCR without busy	R3	SEND_OP_COND	Asks all eMMC and SD Memory cards in idle state to send their operation conditions register contents in the response on the CMD line.
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	Asks all cards to send their CID numbers on the CMD line.
CMD3[1]	ac	[31:6] RCA [15:0] stuff bits	R1 R6 (SDIO)	SET/ SEND_RELATIVE_ADDR	Assigns relative address to the card.
CMD4	bc	[31:0] DSR [15:0] stuff bits	-	SET_DSR	Programs the DSR of all cards.
CMD5	bc	[31:0] OCR without busy	R4	IO_SEND_OP_COND	Asks all SDIO in idle state to send them operation conditions register contents in the response on the CMD line.
CMD6[2]	adtc	[31] Mode 0: Check function 1: Switch function [30:8] Reserved for function groups 6 ~ 3 (All 0 or 0xFFFF) [7:4] Function group1 for command system [3:0] Function group2 for access mode	R1	SWITCH_FUNC	Checks switch ability (mode 0) and switch card function (mode 1). Refer to "SD Physical Specification V1.1" for more details.
CMD6[3]	ac	[31:26] Set to 0 [25:24] Access	R1b	SWITCH	Switches the mode of operation of the selected card or modifies the EXT_CSD registers. Refer to "The

Table continues on the next page...

Table 246. Commands for eMMC/SD card/SDIO (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
		[23:16] Index [15:8] Value [7:3] Set to 0 [2:0] Cmd Set			Embedded MultiMediaCard System Specification Version 4.0 Final draft 2" for more details.
CMD7	ac	[31:6] RCA [15:0] stuff bits	R1b	SELECT/ DESELECT_CARD	Toggles a card between the stand-by and transfer states or between the programming and disconnect states. In both cases, the card is selected by its own relative address and gets deselected by any other address. Address 0 deselects all.
CMD8	adtc	[31:0] stuff bits	R1	SEND_EXT_CSD	The card sends its EXT_CSD register as a block of data, with a block size of 512 bytes.
CMD9	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CSD	Addressed card sends its card-specific data (CSD) on the CMD line.
CMD10	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CID	Addressed card sends its card-identification (CID) on the CMD line.
CMD11	adtc	[31:0] data address	R1	READ_DAT_UNTIL_STOP	Reads data stream from the card, starting at the given address, until a STOP_TRANSMISSION follows.
CMD12	ac	[31:0] stuff bits	R1b	STOP_TRANSMISSION	Forces the card to stop transmission.
CMD13	ac	[31:6] RCA [15:0] stuff bits	R1	SEND_STATUS	Addressed card sends its status register.
CMD14	Reserved				
CMD15	ac	[31:6] RCA [15:0] stuff bits	-	GO_INACTIVE_STATE	Sets the card to inactive state in order to protect the card stack against communication breakdowns.
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	Sets the block length (in bytes) for all following block commands (read and write). Default block length is specified in the CSD.

Table continues on the next page...

Table 246. Commands for eMMC/SD card/SDIO (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command.
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_BLOCK	Continuously transfers data blocks from card to host until interrupted by a stop command.
CMD19	adtc	[31:0] reserved bits(all 0)	R1	SEND_TUNING_BLOCK	64 bytes tuning pattern is sent for SDR50 and SDR104.
CMD20	adtc	[31:0] data address	R1	WRITE_DAT_UNTIL_STOP	Writes data stream from the host, starting at the given address, until a STOP_TRANSMISION follows.
CMD21	adtc	[31:0] stuff bits	R1	SEND_TUNING_BLOCK	128 clocks of tuning pattern (64 byte in 4 bit mode or 128 byte in 8 bit mode) is sent for HS200 optimal sampling point detection.
CMD22	Reserved				
CMD23	ac	[31] reliable write request [30:16] set to 0 [15:0] number of blocks	R1	SET_BLOCK_COUNT	Defines the number of blocks (read/write) and the reliable writer parameter (write) for a block read or write command.
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command.
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until a STOP_TRANSMISSION follows.
CMD26	adtc	[31:0] stuff bits	R1	PROGRAM_CID	Programming of the card identification register. This command is issued only once per card. The card contains hardware to prevent this operation after the first programming. Normally this command is reserved for the manufacturer.
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	Programming of the programmable bits of the CSD.

Table continues on the next page...

Table 246. Commands for eMMC/SD card/SDIO (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD28	ac	[31:0] data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	ac	[31:0] data address	R1b	CLR_WRITE_PROT	If the card provides write protection features, this command clears the write protection bit of the addressed group.
CMD30	adtc	[31:0] write protect data address	R1	SEND_WRITE_PROT	If the card provides write protection features, this command asks the card to send the status of the write protection bits.
CMD31	Reserved				
CMD32	ac	[31:0] data address	R1	TAG_SECTOR_START	Sets the address of the first sector of the erase group.
CMD33	ac	[31:0] data address	R1	TAG_SECTOR_END	Sets the address of the last sector in a continuous range within the selection of a single sector to be selected for erase.
CMD34	ac	[31:0] data address	R1	UNTAG_SECTOR	Removes one previously selected sector from the erase selection.
CMD35	ac	[31:0] data address	R1	TAG_ERASE_GROUP_START	Sets the address of the first erase group within a range to be selected for erase.
CMD36	ac	[31:0] data address	R1	TAG_ERASE_GROUP_END	Sets the address of the last erase group within a continuous range to be selected for erase.
CMD37	ac	[31:0] data address	R1	UNTAG_ERASE_GROUP	Removes one previously selected erase group from the erase selection.
CMD38	ac	[31:0] stuff bits	R1b	ERASE	Erase all previously selected sectors.
CMD39	ac	[31:0] RCA [15] register write flag	R4	FAST_IO	Used to write and read 8-bit (register) data fields. The command addresses a card, and a register,

Table continues on the next page...

Table 246. Commands for eMMC/SD card/SDIO (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
		[14:8] register address [7:0] register data			and provides the data for writing if the write flag is set. The R4 response contains data read from the address register. This command accesses application dependent registers which are not defined in the eMMC standard.
CMD40	bcr	[31:0] stuff bits	R5	GO_IRQ_STATE	Sets the system into interrupt mode.
CMD41	Reserved				
CDM42	adtc	[31:0] stuff bits	R1b	LOCK_UNLOCK	Used to set/reset the password or lock/unlock the card. The size of the data block is set by the SET_BLOCK_LEN command.
CMD43~51	Reserved				
CMD52	ac	[31:0] stuff bits	R5	IO_RW_DIRECT	Access a single register within the total 128k of register space in any I/O function.
CMD53	ac	[31:0] stuff bits	R5	IO_RW_EXTENDED	Accesses a multiple I/O register with a single command. Allows the reading or writing of a large number of I/O registers.
CMD54	Reserved				
CMD55	ac	[31:16] RCA [15:0] stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application specific command rather than a standard command.
CMD56	adtc	[31:1] stuff bits [0]: RD/WR	R1b	GEN_CMD	Used either to transfer a data block to the card or to get a data block from the card for general purpose / application specific commands. The size of the data block is set by the SET_BLOCK_LEN command.
CMD57-59	Reserved				
CMD60	adtc	[31] WR [30:24] stuff bits [23:16] address	R1b	RW_MULTIPLE_REGISTER	These registers are used to control the behavior of the device and to retrieve status information regarding the operation of the

Table continues on the next page...

Table 246. Commands for eMMC/SD card/SDIO (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
		[15:8] stuff bits [7:0] byte count			device. All Status and Control registers are WORD (32-bit) in size and are WORD aligned. CMD60 is used to read and write these registers.
CMD61	adtc	[31] WR [30:16] stuff bits [15:0] data unit count	R1b	RW_MULTIPLE_BLOCK	The host issues a RW_MULTIPLE_BLOCK (CMD61) to begin the data transfer.
CMD62-63	Reserved				
ACMD6[4]	ac	[31:2] stuff bits [1:0] bus width	R1	SET_BUS_WIDTH	Defines the data bus width ('00'=1bit or '10'=4bit bus) to be used for data transfer. The allowed data bus widths are given in SCR register.
ACMD13[4]	adtc	[31:0] stuff bits	R1	SD_STATUS	Send the SD Memory Card status.
ACMD22[4]	adtc	[31:0] stuff bits	R1	SEND_NUM_WR_SECTORS	Send the number of the written sectors (without errors). Responds with 32-bit plus the CRC data block.
ACMD23[4]	ac	[31:23] stuff bits [22:0] Number of blocks	R1	SET_WR_BLK_ERASE_COUNT	Set the number of write blocks to be pre-erased before writing (to be used for fast Multiple Block WR command). "1"=default(one write block).
ACMD41[4]	bcr	[31:0] OCR	R3	SD_APP_OP_COND	Asks the accessed card to send its operating condition register (OCR) contents in the response on the CMD line.
ACMD42[4]	ac	[31:1] stuff bits [0] set_cd	R1	SET_CLR_CARD_DETECT	Connect(1)/Disconnect(0) the 50KOhm pull-up resistor on CD_B/DATA3 of the card.
ACMD51[4]	adtc	[31:0] stuff bits	R1	SEND_SCR	Reads the SD Configuration Register (SCR).

1. CMD3 differs for eMMC and SD cards. For eMMC, it is referred to as SET_RELATIVE_ADDR, with a response type of R1. For SD cards, it is referred to as SEND_RELATIVE_ADDR, with a response type of R6 (with RCA inside).
2. CMD6 differs completely between high speed eMMC and high-speed SD cards. Command SWITCH_FUNC is for high-speed SD cards.
3. Command SWITCH is for high speed eMMC. The Index field can contain any value from 0-255, but only values 0-191 are valid. If the Index value is in the 192-255 range the card does not perform any modification and the SWITCH_ERROR status bit in the EXT_CSD register is set. The Access Bits are shown in Table 247.

4. ACMDs is preceded with the APP_CMD command. Commands listed are used for SD only, other SD commands not listed are not supported on this module.

The access bits for the EXT_CSD access modes are listed in the following table.

Table 247. EXT_CSD access modes

Bits	Access name	Operation
00	Command set	The command set is changed according to the Cmd Set field of the argument.
01	Set bits	The bits in the pointed byte are set, according to the bits set to 1 in the Value field.
10	Clear bits	The bits in the pointed byte are cleared, according to the bits set to 1 in the Value field.
11	Write byte	The Value field is written into the pointed byte.

33.3.9 SDIO interrupt

Information on interrupts in 1-bit mode, interrupts in 4-bit mode, and card interrupt handling are detailed in the sections below.

33.3.9.1 Interrupts in 1-bit mode

In this case, the DATA1 pin provides the interrupt function. An interrupt is asserted by pulling the DATA1 low from SDIO, until the interrupt service is finished to clear the interrupt.

33.3.9.2 Interrupt in 4-bit mode

As the interrupt and data line 1 share pin 8 in a 4-bit mode, an interrupt is only sent by the card and recognized by the host during a specific time. This is known as the interrupt period. The uSDHC module only provides samples the level on pin 8 during the interrupt period. At all other times, the host treats it as the data signal. The definition of the interrupt period is different for operations with single block and multiple block data transfers.

In the case of normal single data block transmissions, the interrupt period becomes active two clock cycles after the completion of a data packet. This interrupt period lasts until after the card receives the end bit of the next command that has a data block transfer associated with it.

For multiple block data transfers in a 4-bit mode, there is only a limited period of time that the interrupt period can be active because of the limited period of data line availability between the multiple blocks of data. This requires stricter definition of the interrupt period. For this case, the interrupt period is limited to two clock cycles. This begins two clocks after the end bit of the previous data block. During this 2-clock cycle interrupt period, if an interrupt is pending, the DATA1 line holds low for one clock cycle, then pulls high for the next clock cycle . On completion of the interrupt period, the card releases the DATA1 line into the high Z state. The uSDHC module provides sample of the DATA1 during the interrupt period when the IABG bit in the Protocol Control register is set.

Refer to SDIO Specification v1.10 for further information about the SDIO interrupt.

33.3.9.3 Card interrupt handling

When the CINTIEN bit in the Interrupt Signal Enable Register is set to 0, uSDHC clears the interrupt request to the host system. The host driver should clear this bit before servicing the SDIO interrupt, and should set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.

The SDIO Interrupt Status can be cleared by writing 1 to this bit. But as the interrupt source from SDIO does not clear, this bit is set again. To clear this bit, it is required to reset the interrupt source from the external card followed by writing 1 to this bit. In a 1-bit mode, uSDHC detects the SDIO interrupt with or without the SD clock (to support wakeup). In a 4-bit mode, the interrupt signal is sampled during the interrupt period, so there are some sample delays between the interrupt signal from SDIO and the interrupt to the Host System Interrupt Controller. When the SDIO status is set and the host driver needs to service this interrupt, the SDIO bit in the Interrupt Control Register of SDIO is cleared. This is required to clear the SDIO interrupt status latched in uSDHC and to stop driving the interrupt signal to the System Interrupt Controller. The host driver must issue a CMD52 to clear the card interrupt. After

completion of the card interrupt service, the SDIO Interrupt Status Enable bit is set to 1 and uSDHC starts sampling the interrupt signal again.

See Figure 254 for an illustration of the SDIO interrupt scheme and for the sequences of software and hardware events that take place during a card interrupt handling procedure.

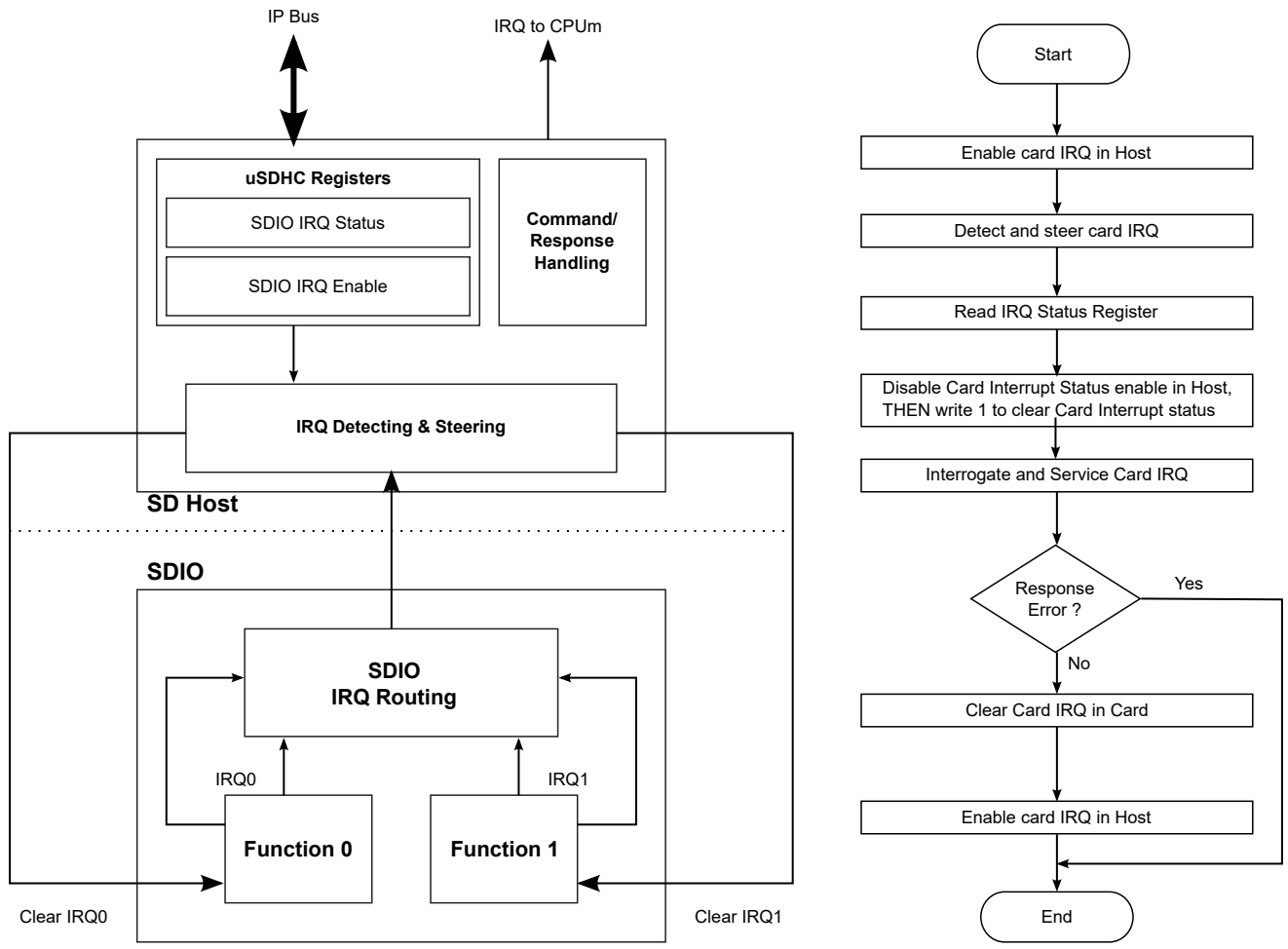


Figure 254. Card interrupt scheme, card interrupt detection, and handling procedure

33.3.10 Speed mode selection

To switch to HS200 or HS400 from High Speed (26 MHz-52 MHz), see the procedure below.

NOTE

While the actual timing change is done, the behavior of any command given (such as CMD13) cannot be guaranteed due to the asynchronous operation. Therefore, you must not use CMD13 to check the busy completion of the timing change indication. In case, CMD13 is used, the host must ignore CRC errors, if appear.

The max frequency for HS200 mode and HS400 mode could be slower than 200 MHz based on the chip implementation, you must set the target frequency for these two speed modes accordingly. Following sections describe the mode selections for HS200 mode and HS400 mode in detail.

33.3.10.1 HS200 mode selection

To switch to HS200, the Driver must perform the following steps:

1. Select the chip (through sending CMD7) and make sure it is unlocked (through CMD42).
2. Read the DEVICE_TYPE [196] field of the Extended CSD register to validate if the chip supports HS200 at the IO voltage appropriate for both host and chip (through CMD8).
3. Read the DRIVER_STRENGTH [197] field of the Extended CSD register to find the supported chip Driver Strengths (through CMD8). The default setting is 50 Ohm. The Host Designer might simulate its specific system, using a device driver models. Host can select the optimal Driver Type that might drive the host system load at the required operating frequency with the minimal noise generated.

NOTE

This step can be skipped if changes of Driver strength is not needed.

4. Check the setting of PROT_CTRL[DTW], make sure the data transfer width is set as expected.

   Make sure the I/O pad voltage is 1.8V and corresponding drive strength is set to proper value.

   Set HS200 bit and Driver Strength value in the HS_TIMING [185] field of the Extended CSD register by issuing CMD6. If the host attempts to write an invalid value, the HS_TIMING byte is not changed, the HS200 interface timing is not enabled, the Driver Strength is not changed, and the SWITCH_ERROR field is set. After the chip responds with R1, it might assert Busy signal. Once the busy signal gets de-asserted, the host may send a SEND_STATUS Command (CMD13) using the HS200 timing and after it receives a Trans State indication and No Error it means that the chip is set to HS200 timing and the Driver Strength is set to the selected settings.
5. At this point, the host can set the frequency to ≤ 200 MHz.
6. The host might invoke the HS200 standard tuning procedure, by sending CMD21 to the chip.

NOTE

The host must switch to the required bus width before starting the tuning operation to allow the tuning sequence to be done using the proper bus operating conditions.

33.3.10.2 Standard tuning procedure

By default, lower frequency operation, a fixed sampling clock is used to receive signals on CMD and DAT[3:0]. Before using the HS200, HS400, SDR104, or SDR50 modes, the Host Driver executes the tuning procedure at the mode switch sequence.

1. Issue uSDHC SW reset, set SYS_CTRL[RSTT] to 1.
2. Clear VEND_SPEC[FRC_SDCLK_ON] to 0.
3. Clean the history status by following these steps:
   a. Clear AUTOCMD12_ERR_STATUS[EXECUTE_TUNING] to 0.
   b. Read Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS) until AUTOCMD12_ERR_STATUS[EXECUTE_TUNING] reads 0.
   c. Write 1 to clear INT_STATUS[BRR].
4. Set TUNING_CTRL to 0x01312880 to enable the standard tuning and disable the CMD line check.
5. Start the tuning procedure by setting AUTOCMD12_ERR_STATUS[EXECUTE_TUNING] to 1.
6. Issue CMD19(SD) / CMD21(eMMC) with the proper Command Transfer Type (CMD_XFR_TYP) and Mixer Control (MIX_CTRL) settings.
7. Wait for uSDHC BRR (Buffer Read Ready) interrupt signal is 1.
8. Check AUTOCMD12_ERR_STATUS[EXECUTE_TUNING]. If AUTOCMD12_ERR_STATUS[EXECUTE_TUNING] = 1, write 1 to clear INT_STATUS[BRR] and then repeat steps 6~7. If AUTOCMD12_ERR_STATUS[EXECUTE_TUNING] = 0, standard tuning has completed, or the tuning has not completed within 40 attempts. The Host Driver might abort this loop if the number of loops exceeds 40 or 150 ms timeout occurs. In this case, a fixed sampling clock should be used (AUTOCMD12_ERR_STATUS[SMP_CLK_SEL] = 0).

9. Sampling Clock Select (AUTOCMD12_ERR_STATUS[SMP_CLK_SEL]) is valid after AUTOCMD12_ERR_STATUS[EXECUTE_TUNING] has changed from 1 to 0.
   AUTOCMD12_ERR_STATUS[SMP_CLK_SEL] = 1, indicates tuning procedure passed.
   AUTOCMD12_ERR_STATUS[SMP_CLK_SEL] = 0, indicates tuning procedure failed. The tuning result is applied to the delay chain, CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS) [30:16], upon the successful tuning procedure completion.
10. Set MIX_CTRL[AUTO_TUNE_EN] to 1.

While the tuning sequence is being performed, the Host Controller does not generate interrupts (including Command Complete), except Buffer Read Ready. CMD19 response errors are not indicated.

Writing AUTOCMD12_ERR_STATUS[SMP_CLK_SEL] to 0 forces the Host Controller to use a fixed sampling clock and resets the tuning circuit of the Host Controller.

NOTE

- There could be slight difference on delay cell delay in each project, and this difference can lead to different loop number needed. TUNING_CTRL[TUNING_STEP] can be used to control how may taps are to be added for each step.
- Manual tuning might be required in cases where standard tuning is not able to close passing window with CMD19/21 transfer fail.

33.3.10.3 Manual tuning procedure

Manual tuning controls more details in the tuning procedure, and some corner cases. Manual tuning provides control to tuning start point, tuning steps, and delay chains. To start manual tuning:

1. Clear TUNING_CTRL[STD_TUNING_EN] to 0.
2. Clear VEND_SPEC[FRC_SDCLK_ON] and MIX_CTRL[FBCLK_SEL] to 0, set MIX_CTRL[SMP_CLK_SEL] and MIX_CTRL[EXE_TUNE] to 1 to start tuning.
3. Set SYS_CTRL[RSTA] to 1, and wait for self-clear.
4. Config delay chain, CLK_TUNE_CTRL_STATUS[DLY_CELL_SET_PRE] with the valueA.

NOTE

The valueA must start from 0.

5. Set SYS_CTRL[RST_FIFO] to 1 and wait for self-clear.
6. Send CMD19/21 to card.
7. Poll for CC, CIE, CEBE, CCE and CTOE of Interrupt Status (INT_STATUS) assertion.
8. Poll for TC, DEBE, DCE and DTOE of Interrupt Status (INT_STATUS) assertion.
9. Check TP of Interrupt Status (INT_STATUS); TP = 1 & no command/data error indicates the current CMD19/21 transfer is passed; otherwise, indicates failed.
10. If TP = 0 or command/data error is observed, increase the valueA with the required tuning step(1 to 127) and repeat steps 3~9.
11. If TP = 1 & no command/data error is met after a loop with TP=0 (or command/data error), save the current valueA as value_start. Increase valueA with the required tuning step, and repeat steps 3~9 until TP = 0 (or command/data error) again. Save the current valueA as value_end.
12. Clear MIX_CTRL[EXE_TUNE] to 0.
13. Set SYS_CTRL[RSTA] to 1, and wait for self-clear.
14. Set MIX_CTRL[SMP_CLK_SEL] to 1.

15. Set bit[14:0] of CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS) with the output of expression {[(((value_start+value_end)/2)< < 8)& 0xffffff00] -0x300 } | 0x33.
16. Poll bit[30:16] of CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS) until the value written in the previous step is read.
17. Set MIX_CTRL[AUTO_TUNE_EN] to 1.

The manual tuning procedure defines a passing window, inside which the tuned clock can capture the correct data, and delay chain is to be configured as centered within the passing window.

It is not a fixed flow as the standard tuning. Any step can be changed based on the real case. In cases where passing window is not closed with a tuning fail, the center of the passing region must be applied for the delay chain setting. In cases where two passing windows separated by tuning fail(s) is observed, the center of the first passing window must be applied for the delay chain setting.

NOTE

Both the standard tuning and manual tuning are only for tuning of SD_CLK, cannot be used to tune STROBE clock of HS400 and HS400 enhanced mode.

33.3.10.4 Switch to HS400 mode

The valid IO voltage for HS400 is 1.8 V. The bus width is set to only DDR 8bit in HS400 mode. HS400 supports the same commands as DDR52. To switch to HS400 mode, the host must perform the following steps:

1. Initialize chip with “Backward Compatible Timings", after power-on or software reset (CMD0), the interface timing of the chip is set as the default "Backward Compatible Timing".
2. Select the chip with CMD7.
3. Read the DEVICE_TYPE [196] field of the Extended CSD register to validate whether the chip supports HS400 by sending CMD8.
4. Read the DRIVER_STRENGTH [197] field of the Extended CSD register to find the supported chip's Driver Strengths by sending CMD8.

NOTE

You may skip this step if changes of driver strength is not needed.

5. Set the "Selected Driver Strength" parameter in the HS_TIMING [185] field of the Extended CSD register to the appropriate driver strength for HS400 operation and set the "Timing Interface" parameter to 0x2 to switch to HS200 mode with CMD6. Send CMD13 to make sure mode switch to HS200 mode has been successfully done.
6. Perform the tuning procedure (standard or manual) at the HS400 target operating frequency, save the tuning result (68h [30:16]) and clear std_tuning_en.

NOTE

Tuning procedure in HS200 mode is required to synchronize the command response on the CMD line to CLK for HS400 operation.

7. Set the "Timing Interface" parameter in the HS_TIMING [185] field of the Extended CSD register to 0x1 to switch to High Speed mode with CMD6 and then set the clock frequency to a value not greater than 52 MHz.
8. Set BUS_WIDTH[183] to 0x06 to select the dual data rate x8 bus mode. Check the setting of PROT_CTRL[DTW]. Make sure the data transfer width is set as expected.
9. Set the "Timing Interface" parameter in the HS_TIMING [185] field of the Extended CSD register to 0x3 to switch to HS400 mode with CMD6. Set the MIX_CTRL[DDR_EN] and MIX_CTRL[HS400_MODE] to 1 to enable HS400 mode in controller.
10. Set bit[14:0] of CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS) with saved tuning result value of 0x68 from standard tuning flow in step 6).

11. Poll bit[30:16] of CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS) until the value written in the step 10 is read.
12. Set MIX_CTRL[SMP_CLK_SEL] to 1'b1 to apply the delay on capture clock of CMD channel, then set the clock frequency to ≤ 200 MHz.

    Ensure that uSDHC function clock source is 2x HS400 mode target frequency, and both SDCLKFS and DVS are configured to 0.
13. Set Strobe DLL control (STROBE_DLL_CTRL) to 0x00400039 to have strobe clock automatically tuned to 1/4 cycle of input ipp_strobe to capture data. Ensure that both STROBE_DLL_STATUS[STROBE_DLL_STS_REF_LOCK] and STROBE_DLL_STATUS[STROBE_DLL_STS_SLV_LOCK] are set before any transfer is started.

NOTE

For speed mode switch like from High Speed mode to HS200 mode or HS400(enhanced) mode, you must calibrate the right configuration of slew rate and drive strength for chip pad that are used for uSDHC.

33.3.10.5 Switch to HS400 enhanced mode

This selection flow describes how to initialize the eMMC device in HS400 mode while enabling enhanced strobe without the need for tuning procedure. To switch to HS400 mode with enhanced strobe, the host must perform the following steps:

1. Initialize device with "Backward Compatible Timings", after power-on or software reset (CMD0), the interface timing of the device is set as the default "Backward Compatible Timing".
2. Select the device with CMD7
3. Read the DEVICE_TYPE [196] field of the Extended CSD register to validate whether the device supports HS400 with CMD8.
4. Read the STROBE_SUPPORT[184] field of the Extended CSD register to validate whether the device supports enhanced strobe with CMD8.
5. Set the "Timing Interface" parameter in the HS_TIMING [185] field of the Extended CSD register to 0x1 to switch to High Speed mode by sending CMD6 and then set the clock frequency to a value not greater than 52 MHz.
6. Set BUS_WIDTH[183] to 0x86 to select the dual data rate x8 bus mode and enable enhanced strobe (this is active only after HS400 mode is selected) with CMD6. Check the setting of PROT_CTRL[DTW]. Make sure the data transfer width is set as expected.
7. Read the DRIVER_STRENGTH [197] field of the Extended CSD register to find the supported device Driver Strengths with CMD8.

NOTE

You may skip this step if changes of driver strength is not needed, if needed host may change the device Driver Strength.

8. Set the "Timing Interface" parameter in the HS_TIMING [185] field of the Extended CSD register to 0x3 to switch to HS400 mode with CMD6.
9. Set the MIX_CTRL[DDR_EN], MIX_CTRL[HS400_MODE], and MIX_CTRL[EN_HS400_MODE] bits to 1 to enable enhanced HS400 mode in controller.
10. Host may set the clock frequency to a value not greater than 200 MHz. Ensure uSDHC function clock source is 2x HS400 mode target frequency, and both SDCLKFS and DVS are configured to 0 .
11. Set Strobe DLL control (STROBE_DLL_CTRL) to 0x00400039 to have strobe clock automatically delayed 1/4 cycle of input ipp_strobe to capture data. Make sure both STROBE_DLL_STATUS[STROBE_DLL_STS_REF_LOCK] and STROBE_DLL_STATUS[STROBE_DLL_STS_SLV_LOCK] are set before any transfer is started.

NOTE

For speed mode switch like from high speed mode to HS200 mode or HS400(enhanced) mode, may must calibrate the right configuration of slew rate and drive strength for chip pad that are used for uSDHC.

33.3.11 Software restrictions

33.3.11.1 Initialization active

The driver cannot set INITA bit in System Control register when any of the command line or data lines are active, so the driver must ensure both CDIHB and CIHB bits are cleared.

33.3.11.2 Software polling procedure

For polling read or write, after the software begins a buffer read or write, it must access exactly the number of times as the values set in the Watermark Level Register; moreover, if the block size is not a multiple of the value in the Watermark Level Register (read and write respectively), the software must access exactly the remaining number of words at the end of each block. For example, for a read operation, if the RD_WML is 4, indicating the watermark level is 16 bytes, block size is 40 bytes, and the block number is 2, then the access times for the burst sequence in the whole transfer process must be 4, 4, 2, 4, 4, 2.

33.3.11.3 Suspend operation

To suspend the data transfer, the software must inform uSDHC that the suspend command is successfully accepted. To achieve this, after the Suspend command is accepted by SDIO, software must send another normal command marked as suspend command (CMDTYP bits set as '01') to inform uSDHC that the transfer is suspended.

If software needs to resume the suspended transfer, it should read the value in BLKCNT register to save the remaining number of blocks before sending the normal command marked as suspend, otherwise on sending such a 'suspend' command, uSDHC treats the current transfer is aborted and change the BLKCNT register to its original value, instead of retaining the remaining number of blocks.

33.3.11.4 Data length setting

For either ADMA (ADMA1 or ADMA2) transfer, the data in the data buffer must be word aligned, so the data length set in the descriptor must be a multiple of 4.

ADMA1 is 4KB aligned.

33.3.11.5 (A)DMA address setting

To configure the ADMA1/ADMA2/DMA address register, when TC bit is set, the register always updates itself with the internal address value to support dynamic address synchronization, so the software must ensure that the TC bit is cleared prior to configuring the ADMA1/ADMA2/DMA address register.

33.3.11.6 Data port access

Data port does not support parallel access. For example, during an internal DMA access, it is not allowed to write any data to the data port by CPU; or during a CPU read operation, it is also prohibited to write any data to the data port, by either CPU or internal DMA. Otherwise the data is corrupted inside the uSDHC buffer.

33.3.11.7 Change clock frequency

The uSDHC module does not automatically gate off the card clock when the host driver changes the clock frequency. To prevent possible glitch on the card clock, clear the FRC_SDCLK_ON bit when changing the clock divisor value (SDCLKFS or DVS in System Control Register) or setting the RSTA bit.

Also, before changing the clock divisor value, the host driver should make sure that the SDSTB bit is high.

33.3.11.8 Multi-block read

For pre-defined multi-block read operation, that is, the number of blocks to read has been defined by previous CMD23 for eMMC, or pre-defined number of blocks in CMD53 for SDIO/SDCombo, or whatever multi-block read without abort command at card side, an abort command, either automatic or manual CMD12/CMD52, is still required by uSDHC after the pre-defined number of blocks are done, to drive the internal state machine to idle mode. In this case, the card may not respond to this extra abort command and uSDHC gets response timeout. It is recommended to manually send an abort command with RSPTYP[1:0] both bits cleared.

33.3.12 Clocking

The table found here describes the clock sources for uSDHC. For more information on clocking, see the Clocking chapter.

Table 248. Clocks

Clock name	Description
hclk	Bus clock
ipg_clk	Peripheral clock
ipg_clk_perclk	Base clock
ipg_clk_lp	Low power clock

33.3.12.1 Clock and reset manager

This module controls all four kinds reset signals within uSDHC:

- Hardware reset
- Software reset for all logic
- Software reset for the data logic
- Software reset for the command logic

All these signals are fed into this module and stable signals are generated inside the module to reset all other modules. The module also gates off all the inside signals.

33.3.12.2 Clock generator

The clock generator generates the card CLK by peripheral source clock in two stages. The clock divisor can be configured through register SYS_CTRL[SDCLKFS] for prescaler configuration while the [DVS] for divisor configuration. Details can be found in the register function description. See the following figure for the structure of the divider. The term "Base" represents the frequency of peripheral source clock (see ipg_clk_perclk in Table 248).

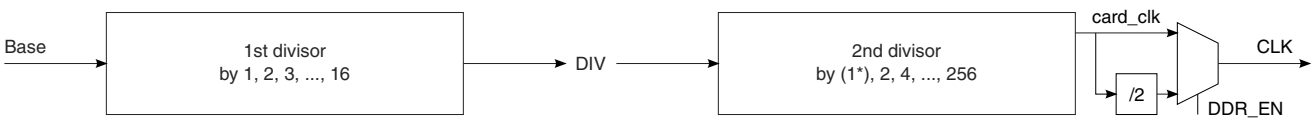


Figure 255. Two stages of the clock divider

The first stage outputs an intermediate clock (DIV) that can be Base, Base/2, Base/3, ..., or Base/16.

The second stage is a prescaler and outputs the actual internal working clock (card_clk). This clock is the driving clock for all the sub modules of the SD protocol unit, and helps in syncing FIFOs (see Figure 241) to synchronize with the data rate from the internal data buffer. The frequency of the clock output from this stage can be DIV, DIV/2, DIV/4,..., or DIV/256. Therefore, the highest frequency of the card_clk is base, and the next highest is Base/2, while the lowest frequency is Base/4096. If the duty cycle of the base clock is 50%, the duty cycle of card_clk is also 50%, even when the compound divisor is an odd value.

NOTE

CLK is different for the SDR and DDR modes.

- In the SDR mode, CLK is equal to the internal working clock (card_clk).

$$f_{card\_clk} = \frac{f_{Base}}{SYS\_CTRL[DVS] \times SYS\_CTRL[SDCLKFS]}$$

$$f_{CLK} = f_{card\_clk}$$

Equation 2. Equation for f_{card_clk} in SDR mode

For HS200 mode, if the source clock f_{Base} = 400 MHz, and the target clock f_{CLK} = 200 MHz, then configuring DVS to 'h00 and SDCLKFS to 'h01, the mapping value of SYS_CTRL[DVS] is 1 and the mapping value of SYS_CTRL[SDCLKFS] is 2.

$$f_{card\_clk} = \frac{400MHz}{1x(2)} = 200MHz$$

$$f_{CLK} = f_{card\_clk} = 200\ MHz$$

- In the DDR mode, CLK is equal to card_clk/2

$$f_{card\_clk} = \frac{f_{Base}}{SYS\_CTRL[DVS] \times (SYS\_CTRL[SDCLKFS]/2)}$$

$$f_{CLK} = \frac{f_{card\_clk}}{2}$$

Equation 3. Equation for f_{card_clk} in DDR mode

For HS400 mode, if the source clock f_{Base} = 400 MHz, and the target clock f_{CLK} = 200 MHz, then configuring DVS to 'h00 and SDCLKFS to 'h00, the mapping value of SYS_CTRL[DVS] is 1 and the mapping value of SYS_CTRL[SDCLKFS] is 2.

$$f_{card\_clk} = \frac{400MHz}{1x(2/2)} = 400MHz$$

$$f_{CLK} = \frac{f_{card\_clk}}{2} = \frac{400MHz}{2} = 200MHz$$

33.3.13 Command Queuing (CQE) operation

To facilitate command queuing in eMMC, the device manages an internal task queue that the host can queue data transfer tasks. Initially, the task queue is empty. Every task is issued by the host and initially queued as pending. The device controller works to prepare the pending tasks for execution. The software flow is described below:

1. Read CMDQ_SUPPORT[308] and CMDQ_DEPTH[307] of the extended CSD register (through CMD8) to verify eMMC chip support of command queue.
2. Set CMDQ_MODE_EN in CMD_MODE_EN[15] field of the extended CSD by issuing CMD6.
3. Send CMD13 to ensure a successful mode switch.
4. Define Task Descriptor List (TDL) comprised of task descriptor and transfer descriptor sets.
    a. 0x0010002b,0x00000000,0x20000023,0x34300000, //Write example
    b. 0x0010102b,0x00000000,0x20000023,0x34500000, //Read example
5. Set PROT_CTRL[DMASEL] to 2 to select ADMA2. CMDQ descriptors are in ADMA2 format.

6. Set CQSSC2[SSC2] to RCA.
7. Set Command Queuing Task Descriptor List Base Address (CQTDLBA) and Command Queuing Task Descriptor List Base Address Upper 32 Bits (CQTDLBAU) to TDL start address.
8. Set CQCFG[CQUE] to 1 to enable the command queuing.
9. Set CQTDBR[TDBR] field(s) to trigger the required task(s).
10. Read CQTCN[TCN] field(s) to confirm the task completion(s).
11. Set CQTCN[TCN] field(s) to 1 to clear status.

33.4 External signals

The following table describes the uSDHC external signals:

Table 249. uSDHC external signals

Signal	Description	Direction
CLK	Is an Internally generated clock used to drive the eMMC, SD card, and SDIO.	O
CMD	Is used to send commands and receive responses to and from the card.	I/O
DAT7	DAT7 line in the 8-bit mode — Not used in other modes	I/O
DAT6	DAT6 line in the 8-bit mode — Not used in other modes	I/O
DAT5	DAT5 line in the 8-bit mode — Not used in other modes	I/O
DAT4	DAT4 line in the 8-bit mode — Not used in other modes	I/O
NOTE If uSDHC needs to support a 4-bit data transfer, DAT7~DAT4 can also be optional and tied to high.		
DAT3	DAT3 line in the 4/8-bit mode or configured as card detection pin. The bit may be configured as card detection pin in the 1-bit mode.	I/O
DAT2	DAT2 line or Read Wait in the 4-bit mode Read Wait in 1-bit mode	I/O
DAT1	DAT1 line in the 4/8-bit mode Also, used to detect interrupt in 1/4-bit mode	I/O
DAT0	DAT0 line in all the modes Also, used to detect busy state	I/O

Table continues on the next page...

Table 249. uSDHC external signals (continued)

Signal	Description	Direction
CD_B	Card detection signal directly routed from the socket. Low value (0) indicates there is a card inserted. In the case the pin is not used (for the embedded memory), tie low. Optional for system implementation.	I
WP	Write protection signals directly routed from the socket. Low value (0) indicates it is not write protected. In the case the pin is not used (for the embedded memory), tie low. Optional for system implementation.	I
RESET_B	Is used to reset the eMMC.	O
VSELECT	Is used to change the voltage of the external power supplier. Optional for system implementation.	O
STROBE	Input clock for eMMC HS400 mode. **NOTE** If the uSDHC does not support the HS400 mode, STROBE can also be optional and tied to low.	I

33.5 Application information

All communication between the system and cards are controlled by the host. The host sends commands of two types: broadcast and addressed (point-to-point).

Broadcast commands are intended for all cards, such as GO_IDLE_STATE, SEND_OP_COND, and ALL_SEND_CID. In the Broadcast mode, eMMC are in the open-drain mode to avoid bus contention. See Commands for SD card, SDIO, and eMMC for the commands of bc and bcr categories.

After the broadcast command CMD3 is issued, the cards enter the standby mode. Addressed type commands are used from this point. In this mode, for eMMC, the CMD/DATA I/O pads turns to the push-pull mode to have the driving capability for maximum frequency operation. See Commands for SD card, SDIO, and eMMC for the commands of ac and adtc categories.

33.5.1 Command send and response receive basic operation

Assuming that the data type WORD is an unsigned 32-bit integer, the flow indicated below presents a guideline for sending a command to the card(s):

```
send_command(cmd_index, cmd_arg, other requirements)

{

    WORD wCmd; // 32-bit integer to make up the data to write into Transfer Type register, it is
recommended to implement in a bit-field manner

    wCmd = (<cmd_index> & 0x3f) << 24; // set the first 8 bits as '00'+<cmd_index>

    set CMDTYP, DPSEL, CICEN, CCCEN, RSTTYP, DTDSEL according to the command index;

    if (internal DMA is used) wCmd |= 0x1;
```

```
    if (multi-block transfer) {

        set MSBSEL bit;

        if (finite block number) {

            set BCEN bit;

            if (auto12 command is to use) set AC12EN bit;

        }

    }

    write_reg(CMDARG, <cmd_arg>); // configure the command argument

    write_reg(XFERTYP, wCmd); // set Transfer Type register as wCmd value to issue the command

}

wait_for_response(cmd_index)

{

    while (CC bit in IRQ Status register is not set); // wait until Command Complete bit is set

    read IRQ Status register and check if any error bits about Command are set

    if (any error bits are set) report error;

    write 1 to clear CC bit and all Command Error bits;

}
```

For the sake of simplicity, the function wait_for_response is implemented here by means of polling. For an effective and formal way, the response is usually checked after the Command Complete Interrupt is received. When doing this, make sure that the corresponding interrupt status bits are enabled.

For some scenarios, the response time-out is expected. For instance, after all cards respond to CMD3 and move to the Standby state no response to the Host when CMD2 is sent. The host driver deals with "fake" errors like this with caution.

33.5.2 Card identification mode

When a card is inserted to the socket or the card is reset by the host, the host needs to validate the operation voltage range, identify the cards, request the cards to publish the Relative Card Address (RCA) or to set the RCA for eMMC.

33.5.2.1 Card detect

See Figure 256 for a flow diagram showing the detection of SD card and SDIO using uSDHC.

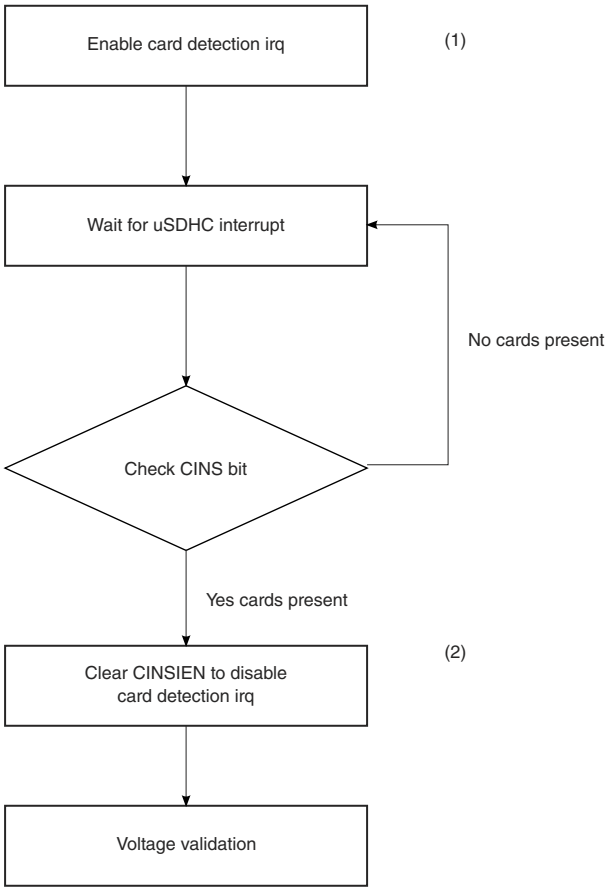


Figure 256. Flow diagram for card detection

Here is the card detect sequence:

- Set the CINSIEN bit to enable card detection interrupt.
- When an interrupt from uSDHC is received, check the CINS bit in the Interrupt Status register to see if it was caused by card insertion.
- Clear the CINSIEN bit to disable the card detection interrupt and ignore all card insertion interrupts afterwards.

33.5.2.2 Reset

The host consists of three types of resets:

- Hardware reset (Card and Host) that is driven by Power On Reset (POR).
- Software reset (Host only) is initiated by the write operation on the RSTD, RSTC, or RSTA bits of the System Control register to reset the data part, command part, or all parts of the host controller, respectively.
- Card reset (Card only): The command, "Go_Idle_State" (CMD0), is the software reset command for all types of eMMC and SD memory cards. This command sets each card into the idle state regardless of the current card state. For an SDIO, CMD52 is used to write an I/O reset in CCCR. The cards are initialized with a default relative card address (RCA=0x0000) and with a default driver stage register setting (lowest speed, highest driving current capability).

After the card is reset, the host needs to validate the voltage range of the card. See the figure below for the software flow to reset both uSDHC and the card.

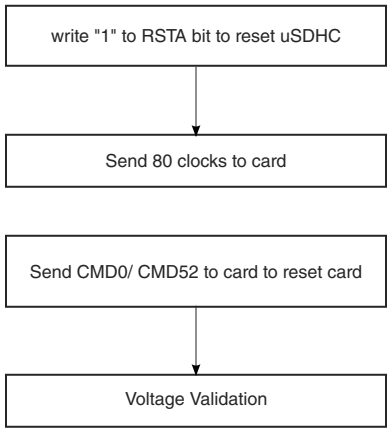


Figure 257. Flow chart for resetting uSDHC and SD card or SDIO

```
software_reset() {
    set_bit(SYSCTRL, RSTA); // software reset the Host set DTOCV and SDCLKFS fields to get the CLK of
    //frequency around 400kHz
    //configure IO pad to set the power voltage of external card to around 3.0V
    //poll bits CIHB and CDIHB
    //bits of PRSSTAT to wait both fields are cleared
    set_bit(SYSCTRL, INTIA); // send 80 clock ticks for card to power up
    send_command(CMD_GO_IDLE_STATE, <other parameter>); // reset the card with CMD0 or
    send_command(CMD_IO_RW_DIRECT, <other parameter>);
}
```

33.5.2.3 Voltage validation

All cards should be able to establish communication with the host using any operation voltage in the maximum allowed voltage range specified in the card specification. However, the supported minimum and maximum values for VDD are defined in the Operation Conditions Register (OCR) and may not cover the whole range. Cards that store the CID and CSD data in the preload memory are only able to communicate this information under data transfer VDD conditions. This means that if the host and card have non-common VDD ranges, the card is neither able to complete the identification cycle nor able to send CSD data.

Therefore, special commands Send_Op_Cont (CMD1 for eMMC), SD_Send_Op_Cont (ACMD41 for SD Memory), and IO_Send_Op_Cont (CMD5 for SD I/O) are used. The voltage validation procedure is designed to provide a mechanism to identify and reject cards that do not match the VDD range(s) desired by the host. This is accomplished when the host sends the desired VDD voltage window as the operand of this command. Cards that cannot perform the data transfer in the specified range must discard themselves from further bus operations and go into the Inactive state. By omitting the voltage range in the command, the host can query each card and determine the common voltage range before sending out-of-range cards into the Inactive state. This query should be used if the host is able to select a common voltage range or if a notification is sent to the system when a non-usable card in the stack is detected.

The following steps show how to perform voltage validation when a card is inserted:

```
voltage_validation(voltage_range_argument) {
    label the card as UNKNOWN;
    send_command(IO_SEND_OP_COND, 0x0, <other parameters are omitted>); // CMD5, check SDIO operation
voltage, command argument is zero
    if (RESP_TIMEOUT != wait_for_response(IO_SEND_OP_COND)) { // SDIO command is accepted
        if (0 < number of IO functions) {
            label the card as SDIO;
            IORDY = 0;
```

```
            while (!(IORDY in IO OCR response)) { // set voltage range for each IO function
                send_command(IO_SEND_OP_COND, <voltage range>, <other parameter>);
                wait_for_response(IO_SEND_OP_COND);
            } // end of while ...
        } // end of if (0 < ...
        if (memory part is present inside SDIO)
            Label the card as SDCombo; //this is an SD-Combo card
    } // end of if (RESP_TIMEOUT ...
    if (the card is labeled as SDIO)
        return; // card type is identified and voltage range is set, so exit the function;
    send_command(APP_CMD, 0x0, <other parameter>); // CMD55, Application specific CMD prefix
    if (no error calling wait_for_response(APP_CMD, <...>) { // CMD55 is accepted
        send_command(SD_APP_OP_COND, <voltage range>, <...>); // ACMD41, to set voltage range for
memory part or SD card
        wait_for_response(SD_APP_OP_COND); // voltage range is set
        if (card type is UNKNOWN)
           label the card as SD;
        return; //
    } // end of if (no error ...
    else if (errors other than time-out occur) { // command/response pair is corrupted
        deal with it by program specific manner;
    } // of else if (response time-out
    else { // CMD55 is refuse, it must be eMMC
        if (card is already labeled as SDCombo) { // change label
            re-label the card as SDIO;
            ignore the error or report it;
            return; // card is identified as SDIO
        } // end of if (card is ...
        send_command(SEND_OP_COND, <voltage range>, <...>);
        if(RESP_TIMEOUT == wait_for_response(SEND_OP_COND)) { // CMD1 is not accepted,
            either label the card as UNKNOWN;
            return;
        } // end of if (RESP_TIMEOUT ...
    } // end of else
}
```

33.5.2.4 Card registry

This section briefly describes the registry flow. For details, please refer to the card specifications. Card registry for the eMMC and SD/SDIO/SD combo cards are different. For the SD card, the identification process starts at a clock rate lower than 400 kHz and the power voltage higher than 2.7 V (as defined by the card spec). Currently, the CMD line output drives are push-pull drivers instead of open-drain. After the bus is activated, the host requests the card to send their valid operation conditions. The response to ACMD41 is the operation condition register of the card. The same command is sent to all the new cards in the system. Incompatible cards are put into the Inactive state. The host then issues the command, All_Send_CID (CMD2), to each card to get its unique card identification (CID) number. Cards that are currently unidentified (in the Ready state), send their CID number as the response. After the CID is sent by the card, the card goes into the Identification state.

The host then issues Send_Relative_Addr (CMD3), requesting the card to publish a new relative card address (RCA) that is shorter than the CID. This RCA is used to address the card for future data transfer operations. After the RCA is received, the card changes its state to the Standby state. At this point, if the host wants the card to have an alternative RCA number, it may ask the card to publish a new number by sending another Send_Relative_Addr command to the card. The last published RCA is the actual RCA of the card.

The host repeats the identification process with CMD2 and CMD3 for each card in the system until the last CMD2 gets no response from any of the cards in the system.

For eMMC operation, the host starts the card identification process in the open-drain mode with the identification clock rate lower than 400 kHz and the power voltage higher than 2.7 V. The open drain driver stages on the CMD line to allow parallel card operation during card identification. After the bus is activated, the host requests the cards to send their valid operation conditions

(CMD1). The response to CMD1 is the "wired AND " operation on the condition restrictions of all cards in the system. Incompatible cards are sent into the Inactive state. The host then issues the broadcast command All_Send_CID (CMD2), asking all cards for their unique card identification (CID) number. All unidentified cards (the cards in the Ready state) simultaneously start sending their CID numbers serially, while bit-wise monitoring their outgoing bit stream. Those cards, whose outgoing CID bits do not match the corresponding bits on the command line in any one of the bit periods, stop sending their CID immediately and must wait for the next identification cycle. As the CID is unique for each card, only one card can be successfully sent its full CID to the host. This card then goes into the Identification state. Thereafter, the host issues Set_Relative_Addr (CMD3) to assign a relative card address (RCA) to the card. After the RCA is received, the state of the card changes to standby, and the card does not react in further identification cycles. Also, its output driver switches from open-drain to push-pull. The host repeats the process, mainly CMD2 and CMD3, until the host receives a time-out condition to recognize the completion of the identification process.

The following steps show how to perform an operation using eMMC:

```
card_registry() {
    do { // decide RCA for each card until response time-out
        if(card is labelled as SDCombo or SDIO) { // for SDIO like device
            send_command(SET_RELATIVE_ADDR, 0x00, <...>); // ask SDIO to publish its RCA
            retrieve RCA from response;
        } // end if (card is labelled as SDCombo ...
        else if (card is labelled as SD) { // for SD card
            send_command(ALL_SEND_CID, <...>);
            if(RESP_TIMEOUT == wait_for_response(ALL_SEND_CID))
                break;
            send_command(SET_RELATIVE_ADDR, <...>);
            retrieve RCA from response;
        } // else if (card is labelled as SD ...
        else if (card is labelled as eMMC) {
            send_command(ALL_SEND_CID, <...>);
            rca = 0x1; //arbitrarily set RCA, 1 here for example
            send_command(SET_RELATIVE_ADDR, 0x1 << 16, <...>); // send RCA at upper 16 bits
        } // end of else if (card is labelled as eMMC...
    } while (response is not time-out);
}
```

33.5.3 Card access

Information about Block Write, Block Read, Suspense Resume, ADMA Usage, Transfer Error, and Card Interrupt are detailed in the sections below.

33.5.3.1 Block write

Information on Normal Write, DDR Write, and Write with Pause are detailed in the sections below.

33.5.3.1.1 Normal Write

During a block write (CMD24 - 27, CMD60, CMD61), one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block by the host. If the CRC fails, the card indicates that the failure on the DATA line. The transferred data is discarded and not written, and all further transmitted blocks (in multiple block write mode) are ignored.

If the host uses partial blocks whose accumulated length is not block aligned and block misalignment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the card detects the block misalignment error and aborts the programming before the beginning of the first misaligned block. The card sets the ADDRESS_ERROR error bit in the status register, and while ignoring all further data transfer, waits in the Receive-data-State for a stop command. The write operation is also aborted if the host tries to write over a write-protected area.

For eMMC and SD cards, programming of the CID and CSD registers does not require a previous block length setting. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in ROM, then this unchangeable part must match the corresponding part of the receive buffer. If this match fails, then the card reports an error and does not change any register contents.

For all types of cards, some may require long and unpredictable periods of time to write a block of data. After receiving a block of data and completing the CRC check, the card begins writing and holds the DATA line low if its write buffer is full and unable to accept new data from a new WRITE_BLOCK command. The host may poll the status of the card with a SEND_STATUS command (CMD13) or other means for SDIO at any time, and the card responds with its status. The responded status indicates whether the card can accept new data or whether the write process is still in progress. The host may deselect the card by issuing a CMD7 (to select a different card) to place the card into the Standby state and release the DATA line without interrupting the write operation. When re-selecting the card, it reactivates the busy indication by pulling data to low if the programming is still in progress and the write buffer is unavailable.

The software flow to write to a card that incorporates the internal DMA and the write operation is a multi-block write with the Auto CMD12 enabled. For the other methods (CPU polling status with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to write to a card is described below:

1. Check the card status, wait until the card is ready for data.
2. Set the card block length/size:
   - For eMMC and SD cards, use SET_BLOCKLEN (CMD16).
   - For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT (CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred, or another error that occurred during the auto12 command sending and response receiving.

33.5.3.1.2 DDR write

uSDHC supports the dual data rate mode.

The software flow to write to a card in the DDR mode is described as below:

1. Check the card status and wait until the card is ready for data.

   NOTE

   For eMMC, block length only can be set to 512byte.

2. Set the uSDHC number block register (NOB), where nob is 5, for instance.
3. Set the eMMC, SD card, or SDIO to high-speed mode and use SWITCH(CMD6).
4. Set the eMMC bus or SD with 4-bit/8-bit DDR mode and use SWITCH(CMD6).
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The DDR_EN and AC12EN bits should be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred or another error that occurred during the auto12 command sending and response receiving.

33.5.3.1.3 Write with Pause

The write operation can be paused during the transfer. Instead of stopping the CLK at any time to pause all the operations, which is also inaccessible to the host driver, the driver can set the Stop At Block Gap Request (SABGREQ) bit in the Protocol

Control register to pause the transfer between the data blocks. As there is no time-out condition in a write operation during the data blocks, a write to all types of cards can be paused in this way, and if the DATA0 line is not required to de-assert to release the busy state, no suspend command is needed.

Similar to the flow described in Normal Write, the write with pause is shown with the same kind of write operation:

1. Check the card status and wait until the card is ready for data.
2. Set the card block length/size:
   - For eMMC and SD cards, use SET_BLOCKLEN (CMD16).
   - For SDIO or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Set the SABGREQ bit.
7. Wait for the Transfer Complete interrupt.
8. Clear the SABGREQ bit.
9. Check the status bit to see if a write CRC error occurred.
10. Set the CREQ bit to continue the write operation.
11. Wait for the Transfer Complete interrupt.
12. Check the status bit to see if a write CRC error occurred or some another error that occurred during the auto12 command sending and response receiving.

The number of blocks left during the data transfer is accessible by reading the contents of the BLKCNT field in the Block Attribute register. As the data transfer and the setting of the SABGREQ bit are concurrent, and the delay of register read and the register setting, the actual number of blocks left may not be exactly the value read earlier. The host driver reads the value of BLKCNT after the transfer is paused and the Transfer Complete interrupt is received.

It is also possible that the last block has begun when the Stop At Block Gap Request is sent to the buffer. In this case, the next block gap is the end of the transfer. These types of requests are ignored, the driver should treat these as a non-pause transfer, and deal with it as a common write operation.

When the write operation is paused, the data transfer inside the host system is not stopped, and the transfer is active until the data buffer is full. Because of this, it is recommended to avoid using the Suspend command for SDIO. This is because when such a command is sent, uSDHC interprets the system that switches to another function on SDIO and flush the data buffer. uSDHC takes the Resume command as a normal command with data transfer, and it is left for the host driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set as well as the AC12EN bit. However, the uSDHC module automatically sends a CMD12 to mark the end of the multi-block transfer.

33.5.3.2 Block read

Information about Normal read, DDR read, Read with Pause, and Delay Line (DLL) in Read Path are detailed in the sections below.

33.5.3.2.1 Normal read

For block reads, the basic unit of data transfer is a block whose maximum size is stored in areas defined by the corresponding card specification. A CRC is appended to the end of each block, ensuring data transfer integrity. The CMD17, CMD18, CMD53, CMD60, CMD61, and so on, can initiate a block read. After completing the transfer, the card returns to the Transfer state. For multi blocks read, data blocks are continuously transferred until a stop command is issued.

The software flow to read from a card that incorporates the internal DMA and the read operation is a multi-block read with the Auto CMD12 enabled. For the other methods (CPU polling status with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to read from a card is described below:

1. Check the card status and wait until card is ready for data.
2. Set the card block length/size:
   - For eMMC and SD cards, use SET_BLOCKLEN (CMD16).
   - For SDIO or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer read ready interrupt, configure the DMA settings, and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a read CRC error occurred, or another error occurred during the auto12 command sending and response receiving.

33.5.3.2.2 DDR read

The uSDHC module supports dual data rate mode.

The software flow to write to a card in the DDR mode is described below:

1. Check the card status and wait until the card is ready for data.

NOTE

For eMMC, block length can be set to only 512 bytes.

2. Set the uSDHC number block register (NOB) where nob is 5, for instance.
3. Set the eMMC, SD card, or SDIO to high-speed mode and use SWITCH (CMD6).
4. Set the eMMC bus or SD with 4-bit /8-bit DDR mode and use SWITCH(CMD6).
5. Disable the buffer write ready interrupt, configure the DMA settings, and enable the uSDHC DMA when sending the command with data transfer. The DDR_EN and AC12EN bits should be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred, or another error that occurred during the auto12 command sending and response receiving.

33.5.3.2.3 Read with Pause

The read operation is not generally able to pause. Only SDIO (and SDCombo card working under I/O mode) supporting the Read Wait feature can pause during the read operation. If SDIO supports Read Wait (SRW bit in CCCR register is 1), the host driver can set the SABGREQ bit in the Protocol Control register to pause the transfer between the data blocks. Before setting the SABGREQ bit, ensure that the RWCTL bit in the Protocol Control register is set, otherwise uSDHC does not assert the Read Wait signal during the block gap and data corruption occurs. It is recommended to set the RWCTL bit after the Read Wait capability of SDIO is recognized.

Similar to the flow described in Normal read, the read with pause is shown with the same kind of read operation:

1. Check the SRW bit in the CCR register on SDIO to confirm that the card supports the Read Wait mode.
2. Set the RWCTL bit.

3. Check the card status and wait until the card is ready for data.
4. Set the card block length/size:
   - For eMMC and SD cards, use SET_BLOCKLEN (CMD16).
   - For SDIO or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
5. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
6. Set the uSDHC number block register (NOB), where nob is 5, for instance.
7. Disable the buffer read ready interrupt, configure the DMA setting, and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
8. Set the SABGREQ bit.
9. Wait for the Transfer Complete interrupt.
10. Clear the SABGREQ bit.
11. Check the status bit to see if read CRC error occurred.
12. Set the CREQ bit to continue the read operation.
13. Wait for the Transfer Complete interrupt.
14. Check the status bit to see if a read CRC error occurred, or another error occurred during the auto12 command sending and response receiving.

Similar to the Write operation, it is possible to meet the ending block of the transfer when paused. In this case, uSDHC ignores the Stop At Block Gap Request and treats it as a command read operation.

Unlike the write operation, there is no remaining data inside the buffer when the transfer is paused. All data received before the pause is transferred to the host system. No matter if the Suspend command is sent or not, the internal data buffer is not flushed.

If the Suspend command is sent and the transfer is later resumed by means of a Resume command, uSDHC takes the command as a normal one accompanied with data transfer. It is left for the host driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set, as well as the AC12EN bit. However, the uSDHC automatically sends CMD12 to mark the end of multi-block transfer.

33.5.3.2.4 Delay Line (DLL) in read path

The DLL is newly added to assist in sampling read data. The DLL provides the ability to programmatically select a quantized delay (in fractions of the clock period) regardless of on-chip variations such as process, voltage, and temperature (PVT).

The reasons why DLL is needed for uSDHC are these:

- The path of read data traveling from card to host varies.
- In the eMMC and SD cards DDR mode, the minimum input setup and hold time are both at 2.5 ns.

The data sampling window is so small that the delay of loopback clock needs to be accurate and consistent regardless of PVT. The DLL takes the divided card_clk as the reference clock and loopback clock as the input clock. It then generates a delayed version of the input clock according to the programmed target delay.

The DLL can be disabled or bypassed, and it can also be manually set for a fixed delay in the override mode. The override value set is the number of delay cells. In the override mode, there is no need to set the DLL_enable. Another DLL mode is target value mode. In this mode, the DLL automatically adjusts the number of delay cells according to the target value set by the user and PVT changes. Be aware that the target value is in units of 1/32 of the clock reference period. If the card_clk is 100Mhz, then the reference clock period is 10ns; setting target value of 16 means 5ns =(16/32)*10ns. The software can disable automatic update by the setting dll_gate_update bit.

As you might change the frequency of card_clk from time to time by changing SDCLKFS[7:0]/DVS[3:0], the software must adjust the delay value to ensure it works correctly when the reference clock (card_clk) is changed. If DLL is to be used, make sure SDCLKFS is used (at least be set to divided by 2) so that the REF_CLK generated are the same frequency as card_clk. There are

two DLLs, PARTS DLL and STROBE DLL. PARTS DLL controls the SD_CLK loop back delay for which DLL (Delay Line) Control (DLL_CTRL) and DLL Status (DLL_STATUS) are used. STROBE DLL controls the STROBE loop back delay for which Strobe DLL control (STROBE_DLL_CTRL) and Strobe DLL status (STROBE_DLL_STATUS) are used. For PARTS DLL, below flow can be taken as reference. For STROBE DLL, similar register field with prefix STROBE in Strobe DLL control (STROBE_DLL_CTRL) and Strobe DLL status (STROBE_DLL_STATUS) must be used.

Step 1: Set the DLL_CTRL_RESET and DLL_CTRL_ENABLE fields

Step 2: Configure the SDCLKFS[7:0] and DVS[3:0]

Step 3: Wait until SDSTB is asserted

Step 4: Clear the DLL_CTRL_RESET field

Step 5: Wait until both the DLL_STS_SLV_LOCK and DLL_STS_REF_LOCK are asserted

Step 6: Set the DLL_CTRL_SLV_FORCE_UPD

Step 7: Clear the DLL_CTRL_SLV_FORCE_UPD

NOTE

The software should make sure that the DLL_CTRL_SLV_FORCE_UPD lasts for at least one card_clk. So, the software may need to add some delay between step 6 and step 7.

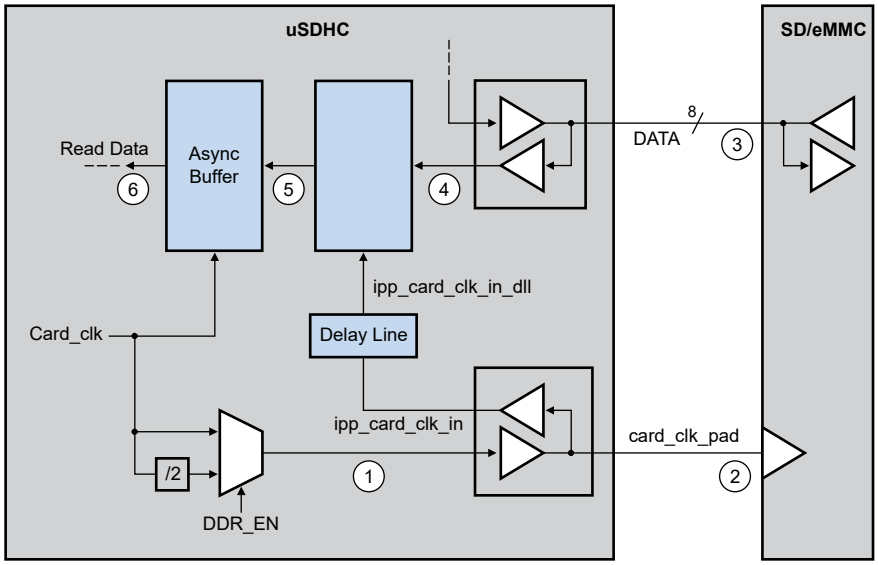


Figure 258. DLL in read path

33.5.3.3 Suspend Resume

The uSDHC module supports the Suspend Resume operations of SDIO, although slightly different than the suggested implementation of Suspend in the SDIO specification.

33.5.3.3.1 Suspend

After setting the SABGREQ bit, the host driver may send a Suspend command to switch to another function of SDIO. The uSDHC module does not monitor the content of the response, so it does not know if the Suspend command succeeded or not. Accordingly, it does not de-assert Read Wait for read pause. To solve this problem, the host driver does not mark the Suspend command as "Suspend", (that is, setting the CMDTYP bits to 01). Instead, the driver sends this command as if it were a normal command, and only when the command succeeds, and the BS bit is set in the response, can the Driver send another command marked as "Suspend" to inform uSDHC that the current transfer is suspended. Here is the sequence for the Suspend operation:

1. Set the SABREQ bit to pause the current data transfer at block gap.
2. After the BGE bit is set, send the Suspend command to suspend the active function. The CMDTYP bit field must be 2'b00.
3. Check the BS bit of the CCCR in the response. If it is 1, repeat this step until the BS bit is cleared or abandon the suspend operation according to the Driver strategy.
4. Send another normal I/O command to the suspended function. The CMDTYP of this command must be 2'b01, so uSDHC can detect this special setting and be informed that the paused operation has successfully suspended. If the paused transfer is a read operation, uSDHC stops driving DATA2 and goes to the idle state.
5. Save the context registers in the system memory for later use, including the DMA System Address Register (for internal DMA operation), and the Block Attribute Register.
6. Begin operation for another function on SDIO.

33.5.3.3.2 Resume

To resume the data transfer, a Resume command is issued:

1. To resume the suspended function, restore the context register with the saved value in step #5 of the Suspend operation.
2. Send the Resume command: In the Transfer Type register, all the bit fields are set to the value as if this were another ordinary data transfer instead of a transfer resume (except the CMDTYP is set to 2'b10).
3. If the Resume command has responded, the data transfer is resumed.

33.5.3.4 ADMA usage

To use the ADMA in a data transfer, the host driver must prepare the correct descriptor chain prior to sending the read/write command. The steps to prepare the correct descriptor chain are these:

1. Create a descriptor to set the data length that the current descriptor group is about to transfer. The data length should be even numbers of the block size.
2. Create another descriptor to transfer the data from the address setting in this descriptor. The data address must be at a page boundary (4KB address aligned).
3. If necessary, create a Link descriptor containing the address of the next descriptor. The descriptor group is created in steps 1 ~ 3.
4. Repeat steps 1 ~ 3 until all descriptors are created.
5. In the last descriptor, set the End flag to 1 and make sure the total length of all descriptors matches the product of the block size and block number configured in the Block Attribute Register.
6. Set the ADMA System Address Register to the address of the first descriptor and set the DMAS field in the Protocol Control Register to 01 to select the ADMA.
7. Issue a write or read command with the DMAEN bit set to 1 in the Transfer Type Register.

Steps 1 ~ 5 are independent of step 6, so step 6 can finish before steps 1 ~ 5. Regarding the descriptor configuration, it is recommended not to use the Link descriptor as it requires extra system memory access.

33.5.3.5 Transfer error

Information about CRC, Internal DMA, Transfer ADMA, and Auto CMD12 errors are detailed in the sections below.

33.5.3.5.1 CRC error

It is possible at the end of a block transfer that a write CRC status error or read CRC error occurs. For this type of error, the latest block received is discarded. This is because the integrity of the data block is not guaranteed. It is recommended to discard the following data blocks and re-transfer the block from the corrupted one. For a multi-block transfer, the host driver

issues a CMD12 to abort the current process and start the transfer by a new data command. In this scenario, even when the AC12EN and BCEND bits are set, uSDHC does not automatically send a CMD12 because the last block is not transferred. On the other hand, if it is within the last block that the CRC error occurs, an Auto CMD12 is sent by uSDHC. In this case, the host driver re-sends or re-obtains the last block with a single block transfer.

33.5.3.5.2 Internal DMA error

During the data transfer with internal Simple DMA, if the DMA engine encounters an error on the AXI bus, the DMA operation is aborted, and the DMA error interrupt is sent to the host system. When acknowledged by such an interrupt, the host driver calculates the start address of data block in which the error occurs. The start address can be calculated by either:

- Reading the DMA System Address register: The error occurs during the previous burst. Considering the block size, and the start address of the next burst transfer, it is straight forward to obtain the start address of the corrupted block.
- Reading the BLKCNT field of the Block Attribute register: Considering the number of blocks left, the total number to transfer, the start address of transfer, and the size of each block, the start address of corrupted block can be determined.To use this method, MIX_CTRL[BCEN] bit has to be set to enable Block Attribute register update.

When a DMA error occurs, it is recommended to abort the current transfer by means of a CMD12 (for multi block transfer), apply a reset for data, and restart the transfer from the corrupted block to recover from the error.

33.5.3.5.3 Transfer ADMA error

There are three kinds of possible ADMA errors: The AXI transfer, invalid descriptor, and data-length mismatch. Whenever these errors occur, the DMA transfer stops and the corresponding error status bit is set. For acknowledging the status, the host driver should recover the error as shown below and re-transfer from the place of interruption.

- AXI transfer error: Such errors may occur during data transfer or descriptor fetch. For either scenario, it is recommended to retrieve the transfer context, reset for the data part and re-transfer the block that was corrupted, or to the next block if no block is corrupted.
- Invalid descriptor error: For such errors, it is recommended to retrieve the transfer context, reset for the data part and recreate the descriptor chain from the invalid descriptor and issue a new transfer. As the data to transfer now may be less than the previous setting, the data length configured in the new descriptor chain should match the new value.
- Data-length mismatch error: It is similar to recover from this error. The Host Driver polls relating registers to retrieve the transfer context, apply a reset for the data part, configure a new descriptor chain and make another transfer if there is data left. Like the previous scenario of the invalid descriptor error, the data length must match the new transfer.

33.5.3.5.4 Auto CMD12 error

After the last block of the multi-block transfer is sent or received, and the AC12EN bit is set when the data transfer is initiated by the data command, uSDHC automatically sends a CMD12 to the card to stop the transfer. When errors with this command occur, it is recommended to the host driver to deal with the situations in the following manner:

- Auto CMD12 response time-out: It is not certain whether the command is accepted by the card or not. The host driver clears the auto CMD12 error status bits and re-send CMD12 until it is accepted by the card.
- Auto CMD12 response CRC error: As the card responds to CMD12, it aborts the transfer. The host driver may ignore the error and clear the error status bit.
- Auto CMD12 conflict error or not sent: The command is not sent; therefore, the host driver sends a CMD12 manually.

33.5.3.6 Card interrupt

The external cards can inform the host controller by means of some special signals. For SDIO, it can be the low-level on the DATA1 line during some special period. The uSDHC module only monitors the DATA1 line and supports the SDIO interrupt.

When the SDIO interrupt is captured by uSDHC, and the host system is informed by uSDHC asserting the uSDHC interrupt line, the interrupt service from the host driver is called.

As the interrupt source is controlled by the external card, the interrupt from SDIO must be serviced before the CINT bit is cleared. Refer to Card interrupt handling for the card interrupt handling flow.

33.5.4 Switch function

A switch command is issued by the host driver to enable new features added to the eMMC and SD cards. The eMMC and SD cards can transfer data at bus widths other than 1-bit. Different speed modes are also defined. To enable these features, a switch command is issued by the host driver.

For SDIO, the high-speed mode /DDR50/SDR50/SDR104 are enabled by writing the EHS bit in the CCCR register after the SHS bit is confirmed as high. For SD cards, the high-speed mode/DDR50/SDR50/SDR104 are queried and enabled by a CMD6 (with the mnemonic symbol as SWITCH_FUNC). For eMMC, the high-speed mode HS200 or HS400 is queried by a CMD8 and enabled by a CMD6 (with the mnemonic symbol as SWITCH).

The SDR4-bit, SDR8-bit, DDR4-bit, and DDR8-bit width of eMMC is also enabled by the SWITCH command, but with a different argument.

These new functions can also be disabled by a software reset. For SDIO, it can be done by setting the RES bit in the CCCR register. For other cards, it can be accomplished by issuing a CMD0. This method of restoring to the normal mode is not recommended because a complete identification process is needed before the card is ready for data transfer.

For the sake of simplicity, the following pseudo-code examples do not show current capability check, which is recommended in the function switch process.

33.5.4.1 Query, enable, and disable SDIO high-speed mode

```
enable_sdio_high_speed_mode(void)
{
send CMD52 to query bit SHS at address 0x13;
if (SHS bit is '0') report SDIO does not support high speed mode and return;
send CMD52 to set bit EHS at address 0x13 and read after write to confirm EHS bit is set;
change clock divisor value or configure the system clock feeding into uSDHC to generate the card_clk
of around 50MHz;
(data transactions like normal peers)
}
disable_sdio_high_speed_mode(void)
{
send CMD52 to clear bit EHS at address 0x13 and read after write to confirm EHS bit is cleared;
change clock divisor value or configure the system clock feeding into uSDHC to generate the card_clk
of the desired value below 25MHz;
(data transactions like normal peers)
}
```

33.5.4.2 Query, enable, and disable SD high-speed mode/DDR50 /SDR50 /SDR104

```
enable_sd_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0xFFFFFx and read 64 bytes of data accompanying the R1 response;(high speed
mode,x=1; SDR50,x=2; SDR104,x=3; DDR50,x=4;)
wait data transfer done bit is set;
check if the bit x of received 512 bits is set;
if (bit 401 is '0') report the SD card does not support high speed mode and return;
if (bit 402 is '0') report the SD card does not support SDR50 mode and return;
if (bit 403 is '0') report the SD card does not support SDR104 mode and return;
if (bit 404 is '0') report the SD card does not support DDR50 mode and return;
      send CMD6, with argument 0x80FFFFFx and read 64 bytes of data accompanying the R1 response;
(high speed mode,x=1; SDR50,x=2; SDR104,x=3; DDR50,x=4;)
check if the bit field 379~376 is 0xF;
```

```
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the card_clk
of around 50MHz for high speed mode, 100MHz for SDR50, 200MHz for SDR104, 50MHz for DDR50;
(data transactions like normal peers)
}
disable_sd_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0x80FFFFF0 and read 64 bytes of data accompanying the R1 response;
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

33.5.4.3 Query, enable, and disable eMMC high-speed mode

```
enable_mmc_high_speed_mode(void)
{
send CMD9 to get CSD value of eMMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the eMMC does not support high speed mode and return;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of eMMC;
extract the value of CARD_TYPE field to check the 'high speed mode' in this eMMC is 26MHz or
52MHz;
send CMD6 with argument 0x1B90100;
send CMD13 to wait card ready (busy line released);
send CMD8 to get EXT_CSD value of eMMC;
check if HS_TIMING byte (byte number 185) is 1;
if (HS_TIMING is not 1) report eMMC switching to high speed mode failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of around 26MHz or 52MHz according to the CARD_TYPE;
(data transactions like normal peers)
}
disable_mmc_high_speed_mode(void)
{
send CMD6 with argument 0x2B90100;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of eMMC;
check if HS_TIMING byte (byte number 185) is 0;
if (HS_TIMING is not 0) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 20MHz;
(data transactions like normal peers)
}
```

33.5.4.4 Set eMMC bus width

```
change_mmc_bus_width(void)
{
send CMD9 to get CSD value of eMMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the eMMC does not support multiple bit width and return;
send CMD6 with argument 0x3B70x00; (8-bit(dual data rate), x=6; 4-bit(dual data rate), x=5;8-bit,
x=2; 4-bit, x=1; 1-bit, x=0)
send CMD13 to wait card ready (busy line released);
```

```
(data transactions like normal peers)
}
```

33.5.5 ADMA operation

Here are the codes for the ADMA1 and ADMA2 operations.

33.5.5.1 ADMA1 operation

```
Set_adma1_descriptor
{
if (to start data transfer) {
// Make sure the address is 4KB align.
Set 'Set' type descriptor;
{
Set Act bits to 01;
Set [31:12] bits data length (byte unit);
}
Set 'Tran' type descriptor;
{
Set Act bits to 10;
Set [31:12] bits address (4KB align);
}
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to 11;
Set [31:12] bits the next descriptor address (4KB aligned);
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set End bit to 1;
}
if (to generate interrupt for this descriptor) {
Set Int bit to 1;
}
Set Valid bit to 1;
}
```

33.5.5.2 ADMA2 operation

```
Set_adma2_descriptor
{
if (to start data transfer) {
// Make sure the address is a 32-bit boundary (lower 2-bit are always '00').
Set higher 32-bit of descriptor for this data transfer initial address;
Set [31:16] bits data length (byte unit);
Set Act bits to '10';
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to '11';
// Make sure the address is 32-bit boundary (lower 2-bit are always set to '00').
Set higher 32-bit of descriptor for the next descriptor address;
}
else { // other types of descriptor
Set Act bits accordingly
}
```

```
if (this descriptor is the last one) {
Set 'End' bit '1';
}
if (to generate interrupt for this descriptor) {
Set 'Int' bit '1';
}
Set the 'Valid' bit to '1';
}
```

33.5.6 Fast boot operation

33.5.6.1 Normal fast boot flow

Here are the steps for normal fast boot flow:

1. Software must configure the SYS_CTRL[INITA] bit to make sure that 74 card clocks are finished.
2. Software must configure the eMMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 0 (normal fast boot), and bit 4 to select the ack mode. If the data is sent through the DMA mode, the software should configure bit 7 to enable the automatic stop at block gap feature, and configure bit 3-bit 0 to select the ack timeout value according to the SD CLK frequency.
3. Software then needs to configure the Block Attributes Register to set the block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes.
4. Software must configure the Protocol control register to set Data Transfer Width (DTW). If in the DDR fast boot mode, DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software needs to configure the Command Argument Register to set argument if needed (no need in normal fast boot).
6. Software must configure the Transfer Type Register to start the boot process. In normal boot mode, CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN and DMAEN retain the default value, where DPSEL bit is set to 1, DTDSEL is set to 1 and MSBSEL is set to 1.
7. DMAEN should be configured as 0 in the polling mode and if BCEN is configured as 1, it is recommended to configure the number of blocks in the Block Attributes Register to the maximum value. If in DDR fast boot mode, DDR_EN needs to be set to 1.
8. When step 6 is configured, the boot process begins. The software needs to poll the data buffer ready status to read the data from the buffer in time. If a boot timeout happens (ack times out or the first data read times out), an interrupt is triggered, and the software must configure eMMC Boot Register to bit 6 to 0 to disable boot. This makes CMD high, then after at least 56 clocks, it is ready to begin a normal initialization process.
9. If there is no timeout, software needs to determine when the data read is finished and then configure eMMC Boot Register bit 6 to 0 to disable boot. This render CMD line high and command completed asserted. After at least 56 clocks, it is ready to begin the normal initialization process.
10. You must reset the host and begin the normal process.

33.5.6.2 Alternative fast boot flow

Here are the steps for alternative fast boot flow:

1. Software needs to configure SYS_CTRL[INITA] to make sure 74 card clocks are finished.
2. Software needs to configure eMMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 1 (alternative boot), and bit 4 to select the ack mode or not. If data needs to be sent through the DMA mode, then configure bit 7 to enable the automatic stop at block gap feature. Software should also configure bit 3-bit 0 to select the ack timeout value according to the SD clock frequency.
3. Software then needs to configure the Block Attributes Register to set the block size and count. If in the DDR fast boot mode, the block size only can be configured to 512 bytes.

4. Software needs to configure the Protocol control register to set the data transfer width (DTW). If in the DDR fast boot mode, DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software needs to configure Command Argument Register to set argument to 0xFFFFFFFA.
6. Software needs to configure the Transfer Type Register to start the boot process by CMD0 with the 0xFFFFFFFA argument. In alternative boot, CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN, and DMAEN retain the default value. DPSEL bit is set to 1, DTDSEL is set to 1, and MSBSEL is set to 1. Note DMAEN should be configured as 0 in the polling mode, and if BCEN is configured as 1 in polling mode, it is recommended to configure the block count in the Block Attributes Register to the maximum value. If in the DDR fast boot mode, DDR_EN needs to be set to 1.
7. When step 6 is configured, the boot process begins. Software needs to poll the data buffer ready status to read the data from the buffer in time. If there is a boot timeout (ack data timeout in 50ms or data timeout in 1s), the host sends out the interrupt and the software needs to send CMD0 with reset and then configure the boot enable bit to 0 to stop this process.
8. If there is no time out, the software needs to decide when to stop the boot process, and send out the CMD0 with reset and then after the command is completed, configure the eMMC Boot Register bit 6 to stop the process. After 8 clocks from the command completion, the slave (card) is ready for the identification step.
9. You must reset the host and begin the normal process.

33.5.6.3 Fast boot application case (in DMA mode)

In the boot application case, because the image destination and the image size are contained in the beginning of the image, it is necessary to switch DMA parameters on the fly during eMMC fast boot.

In fast boot, the host can use Advanced DMA2 (ADMA2) with two destinations.

The detailed flow is described below:

1. The software needs to configure INIT_ACTIVE bit (system control register bit 27) to make sure that 74 card clocks are finished.
2. The software needs to configure the eMMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot); and bit 5 to 0 (normal fast boot) or 1 (alternative boot); and bit 4 to select the ack mode. In DMA mode, configure bit 7 to 1 to enable the automatic stop at block gap feature. Also configure bits[31-16] to set the (BLK_CNT - VALUE1). Here VALUE1 is the value of the block count that needs to transfer the first time, so that the host stops at the block gap when the uSDHC controller gets VAULE1 blocks from the device. Also, configure bits[3-0] to select the ack timeout value according to the SD clock frequency.
3. The software then needs to configure the Block Attributes Register to set block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes. In DMA mode, it is recommended to set the block count (BLK_CNT) to the max value (16'hffff).
4. The software needs to configure Protocol Control Register to set DTW (data transfer width). If in DDR fast boot mode, the DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software enable ADMA2 by configuring Protocol Control Register bits [9-8].
6. The software needs to set at least three pairs of ADMA2 descriptor in boot memory (that is, in IRAM, at least six words). The first pair descriptor defines the start address (that is, IRAM) and data length (that is,512byte*VALUE1) of the first part boot code. The software also needs to set the second pair descriptor, the second start address (any value that is writable), and data length is suggested to set 1~2word (record as VALUE2). Note that the second couple desc also transfers useful data even at lease 1 word, because our ADMA2 cannot support 0 data_length data transfer descriptor.
7. The software needs to configure Command Argument Register to set argument to 0xFFFFFFFA in alternative fast boot and do not need to be set in normal fast boot.
8. The software needs to configure Transfer Type Register to start the boot process. CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN, and DMAEN retain the default value. DPSEL bit is set to 1, DTDSEL is set to 1, and MSBSEL is set to 1. DMAEN is configured as 1 in the DMA mode. And, if BCEN is configured as 1, then configure blk no in Bock Attributes Register to the max value. And, if in the DDR fast boot mode, DDR_EN needs to be set to 1.

9. When step 8 is configured, boot process begins, the first VALUE1 block number data gets transferred. The software needs to poll the TC bit (bit1 in Interrupt Status Register) to determine first transfer is ended. Also, the software needs to polling the BGE bit (bit2 in Interrupt Status Register) to determine if the first transfer stops at the block gap.
10. When TC and BGE bits are set to 1, the software can analyze the first code of VALUE1 block, initializes the new memory device, if required, and sets the third pair of descriptors to define the start address and length of the remaining part of the boot code (VALUE3, the remain boot code block). Remember to set the last descriptor with END.
11. The software needs to configure the eMMC Boot Register (offset 0xc4) again. Set bit 6 to 1 (enable boot); and bit 5 to 0 (normal fast boot), to 1 (alternative boot); and bit 4 to select the ack mode or not. In the DMA mode, configure bit 7 to 1 for enabling the automatically stop at block gap feature. Also, configure bit31-bit16 to set the (BLK_CNT - (VALUE1+1+VALUE3)), that host stops at block gap when the uSDHC controller gets (VALUE1+1+VALUE3)) blocks from device totally include the blocks received in step 9. And need to configure bit 3-bit0 to select the ack timeout value according to the sd clk frequency. Note that the software does not need to configure the BLK_CNT again, because it is counted down automatically by the uSDHC controller.
12. The software needs to clear the TC and BGE bits and the software needs to clear SABGREQ (bit 16 in the Protocol control register) and set CREQ (bit17 in the Protocol control register) to 1 to resume the data transfer. Host transfers the VALUE2 and VALUE3 data to the destination that is set by descriptor.
13. The software needs to do poll BGE bit to determine if the fast boot is over.

Note:

- When ADMA boot flow starts, for uSDHC, it is like a normal ADMA read operation. So, set ADMA2 descriptor as the normal ADMA2 transfer.
- Need a few words length memory to keep descriptor.
- For the 1~2-word data in second descriptor setting, it is a useful data, so the software needs to deal the data because of the application case.

33.6 uSDHC memory map and register definition

33.6.1 uSDHC register descriptions

This section includes the module memory map and detailed descriptions of all registers.

See the table below for the register memory map of uSDHC. All these registers only support 32-bit accesses.

NOTE

The uSDHC registers are 32-bit wide and only support 32-bit access.

33.6.1.1 uSDHC memory map

USDHC1 base address: 4285_0000h

USDHC2 base address: 4286_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	DMA System Address (DS_ADDR)	32	RW	0000_0000h
4h	Block Attributes (BLK_ATT)	32	RW	0001_0000h
8h	Command Argument (CMD_ARG)	32	RW	0000_0000h
Ch	Command Transfer Type (CMD_XFR_TYP)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10h	Command Response0 (CMD_RSP0)	32	R	0000_0000h
14h	Command Response1 (CMD_RSP1)	32	R	0000_0000h
18h	Command Response2 (CMD_RSP2)	32	R	0000_0000h
1Ch	Command Response3 (CMD_RSP3)	32	R	0000_0000h
20h	Data Buffer Access Port (DATA_BUFF_ACC_PORT)	32	RW	0000_0000h
24h	Present State (PRES_STATE)	32	R	See section
28h	Protocol Control (PROT_CTRL)	32	RW	0880_0020h
2Ch	System Control (SYS_CTRL)	32	RW	0080_800Fh
30h	Interrupt Status (INT_STATUS)	32	RW	0000_0000h
34h	Interrupt Status Enable (INT_STATUS_EN)	32	RW	0000_0000h
38h	Interrupt Signal Enable (INT_SIGNAL_EN)	32	RW	0000_0000h
3Ch	Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS)	32	RW	0000_0000h
40h	Host Controller Capabilities (HOST_CTRL_CAP)	32	RW	07F3_B407h
44h	Watermark Level (WTMK_LVL)	32	RW	0810_0810h
48h	Mixer Control (MIX_CTRL)	32	RW	8000_0000h
50h	Force Event (FORCE_EVENT)	32	RW	0000_0000h
54h	ADMA Error Status (ADMA_ERR_STATUS)	32	R	0000_0000h
58h	ADMA System Address (ADMA_SYS_ADDR)	32	RW	0000_0000h
60h	DLL (Delay Line) Control (DLL_CTRL)	32	RW	0000_0000h
64h	DLL Status (DLL_STATUS)	32	R	0000_0200h
68h	CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS)	32	RW	0000_0000h
70h	Strobe DLL control (STROBE_DLL_CTRL)	32	RW	0000_0000h
74h	Strobe DLL status (STROBE_DLL_STATUS)	32	R	0000_0200h
C0h	Vendor Specific Register (VEND_SPEC)	32	RW	3000_7809h
C4h	eMMC Boot (MMC_BOOT)	32	RW	0000_0000h
C8h	Vendor Specific 2 Register (VEND_SPEC2)	32	RW	0001_9006h
CCh	Tuning Control (TUNING_CTRL)	32	RW	0021_2800h
100h	Command Queuing Version (CQVER)	32	R	0000_0510h
104h	Command Queuing Capabilities (CQCAP)	32	RW	0000_310Ah
108h	Command Queuing Configuration (CQCFG)	32	RW	0000_0000h
10Ch	Command Queuing Control (CQCTL)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
110h	Command Queuing Interrupt Status (CQIS)	32	RW	0000_0000h
114h	Command Queuing Interrupt Status Enable (CQISTE)	32	RW	0000_0000h
118h	Command Queuing Interrupt Signal Enable (CQISGE)	32	RW	0000_0000h
11Ch	Command Queuing Interrupt Coalescing (CQIC)	32	RW	0000_0000h
120h	Command Queuing Task Descriptor List Base Address (CQTDLBA)	32	RW	0000_0000h
124h	Command Queuing Task Descriptor List Base Address Upper 32 Bits (CQTDLBAU)	32	RW	0000_0000h
128h	Command Queuing Task Doorbell (CQTDBR)	32	RW	0000_0000h
12Ch	Command Queuing Task Completion Notification (CQTCN)	32	RW	0000_0000h
130h	Command Queuing Device Queue Status (CQDQS)	32	R	0000_0000h
134h	Command Queuing Device Pending Tasks (CQDPT)	32	R	0000_0000h
138h	Command Queuing Task Clear (CQTCLR)	32	RW	0000_0000h
140h	Command Queuing Send Status Configuration 1 (CQSSC1)	32	RW	0001_1000h
144h	Command Queuing Send Status Configuration 2 (CQSSC2)	32	RW	0000_0000h
148h	Command Queuing Command Response for Direct-Command Task (CQCRDCT)	32	R	0000_0000h
150h	Command Queuing Response Mode Error Mask (CQRMEM)	32	RW	FDF9_A080h
154h	Command Queuing Task Error Information (CQTERRI)	32	R	0000_0000h
158h	Command Queuing Command Response Index (CQCRI)	32	R	0000_0000h
15Ch	Command Queuing Command Response Argument (CQCRA)	32	R	0000_0000h

33.6.1.2 DMA System Address (DS_ADDR)

Offset

Register	Offset
DS_ADDR	0h

Function

This register contains the physical system memory address used for DMA transfers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DS_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DS_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DS_ADDR	System address **DMA system address / argument 2** When ACMD23_ARGU2_EN is set to 0, SDMA uses this register as system address and supports only 32-bit addressing mode. Auto CMD23 cannot be used with SDMA. When ACMD23_ARGU2_EN is set to 1, SDMA uses ADMA System Address register (05Fh – 058h) instead of this register to support both ADMA1 and ADMA2. This register is used only for Argument2 and SDMA may use Auto CMD23. 1. **SDMA system address** Because the address must be word (4 bytes) aligned, the least two bits are reserved and always set to 0. When uSDHC stops a DMA transfer, this register points out the system address of the next contiguous data position. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operation during transfers may return an invalid value. The host driver initializes this register before starting a DMA transaction. After DMA has stopped, the system address of the next contiguous data position can be read from this register. This register is protected during a data transfer. When data lines are active, write to this register is ignored. The host driver waits until the DLA field in the Present State register is cleared, before writing to this register. The uSDHC internal DMA does not support a virtual memory system. It only supports continuous physical memory access. Because this register supports dynamic address reflecting, when TC field is set, it automatically alters the value of internal address counter, so the software cannot change this register when TC field is set. Such restriction is also listed in Software restrictions. 2. **Argument 2** This register is used with the Auto CMD23 to set a 32-bit block count value to the argument of the CMD23 while executing Auto CMD23. If Auto CMD23 is used with ADMA, the full 32-bit block count value can be used. If Auto CMD23 is used without ADMA, the available block count value is limited by the Block Count register. 65535 blocks is the maximum value in this case.

33.6.1.3 Block Attributes (BLK_ATT)

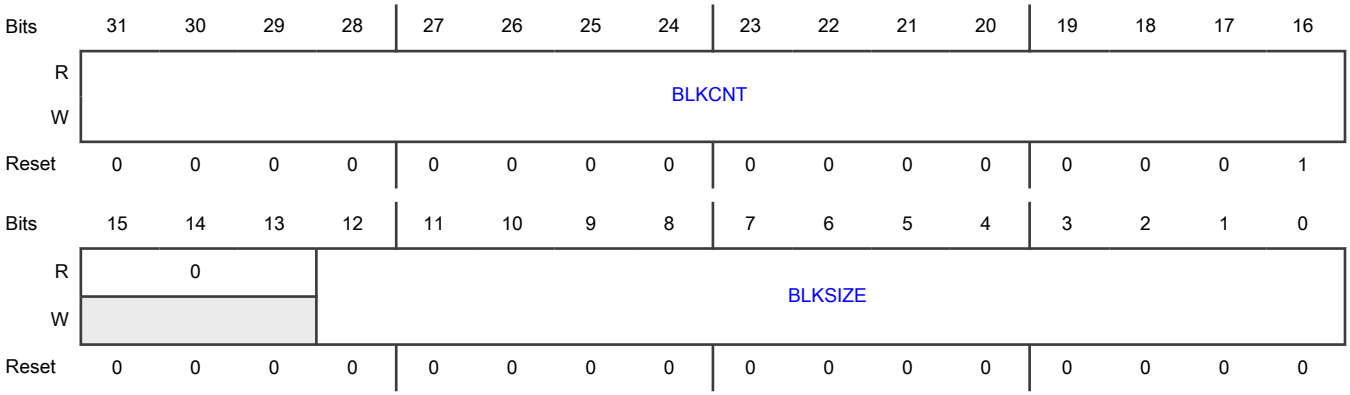
Offset

Register	Offset
BLK_ATT	4h

Function

This register is used to configure the number of data blocks and the number of bytes in each block.

Diagram



Fields

Field	Function
31-16 BLKCNT	Blocks count for current transfer This field is enabled when the Block Count Enable field in the Transfer Mode register is set to 1 and is valid only for multiple block transfers. For single block transfer, this field always reads as 1. The host driver sets this field to a value between 1 and the maximum block count. The uSDHC module decrements the block count after each block transfer and stops when the count reaches zero. Setting the block count to zero results in no data blocks being transferred. This field should be accessed only when no transaction is executing (that is, after transactions are stopped). During data transfer, read operations on this field may return an invalid value and write operations are ignored. When saving transfer content because of a Suspend command, the number of blocks yet to be transferred can be determined by reading this field. The reading of this field should be applied after transfer is paused by stop at block gap operation and before sending the command marked as suspend. This is because when the Suspend command is sent out, uSDHC treats the current transfer as aborted and change the BLKCNT field back to its original value instead of keeping the dynamical indicator of the remaining block count. When restoring transfer content prior to issuing a Resume command, the host driver restores the previously saved block count.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Although the BLKCNT field is 0 after reset, the read of reset value is 0x1. This is because when MSBSEL field is indicating a single block transfer, the read value of BLKCNT is always 1. 0000_0000_0000_0000b - Stop count 0000_0000_0000_0001b - 1 block 0000_0000_0000_0010b - 2 blocks 1111_1111_1111_1111b - 65535 blocks
15-13 —	Reserved
12-0 BLKSIZE	Transfer block size This field specifies the block size for block data transfers. Values ranging from 1 byte up to the maximum buffer size can be set. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operations during transfers may return an invalid value, and write operations are ignored. 0_0000_0000_0000b - No data transfer 0_0000_0000_0001b - 1 byte 0_0000_0000_0010b - 2 bytes 0_0000_0000_0011b - 3 bytes 0_0000_0000_0100b - 4 bytes 0_0001_1111_1111b - 511 bytes 0_0010_0000_0000b - 512 bytes 0_1000_0000_0000b - 2048 bytes 1_0000_0000_0000b - 4096 bytes

33.6.1.4 Command Argument (CMD_ARG)

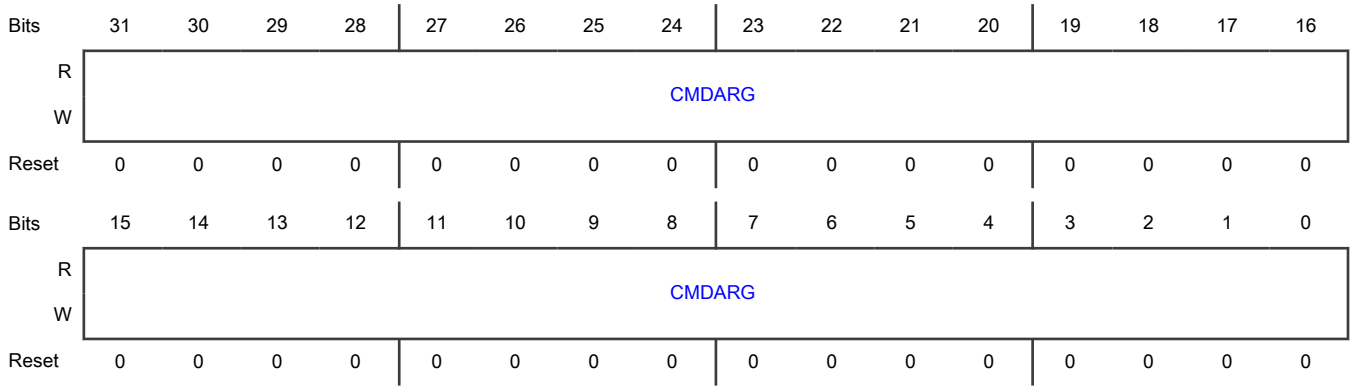
Offset

Register	Offset
CMD_ARG	8h

Function

This register contains the SD/eMMC command argument.

Diagram



Fields

Field	Function
31-0 CMDARG	Command argument The SD/eMMC command argument is specified as bits 39-8 of the command format in the SD or eMMC specification. This field is write protected when the Command Inhibit (CMD) field in the Present State register is set.

33.6.1.5 Command Transfer Type (CMD_XFR_TYP)

Offset

Register	Offset
CMD_XFR_TYP	Ch

Function

This register is used to control the operation of data transfers. The host driver sets this register before issuing a command followed by a data transfer or before issuing a Resume command. To prevent data loss, uSDHC prevents writing to the bits, which are involved in the data transfer of this register, when data transfer is active. These fields are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN, and DMAEN.

The host driver checks the Command Inhibit DAT field (PRES_STATE[CDIHB]) and the Command Inhibit CMD field (PRES_STATE[CIHB]) in the Present State register before writing to this register. When the CDIHB field in the Present State register is set, any attempt to send a command with data by writing to this register is ignored; when the CIHB field is set, any write to this register is ignored.

On sending commands with data transfer involved, it is mandatory that the block size is non-zero. Block count must also be non-zero, or indicated as a single block transfer (bit 5 of this register is '0' when written), or block count is disabled (bit 1 of this register is '0' when written), otherwise uSDHC ignores the sending of this command and do nothing. For write command, with all above restrictions, it is also mandatory that the write protect switch is not active (PRES_STATE[WPSPL] field of Present State register is '1'); otherwise, uSDHC also ignores the command.

If the commands with data transfer do not receive the response in 64 clock cycles, that is, if response time-out happens, uSDHC treats the external device, does not accept the command, and aborts the data transfer. In this scenario, the driver should issue the command again to retry the transfer. It is also possible that for some reason the card responds to the command but uSDHC

does not receive the response, and if it is internal DMA (either simple DMA or ADMA) read operation, the external system memory is over-written by the internal DMA with data sent back from the card.

The table below shows the summary of how register settings determine the type of data transfer.

Table 250. Transfer type register setting for various transfer types

Multi/single block select	Block count enable	Block count	Function
0	Do not care	Do not care	Single transfer
1	0	Do not care	Infinite transfer
1	1	Positive number	Multiple transfer
1	1	Zero	No data transfer

The table below shows the relationship between the Command Index Check Enable and the Command CRC Check Enable, regarding the Response Type bits as well as the name of the response type.

Table 251. Relationship between parameters and the name of the response type

Response type	Index check enable	CRC check enable	Name of response type
00	0	0	No response
01	0	1	R2
10	0	0	R3, R4
10	1	1	R1, R5, R6
11	1	1	R1b,R5b

- In the SDIO specification, response type notation for R5b is not defined. R5 includes R5b in the SDIO specification, but R5b is defined in this specification to specify that uSDHC checks the busy status after receiving a response. For example, usually CMD52 is used with R5, but the I/O abort command is used with R5b.
- The CRC fields for R3 and R4 are expected to be all 1 bits. The CRC check is disabled for these response types.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		CMDINX						CMDTYP		DPSEL	CICEN	CCCEN	Reserved	RSPTYP	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								AC23EN	NIBBLE_...	MSBSEL	DTDSEL	DDR_EN	AC12EN	BCEN	DMAEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 CMDINX	Command index These fields are set to the command number that is specified in bits 45-40 of the command-format in the SD Memory Card Physical Layer Specification and SDIO Specification.
23-22 CMDTYP	Command type There are three types of special commands: Suspend, Resume, and Abort. These fields are set to 00b for all other commands. • Suspend command: If the Suspend command succeeds, uSDHC assumes that the card bus has been released and that it is possible to issue the next command that uses the DATA line. Because uSDHC does not monitor the content of command response, it does not know if the Suspend command succeeded or not. It is the host driver's responsibility to check the status of the Suspend command and send another command marked as Suspend to inform uSDHC that a Suspend command was successfully issued. See Suspend Resume for more details. After the end bit of command is sent, uSDHC deasserts Read Wait for read transactions and stops checking busy for write transactions. In a 4-bit mode, the interrupt cycle starts. If the Suspend command fails, uSDHC maintains its current state, and the host driver restarts the transfer by setting the Continue Request field in the Protocol Control register. • Resume command: The host driver re-starts the data transfer by restoring the registers saved before sending the Suspend command and then sends the Resume command. The uSDHC module checks for a pending busy state before starting write transfers. • Abort command: If this command is set when executing a read transfer, uSDHC stops reads to the buffer. If this command is set when executing a write transfer, uSDHC stops driving the DATA line. After issuing the Abort command, the host driver should issue a software reset (Abort Transaction). 00b - Normal other commands 01b - Suspend CMD52 for writing bus suspend in CCCR 10b - Resume CMD52 for writing function select in CCCR 11b - Abort CMD12, CMD52 for writing I/O Abort in CCCR
21 DPSEL	Data present select This field is set to 1 to indicate that data is present and is transferred using the DATA line. It is set to 0 for the following: • Commands using only the CMD line (for example, CMD52) • Commands with no data transfer, but using the busy signal on DATA0 line (R1b or R5b (for example, CMD38)) **NOTE** In resume command, this field is set, and other bits in this register is set the same as when the transfer was initially launched. When the write protect switch is on, (that is, the WPSPL field is active as '0'), any command with a write operation ignored. When this field is set, while the DTDSEL field is 0, writes to the register Transfer Type are ignored.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No data present 1b - Data present
20 CICEN	Command index check enable If this field is set to 1, uSDHC checks the Index field in the response to see if it has the same value as the command index. If it is not, it is reported as a Command Index Error. If this field is set to 0, the Index field is not checked. 0b - Disable command index check 1b - Enables command index check
19 CCCEN	Command CRC check enable If this field is set to 1, uSDHC checks the CRC field in the response. If an error is detected, it is reported as a Command CRC Error. If this field is set to 0, the CRC field is not checked. The number of bits checked by the CRC field value changes according to the length of the response. See RSPTYP[1:0] and Command Transfer Type (CMD_XFR_TYP). 0b - Disables command CRC check 1b - Enables command CRC check
18 —	Reserved
17-16 RSPTYP	Response type select 00b - No response 01b - Response length 136 10b - Response length 48 11b - Response length 48, check busy after response
15-8 —	Reserved
7 AC23EN	AC23EN This field is read when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is tied to '0'. When this field is set to 1, the host controller issues a CMD23 automatically before issuing a command specified in the Command Register. 0b - Disable 1b - Enable
6 NIBBLE_POS	NIBBLE_POS This field indicates the nibble position in the DDR 4-bit mode. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. 0- the sequence is 'odd high nibble -> even high nibble -> odd low nibble -> even low nibble'; 1- the sequence is 'odd high nibble -> odd low nibble -> even high nibble -> even low nibble'.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
5 MSBSEL	MSBSEL This field enables multiple block DATA line data transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. For any other commands, this field can be set to 0. If this field is 0, it is not necessary to set the Block Count register. See Command Transfer Type (CMD_XFR_TYP). 0b - Disable 1b - Enable
4 DTDSEL	DTDSEL This field defines the direction of DATA line data transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. The field is set to 1 by the host driver to transfer data from the SD card to uSDHC and is set to 0 for all other commands. 0b - Disable 1b - Enable
3 DDR_EN	DDR_EN Dual data rate mode selection. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. 0b - Disable 1b - Enable
2 AC12EN	AC12EN Multiple block transfers for memory require a CMD12 to stop the transaction. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. When this field is set to 1, uSDHC issues a CMD12 automatically when the last block transfer has completed. The host driver is not set this field to issue commands that do not require CMD12 to stop a multiple block data transfer. In particular, secure commands defined in File Security Specification (see reference list) do not require CMD12. In single block transfer, uSDHC ignores this field no matter it is set or not. 0b - Disable 1b - Enable
1 BCEN	BCEN This field is used to enable the Block Count register, which is only relevant for multiple block transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. When this field is 0, the internal counter for block is disabled, which is useful in executing an infinite transfer. 0b - Disable 1b - Enable
0	DMAEN

Table continues on the next page...

Table continued from the previous page...

Field	Function
DMAEN	This field enables DMA functionality. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is enabled; otherwise, this field is read-only. If this field is set to 1, a DMA operation begins when the host driver sets the DPSEL field of this register. Whether the simple DMA or the advanced DMA is active depends on the DMA Select field of the Protocol Control register. 0b - Disable 1b - Enable

33.6.1.6 Command Response0 (CMD_RSP0)

Offset

Register	Offset
CMD_RSP0	10h

Function

This register is used to store part 0 of the response bits from the card.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMDRSP0	Command response 0 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

33.6.1.7 Command Response1 (CMD_RSP1)

Offset

Register	Offset
CMD_RSP1	14h

Function

This register is used to store part 1 of the response bits from the card.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMDRSP1	Command response 1 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

33.6.1.8 Command Response2 (CMD_RSP2)

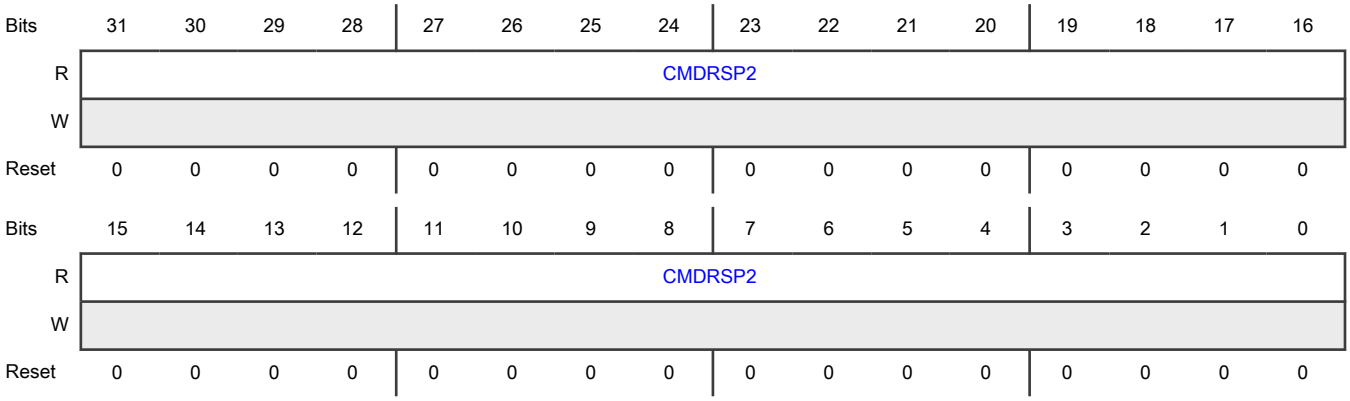
Offset

Register	Offset
CMD_RSP2	18h

Function

This register is used to store part 2 of the response bits from the card.

Diagram



Fields

Field	Function
31-0 CMDRSP2	Command response 2 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

33.6.1.9 Command Response3 (CMD_RSP3)

Offset

Register	Offset
CMD_RSP3	1Ch

Function

This register is used to store part 3 of the response bits from the card.

The table below describes the mapping of command responses from the SD bus to Command Response registers for each response type. In this table, R[ ] refers to a bit range within the response data as transmitted on the SD bus.

Table 252. Response bit definition for each response type

Response type	Meaning of response	Response field	Response register
R1,R1b (normal response)	Card status	R[39:8]	CMDRSP0
R1b (auto CMD12 response)	Card status for auto CMD12	R[39:8]	CMDRSP3
R2 (CID, CSD register)	CID/CSD register [127:8]	R[127:8]	{CMDRSP3[23:0], CMDRSP2, CMDRSP1, CMDRSP0}

Table continues on the next page...

Table 252. Response bit definition for each response type (continued)

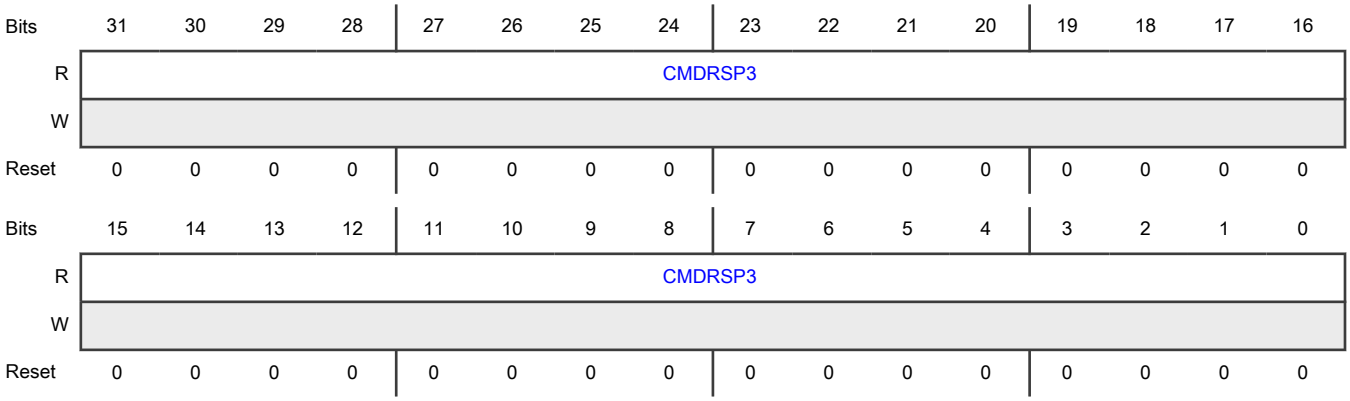
Response type	Meaning of response	Response field	Response register
R3 (OCR register)	OCR register for memory	R[39:8]	CMDRSP0
R4 (OCR register)	OCR register for I/O etc.	R[39:8]	CMDRSP0
R5, R5b	SDIO response	R[39:8]	CMDRSP0
R6 (publish RCA)	New published RCA[31:16] and card status[15:0]	R[39:9]	CMDRSP0

This table shows that most responses with a length of 48 (R[47:0]) have 32-bits of the response data (R[39:8]) stored in the CMDRSP0 register. Responses of type R1b (Auto CMD12 responses) have response data bits (R[39:8]) stored in the CMDRSP3 register. Responses with length 136 (R[135:0]) have 120-bits of the response data (R[127:8]) stored in the CMDRSP0, 1, 2, and 3 registers.

To be able to read the response status efficiently, uSDHC only stores part of the response data in the Command Response registers. This enables the host driver to efficiently read 32-bits of response data in one read cycle on a 32-bit bus system. Parts of the response, the Index field and the CRC, are checked by uSDHC (as specified by the Command Index Check Enable and the Command CRC Check Enable bits in the Transfer Type register) and generate an error interrupt if any error is detected. The bit range for the CRC check depends on the response length. If the response length is 48, uSDHC checks R[47:1], and if the response length is 136 the uSDHC checks R[119:1].

Because uSDHC may have a multiple block data transfer executing concurrently with a CMD_wo_DAT command, uSDHC stores the Auto CMD12 response in the CMDRSP3 register. The CMD_wo_DAT response is stored in CMDRSP0. This allows uSDHC to avoid overwriting the Auto CMD12 response with the CMD_wo_DAT and vice versa. When uSDHC modifies part of the Command Response registers, as shown in the table above, it preserves the unmodified bits.

Diagram



Fields

Field	Function
31-0 CMDRSP3	Command response 3 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

33.6.1.10 Data Buffer Access Port (DATA_BUFF_ACC_PORT)

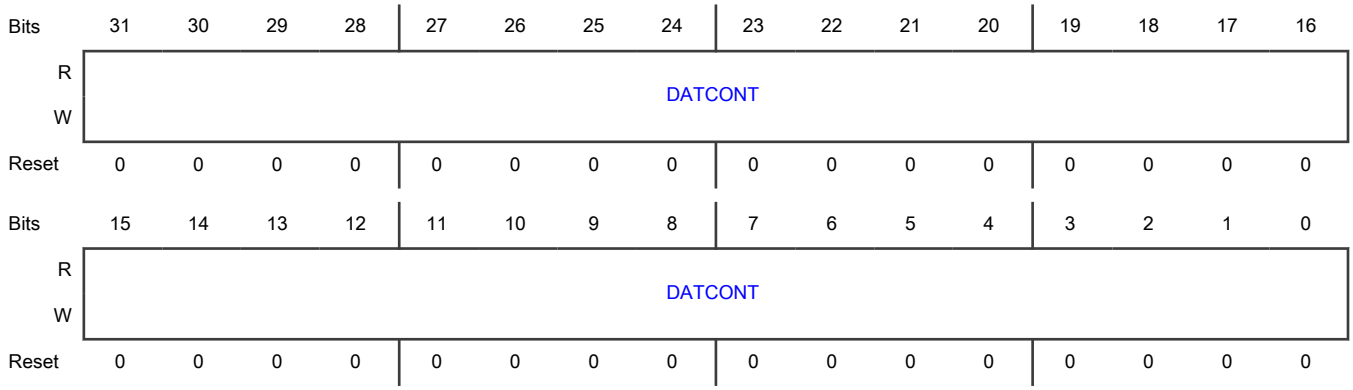
Offset

Register	Offset
DATA_BUFF_ACC_PORT	20h

Function

The Buffer Data Port register is for 32-bit data access by the Arm platform. When the internal DMA is enabled, any write to this field is ignored, and any read from this field always yields 0s.

Diagram



Fields

Field	Function
31-0 DATCONT	Data content This field is used to access the internal buffer.

33.6.1.11 Present State (PRES_STATE)

Offset

Register	Offset
PRES_STATE	24h

Function

The host driver can get status of uSDHC from this 32-bit read only register.

The host driver can issue CMD0, CMD12, CMD13 (for memory) and CMD52 (for SDIO) when the DATA lines are busy during a data transfer. These commands can be issued when Command Inhibit (CMD) is set to zero. Other commands are issued when Command Inhibit (DATA) is set to zero. Possible changes to the SD Physical Specification may add other commands to this list in the future.

NOTE

The reset value of Present State register depends on board connectivity.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLSL								CLSL	0			WPSPL	CDPL	0	CINST
W																
Reset	u	u	u	u	u	u	u	u	u	0	0	0	u	u	0	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TSCD	0		RTR	BREN	BWEN	RTA	WTA	Reserved				SDSTB	DLA	CDIHB	CIHB
W																
Reset	1	0	0	0	0	0	0	0	1	0	0	0	u	0	0	0

Fields

Field	Function
31-24 DLSL	DATA[7:0] line signal level This field is used to check the DATA line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from DATA0. The reset value is affected by the external pull-up / pull-down resistors. By default, the read value of this field after reset is 8'b11110111, when DATA3 is pulled down and the other lines are pulled up. 0000_0000b - Data 0 line signal level 0000_0001b - Data 1 line signal level 0000_0010b - Data 2 line signal level 0000_0011b - Data 3 line signal level 0000_0100b - Data 4 line signal level 0000_0101b - Data 5 line signal level 0000_0110b - Data 6 line signal level 0000_0111b - Data 7 line signal level
23 CLSL	CMD line signal level This field is used to check the CMD line level to recover from errors, and for debugging. The reset value is affected by the external pull-up / pull-down resistor, by default, the read value of this field after reset is 1'b1, when the command line is pulled up.
22-20 —	Reserved
19	Write protect switch pin level

Table continues on the next page...

Table continued from the previous page...

Field	Function
WPSPL	The Write Protect switch is supported for memory and combo cards. This field reflects the inverted value of the WP pin of the card socket. A software reset does not affect this field. The reset value is affected by the external write protect switch. If the WP pin is not used, it should be tied low, so that the reset value of this field is high and write is enabled. 0b - Write protected (WP = 1) 1b - Write enabled (WP = 0)
18 CDPL	Card detect pin level This field reflects the inverse value of the CD_B pin for the card socket. Debouncing is not performed on this field. This field may be valid, but is not guaranteed, because of propagation delay. Use of this field is limited to testing because it must be debounced by software. A software reset does not affect this field. A write to the Force Event Register does not affect this field. The reset value is affected by the external card detection pin. This field shows the value on the CD_B pin (that is, when a card is inserted in the socket, it is 0 on the CD_B input, and consequently, the CDPL reads 1. 0b - No card present (CD_B = 1) 1b - Card present (CD_B = 0)
17 —	Reserved
16 CINST	Card inserted This field indicates whether a card has been inserted. The uSDHC module debounces this signal so that the host driver does not need to wait for it to stabilize. Changing from a 0 to 1 generates a Card Insertion interrupt in the Interrupt Status register. Changing from a 1 to 0 generates a Card Removal interrupt in the Interrupt Status register. A write to the Force Event Register does not affect this field. The Software Reset For All in the System Control register does not affect this field. A software reset does not affect this field. 0b - Power on reset or no card 1b - Card inserted
15 TSCD	Tap select change done This field indicates the delay setting is effective after write CLK_TUNE_CTRL_STATUS register. 0b - Delay cell select change is not finished. 1b - Delay cell select change is finished.
14-13 —	Reserved
12 RTR	Re-Tuning Request (only for SD3.0 SDR104 mode, and eMMC HS200 mode) Host controller may request host driver to execute re-tuning sequence by setting this field when the data window is shifted by temperature drift and a tuned sampling point does not have a good margin to receive correct data.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is cleared when a command is issued with setting Execute Tuning field in MIX_CTRL register. Changing of this field from 0 to 1 generates Re-Tuning Event. See Interrupt status registers for more detail. This field isn't set to 1 if Sampling Clock Select in the MIX_CTRL register is set to 0 (using fixed sampling clock). 0b - Fixed or well tuned sampling clock 1b - Sampling clock needs re-tuning
11 BREN	Buffer read enable This status field is used for non-DMA read transfers. The uSDHC module implements an internal buffer to transfer data efficiently. This read only flag indicates that valid data exists in the host side buffer. If this field is high, valid data greater than the watermark level exist in the buffer. A change of this field from 1 to 0 occurs when some reads from the buffer (read DATPORT (Base + 0x20)) are made and the buffer hasn't valid data greater than the watermark level. A change of this field from 0 to1 occurs when there is enough valid data ready in the buffer and the Buffer Read Ready interrupt has been generated and enabled. 0b - Read disable 1b - Read enable
10 BWEN	Buffer write enable This status field is used for non-DMA write transfers. The uSDHC module implements an internal buffer to transfer data efficiently. This read only flag indicates if space is available for write data. If this field is 1, valid data greater than the watermark level can be written to the buffer. A change of this field from 1 to 0 occurs when some writes to the buffer (write DATPORT (Base + 0x20)) are made and the buffer hasn't valid space greater than the watermark level. A change of this field from 0 to 1 occurs when the buffer can hold valid data greater than the write watermark level and the Buffer Write Ready interrupt is generated and enabled. 0b - Write disable 1b - Write enable
9 RTA	Read transfer active This status field is used for detecting completion of a read transfer. This field is set for either of the following conditions: • After the end field of the read command • When writing a 1 to the Continue Request field in the Protocol Control register to restart a read transfer A transfer complete interrupt is generated when this field changes to 0. This field is cleared for either of the following conditions: • When the last data block as specified by block length is transferred to the System, that is, all data are read away from uSDHC internal buffer. • When all valid data blocks have been transferred from uSDHC internal buffer to the System and no current block transfers are being sent because of the Stop At Block Gap Request being set to 1. 0b - No valid data

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Transferring data
8 WTA	Write transfer active This status field indicates a write transfer is active. If this field is 0, it means no valid write data exists in uSDHC. This field is set in either of the following cases: • After the end field of the write command • When writing 1 to the Continue Request field in the Protocol Control register to restart a write transfer This field is cleared in either of the following cases: • After getting the CRC status of the last data block as specified by the transfer count (Single and Multiple) • After getting the CRC status of any block where data transmission is about to be stopped by a Stop At Block Gap Request During a write transaction, a Block Gap Event interrupt is generated when this field is changed to 0, as result of the Stop At Block Gap Request being set. This status is useful for the host driver in determining when to issue commands during Write Busy state. 0b - No valid data 1b - Transferring data
7-4 —	Reserved
3 SDSTB	SD clock stable This status field indicates that the internal card clock is stable. This field is for the host driver to poll clock status when changing the clock frequency. It is recommended to clear FRC_SDCLK_ON field in System Control register to remove glitches on the card clock when the frequency is changing. Before changing clock divisor value (SDCLKFS or DVS), host driver should make sure the SDSTB field is high. 0b - Clock is changing frequency and not stable. 1b - Clock is stable.
2 DLA	Data line active This status field indicates whether one of the DATA lines on the SD bus is in use. In the case of read transactions: This status indicates if a read transfer is executing on the SD bus. Changes in this value from 1 to 0, between data blocks, generates a Block Gap Event interrupt in the Interrupt Status register. This field is set in either of the following cases: • After the end field of the read command

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• When writing a 1 to the Continue Request field in the Protocol Control register to restart a read transfer This field is cleared in either of the following cases: • When the end field of the last data block is sent from the SD bus to uSDHC. • When the Read Wait state is stopped by a Suspend command and the DATA2 line is released. The uSDHC module waits at the next block gap by driving Read Wait at the start of the interrupt cycle. If the Read Wait signal is already driven (data buffer cannot receive data), uSDHC can wait for a current block gap by continuing to drive the Read Wait signal. It is necessary to support Read Wait to use the suspend / resume function. This field remains 1 during Read Wait. In the case of write transactions: This status indicates that a write transfer is executing on the SD bus. Changes in this value from 1 to 0 generate a Transfer Complete interrupt in the Interrupt Status register. This field is set in either of the following cases: • After the end field of the write command • When writing to 1 to the Continue Request field in the Protocol Control register to continue a write transfer This field is cleared in either of the following cases: • When the SD card releases Write Busy of the last data block, uSDHC also detects if the output is not busy. If the SD card does not drive the busy signal after the CRC status is received, uSDHC assumes the card drive "Not Busy". • When the SD card releases write busy, prior to waiting for write transfer, and because of a Stop At Block Gap Request. In the case of command with busy pending: This status indicates that a busy state follows the command and the data line is in use. This field is cleared when the DATA0 line is released. 0b - DATA line inactive 1b - DATA line active
1 CDIHB	Command Inhibit Data (DATA) This status field is generated if either the DAT Line Active or the Read Transfer Active is set to 1. If this field is 0, it indicates that uSDHC can issue the next SD / eMMC Command. Commands with a busy signal belong to Command Inhibit (DATA) (for example. R1b, R5b type). Changing from 1 to 0 generates a Transfer Complete interrupt in the Interrupt Status register. **NOTE** The SD host driver can save registers for a suspend transaction after this field has changed from 1 to 0. 0b - Can issue command that uses the DATA line 1b - Cannot issue command that uses the DATA line

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 CIHB	Command inhibit (CMD) If this status bit is 0, it indicates that the CMD line is not in use and uSDHC can issue a SD / eMMC command using the CMD line. This field is set also immediately after the Transfer Type register is written. This field is cleared when the command response is received. Even if the Command Inhibit (DATA) is set to 1, commands using only the CMD line can be issued if this field is 0. Changing from 1 to 0 generates a Command Complete interrupt in the Interrupt Status register. If uSDHC cannot issue the command because of a command conflict error (see Command CRC Error) or because of a Command Not Issued By Auto CMD12 Error, this field remains 1 and the Command Complete is not set. The Status of issuing an auto CMD12 does not show on this field. 0b - Can issue command using only CMD line 1b - Cannot issue command

33.6.1.12 Protocol Control (PROT_CTRL)

Offset

Register	Offset
PROT_CTRL	28h

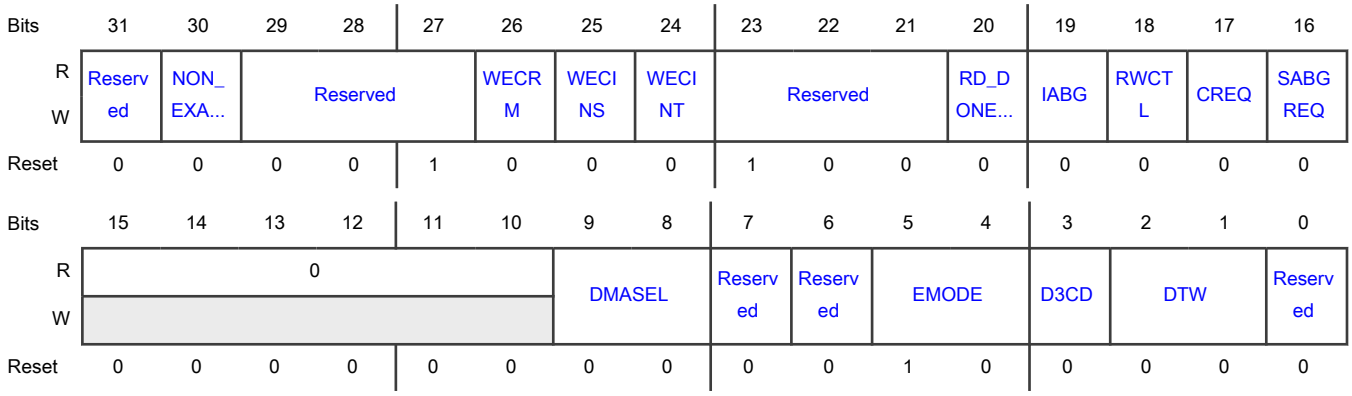
Function

This register controls three cases to restart the transfer after stop at the block gap. Which case is appropriate depends on whether uSDHC issues a Suspend command or the SD card accepts the Suspend command.

- If the host driver does not issue a Suspend command, the Continue request is used to restart the transfer.
- If the host driver issues a Suspend command and the SD card accepts it, a Resume command is used to restart the transfer.
- If the host driver issues a Suspend command and the SD card does not accept it, the Continue request is used to restart the transfer.

Any time stop at block gap request stops the data transfer, the host driver waits for a Transfer Complete (in the Interrupt Status register), before attempting to restart the transfer. When restarting the data transfer by Continue Request, the host driver clears the Stop At Block Gap Request before or simultaneously.

Diagram



Fields

Field	Function
31 —	Reserved Always write as 0
30 NON_EXACT_BLK_RD	Non-exact block read Current block read is non-exact block read. It is only used for SDIO. 0b - The block read is exact block read. Host driver does not need to issue abort command to terminate this multi-block read. 1b - The block read is non-exact block read. Host driver needs to issue abort command to terminate this multi-block read.
29-27 —	Reserved
26 WECRM	Wakeup event enable on SD card removal This field enables a wakeup event, via a card removal, in the Interrupt Status register. FN_WUS (Wakeup Support) in CIS does not affect this field. When this field is set, the Card Removal Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Removal Status and uSDHC interrupt. 0b - Disables wakeup event enable on SD card removal 1b - Enables wakeup event enable on SD card removal
25 WECINS	Wakeup event enable on SD card insertion This field enables a wakeup event, via a card insertion, in the Interrupt Status register. FN_WUS (Wakeup Support) in CIS does not affect this field. When this field is set, the Card Insertion Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Insertion Status and uSDHC interrupt. 0b - Disable wakeup event enable on SD card insertion 1b - Enable wakeup event enable on SD card insertion

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 WECINT	Wakeup event enable on card interrupt This field enables a wakeup event, via a card interrupt, in the Interrupt Status register. This field can be set to 1 if FN_WUS (Wakeup Support) in CIS is set to 1. When this field is set, the Card Interrupt Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Interrupt Status and uSDHC interrupt. 0b - Disables wakeup event enable on card interrupt 1b - Enables wakeup event enable on card interrupt
23-21 —	Reserved Always write as 3'b100
20 RD_DONE_NO_8CLK	Read performed number 8 clock According to the SD/eMMC spec, for read data transaction, 8 clocks are needed after the end field of the last data block. So, by default(RD_DONE_NO_8CLK=0), eight clocks are active after the end field of the last read data transaction. However, these 8 clocks should not be active if user wants to use stop at block gap (include the auto stop at block gap in boot mode) feature for read and the RWCTL field (bit18) is not enabled. In this case, software should set RD_DONE_NO_8CLK to avoid these 8 clocks. Otherwise, the device might send extra data to uSDHC while uSDHC ignores these data. In a summary, this field should be set only if the use case needs to use stop at block gap feature while the device can't support the read wait feature.
19 IABG	Interrupt at block gap This field is valid only in 4-bit mode, of SDIO, and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. Setting to 0 disables interrupt detection during a multiple block transfer. If SDIO cannot signal an interrupt during a multiple block transfer, this field should be set to 0 to avoid an inadvertent interrupt. When the host driver detects an SDIO insertion, it sets this field according to the CCCR of the card. 0b - Disables interrupt at block gap 1b - Enables interrupt at block gap
18 RWCTL	Read wait control The read wait function provided by this field is optional for SDIO. If the card supports read wait, set this field to enable use of the read wait protocol to stop read data using the DATA2 line. Otherwise, uSDHC has to stop the SD clock to hold read data, which restricts commands generation. When the host driver detects an SDIO insertion, it sets this field according to the CCCR of the card. If the card does not support read wait, this field should never be set to 1; otherwise, DATA line conflicts might occur. If this field is set to 0, stop at block gap during read operation is also supported, but uSDHC stops the SD clock to pause reading operation. 0b - Disables read wait control and stop SD clock at block gap when SABGREQ field is set 1b - Enables read wait control and assert read wait without stopping SD clock at block gap when SABGREQ field is set
17	Continue request

Table continues on the next page...

Table continued from the previous page...

Field	Function
CREQ	This field is used to restart a transaction which was stopped using the stop at block gap request. When a suspend operation is not accepted by the card, it is also by setting this field to restart the paused transfer. To cancel stop at the block gap, set stop at block gap request to 0 and set this field to 1 to restart the transfer. The uSDHC module automatically clears this field, therefore it is not necessary for the host driver to set this field to 0. If both stop at block gap request and this field are 1, the continue request is ignored. 0b - No effect 1b - Restart
16 SABGREQ	Stop at block gap request This field is used to stop executing a transaction at the next block gap for both DMA and non-DMA transfers. Until the transfer complete is set to 1, indicating a transfer completion, the host driver leaves this field set to 1. Clearing both the stop at block gap request and continue request does not cause the transaction to restart. Read Wait is used to stop the read transaction at the block gap. The uSDHC module supports the stop at block gap request for write transfers, but for read transfers it requires that SDIO support read wait. Therefore, the host driver does not set this field during read transfers unless SDIO supports Read Wait and has set the read wait control to 1; otherwise, uSDHC stops the SD bus clock to pause the read operation during block gap. In the case of write transfers in which the host driver writes data to the Data Port register, the host driver sets this field after all block data is written. If this field is set to 1, the host driver does not write data to the Data Port register after a block is sent. Once this field is set, the host driver does not clear this field before the Transfer Complete field in Interrupt Status register is set, otherwise uSDHC's behavior is undefined. This field effects read transfer active, write transfer active, DATA Line Active and Command Inhibit (DATA) in the Present State register. 0b - Transfer 1b - Stop
15-10 —	Reserved
9-8 DMASEL	DMA select This field is valid while DMA (SDMA or ADMA) is enabled and selects the DMA operation. 00b - No DMA or simple DMA is selected. 01b - ADMA1 is selected. 10b - ADMA2 is selected. 11b - Reserved
7 —	Reserved
6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-4 EMODE	Endian mode This field supports all three endian modes in data transfer. See Data buffer for more details. 00b - Big endian mode 01b - Half word big endian mode 10b - Little endian mode 11b - Reserved
3 D3CD	DATA3 as card detection pin If this field is set, DATA3 should be pulled down to act as a card detection pin. Be cautious when using this feature, because DATA3 is also a chip-select for the SPI mode. A pull-down on this pin and CMD0 might set the card into the SPI mode, which uSDHC does not support. 0b - DATA3 does not monitor card insertion 1b - DATA3 as card detection pin
2-1 DTW	Data transfer width This field selects the data width of the SD bus for a data transfer. The host driver sets it to match the data width of the card. Possible data transfer width is 1-bit, 4-bits or 8-bits. 00b - 1-bit mode 01b - 4-bit mode 10b - 8-bit mode 11b - Reserved
0 —	Reserved

33.6.1.13 System Control (SYS_CTRL)

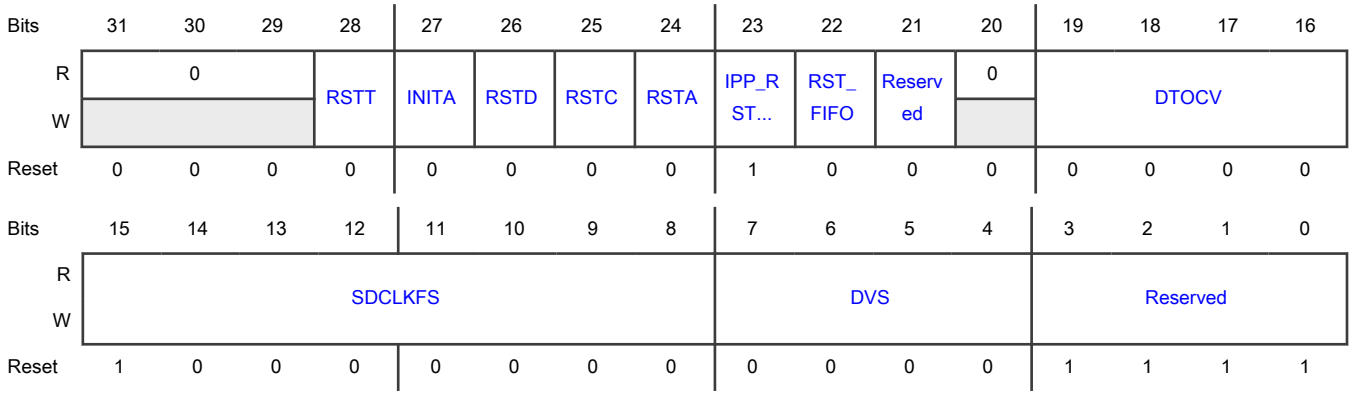
Offset

Register	Offset
SYS_CTRL	2Ch

Function

This register provides control of the system. See detail in the field description.

Diagram



Fields

Field	Function
31-29 —	Reserved
28 RSTT	Reset tuning When set this field to 1, it resets tuning circuit. After tuning circuits are reset, bit value is 0. Clearing execute_tuning field in AUTOCMD12_ERR_STATUS also sets this field to 1 to reset tuning circuit
27 INITA	Initialization active When this field is set, 80 SD-clocks are sent to the card. After the 80 clocks are sent, this field is self cleared. This field is very useful during the card power-up period when 74 SD-clocks are needed and the clock auto gating feature is enabled. Writing 1 to this bit when this field is already 1 has no effect. Writing 0 to this field at any time has no effect. When either of the CIHB and CDIHB fields in the Present State register are set, writing 1 to this field is ignored (that is, when command line or data lines are active, write to this field is not allowed). On the other-hand, when this field is set, that is, during initialization active period, it is allowed to issue command, and the command bit stream appears on the CMD pad after all 80 clock cycles are done. So, when this command ends, the driver can make sure the 80 clock cycles are sent out. This is very useful when the driver needs to send 80 cycles to the card and does not want to wait till this field is self cleared.
26 RSTD	Software reset for data line Only part of the data circuit is reset. DMA circuit is also reset. After this field is set, the software waits for self-clear. The following registers and bits are cleared by this field: • Data Port register — Buffer is cleared and initialized • Present State register — Buffer read enable — Buffer write enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	— Read transfer active — Write transfer active — DATA line active — Command Inhibit (DATA) • Protocol Control register — Continue request • Interrupt Status register — Buffer read ready — Buffer write ready — DMA interrupt — Block gap event — Transfer complete **NOTE** When reset, the software must make sure there is no incomplete data transferring. If there is data transfer going on, the software needs to wait TC or DC INT_STATUS register is set. 0b - No reset 1b - Reset
25 RSTC	Software reset for CMD line Only part of the command circuit is reset. After this field is set, the software waits for self-clear. The following registers and bits are cleared by this field: • Present State Register — Command Inhibit (CMD) • Interrupt Status Register — Command Complete 0b - No reset 1b - Reset
24 RSTA	Software reset for all This reset affects the entire host controller except for the card detection circuit. RSTA resets all the registers that can be reset by RSTC/RSTD. During its initialization, the host driver is set this field to 1 to reset uSDHC. The uSDHC module resets this field to 0 when the capabilities registers are valid and the host driver can read them. Additional use of Software Reset For All does not affect the value of the capabilities registers. After this field is set, it is recommended that the host driver reset the external card and re-initialize it. After this field is set, the software should wait for self-clear. In tuning process, after every CMD19 is finished, this field is set to retest uSDHC.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE When reset, the software must make sure there is no incomplete data transferring. If there is data transfer going on, the software needs to wait TC or DC INT_STATUS register is set. 0b - No reset 1b - Reset
23 IPP_RST_N	Hardware reset This field's value is output to card through pad directly to hardware reset pin of the card if the card supports this feature.
22 RST_FIFO	Reset the async FIFO Reset the async FIFO between card interface and the internal logic. After this field is set, the software waits for self-clear.
21 —	Reserved
20 —	Reserved
19-16 DTOCV	Data timeout counter value This value determines the interval by which DAT line timeouts are detected. See the Data Timeout Error field in the Interrupt Status register for information on factors that dictate time-out generation. Time-out clock frequency is generated by dividing the base clock SDCLK value by this value. The host driver can clear the Data Timeout Error Status Enable (in the Interrupt Status Enable register) to prevent inadvertent time-out events. 0000b - SDCLK x 2^{32} 0001b - SDCLK x 2^{33} 0010b - SDCLK x 2^{18} 0011b - SDCLK x 2^{19} 1101b - SDCLK x 2^{29}, recommend to use for supported speed modes except HS200, HS400, SDR104 mode 1110b - SDCLK x 2^{30}, recommend to use for HS200 and SDR104 mode 1111b - SDCLK x 2^{31}, recommend to use for HS400 mode
15-8 SDCLKFS	SDCLK frequency select This field is used to select the frequency of the SDCLK pin. The frequency is not programmed directly, rather this field holds the prescaler (of this register) and divisor (next register) of the Base Clock Frequency register. In Single Data Rate mode (DDR_EN field of MIXERCTRL is '0')

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Only the following settings are allowed: 80h) Base clock divided by 256 40h) Base clock divided by 128 20h) Base clock divided by 64 10h) Base clock divided by 32 08h) Base clock divided by 16 04h) Base clock divided by 8 02h) Base clock divided by 4 01h) Base clock divided by 2 00h) Base clock divided by 1 While in Dual Data Rate mode (DDR_EN field of MIXERCTRL is '1') Only the following settings are allowed: 80h) Base clock divided by 512 40h) Base clock divided by 256 20h) Base clock divided by 128 10h) Base clock divided by 64 08h) Base clock divided by 32 04h) Base clock divided by 16 02h) Base clock divided by 8 01h) Base clock divided by 4 00h) Base clock divided by 2 When the software changes the DDR_EN field, SDCLKFS might need to be changed also. In Single Data Rate mode, setting 00h bypasses the frequency prescaler of the SD clock. Multiple bits must not be set, or the behavior of this prescaler is undefined. The two default divider values can be calculated by the frequency of ipg_perclk and the following Divisor bits. The frequency of SDCLK is set by the following formula: Clock Frequency = (Base Clock) / (prescaler x divisor) For example, in Single Data Rate mode, if the Base Clock Frequency is 96 MHz, and the target frequency is 25 MHz, then choosing the prescaler value of 01h and divisor value of 1h yields 24 MHz, which is the nearest frequency less than or equal to the target. Similarly, to approach a clock value of 400 kHz, the prescaler value of 08h and divisor value of eh yields the exact clock value of 400 kHz. The reset value of this field is 80h, so if the input Base Clock (ipg_perclk) is about 96 MHz, the default SD clock after reset is 375 kHz. Before changing clock divisor value (SDCLKFS or DVS), host driver should make sure the SDSTB field is high.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If setting SDCLKFS and DVS can generate the same clock frequency,(for example, in SDR mode, SDCLKFS = 01h is same as DVS = 01h.), SDCLKFS is highly recommended. For HS400 mode, SDCLKFS must be used so that REF_CLK, generated for STROBE DLL, is at the same frequency as STROBE clock. For HS200 mode, if DLL is to be used, make sure SDCLKFS is used so that REF_CLK, generated for PARTS DLL, is at the same frequency as SD_CLK. For SDR104 mode, if DLL is to be used, make sure SDCLKFS is used so that REF_CLK, generated for PARTS DLL, is at the same frequency as SD_CLK.
7-4 DVS	Divisor This field is used to provide a more exact divisor to generate the desired SD clock frequency. Note the divider can even support odd divisors without deterioration of duty cycle. Before changing clock divisor value (SDCLKFS or DVS), Host driver should make sure the SDSTB field is high. The settings are as follows: 0000b - Divide-by-1 0001b - Divide-by-2 1110b - Divide-by-15 1111b - Divide-by-16
3-0 —	Reserved Always write as 1.

33.6.1.14 Interrupt Status (INT_STATUS)

Offset

Register	Offset
INT_STATUS	30h

Function

An interrupt is generated when the normal interrupt signal enable is enabled and at least one of the status fields is set to 1. For all fields, writing 1 to a bit clears it; writing to 0 keeps the bit unchanged. More than one status can be cleared with a single register write. For card interrupt, before writing 1 to clear, it is required that the card stops asserting the interrupt, meaning that when the card driver services the interrupt condition; otherwise, the CINT field is asserted again.

The table below shows the relationship between the command timeout error and the command complete.

Table 253. uSDHC status for command timeout error/command complete bit combinations

Command complete	Command timeout error	Meaning of the status
0	0	X
X	1	Response not received within 64 SDCLK cycles
1	0	Response received

The table below shows the relationship between the transfer complete and the data timeout error.

Table 254. uSDHC status for data timeout error/transfer complete bit combinations

Transfer complete	Data timeout error	Meaning of the status
0	0	X
0	1	Timeout occurred during transfer
1	X	Data transfer complete

The table below shows the relationship between the command CRC error and command timeout error.

Table 255. uSDHC status for command CRC error/command timeout error bit combinations

Command CRC error	Command timeout error	Meaning of the status
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	Reserved		DMAE	0	TNE	0	AC12E	0	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE
W		W1C		W1C		W1C		W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERR_INT...	CQI	TP	RTE	0	Reserved		CINT	CRM	CINS	BRR	BWR	DINT	BGE	TC	CC
W		W1C	W1C	W1C		W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-29 —	Reserved
28 DMAE	DMA error Occurs when an Internal DMA transfer has failed. This field is set to 1, when some error occurs in the data transfer. This error can be caused by either simple DMA or ADMA, depending on which DMA is in use. The value in DMA System Address register is the next fetch address where the error occurs. Because any error corrupts the whole data block, the host driver restarts the transfer from the corrupted block boundary. The address of the block boundary can be calculated either from the current DS_ADDR value or from the remaining number of blocks and the block size. 0b - No error 1b - Error
27 —	Reserved
26 TNE	Tuning error: (only for SD3.0 SDR104 mode and eMMC HS200 mode) This field is set when an unrecoverable error is detected in a tuning circuit. By detecting Tuning Error, host driver needs to abort a command executing and perform tuning. This field is relevant in combination with the bus error only and must be cleared prior to all the write and read transactions.
25 —	Reserved
24 AC12E	Auto CMD12 error Occurs when detecting that one of the fields in the Auto CMD12 Error Status register has changed from 0 to 1. This field is set to 1, not only when the errors in Auto CMD12 occur, but also, when the Auto CMD12 is not executed due to the previous command error. 0b - No error 1b - Error
23 —	Reserved
22 DEBE	Data end bit error Occurs either when detecting 0 at the end field position of read data that uses the DATA line, or at the end field position of the CRC. This field is not asserted in tuning process.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No error 1b - Error
21 DCE	Data CRC error Occurs when detecting a CRC error when transferring read data that uses the DATA line, or when detecting the Write CRC status having a value other than 010. This field is not asserted in tuning process. 0b - No error 1b - Error
20 DTOE	Data timeout error Occurs when detecting one of following time-out conditions. • Busy time-out for R1b, R5b type • Busy time-out after Write CRC status • Read Data time-out. This field is not asserted in tuning process. 0b - No error 1b - Time out
19 CIE	Command index error Occurs if a command index error occurs in the command response. This field is not asserted in tuning process. 0b - No error 1b - Error
18 CEBE	Command end bit error Occurs when detecting that the end field of a command response is 0. This field is not asserted in tuning process. 0b - No error 1b - End bit error generated
17 CCE	Command CRC error Command CRC Error is generated in two cases. • If a response is returned and the Command Timeout Error is set to 0 (indicating no time-out), this field is set when detecting a CRC error in the command response. • The uSDHC module detects a CMD line conflict by monitoring the CMD line when a command is issued. If uSDHC drives the CMD line to 1, but detects 0 on the CMD line at the next SDCLK edge, then uSDHC aborts the command (Stop driving CMD line) and set this bit to 1. The Command Timeout Error should also be set to 1 to distinguish CMD line conflict.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is not asserted in tuning process. 0b - No error 1b - CRC error generated
16 CTOE	Command timeout error Occurs only if no response is returned within 64 SDCLK cycles from the end field of the command. If uSDHC detects a CMD line conflict, in which case a Command CRC Error is also be set (as shown in Interrupt Status (INT_STATUS)), this field is set without waiting for 64 SDCLK cycles. This is because the command is aborted by uSDHC. This field is not asserted in tuning process. 0b - No error 1b - Time out
15 ERR_INT_STATUS	Error Interrupt Status This bit is set when any of the error status bit DMAE, AC12E, DEBE, DCE, DTOE, CIE, CEBE, CCE and CTOE is set.
14 CQI	Command queuing interrupt This interrupt is asserted when at least one of the bits in CQIS register is set. This interrupt is cleared only by clearing the source interrupt in CQIS register.
13 TP	Tuning pass:(only for SD3.0 SDR104 mode and eMMC HS200 mode) Current CMD19 transfer is done successfully. That is, current sampling point is correct.
12 RTE	Re-tuning event: (only for SD3.0 SDR104 mode and eMMC HS200 mode) This status is set if Re-Tuning Request in the Present State register changes from 0 to 1. host controller requests host driver to perform re-tuning for next data transfer. Current data transfer (not large block count) can be completed without re-tuning. 0b - Re-tuning is not required. 1b - Re-tuning should be performed.
11 —	Reserved
10-9 —	Reserved
8 CINT	Card interrupt This status field is set when an interrupt signal is detected from the external card. In 1-bit mode, uSDHC detects the Card Interrupt without the SD clock to support wakeup. In 4-bit mode, the card interrupt signal is sampled during the interrupt cycle, so the interrupt from card can only be sampled during interrupt cycle, introducing some delay between the interrupt signal from SDIO and the interrupt to the host system. Writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	this field to 1 can clear this field, but as the interrupt source from SDIO does not clear, this field is set again. to clear this field, it is required to reset the interrupt source from the external card followed by a writing 1 to this field. When this status has been set, and the host driver needs to service this interrupt, the Card Interrupt Signal Enable in the Interrupt Signal Enable register should be 0 to stop driving the interrupt signal to the host system. After completion of the card interrupt service (It should reset the interrupt sources in SDIO and the interrupt signal might not be asserted), write 1 to clear this field, set the Card Interrupt Signal Enable to 1, and start sampling the interrupt signal again. 0b - No card interrupt 1b - Generate card interrupt
7 CRM	Card removal This status field is set if the Card Inserted field in the Present State register changes from 1 to 0. When the host driver writes this field to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state might possibly be changed when the host driver clears this field and the interrupt event might not be generated. When this field is cleared, it is set again if no card is inserted. to leave it cleared, clear the Card Removal Status Enable field in Interrupt Status Enable register. 0b - Card state unstable or inserted 1b - Card removed
6 CINS	Card insertion This status field is set if the Card Inserted field in the Present State register changes from 0 to 1. When the host driver writes this field to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state might possibly be changed when the host driver clears this field and the interrupt event might not be generated. When this field is cleared, it is set again if a card is inserted. to leave it cleared, clear the Card Inserted Status Enable field in Interrupt Status Enable register. 0b - Card state unstable or removed 1b - Card inserted
5 BRR	Buffer read ready This status field is set if the Buffer Read Enable field, in the Present State register, changes from 0 to 1. See the Buffer Read Enable field in the Present State register for additional information. This field indicates that cmd19 is finished in tuning process. 0b - Not ready to read buffer 1b - Ready to read buffer
4 BWR	Buffer write ready This status field is set if the Buffer Write Enable field, in the Present State register, changes from 0 to 1. See the Buffer Write Enable field in the Present State register for additional information. 0b - Not ready to write buffer 1b - Ready to write buffer

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 DINT	DMA interrupt Occurs only when the internal DMA finishes the data transfer successfully. Whenever errors occur during data transfer, this field does not be set. Instead, the DMAE field is set. Either Simple DMA or ADMA finishes data transferring, this field is set. 0b - No DMA interrupt 1b - DMA interrupt is generated.
2 BGE	Block gap event If the Stop At Block Gap Request field in the Protocol Control register is set, this field is set when a read or write transaction is stopped at a block gap. If Stop At Block Gap Request is not set to 1, this field is not set to 1. In the case of a Read Transaction: This field is set at the falling edge of the DATA Line Active Status (When the transaction is stopped at SD bus timing). The Read Wait must be supported to use this function. In the case of Write Transaction: This field is set at the falling edge of Write Transfer Active Status (After getting CRC status at SD Bus timing). 0b - No block gap event 1b - Transaction stopped at block gap
1 TC	Transfer complete This field is set when a read or write transfer is completed. In the case of a Read Transaction: This field is set at the falling edge of the Read Transfer Active Status. There are two cases in which this interrupt is generated. The first is when a data transfer is completed as specified by the data length (after the last data has been read to the host system). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request field in the Protocol Control register (after valid data has been read to the host system). In the case of a Write Transaction: This field is set at the falling edge of the DATA Line Active Status. There are two cases in which this interrupt is generated. The first is when the last data is written to the SD card as specified by the data length and the busy signal is released. The second is when data transfers are stopped at the block gap, by setting the Stop At Block Gap Request field in the Protocol Control register, and the data transfers are completed. (after valid data is written to the SD card and the busy signal released). In the case of a command with busy, this field is set when busy is deasserted. This field is not asserted in tuning process. 0b - Transfer does not complete 1b - Transfer complete
0 CC	Command complete This field is set when you receive the end field of the command response (except auto CMD12). See the Command Inhibit (CMD) in the Present State register. This field is not asserted in tuning process. 0b - Command not complete

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Command complete

33.6.1.15 Interrupt Status Enable (INT_STATUS_EN)

Offset

Register	Offset
INT_STATUS_EN	34h

Function

Setting any bit in this register to 1 enables the corresponding interrupt status be requested to the system. If any bit is set to 0, the corresponding interrupt request gets blocked.

- Depending on IABG field setting, uSDHC might be programmed to sample the card interrupt signal during the interrupt period and hold its value in the flip-flop. There are some delays on the Card Interrupt, asserted from the card, to the time the host system is informed.
- To detect a CMD line conflict, the host driver must set both Command Timeout Error Status Enable and Command CRC Error Status Enable to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	Reserved		DMAESEN	0	TNESEN	0	AC12ESEN	0	DEBESEN	DCESEN	DTOESEN	CIESEN	CEBESEN	CCESEN	CTOESEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CQISEN	TPSEN	RTESEN	0	Reserved		CINTSEN	CRMSEN	CINSSEN	BRRSEN	BWRSEN	DINTSEN	BGESEN	TCSEN	CCSEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-29 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 DMAESEN	DMA error status enable 0b - Masked 1b - Enabled
27 —	Reserved
26 TNESEN	Tuning error status enable 0b - Masked 1b - Enabled
25 —	Reserved
24 AC12ESEN	Auto CMD12 error status enable 0b - Masked 1b - Enabled
23 —	Reserved
22 DEBESEN	Data end bit error status enable 0b - Masked 1b - Enabled
21 DCESEN	Data CRC error status enable 0b - Masked 1b - Enabled
20 DTOESEN	Data timeout error status enable 0b - Masked 1b - Enabled
19 CIESEN	Command index error status enable 0b - Masked 1b - Enabled
18 CEBESEN	Command end bit error status enable 0b - Masked 1b - Enabled
17	Command CRC error status enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
CCESEN	0b - Masked 1b - Enabled
16 CTOESEN	Command timeout error status enable 0b - Masked 1b - Enabled
15 —	Reserved
14 CQISEN	Command queuing status enable 0b - Masked 1b - Enabled
13 TPSEN	Tuning pass status enable 0b - Masked 1b - Enabled
12 RTESEN	Re-tuning event status enable 0b - Masked 1b - Enabled
11 —	Reserved
10-9 —	Reserved
8 CINTSEN	Card interrupt status enable If this field is set to 0, uSDHC clears the interrupt request to the system. The Card Interrupt detection is stopped when this field is cleared and restarted when this field is set to 1. The host driver should clear the Card Interrupt Status Enable before servicing the Card Interrupt and should set this field again after all interrupt requests from the card are cleared to prevent inadvertent interrupts. 0b - Masked 1b - Enabled
7 CRMSEN	Card removal status enable 0b - Masked 1b - Enabled
6 CINSSEN	Card insertion status enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Masked 1b - Enabled
5 BRRSEN	Buffer read ready status enable 0b - Masked 1b - Enabled
4 BWRSEN	Buffer write ready status enable 0b - Masked 1b - Enabled
3 DINTSEN	DMA interrupt status enable 0b - Masked 1b - Enabled
2 BGESEN	Block gap event status enable 0b - Masked 1b - Enabled
1 TCSEN	Transfer complete status enable 0b - Masked 1b - Enabled
0 CCSEN	Command complete status enable 0b - Masked 1b - Enabled

33.6.1.16 Interrupt Signal Enable (INT_SIGNAL_EN)

Offset

Register	Offset
INT_SIGNAL_EN	38h

Function

This register is used to select which interrupt status is indicated to the host system as the interrupt. These status fields all share the same interrupt lines. Setting any of these fields to 1 enables interrupt generation. The corresponding Status register field generates an interrupt when the corresponding interrupt signal enable field is set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	Reserved			0	TNEIEN	0	AC12EIEN	0	DEBEIEN	DCEIEN	DTOEIEN	CIEIEN	CEBEIEN	CCEIEN	CTOEIEN
W				DMAEIEN												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CQIIEN	TPIEN	RTEIEN	0	Reserved		CINTIEN	CRMIEN	CINSIEN	BRRIEN	BWRIEN	DINTIEN	BGEIEN	TCIEN	CCIEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-29 —	Reserved
28 DMAEIEN	DMA error interrupt enable 0b - Masked 1b - Enable
27 —	Reserved
26 TNEIEN	Tuning error interrupt enable 0b - Masked 1b - Enabled
25 —	Reserved
24 AC12EIEN	Auto CMD12 error interrupt enable 0b - Masked 1b - Enabled
23 —	Reserved
22 DEBEIEN	Data end bit error interrupt enable 0b - Masked

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
21 DCEIEN	Data CRC error interrupt enable 0b - Masked 1b - Enabled
20 DTOEIEN	Data timeout error interrupt enable 0b - Masked 1b - Enabled
19 CIEIEN	Command index error interrupt enable 0b - Masked 1b - Enabled
18 CEBEIEN	Command end bit error interrupt enable 0b - Masked 1b - Enabled
17 CCEIEN	Command CRC error interrupt enable 0b - Masked 1b - Enabled
16 CTOEIEN	Command timeout error interrupt enable 0b - Masked 1b - Enabled
15 —	Reserved
14 CQIIEN	Command queuing signal enable 0b - Masked 1b - Enabled
13 TPIEN	Tuning pass interrupt enable 0b - Masked 1b - Enabled
12 RTEIEN	Re-tuning event interrupt enable 0b - Masked 1b - Enabled
11	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
10-9 —	Reserved
8 CINTIEN	Card interrupt enable 0b - Masked 1b - Enabled
7 CRMIEN	Card removal interrupt enable 0b - Masked 1b - Enabled
6 CINSIEN	Card insertion interrupt enable 0b - Masked 1b - Enabled
5 BRRIEN	Buffer read ready interrupt enable 0b - Masked 1b - Enabled
4 BWRIEN	Buffer write ready interrupt enable 0b - Masked 1b - Enabled
3 DINTIEN	DMA interrupt enable 0b - Masked 1b - Enabled
2 BGEIEN	Block gap event interrupt enable 0b - Masked 1b - Enabled
1 TCIEN	Transfer complete interrupt enable 0b - Masked 1b - Enabled
0 CCIEN	Command complete interrupt enable 0b - Masked 1b - Enabled

33.6.1.17 Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS)

Offset

Register	Offset
AUTOCMD12_ERR_STATUS	3Ch

Function

When the Auto CMD12 Error Status field in the Status register is set, the host driver checks this register to identify what kind of error the Auto CMD12 / CMD 23 indicated. Auto CMD23 errors are indicated in field 04-01. This register is valid only when the Auto CMD12 Error status field is set.

The table below shows the relationship between the Auto CMGD12 CRC Error and the Auto CMD12 Command Timeout Error.

Table 256. Relationship between command CRC error and command timeout error for auto CMD12

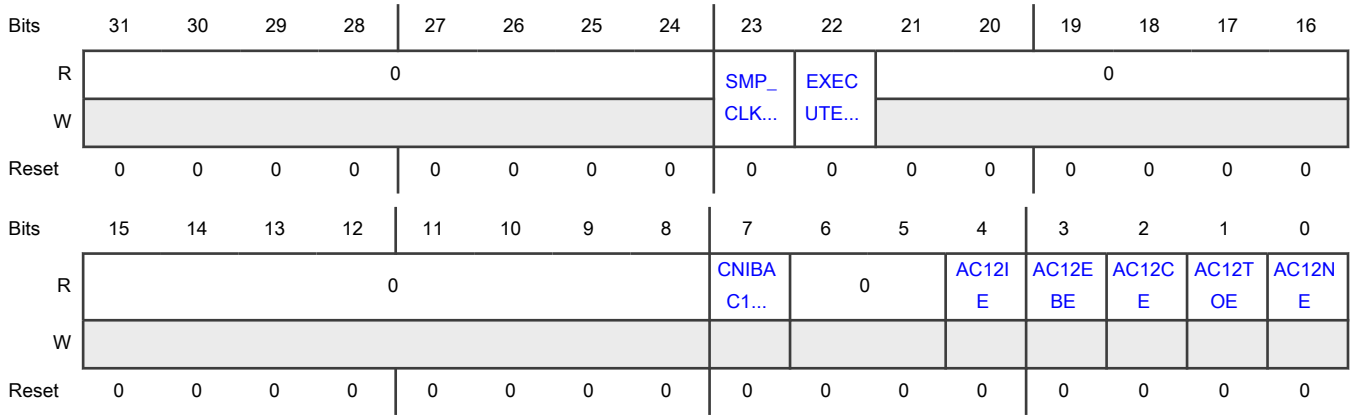
Auto CMD12 CRC error	Auto CMD12 timeout error	Type of error
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

Changes in Auto CMD12 Error Status register can be classified in three scenarios:

- When uSDHC is going to issue an Auto CMD12
  - Set field 0 to 1 if the Auto CMD12 cannot be issued due to an error in the previous command
  - Set field 0 to 0 if the Auto CMD12 is issued
- At the end field of an Auto CMD12 response
  - Check errors correspond to fields 1-4
  - Set fields 1-4 corresponding to detected errors
  - Clear fields 1-4 corresponding to detected errors
- Before reading the Auto CMD12 Error Status field 7
  - Set field 7 to 1 if there is a command that can't be issued
  - Clear field 7 if there is no command to issue

The timing for generating the Auto CMD12 Error and writing to the Command register are asynchronous. After that, field 7 is sampled when the driver is not writing to the Command register. So, it is suggested to read this register only when the AC12E field in Interrupt Status register is set. An Auto CMD12 Error Interrupt is generated when one of the error fields (0-4) is set to 1. The Command Not Issued By Auto CMD12 Error does not generate an interrupt.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 SMP_CLK_SEL	Sample clock select This field is used to select sampling clock to receive CMD and DATA. This field is functional only when TUNING_CTRL[STD_TUNING_EN] is set. Otherwise, this field is reserved. This field is controlled by re-tuning procedure and is valid after the tuning procedure is complete (AUTOCMD12_ERR_STATUS[EXECUTE_TUNING]=0). A value of '1' means that tuning has been completed successfully and a value of '0' implies the tuning has failed. Writing '1' to this field is meaningless and ignored when the tuning has failed. The tuning circuit is reset by writing '0' to this field. This field can be cleared by setting AUTOCMD12_ERR_STATUS[EXECUTE_TUNING]=1. Once the tuning circuit is reset, it takes time to complete tuning sequence. Therefore, host driver should keep this field as 1 to perform the re-tuning sequence in a short time. This field cannot be changed while the Host controller is receiving response or a read data block. 0b - Fixed clock is used to sample data 1b - Tuned clock is used to sample data
22 EXECUTE_TUNING	Execute tuning This field is used to start the tuning procedure. This field is functional only when TUNING_CTRL[STD_TUNING_EN] is set. Otherwise, this field is reserved. This field is set to start tuning procedure and automatically cleared when tuning procedure is completed. The result of tuning is indicated to AUTOCMD12_ERR_STATUS[SMP_CLK_SEL]. 0b - Tuning procedure is aborted 1b - Start tuning procedure
21-8 —	Reserved
7	Command not issued by Auto CMD12 error

Table continues on the next page...

Table continued from the previous page...

Field	Function
CNIBAC12E	Setting this field to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 error (D04-D01) in this register. 0b - No error 1b - Not issued
6-5 —	Reserved
4 AC12IE	Auto CMD12 / 23 index error Occurs if the command index error occurs in response to a command. 0b - No error 1b - Error, the CMD index in response is not CMD12/23
3 AC12EBE	Auto CMD12 / 23 end bit error Occurs when detecting that the end field of command response is 0 which should be 1. 0b - No error 1b - End bit error generated
2 AC12CE	Auto CMD12 / 23 CRC error Occurs when detecting a CRC error in the command response. 0b - No CRC error 1b - CRC error met in Auto CMD12/23 response
1 AC12TOE	Auto CMD12 / 23 timeout error Occurs if no response is returned within 64 SDCLK cycles from the end field of the command. If this field is set to1, the other error status fields (2-4) have no meaning. 0b - No error 1b - Time out
0 AC12NE	Auto CMD12 not executed If memory multiple block data transfer is not started, due to a command error, this field is not set because it is not necessary to issue an Auto CMD12. Setting this field to 1 means uSDHC cannot issue the Auto CMD12 to stop a memory multiple block data transfer due to some error. If this field is set to 1, other error status fields (1-4) have no meaning. 0b - Executed 1b - Not executed

33.6.1.18 Host Controller Capabilities (HOST_CTRL_CAP)

Offset

Register	Offset
HOST_CTRL_CAP	40h

Function

This register provides the host driver with information specific to uSDHC implementation. The value in this register is the power-on-reset value and does not change with a software reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					VS18	VS30	VS33	SRS	DMAS	HSS	ADMAS	0	MBL		
W																
Reset	0	0	0	0	0	1	1	1	1	1	1	1	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		USE_TUN...	Reserved	Reserved				Reserved					DDR50_S...	SDR104_...	SDR50_S...
W																
Reset	1	0	1	1	0	1	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26 VS18	Voltage support 1.8 V This field depends on the host system ability. 0b - 1.8 V not supported 1b - 1.8 V supported
25 VS30	Voltage support 3.0 V This field depends on the host system ability. 0b - 3.0 V not supported 1b - 3.0 V supported
24 VS33	Voltage support 3.3 V This field depends on the host system ability.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - 3.3 V not supported 1b - 3.3 V supported
23 SRS	Suspend / resume support This field indicates whether uSDHC supports Suspend / Resume functionality. If this field is 0, the Suspend and Resume mechanism, as well as the read wait, are not supported, and the host driver does not issue either Suspend or Resume commands. 0b - Not supported 1b - Supported
22 DMAS	DMA support This field indicates whether uSDHC can use the internal DMA to transfer data between system memory and the data buffer directly. 0b - DMA not supported 1b - DMA supported
21 HSS	High speed support This field indicates whether uSDHC supports High Speed mode and the host system can supply a SD clock frequency from 25 MHz to 50 MHz. 0b - High speed not supported 1b - High speed supported
20 ADMAS	ADMA support This field indicates whether uSDHC supports the ADMA feature. 0b - Advanced DMA not supported 1b - Advanced DMA supported
19 —	Reserved
18-16 MBL	Max block length This field indicates the maximum block size that the host driver can read and write to the buffer in uSDHC. The buffer transfers block size without wait cycles. 000b - 512 bytes 001b - 1024 bytes 010b - 2048 bytes 011b - 4096 bytes
15-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 USE_TUNING_SDR50	Use Tuning for SDR50 This field is used to enable tuning for SDR50 mode 0b - SDR50 does not support tuning 1b - SDR50 supports tuning
12 —	Reserved
11-8 —	Reserved
7-3 —	Reserved
2 DDR50_SUPPORT	DDR50 support This field indicates support of DDR50 mode.
1 SDR104_SUPPORT	SDR104 support This field indicates support of SDR104 mode.
0 SDR50_SUPPORT	SDR50 support This field indicates support of SDR50 mode.

33.6.1.19 Watermark Level (WTMK_LVL)

Offset

Register	Offset
WTMK_LVL	44h

Function

This register indicates configurability of write and read watermark levels (FIFO threshold). Their value can range from 1 to 128 words.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				Reserved				WR_WML							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				Reserved				RD_WML							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 —	Reserved
23-16 WR_WML	Write watermark level This field indicates the number of words used as the watermark level (FIFO threshold) in a DMA write operation. Also, the number of words as a sequence of write bursts in back-to-back mode. The maximum legal value for the write watermark level is 128.When using internal DMA, WR_WML can be set any value between 1 and 128. To get max AXI throughput, we recommend to use integer multiple of 16.
15-13 —	Reserved
12-8 —	Reserved
7-0 RD_WML	Read watermark level This field indicates the number of words used as the watermark level (FIFO threshold) in a DMA read operation. Also, the number of words as a sequence of read bursts in back-to-back mode. The maximum legal value for the read water mark level is 128.When using internal DMA mode, RD_WML must be set to the value less than 16. To get max AXI throughput, recommend to use 16.

33.6.1.20 Mixer Control (MIX_CTRL)

Offset

Register	Offset
MIX_CTRL	48h

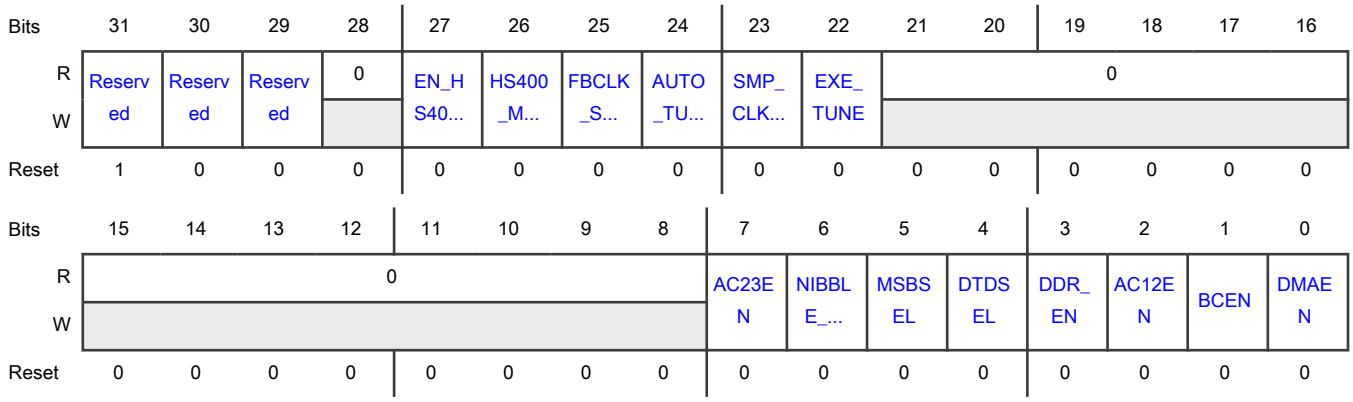
Function

This register is used to DMA and data transfer. To prevent data loss, the software should check if data transfer is active before writing this register. These fields are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN, and DMAEN.

Table 257. Transfer type register setting for various transfer types

Multi/single block select	Block count enable	Block count	Function
0	Do not care	Do not care	Single transfer
1	0	Do not care	Infinite transfer
1	1	Positive number	Multiple transfer
1	1	Zero	No data transfer

Diagram



Fields

Field	Function
31 —	Reserved Always write as 1.
30 —	Reserved Always write as 0.
29 —	Reserved Always write as 0.
28 —	Reserved
27 EN_HS400_MODE	Enable enhance HS400 mode Enhance HS400 enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 HS400_MODE	Enable HS400 mode HS400 Enable
25 FBCLK_SEL	Feedback clock source selection (Only used for SD3.0, SDR104 mode and eMMC HS200 mode) 0b - Feedback clock comes from the loopback CLK 1b - Feedback clock comes from the ipp_card_clk_out
24 AUTO_TUNE_EN	Auto tuning enable (Only used for SD3.0, SDR104 mode and eMMC HS200 mode) 0b - Disable auto tuning 1b - Enable auto tuning
23 SMP_CLK_SEL	Clock selection When TUNING_CTRL[STD_TUNING_EN] is 0, this field is used to select Tuned clock or Fixed clock to sample data / cmd (Only used for SD3.0, SDR104 mode and eMMC HS200 mode) 0b - Fixed clock is used to sample data / cmd 1b - Tuned clock is used to sample data / cmd
22 EXE_TUNE	Execute tuning: (Only used for SD3.0, SDR104 mode and eMMC HS200 mode) When TUNING_CTRL[STD_TUNING_EN] is 0, this field is set to 1 to indicate the host driver is starting tuning procedure. Tuning procedure is aborted by writing 0. 0b - Not tuned or tuning completed 1b - Execute tuning
21-8 —	Reserved
7 AC23EN	Auto CMD23 enable This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. When this field is set to 1, the host controller issues a CMD23 automatically before issuing a command specified in the Command Register.
6 NIBBLE_POS	Nibble position indication This field indicates the nibble position in the DDR 4-bit mode. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. 0- the sequence is 'odd high nibble -> even high nibble -> odd low nibble -> even low nibble'; 1- the sequence is 'odd high nibble -> odd low nibble -> even high nibble -> even low nibble'.
5 MSBSEL	Multi / Single block select This field enables multiple block DATA line data transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. For any other commands, this field can be set to 0. If this field is 0, it is not necessary to set the Block Count register. See Command Transfer Type (CMD_XFR_TYP).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Single block 1b - Multiple blocks
4 DTDSEL	Data transfer direction select This field defines the direction of DATA line data transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. The field is set to 1 by the host driver to transfer data from the SD card to uSDHC and is set to 0 for all other commands. 0b - Write (Host to card) 1b - Read (Card to host)
3 DDR_EN	Dual data rate mode selection This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only.
2 AC12EN	Auto CMD12 enable Multiple block transfers for memory require a CMD12 to stop the transaction. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. When this field is set to 1, uSDHC issues a CMD12 automatically when the last block transfer has completed. The host driver is not set this field to issue commands that do not require CMD12 to stop a multiple block data transfer. In particular, secure commands defined in File Security Specification (see reference list) do not require CMD12. In single block transfer, uSDHC ignores this field no matter it is set or not. 0b - Disable 1b - Enable
1 BCEN	Block count enable This field is used to enable the Block Count register, which is only relevant for multiple block transfers. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. When this field is 0, the internal counter for block is disabled, which is useful in executing an infinite transfer. 0b - Disable 1b - Enable
0 DMAEN	DMA enable This field enables DMA functionality. This field is read/write when VEND_SPEC[CMD_BYTE_EN] is disabled; otherwise, this field is read-only. If this field is set to 1, a DMA operation begins when the host driver sets the DPSEL field of this register. Whether the simple DMA or the advanced DMA is active depends on the DMA Select field of the Protocol Control register. 0b - Disable 1b - Enable

33.6.1.21 Force Event (FORCE_EVENT)

Offset

Register	Offset
FORCE_EVENT	50h

Function

This register is not a physically implemented register. Rather, it is an address at which the Interrupt Status register can be written if the corresponding field of the Interrupt Status Enable Register is set. This register is a write only register and writing 0 to it has no effect. Writing 1 to this register sets the corresponding field of Interrupt Status register. A read from this register always results in 0's. to change the corresponding status fields in the Interrupt Status register, make sure to set IPGEN field in System Control Register so that peripheral clock is always active.

Forcing a card interrupt generates a short pulse on the DATA1 line, and the driver might treat this interrupt as a normal interrupt. The interrupt service routine might skip polling the card interrupt factor as the interrupt is self cleared.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0
W	FEVTCINT			FEVTDMAE		FEVTTNE		FEVTAC1...		FEVTDEBE	FEVTDCE	FEVTDTOE	FEVTCIE	FEVTCEBE	FEVTCCE	FEVTCTOE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	0		0	0	0	0	0
W									FEVTCNI...			FEVTAC1...	FEVTAC1...	FEVTAC1...	FEVTAC1...	FEVTAC1...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 FEVTCINT	Force event card interrupt Writing 1 to this field generates a short low-level pulse on the internal DATA1 line, as if a self-clearing interrupt was received from the external card. If enabled, the CINT field is set and the interrupt service routine might treat this interrupt as a normal interrupt from the external card.
30-29 —	Reserved
28 FEVTDMAE	Force event DMA error Forces the DMAE field of Interrupt Status register to be set
27	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
26 FEVTTNE	Force tuning error Forces the TNE field of Interrupt Status register to be set
25 —	Reserved
24 FEVTAC12E	Force event Auto Command 12 error Forces the AC12E field of Interrupt Status register to be set
23 —	Reserved
22 FEVTDEBE	Force event data end bit error Forces the DEBE field of Interrupt Status register to be set
21 FEVTDCE	Force event data CRC error Forces the DCE field of Interrupt Status register to be set
20 FEVTDTOE	Force event data time out error Force the DTOE field of Interrupt Status register to be set
19 FEVTCIE	Force event command index error Forces the CCE field of Interrupt Status register to be set
18 FEVTCEBE	Force event command end bit error Forces the CEBE field of Interrupt Status register to be set
17 FEVTCCE	Force event command CRC error Forces the CCE field of Interrupt Status register to be set
16 FEVTCTOE	Force event command time out error Forces the CTOE field of Interrupt Status register to be set
15-8 —	Reserved
7 FEVTCNIBAC12E	Force event command not executed by Auto Command 12 error Forces the CNIBAC12E field in the Auto Command12 Error Status register to be set
6-5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4 FEVTAC12IE	Force event Auto Command 12 index error Forces the AC12IE field in the Auto Command12 Error Status register to be set
3 FEVTAC12EBE	Force event Auto Command 12 end bit error Forces the AC12EBE field in the Auto Command12 Error Status register to be set
2 FEVTAC12CE	Force event auto command 12 CRC error Forces the AC12CE field in the Auto Command12 Error Status register to be set
1 FEVTAC12TOE	Force event auto command 12 time out error Forces the AC12TOE field in the Auto Command12 Error Status register to be set
0 FEVTAC12NE	Force event auto command 12 not executed Forces the AC12NE field in the Auto Command12 Error Status register to be set

33.6.1.22 ADMA Error Status (ADMA_ERR_STATUS)

Offset

Register	Offset
ADMA_ERR_STATUS	54h

Function

When an ADMA Error Interrupt has occurred, the ADMA error sates field in this register holds the ADMA state and the ADMA System Address register holds the address around the error descriptor.

For recovering from this error, the host driver requires the ADMA state to identify the error descriptor address as follows:

- ST_STOP: Previous location set in the ADMA System Address register is the error descriptor address.
- ST_FDS: Current location set in the ADMA System Address register is the error descriptor address.
- ST_CADR: This state is never set because it only increments the descriptor pointer and does not generate an ADMA error.
- ST_TFR: Previous location set in the ADMA System Address register is the error descriptor address.

In case of a write operation, the host driver should use the ACMD22 to get the number of the written block, rather than using this information, because unwritten data might exist in the host controller.

The host controller generates the ADMA Error Interrupt when it detects invalid descriptor data (Valid=0) in the ST_FDS state. The host driver can distinguish this error by reading the Valid field of the error descriptor.

Table 258. ADMA error state coding

D01-D00	ADMA error state (when error has occurred)	Contents of ADMA System Address register
00	ST_STOP (Stop DMA)	Holds the address of the next executable descriptor command
01	ST_FDS (Fetch descriptor)	Holds the valid descriptor address
10	ST_CADR (Change address)	No ADMA error is generated
11	ST_TFR (Transfer data)	Holds the address of the next executable descriptor command

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												ADMA DCE	ADMA LME	ADMAES	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3 ADMADCE	ADMA descriptor error This error occurs when invalid descriptor fetched by ADMA. 0b - No error 1b - Error
2 ADMALME	ADMA length mismatch error This error occurs in the following two cases: • While the block count enable is being set, the total data length specified by the descriptor table is different from that specified by the block count and block length. • Total data length cannot be divided by the block length.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No error 1b - Error
1-0 ADMAES	ADMA error state (when ADMA error is occurred) This field indicates the state of the ADMA when an error has occurred during an ADMA data transfer. See ADMA Error Status (ADMA_ERR_STATUS) for more details.

33.6.1.23 ADMA System Address (ADMA_SYS_ADDR)

Offset

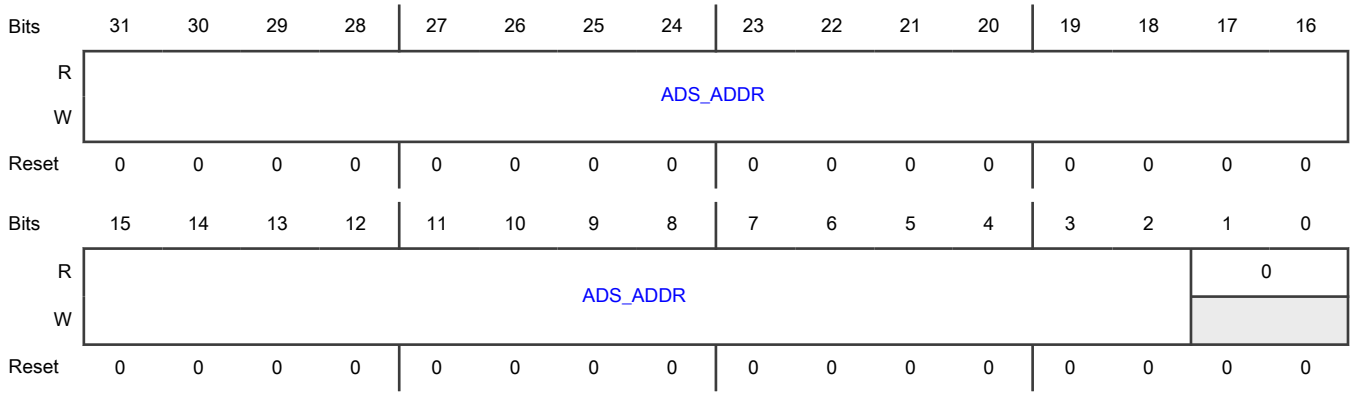
Register	Offset
ADMA_SYS_ADDR	58h

Function

This register holds the word address of the executing command in the Descriptor table. At the start of ADMA, the host driver sets the start address of the Descriptor table. The ADMA engine increments this register address whenever fetching a Descriptor command. When the ADMA is stopped at the Block Gap, this register indicates the address of the next executable Descriptor command. When the ADMA Error Interrupt is generated, this register holds the valid Descriptor address depending on the ADMA state. The lower 2 bits of this register is tied to '0' so the ADMA address is always word aligned.

Because this register supports dynamic address reflecting, when TC field is set, it automatically alters the value of internal address counter, so the software cannot change this register when TC field is set. Such restriction is also listed in Software restrictions.

Diagram



Fields

Field	Function
31-2	ADMA system address

Table continues on the next page...

Table continued from the previous page...

Field	Function
ADS_ADDR	This field contains the physical system memory address used for ADMA transfers.
1-0 —	Reserved

33.6.1.24 DLL (Delay Line) Control (DLL_CTRL)

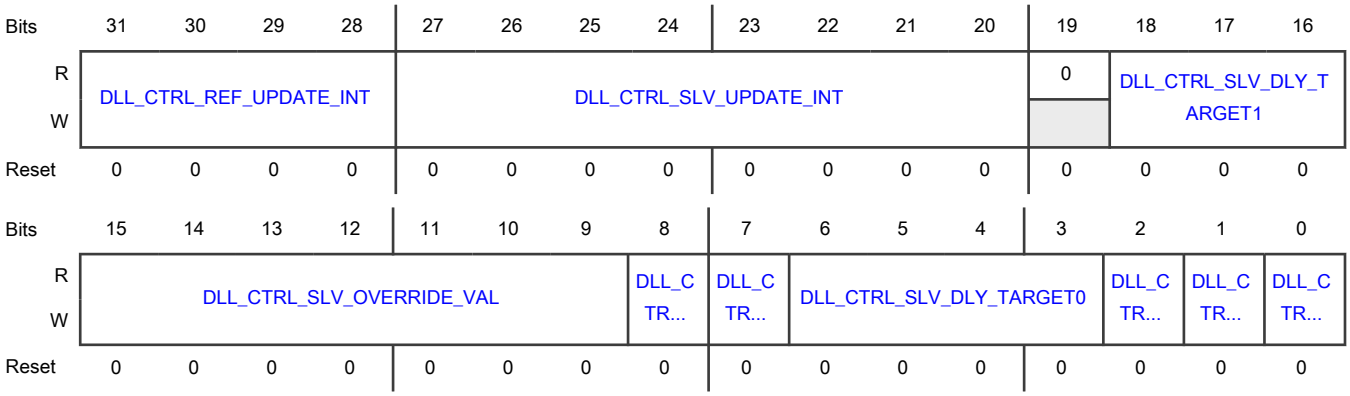
Offset

Register	Offset
DLL_CTRL	60h

Function

This register contains control fields for DLL.

Diagram



Fields

Field	Function
31-28 DLL_CTRL_REF_UPDATE_INT	DLL control loop update interval The interval cycle is (2 + REF_UPDATE_INT) * REF_CLOCK. By default, the DLL control loop updates every two REF_CLOCK cycles. It should be noted that increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that might effect the delay (such as voltage and temperature)
27-20	Slave delay line update interval If default 0 is used, it means 256 cycles of REF_CLOCK. A value of 0x0f results in 15 cycles and so on. Note that software can always cause an update of the slave-delay line using the SLV_FORCE_UPDATE register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
DLL_CTRL_SLV_UPDATE_INT	Note that the slave delay line also updates automatically when the reference DLL transitions to a locked state (from an un-locked state).
19 —	Reserved
18-16 DLL_CTRL_SLV_DLY_TARGET1	DLL slave delay target1 See DLL_CTRL_SLV_DLY_TARGET0 below.
15-9 DLL_CTRL_SLV_OVERRIDE_VAL	DLL slave override val When SLV_OVERRIDE = 1, this field is used to select 1 of 128 physical taps manually. A value of 0 selects tap 1, and a value of 0x7f selects tap 128.
8 DLL_CTRL_SLV_OVERRIDE	DLL slave override Set this field to 1 to Enable manual override for slave delay chain using SLV_OVERRIDE_VAL; to set 0 to disable manual override. This feature does not require the DLL to be enabled using the ENABLE field. In fact to reduce power, if SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE = 0
7 DLL_CTRL_GATE_UPDATE	DLL gate update Set this field to 1 to prevent the DLL from updating (because when clock_in exists, glitches might appear during DLL updates). This field might be used by software if such a condition occurs. Clear the bit to 0 to allow the DLL to update automatically.
6-3 DLL_CTRL_SLV_DLY_TARGET0	DLL slave delay target0 The delay target for uSDHC loopback read clock can be programmed in 1/16th increments of an ref_clock half-period. The delay is (({DLL_CTRL_SLV_DLY_TARGET1,DLL_CTRL_SLV_DLY_TARGET0} +1)* REF_CLOCK / 2) / 16 So the input read-clock can be delayed relative input data from (REF_CLOCK / 2) / 16 to REF_CLOCK * 4. **NOTE** For the restrictions of delay cell implementation, the delay target must be set between REF_CLOCK/16 and REF_CLOCK*2 when REF_CLOCK is running at 200 MHz. When REF_CLOCK frequency is slower than 100 MHz, the maximum delay target might not reach REF_CLOCK*2.
2 DLL_CTRL_SLV_FORCE_UPD	DLL slave delay line Setting this field to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line updates automatically based on the SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered). Be sure to use it when uSDHC is idle. This function might not work when uSDHC is working on data / cmd / response.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DLL_CTRL_RESET	DLL reset Setting this field to 1 force a reset on DLL. This causes the DLL to lose lock and re-calibrate to detect an REF_CLOCK half period phase shift. This signal is used by the DLL as edge-sensitive, so to create a subsequent reset, RESET must be taken low and then asserted again.
0 DLL_CTRL_ENABLE	DLL and delay chain Set this field to 1 to enable the DLL and delay chain; otherwise; set to 0 to bypasses DLL. Note that using the slave delay line override feature with SLV_OVERRIDE and SLV_OVERRIDE VAL, the DLL does not need to be enabled.

33.6.1.25 DLL Status (DLL_STATUS)

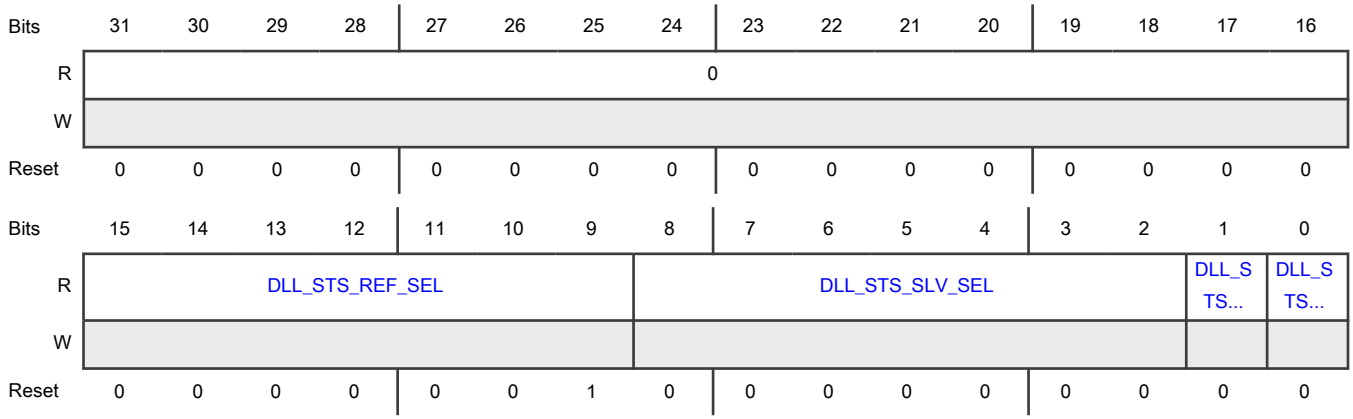
Offset

Register	Offset
DLL_STATUS	64h

Function

This register contains the DLL status information. All fields are read only and reads the same as the power-reset value.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-9	Reference delay line select taps

Table continues on the next page...

Table continued from the previous page...

Field	Function
DLL_STS_REF_SEL	This is encoded by 7 fields for 127 taps.
8-2 DLL_STS_SLV_SEL	Slave delay line select status This is the instant value generated from reference chain. Because the reference chain can only be updated when REF_CLOCK is detected, this value should be the right value to be updated when the reference is locked.
1 DLL_STS_REF_LOCK	Reference DLL lock status This signifies that the DLL has detected and locked to a half-phase ref_clock shift, allowing the slave delay-line to perform programmed clock delays
0 DLL_STS_SLV_LOCK	Slave delay-line lock status This signifies that a valid calibration has been set to the slave-delay line and that the slave-delay line is implementing the programmed delay value

33.6.1.26 CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS)

Offset

Register	Offset
CLK_TUNE_CTRL_STATUS	68h

Function

This register contains the Clock Tuning Control status information. This register is added to support SD3.0 UHS-I SDR104 mode and eMMC HS200 mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRE_ERR	TAP_SEL_PRE							TAP_SEL_OUT				TAP_SEL_POST			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NXT_ERR	DLY_CELL_SET_PRE							DLY_CELL_SET_OUT				DLY_CELL_SET_POST			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PRE_ERR	PRE error PRE error which means the number of delay cells added on the feedback clock is too small. It is valid only when SMP_CLK_SEL of Mix control register (bit23 of 0x48) is enabled.
30-24 TAP_SEL_PRE	TAP_SEL_PRE Reflects the number of delay cells added on the feedback clock between the feedback clock and CLK_PRE. When AUTO_TUNE_EN (bit24 of 0x48) is disabled, TAP_SEL_PRE is always equal to DLY_CELL_SET_PRE. When AUTO_TUNE_EN (bit24 of 0x48) is enabled, TAP_SEL_PRE is updated automatically according to the status of the auto tuning circuit to adjust the sample clock phase.
23-20 TAP_SEL_OUT	Delay cells added on the feedback clock between CLK_PRE and CLK_OUT Reflects the number of delay cells added on the feedback clock between CLK_PRE and CLK_OUT.
19-16 TAP_SEL_POST	Delay cells added on the feedback clock between CLK_OUT and CLK_POST Reflects the number of delay cells added on the feedback clock between CLK_OUT and CLK_POST.
15 NXT_ERR	NXT error NXT error which means the number of delay cells added on the feedback clock is too large. It's valid only when SMP_CLK_SEL of Mix control register (bit23 of 0x48) is enabled.
14-8 DLY_CELL_SET_PRE	delay cells on the feedback clock between the feedback clock and CLK_PRE Set the number of delay cells on the feedback clock between the feedback clock and CLK_PRE.
7-4 DLY_CELL_SET_OUT	Delay cells on the feedback clock between CLK_PRE and CLK_OUT Set the number of delay cells on the feedback clock between CLK_PRE and CLK_OUT.
3-0 DLY_CELL_SET_POST	Delay cells on the feedback clock between CLK_OUT and CLK_POST Set the number of delay cells on the feedback clock between CLK_OUT and CLK_POST.

33.6.1.27 Strobe DLL control (STROBE_DLL_CTRL)

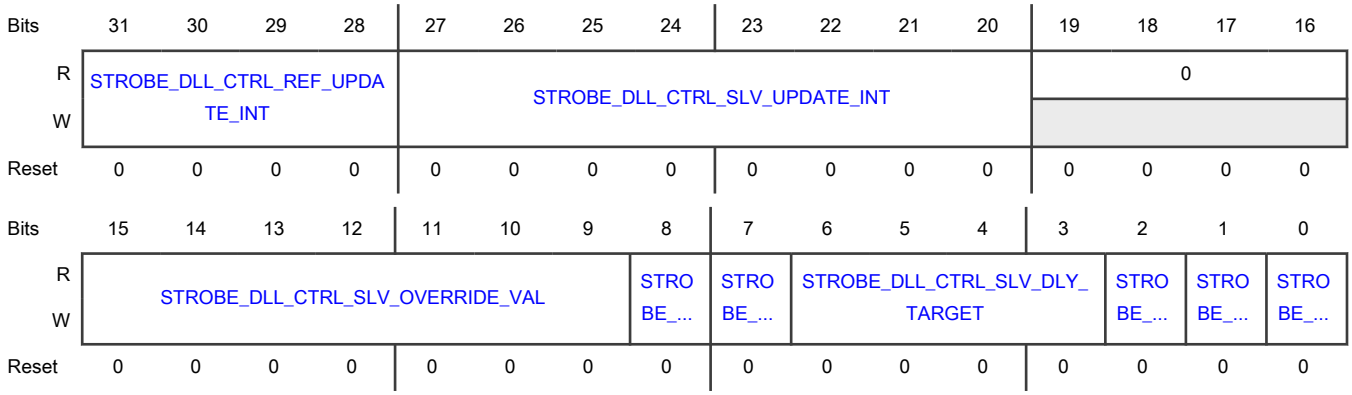
Offset

Register	Offset
STROBE_DLL_CTRL	70h

Function

This register contains the strobe DLL control information.

Diagram



Fields

Field	Function
31-28 STROBE_DLL_CTRL_REF_UPDATE_INT	Strobe DLL control reference update interval The interval cycle is (2 + STROBE_REF_UPDATE_INT) * STROBE_REF_CLOCK. By default, the DLL control loop updates every two STROBE_REF_CLOCK cycles. **NOTE** Increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that might effect the delay (such as voltage and temperature)
27-20 STROBE_DLL_CTRL_SLV_UPDATE_INT	Strobe DLL control slave update interval Slave delay line update interval. If default 0 is used, it means 256 cycles of STROBE_REF_CLOCK. A value of 0x0F results in 15 cycles and so on. **NOTE** software can always cause an update of the slave-delay line using the STROBE_SLV_FORCE_UPDATE register. The slave delay line also updates automatically when the reference DLL transitions to a locked state (from an un-locked state).
19-16 —	This field is reserved. This read-only field is reserved and always has the value 0.
15-9 STROBE_DLL_CTRL_SLV_OVERRIDE_VAL	Strobe DLL control slave Override value When STROBE_SLV_OVERRIDE = 1, this field is used to manually select one of 128 physical taps. A value of 0 selects tap 1, and a value of 0x7F selects tap 128.
8 STROBE_DLL_CTRL_SLV_OVERRIDE	Strobe DLL control slave override Set this field to 1 to Enable manual override for slave delay chain using STROBE_SLV_OVERRIDE_VAL; set this field to 0 to disable manual override. This feature does not require the DLL to be enabled using the ENABLE field. In fact to reduce power, if STROBE_SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE = 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 STROBE_DLL_CTRL_GATE_UPDATE	Strobe DLL control gate update Set this field to 1 to prevent the DLL from updating (because when STROBE_CLOCK_IN exists, glitches might appear during DLL updates). This field can be used by software if such a condition occurs. Clear the field to 0 to allow the DLL to update automatically.
6-3 STROBE_DLL_CTRL_SLV_DLY_TARGET	Strobe DLL Control Slave Delay Target The delay target for uSDHC loopback read clock can be programmed in 1/16th increments of an STROBE_REF_CLOCK half-period. The delay is ((STROBE_DLL_CTRL_SLV_DLY_TARGET + 1) * STROBE_REF_CLOCK / 2) / 16, So, the input read-clock can be delayed relative input data from (STROBE_REF_CLOCK / 2) / 16 to (STROBE_REF_CLOCK *8)/16. **NOTE** For the restrictions of delay cell implementation, the minimum delay target that can be set is REF_CLOCK/16.
2 STROBE_DLL_CTRL_SLV_FORCE_UPD	Strobe DLL control slave force updated Setting this field to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line should automatically update the STROBE_SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if STROBE_SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered). Be sure to use it when uSDHC is idle. This function might not work when uSDHC is working on data / cmd / response.
1 STROBE_DLL_CTRL_RESET	Strobe DLL control reset Setting this field to 1 to force a reset on DLL. This causes the DLL to lose lock and re-calibrate to detect an REF_CLOCK half period phase shift. This signal is used by the DLL as edge-sensitive, to create a subsequent reset, RESET must be taken low and then asserted again.
0 STROBE_DLL_CTRL_ENABLE	Strobe DLL control enable Set this field to 1 to enable the DLL and delay chain; otherwise, set to 0 to bypasses DLL. **NOTE** Using the slave delay line override feature with STROBE_SLV_OVERRIDE and STROBE_SLV_OVERRIDE VAL, the DLL does not need to be enabled.

33.6.1.28 Strobe DLL status (STROBE_DLL_STATUS)

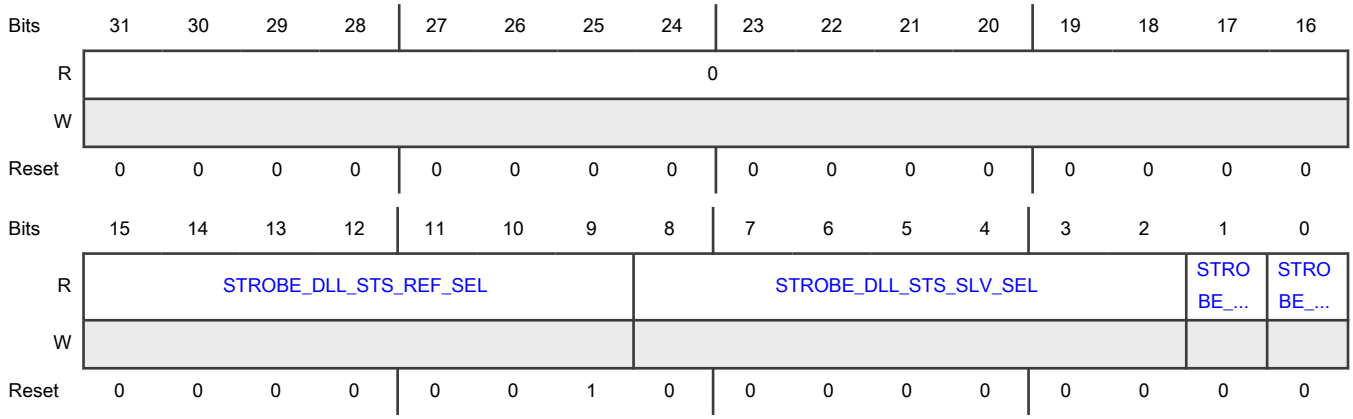
Offset

Register	Offset
STROBE_DLL_STATUS	74h

Function

This register contains the strobe DLL status information. All fields are read only and read the same as the power-reset value.

Diagram



Fields

Field	Function
31-16 —	This field is reserved. This read-only field is reserved and always has the value 0.
15-9 STROBE_DLL_STS_REF_SEL	Strobe DLL status reference select Reference delay line select taps. This is encoded by 7 fields for 127 taps.
8-2 STROBE_DLL_STS_SLV_SEL	Strobe DLL status slave select Slave delay line select status. This is the instant value generated from reference chain. Because the reference chain can only be updated when STROBE_REF_CLOCK is detected, this value can be updated with the right value when the reference is locked.
1 STROBE_DLL_STS_REF_LOCK	Strobe DLL status reference lock This signifies that the DLL has detected and locked to a half-phase REF_CLOCK shift, it allows the slave delay-line to perform programmed clock delays.
0 STROBE_DLL_STS_SLV_LOCK	Strobe DLL status slave lock Slave delay-line lock status. This signifies that a valid calibration has been set to the slave-delay line, and the slave-delay line is implementing the programmed delay value.

33.6.1.29 Vendor Specific Register (VEND_SPEC)

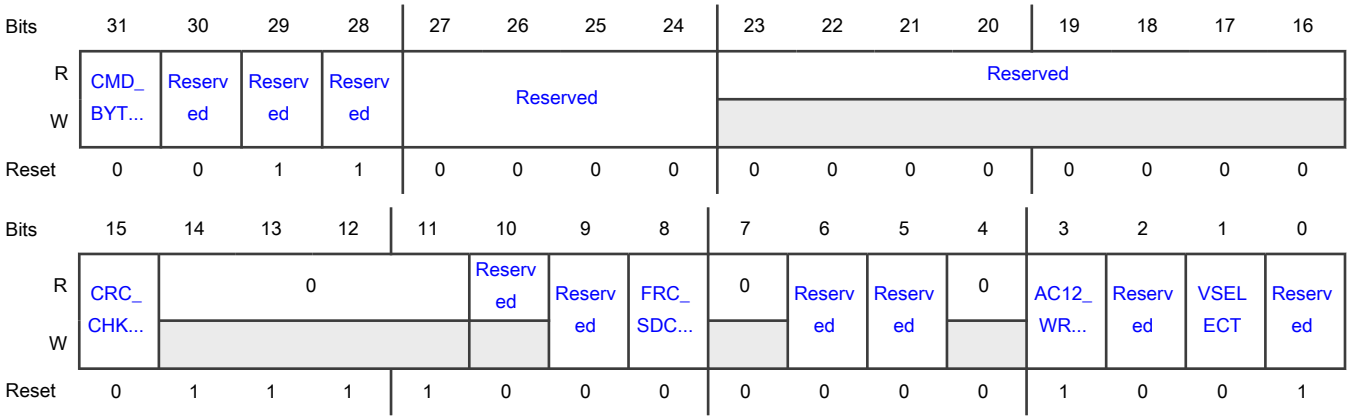
Offset

Register	Offset
VEND_SPEC	C0h

Function

This register contains the vendor specific control/status register.

Diagram



Fields

Field	Function
31 CMD_BYTE_EN	Register byte access for CMD_XFR_TYP This field controls the register byte access for Command Transfer Type (CMD_XFR_TYP). If this field is enabled, the register can be configured through byte write enable. IPS_bus can write only one byte once for one write operation. 0b - Disable. MIX_CTRL[7:0] is read/write and CMD_XFR_TYP[7:0] is read-only. 1b - Enable. MIX_CTRL[7:0] is read-only and CMD_XFR_TYP[7:0] is read/write.
30 —	Reserved Always write as 0.
29 —	Reserved Always write as 1
28 —	Reserved Always write as 0.
27-24 —	Reserved Always write as 4'b0000.
23-16 —	Reserved Always write as 8'h00.
15 CRC_CHK_DIS	CRC Check Disable 0b - Check CRC16 for every read data packet and check CRC fields for every write data packet

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Ignore CRC16 check for every read data packet and ignore CRC fields check for every write data packet
14-11 —	Reserved Reserved
10 —	Reserved Always write as 0.
9 —	Reserved Always write as 0.
8 FRC_SDCLK_ON	Force CLK Force CLK output active Do not set this bit to 1 unless it is necessary. Also, make sure that this bit is cleared when uSDHC's clock is about to be changed (frequency change, clock source change, or delay chain tuning). 0b - CLK active or inactive is fully controlled by the hardware. 1b - Force CLK active
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 AC12_WR_CHKBUSY_EN	Check busy enable Check busy enable after auto CMD12 for write data packet 0b - Do not check busy after auto CMD12 for write data packet 1b - Check busy after auto CMD12 for write data packet
2 —	Reserved
1 VSELECT	Voltage selection Change the value of output signal VSELECT to control the voltage on pads for external card. There must be a control circuit out of uSDHC to change the voltage on pads.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Change the voltage to high voltage range, around 3.0 V 1b - Change the voltage to low voltage range , around 1.8 V
0 —	Reserved Always write as 0.

33.6.1.30 eMMC Boot (MMC_BOOT)

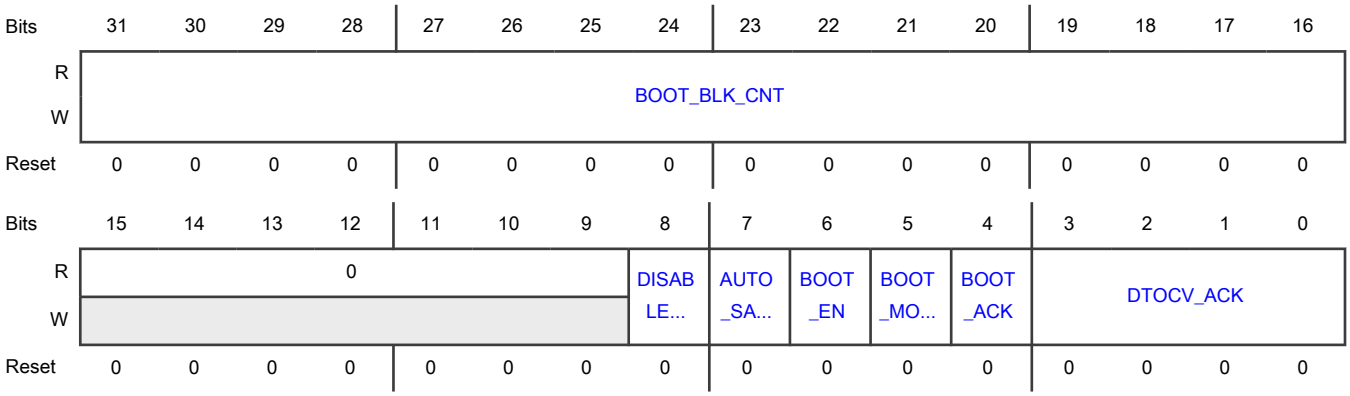
Offset

Register	Offset
MMC_BOOT	C4h

Function

This register contains the eMMC Fast Boot control register.

Diagram



Fields

Field	Function
31-16 BOOT_BLK_CNT	Stop At Block Gap value of automatic mode The value defines the Stop At Block Gap value of automatic mode. When received, card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT) and AUTO_SABG_EN is 1, then Stop At Block Gap. Here, BLK_CNT is defined in the Block Attributes Register, field 31 - 16 of 0x04.
15-9 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 DISABLE_TIME_OUT	Time out **NOTE** When this field is set, there is no timeout check no matter whether BOOT_EN is set or not. 0b - Enable time out 1b - Disable time out
7 AUTO_SABG_EN	Auto stop at block gap During boot, enable auto stop at block gap function. This function is triggered, and host stops at block gap when received card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT).
6 BOOT_EN	Boot enable Boot mode enable 0b - Fast boot disable 1b - Fast boot enable
5 BOOT_MODE	Boot mode Boot mode select 0b - Normal boot 1b - Alternative boot
4 BOOT_ACK	BOOT ACK Boot ACK mode select 0b - No ack 1b - Ack
3-0 DTOCV_ACK	DTOCV_ACK Boot ACK time out counter value 0000b - SDCLK x 2^32 0001b - SDCLK x 2^33 0010b - SDCLK x 2^18 0011b - SDCLK x 2^19 0100b - SDCLK x 2^20 0101b - SDCLK x 2^21 0110b - SDCLK x 2^22 0111b - SDCLK x 2^23 1110b - SDCLK x 2^30 1111b - SDCLK x 2^31

33.6.1.31 Vendor Specific 2 Register (VEND_SPEC2)

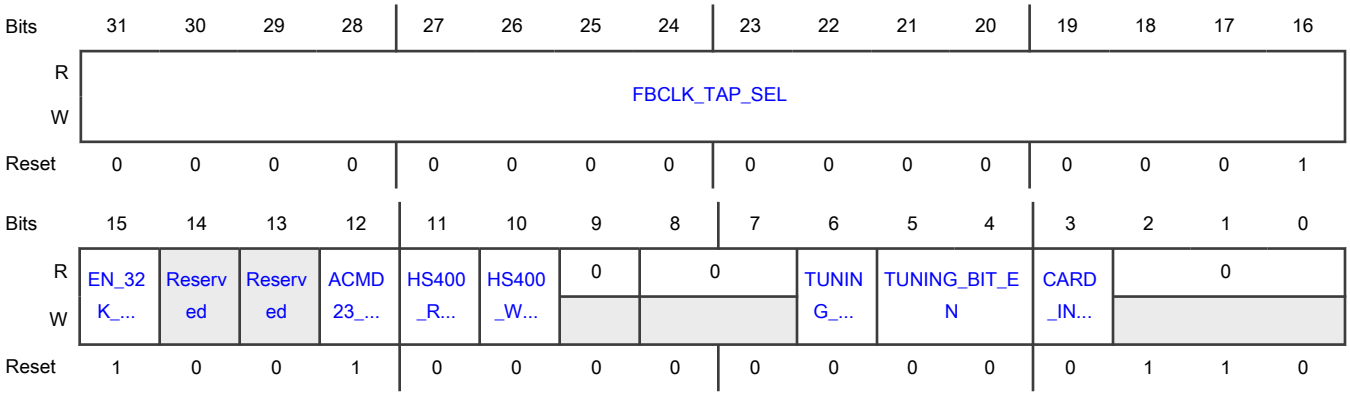
Offset

Register	Offset
VEND_SPEC2	C8h

Function

This register contains the vendor specific control 2 register.

Diagram



Fields

Field	Function
31-16 FBCLK_TAP_SEL	Enable extra delay on internal feedback clock It is a hot code. The default value is 16'h0001 that means the length of feedback delay chain is 1. The value of 16'h8000 means the length of feedback delay chain is 16.
15 EN_32K_CLK	Select the clock source for host card detection. It can use low power clock for card detection, by setting this bit to 1. 0b - Use the peripheral clock (ipg_clk) for card detection. 1b - Use the low power clock (ipg_clk_lp) for card detection.
14 —	Reserved
13 —	Reserved
12 ACMD23_ARGU2_EN	Argument2 register enable for ACMD23 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Argument2 register enable for ACMD23 sharing with SDMA system address register. Default is enabled.
11 HS400_RD_CLK_STOP_EN	HS400 read clock stop enable Only stop clock at read block gap.
10 HS400_WR_CLK_STOP_EN	HS400 write clock stop enable Only stop clock at write block gap.
9 —	Reserved
8-7 —	Reserved
6 TUNING_CMD_EN	Tuning command enable Enable the auto tuning circuit to check the CMD line. 0b - Auto tuning circuit does not check the CMD line. 1b - Auto tuning circuit checks the CMD line.
5-4 TUNING_BIT_EN	Tuning bit enable Enable the auto tuning circuit to check the DATA[7:0]. The value option is as the list below. **NOTE** In SDIO mode, auto-tuning only supports 1-bit (i.e. DATA[0]) tuning, and the 4-bit (i.e. DATA[3:0]) auto tuning is **not** supported for SDIO mode. SD mode supports 1-bit/4-bit (DATA[0] and DATA[3:0]) auto tuning. eMMC mode supports 1-bit/4-bit/8-bit (DATA[0], DATA[3:0] and DATA[7:0]) auto tuning. 00b - Enable Tuning circuit for DATA[3:0] 01b - Enable Tuning circuit for DATA[7:0] 10b - Enable Tuning circuit for DATA[0] 11b - Invalid
3 CARD_INT_D3_TEST	Card interrupt detection test This field is used only for debugging. 0b - Check the card interrupt only when DATA3 is high. 1b - Check the card interrupt by ignoring the status of DATA3.
2-0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

33.6.1.32 Tuning Control (TUNING_CTRL)

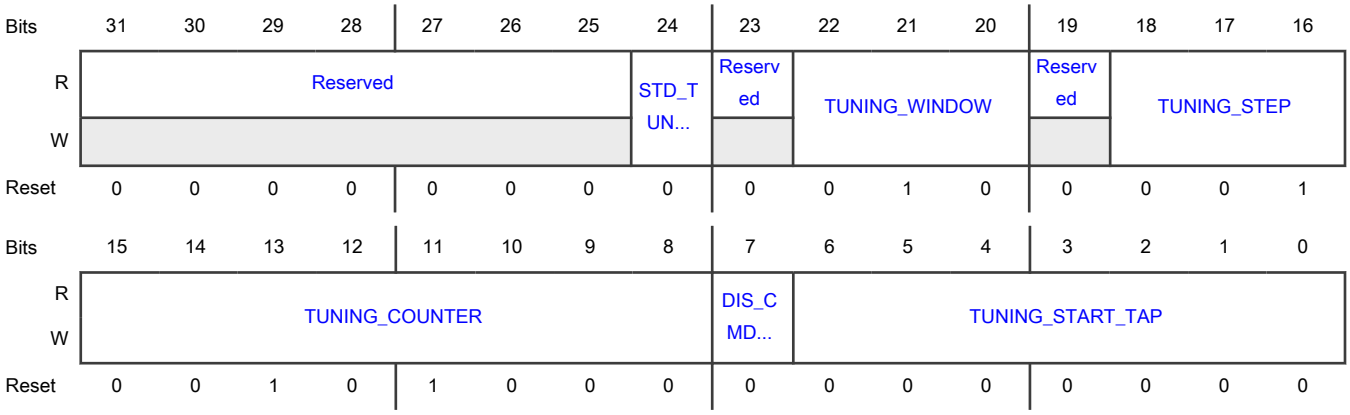
Offset

Register	Offset
TUNING_CTRL	CCh

Function

The register contains configuration of tuning circuit.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 STD_TUNING_EN	Standard tuning circuit and procedure enable This field is used to enable standard tuning circuit and procedure.
23 —	Reserved
22-20	Data window Select data window value for auto tuning

Table continues on the next page...

Table continued from the previous page...

Field	Function
TUNING_WINDOW	
19 —	Reserved
18-16 TUNING_STEP	TUNING_STEP The increasing delay cell steps in tuning procedure.
15-8 TUNING_COUNTER	Tuning counter The MAX repeat CMD19 times in tuning procedure.
7 DIS_CMD_CHK_FOR_STD_TUNING	Disable command check for standard tuning Writing 1 to this field disables command check (command CRC, CMD end bit error, and CMD index error or CMD timeout error) for standard tuning flow after each tuning command is sent.
6-0 TUNING_START_TAP	Tuning start The start delay cell point when send first CMD19 in tuning procedure.

33.6.1.33 Command Queuing Version (CQVER)

Offset

Register	Offset
CQVER	100h

Function

This register provides information about the version of the eMMC CQ standard that is implemented by CQE, in BCD format. The current version is 5.1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				MAJOR_VN				MINOR_VN				VERSION_SUFFIX			
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-8 MAJOR_VN	eMMC major version number Digit right of decimal point, in BCD format
7-4 MINOR_VN	eMMC minor version number Digit right of decimal point, in BCD format
3-0 VERSION_SUFFIX	eMMC version suffix The second digit right of decimal point, in BCD format

33.6.1.34 Command Queuing Capabilities (CQCAP)

Offset

Register	Offset
CQCAP	104h

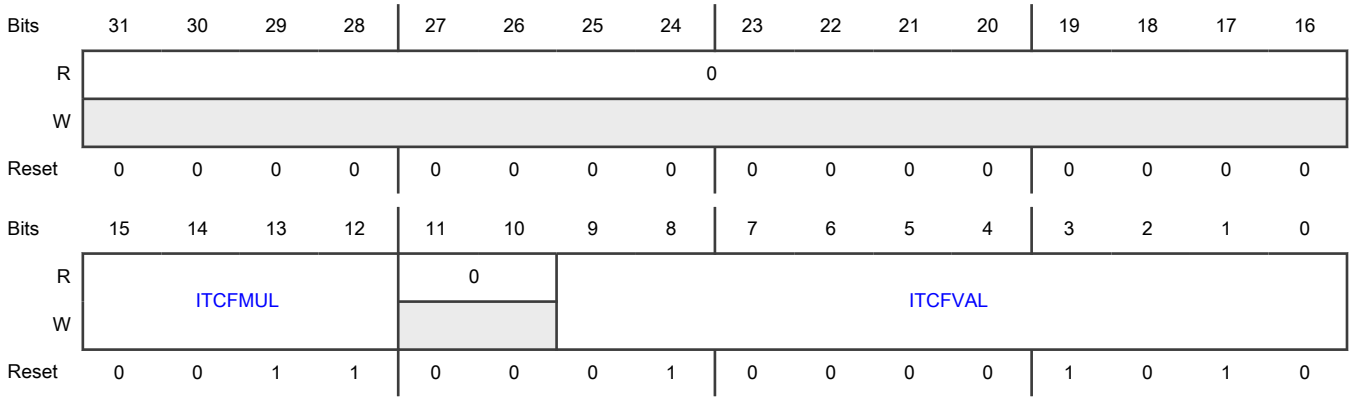
Function

This register indicates hardware capabilities.

NOTE

This register must not be configured and must be used as a read-only register. Writing to the read-write fields of this register does not affect any functionality. Reset values reflect the correct capability information.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-12 ITCFMUL	Internal timer clock frequency multiplier ITCFMUL and ITCFVAL indicate the frequency of the clock used for interrupt coalescing timer and for determining the SQS polling period. See ITCFVAL definition for details. 0001b - 0.001 MHz 0010b - 0.01 MHz 0011b - 0.1 MHz 0100b - 1 MHz 0101b - 10 MHz 0110b-1001b - Reserved
11-10 —	Reserved
9-0 ITCFVAL	Internal timer clock frequency value ITCFMUL and ITCFVAL indicate the frequency of the clock used for interrupt coalescing timer and for determining the polling period when using periodic SEND_QUEUE_ STATUS (CMD13) polling. The clock frequency is calculated as ITCFVAL* ITCFMUL. For example, to encode 19.2 MHz, ITCFVAL is C0h (= 192 decimal) and ITCFMUL is 3h (0.1 MHz): 192 * 0.1 MHz = 19.2 MHz.

33.6.1.35 Command Queuing Configuration (CQCFG)

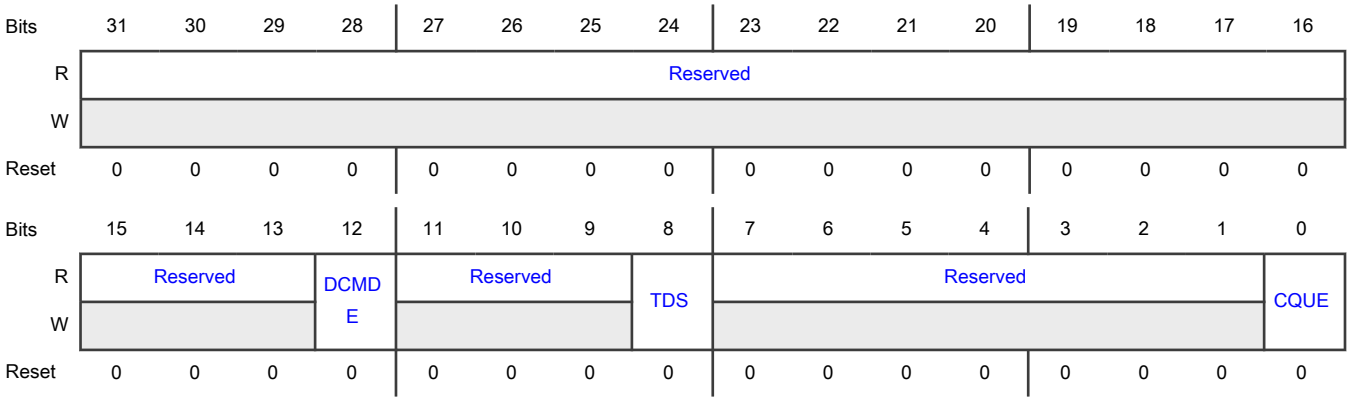
Offset

Register	Offset
CQCFG	108h

Function

This register controls CQE behavior affecting the general operation of command queuing module or operation of multiple tasks in the same time.

Diagram



Fields

Field	Function
31-13 —	Reserved
12 DCMDE	Direct command (DCMD) enable This bit indicates to the hardware whether the Task Descriptor in slot #31 of the TDL is a Data Transfer Task Descriptor, or a Direct Command Task Descriptor. CQE uses this bit when a task is issued in slot #31, to determine how to decode the Task Descriptor. 0b - Task descriptor in slot #31 is a Data Transfer Task Descriptor 1b - Task descriptor in slot #31 is a DCMD Task Descriptor
11-9 —	Reserved
8 TDS	Task descriptor size This bit indicates whether the task descriptor size is 128 bits or 64 bits as detailed in Data Structures section. This bit can only be configured when Command Queuing Enable bit is ‘0’ (command queuing is disabled)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Task descriptor size is 64 bits 1b - Task descriptor size is 128 bits
7-1 —	Reserved
0 CQUE	Command queuing enable Software shall write '1' this bit when in order to enable command queuing mode (for example, enable CQE). When this bit is 0, CQE is disabled and software controls the eMMC bus using the legacy eMMC host controller. Before software writes '1' to this bit, software shall verify that the eMMC host controller is in idle state and there are no commands or data transfers ongoing. When software wants to exit command queuing mode, it shall clear all previous tasks if such exist before setting this bit to 0.

33.6.1.36 Command Queuing Control (CQCTL)

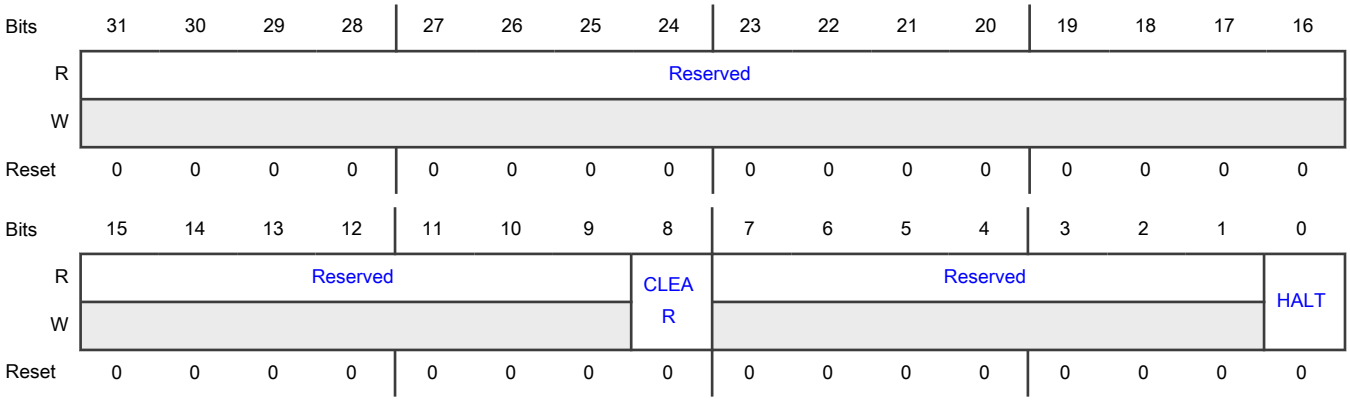
Offset

Register	Offset
CQCTL	10Ch

Function

This register controls CQE behavior affecting the general operation of command queuing module or operation of multiple tasks in the same time.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 CLEAR	Clear all tasks Software shall write '1' this bit when it wants to clear all the tasks sent to the chip. This bit can only be written when CQE is in halt state (for example, Halt bit is 1). When software writes 1, the value of the register is updated to '1', and CQE shall reset CQTDBR register and all other context information for all unfinished tasks. Then CQE will clear this bit. Software should poll on this bit until it is set to back 0 and may then resume normal operation, by clearing the Halt bit. CQE does not communicate to the chip that the tasks were cleared. It is software's responsibility to order the chip to discard the tasks in its queue using CMDQ_TASK_MGMT command. Writing '0' to this register shall have no effect.
7-1 —	Reserved
0 HALT	Halt Host software shall write '1' to the bit when it wants to acquire software control over the eMMC bus and disable CQE from issuing commands on the bus. For example, issuing a Discard Task command (CMDQ_TASK_MGMT) When software writes '1', CQE shall complete the ongoing task if such a task is in progress. Once the task is completed and CQE is in idle state, CQE shall not issue new commands and shall indicate so to software by setting this bit to 1. Software may poll on this bit until it is set to 1, and may only then send commands on the eMMC bus. In order to exit halt state (for example, resume CQE activity), software shall clear this bit (write '0'). Writing '0' when the value is already '0' shall have no effect.

33.6.1.37 Command Queuing Interrupt Status (CQIS)

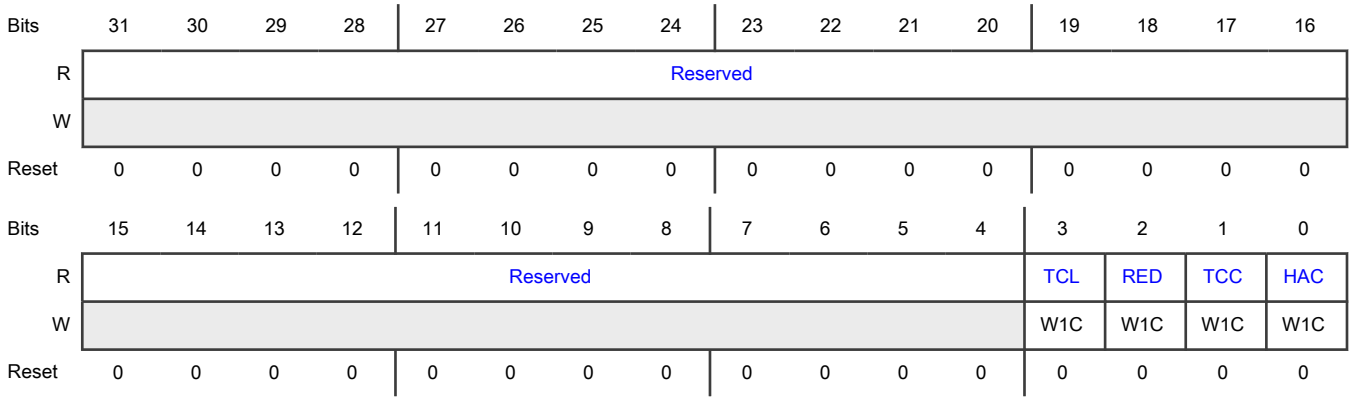
Offset

Register	Offset
CQIS	110h

Function

This register indicates pending interrupts that require service. Each bit in this registers is asserted in response a specific event, only if the respective bit is set in CQISTE register.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 TCL	Task cleared This status bit is asserted (if CQISTE[TCL_STE]=1) when a task clear operation is completed by CQE. The completed task clear operation is either an individual task clear (CQTCLR) or clearing of all tasks (CQCTL).
2 RED	Response error detected interrupt This status bit is asserted (if CQISTE[RED_STE]=1) when a response is received with an error bit set in the device status field. The contents of the device status field are listed in 6.13. Software uses CQRMEM register to configure which device status bit fields may trigger an interrupt, and which are masked.
1 TCC	Task complete interrupt This status bit is asserted (if CQISTE[TCC_STE]=1) when at least one of the following two conditions are met: • A task is completed and the INT bit is set in its Task Descriptor • Interrupt caused by Interrupt Coalescing logic (see 0)
0 HAC	Halt complete interrupt This status bit is asserted (if CQISTE[HAC_STE]=1) when halt bit in CQCTL register transitions from 0 to 1 indicating that host controller has completed its current ongoing task and has entered halt state.

33.6.1.38 Command Queuing Interrupt Status Enable (CQISTE)

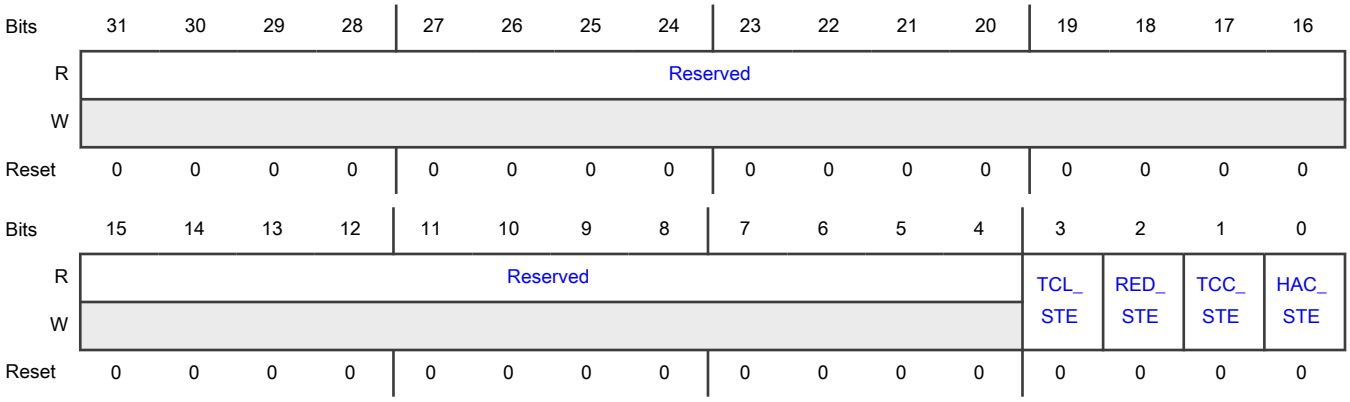
Offset

Register	Offset
CQISTE	114h

Function

This register enables and disables the reporting of the corresponding interrupt to host software in CQIS register. When a bit is set ('1') and the corresponding interrupt condition is active, then the bit is CQIS is asserted. Interrupt sources that are disabled (‘0’) are not indicated in the CQIS register. This register is bit-index matched to CQIS register.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 TCL_STE	Task cleared status enable Related to CQIS[TCL] 0b - CQIS[TCL] is disabled 1b - CQIS[TCL] is set when its interrupt condition is active
2 RED_STE	Response error detected status enable Related to CQIS[RED] 0b - CQIS[RED]is disabled 1b - CQIS[RED] is set when its interrupt condition is active
1 TCC_STE	Task complete status enable Related to CQIS[TCC] 0b - CQIS[TCC] is disabled 1b - CQIS[TCC] is set when its interrupt condition is active
0 HAC_STE	Halt complete status enable Related to CQIS[HAC] 0b - CQIS[HAC] is disabled 1b - CQIS[HAC] is set when its interrupt condition is active

33.6.1.39 Command Queuing Interrupt Signal Enable (CQISGE)

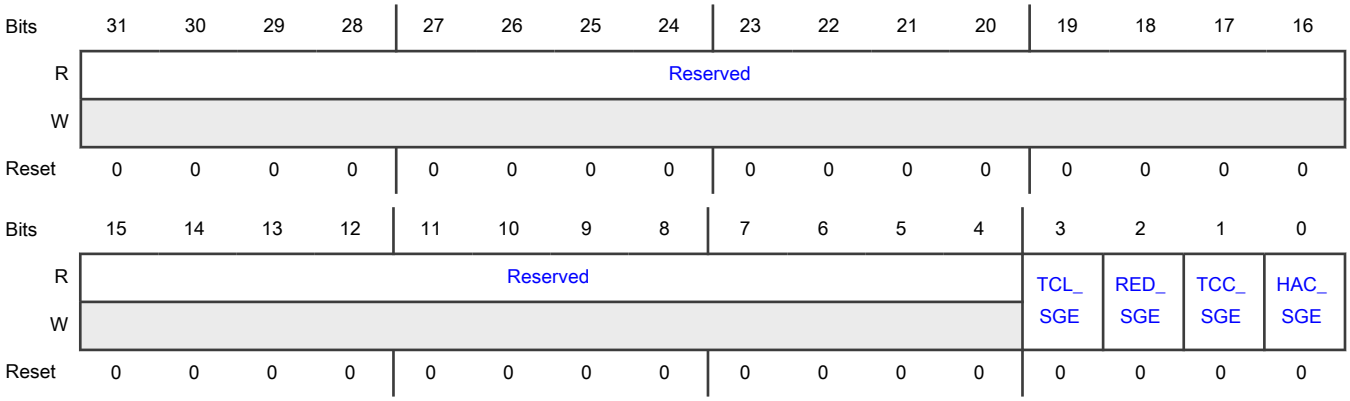
Offset

Register	Offset
CQISGE	118h

Function

This register enables and disables the generation of interrupts to host software. When a bit is set ('1') and the corresponding bit in CQIS is set, then an interrupt is generated. Interrupt sources that are disabled (‘0’) are still indicated in the CQIS register. This register is bit-index matched to CQIS register.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 TCL_SGE	Task cleared signal enable When set and CQIS[TCL] is asserted, the CQE generates an interrupt.
2 RED_SGE	Response error detected signal enable When set and CQIS[RED] is asserted, the CQE generates an interrupt.
1 TCC_SGE	Task complete signal enable When set and CQIS[TCC] is asserted, the CQE generates an interrupt.
0 HAC_SGE	Halt complete signal enable When set and CQIS[HAC] is asserted, the CQE generates an interrupt.

33.6.1.40 Command Queuing Interrupt Coalescing (CQIC)

Offset

Register	Offset
CQIC	11Ch

Function

This register controls the interrupt coalescing feature.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICENDIS	Reserved										ICSB	Reserved			
W																ICCTR
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R		Reserved			ICCTH					ICTOVAL...			ICTOVAL			
W	ICCTHWEN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ICENDIS	Interrupt coalescing enable/disable When host driver writes ‘1’, the interrupt coalescing timer and counter are reset.
30-21 —	Reserved
20 ICSB	Interrupt coalescing status This bit indicates to software whether any tasks (with INT=0) have completed and counted towards interrupt coalescing (for example, ICSB is set if and only if IC counter > 0). 0b - No task completions have occurred since last counter reset (IC counter =0) 1b - At least one task completion has been counted (IC counter >0)
19-17 —	Reserved
16 ICCTR	Counter and timer reset When host driver writes ‘1’, the interrupt coalescing timer and counter are reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 ICCTHWEN	Interrupt coalescing counter threshold write enable When software writes ‘1’, the value ICCTH is updated with the contents written at the same cycle. When software writes ‘0’, the value in ICCTH is not updated. **NOTE** Write operations to ICCTH are only allowed when the task queue is empty.
14-13 —	Reserved
12-8 ICCTH	Interrupt coalescing counter threshold Software uses this field to configure the number of task completions (only tasks with INT=0 in the Task Descriptor) which are required in order to generate an interrupt. Counter Operation: As data transfer tasks with INT=0 complete, they are counted by CQE. The counter is reset by software during the interrupt service routine. The counter stops counting when it reaches the value configured in ICCTH. The maximum allowed value is 31. **NOTE** When ICCTH is 0, task completions are not counted, and counting-based interrupts are not generated. In order to write to this field, the ICCTHWEN field must be set at the same write operation.
7 ICTOVALWEN	Interrupt coalescing timeout value write enable When software writes ‘1’, the value ICTOVAL is updated with the contents written at the same cycle. When software writes ‘0’, the value in ICTOVAL is not updated. **NOTE** Write operations to ICTOVAL are only allowed when the task queue is empty.
6-0 ICTOVAL	Interrupt coalescing timeout value Software uses this field to configure the maximum time allowed between the completion of a task on the bus and the generation of an interrupt. Timer Operation: The timer is reset by software during the interrupt service routine. It starts running when a data transfer task with INT=0 is completed, after the timer was reset. When the timer reaches the value configured in ICTOVAL field it generates an interrupt and stops. The timer’s unit is equal to 1024 clock periods of the clock whose frequency is specified in the Internal Timer Clock Frequency field CQCAP register. The minimum value is 01h (1024 clock periods) and the maximum value is 7Fh (127*1024 clock periods). For example, a CQCAP field value of 0 indicates a 19.2 MHz clock frequency (period = 52.08 ns). If the setting in ICTOVAL is 10h, the calculated polling period is 16*1024*52.08 ns= 853.33 us

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE When ICTOVAL is 0, the timer is not running, and timer-based interrupts are not generated. In order to write to this field, the ICTOVALWEN bit must be set at the same write operation.

33.6.1.41 Command Queuing Task Descriptor List Base Address (CQTDLBA)

Offset

Register	Offset
CQTDLBA	120h

Function

This register is used for configuring the lower 32 bits of the byte address of the head of the Task Descriptor List in the host memory.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TDLBA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TDLBA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TDLBA	Task descriptor list base address This register stores the LSB bits (bits 31:0) of the byte address of the head of the Task Descriptor List in system memory. The size of the task descriptor list is 32 * (Task Descriptor size + Transfer Descriptor size) as configured by Host driver. This address shall be set on 1 KByte boundary: The lower 10 bits of this register shall be set to 0 by software and shall be ignored by CQE.

33.6.1.42 Command Queuing Task Descriptor List Base Address Upper 32 Bits (CQTDLBAU)

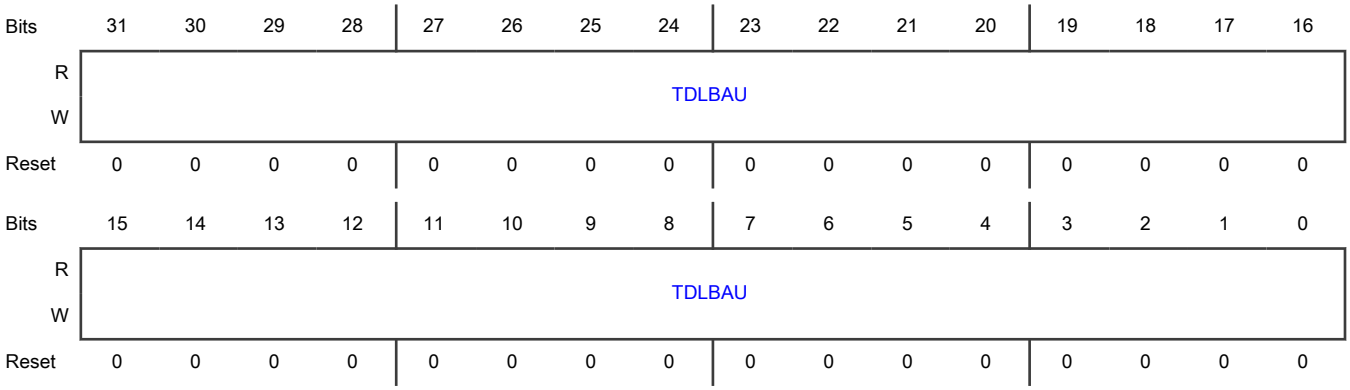
Offset

Register	Offset
CQTDLBAU	124h

Function

This register is used for configuring the upper 32 bits of the byte address of the head of the Task Descriptor List in the host memory.

Diagram



Fields

Field	Function
31-0 TDLBAU	Task descriptor list base address This register stores the MSB bits (bits 63:32) of the byte address of the head of the Task Descriptor List in system memory. The size of the task descriptor list is 32 * (Task Descriptor size + Transfer Descriptor size) as configured by Host driver. This register is reserved when using 32-bit addressing mode.

33.6.1.43 Command Queuing Task Doorbell (CQTDBR)

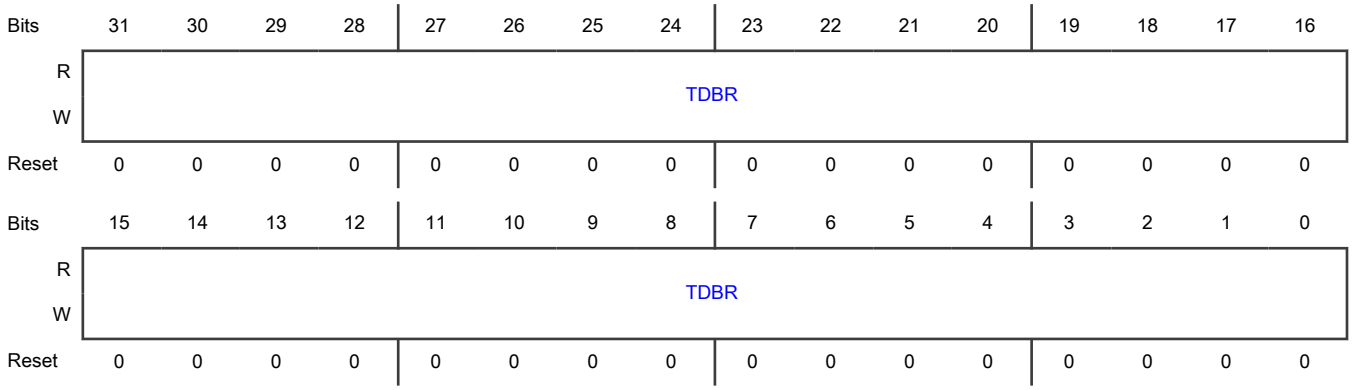
Offset

Register	Offset
CQTDBR	128h

Function

Using this register, software triggers CQE to process a new task.

Diagram



Fields

Field	Function
31-0 TDBR	Task doorbell Software configures TDLBA and TDLBAU, and enable CQE in CQCFG before using this register. Writing 1 to bit n of this register triggers CQE to start processing the task encoded in slot n of the TDL. CQE always processes tasks in-order according to the order submitted to the list by CQTDBR write transactions. CQE processes Data Transfer tasks by reading the Task Descriptor and sending QUEUED_TASK_PARAMS (CMD44) and QUEUED_TASK_ADDRESS (CMD45) commands to the device. CQE processes DCMD tasks (in slot #31, when enabled) by reading the Task Descriptor, and generating the command encoded by its index and argument. The corresponding bit is cleared to '0' by CQE in one of the following events: 1. When a task execution is completed (with success or error) 2. The task is cleared using CQTCLR register 3. All tasks are cleared using CQCTL register 4. CQE is disabled using CQCFG register Software may initiate multiple tasks at the same time (batch submission) by writing 1 to multiple bits of this register in the same transaction. In the case of batch submission: CQE shall process the tasks in order of the task index, starting with the lowest index. If one or more tasks in the batch are marked with QBR, the ordering of execution will be based on said processing order. Writing 0 by software shall have no impact on the hardware, and will not change the value of the register bit.

33.6.1.44 Command Queuing Task Completion Notification (CQTCN)

Offset

Register	Offset
CQTCN	12Ch

Function

This register is used by CQE to notify software about completed tasks.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TCN															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TCN															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TCN	Task complete notification CQE shall set bit n of this register (at the same time it clears bit n of CQTDBR) when a task execution is completed (with success or error). When receiving interrupt for task completion, software may read this register to know which tasks have finished. After reading this register, software may clear the relevant bit fields by writing 1 to the corresponding bits.

33.6.1.45 Command Queuing Device Queue Status (CQDQS)

Offset

Register	Offset
CQDQS	130h

Function

This register stores the most recent value of the device’s queue status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DQS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DQS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DQS	Device queue status Every time the Host controller receives a queue status register (QSR) from the device, it updates this register with the response of status command (For example, the chip's queue status).

33.6.1.46 Command Queuing Device Pending Tasks (CQDPT)

Offset

Register	Offset
CQDPT	134h

Function

This register indicates to software which tasks are queued in the device, awaiting execution.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DPT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DPT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DPT	Device pending tasks Bit n of this register is set if and only if QUEUED_TASK_PARAMS (CMD44) and QUEUED_TASK_ADDRESS (CMD45) were sent for this specific task and if this task hasn't been executed yet. CQE shall set this bit after receiving a successful response for CMD45. CQE shall clear this bit after the task has completed execution. Software needs to read this register in the task-discard procedure, when the controller is halted, to determine if the task is queued in the device. If the task is queued, the driver sends a CMDQ_TASK_MGMT (CMD48) to the device ordering it to discard the task. Then software clears the task in the CQE. Only then the software orders CQE to resume its operation using CQCTL register.

33.6.1.47 Command Queuing Task Clear (CQTCLR)

Offset

Register	Offset
CQTCLR	138h

Function

This register is used for removing an outstanding task in the CQE. The register should be used only when CQE is in Halt state.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TCLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TCLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TCLR	Task clear Writing 1 to bit n of this register orders CQE to clear a task which software has previously issued. This bit can only be written when CQE is in Halt state as indicated in CQCFG register Halt bit.

Table continues on the next page...

Field	Function
	When software writes ‘1’ to a bit in this register, CQE updates the value to ‘1’, and starts clearing the data structures related to the task. CQE clears the bit fields (sets a value of 0) in CQTCLR and in CQTDBR once clear operation is complete. Software should poll on the CQTCLR until it is cleared to verify clear operation was complete. Writing to this register only clears the task in the CQE and does not have impact on the device. In order to discard the task in the device, host software shall send CMDQ_TASK _MGMT while CQE is still in Halt state. Host driver is not allowed to use this register to clear multiple tasks at the same time. Clearing multiple tasks can be done using CQCTL register. Writing 0 to a register bit shall have no impact.

33.6.1.48 Command Queuing Send Status Configuration 1 (CQSSC1)

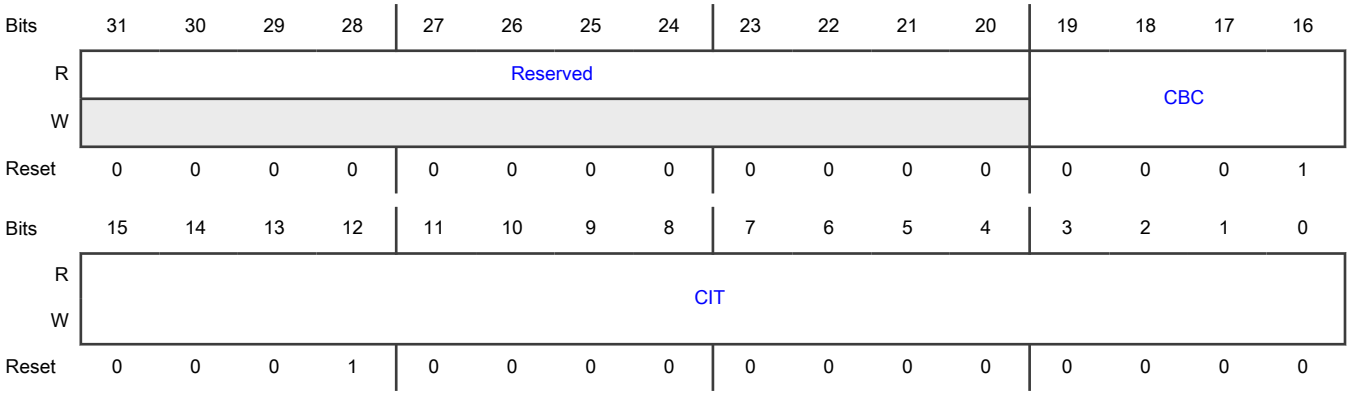
Offset

Register	Offset
CQSSC1	140h

Function

The register controls the when SEND_QUEUE_STATUS commands are sent.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16	Send status command block counter

Table continues on the next page...

Table continued from the previous page...

Field	Function
CBC	This field indicates to CQE when to send SEND_QUEUE_STATUS (CMD13) command to inquire the status of the device's task queue. A value of *n* means CQE sends status command on the CMD line, during the transfer of data block BLOCK_CNT-n, on the data lines, where BLOCK_CNT is the number of blocks in the current transaction. A value of 0 means that SEND_QUEUE_STATUS (CMD13) command is not sent during the transaction. Instead it will be sent only when the data lines are idle. A value of 1 means that STATUS command is to be sent during the last block of the transaction.
15-0 CIT	Send status command idle timer This field indicates to CQE the polling period to use when using periodic SEND_QUEUE_STATUS (CMD13) polling. Periodic polling is used when tasks are pending in the device, but no data transfer is in progress. When a SEND_QUEUE_STATUS response indicating that no task is ready for execution, CQE counts the configured time until it issues the next SEND_QUEUE_STATUS. Timer units are clock periods of the clock whose frequency is specified in the Internal Timer Clock Frequency field CQCAP register. The minimum value is 0001h (1 clock period) and the maximum value is FFFFh (65535 clock periods). Default interval is: 4096 clock periods. For example, a CQCAP field value of 0 indicates a 19.2 MHz clock frequency (period = 52.08 ns). If the setting in CQSSC1.CIT is 1000h, the calculated polling period is 4096*52.08 ns= 213.33 us

33.6.1.49 Command Queuing Send Status Configuration 2 (CQSSC2)

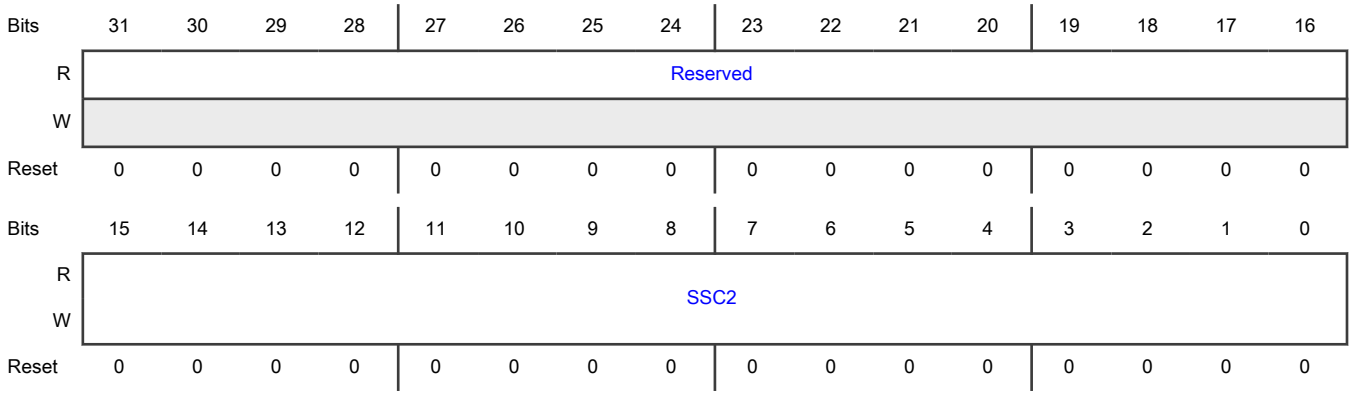
Offset

Register	Offset
CQSSC2	144h

Function

This register is used for configuring RCA field in SEND_QUEUE_STATUS command argument.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 SSC2	Send queue status RCA This field provides CQE with the contents of the 16-bit RCA field in SEND_QUEUE_ STATUS (CMD13) command argument. CQE copies this field to bits 31:16 of the argument when transmitting SEND_ QUEUE_STATUS (CMD13) command.

33.6.1.50 Command Queuing Command Response for Direct-Command Task (CQCRDCT)

Offset

Register	Offset
CQCRDCT	148h

Function

This register is used for passing the response of a DCMD task to software.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CRDCT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CRDCT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CRDCT	Direct command last response This register contains the response of the command generated by the last direct-command (DCMD) task which was sent. CQE updates this register when it receives the response for a DCMD task. This register is considered valid only after bit 31 of CQTDBR register is cleared by CQE.

33.6.1.51 Command Queuing Response Mode Error Mask (CQRMEM)

Offset

Register	Offset
CQRMEM	150h

Function

This register controls the generation of Response Error Detection (RED) interrupt.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RMEM															
W																
Reset	1	1	1	1	1	1	0	1	1	1	1	1	1	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RMEM															
W																
Reset	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RMEM	Response mode error mask This bit is used as in interrupt mask on the device status filed which is received in R1/R1b responses. The reset value of this register is set to trigger an interrupt on all “Error” type bits in the device status, as defined in 6.13. **NOTE** Responses to CMD13 (SQS) encode the QSR, so they are ignored by this logic. 0000_0000_0000_0000_0000_0000_0000_0000b - When a R1/R1b response is received, bit i in the device status is ignored 0000_0000_0000_0000_0000_0000_0000_0001b - When a R1/R1b response is received, with bit i in the device status set, a RED interrupt is generated

33.6.1.52 Command Queuing Task Error Information (CQTERRI)

Offset

Register	Offset
CQTERRI	154h

Function

This register is updated by CQE when an error occurs on data or command related to a task activity.

When such error is detected by CQE or indicated by the eMMC controller CQE stores in CQTERRI the task IDs and the command indices of the commands which were executed on the command line and data lines when the error occurred.

Software is expected to use this information in the error recovery procedure. See B.2.8 for more details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DTEFV	Reserved			DTETID				Reserved				DTECI			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RMEFV	Reserved			RMETID				Reserved				RMECI			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DTEFV	Data transfer error fields valid This bit is updated when an error is detected by CQE, or indicated by eMMC controller. If a data transfer is in progress when the error is detected/indicated, the bit is set to 1. If a no data transfer is in progress when the error is detected/indicated, the bit is cleared to 0.
30-29 —	Reserved
28-24 DTETID	Data transfer error task ID This field indicates the ID of the task which was executed on the data lines when an error occurred. The field is updated if a data transfer is in progress when an error is detected by CQE, or indicated by eMMC controller.
23-22 —	Reserved
21-16 DTECI	Data transfer error command index This field indicates the index of the command which was executed on the data lines when an error occurred. The index shall be set to EXECUTE_READ_TASK (CMD46) or EXECUTE_WRITE_TASK (CMD47) according to the data direction. The field is updated if a data transfer is in progress when an error is detected by CQE, or indicated by eMMC controller.
15 RMEFV	Response mode error fields valid This bit is updated when an error is detected by CQE, or indicated by eMMC controller. If a command transaction is in progress when the error is detected/indicated, the bit is set to 1. If a no command transaction is in progress when the error is detected/indicated, the bit is cleared to 0.
14-13 —	Reserved
12-8 RMETID	Response mode error task ID This field indicates the index of the command which was executed on the command line when an error occurred. The field is updated if a command transaction is in progress when an error is detected by CQE, or indicated by eMMC controller.
7-6 —	Reserved
5-0	Response mode error command index

Table continues on the next page...

Table continued from the previous page...

Field	Function
RMECI	This field indicates the index of the command which was executed on the command line when an error occurred. The field is updated if a command transaction is in progress when an error is detected by CQE, or indicated by eMMC controller.

33.6.1.53 Command Queuing Command Response Index (CQCRI)

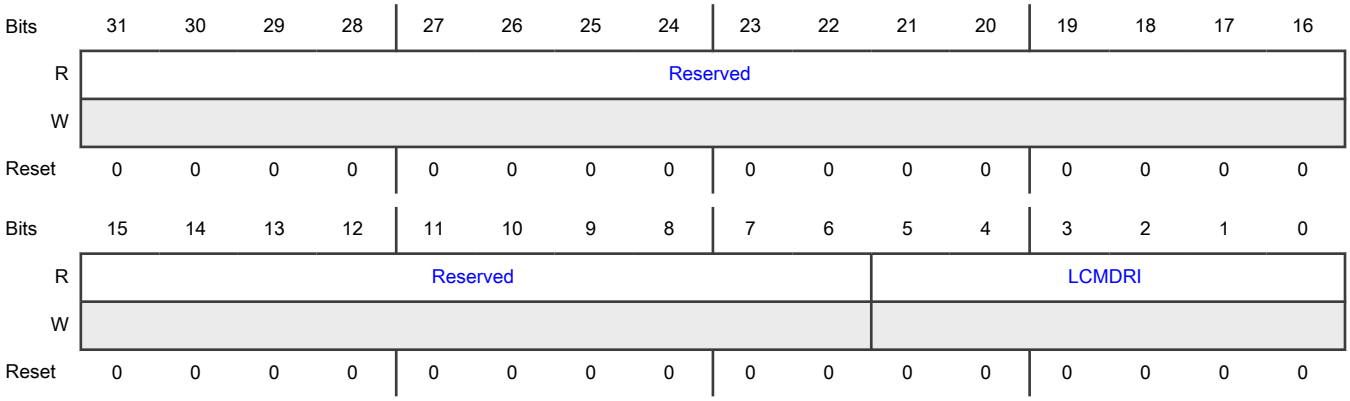
Offset

Register	Offset
CQCRI	158h

Function

This register stores the index of the last received command response.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 LCMDRI	Last command response index This field stores the index of the last received command response. CQE shall update the value every time a command response is received..

33.6.1.54 Command Queuing Command Response Argument (CQCRA)

Offset

Register	Offset
CQCRA	15Ch

Function

This register stores the argument of the last received command response.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LCMDRA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LCMDRA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 LCMDRA	Last command response argument This field stores the argument of the last received command. CQE shall update the value every time a command response is received.

Chapter 34
Flexible Serial Peripheral Interface Follower (FlexSPI_FLR)

34.1 Chip-specific FlexSPI_FLR information

Table 259. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

Table 260. RT1180 FlexSPI_FLR

Description	FlexSPI_Follower
AXI RX BUFFER size	(2^8)*8=2K Byte
AXI RX BUFFER Address Range	0x00 – 0xFF
AXI TX BUFFER size	(2^7)*8=1K Byte
AXI TX BUFFER Address Range	0x00 – 0x7F

34.2 Overview

FlexSPI_FLR is a Serial Peripheral Interface (SPI) block working as an SPI follower. It communicates with other chips or with FPGA by using the SPI bus protocol.

A remote chip with an SPI leader can read or write this block's internal registers. Alternatively, it can access the chip address via this block's AXI bus.

34.2.1 Block diagram

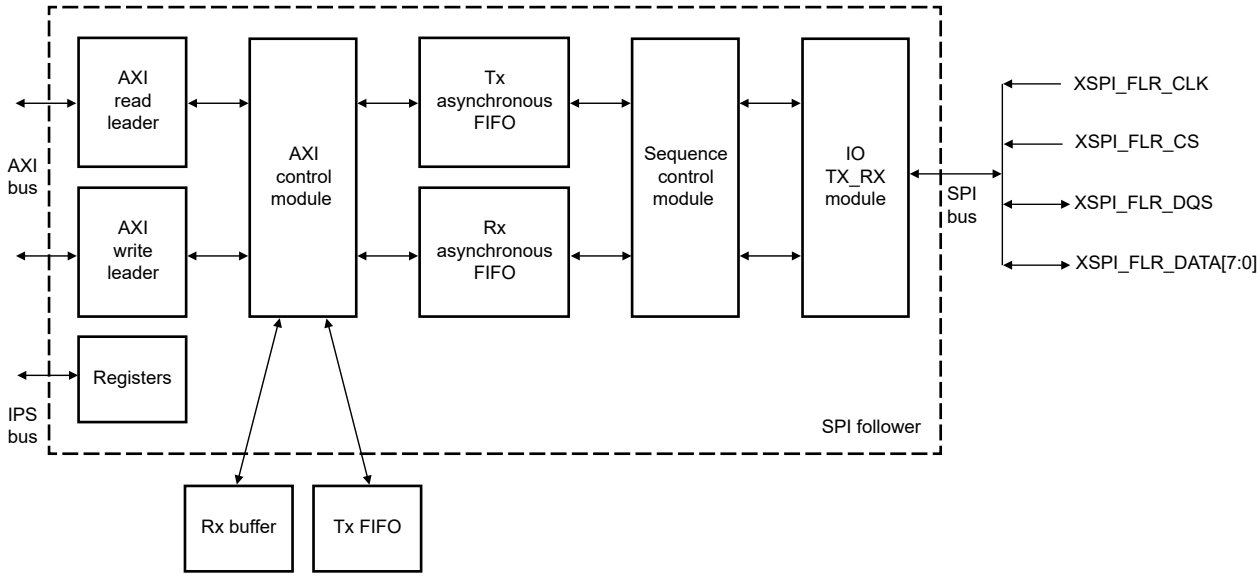


Figure 259. Block diagram

34.2.2 Features

- SPI support for these modes:
  - DDR
  - SDR
  - Octal
  - Quad
- DQS output during read commands, assisting the SPI leader with data sampling
- DQS input during write commands, as DATA strobe only in Octal DDR mode to support unaligned writes
- Automatic stop of DQS toggling during read commands to avoid FIFO underflow
- Memory read and write commands, and register read and write commands with configurable command code and dummy cycles
- Inter-Processor Communication (IPC) protocol for SPI leader and follower communication
- 9x32-bit mailbox registers which are writable and readable via the SPI leader, and are only readable via the chip
- Asynchronous interrupt through mailbox which can be set by the SPI leader
- Independent AXI write and read leaders
- Four independent read leaders
- Two different pairs of SPI memory read and write commands to access two different locations in the chip via the AXI bus
- Configurable AXI memory-base-address and address-range setting via register
- Blocking of AXI reads and writes to ignore requests from the SPI leader

34.3 Functional description

34.3.1 Modes

34.3.1.1 SDR and DDR modes

In SDR mode, FlexSPI_FLR captures the received data on the SCK rising edge. The FlexSPI leader must center-align the SCK and write data. FlexSPI_FLR sends the read data after the SCK falling edge. DQS also toggles a half cycle before the valid data emerges. See Application information.

During DDR mode, FlexSPI_FLR captures the received data on both the rising edge and SCK falling edge. The FlexSPI leader center-aligns both edges of SCK with write data. FlexSPI_FLR sends out the read data after the rising edge of SCK. DQS also toggles with the same phase as valid data. See Application information.

34.3.1.2 Quad and Octal modes

In Quad mode, FlexSPI captures and sends data through XSPI_FLR_DATA[3:0]. FlexSPI_FLR does not drive the upper four bits, and the bits are inputs relative to the block.

In Octal mode, FlexSPI captures and sends data through XSPI_FLR_DATA[7:0]. DQS is also sampled during the bit-capture phase.

34.3.1.3 Low-power modes

FlexSPI_FLR's AXI clock and module clock can shut down to enter a power-saving mode. The SPI clock, however, must remain active to allow the remote SPI leader to wake the block. When neither an AXI nor module clock exists, the remote SPI leader cannot issue memory reads or writes. The mailbox inside FlexSPI_FLR is accessible via register read and write commands. The remote SPI leader sets the interrupt inside the mailbox. The SPI clock domain sets the block's interrupt signal. After the system sees this interrupt, it can restart the AXI clock and module clock signals and clear the interrupt when the system is ready.

Meanwhile, the remote SPI leader polls the interrupt signal in the mailbox. After the leader sees that the interrupt field is cleared, you can safely issue further SPI commands.

34.3.1.4 Debug mode

FlexSPI_FLR does not support this mode.

34.3.2 Operations

34.3.2.1 Read memory

If an incoming SPI command matches the memory read command set, FlexSPI_FLR triggers an AXI read to carry read data from the chip memory. The chip access base address and address range are configurable via registers, and you can configure the shift address directly from the SPI bus.

NOTE

If an incoming request exceeds the access address range, the part beyond the range will not be read from the AXI bus, and FlexSPI_FLR's AXI interface sends no AXI read request.

The following figure shows the flow of filling the read data into the AXI RX buffer. Meanwhile, internal logic dynamically monitors the valid content inside the AXI RX buffer. After, the read data becomes valid, it is pushed into IO TX FIFO. The IO TX FIFO starts to pop the read data when it reaches the dummy cycles.

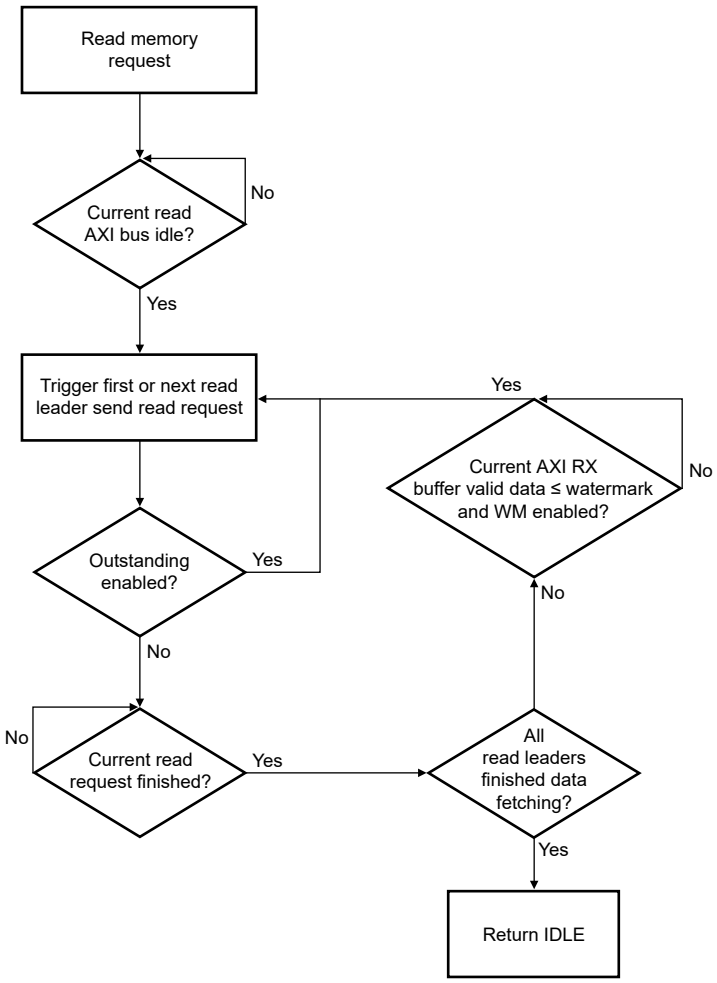


Figure 260. AXI read data flow

RDFETCHSIZE defines the maximum size for AXI fetch data. The leader must not issue a read request size larger than the limit; otherwise, the return data wraps back to address 0 of the AXI RX buffer.

If a previous request-triggered AXI read is still ongoing, the upcoming read request must wait until the current AXI read becomes idle, and the previous read request latency is added to the current read. If any read leader does not send a read request and sees the CS signal assert, it terminates the waiting and returns to idle. It is useful when the general SPI read size is small.

34.3.2.2 Write memory

If an incoming SPI command matches the memory write command set, FlexSPI_FLR triggers an AXI write to carry write data to the chip memory. The chip access base address and address range are configurable via registers, and you can configure the shift address directly from the SPI bus.

NOTE

If an incoming request exceeds the access address range, the part beyond the range is not written to the AXI bus. Additionally, FlexSPI_FLR's AXI interface does not send any more AXI write requests. All incoming write data is discarded inside the block.

The AXI write operation does not support multi-leader outstanding with only the bus leader inside of the IP. The internal logic continuously pushes write data into the AXI TX FIFO. The module monitors the filling level of AXI TX FIFO in real time.

After the filling level exceeds the watermark level that WRM sets, a write request is sent to the AXI write leader.

If the XSPI_FLR_CS signal is deasserted, all FIFO content is pushed in one request.

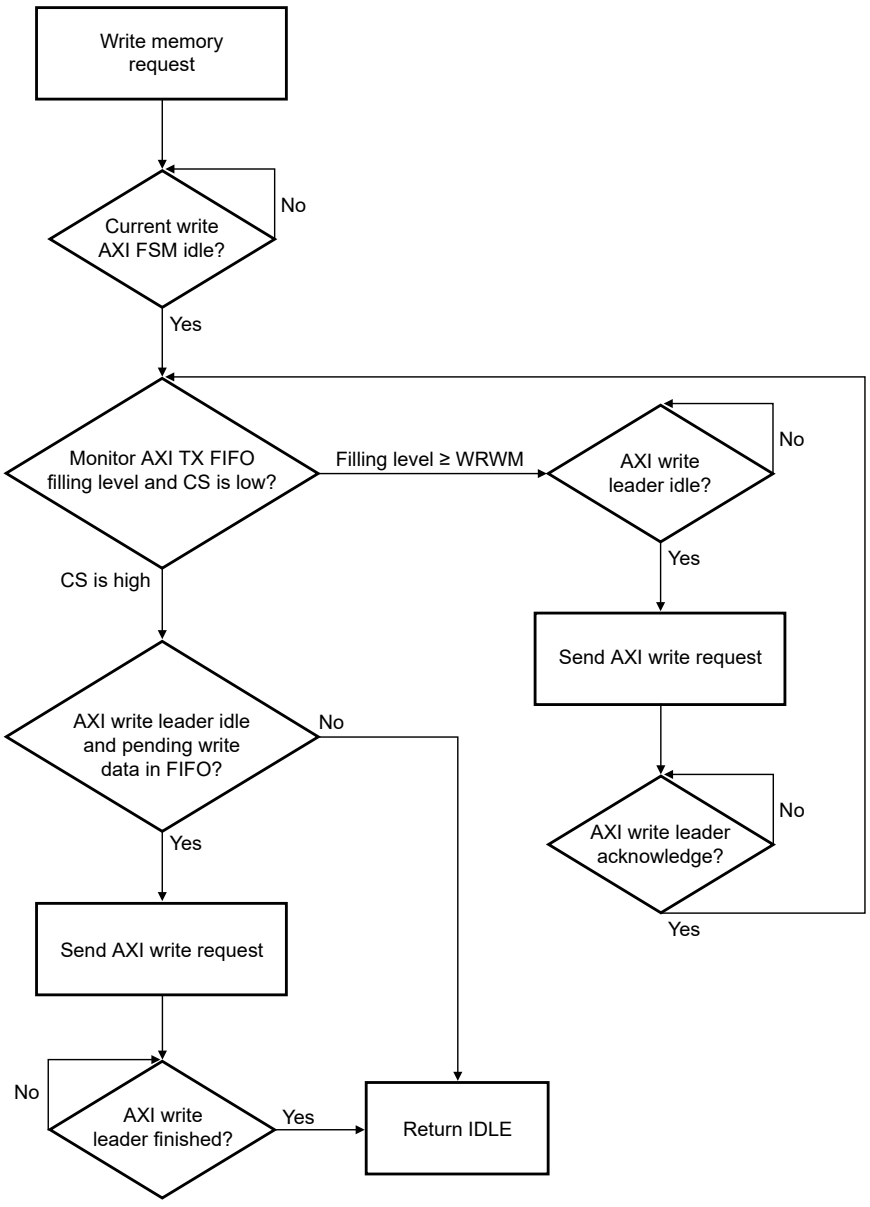


Figure 261. AXI write data flow

FlexSPI_FLR cannot accept a new write request until it finishes the current memory write command. If you attempt a write in this case, you may write the data into the wrong location.

Because the module monitors the TX FIFO in real time, the AXI bus speed restricts the maximum write size that the module can accept. To avoid a potential overflow, do not transfer a write size larger than the TX FIFO depth.

34.3.2.3 Read/write registers

If FlexSPI_FLR decodes an incoming SPI command as a register read or write, FlexSPI_FLR temporarily takes control of the IPS bus for a register read or write. Any read or write register request from the chip causes assertion of IPS_bus_transfer_wait.

There is an internal command FIFO for register reads and writes, with the default FIFO depth set to 16, which means that if the IPS domain is slow, the maximum command it can receive is 16.

For a register write operation, FlexSPI_FLR only takes the first 32 bits of data from the SPI bus as written data. Any subsequent incoming write data is received but discarded. The FlexSPI leader can only write to SPIMAIL*n* registers; writes to other internal registers in the module are discarded.

For a register read operation, if the CS signal is asserted, IO continuously sends out the same register read value. The FlexSPI leader can only read from the SPIMAIL*n* registers and FlexSPI_FLR status register. Reading from other registers returns read data 0.

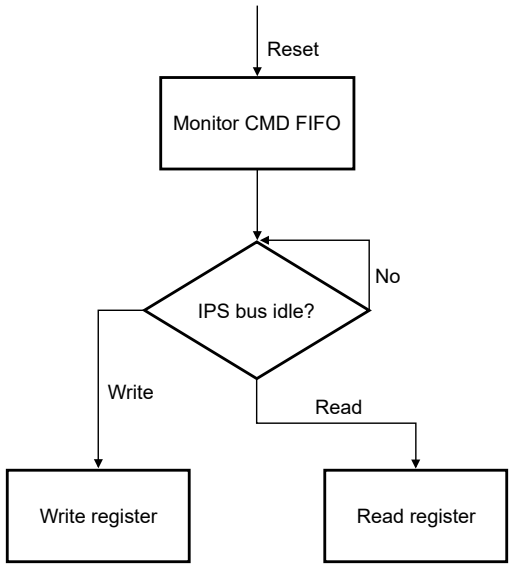


Figure 262. Read and write register flow

34.3.2.4 AXI read and write blocking

FlexSPI_FLR supports block AXI reads and writes via MODULE_CONTROL[5:10].

MODULE_CONTROL[BLKWRITE] and MODULE_CONTROL[BLKREAD] can block AXI reading and writing completely. After these fields become 1, any incoming write command is accepted, but the write data is discarded, and any incoming read command is accepted, but all returned read data are 0s. No action occurs on FlexSPI_FLR's AXI leader interface.

MODULE_CONTROL[BLKNXTWR] and MODULE_CONTROL[BLKNXTRD] control the SPI leader's access frequency.

After MODULE_CONTROL[BLKNXTRD], MODULE_CONTROL[BLKNXTWR] or both are 1, the module only supports the next incoming SPI read or write request. (An ongoing SPI read or write also counts, so the upper-level software controls access.) After FlexSPI_FLR finishes processing one more read or write, any further incoming write is accepted but discarded, and any incoming read returns zeroes.

When MODULE_CONTROL[BLKNXTRD] or MODULE_CONTROL[BLKNXTWR] is 1, you can allow one more read or write by writing MODULE_CONTROL[ALLONERD] or MODULE_CONTROL[ALLONEWR]. (An ongoing SPI read or write also counts, so the upper-level software controls the access.) It is self-cleared. It allows the SPI leader to access the SPI follower one more time.

34.3.3 Clocking

In FlexSPI_FLR:

- The AXI clock, module clock, and SPI clock are fully asynchronous.
- The SCK clock from the SPI leader is asynchronous with all internal clocks from the chip.

Due to bandwidth concerns, the AXI clock must be fast enough to handle the data input or output on the SPI bus. The AXI bus bandwidth is roughly equal to 8 bytes × AXI clock speed.

The frequency of the module clock impacts the performance of SPI register read or write commands, but normally the impact is small.

Set the frequency of SPI clock = 2 × XSPI_FLR_CLK. This frequency allows FlexSPI_FLR to respond quickly to the requests from the SPI bus.

34.3.4 Reset

FlexSPI_FLR has an internal synchronization module to synchronize the reset release in the AXI, SPI, and module domain. The reset is released after AXI clock, SPI clock, and module clock start to toggle.

The reset assert is fully asynchronous; it does not require a clock.

34.3.5 Interrupts

Table 261. Interrupts

Interrupt	Description
IO TX FIFO underflow	If the DQS stop feature is disabled (MODULE_CONTROL[DQSSTOP] = 0), it is possible that the AXI bus returns data too slowly to meet the SPI bus bandwidth during a read command. This causes a TX FIFO overflow in the IO module. The SPI leader side receives the wrong data. If MODULE_CONTROL[DQSSTOP] = 1, this interrupt only indicates that a DQS stop happened during a read command.
IO RX FIFO overflow	If the AXI and root clock domain is too slow to handle the received data from the SPI bus, an IO RX FIFO overflow occurs. After that it drops any following data coming through the IO RX FIFO. If the command is a write command, no write data is written through the AXI write leader. If it is a read command, the block ignores this overflow during the read data phase, because the IO RX FIFO is not used during that time.
Error command received	If an incoming command does not match any command set within the block, then the block ignores this command and waits for CS to deassert.
SPIMAIL interrupt	The SPI leader can write 1 to SPIMAIL0[0] to generate this interrupt; the chip can clear this.

34.4 External signals

Table 262. External signals

Signal	Description	Direction
XSPI_FLR_CLK	Clock input from SPI leader	I
XSPI_FLR_CS	Chip select input from SPI leader	I
XSPI_FLR_DQS	Input DQS as write mask from SPI leader, output DQS to SPI leader	I/O
XSPI_FLR_DATA[7:0]	Data I and O to SPI leader	I/O

34.5 Initialization

To initialize FlexSPI_FLR:

1. Enable the clock (AXI clock, IP bus clock, or root clock) for XSPI_SLV.
2. Write 1 to MODULE_CONTROL[SWRESET] to initiate software reset. Then, write 0 to this field.
3. Configure MODULE_CONTROL[IOMODE] to select I or O mode.
4. Configure RW_COMMAND_BASE[ADDRBASE2] or RW_COMMAND_BASE[ADDRBASE1] to set the read and write base address.

5. Write 1 to MODULE_CONTROL[CMDRANGEBASEUPDATE] to update the AXI command's base address.
6. Write the read and write command address range to CMD1_RANGE[RANGE] and CMD2_RANGE[RANGE].
7. Write 1 to MODULE_CONTROL[CMDRANGEBASEUPDATE] to update the AXI command's base address and range.
8. Write 1 to READ_COMMAND_CONTROL[RDWM] and READ_COMMAND_CONTROL[WMEN] to read and write watermark levels.
9. After performing the aforementioned configuration, write 0 to MODULE_CONTROL[CSMASK].

34.6 Application information

34.6.1 Write command

Table 263. Write sequence for FlexSPI_FLR

Instruction #	Command sequence input	Mode select	Settings
1	CMD	SDR write (see Figure 263)	• MODULE_CONTROL[IOMODE] = 00b or 01b • Configure Write Command 1 Setting (WRITE_COMMAND1) and Write Command 2 Setting (WRITE_COMMAND2)
		DDR write (see Figure 264)	• MODULE_CONTROL[IOMODE] = 10b or 11b • Configure WRITE_COMMANDx
2	ADDR	Same as CMD (you must configure the base address and address range.	RW_COMMAND_BASE[ADDRBASEx] is for the base address. CMDx_RANGE[RANGE] is for the address range.
3	DATA	SDR×4	MODULE_CONTROL[IOMODE] = 00b
		SDR×8	MODULE_CONTROL[IOMODE] = 01b
		DDR×4	MODULE_CONTROL[IOMODE] = 10b
		DDR×8	MODULE_CONTROL[IOMODE] = 11b

As seen in Figure 263 and Figure 264:

- Write: Command + Address + Write Data In
- No DQS toggling is allowed for SDR or DDR×4. All incoming CMD, ADDR, and DATA are sampled by the rising edge of SCK.

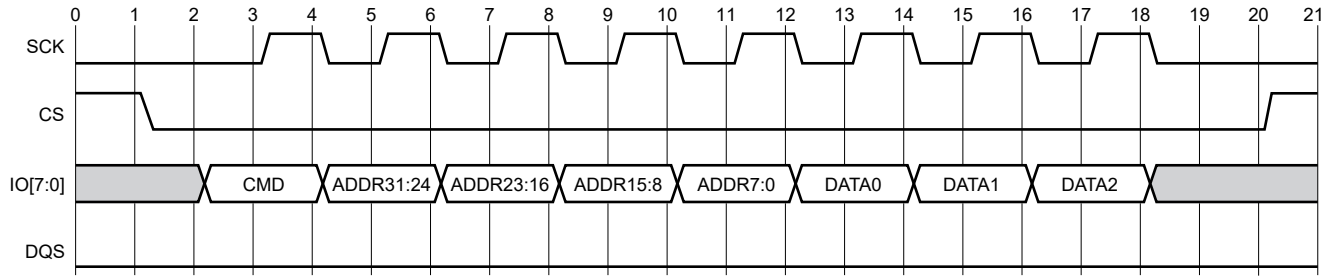


Figure 263. SDR write command

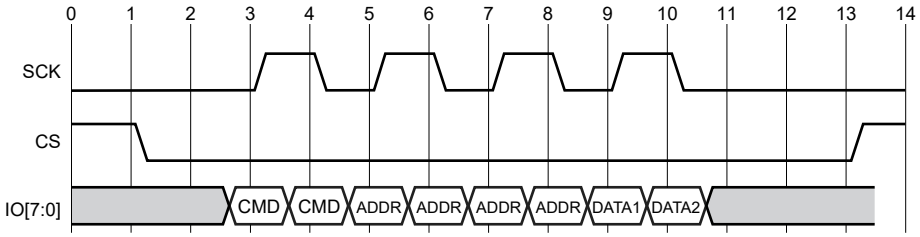


Figure 264. DDR write command

34.6.2 Read command

Table 264. Read sequence for FlexSPI_FLR

Instruction #	Command sequence input	Mode select	Settings
1	CMD	SDR write (see Figure 265 and Figure 266)	• MODULE_CONTROL[IOMODE] = 00b or 01b • Configure Read Command 1 setting (READ_COMMAND1) and Read Command 2 setting (READ_COMMAND2)
		DDR write (see Figure 267 and Figure 268)	• MODULE_CONTROL[IOMODE] = 10b or 11b • Configure READ_COMMANDx
2	ADDR	Same as CMD (you must configure the base address and address range).	RW_COMMAND_BASE[ADDRBASEx] is for the base address. CMDx_RANGE[RANGE] is for the address range.
3	DUMMY_CYCLES	Dummy clock cycles between the ADDR and DATA sequence.	READ_COMMANDx[DUMMYCYCLES] defines dummy cycles.
4	DATA	SDR×4	MODULE_CONTROL[IOMODE] = 00b
		SDR×8	MODULE_CONTROL[IOMODE] = 01b

Table continues on the next page...

Table 264. Read sequence for FlexSPI_FLR (continued)

Instruction #	Command sequence input	Mode select	Settings
		DDR×4	MODULE_CONTROL[IOMODE] = 10b
		DDR×8	MODULE_CONTROL[IOMODE] = 11b

As shown in the following figures:

- Read data: Command + Address + Dummy cycles + Read data out
- Read status: Command + Address+ Dummy cycles + Read status out
- Dummy cycles exist between the ADDR and DATA phases. The clock uses these cycles to process the read command and compensate internal asynchronous FIFO read data transferring latency.
- The SCK falling edge (see Read Command 2 setting (READ_COMMAND2) SCK edge 4) pops out the read data. A chip delay exists between SCK input to I/O valid data output ("chip delay" in Read Command2). This delay could be larger than one cycle of SCK, depending on chip implementation. Therefore, the DQS output must arrive along with valid data from the block. DQS must always be in phase with the read data output. The SPI leader must use DQS to sample the read data.

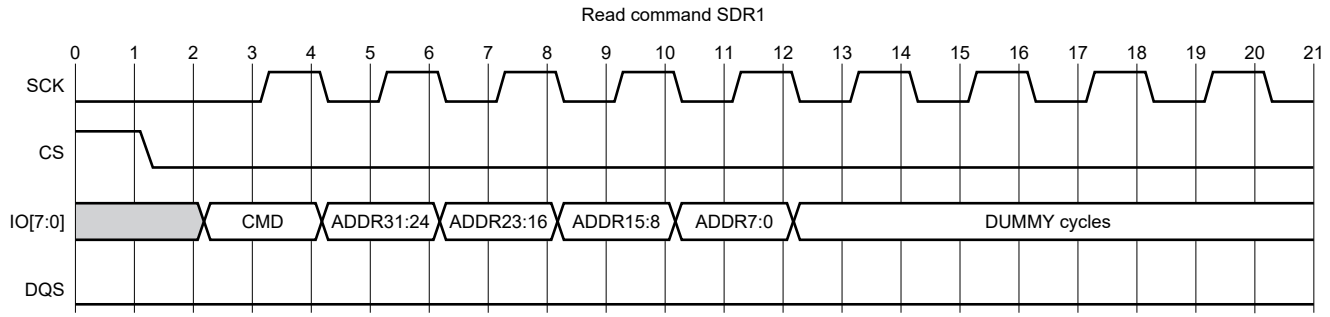


Figure 265. Read command SDR1

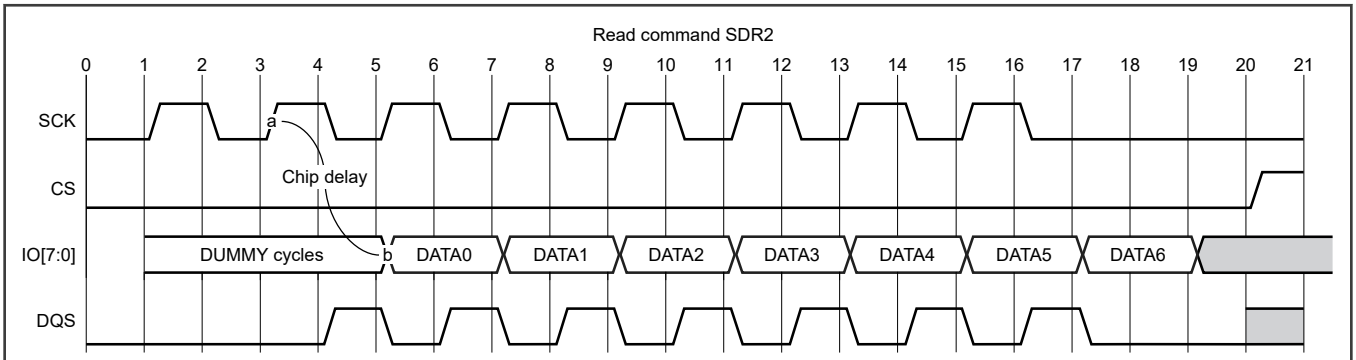


Figure 266. Read command SDR2

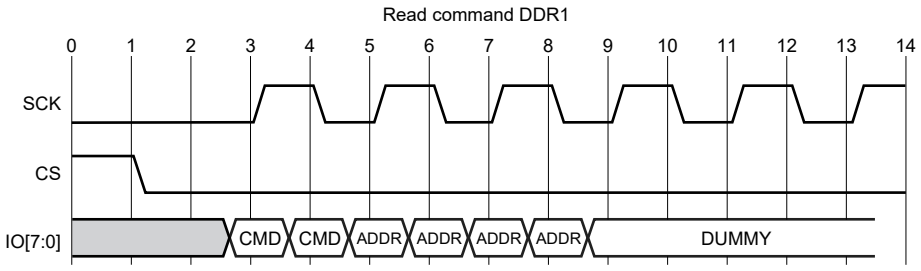


Figure 267. Read command DDR1

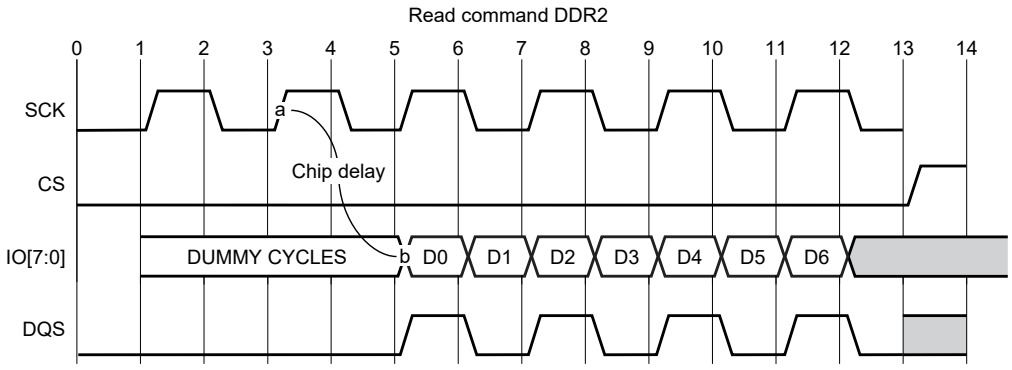


Figure 268. Read command DDR2

NOTE

For DDR×8 read, the start address must be an even address.

34.6.3 CS interval

You must maintain a minimum CS interval between each command to prevent two commands arriving at the SPI follower too closely together. During this period of time, the SPI follower must clean up all redundant data in each stage of the asynchronous FIFO to prepare a clean data path for the next command.

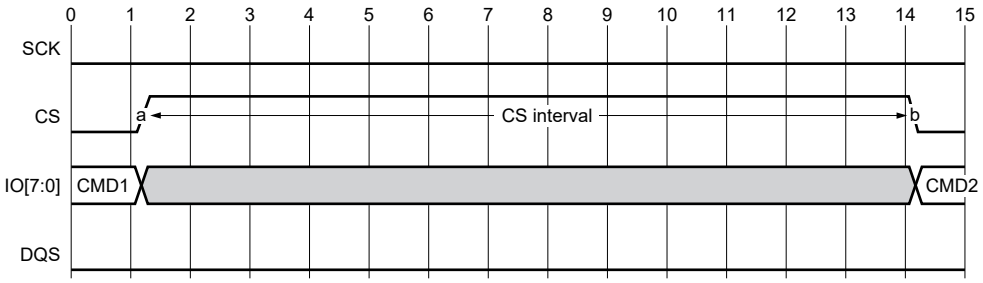


Figure 269. CS interval

34.6.4 TCSS and TCSH timing

TCSH is the period between when SCK stops toggling and CS deasserts. This period of time is necessary for FlexSPI_FLR to clean up the internal state from the previous transaction.

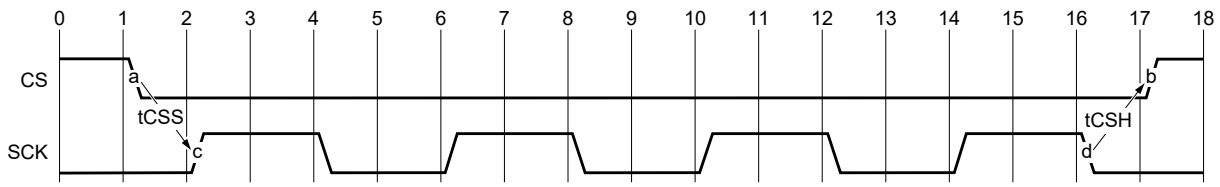


Figure 270. TCSS and TCSH timing

TCSS is the period between CS being asserted and SCK starting to toggle. This period of time is necessary for FlexSPI_FLR to initialize the block's internal IO logic.

34.6.5 Read DQS stop feature in SDR mode

Because of chip bus latency, if read data is not returned quickly enough, it is possible for I/O TX FIFO to be empty. In this case, the IO TX FIFO pop must be negated so that DQS does not toggle.

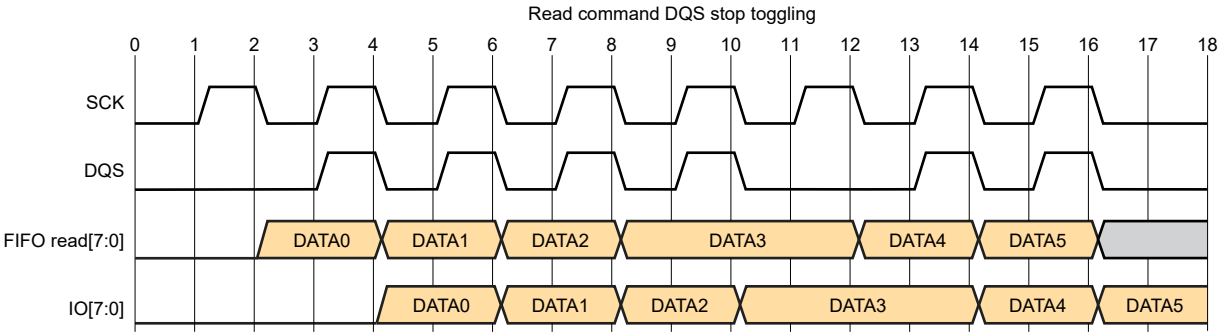


Figure 271. Read command DQS stop toggling

34.6.6 Write DQS mask in DDR mode

DQS working as mask is only supported in DDR×8 mode. In this mode, the start address and write data beat must always be even.

In the following figure, DATA0 and DATA7 are discarded.

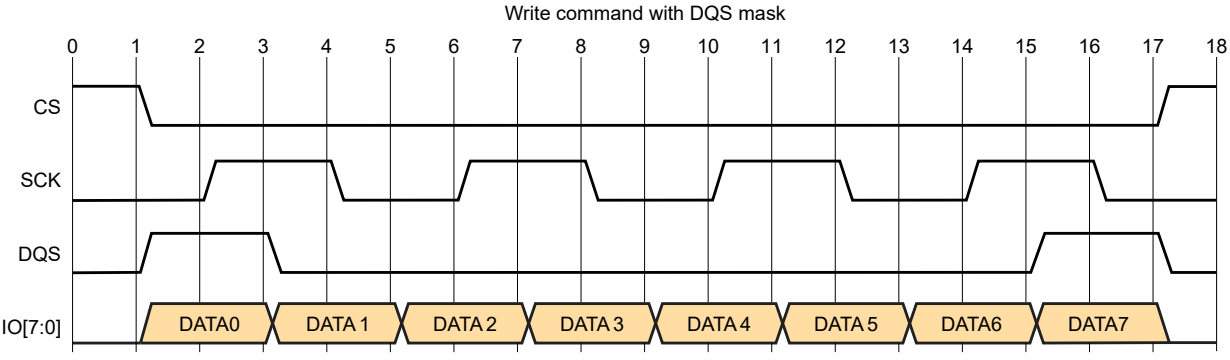


Figure 272. Write command with DQS mask

34.7 Memory map and register definition

34.7.1 FlexSPI_FLR register descriptions

34.7.1.1 FlexSPI_FLR memory map

FLEXSPI_SLV base address: 4290_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Control (MODULE_CONTROL)	32	RW	0000_0018h
4h	Read Command Control (READ_COMMAND_CONTROL)	32	RW	0000_0000h
8h	Read Register Command Setting (READ_REGISTER_COMMAND0)	32	RW	0101_0014h
Ch	Read Command 1 setting (READ_COMMAND1)	32	RW	0202_0014h
10h	Read Command 2 setting (READ_COMMAND2)	32	RW	0303_0014h
14h	Write Command Control (WRITE_COMMAND_CONTROL)	32	RW	0000_0002h
18h	Write Register Command 0 Setting (WRITE_REGISTER_COMMAND0)	32	RW	0404_0000h
1Ch	Write Command 1 Setting (WRITE_COMMAND1)	32	RW	0505_0000h
20h	Write Command 2 Setting (WRITE_COMMAND2)	32	RW	0606_0000h
24h	Read Write Command Address Base (RW_COMMAND_BASE)	32	RW	1000_0000h
28h	Command Suit 1 Range (CMD1_RANGE)	32	RW	0000_0000h
2Ch	Command Suit 2 Range (CMD2_RANGE)	32	RW	0000_0000h
30h	Module Status (MODULE_STATUS)	32	R	0000_300Eh
34h	SPI FLR interrupt (MODULE_INT)	32	RW	0000_0000h
38h	SPI FLR Interrupt Enable (MODULE_INTEN)	32	RW	0000_0000h
3Ch	SPI Mailbox control (SPI_MAIL_CTRL)	32	RW	0000_0000h
40h	SPI Mail Interrupt (SPIMAIL0)	32	R	0000_0000h
44h - 60h	SPI Mail Interrupt (SPIMAIL1 - SPIMAIL8)	32	R	0000_0000h

34.7.1.2 Module Control (MODULE_CONTROL)

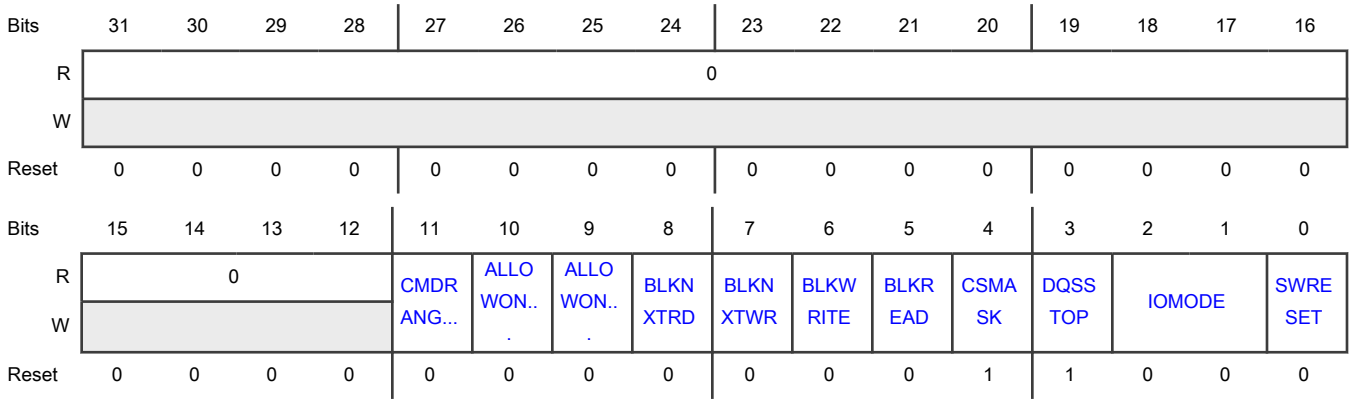
Offset

Register	Offset
MODULE_CONTROL	0h

Function

Contains miscellaneous control fields.

Diagram



Fields

Field	Function
31-12 —	Reserved
11 CMDRANGEBASEUPDATE	AXI Command Range Base Update After changing the AXI command's base address and range, set this field until it is self-cleared. 0b - Not updated 1b - Updated
10 ALLOWONERD	Allow One More Read Specifies whether one or more read command is allowed when BLKNXTRD is 1. You can use this field only when BLKNXTRD is 1. This field automatically becomes 0 if it is 1. 0b - Not allowed 1b - Allowed
9 ALLOWONEWR	Allow One More Write Specifies whether one or more write command is allowed when BLKNXTWR is 1. You can use this only when BLKNXTWR is 1. This field automatically becomes 0 if it is 1. 0b - Not allowed 1b - Allowed
8 BLKNXTRD	Block Next Read Specifies whether the next read command is allowed. If it is blocked, a read command returns all zeros. 0b - Allowed 1b - Blocked
7	Block Next Write Command Specifies whether the next write command is allowed. If not, all incoming write data will be discarded.

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLKNXTWR	0b - Allowed 1b - Blocked
6 BLKWRITE	Block Write Specifies whether the incoming write command is allowed. If this field is 1, all incoming write data is dropped. 0b - Allowed 1b - Blocked
5 BLKREAD	Block Read Specifies whether the incoming read commands are allowed. If not, return 0h to SPI bus. 0b - Allowed 1b - Blocked
4 CSMASK	Chip Select Mask Masks the chip select. You must write 0 to this field after I/O configuration is completed to avoid any mistriggering of internal logic. 0b - Not masked 1b - Masked
3 DQSSTOP	DQS Stop Feature Specifies that when you write 1 to this field during a read command, DQS stops toggling if internal FIFO has no read data to pop out. If this field is 0, the chip must ensure that no FIFO underflow happens. 0b - Disable 1b - Enable
2-1 IOMODE	SPI IO Mode Control 00b - SDR×4 01b - SDR×8 10b - DDR×4 11b - DDR×8
0 SWRESET	Software Reset Specifies whether to initiate software reset. Before writing 1, all state machines must be idle. 0b - Finished 1b - Initiate

34.7.1.3 Read Command Control (READ_COMMAND_CONTROL)

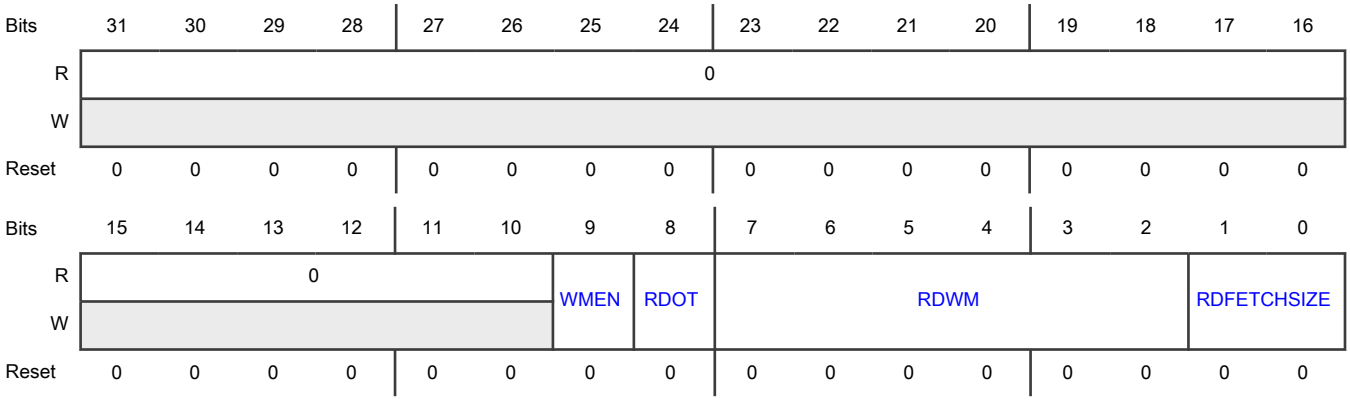
Offset

Register	Offset
READ_COMMAND_CONTROL	4h

Function

Contains read command control fields.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 WMEN	Read Water Mark Enable Specifies whether the read water mark is enabled. If this field is 1, then READ_COMMAND_CONTROL[RDWM] is valid. If this field is 0, then READ_COMMAND_CONTROL[RDWM] is invalid. 0b - Disable 1b - Enable
8 RDOT	Read Outstanding Contains 4 read leaders inside the block. When this field becomes 1, each leader issues a read request with a total size [RDFETCHSIZE]/4 per leader. If this field is 1, 4 leaders send out requests outstandingly, and [RDWM] is ignored. If this field is 0, 4 leaders send requests only after previous leaders finish their data fetch. 0b - Send requests after previous leaders finish 1b - Send requests outstandingly

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-2 RDWM	Read Watermark Level Triggers a new AXI read to fetch more data through the IP AXI header if two conditions are met: • The number of current valid items is equal to or smaller than RDWM. • The maximum read fetch size is not yet reached during the read command.
1-0 RDFETCHSIZE	Read Fetch Size Specifies the maximum read size triggered by a single read command. 00b - 256 bytes 01b - 512 bytes 10b - 1 KB 11b - 2 KB

34.7.1.4 Read Register Command Setting (READ_REGISTER_COMMAND0)

Offset

Register	Offset
READ_REGISTER_COMMAND0	8h

Function
Contains read register command settings.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COMMANDSET															
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DUMMYCYCLES															
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

Fields

Field	Function
31-16 COMMANDSET	Read Register Command Setting Contains values used for command decoding for incoming SPI requests. Use the higher 8 bits only in DDR×8 mode. The higher 8 bits of the 16-bit register on DDR×8 mode are for the rising edge of the SPI clock, and the lower 8 bits are for the falling edge. For example, if the command input is FDh for both rising and falling edges of the SPI clock in DDR×8 mode, you must write FDFDh to the command set.
15-0 DUMMYCYCLES	Read Register Dummy Cycles Specifies the dummy cycle setting for read register command.

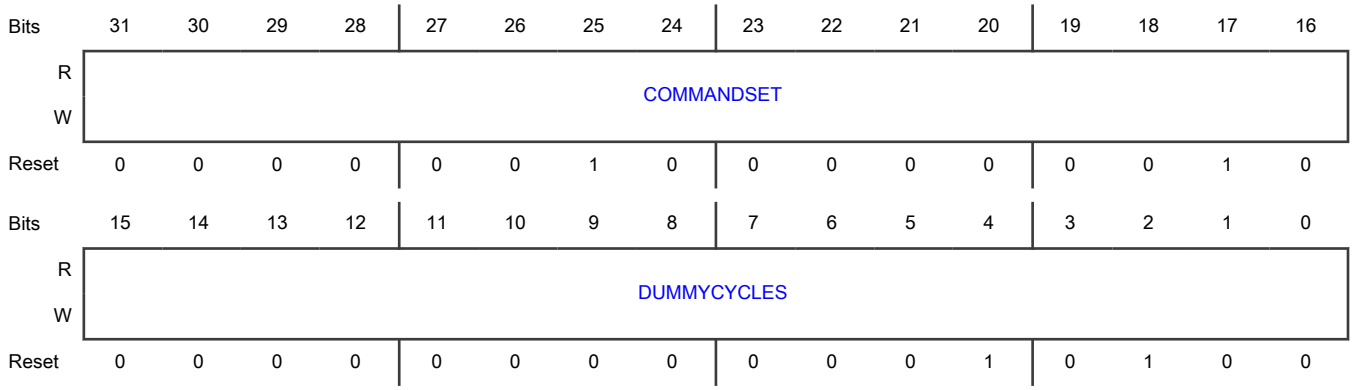
34.7.1.5 Read Command 1 setting (READ_COMMAND1)

Offset

Register	Offset
READ_COMMAND1	Ch

Function
Contains read command 1 settings.

Diagram



Fields

Field	Function
31-16 COMMANDSET	Read Command 1 Setting Contains value used for command decoding for incoming SPI requests. Use the higher 8 bits only in DDR×8 mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The higher 8 bits of the 16-bit register on DDR×8 mode are for the rising edge of the SPI clock, and the lower 8 bits are for the falling edge. For example, if the command input is FDh for both rising and falling edges of the SPI clock in DDR×8 mode, you must write FDFDh to the command set.
15-0 DUMMYCYCLES	Read Command 1 Dummy Cycles Specifies the dummy cycle setting for read command 1.

34.7.1.6 Read Command 2 setting (READ_COMMAND2)

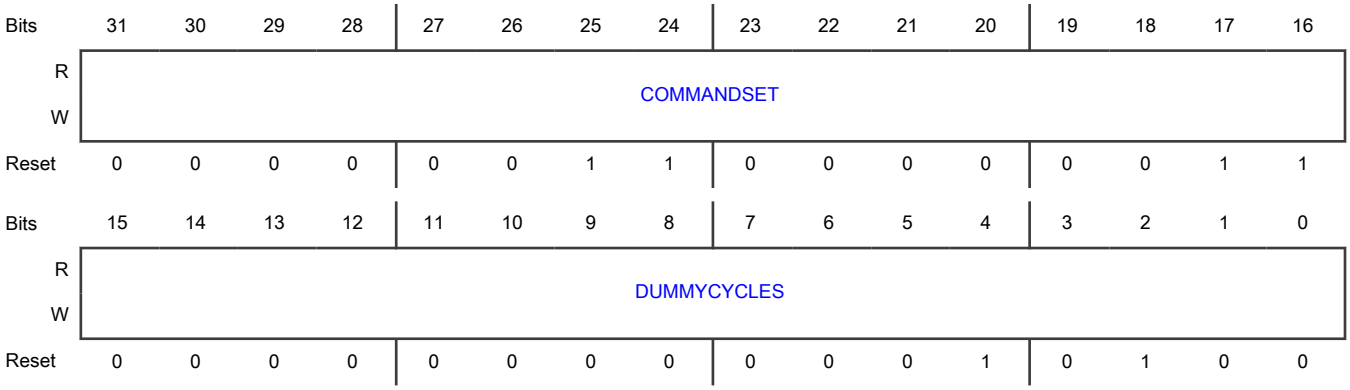
Offset

Register	Offset
READ_COMMAND2	10h

Function

Contains read command 2 setting.

Diagram



Fields

Field	Function
31-16 COMMANDSET	Read Command 2 Setting Contains value used for command decoding for incoming SPI request. Use the higher 8 bits only in DDR×8 mode. Note that the higher 8 bits of the 16-bit register on DDR×8 mode are for the rising edge of the SPI clock, and the lower 8 bits are for the falling edge of the SPI clock. For example, if the command input is FDh for both the rising and falling edges of the SPI clock in DDR×8 mode, you must write FDFDh to the command set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 DUMMYCYCLES	Read Command 2 Dummy Cycles Specifies the dummy cycle setting for read command 2.

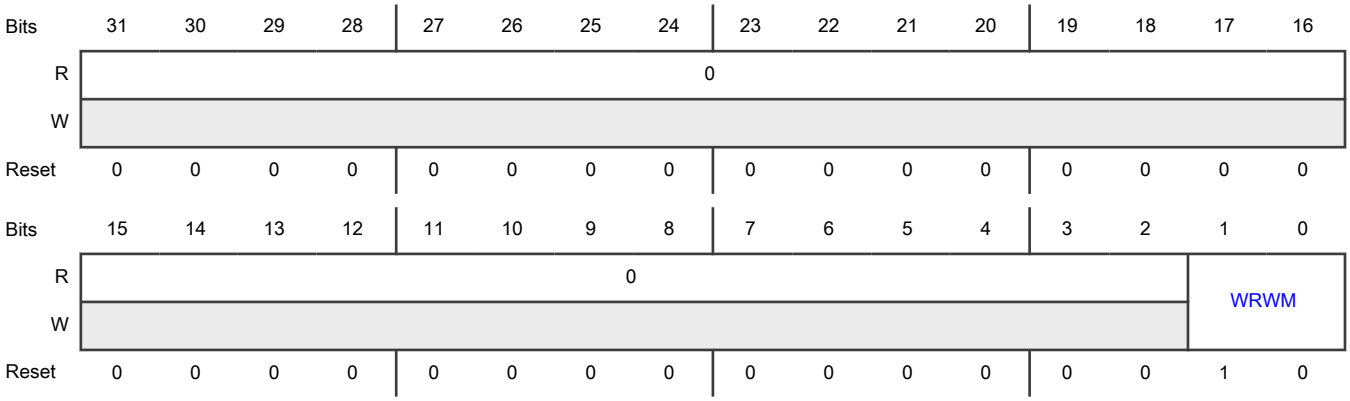
34.7.1.7 Write Command Control (WRITE_COMMAND_CONTROL)

Offset

Register	Offset
WRITE_COMMAND_CONTROL	14h

Function
Contains write command control information.

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 WRWM	Write Watermark Specifies the watermark value. During the write command, each one case below can trigger a new AXI write: • If pending write data equals or exceeds the watermark level. • If XSPI_FLR_CS is deasserted, even though pending write data is less than watermark level.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - 32 bytes 01b - 64 bytes 10b - 128 bytes 11b - 256 bytes

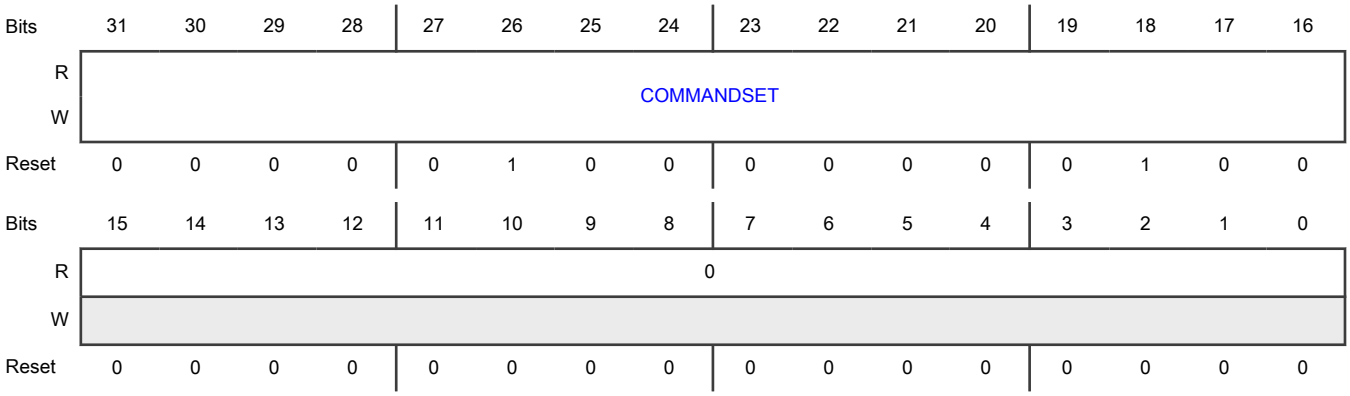
34.7.1.8 Write Register Command 0 Setting (WRITE_REGISTER_COMMAND0)

Offset

Register	Offset
WRITE_REGISTER_COMMAND0	18h

Function
Contains write register command 0 setting information.

Diagram



Fields

Field	Function
31-16 COMMANDSET	Write Register Command Setting Contains the value used for command decoding for incoming SPI request Use the higher 8 bits only in DDR×8 mode. The higher 8 bits of the 16-bit register on DDR×8 mode are for the rising edge of the SPI clock, and the lower 8 bits are for the falling edge of the SPI clock. For example, if the command input is FDh for both the rising and falling edges of the SPI clock in DDR×8 mode, you must write FDFDh to the command set.
15-0 —	Reserved

34.7.1.9 Write Command 1 Setting (WRITE_COMMAND1)

Offset

Register	Offset
WRITE_COMMAND1	1Ch

Function
Specifies write command 1 setting.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COMMANDSET															
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 COMMANDSET	Write Command 1 Setting Specifies the use of command decoding for incoming SPI request. Use the higher 8 bits only in DDR×8 mode. The higher 8 bits of the 16-bit register on DDR×8 mode are for the rising edge of the SPI clock, and the lower 8 bits are for the falling edge of the SPI clock. For example, if the command input is FDh for both the rising and falling edges of the SPI clock in DDR×8 mode, you must write FDFDh to the command set.
15-0 —	Reserved

34.7.1.10 Write Command 2 Setting (WRITE_COMMAND2)

Offset

Register	Offset
WRITE_COMMAND2	20h

Function
Specifies write command 2 setting.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COMMANDSET															
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 COMMANDSET	Write Command 2 Setting Specifies the use of command decoding for incoming SPI request. Use the higher 8 bits only in DDR×8 mode. The higher 8 bits of the 16-bit register on DDR×8 mode are for the rising edge of the SPI clock, and the lower 8 bits are for the falling edge of the SPI clock. For example, if the command input is FDh for both the rising and falling edges of the SPI clock in DDR×8 mode, you must write FDFDh to the command set.
15-0 —	Reserved

34.7.1.11 Read Write Command Address Base (RW_COMMAND_BASE)

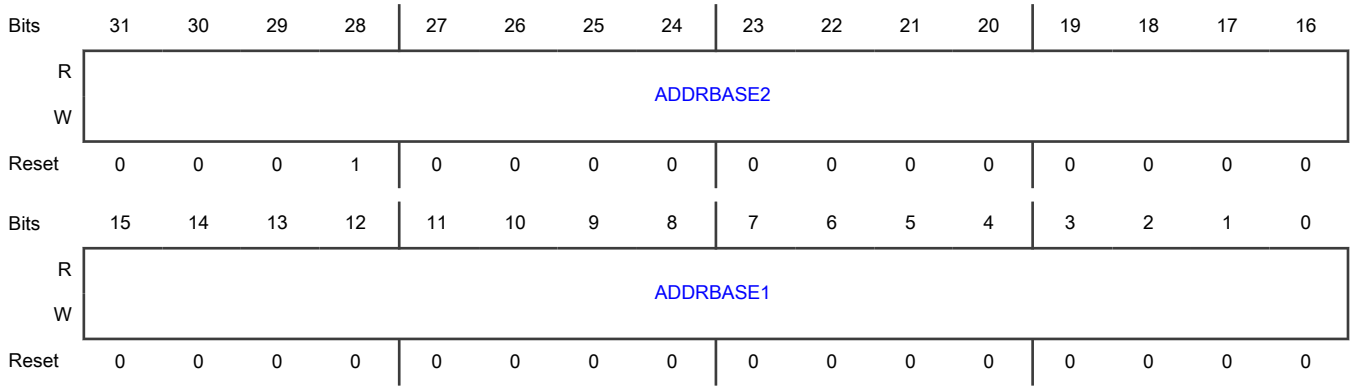
Offset

Register	Offset
RW_COMMAND_BASE	24h

Function

Contains the address base for read/write command 1 and 2.

Diagram



Fields

Field	Function
31-16 ADDRBASE2	Address Base 2 Contains the upper 16-bit address base added to any incoming SPI read or write request that is decoded as Read or Write command 2. You must ensure that the base address does not overlap with an incoming address.
15-0 ADDRBASE1	Address Base 1 Contains the upper 16-bit address base added to any incoming SPI read or write request that is decoded as Read or Write command 1. You must ensure that the base address does not overlap with an incoming address.

34.7.1.12 Command Suit 1 Range (CMD1_RANGE)

Offset

Register	Offset
CMD1_RANGE	28h

Function

Contains the read and write command 1 range. If a write exceeds this range, any additional data are dropped. If any read exceeds this range, no more read data beyond it are issued.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RANGE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RANGE								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 RANGE	Memory Range Contains the size of the memory range in 1 KB units.
9-0 —	Reserved

34.7.1.13 Command Suit 2 Range (CMD2_RANGE)

Offset

Register	Offset
CMD2_RANGE	2Ch

Function

Contains the read or write command 2 range. If a write exceeds this range, any additional data are dropped. If any read exceeds this range, no more read data beyond it are issued.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RANGE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RANGE								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 RANGE	Memory Range
9-0 —	Reserved

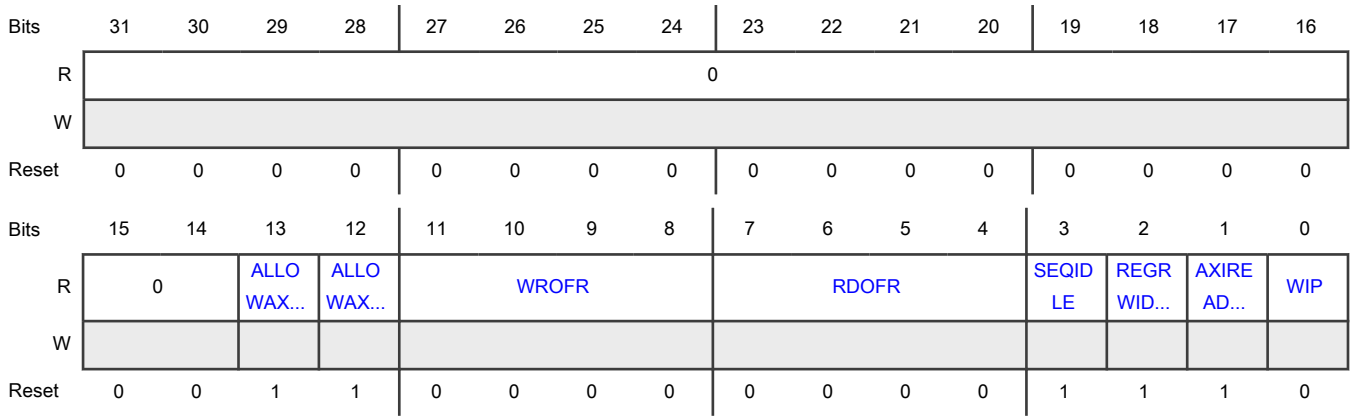
34.7.1.14 Module Status (MODULE_STATUS)

Offset

Register	Offset
MODULE_STATUS	30h

Function
Contains module status information.

Diagram



Fields

Field	Function
31-14 —	Reserved
13 ALLOWAXIWR	Allow AXI Write Access Indicates whether AXI write access is allowed. If this field is 1, no AXI write blocking feature is enabled. 0b - Denied 1b - Allowed

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 ALLOWAXIRD	Allow AXI Read Access Indicates whether AXI read access is allowed. If this field is 1, no AXI read blocking feature is enabled. 0b - Denied 1b - Allowed
11-8 WROFR	Write Out-of-Range Counter Contains the SPI leader write out-of-allowed-range counter.
7-4 RDOFR	Read Out-of-Range Counter Contains the SPI leader read out-of-allowed-range counter.
3 SEQIDLE	SEQ Controller Idle Indicates whether the SEQ control logic is idle or else busy with an ongoing SPI request. 0b - Busy 1b - Idle
2 REGRWIDLE	Register Read Write Idle Indicates whether the SPI to read/write register queue is idle. 0b - Busy 1b - Idle
1 AXIREADIDLE	AXI Read Leader Idle Indicates whether the AXI read leader is busy with a read request or else idle with no pending AXI read request. 0b - Busy 1b - Idle
0 WIP	Write in Progress Indicates whether the current AXI write leader is busy with a write command. 0b - Not busy 1b - Busy

34.7.1.15 SPI FLR interrupt (MODULE_INT)

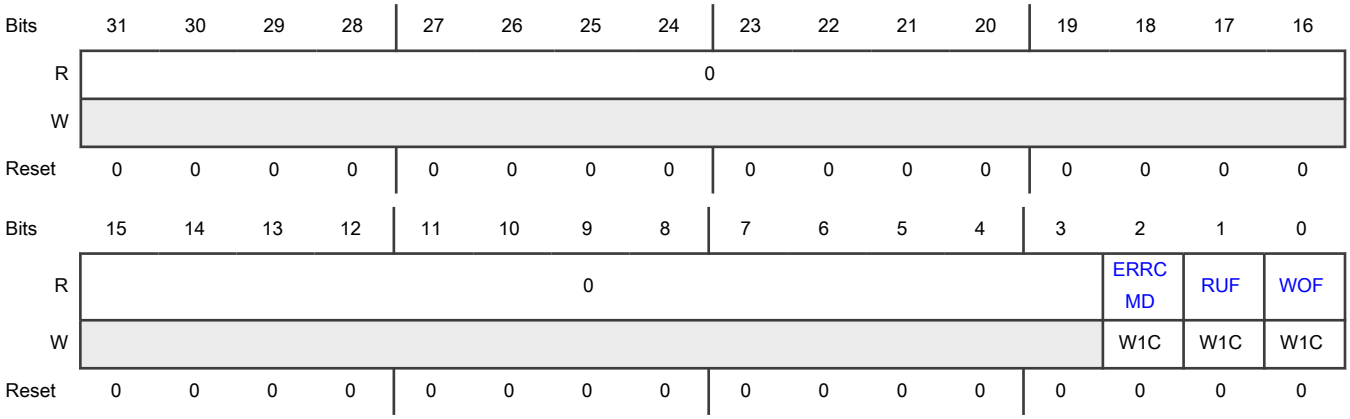
Offset

Register	Offset
MODULE_INT	34h

Function

Provides interrupt information.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 ERRCMD	Error Command Indicates whether an unknown command has been received from the SPI bus. 0b - Not received 1b - Received
1 RUF	Read Underflow Indicates whether IO TX FIFO underflow has occurred during a read command. If this field is 1, and If MODULE_CONTROL[DQSSTOP] = 1, then this interrupt indicates that a DQS stop occurred during a read command. 0b - Did not occur 1b - Occurred
0 WOF	Write Overflow Interrupt Indicates whether an IO RX FIFO overflow occurred during command/address/write data phase. 0b - Did not occur 1b - Occurred

34.7.1.16 SPI FLR Interrupt Enable (MODULE_INTEN)

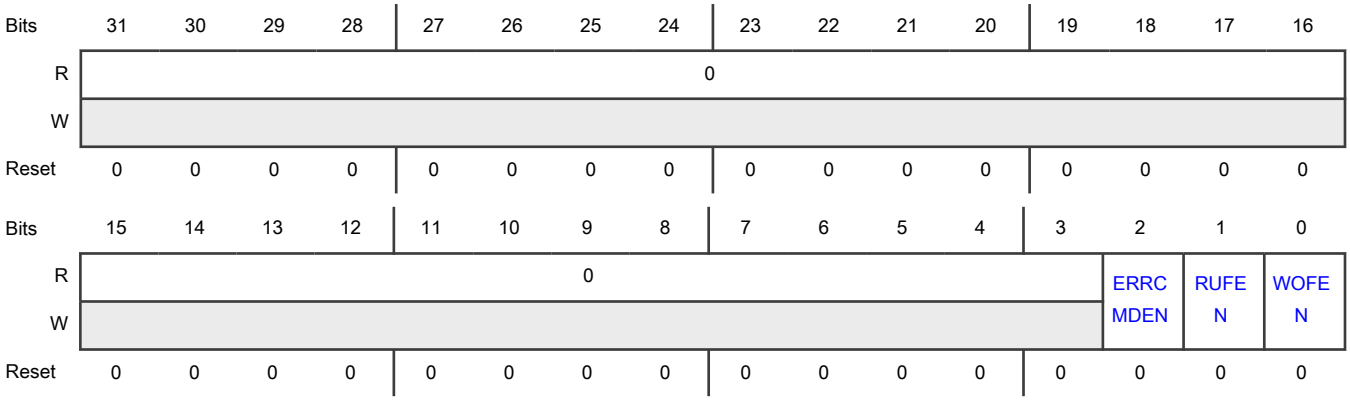
Offset

Register	Offset
MODULE_INTEN	38h

Function

Contains SPI FLR Interrupt enable information.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 ERRCMDEN	Error Command Interrupt Enable Enables the error command interrupt. 0b - Disable 1b - Enable
1 RUFEN	Read Underflow Interrupt Enable Enables the read underflow interrupt MODULE_INT[RUF]. 0b - Disable 1b - Enable
0 WOFEN	Write Overflow Interrupt Enable Enables the overflow interrupt MODULE_INT[WOF]. 0b - Disable 1b - Enable

34.7.1.17 SPI Mailbox control (SPI_MAIL_CTRL)

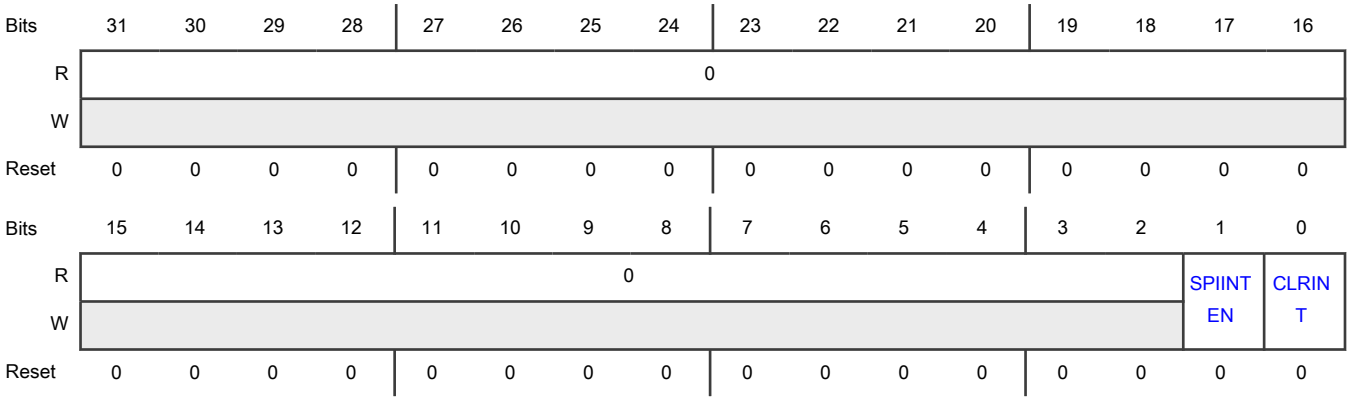
Offset

Register	Offset
SPI_MAIL_CTRL	3Ch

Function

Contains SPI mailbox control information.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 SPIINTEN	SPI Leader Interrupt Enable Specifies whether the SPI leader interrupt is enabled. 0b - Disable 1b - Enable
0 CLRINT	Clear Interrupt Clears the mailbox interrupt. 0b - Do not clear 1b - Clear

34.7.1.18 SPI Mail Interrupt (SPIMAIL0)

Offset

Register	Offset
SPIMAIL0	40h

Function

Generates an interrupt MAILn[0].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAILn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAILn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MAILn	Specifies the use of MAILn for Inter-Processor Communication (IPC) protocol for SPI leader and follower communication. MAILn[0] is used to generate an interrupt.

34.7.1.19 SPI Mail Interrupt (SPIMAIL1 - SPIMAIL8)

Offset

For n = 1 to 8:

Register	Offset
SPIMAILn	40h + (n × 4h)

Function

Specifies MAIL*n* when n = 1–8.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAILn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAILn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MAILn	Specifies the use of MAILn for Inter-Processor Communication (IPC) protocol for SPI leader and follower communication. MAILn[0] is used to generate an interrupt.

Chapter 35
ARM Cortex M7 Platform (M7)

35.1 Chip-specific M7 information

Table 265. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments
System Debug	-	System Debug Overview

The Arm Cortex-M7 core revision is CORTEX-M7-r1p2-00rel0. The CoreSight™ revision is CoreSight-SoC-TM100-r3p2. The list shows related Arm documents:

- DDI0403E_B_armv7m_arm.pdf
- DDI0489F_cortex_m7_r1p2_trm.pdf
- DUI0644E_cortex_m7_r1p2_ugrm.pdf
- DDI0480G_coresight_soc_r3p2_trm.pdf
- DUI0563G_coresight_soc_r3p2_ug.pdf

For this chip, the Cortex-M7 core configurations are listed in the following table.

Table 266. Cortex-M7 Core Configuration

Feature	Description	Configuration
FPU	Floating point unit (FPU) configuration	2
ICACHE	I-cache configuration	1
DCACHE	D-cache configuration	1
CACHEECC	Cache ECC configuration	1

Table continues on the next page...

Table 266. Cortex-M7 Core Configuration (continued)

MPU	MPU region configuration	16
IRQNUM	Number of IRQs	240
IRQLVL	IRQ priority width configuration	4
DBGLVL	Debug (breakpoint/watchpoint) configuration	1
TRC	Internal Trace support	1
ETM	ETM support	1
CTI	CTI support	1
WIC	WIC support	0
LOCKSTEP	Dual-redundant (lock-step) CPU functionality	0
RAR	Reset-all-registers	1
DW	Use Synopsys DesignWare	1
ICACHESIZE	I-cache Size (in k Bytes)	32
DCACHESIZE	D-cache Size (in k Bytes)	32

The Cortex-M7 provides access to ITCM and DTCM through a direct SRAM memory port. The address used for the I/DTCM are fixed within the Cortex-M7 IP core. These addresses are:

- ITCM => 0x0000_0000 (of size 0, 256kB, or 512kB)
- DTCM => 0x2000_0000 (of size 0, 256kB, or 512kB)

The system access from non-Cortex-M7 initiators to the TCM is mapped to address:

- ITCM => 0x2038_0000 or 0x3038_0000 (remapped to 0x0000_0000)
- DTCM => 0x2040_0000 or 0x3040_0000 (remapped to 0x2000_0000)

The valid range of these address ranges depends on the value of M7_CFG[TCM_SIZE].

35.2 Overview

Cortex-M7 platform features a single Arm® Cortex®-M7 processor in this chip. The Arm Cortex-M7 processor is a highly efficient, high-performance, embedded processor that features low interrupt latency, low-cost debug, and has backward compatibility with existing Cortex-M profile processors. The processor has an in-order super-scalar pipeline by which many instructions can be dual-issued, including load/load and load/store instruction pairs because of multiple memory interfaces.

Memory interfaces that the processor supports include:

- Tightly-Coupled Memory (TCM)
- Harvard instruction and data caches and AXI Master (AXIM) interface
- Dedicated low-latency AHB-Lite Peripheral (AHBP) interface

35.2.1 Block diagram

A block diagram for Cortex-M7 is given below:

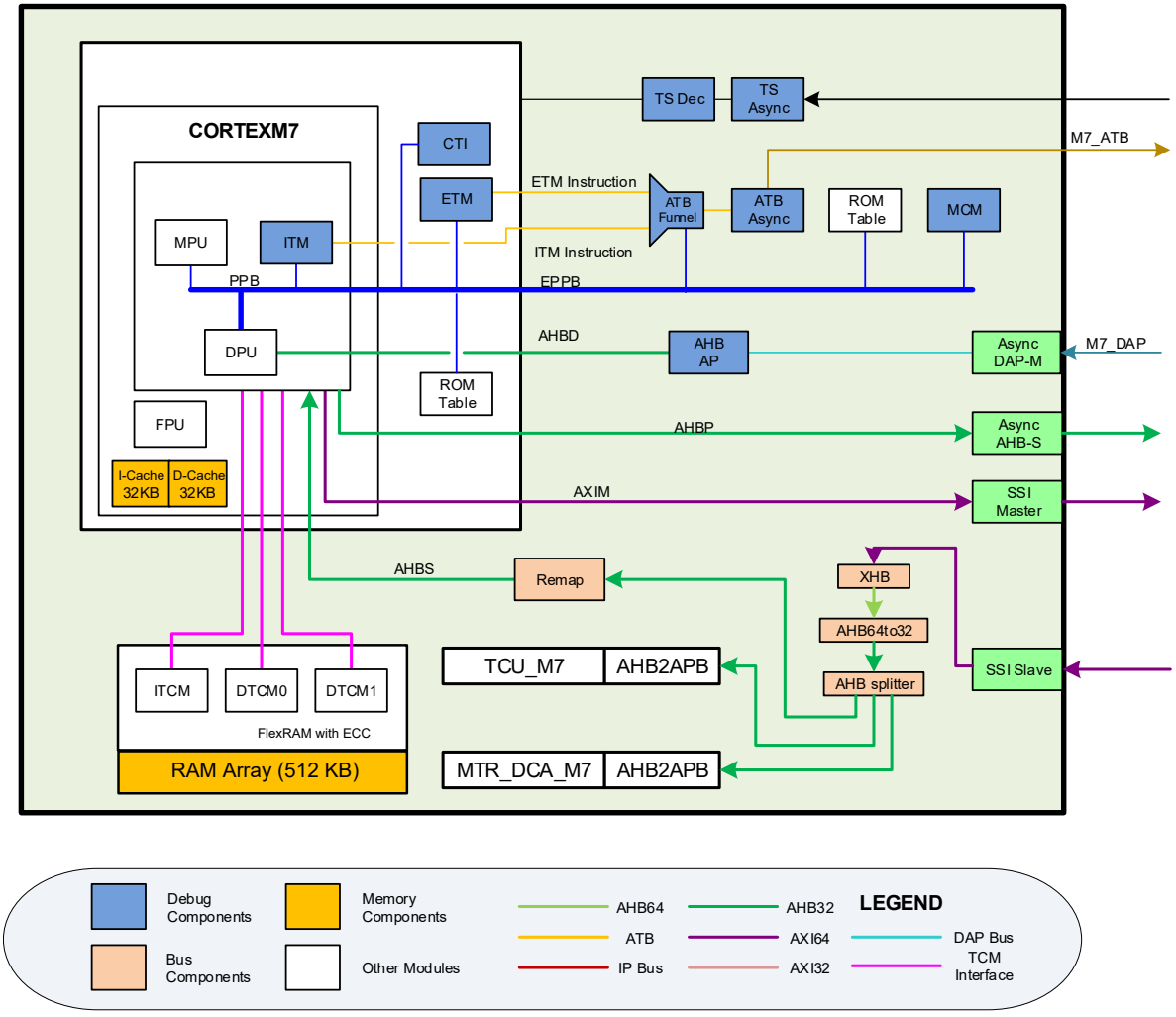


Figure 273. Cortex-M7 Platform Block Diagram

35.2.2 Features

Cortex-M7 platform in this chip has the following features.

- 1× Cortex-M7 processor: architecture Arm v7E-M.
- Floating Point Unit (FPU) FPv5 including single and double precision.
- Bus interface 64-bit AMBA4 AXI, 32-bit AHB peripheral port, 32-bit AMBA AHB slave port, AMBA APB interface for CoreSight debug components.
- 32 KB instruction / 32 KB data L1 caches with ECC.
- Configurable size instruction and data TCM (default 256 KB ITCM / 256 KB DTCM) with ECC interface.
- 16 regions of MPU.
- Non-maskable Interrupt (NMI) + 240 NVIC IRQs.
- 16 interrupt priority levels (4-bits).
- Sleep (WAIT) and deep sleep (STOP) power modes with integrated WFI, WFE, SLEEPONEXIT.
- SWD and JTAG, with up to 8 breakpoints and 4 watchpoints.

35.3 Functional description

35.3.1 CM7_MCM sub-module

CM7_MCM is a miscellaneous control module for ARM Cortex-M7. It is connected onto the Cortex-M7 PPB bus. It mainly contains programming registers to control the ECC features for CM7 ITCM and DTCM (CM7_ECC_MCM registers). It also has registers for Cortex-M7 FPU exceptions (CM7_MCM registers).

35.3.2 Tightly-Coupled Memory (TCM)

The Cortex-M7 uses TCM to perform low-latency memory operations. There are two TCM memory regions called ITCM and DTCM. The ITCM interface has a 64-bit data width and DTCM interface has a 32-bit data width.

The project supports total 512 KB TCM. By default, the ITCM and DTCM are 256 KB each. The valid address ranges of those memories depend on the value of M7_CFG[TCM_SIZE]. See M7_CFG register in BLK_CTRL_S_AONMIX chapter in the chip's Security Reference Manual. M7_CFG[TCM_SIZE] is a 3-bit field with 5 valid encodings:

- 000b - Regular TCM, 256 KB ITCM and 256 KB DTCM
- 001b - Double ITCM, 512 KB ITCM
- 010b - Double DTCM, 512 KB DTCM
- 100b - HALF ITCM, 128 KB ITCM and 384 KB DTCM
- 101b - HALF DTCM, 384 KB ITCM and 128 KB DTCM
- 011b - Reserved
- 110b - Reserved
- 111b - Reserved

When the TCM access is out of its register defined space, then it gets hard fault interrupt for illegal access indication. For example, when the DTCM size is defined to 384 KB size, then the read access on CM7 DTCM gets error response if the address is out of 384 KB space.

NOTE

For CM7 core, the base address of each TCM is fixed:

- ITCM at 0x00000000.
- DTCM at 0x20000000.

For the system access from non-Cortex-M7 initiators to the TCM, the address is remapped. The ITCM and DTCM regions are always in continuous totally 512 KB region, but based on different TCM size configuration, the address has different mapping, only the address defined in the valid 512 KB region is legal, the access out of this 512 KB region are illegal access, and it causes error response by hard fault indication if accessed. For example with following memory map table from CM33 core view, there are following several different memory remaps:

Table 267. CM7 TCM Memory Map Example (CM33 View)

Alias	Start Address(hex) (Non-Secure)	End Address(hex) (Non-Secure)	Start Address (hex) (Secure)	End Address(hex) (Secure)
M7 ITCM	20380000	203BFFFF	30380000	303BFFFF
M7 ITCM	203C0000	203FFFFF	303C0000	303FFFFF
M7 DTCM	20400000	2043FFFF	30400000	3043FFFF
M7 DTCM	20440000	2047FFFF	30440000	3047FFFF

When TCM is configured to 256 KB ITCM and 256 KB DTCM, the ITCM base and DTCM base address are remapped as following mapping:

- ITCM => 0x203C_0000 or 0x303C_0000 (remapped to 0x0000_0000)
- DTCM => 0x2040_0000 or 0x3040_0000 (remapped to 0x2000_0000)

When TCM is configured to 128 KB ITCM and 384 KB DTCM, the ITCM base and DTCM base address are remapped as following mapping:

- ITCM => 0x203E_0000 or 0x303E_0000 (remapped to 0x0000_0000)
- DTCM => 0x2040_0000 or 0x3040_0000 (remapped to 0x2000_0000)

When TCM is configured to 384 KB ITCM and 128 KB DTCM, the ITCM base and DTCM base address are remapped as following mapping:

- ITCM => 0x203A_0000 or 0x303A_0000 (remapped to 0x0000_0000)
- DTCM => 0x2040_0000 or 0x3040_0000 (remapped to 0x2000_0000)

When TCM is configured to 512 KB ITCM and no DTCM, the ITCM base address is remapped as following mapping and since no DTCM is defined, the read access on all address from 0x2040_0000 to 0x2047_FFFF gets error response.

- ITCM => 0x2038_0000 or 0x3038_0000 (remapped to 0x0000_0000)

When TCM is configured to 512 KB DTCM and no ITCM, the DTCM base address is remapped as following mapping and since no ITCM is defined, the read access on all address from 0x2038_0000 to 0x203F_FFFF gets error response.

- DTCM => 0x2040_0000 or 0x3040_0000 (remapped to 0x2000_0000)

In the project, The Cortex-M7 also supports ECC function with Double Error Detection and Single Error Correction (SEC-DED) to the TCM memories. Where, the following features are also supported:

- ECC errors detected are capable of being reported.
- ECC errors are capable of being injected and tested.

  For the details how to inject ECC error and report error, see CM7_ECC_MCM chapter.

35.3.3 Cache

The Cortex-M7 uses the instruction and data caches to provide fast access to frequently accessed data and instructions, that support increased average performance when using system-based memory. Cortex-M7 processor supports following feature on Cache function:

- Two-way set-associative 32 KB Instruction Cache.
- Four-way set-associative 32 KB Data Cache.
- ECC supports for both Instruction Cache and Data Cache with Double Error Detection and Single Error Correction (SEC-DED) on following cache memory region:
  - Instruction cache data RAM.
  - Instruction cache tag RAM.
  - Data cache data RAM.
  - Data cache tag RAM.

For Cache ECC function, it can also support the following features:

- ECC errors detected are capable of being reported.
- ECC errors are capable of being injected and tested.

For the detailed information on how to inject and report ECC error on cache, see Error Injection Module (EIM) and Error Reporting Module (ERM) chapters.

35.3.4 Memory Protection Unit (MPU)

The Cortex-M7 platform supports Memory Protection Unit (MPU). The MPU divides the memory map into a number of regions, and defines the location, size, access permissions, and memory attributes of each region. It supports:

- Independent attribute settings for each region.
- Overlapping regions.
- Export of memory attributes to the system.

The memory attributes affect the behavior of memory accesses to the region. The Cortex-M7 MPU defines up to 16 separate memory regions.

The Cortex-M7 MPU memory map is unified. This means instruction accesses and data accesses have same region settings. If a program accesses a memory location that is prohibited by the MPU, the processor generates a MemManage fault.

For the detailed information about MPU, see "Memory Protection Unit" chapter in DUI0644E_cortex_m7_r1p2_ugrm.pdf.

35.3.5 Clocking

Table 268. Arm Cortex-M7 Clocks

Clock Name	Description
forerun_clk	Clock for trout.ref_clk
osc_24m_clk	Clock for CMU.ref_clk
ccm_m7_clk_root	Clock for arm core / TCM / Cache
ccm_m7_bus_clk_root	Clock for m7 peripherals
ccm_m7_stclk_root	Used as cm7.STCLKEN

Table continues on the next page...

Table 268. Arm Cortex-M7 Clocks (continued)

Clock Name	Description
ssi_ahbs_ssi_fwd_clk	Clock for ssi slave forward direction clock
ssi_ahbs_ssi_rev_clk	Clock for ssi slave reverse direction clock
ssi_axim_ssi_fwd_clk	Clock for ssi master forward direction clock
ssi_axim_ssi_rev_clk	Clock for ssi master reverse direction clock

35.3.6 Reset

CM7_MCM uses reset of Cortex-M7. Cold or soft resetting the CM7 core would also reset this module.

35.3.7 Interrupts

CM7_MCM has three interrupt requests: mcm_irq, tcm_ecc1bit_irq and tcm_ecc2bit_irq.

The mcm_irq asserts when any of the interrupts in register ISCR occurs.

The tcm_ecc1bit_irq asserts when a single bit ECC error is detected on TCM including ITCM, D0TCM and D1TCM and the corresponding interrupt is enabled.

The tcm_ecc2bit_irq asserts when a multi-bit ECC error is detected on TCM including ITCM, D0TCM and D1TCM and the corresponding interrupt is enabled.

35.4 Memory Map and register definition

This section includes CM7_MCM memory map and detailed description of all registers.

35.4.1 CM7_MCM register descriptions

35.4.1.1 CM7_MCM memory map

M7_MCM base address: E008_0000h

Offset	Register	Width (In bits)	Access	Reset value
10h	Interrupt Status and Control Register (ISCR)	32	RW	0000_0000h
14h	SysTick Calibration Register (CFGSTCALIB)	32	RW	0100_0000h
18h	Process ID (PID)	32	RW	0000_0000h

35.4.1.2 Interrupt Status and Control Register (ISCR)

Offset

Register	Offset
ISCR	10h

Function

Defines the configuration and reports the status for a number of core-related interrupt exception conditions. It includes the enable and status bits associated with the core's floating-point exceptions, and bus errors on TCM. The individual event indicators are first qualified with their exception enables and then logically summed to form an interrupt request sent to the core's NVIC. Bits 15-8 are read-only indicator flags based on the processor's FPSCR register. Attempted writes to these bits are ignored. Once set, the flags remain asserted until software clears the corresponding FPSCR bit.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FIDCE	0		FIXCE	FUFCE	FOFCE	FDZCE	FIOCE	0		WABE	0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIDC	0		FIXC	FUFC	FOFC	FDZC	FIOC	0	WABSO	WABS	0				
W											W1C					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 FIDCE	FPU Input Denormal Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
30-29 —	Reserved
28 FIXCE	FPU Inexact Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
27 FUFCE	FPU Underflow Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
26 FOFCE	FPU Overflow Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
25 FDZCE	FPU Divide-by-Zero Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 FIOCE	FPU Invalid Operation Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
23-22 —	Reserved
21 WABE	TCM Write Abort Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
20-16 —	Reserved
15 FIDC	FPU Input Denormal Interrupt Status This read-only bit is a copy of the core's FPSCR[IDC] bit and signals input denormalized number has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IDC] bit. 0b - No interrupt 1b - Interrupt occurred
14-13 —	Reserved
12 FIXC	FPU Inexact Interrupt Status This read-only bit is a copy of the core's FPSCR[IXC] bit and signals an inexact number has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IXC] bit. 0b - No interrupt 1b - Interrupt occurred
11 FUFC	FPU Underflow Interrupt Status This read-only bit is a copy of the core's FPSCR[UFC] bit and signals an underflow has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[UFC] bit. 0b - No interrupt 1b - Interrupt occurred
10 FOFC	FPU Overflow Interrupt Status This read-only bit is a copy of the core's FPSCR[OFC] bit and signals an overflow has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[OFC] bit. 0b - No interrupt 1b - Interrupt occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 FDZC	FPU Divide-by-Zero Interrupt Status This read-only bit is a copy of the core's FPSCR[DZC] bit and signals a divide-by-zero operation has been detected in the processor's FPU. Once set, this bit remains set until software clears FPSCR[DZC] bit. 0b - No interrupt 1b - Interrupt occurred
8 FIOC	FPU Invalid Operation interrupt Status This read-only bit is a copy of the core's FPSCR[IOC] bit and signals an illegal operation has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IOC] bit. 0b - No interrupt 1b - Interrupt occurred
7 —	Reserved
6 WABSO	Write Abort on Slave Overrun Indicates if another write abort is detected before system software has retrieved all the information from the original event. When the WABS is cleared, this bit is also cleared. 0b - No write abort overrun 1b - Write abort overrun occurred
5 WABS	Write Abort on Slave Indicates when a write abort has occurred on AHBS interface. 0b - No abort 1b - Abort
4-0 —	Reserved

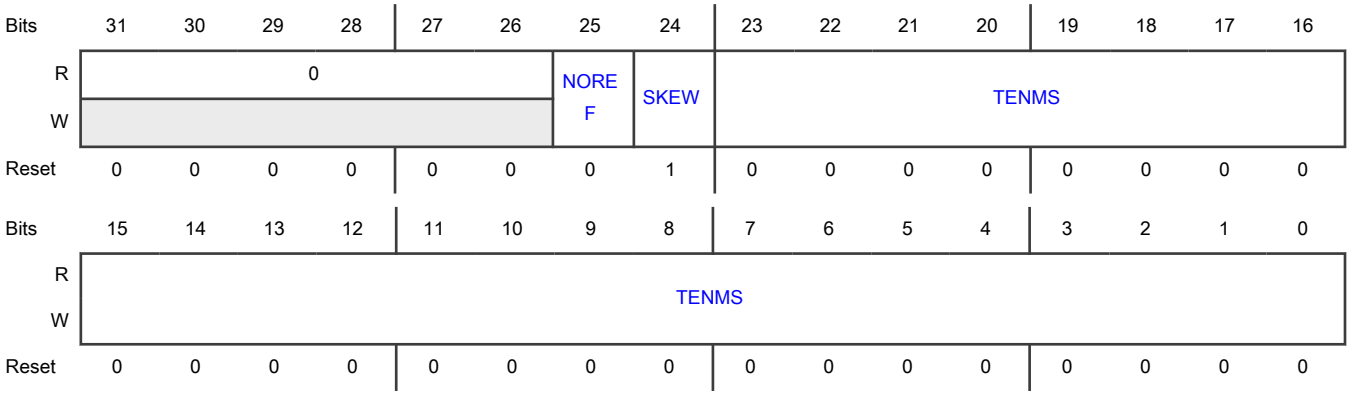
35.4.1.3 SysTick Calibration Register (CFGSTCALIB)

Offset

Register	Offset
CFGSTCALIB	14h

Function
Configures the CM7 SysTick's calibration.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 NOREF	No Reference Clock Indicates that no alternative reference clock source has been integrated in this chip.
24 SKEW	Clock Skew Program this field to 0 if the system timer clock can guarantee an exact multiple of 10ms. Otherwise, write this bit to 1. 0b - No clock skew 1b - Having clock skew
23-0 TENMS	Ten milliseconds Provides an integer value to compute a 10ms (100Hz) delay from the SysTick's clock. For an example, apply the value 0x07A11F if the clock is 50MHz. Apply 0x3E7 if the clock is 100KHz.

35.4.1.4 Process ID (PID)

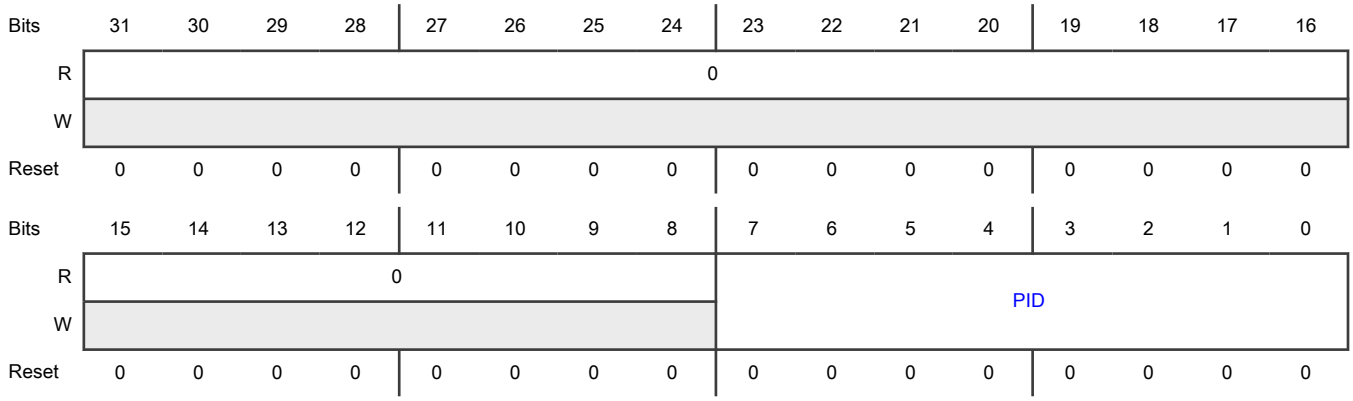
Offset

Register	Offset
PID	18h

Function

Provides a way for software to write an ID for different processes. Whenever entering a new process, software like OS(Operating System) can write this register to set an ID for the process. The XRDC or TRDC on this chip can use the process ID to control the permission of the process, determining what memories/peripherals can be or cannot be accessed by the specific process.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 PID	Process ID

Chapter 36
ARM Cortex CM33 Platform (CM33)

36.1 Chip-specific CM33 information

Table 269. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments
System Debug	-	System Debug Overview

Related Arm Documents:

- Arm Cortex M33 Technical Reference Manual, Revision r1p0
- Arm Cortex M33 Devices Generic User Guide, Revision r1p0
- Arm Cortex M33 Processor Datasheet

36.1.1 CM33 Configuration

For this chip, the Cortex-CM33 core configurations are listed in the following table.

Table 270. Cortex-CM33 Core Configuration

Feature	Description	Configuration
FPU	Floating point unit (FPU) configuration	1
DSP	DSP Extensions included	1
SECEXT	Security Extensions included	1
CPIF	Coprocessor interface included (set of 1 if PowerQuad is used)	0
MPU_NS	Non-secure MPU region configuration (CM33 supports 0, 4, 8, 12, 16)	16

Table continues on the next page...

Table 270. Cortex-CM33 Core Configuration (continued)

MPU_S	Secure MPU region configuration	Same as MPU_NS
SAU	Number of Security Attribution Unit regions	8
NUMIRQ	Number of IRQs (rounded to a multiple of 32)	256
IRQLVL	IRQ priority width configuration	4
IRQLATENCY	Individual interrupts with lowest interrupt latency (1=low latency)	X'b1
IRQDIS	Individual interrupts disabled	X'b0
DBGLVL	Debug (breakpoint/watchpoint) configuration	2
ITM	Internal Trace support	1
ETM	ETM support	1
MTB	Micro Trace Buffer trace	1
MTBAWIDTH	MTB RAM interface address width (5-12)	32
WIC	WIC support	1
WICLINES	-	NUMIRQ + 3
CTI	CTI support	1
RAR	Reset-all-registers	1

36.2 Overview

CM33 implements the Armv8-M architecture with main extensions. The Arm Cortex-M33 is a high-efficiency processor with security considerations. CM33 includes:

- A single-precision floating-point unit (FPU)
- A nested vectored interrupt controller (NVIC)
- A flash patch breakpoint (FPB)
- The data watchpoint and trace (DWT) unit
- An instrumentation trace macrocell (ITM)

As shown in Figure 274, the CM33 platform also includes:

- Miscellaneous control module CM33_TCM_MCM, which contains programming registers to control the ECC features for CM33 code TCM and system TCM
- Miscellaneous control module CM33_CACHE_ECC_MCM, which contains programming registers to control the ECC features for TAG0, TAG1, DATA0, and DATA1 in the CM33 code cache and system cache

36.2.1 Block diagram

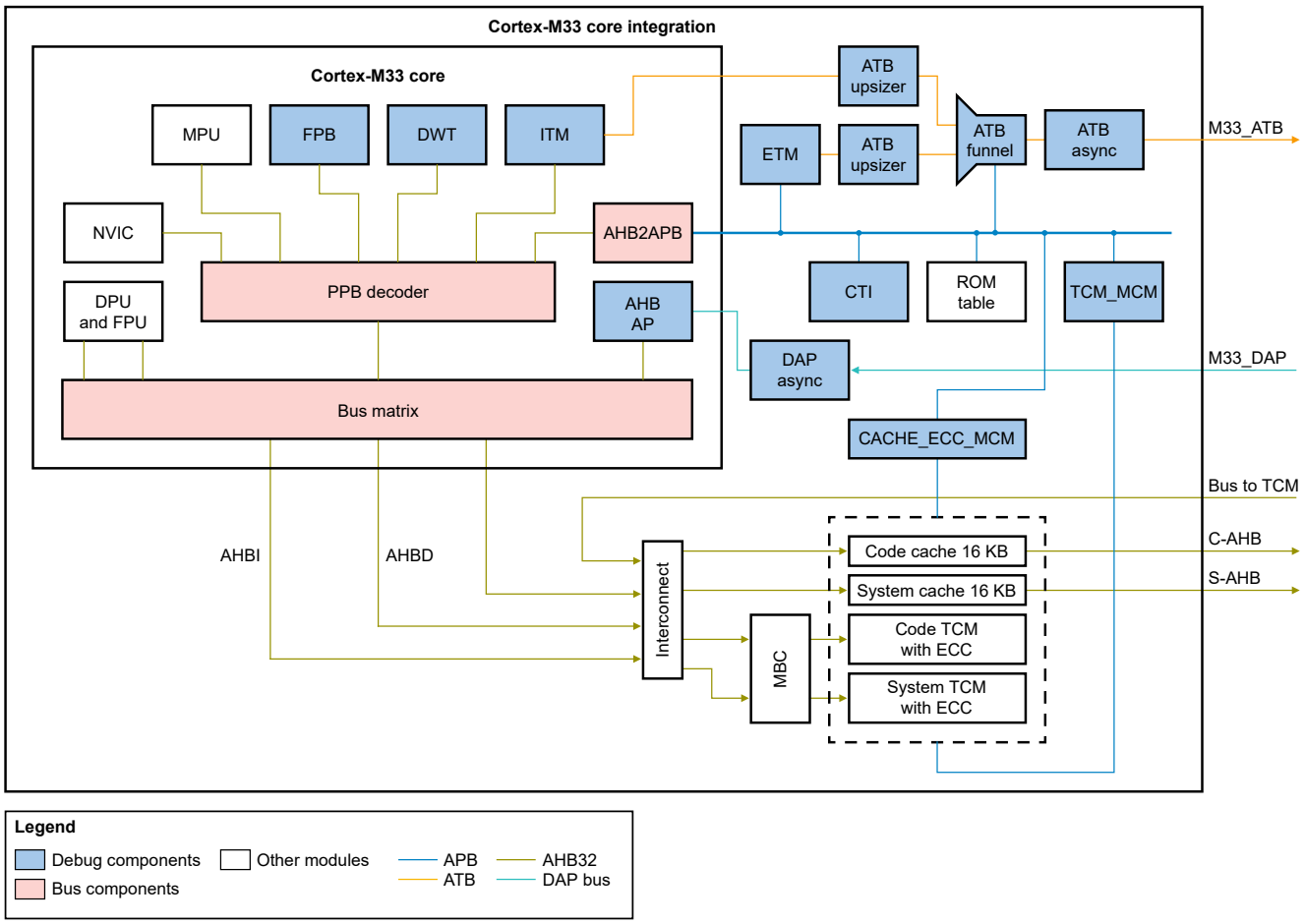


Figure 274. Block diagram

36.2.2 Features

- Cortex-M33 processor
- Single-precision FPU
- NVIC support
- FPB support (does not include patching)
- DWT and ITM support
- ETM support
- Two-way set-associative 16-KB system cache
- Two-way set-associative 16-KB code cache
- Configurable 256-KB tightly coupled memory (TCM):
  - 128-KB system TCM on system bus
  - 128-KB code TCM on code bus
- TCM ECC support
- Cache ECC support

- DSP extensions
- Security extensions
- Nonsecure memory protection unit (MPU) with 16 regions
- Secure MPU with 16 regions
- Cross trigger interface (CTI) support

The AON domain, which incorporates the Cortex-M33 core, includes one messaging unit (MU) and one hardware semaphore (SEMA42) for inter-core communication with the Cortex-M33 core.

For the Cortex-M33 core, the CM33 platform also supports execution in place (XIP) from FlexSPI to facilitate fast power-on wake-up. The Cortex-M33 debug interface is connected to the DAP and CoreSight debug modules in the chip to allow debugging through the JTAG interface. See the "System Debug" chapter for more details.

36.3 Functional description

The following sections contain a high-level overview of the CM33 components.

36.3.1 Cortex-M33 processor

The Cortex-M33 processor is based on the Armv8-M architecture and has a low gate count and high energy efficiency.

36.3.2 NVIC

NVIC supports 6 priority levels for interrupts. The NVIC IPR registers define 4 bits per IRQ.

36.3.3 FPU

The FPU implements FPv5 floating-point extensions, which support these single-precision operations: add, subtract, multiply, divide, multiply and accumulate, and square root. The FPU also provides conversions between fixed-point and floating point data formats, and it provides floating-point constant instructions.

The FPU is compliant with ANSI/IEEE Std 754-2008.

36.3.4 MPU

The MPU has independent attribute settings for each region and can export memory attributes.

CM33 also supports a secure MPU.

When a core access hits an overlapping memory region, the processor generates a fault.

Instruction accesses and data accesses in a given region have the same region settings.

36.3.5 Debug

See the "System Debug" chapter for more information about the CM33 platform debug components.

36.3.6 Code and system TCM

CM33 accesses its code TCM and system TCM through its C-AHB and S-AHB buses, respectively.

By default, the code TCM and system TCM are 128 KB each. The BLK_CTRL_S_AON register field M33_CFG[TCM_SIZE] is a 2-bit field with three valid encodings:

- 00 (default) = REGULAR_TCM: 50% code TCM and 50% system TCM
- 01 = DOUBLE_CODE_TCM: 100% code TCM and 0% system TCM
- 10 = DOUBLE_SYS_TCM: 100% system TCM and 0% code TCM
- 11 = Reserved

NOTE

To avoid unexpected software issues, perform writes to BLK_CTRL_S_AON.M33_CFG[TCM_SIZE] only when the Cortex-M33 core is in reset.

Code TCM is mapped to 0FFE0000h (nonsecure) and 1FFE0000h (secure) (plus the additional 128 KB below this if M33_CFG[TCM_SIZE] = DOUBLE_CODE_TCM)

System TCM is mapped to 20000000h (nonsecure) and 30000000h (secure) (plus the additional 128 KB above this address if M33_CFG[TCM_SIZE] = DOUBLE_DATA_TCM)

The system bus can access both TCM memories as a continuous address space via their aliases:

- Code TCM is mapped to the aliased code TCM base address (plus the additional 128 KB below this if TCM_SIZE = DOUBLE_CODE_TCM)
- System TCM is mapped to the aliased system base address (plus the additional 128 KB above this address if TCM_SIZE = DOUBLE_DATA_TCM)

See the chip's memory map information for aliased code and system TCM base address details.

The valid range of these address ranges depends on the value of BLK_CTRL_S_AON.M33_CFG[TCM_SIZE].

See the M33_CFG register in the BLK_CTRL_S_AON chapter for more information about the CM33 and TCM configuration options.

36.3.6.1 CM33_TCM_MCM

CM33_TCM_MCM is a miscellaneous control module for the Cortex-M33 core. It contains programming registers to control the ECC features of CM33 TCM.

36.3.6.2 CM33 TCM ECC support

You can use the syndrome value for single-bit ECC to locate the error bit in the CM33 system and code TCM. See the following lookup table for these values.

Table 271. Lookup table for 32-bit data single-bit ECC syndrome

Syndrome value	Error bit
07h	31
7Fh	30
6Eh	29
6Dh	28
6Bh	27
67h	26
5Eh	25
5Dh	24
5Bh	23
57h	22
4Fh	21
7Ch	20
7Ah	19

Table continues on the next page...

Table 271. Lookup table for 32-bit data single-bit ECC syndrome (continued)

Syndrome value	Error bit
79h	18
76h	17
75h	16
73h	15
70h	14
68h	13
64h	12
62h	11
61h	10
58h	9
54h	8
52h	7
51h	6
4Ch	5
4Ah	4
49h	3
46h	2
45h	1
43h	0

NOTE

- You cannot use the syndrome value for multibit ECC to locate the error bit in the CM33 system and code TCM.
- You cannot use the syndrome value for single-bit and multibit ECC to locate the error bit in the CM33 system and code cache.

36.3.7 CM33 code and system cache

The following caches are included in the CM33 platform:

- Two-way set-associative 16-KB code cache with ECC and write path error injection support
- Two-way set-associative 16-KB system cache with ECC and write path error injection support

36.3.7.1 CM33_CACHE_ECC_MCM

CM33_CACHE_ECC_MCM is a miscellaneous control module for the Arm Cortex-M33 cache ECC. It contains programming registers to control the ECC features of CM33 TAG0, TAG1, DATA0, and DATA1.

36.3.8 Clocking

See the chip's clocking sections for more information on CM33 clocks.

Table 272. Clocks

Clock name	Description
m33_clk	Main M33 clock
m33_systick_clk	M33 Systick clock

36.3.9 Reset

You can use a CM33 reset to reset the CM33_TCM_MCM and CM33_CACHE_ECC_MCM submodules. Performing a cold or soft reset on the CM33 core also resets the submodules.

36.3.10 Interrupts

Table 273. CM33_TCM_MCM interrupt requests

Interrupt request	Asserted when
tcm_ecc1bit_irq	CM33 detects a single-bit ECC error on TCM (including code and system TCM) and the corresponding interrupt is enabled.
tcm_ecc2bit_irq	CM33 detects a multi-bit ECC error on TCM (including code and system TCM) and the corresponding interrupt is enabled.

Table 274. CM33_CACHE_ECC_MCM interrupt requests

Interrupt request	Asserted when
ecc1bit_irq	CM33 detects a single-bit ECC error on cache (including cache TAG0, cache TAG1, cache DATA0, and cache DATA1) and the corresponding interrupt is enabled.
ecc2bit_irq	CM33 detects a multibit ECC error on cache (including cache TAG0, cache TAG1, cache DATA0, and cache DATA1) and the corresponding interrupt is enabled.

36.4 External signals

The CM33 platform has no external signals.

36.5 Initialization

To initialize CM33:

1. Verify that the clock for CM33 is enabled.
2. Enable the ECC features by writing 0 to the following fields before checking the cache or TCM ECC on read or write data:
   - CACHE_ECCR[RECC_DIS]
   - CACHE_ECCR[WECC_DIS]
   - TCMECCR[RECC_DIS]
   - TCMECCR[WECC_DIS]

36.6 Memory map and register definition

This section includes the memory map and detailed descriptions of all registers.

NOTE

For CM33_TCM_MCM:

- When you configure the code TCM to be double sized (BLK_CTRL_Secure_AON.M33_CFG[TCM_SIZE] = 01b), all of the TCM is used as code TCM.
- When you configure the system TCM to be double sized (BLK_CTRL_Secure_AON.M33_CFG[TCM_SIZE] = 10b), all the TCM is used as system TCM.
- However, the code injection and status register always correspond to the TCM space that starts at FFE0000h, and the system injection and status register always correspond to the TCM space that starts at 20000000h.

36.6.1 CM33_CACHE_ECC_MCM register descriptions

36.6.1.1 CM33_CACHE_ECC_MCM memory map

CP_CM33_IMX9RTC.CM33_CACHE_ECC_MCM base address: 4440_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	CACHE ECC Control (CACHE_ECCR)	32	RW	0000_0000h
20h	Interrupt Status (INT_STATUS)	32	RW	0000_0000h
24h	Interrupt Status Enable (INT_STAT_EN)	32	RW	0000_0000h
28h	Interrupt Enable (INT_SIG_EN)	32	RW	0000_0000h
5Ch	Code Cache Single-Bit ECC Error Information (CODE_CACHE_ECC_SINGLE_ERROR_INFO)	32	R	0000_0000h
60h	Code Cache Single-Bit ECC Error Address (CODE_CACHE_ECC_SINGLE_ERROR_ADDR)	32	R	0000_0000h
68h	Code Cache Multibit ECC Error Information (CODE_CACHE_ECC_MULTI_ERROR_INFO)	32	R	0000_0000h
74h	System Cache Single-Bit ECC Error Information (SYSTEM_CACHE_ECC_SINGLE_ERROR_INFO)	32	R	0000_0000h
78h	System Cache Single-Bit ECC Error Address (SYSTEM_CACHE_ECC_SINGLE_ERROR_ADDR)	32	R	0000_0000h
80h	System Cache Multibit ECC Error Information (SYSTEM_CACHE_ECC_MULTI_ERROR_INFO)	32	R	0000_0000h
84h	System Cache Multibit ECC Error Data (SYSTEM_CACHE_ECC_MULTI_ERROR_DATA)	32	R	0000_0000h
8Ch	Code Cache TAG0 ECC Error Injection (CODE_CACHE_TAG0_ECC_ERROR_INJEC)	32	RW	0000_0000h
90h	Code Cache TAG1 ECC Error Injection (CODE_CACHE_TAG1_ECC_ERROR_INJEC)	32	RW	0000_0000h
94h	Code Cache DATA0 ECC Error Injection (CODE_CACHE_DATA0_ECC_ERROR_INJEC)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
98h	Code Cache DATA1 ECC Error Injection (CODE_CACHE_DATA1_ECC_ERROR_INJEC)	32	RW	0000_0000h
9Ch	System Cache TAG0 ECC Error Injection (SYTEM_CACHE_TAG0_ECC_ERROR_INJEC)	32	RW	0000_0000h
A0h	System Cache TAG1 ECC Error Injection (SYSTEM_CACHE_TAG1_ECC_ERROR_INJEC)	32	RW	0000_0000h
A4h	System Cache DATA0 ECC Error Injection (SYSTEM_CACHE_DATA0_ECC_ERROR_INJEC)	32	RW	0000_0000h
A8h	System Cache DATA1 ECC Error Injection (STSTEM_CACHE_DATA1_ECC_ERROR_INJEC)	32	RW	0000_0000h

36.6.1.2 CACHE ECC Control (CACHE_ECCR)

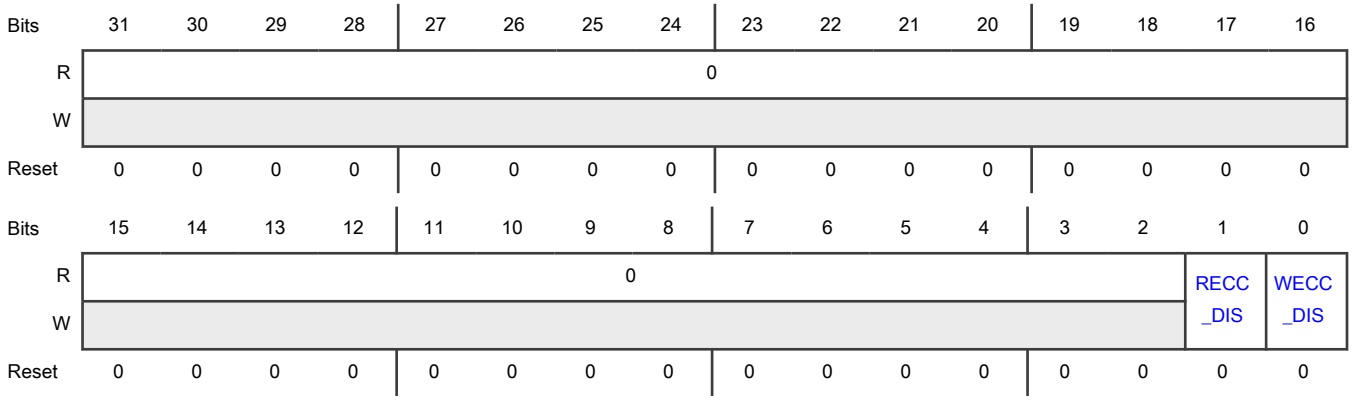
Offset

Register	Offset
CACHE_ECCR	0h

Function

Enables the ECC logic for the Cortex-M33 cache. The cache ECC is enabled by default.

Diagram



Fields

Field	Function
31-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 RECC_DIS	Disable Cache ECC Read Check Disables the ECC check on cache read data. 0b - Enable 1b - Disable
0 WECC_DIS	Disable CACHE ECC Write Generation Disables ECC generation on cache write data. 0b - Enable 1b - Disable

36.6.1.3 Interrupt Status (INT_STATUS)

Offset

Register	Offset
INT_STATUS	20h

Function
Contains the ECC interrupt status for the code and system cache.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SYSTEM_...	SYSTEM_...	SYSTEM_...	SYSTEM_...	CODE_CA...	CODE_CA...	CODE_CA...	CODE_CA...	0							
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 SYSTEM_CACHE_ECC_ERRS_OVER_INT	System Cache Access Multiple Single-Bit ECC Error Interrupt Status Indicates more than one single-bit ECC error in system cache read data when INT_STAT_EN[SYSTEM_CACHE_ECC_ERRS_OVER_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not more than one error 1b - Multiple errors When writing 0b - No effect 1b - Clear the flag
14 SYSTEM_CACHE_ECC_ERRM_OVER_INT	System Cache Access Multiple Multibit ECC Error Interrupt Status Indicates more than one multibit ECC error in system cache read data when INT_STAT_EN[SYSTEM_CACHE_ECC_ERRM_OVER_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not more than one error 1b - Multiple errors When writing 0b - No effect 1b - Clear the flag
13 SYSTEM_CACHE_ECC_ERRS_INT	System Cache Access Single-Bit ECC Error Interrupt Status Indicates a single-bit ECC error in system cache read data when INT_STAT_EN[SYSTEM_CACHE_ECC_ERRS_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
12 SYSTEM_CACHE_ECC_ERRM_INT	System Cache Access Multibit ECC Error Interrupt Status Indicates a multibit ECC error in system cache read data when INT_STAT_EN[SYSTEM_CACHE_ECC_ERRM_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
11 CODE_CACHE_ECC_ERRS_OVER_INT	Code Cache Access Multiple Single-Bit ECC Error Interrupt Status Indicates more than one single-bit ECC error in code cache read data when INT_STAT_EN[CODE_CACHE_ERRS_OVER_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not more than one error 1b - Multiple errors When writing 0b - No effect 1b - Clear the flag
10 CODE_CACHE_ECC_ERRM_OVER_INT	Code Cache Access Multiple Multibit ECC Error Interrupt Status Indicates more than one multibit ECC error in code cache read data when INT_STAT_EN[CODE_CACHE_ERRM_OVER_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not more than one error 1b - Multiple errors When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
9 CODE_CACHE_ECC_ERRS_INT	Code Cache Access Single-Bit ECC Error Interrupt Status Indicates a single-bit ECC error in code cache read data when INT_STAT_EN[CODE_CACHE_ERRS_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
8 CODE_CACHE_ECC_ERRM_INT	Code Cache Access Multibit ECC Error Interrupt Status Indicates a multibit ECC error in code cache read data when INT_STAT_EN[CODE_CACHE_ERRM_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
7-0 —	Reserved

36.6.1.4 Interrupt Status Enable (INT_STAT_EN)

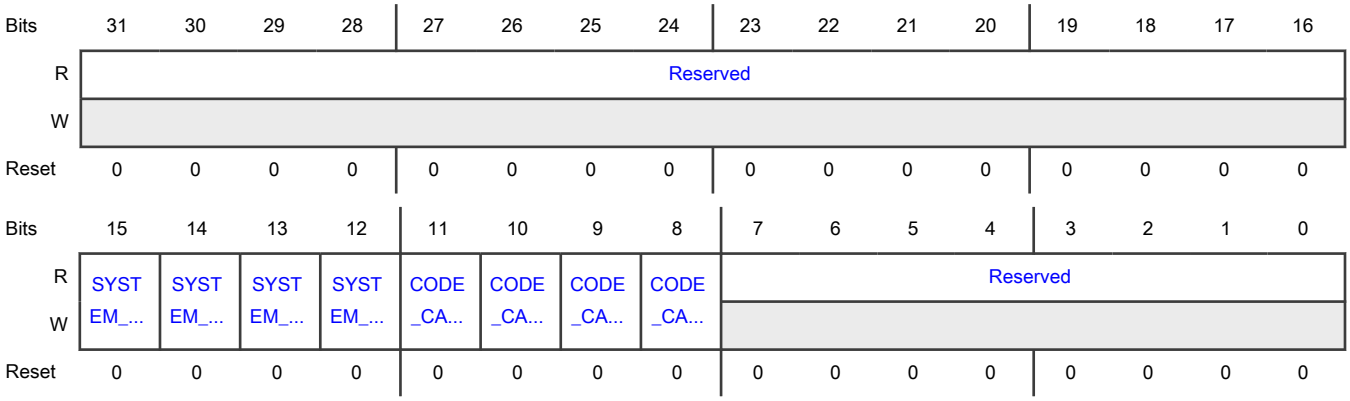
Offset

Register	Offset
INT_STAT_EN	24h

Function

Enables the interrupt statuses in Interrupt Status (INT_STATUS).

Diagram



Fields

Field	Function
31-16 —	Reserved
15 SYSTEM_CACHE_ECC_ERRS_OVER_INT_EN	System Cache Access Multiple Single-Bit ECC Error Interrupt Status Enable Enables INT_STATUS[SYSTEM_CACHE_ECC_ERRS_OVER_INT]. 0b - Mask 1b - Enable
14 SYSTEM_CACHE_ECC_ERRM_OVER_INT_EN	System Cache Access Multiple Multibit ECC Error Interrupt Status Enable Enables INT_STATUS[SYSTEM_CACHE_ECC_ERRM_OVER_INT]. 0b - Mask 1b - Enable
13 SYSTEM_CACHE_ECC_ERRS_INT_EN	System Cache Access Single-Bit ECC Error Interrupt Status Enable Enables INT_STATUS[SYSTEM_CACHE_ECC_ERRS_INT]. 0b - Mask 1b - Enable
12 SYSTEM_CACHE_ECC_ERRM_INT_EN	System Cache Access Multibit ECC Error Interrupt Status Enable Enables INT_STATUS[SYSTEM_CACHE_ECC_ERRM_INT]. 0b - Mask 1b - Enable
11	Code Cache Access Multiple Single-Bit ECC Error Interrupt Status Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
CODE_CACHE_ERRS_OVER_INT_EN	Enables INT_STATUS[CODE_CACHE_ECC_ERRS_OVER_INT]. 0b - Mask 1b - Enable
10 CODE_CACHE_ERRM_OVER_INT_EN	Code Cache Access Multiple Multibit ECC Error Interrupt Status Enable Enables INT_STATUS[CODE_CACHE_ECC_ERRM_OVER_INT]. 0b - Mask 1b - Enable
9 CODE_CACHE_ERRS_INT_EN	Code Cache Access Single-Bit ECC Error Interrupt Status Enable Enables INT_STATUS[CODE_CACHE_ECC_ERRS_INT]. 0b - Mask 1b - Enable
8 CODE_CACHE_ERRM_INT_EN	Code Cache Access Multibit ECC Error Interrupt Status Enable Enables INT_STATUS[CODE_CACHE_ECC_ERRM_INT]. 0b - Mask 1b - Enable
7-0 —	Reserved

36.6.1.5 Interrupt Enable (INT_SIG_EN)

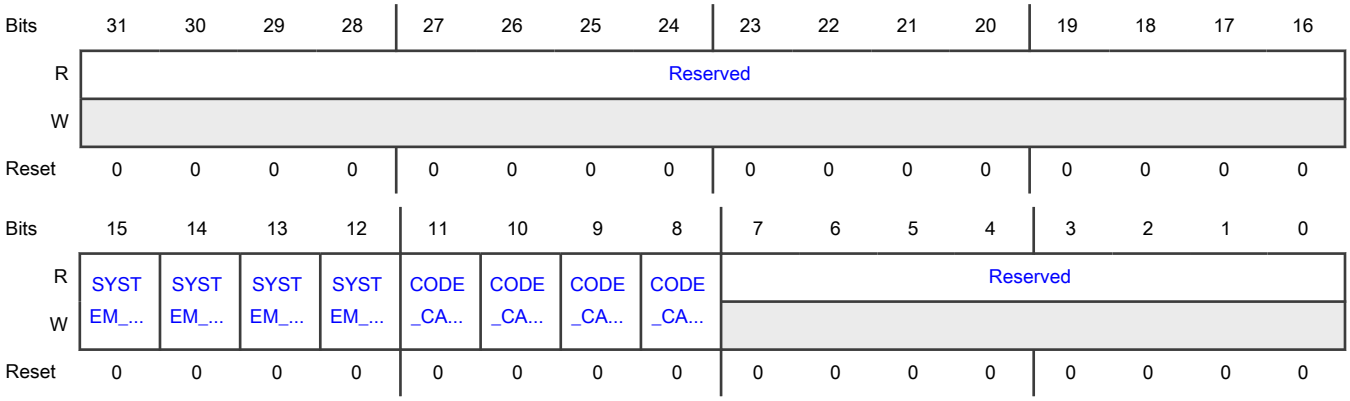
Offset

Register	Offset
INT_SIG_EN	28h

Function

Selects which interrupt statuses are available to interrupt the host system. All these status fields share the same interrupt line (that is, all interrupts are combined with a logical OR). Writing 1 to any of these fields enables interrupt generation. The corresponding field in Interrupt Status (INT_STATUS) generates an interrupt when the corresponding field in Interrupt Status Enable (INT_STAT_EN) is 1.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 SYSTEM_CACHE_ERRS_OVER_INT_SIG_EN	System Cache Access Multiple Single-Bit ECC Error Interrupt Signal Enable Selects INT_STATUS[SYSTEM_CACHE_ECC_ERRS_OVER_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Mask 1b - Enable
14 SYSTEM_CACHE_ERRM_OVER_INT_SIG_EN	System Cache Access Multiple Multibit ECC Error Interrupt Signal Enable Selects INT_STATUS[SYSTEM_CACHE_ECC_ERRM_OVER_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Masked 1b - Enabled
13 SYSTEM_CACHE_ERRS_INT_SIG_EN	System Cache Access Single-Bit ECC Error Interrupt Signal Enable Selects INT_STATUS[SYSTEM_CACHE_ECC_ERRS_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Mask 1b - Enable
12 SYSTEM_CACHE_ERRM_INT_SIG_EN	System Cache Access Multibit ECC Error Interrupt Signal Enable Selects INT_STATUS[SYSTEM_CACHE_ECC_ERRM_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Mask 1b - Enable
11	Code Cache Access Multiple Single-Bit ECC Error Interrupt Signal Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
CODE_CACHE_ERRS_OVER_INT_SIG_EN	Selects INT_STATUS[CODE_CACHE_ECC_ERRS_OVER_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Mask 1b - Enable
10 CODE_CACHE_ERRM_OVER_INT_SIG_EN	Code Cache Access Multiple Multibit ECC Error Interrupt Signal Enable Selects INT_STATUS[CODE_CACHE_ECC_ERRM_OVER_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Mask 1b - Enable
9 CODE_CACHE_ERRS_INT_SIG_EN	Code Cache Access Single-Bit ECC Error Interrupt Signal Enable Selects INT_STATUS[CODE_CACHE_ECC_ERRS_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Mask 1b - Enable
8 CODE_CACHE_ERRM_INT_SIG_EN	Code Cache Access Multibit ECC Error Interrupt Signal Enable Selects INT_STATUS[CODE_CACHE_ECC_ERRM_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Mask 1b - Enable
7-0 —	Reserved

36.6.1.6 Code Cache Single-Bit ECC Error Information (CODE_CACHE_ECC_SINGLE_ERROR_INFO)

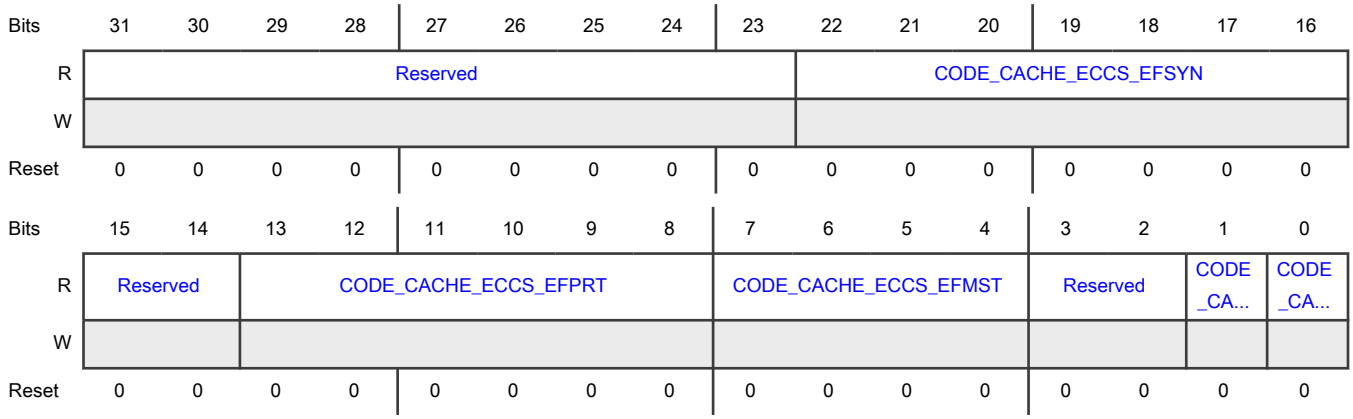
Offset

Register	Offset
CODE_CACHE_ECC_SINGLE_ERROR_INFO	5Ch

Function

Stores detailed error information for the corresponding code cache single-bit ECC interrupt when the interrupt is enabled.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 CODE_CACHE_ECCS_EFSYN	Code Cache Single-Bit ECC Error Corresponding Syndrome Indicates the corresponding syndrome when a single-bit error is reported. You cannot use the syndrome for single-bit ECC to locate the error bit.
15-14 —	Reserved
13-8 CODE_CACHE_ECCS_EFPRT	Code Cache Single-Bit ECC Error Protection Indicates the master access protection attribute of the access that created the error when a single-bit error is reported. For details about the protection attribute decoding, see the Arm AMBA AHB specification.
7-4 CODE_CACHE_ECCS_EFMST	Code Cache Single-Bit ECC Error Master Number Indicates the ECC error master number when a single ECC error is reported.
3-2 —	Reserved
1 CODE_CACHE_ECCS_CMD	Code Cache Single-Bit ECC Error on Cache Command Indicates a code cache single-bit ECC error on the cache command when a single ECC error is reported. 0b - No error 1b - Error
0	Code Cache Single-Bit ECC Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
CODE_CACHE_ECCS_TAG	Indicates whether the single-bit ECC error is in the cache tag or the cache data. 0b - Data 1b - Tag

36.6.1.7 Code Cache Single-Bit ECC Error Address (CODE_CACHE_ECC_SINGLE_ERROR_ADDR)

Offset

Register	Offset
CODE_CACHE_ECC_SINGLE_ERROR_ADDR	60h

Function

Stores the error address for a single-bit ECC interrupt.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CODE_CACHE_ECCS_ERRED_ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CODE_CACHE_ECCS_ERRED_ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CODE_CACHE_ECCS_ERRED_ADDR	Code Cache Single-Bit ECC Error Address Stores the error address for the corresponding code cache single-bit ECC interrupt when the interrupt is enabled.

36.6.1.8 Code Cache Multibit ECC Error Information (CODE_CACHE_ECC_MULTI_ERROR_INFO)

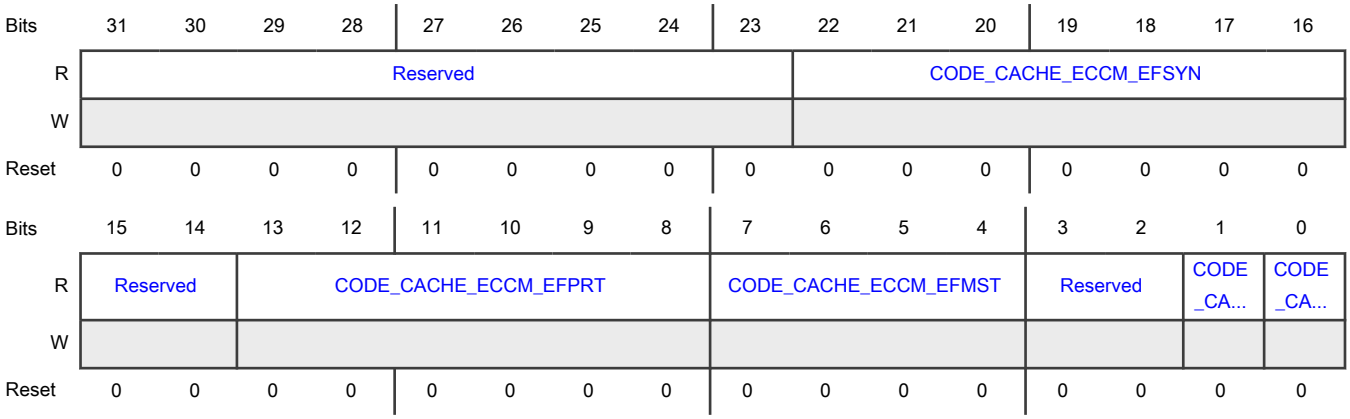
Offset

Register	Offset
CODE_CACHE_ECC_MULTI_ERROR_INFO	68h

Function

Stores detailed error information for the corresponding code cache multibit ECC interrupt when the interrupt is enabled.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 CODE_CACHE_ECCM_EFSYN	Code Cache Multibit ECC Error Corresponding Syndrome Indicates the corresponding syndrome when a multibit error is reported. You cannot use the syndrome for multibit ECC to locate the error bit.
15-14 —	Reserved
13-8 CODE_CACHE_ECCM_EFPRT	Code Cache Multibit ECC Error Protection Indicates the master access protection attribute of the access creating the error when a multibit error is reported. For details about the protection attribute decoding, see the Arm AMBA AHB specification.
7-4	Code Cache Multibit ECC Error Master Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
CODE_CACHE_ECCM_EFMST	Indicates the ECC error master number when a multibit ECC error is reported.
3-2 —	Reserved
1 CODE_CACHE_ECCM_CMD	Code Cache Multibit ECC Error on Code Cache Command Indicates whether the error has occurred on a code cache command when a multibit ECC error is reported. 0b - No error 1b - Error
0 CODE_CACHE_ECCM_TAG	Code Cache Multibit ECC Error Indicates whether the multibit ECC error is in the cache tag or the cache data. 0b - Data 1b - Tag

36.6.1.9 System Cache Single-Bit ECC Error Information (SYSTEM_CACHE_ECC_SINGLE_ERROR_INFO)

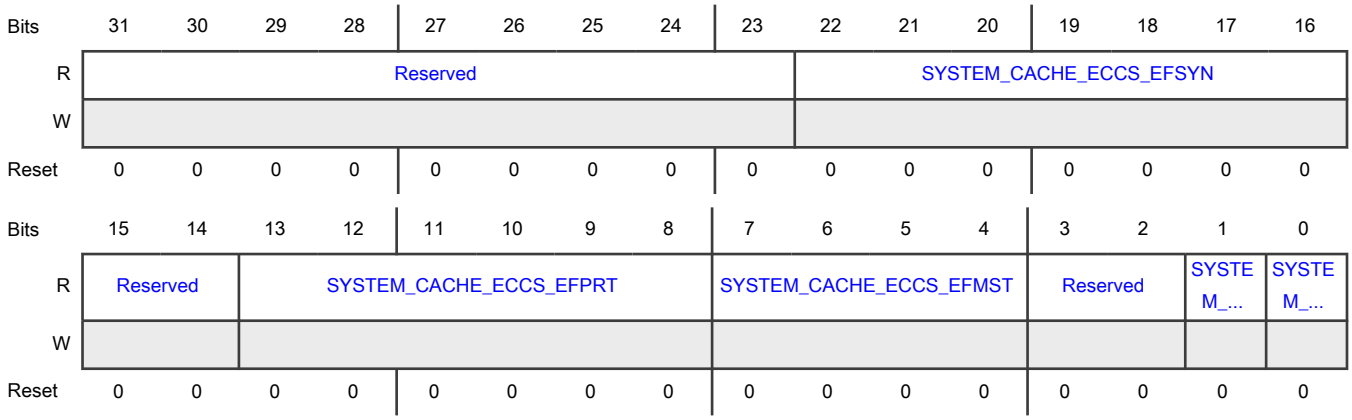
Offset

Register	Offset
SYSTEM_CACHE_ECC_SINGLE_ERROR_INFO	74h

Function

Stores error information for the corresponding system cache single-bit ECC interrupt when the interrupt is enabled.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 SYSTEM_CACHE_ECCS_EFSYN	System Cache Single-Bit ECC Error Corresponding Syndrome Indicates the corresponding syndrome when a single-bit error is reported. You cannot use the syndrome for single-bit ECC to locate the error bit.
15-14 —	Reserved
13-8 SYSTEM_CACHE_ECCS_EFPRT	System Cache Single-Bit ECC Error Protection Indicates the master access protection attribute of the access creating the error when a single-bit error is reported. For details about the protection attribute decoding, see the Arm AMBA AHB specification.
7-4 SYSTEM_CACHE_ECCS_EFMST	System Cache Single-Bit ECC Error Master Number Indicates the ECC error master number when a single-bit ECC error is reported.
3-2 —	Reserved
1 SYSTEM_CACHE_ECCS_CMD	System Cache Single-Bit ECC Error on Cache Command Indicates whether a system cache single-bit ECC error has occurred on the cache command when a single ECC error is reported. 0b - No error 1b - Error
0 SYSTEM_CACHE_ECCS_TAG	System Cache Single-Bit ECC Error Indicates whether the single-bit ECC error is in the system tag or the system data. 0b - Data 1b - Tag

36.6.1.10 System Cache Single-Bit ECC Error Address (SYSTEM_CACHE_ECC_SINGLE_ERROR_ADDR)

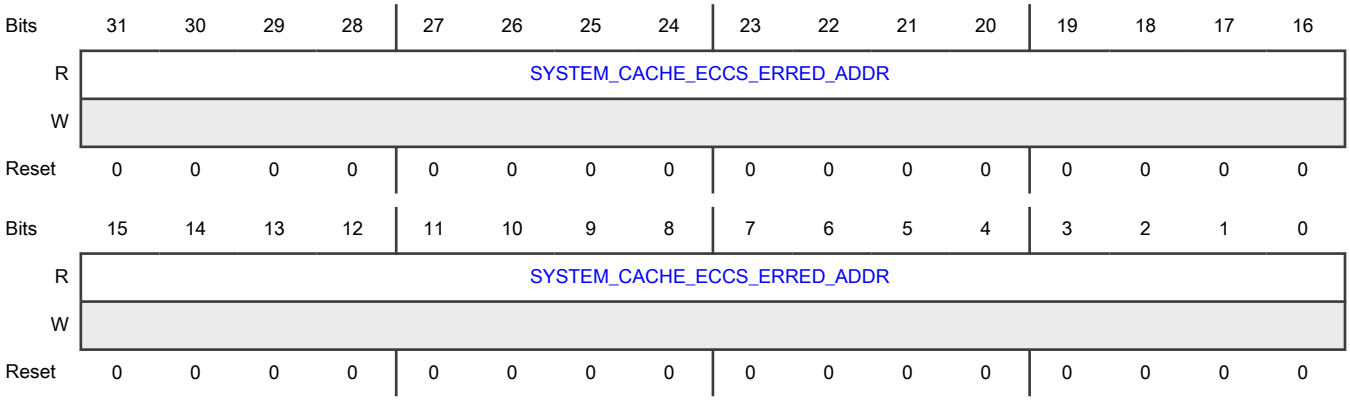
Offset

Register	Offset
SYSTEM_CACHE_ECC_SINGLE_ERROR_ADDR	78h

Function

Stores the error address for a system cache single-bit ECC interrupt.

Diagram



Fields

Field	Function
31-0 SYSTEM_CACHE_ECCS_ERRED_ADDR	System Cache Single-Bit ECC Error Address Stores the error address for the corresponding system cache single-bit ECC interrupt when the interrupt is enabled.

36.6.1.11 System Cache Multibit ECC Error Information (SYSTEM_CACHE_ECC_MULTI_ERROR_INFO)

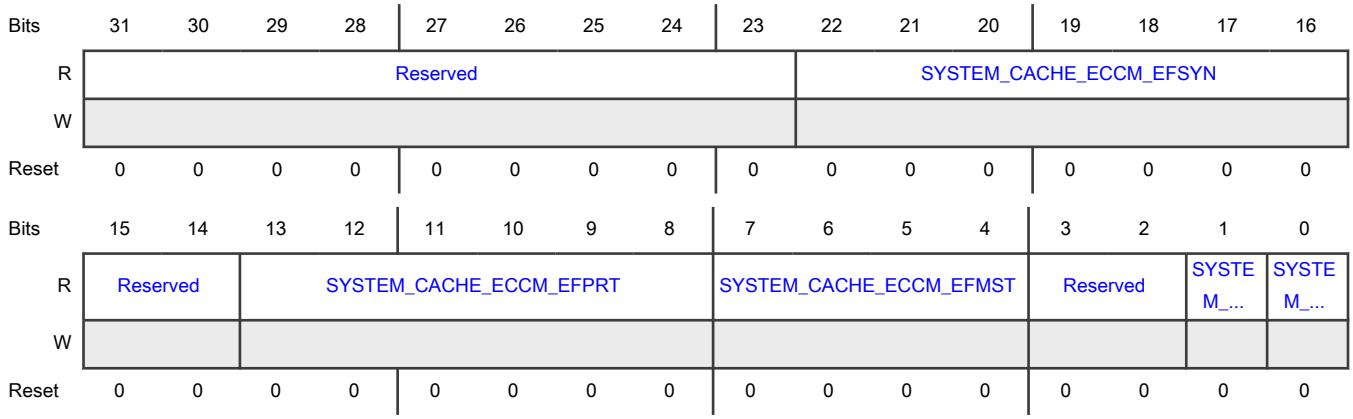
Offset

Register	Offset
SYSTEM_CACHE_ECC_MULTI_ERROR_INFO	80h

Function

Stores error information for the corresponding system cache multibit ECC interrupt when the interrupt is enabled.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 SYSTEM_CACHE_ECCM_EFSYN	System Cache Multibit ECC Error Corresponding Syndrome Indicates the corresponding syndrome when a multibit error is reported. You cannot use the syndrome for multibit ECC to locate the error bit.
15-14 —	Reserved
13-8 SYSTEM_CACHE_ECCM_EFPRT	System Cache Multibit ECC Error Protection Indicates the master access protection attribute of the access that created the error when a multibit error is reported. For details about the protection attribute decoding, refer to the Arm AMBA AHB specification.
7-4 SYSTEM_CACHE_ECCM_EFMST	System Cache Multibit ECC Error Master Number Indicates the ECC error master number when a multibit ECC error is reported.
3-2 —	Reserved
1 SYSTEM_CACHE_ECCM_CMD	System Cache Multibit ECC Error on System Cache Command Indicates whether a system cache multibit ECC error has occurred on a cache command when a multibit ECC error is reported. 0b - No error 1b - Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 SYSTEM_CACHE_ECCM_TAG	System Cache Multibit ECC Error Indicates whether the multibit ECC error is in the system tag or the system data. 0b - Data 1b - Tag

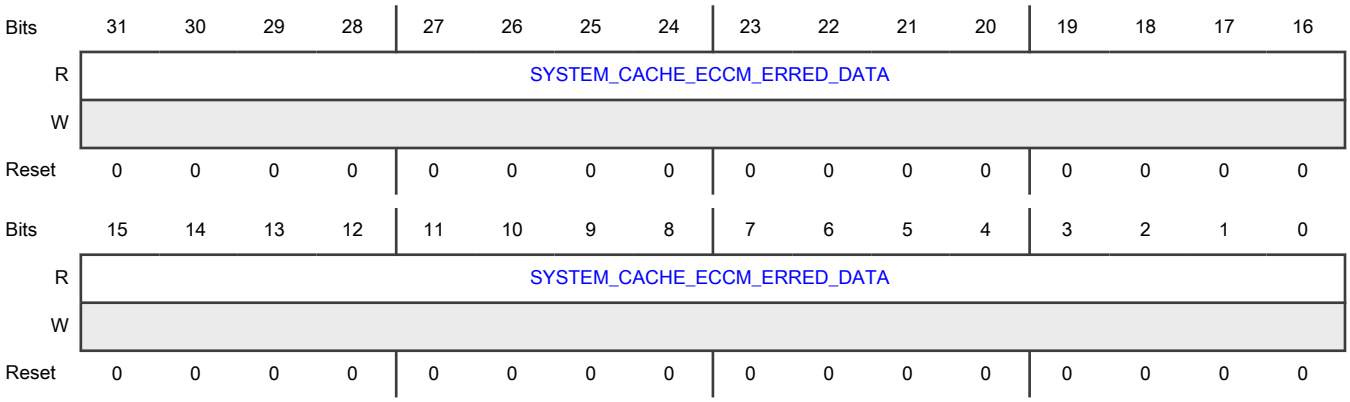
36.6.1.12 System Cache Multibit ECC Error Data (SYSTEM_CACHE_ECC_MULTI_ERROR_DATA)

Offset

Register	Offset
SYSTEM_CACHE_ECC_MULTI_ERROR_DATA	84h

Function
Stores error data for the system cache multibit ECC interrupt.

Diagram



Fields

Field	Function
31-0 SYSTEM_CACHE_ECCM_ERRED_DATA	System Cache Multibit ECC Error Data Stores error data for the corresponding system cache multibit ECC interrupt when the interrupt is enabled.

36.6.1.13 Code Cache TAG0 ECC Error Injection (CODE_CACHE_TAG0_ECC_ERROR_INJEC)

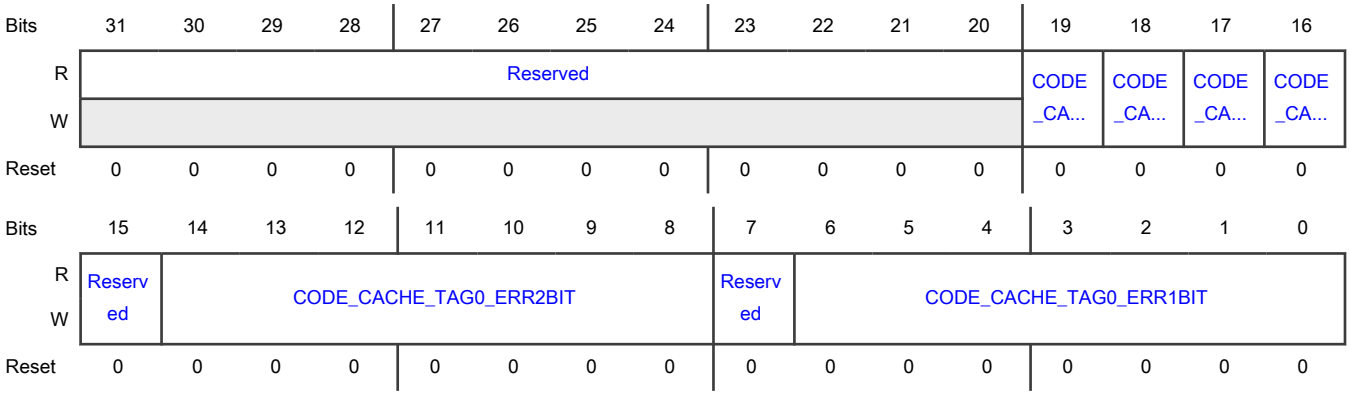
Offset

Register	Offset
CODE_CACHE_TAG0_ECC_ERROR_INJEC	8Ch

Function

Injects an ECC error into code cache TAG0. This register is used only for verification and debugging.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CODE_CACHE_TAG0_FRCNCI	Force Continuous Noncorrectable Data Inversions on Code Cache TAG0 Write Access Injects a multibit ECC error on the bits specified by CODE_CACHE_TAG0_ERR1BIT and CODE_CACHE_TAG0_ERR2BIT into every write access to code cache TAG0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
18 CODE_CACHE_TAG0_FRC1BI	Force Continuous 1-Bit Data Inversions on Code Cache TAG0 Write Access Injects a single-bit ECC error on the bit specified by CODE_CACHE_TAG0_ERR1BIT into every write access to code cache TAG0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
17	Force One Noncorrectable Data Inversion on Code Cache TAG0 Write Access

Table continues on the next page...

Table continued from the previous page...

Field	Function
CODE_CACHE_TAG0_FR1NCI	Injects a multibit ECC error on the bits specified by CODE_CACHE_TAG0_ERR1BIT and CODE_CACHE_TAG0_ERR2BIT into the first write access to code cache TAG0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
16 CODE_CACHE_TAG0_FR11BI	Force One 1-Bit Data Inversion on Code Cache TAG0 Write Access Injects a single-bit ECC error on the bit specified by CODE_CACHE_TAG0_ERR1BIT into the first write access to code cache TAG0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
15 —	Reserved
14-8 CODE_CACHE_TAG0_ERR2BIT	Position of Second Bit to Inject ECC Error Determines the position of the second bit into which CM33 injects an ECC error. This field is used only for multibit ECC error injection.
7 —	Reserved
6-0 CODE_CACHE_TAG0_ERR1BIT	Position of First Bit to Inject ECC Error Determines the position of the first bit into which CM33 injects an ECC error. This field can be used for both single-bit and multibit ECC error injection.

36.6.1.14 Code Cache TAG1 ECC Error Injection (CODE_CACHE_TAG1_ECC_ERROR_INJEC)

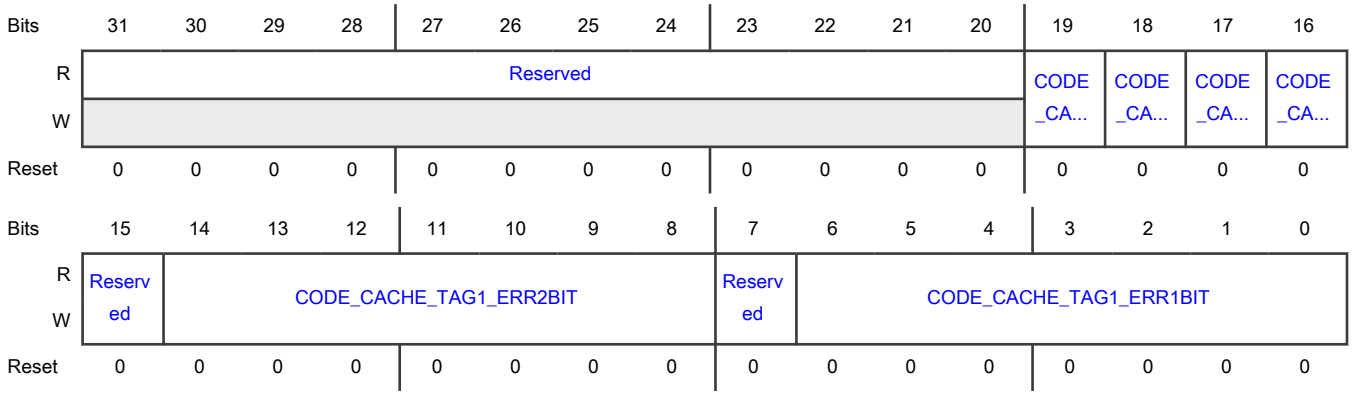
Offset

Register	Offset
CODE_CACHE_TAG1_ECC_ERROR_INJEC	90h

Function

Injects an ECC error into code cache TAG1. This register is used only for verification and debugging.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CODE_CACHE_TAG1_FRCNCI	Force Continuous Noncorrectable Data Inversions on Code Cache TAG1 Write Access Injects a multibit ECC error on the bits specified by CODE_CACHE_TAG1_ERR1BIT and CODE_CACHE_TAG1_ERR2BIT into every write access to code cache TAG1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
18 CODE_CACHE_TAG1_FRC1BI	Force Continuous 1-Bit Data Inversions on Code Cache TAG1 Write Access Injects a single-bit ECC error on the bit specified by CODE_CACHE_TAG1_ERR1BIT into every write access to code cache TAG1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
17 CODE_CACHE_TAG1_FR1NCI	Force One Noncorrectable Data Inversion on Code Cache TAG1 Write Access Injects a multibit ECC error on the bits specified by CODE_CACHE_TAG1_ERR1BIT and CODE_CACHE_TAG1_ERR2BIT into the first write access to code cache TAG1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
16 CODE_CACHE_TAG1_FR11BI	Force One 1-Bit Data Inversion on Code Cache TAG1 Write Access Injects a single-bit ECC error on the bit specified by CODE_CACHE_TAG1_ERR1BIT into the first write access to code cache TAG1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 —	Reserved
14-8 CODE_CACHE_TAG1_ERR2BIT	Position of Second Bit to Inject ECC Error Determines the position of the second bit into which CM33 injects an ECC error. This field is used only for multibit ECC error injection.
7 —	Reserved
6-0 CODE_CACHE_TAG1_ERR1BIT	Position of First Bit to Inject ECC Error Determines the position of the first bit into which CM33 injects an ECC error. This field can be used for both single-bit and multibit ECC error injection.

36.6.1.15 Code Cache DATA0 ECC Error Injection (CODE_CACHE_DATA0_ECC_ERROR_INJEC)

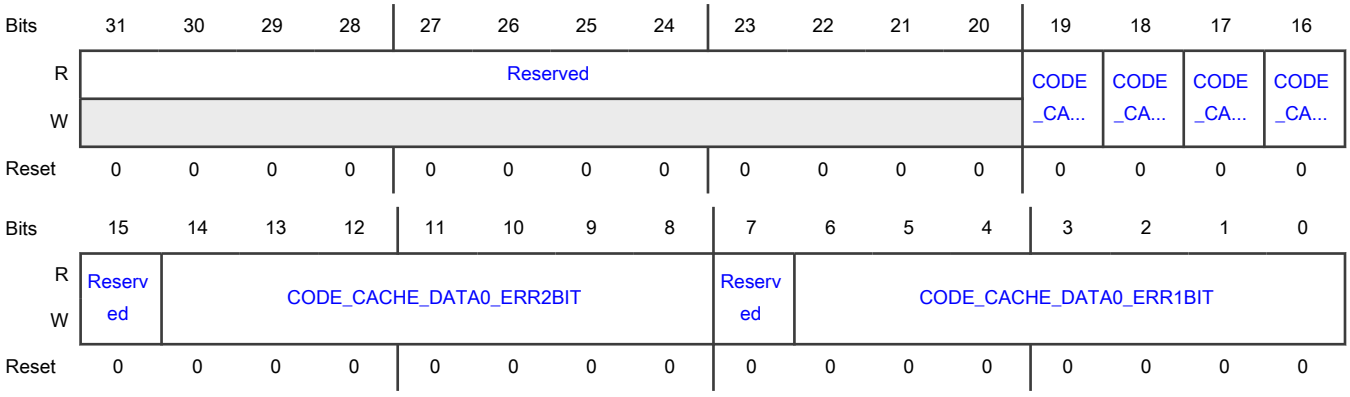
Offset

Register	Offset
CODE_CACHE_DATA0_ECC_ERROR_INJEC	94h

Function

Injects an ECC error into code cache DATA0. This register is used only for verification and debugging.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CODE_CACHE_DATA0_FRCNCI	Force Continuous Noncorrectable Data Inversions on Code Cache DATA0 Write Access Injects a multibit ECC error on the bits specified by CODE_CACHE_DATA0_ERR1BIT and CODE_CACHE_DATA0_ERR2BIT into every write access to code cache DATA0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
18 CODE_CACHE_DATA0_FRC1BI	Force Continuous 1-Bit Data Inversions on Code Cache DATA0 Write Access Injects a single-bit ECC error on the bit specified by CODE_CACHE_DATA0_ERR1BIT into every write access to code cache DATA0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
17 CODE_CACHE_DATA0_FR1NCI	Force One Noncorrectable Data Inversion on Code Cache DATA0 Write Access Injects a multibit ECC error on the bits specified by CODE_CACHE_DATA0_ERR1BIT and CODE_CACHE_DATA0_ERR2BIT into the first write access to code cache DATA0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
16 CODE_CACHE_DATA0_FR11BI	Force One 1-Bit Data Inversion on Code Cache DATA0 Write Access Injects a single-bit ECC error on the bit specified by CODE_CACHE_DATA0_ERR1BIT into the first write access to code cache DATA0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
15 —	Reserved
14-8 CODE_CACHE_DATA0_ERR2BIT	Position of Second Bit to Inject ECC Error Determines the position of the second bit into which CM33 injects an ECC error. This field is used only for multibit ECC error injection.
7 —	Reserved
6-0	Position of First Bit to Inject ECC Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
CODE_CACHE_DATA0_ERR1BIT	Determines the position of the first bit into which CM33 injects an ECC error. This field can be used for both single-bit and multibit ECC error injection.

36.6.1.16 Code Cache DATA1 ECC Error Injection (CODE_CACHE_DATA1_ECC_ERROR_INJEC)

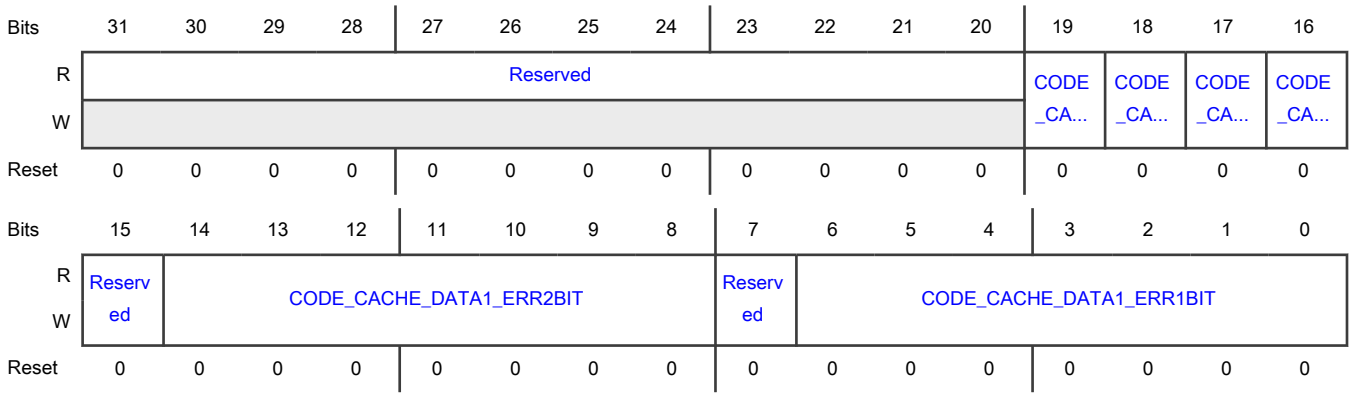
Offset

Register	Offset
CODE_CACHE_DATA1_ECC_ERROR_INJEC	98h

Function

Injects an ECC error into code cache DATA1. This register is used only for verification and debugging.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CODE_CACHE_DATA1_FRCNCI	Force Continuous Noncorrectable Data Inversions on Code Cache DATA1 Write Access Injects a multibit ECC error on the bits specified by CODE_CACHE_DATA1_ERR1BIT and CODE_CACHE_DATA1_ERR2BIT into every write access to code cache DATA1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 CODE_CACHE_DATA1_FRC1BI	Force Continuous 1-Bit Data Inversions on Code Cache DATA1 Write Access Injects a single-bit ECC error on the bit specified by CODE_CACHE_DATA1_ERR1BIT into every write access to code cache DATA1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
17 CODE_CACHE_DATA1_FR1NCI	Force One Noncorrectable Data Inversion on Code Cache DATA1 Write Access Injects a multibit ECC error on the bits specified by CODE_CACHE_DATA1_ERR1BIT and CODE_CACHE_DATA1_ERR2BIT into the first write access to code cache DATA1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
16 CODE_CACHE_DATA1_FR11BI	Force One 1-Bit Data Inversion on Code Cache DATA1 Write Access Injects a single-bit ECC error on the bit specified by CODE_CACHE_DATA1_ERR1BIT into the first write access to code cache DATA1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
15 —	Reserved
14-8 CODE_CACHE_DATA1_ERR2BIT	Position of Second Bit to Inject ECC Error Determines the position of the second bit into which CM33 injects an ECC error. This field is used only for multibit ECC error injection.
7 —	Reserved
6-0 CODE_CACHE_DATA1_ERR1BIT	Position of First Bit to Inject ECC Error Determines the position of the first bit into which CM33 injects an ECC error. This field can be used for both single-bit and multibit ECC error injection.

36.6.1.17 System Cache TAG0 ECC Error Injection (SYTEM_CACHE_TAG0_ECC_ERROR_INJEC)

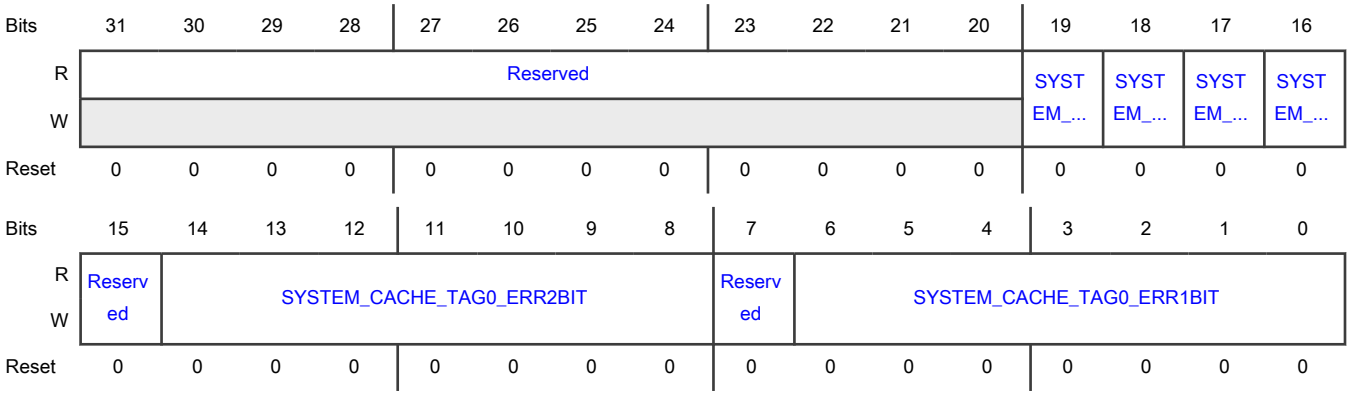
Offset

Register	Offset
SYTEM_CACHE_TAG0_ECC_ERROR_INJEC	9Ch

Function

Injects an ECC error into system cache TAG0. This register is used only for verification and debugging.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 SYSTEM_CACHE_TAG0_FRCNCI	Force Continuous Noncorrectable Data Inversions on System Cache TAG0 Write Access Injects a multibit ECC error on the bits specified by SYSTEM_CACHE_TAG0_ERR1BIT and SYSTEM_CACHE_TAG0_ERR2BIT into every write access to system cache DATA1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
18 SYSTEM_CACHE_TAG0_FRC1BI	Force Continuous 1-Bit Data Inversions on System Cache TAG0 Write Access Injects a single-bit ECC error on the bit specified by SYSTEM_CACHE_TAG0_ERR1BIT into every write access to system cache TAG0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
17 SYSTEM_CACHE_TAG0_FR1NCI	Force One Noncorrectable Data Inversion on System Cache TAG0 Write Access Injects a multibit ECC error on the bits specified by SYSTEM_CACHE_TAG0_ERR1BIT and SYSTEM_CACHE_TAG0_ERR2BIT into the first write access to system cache TAG0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
16	Force One 1-Bit Data Inversion on System Cache TAG0 Write Access Injects a single-bit ECC error on the bit specified by SYSTEM_CACHE_TAG0_ERR1BIT into the first write access to system cache TAG0 that occurs after you write 1 to this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYSTEM_CACHE_TAG0_FR11BI	0b - Disable injection 1b - Enable injection
15 —	Reserved
14-8 SYSTEM_CACHE_TAG0_ERR2BIT	Position of Second Bit to Inject ECC Error Determines the position of the second bit into which CM33 injects an ECC error. This field is used only for multibit ECC error injection.
7 —	Reserved
6-0 SYSTEM_CACHE_TAG0_ERR1BIT	Position of First Bit to Inject ECC Error Determines the position of the first bit into which CM33 injects an ECC error. This field can be used for both single-bit and multibit ECC error injection.

36.6.1.18 System Cache TAG1 ECC Error Injection (SYSTEM_CACHE_TAG1_ECC_ERROR_INJEC)

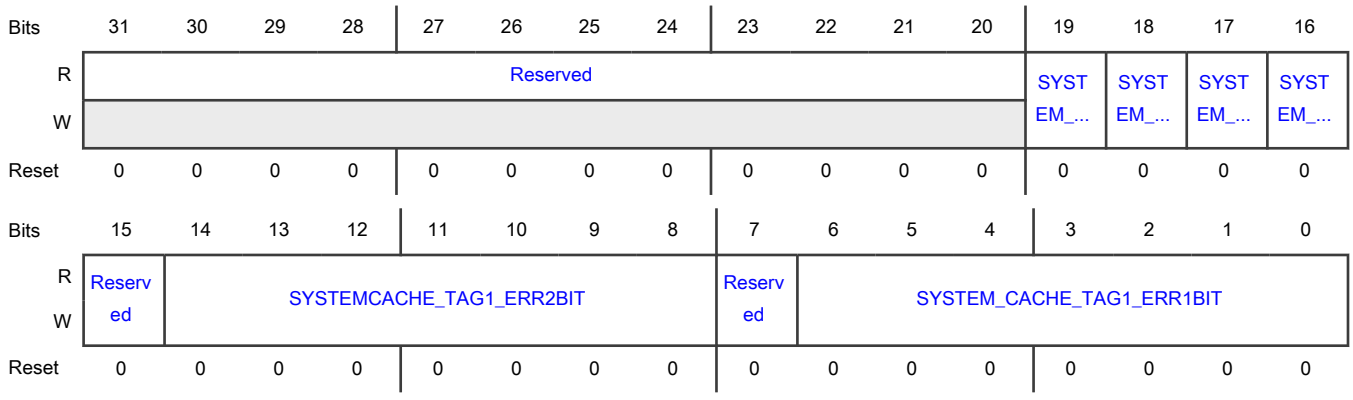
Offset

Register	Offset
SYSTEM_CACHE_TAG1_ECC_ERROR_INJEC	A0h

Function

Injects an ECC error into system cache TAG1. This register is used only for verification and debugging.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 SYSTEM_CACHE_TAG1_FRCNCI	Force Continuous Noncorrectable Data Inversions on System Cache TAG1 Write Access Injects a multibit ECC error on the bits specified by SYSTEM_CACHE_TAG1_ERR1BIT and SYSTEM_CACHE_TAG1_ERR2BIT into every write access to system cache TAG1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
18 SYSTEM_CACHE_TAG1_FRC1BI	Force Continuous 1-Bit Data Inversions on System Cache TAG1 Write Access Injects a single-bit ECC error on the bit specified by SYSTEM_CACHE_TAG1_ERR1BIT into every write access to system cache TAG1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
17 SYSTEM_CACHE_TAG1_FR1NCI	Force One Noncorrectable Data Inversion on System Cache TAG1 Write Access Injects a multibit ECC error on the bits specified by SYSTEM_CACHE_TAG1_ERR1BIT and SYSTEM_CACHE_TAG1_ERR2BIT into the first write access to system cache TAG1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
16 SYSTEM_CACHE_TAG1_FR11BI	Force One 1-Bit Data Inversion on System Cache TAG1 Write Access Injects a single-bit ECC error on the bit specified by SYSTEM_CACHE_TAG1_ERR1BIT into the first write access to system cache TAG1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
15 —	Reserved
14-8 SYSTEMCACHE_TAG1_ERR2BIT	Position of Second Bit to Inject ECC Error Determines the position of the second bit into which CM33 injects an ECC error. This field is used only for multibit ECC error injection.
7 —	Reserved
6-0	Position of First Bit to Inject ECC Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYSTEM_CACHE_TAG1_ERR1BIT	Determines the position of the first bit into which CM33 injects an ECC error. This field can be used for both single-bit and multibit ECC error injection.

36.6.1.19 System Cache DATA0 ECC Error Injection (SYSTEM_CACHE_DATA0_ECC_ERROR_INJEC)

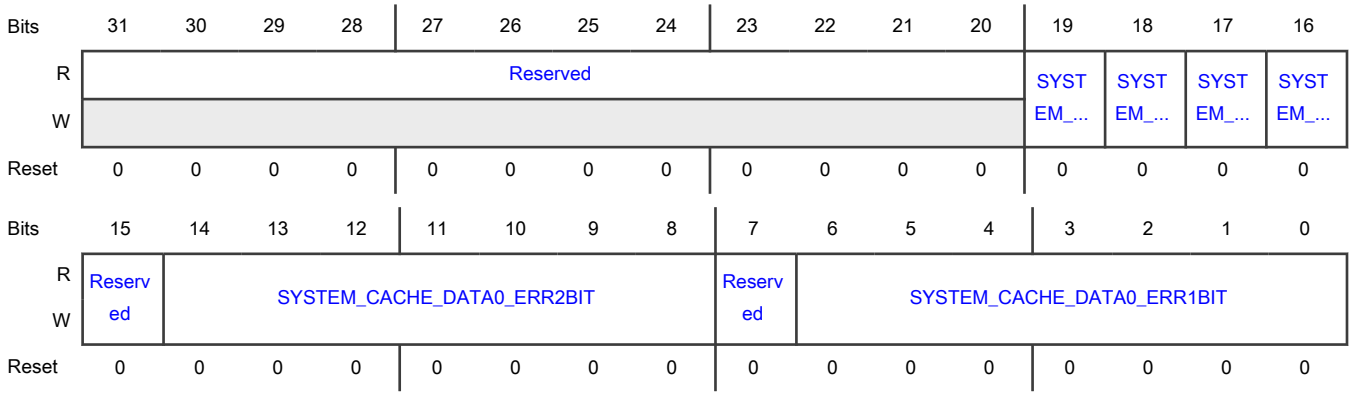
Offset

Register	Offset
SYSTEM_CACHE_DATA0_ECC_ERROR_INJEC	A4h

Function

Injects an ECC error into system cache DATA0. This register is used only for verification and debugging.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 SYSTEM_CACHE_DATA0_FRCNCI	Force Continuous Noncorrectable Data Inversions on System Cache DATA0 Write Access Injects a multibit ECC error on the bits specified by SYSTEM_CACHE_DATA0_ERR1BIT and SYSTEM_CACHE_DATA0_ERR2BIT into every write access to system cache DATA0 that occurs after you write 1 to this field. 0b - Disable injection

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable injection
18 SYSTEM_CACHE_DATA0_FRC1BI	Force Continuous 1-Bit Data Inversions on System Cache DATA0 Write Access Injects a single-bit ECC error on the bit specified by SYSTEM_CACHE_DATA0_ERR1BIT into every write access to system cache DATA0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
17 SYSTEM_CACHE_DATA0_FR1NCI	Force One Noncorrectable Data Inversion on System Cache DATA0 Write Access Injects a multibit ECC error on the bits specified by SYSTEM_CACHE_DATA0_ERR1BIT and SYSTEM_CACHE_DATA0_ERR2BIT into the first write access to system cache DATA0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
16 SYSTEM_CACHE_DATA0_FR11BI	Force One 1-Bit Data Inversion on System Cache DATA0 Write Access Injects a single-bit ECC error on the bit specified by SYSTEM_CACHE_DATA0_ERR1BIT into the first write access to system cache DATA0 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
15 —	Reserved
14-8 SYSTEM_CACHE_DATA0_ERR2BIT	Position of Second Bit to Inject ECC Error Determines the position of the second bit into which CM33 injects an ECC error. This field is used only for multibit ECC error injection.
7 —	Reserved
6-0 SYSTEM_CACHE_DATA0_ERR1BIT	Position of First Bit to Inject ECC Error Determines the position of the first bit into which CM33 injects an ECC error. This field can be used for both single-bit and multibit ECC error injection.

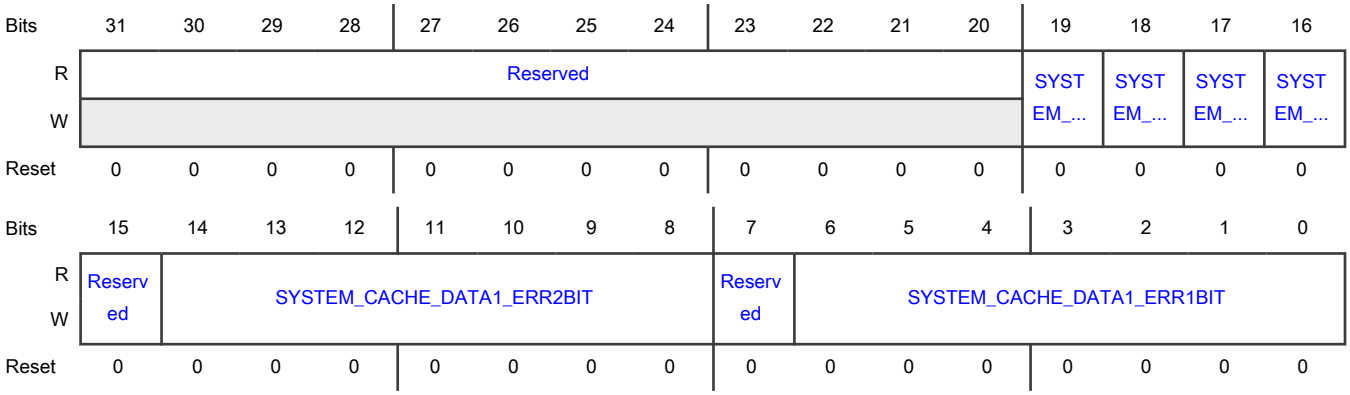
36.6.1.20 System Cache DATA1 ECC Error Injection (STSTEM_CACHE_DATA1_ECC_ERROR_INJEC)

Offset

Register	Offset
STSTEM_CACHE_DATA1_ECC_ERROR_INJEC	A8h

Function
Injects an ECC error into system cache DATA1. This register is used only for verification and debugging.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 SYSTEM_CACHE_DATA1_FRCNCI	Force Continuous Noncorrectable Data Inversions on System Cache DATA1 Write Access Injects a multibit ECC error on the bits specified by SYSTEM_CACHE_DATA1_ERR1BIT and SYSTEM_CACHE_DATA1_ERR2BIT into every write access to system cache DATA1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
18 SYSTEM_CACHE_DATA1_FRC1BI	Force Continuous 1-Bit Data Inversions on System Cache DATA1 Write Access Injects a single-bit ECC error on the bit specified by SYSTEM_CACHE_DATA1_ERR1BIT into every write access to system cache DATA1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
17	Force One Noncorrectable Data Inversion on System Cache DATA1 Write Access

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYSTEM_CACHE_DATA1_FR1NCI	Injects a multibit ECC error on the bits specified by SYSTEM_CACHE_DATA1_ERR1BIT and SYSTEM_CACHE_DATA1_ERR2BIT into the first write access to system cache DATA1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
16 SYSTEM_CACHE_DATA1_FR11BI	Force One 1-Bit Data Inversion on System Cache DATA1 Write Access Injects a single-bit ECC error on the bit specified by SYSTEM_CACHE_DATA1_ERR1BIT into the first write access to system cache DATA1 that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
15 —	Reserved
14-8 SYSTEM_CACHE_DATA1_ERR2BIT	Position of Second Bit to Inject ECC Error Determines the position of the second bit into which CM33 injects an ECC error. This field is used only for multibit ECC error injection.
7 —	Reserved
6-0 SYSTEM_CACHE_DATA1_ERR1BIT	Position of First Bit to Inject ECC Error Determines the position of the first bit into which CM33 injects an ECC error. This field can be used for both single-bit and multibit ECC error injection.

36.6.2 CM33_TCM_MCM register descriptions

36.6.2.1 CM33_TCM_MCM memory map

CP_CM33_IMX9RTC.CM33_TCM_MCM base address: 4442_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	TCM ECC Control (TCMECCR)	32	RW	0000_0000h
20h	Interrupt Status (INT_STATUS)	32	RW	0000_0000h
24h	Interrupt Status Enable (INT_STAT_EN)	32	RW	0000_0000h
28h	Interrupt Enable (INT_SIG_EN)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5Ch	Code TCM Single-Bit ECC Error Information (CODE_TCM_ECC_SINGLE_ERROR_INFO)	32	R	0000_0000h
60h	Code TCM Single-Bit ECC Error Address (CODE_TCM_ECC_SINGLE_ERROR_ADDR)	32	R	0000_0000h
68h	Code TCM Multibit ECC Error Information (CODE_TCM_ECC_MULTI_ERROR_INFO)	32	R	0000_0000h
6Ch	Code TCM Multibit ECC Error Address (CODE_TCM_ECC_MULTI_ERROR_ADDR)	32	R	0000_0000h
74h	System TCM Single-Bit ECC Error Information (SYS_TCM_ECC_SINGLE_ERROR_INFO)	32	R	0000_0000h
78h	System TCM Single-Bit ECC Error Address (SYS_TCM_ECC_SINGLE_ERROR_ADDR)	32	R	0000_0000h
80h	System TCM Multibit ECC Error Information (SYS_TCM_ECC_MULTI_ERROR_INFO)	32	R	0000_0000h
84h	System TCM Multibit ECC Error Address (SYS_TCM_ECC_MULTI_ERROR_ADDR)	32	R	0000_0000h
94h	Code TCM ECC Error Injection (CODE_TCM_ECC_ERROR_INJEC)	32	RW	0000_0000h
98h	System TCM ECC Error Injection (SYS_TCM_ECC_ERROR_INJEC)	32	RW	0000_0000h

36.6.2.2 TCM ECC Control (TCMECCR)

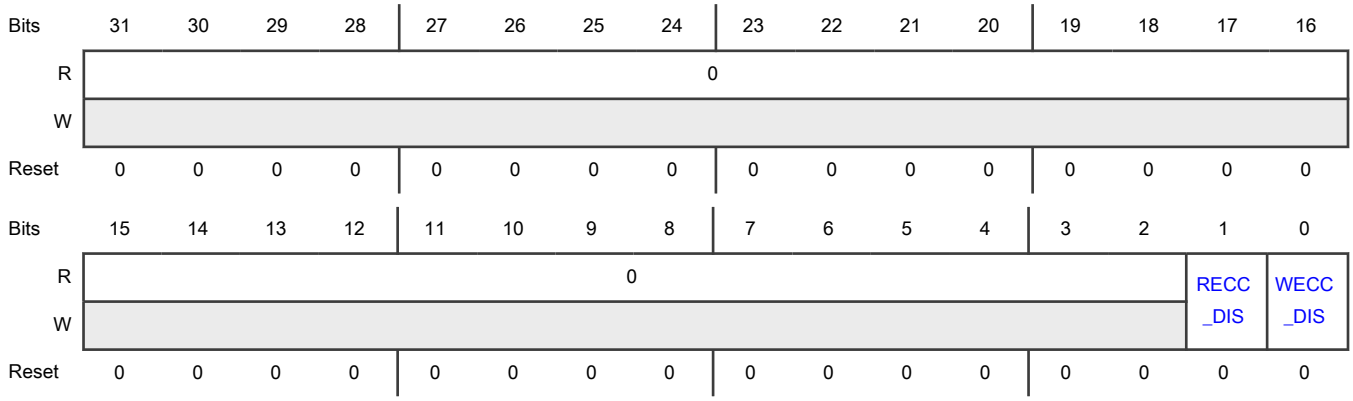
Offset

Register	Offset
TCMECCR	4h

Function

Disables the ECC logic for Cortex-M33 TCM. TCM ECC is enabled by default.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RECC_DIS	TCM ECC Read Check Disable Disables the ECC check on TCM read data. 0b - Enable 1b - Disable
0 WECC_DIS	TCM ECC Write Generation Disable Disables ECC generation on TCM write data. 0b - Enable 1b - Disable

36.6.2.3 Interrupt Status (INT_STATUS)

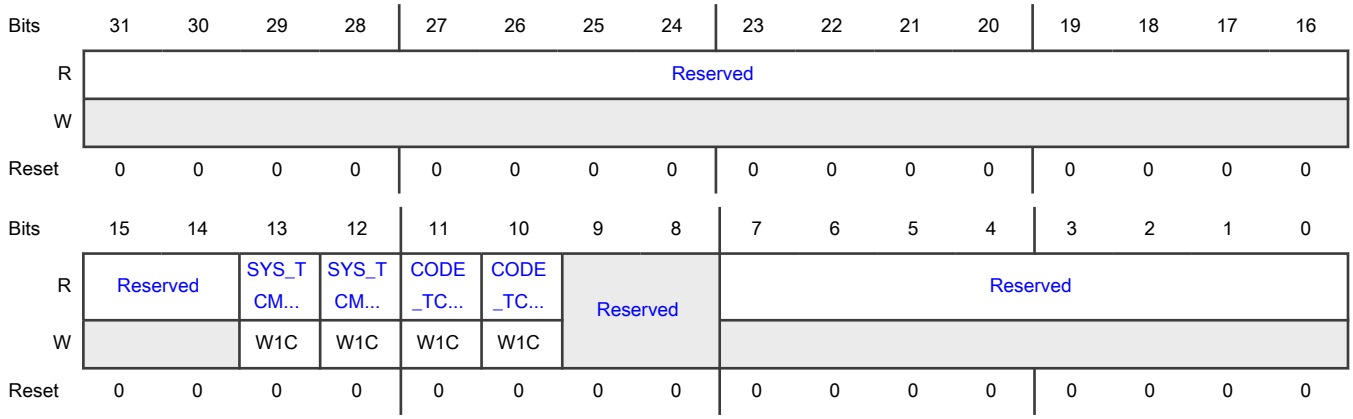
Offset

Register	Offset
INT_STATUS	20h

Function

Contains the status of ECC errors for system TCM and code TCM.

Diagram



Fields

Field	Function
31-14 —	Reserved
13 SYS_TCM_ECC_ERRS_INT	System TCM Access Single-Bit ECC Error Interrupt Status Indicates a single-bit ECC error in system TCM read data when INT_STAT_EN[SYS_TCM_ERRS_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
12 SYS_TCM_ECC_ERRM_INT	System TCM Access Multibit ECC Error Interrupt Status Indicates a multibit ECC error in system TCM read data when INT_STAT_EN[SYS_TCM_ERRM_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
11 CODE_TCM_ECC_ERRS_INT	Code TCM Access Single-Bit ECC Error Interrupt Status Indicates a single-bit ECC error in code TCM read data when INT_STAT_EN[CODE_TCM_ERRS_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
10 CODE_TCM_ECC_ERRM_INT	Code TCM Access Multibit ECC Error Interrupt Status Indicates a multibit ECC error in code TCM read data when INT_STAT_EN[CODE_TCM_ERRM_INT_EN] = 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
9-8 —	Reserved
7-0 —	Reserved

36.6.2.4 Interrupt Status Enable (INT_STAT_EN)

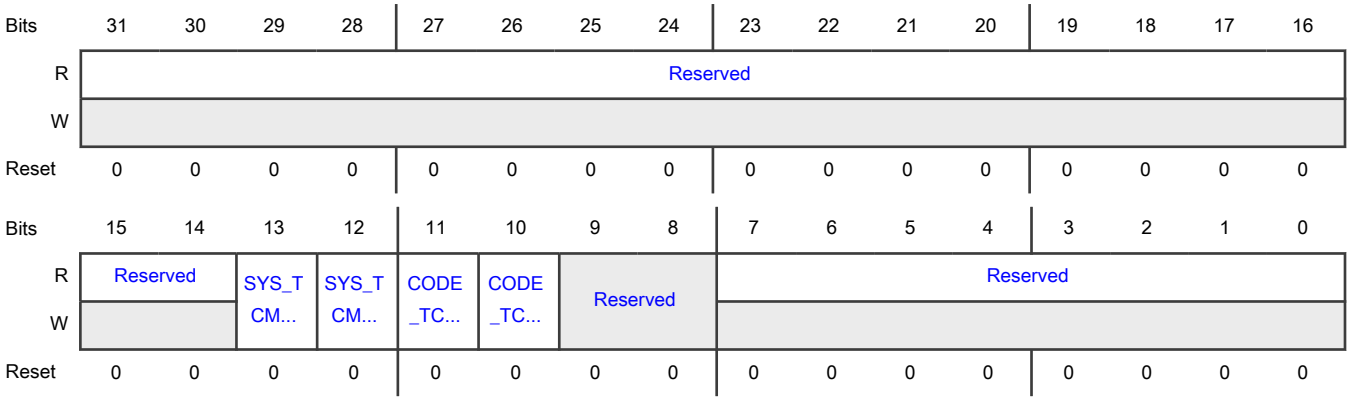
Offset

Register	Offset
INT_STAT_EN	24h

Function

Enables the interrupt statuses in Interrupt Status (INT_STATUS).

Diagram



Fields

Field	Function
31-14 —	Reserved
13 SYS_TCM_ERRS_INT_EN	System TCM Access Single-Bit ECC Error Interrupt Status Enable Enables INT_STATUS[SYS_TCM_ECC_ERRS_INT]. 0b - Mask 1b - Enable
12 SYS_TCM_ERRM_INT_EN	System TCM Access Multibit ECC Error Interrupt Status Enable Enables INT_STATUS[SYS_TCM_ECC_ERRM_INT]. 0b - Mask 1b - Enable
11 CODE_TCM_ERRS_INT_EN	Code TCM Access Single-Bit ECC Error Interrupt Status Enable Enables INT_STATUS[CODE_TCM_ECC_ERRS_INT]. 0b - Mask 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 CODE_TCM_ERRM_INT_EN	Code TCM Access Multibit ECC Error Interrupt Status Enable Enables INT_STATUS[CODE_TCM_ECC_ERRM_INT]. 0b - Mask 1b - Enable
9-8 —	Reserved
7-0 —	Reserved

36.6.2.5 Interrupt Enable (INT_SIG_EN)

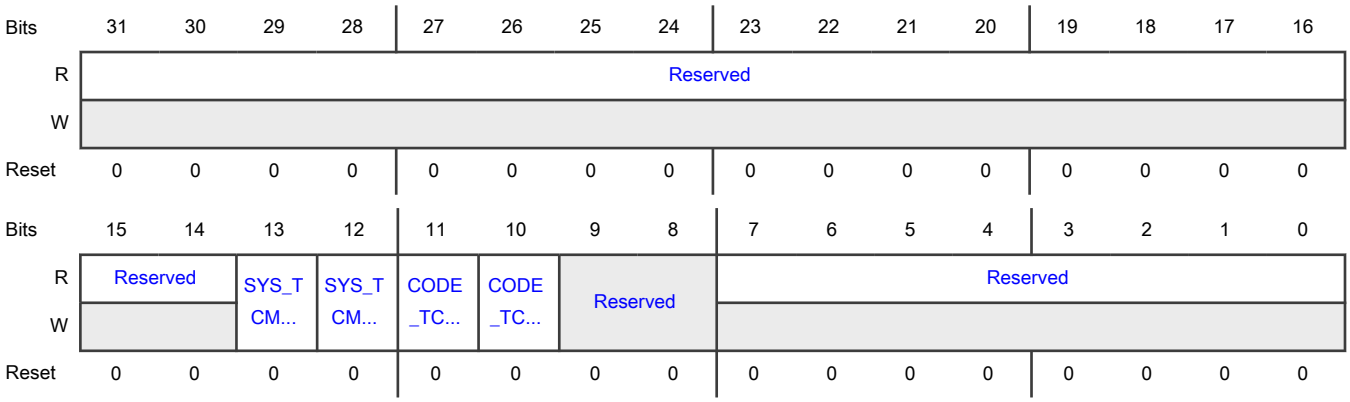
Offset

Register	Offset
INT_SIG_EN	28h

Function

Selects which interrupt statuses are available to interrupt the host system. All these status fields share the same interrupt line (that is, all interrupts are combined with a logical OR). Writing 1 to any of these fields enables interrupt generation. The corresponding field in Interrupt Status (INT_STATUS) generates an interrupt when the corresponding field in Interrupt Status Enable (INT_STAT_EN) is 1.

Diagram



Fields

Field	Function
31-14 —	Reserved
13 SYS_TCM_ERRS_INT_SIG_EN	System TCM Access Single-Bit ECC Error Interrupt Signal Enable Selects INT_STATUS[SYS_TCM_ECC_ERRS_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Mask 1b - Enable
12 SYS_TCM_ERRM_INT_SIG_EN	System TCM Access Multibit ECC Error Interrupt Signal Enable Selects INT_STATUS[SYS_TCM_ECC_ERRM_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Mask 1b - Enable
11 CODE_TCM_ERRS_INT_SIG_EN	Code TCM Access Single-Bit ECC Error Interrupt Signal Enable Selects INT_STATUS[CODE_TCM_ECC_ERRS_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Mask 1b - Enable
10 CODE_TCM_ERRM_INT_SIG_EN	Code TCM Access Multibit ECC Error Interrupt Signal Enable Selects INT_STATUS[CODE_TCM_ECC_ERRM_INT] as one of the interrupt statuses available to interrupt the host system. 0b - Mask 1b - Enable
9-8 —	Reserved
7-0 —	Reserved

36.6.2.6 Code TCM Single-Bit ECC Error Information (CODE_TCM_ECC_SINGLE_ERROR_INFO)

Offset

Register	Offset
CODE_TCM_ECC_SINGLE_ERROR_INFO	5Ch

Function

Stores detailed error information for the corresponding code TCM single-bit ECC interrupt when the interrupt is enabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													CODE_TCM_ECCS_EFSYN		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CODE_TCM_ECCS_EFSYN				CODE_TCM_ECCS_EFPRT				CODE_TCM_ECCS_EFMST				CODE_TCM_ECCS_EFSIZ			0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-12 CODE_TCM_ECCS_EFSYN	Code TCM Single-Bit ECC Error Corresponding Syndrome Contains the corresponding syndrome when a single-bit error is reported. You can use the ECC syndrome to locate the error bit via a lookup table. See CM33 TCM ECC support for more information.
11-8 CODE_TCM_ECCS_EFPRT	Code TCM Single-Bit ECC Error for Corresponding TCM Access Protection Contains the master access protection attribute (tcm_priv) of the access that created the error when a single-bit error is reported. For details about the protection attribute decoding, see the Arm AMBA AHB specification.
7-4 CODE_TCM_ECCS_EFMST	Code TCM Single-Bit ECC Error for Corresponding TCM Master Contains the code TCM single-bit ECC error for master number (manager identifier).
3-1 CODE_TCM_ECCS_EFSIZ	Code TCM Single-Bit ECC Error for Corresponding TCM Access Size Contains the code TCM single-bit ECC error for TCM access size. For code TCM, this value must be 2 (32 bits).
0 —	Reserved

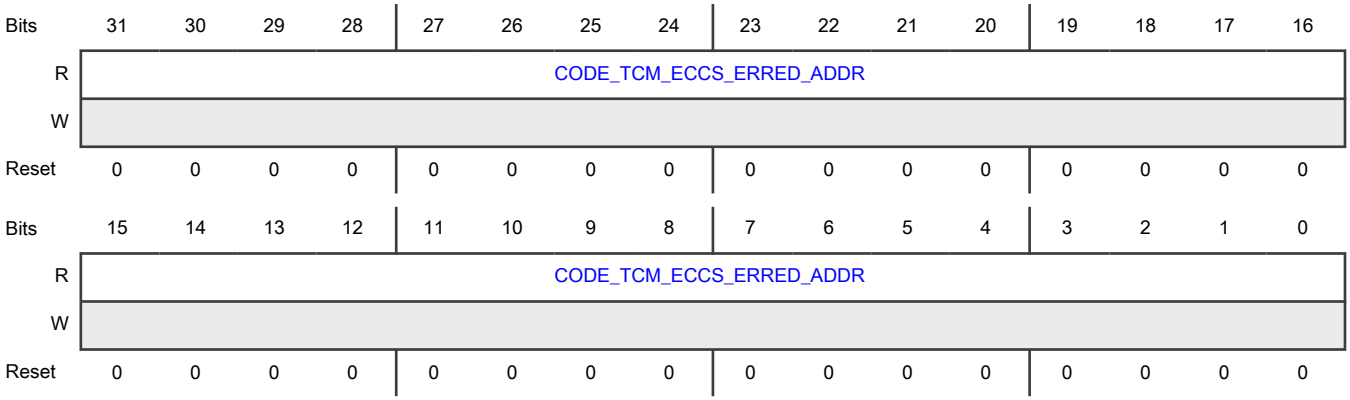
36.6.2.7 Code TCM Single-Bit ECC Error Address (CODE_TCM_ECC_SINGLE_ERROR_ADDR)

Offset

Register	Offset
CODE_TCM_ECC_SINGLE_ERROR_ADDR	60h

Function
Stores the relative error address for the corresponding code TCM single-bit ECC interrupt.

Diagram



Fields

Field	Function
31-0 CODE_TCM_ECCS_ERRED_ADDR	Code TCM Single-Bit ECC Error Address Stores the relative error address for the corresponding code TCM single-bit ECC interrupt when the interrupt is enabled.

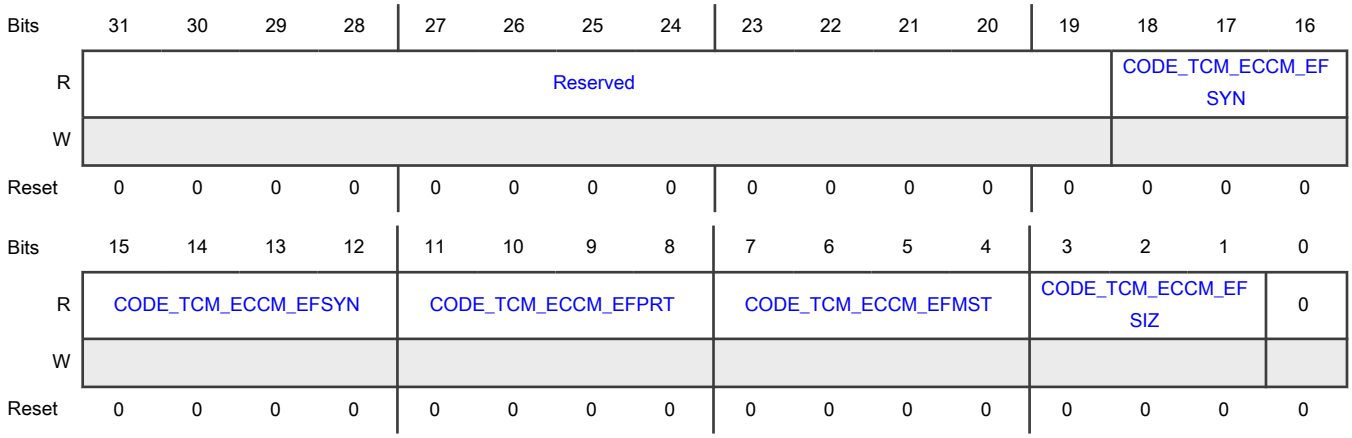
36.6.2.8 Code TCM Multibit ECC Error Information (CODE_TCM_ECC_MULTI_ERROR_INFO)

Offset

Register	Offset
CODE_TCM_ECC_MULTI_ERROR_INFO	68h

Function
Stores detailed error information for the corresponding code TCM multibit ECC interrupt when the interrupt is enabled.

Diagram



Fields

Field	Function
31-19 —	Reserved
18-12 CODE_TCM_ECCM_EFSYN	Code TCM Multibit ECC Error for Corresponding Syndrome Contains the corresponding syndrome when a multibit error is reported. You cannot use the syndrome for multibit ECC to locate the error bit.
11-8 CODE_TCM_ECCM_EFPRT	CODE_TCM Multibit ECC Error for Corresponding Access Protection Attribute Contains the master access protection attribute (tcm_priv) of the access that created the error when a multibit error is reported. For details about the protection attribute decoding, see the Arm AMBA AHB specification.
7-4 CODE_TCM_ECCM_EFMST	Code TCM Multibit ECC Error for Corresponding TCM Master Contains the code TCM multibit ECC error master number (manager identifier).
3-1 CODE_TCM_ECCM_EFSIZ	Code TCM Multibit ECC Error for Corresponding TCM Access Size Contains the code TCM multibit ECC error for TCM access size. For code TCM, this value must be 2 (32 bits).
0 —	Reserved

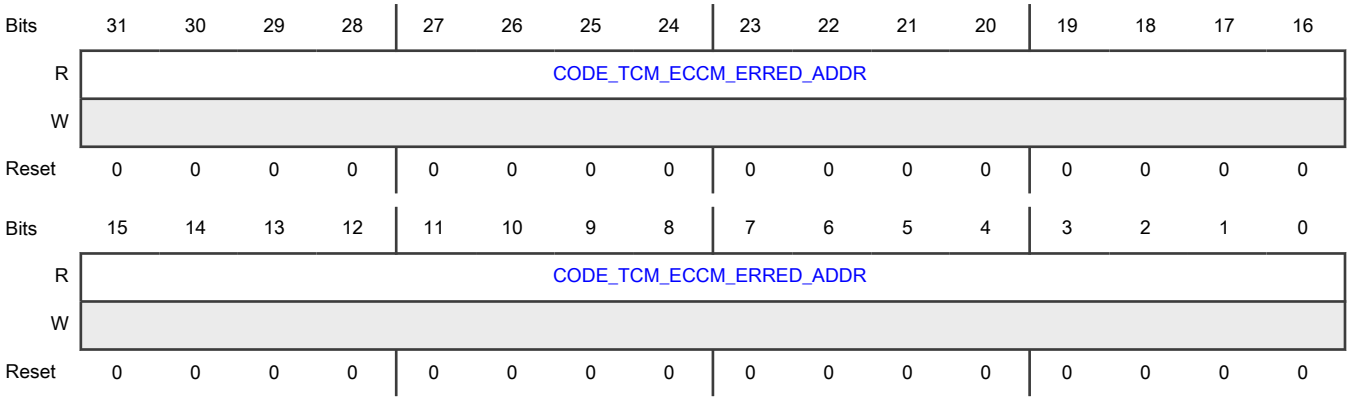
36.6.2.9 Code TCM Multibit ECC Error Address (CODE_TCM_ECC_MULTI_ERROR_ADDR)

Offset

Register	Offset
CODE_TCM_ECC_MULTI_ERROR_ADDR	6Ch

Function
Stores the relative error address for the corresponding code TCM multibit ECC interrupt.

Diagram



Fields

Field	Function
31-0 CODE_TCM_ECCM_ERRED_ADDR	Code TCM Multibit ECC Error Address Stores the relative error address for the corresponding code TCM multibit ECC interrupt when the interrupt is enabled.

36.6.2.10 System TCM Single-Bit ECC Error Information (SYS_TCM_ECC_SINGLE_ERROR_INFO)

Offset

Register	Offset
SYS_TCM_ECC_SINGLE_ERROR_INFO	74h

Function
Stores detailed error information for the corresponding system TCM single-bit ECC interrupt when the interrupt is enabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													SYS_TCM_ECCS_EFSYN		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SYS_TCM_ECCS_EFSYN				SYS_TCM_ECCS_EFPRT				SYS_TCM_ECCS_EFMST				SYS_TCM_ECCS_EFSIZ			0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-12 SYS_TCM_ECCS_EFSYN	System TCM Single-Bit ECC Error for Corresponding Syndrome Contains the corresponding syndrome when a single-bit error is reported. You can use the ECC syndrome to locate the error bit via a lookup table. See CM33 TCM ECC support for more information.
11-8 SYS_TCM_ECCS_EFPRT	System TCM Single-Bit ECC Error for Corresponding Access Protection Attribute Contains the master access protection attribute (tcm_priv) of the access that created the error when a single-bit error is reported. For details about the protection attribute decoding, see the Arm AMBA AHB specification.
7-4 SYS_TCM_ECCS_EFMST	System TCM Single-Bit ECC Error for Corresponding TCM Master Contains the system TCM single-bit ECC error master number (manager identifier).
3-1 SYS_TCM_ECCS_EFSIZ	System TCM Single-Bit ECC Error for Corresponding TCM Access Size Contains the code TCM single-bit ECC error for TCM access size. For system TCM, this value must be 2 (32 bits).
0 —	Reserved

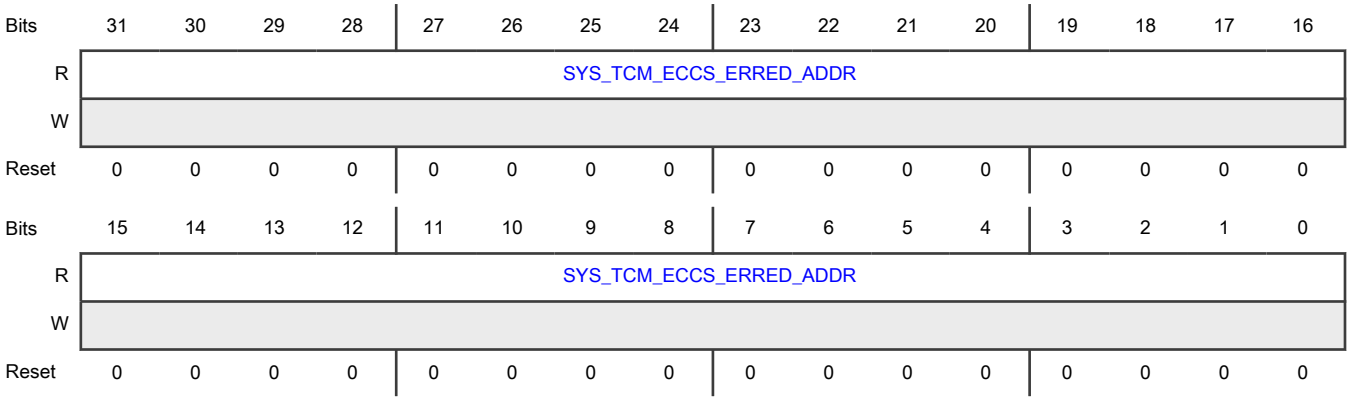
36.6.2.11 System TCM Single-Bit ECC Error Address (SYS_TCM_ECC_SINGLE_ERROR_ADDR)

Offset

Register	Offset
SYS_TCM_ECC_SINGLE_ERROR_ADDR	78h

Function
Stores the relative error address for the corresponding system TCM single-bit ECC interrupt.

Diagram



Fields

Field	Function
31-0 SYS_TCM_ECCS_ERRED_ADDR	System TCM Single-Bit ECC Error Address Stores the relative error address for the corresponding system TCM single-bit ECC interrupt when the interrupt is enabled.

36.6.2.12 System TCM Multibit ECC Error Information (SYS_TCM_ECC_MULTI_ERROR_INFO)

Offset

Register	Offset
SYS_TCM_ECC_MULTI_ERROR_INFO	80h

Function
Stores detailed error information for the corresponding system TCM multibit ECC interrupt when the interrupt is enabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													SYS_TCM_ECCM_EFSYN		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SYS_TCM_ECCM_EFSYN				SYS_TCM_ECCM_EFPRT				SYS_TCM_ECCM_EFMST				SYS_TCM_ECCM_EFSIZ			0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-12 SYS_TCM_ECCM_EFSYN	System TCM Multibit ECC Error for Corresponding Syndrome Contains the corresponding syndrome when a multibit error is reported. You cannot use the syndrome for multibit ECC to locate the error bit.
11-8 SYS_TCM_ECCM_EFPRT	System TCM Multibit ECC Error for Corresponding Access Protection Attribute Contains the master access protection attribute (tcm_priv) of the access that created the error when a multibit error is reported. For details about the protection attribute decoding, see the Arm AMBA AHB specification.
7-4 SYS_TCM_ECCM_EFMST	System TCM Multibit ECC Error for Corresponding TCM Master Contains the code TCM multibit ECC error master number (manager identifier).
3-1 SYS_TCM_ECCM_EFSIZ	System TCM Multibit ECC Error for Corresponding TCM Access Size Contains the system TCM multibit ECC error for TCM access size. For system TCM, this value must be 2 (32 bits).
0 —	Reserved

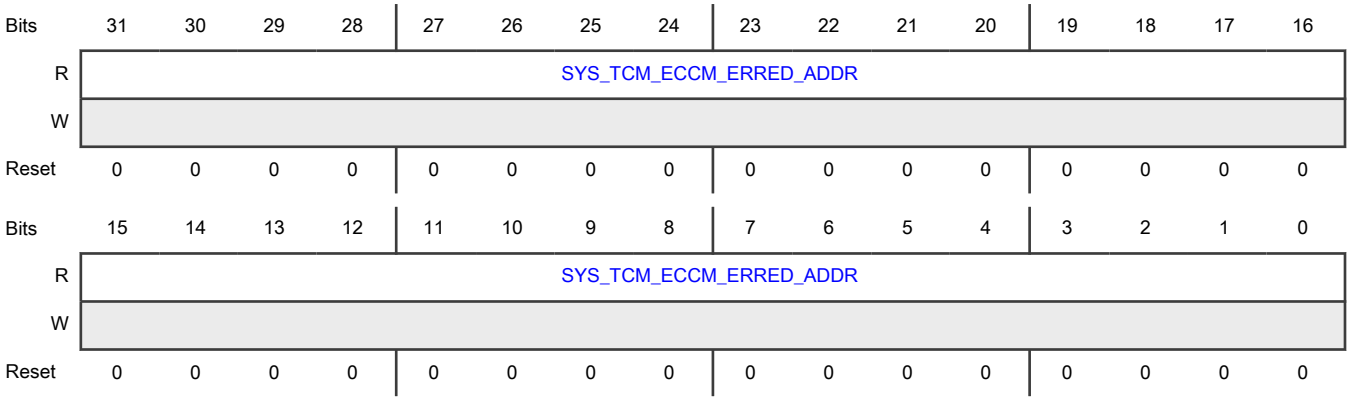
36.6.2.13 System TCM Multibit ECC Error Address (SYS_TCM_ECC_MULTI_ERROR_ADDR)

Offset

Register	Offset
SYS_TCM_ECC_MULTI_ERROR_ADDR	84h

Function
Stores the relative error address for the corresponding system TCM multibit ECC interrupt.

Diagram



Fields

Field	Function
31-0 SYS_TCM_ECCM_ERRED_ADDR	System TCM Multibit ECC Error Address Stores the relative error address for the corresponding system TCM multibit ECC interrupt when the interrupt is enabled.

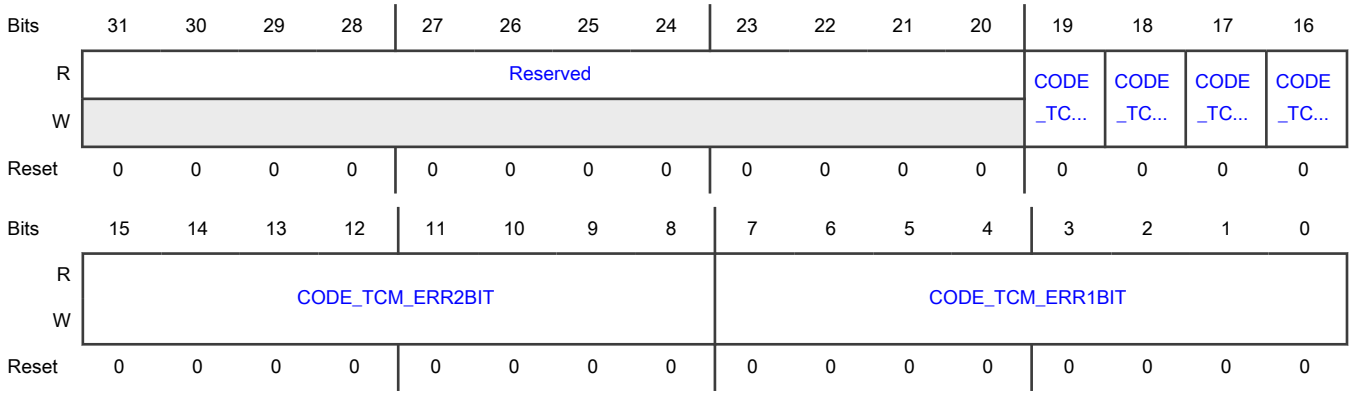
36.6.2.14 Code TCM ECC Error Injection (CODE_TCM_ECC_ERROR_INJEC)

Offset

Register	Offset
CODE_TCM_ECC_ERROR_INJEC	94h

Function
Injects an ECC error into code TCM. This register is used only for verification and debugging.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 CODE_TCM_FRCNCI	Force Continuous Noncorrectable Data Inversions on Code TCM Write Access Injects a multibit ECC error on the bits specified by CODE_TCM_ERR1BIT and CODE_TCM_ERR2BIT into every write access to code TCM that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
18 CODE_TCM_FRC1BI	Force Continuous 1-Bit Data Inversions on Code TCM Write Access Injects a single-bit ECC error on the bit specified by CODE_TCM_ERR1BIT into every write access to code TCM that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
17 CODE_TCM_FR1NCI	Force One Noncorrectable Data Inversion on Code TCM Write Access Injects a multibit ECC error on the bits specified by CODE_TCM_ERR1BIT and CODE_TCM_ERR2BIT into the first write access to code TCM that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
16 CODE_TCM_FR11BI	Force One 1-Bit Data Inversion on Code TCM Write Access Injects a single-bit ECC error on the bit specified by CODE_TCM_ERR1BIT into the first write access to code TCM that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
15-8	Position of Second Bit to Inject ECC Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
CODE_TCM_ERR2BIT	Determines the position of the second bit into which CM33 injects an ECC error. This field is used only for multibit ECC error injection.
7-0 CODE_TCM_ERR1BIT	Position of First Bit to Inject ECC Error Determines the position of the first bit into which CM33 injects an ECC error. This field can be used for both single-bit and multibit ECC error injection.

36.6.2.15 System TCM ECC Error Injection (SYS_TCM_ECC_ERROR_INJEC)

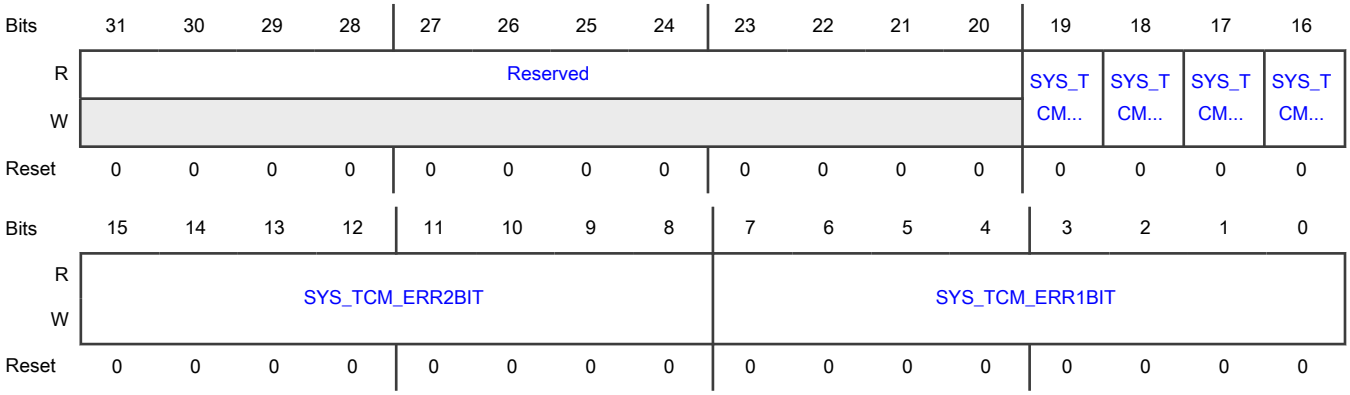
Offset

Register	Offset
SYS_TCM_ECC_ERROR_INJEC	98h

Function

Injects an ECC error into system TCM. This register is used only for verification and debugging.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 SYS_TCM_FRCNCI	Force Continuous Noncorrectable Data Inversions on System TCM Write Access Injects a multibit ECC error on the bits specified by SYS_TCM_ERR1BIT and SYS_TCM_ERR2BIT into every write access to system TCM that occurs after you write 1 to this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable injection 1b - Enable injection
18 SYS_TCM_FRC1BI	Force Continuous 1-Bit Data Inversions on System TCM Write Access Injects a single-bit ECC error on the bit specified by SYS_TCM_ERR1BIT into every write access to system TCM that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
17 SYS_TCM_FR1NCI	Force One Noncorrectable Data Inversion on System TCM Write Access Injects a multibit ECC error on the bits specified by SYS_TCM_ERR1BIT and SYS_TCM_ERR2BIT into the first write access to system TCM that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
16 SYS_TCM_FR11BI	Force One 1-Bit Data Inversion on System TCM Write Access Injects a single-bit ECC error on the bit specified by SYS_TCM_ERR1BIT into the first write access to system TCM that occurs after you write 1 to this field. 0b - Disable injection 1b - Enable injection
15-8 SYS_TCM_ERR2BIT	Position of Second Bit to Inject ECC Error Determines the position of the second bit into which CM33 injects an ECC error. This field is used only for multibit ECC error injection.
7-0 SYS_TCM_ERR1BIT	Position of First Bit to Inject ECC Error Determines the position of the first bit into which CM33 injects an ECC error. This field can be used for both single-bit and multibit ECC error injection.

Chapter 37
Messaging Unit (MU)

37.1 Chip-specific MU Information

There are two MU instances.

Table 275. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

37.2 Overview

MU enables two processors on a chip to communicate and coordinate by passing messages (for example, data, status, and control) through the MU interface. MU also allows one processor to signal the other processor using interrupts.

MU must synchronize the accesses from one side to the other because MU can manage messaging between processors using different clocks. MU accomplishes synchronization using two sets of matching registers: processor A-facing and processor B-facing.

37.2.1 Block diagram

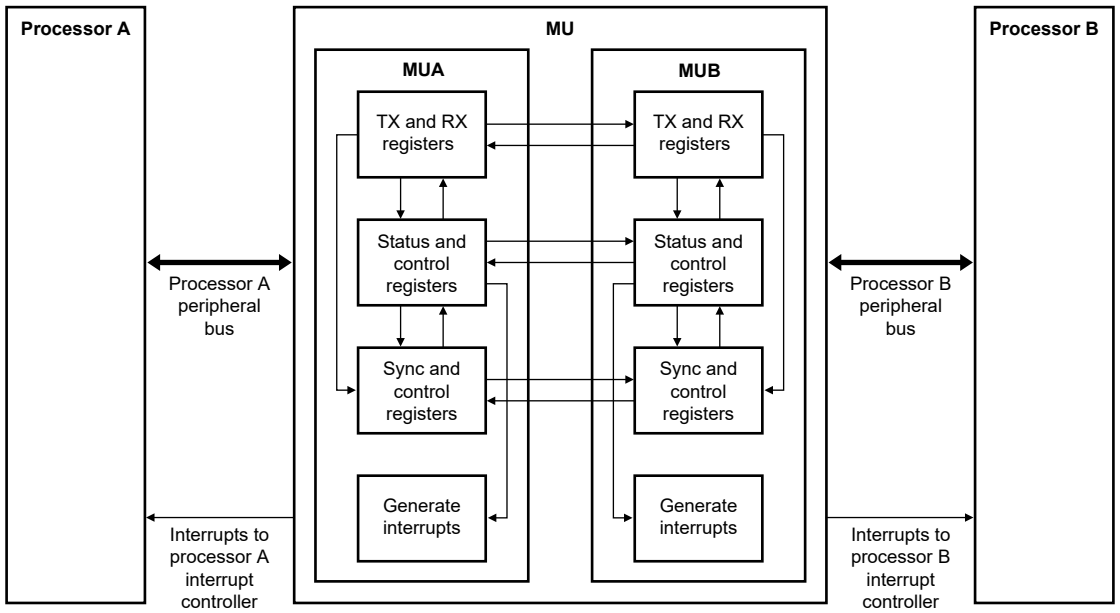


Figure 275. Block diagram

37.2.2 Features

- Memory-mapped registers (MU is connected as a peripheral under the peripheral bus on the processor A-side and processor B-side)
- Synchronized message transfers between cores
  - To send data or messages from one side to the other, messaging unit A (MUA) provides 4 transmit registers and 4 receive registers. Messaging unit B (MUB) provides 4 transmit registers and 4 receive registers.
  - Transmit empty flags (TSR[TEn]) and receive full flags (RSR[RFn]) facilitate the transfer of data or messages between cores on both sides of MU.
  - A synchronization mechanism updates the transmit and receive flags. There is an inherent latency between updating the flag on one side and reflecting its status on the other side. See Event update timing.
  - MU has a 3-bit flag data register, which you can use to send flag data between the two MU sides.
- Interprocessor interrupts
  - MU has 12 interrupt sources on each side (processor A-side, processor B-side) for signaling the other processor. You can use the interrupts for notification of receive and transmit events and for general-purpose signaling between processors. There are 4 general-purpose interrupt requests available and 8 receive and transmit interrupt sources.
- Reset (Each processor can issue a reset to MU via CR[MUR], which is a self-clearing field)

37.3 Functional description

MU enables two cores (processor A and processor B) to communicate with each other:

- By sharing messages and data.
- By enabling one core to wake the other core by using interrupts.

The messaging, control, and status registers of the two cores are mapped to processor A memory and processor B memory as a regular peripheral. The peripheral data bus is 32 bits wide inside MU.

Messaging logic is used with shared memory. Various messaging methods can implement a messaging protocol. For example, a message could mean one of the following:

- A message of *n* words has been written starting at offset *x* in the memory.
- The previous data block that was sent has been read.

The ability to keep messaging logic independent of the shared memory is not restricted to a predefined hardware protocol. The software required to manage the messaging is short and straightforward.

Most of the messaging mechanisms are symmetric. They are duplicated and are available on both the processor A side and processor B side. The messaging mechanisms are:

- 4 32-bit transmit registers, which are each reflected in 4 read-only receive registers on the side of the other processor. These registers can transfer 32-bit word messages or the frame information of the messages written to the shared memory. For example, they can transfer the number of words, initial address, and message type code.
- Writing to a transmitter-side transmit register (TR*n*) clears the Transmitter Empty flag in the transmitter-side Status register (TS*n*[TE*n*]), and sets the Receiver Full flag in the receiver-side Status register (RS*n*[RF*n*]). Setting the flag on the receiver side can trigger an interrupt on the receiver side (a maskable receive interrupt).
- Reading a receiver-side receive register (RR*n*) clears the Receiver Full flag in the receiver-side Status register (RS*n*[RF*n*]), and sets the Transmitter Empty flag in the transmitter-side Status register (TS*n*[TE*n*]). Setting the Transmitter Empty flag can trigger an interrupt on the transmitter side (a maskable transmit interrupt).
- 4 general-purpose interrupt request flags are reflected in General-purpose Status (GSR) on the receiver side.
- 3-bit flag data is transmitted from Flag Control (FCR) to Flag Status (FSR) on the side receiving the flag data. SR[FUP] sets when the flag data is transmitted and clears when the receiving side acknowledges the flag data (the flag is updated).

Writing to a transmit register signals to the receiver side that data is ready for retrieval.

Do not write to the transmit register again without verifying that the data is retrieved. The transmitter side cannot determine the exact time that the receiver attempts to retrieve the data. Before attempting to write to the transmit register again, the transmitter side must wait for the Transmitter Empty interrupt or it must poll TSR[TE*n*].

Reading a receive register signals to the transmitter side that data can be written to that register.

Do not read the receive register again without verifying that the data is written. The receiver side cannot determine the exact time that the transmitter attempts to write the data. Before attempting to read the receive register again, the receiver side must wait for the Receiver Full interrupt or it must poll RSR[RF*n*].

37.3.1 Submodules

37.3.1.1 MUA side

MUA is receives its register configuration via the processor A peripheral bus. It sends or receives messages to or from MUB. Processor A can receive messages by reading MUA registers, and MUA can send interrupts to processor A when interrupts are enabled.

NOTE

Processor B is not allowed to access MUA register. Processor C, if exist, is also not allowed to access MUA register.

TRDC might be needed to prevent such illegal access.

37.3.1.2 MUB side

MUB receives its register configuration via the processor B peripheral bus. It sends or receives messages to or from MUA. Processor B can receive messages by reading MUB registers, and MUB can send interrupts to processor B when interrupts are enabled.

NOTE

Processor A is not allowed to access MUB register. Processor C, if exist, is also not allowed to access MUB register.

TRDC might be needed to prevent such illegal access.

37.3.2 Event update timing

The messaging side of each processor has a hardware mechanism to send event update requests to the other processor. An event occurs when the status register of the receiving processor must reflect any information change. The event update latency is the delay between the event being ready at one processor and the resulting update at the status register of the other processor:

- The minimum event latency is (1 clock cycle of the sending side) + (2.5 clock cycles of the receiving side). This minimum case happens when no event is pending when the new event occurs.
- The maximum event latency is (6 clock cycles of the sending side) + (6.5 clock cycles of the receiving side). The maximum case happens when the event occurs just after a previous event is sent to the other side.

The event update latency varies depending on the time at which the subsequent event is triggered.

37.3.3 Clocking

Table 276. MU clocks

Clock name	Description
Bus clock MUA	Used only for bus accesses to MUA control and configuration registers
Bus clock MUB	Used only for bus accesses to MUB control and configuration registers

37.3.4 Resets

The following sections list the reset sources included in MU. Each reset performs a different function for the MU module compared to the function it performs for the system.

37.3.4.1 Asynchronous system reset

When the asynchronous system reset on either side of MU is asserted, SR[MURS] becomes 1 until the asynchronous system reset sequence on both the MUA and MUB sides ends. Verify that SR[MURS] becomes 0 before accessing MU.

The asynchronous system reset on one side of MU resets the other side of MU. The reset forces all control and status registers to return to their default values and clears all internal states. The following table shows the exceptions to this behavior.

Table 277. Exceptions to asynchronous system reset

MUA-side exceptions	MUB-side exceptions
MUB_CCR0[HR]	MUA_CCR0[HR]
MUB_CCR0[HRM]	MUA_CCR0[HRM]
MUB_CR[MURIE]	MUA_CR[MURIE]
MUB_SR[MURIP]	MUA_SR[MURIP]
MUB_SR[MURS]	MUA_SR[MURS]

37.3.4.2 MU software reset

Writing 1 to CR[MUR] causes most control and status registers to return to their default values and clears all internal states.

The instruction immediately following the assertion of CR[MUR] must not write to the MU registers. The reset sequence may overwrite such a write, with the register retaining the reset value. To know the end of the reset sequence for both processors,

monitor the value of SR[MURS]. After the reset sequence on both processors has ended, a write to the MU registers can be attempted.

NOTE

The process of CR[MUR] becoming 1 is delicate because it asynchronously affects the registers on the other side. CR[MUR] becoming 1 may cause unpredictable behavior if, for example, the other processor is concurrently testing an MU register field (TSR[TE*n*] in the other processor). Before writing 1 to CR[MUR], verify that the other processor is not engaged in an MU signaling activity.

37.3.5 Interrupts

MU controls interrupt requests that one processor makes to the other processor. This section describes all the interrupts that the module generates.

MU can generate these interrupt sources individually to send to the processors:

- 4 receive interrupts (asserted when the Receive Full flags are set in Receive Status (RSR) and enabled in Receive Control (RCR)) for each receive register
- 4 transmit interrupts (asserted when the Transmit Empty flags are set in Transmit Status (TSR) and enabled in Transmit Control (TCR)) for each transmit register
- 4 general-purpose interrupts (asserted when the GIP flags are set in General-Purpose Interrupt Enable (GIER) and enabled in General-Purpose Interrupt Enable (GIER))

All interrupts are maskable in the processor control registers: TCR, RCR, GIER, and CR. MU does not assume any internal priority of these interrupts. Multiple interrupts (for example, receive 0 and receive 1, or any transmit or general-purpose interrupts) can be asserted simultaneously. The interrupt controller must resolve the priority of these interrupts at the chip level.

Triggering any enabled interrupt wakes the processor from below mode before servicing the interrupt

The software (as part of the interrupt handler) must clear the general-purpose interrupt pending flags (GSR[GIP*n*]) to deassert the request to the interrupt controller.

When a processor writes to the general-purpose interrupt (GCR[GIR*n*]), MU synchronizes the write event to the other processor to set the general-purpose interrupt request pending flag (GSR[GIP*n*]). When GSR[GIP*n*] is set and the general-purpose interrupt is enabled on the receiving side (GIER[GIE*n*] is 1), the transmitting-side general-purpose interrupt is issued to the receiving processor, which clears this interrupt by writing 1 to GSR[GIP*n*]. The interrupt is deasserted as soon as the write to GSR[GIP*n*] occurs. MU synchronizes the write event of GSR[GIP*n*] to the other processor. The synchronized signal writes 0 to the GIR*n* interrupt.

Before writing 1 to GCR[GIR*n*], verify that the field is 0, which means that a general interrupt is not pending. Generally, MU ignores writing 1 to this field while the field is already 1, but in some cases it may issue a second interrupt.

37.4 External signals

This module has no external signals.

37.5 Initialization

37.5.1 Sending messages with interrupt

Perform the following procedure in which, for example, processor A sends messages to processor B:

1. Prepare processor A with the data to be sent. Write data to MUA_TR0, MUA_TR1, MUA_TR2, and MUA_TR3.
2. Configure the interrupts.
    a. Write 1 to MUA_TCR[TIE3] to enable the interrupt to processor A.
    b. Write 1 to MUB_RCR[RIE3] to enable the interrupt to processor B.

3. Wait for processor B to enter into interrupt service. Processor B reads MUB_RR0, MUB_RR1, MUB_RR2, and MUB_RR3. If processor B continues to receive data, it can exit the interrupt service program without MUB_RCR[RIE3] becoming 0.

   Because processor B has read all the data, processor A receives the interrupt.

4. Write data to MUA_TR0, MUA_TR1, MUA_TR2, and MUA_TR3. If processor A continues to send data, it can exit the interrupt service program without MUA_TCR[TIE3] becoming 0.

NOTE

The aforementioned example is not the only method to send or receive messages for MU. You must configure MU for methods specific to your application.

37.6 Application information

MU facilitates messages between processors. For example, MU passes:

- Short messages. Transmit registers can pass short messages from 1 to 4 words in length for processor A and from 1 to 4 words for processor B. For example, for a four-word message, only one register must have its corresponding interrupt enabled on the receiving side. The first three words of the message are written to the registers with masked interrupts. The fourth word is written to the last register, triggering an interrupt on the receiving side.
- Frame information. Transmit registers can pass frame information for long messages written to the shared system memory. Such frame information normally includes a start address and a number of words. It can include a message type code.
- Event notices and requests. MU can signal events and requests that do not include data words between processors using general-purpose interrupts. For example, one such event is acknowledging that a long message is read from the shared system memory.
- Fixed-length data. Formatted data with a fixed length can be written to predetermined locations in the shared memory. A processor can use general-purpose interrupts to signal to the other processor that the data is ready.
- Announcements. A processor can use the 3 flags to announce its current program state or other billboard messages to the other processor.

37.6.1 Messaging protocols using interrupts

The example below describes a four-word messaging sequence sent between the processor A and processor B.

The transmitting processor writes to the transmit registers sequentially. When n = 0, 1, and 2, the interrupts are disabled, so no interrupt goes to the processor (although interrupt conditions occur). For n = 3, the interrupt is enabled, and MU generates the last receive interrupt request.

1. Write sequence:
   a. The transmitting processor writes the message information sequentially to its Transmit registers 0, 1, 2.
   b. When the write to Transmit register 3 occurs, RSR[RF3] is set after synchronization. It immediately triggers the receive full 3 interrupt on the receiving processor.
2. Read sequence:
   a. The receiving processor receives the receive full 3 interrupt and starts reading the message transferred from the receive registers.
   b. After the receiving processor reads Receive register 3, MU clears RSR[RF3].

Table 278 and Figure 276 describe the messaging model, using transmit and receive registers and the interrupt messaging protocol.

Table 278. Interrupt messaging protocol (generalized)

Step	Action	Description
1	Processor A writes data.	RR*n* on processor B's side reflects a data write to TR*n* by processor A.
2	Clear the transmitter empty flag and set the receiver full flag.	The data write to TR*n*: • Clears TSR[TE*n*] on the processor A side. • Sets RSR[RF*n*] on the processor B side.
3	Generate the receive interrupt request.	Setting RSR[RF*n*] generates a receive interrupt request to processor B.
4	Processor B reads the data.	After receiving the receive interrupt request, processor B performs a data read of RR*n*.
5	Clear the receiver full flag and set the transmitter empty flag.	Reading the data from the RR*n* register: • Clears RSR[RF*n*] on the processor B side. • Sets TSR[TE*n*] on the processor A side.
6	Generate the transmit interrupt request.	Setting TSR[TE*n*] generates a transmit interrupt request to processor A.

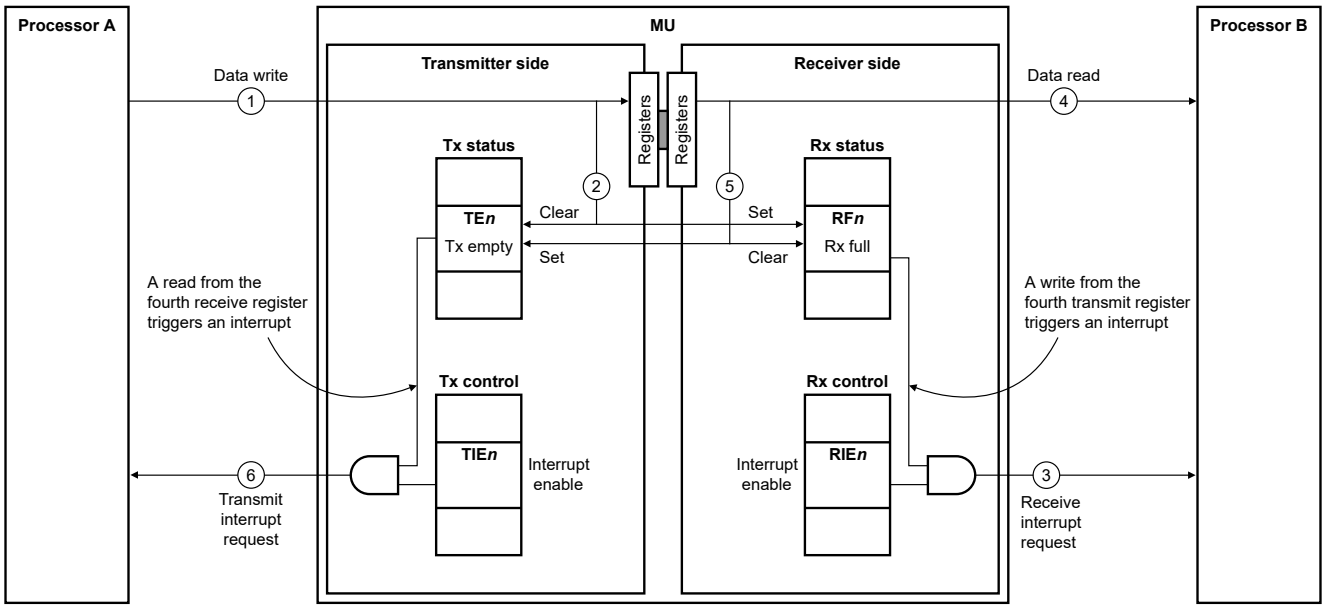


Figure 276. Messaging model using transmit and receive registers

You can use the messaging hardware to implement messaging protocols for an array of message types. MU provides full support for interrupt and polling management schemes.

37.6.2 Messaging protocols using event interrupts

MU can signal events and requests that do not include data words between processors using general-purpose interrupts.

Formatted data with a fixed length can be written to predetermined locations in the shared memory. A processor can use a general-purpose interrupt to signal to the other processor that the data is ready.

A processor can use the 3 flags to announce its current program state (or similar messages) to the other processor.

Table 279 and Figure 277 describe the event steps when the processor triggers a general-purpose interrupt.

Table 279. General-purpose interrupt messaging protocol (generalized)

Step	Action	Description
1	Processor A sets its associated general-purpose interrupt request flag.	Processor A sets GCR[GIR*n*].
2	The general-purpose interrupt request pending status flag is set.	GSR[GIP*n*] is set.
3	MU generates the general-purpose interrupt request to processor B.	Setting GSR[GIP*n*] generates the general-purpose interrupt request to processor B. GIER[GIE*n*] must be set for processor B.
4	Processor B reads the status register.	Processor B reads GSR[GIP*n*].
5	Processor B services the interrupt.	—
6	Processor B sets GSR[GIP*n*] to clear the interrupt.	Processor B writes 1 to the corresponding GSR[GIP*n*] flag to clear the interrupt.
7	GCR[GIR*n*] is cleared.	Setting GSR[GIP*n*] clears GCR[GIR*n*] on the processor A side.

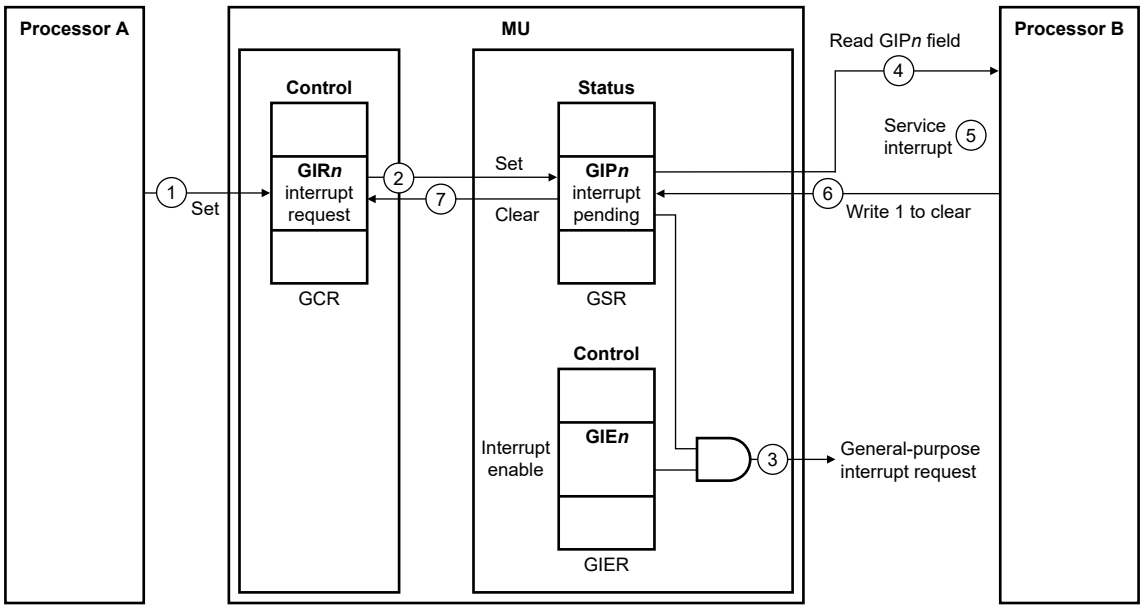


Figure 277. Messaging model using a general-purpose interrupt

37.6.3 Exclusive access to shared memory

MU can signal one processor about its current access to shared memory. This signaling prevents the other processor from overwriting data during the exclusive memory access period.

The following tables describe the signaling protocol that processor A uses to inform processor B about its current access (write) to shared memory:

- Table 280 shows processor A performing an exclusive access to shared memory.
- Table 281 shows processor B scanning for transaction information.
- Table 282 shows processor B accepting exclusive access from processor A.
- Table 283 shows processor B rejecting exclusive access from processor A.

According to the examples shown in the following tables, GCR[GIR*n*], RR*n*, and TR*n* are reserved to support exclusive access to the shared memory protocol.

Table 280. Processor A performs an exclusive access to shared memory

Step	Action	Description
1	Processor A sends the GIR*n* request to processor B using the processor A control register.	Before processor A performs an exclusive access to the shared memory, it sends a GIR*n* request to processor B.
2	Processor A sends an exclusive-access request using a transmit data register (TR*n*).	Processor A sends an exclusive-access request (command, location, and length of target access) to processor B using a selected transmit data register (TR0).
3	Processor A waits for a dedicated interrupt from processor B.	Processor A waits for a dedicated interrupt (as an acknowledgment) triggered by processor B before proceeding.
4	Processor A accesses the shared memory.	After receiving a dedicated interrupt from processor B, processor A proceeds.

Table 281. Processor B scans for transaction information

Step	Action	Description
1	Processor B receives an interrupt from a receive data register (RR*n*).	—
2	Processor B reads the receive data register (RR*n*).	—
3	Processor B scans the receive data register contents.	Processor B scans for transaction information (whether processor A has requested exclusive access).

Table 282. Processor B accepts exclusive access from processor A

Step	Action	Description
1	Processor B triggers a dedicated interrupt.	Processor B acknowledges the request from processor A by triggering a dedicated interrupt (ack) to processor A.
2	Processor B sends a code message to processor A.	With the acknowledge interrupt, processor B sends a code message to processor A through the selected transmit register (TR*n*). The message informs processor A that it can exclusively access the shared memory.

Table 283. Processor B rejects exclusive access from processor A

Step	Action	Description
1	Processor B ignores the request from processor A for exclusive access.	If processor B does not provide permission to processor A, processor B ignores the exclusive-access request.

37.6.4 Packet data transfers

The following table describes an example packet transfer sequence between processor B and processor A subsystems.

Table 284. Packet data transfer sequence

Step	Action	Description
1	Processor B requests DMA.	Processor B sends a DMA request to initiate the packet data transfer.
2	DMA data transfer	DMA acknowledges.
3		DMA starts transferring data from the specified processor B memory location to the specified shared memory.
4		DMA interrupts processor B to signal that the packet transfer has finished.
5	Processor B informs processor A that data is in shared memory.	Using a B-side transmit register, processor B sends a packet information message to processor A about the arrival of new packet data stored in shared memory. The message contains the command, location, and length of packet data.
6	Processor A receives an interrupt.	Processor A receives an interrupt (assuming its corresponding processor A MU-side receive interrupt is enabled). The pending processing task becomes active and processes packet data from memory.
7	Processor A reads data, then writes data.	Processor A reads or processes packet data from shared memory.
8		Processor A writes the result from packet processing to a separate buffer.
9	Processor A informs processor B that the transfer is finished.	After the processing of the packet data finishes, processor A informs processor B (using MU processor A-side transmit register, MUA_TR*n*).
10	Processor A sends an interrupt to processor B (a request for more data).	Processor B receives the next interrupt from processor A, in which processor A requests more packet data.

37.6.5 Freeing a processor from deadlock

During normal operation, one processor may determine that the other processor is not working or is deadlocked. Using Status (SR), the processor can use the methods in the following table to identify and correct the problem.

Table 285. Freeing a processor from deadlock

Processor mode	Technique	Description
—	Processor issues an interrupt.	The other processor can interrupt the processor by issuing one of the 12 (general purpose, receive, or transmit) interrupts.

37.7 Register definition

MU provides transmit and receive data registers (*x*TR0–*n*, *x*RR0–*n*) for communication between processor A and processor B. It also provides control registers (*x*CR) for operations such as interrupts and resets, and status registers (*x*SR) for checking the status of the other MU-side. Figure 278 shows the schematic for MU registers.

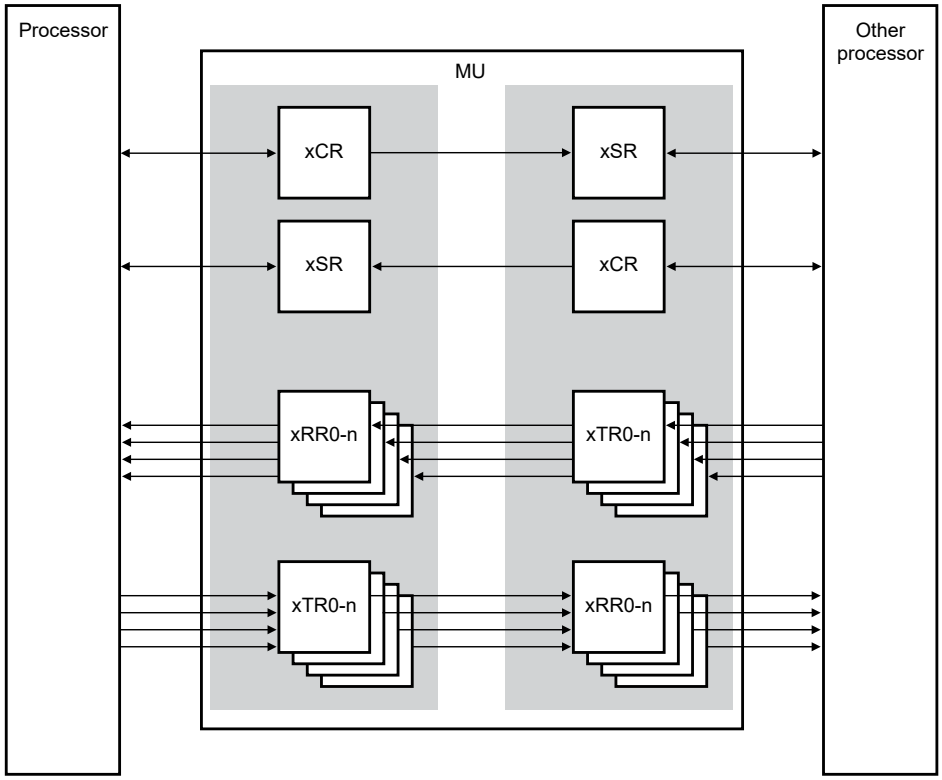


Figure 278. MU registers

37.7.1 MUA register descriptions

This section contains the detailed register descriptions for MUA registers.

NOTE

A module transfer error to processor A or processor B is generated when:

- A read or write access is made to an invalid location.
- A write operation is performed on a read-only register on the processor A side or processor B side of MU.

37.7.1.1 MUA memory map

MU0.MUA base address: 4422_0000h

MU1.MUA base address: 4243_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VER)	32	R	0309_000Fh
4h	Parameter (PAR)	32	R	0304_0404h
8h	Control (CR)	32	RW	0000_0000h
Ch	Status (SR)	32	RW	See section
10h	Core Control 0 (CCR0)	32	R	0000_0014h
14h	Core Interrupt Enable 0 (CIER0)	32	R	0000_0000h
100h	Flag Control (FCR)	32	RW	0000_0000h
104h	Flag Status (FSR)	32	R	0000_0000h
110h	General-Purpose Interrupt Enable (GIER)	32	RW	0000_0000h
114h	General-Purpose Control (GCR)	32	RW	0000_0000h
118h	General-purpose Status (GSR)	32	RW	0000_0000h
120h	Transmit Control (TCR)	32	RW	0000_0000h
124h	Transmit Status (TSR)	32	R	0000_000Fh
128h	Receive Control (RCR)	32	RW	0000_0000h
12Ch	Receive Status (RSR)	32	R	0000_0000h
200h - 20Ch	Transmit (TR0 - TR3)	32	W	0000_0000h
280h - 28Ch	Receive (RR0 - RR3)	32	R	0000_0000h

37.7.1.2 Version ID (VER)

Offset

Register	Offset
VER	0h

Function

Determines the version ID and feature set number of MUA.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Fields

Field	Function
31-24 MAJOR	Major Version Number
23-16 MINOR	Minor Version Number
15-0 FEATURE	Feature Set Number Indicates the feature set number. MU implements: • Standard features • Expanded number of TR*n*/RR*n* registers

37.7.1.3 Parameter (PAR)

Offset

Register	Offset
PAR	4h

Function

Defines the number of flags, transmit registers, receive registers, and general-purpose interrupt requests available for MU.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FLAG_WIDTH								GIR_NUM							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RR_NUM								TR_NUM							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-24 FLAG_WIDTH	Flag Width Specifies the number of flag bits (3) in the Flag Control (FCR) and Flag Status (FSR) registers.
23-16 GIR_NUM	General-Purpose Interrupt Request Number Specifies the number of general-purpose interrupt requests available (4).
15-8 RR_NUM	Receive Register Number Specifies the number of receive registers (4).
7-0 TR_NUM	Transmit Register Number Specifies the number of transmit registers (4).

37.7.1.4 Control (CR)

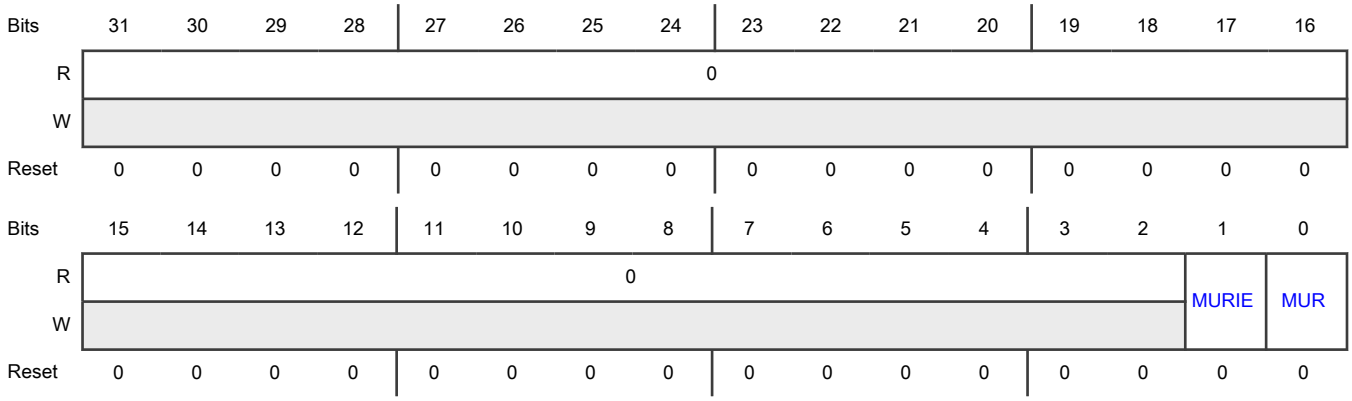
Offset

Register	Offset
CR	8h

Function

Controls MU reset and reset interrupt enable.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 MURIE	MUA Reset Interrupt Enable Enables the processor A-side MU reset interrupt request due to MU reset issued by MUB. If the value of this field is 1, an MU reset interrupt request is issued to processor A when MUA_SR[MURIP] = 1. If the value of this field is 0, MU ignores the value of MURIP and issues no MU reset interrupt request. Only a system reset can reset this field. CR[MUR] cannot reset this field. 0b - Disable 1b - Enable
0 MUR	MU Reset Resets MU. Writing 1 to this field resets the MUA and MUB sides. All internal states are cleared. It forces all control and status registers to return to their default values (except HR, HRM in MUA/B_CCR0 registers; MURIE in MUA/B_CR registers; MURIP and MURS in MUA/B_SR registers). Before writing 1 to this field, interrupt processor B because writing 1 to this field may affect the ongoing processor B program. After writing 1 to this field, monitor the value of MUA_SR[MURS] to know when the reset sequence on the processor B-side has ended. This field always reads 0, and it becomes 0 during the MU reset sequence. 0b - Idle 1b - Reset

37.7.1.5 Status (SR)

Offset

Register	Offset
SR	Ch

Function

Shows the status of MU resets and the status of pending events and requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	RFP	TEP	GIRP	FUP	EP	MURIP	MURS
W															W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	u

Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 RFP	MUA Receive Full Pending Indicates whether a receive full message is pending. This field becomes 1 when MUB writes to a TR*n* register to send data to MUA. After this field becomes 1, MU checks RSR[RF*n*] to determine whether the data in the Receive register is ready for MUA to read it. This field becomes 0 when all MUA RR*n* registers are read, or when MU is reset. 0b - Not pending; MUB is not writing to a Transmit register 1b - Pending; MUB is writing to a Transmit register
5 TEP	MUA Transmit Empty Pending Indicates whether a transmit empty message is pending.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 when any TCR[TIE*n*] field is 1 and MUB reads the corresponding Receive (RR*n*) register. After this field becomes 1, MU checks TSR[TE*n*] to determine whether the data in the Transmit (TR*n*) register is ready for MUA to write to it. This field becomes 0 when write operations to all MUA Transmit (TR*n*) registers where TCR[TIE*n*] = 1 (transfer interrupt enabled) are completed, or when MU is reset. 0b - Not pending; MUB is reading no Receive (RRn) register 1b - Pending; MUB is reading a Receive (RRn) register
4 GIRP	MUA General-Purpose Interrupt Pending Indicates that MUB has sent a general-purpose interrupt request. This field becomes 1 when the MUB side sends a general-purpose interrupt request to the MUA side. GSR[GIP*n*] identifies which general-purpose interrupt request is received. This field becomes 0 when all MUA_GSR[GIP*n*] fields are cleared, or when MU is reset. 0b - No request sent 1b - Request sent
3 FUP	MUA Flag Update Pending Indicates whether a flag update request is pending. MU generates this request when there is a change to the F*n*[2:0] bits of MUA_FCR. This field becomes 1 when the MUA side sends a flag update request to the MUB side. This field becomes 0 when MU acknowledges this flag update request internally (the flag is updated) from the MUB side, or during MU reset. No flag update changes are allowed when this field is 1. When FUP = 1, a write to the F*n*[2:0] bits of MUA_FCR does not generate a flag update event. The F*n*[2:0] bits do not change. If SR[EP] = 1 (event pending), writing to MUA_FCR does not immediately cause this field to become 1. 0b - No pending update flags (initiated by MUA) 1b - Pending update flags (initiated by MUA)
2 EP	MUA Side Event Pending Indicates a pending side event when the MUA side sends an event update request to the MUB side. An event is any hardware message that the Status register on the MUB side reflects. For example, an event occurs when Transmit register 0 is the target of a write operation. During normal operations, the update mechanism for this field operates automatically. MU clears this field automatically when the event update acknowledgment is received, or when MU resets. To ensure that events are posted to MUB, verify that this field is 0. If it is 1, wait and continue to poll this field until it becomes 0. 0b - Not pending 1b - Pending
1	MU Reset Interrupt Pending Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
MURIP	Indicates whether processor B has issued an MU reset. This flag is set after processor B initiates an MU reset by setting MUB_CR[MUR]. If CR[MURIE] = 1, the processor A MU reset interrupt request is issued when processor B writes 1 to MUB_CR[MUR]. Clearing this flag also clears the MU reset interrupt request. Only a system reset can reset this flag. MU reset cannot reset this flag. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Reset not issued 1b - Reset issued When writing 0b - No effect 1b - Clear the flag
0 MURS	MUA and MUB Reset State Indicates the reset state of MUA and MUB. This field becomes 1 during any system reset or MU reset from the MUA or MUB side. This field becomes 0 when the reset sequence on both MUA and MUB sides ends. After issuing any of the aforementioned reset events, verify that this field is 0 before starting any access. 0b - Out of reset 1b - In reset

37.7.1.6 Core Control 0 (CCR0)

Offset

Register	Offset
CCR0	10h

Function

Allows MUA to control the processor on the MUB side.

Also allows control of the processor A hardware reset mask.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0			Reserved	0	Reserved	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

Fields

Field	Function
31-7 —	Reserved
6-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

37.7.1.7 Core Interrupt Enable 0 (CIER0)

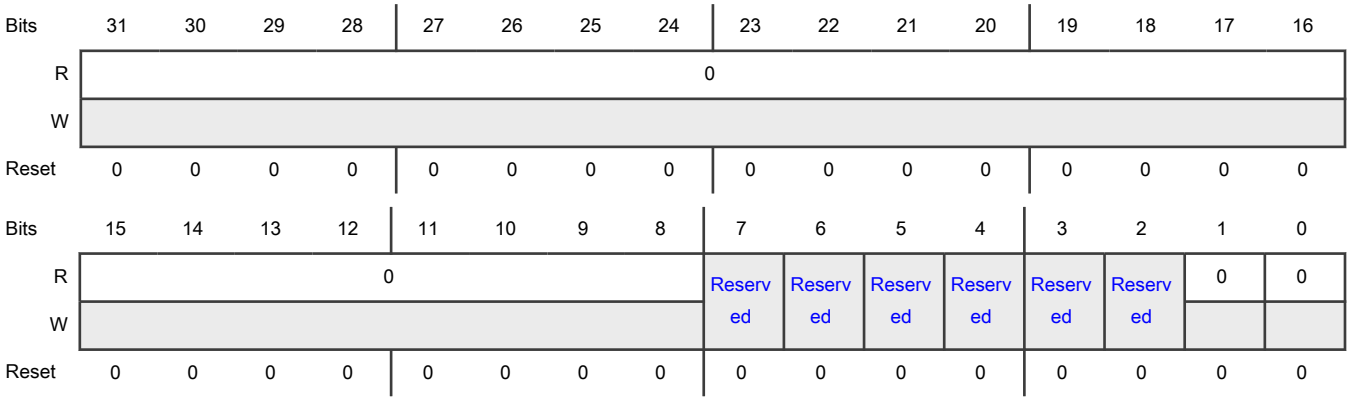
Offset

Register	Offset
CIER0	14h

Function

Controls interrupt enables.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

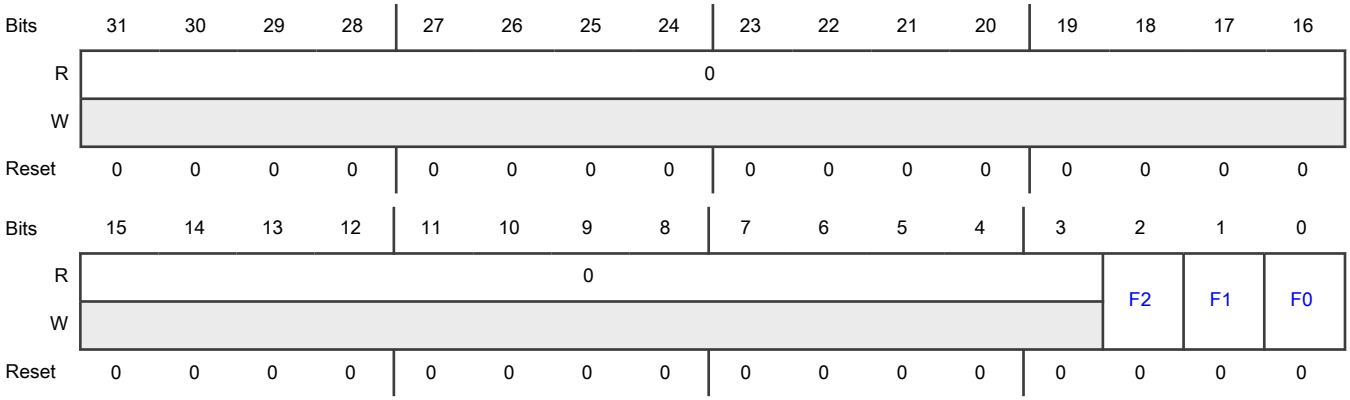
37.7.1.8 Flag Control (FCR)

Offset

Register	Offset
FCR	100h

Function
Configures MUB_FSR[F*n*] flags.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 Fn	MUA to MUB Flag Configures MUB_FSR[F*n*] flags, where *n* = 0 to 2. F*n* configures the corresponding MUB_FSR[F*n*] flag. F*n* becomes 0 when MU resets. 0b - Clear MUB_FSR[Fn] 1b - Set MUB_FSR[Fn]

37.7.1.9 Flag Status (FSR)

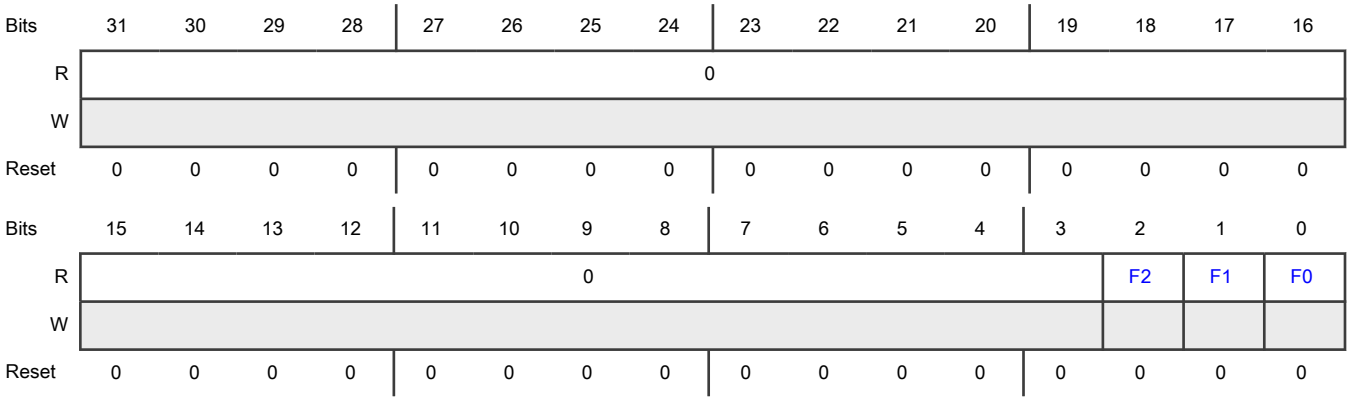
Offset

Register	Offset
FSR	104h

Function

Contains flags configured by the values written to MUB_FCR[F*n*].

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 Fn	MUB to MUA-Side Flag Contains flags configured by the values written to MUB_FCR[F*n*], where *n* = 0 to 2. F*n* is the MUA-side flag configured by the values written to MUB_FCR[F*n*]. When MUB_FCR[F*n*] is written to, the write event updates MUA_FSR[F*n*], after the event update latency. 0b - MUB_FCR[Fn] = 0 1b - MUB_FCR[Fn] = 1

37.7.1.10 General-Purpose Interrupt Enable (GIER)

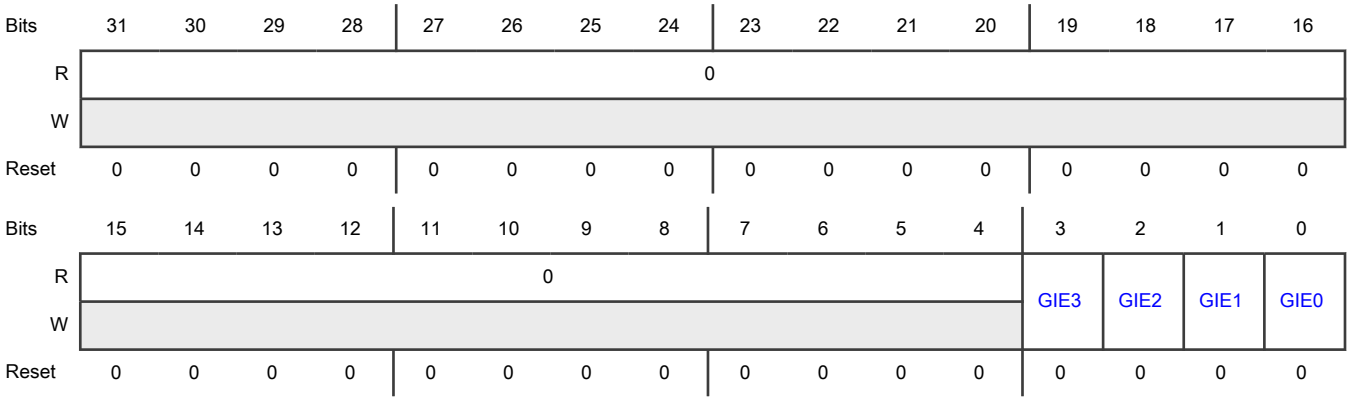
Offset

Register	Offset
GIER	110h

Function

Contains the MUA general-purpose interrupt enable fields.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 GIEn	MUA General-purpose Interrupt Enable Enables general-purpose interrupt. There are 4 general-purpose interrupts (*n* = 0 to 3). When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor A when MUA GSR[GIP*n*] = 1. If GIE*n* = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt. GIE*n* becomes 0 when MU resets. 0b - Disable 1b - Enable

37.7.1.11 General-Purpose Control (GCR)

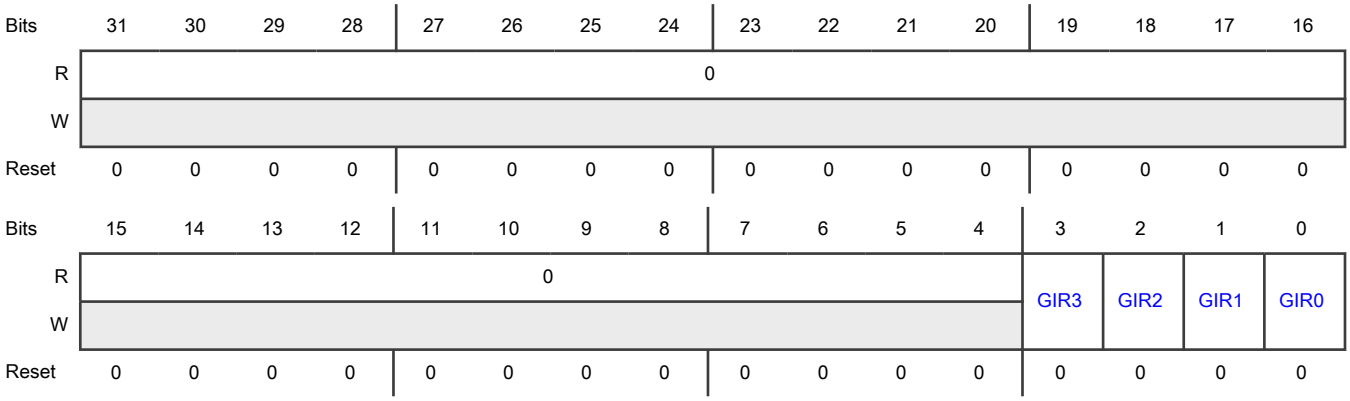
Offset

Register	Offset
GCR	114h

Function

Contains the MUA general-purpose interrupt request fields.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 GIRn	MUA General-Purpose Interrupt Request Specifies whether general-purpose interrupts are requested to MUB. There are 4 general-purpose interrupts (*n* = 0 to 3). Writing 1 to GIR*n* sets MUB_GSR[GIP*n*]. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor B. This field becomes 0 when MUB_GSR[GIP*n*] is cleared. This clearing informs MUA that the interrupt is accepted (cleared by software). To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it. This field becomes 0 when MU resets. 0b - Not requested 1b - Requested

37.7.1.12 General-purpose Status (GSR)

Offset

Register	Offset
GSR	118h

Function

Contains the status of the MUA general-purpose interrupt pending requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												GIP3	GIP2	GIP1	GIP0
W													W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 GIPn	MUA General-Purpose Interrupt Request Pending Indicates whether a general-purpose interrupt request is pending. There are 4 general-purpose interrupts (*n* = 0 to 3). GIP*n* informs MUA that MUB_GCR[GIR*n*] changed from 0 to 1. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor A. GIP*n* is cleared when MU resets. After GIP*n* is cleared, if MUA_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUA side. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag

37.7.1.13 Transmit Control (TCR)

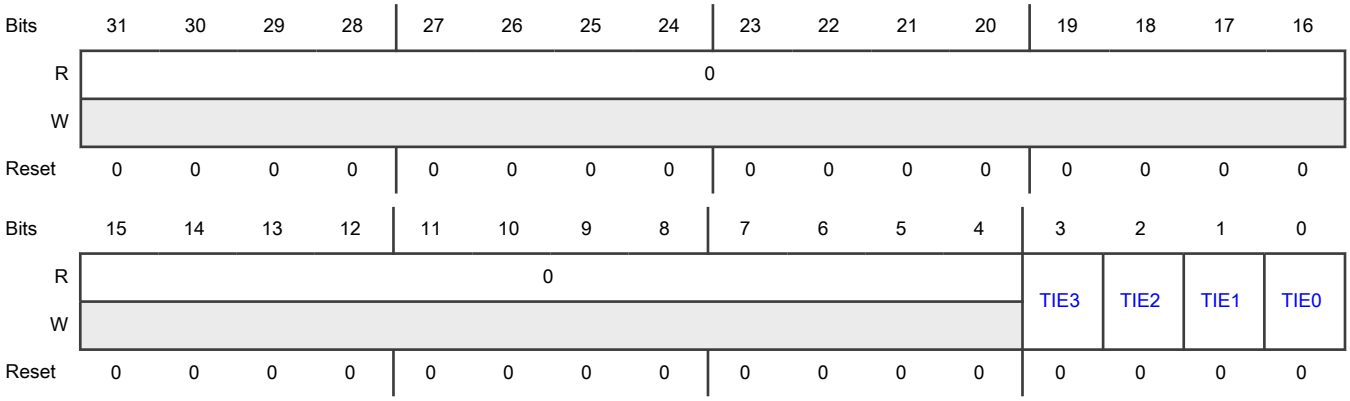
Offset

Register	Offset
TCR	120h

Function

Contains the MUA transmit interrupt enable fields.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TIEn	MUA Transmit Interrupt Enable Enables MUA transmit interrupt *n*, where *n* = 0 to 3. If this field is 1, an MUA transmit interrupt *n* request is issued when MUA_TSR[TE*n*] is set. If this field is 0, MU ignores the value of MUA_TSR[TE*n*], and no MUA transmit interrupt *n* request is issued. 0b - Disable 1b - Enable

37.7.1.14 Transmit Status (TSR)

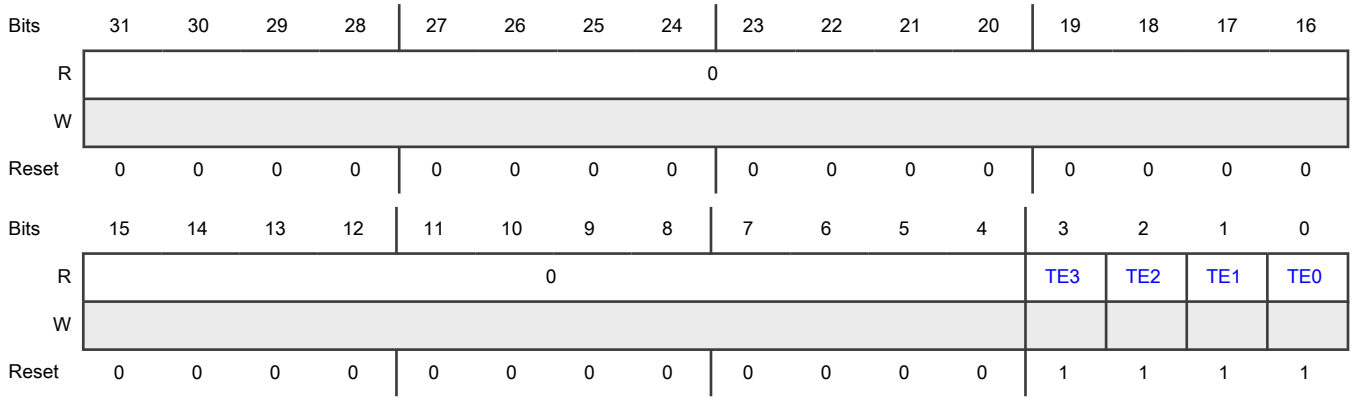
Offset

Register	Offset
TSR	124h

Function

Indicates whether the MUA transmit registers are empty.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TEn	MUA Transmit Empty Indicates whether MUA Transmit (TR*n*) register is empty, where *n* = 0 to 3. This field becomes 1 after the MUB_RR*n* register is read on the MUB side. When TE*n* = 1, it indicates to the MUA side that the MUA_TR*n* register is ready to be written on the MUA side. If MUA_TCR[TIE*n*] = 1, a transmit *n* interrupt is issued on the MUA side. This field becomes 0 after the MUA_TR*n* register is written to on the MUA side. After this field becomes 0, if MUA_TCR[TIE*n*] = 1, the transmit *n* interrupt request is cleared on the MUA side. This field becomes 1 when MU resets. 0b - Not empty 1b - Empty

37.7.1.15 Receive Control (RCR)

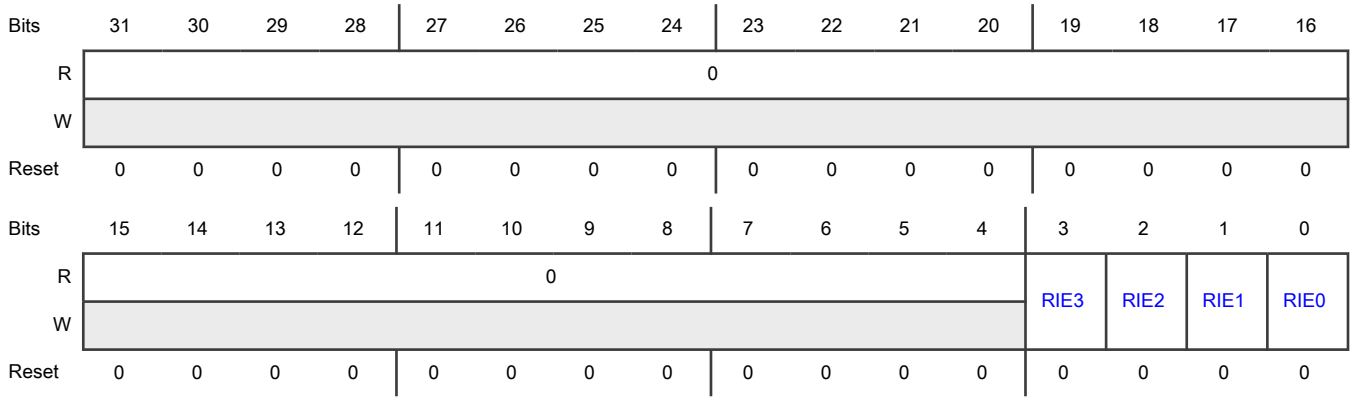
Offset

Register	Offset
RCR	128h

Function

Contains the MUA receive interrupt enables.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RIEn	MUA Receive Interrupt Enable Enables MUA receive interrupt *n*, where *n* = 0 to 3. If this field is 1, an MUA receive interrupt *n* request is issued when MUA_RSR[RF*n*] is set. If this field is 0, MU ignores the value of MUA_RSR[RF*n*], and no MUA receive interrupt request is issued. 0b - Disable 1b - Enable

37.7.1.16 Receive Status (RSR)

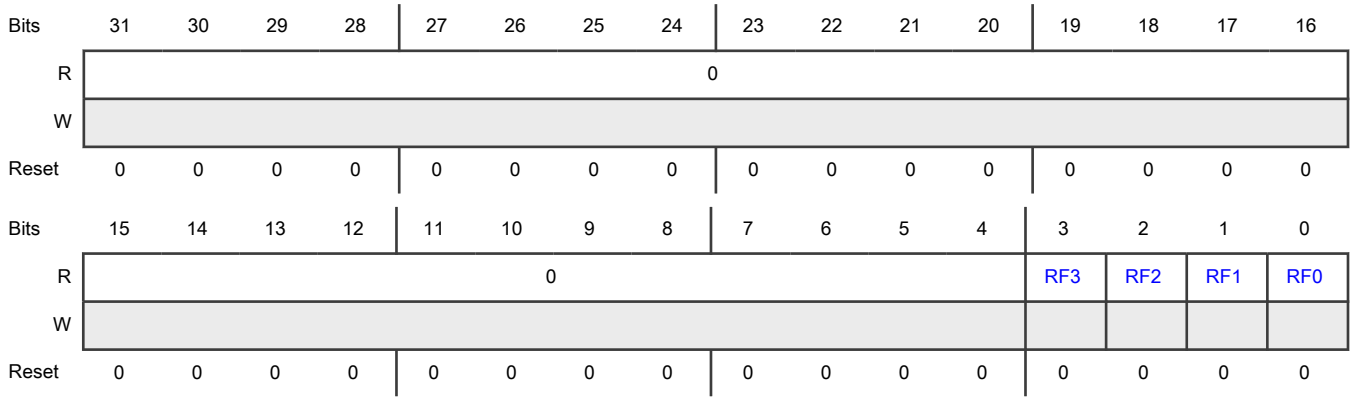
Offset

Register	Offset
RSR	12Ch

Function

Indicates whether the MUA receive registers are full.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RFn	MUA Receive Register Full Indicates whether MUA Receive register (RR*n*) is full, where *n* = 0 to 3. This field becomes 1 when the MUB_TR*n* register is written to on the MUB side. When this field is 1, it indicates to the MUA side that new data in the MUA_RR*n* register is ready for MUA to read it. If MUA_RCR[RIE*n*] = 1, a receive *n* interrupt is issued on the MUA side. This field becomes 0 when the MUA_RR*n* register is read, or when MU is reset. After this field becomes 0, if MUA_RCR[RIE*n*] = 1, the receive *n* interrupt request is cleared on the MUA side. 0b - Not full 1b - Full

37.7.1.17 Transmit (TR0 - TR3)

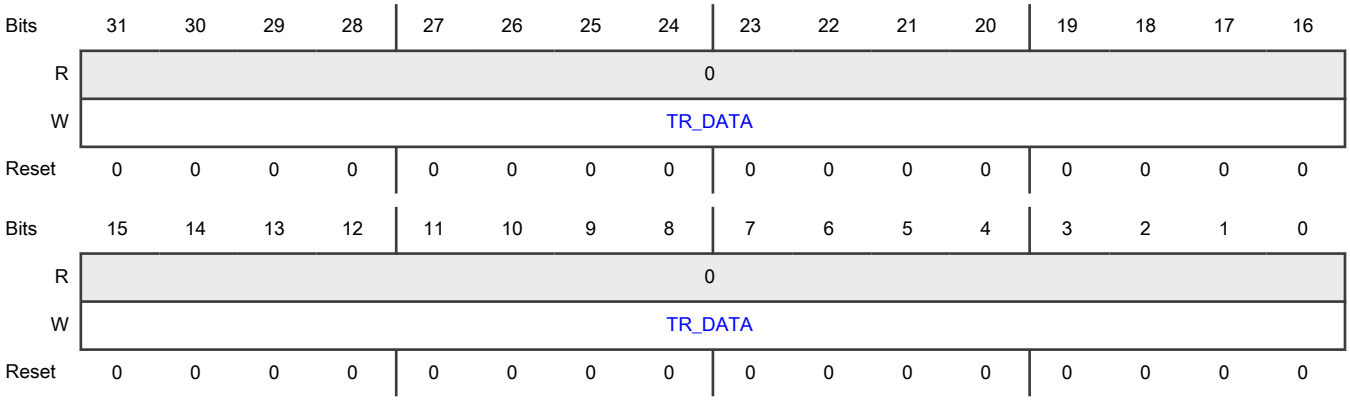
Offset

Register	Offset
TR0	200h
TR1	204h
TR2	208h
TR3	20Ch

Function

Contains MUA transmit data.

Diagram



Fields

Field	Function
31-0 TR_DATA	MUA Transmit Data Contains MUA transmit data. MUB_RR*n* reflects the data written to this register. The TR*n* and RR*n* registers are not double-buffered. Writing to MUA_TR*n* overrides the data readable in the MUA_RR*n* register. A write to the Transmit register clears MUA_TSR[TE*n*] on the transmitter side, and sets MUB_RSR[RF*n*] on the receiver side. You can write to this register only when MUA_TSR[TE*n*] = 1. Reading this register returns all zeroes.

37.7.1.18 Receive (RR0 - RR3)

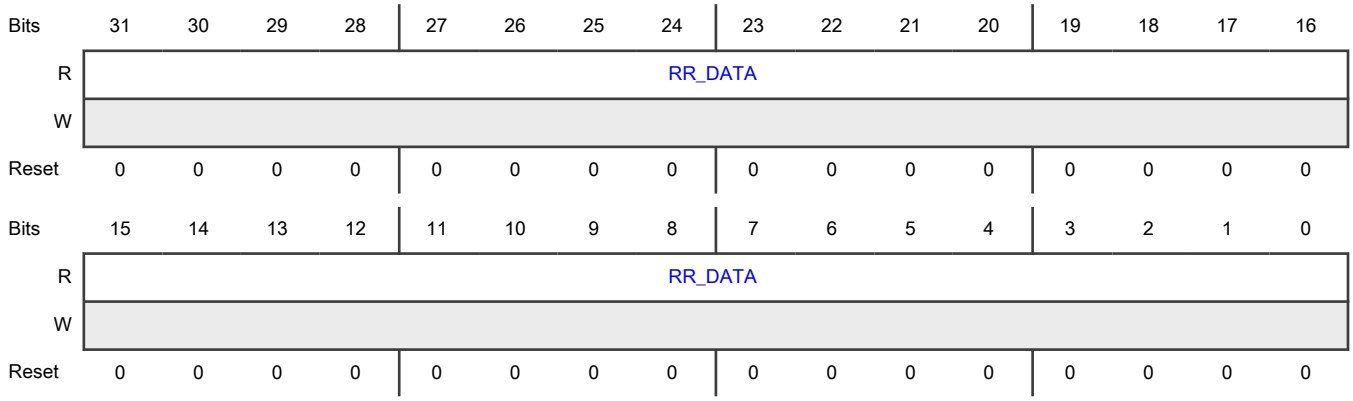
Offset

Register	Offset
RR0	280h
RR1	284h
RR2	288h
RR3	28Ch

Function

Contains MUA receive data.

Diagram



Fields

Field	Function
31-0 RR_DATA	MUA Receive Data Reflects the data written to MUB TR*n*. Reading this register clears MUA_RSR[RF*n*] on the receiver side, and sets MUB_TSR[TE*n*] on the transmitter side. You can read this register only when MUA_RSR[RF*n*] = 1. Reading it before MUA_RSR[RF*n*] becomes 1 may result in reading incorrect data. Poll MUA_RSR[RF*n*] to confirm it is set before reading RR*n*. Writing to this register generates an error response to MUA.

37.7.2 MUB register descriptions

This section contains the detailed register descriptions for MUB registers.

NOTE

A module transfer error to processor A or processor B is generated when:

- A read or write access is made to an invalid location.
- A write operation is performed on a read-only register on the processor A side or processor B side of MU.

37.7.2.1 MUB memory map

MU0.MUB base address: 4423_0000h

MU1.MUB base address: 4244_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VER)	32	R	0309_000Fh
4h	Parameter (PAR)	32	R	0304_0404h
8h	Control (CR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
Ch	Status (SR)	32	RW	See section
10h	Core Control 0 (CCR0)	32	R	0000_0014h
14h	Core Interrupt Enable 0 (CIER0)	32	R	0000_0000h
100h	Flag Control (FCR)	32	RW	0000_0000h
104h	Flag Status (FSR)	32	R	0000_0000h
110h	General-Purpose Interrupt Enable (GIER)	32	RW	0000_0000h
114h	General-Purpose Control (GCR)	32	RW	0000_0000h
118h	General-purpose Status (GSR)	32	RW	0000_0000h
120h	Transmit Control (TCR)	32	RW	0000_0000h
124h	Transmit Status (TSR)	32	R	0000_000Fh
128h	Receive Control (RCR)	32	RW	0000_0000h
12Ch	Receive Status (RSR)	32	R	0000_0000h
200h - 20Ch	Transmit (TR0 - TR3)	32	W	0000_0000h
280h - 28Ch	Receive (RR0 - RR3)	32	R	0000_0000h

37.7.2.2 Version ID (VER)

Offset

Register	Offset
VER	0h

Function

Determines the version ID and feature set number of MUB.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Fields

Field	Function
31-24 MAJOR	Major Version Number
23-16 MINOR	Minor Version Number
15-0 FEATURE	Feature Set Number Indicates the feature set number. MU implements: • Standard features • Expanded number of TR*n*/RR*n* registers

37.7.2.3 Parameter (PAR)

Offset

Register	Offset
PAR	4h

Function

Defines the number of flags, transmit registers, receive registers, and general-purpose interrupt requests available for MU.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FLAG_WIDTH								GIR_NUM							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RR_NUM								TR_NUM							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-24 FLAG_WIDTH	Flag Width Specifies the number of flag bits (3) in the Flag Control (FCR) and Flag Status (FSR) registers.
23-16 GIR_NUM	General-Purpose Interrupt Request Number Specifies the number of general-purpose interrupt requests available (4).
15-8 RR_NUM	Receive Register Number Specifies the number of receive registers (4).
7-0 TR_NUM	Transmit Register Number Specifies the number of transmit registers (4).

37.7.2.4 Control (CR)

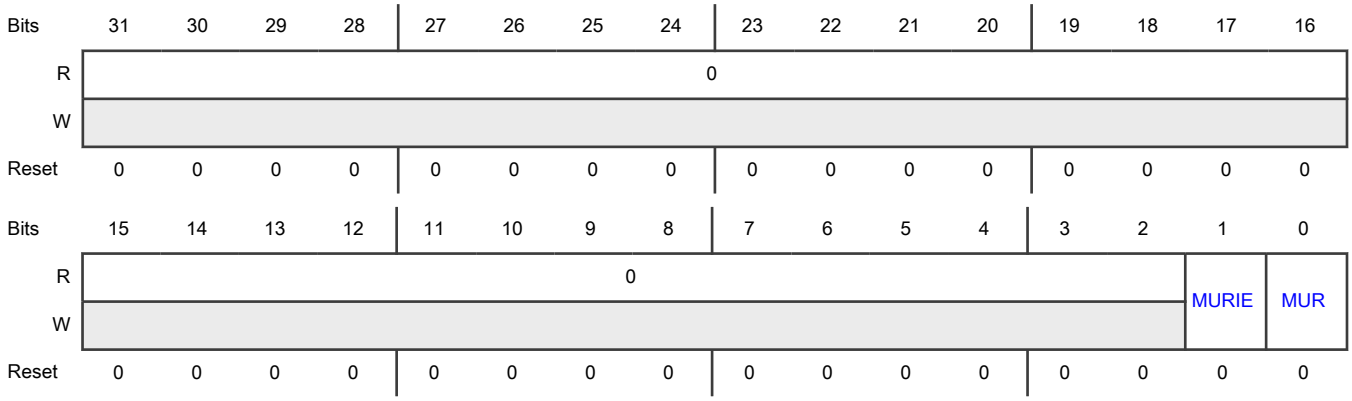
Offset

Register	Offset
CR	8h

Function

Controls MU reset and reset interrupt enable.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 MURIE	MUB Reset Interrupt Enable Enables the processor B-side MU reset interrupt request due to MU reset issued by MUA. If the value of this field is 1, an MU reset interrupt request is issued to processor B when MUB_SR[MURIP] = 1. If the value of this field is 0, MU ignores the value of MURIP and issues no MU reset interrupt request. Only a system reset can reset this field. CR[MUR] cannot reset this field. 0b - Disable 1b - Enable
0 MUR	MU Reset Resets MU. Writing 1 to this field resets the MUB and MUA sides. All internal states are cleared. It forces all control and status registers to return to their default values (except HR, HRM in MUB/A_CCR0 registers; MURIE in MUB/A_CR registers; MURIP and MURS in MUB/A_SR registers). Before writing 1 to this field, interrupt processor A because writing 1 to this field may affect the ongoing processor A program. After writing 1 to this field, monitor the value of MUB_SR[MURS] to know when the reset sequence on the processor A-side has ended. This field always reads 0, and it becomes 0 during the MU reset sequence. 0b - Idle 1b - Reset

37.7.2.5 Status (SR)

Offset

Register	Offset
SR	Ch

Function

Shows the status of MU resets and the status of pending events and requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	RFP	TEP	GIRP	FUP	EP	MURIP	MURS
W															W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	u

Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 RFP	MUB Receive Full Pending Indicates whether a receive full message is pending. This field becomes 1 when MUA writes to a TR*n* register to send data to MUB. After this field becomes 1, MU checks RSR[RF*n*] to determine whether the data in the Receive register is ready for MUB to read it. This field becomes 0 when all MUB RR*n* registers are read, or when MU is reset. 0b - Not pending; MUA is not writing to a Transmit register 1b - Pending; MUA is writing to a Transmit register
5 TEP	MUB Transmit Empty Pending Indicates whether a transmit empty message is pending.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 when any TCR[TIE*n*] field is 1 and MUA reads the corresponding Receive (RR*n*) register. After this field becomes 1, MU checks TSR[TE*n*] to determine whether the data in the Transmit (TR*n*) register is ready for MUB to write to it. This field becomes 0 when write operations to all MUB Transmit (TR*n*) registers where TCR[TIE*n*] = 1 (transfer interrupt enabled) are completed, or when MU is reset. 0b - Not pending; MUA is reading no Receive (RRn) register 1b - Pending; MUA is reading a Receive (RRn) register
4 GIRP	MUB General-Purpose Interrupt Pending Indicates that MUA has sent a general-purpose interrupt request. This field becomes 1 when the MUA side sends a general-purpose interrupt request to the MUB side. GSR[GIP*n*] identifies which general-purpose interrupt request is received. This field becomes 0 when all MUB_GSR[GIP*n*] fields are cleared, or when MU is reset. 0b - No request sent 1b - Request sent
3 FUP	MUB Flag Update Pending Indicates whether a flag update request is pending. MU generates this request when there is a change to the F*n*[2:0] bits of MUB_FCR. This field becomes 1 when the MUB side sends a flag update request to the MUA side. This field becomes 0 when MU acknowledges this flag update request internally (the flag is updated) from the MUA side, or during MU reset. No flag update changes are allowed when this field is 1. When FUP = 1, a write to the F*n*[2:0] bits of MUB_FCR does not generate a flag update event. The F*n*[2:0] bits do not change. If SR[EP] = 1 (event pending), writing to MUB_FCR does not immediately cause this field to become 1. 0b - No pending update flags (initiated by MUB) 1b - Pending update flags (initiated by MUB)
2 EP	MUB Side Event Pending Indicates a pending side event when the MUB side sends an event update request to the MUA side. An event is any hardware message that the Status register on the MUA side reflects. For example, an event occurs when Transmit register 0 is the target of a write operation. During normal operations, the update mechanism for this field operates automatically. MU clears this field automatically when the event update acknowledgment is received, or when MU resets. To ensure that events are posted to MUA, verify that this field is 0. If it is 1, wait and continue to poll this field until it becomes 0. 0b - Not pending 1b - Pending
1	MU Reset Interrupt Pending Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
MURIP	Indicates whether processor A has issued an MU reset. This flag is set after processor A initiates an MU reset by setting MUB_CR[MUR]. If CR[MURIE] = 1, the processor B MU reset interrupt request is issued when processor A writes 1 to MUA_CR[MUR]. Clearing this flag also clears the MU reset interrupt request. Only a system reset can reset this flag. MU reset cannot reset this flag. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Reset not issued 1b - Reset issued When writing 0b - No effect 1b - Clear the flag
0 MURS	MUA and MUB Reset State Indicates the reset state of MUA and MUB. This field becomes 1 during any system reset or MU reset from the MUA or MUB side. This field becomes 0 when the reset sequence on both MUA and MUB sides ends. After issuing any of the aforementioned reset events, verify that this field is 0 before starting any access. 0b - Out of reset 1b - In reset

37.7.2.6 Core Control 0 (CCR0)

Offset

Register	Offset
CCR0	10h

Function

Allows MUB to control the processor on the MUA side.

Also allows control of the processor B hardware reset mask.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0									0		Reserved	0	Reserved	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

Fields

Field	Function
31-7 —	Reserved
6-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

37.7.2.7 Core Interrupt Enable 0 (CIER0)

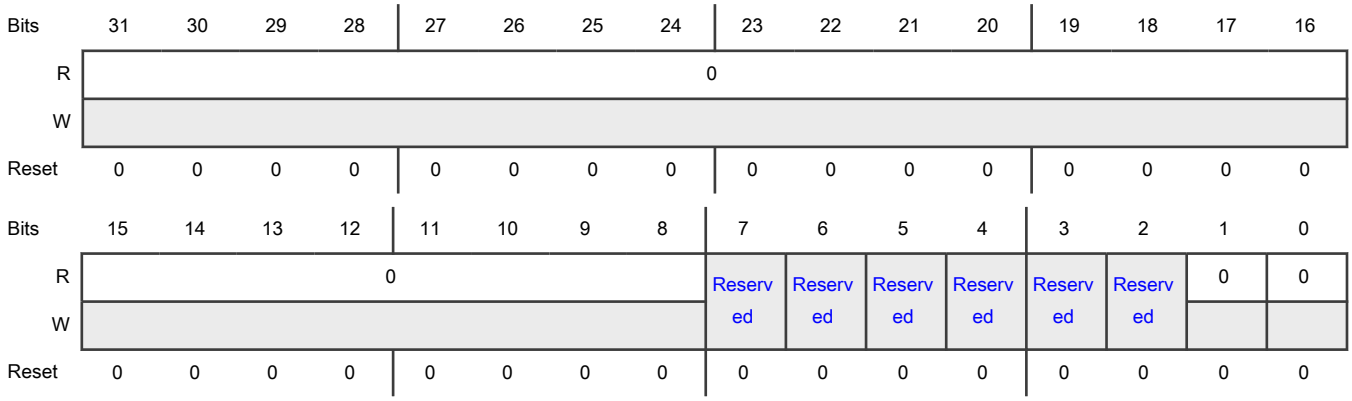
Offset

Register	Offset
CIER0	14h

Function

Controls interrupt enables.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

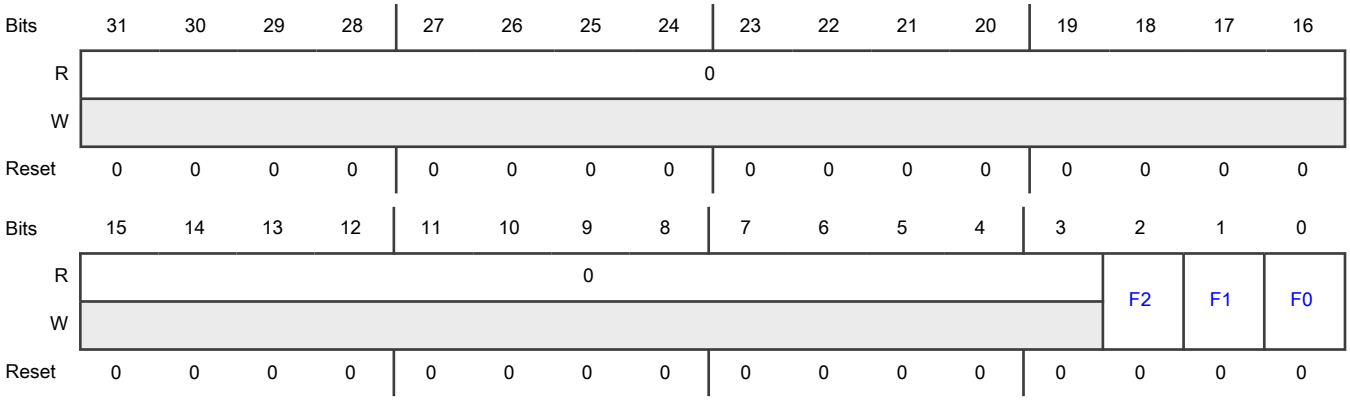
37.7.2.8 Flag Control (FCR)

Offset

Register	Offset
FCR	100h

Function
Configures MUA_FSR[F*n*] flags.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 Fn	MUB to MUA Flag Configures MUA_FSR[F*n*] flags, where *n* = 0 to 2. F*n* configures the corresponding MUA_FSR[F*n*] flag. F*n* becomes 0 when MU resets. 0b - Clear MUA_FSR[Fn] 1b - Set MUA_FSR[Fn]

37.7.2.9 Flag Status (FSR)

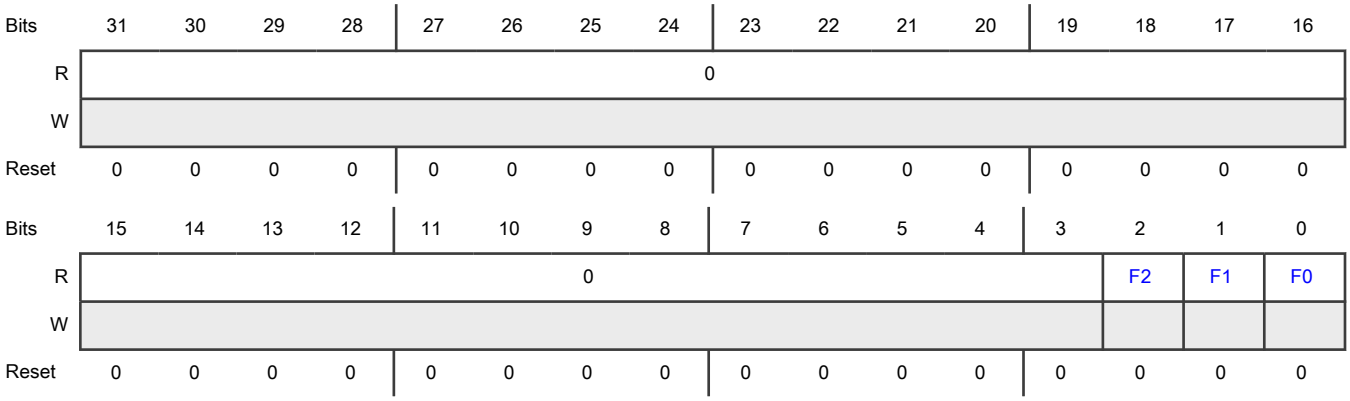
Offset

Register	Offset
FSR	104h

Function

Contains flags configured by the values written to MUA_FCR[F*n*].

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 Fn	MUA to MUB-Side Flag Contains flags configured by the values written to MUA_FCR[F*n*], where *n* = 0 to 2. F*n* is the MUB-side flag configured by the values written to MUA_FCR[F*n*]. When MUA_FCR[F*n*] is written to, the write event updates MUB_FSR[F*n*], after the event update latency. 0b - MUA_FCR[Fn] = 0 1b - MUA_FCR[Fn] = 1

37.7.2.10 General-Purpose Interrupt Enable (GIER)

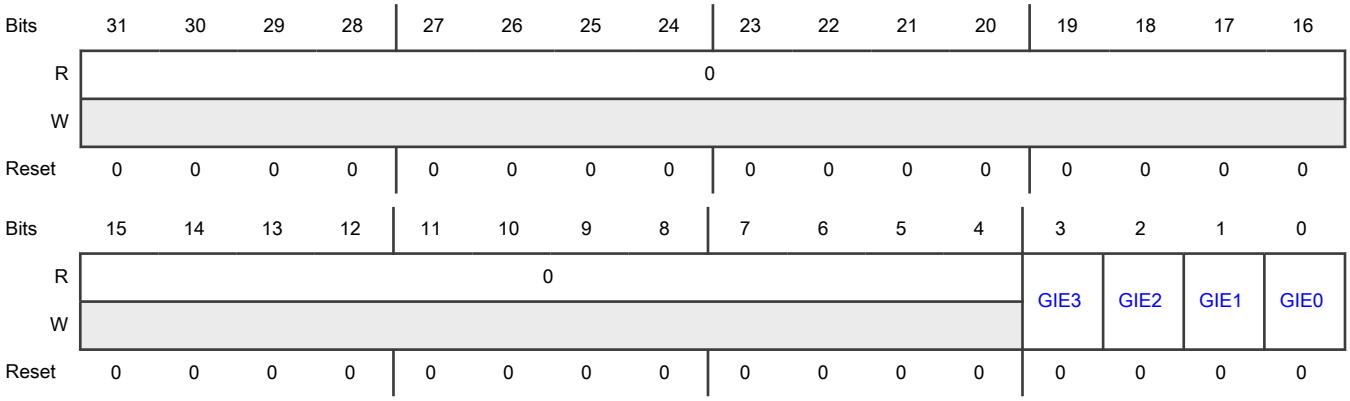
Offset

Register	Offset
GIER	110h

Function

Contains the MUB general-purpose interrupt enable fields.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 GIEn	MUB General-purpose Interrupt Enable Enables general-purpose interrupt. There are 4 general-purpose interrupts (*n* = 0 to 3). When GIE*n* = 1, a general-purpose interrupt *n* request is issued to processor B when MUB GSR[GIP*n*] = 1. If GIE*n* = 0, the general-purpose interrupt request pending does not trigger the general-purpose interrupt. GIE*n* becomes 0 when MU resets. 0b - Disable 1b - Enable

37.7.2.11 General-Purpose Control (GCR)

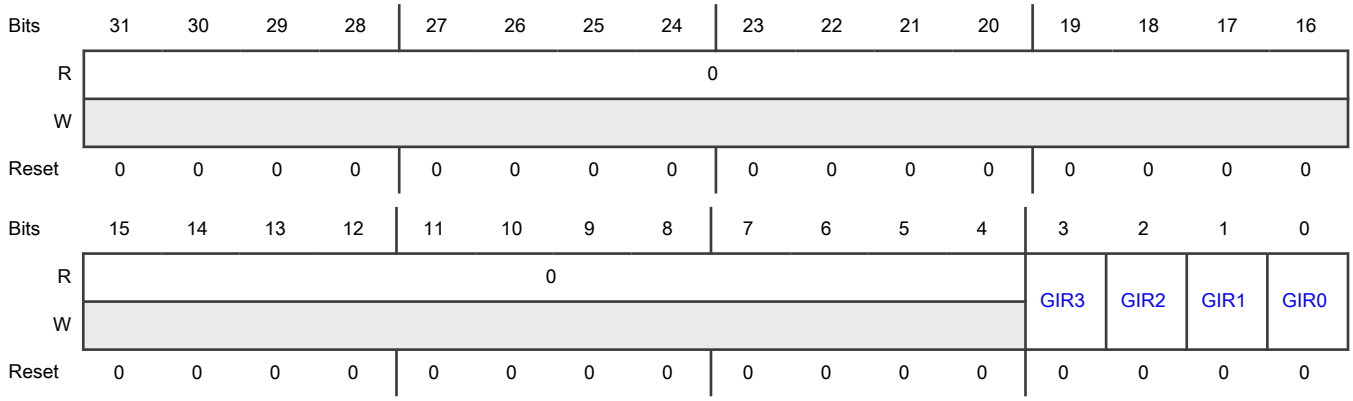
Offset

Register	Offset
GCR	114h

Function

Contains the MUB general-purpose interrupt request fields.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 GIRn	MUB General-Purpose Interrupt Request Specifies whether general-purpose interrupts are requested to MUA. There are 4 general-purpose interrupts (*n* = 0 to 3). Writing 1 to GIR*n* sets MUA_GSR[GIP*n*]. If MUA_GIER[GIE*n*] = 1, a general-purpose interrupt request is triggered on processor A. This field becomes 0 when MUA_GSR[GIP*n*] is cleared. This clearing informs MUB that the interrupt is accepted (cleared by software). To ensure proper operations, verify that GIR*n* is 0 (no pending interrupt) before writing 1 to it. This field becomes 0 when MU resets. 0b - Not requested 1b - Requested

37.7.2.12 General-purpose Status (GSR)

Offset

Register	Offset
GSR	118h

Function

Contains the status of the MUB general-purpose interrupt pending requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												GIP3	GIP2	GIP1	GIP0
W													W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 GIPn	MUB General-Purpose Interrupt Request Pending Indicates whether a general-purpose interrupt request is pending. There are 4 general-purpose interrupts (*n* = 0 to 3). GIP*n* informs MUB that MUA_GCR[GIR*n*] changed from 0 to 1. If MUB_GIER[GIE*n*] = 1, a general-purpose interrupt request is issued to processor B. GIP*n* is cleared when MU resets. After GIP*n* is cleared, if MUB_GIER[GIE*n*] = 1, the general-purpose interrupt request is cleared on the MUB side. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not pending 1b - Pending When writing 0b - No effect 1b - Clear the flag

37.7.2.13 Transmit Control (TCR)

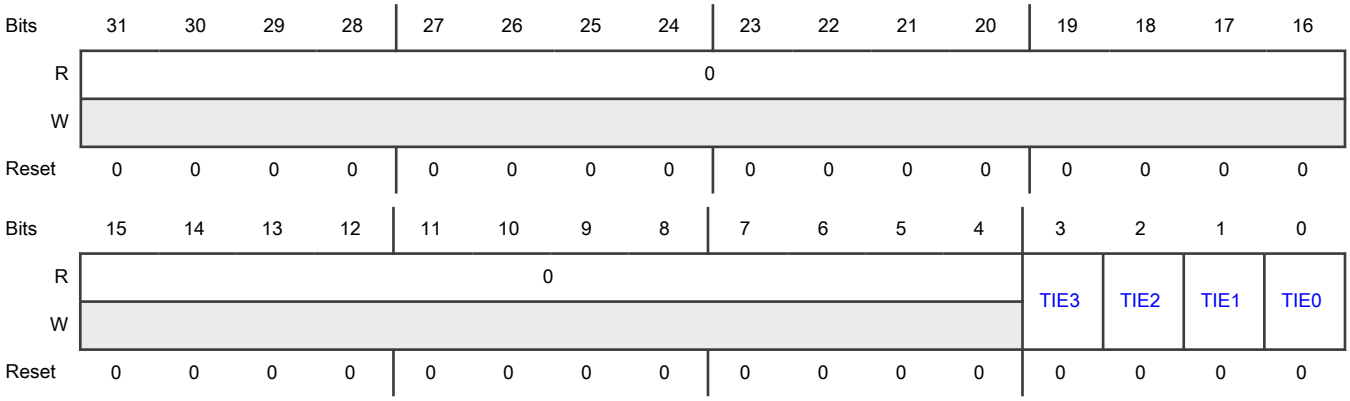
Offset

Register	Offset
TCR	120h

Function

Contains the MUB transmit interrupt enable fields.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TIEn	MUB Transmit Interrupt Enable Enables MUB transmit interrupt *n*, where *n* = 0 to 3. If this field is 1, an MUB transmit interrupt *n* request is issued when MUB_TSR[TE*n*] is set. If this field is 0, MU ignores the value of MUB_TSR[TE*n*], and no MUB transmit interrupt *n* request is issued. 0b - Disable 1b - Enable

37.7.2.14 Transmit Status (TSR)

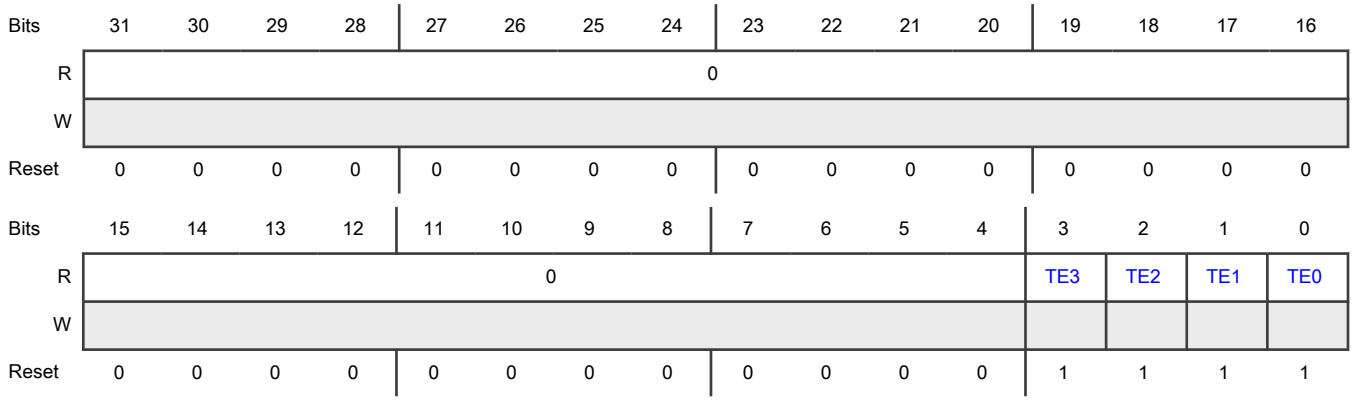
Offset

Register	Offset
TSR	124h

Function

Indicates whether the MUB transmit registers are empty.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TEn	MUB Transmit Empty Indicates whether MUB Transmit (TR*n*) register is empty, where *n* = 0 to 3. This field becomes 1 after the MUA_RR*n* register is read on the MUA side. When TE*n* = 1, it indicates to the MUB side that the MUB_TR*n* register is ready to be written on the MUB side. If MUB_TCR[TIE*n*] = 1, a transmit *n* interrupt is issued on the MUB side. This field becomes 0 after the MUB_TR*n* register is written to on the MUB side. After this field becomes 0, if MUB_TCR[TIE*n*] = 1, the transmit *n* interrupt request is cleared on the MUB side. This field becomes 1 when MU resets. 0b - Not empty 1b - Empty

37.7.2.15 Receive Control (RCR)

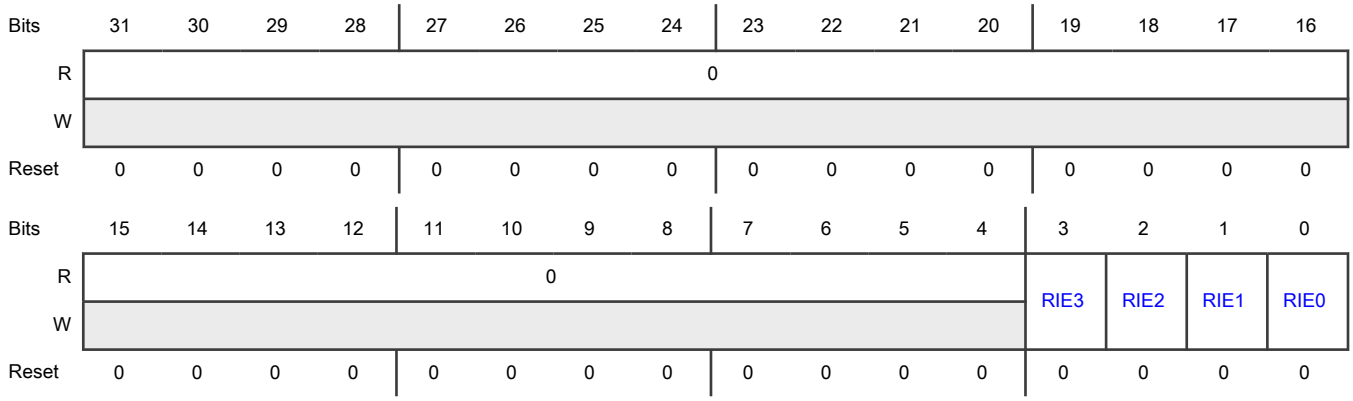
Offset

Register	Offset
RCR	128h

Function

Contains the MUB receive interrupt enables.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RIEn	MUB Receive Interrupt Enable Enables MUB receive interrupt *n*, where *n* = 0 to 3. If this field is 1, an MUB receive interrupt *n* request is issued when MUB_RSR[RF*n*] is set. If this field is 0, MU ignores the value of MUB_RSR[RF*n*], and no MUB receive interrupt request is issued. 0b - Disable 1b - Enable

37.7.2.16 Receive Status (RSR)

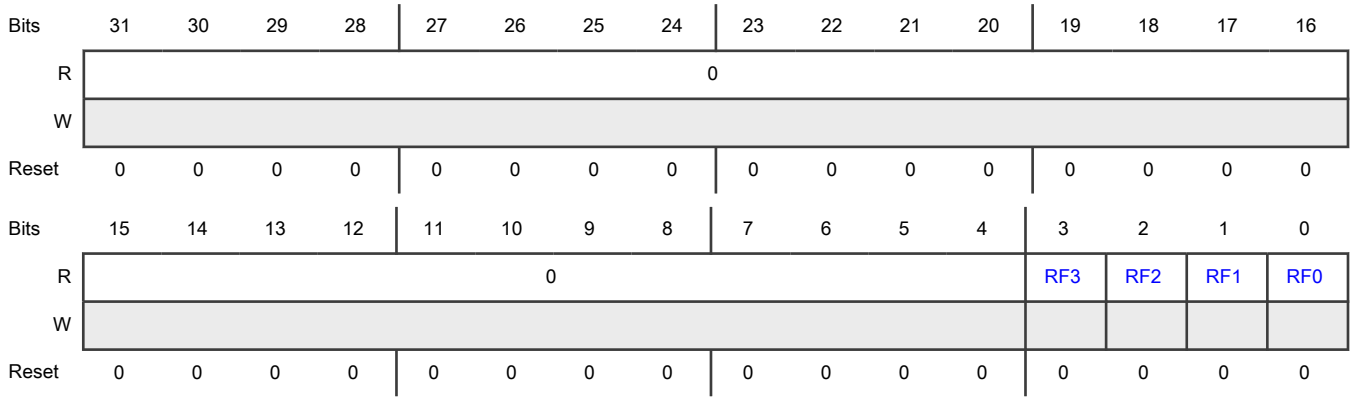
Offset

Register	Offset
RSR	12Ch

Function

Indicates whether the MUB receive registers are full.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RFn	MUB Receive Register Full Indicates whether MUB Receive register (RR*n*) is full, where *n* = 0 to 3. This field becomes 1 when the MUA_TR*n* register is written to on the MUA side. When this field is 1, it indicates to the MUB side that new data in the MUB_RR*n* register is ready for MUB to read it. If MUB_RCR[RIE*n*] = 1, a receive *n* interrupt is issued on the MUB side. This field becomes 0 when the MUB_RR*n* register is read, or when MU is reset. After this field becomes 0, if MUB_RCR[RIE*n*] = 1, the receive *n* interrupt request is cleared on the MUB side. 0b - Not full 1b - Full

37.7.2.17 Transmit (TR0 - TR3)

Offset

Register	Offset
TR0	200h
TR1	204h
TR2	208h
TR3	20Ch

Function

Contains MUB transmit data.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	TR_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	TR_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TR_DATA	MUB Transmit Data Contains MUB transmit data. MUA_RR*n* reflects the data written to this register. The TR*n* and RR*n* registers are not double-buffered. Writing to MUB_TR*n* overrides the data readable in the MUA_RR*n* register. A write to the Transmit register clears MUB_TSR[TE*n*] on the transmitter side, and sets MUA_RSR[RF*n*] on the receiver side. You can write to this register only when MUB_TSR[TE*n*] = 1. Reading this register returns all zeroes.

37.7.2.18 Receive (RR0 - RR3)

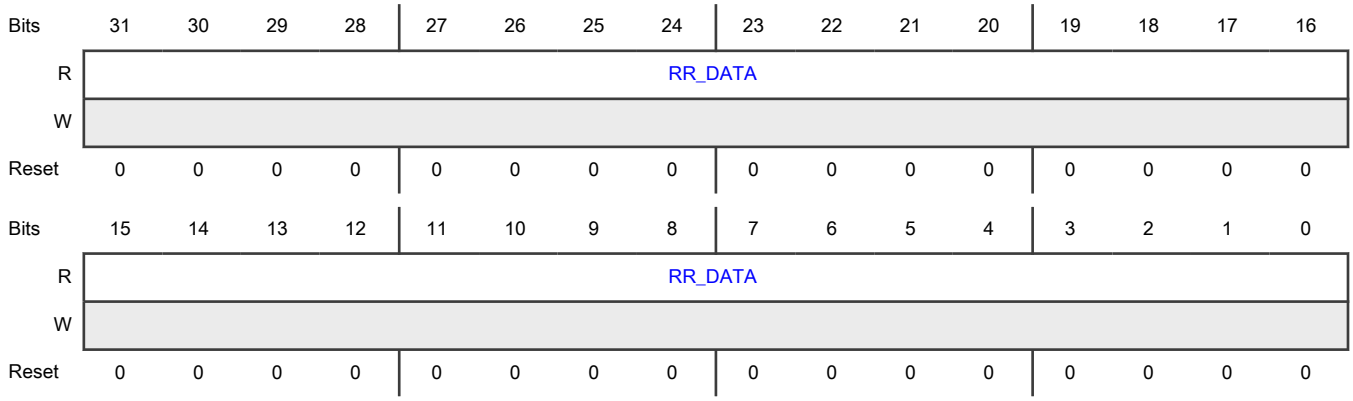
Offset

Register	Offset
RR0	280h
RR1	284h
RR2	288h
RR3	28Ch

Function

Contains MUB receive data.

Diagram



Fields

Field	Function
31-0 RR_DATA	MUB Receive Data Reflects the data written to MUA TR*n*. Reading this register clears MUB_RSR[RF*n*] on the receiver side, and sets MUA_TSR[TE*n*] on the transmitter side. You can read this register only when MUB_RSR[RF*n*] = 1. Reading it before MUB_RSR[RF*n*] becomes 1 may result in reading incorrect data. Poll MUB_RSR[RF*n*] to confirm it is set before reading RR*n*. Writing to this register generates an error response to MUB.

Chapter 38
Edgelock Messaging Unit (MU)

38.1 Overview

The Messaging Unit module enables two processing elements within the SoC to communicate and coordinate by passing messages (e.g., data, status and control) through its interfaces. The Messaging Unit (ELEMU) is specifically targeted for use in ELE. The ELEMU also provides the ability for one processing element to signal the other processing element using interrupts based on data transmission status.

This module's design is based on synchronous clock domain crossings, that is, each side of the ELEMU operates in a phase-aligned clock domain. The two clock domains may be operating at integer multipliers or dividers, but they are phase aligned so there is no need for logic to handle asynchronous clock domain crossings. The ELEMU accomplishes synchronization using two sets of matching registers (Processor A-facing, Processor B-facing).

In the ELE application, the ELEMU "A-side" port corresponds to the SoC host processor while the ELEMU "B-side" port is the security subsystem.

Throughout this chapter, "ELEMUA" is used to denote hardware resources associated with the "A-side" port, and "ELEMUB" denotes those associated with the "B-side" port. This manual only documents the MU "A-side" programming model as it is visible and accessible to the host processor, while the MU "B-side" programming model is restricted to the internal operation of the EdgeLock enclave and not visible nor accessible to the host processor.

38.1.1 Block Diagram

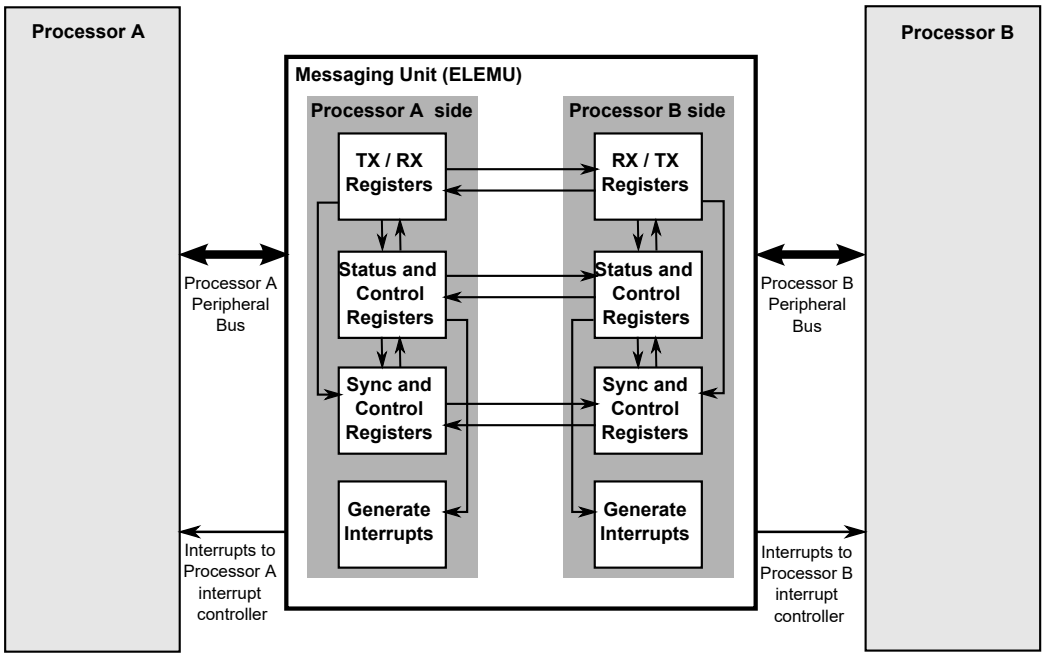


Figure 279. ELEMU Block Diagram

38.1.2 Features

The ELEMU includes the following features:

1. Memory-Mapped Registers

- The ELEMU is connected with separate peripheral buses on Processor A-side and Processor B-side.

2. Synchronous Message Transfers between Processing Elements
   - For sending data or messages between the processing elements, ELEMUA provides 8 transmit registers and 4 receive registers, while ELEMUB provides 4 transmit registers and 8 receive registers.
   - The transfer of data messages between processing elements uses transmit empty and receive full flags provided on both sides of the ELEMU.
   - The update of these transmit and receive flags is accomplished using a fully synchronous mechanism. Upon a register read or write, the corresponding transmit or receive flags are updated at the completion of the register access cycle.

38.2 Functional Description

The Messaging Unit (ELEMU) enables two processing elements (Processor A and Processor B) to communicate with each other, by passing message/data information to each other, and by enabling one side to wake up the other side using data transmission interrupts.

The messaging, control, and status registers of the Processor A and Processor B sides for the ELEMU are mapped to the processing element A and processing element B address spaces using a standard peripheral bridge memory slot. The peripheral data bus is 32 bits wide inside the ELEMU module.

Most of the messaging mechanisms are symmetric. They are duplicated and are available on both the A-side and the B-side. The messaging mechanisms are:

- 8 32-bit ELEMUA transmit registers, which are each reflected in 8 read-only ELEMUB receive registers. These registers can be used to transfer 32-bit word messages or pass information for messages written to the shared memory (number of words, address pointers, and message type code).
- 4 32-bit ELEMUB transmit registers, which are each reflected in 4 read-only ELEMUA receive registers. These registers can be used to transfer 32-bit word messages or pass information for messages written to the shared memory (number of words, address pointers, and message type code).
- A write to a transmit register clears the corresponding “transmitter empty” bit in the Status Register on the transmitter side, and sets the appropriate “receiver full” bit in the Status Register on the receiver side. The setting of the bit at the receiver side can optionally trigger an interrupt at the receiver side (maskable receive interrupt).
- A read of one of the receive registers clears the corresponding “receiver full” bit in the Status Register at the receiver side, and sets the appropriate “transmitter empty” bit in the Status Register on the transmitter side. The setting of the “transmitter empty” bit can optionally trigger an interrupt at the transmitter side (maskable transmit interrupt).

A write to a transmit register signals the receiver side that data is ready for retrieval.

- Writing to the transmit register again without verifying that the data was retrieved is prohibited, because the transmitter side has no way of knowing the exact time that the receiver retrieves the data.
- Before attempting to write the transmit register again, the transmitter side waits for a “Transmitter Empty” interrupt, or polls the “Transmitter Empty” bit in the Transmit Status Register.

A read of a receive register signals the transmitter side that new data can be written to that register. In the same way, the receiver processor should not read a receive register before receiving a “Receiver Full” interrupt or polling the “Receiver Full” bit in the Receive Status Register.

- Reading the receive register again without verifying that the data was written is prohibited, because the receiver side has no way of knowing the exact time that the transmitter writes the data.
- Before attempting to read the receive register again, the receiver side waits for a “Receiver Full” interrupt, or polls the “Receiver Full” bit in the Receive Status Register.

38.3 Register Definition

The ELEMU provides transmit and receive data registers for the communication between Processor A and Processor B. Some control and status registers to the Processor A and Processor B sides are for control operations (such as interrupts), and for status checking of the other ELEMU-side. The following diagram shows the ELEMU registers schematic.

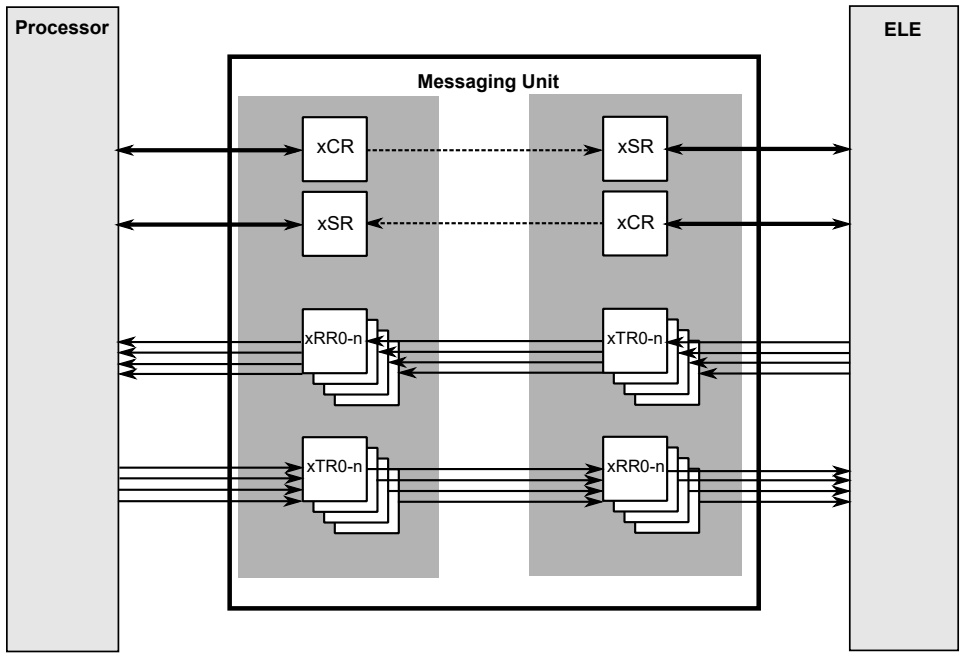


Figure 280. ELEMU Registers

The detailed ELEMU register definition can be found below.

NOTE

Read of a write-only, read zero (WORZ) register returns 0. Writes to a read-only (RO) register are ignored. Byte/halfword accesses are supported.

A read/write access to any illegal location of the ELEMU generates a bus transfer error acknowledge to the Processor A or Processor B. Writes to "RSVD" registers are ignored and reads to "RSVD" registers return 0.

38.3.1 ELEMUA register descriptions

This section contains the detailed register descriptions for the ELEMUA registers.

38.3.1.1 ELEMUA memory map

MU_APPS.ELEMUA base address: 4752_0000h

MU_RT.ELEMUA base address: 4754_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VER)	32	R	0100_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4h	Parameter Register (PAR)	32	R	0000_0408h
8h	Unused Register 0 (UNUSED0)	32	R	0000_0000h
Ch	Status Register (SR)	32	R	0000_0020h
120h	Transmit Control Register (TCR)	32	RW	0000_0000h
124h	Transmit Status Register (TSR)	32	R	0000_0000h
128h	Receive Control Register (RCR)	32	RW	0000_0000h
12Ch	Receive Status Register (RSR)	32	R	0000_0000h
1FCh	Unused Register 1 (UNUSED1)	32	RW	0000_0000h
200h - 21Ch	Transmit Register (TR0 - TR7)	32	W	0000_0000h
280h - 28Ch	Receive Register (RR0 - RR3)	32	R	0000_0000h

38.3.1.2 Version ID Register (VER)

Offset

Register	Offset
VER	0h

Function

The VER register can be used to determine the version ID and feature set number of ELEMUA.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 MAJOR	Major Version Number (0x01)
23-16 MINOR	Minor Version Number (0x00)
15-0 FEATURE	Feature Set Number 0000_0000_0000_0000b - Standard features are implemented.

38.3.1.3 Parameter Register (PAR)

Offset

Register	Offset
PAR	4h

Function
The PAR register reports the number of Transmit (TR) registers and Receive (RR) registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RR_NUM								TR_NUM							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 RR_NUM	Number of Receive (RRn) registers (4)

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 TR_NUM	Number of Transmit (TRn) registers (8)

38.3.1.4 Unused Register 0 (UNUSED0)

Offset

Register	Offset
UNUSED0	8h

Function

DO NOT WRITE

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 —	DO NOT WRITE TO THIS BIT FIELD.

38.3.1.5 Status Register (SR)

Offset

Register	Offset
SR	Ch

Function

The SR register reports the status of the Transmit Empty Pending and Receive Full Pending flags.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0									RFP	TEP	0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved
6 RFP	Receive Full Pending Flag Indicates if any of the receive registers are ready to be read. When the RFP bit is set, then software should check the RSR[RFn] flags to determine which RRn register is ready to be read. 0b - No data is ready to be read. All RSR[RFn] bits are clear. 1b - Data is ready to be read. One or more RSR[RFn] bits are set.
5 TEP	Transmit Empty Pending • The TEP bit reads as "1" when any TSR[TEn] bit is set. • The TEP bit reads as "0" when all TSR[TEn] bits are clear. • When the TEP bit reads as "1", check the TSR[TEn] flags to determine which TRn register is ready to be written.
4-0 —	Reserved

38.3.1.6 Transmit Control Register (TCR)

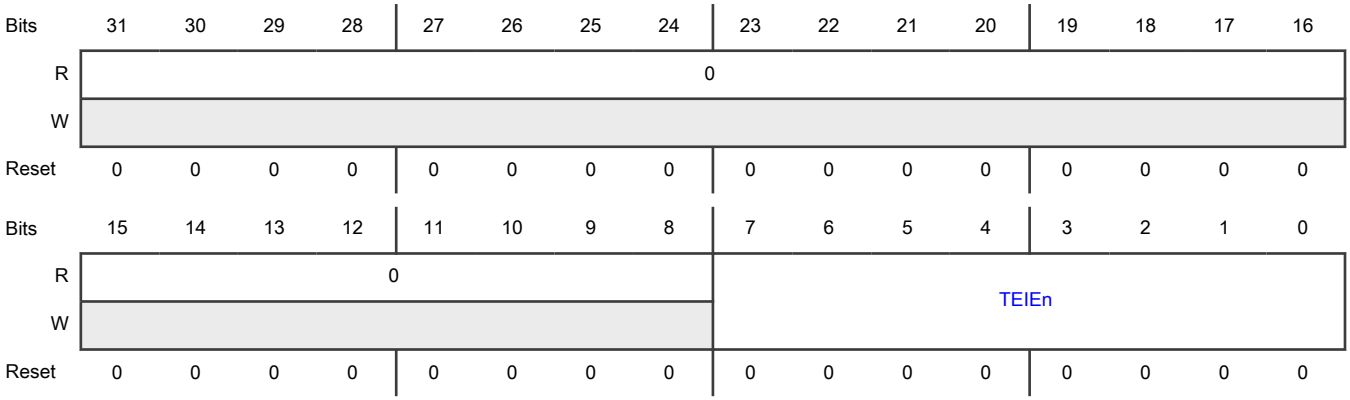
Offset

Register	Offset
TCR	120h

Function

The TCR register is used to configure which Transmit Empty interrupts are generated when the corresponding TSR[TEn] bits are set.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 TEIEn	Transmit Register n Empty Interrupt Enable When set, causes a Transmit Empty interrupt to be generated when the corresponding TSR[TEn] bit is set.

38.3.1.7 Transmit Status Register (TSR)

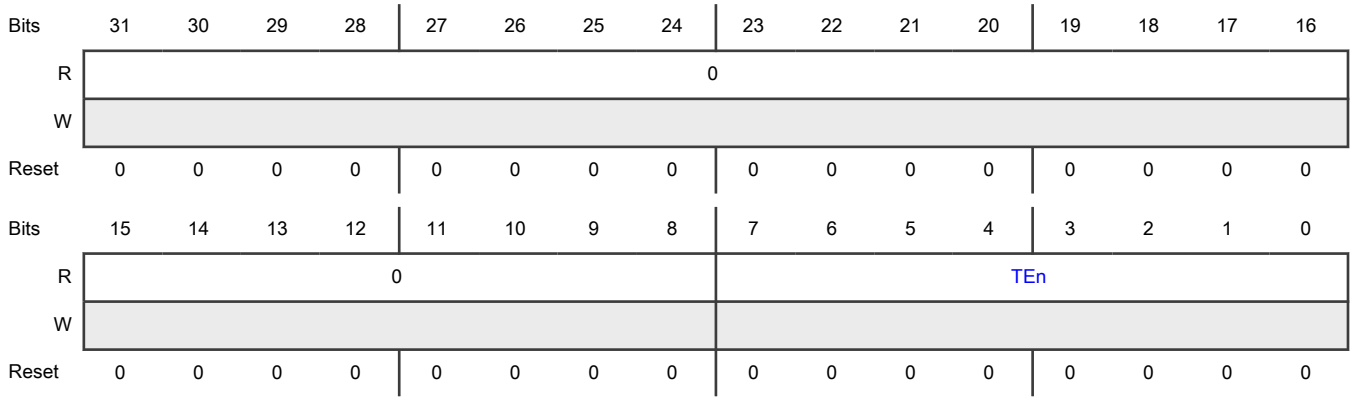
Offset

Register	Offset
TSR	124h

Function

The TSR register shows which TR registers are ready to be written.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 TEn	Transmit Register n Empty • The TEn bit is used to indicate to the ELEMUA processing element that the corresponding ELEMUA TRn register is ready to be written. • The TEn bit is set after the corresponding ELEMUB RRn register is read, indicating the ELEMUA TRn register is empty. • The TEn bit is cleared after the ELEMUA TRn register is written, indicating the ELEMUA TRn register is full. • After the TEn bit is cleared, the Transmit Empty interrupt n (if the TCR[TEIEn] bit is set) is cleared on the ELEMUA side.

38.3.1.8 Receive Control Register (RCR)

Offset

Register	Offset
RCR	128h

Function

This register can be used for status, as the receive interrupts are not connected to the host CPU.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												RFIEn			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 RFIEn	Receive Register n Full Interrupt Enable When set, causes a Receive Full interrupt to be generated when the corresponding RSR[RFn] bit is set.

38.3.1.9 Receive Status Register (RSR)

Offset

Register	Offset
RSR	12Ch

Function

The RSR register shows which RR registers are ready to be read.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												RFn			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 RFn	Receive Register n Full • The RFn bit is used to indicate to the ELEMUA processing element that the corresponding ELEMUA RRn register is ready to be read. • The RFn bit is set after the corresponding ELEMUB TRn register is written, indicating the ELEMUA RRn register is full. • The RFn bit is cleared after the ELEMUA RRn register is read, indicating the ELEMUA RRn register is empty. • After the RFn bit is cleared, the Receive Full interrupt n (if the RCR[RFIEn] bit is set) is cleared on the ELEMUA side.

38.3.1.10 Unused Register 1 (UNUSED1)

Offset

Register	Offset
UNUSED1	1FCh

Function

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA16															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-0 DATA16	Unused 16-bit Register

38.3.1.11 Transmit Register (TR0 - TR7)

Offset

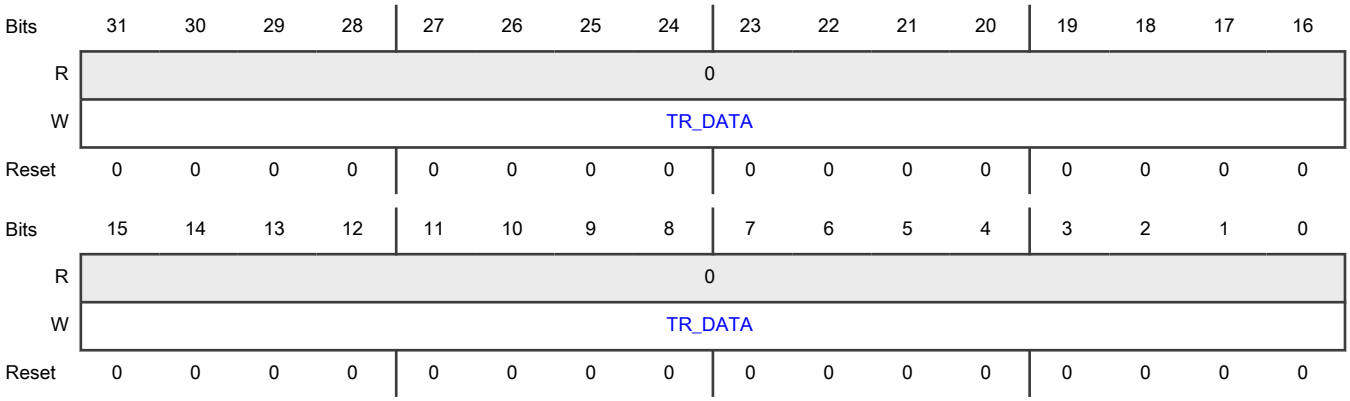
For n = 0 to 7:

Register	Offset
TRn	200h + (n × 4h)

Function

The TR registers contain Transmit Data.

Diagram



Fields

Field	Function
31-0 TR_DATA	Transmit Data • Data written to the TRn register is reflected in the ELEMUB Receive Register n (RRn). The TRn and RRn registers are not double-buffered; a write to the TRn register overrides the data readable at the RRn register. • A write to the transmit register clears the ELEMUA TSR[TEn] bit on the transmitter side, and sets the ELEMUB RSR[RFn] bit on the receiver side. • TRn register should be written only when the ELEMUA TSR[TEn] bit is set to "1". • Reading the TRn register returns all zeros.

38.3.1.12 Receive Register (RR0 - RR3)

Offset

Register	Offset
RR0	280h
RR1	284h
RR2	288h
RR3	28Ch

Function

The RR registers contain Receive Data.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RR_DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RR_DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RR_DATA	Receive Data • Reflects the data written to ELEMUB Transmit Register (TRn). • Reading the RRn register clears the ELEMUA RSR[RFn] bit on the receiver side, and sets the ELEMUB TSR[TEn] bit on the transmitter side. • RRn register should be read only when the ELEMUB RSR[RFn] bit is set to "1".

Chapter 39
Cache Memory Controller (XCACHE)

39.1 Chip-specific XCACHE Information

Table 286. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

NOTE

This cache is for CM33. For further details about cache related information, see AMBA AHB protocol specification.

39.2 Overview

XCACHE is a general-purpose AMBA AHB bus protocol cache. XCACHE provides lower latency access to slower memories.

39.2.1 Block diagram

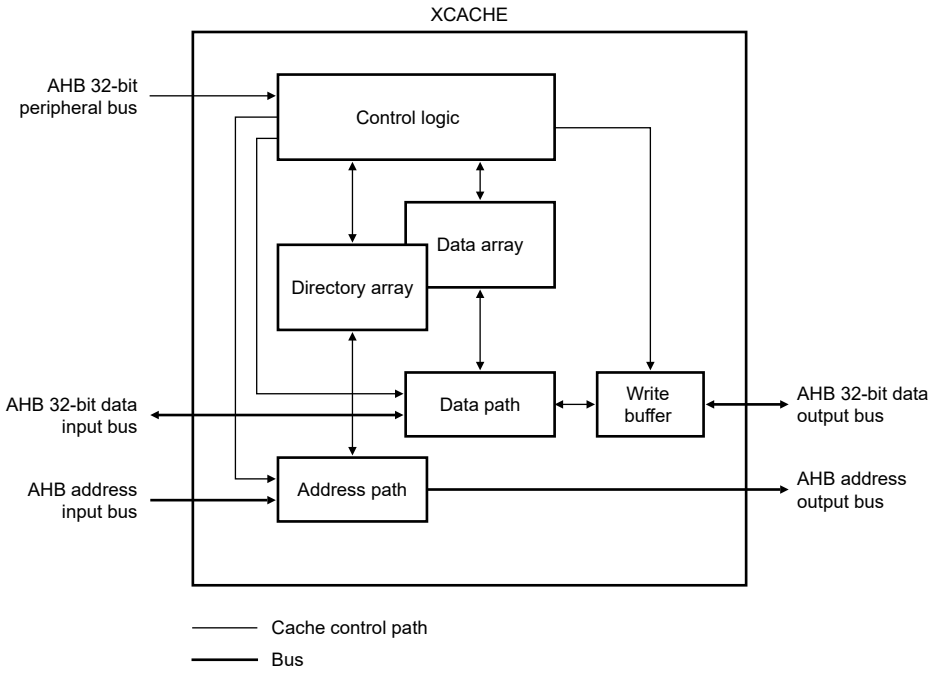


Figure 281. Block diagram

39.2.2 Features

XCACHE is designed to work with AMBA AHB bus systems. It includes:

- An AHB input bus. Accesses have cache mode attributes for cacheable copyback, cacheable write-through, or noncacheable mode operation and write bufferable mode operation on an access-by-access basis.
- 32-bit data path AHB output bus for downstream cache read miss, cache write-through write, and noncacheable accesses.
- Connections to tag and data RAMs to implement the cache functions.
- A peripheral interface for accessing the cache's programming model.

XCACHE includes the following AMBA_AHB buses:

- AHB input bus
- AHB output bus

39.3 Functional description

XCACHE is a block of high-speed memory locations that contains address information (commonly known as a tag) and its associated data. It decreases the average time of a memory access.

XCACHE operates on two principles of locality:

- Spatial locality — Access to one location possibly follows an access from adjacent locations (for example, sequential instruction execution or usage of a data structure).
- Temporal locality — Access to an area of memory possibly repeats within a short period of time (for example, execution of a code loop).

To minimize the quantity of control information stored, the spatial locality property groups several locations together, under the same tag. This logical block is commonly known as a cache line.

You achieve temporal locality by retaining recently accessed lines in the cache.

When data loads into a cache, access times for subsequent loads and stores are reduced, resulting in overall performance benefits. An access to information already in a cache is known as a cache hit, and other accesses are known as cache misses.

Normally, caches are self-managing, with the updates occurring automatically. Whenever the input bus wants to access a cacheable location, the cache is checked. If the access is a cache hit, the access occurs immediately. Otherwise, a location is allocated and the cache line is loaded from memory. Different cache topologies and access policies are possible. However, they must comply with the memory coherency model of the underlying architecture.

Caches introduce a number of potential problems, mainly because of:

- Memory accesses occurring at times other than when the programmer would normally expect them.
- The existence of multiple physical locations where a data item can be held.

39.3.1 Modes of operation

Table 287. Modes of operation

Mode	Description
Write-Through	Access to address spaces with this cache mode are cacheable. • If all cacheable spaces are read-only spaces, then cache contains read-only data and all writes to the cache cause a fault. See the chip-specific cacheable space information. • A read miss on the input bus causes a line read on the output bus of a 16-byte-aligned memory address that contains the desired address. This miss data loads into the cache and marks as valid and not modified. • A read hit to a valid cache location returns data from the cache with no output bus access. • A write miss bypasses the cache and writes to the output bus (no allocation on write-miss policy for Write-Through mode spaces). • A write hit updates the cache hit line with the write data and writes to the output bus. • The caches are input-bus-local and do not support hardware cache coherency. If the input bus accesses write-through regions and an external input bus (such as DMA), that needs to update these regions, you must first perform explicit cache clears to any needed cache memory range to ensure all modified cache lines update their associated memories before they are modified by external inputs, and subsequent cache input bus accesses get the updated memory.
Write-back	Access to address spaces with this cache mode are cacheable. • A read miss on the input bus causes a line read on the output bus of a 16-byte-aligned memory address that contains the desired address. This miss data loads into the cache and marks as valid and not modified. • A read hit to a valid cache location returns data from the cache with no output bus access. • A write miss performs a read-to-write (allocate on write miss policy for write-back mode spaces). You perform a line read on the output bus of a 16-byte-aligned memory address containing the desired write address. This

Table continues on the next page...

Table 287. Modes of operation (continued)

Mode	Description
	miss data is loaded into the cache and marked as valid and modified. Then the write data updates the appropriate cache data locations. • A write hit updates the cache hit line with the write data plus marks the line as modified. • Caches do not support hardware cache coherency. If the cache's input bus has written to write-back regions and another input bus that can access the same target memory without going through the cache, to then needs to see these updates, software must perform explicit cache pushes to any needed cache memory range to ensure all modified cache lines update their associated memories before being read by this input. Likewise, if the cache's input bus has accessed write-back or write-through regions and another input bus that can access the same target memory without going through the cache then needs to update these regions, you must first perform explicit cache clears to any needed cache memory range to ensure all modified cache lines update their associated memories before this input modifies them, and subsequent cache input bus accesses get the updated memory.
Non-cacheable	Access to address spaces with this cache mode are not cacheable. These accesses bypass the cache and access the output bus.

There is an optional ECC (Error correcting code) support of the cache tag and data storage RAMs. The optional ECC is SECDED (Single error correct, double error detect).

39.3.2 XCACHE function

XCACHE receives bus requests on the input bus.

You can access the programming model for the cache via the XCACHE's AMBA APB bus.

XCACHE then processes the cacheable accesses as needed, when bypassing the noncacheable, cache write-through, cache miss, and cache maintenance accesses to its output bus.

The cache on this chip is structured as follows. The cache has a 2-way set-associative cache structure with a total size of 16 KB for the cache. The caches have 32-bit address and data paths and a 16-byte line size. The cache tags and data storage use single-port, synchronous RAMs.

The TAG contains the required upper address bits, a modify bit, a valid bit, and up to 8 bits of extended address attributes. The extended address attribute field has information such as the supervisor/user and/or domain ID of the accesses that loads this tag. If any extended address attribute bits are provided, for subsequent access that hit a valid tag but has different attributes, the cache coherently forces a miss so downstream memory protection logic can check the access.

For this 16 KB cache, each cache TAG function uses two 512 x 21-bit RAM arrays or optionally two up to 512 x 29-bit RAM arrays (if all 8 bits of the extended address attribute are used). The cache DATA function uses two 2048 x 32-bit RAM arrays. The cache TAG entries store 19 bits of upper address as well as a modified and valid bit per cache line, and any optional extended address attribute bits. The cache DATA entries store four bytes of code or data.

The tag and data RAM ECC codes are stored in RAM arrays with their associated tag and data. The 7Bits tag ECC code word per cache way is stored along side the associated tag way. The 7Bit data ECC code word per cache way is stored along side the associated data way.

All normal cache accesses use physical addresses. This leads to the use of the following cache address:

CACHE - 16 KB size = (512 sets) x (16-byte lines) x (2-way set associative)

Table 288. Tag cache address use

Tag cache address use	Cache
Tag hit address range	Address[31:13]
Tag set select address range	Address[12:4] selects 1 of 512 sets
Not used	Address[3:0]

Table 289. Data cache address use

Data cache address use	System cache
Not used	Address[31:13]
Data set select address range	Address[12:4] used to select one of 512 sets
32-bit word select	Address[3:2] used to select one of four 32-bit words within a set
Byte select	Address[1:0] used to select the byte within the 32-bit word

39.3.3 XCACHE control

The cache is disabled at reset. Cache tag and data arrays are not cleared at reset. Therefore, to enable and configure the cache, you must clear the cache commands and initialize the required tag array.

39.3.3.1 XCACHE set commands

The XCACHE set commands may operate on:

- All of way 0
- All of way 1
- All of both ways (complete cache)

Cache set commands are initiated using the upper bits in Cache Control (CCR). Cache set commands perform their operation on the cache independent of CCR[ENCACHE].

You initiate a cache set command by writing 1 to CCR[GO]. This field also acts as a busy bit for set commands. This field remains 1 when the command is active, and XCACHE writes 0 to it after the set command completes.

Supported cache set commands are provided in Table 290. Following are the set commands:

- Invalidate − Unconditionally clears valid and modify bits of a cache entry.
- Push − Pushes a cache entry if it is valid and modified, then clears the modify bit. If the entry is not valid or not modified, leave it as is.
- Clear − Pushes a cache entry if it is valid and modified, then clears the valid and modify bits. If the entry is not valid or not modified, clear the valid bit.

Table 290. Cache set commands

Cache Control (CCR)[27:24]				Command
CCR[PUSHW1]	CCR[INVW1]	CCR[PUSHW0]	CCR[INVW0]	
0	0	0	0	NOP
0	0	0	1	Invalidate all way 0

Table continues on the next page...

Table 290. Cache set commands (continued)

Cache Control (CCR)[27:24]				Command
CCR[PUSHW1]	CCR[INVW1]	CCR[PUSHW0]	CCR[INVW0]	
0	0	1	0	Push all way 0
0	0	1	1	Clear all way 0
0	1	0	0	Invalidate all way 1
0	1	0	1	Invalidate all way 1; invalidate all way 0 (invalidate cache)
0	1	1	0	Invalidate all way 1; push all way 0
0	1	1	1	Invalidate all way 1; clear all way 0
1	0	0	0	Push all way 1
1	0	0	1	Push all way 1; invalidate all way 0
1	0	1	0	Push all way 1; push all way 0 (push cache)
1	0	1	1	Push all way 1; clear all way 0
1	1	0	0	Clear all way 1
1	1	0	1	Clear all way 1; invalidate all way 0
1	1	1	0	Clear all way 1; push all way 0
1	1	1	1	Clear all way 1 and way 0 (clear cache)

After a reset, complete an invalidate cache command before using the cache. It is possible to combine the cache invalidate command with the cache enable. That is, writing 8500_0001h to Cache Control (CCR) invalidates the cache and enables the cache.

39.3.3.2 XCACHE line commands

XCACHE line commands operate on a single line in the cache at a time. You can perform cache line commands using a physical or cache address that consists of a set address and a way select. The line command acts on the specified cache line.

Cache line commands with physical addresses first search both ways of the cache set specified by the following physical address bits. If they hit, the commands perform their action on the hit way: For cache - [12:4]

You specify cache line commands using the upper bits in Cache Line Control (CLCR). Cache line commands perform their operation on the cache independent of CCR[ENCACHE]. Using a cache address, the command can be specified using Cache Line Control (CLCR). Using a physical address, the command must also use Cache Search Address (CSAR) to specify the physical address.

You can initiate a line cache command by writing 1 to CLCR[LGO] or CSAR[LGO]. This field also acts a busy bit for line commands. This field remains 1 when the command is active and XCACHE writes 0 to it after the command completes.

Table 291. Cache line commands

Cache Line Control (CLCR)[27:24]			Command
CLCR[LACC]	CLCR[LADSEL]	CLCR[LCMD]	
0	0	00	Search by cache address and way
0	0	01	Invalidate by cache address and way

Table continues on the next page...

Table 291. Cache line commands (continued)

Cache Line Control (CLCR)[27:24]			Command
CLCR[LACC]	CLCR[LADSEL]	CLCR[LCMD]	
0	0	10	Push by cache address and way
0	0	11	Clear by cache address and way
0	1	00	Search by physical address
0	1	01	Invalidate by physical address
0	1	10	Push by physical address
0	1	11	Clear by physical address
1	0	00	Write by cache address and way
1	0	01	Reserved, NOP
1	0	10	Reserved, NOP
1	0	11	Reserved, NOP
1	1	*xx*	Reserved, NOP

39.3.3.2.1 Executing a series of line commands using cache addresses

You can perform a series of line commands with incremental cache addresses by writing to Cache Line Control (CLCR).

- Place the command in the following fields of Cache Line Control (CLCR):
  - LACC
  - LADSEL
  - LCMD
- Write 1 to CLCR[WSEL] and CLCR[TDSEL] as required.
- Place the cache address in CLCR[CACHEADDR].
- Write 1 to CLCR[LGO].

After one line command completes, follow these steps to initiate the next command:

- Increment the cache address (at bit 2 to step through data or at bit 4 to step through lines).
- Write 1 to CLCR[LGO].

39.3.3.2.2 Executing a series of line commands using physical addresses

You can perform a series of line commands with incremental physical addresses using the following steps:

- Write to Cache Line Control (CLCR).
  - Place the command in the following fields of Cache Line Control (CLCR):
    - LACC
    - LADSEL
    - LCMD
  - Write 1 to CLCR[TDSEL]
- Place the physical address in CSAR[PHYADDR] and write 1 to CSAR[LGO].

After one line command completes, follow these steps to initiate the next command:

- Increment the physical address (at bit 2 to step through data or at bit 4 to step through lines).
- Write 1 to CSAR[LGO].

Cache Line Control (CLCR) and Cache Search Address (CSAR) both contains the field called LGO. Therefore, you can complete the above steps in a single write to Cache Search Address (CSAR).

39.3.3.2.3 Line command results

At completion of a line command, Cache Line Control (CLCR) contains information on the initial state of the line targeted by the command. For line commands with cache addresses, read this information before performing a line command action from the targeted cache line. For line commands with physical addresses, read this information on a hit before performing a line command action from the hit cache line or has initial valid bit cleared if the command misses. In general, if CLCR[LCIVB] becomes 0, the targeted line is invalid at the start of the line command and no line operation is performed.

Table 292. Line command results

Cache Line Control (CLCR)[22:20]			For cache address commands	For physical address commands
CLCR[LCWAY]	CLCR[LCIMB]	CLCR[LCIVB]		
0	0	0	Way 0 line is invalid	No hit
0	0	1	Way 0 valid and not modified	Way 0 valid and not modified
0	1	0	Way 0 line is invalid	No hit
0	1	1	Way 0 valid and modified	Way 0 valid and modified
1	0	0	Way 1 line is invalid	No hit
1	0	1	Way 1 valid and not modified	Way 1 valid and not modified
1	1	0	Way 1 line is invalid	No hit
1	1	1	Way 1 valid and modified	Way 1 valid and modified

On completion of a line command other than a write, Cache Read/Write Value (CCVR) contains information on the initial state of the line tag or data targeted by the command. For line commands, CLCR[TDSEL] selects between tag and data. If the line command used a physical address and missed, the data is don't care. For write commands, Cache Read/Write Value (CCVR) holds the write data.

The cache does not have line lock capability.

39.3.3.3 XCACHE ECC enable

There are two ECC control enable inputs - one to enable ECC generation on writes and another to enable ECC checking and correction on reads. These inputs are controlled in an external program model. These signals are:

- ecc_wr_enable - Enable ECC generation on writes.
- ecc_rd_enable - Enable ECC checking and correction on reads.

At reset, the cache is disabled. After reset, to enable the cache with ECC enabled, perform the following:

1. Assert ecc_wr_enable and negate ecc_rd_enabled.
2. Perform the required cache "invalidated all" command. This initializes the TAG ECC.
3. After the invalidate completes, assert both ecc_wr_enable and ecc_rd_enabled.
4. Enable the cache - now the cache is enabled with ECC protection on.

39.3.3.4 XCACHE ECC function

When XCACHE ECC is enabled, for cache tag writes:

- All cache tag write generate the required tag ECC word.
- The cache tag and the associated tag ECC word are written to the tag RAM and tag ECC word RAM simultaneously.

When cache ECC is enabled, for cache data writes:

- All cache data writes of less than 32 bits are converted to a 32-bit read-modify-write sequence.
- All cache data writes generate the required data ECC word.
- The cache data and associated data ECC word are written to the data RAM and data ECC word RAM simultaneously.

When cache ECC is enabled, for cache tag reads:

- The cache tag and its ECC word are read in parallel.
- ECC is checked, any single bit error is corrected on-the-fly, any single, or multi-bit error is flagged.

When cache ECC is enabled, for cache data reads:

- For every data 32-bit word read, the data 32-bit word and its ECC word are read in parallel.
- ECC is checked, any single-bit error is corrected on-the-fly. Any single-, or multi-bit error is flagged.

The cache has single- and multi-bit ECC error indicator outputs.

Because all single-bit ECC errors are corrected on-the-fly, cache operations continue as normal when only single-bit ECC errors occur. Note it is possible to get multiple single- or multi-bit ECC errors on one cache access.

Cache read accesses that get multi-bit ECC errors are handled as follows:

- Any access that gets a tag multi-bit error terminates with a bus fault.
- Any access that gets a data multi-bit error on its hit data terminates with a bus fault.
- Data multi-bit error on miss or invalidate ways are ignored.

NOTE

Cache allocate pushes read the cache data arrays and may get ECC errors. Because the cache access that caused the allocate push may have completed before the ECC error, bus errors do not occur on multi-bit ECC errors. Only single or multi-bit ECC error indicators provide indications for an ECC error.

NOTE

Cache commands, which initiate via the cache's program model, may also access the cache tags and cache data RAMs. If these accesses get a multi-bit ECC error, the peripheral bus access that initiated the cache command does not terminate with a bus error. Only single or multi-bit ECC error indicators provide indications for an ECC error.

39.3.4 Clocks

XCACHE uses the AHB input bus clock.

39.3.5 Interrupts

XCACHE has no interrupts.

39.4 External signals

XCACHE has no external signals.

39.5 Initialization

To configure and enable the cache, you must perform XCACHE commands to clear and initialize the required tag array fields. See the following sequence:

- To enable the cache after reset with the write buffer also enabled, write 8500_0003h to Cache Control (CCR). After you write CCR[GO], it initiates the cache command that the following fields in Cache Control (CCR) specify:
  - PUSHW1
  - INVW1
  - PUSHW0
  - INVW0

  CCR[27:24] indicates a command of clear all entries in both cache ways, CCR[ENWRBUF] enables the write buffer, and CCR[ENCACHE] enables the cache. See XCACHE control for more information.
- To enable the cache after reset with the write buffer disabled, write 8500_0001h to Cache Control (CCR). After you write CCR[GO], it initiates the cache command that the following fields in Cache Control (CCR) specify:
  - PUSHW1
  - INVW1
  - PUSHW0
  - INVW0

  CCR[27:24] indicates a command of clear all entries in both cache ways, and CCR[ENCACHE] enables the cache. See XCACHE control description for more information.

39.6 Application information

XCACHE provides low-latency access to instructions or data. This decouples processor performance from system memory performance, increasing bus availability for other modules, and improving system performance.

39.7 Memory map and registers

The XACHE programming model provides various registers for configuring and controlling the cache, as well as indirect access paths to all cache tag and data storage.

NOTE

The XCACHE registers are accessible in Supervisor mode only.

39.7.1 XCACHE register descriptions

39.7.1.1 XCACHE memory map

XCACHE_PC base address: 4440_0000h

XCACHE_PS base address: 4440_0800h

Offset	Register	Width (In bits)	Access	Reset value
0h	Cache Control (CCR)	32	RW	0000_0000h
4h	Cache Line Control (CLCR)	32	RW	0000_0000h
8h	Cache Search Address (CSAR)	32	RW	0000_0000h
Ch	Cache Read/Write Value (CCVR)	32	RW	0000_0000h

39.7.1.2 Cache Control (CCR)

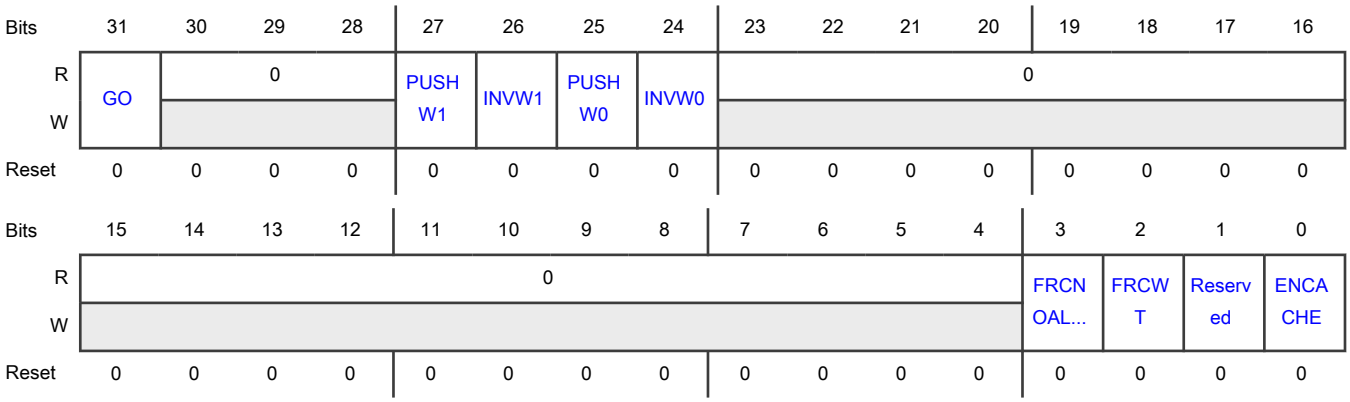
Offset

Register	Offset
CCR	0h

Function

Configures the cache control.

Diagram



Fields

Field	Function
31 GO	Initiate Cache Command Initiates the cache command that CCR[PUSHW1], CCR[INVW1], CCR[PUSHW0], and CCR[INVW0] indicate, if you write 1 to this field. Reading this field indicates whether a command is active. **NOTE** This field remains 1 until the command completes. Writing 0 has no effect. 0b - Write: no effect. Read: no cache command active 1b - Write: initiates command; Read: cache command active
30-28 —	Reserved
27 PUSHW1	Push Way 1 0b - No operation 1b - When you write 1 to GO, push all modified lines in way 1
26 INVW1	Invalidate Way 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If CCR[PUSHW1] and CCR[INVW1] are 1, then after you write 1 to CCR[GO], this field pushes all modified lines in way 1 and invalidates all lines in way 1 (clear way 1). 0b - No operation 1b - When you write 1 to GO, invalidates all lines in way 1
25 PUSHW0	Push Way 0 0b - No operation 1b - When you write 1 to GO, push all modified lines in way 0
24 INVW0	Invalidate Way 0 **NOTE** If CCR[PUSHW0] and CCR[INVW0] are 1, then after you write 1 to CCR[GO], this field pushes all modified lines in way 0 and invalidates all lines in way 0 (clear way 0). 0b - No operation 1b - When you write 1 to GO, invalidates all lines in way 0.
23-4 —	Reserved
3 FRCNOALLC	Forces No Allocation on Cache Misses 0b - Allocation on cache misses 1b - Forces no allocation on cache misses (must also have FRCWT asserted)
2 FRCWT	Force Write Through Mode Specifies whether all cacheable spaces to write through are forced. 0b - Does not force 1b - Force
1 —	Reserved Always write 0 to this field to maintain compatibility.
0 ENCACHE	Cache Enable Enables the cache. 0b - Disable 1b - Enable

39.7.1.3 Cache Line Control (CLCR)

Offset

Register	Offset
CLCR	4h

Function

Defines the specific line-sized cache operations that you perform using a specific cache line address or a physical address.

If you specify the physical address, you can search cache in both the ways, and the command only performs on the way that hits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				LACC	LADSEL	LCMD		Reserved	LCWAY	LCIMB	LCIVB	0			TDSEL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	WSEL	CACHEADDR												0	LGO
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 LACC	Line Access Type 0b - Read 1b - Write
26 LADSEL	Line Address Select Specifies the address (cache or physical). When using the cache address, the way must also be specified in CLCR[WSEL]. When using the physical address, both ways are searched and the command is performed only if a hit. 0b - Cache address 1b - Physical address
25-24	Line Command

Table continues on the next page...

Table continued from the previous page...

Field	Function
LCMD	00b - Search and read or write 01b - Invalidate 10b - Push 11b - Clear
23 —	Reserved
22 LCWAY	Line Command Way Indicates the way that the line command uses. Applies only if CLCR[LCIVB] = 1.
21 LCIMB	Line Command Initial Modified Specifies the initial state of the modified bit. If the command uses cache address and way, then this field shows the initial state of the modified bit. If the command uses physical address and a hit, then this field shows the initial state of the modified bit. If a miss, this field reads zero.
20 LCIVB	Line Command Initial Valid Specifies the initial state of the valid bit. If the command uses cache address and way, then this field shows the initial state of the valid bit. If the command uses physical address and a hit, then this field shows the initial state of the valid bit. If a miss, this field reads zero.
19-17 —	Reserved
16 TDSEL	Tag or Data Select Selects tag or data for search and read or write commands. 0b - Data 1b - Tag
15 —	Reserved
14 WSEL	Way Select Selects the way for line commands. 0b - Way 0 1b - Way 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-2 CACHEADDR	Cache Address Represents bits [13:2] of the cache address: CLCR [12:4] accesses the tag arrays. CLCR [12:2] accesses the data arrays.
1 —	Reserved
0 LGO	Initiate Cache Line Command Initiates the cache command that CLCR[LCMD], CLCR[LADSEL], and CLCR[LACC] indicate, if you write 1 to this field. Reading this field indicates whether a line command is active. **NOTE** This field remains 1 until the command completes. Writing 0 has no effect. **NOTE** This field is shared with CSAR[LGO] 0b - Write: no effect. Read: no line command active. 1b - Write: initiate line command. Read: line command active.

39.7.1.4 Cache Search Address (CSAR)

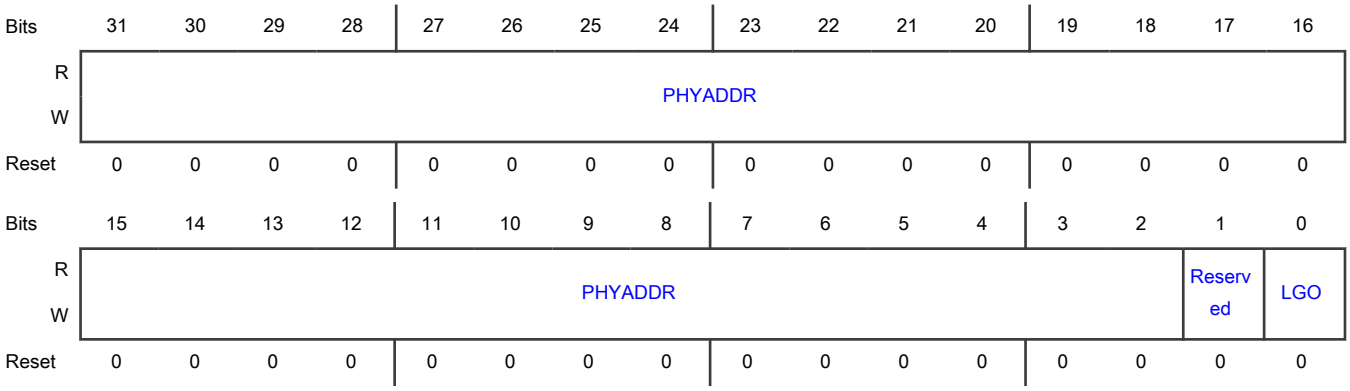
Offset

Register	Offset
CSAR	8h

Function

Defines the explicit cache address or the physical address for line-sized commands specified in CLCR[LADSEL].

Diagram



Fields

Field	Function
31-2 PHYADDR	Physical Address Represents bits [31:2] of the system address. CSAR [31:13] is used for tag compare. CSAR [12:4] accesses the tag arrays. CSAR [12:2] accesses the data arrays.
1 —	Reserved
0 LGO	Initiate Cache Line Command Initiates the cache line command that CLCR[LCMD], CLCR[LADSEL], and CLCR[LACC] indicate. Reading this field indicates whether a line command is active. **NOTE** This field remains 1 until the command completes. Writing 0 has no effect. **NOTE** This field is shared with CLCR[LGO] 0b - Write: no effect. Read: no line command active. 1b - Write: initiate line command. Read: line command active.

39.7.1.5 Cache Read/Write Value (CCVR)

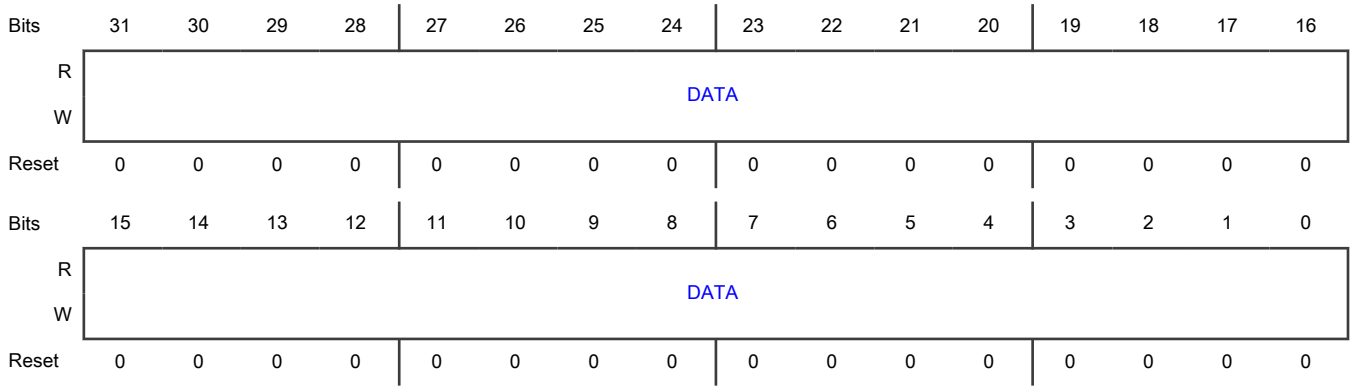
Offset

Register	Offset
CCVR	Ch

Function

Sources write data or return read data for the commands specified in Cache Line Control (CLCR).

Diagram



Fields

Field	Function
31-0 DATA	Cache Read/Write Data Specifies the data that you want to read or write during tag search or data search. For tag search, read or write: • CCVR [31:13] for tag array R/W value. • CCVR [12:4] for tag set address on reads; unused on writes. • CCVR[3:2] are reserved. • CCVR[1] tag modify bit. • CCVR[0] tag valid bit. For data search, read or write: • CCVR[31:0] for data array R/W value.

Chapter 40
Performance Monitor (PERFMON)

40.1 Chip-specific CMX_PERFMON Information

Table 293. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

40.2 Overview

CMX_PERFMON contains counters that you can configure to count events used to calculate the performance of a CPU, cache, or memory. You can configure CMX_PERFMON to select the events to be counted for each counter. An additional instruction counter is available to count CPU instruction fetches.

Block diagram provides an overview of CMX_PERFMON, which includes three event counters and an instruction counter.

40.2.1 Block diagram

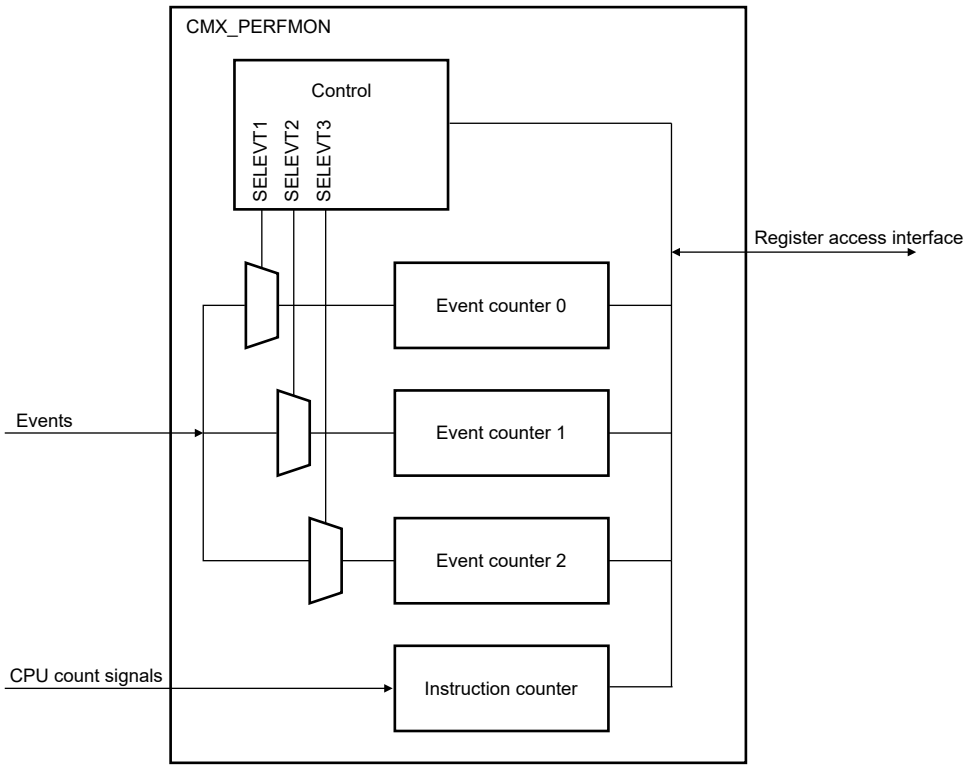


Figure 282. Block diagram

40.2.2 Features

- Three 40-bit event counters
- One 40-bit instruction counter
- Programmable event select
- Countable events that include cache, branch fetches, and stalls
- Memory mapped performance monitor that counts instructions, cache, branch, IPS, and TCM stats
- Count filter based on privilege level

40.3 Functional description

40.3.1 Operations

See the chip-specific CMX_PERFMON information for details about the countable events for the performance counters per CMX_PERFMON instance on this chip.

CMX_PERFMON has 3 programmable counters to count CPU events. The event to be counted is configured using the PMCR*m* register.

Each counter is a 40-bit register that tracks the number of occurrences of the selected event.

The counters can be programmed to take one of the 3 states:

- At idle state, the counters are in idle and do not perform any operation.

- At local start state, the counters are actively listening to CPU and cache and start incrementing when the selected events are observed.
- At local stop state, the counters stop listening to all components. The 3 event counter can be independently cleared by writing to PMCR*m*[RECTR*n*], where *n* denotes the counter instance.

40.3.2 Operating modes

The following table shows:

- The operating modes that you can specify for CMX_PERFMON.
- The PMCR*n*[CMODE] value that you must specify to count events in each mode.

Table 294. Operating modes

Mode	Description	CMODE value (binary)
Privileged	Only events in privilege mode are counted	11
User	Only events in user mode are counted	10
Both Privileged and User	Events in either privilege or user mode can be counted	00

Use PMCR*n*[SSC] to start and stop the counters.

40.3.3 Clocking

This module has no clocking considerations.

40.3.4 Interrupts

This module has no interrupts.

40.4 External signals

CMX_PERFMON has no external signals.

40.5 Initialization

To initialize CMX_PERFMON:

1. Program PMCR*m*[SELEVT*n*] to select the specific CPU event to be counted by this counter, where *n* denotes the counter instance.
2. Set PMCR*n*[CMODE] to filter each counter for the selected event while the CPU is in privilege, user, or either mode.
3. Set PMCR*n*[SSC] to program the counter to be in one of idle, local start, or local stop states.
4. Set PMCR*m*[RECTR*n*] to clear the counters independently where *n* denotes the counter instance.

40.6 Memory map and register definition

40.6.1 CMX_PERFMON register descriptions

40.6.1.1 CMX_PERFMON memory map

M33_PCF1 base address: 443E_0000h

M33_PSF1 base address: 443F_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Performance Monitor Control (PMCR)	32	RW	0000_0000h
10h	Performance Monitor Instruction Counter (PMICTR_HI)	8	R	00h
14h	Performance Monitor Instruction Counter (PMICTR_LO)	32	R	0000_0000h
18h	Performance Monitor Event Counter (PMECTR1_HI)	8	R	00h
1Ch	Performance Monitor Event Counter (PMECTR1_LO)	32	R	0000_0000h
20h	Performance Monitor Event Counter (PMECTR2_HI)	8	R	00h
24h	Performance Monitor Event Counter (PMECTR2_LO)	32	R	0000_0000h
28h	Performance Monitor Event Counter (PMECTR3_HI)	8	R	00h
2Ch	Performance Monitor Event Counter (PMECTR3_LO)	32	R	0000_0000h

40.6.1.2 Performance Monitor Control (PMCR)

Offset

Register	Offset
PMCR	0h

Function

Specifies:

- The events that the PMECTR*n*HI and PMECTR*n*LO count.
- The count mode.
- The start and stop control.
- The enables for the counters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SELEVT3								SELEVT2						SELEVT1	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SELEVT1				RECTR3	RECTR2	RECTR1	RICTR	DCIFSH	CMODE			SSC			MENB
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 SELEVT3	Select Event 3 Selects the event to be counted in PMECTR3. See the chip-specific CMX_PERFMON information to select which event PMECTR3 counts.
24-18 SELEVT2	Select Event 2 Selects the event to be counted in PMECTR2. See the chip-specific CMX_PERFMON information to select which event PMECTR2 counts.
17-11 SELEVT1	Select Event 1 Selects the event to be counted in PMECTR1. See the chip-specific CMX_PERFMON information to select which event PMECTR1 counts.
10 RECTR3	Reset Event Counter 3 Specifies whether the counter runs normally or the counter value resets at the end of the cycle. Write 1 to this field to clear this counter. This field does not return to 0 automatically, so if you want to restart the counter after clearing it, write 0 to this field. 0b - Run normally 1b - Reset
9 RECTR2	Reset Event Counter 2 Specifies whether the counter runs normally or the counter value resets at the end of the cycle. Write 1 to this field to clear this counter. This field does not return to 0 automatically, so if you want to restart the counter after clearing it, write 0 to this field. 0b - Run normally 1b - Reset
8 RECTR1	Reset Event Counter 1 Specifies whether the counter runs normally or the counter value resets at the end of the cycle. Write 1 to this field to clear this counter. This field does not return to 0 automatically, so if you want to restart the counter after clearing it, write 0 to this field. 0b - Run normally 1b - Reset
7 RICTR	Reset Instruction Counter Determines whether events reset or clear the instruction counter. Write 1 to clear the instruction counter. This field must be 0 to restart the instruction counter. 0b - Do not reset 1b - Clear
6	Disable Counters if Stopped or Halted

Table continues on the next page...

Table continued from the previous page...

Field	Function
DCIFSH	Specifies whether counters continue counting or stop counting when the CPU is halted. 0b - Continue 1b - Stop
5-4 CMODE	Count Mode Determines whether events are counted for Privileged mode, User mode, or both Privileged and User modes. This setting affects the operation of all event counters. 00b - Counted in both User and Privileged modes 01b - Reserved 10b - Counted only in User mode 11b - Counted only in Privileged mode
3-1 SSC	Start and Stop Control Provides a three-phase mechanism to start and stop the counters. It includes a prioritized scheme with the following relative priorities: local start > local stop The start or stop affects all performance monitor event counters concurrently, so that the counters are coherent. 000b - Idle or no-op 001b - Local stop 010b,011b - Local start 100b - Reserved 101b - Reserved 110b,111b - Reserved
0 MENB	Module Is Enabled Indicates whether CMX_PERFMON is enabled. Enabling CMX_PERFMON does not start the counters. To start the counters, write a nonzero value to PMCR*n*[SSC]. 0b - Disabled 1b - Enabled

40.6.1.3 Performance Monitor Instruction Counter (PMICTR_HI)

Offset

Register	Offset
PMICTR_HI	10h

Function

Provides part of a 40-bit counter that counts executed instructions.

Diagram

Bits	7	6	5	4	3	2	1	0
R	ICTR							
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-0 ICTR	Instruction Counter Specifies the upper 8 bits of the 40-bit instruction counter.

40.6.1.4 Performance Monitor Instruction Counter (PMICTR_LO)

Offset

Register	Offset
PMICTR_LO	14h

Function

Provides part of a 40-bit counter that counts executed instructions.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ICTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0	Instruction Counter

Table continues on the next page...

Field	Function
ICTR	Specifies the lower 32 bits of the instruction counter. The value in this field increments each time an instruction is executed.

40.6.1.5 Performance Monitor Event Counter (PMECTR1_HI - PMECTR3_HI)

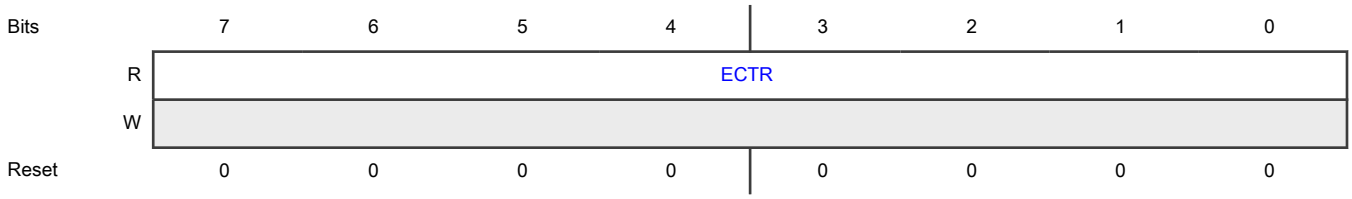
Offset

Register	Offset
PMECTR1_HI	18h
PMECTR2_HI	20h
PMECTR3_HI	28h

Function

Provides part of a 40-bit event counter that you configure with PMCR*m*[SELEVT*n*].

Diagram



Fields

Field	Function
7-0 ECTR	Event Counter Specifies the upper 8 bits of the event[*n*] counter.

40.6.1.6 Performance Monitor Event Counter (PMECTR1_LO - PMECTR3_LO)

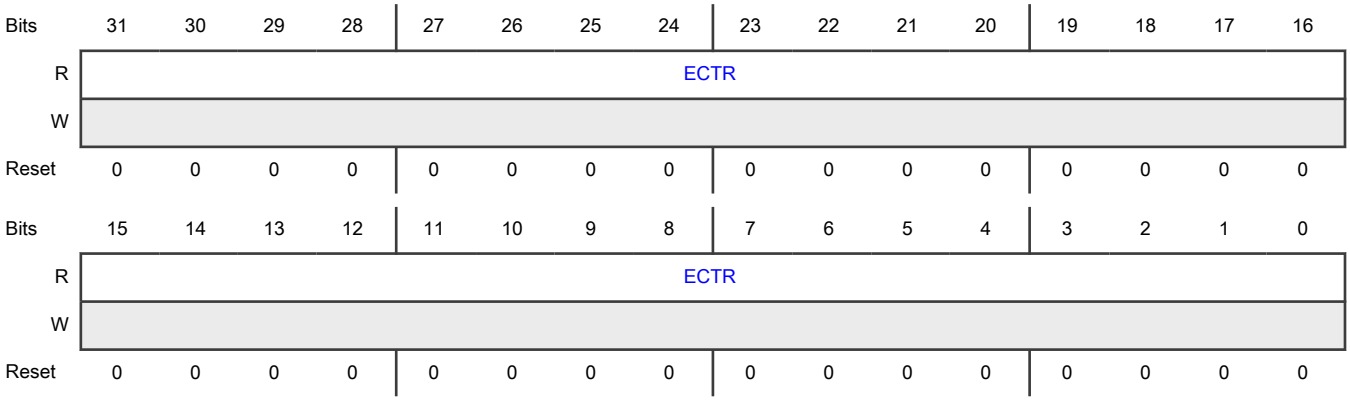
Offset

Register	Offset
PMECTR1_LO	1Ch
PMECTR2_LO	24h
PMECTR3_LO	2Ch

Function

Provides part of a 40-bit event counter that you configure with PMCR*m*[SELEVT*n*].

Diagram



Fields

Field	Function
31-0 ECTR	Event Counter Specifies the lower 32 bits of the event[*n*] counter. The 40-bit counter value increments each time the event selected in PMCR*m*[SELEVT*n*] occurs.

Chapter 41
Memory ECC Controller (MECC64)

41.1 Chip-specific MECC64 Information

Table 295. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

41.2 Overview

The Memory ECC Controller (MECC64) module supports Single Error Correction and Double Error Detection (SEC-DED) ECC functions to On-Chip RAM (OCRAM) access. It can generate a 8-bit ECC code by Hsiao Hamming algorithm for 64-bit data. Corresponding OCRAM space is available to store ECC code.

41.2.1 Block diagram

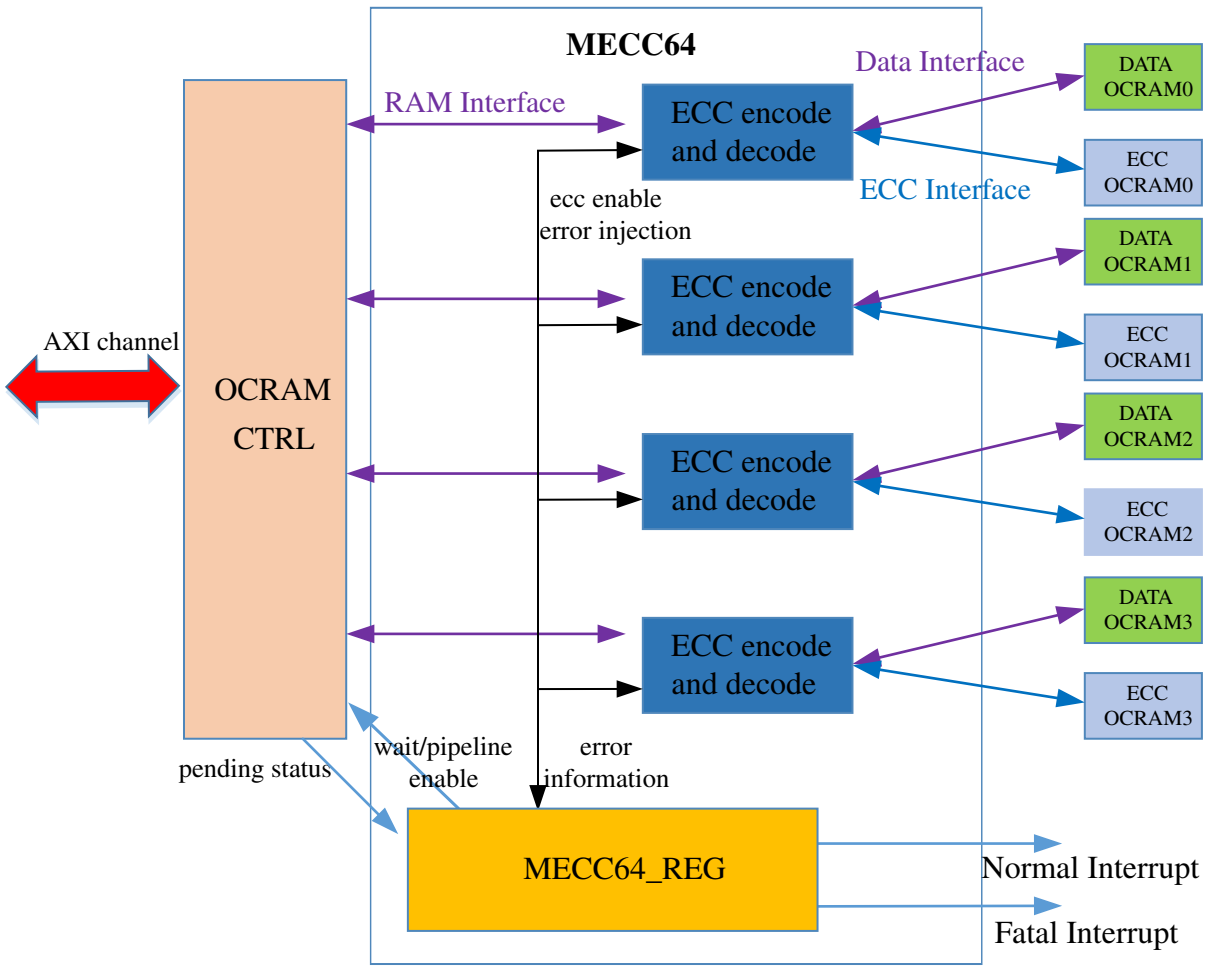


Figure 283. MECC64 block diagram

The following list shows detailed register/bitfield correspondance in the block diagram.

- ecc enable:
  - PIPE_ECC_EN[ECC_EN]
- error injection:
  - ERR_DATA_INJ_LOW/HIGH*n*
  - ERR_ECC_INJ*n*
- error information:
  - SINGLE_ERR_ADDR_ECC*n*
  - SINGLE_ERR_DATA_LOW/HIGH*n*
  - SINGLE_ERR_POS_LOW/HIGH*n*
  - MULTI_ERR_ADDR_ECC*n*
  - MULTI_ERR_DATA_LOW/HIGH*n*
- pending status:
  - PENDING_STAT

- wait/pipeline enable:
  - PIPE_ECC_EN[READ_DATA_WAIT_EN]
  - PIPE_ECC_EN[READ_ADDR_PIPE_EN]
  - PIPE_ECC_EN[WRITE_DATA_PIPE_EN]
  - PIPE_ECC_EN[WRITE_ADDR_PIPE_EN]

41.2.2 Features

The MECC64 includes the following features:

- Generates a 8-bit ECC code for 64-bit write data. Write data and ECC code are written into seperate OCRAM banks.
- Supports SEC-DED function for 64-bit read data and 8-bit ECC code.
- Integrate 4 banks OCRAM controller for converting AXI transaction to RAM interface.
- Error injection on write data and ECC code for debug purpose.
- Two different interrupt types: Normal and Fatal.

41.3 Functional description

The MECC64 converts AXI interface to RAM interface signals, and provides ECC encode/decode function for OCRAM write/read. For each data in OCRAM bank, corresponding space in ECC OCRAM banks is available to store ECC code when ECC function is enabled.

The OCRAM bank address allocation is as below:

- Bank0: ocram_base_address+0x20*i
- Bank1: ocram_base_address+0x20*i+0x8
- Bank2: ocram_base_address+0x20*i+0x10
- Bank3: ocram_base_address+0x20*i+0x18

where i = 0,1,2,3,4,...

41.3.1 Interface Conversion

The MECC64 integrates 4 banks OCRAM controller which converts AXI master interface signals to 4 groups of OCRAM interface signals (see OCRAM chapter for details). The 4 OCRAM banks are organized with lower 2 bits of address interleaved. This allows a read access and a write access to be processed at the same time if they are targeted to different OCRAM banks. OCRAM controller can support to add pipeline or wait-state in read/write access by register configuration.

41.3.2 ECC encode

When ECC is enabled, the 4 groups of OCRAM write access will enter into seperate ECC encode channels. In a ECC encode channel, it will generate 8 bits ECC code for 64 bits OCRAM write data. Then 64 bits OCRAM write access will split into one 64 bits data write and one 8 bits ECC code write. The 64 bits data will be written into one of the DATA OCRAM banks according to address, and 8 bits ECC code will be written simutaneously into the corresponding ECC OCRAM bank.

41.3.3 Error injection

The MECC64 can inject error in write data and ECC code by configuring ERR_DATA_INJ* and ERR_ECC_INJ* registers. The 64 bits write data and 8 bits ECC code can be inverted if corresponding bits in ERR_DATA_INJ* and ERR_ECC_INJ* registers are set. Each OCRAM bank has separate error injection registers. This feature is mainly used for debug.

41.3.4 ECC decode

When ECC is enabled, the 64 bits OCRAM read access will be split into one 64 bits data read and one 8 bits ECC code read. After getting read data from DATA OCRAM and ECC code from the corresponding ECC OCRAM, ECC decode channel can report single bit error or multiple bits error if ECC check is failed. At the same time, the error information will be saved in the module's error registers for debug. When single bit error is detected on read data, the error bit in read data will be corrected. In this case, AXI master will get read data correctly. When multiple bits error is detected on read data, error read data will not be corrected. In this case, master will get read data in error. When single bit error or multiple bits error happens, interrupt will be asserted if the corresponding enable bits are set in ERROR_STAT_EN and ERROR_SIG_EN registers. Each OCRAM bank has separate error and interrupt registers.

NOTE

- If read access is 64-bit, the MECC64 will generate single or multi error interrupt once when meeting ECC error.
- If read access is 32-bit (or 16-bit/8-bit), the MECC64 will generate interrupt twice when meeting ECC error, once each in both the low 32 bits and the high 32 bits, even if it has a bit error in only the low 32 bits or the high 32 bits. This is because the ECC function will do ECC code validation on the full 64 bits for one read access.

41.3.5 Interrupts

The MECC64 can generate two kinds of interrupts:

- Normal interrupt can be generated for the following cases:
  - Single bit error is corrected for read;
  - AXI access address exceeds OCRAM address range;
  - AXI write access to OCRAM bank is not full size (wstrb=0xff).
- Fatal interrupt can be generated when 'multiple bits error' is detected for read.

41.3.6 Clocks

There are two clocks in MECC64:

Clock	Description
clk	Main clock
ips_clk	This is for IPS bus, used for register write/read.

There are no special requirements for these two clocks.

41.4 Signals

There are no external signals for MECC.

41.5 Initialization

The steps for initialization process is as below:

1. Enable MECC function (set PIPE_ECC_EN[ECC_EN] = 1).
2. Enable interrupts (set ERR_STAT_EN = 0xFFFF, ERR_SIG_EN = 0xFFFF).

NOTE

- When the MECC function is enabled, the OCRAM memory region can be initialized with 0x00 or 0xFF (initialization data for the first ECC encode), to generate ECC code within OCRAM ECC code region.
  - Initialization is required because the MECC64 needs a complete OCRAM and OCRAM ECC data. When initialized, it generates ECC code depending on initialization data, store data and its ECC code in OCRAM and OCRAM ECC memory.
  - Do such initialization with 64 bits access, with STRD or STMIA instruction. Do not use memset() API, which is STR instruction, in software coding level.
- OCRAM memory with ECC should be non-cacheable, or cacheable with write through, to assure data in OCRAM memory at the write time. It may have different methods to achieve this goal, such as Memory Protection Unit (MPU) or disable cache.

41.6 Application information

There are no application pseudocode examples for MECC64.

41.7 Memory Map and register definition

This section includes the MECC64 module memory map and detailed descriptions of all registers.

41.7.1 MECC64 register descriptions

41.7.1.1 MECC64 memory map

MECC1 base address: 4292_0000h

MECC2 base address: 4293_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Interrupt Status Register (ERR_STATUS)	32	RW	0000_0000h
4h	Error Interrupt Status Enable Register (ERR_STAT_EN)	32	RW	0000_0000h
8h	Error Interrupt Enable Register (ERR_SIG_EN)	32	RW	0000_0000h
Ch - 30h	Error Injection On LOW 32 bits Of OCRAM Banka Write Data (ERR_DATA_INJ_LOW0 - ERR_DATA_INJ_LOW3)	32	RW	0000_0000h
10h - 34h	Error Injection On HIGH 32 bits Of OCRAM Banka Write Data (ERR_DATA_INJ_HIGH0 - ERR_DATA_INJ_HIGH3)	32	RW	0000_0000h
14h - 38h	Error Injection On 8 bits ECC code Of OCRAM Banka Write Data (ERR_ECC_INJ0 - ERR_ECC_INJ3)	32	RW	0000_0000h
3Ch - 78h	Single Error Address And ECC code On OCRAM Banka (SINGLE_ERR_ADDR_ECC0 - SINGLE_ERR_ADDR_ECC3)	32	R	0000_0000h
40h - 7Ch	LOW 32 Bits Single Error Read Data On OCRAM Banka (SINGLE_ERR_DATA_LOW0 - SINGLE_ERR_DATA_LOW3)	32	R	0000_0000h
44h - 80h	HIGH 32 Bits Single Error Read Data On OCRAM Banka (SINGLE_ERR_DATA_HIGH0 - SINGLE_ERR_DATA_HIGH3)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
48h - 84h	LOW Single Error Bit Position On OCRAM Banka (SINGLE_ERR_POS_LOW0 - SINGLE_ERR_POS_LOW3)	32	R	0000_0000h
4Ch - 88h	HIGH Single Error Bit Position On OCRAM Banka (SINGLE_ERR_POS_HIGH0 - SINGLE_ERR_POS_HIGH3)	32	R	0000_0000h
8Ch - B0h	Multiple Error Address And ECC code On OCRAM Banka (MULTI_ERR_ADDR_ECC0 - MULTI_ERR_ADDR_ECC3)	32	R	0000_0000h
90h - B4h	LOW 32 Bits Multiple Error Read Data On OCRAM Banka (MULTI_ERR_DATA_LOW0 - MULTI_ERR_DATA_LOW3)	32	R	0000_0000h
94h - B8h	HIGH 32 Bits Multiple Error Read Data On OCRAM Banka (MULTI_ERR_DATA_HIGH0 - MULTI_ERR_DATA_HIGH3)	32	R	0000_0000h
100h	OCRAM Pipeline And ECC Enable (PIPE_ECC_EN)	32	RW	0000_0000h
104h	Pending Status (PENDING_STAT)	32	R	0000_0000h

41.7.1.2 Error Interrupt Status Register (ERR_STATUS)

Offset

Register	Offset
ERR_STATUS	0h

Function

Error Interrupt Status Register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR_ER...	ADDR_ER...	ADDR_ER...	ADDR_ER...	STRB_ER...	STRB_ER...	STRB_ER...	STRB_ER...	MULTI_E...	MULTI_E...	MULTI_E...	MULTI_E...	SINGLE_...	SINGLE_...	SINGLE_...	SINGLE_...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 ADDR_ERR3	OCRAM Access Error On Bank3 When OCRAM access unallocated address on bank3, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - OCRAM access error does not happen on OCRAM bank3. 1b - OCRAM access error happens on OCRAM bank3.
14 ADDR_ERR2	OCRAM Access Error On Bank2 When OCRAM access unallocated address on bank2, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - OCRAM access error does not happen on OCRAM bank2. 1b - OCRAM access error happens on OCRAM bank2.
13 ADDR_ERR1	OCRAM Access Error On Bank1 When OCRAM access unallocated address on bank1, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - OCRAM access error does not happen on OCRAM bank1. 1b - OCRAM access error happens on OCRAM bank1.
12 ADDR_ERR0	OCRAM Access Error On Bank0 When OCRAM access unallocated address on bank0, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - OCRAM access error does not happen on OCRAM bank0. 1b - OCRAM access error happens on OCRAM bank0.
11 STRB_ERR3	AXI Strobe Error On OCRAM Bank3 When AXI write access to OCRAM bank3 is not full size(wstrb=0xff), this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - AXI strobe error does not happen on OCRAM bank3. 1b - AXI strobe error happens on OCRAM bank3.
10 STRB_ERR2	AXI Strobe Error On OCRAM Bank2 When AXI write access to OCRAM bank2 is not full size(wstrb=0xff), this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - AXI strobe error does not happen on OCRAM bank2. 1b - AXI strobe error happens on OCRAM bank2.
9	AXI Strobe Error On OCRAM Bank1

Table continues on the next page...

Table continued from the previous page...

Field	Function
STRB_ERR1	When AXI write access to OCRAM bank1 is not full size(wstrb=0xff), this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - AXI strobe error does not happen on OCRAM bank1. 1b - AXI strobe error happens on OCRAM bank1.
8 STRB_ERR0	AXI Strobe Error On OCRAM Bank0 When AXI write access to OCRAM bank0 is not full size(wstrb=0xff), this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - AXI strobe error does not happen on OCRAM bank0. 1b - AXI strobe error happens on OCRAM bank0.
7 MULTI_ERR3	Multiple Bits Error On OCRAM Bank3 When detect multiple bits error on read data from OCRAM bank3, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - Multiple bits error does not happen on OCRAM bank3. 1b - Multiple bits error happens on OCRAM bank3.
6 MULTI_ERR2	Multiple Bits Error On OCRAM Bank2 When detect multiple bits error on read data from OCRAM bank2, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - Multiple bits error does not happen on OCRAM bank2. 1b - Multiple bits error happens on OCRAM bank2.
5 MULTI_ERR1	Multiple Bits Error On OCRAM Bank1 When detect multiple bits error on read data from OCRAM bank1, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - Multiple bits error does not happen on OCRAM bank1. 1b - Multiple bits error happens on OCRAM bank1.
4 MULTI_ERR0	Multiple Bits Error On OCRAM Bank0 When detect multiple bits error on read data from OCRAM bank0, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - Multiple bits error does not happen on OCRAM bank0. 1b - Multiple bits error happens on OCRAM bank0.
3 SINGLE_ERR3	Single Bit Error On OCRAM Bank3 When detect single bit error on read data from OCRAM bank3, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - Single bit error does not happen on OCRAM bank3. 1b - Single bit error happens on OCRAM bank3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 SINGLE_ERR2	Single Bit Error On OCRAM Bank2 When detect single bit error on read data from OCRAM bank2, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - Single bit error does not happen on OCRAM bank2. 1b - Single bit error happens on OCRAM bank2.
1 SINGLE_ERR1	Single Bit Error On OCRAM Bank1 When detect single bit error on read data from OCRAM bank1, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - Single bit error does not happen on OCRAM bank1. 1b - Single bit error happens on OCRAM bank1.
0 SINGLE_ERR0	Single Bit Error On OCRAM Bank0 When detect single bit error on read data from OCRAM bank0, this bit will be asserted if corresponding status enable bit in ERR_STAT_EN register is set. 0b - Single bit error does not happen on OCRAM bank0. 1b - Single bit error happens on OCRAM bank0.

41.7.1.3 Error Interrupt Status Enable Register (ERR_STAT_EN)

Offset

Register	Offset
ERR_STAT_EN	4h

Function

Setting the bits in this register to 1 enables the corresponding Error Interrupt Status to be set by the specified event.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR_ER...	ADDR_ER...	ADDR_ER...	ADDR_ER...	STRB_ER...	STRB_ER...	STRB_ER...	STRB_ER...	MULTI_E...	MULTI_E...	MULTI_E...	MULTI_E...	SINGLE_...	SINGLE_...	SINGLE_...	SINGLE_...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 ADDR_ERR3_STAT_EN	OCRAM Access Error Status Enable On Bank3 0b - Disabled 1b - Enabled
14 ADDR_ERR2_STAT_EN	OCRAM Access Error Status Enable On Bank2 0b - Disabled 1b - Enabled
13 ADDR_ERR1_STAT_EN	OCRAM Access Error Status Enable On Bank1 0b - Disabled 1b - Enabled
12 ADDR_ERR0_STAT_EN	OCRAM Access Error Status Enable On Bank0 0b - Disabled 1b - Enabled
11 STRB_ERR3_STAT_EN	AXI Strobe Error Status Enable On OCRAM Bank3 0b - Disabled 1b - Enabled
10 STRB_ERR2_STAT_EN	AXI Strobe Error Status Enable On OCRAM Bank2 0b - Disabled 1b - Enabled
9 STRB_ERR1_STAT_EN	AXI Strobe Error Status Enable On OCRAM Bank1 0b - Disabled 1b - Enabled
8 STRB_ERR0_STAT_EN	AXI Strobe Error Status Enable On OCRAM Bank0 0b - Disabled 1b - Enabled
7 MULTI_ERR3_STAT_EN	Multiple Bits Error Status Enable On OCRAM Bank3 0b - Disabled 1b - Enabled
6 MULTI_ERR2_STAT_EN	Multiple Bits Error Status Enable On OCRAM Bank2 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 MULTI_ERR1_STAT_EN	Multiple Bits Error Status Enable On OCRAM Bank1 0b - Disabled 1b - Enabled
4 MULTI_ERR0_STAT_EN	Multiple Bits Error Status Enable On OCRAM Bank0 0b - Disabled 1b - Enabled
3 SINGLE_ERR3_STAT_EN	Single Bit Error Status Enable On OCRAM Bank3 0b - Disabled 1b - Enabled
2 SINGLE_ERR2_STAT_EN	Single Bit Error Status Enable On OCRAM Bank2 0b - Disabled 1b - Enabled
1 SINGLE_ERR1_STAT_EN	Single Bit Error Status Enable On OCRAM Bank1 0b - Disabled 1b - Enabled
0 SINGLE_ERR0_STAT_EN	Single Bit Error Status Enable On OCRAM Bank0 0b - Disabled 1b - Enabled

41.7.1.4 Error Interrupt Enable Register (ERR_SIG_EN)

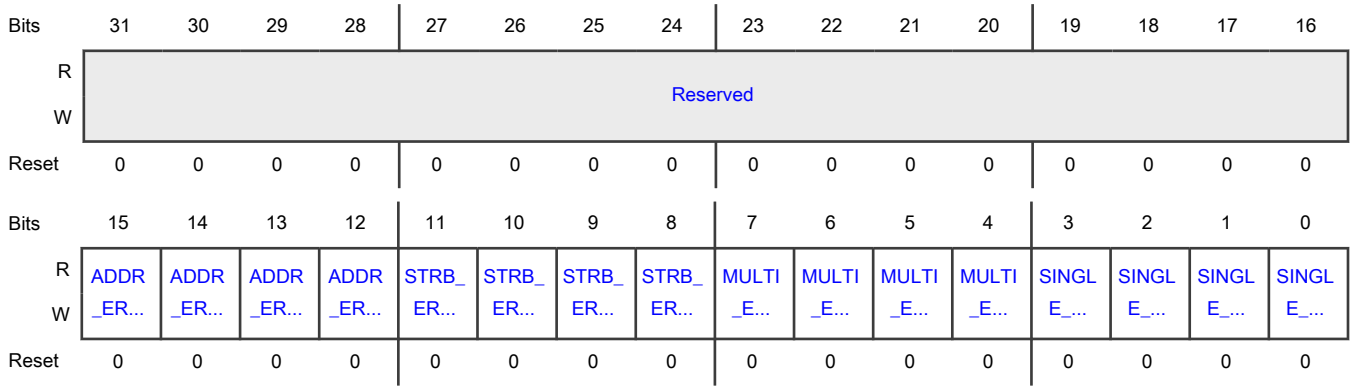
Offset

Register	Offset
ERR_SIG_EN	8h

Function

This register is used to select which interrupt status is indicated to the Host System as the interrupt. Setting any of these bits to 1 enables interrupt generation. The corresponding Status register bit will generate an interrupt when the corresponding interrupt signal enable bit is set.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 ADDR_ERR3_SIG_EN	OCRAM Access Error Interrupt Enable On Bank3 0b - Disabled 1b - Enabled
14 ADDR_ERR2_SIG_EN	OCRAM Access Error Interrupt Enable On Bank2 0b - Disabled 1b - Enabled
13 ADDR_ERR1_SIG_EN	OCRAM Access Error Interrupt Enable On Bank1 0b - Disabled 1b - Enabled
12 ADDR_ERR0_SIG_EN	OCRAM Access Error Interrupt Enable On Bank0 0b - Disabled 1b - Enabled
11 STRB_ERR3_SIG_EN	AXI Strobe Error Interrupt Enable On OCRAM Bank3 0b - Disabled 1b - Enabled
10 STRB_ERR2_SIG_EN	AXI Strobe Error Interrupt Enable On OCRAM Bank2 0b - Disabled 1b - Enabled
9	AXI Strobe Error Interrupt Enable On OCRAM Bank1 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
STRB_ERR1_SIG_EN	1b - Enabled
8 STRB_ERR0_SIG_EN	AXI Strobe Error Interrupt Enable On OCRAM Bank0 0b - Disabled 1b - Enabled
7 MULTI_ERR3_SIG_EN	Multiple Bits Error Interrupt Enable On OCRAM Bank3 0b - Disabled 1b - Enabled
6 MULTI_ERR2_SIG_EN	Multiple Bits Error Interrupt Enable On OCRAM Bank2 0b - Disabled 1b - Enabled
5 MULTI_ERR1_SIG_EN	Multiple Bits Error Interrupt Enable On OCRAM Bank1 0b - Disabled 1b - Enabled
4 MULTI_ERR0_SIG_EN	Multiple Bits Error Interrupt Enable On OCRAM Bank0 0b - Disabled 1b - Enabled
3 SINGLE_ERR3_SIG_EN	Single Bit Error Interrupt Enable On OCRAM Bank3 0b - Disabled 1b - Enabled
2 SINGLE_ERR2_SIG_EN	Single Bit Error Interrupt Enable On OCRAM Bank2 0b - Disabled 1b - Enabled
1 SINGLE_ERR1_SIG_EN	Single Bit Error Interrupt Enable On OCRAM Bank1 0b - Disabled 1b - Enabled
0 SINGLE_ERR0_SIG_EN	Single Bit Error Interrupt Enable On OCRAM Bank0 0b - Disabled 1b - Enabled

41.7.1.5 Error Injection On LOW 32 bits Of OCRAM Banka Write Data (ERR_DATA_INJ_LOW0 - ERR_DATA_INJ_LOW3)

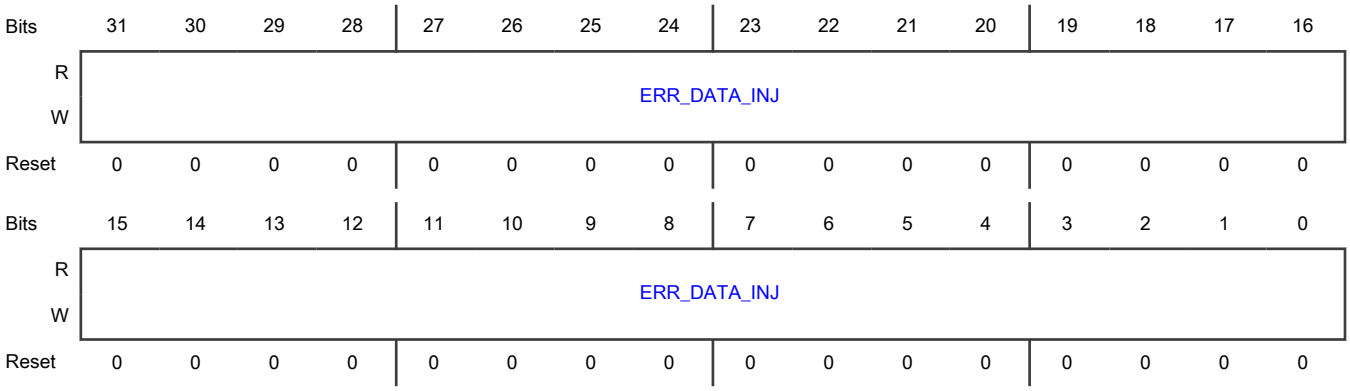
Offset

Register	Offset
ERR_DATA_INJ_LOW0	Ch
ERR_DATA_INJ_LOW1	18h
ERR_DATA_INJ_LOW2	24h
ERR_DATA_INJ_LOW3	30h

Function

LOW 32 bits of OCRAM bank0 write data will be reversed if corresponding bits in this register are set.

Diagram



Fields

Field	Function
31-0 ERR_DATA_INJ	Error Injection On LOW 32 bits Of OCRAM Bank0 Write Data LOW 32 bits of OCRAM bank0 write data will be reversed if corresponding bits in this register are set. For example: • If it is set to 0x00000001, then bit 0 of LOW 32bits of OCRAM bank0 write data will be reversed. • If it is set to 0x00000003, then bit 0 and bit 1 of LOW 32bits of OCRAM bank0 write data will be reversed. • if it is set to 0xFFFFFFFF, all bits of LOW 32bits of OCRAM bank0 write data will be reversed.

41.7.1.6 Error Injection On HIGH 32 bits Of OCRAM Banka Write Data (ERR_DATA_INJ_HIGH0 - ERR_DATA_INJ_HIGH3)

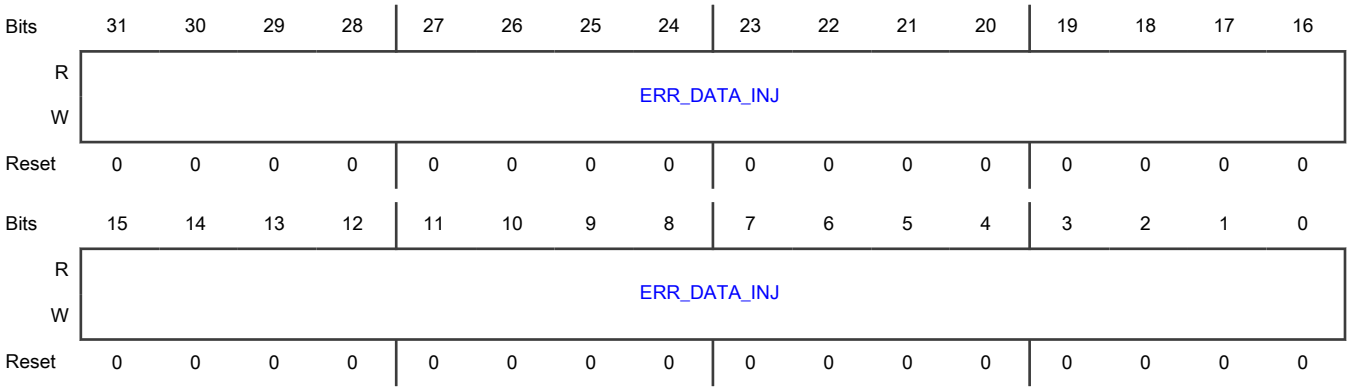
Offset

Register	Offset
ERR_DATA_INJ_HIGH0	10h
ERR_DATA_INJ_HIGH1	1Ch
ERR_DATA_INJ_HIGH2	28h
ERR_DATA_INJ_HIGH3	34h

Function

HIGH 32 bits of OCRAM bank0 write data will be reversed if corresponding bits in this register are set.

Diagram



Fields

Field	Function
31-0 ERR_DATA_INJ	Error Injection On HIGH 32 bits Of OCRAM Bank0 Write Data HIGH 32 bits of OCRAM bank0 write data will be reversed if corresponding bits in this register are set. For example: • If it is set to 0x00000001, then bit 0 of HIGH 32bits of OCRAM bank0 write data will be reversed. • If it is set to 0x00000003, then bit 0 and bit 1 of HIGH 32bits of OCRAM bank0 write data will be reversed. • if it is set to 0xFFFFFFFF, all bits of HIGH 32bits of OCRAM bank0 write data will be reversed.

41.7.1.7 Error Injection On 8 bits ECC code Of OCRAM Banka Write Data (ERR_ECC_INJ0 - ERR_ECC_INJ3)

Offset

Register	Offset
ERR_ECC_INJ0	14h
ERR_ECC_INJ1	20h
ERR_ECC_INJ2	2Ch
ERR_ECC_INJ3	38h

Function

8 bits ECC code of OCRAM bank0 write data will be reversed if corresponding bits in this register are set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								ERR_ECC_INJ							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 ERR_ECC_INJ	Error Injection On 8 bits ECC code Of OCRAM Bank0 Write Data 8 bits ECC code of OCRAM bank0 write data will be reversed if corresponding bits in this register are set.

41.7.1.8 Single Error Address And ECC code On OCRAM Banka (SINGLE_ERR_ADDR_ECC0 - SINGLE_ERR_ADDR_ECC3)

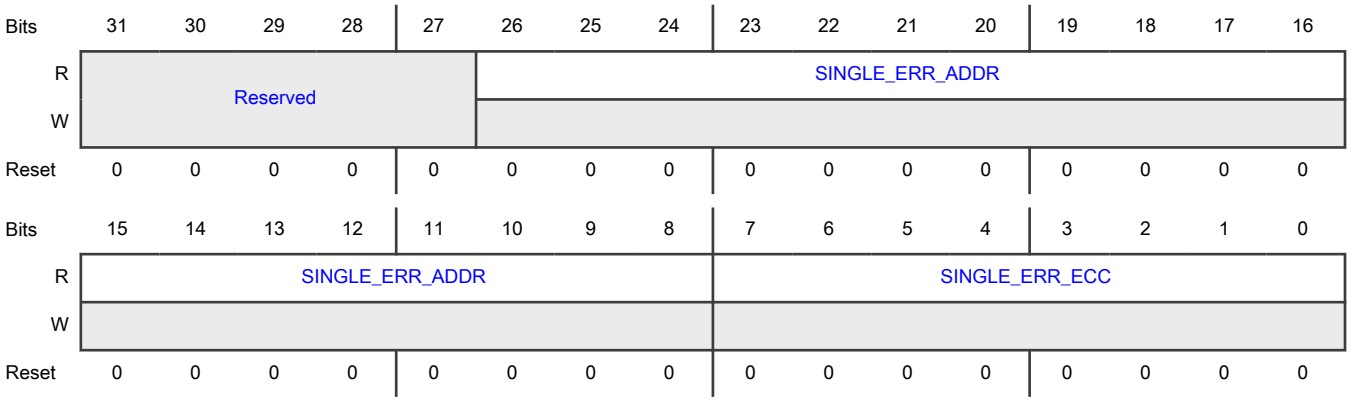
Offset

Register	Offset
SINGLE_ERR_ADDR_ECC0	3Ch
SINGLE_ERR_ADDR_ECC1	50h
SINGLE_ERR_ADDR_ECC2	64h
SINGLE_ERR_ADDR_ECC3	78h

Function

When single bit error is corrected in OCRAM bank0 read access, this register stores OCRAM address and ECC code.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-8 SINGLE_ERR_ADDR	Single Error Address On OCRAM Bank0 When single bit error is corrected in OCRAM bank0 read access, this field stores OCRAM address. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>MECC1</td><td>SINGLE_ERR_ADDR_ECC0– SINGLE_ERR_ADDR_ECC3</td><td>—</td></tr><tr><td>MECC2</td><td>SINGLE_ERR_ADDR_ECC0– SINGLE_ERR_ADDR_ECC3[25–8]</td><td>SINGLE_ERR_ADDR_ECC0– SINGLE_ERR_ADDR_ECC3[26]</td></tr></table>
7-0 SINGLE_ERR_ECC	Single Error ECC code On OCRAM Bank0 When single bit error is corrected in OCRAM bank0 read access, this field stores ECC code.

41.7.1.9 LOW 32 Bits Single Error Read Data On OCRAM Banka (SINGLE_ERR_DATA_LOW0 - SINGLE_ERR_DATA_LOW3)

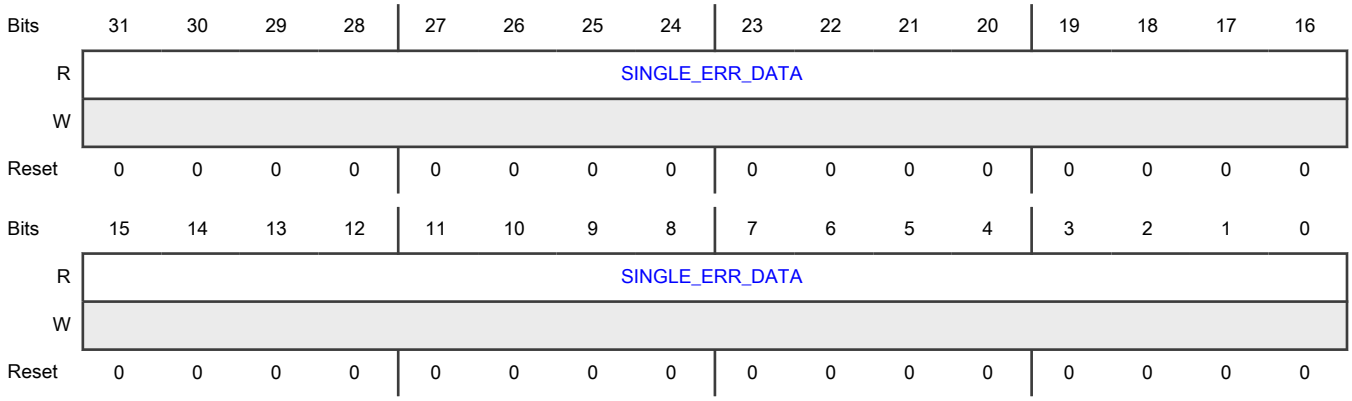
Offset

Register	Offset
SINGLE_ERR_DATA_LOW0	40h
SINGLE_ERR_DATA_LOW1	54h
SINGLE_ERR_DATA_LOW2	68h
SINGLE_ERR_DATA_LOW3	7Ch

Function

When single bit error is corrected in OCRAM bank0 read access, this register stores LOW 32 bits uncorrected read data.

Diagram



Fields

Field	Function
31-0 SINGLE_ERR_DATA	LOW 32 Bits Single Error Read Data On OCRAM Bank0 When single bit error is corrected in OCRAM bank0 read access, this field stores LOW 32 bits uncorrected read data.

41.7.1.10 HIGH 32 Bits Single Error Read Data On OCRAM Banka (SINGLE_ERR_DATA_HIGH0 - SINGLE_ERR_DATA_HIGH3)

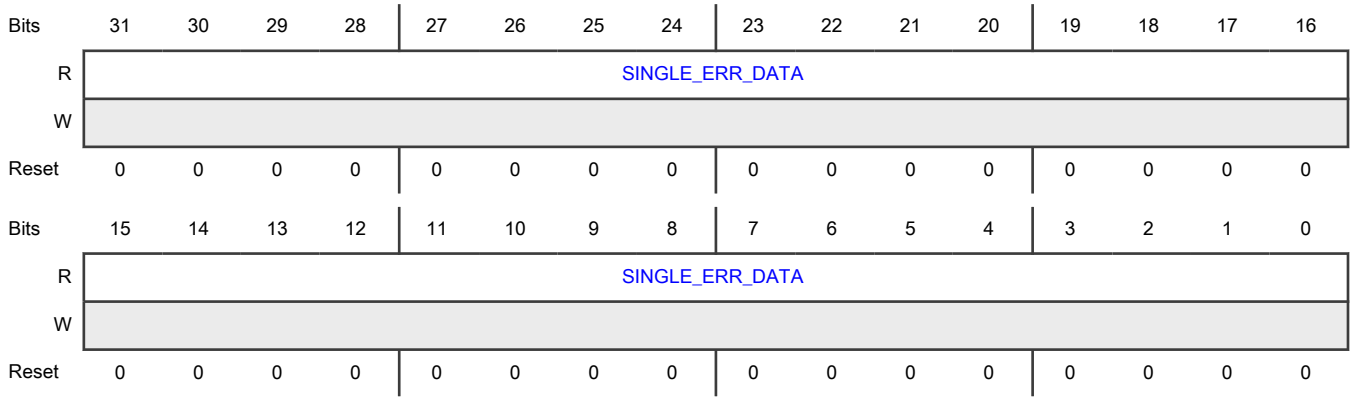
Offset

Register	Offset
SINGLE_ERR_DATA_HIGH0	44h
SINGLE_ERR_DATA_HIGH1	58h
SINGLE_ERR_DATA_HIGH2	6Ch
SINGLE_ERR_DATA_HIGH3	80h

Function

When single bit error is corrected in OCRAM bank0 read access, this register stores HIGH 32 bits uncorrected read data.

Diagram



Fields

Field	Function
31-0 SINGLE_ERR_DATA	HIGH 32 Bits Single Error Read Data On OCRAM Bank0 When single bit error is corrected in OCRAM bank0 read access, this field stores HIGH 32 bits uncorrected read data.

41.7.1.11 LOW Single Error Bit Position On OCRAM Banka (SINGLE_ERR_POS_LOW0 - SINGLE_ERR_POS_LOW3)

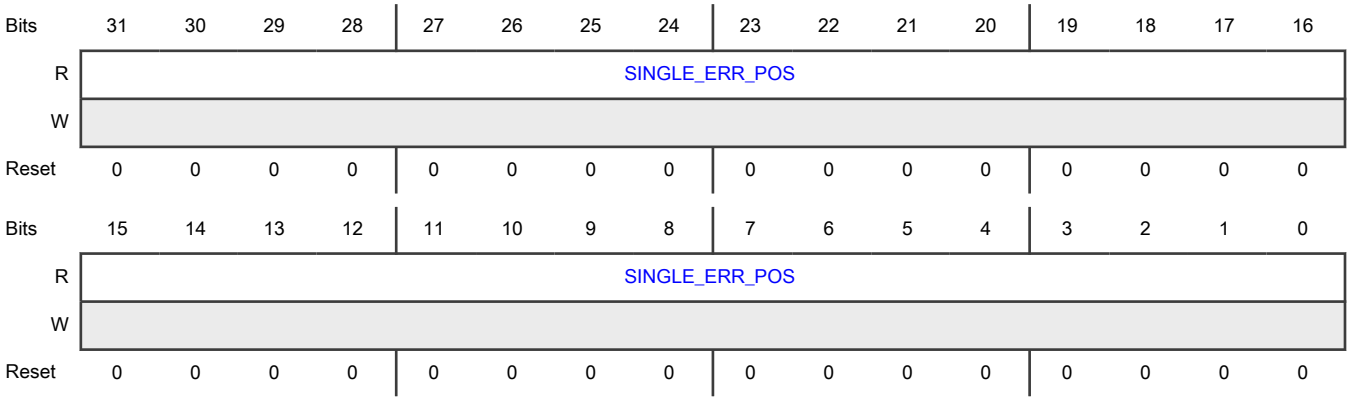
Offset

Register	Offset
SINGLE_ERR_POS_LOW0	48h
SINGLE_ERR_POS_LOW1	5Ch
SINGLE_ERR_POS_LOW2	70h
SINGLE_ERR_POS_LOW3	84h

Function

When single bit error is corrected in OCRAM bank0 read access, this register indicates which bit is corrected in the LOW 32 bits of read data.

Diagram



Fields

Field	Function
31-0 SINGLE_ERR_POS	LOW Single Error Bit Position On OCRAM Bank0 When single bit error is corrected in OCRAM bank0 read access, this field indicates which bit is corrected in LOW 32 bits of read data.

41.7.1.12 HIGH Single Error Bit Position On OCRAM Banka (SINGLE_ERR_POS_HIGH0 - SINGLE_ERR_POS_HIGH3)

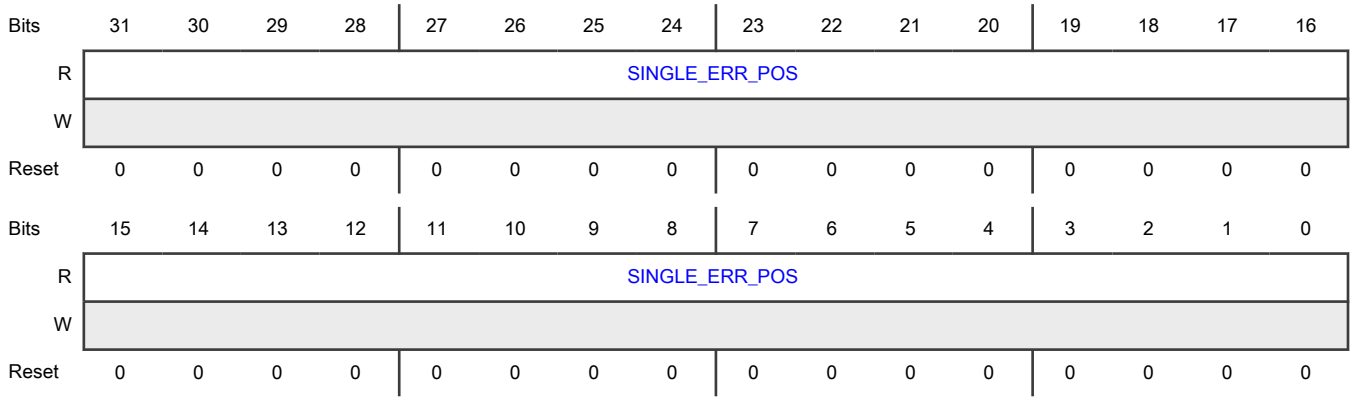
Offset

Register	Offset
SINGLE_ERR_POS_HIGH0	4Ch
SINGLE_ERR_POS_HIGH1	60h
SINGLE_ERR_POS_HIGH2	74h
SINGLE_ERR_POS_HIGH3	88h

Function

When single bit error is corrected in OCRAM bank0 read access, this register indicates which bit is corrected in the HIGH 32 bits of read data.

Diagram



Fields

Field	Function
31-0 SINGLE_ERR_POS	HIGH Single Error Bit Position On OCRAM Bank0 When single bit error is corrected in OCRAM bank0 read access, this field indicates which bit is corrected in HIGH 32 bits of read data.

41.7.1.13 Multiple Error Address And ECC code On OCRAM Banka (MULTI_ERR_ADDR_ECC0 - MULTI_ERR_ADDR_ECC3)

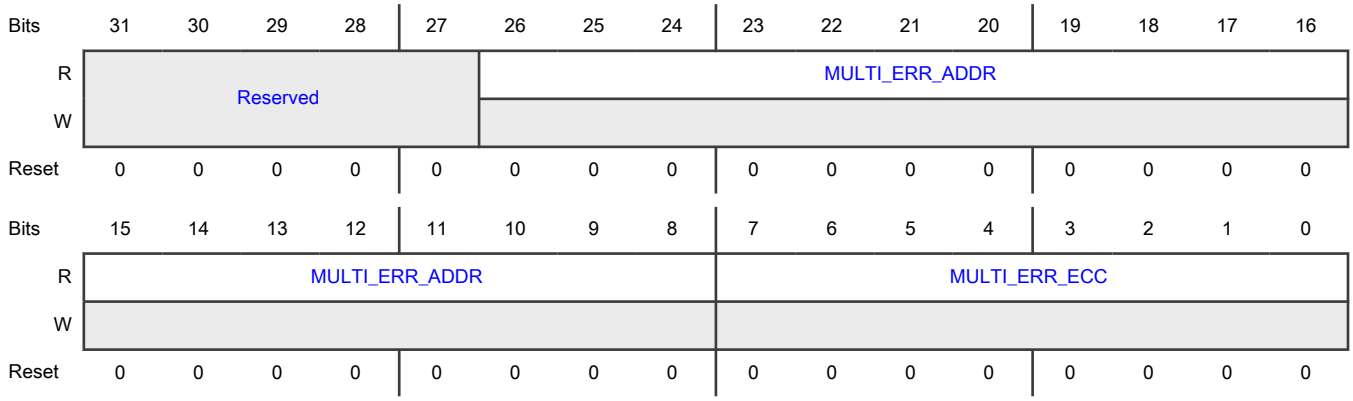
Offset

Register	Offset
MULTI_ERR_ADDR_ECC0	8Ch
MULTI_ERR_ADDR_ECC1	98h
MULTI_ERR_ADDR_ECC2	A4h
MULTI_ERR_ADDR_ECC3	B0h

Function

When multiple bits error are detected in OCRAM bank0 read access, this register stores OCRAM address and ECC code.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-8 MULTI_ERR_ADDR	Multiple Error Address On OCRAM Bank0 When multiple bits error are detected in OCRAM bank0 read access, this field stores OCRAM address. **NOTE** This field is not supported in every instance. The following table includes only supported registers.
7-0 MULTI_ERR_ECC	Multiple Error ECC code On OCRAM Bank0 When multiple bits error are detected in OCRAM bank0 read access, this field stores ECC code.

Instance	Field supported in	Field not supported in
MECC1	MULTI_ERR_ADDR_ECC0– MULTI_ERR_ADDR_ECC3	—
MECC2	MULTI_ERR_ADDR_ECC0– MULTI_ERR_ADDR_ECC3[25–8]	MULTI_ERR_ADDR_ECC0– MULTI_ERR_ADDR_ECC3[26]

41.7.1.14 LOW 32 Bits Multiple Error Read Data On OCRAM Banka (MULTI_ERR_DATA_LOW0 - MULTI_ERR_DATA_LOW3)

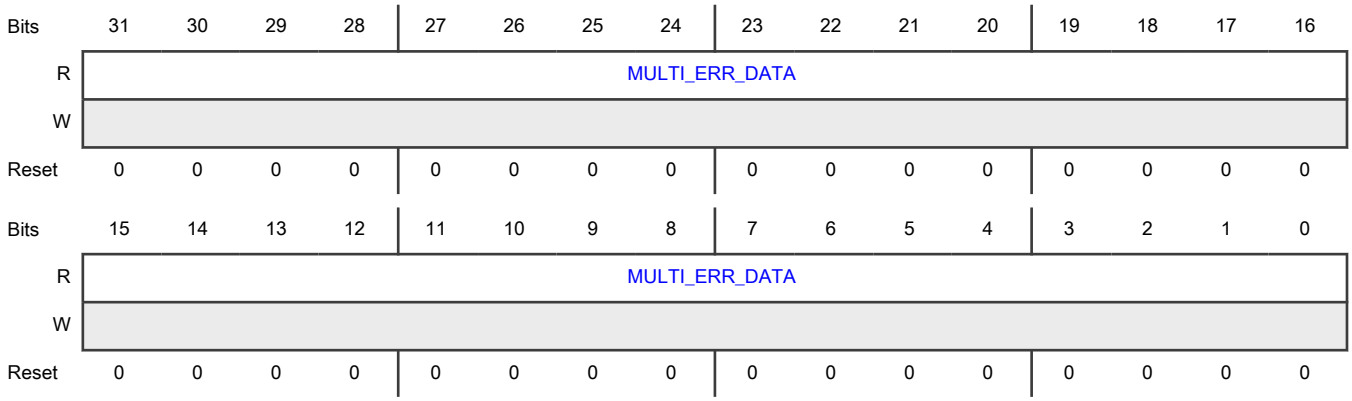
Offset

Register	Offset
MULTI_ERR_DATA_LOW0	90h
MULTI_ERR_DATA_LOW1	9Ch
MULTI_ERR_DATA_LOW2	A8h
MULTI_ERR_DATA_LOW3	B4h

Function

When multiple bits error are detected in OCRAM bank0 read access, this register stores LOW 32 bits read data.

Diagram



Fields

Field	Function
31-0 MULTI_ERR_DATA	LOW 32 Bits Multiple Error Read Data On OCRAM Bank0 When multiple bits error are detected in OCRAM bank0 read access, this field stores LOW 32 bits read data.

41.7.1.15 HIGH 32 Bits Multiple Error Read Data On OCRAM Banka (MULTI_ERR_DATA_HIGH0 - MULTI_ERR_DATA_HIGH3)

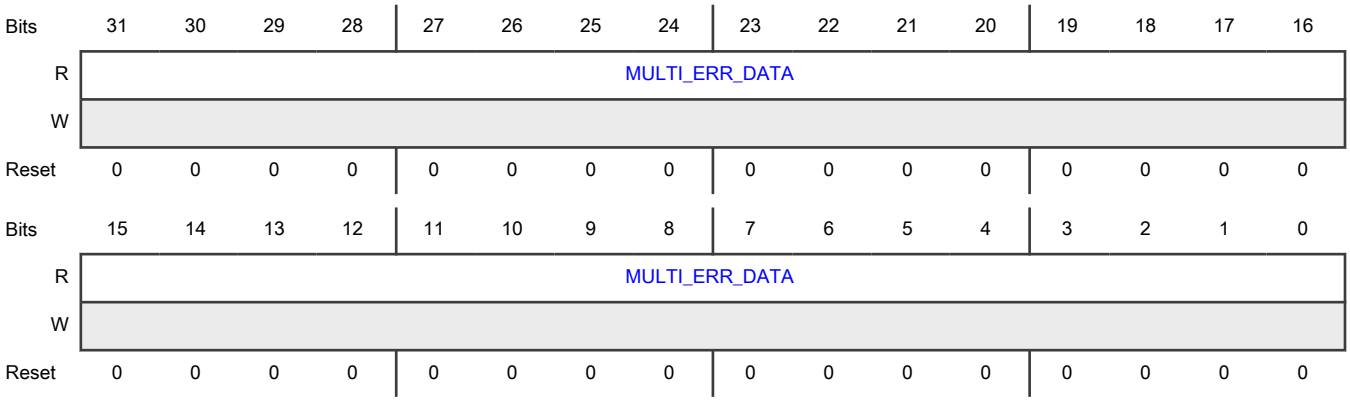
Offset

Register	Offset
MULTI_ERR_DATA_HIGH0	94h
MULTI_ERR_DATA_HIGH1	A0h
MULTI_ERR_DATA_HIGH2	ACh
MULTI_ERR_DATA_HIGH3	B8h

Function

When multiple bits error are detected in OCRAM bank0 read access, this register stores HIGH 32 bits read data.

Diagram



Fields

Field	Function
31-0 MULTI_ERR_DATA	HIGH 32 Bits Multiple Error Read Data On OCRAM Bank0 When multiple bits error are detected in OCRAM bank0 read access, this field stores HIGH 32 bits read data.

41.7.1.16 OCRAM Pipeline And ECC Enable (PIPE_ECC_EN)

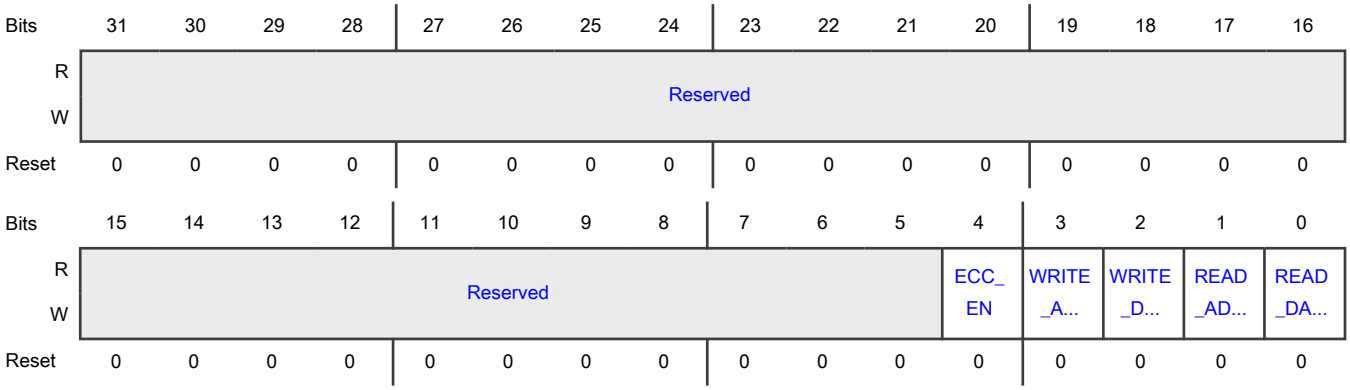
Offset

Register	Offset
PIPE_ECC_EN	100h

Function

OCRAM pipeline and ECC enable.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 ECC_EN	ECC Function Enable Enable ECC64 function for OCRAM access. This register bit reset value comes from IOMUX GPR. 0b - Disable. 1b - Enable.
3 WRITE_ADDR_PIPE_EN	Write Address Pipeline Enable Pipeline enable signal for write address path, when this signal is set to 1, the write address will be registered before can be applied to memory cell. 0b - Disable. 1b - Enable.
2 WRITE_DATA_PIPE_EN	Write Data Pipeline Enable Pipeline enable signal for write data path, when this signal is set to 1, the write data will be registered before can be applied to memory cell. 0b - Disable. 1b - Enable.
1 READ_ADDR_PIPE_EN	Read Address Pipeline Enable Pipeline enable signal for read address path, when this signal is set to 1, the read address will be registered before can be applied to memory cell. 0b - Disable. 1b - Enable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 READ_DATA_WAIT_EN	Read Data Wait Enable 1-cycle wait state in read access enable signal, when this signal is set to 1, 1-cycle wait state will be inserted into each read access. 0b - Disable. 1b - Enable.

41.7.1.17 Pending Status (PENDING_STAT)

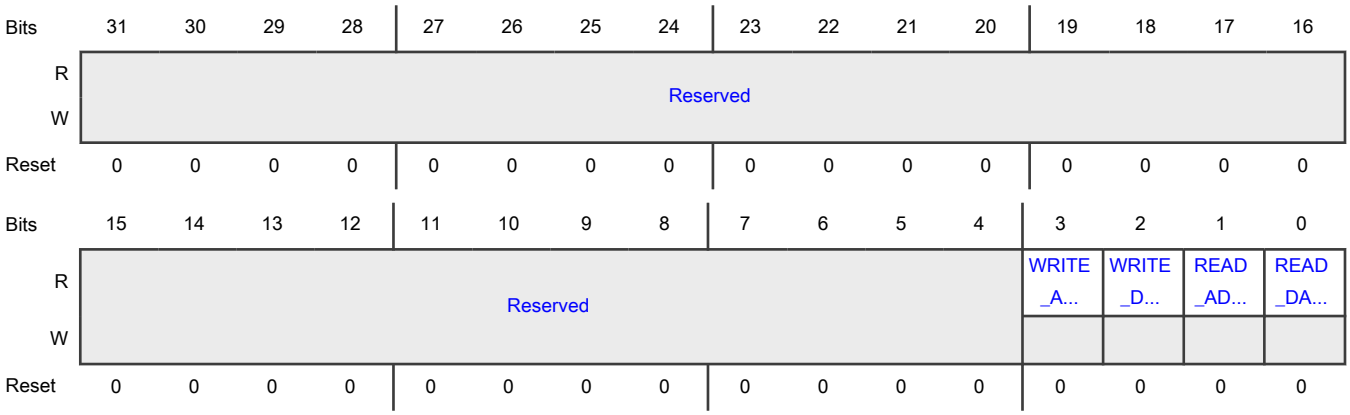
Offset

Register	Offset
PENDING_STAT	104h

Function

Update pending signal for OCRAM wait and pipeline enable.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 WRITE_ADDR_PIPE_PENDING	Write Address Pipeline Pending Update pending status for WRITE_ADDR_PIPE_EN. 0b - No update pending status for WRITE_ADDR_PIPE_EN.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - When WRITE_ADDR_PIPE_EN register bit is changed, this register bit will be set until the new setup becomes valid in the controller.
2 WRITE_DATA_PIPE_PENDING	Write Data Pipeline Pending Update pending status for WRITE_DATA_PIPE_EN. 0b - No update pending status for WRITE_DATA_PIPE_EN. 1b - When WRITE_DATA_PIPE_EN register bit is changed, this register bit will be set until the new setup becomes valid in the controller.
1 READ_ADDR_PIPE_PENDING	Read Address Pipeline Pending Update pending status for READ_ADDR_PIPE_EN. 0b - No update pending status for READ_ADDR_PIPE_EN. 1b - When READ_ADDR_PIPE_EN register bit is changed, this register bit will be set until the new setup becomes valid in the controller.
0 READ_DATA_WAIT_PENDING	Read Data Wait Pending Update pending status for READ_DATA_WAIT_EN. 0b - No update pending status for READ_DATA_WAIT_EN. 1b - When READ_DATA_WAIT_EN register bit is changed, this register bit will be set until the new setup becomes valid in the controller.

Chapter 42
Semaphores2 (SEMA42)

42.1 Chip-specific SEMA42 Information

Table 296. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

42.2 Overview

SEMA42 is a memory-mapped module that provides robust hardware support needed in multi-core systems for implementing semaphores and provides a simple mechanism to achieve "lock and unlock" operations via a single - write access. The hardware semaphore module provides hardware-enforced gates as well as other useful system functions related to the gating mechanisms.

42.2.1 Block diagram

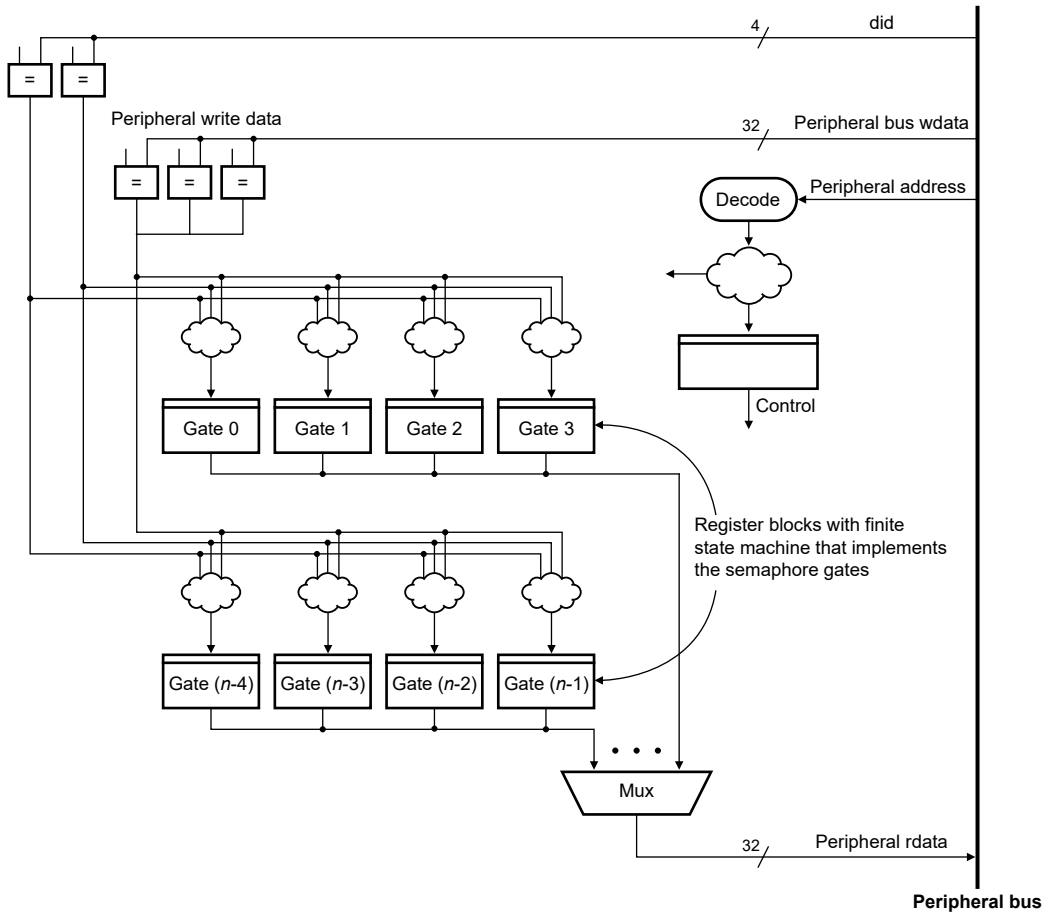


Figure 284. SEMA42 block diagram

42.2.2 Features

SEMA42 implements hardware-enforced semaphores as an IPS-mapped slave peripheral device. The feature set includes:

- Support for 64 hardware-enforced gates in a multi-domain configuration that supports up to 15 domains.
- Each hardware gate appears as a 16-state, 4-bit state machine.
- Support for secure reset mechanisms to clear the contents of individual gates, as well as a clear-all capability.
- Memory-mapped slave peripheral that offers programming-model accesses.

42.3 Functional description

The intent of the hardware gate implementation is to specify a protocol where the locking domain must unlock the gate. However, some systems may require a reset function to re-initialize the state of any gate(s) without requiring a system-level reset. To support this special gate reset requirement, SEMA42 implements a secure reset mechanism that allows you to initialize a hardware gate (or all the gates) by following a specific dual-write access pattern. The secure-gate reset:

- Uses a technique similar to that required for the servicing of a software watchdog timer
- Requires two consecutive writes with pre-defined data patterns from the same domain

You must do this to force the clearing of the specified gate(s). The required access pattern as follows:

1. A domain performs a 16-bit write to the RSTGT memory location. The most significant byte (RSTGT_W[RSTGDP]) must be E2h. The value of least significant byte is irrelevant for this reference and can be anything.
2. The same domain then performs a second 16-bit write to the RSTGT location. For this write, the upper byte (RSTGT_W[RSTGDP]) is the logical complement of the first data pattern (1Dh) and the lower byte (RSTGT_W[RSTGTN]) specifies the gate(s) to be reset. This gate field can specify a single gate or all gates to be cleared. If the same domain writes incorrect data on the second access or another domain performs the second write access, SEMA42 aborts the special gate reset sequence and does not assert an error signal.
3. Reads of the RSTGT location return information on the 2-bit reset state machine (RSTGT_R[RSTGSM]) that implements these functions:
   - The domain performing the reset (RSTGT_R[RSTGMS])
   - The last-cleared gate number(s) (RSTGT_R[RSTGTN])

   Reads of the RSTGT register do not affect the secure-reset finite state machine in any manner.

42.3.1 Multi-core programming: software gates

Multi-processor systems require a function that can be used to safely and easily provide a locking mechanism for system software to control access to shared data structures, shared hardware resources, and so on. The software uses the gating mechanisms to serialize (and synchronize) accesses to shared data and/or resources to prevent race conditions and preserve memory coherency between different processes and domains.

Consider the following description of a typical use-case: domain*X* enters a section of code, where shared data values are to be updated. The domain must first acquire a semaphore. Think of this as the locking (or closing) of a software gate. After the gate locks, a properly-architected software system does not allow other processes (or domains) to execute the same code segment or modify the shared data structure protected by the gate. In other words, the system locks out other processes/domains. Many software implementations include a spin-wait loop within the lock function until the gate locks. After domain X obtains the lock, domain X continues execution and updates the data values protected by the particular lock. After domain X completes the updates, it unlocks (or opens) the software gate, allowing other processes/domains access to the updated data values.

A correctly-implemented system solution must follow these important rules:

- A gate variable must protect all writes to shared data values or shared hardware resources.
- After a domain locks a gate, the system must block other processes/domains from accessing the shared data or resources. Software conventions enforce this.
- The domain that locks a particular gate is the only domain that can open (unlock) that gate.

Information in the hardware gate identifying the locking domain can be extremely useful for system-level debugging.

42.3.2 64 Hardware-enforced gates

Gates appear as a 64-entry byte-size array with read and write accesses.

Domains lock gates by writing "domainID_number+1" to the appropriate gate and must read back the gate value to verify that the lock operation succeeded.

After the gate locks, the locking domain unlocks the gate by writing zeroes.

- 16-state implementation

  If gate = 0h, then state = unlocked

  if gate = 1h, then state = locked by domain (master) 0

  if gate = 2h, then state = locked by domain (master) 1

  …

  if gate = Fh, then state = locked by domain (master) 14

SEMA42 uses the logical domain number and the specified data patterns as reference attributes to validate all write operation.

After a gate locks, the locking domain must unlock the gate by writing zeroes.

42.3.3 State machine of the GATE*n* registers

This section describes more about the SEMA42 functional operation and specific details of the state machines of the GATE*n* registers.

As described previously, each of the GATE*n* registers implements a 4-bit, 16-state machine. The following figure shows a simplified diagram of the state transitions for each gate.

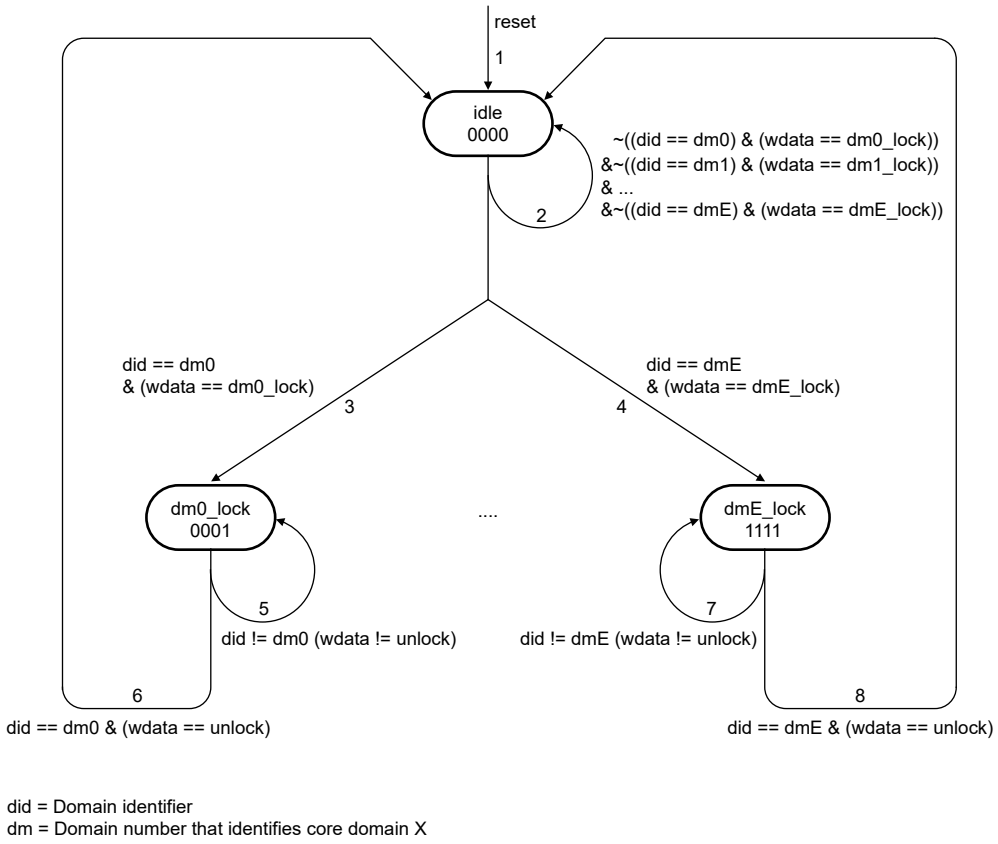


Figure 285. GATE*n* state machine

In the figure above, "dmE" represents domain 14 (E in hexadecimal). The platform passes the domain number to SEMA42.

The following table defines the GATE*n* state transitions.

Table 297. GATE*n* state transitions

Current state	Next state	Transition	Description
–	idle	1	Any reset, whether a system reset or a software-initiated gate reset, unconditionally forces the gate into the idle state.
idle	idle	2	Unless a write of the appropriate lock value from the corresponding domain occurs, the gate remains in the idle state.
idle	dm0_lock	3	When domain 0h initiates a write of the dm0_lock data value, the gate transitions into the dm0_lock state.

Table continues on the next page...

Table 297. GATE*n* state transitions (continued)

Current state	Next state	Transition	Description
idle	dmE_lock	4	When domain Eh initiates a write of the dmE_lock value, the gate transitions into the dmE_lock state.
dm0_lock	dm0_lock	5	When in this state, the gate remains here if any attempted write is not from domain 0h with the unlock data value.
dm0_lock	idle	6	The gate returns to the idle (unlocked) state after a write from domain 0h with the unlock data value occurs.
dmE_lock	dmE_lock	7	When in this state, the gate remains here if any attempted write is not from domain Eh with the unlock data value.
dmE_lock	idle	8	The gate returns to the idle (unlocked) state after a write from domain Eh with the unlock data value occurs.

SEMA42 uses these gate data values:

- The lock data value is (domain number) + 1.
- The unlock data value is 00h.

42.3.4 Clocking

This module has no clocking considerations.

42.3.5 Interrupts

This module has no interrupts.

42.4 Memory map/register definition

You can access these registers only in Supervisor mode. User accesses terminate with an error.

42.4.1 SEMA42 register descriptions

42.4.1.1 SEMA42 memory map

SEMA1 base address: 4426_0000h

SEMA2 base address: 4245_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 3Fh	Gate (GATE0 - GATE63) [1]	8	RW	00h
42h	Reset Gate Read (RSTGT_R)	16	R	0000h
42h	Reset Gate Write (RSTGT_W)	16	W	See section

1. In this array, the index and offset values of the registers do not increment in direct alignment. For details, see the register description.

42.4.1.2 Gate (GATE0 - GATE63)

Offset

For n = 0 to 63:

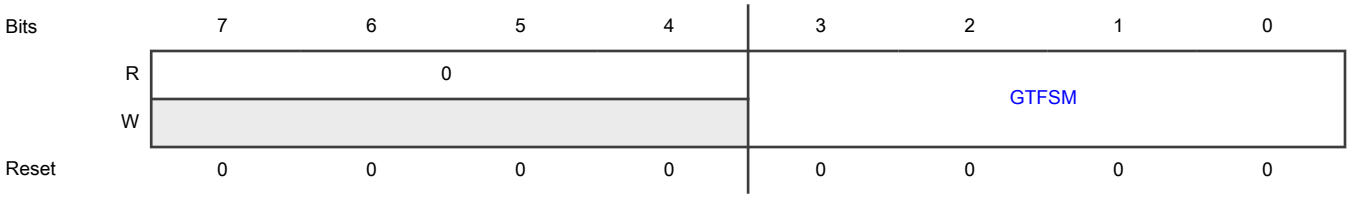
Register	Offset
GATEn	0h + (n + 3 - 2 × (n mod 4))

Function

Implements each semaphore gate in a 4-bit finite state machine, right-justified in a byte data structure. The hardware uses the logical domain-identifier number in conjunction with the data patterns to validate all attempted write operations. Only domain masters can modify the gate registers. After a gate locks, only the locking domain must open (unlock) the gate.

You can read multiple gate values in a single access. However, you can update only a single gate at a time, via a write operation. If you attempt to write a byte-wide value that is neither the unlock value (00h) nor the appropriate lock value (domainID_number + 1), SEMA42 considers this as "no operation" and does not change any gate state. Attempts to write multiple gates in a single-aligned access with a size larger than 8 bits (byte) generate an error termination and do not allow any gate state changes.

Diagram



Fields

Field	Function
7-4 —	Reserved
3-0 GTFSM	Gate Finite State Machine Indicates the state of the gate for the last domain that locked the gate. This can be useful during system debug. The hardware gate has a 16-state implementation, defined as: 0000b - The gate is unlocked (free). 0001b - Domain 0 locked the gate. 0010b - Domain 1 locked the gate. 0011b - Domain 2 locked the gate. 0100b - Domain 3 locked the gate. 0101b - Domain 4 locked the gate. 0110b - Domain 5 locked the gate.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0111b - Domain 6 locked the gate. 1000b - Domain 7 locked the gate. 1001b - Domain 8 locked the gate. 1010b - Domain 9 locked the gate. 1011b - Domain 10 locked the gate. 1100b - Domain 11 locked the gate. 1101b - Domain 12 locked the gate. 1110b - Domain 13 locked the gate. 1111b - Domain 14 locked the gate.

42.4.1.3 Reset Gate Read (RSTGT_R)

Offset

Register	Offset
RSTGT_R	42h

Function

Describes the specific hardware gate to be reset and records the logic number of domain. Reset Gate Write (RSTGT_W) also describe the same register showing the fields when you write it.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		RSTGSM		RSTGMS				RSTGTN							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 —	Reserved
13-12 RSTGSM	Reset Gate Finite State Machine

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates the encoded state machine value when you read the register. RSTGSM = 10b is valid for only a single machine cycle, so a read can never return this value. SEMA42 maintains the reset state machine in a 2-bit, 3-state implementation, defined as follows: 00b - Idle, waiting for the first data pattern write. 01b - Waiting for the second data pattern write 10b - The 2-write sequence has completed. Generate the specified gate reset(s). After the reset is performed, this machine returns to the idle (waiting for first data pattern write) state. 11b - This state encoding is never used and therefore reserved.
11-8 RSTGMS	Reset Gate Domain Records the logical number of the domain performing the gate reset function. To succeed, this function requires that the same domain initiate the two consecutive writes to this register. SEMA42 updates the field each time a write to this register occurs.
7-0 RSTGTN	Reset Gate Number Specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

42.4.1.4 Reset Gate Write (RSTGT_W)

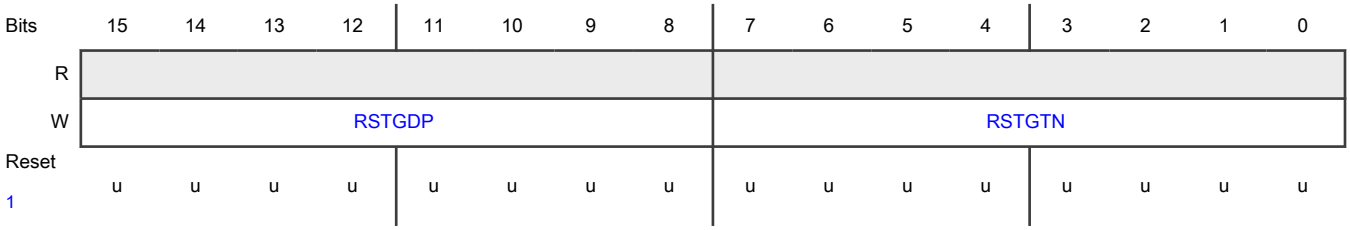
Offset

Register	Offset
RSTGT_W	42h

Function

Specifies the hardware gate to reset when data patterns are specified.

Diagram



1. Reset value is not applicable for writes.

Fields

Field	Function
15-8 RSTGDP	Reset Gate Data Pattern You access this field with the specified data patterns on the two consecutive writes to enable the gate-reset mechanism. For the first write, RSTGDP must be E2h. For the second write, RSTGDP must be 1Dh.
7-0 RSTGTN	Reset Gate Number Specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

42.5 External signals

This module has no external signals.

42.6 Initialization

This module does not require initialization.

Chapter 43
Trusted Resource Domain Controller (TRDC)

43.1 Chip-specific TRDC information

Table 298. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

This device has three instances of the TRDC module:

- TRDC1 (AON Domain)
- TRDC2 (WAKEUP Domain)
- TRDC3 (MEGA Domain)

The following table shows the TRDC1 MDAC Configuration:

Table 299. TRDC1 MDAC Configuration

MDAC	MASTER	MDAC_REG_NUM	MDAC_CPU	DEFAULT_DID	MDAC_INST_NUM	MDAC_INCL_PID
MDAC_A1	CM33_I	3	1	2	1	1
	CM33_S					
MDAC_A2	eDMA1	1	0	4	2	0

The following table shows the TRDC2 MDAC Configuration:

Table 300. TRDC2 MDAC Configuration

MDAC	MASTER	MDAC_REG_NUM	MDAC_CPU	DEFAULT_DID	MDAC_INST_NUM	MDAC_INCL_PID
MDAC_W0	CM7 AHBP	2	1	4	0	0
MDAC_W1	CM7 AXI	3	1	4	1	0
MDAC_W2	DAP AHB_AP_SYS	1	0	9	2	0
MDAC_W3	CoreSight ETR	1	0	8	3	0
MDAC_W4	eDMA2	1	0	7	4	0
MDAC_W5	NETC ACE-Lite	1	0	10	5	0

The following table shows the TRDC3 MDAC Configuration:

Table 301. TRDC3 MDAC Configuration

MDAC	MASTER	MDAC_REG_NUM	MDAC_CPU	DEFAULT_DID	MDAC_INST_NUM	MDAC_INCL_PID
MDAC_M0	uSDHC1	1	0	5	0	0
MDAC_M1	uSDHC2	1	0	6	1	0
MDAC_M3	USB	1	0	11	3	0
MDAC_M4	FlexSPI_FLR	1	0	10	4	0

The following table shows the TRDC1 MBC Configuration:

Table 302. TRDC1 MBC Configuration

MBC	Memory	Memory slave index	Block Size	Number of Blocks
MBC_A0	AIPS1	0	64 KB	128
	Edgelock	1	64 KB	8
	GPIO1	2	64 KB	1
MBC_A1	CM33 Code-TCM	0	4 KB	32
	CM33 System-TCM	1	4 KB	32

NOTE

MDACs other than those listed in the tables above might have registers shown in the memory map (ex: TRDC1_PID0 and TRDC1_DACFG0). These registers should be treated as reserved.

The following table shows the TRDC2 MBC Configuration:

Table 303. TRDC2 MBC Configuration

MBC	Memory	Memory slave index	Block Size	Number of Blocks
MBC_W0	AIPS2	0	64 KB	128
	GPIO2, GPIO4, GPIO6	1	64 KB	6
	GPIO3, GPIO5	2	64 KB	5
	DAP (Debug)	3	1 MB	17
MBC_W1	AIPS3	0	64 KB	128
	AHB_ISPAP	1	64 KB	1
	NIC_MAIN GPV	2	1 MB	1
	SRAMC	3	512 KB	1

The following table shows the TRDC1 MRC Configuration:

Table 304. TRDC1 MRC Configuration

MRC Number	Memory	Number of Regions
MRC_A0	CM33 ROM	8
MRC_A1	FlexSPI2	8

The following table shows the TRDC2 MRC Configuration:

Table 305. TRDC2 MRC Configuration

MRC Number	Memory	Number of Regions
MRC_W1	FlexSPI1	8
MRC_W2	CM7 I-TCM, D-TCM	16
MRC_W3	OCRAM1	16
MRC_W4	OCRAM2	16
MRC_W5	SEMC	16
MRC_W6	NETC	16

NOTE

DACFG0 register is for non-existent master and should be ignored.

43.2 CM7 PID generation

CM7_MCM's PID register is used to generate the input PID to the TRDC for CM7 accesses. TRDC only uses 6-bits of the PID.

43.3 Overview

TRDC provides an integrated and scalable architectural framework for access control, system memory protection, and peripheral isolation. It allows software to assign chip resources including processor cores, non-core bus masters, memory regions, and slave peripherals to processing domains to support enforcement of robust operational environments.

- First, each bus mastering resource is assigned to a domain identifier (domainID, DID). Typically, each processor is assigned to a unique domainID and all remaining non-processor bus masters assigned to a different domainID.

- Next, the access control policies for the individual domains are programmed into any number of registers implemented in the slave memory block and region checkers.
- Finally, all accesses throughout the device are monitored concurrently to determine the validity of each access. If a reference from a given domain has sufficient access rights, it is allowed to continue, else the access is aborted and error information captured.

The access control scheme that TRDC defines supports a 4-level model, combining the traditional privileged (also known as supervisor) and user modes with an additional signal defining the secure, nonsecure attributes of each memory reference. The result is a 4-level hierarchical access control mechanism with different access control policies based on read, write, and execute references, where:

SecurePriv > SecureUser > NonsecurePriv > NonsecureUser

Combined with the secure/nonsecure and privileged/user attributes, a domainID is associated with every system bus transaction and forms the hardware basis for the implementation of TRDC's access control mechanisms.

43.3.1 Block diagram

As previously noted, the TRDC implementation is distributed across multiple submodules instantiated throughout the device. The TRDC submodules include:

- TRDC_MGR
  - The Manager (MGR) submodule coordinates all programming model reads and writes.
- TRDC_DAC
  - The Domain Assignment Controller (DAC) submodule handles resource assignments and generation of the domain identifiers.
- TRDC_MBC
  - The Memory Block Checker (MBC) submodule implements access controls for on-chip internal memories and slave peripherals based on a fixed-sized block format.
- TRDC_MRC
  - The Memory Region Checker (MRC) submodule implements the access controls for external off-chip memories and peripherals based on the pre-programmed region descriptor registers.

See Figure 286 for a simplified TRDC block diagram focusing on topology and connections.

NOTE

TRDC typically consists of various submodules like DAC, MGR, MBC, and MRC. However, all the submodules as shown in the block diagram below may not be present for this configuration. For chip-specific implementation details of this module's instances, see the chip-specific information.

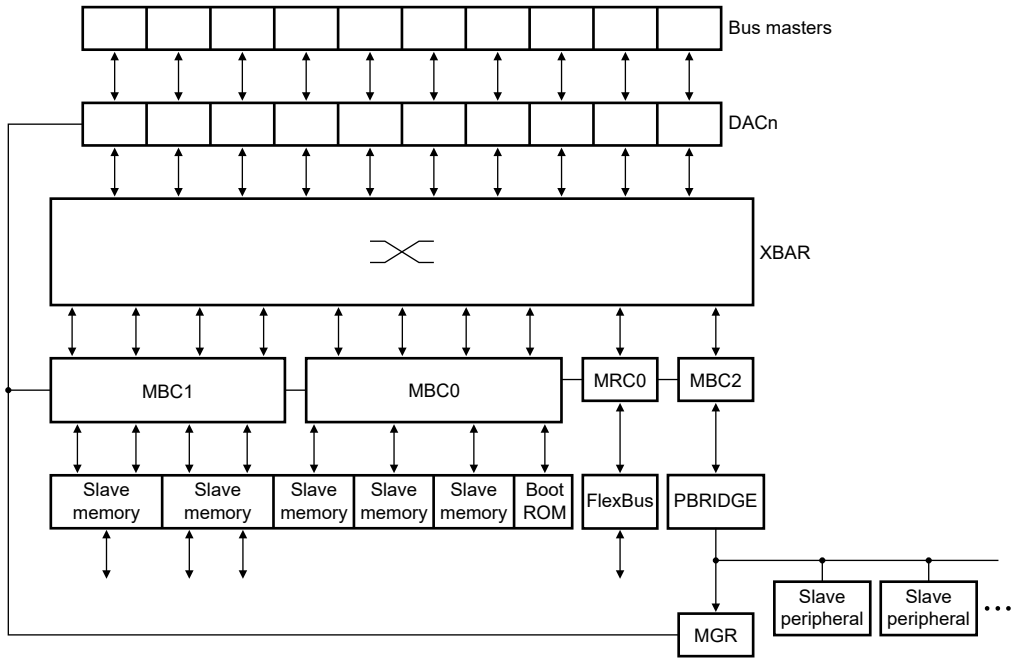


Figure 286. Block diagram

43.3.2 Features

- Ability to assign chip resources to processing "domains":
  - Processor cores, non-core bus masters, slave memories, and slave peripherals
  - Each processing domain is assigned a unique domain identifier (domainID, DID)
  - DomainID is an attribute associated with every system bus transaction
  - Used in conjunction with secure/nonsecure, privileged/user attributes
- Defines access rights to slave targets in MBCs for on-chip memories and peripherals and in MRCs for external memories and peripherals
- Built upon a 4-level hierarchical access control model:
  - SecurePriv > SecureUser > NonsecurePriv > NonsecureUser
  - Defined by multiple sets of user-programmable R/W/X (Read, Write, Execute) flags per access state
- Distributes programming model and hardware implementation across multiple submodules:
  - Supports a broad, highly-configurable architecture definition
  - Supports efficient mechanisms for memory space context switch state changes using nonsecure enables (NSE) for memory blocks and regions

43.4 Functional description

This section provides more details on the operation and implementation of the TRDC submodules.

43.4.1 Modes of operation

TRDC does not support any special modes of operation. As a memory-mapped device located in the core platform's clock domain, it responds strictly on the basis of memory addresses of the connected system buses. TRDC evaluates domain assignment

and access control functions on a reference-by-reference basis using the addresses and attributes connected to the system bus port(s).

43.4.2 Manager (TRDC_MGR)

The MGR submodule is responsible for coordinating TRDC's programming model reads and writes.

The complete programming model is large. Additionally, the programming model is distributed across the various submodule instances (DAC, MBC, MRC).

MGR provides:

- The required hardware management for all programming model accesses.
- Generating and distributing the appropriate address decodes and write data to the DAC, MBC, and MRC submodule instances.
- Collecting and combining all the register read data buses and responses.

MGR implements:

- The Control Register
- All the read-only hardware configuration registers (HWCFG[0-1], DACFGn, MBCm_MEMs_GLBCFG, MRC_GLBCFG)
- The program-visible versions of the domain error capture registers.

To support the domain error reporting functionality, MGR collects the individual captured error indicator output signals from all the submodule instances and remaps them so that they are organized into an instance bitmap by domain that the DERRLOCn register array specifies.

43.4.3 Domain Assignment Controller (TRDC_DAC)

The DAC submodule is responsible for the generation of the domainID on every memory transaction for every bus master in the device. The resulting domainID, and {nonsecure, privileged} signals are generated and then treated as address attributes and associated with each transaction as it moves through the system.

DAC modules that drive the domainIDs (DIDs) for processor cores include an option of processor-identifier (PID) matching to select between DIDs. You can accomplish this by first enabling PID checking by writing TRDC_MDA_Wm_n[PE] = [2,3]. With PID matching enabled, a domain hit is determined by matching TRDC_MDA_Wm_n[PID] with the PID input or PID in the TRDC_PID register. The PIDM field can be used to match specific groups of PIDs.

As a simple example, assume a system with two domainIDs.

- domainID 0 is for critical tasks.
- domainID 1 is for non-critical tasks.
- Two domainIDs require two MDA_Wm_n registers, MDA_W0 and MDA_W1.

Further assume the processor’s task identifier defines whether a task is critical or non-critical as defined by system software.

- PIDs [0-15] are assigned to critical tasks.
- PIDs != [0-15] are assigned to noncritical tasks.

During startup, software initializes the MDA_W[0-1] registers. MDA_W0 defines DID = 1 for PIDs [0-15] and MDA_W1 defines DID = 2 for PIDs != [0-15].

1. MDA_W0 = 0x8000_0F81
    - VLD = 1
    - PID = 0x0
    - PIDM = 0xF
    - PE = 2

- DIDS = 0
- DID = 1

2. MDA_W1 = 0x8000_0FC2
   - VLD = 1
   - PID = 0x0
   - PIDM = 0xF
   - PE = 3
   - DIDS = 0
   - DID = 2

As the processor executes, it loads the appropriate task ID into its PID register as the task is started. The processor's PID register value is input to the corresponding DAC module and used by the MDA_Wr_m comparison logic. The system then dynamically generates two domainIDs and the downstream TRDC access check logic can distinguish and enforce different access control rights based on the different domainIDs.

A special mode is provided for driving the DID of non-processor masters. This is enabled with the DID bypass (DIDB) bit. The DAC has a DID input that can be driven by a master and used directly as the DID output. This allows variability in the DID as dictated by the master and is intended to be used by DMA masters that masquerade as, (that is, drive the DID of) the master that programmed it.

43.4.4 Access evaluation

Fundamental to the TRDC's operation is the actual access violation check performed by the memory block checker (MBC) and memory region (MRC).

Previous descriptions have detailed the domain assignment mechanisms and the tracking of the domain identifier as an address attribute through the system bus switching fabric(s). As transactions reach the slave memory and peripheral controllers, the address is used to select the appropriate memory region descriptor or block configuration register. Once the appropriate register has been selected, the next function is the actual access evaluation.

For this step, the 3-bit M{B,R}ACSEL field of the selected RGD or BLK_CFG register selects the appropriate GLBAC (global access control policy). The GLBAC register defines the access rights for the domain based on the 4-level model combined with the transaction's access attributes (read = 0/write = 1, secure = 0/nonsecure = 1, execute =1/non-execute=0, user = 0/privileged = 1).

The 4-level model is defined by the concatenation of two address attributes: secure/nonsecure and user/privileged attributes. These two attribute signals (nonsecure, privileged) and the resulting access levels are defined as:

```
if {nonsecure, privileged} == 0b00, then level = SecureUser
if {nonsecure, privileged} == 0b01, then level = SecurePriv(ileged)
if {nonsecure, privileged} == 0b10, then level = NonsecureUser
if {nonsecure, privileged} == 0b11, then level = NonsecurePriv(ileged)
```

The resulting hierarchical access control model specifies:

```
SecurePriv , SecureUser , NonsecurePriv , NonsecureUser
```

After the access evaluation is complete and the access error signal generated, the transaction is allowed to continue if it has sufficient access rights, else the access is aborted with the address and attribute information captured in the appropriate error registers.

43.4.5 Memory Block Checker (TRDC_MBC)

The TRDC's Memory Block Checker provides domain-based access control for all system bus references targeted to on-chip internal memories and slave peripherals. Using programmed block configuration registers which define the access rights per domain and block, the MBC concurrently monitors system bus transactions and evaluates the appropriateness of each transfer.

Memory references that have sufficient access rights are allowed to complete, while references that have insufficient rights are terminated with an access error response.

The TRDC architectural framework supports up to 8 MBC instances where each MBC can support up to 4 sub-regions (MEM[s]). Note monitoring of an AXI memory counts as 2 sub-regions since there are independent read and write channels per AXI memory. Conversely, an AHB connection counts as a single subregion.

Each sub-memory is divided up in equal sized consecutive blocks. Sub-memory 0 (MEM0) can support up to 512 blocks while sub-memory 1-3 (MEM1-MEM3) can each support up to 64 blocks. The number of blocks and size of the block is defined in MBCm_MEMs_GLBCFG[NBLKS,SIZE_LOG2] register fields, where m is the MBC instance number, and s is the sub-memory number. For example, Sub-memory 0 (MEM0) may have 16 consecutive blocks, each of size 64 KB, and corresponding MBCm_DOMd_BLK_CFG_W register to a block will control access to that block.

Each MBCm_DOMd_BLK_CFG_W contains eight access control structures. Each structure is comprised of a 1-bit NSE (NonSecure Enable) plus a 3-bit MBACSEL (Memory Block Access Control Select). The NSE controls secure/nonsecure access to the corresponding block, while the MBACSEL field selects a MBCm_MEMN_GLBACr register which controls the read/write/execute access to the corresponding block. There are 8 programmable MBCm_MEMn_GLBACr registers per block checker, each containing contain read-write-execute control flags for each of the four operating modes:

```
SPR, SPW, SPX - Secure    Privileged {Read, Write, Execute}
SUR, SUW, SUX - Secure    User       {Read, Write, Execute}
NPR, NPW, NPX - Nonsecure Privileged {Read, Write, Execute}
NUR, NUW, NUX - Nonsecure User       {Read, Write, Execute}
```

GLBAC1-7 also have a lock bit MBCm_MEMn_GLBACr[LK]. When LK=1, the GLBACr register is read-only until the next reset. Furthermore, if MBCm_DOMd_BLK_CFG_W[MBACSEL] selects a GLBAC that is locked, the 3-bit MBCACSEL field is also locked until the next reset. GLBAC0 cannot be locked.

The MBC also has MBCm_DOMd_MEMs_BLK_NSE_W registers. These registers are a bit map of the block NSE bits. Each word supports up to 32 blocks.

```
MBCm_DOMd_MEM0_BLK_NSE_W[0-15] has a maximum of 16 words to support 512 blocks
MBCm_DOMd_MEM[1-3]_BLK_NSE_W[0-1] has a maximum of 2 words to support 64 blocks.
```

The block NSE bits can be programmed by writing these registers or by writing the MBCm_MEMN_BLK_{INDEX, SET, CLR, CLR_ALL} registers.

The MBCm_MEMn_NSE_BLK_{INDEX, SET, CLR, CLR_ALL} registers provide a mechanism to efficiently and quickly manipulate the NSE bitmap with bitmasks to set or clear individual blocks within the bitmap and a single operation to perform a clear all across multiple DIDs. The BLK_INDEX register provides the control definition for the BLK_SET and BLK_CLR operations.

43.4.5.1 Memory block hit determination

MBCm_MEMs_GLBCFG[NBLK, SIZE_LOG2] is used to determine which bits of the address correspond to block number.

```
block_n_hit = (addr[MSB:LSB] == n)
where LSB=SIZE_LOG2
MSB=f{NBLKS,LSB}
```

Note that the block hit determination is based only on the address comparison of the first byte being accessed; that is, the MBC does not check the size of the access to make sure it entirely fits within the region.

43.4.5.2 Memory block access evaluation

For a block n hit, the MBC logic evaluates the access rights defined by the MBCm_DOMd_BLK_CFG_Ww registers. The domainID attribute and block number hit selects the appropriate MBCm_DOMd_BLK_CFG_Ww register to use in the access evaluation. The {R,W,X}, nonsecure and privilege attributes of the access are compared with the MBC global access control policy defined in MBCm_DOMd_BLK_CFG_W[MBACSEL, NSE] fields.

While TRDC_CR[GVLDB] = 1, and for each sub-memory being monitored, the MBC performs a reduction-AND of all the individual error terms from each access evaluation.

Unlike the MRC access evaluation and termination, the MBC access evaluation terminates the access with a bus error and reports an access error for only one condition - when the access doesn't have sufficient access rights. By nature, the block checker can only hit in ONE block. And by implementation, there are no "miss" accesses. Accesses outside of the range covered by the MBC are not sent to the MBC. There also aren't any individual block valid bits. When TRDC_CR[GVLDB] = 1, checking on all blocks is always enabled.

43.4.6 Memory Region Checker (TRDC_MRC)

The TRDC's Memory Region Controller provides domain-based, hardware access control for all system bus references targeted at non-peripheral memory spaces. Using pre-programmed *region descriptors* which define memory spaces and their associated access rights per domain identifier, the MRC concurrently monitors system bus transactions and evaluates the appropriateness of each transfer. Memory references that have sufficient access control rights are allowed to complete, while references that are not mapped to any region descriptor or have insufficient rights are terminated with an access error response.

The TRDC architectural framework supports up to 16 MRC instances, where each MRC instance can support [1-16] region descriptors, concurrently monitoring up to 8 slave memory ports. Note, monitoring an AXI protocol port counts as two ports since there are independent read address and write address channels. Conversely, an AHB connection counts as a single port.

A partial, simplified one-dimensional block diagram of an instance of the MRC module is shown in Figure 287. In this figure, a single slave bus port is shown. When the remaining slave ports are considered, the MRC hardware's two-dimensional connection matrix, broadcasting each system bus slave port access across the implemented memory region descriptors, is the resulting structure with the basic access evaluation macro shown as the replicated submodule block. In Figure 287, the system bus fabric slave port is shown on the left, the memory region descriptor registers are in the middle, and the peripheral bus interface on the right side. The evaluation macro contains two magnitude comparators connected to the start and end address registers from each region descriptor as well as the combinational logic blocks to determine the region hit and, based on the domainID associated with the reference, access violations.

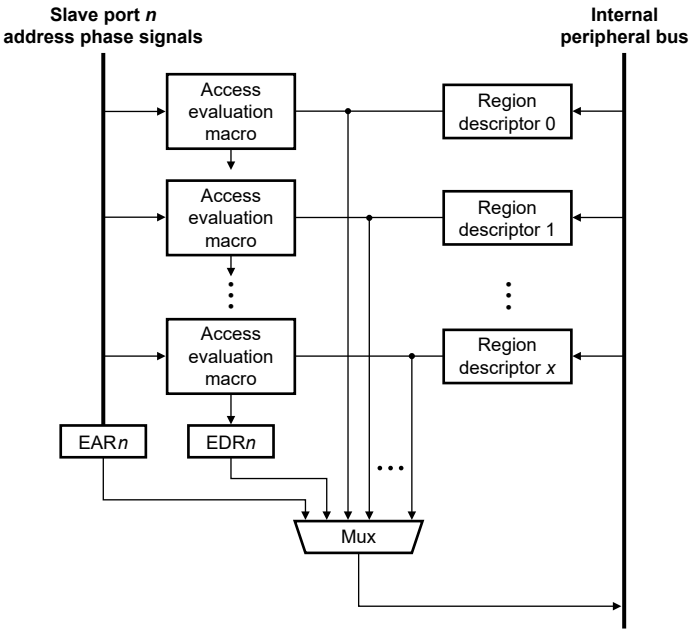


Figure 287. MRC access evaluation macro hardware structure

For each access, the MRC hardware performs a two-step evaluation process. First, the access is compared to all memory region descriptors to determine hit/miss status, and then, using the domainID associated with the reference plus the address attributes {write, nonsecure, privileged}, the access rights are examined using the method detailed in Access evaluation.

43.4.6.1 Memory region descriptor hit determination

To determine if the current reference hits in a given region, two magnitude comparators are used in conjunction with the region's start and end addresses. The boolean equation for this portion of the hit determination is defined as:

```
region_hit_n =
   ((addr >= start_addr) & (addr <= end_addr)) & valid
```

where `addr` is the current system reference address, `start_addr` and `end_addr` are the start and end addresses, and `valid` is the valid bit, all from memory region descriptor n.

Note there are *no hardware checks* to verify that `end_addr ≥ start_addr`, and it is software's responsibility to properly load appropriate values into these fields of the region descriptor.

Also, the region hit determination is based only on an address comparison of the first byte being accessed; that is, the MRC does *not* check the size of the access to make sure it entirely fits within the region.

43.4.6.2 Memory region access evaluation

For each region descriptor hit, the MRC logic evaluates the access rights defined by the MRCm_DOMd_RGDr_Ww registers. Specifically, the domainID attribute selects the appropriate MRCm_DOMd_RGDr_Ww register to use in the access evaluation. The {R,W,X}, nonsecure and privilege attributes of the access are compared with the MRC global access control policy defined in MRCm_DOMd_RGDr_W0[MRACSEL] and MRCm_DOMd_RGDr_W1[NSE] fields. See Access evaluation for additional details on this function.

While CR[GVLDR] = 1, and for each bus slave port being monitored, the MRC performs a **reduction-AND** of all the individual *(no_hit | error)* terms from each access evaluation macro. Recall as specified in Memory region descriptor hit determination, an *invalid* MRC_DOMd_RGDr_Ww (MRCm_DOMd_RGDr_W1[VLD] = 0) forces the region_hit_n evaluation to be negated.

This expression then terminates the bus cycle with an error and reports an access error for three conditions:

1. If the access does not hit in *any* region descriptor, an access error is reported.
2. If the access hits in a single region descriptor and that region signals a domain violation, then an access error is reported.
3. If the access hits in multiple (overlapping) regions and one region signals a violation, then an access error is reported.

The third condition reflects that priority is given to access denying over access allowing for overlapping regions.

Unimplemented domain IDs (DIDs) do not have any associated region descriptors and therefore have no access rights.

43.4.7 Interrupts

This module outputs an interrupt signal which can be connected to the system's interrupt controller. Please check chip-specific interrupt assignment for details. Interrupt is asserted on detection of access violation by any checker, and it remains asserted until DERRLOCn registers are cleared. This interrupt is in addition to interrupt generated by master after receiving bus error due to access violation by memory block checker or memory region checker. So, system may receive interrupt through different channels but from the same access violation at checker.

43.5 External signals

The TRDC module does not directly support any external interfaces.

The *internal* interfaces include a standard 32-bit slave bus for all programming model accesses, connections to the address phase signals associated with AHB and/or AXI system buses and connections to the slave peripheral buses as shown in TRDC block diagram.

43.6 Initialization

Out of reset, CR[GVLDR, GVLDB, GVLDM] bits are cleared, TRDC is disabled and in deny by default mode allowing secure privileged startup code to configure the entire programming model. In deny by default mode, only the DIDs that aren't denied by

default are allowed to access memory and peripherals. The allowed DIDs are implementation defined. The initialization process typically is performed in 3 steps:

1. Read the various hardware configuration registers (HWCFG{0,1}, DACFGn, MBCm_MEMs_GLBCFG, MRC_GLBCFG) to obtain the implemented hardware capabilities for the device.
2. Using the information retrieved in Step 1 coupled with the desired domain architecture, program all the register data structures for domain assignment (MDA_Wm_n), memory block configuration (MBCm_DOMd_MEMs_BLK_CFG_Ww) and memory region descriptors (MRCm_DOMd_RGDr_Ww). There are individual valid bits included in these registers which typically would be asserted. Additionally, there are also lock mechanisms available if register settings need to be configured and marked as read-only.
3. Once the TRDC programming model is loaded, the CR is written with GVLD {R, B, M} asserted. Upon the assertion of the 3 GVLD bits, deny by default mode is exited and at that point, the TRDC is fully operational. The TRDC remains enabled until the next reset.

43.7 Application information

As described in Overview, the TRDC architecture is intended to provide a broad, highly-capable framework for access control, system memory protection and peripheral isolation. The resulting microarchitecture implementation is distributed throughout the core platform, highly configurable via hardware design parameters and software programmability.

43.7.1 Master domain assignments

The typical use case related to master domain assignments is to include one (or more) processor core(s) in a single domain, possibly coupled with other bus master devices like DMA, etc. The definition of a domain may be static, based simply on the combination of a processor and other optional bus masters. Alternatively, the processor's operation may be configured to dynamically select between a very small number of domains. In particular, the optional inclusion of PID value(s) to be used to create multiple classes of CPU with different domain values.

For example, "critical" tasks, whether safety-critical, performance-critical, etc. can be grouped together in one domain and all other tasks into a second domain. Non-processor bus masters typically have a single MDA_Wm_n configuration register associated with them. The domainID and {nonsecure, privileged} attributes are usually statically assigned, unless the module can "inherit" attributes from a processor programming it (certain modules like DMAs).

The MDA_Wm_n registers, along with the memory region descriptors and the peripheral domain access control registers all have lock features that allow the register resources to be converted into read-only resources to protect the written configuration.

43.7.2 Memory region descriptor management

There are important concepts to consider in managing the memory region descriptors.

Recall that the association between each MRCn instance and the slave memory regions it monitors is SoC-dependent. The device-specific configuration specifies the number of implemented memory region descriptors in a given instance and the specific port numbers associated with the slave memories being monitored.

43.7.3 Domain error capture management

Recall when a domain access violation is detected by either a memory region checker or a memory block checker, address and attribute information of the offending access is captured. The DERRLOCn read-only registers provide additional information signaling the instance number of the submodule where the access violation(s) occurred. Since the resulting exception handler needs the submodule instance to retrieve the captured address and attribute information from DERR_W0_n and DERR_W1_n, the DERRLOCn registers provide the instance number details.

These registers are organized as a word array, *indexed by the faulting domain number*, with the contents of each register providing a bitmap signaling the instance number(s) associated with all submodules containing captured error information for that domain.

It is recommended that the exception handler begin by reading the HWCFG1 register to determine its domainID. Next, it uses the just-retrieved domainID to index into the DERRLOCn array. The resulting DERRLOCn value is then examined to determine the instance number of the reporting MBC and/or MRC submodule.

There may be multiple access violations, across multiple instances, pending for a given domain. It is suggested that exception handler use a "find first one" instruction (alternatively known as "count leading zeroes") to quickly and efficiently find the instance number containing access violation details. Once the instance number has been determined, the captured error address (DERR_W0_n) and attribute information (DERR_W1_n) can easily be retrieved from the domain error registers. The 2-bit DERR_W1_n[EST] field provides information signaling whether one or more errors have been detected by the submodule instance. A special write to DERR_W3_n is required to reset (and rearm) the EST error capture state machine.

This process is repeated until all errors associated with a given domainID have been processed.

A graphical representation of this 2-step retrieval of the domain error address and attributes is shown in Figure 288.

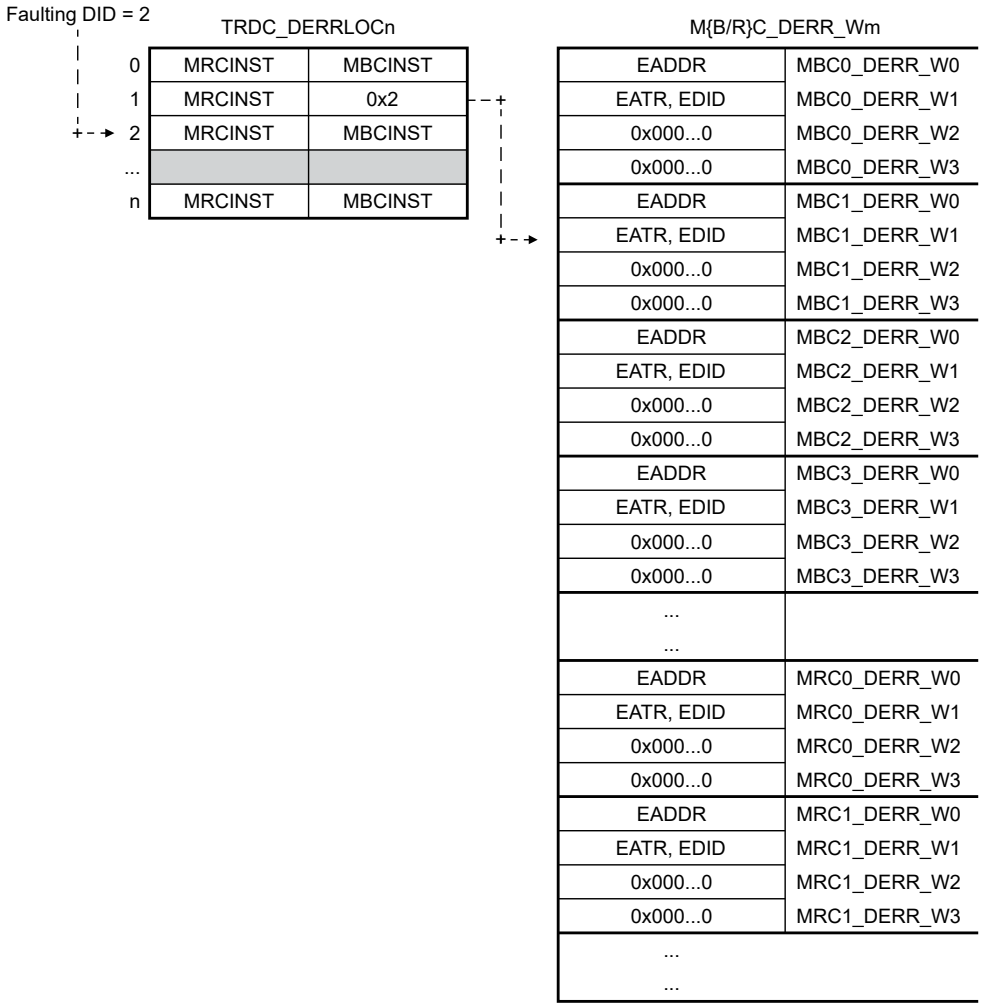


Figure 288. Two-step retrieval of domain error address and attributes

43.8 Register descriptions

The MBCm_NSE_BLK_SET, MBCm_NSE_BLK_CLR and MBCm_NSE_BLK_CLR_ALL registers exist at offset for only domain ID value 0 even when more than 1 domains exist, and address offsets for non-zero domain ID value should be considered reserved space. So, this reserved address space in MBC space does not return transfer error: 0xn14 to 0xn1C where n = (1 to NUM_DID-1) * 2 for each MBC.

If TZ-M is enabled, the TRDC programming model can only be accessed in SecurePrivileged mode. If TZ-M is disabled, it can only be accessed in Privileged mode regardless of Secure or NonSecure; that is it can't be accessed in User mode. Unless noted otherwise, the programming model registers can be accessed via 8-, 16- or 32-bit reads and 32-bit write references. Attempted

accesses in a different operating mode, using unsupported write data sizes, writes to read-only resources, or access to reserved spaces are terminated with an error unless noted otherwise. The TRDC programming model is partitioned into these groups of registers:

- Basic hardware control and configuration
- Domain errors: location and details
- Master domain assignments
- Memory block and region checkers

It should be noted that many of the programming model registers in TRDC are organized as 2-, 3- or 4- dimensional data structures. For the 2-dimensional structures, the generic arrays contain "m" words representing the "columns" and "n" instances of the structure representing the "rows". These may be described as `structure[n][m]`. They appear in the address space of the programming model memory map in the standard C language row-major layout, that is, the "m" words representing the "column" appear sequentially with the entire row replicated "n" times. The TRDC register structure uses the naming convention of *TRDC_<regname>_Wm_n*, where the column number "m" appears as a numeric suffix on the W (32-bit word) identifier, and the row identifier "n" appears as the final numerical suffix.

For the 4-dimensional structures, the generic arrays contain "m", "d", "s" or "r", and "w" indices corresponding to MBC/MRC instance, domain index, sub-memory or region index and word index. The naming convention for the MBC block configuration and MRC region descriptor word registers are: *MBCm_DOMd_MEMs_BLK_CFG_Ww* and *MRCm_DOMd_RGDr_Ww*

For the reset values in this section, a "*" indicates an initial value determined by the hardware configuration.

43.8.1 TRDC register descriptions

43.8.1.1 TRDC memory map

TRDC1 base address: 4427_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRDC Register (TRDC_CR)	32	RW	0000_0010h
F0h	TRDC Hardware Configuration Register 0 (TRDC_HWCFG0)	32	R	2202_0410h
F4h	TRDC Hardware Configuration Register 1 (TRDC_HWCFG1)	32	R	See section
F8h	TRDC Hardware Configuration Register 2 (TRDC_HWCFG2)	32	R	0000_0000h
FCh	TRDC Hardware Configuration Register 3 (TRDC_HWCFG3)	32	R	0000_0000h
100h - 103h	Domain Assignment Configuration Register (DACFG0 - DACFG3)	8	R	See section
1C0h	TRDC IDAU Control Register (TRDC_IDAU_CR)	32	RW	0000_0008h
1E0h	TRDC FLW Control (TRDC_FLW_CTL)	32	RW	0000_0000h
1E4h	TRDC FLW Physical Base (TRDC_FLW_PBASE)	32	R	0100_0000h
1E8h	TRDC FLW Array Base (TRDC_FLW_ABASE)	32	RW	0000_0000h
1ECh	TRDC FLW Block Count (TRDC_FLW_BCNT)	32	RW	0000_0000h
1FCh	TRDC Fault Domain ID (TRDC_FDID)	32	RW	0000_0000h
200h - 23Ch	TRDC Domain Error Location Register (TRDC_DERRLOC0 - TRDC_DERRLOC15)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
400h	MBC Domain Error Word0 Register (MBC0_DERR_W0)	32	R	0000_0000h
404h	MBC Domain Error Word1 Register (MBC0_DERR_W1)	32	R	0000_0000h
40Ch	MBC Domain Error Word3 Register (MBC0_DERR_W3)	32	RW	0000_0000h
410h	MBC Domain Error Word0 Register (MBC1_DERR_W0)	32	R	0000_0000h
414h	MBC Domain Error Word1 Register (MBC1_DERR_W1)	32	R	0000_0000h
41Ch	MBC Domain Error Word3 Register (MBC1_DERR_W3)	32	RW	0000_0000h
480h	MRC Domain Error Word0 Register (MRC0_DERR_W0)	32	R	0000_0000h
484h	MRC Domain Error Word1 Register (MRC0_DERR_W1)	32	R	0000_0000h
48Ch	MRC Domain Error Word3 Register (MRC0_DERR_W3)	32	RW	0000_0000h
490h	MRC Domain Error Word0 Register (MRC1_DERR_W0)	32	R	0000_0000h
494h	MRC Domain Error Word1 Register (MRC1_DERR_W1)	32	R	0000_0000h
49Ch	MRC Domain Error Word3 Register (MRC1_DERR_W3)	32	RW	0000_0000h
700h	Process Identifier (PID0)	32	RW	0000_0000h
704h	Process Identifier (PID1)	32	RW	0000_0000h
800h	DAC Master Domain Assignment Register (MDA_W0_0_DFMT0)	32	RW	0000_0000h
820h	DAC Master Domain Assignment Register (MDA_W0_1_DFMT0)	32	RW	0000_0000h
824h	DAC Master Domain Assignment Register (MDA_W1_1_DFMT0)	32	RW	0000_0000h
828h	DAC Master Domain Assignment Register (MDA_W2_1_DFMT0)	32	RW	0000_0000h
840h	DAC Master Domain Assignment Register (MDA_W0_2_DFMT1)	32	RW	2000_0000h
860h	DAC Master Domain Assignment Register (MDA_W0_3_DFMT1)	32	RW	2000_0000h
1_0000h	MBC Global Configuration Register (MBC0_MEM0_GLBCFG)	32	R	000C_0080h
1_0004h	MBC Global Configuration Register (MBC0_MEM1_GLBCFG)	32	R	000C_0008h
1_0008h	MBC Global Configuration Register (MBC0_MEM2_GLBCFG)	32	R	000C_0001h
1_000Ch	MBC Global Configuration Register (MBC0_MEM3_GLBCFG)	32	R	000C_0001h
1_0010h	MBC NonSecure Enable Block Index (MBC0_NSE_BLK_INDEX)	32	RW	0000_0000h
1_0014h	MBC NonSecure Enable Block Set (MBC0_NSE_BLK_SET)	32	W	0000_0000h
1_0018h	MBC NonSecure Enable Block Clear (MBC0_NSE_BLK_CLR)	32	W	0000_0000h
1_001Ch	MBC NonSecure Enable Block Clear All (MBC0_NSE_BLK_CLR_ALL)	32	RW	0000_0000h
1_0020h	MBC Global Access Control (MBC0_MEMN_GLBAC0)	32	RW	0000_0000h
1_0024h	MBC Global Access Control (MBC0_MEMN_GLBAC1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0028h	MBC Global Access Control (MBC0_MEMN_GLBAC2)	32	RW	0000_0000h
1_002Ch	MBC Global Access Control (MBC0_MEMN_GLBAC3)	32	RW	0000_0000h
1_0030h	MBC Global Access Control (MBC0_MEMN_GLBAC4)	32	RW	0000_0000h
1_0034h	MBC Global Access Control (MBC0_MEMN_GLBAC5)	32	RW	0000_0000h
1_0038h	MBC Global Access Control (MBC0_MEMN_GLBAC6)	32	RW	0000_0000h
1_003Ch	MBC Global Access Control (MBC0_MEMN_GLBAC7)	32	RW	0000_0000h
1_0040h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0044h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0048h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_004Ch	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0050h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0054h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0058h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_005Ch	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0060h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0064h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0068h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_006Ch	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0070h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0074h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0078h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W14)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_007Ch	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0140h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0144h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0148h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_014Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0180h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_01A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_01A8h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_01C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_01D0h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_01F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0240h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0244h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0248h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_024Ch	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0250h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0254h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0258h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W6)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_025Ch	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0260h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0264h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0268h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_026Ch	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0270h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0274h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0278h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_027Ch	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0340h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0344h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0348h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_034Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0380h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_03A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_03A8h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_03C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_03D0h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM3_BLK_CFG_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_03F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0440h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0444h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0448h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_044Ch	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0450h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0454h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0458h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_045Ch	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0460h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0464h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0468h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_046Ch	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0470h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0474h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0478h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_047Ch	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0540h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM0_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0544h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0548h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_054Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0580h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_05A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_05A8h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_05C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_05D0h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_05F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0640h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0644h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0648h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_064Ch	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0650h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0654h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0658h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_065Ch	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0660h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W8)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0664h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0668h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_066Ch	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0670h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0674h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0678h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_067Ch	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0740h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0744h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0748h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_074Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0780h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_07A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_07A8h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_07C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_07D0h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_07F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0840h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0844h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0848h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_084Ch	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0850h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0854h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0858h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_085Ch	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0860h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0864h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0868h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_086Ch	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0870h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0874h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0878h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_087Ch	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0940h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0944h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0948h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM0_BLK_NSE_W2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_094Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0980h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_09A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_09A8h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_09C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_09D0h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_09F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0A40h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0A44h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0A48h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_0A4Ch	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0A50h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0A54h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0A58h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_0A5Ch	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0A60h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0A64h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0A68h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W10)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0A6Ch	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0A70h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0A74h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0A78h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_0A7Ch	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0B40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0B44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0B48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_0B4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0B80h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_0BA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_0BA8h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_0BC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_0BD0h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_0BF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0C40h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0C44h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0C48h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0C4Ch	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0C50h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0C54h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0C58h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_0C5Ch	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0C60h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0C64h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0C68h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_0C6Ch	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0C70h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0C74h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0C78h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_0C7Ch	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0D40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0D44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0D48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_0D4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0D80h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM1_BLK_CFG_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0DA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_0DA8h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_0DC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_0DD0h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_0DF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0E40h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0E44h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0E48h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_0E4Ch	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0E50h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0E54h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0E58h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_0E5Ch	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0E60h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0E64h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0E68h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_0E6Ch	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0E70h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W12)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0E74h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0E78h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_0E7Ch	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0F40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0F44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0F48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_0F4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0F80h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_0FA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_0FA8h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_0FC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_0FD0h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_0FF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1040h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1044h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1048h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_104Ch	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1050h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1054h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1058h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_105Ch	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1060h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1064h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1068h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_106Ch	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1070h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1074h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1078h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_107Ch	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1140h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1144h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1148h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_114Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1180h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_11A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_11A8h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM2_BLK_CFG_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_11C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_11D0h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_11F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1240h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1244h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1248h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_124Ch	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1250h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1254h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1258h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_125Ch	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1260h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1264h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1268h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_126Ch	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1270h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1274h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1278h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W14)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_127Ch	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1340h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1344h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1348h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_134Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1380h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_13A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_13A8h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_13C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_13D0h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_13F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1440h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1444h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1448h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_144Ch	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1450h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1454h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1458h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W6)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_145Ch	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1460h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1464h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1468h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_146Ch	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1470h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1474h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1478h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_147Ch	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1540h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1544h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1548h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_154Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1580h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_15A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_15A8h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_15C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_15D0h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM3_BLK_CFG_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_15F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1640h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1644h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1648h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_164Ch	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1650h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1654h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1658h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_165Ch	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1660h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1664h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1668h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_166Ch	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1670h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1674h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1678h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_167Ch	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1740h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM0_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1744h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1748h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_174Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1780h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_17A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_17A8h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_17C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_17D0h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_17F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1840h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1844h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1848h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_184Ch	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1850h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1854h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1858h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_185Ch	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1860h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W8)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1864h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1868h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_186Ch	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1870h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1874h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1878h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_187Ch	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1940h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1944h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1948h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_194Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1980h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_19A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_19A8h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_19C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_19D0h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_19F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1A40h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1A44h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1A48h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_1A4Ch	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1A50h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1A54h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1A58h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_1A5Ch	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1A60h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1A64h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1A68h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_1A6Ch	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1A70h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1A74h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1A78h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_1A7Ch	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1B40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1B44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1B48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM0_BLK_NSE_W2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1B4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1B80h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_1BA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_1BA8h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_1BC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_1BD0h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_1BF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1C40h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1C44h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1C48h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_1C4Ch	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1C50h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1C54h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1C58h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_1C5Ch	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1C60h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1C64h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1C68h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W10)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1C6Ch	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1C70h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1C74h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1C78h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_1C7Ch	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1D40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1D44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1D48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_1D4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1D80h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_1DA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_1DA8h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_1DC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_1DD0h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_1DF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1E40h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1E44h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1E48h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1E4Ch	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1E50h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1E54h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1E58h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_1E5Ch	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1E60h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1E64h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1E68h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_1E6Ch	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1E70h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1E74h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1E78h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_1E7Ch	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1F40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1F44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1F48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_1F4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1F80h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM1_BLK_CFG_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1FA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_1FA8h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_1FC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_1FD0h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_1FF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_2000h	MBC Global Configuration Register (MBC1_MEM0_GLBCFG)	32	R	000C_0020h
1_2004h	MBC Global Configuration Register (MBC1_MEM1_GLBCFG)	32	R	000C_0020h
1_2008h	MBC Global Configuration Register (MBC1_MEM2_GLBCFG)	32	R	000C_0000h
1_200Ch	MBC Global Configuration Register (MBC1_MEM3_GLBCFG)	32	R	000C_0000h
1_2010h	MBC NonSecure Enable Block Index (MBC1_NSE_BLK_INDEX)	32	RW	0000_0000h
1_2014h	MBC NonSecure Enable Block Set (MBC1_NSE_BLK_SET)	32	W	0000_0000h
1_2018h	MBC NonSecure Enable Block Clear (MBC1_NSE_BLK_CLR)	32	W	0000_0000h
1_201Ch	MBC NonSecure Enable Block Clear All (MBC1_NSE_BLK_CLR_ALL)	32	RW	0000_0000h
1_2020h	MBC Global Access Control (MBC1_MEMN_GLBAC0)	32	RW	0000_0000h
1_2024h	MBC Global Access Control (MBC1_MEMN_GLBAC1)	32	RW	0000_0000h
1_2028h	MBC Global Access Control (MBC1_MEMN_GLBAC2)	32	RW	0000_0000h
1_202Ch	MBC Global Access Control (MBC1_MEMN_GLBAC3)	32	RW	0000_0000h
1_2030h	MBC Global Access Control (MBC1_MEMN_GLBAC4)	32	RW	0000_0000h
1_2034h	MBC Global Access Control (MBC1_MEMN_GLBAC5)	32	RW	0000_0000h
1_2038h	MBC Global Access Control (MBC1_MEMN_GLBAC6)	32	RW	0000_0000h
1_203Ch	MBC Global Access Control (MBC1_MEMN_GLBAC7)	32	RW	0000_0000h
1_2040h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2044h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2048h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_204Ch	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2140h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2180h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_2184h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_2188h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_218Ch	MBC Memory Block Configuration Word (MBC1_DOM0_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_21A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_2240h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2244h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2248h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_224Ch	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2340h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2380h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_2384h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_2388h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_238Ch	MBC Memory Block Configuration Word (MBC1_DOM1_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_23A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_2440h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2444h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2448h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_244Ch	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2540h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2580h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_2584h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_2588h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_258Ch	MBC Memory Block Configuration Word (MBC1_DOM2_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_25A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_2640h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2644h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2648h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_264Ch	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2740h	MBC Memory Block NonSecure Enable Word (MBC1_DOM3_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2780h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_2784h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_2788h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_278Ch	MBC Memory Block Configuration Word (MBC1_DOM3_MEM1_BLK_CFG_W3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_27A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM3_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_2840h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2844h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2848h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_284Ch	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2940h	MBC Memory Block NonSecure Enable Word (MBC1_DOM4_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2980h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_2984h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_2988h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_298Ch	MBC Memory Block Configuration Word (MBC1_DOM4_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_29A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM4_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_2A40h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2A44h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2A48h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_2A4Ch	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2B40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM5_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2B80h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_2B84h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM1_BLK_CFG_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2B88h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_2B8Ch	MBC Memory Block Configuration Word (MBC1_DOM5_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_2BA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM5_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_2C40h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2C44h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2C48h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_2C4Ch	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2D40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM6_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2D80h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_2D84h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_2D88h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_2D8Ch	MBC Memory Block Configuration Word (MBC1_DOM6_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_2DA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM6_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_2E40h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2E44h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2E48h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_2E4Ch	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2F40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM7_MEM0_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2F80h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_2F84h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_2F88h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_2F8Ch	MBC Memory Block Configuration Word (MBC1_DOM7_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_2FA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM7_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_3040h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3044h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3048h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_304Ch	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3140h	MBC Memory Block NonSecure Enable Word (MBC1_DOM8_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3180h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_3184h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_3188h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_318Ch	MBC Memory Block Configuration Word (MBC1_DOM8_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_31A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM8_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_3240h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3244h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3248h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_324Ch	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3340h	MBC Memory Block NonSecure Enable Word (MBC1_DOM9_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3380h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_3384h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_3388h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_338Ch	MBC Memory Block Configuration Word (MBC1_DOM9_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_33A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM9_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_3440h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3444h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3448h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_344Ch	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3540h	MBC Memory Block NonSecure Enable Word (MBC1_DOM10_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3580h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_3584h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_3588h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_358Ch	MBC Memory Block Configuration Word (MBC1_DOM10_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_35A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM10_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_3640h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3644h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3648h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_364Ch	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3740h	MBC Memory Block NonSecure Enable Word (MBC1_DOM11_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3780h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_3784h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_3788h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_378Ch	MBC Memory Block Configuration Word (MBC1_DOM11_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_37A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM11_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_3840h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3844h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3848h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_384Ch	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3940h	MBC Memory Block NonSecure Enable Word (MBC1_DOM12_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3980h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_3984h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_3988h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_398Ch	MBC Memory Block Configuration Word (MBC1_DOM12_MEM1_BLK_CFG_W3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_39A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM12_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_3A40h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3A44h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3A48h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_3A4Ch	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3B40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM13_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3B80h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_3B84h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_3B88h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_3B8Ch	MBC Memory Block Configuration Word (MBC1_DOM13_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_3BA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM13_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_3C40h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3C44h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3C48h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_3C4Ch	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3D40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM14_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3D80h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_3D84h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM1_BLK_CFG_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3D88h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_3D8Ch	MBC Memory Block Configuration Word (MBC1_DOM14_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_3DA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM14_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_3E40h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3E44h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3E48h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_3E4Ch	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3F40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM15_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3F80h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_3F84h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM1_BLK_CFG_W1)	32	RW	0000_0000h
1_3F88h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM1_BLK_CFG_W2)	32	RW	0000_0000h
1_3F8Ch	MBC Memory Block Configuration Word (MBC1_DOM15_MEM1_BLK_CFG_W3)	32	RW	0000_0000h
1_3FA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM15_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_4000h	MRC Global Configuration Register (MRC0_GLBCFG)	32	R	0000_0008h
1_4010h	MRC NonSecure Enable Region Indirect (MRC0_NSE_RGN_INDIRECT)	32	RW	0000_0000h
1_4014h	MRC NonSecure Enable Region Set (MRC0_NSE_RGN_SET)	32	RW	0000_0000h
1_4018h	MRC NonSecure Enable Region Clear (MRC0_NSE_RGN_CLR)	32	RW	0000_0000h
1_401Ch	MRC NonSecure Enable Region Clear All (MRC0_NSE_RGN_CLR_ALL)	32	RW	0000_0000h
1_4020h	MRC Global Access Control (MRC0_GLBAC0)	32	RW	0000_0000h
1_4024h	MRC Global Access Control (MRC0_GLBAC1)	32	RW	0000_0000h
1_4028h	MRC Global Access Control (MRC0_GLBAC2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_402Ch	MRC Global Access Control (MRC0_GLBAC3)	32	RW	0000_0000h
1_4030h	MRC Global Access Control (MRC0_GLBAC4)	32	RW	0000_0000h
1_4034h	MRC Global Access Control (MRC0_GLBAC5)	32	RW	0000_0000h
1_4038h	MRC Global Access Control (MRC0_GLBAC6)	32	RW	0000_0000h
1_403Ch	MRC Global Access Control (MRC0_GLBAC7)	32	RW	0000_0000h
1_4040h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD0_W0)	32	RW	0000_0000h
1_4044h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD0_W1)	32	RW	0000_0000h
1_4048h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD1_W0)	32	RW	0000_0000h
1_404Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD1_W1)	32	RW	0000_0000h
1_4050h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD2_W0)	32	RW	0000_0000h
1_4054h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD2_W1)	32	RW	0000_0000h
1_4058h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD3_W0)	32	RW	0000_0000h
1_405Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD3_W1)	32	RW	0000_0000h
1_4060h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD4_W0)	32	RW	0000_0000h
1_4064h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD4_W1)	32	RW	0000_0000h
1_4068h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD5_W0)	32	RW	0000_0000h
1_406Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD5_W1)	32	RW	0000_0000h
1_4070h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD6_W0)	32	RW	0000_0000h
1_4074h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD6_W1)	32	RW	0000_0000h
1_4078h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD7_W0)	32	RW	0000_0000h
1_407Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD7_W1)	32	RW	0000_0000h
1_40C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM0_RGD_NSE)	32	RW	0000_0000h
1_4140h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD0_W0)	32	RW	0000_0000h
1_4144h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD0_W1)	32	RW	0000_0000h
1_4148h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD1_W0)	32	RW	0000_0000h
1_414Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD1_W1)	32	RW	0000_0000h
1_4150h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD2_W0)	32	RW	0000_0000h
1_4154h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD2_W1)	32	RW	0000_0000h
1_4158h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD3_W0)	32	RW	0000_0000h
1_415Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD3_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4160h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD4_W0)	32	RW	0000_0000h
1_4164h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD4_W1)	32	RW	0000_0000h
1_4168h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD5_W0)	32	RW	0000_0000h
1_416Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD5_W1)	32	RW	0000_0000h
1_4170h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD6_W0)	32	RW	0000_0000h
1_4174h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD6_W1)	32	RW	0000_0000h
1_4178h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD7_W0)	32	RW	0000_0000h
1_417Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD7_W1)	32	RW	0000_0000h
1_41C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM1_RGD_NSE)	32	RW	0000_0000h
1_4240h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD0_W0)	32	RW	0000_0000h
1_4244h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD0_W1)	32	RW	0000_0000h
1_4248h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD1_W0)	32	RW	0000_0000h
1_424Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD1_W1)	32	RW	0000_0000h
1_4250h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD2_W0)	32	RW	0000_0000h
1_4254h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD2_W1)	32	RW	0000_0000h
1_4258h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD3_W0)	32	RW	0000_0000h
1_425Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD3_W1)	32	RW	0000_0000h
1_4260h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD4_W0)	32	RW	0000_0000h
1_4264h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD4_W1)	32	RW	0000_0000h
1_4268h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD5_W0)	32	RW	0000_0000h
1_426Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD5_W1)	32	RW	0000_0000h
1_4270h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD6_W0)	32	RW	0000_0000h
1_4274h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD6_W1)	32	RW	0000_0000h
1_4278h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD7_W0)	32	RW	0000_0000h
1_427Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD7_W1)	32	RW	0000_0000h
1_42C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM2_RGD_NSE)	32	RW	0000_0000h
1_4340h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD0_W0)	32	RW	0000_0000h
1_4344h	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD0_W1)	32	RW	0000_0000h
1_4348h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD1_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_434Ch	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD1_W1)	32	RW	0000_0000h
1_4350h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD2_W0)	32	RW	0000_0000h
1_4354h	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD2_W1)	32	RW	0000_0000h
1_4358h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD3_W0)	32	RW	0000_0000h
1_435Ch	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD3_W1)	32	RW	0000_0000h
1_4360h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD4_W0)	32	RW	0000_0000h
1_4364h	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD4_W1)	32	RW	0000_0000h
1_4368h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD5_W0)	32	RW	0000_0000h
1_436Ch	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD5_W1)	32	RW	0000_0000h
1_4370h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD6_W0)	32	RW	0000_0000h
1_4374h	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD6_W1)	32	RW	0000_0000h
1_4378h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD7_W0)	32	RW	0000_0000h
1_437Ch	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD7_W1)	32	RW	0000_0000h
1_43C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM3_RGD_NSE)	32	RW	0000_0000h
1_4440h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD0_W0)	32	RW	0000_0000h
1_4444h	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD0_W1)	32	RW	0000_0000h
1_4448h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD1_W0)	32	RW	0000_0000h
1_444Ch	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD1_W1)	32	RW	0000_0000h
1_4450h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD2_W0)	32	RW	0000_0000h
1_4454h	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD2_W1)	32	RW	0000_0000h
1_4458h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD3_W0)	32	RW	0000_0000h
1_445Ch	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD3_W1)	32	RW	0000_0000h
1_4460h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD4_W0)	32	RW	0000_0000h
1_4464h	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD4_W1)	32	RW	0000_0000h
1_4468h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD5_W0)	32	RW	0000_0000h
1_446Ch	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD5_W1)	32	RW	0000_0000h
1_4470h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD6_W0)	32	RW	0000_0000h
1_4474h	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD6_W1)	32	RW	0000_0000h
1_4478h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD7_W0)	32	RW	0000_0000h
1_447Ch	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD7_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_44C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM4_RGD_NSE)	32	RW	0000_0000h
1_4540h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD0_W0)	32	RW	0000_0000h
1_4544h	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD0_W1)	32	RW	0000_0000h
1_4548h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD1_W0)	32	RW	0000_0000h
1_454Ch	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD1_W1)	32	RW	0000_0000h
1_4550h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD2_W0)	32	RW	0000_0000h
1_4554h	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD2_W1)	32	RW	0000_0000h
1_4558h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD3_W0)	32	RW	0000_0000h
1_455Ch	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD3_W1)	32	RW	0000_0000h
1_4560h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD4_W0)	32	RW	0000_0000h
1_4564h	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD4_W1)	32	RW	0000_0000h
1_4568h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD5_W0)	32	RW	0000_0000h
1_456Ch	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD5_W1)	32	RW	0000_0000h
1_4570h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD6_W0)	32	RW	0000_0000h
1_4574h	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD6_W1)	32	RW	0000_0000h
1_4578h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD7_W0)	32	RW	0000_0000h
1_457Ch	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD7_W1)	32	RW	0000_0000h
1_45C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM5_RGD_NSE)	32	RW	0000_0000h
1_4640h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD0_W0)	32	RW	0000_0000h
1_4644h	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD0_W1)	32	RW	0000_0000h
1_4648h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD1_W0)	32	RW	0000_0000h
1_464Ch	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD1_W1)	32	RW	0000_0000h
1_4650h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD2_W0)	32	RW	0000_0000h
1_4654h	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD2_W1)	32	RW	0000_0000h
1_4658h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD3_W0)	32	RW	0000_0000h
1_465Ch	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD3_W1)	32	RW	0000_0000h
1_4660h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD4_W0)	32	RW	0000_0000h
1_4664h	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD4_W1)	32	RW	0000_0000h
1_4668h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD5_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_466Ch	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD5_W1)	32	RW	0000_0000h
1_4670h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD6_W0)	32	RW	0000_0000h
1_4674h	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD6_W1)	32	RW	0000_0000h
1_4678h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD7_W0)	32	RW	0000_0000h
1_467Ch	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD7_W1)	32	RW	0000_0000h
1_46C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM6_RGD_NSE)	32	RW	0000_0000h
1_4740h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD0_W0)	32	RW	0000_0000h
1_4744h	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD0_W1)	32	RW	0000_0000h
1_4748h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD1_W0)	32	RW	0000_0000h
1_474Ch	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD1_W1)	32	RW	0000_0000h
1_4750h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD2_W0)	32	RW	0000_0000h
1_4754h	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD2_W1)	32	RW	0000_0000h
1_4758h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD3_W0)	32	RW	0000_0000h
1_475Ch	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD3_W1)	32	RW	0000_0000h
1_4760h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD4_W0)	32	RW	0000_0000h
1_4764h	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD4_W1)	32	RW	0000_0000h
1_4768h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD5_W0)	32	RW	0000_0000h
1_476Ch	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD5_W1)	32	RW	0000_0000h
1_4770h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD6_W0)	32	RW	0000_0000h
1_4774h	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD6_W1)	32	RW	0000_0000h
1_4778h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD7_W0)	32	RW	0000_0000h
1_477Ch	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD7_W1)	32	RW	0000_0000h
1_47C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM7_RGD_NSE)	32	RW	0000_0000h
1_4840h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD0_W0)	32	RW	0000_0000h
1_4844h	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD0_W1)	32	RW	0000_0000h
1_4848h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD1_W0)	32	RW	0000_0000h
1_484Ch	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD1_W1)	32	RW	0000_0000h
1_4850h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD2_W0)	32	RW	0000_0000h
1_4854h	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD2_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4858h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD3_W0)	32	RW	0000_0000h
1_485Ch	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD3_W1)	32	RW	0000_0000h
1_4860h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD4_W0)	32	RW	0000_0000h
1_4864h	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD4_W1)	32	RW	0000_0000h
1_4868h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD5_W0)	32	RW	0000_0000h
1_486Ch	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD5_W1)	32	RW	0000_0000h
1_4870h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD6_W0)	32	RW	0000_0000h
1_4874h	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD6_W1)	32	RW	0000_0000h
1_4878h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD7_W0)	32	RW	0000_0000h
1_487Ch	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD7_W1)	32	RW	0000_0000h
1_48C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM8_RGD_NSE)	32	RW	0000_0000h
1_4940h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD0_W0)	32	RW	0000_0000h
1_4944h	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD0_W1)	32	RW	0000_0000h
1_4948h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD1_W0)	32	RW	0000_0000h
1_494Ch	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD1_W1)	32	RW	0000_0000h
1_4950h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD2_W0)	32	RW	0000_0000h
1_4954h	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD2_W1)	32	RW	0000_0000h
1_4958h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD3_W0)	32	RW	0000_0000h
1_495Ch	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD3_W1)	32	RW	0000_0000h
1_4960h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD4_W0)	32	RW	0000_0000h
1_4964h	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD4_W1)	32	RW	0000_0000h
1_4968h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD5_W0)	32	RW	0000_0000h
1_496Ch	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD5_W1)	32	RW	0000_0000h
1_4970h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD6_W0)	32	RW	0000_0000h
1_4974h	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD6_W1)	32	RW	0000_0000h
1_4978h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD7_W0)	32	RW	0000_0000h
1_497Ch	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD7_W1)	32	RW	0000_0000h
1_49C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM9_RGD_NSE)	32	RW	0000_0000h
1_4A40h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD0_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4A44h	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD0_W1)	32	RW	0000_0000h
1_4A48h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD1_W0)	32	RW	0000_0000h
1_4A4Ch	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD1_W1)	32	RW	0000_0000h
1_4A50h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD2_W0)	32	RW	0000_0000h
1_4A54h	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD2_W1)	32	RW	0000_0000h
1_4A58h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD3_W0)	32	RW	0000_0000h
1_4A5Ch	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD3_W1)	32	RW	0000_0000h
1_4A60h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD4_W0)	32	RW	0000_0000h
1_4A64h	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD4_W1)	32	RW	0000_0000h
1_4A68h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD5_W0)	32	RW	0000_0000h
1_4A6Ch	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD5_W1)	32	RW	0000_0000h
1_4A70h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD6_W0)	32	RW	0000_0000h
1_4A74h	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD6_W1)	32	RW	0000_0000h
1_4A78h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD7_W0)	32	RW	0000_0000h
1_4A7Ch	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD7_W1)	32	RW	0000_0000h
1_4AC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM10_RGD_NSE)	32	RW	0000_0000h
1_4B40h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD0_W0)	32	RW	0000_0000h
1_4B44h	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD0_W1)	32	RW	0000_0000h
1_4B48h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD1_W0)	32	RW	0000_0000h
1_4B4Ch	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD1_W1)	32	RW	0000_0000h
1_4B50h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD2_W0)	32	RW	0000_0000h
1_4B54h	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD2_W1)	32	RW	0000_0000h
1_4B58h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD3_W0)	32	RW	0000_0000h
1_4B5Ch	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD3_W1)	32	RW	0000_0000h
1_4B60h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD4_W0)	32	RW	0000_0000h
1_4B64h	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD4_W1)	32	RW	0000_0000h
1_4B68h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD5_W0)	32	RW	0000_0000h
1_4B6Ch	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD5_W1)	32	RW	0000_0000h
1_4B70h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD6_W0)	32	RW	0000_0000h
1_4B74h	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD6_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4B78h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD7_W0)	32	RW	0000_0000h
1_4B7Ch	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD7_W1)	32	RW	0000_0000h
1_4BC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM11_RGD_NSE)	32	RW	0000_0000h
1_4C40h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD0_W0)	32	RW	0000_0000h
1_4C44h	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD0_W1)	32	RW	0000_0000h
1_4C48h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD1_W0)	32	RW	0000_0000h
1_4C4Ch	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD1_W1)	32	RW	0000_0000h
1_4C50h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD2_W0)	32	RW	0000_0000h
1_4C54h	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD2_W1)	32	RW	0000_0000h
1_4C58h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD3_W0)	32	RW	0000_0000h
1_4C5Ch	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD3_W1)	32	RW	0000_0000h
1_4C60h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD4_W0)	32	RW	0000_0000h
1_4C64h	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD4_W1)	32	RW	0000_0000h
1_4C68h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD5_W0)	32	RW	0000_0000h
1_4C6Ch	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD5_W1)	32	RW	0000_0000h
1_4C70h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD6_W0)	32	RW	0000_0000h
1_4C74h	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD6_W1)	32	RW	0000_0000h
1_4C78h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD7_W0)	32	RW	0000_0000h
1_4C7Ch	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD7_W1)	32	RW	0000_0000h
1_4CC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM12_RGD_NSE)	32	RW	0000_0000h
1_4D40h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD0_W0)	32	RW	0000_0000h
1_4D44h	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD0_W1)	32	RW	0000_0000h
1_4D48h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD1_W0)	32	RW	0000_0000h
1_4D4Ch	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD1_W1)	32	RW	0000_0000h
1_4D50h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD2_W0)	32	RW	0000_0000h
1_4D54h	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD2_W1)	32	RW	0000_0000h
1_4D58h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD3_W0)	32	RW	0000_0000h
1_4D5Ch	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD3_W1)	32	RW	0000_0000h
1_4D60h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD4_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4D64h	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD4_W1)	32	RW	0000_0000h
1_4D68h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD5_W0)	32	RW	0000_0000h
1_4D6Ch	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD5_W1)	32	RW	0000_0000h
1_4D70h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD6_W0)	32	RW	0000_0000h
1_4D74h	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD6_W1)	32	RW	0000_0000h
1_4D78h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD7_W0)	32	RW	0000_0000h
1_4D7Ch	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD7_W1)	32	RW	0000_0000h
1_4DC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM13_RGD_NSE)	32	RW	0000_0000h
1_4E40h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD0_W0)	32	RW	0000_0000h
1_4E44h	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD0_W1)	32	RW	0000_0000h
1_4E48h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD1_W0)	32	RW	0000_0000h
1_4E4Ch	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD1_W1)	32	RW	0000_0000h
1_4E50h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD2_W0)	32	RW	0000_0000h
1_4E54h	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD2_W1)	32	RW	0000_0000h
1_4E58h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD3_W0)	32	RW	0000_0000h
1_4E5Ch	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD3_W1)	32	RW	0000_0000h
1_4E60h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD4_W0)	32	RW	0000_0000h
1_4E64h	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD4_W1)	32	RW	0000_0000h
1_4E68h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD5_W0)	32	RW	0000_0000h
1_4E6Ch	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD5_W1)	32	RW	0000_0000h
1_4E70h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD6_W0)	32	RW	0000_0000h
1_4E74h	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD6_W1)	32	RW	0000_0000h
1_4E78h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD7_W0)	32	RW	0000_0000h
1_4E7Ch	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD7_W1)	32	RW	0000_0000h
1_4EC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM14_RGD_NSE)	32	RW	0000_0000h
1_4F40h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD0_W0)	32	RW	0000_0000h
1_4F44h	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD0_W1)	32	RW	0000_0000h
1_4F48h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD1_W0)	32	RW	0000_0000h
1_4F4Ch	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD1_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4F50h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD2_W0)	32	RW	0000_0000h
1_4F54h	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD2_W1)	32	RW	0000_0000h
1_4F58h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD3_W0)	32	RW	0000_0000h
1_4F5Ch	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD3_W1)	32	RW	0000_0000h
1_4F60h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD4_W0)	32	RW	0000_0000h
1_4F64h	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD4_W1)	32	RW	0000_0000h
1_4F68h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD5_W0)	32	RW	0000_0000h
1_4F6Ch	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD5_W1)	32	RW	0000_0000h
1_4F70h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD6_W0)	32	RW	0000_0000h
1_4F74h	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD6_W1)	32	RW	0000_0000h
1_4F78h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD7_W0)	32	RW	0000_0000h
1_4F7Ch	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD7_W1)	32	RW	0000_0000h
1_4FC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM15_RGD_NSE)	32	RW	0000_0000h
1_5000h	MRC Global Configuration Register (MRC1_GLBCFG)	32	R	0000_0008h
1_5010h	MRC NonSecure Enable Region Indirect (MRC1_NSE_RGN_INDIRECT)	32	RW	0000_0000h
1_5014h	MRC NonSecure Enable Region Set (MRC1_NSE_RGN_SET)	32	RW	0000_0000h
1_5018h	MRC NonSecure Enable Region Clear (MRC1_NSE_RGN_CLR)	32	RW	0000_0000h
1_501Ch	MRC NonSecure Enable Region Clear All (MRC1_NSE_RGN_CLR_ALL)	32	RW	0000_0000h
1_5020h	MRC Global Access Control (MRC1_GLBAC0)	32	RW	0000_0000h
1_5024h	MRC Global Access Control (MRC1_GLBAC1)	32	RW	0000_0000h
1_5028h	MRC Global Access Control (MRC1_GLBAC2)	32	RW	0000_0000h
1_502Ch	MRC Global Access Control (MRC1_GLBAC3)	32	RW	0000_0000h
1_5030h	MRC Global Access Control (MRC1_GLBAC4)	32	RW	0000_0000h
1_5034h	MRC Global Access Control (MRC1_GLBAC5)	32	RW	0000_0000h
1_5038h	MRC Global Access Control (MRC1_GLBAC6)	32	RW	0000_0000h
1_503Ch	MRC Global Access Control (MRC1_GLBAC7)	32	RW	0000_0000h
1_5040h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD0_W0)	32	RW	0000_0000h
1_5044h	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD0_W1)	32	RW	0000_0000h
1_5048h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD1_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_504Ch	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD1_W1)	32	RW	0000_0000h
1_5050h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD2_W0)	32	RW	0000_0000h
1_5054h	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD2_W1)	32	RW	0000_0000h
1_5058h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD3_W0)	32	RW	0000_0000h
1_505Ch	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD3_W1)	32	RW	0000_0000h
1_5060h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD4_W0)	32	RW	0000_0000h
1_5064h	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD4_W1)	32	RW	0000_0000h
1_5068h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD5_W0)	32	RW	0000_0000h
1_506Ch	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD5_W1)	32	RW	0000_0000h
1_5070h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD6_W0)	32	RW	0000_0000h
1_5074h	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD6_W1)	32	RW	0000_0000h
1_5078h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD7_W0)	32	RW	0000_0000h
1_507Ch	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD7_W1)	32	RW	0000_0000h
1_50C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM0_RGD_NSE)	32	RW	0000_0000h
1_5140h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD0_W0)	32	RW	0000_0000h
1_5144h	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD0_W1)	32	RW	0000_0000h
1_5148h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD1_W0)	32	RW	0000_0000h
1_514Ch	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD1_W1)	32	RW	0000_0000h
1_5150h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD2_W0)	32	RW	0000_0000h
1_5154h	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD2_W1)	32	RW	0000_0000h
1_5158h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD3_W0)	32	RW	0000_0000h
1_515Ch	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD3_W1)	32	RW	0000_0000h
1_5160h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD4_W0)	32	RW	0000_0000h
1_5164h	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD4_W1)	32	RW	0000_0000h
1_5168h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD5_W0)	32	RW	0000_0000h
1_516Ch	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD5_W1)	32	RW	0000_0000h
1_5170h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD6_W0)	32	RW	0000_0000h
1_5174h	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD6_W1)	32	RW	0000_0000h
1_5178h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD7_W0)	32	RW	0000_0000h
1_517Ch	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD7_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_51C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM1_RGD_NSE)	32	RW	0000_0000h
1_5240h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD0_W0)	32	RW	0000_0000h
1_5244h	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD0_W1)	32	RW	0000_0000h
1_5248h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD1_W0)	32	RW	0000_0000h
1_524Ch	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD1_W1)	32	RW	0000_0000h
1_5250h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD2_W0)	32	RW	0000_0000h
1_5254h	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD2_W1)	32	RW	0000_0000h
1_5258h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD3_W0)	32	RW	0000_0000h
1_525Ch	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD3_W1)	32	RW	0000_0000h
1_5260h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD4_W0)	32	RW	0000_0000h
1_5264h	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD4_W1)	32	RW	0000_0000h
1_5268h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD5_W0)	32	RW	0000_0000h
1_526Ch	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD5_W1)	32	RW	0000_0000h
1_5270h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD6_W0)	32	RW	0000_0000h
1_5274h	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD6_W1)	32	RW	0000_0000h
1_5278h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD7_W0)	32	RW	0000_0000h
1_527Ch	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD7_W1)	32	RW	0000_0000h
1_52C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM2_RGD_NSE)	32	RW	0000_0000h
1_5340h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD0_W0)	32	RW	0000_0000h
1_5344h	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD0_W1)	32	RW	0000_0000h
1_5348h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD1_W0)	32	RW	0000_0000h
1_534Ch	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD1_W1)	32	RW	0000_0000h
1_5350h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD2_W0)	32	RW	0000_0000h
1_5354h	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD2_W1)	32	RW	0000_0000h
1_5358h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD3_W0)	32	RW	0000_0000h
1_535Ch	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD3_W1)	32	RW	0000_0000h
1_5360h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD4_W0)	32	RW	0000_0000h
1_5364h	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD4_W1)	32	RW	0000_0000h
1_5368h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD5_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_536Ch	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD5_W1)	32	RW	0000_0000h
1_5370h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD6_W0)	32	RW	0000_0000h
1_5374h	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD6_W1)	32	RW	0000_0000h
1_5378h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD7_W0)	32	RW	0000_0000h
1_537Ch	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD7_W1)	32	RW	0000_0000h
1_53C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM3_RGD_NSE)	32	RW	0000_0000h
1_5440h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD0_W0)	32	RW	0000_0000h
1_5444h	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD0_W1)	32	RW	0000_0000h
1_5448h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD1_W0)	32	RW	0000_0000h
1_544Ch	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD1_W1)	32	RW	0000_0000h
1_5450h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD2_W0)	32	RW	0000_0000h
1_5454h	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD2_W1)	32	RW	0000_0000h
1_5458h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD3_W0)	32	RW	0000_0000h
1_545Ch	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD3_W1)	32	RW	0000_0000h
1_5460h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD4_W0)	32	RW	0000_0000h
1_5464h	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD4_W1)	32	RW	0000_0000h
1_5468h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD5_W0)	32	RW	0000_0000h
1_546Ch	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD5_W1)	32	RW	0000_0000h
1_5470h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD6_W0)	32	RW	0000_0000h
1_5474h	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD6_W1)	32	RW	0000_0000h
1_5478h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD7_W0)	32	RW	0000_0000h
1_547Ch	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD7_W1)	32	RW	0000_0000h
1_54C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM4_RGD_NSE)	32	RW	0000_0000h
1_5540h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD0_W0)	32	RW	0000_0000h
1_5544h	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD0_W1)	32	RW	0000_0000h
1_5548h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD1_W0)	32	RW	0000_0000h
1_554Ch	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD1_W1)	32	RW	0000_0000h
1_5550h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD2_W0)	32	RW	0000_0000h
1_5554h	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD2_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5558h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD3_W0)	32	RW	0000_0000h
1_555Ch	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD3_W1)	32	RW	0000_0000h
1_5560h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD4_W0)	32	RW	0000_0000h
1_5564h	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD4_W1)	32	RW	0000_0000h
1_5568h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD5_W0)	32	RW	0000_0000h
1_556Ch	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD5_W1)	32	RW	0000_0000h
1_5570h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD6_W0)	32	RW	0000_0000h
1_5574h	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD6_W1)	32	RW	0000_0000h
1_5578h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD7_W0)	32	RW	0000_0000h
1_557Ch	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD7_W1)	32	RW	0000_0000h
1_55C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM5_RGD_NSE)	32	RW	0000_0000h
1_5640h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD0_W0)	32	RW	0000_0000h
1_5644h	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD0_W1)	32	RW	0000_0000h
1_5648h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD1_W0)	32	RW	0000_0000h
1_564Ch	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD1_W1)	32	RW	0000_0000h
1_5650h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD2_W0)	32	RW	0000_0000h
1_5654h	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD2_W1)	32	RW	0000_0000h
1_5658h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD3_W0)	32	RW	0000_0000h
1_565Ch	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD3_W1)	32	RW	0000_0000h
1_5660h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD4_W0)	32	RW	0000_0000h
1_5664h	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD4_W1)	32	RW	0000_0000h
1_5668h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD5_W0)	32	RW	0000_0000h
1_566Ch	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD5_W1)	32	RW	0000_0000h
1_5670h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD6_W0)	32	RW	0000_0000h
1_5674h	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD6_W1)	32	RW	0000_0000h
1_5678h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD7_W0)	32	RW	0000_0000h
1_567Ch	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD7_W1)	32	RW	0000_0000h
1_56C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM6_RGD_NSE)	32	RW	0000_0000h
1_5740h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD0_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5744h	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD0_W1)	32	RW	0000_0000h
1_5748h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD1_W0)	32	RW	0000_0000h
1_574Ch	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD1_W1)	32	RW	0000_0000h
1_5750h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD2_W0)	32	RW	0000_0000h
1_5754h	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD2_W1)	32	RW	0000_0000h
1_5758h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD3_W0)	32	RW	0000_0000h
1_575Ch	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD3_W1)	32	RW	0000_0000h
1_5760h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD4_W0)	32	RW	0000_0000h
1_5764h	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD4_W1)	32	RW	0000_0000h
1_5768h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD5_W0)	32	RW	0000_0000h
1_576Ch	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD5_W1)	32	RW	0000_0000h
1_5770h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD6_W0)	32	RW	0000_0000h
1_5774h	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD6_W1)	32	RW	0000_0000h
1_5778h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD7_W0)	32	RW	0000_0000h
1_577Ch	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD7_W1)	32	RW	0000_0000h
1_57C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM7_RGD_NSE)	32	RW	0000_0000h
1_5840h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD0_W0)	32	RW	0000_0000h
1_5844h	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD0_W1)	32	RW	0000_0000h
1_5848h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD1_W0)	32	RW	0000_0000h
1_584Ch	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD1_W1)	32	RW	0000_0000h
1_5850h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD2_W0)	32	RW	0000_0000h
1_5854h	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD2_W1)	32	RW	0000_0000h
1_5858h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD3_W0)	32	RW	0000_0000h
1_585Ch	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD3_W1)	32	RW	0000_0000h
1_5860h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD4_W0)	32	RW	0000_0000h
1_5864h	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD4_W1)	32	RW	0000_0000h
1_5868h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD5_W0)	32	RW	0000_0000h
1_586Ch	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD5_W1)	32	RW	0000_0000h
1_5870h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD6_W0)	32	RW	0000_0000h
1_5874h	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD6_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5878h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD7_W0)	32	RW	0000_0000h
1_587Ch	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD7_W1)	32	RW	0000_0000h
1_58C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM8_RGD_NSE)	32	RW	0000_0000h
1_5940h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD0_W0)	32	RW	0000_0000h
1_5944h	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD0_W1)	32	RW	0000_0000h
1_5948h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD1_W0)	32	RW	0000_0000h
1_594Ch	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD1_W1)	32	RW	0000_0000h
1_5950h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD2_W0)	32	RW	0000_0000h
1_5954h	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD2_W1)	32	RW	0000_0000h
1_5958h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD3_W0)	32	RW	0000_0000h
1_595Ch	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD3_W1)	32	RW	0000_0000h
1_5960h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD4_W0)	32	RW	0000_0000h
1_5964h	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD4_W1)	32	RW	0000_0000h
1_5968h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD5_W0)	32	RW	0000_0000h
1_596Ch	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD5_W1)	32	RW	0000_0000h
1_5970h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD6_W0)	32	RW	0000_0000h
1_5974h	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD6_W1)	32	RW	0000_0000h
1_5978h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD7_W0)	32	RW	0000_0000h
1_597Ch	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD7_W1)	32	RW	0000_0000h
1_59C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM9_RGD_NSE)	32	RW	0000_0000h
1_5A40h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD0_W0)	32	RW	0000_0000h
1_5A44h	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD0_W1)	32	RW	0000_0000h
1_5A48h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD1_W0)	32	RW	0000_0000h
1_5A4Ch	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD1_W1)	32	RW	0000_0000h
1_5A50h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD2_W0)	32	RW	0000_0000h
1_5A54h	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD2_W1)	32	RW	0000_0000h
1_5A58h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD3_W0)	32	RW	0000_0000h
1_5A5Ch	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD3_W1)	32	RW	0000_0000h
1_5A60h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD4_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5A64h	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD4_W1)	32	RW	0000_0000h
1_5A68h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD5_W0)	32	RW	0000_0000h
1_5A6Ch	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD5_W1)	32	RW	0000_0000h
1_5A70h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD6_W0)	32	RW	0000_0000h
1_5A74h	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD6_W1)	32	RW	0000_0000h
1_5A78h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD7_W0)	32	RW	0000_0000h
1_5A7Ch	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD7_W1)	32	RW	0000_0000h
1_5AC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM10_RGD_NSE)	32	RW	0000_0000h
1_5B40h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD0_W0)	32	RW	0000_0000h
1_5B44h	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD0_W1)	32	RW	0000_0000h
1_5B48h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD1_W0)	32	RW	0000_0000h
1_5B4Ch	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD1_W1)	32	RW	0000_0000h
1_5B50h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD2_W0)	32	RW	0000_0000h
1_5B54h	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD2_W1)	32	RW	0000_0000h
1_5B58h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD3_W0)	32	RW	0000_0000h
1_5B5Ch	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD3_W1)	32	RW	0000_0000h
1_5B60h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD4_W0)	32	RW	0000_0000h
1_5B64h	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD4_W1)	32	RW	0000_0000h
1_5B68h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD5_W0)	32	RW	0000_0000h
1_5B6Ch	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD5_W1)	32	RW	0000_0000h
1_5B70h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD6_W0)	32	RW	0000_0000h
1_5B74h	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD6_W1)	32	RW	0000_0000h
1_5B78h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD7_W0)	32	RW	0000_0000h
1_5B7Ch	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD7_W1)	32	RW	0000_0000h
1_5BC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM11_RGD_NSE)	32	RW	0000_0000h
1_5C40h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD0_W0)	32	RW	0000_0000h
1_5C44h	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD0_W1)	32	RW	0000_0000h
1_5C48h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD1_W0)	32	RW	0000_0000h
1_5C4Ch	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD1_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5C50h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD2_W0)	32	RW	0000_0000h
1_5C54h	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD2_W1)	32	RW	0000_0000h
1_5C58h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD3_W0)	32	RW	0000_0000h
1_5C5Ch	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD3_W1)	32	RW	0000_0000h
1_5C60h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD4_W0)	32	RW	0000_0000h
1_5C64h	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD4_W1)	32	RW	0000_0000h
1_5C68h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD5_W0)	32	RW	0000_0000h
1_5C6Ch	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD5_W1)	32	RW	0000_0000h
1_5C70h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD6_W0)	32	RW	0000_0000h
1_5C74h	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD6_W1)	32	RW	0000_0000h
1_5C78h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD7_W0)	32	RW	0000_0000h
1_5C7Ch	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD7_W1)	32	RW	0000_0000h
1_5CC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM12_RGD_NSE)	32	RW	0000_0000h
1_5D40h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD0_W0)	32	RW	0000_0000h
1_5D44h	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD0_W1)	32	RW	0000_0000h
1_5D48h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD1_W0)	32	RW	0000_0000h
1_5D4Ch	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD1_W1)	32	RW	0000_0000h
1_5D50h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD2_W0)	32	RW	0000_0000h
1_5D54h	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD2_W1)	32	RW	0000_0000h
1_5D58h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD3_W0)	32	RW	0000_0000h
1_5D5Ch	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD3_W1)	32	RW	0000_0000h
1_5D60h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD4_W0)	32	RW	0000_0000h
1_5D64h	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD4_W1)	32	RW	0000_0000h
1_5D68h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD5_W0)	32	RW	0000_0000h
1_5D6Ch	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD5_W1)	32	RW	0000_0000h
1_5D70h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD6_W0)	32	RW	0000_0000h
1_5D74h	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD6_W1)	32	RW	0000_0000h
1_5D78h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD7_W0)	32	RW	0000_0000h
1_5D7Ch	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD7_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5DC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM13_RGD_NSE)	32	RW	0000_0000h
1_5E40h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD0_W0)	32	RW	0000_0000h
1_5E44h	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD0_W1)	32	RW	0000_0000h
1_5E48h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD1_W0)	32	RW	0000_0000h
1_5E4Ch	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD1_W1)	32	RW	0000_0000h
1_5E50h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD2_W0)	32	RW	0000_0000h
1_5E54h	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD2_W1)	32	RW	0000_0000h
1_5E58h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD3_W0)	32	RW	0000_0000h
1_5E5Ch	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD3_W1)	32	RW	0000_0000h
1_5E60h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD4_W0)	32	RW	0000_0000h
1_5E64h	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD4_W1)	32	RW	0000_0000h
1_5E68h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD5_W0)	32	RW	0000_0000h
1_5E6Ch	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD5_W1)	32	RW	0000_0000h
1_5E70h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD6_W0)	32	RW	0000_0000h
1_5E74h	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD6_W1)	32	RW	0000_0000h
1_5E78h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD7_W0)	32	RW	0000_0000h
1_5E7Ch	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD7_W1)	32	RW	0000_0000h
1_5EC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM14_RGD_NSE)	32	RW	0000_0000h
1_5F40h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD0_W0)	32	RW	0000_0000h
1_5F44h	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD0_W1)	32	RW	0000_0000h
1_5F48h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD1_W0)	32	RW	0000_0000h
1_5F4Ch	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD1_W1)	32	RW	0000_0000h
1_5F50h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD2_W0)	32	RW	0000_0000h
1_5F54h	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD2_W1)	32	RW	0000_0000h
1_5F58h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD3_W0)	32	RW	0000_0000h
1_5F5Ch	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD3_W1)	32	RW	0000_0000h
1_5F60h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD4_W0)	32	RW	0000_0000h
1_5F64h	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD4_W1)	32	RW	0000_0000h
1_5F68h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD5_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5F6Ch	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD5_W1)	32	RW	0000_0000h
1_5F70h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD6_W0)	32	RW	0000_0000h
1_5F74h	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD6_W1)	32	RW	0000_0000h
1_5F78h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD7_W0)	32	RW	0000_0000h
1_5F7Ch	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD7_W1)	32	RW	0000_0000h
1_5FC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM15_RGD_NSE)	32	RW	0000_0000h

43.8.1.2 TRDC Register (TRDC_CR)

Offset

Register	Offset
TRDC_CR	0h

Function

This register provides status about the TRDC and global enable bits for the entire module's operation. A fully operational TRDC requires all global bits GVLD {R,B,M} to be asserted. Undefined behavior results if they are not all asserted. If there are no region checkers (MRCs) present in the configuration, GVLDR need not be asserted for a fully operational TRDC.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	LK1	0										0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GVLDR	GVLDB	0						0				HRL			GVLDM
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 LK1	Lock Status This field is the lock status of the TRDC_CR. This field is set when all GVLD bits are set; GVLDR=GVLDB=GVLDM=1b1. Once set, this bit remains asserted until the next reset. 0b - The CR can be written by any secure privileged write. 1b - The CR is locked (read-only) until the next reset.
29-20 —	Reserved
19-16 —	Reserved
15 GVLDR	Global Valid for Memory Region Checkers TRDC global MRC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC MRCs are disabled. 1b - TRDC MRCs are enabled.
14 GVLDB	Global Valid for Memory Block Checkers TRDC global MBC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC MBCs are disabled. 1b - TRDC MBCs are enabled.
13-9 —	Reserved
8-5 —	Reserved
4-1 HRL	Hardware Revision Level This read-only field specifies the TRDC’s hardware and definition revision level. It can be read by software to determine the functional definition of the module.
0 GVLDM	Global Valid for Domain Assignment Controllers TRDC global DAC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC DACs are disabled. 1b - TRDC DACs are enabled.

43.8.1.3 TRDC Hardware Configuration Register 0 (TRDC_HWCFG0)

Offset

Register	Offset
TRDC_HWCFG0	F0h

Function

This read-only register contains information on the TRDC's hardware configuration. Specifically, it defines the number of implemented domains and bus masters along with the number of instances of memory block checkers (MBCs) and memory region checkers (MRCs). The register value at reset is device-specific. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MID				NMRC				0				NMBC			
W																
Reset	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NMSTR								0				NDID			
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-29 MID	Module ID This field defines major version ID of this module.
28-24 NMRC	Number of MRCs This field defines the number of MRCs on the device [1-16].
23-20 —	Reserved
19-16 NMBC	Number of MBCs This field defines the number of MBCs on the device [1-8].
15-8 NMSTR	Number of bus masters This read-only field defines the number of bus masters.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-5 —	Reserved
4-0 NDID	Number of domains This read-only field defines the number of domains on the device [1-16].

43.8.1.4 TRDC Hardware Configuration Register 1 (TRDC_HWCFG1)

Offset

Register	Offset
TRDC_HWCFG1	F4h

Function

This register contains information on the TRDC's hardware configuration. It provides a mechanism for software to determine its domain number by simply reading the register. See Domain error capture management for more details on typical usage. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												DID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	u	u	u	u

Fields

Field	Function
31-4 —	Reserved
3-0 DID	Domain identifier number This field provides the domain number [0-15] of the requesting bus master.

43.8.1.5 TRDC Hardware Configuration Register 2 (TRDC_HWCFG2)

Offset

Register	Offset
TRDC_HWCFG2	F8h

Function

This register contains information on the TRDC's hardware configuration. It provides a bitmap signaling the presence of a process identifier (PID) register sourced from the given bus master. The HWCFG2 register is associated with bus masters [31-0]. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PIDPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PIDPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PIDPn	Process identifier present Process identifier present from bus master n, where n = [31-0]. This field provides a bitmap to signal that bus master (where n is defined as bit number) sources a process identifier register to the TRDC_DAC logic. For each bit in this field: • 0 - Bus master n does not source a process identifier register. The TRDC_DAC logic provides the needed PID for processor cores. • 1 - Bus master n sources a process identifier register.

43.8.1.6 TRDC Hardware Configuration Register 3 (TRDC_HWCFG3)

Offset

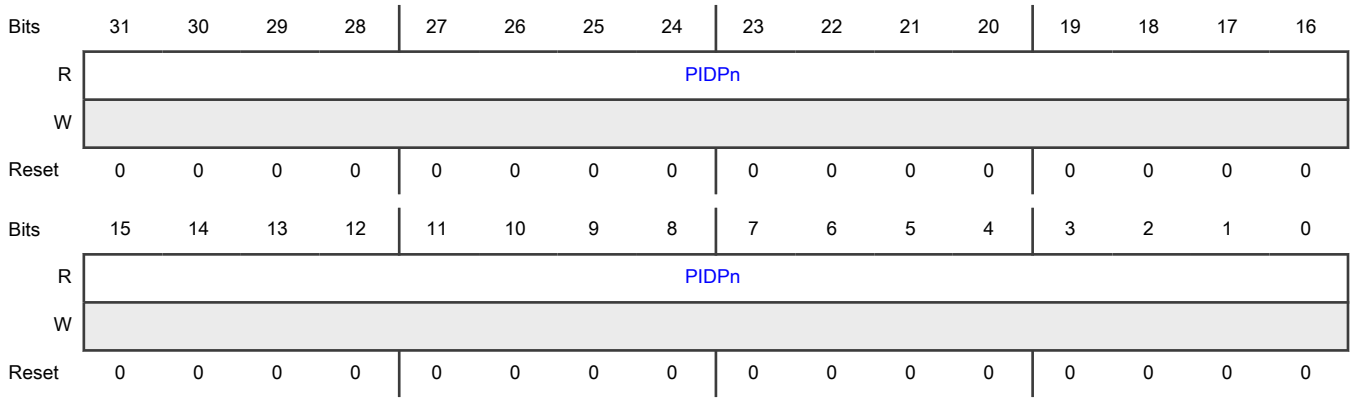
Register	Offset
TRDC_HWCFG3	FCh

Function

This register contains information on the TRDC's hardware configuration. Specifically, it provides a bitmap signaling the presence of a process identifier (PID) register sourced from the given bus master. The HWCFG3 register is associated with bus masters [63-32], while the HWCFG2 register covers bus masters [31-0]. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 PIDPn	Process identifier present Process identifier present from bus master n, where n = [63-32]. This field provides a bitmap to signal that bus master n (where n is defined as 32 + bit number) sources a process identifier register to the TRDC_DAC logic. Process identifier present from bus master n, where n = [31-0]. This field provides a bitmap to signal that bus master (where n is defined as bit number) sources a process identifier register to the TRDC_DAC logic. For each bit in this field: • 0 - Bus master n does not source a process identifier register. The TRDC_DAC logic provides the needed PID for processor cores. • 1 - Bus master n sources a process identifier register.

43.8.1.7 Domain Assignment Configuration Register (DACFG0 - DACFG3)

Offset

Register	Offset
DACFG0	100h
DACFG1	101h
DACFG2	102h
DACFG3	103h

Function

This register defines the number of implemented domain assignment registers for bus master m, where m+1 can specify from 1 to 64 bus masters. These registers are organized as a byte-sized data array and can be read using 8-, 16- or 32-bit accesses. An all-zero value (NCM = 0, NMDAR = 0) indicates a non-existent bus master. Attempted writes are error terminated.

Register read will not return transfer error when DACFG for a master doesn't exist but it is in the same 32-bit group as DACFG for an existing master. For example, when there are only 2 DACFG0-1 registers for 2 master, and masters for DACFG2-3 don't exist, then access to DACFG2-3 won't return transfer error.

Typically, processor bus masters have one or more domain assignment registers, while non-processor masters have a single domain assignment register.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	7	6	5	4	3	2	1	0
R	NCM	0			NMDAR			
W								
Reset	See Register reset values.							

Register reset values

Register	Reset value
DACFG0	01h
DACFG1	03h
DACFG2–DACFG3	81h

Fields

Field	Function
7 NCM	Non-CPU Master This read-only field signals that bus master m is a non-CPU master. It specifies that the format of the associated MDA_Wr_m register defines a non-processor domain assignment. This field is zero for a non-existent bus master. 0b - Bus master is a processor. 1b - Bus master is a non-processor.
6-4 —	Reserved
3-0 NMDAR	Number of master domain assignment registers for bus master m This read-only field specifies the number of registers associated with the master domain assignment register for a given bus master. The value is limited to the range [0-8], where zero indicates a non-existent bus master and non-zero values indicate the number of implemented registers associated with this MDAm.

43.8.1.8 TRDC IDAU Control Register (TRDC_IDAU_CR)

Offset

Register	Offset
TRDC_IDAU_CR	1C0h

Function

This register defines the configuration for the Implementation-Defined Attribution Unit which is tightly-coupled to the TZ-M extensions in the processor core. Many of the fields in this register explicitly control processor TZ-M functions.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PCURRNS	0												0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			LKSAU	LKNSMPU	LKSMPU	LKNSVTOR	LKSVTAI...	0	SAUDIS	MPUNSDIS	MPUSDIS	CFGSECE...	0		VLD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31 PCURRNS	Processor current security Current security state of the processor. 0b - Processor is in Secure state 1b - Processor is in Nonsecure state
30-19 —	Reserved
18-16 —	Reserved
15-13 —	Reserved
12	Lock SAU

Table continues on the next page...

Table continued from the previous page...

Field	Function
LKSAU	Asserting this bit prevents changes to the processor's Secure SAU memory regions already programmed. All processor writes to the registers are ignored. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers 1b - Disable writes to the SAU_CTRL, SAU_RNR, SAU_RBAR and SAU_RLAR registers from software or from a debug agent connected to the processor
11 LKNSMPU	Lock Nonsecure MPU Asserting this bit prevents changes to the processor's Nonsecure MPU memory regions already programmed. All processor writes to the registers are ignored. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers 1b - Disable writes to the MPU_CTRL_NS, MPU_RNR_NS, MPU_RBAR_NS, MPU_RLAR_NS, MPU_RBAR_A_NSn and MPU_RLAR_A_NSn from software or from a debug agent connected to the processor
10 LKSMPU	Lock Secure MPU Asserting this signal prevents changes to the processor's programmed Secure MPU memory regions and all processor writes to the registers are ignored. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers 1b - Disable writes to the MPU_CTRL, MPU_RNR, MPU_RBAR, MPU_RLAR, MPU_RBAR_An and MPU_RLAR_An from software or from a debug agent connected to the processor in Secure state
9 LKNSVTOR	Lock Nonsecure Vector Table Offset Register Asserting this signal prevents changes to the processor's Nonsecure vector table base address. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock this register 1b - Disable writes to the VTOR_NS register
8 LKSVTAIRCR	Lock Secure VTOR, Application interrupt and Reset Control Registers This bit is sticky, once set, it can only be cleared with a reset. Asserting this signal prevents processor changes to: • The Secure vector table base address. • Handling of Secure interrupt priority. • BusFault, HardFault, and NMI security target settings in the processor. 0b - Unlock these registers 1b - Disable writes to the VTOR_S, AIRCR[PRIS], and AIRCR[BFHFNMINS] registers
7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 SAUDIS	Security Attribution Unit Disable 0b - SAU is enabled 1b - SAU is disabled
5 MPUNSDIS	NonSecure Memory Protection Unit Disabled 0b - Nonsecure MPU is enabled 1b - Nonsecure MPU is disabled
4 MPUSDIS	Secure Memory Protection Unit Disabled 0b - Secure MPU is enabled 1b - Secure MPU is disabled
3 CFGSECEXT	Configure Security Extension 0b - Armv8M Security Extension is disabled 1b - Armv8-M Security Extension is enabled
2-1 —	Reserved
0 VLD	Valid When VLD =0, all address attribute from IDAU is nonSecure. When VLD=1, values in other fields of this register are valid.

43.8.1.9 TRDC FLW Control (TRDC_FLW_CTL)

Offset

Register	Offset
TRDC_FLW_CTL	1E0h

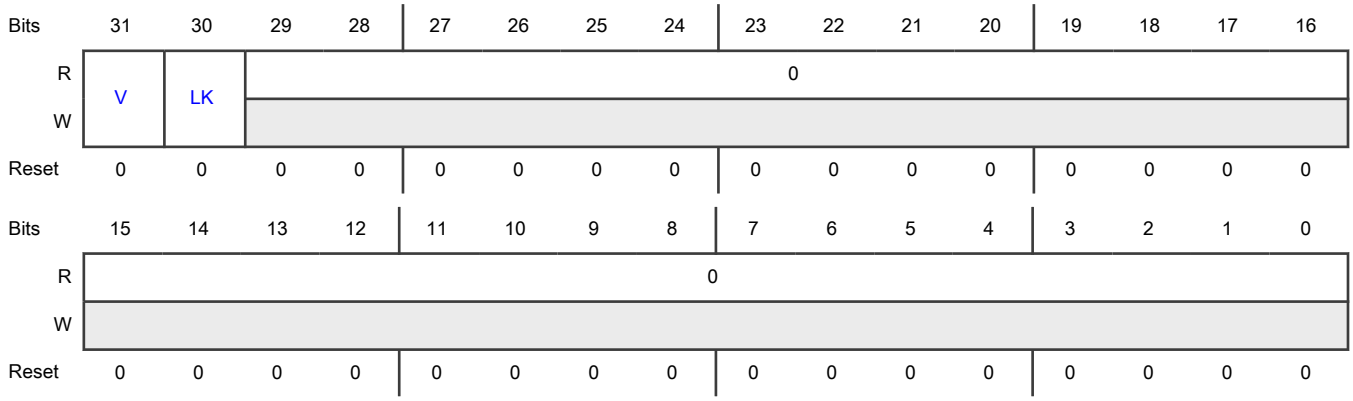
Function

This register provides control of the FLW = Flash Logical Window operation.

This register exists for this device but has no effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31 V	Valid bit 0b - FLW function is disabled. 1b - FLW function is enabled.
30 LK	Lock bit 0b - FLW registers may be modified. 1b - FLW registers are locked until the next reset.
29-0 —	Reserved

43.8.1.10 TRDC FLW Physical Base (TRDC_FLW_PBASE)

Offset

Register	Offset
TRDC_FLW_PBASE	1E4h

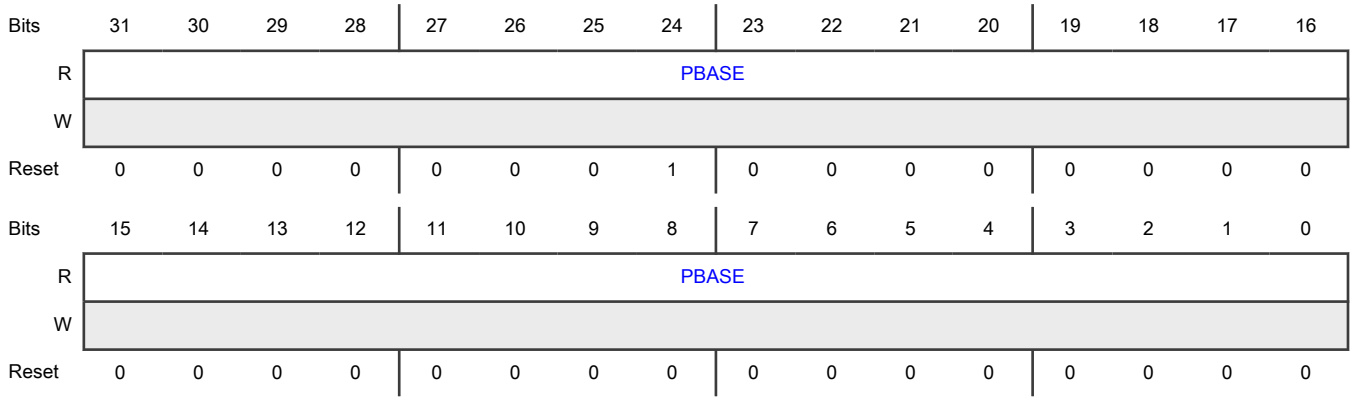
Function

This read-only register gives the physical base address of the Flash Logical Window.

This register exists for this device but has not effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 PBASE	Physical base address Physical address of the base of the FLW (mod 32KBytes).

43.8.1.11 TRDC FLW Array Base (TRDC_FLW_ABASE)

Offset

Register	Offset
TRDC_FLW_ABASE	1E8h

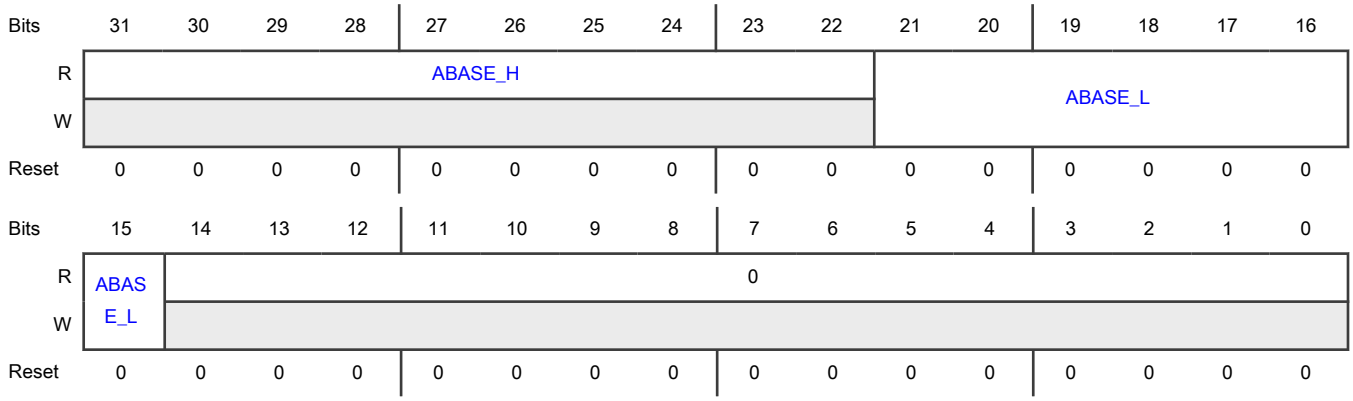
Function

This register gives the flash array base address of the Flash Logical Window.

This register exists for this device but has no effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write. This register is not writable if TRDC_FLW_CTL[LK]=1.

Diagram



Fields

Field	Function
31-22 ABASE_H	Array base address high Flash array address upper bits of the base of the FLW (mod 32KBytes).
21-15 ABASE_L	Array base address low Flash array address lower bits of the base of the FLW (mod 32KBytes).
14-0 —	Reserved

43.8.1.12 TRDC FLW Block Count (TRDC_FLW_BCNT)

Offset

Register	Offset
TRDC_FLW_BCNT	1ECh

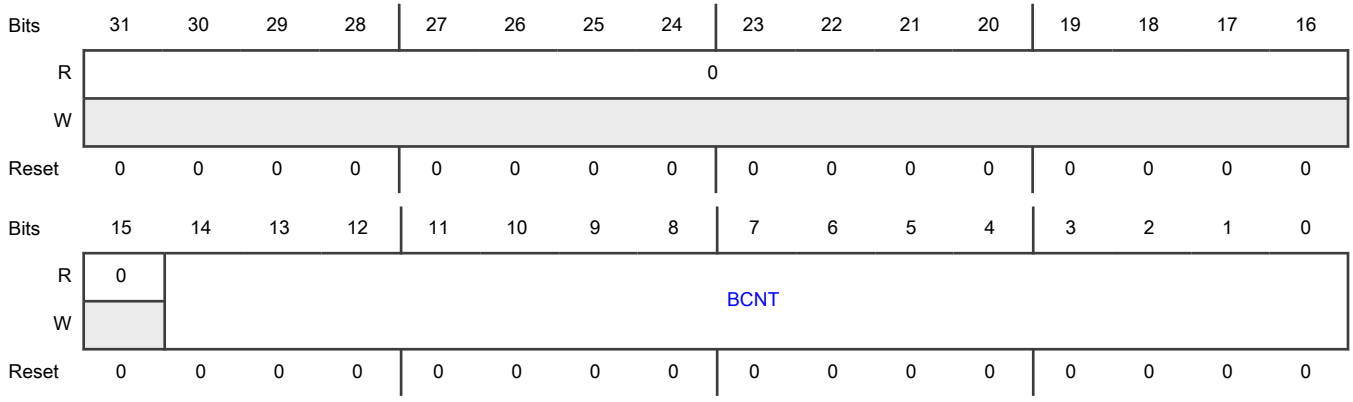
Function

This register gives the size of the Flash Logic Window in 32KByte blocks.

This register exists for this device but has no effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write. This register is not writable if TRDC_FLW_CTL[LK]=1.

Diagram



Fields

Field	Function
31-15	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-0 BCNT	Block Count Size of FLW in number of 32KByte blocks.

43.8.1.13 TRDC Fault Domain ID (TRDC_FDID)

Offset

Register	Offset
TRDC_FDID	1FCh

Function

In the event of an access error, this register is used to specify the domainID of the faulting reference before indexing into the Domain Error registers.

If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												FDID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 FDID	Domain ID of Faulted Access This field indicates the domainID of the fault used to index the array of domain error information. The user queries the DERRLOCx registers to find the DomainID of the faulting access and must write this register with the domain ID before reading the Domain Error registers.

43.8.1.14 TRDC Domain Error Location Register (TRDC_DERRLOC0 - TRDC_DERRLOC15)

Offset

For d = 0 to 15:

Register	Offset
TRDC_DERRLOCd	200h + (d × 4h)

Function

This array of read-only registers provide the instance number of the submodule where (an) access violation(s) occurred. These registers are organized as a word array, indexed by the faulting domain number, d. The two fields of this register provide a bitmap of instances associated with all submodules containing captured error information for that domain. These instance numbers are then used as indices into the DERR_W0_*i*, DERR_W1_*i*, and DERR_W3_*i*register arrays. See Domain error capture management for more details.

When an access violation is detected by either a Memory Region Checker (MRC) or a Memory Block Checker (MBC), address and attribute information of the offending access is captured. Using the faulting domainID number as the index, d, this array of read-only registers provide additional information signaling the instance number of the submodule where the access violation(s) occurred. Since the resulting exception handler needs the submodule instance to retrieve the captured address and attribute information from DERR_W0_*i* and DERR_W1_*i*, these registers provide the instance number details.

Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MRCINST															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								MBCINST							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 MRCINST	MRC instance This field is a bitmap indicating the presence of a detected access violation for domain d in the implemented instances of the MRC. The least-significant bit of this field (bit 16) corresponds to MRC instance 0. The most-significant bit of this field (bit 31) corresponds to MRC instance 15 (MRC instance = i-16 where i is the register bit number), and so on. Multiple bits can be set at any time indicating access violations for this domain have been detected across multiple instances of the MRCs.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For each bit in this field: • 0 - The memory region checker has not detected an access violation or is not physically present. • 1 - The memory region checker has detected one or more access violations for this domain.
15-8 —	Reserved
7-0 MBCINST	MBC instance This field is a bitmap indicating the presence of a detected access violation for domain d in the implemented instances of the MBC. The least-significant bit of this field (bit 0) corresponds to MBC instance 0. The most-significant bit of this field (bit 7) corresponds to MBC instance 7 Multiple bits can be set at any time indicating access violations for this domain have been detected across multiple instances of the MBCs. For each bit in this field: • 0 - The memory block checker has not detected an access violation or is not physically present. • 1 - The memory block checker has detected one or more access violations for this domain.

43.8.1.15 MBC Domain Error Word0 Register (MBC0_DERR_W0 - MBC1_DERR_W0)

Offset

Register	Offset
MBC0_DERR_W0	400h
MBC1_DERR_W0	410h

Function

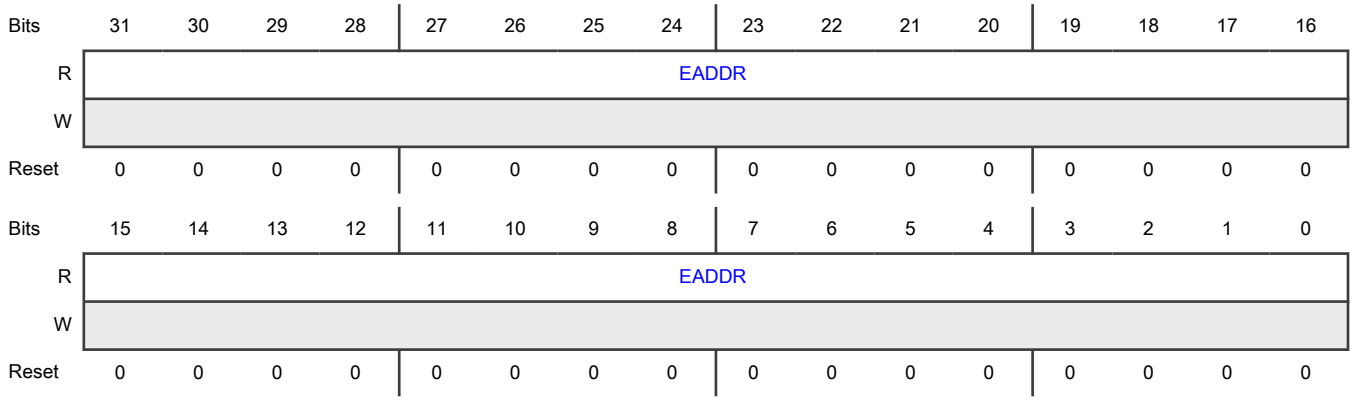
This read-only register array provides the address of an access violation detected by memory block checker (MBC). These registers are organized as a word array, which is indexed by the MBC instance number. That is, the index, i, of this array is the instance number of MBC with the access violation. The submodule instance numbers are provided by the DERRLOC registers. The memory mapped error capture detail registers are organized as 24 sequential 16 byte entries.

When an access violation is detected and the offending information captured, subsequent updates to this register are disabled until the required data pattern is written to the DERR_W3_i register. At that time, this register is cleared and re-enabled to capture the next access violation.

Attempted writes are error terminated as are attempted reads of an MBC instance that is not physically present.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 EADDR	Error address This is the unaliased virtual address of the access that generated the access violation.

43.8.1.16 MBC Domain Error Word1 Register (MBC0_DERR_W1 - MBC1_DERR_W1)

Offset

Register	Offset
MBC0_DERR_W1	404h
MBC1_DERR_W1	414h

Function

This read-only register array provides the attributes of an access violation detected by Memory Block Checker (MBC). These registers are organized as a word array, which is indexed by the violating submodule instance number. Refer to register Domain Error Location(DERRLOCd), Domain Error Word0 Register(DERR_W0_i), and Domain error capture management for more information.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EST		0		EPORT				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				ERW	EATR			0				EDID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 EST	Error state This field signals the state of access violations for this domain in this instance of the block checker or region checker. Once an access violation has been detected and the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. After retrieving the faulting address and attributes, the error capture mechanism must be rearmed by performing a write to DERR_W3_i. 00b - No access violation has been detected. 01b - No access violation has been detected. 10b - A single access violation has been detected. 11b - Multiple access violations for this domain have been detected by this submodule instance. Only the address and attribute information for the first error have been captured in DERR_W0_i and DERR_W1_i.
29-27 —	Reserved
26-24 EPORT	Error port This field identifies the encoded port number of the MBC that detected the access violation. The MBC port number connection is device-specific. See the chip configuration details for more information. This is the MBC slave that has the violation. 000b - mbcxslv0 001b - mbcxslv1 010b - mbcxslv2 011b - mbcxslv3
23-12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 ERW	Error read/write This field signals whether the captured access violation occurred on a read or write reference. 0b - Read access 1b - Write access
10-8 EATR	Error attributes This field captures attributes of the access violation. 000b - Secure user mode, instruction fetch access. 001b - Secure user mode, data access. 010b - Secure privileged mode, instruction fetch access. 011b - Secure privileged mode, data access. 100b - Nonsecure user mode, instruction fetch access. 101b - Nonsecure user mode, data access. 110b - Nonsecure privileged mode, instruction fetch access. 111b - Nonsecure privileged mode, data access.
7-4 —	Reserved
3-0 EDID	Error domain identifier This field captures the domain identifier of the access violation.

43.8.1.17 MBC Domain Error Word3 Register (MBC0_DERR_W3 - MBC1_DERR_W3)

Offset

Register	Offset
MBC0_DERR_W3	40Ch
MBC1_DERR_W3	41Ch

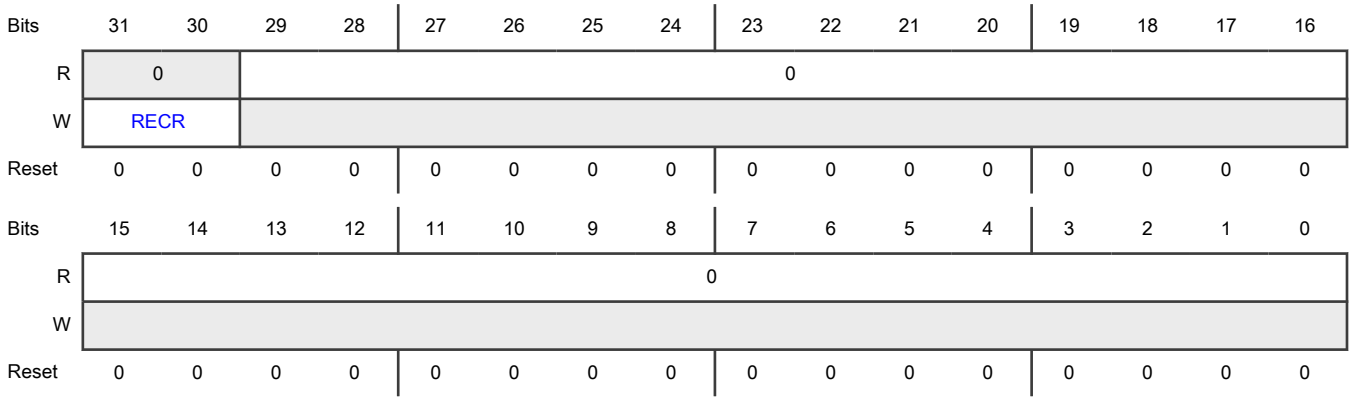
Function

This register is used to rearm the error capture logic and clear the DERR_W0_i and DERR_W1_i registers. After the domain access violation error details have been read, typically in an exception service routine, a 32-bit word write to this register is required to rearm the error capture logic.

A read of this location returns zeroes. Attempted reads of a Memory Region Checker (MRC) or Memory Block Checker (MBC) instance that is not physically present are error terminated.

See Domain error capture management for more details.

Diagram



Fields

Field	Function
31-30 RECR	Rearm Error Capture Registers This 2-bit, write-only field controls the rearming of the domain error capture registers. Once an access violation has been detected with the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. Writing 01b to this field rearms the error capture mechanism and clears the DERR_W0_i and DERR_W1_i registers. A write of any value other than 01b has no effect.
29-0 —	Reserved

43.8.1.18 MRC Domain Error Word0 Register (MRC0_DERR_W0 - MRC1_DERR_W0)

Offset

Register	Offset
MRC0_DERR_W0	480h
MRC1_DERR_W0	490h

Function

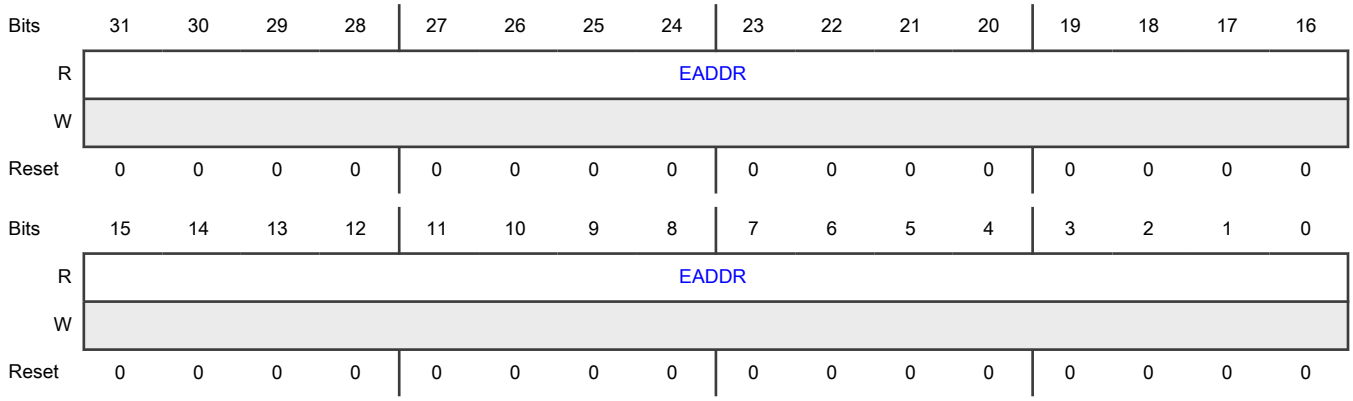
This read-only register array provides the address of an access violation detected by memory region checker (MRC). These registers are organized as a word array, which is indexed by the MRC instance number. That is, the index, i, of this array is the instance number of MRC with the access violation. The submodule instance numbers are provided by the DERRLOC registers. The memory mapped error capture detail registers are organized as 24 sequential 16 byte entries.

When an access violation is detected and the offending information captured, subsequent updates to this register are disabled until the required data pattern is written to the DERR_W3_i register. At that time, this register is cleared and re-enabled to capture the next access violation.

Attempted writes are error terminated as are attempted reads of an MRC instance that is not physically present.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 EADDR	Error address This is the unaliased virtual address of the access that generated the access violation.

43.8.1.19 MRC Domain Error Word1 Register (MRC0_DERR_W1 - MRC1_DERR_W1)

Offset

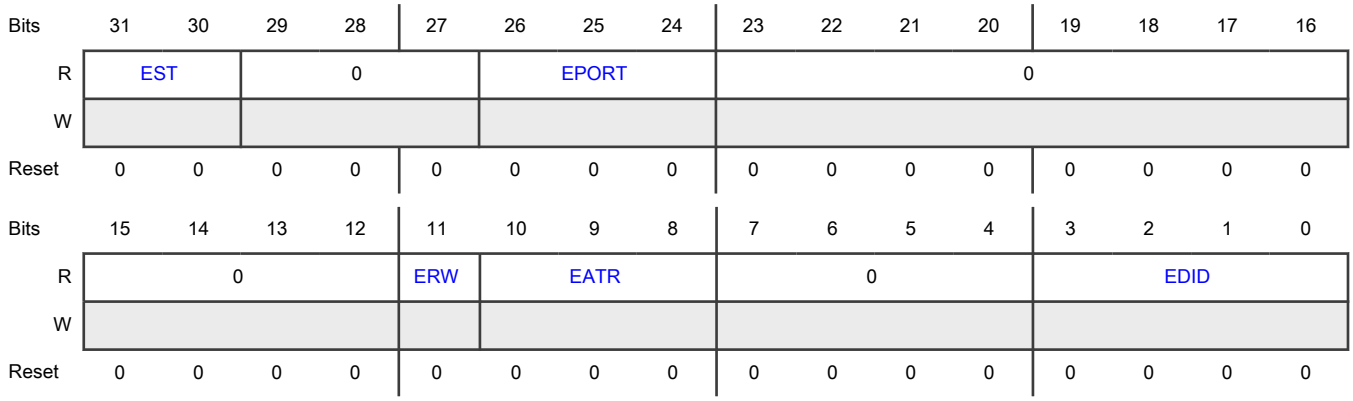
Register	Offset
MRC0_DERR_W1	484h
MRC1_DERR_W1	494h

Function

This read-only register array provides the attributes of an access violation detected by memory region checker (MRC). These registers are organized as a word array, which is indexed by the violating submodule instance number. Refer to register Domain Error Location(DERRLOCd), Domain Error Word0 Register(DERR_W0_i), and Domain error capture management for more information.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-30 EST	Error state This field signals the state of access violations for this domain in this instance of the block checker or region checker. Once an access violation has been detected and the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. After retrieving the faulting address and attributes, the error capture mechanism must be rearmed by performing a write to DERR_W3_i. 00b - No access violation has been detected. 01b - No access violation has been detected. 10b - A single access violation has been detected. 11b - Multiple access violations for this domain have been detected by this submodule instance. Only the address and attribute information for the first error have been captured in DERR_W0_i and DERR_W1_i.
29-27 —	Reserved
26-24 EPORT	Error port This field identifies the encoded port number of the MRC that detected the access violation. The MRC port number connection is device-specific. See the chip configuration details for more information.
23-12 —	Reserved
11 ERW	Error read/write This field signals whether the captured access violation occurred on a read or write reference. 0b - Read access 1b - Write access
10-8	Error attributes

Table continues on the next page...

Table continued from the previous page...

Field	Function
EATR	This field captures certain attributes of the access violation. 000b - Secure user mode, instruction fetch access. 001b - Secure user mode, data access. 010b - Secure privileged mode, instruction fetch access. 011b - Secure privileged mode, data access. 100b - Nonsecure user mode, instruction fetch access. 101b - Nonsecure user mode, data access. 110b - Nonsecure privileged mode, instruction fetch access. 111b - Nonsecure privileged mode, data access.
7-4 —	Reserved
3-0 EDID	Error domain identifier This field captures the domain identifier of the access violation.

43.8.1.20 MRC Domain Error Word3 Register (MRC0_DERR_W3 - MRC1_DERR_W3)

Offset

Register	Offset
MRC0_DERR_W3	48Ch
MRC1_DERR_W3	49Ch

Function

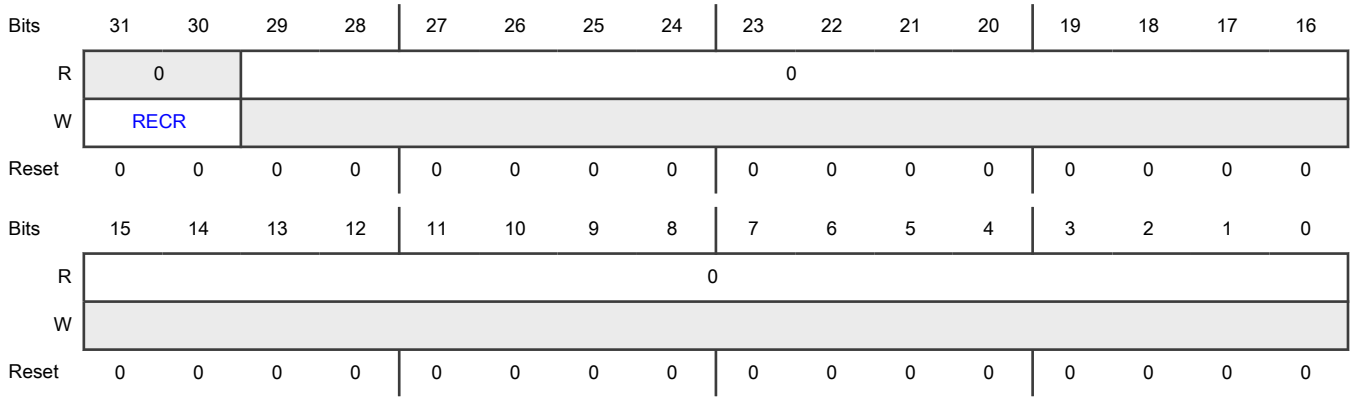
This register is used to rearm the error capture logic and clear the DERR_W0_i and DERR_W1_i registers. After the domain access violation error details have been read, typically in an exception service routine, a 32-bit word write to this register is required to rearm the error capture logic.

A read of this location returns zeroes. Attempted reads of a memory region checker (MRC) or Memory Block Checker (MBC) instance that is not physically present are error terminated.

See Domain error capture management for more details.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31-30 RECR	Rearm Error Capture Registers This 2-bit, write-only field controls the rearming of the domain error capture registers. Once an access violation has been detected with the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. Writing 01b to this field rearms the error capture mechanism and clears the DERR_W0_i and DERR_W1_i registers. A write of any value other than 01b has no effect.
29-0 —	Reserved

43.8.1.21 Process Identifier (PID0 - PID1)

Offset

Register	Offset
PID0	700h
PID1	704h

Function

In the TRDC's access control definition, each processor has a corresponding process identifier (PID) which performs the important functions: the entire value can be used to group tasks into different domains.

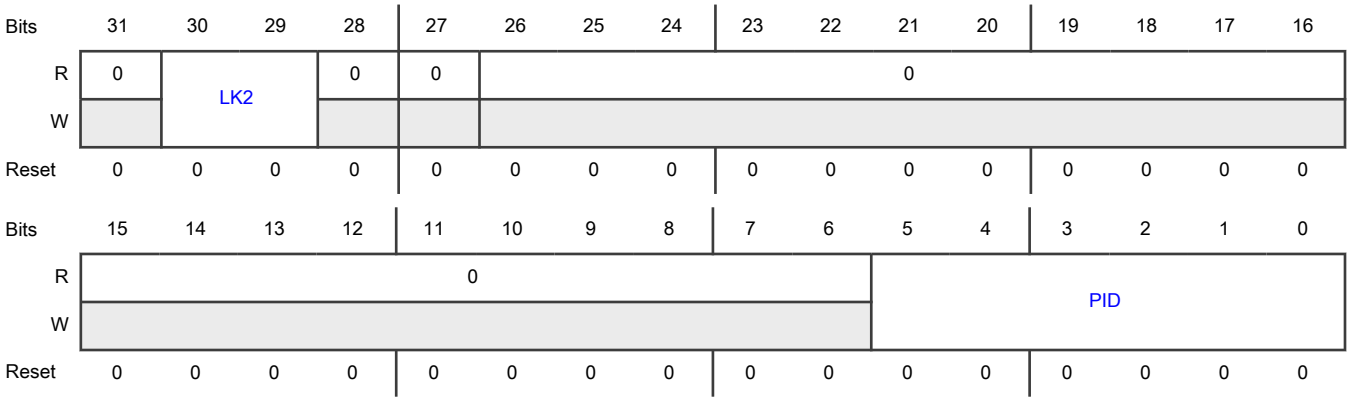
While certain processors include this register in their programming model definitions, others do not. This register is provided for those processors that do *not* include a PID register in their programming models. Secure privileged software saves and restores the PID as part of any context switch.

This data structure defines an array of 32-bit values, one per MDA module, that define the PID. Since this register resource is only applicable to processor cores, the data structure is typically sparsely populated. The HWCFG[2-3] registers provide a bitmap of the implemented PID*n* registers. This data structure is indexed using the corresponding MDA instance number.

Depending on the operating clock domain of each DAC instance, there may be optional information stored in the corresponding PIDm register to properly implement the LK2 = 2 functionality.

Reads of the PID*n* register return the contents of this register, or the PID*n* value directly sourced from a processor. For non-processor bus masters, this register does not exist and any attempted read is error terminated.

Diagram



Fields

Field	Function
31 —	Reserved
30-29 LK2	Lock This field provides a locking mechanism that can be used to limit the ability to write the register. Once set, these bits individually remain asserted until the next reset. If the processor sources its PIDm directly, that is, HWCFG{2-3}[PIDPm = 1], then privileged reads of this memory location return zero for this field. 00b - Register can be written by any secure privileged write. 01b - Register can be written by any secure privileged write. 10b - Register can only be written by a secure privileged write from the bus master that locked the register. 11b - Register is locked (read-only) until the next reset.
28 —	Reserved
27 —	Reserved
26-6 —	Reserved
5-0	Process identifier

Table continues on the next page...

Table continued from the previous page...

Field	Function
PID	This field determines the secure (0) or nonsecure (1) attribute for transactions associated with the corresponding processor. If the processor sources its PID*n* directly, then secure privileged reads of this memory location return the processor register for this field.

43.8.1.22 DAC Master Domain Assignment Register (MDA_W0_0_DFMT0 - MDA_W2_1_DFMT0)

Offset

Register	Offset
MDA_W0_0_DFMT0	800h
MDA_W0_1_DFMT0	820h
MDA_W1_1_DFMT0	824h
MDA_W2_1_DFMT0	828h

Function

The MDA_Wr_m registers provide a 2-dimensional data structure for assigning bus masters to domains. The number of implemented registers is defined by DACFGm[NMDAR]. This per-master domain assignment is then repeated for each bus master (MDAm). Thus, m specifies the master number and r refers to the specific MDA register for a given bus master.

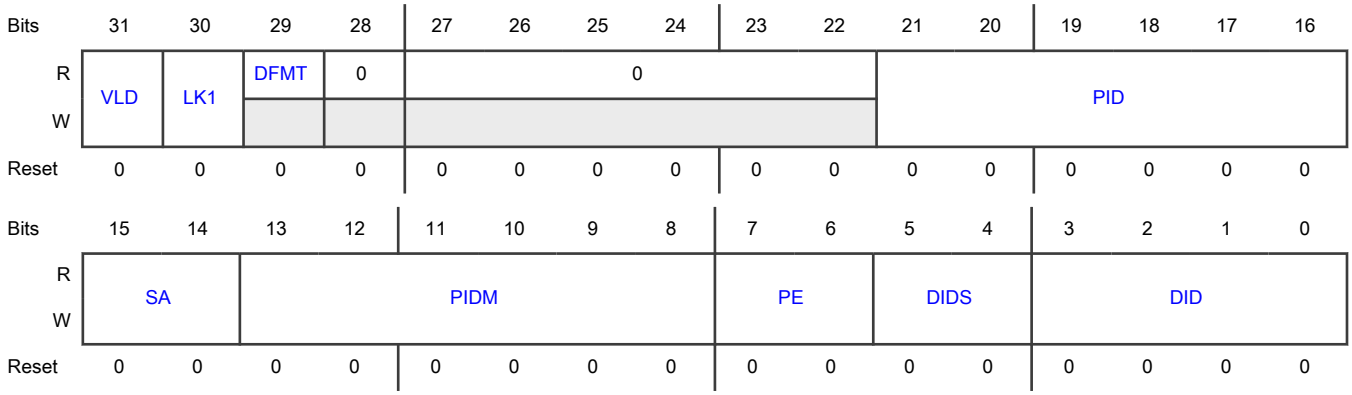
Each Wr within the MDAm structure is a word-sized definition; there are two formats supported, one for processor cores and another for non-processors. Processor masters typically support one or more Wr domain definitions, while non-processor masters support a single Wr. Processor masters Domain Assignment register names use a _DFMT0 suffix. Non-processor masters Domain Assignment register names use a _DFMT1 suffix.

The DAC submodule is responsible for the generation of domain identifiers for every transaction from every bus master. If there is a single Wr for a given master, then the specified domain identifier is used directly. If there are multiple Wr values for a given master, then the DAC evaluates the conditional terms to determine a “hit”. For all Wr hits, their corresponding domain identifiers are simply logically summed together (boolean OR). Use cases are typically expected to *hit in a single Wr* for a processor master. Special care is needed if *none* of the conditional terms hit in any Wr evaluation; for this case, the generated DID = 0 and software needs to be aware of any potential access rights granted for this DID.

Each MDA_Wr_m register has one of two programming models depending on the state of the domain format field, DFMT. The model described in this section is for DFMT = 0. This definition allows three different specifications of the DID for processors. The DFMT = 1 model is described in Master Domain Assignment (MDA_Wr_m_DFMT1) register.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/write.

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates the domain assignment is valid. It is further qualified by CR[GVLDM] = 1. If CR[GVLDM] is cleared, the DID output is defined by the SoC-specific *default DID* value. See the TRDC chip configuration section for more details. If both MDA_Wr_m[VLD] and CR[GVLDM] are asserted, the DID output is defined by the remaining contents of this register. 0b - The Wr domain assignment is invalid. 1b - The Wr domain assignment is valid.
30 LK1	1-bit Lock This field provides a locking mechanism that can be used to limit the ability to write the register. Once set, this bit remains asserted until the next reset. 0b - Register can be written by any secure privileged write. 1b - Register is locked (read-only) until the next reset.
29 DFMT	Domain format Identifies this register's domain assignment format. **NOTE** This bitfield access is ROZ 0b - Processor-core domain assignment 1b - Non-processor domain assignment
28 —	Reserved
27-22 —	Reserved
21-16	Process Identifier

Table continues on the next page...

Table continued from the previous page...

Field	Function
PID	This field specifies that the process identifier is to be combined with the PIDM field and included in the domain hit determination. The optional inclusion of the PID and PIDM is controlled by the MDA_Wr_m[PE] field.
15-14 SA	Secure attribute This field defines the secure/nonsecure attribute for processor cores. **NOTE** The bus master's input secure/nonsecure attribute is used if SA = 1X, or this VLD = 0. If TZM_ENB = 0, the master attribute is forced to nonsecure. A nonsecure write cannot program this field to 2'b00, which is a security level higher than the mode of the process that is writing it. Reset value of SA: • if TZM_ENB=0, SA reset value = 2'b01 • if TZM_ENB=1, SA reset value = 2'b00 00b - Force the bus attribute for this master to secure. 01b - Force the bus attribute for this master to nonsecure. 10b - Use the bus master's secure/nonsecure attribute directly. 11b - Use the bus master's secure/nonsecure attribute directly.
13-8 PIDM	Process Identifier Mask This field provides a masking capability so that multiple process identifiers can be included as part of the domain hit determination. If a bit in the PIDM is set, then the corresponding bit of the PID is ignored in the comparison. The optional inclusion of the PID and PIDM is controlled by the MDA_Wr_m[PE] field.
7-6 PE	Process identifier enable Controls the optional inclusion of the PID, qualified by PIDM, into the domain hit evaluation. It provides the ability to include inclusive or exclusive sets of masked PID values. 00b - No process identifier is included in the domain hit evaluation. 01b - No process identifier is included in the domain hit evaluation. 10b - PE = 2. The process identifier is included in the domain hit evaluation as defined by the expression: `partial_domain_hit = (PE == 2) && ((PID & ~PIDM) == (TRDC_PIDn[PID] & ~PIDM))` 11b - PE = 3. The process identifier is included in the domain hit evaluation as defined by the expression: `partial_domain_hit = (PE == 3) && ~((PID & ~PIDM) == (TRDC_PIDn[PID] & ~PIDM))`
5-4 DIDS	DID Select This field selects the source of the domain identifier. 00b - Use MDAm[3:0] as the domain identifier.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Use the input DID as the domain identifier. 10b - Use MDAm[3:2] concatenated with the low-order 2 bits of the input DID (DID_in[1:0]) as the domain identifier. 11b - Reserved for future use.
3-0 DID	Domain identifier This 4-bit field is the domain ID attribute that is sent on the accesses from the bus master connected to the DAC when DIDS=00b. DID[3:2] is used when DIDS=10b

43.8.1.23 DAC Master Domain Assignment Register (MDA_W0_2_DFMT1 - MDA_W0_3_DFMT1)

Offset

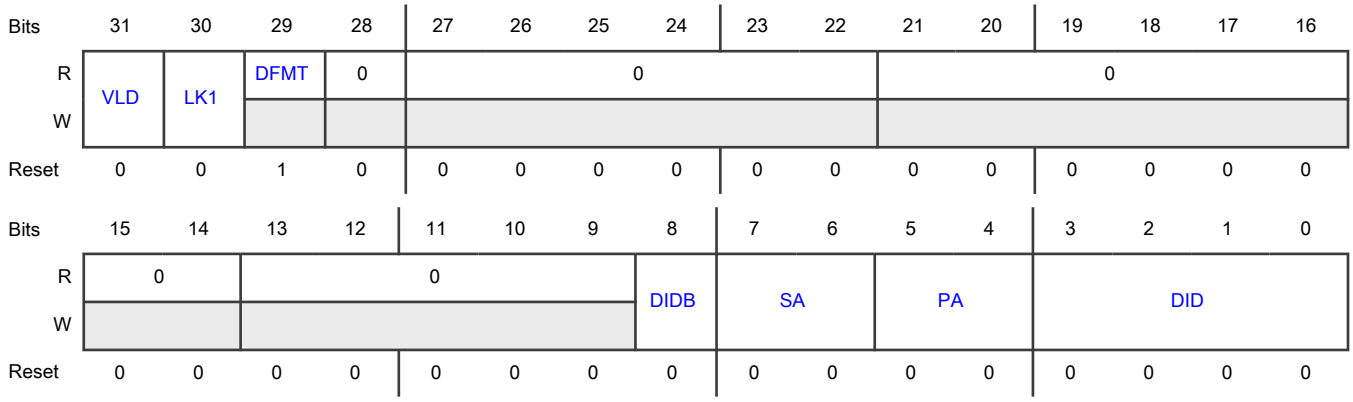
Register	Offset
MDA_W0_2_DFMT1	840h
MDA_W0_3_DFMT1	860h

Function

This register is identical to Master Domain Assignment (MDA_Wr_m_DFMT0) register except that the domain format field, DFMT, is 1 and the PID field is not used. This format supports two different specifications of the DID for non-core bus masters.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/write.

Diagram



Fields

Field	Function
31	Valid

Table continues on the next page...

Table continued from the previous page...

Field	Function
VLD	This field indicates the domain assignment is valid. It is further qualified by CR[GVLDM] = 1. If CR[GVLDM] is cleared, the DID output is defined by the SoC-specific *default DID* value. See the TRDC chip configuration section for more details. If both MDA_Wr_m[VLD] and CR[GVLDM] are asserted, the DID output is defined by the remaining contents of this register. 0b - The Wr domain assignment is invalid. 1b - The Wr domain assignment is valid.
30 LK1	1-bit Lock This field provides a locking mechanism that can be used to limit the ability to write the register. Once set, this bit remains asserted until the next reset. 0b - Register can be written by any secure privileged write. 1b - Register is locked (read-only) until the next reset.
29 DFMT	Domain format Identifies this register's domain assignment format. **NOTE** This bitfield access is ROO 0b - Processor-core domain assignment 1b - Non-processor domain assignment
28 —	Reserved
27-22 —	Reserved
21-16 —	Reserved
15-14 —	Reserved
13-9 —	Reserved
8 DIDB	DID Bypass If asserted, this bit enables the bypassing of an input DID value as the domain identifier for this non-processor bus master. This capability allows non-processor bus masters, for example, a DMA to masquerade as a processor. Once set, this field is “sticky” and remains set until the next reset.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>DAC</th><th>DID Input</th></tr>
<tr><td>0</td><td>0</td></tr>
<tr><td>1</td><td>DMA Domain ID</td></tr>
<tr><td>2</td><td>USB Domain ID</td></tr>
</table>
0b - Use MDAn[3:0] as the domain identifier.
1b - Use the DID input as the domain identifier.
</td></tr>
<tr><td>7-6
SA</td><td>Secure attribute
This field defines the secure/nonsecure attribute for non-processor cores.
NOTE
The bus master’s input secure/nonsecure attribute is used if SA = 1X, or this VLD = 0. If TZM_ENB = 0, the master attribute is forced to nonsecure. A nonsecure write cannot program this field to 2'b00, which is a security level higher than the mode of the process that is writing it.
Reset value of SA:
• if TZM_ENB=0, SA reset value = 2'b01
• if TZM_ENB=1, SA reset value = 2'b00
00b - Force the bus attribute for this master to secure.
01b - Force the bus attribute for this master to nonsecure.
10b - Use the bus master’s secure/nonsecure attribute directly.
11b - Use the bus master’s secure/nonsecure attribute directly.</td></tr>
<tr><td>5-4
PA</td><td>Privileged attribute
This field defines the privileged/user attribute for non-processor cores.
NOTE
The bus master’s input privileged/user attribute is used if PA = 1X, or this VLD = 0.
00b - Force the bus attribute for this master to user.
01b - Force the bus attribute for this master to privileged.
10b - Use the bus master’s privileged/user attribute directly.
11b - Use the bus master’s privileged/user attribute directly.</td></tr>
<tr><td>3-0
DID</td><td>Domain identifier
This 4-bit field is the domain ID attribute that is sent on the accesses from the bus master connected to the DAC when DIDB=0.</td></tr>
</table>

43.8.1.24 MBC Global Configuration Register (MBC0_MEM0_GLBCFG - MBC1_MEM3_GLBCFG)

Offset

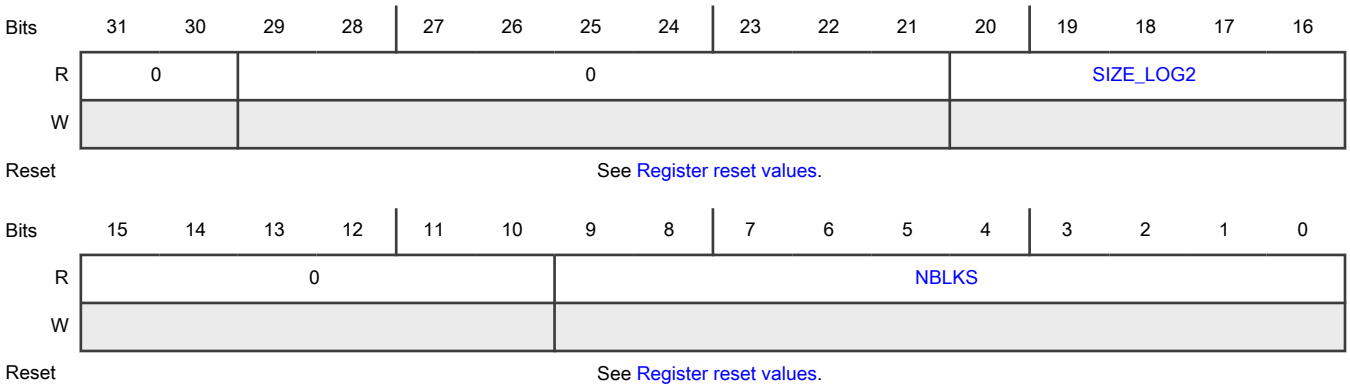
For m = 0 to 1; r = 0 to 3:

Register	Offset
MBCm_MEMr_GLBCFG	1_0000h + (m × 2000h) + (r × 4h)

Function

These MBC global configuration read-only registers contain information on the MBC's hardware configuration. Specifically, it defines the number of memory blocks and the size of each block in each MBC mem (r).

Diagram



Register reset values

Register	Reset value
MBC0_MEM0_GLBCFG	000C_0080h
MBC0_MEM1_GLBCFG	000C_0008h
MBC0_MEM2_GLBCFG–MBC0_MEM3_GLBCFG	000C_0001h
MBC1_MEM0_GLBCFG–MBC1_MEM1_GLBCFG	000C_0020h
MBC1_MEM2_GLBCFG–MBC1_MEM3_GLBCFG	000C_0000h

Fields

Field	Function
31-30 —	Reserved
29-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-16 SIZE_LOG2	Log2 size per block For example SIZE_LOG2=0x0C is 2^12=4 KB blocks.
15-10 —	Reserved
9-0 NBLKS	Number of blocks in this memory

43.8.1.25 MBC NonSecure Enable Block Index (MBC0_NSE_BLK_INDEX - MBC1_NSE_BLK_INDEX)

Offset

Register	Offset
MBC0_NSE_BLK_INDEX	1_0010h
MBC1_NSE_BLK_INDEX	1_2010h

Function

This R/W register defines the selected memories that are affected by the writes to the NSE_BLK_SET and NSE_BLK_CLR registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL9	DID_SEL8	DID_SEL7	DID_SEL6	DID_SEL5	DID_SEL4	DID_SEL3	DID_SEL2	DID_SEL1	DID_SEL0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				MEM_SEL				0		WNDX				0	AI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16	DID Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
DID_SELn	Destination domain bitmap select. 0b - No effect. 1b - Selects NSE bits for this domain.
15-12 —	Reserved
11-8 MEM_SEL	Memory Select Destination memory bitmap select. • Bit [11] - MBC MEM 3 • Bit [10] - MBC MEM 2 • Bit [9] - MBC MEM 1 • Bit [8] - MBC MEM 0
7-6 —	Reserved
5-2 WNDX	Word index into the block NSE bitmap. It selects the BLK_NSE_Wn register, where WNDX determines the value of n.
1 —	Reserved
0 AI	Auto Increment 0b - No effect. 1b - Add 1 to the WNDX field after the register write.

43.8.1.26 MBC NonSecure Enable Block Set (MBC0_NSE_BLK_SET - MBC1_NSE_BLK_SET)

Offset

Register	Offset
MBC0_NSE_BLK_SET	1_0014h
MBC1_NSE_BLK_SET	1_2014h

Function

A write to this register sets the appropriate NSE Bits for the selected domains and memories defined at the word location NSE_BLK_INDEX[WDNX]. If NSE_BLK_INDEX[AI] = 1, then the NSE_BLK_INDEX[WDNX] field is incremented by 1 (modulo-16) after the write is completed. This register reads as zero.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	W1SET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	W1SET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 W1SET	Write-1 Set If set to 1, sets appropriate NSE bit for the selected domains and memories defined at the word location NSE_BLK_INDEX[WDNX]. Write with value 0 has no effect.

43.8.1.27 MBC NonSecure Enable Block Clear (MBC0_NSE_BLK_CLR - MBC1_NSE_BLK_CLR)

Offset

Register	Offset
MBC0_NSE_BLK_CLR	1_0018h
MBC1_NSE_BLK_CLR	1_2018h

Function

A write to this location clears the appropriate NSE bits as defined by the W1CLR[n]=1 for the selected domains and memories Defined at the word location NSE_BLK_INDEX[WDNX]. If NSE_BLK_INDEX[AI] = 1, then the NSE_BLK_INDEX[WDNX] field is incremented by 1 (modulo-16) after the write is completed. This register reads as zero.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	W1CLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	W1CLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 W1CLR	Write-1 Clear If set to 1, Clear the appropriate NSE bit for the selected domains and memories defined at the word location NSE_BLK_INDEX[WDNX]. Write with value 0 has no effect.

43.8.1.28 MBC NonSecure Enable Block Clear All (MBC0_NSE_BLK_CLR_ALL - MBC1_NSE_BLK_CLR_ALL)

Offset

Register	Offset
MBC0_NSE_BLK_CLR_ALL	1_001Ch
MBC1_NSE_BLK_CLR_ALL	1_201Ch

Function

A write to this location clears all the block NSE bits for the selected domains and memories defined in the write data. This register reads as zero.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL9	DID_SEL8	DID_SEL7	DID_SEL6	DID_SEL5	DID_SEL4	DID_SEL3	DID_SEL2	DID_SEL1	DID_SEL0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0							
W					MEMSEL											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 DID_SELn	DID Select Destination domain bitmap select. 0b - No effect. 1b - Clear all NSE bits for this domain.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11-8 MEMSEL	Memory Select Destination memory bitmap select. • Bit [11] - clear all MEM 3 NSE bits • Bit [10] - clear all MEM 2 NSE bits • Bit [9] - clear all MEM 1 NSE bits • Bit [8] - clear all MEM 0 NSE bits
7-0 —	Reserved

43.8.1.29 MBC Global Access Control (MBC0_MEMN_GLBAC0)

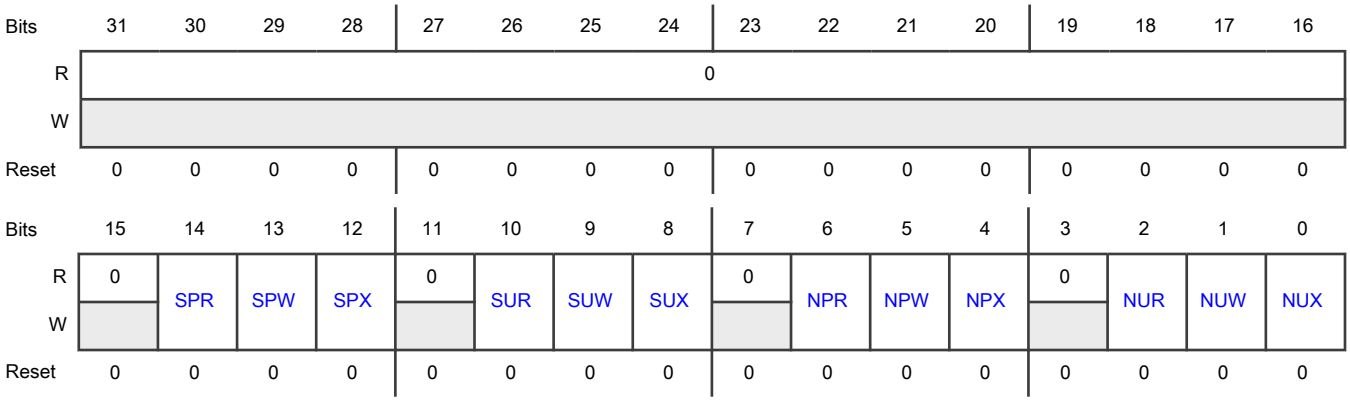
Offset

Register	Offset
MBC0_MEMN_GLBAC0	1_0020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6	NonsecurePriv Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
NPR	NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.1.30 MBC Global Access Control (MBC0_MEMN_GLBAC1 - MBC0_MEMN_GLBAC7)

Offset

Register	Offset
MBC0_MEMN_GLBAC1	1_0024h

Table continues on the next page...

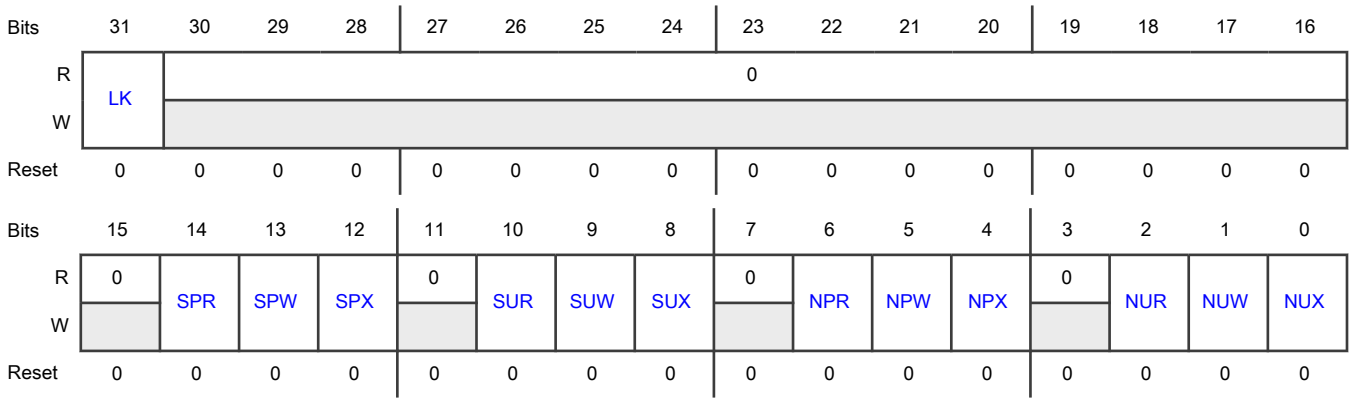
Table continued from the previous page...

Register	Offset
MBC0_MEMN_GLBAC2	1_0028h
MBC0_MEMN_GLBAC3	1_002Ch
MBC0_MEMN_GLBAC4	1_0030h
MBC0_MEMN_GLBAC5	1_0034h
MBC0_MEMN_GLBAC6	1_0038h
MBC0_MEMN_GLBAC7	1_003Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these control access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked and cannot be altered.
30-15 —	Reserved
14	SecurePriv Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
SPR	SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.1.31 MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W0 - MBC1_DOM15_MEM1_BLK_CFG_W3)

Offset

Register	Offset
MBC0_DOM0_MEM0_BLK_CFG_W0	1_0040h
MBC0_DOM0_MEM0_BLK_CFG_W1	1_0044h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM0_MEM0_BLK_CFG_W2	1_0048h
MBC0_DOM0_MEM0_BLK_CFG_W3	1_004Ch
MBC0_DOM0_MEM0_BLK_CFG_W4	1_0050h
MBC0_DOM0_MEM0_BLK_CFG_W5	1_0054h
MBC0_DOM0_MEM0_BLK_CFG_W6	1_0058h
MBC0_DOM0_MEM0_BLK_CFG_W7	1_005Ch
MBC0_DOM0_MEM0_BLK_CFG_W8	1_0060h
MBC0_DOM0_MEM0_BLK_CFG_W9	1_0064h
MBC0_DOM0_MEM0_BLK_CFG_W10	1_0068h
MBC0_DOM0_MEM0_BLK_CFG_W11	1_006Ch
MBC0_DOM0_MEM0_BLK_CFG_W12	1_0070h
MBC0_DOM0_MEM0_BLK_CFG_W13	1_0074h
MBC0_DOM0_MEM0_BLK_CFG_W14	1_0078h
MBC0_DOM0_MEM0_BLK_CFG_W15	1_007Ch
MBC0_DOM0_MEM1_BLK_CFG_W0	1_0180h
MBC0_DOM0_MEM2_BLK_CFG_W0	1_01A8h
MBC0_DOM0_MEM3_BLK_CFG_W0	1_01D0h
MBC0_DOM1_MEM0_BLK_CFG_W0	1_0240h
MBC0_DOM1_MEM0_BLK_CFG_W1	1_0244h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM1_MEM0_BLK_CFG_W2	1_0248h
MBC0_DOM1_MEM0_BLK_CFG_W3	1_024Ch
MBC0_DOM1_MEM0_BLK_CFG_W4	1_0250h
MBC0_DOM1_MEM0_BLK_CFG_W5	1_0254h
MBC0_DOM1_MEM0_BLK_CFG_W6	1_0258h
MBC0_DOM1_MEM0_BLK_CFG_W7	1_025Ch
MBC0_DOM1_MEM0_BLK_CFG_W8	1_0260h
MBC0_DOM1_MEM0_BLK_CFG_W9	1_0264h
MBC0_DOM1_MEM0_BLK_CFG_W10	1_0268h
MBC0_DOM1_MEM0_BLK_CFG_W11	1_026Ch
MBC0_DOM1_MEM0_BLK_CFG_W12	1_0270h
MBC0_DOM1_MEM0_BLK_CFG_W13	1_0274h
MBC0_DOM1_MEM0_BLK_CFG_W14	1_0278h
MBC0_DOM1_MEM0_BLK_CFG_W15	1_027Ch
MBC0_DOM1_MEM1_BLK_CFG_W0	1_0380h
MBC0_DOM1_MEM2_BLK_CFG_W0	1_03A8h
MBC0_DOM1_MEM3_BLK_CFG_W0	1_03D0h
MBC0_DOM2_MEM0_BLK_CFG_W0	1_0440h
MBC0_DOM2_MEM0_BLK_CFG_W1	1_0444h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM2_MEM0_BLK_CFG_W2	1_0448h
MBC0_DOM2_MEM0_BLK_CFG_W3	1_044Ch
MBC0_DOM2_MEM0_BLK_CFG_W4	1_0450h
MBC0_DOM2_MEM0_BLK_CFG_W5	1_0454h
MBC0_DOM2_MEM0_BLK_CFG_W6	1_0458h
MBC0_DOM2_MEM0_BLK_CFG_W7	1_045Ch
MBC0_DOM2_MEM0_BLK_CFG_W8	1_0460h
MBC0_DOM2_MEM0_BLK_CFG_W9	1_0464h
MBC0_DOM2_MEM0_BLK_CFG_W10	1_0468h
MBC0_DOM2_MEM0_BLK_CFG_W11	1_046Ch
MBC0_DOM2_MEM0_BLK_CFG_W12	1_0470h
MBC0_DOM2_MEM0_BLK_CFG_W13	1_0474h
MBC0_DOM2_MEM0_BLK_CFG_W14	1_0478h
MBC0_DOM2_MEM0_BLK_CFG_W15	1_047Ch
MBC0_DOM2_MEM1_BLK_CFG_W0	1_0580h
MBC0_DOM2_MEM2_BLK_CFG_W0	1_05A8h
MBC0_DOM2_MEM3_BLK_CFG_W0	1_05D0h
MBC0_DOM3_MEM0_BLK_CFG_W0	1_0640h
MBC0_DOM3_MEM0_BLK_CFG_W1	1_0644h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM3_MEM0_BLK_CFG_W2	1_0648h
MBC0_DOM3_MEM0_BLK_CFG_W3	1_064Ch
MBC0_DOM3_MEM0_BLK_CFG_W4	1_0650h
MBC0_DOM3_MEM0_BLK_CFG_W5	1_0654h
MBC0_DOM3_MEM0_BLK_CFG_W6	1_0658h
MBC0_DOM3_MEM0_BLK_CFG_W7	1_065Ch
MBC0_DOM3_MEM0_BLK_CFG_W8	1_0660h
MBC0_DOM3_MEM0_BLK_CFG_W9	1_0664h
MBC0_DOM3_MEM0_BLK_CFG_W10	1_0668h
MBC0_DOM3_MEM0_BLK_CFG_W11	1_066Ch
MBC0_DOM3_MEM0_BLK_CFG_W12	1_0670h
MBC0_DOM3_MEM0_BLK_CFG_W13	1_0674h
MBC0_DOM3_MEM0_BLK_CFG_W14	1_0678h
MBC0_DOM3_MEM0_BLK_CFG_W15	1_067Ch
MBC0_DOM3_MEM1_BLK_CFG_W0	1_0780h
MBC0_DOM3_MEM2_BLK_CFG_W0	1_07A8h
MBC0_DOM3_MEM3_BLK_CFG_W0	1_07D0h
MBC0_DOM4_MEM0_BLK_CFG_W0	1_0840h
MBC0_DOM4_MEM0_BLK_CFG_W1	1_0844h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM4_MEM0_BLK_CFG_W2	1_0848h
MBC0_DOM4_MEM0_BLK_CFG_W3	1_084Ch
MBC0_DOM4_MEM0_BLK_CFG_W4	1_0850h
MBC0_DOM4_MEM0_BLK_CFG_W5	1_0854h
MBC0_DOM4_MEM0_BLK_CFG_W6	1_0858h
MBC0_DOM4_MEM0_BLK_CFG_W7	1_085Ch
MBC0_DOM4_MEM0_BLK_CFG_W8	1_0860h
MBC0_DOM4_MEM0_BLK_CFG_W9	1_0864h
MBC0_DOM4_MEM0_BLK_CFG_W10	1_0868h
MBC0_DOM4_MEM0_BLK_CFG_W11	1_086Ch
MBC0_DOM4_MEM0_BLK_CFG_W12	1_0870h
MBC0_DOM4_MEM0_BLK_CFG_W13	1_0874h
MBC0_DOM4_MEM0_BLK_CFG_W14	1_0878h
MBC0_DOM4_MEM0_BLK_CFG_W15	1_087Ch
MBC0_DOM4_MEM1_BLK_CFG_W0	1_0980h
MBC0_DOM4_MEM2_BLK_CFG_W0	1_09A8h
MBC0_DOM4_MEM3_BLK_CFG_W0	1_09D0h
MBC0_DOM5_MEM0_BLK_CFG_W0	1_0A40h
MBC0_DOM5_MEM0_BLK_CFG_W1	1_0A44h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM5_MEM0_BLK_CFG_W2	1_0A48h
MBC0_DOM5_MEM0_BLK_CFG_W3	1_0A4Ch
MBC0_DOM5_MEM0_BLK_CFG_W4	1_0A50h
MBC0_DOM5_MEM0_BLK_CFG_W5	1_0A54h
MBC0_DOM5_MEM0_BLK_CFG_W6	1_0A58h
MBC0_DOM5_MEM0_BLK_CFG_W7	1_0A5Ch
MBC0_DOM5_MEM0_BLK_CFG_W8	1_0A60h
MBC0_DOM5_MEM0_BLK_CFG_W9	1_0A64h
MBC0_DOM5_MEM0_BLK_CFG_W10	1_0A68h
MBC0_DOM5_MEM0_BLK_CFG_W11	1_0A6Ch
MBC0_DOM5_MEM0_BLK_CFG_W12	1_0A70h
MBC0_DOM5_MEM0_BLK_CFG_W13	1_0A74h
MBC0_DOM5_MEM0_BLK_CFG_W14	1_0A78h
MBC0_DOM5_MEM0_BLK_CFG_W15	1_0A7Ch
MBC0_DOM5_MEM1_BLK_CFG_W0	1_0B80h
MBC0_DOM5_MEM2_BLK_CFG_W0	1_0BA8h
MBC0_DOM5_MEM3_BLK_CFG_W0	1_0BD0h
MBC0_DOM6_MEM0_BLK_CFG_W0	1_0C40h
MBC0_DOM6_MEM0_BLK_CFG_W1	1_0C44h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM6_MEM0_BLK_CFG_W2	1_0C48h
MBC0_DOM6_MEM0_BLK_CFG_W3	1_0C4Ch
MBC0_DOM6_MEM0_BLK_CFG_W4	1_0C50h
MBC0_DOM6_MEM0_BLK_CFG_W5	1_0C54h
MBC0_DOM6_MEM0_BLK_CFG_W6	1_0C58h
MBC0_DOM6_MEM0_BLK_CFG_W7	1_0C5Ch
MBC0_DOM6_MEM0_BLK_CFG_W8	1_0C60h
MBC0_DOM6_MEM0_BLK_CFG_W9	1_0C64h
MBC0_DOM6_MEM0_BLK_CFG_W10	1_0C68h
MBC0_DOM6_MEM0_BLK_CFG_W11	1_0C6Ch
MBC0_DOM6_MEM0_BLK_CFG_W12	1_0C70h
MBC0_DOM6_MEM0_BLK_CFG_W13	1_0C74h
MBC0_DOM6_MEM0_BLK_CFG_W14	1_0C78h
MBC0_DOM6_MEM0_BLK_CFG_W15	1_0C7Ch
MBC0_DOM6_MEM1_BLK_CFG_W0	1_0D80h
MBC0_DOM6_MEM2_BLK_CFG_W0	1_0DA8h
MBC0_DOM6_MEM3_BLK_CFG_W0	1_0DD0h
MBC0_DOM7_MEM0_BLK_CFG_W0	1_0E40h
MBC0_DOM7_MEM0_BLK_CFG_W1	1_0E44h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM7_MEM0_BLK_CFG_W2	1_0E48h
MBC0_DOM7_MEM0_BLK_CFG_W3	1_0E4Ch
MBC0_DOM7_MEM0_BLK_CFG_W4	1_0E50h
MBC0_DOM7_MEM0_BLK_CFG_W5	1_0E54h
MBC0_DOM7_MEM0_BLK_CFG_W6	1_0E58h
MBC0_DOM7_MEM0_BLK_CFG_W7	1_0E5Ch
MBC0_DOM7_MEM0_BLK_CFG_W8	1_0E60h
MBC0_DOM7_MEM0_BLK_CFG_W9	1_0E64h
MBC0_DOM7_MEM0_BLK_CFG_W10	1_0E68h
MBC0_DOM7_MEM0_BLK_CFG_W11	1_0E6Ch
MBC0_DOM7_MEM0_BLK_CFG_W12	1_0E70h
MBC0_DOM7_MEM0_BLK_CFG_W13	1_0E74h
MBC0_DOM7_MEM0_BLK_CFG_W14	1_0E78h
MBC0_DOM7_MEM0_BLK_CFG_W15	1_0E7Ch
MBC0_DOM7_MEM1_BLK_CFG_W0	1_0F80h
MBC0_DOM7_MEM2_BLK_CFG_W0	1_0FA8h
MBC0_DOM7_MEM3_BLK_CFG_W0	1_0FD0h
MBC0_DOM8_MEM0_BLK_CFG_W0	1_1040h
MBC0_DOM8_MEM0_BLK_CFG_W1	1_1044h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM8_MEM0_BLK_CFG_W2	1_1048h
MBC0_DOM8_MEM0_BLK_CFG_W3	1_104Ch
MBC0_DOM8_MEM0_BLK_CFG_W4	1_1050h
MBC0_DOM8_MEM0_BLK_CFG_W5	1_1054h
MBC0_DOM8_MEM0_BLK_CFG_W6	1_1058h
MBC0_DOM8_MEM0_BLK_CFG_W7	1_105Ch
MBC0_DOM8_MEM0_BLK_CFG_W8	1_1060h
MBC0_DOM8_MEM0_BLK_CFG_W9	1_1064h
MBC0_DOM8_MEM0_BLK_CFG_W10	1_1068h
MBC0_DOM8_MEM0_BLK_CFG_W11	1_106Ch
MBC0_DOM8_MEM0_BLK_CFG_W12	1_1070h
MBC0_DOM8_MEM0_BLK_CFG_W13	1_1074h
MBC0_DOM8_MEM0_BLK_CFG_W14	1_1078h
MBC0_DOM8_MEM0_BLK_CFG_W15	1_107Ch
MBC0_DOM8_MEM1_BLK_CFG_W0	1_1180h
MBC0_DOM8_MEM2_BLK_CFG_W0	1_11A8h
MBC0_DOM8_MEM3_BLK_CFG_W0	1_11D0h
MBC0_DOM9_MEM0_BLK_CFG_W0	1_1240h
MBC0_DOM9_MEM0_BLK_CFG_W1	1_1244h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM9_MEM0_BLK_CFG_W2	1_1248h
MBC0_DOM9_MEM0_BLK_CFG_W3	1_124Ch
MBC0_DOM9_MEM0_BLK_CFG_W4	1_1250h
MBC0_DOM9_MEM0_BLK_CFG_W5	1_1254h
MBC0_DOM9_MEM0_BLK_CFG_W6	1_1258h
MBC0_DOM9_MEM0_BLK_CFG_W7	1_125Ch
MBC0_DOM9_MEM0_BLK_CFG_W8	1_1260h
MBC0_DOM9_MEM0_BLK_CFG_W9	1_1264h
MBC0_DOM9_MEM0_BLK_CFG_W10	1_1268h
MBC0_DOM9_MEM0_BLK_CFG_W11	1_126Ch
MBC0_DOM9_MEM0_BLK_CFG_W12	1_1270h
MBC0_DOM9_MEM0_BLK_CFG_W13	1_1274h
MBC0_DOM9_MEM0_BLK_CFG_W14	1_1278h
MBC0_DOM9_MEM0_BLK_CFG_W15	1_127Ch
MBC0_DOM9_MEM1_BLK_CFG_W0	1_1380h
MBC0_DOM9_MEM2_BLK_CFG_W0	1_13A8h
MBC0_DOM9_MEM3_BLK_CFG_W0	1_13D0h
MBC0_DOM10_MEM0_BLK_CFG_W0	1_1440h
MBC0_DOM10_MEM0_BLK_CFG_W1	1_1444h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM10_MEM0_BLK_CFG_W2	1_1448h
MBC0_DOM10_MEM0_BLK_CFG_W3	1_144Ch
MBC0_DOM10_MEM0_BLK_CFG_W4	1_1450h
MBC0_DOM10_MEM0_BLK_CFG_W5	1_1454h
MBC0_DOM10_MEM0_BLK_CFG_W6	1_1458h
MBC0_DOM10_MEM0_BLK_CFG_W7	1_145Ch
MBC0_DOM10_MEM0_BLK_CFG_W8	1_1460h
MBC0_DOM10_MEM0_BLK_CFG_W9	1_1464h
MBC0_DOM10_MEM0_BLK_CFG_W10	1_1468h
MBC0_DOM10_MEM0_BLK_CFG_W11	1_146Ch
MBC0_DOM10_MEM0_BLK_CFG_W12	1_1470h
MBC0_DOM10_MEM0_BLK_CFG_W13	1_1474h
MBC0_DOM10_MEM0_BLK_CFG_W14	1_1478h
MBC0_DOM10_MEM0_BLK_CFG_W15	1_147Ch
MBC0_DOM10_MEM1_BLK_CFG_W0	1_1580h
MBC0_DOM10_MEM2_BLK_CFG_W0	1_15A8h
MBC0_DOM10_MEM3_BLK_CFG_W0	1_15D0h
MBC0_DOM11_MEM0_BLK_CFG_W0	1_1640h
MBC0_DOM11_MEM0_BLK_CFG_W1	1_1644h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM11_MEM0_BLK_CFG_W2	1_1648h
MBC0_DOM11_MEM0_BLK_CFG_W3	1_164Ch
MBC0_DOM11_MEM0_BLK_CFG_W4	1_1650h
MBC0_DOM11_MEM0_BLK_CFG_W5	1_1654h
MBC0_DOM11_MEM0_BLK_CFG_W6	1_1658h
MBC0_DOM11_MEM0_BLK_CFG_W7	1_165Ch
MBC0_DOM11_MEM0_BLK_CFG_W8	1_1660h
MBC0_DOM11_MEM0_BLK_CFG_W9	1_1664h
MBC0_DOM11_MEM0_BLK_CFG_W10	1_1668h
MBC0_DOM11_MEM0_BLK_CFG_W11	1_166Ch
MBC0_DOM11_MEM0_BLK_CFG_W12	1_1670h
MBC0_DOM11_MEM0_BLK_CFG_W13	1_1674h
MBC0_DOM11_MEM0_BLK_CFG_W14	1_1678h
MBC0_DOM11_MEM0_BLK_CFG_W15	1_167Ch
MBC0_DOM11_MEM1_BLK_CFG_W0	1_1780h
MBC0_DOM11_MEM2_BLK_CFG_W0	1_17A8h
MBC0_DOM11_MEM3_BLK_CFG_W0	1_17D0h
MBC0_DOM12_MEM0_BLK_CFG_W0	1_1840h
MBC0_DOM12_MEM0_BLK_CFG_W1	1_1844h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM12_MEM0_BLK_CFG_W2	1_1848h
MBC0_DOM12_MEM0_BLK_CFG_W3	1_184Ch
MBC0_DOM12_MEM0_BLK_CFG_W4	1_1850h
MBC0_DOM12_MEM0_BLK_CFG_W5	1_1854h
MBC0_DOM12_MEM0_BLK_CFG_W6	1_1858h
MBC0_DOM12_MEM0_BLK_CFG_W7	1_185Ch
MBC0_DOM12_MEM0_BLK_CFG_W8	1_1860h
MBC0_DOM12_MEM0_BLK_CFG_W9	1_1864h
MBC0_DOM12_MEM0_BLK_CFG_W10	1_1868h
MBC0_DOM12_MEM0_BLK_CFG_W11	1_186Ch
MBC0_DOM12_MEM0_BLK_CFG_W12	1_1870h
MBC0_DOM12_MEM0_BLK_CFG_W13	1_1874h
MBC0_DOM12_MEM0_BLK_CFG_W14	1_1878h
MBC0_DOM12_MEM0_BLK_CFG_W15	1_187Ch
MBC0_DOM12_MEM1_BLK_CFG_W0	1_1980h
MBC0_DOM12_MEM2_BLK_CFG_W0	1_19A8h
MBC0_DOM12_MEM3_BLK_CFG_W0	1_19D0h
MBC0_DOM13_MEM0_BLK_CFG_W0	1_1A40h
MBC0_DOM13_MEM0_BLK_CFG_W1	1_1A44h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM13_MEM0_BLK_CFG_W2	1_1A48h
MBC0_DOM13_MEM0_BLK_CFG_W3	1_1A4Ch
MBC0_DOM13_MEM0_BLK_CFG_W4	1_1A50h
MBC0_DOM13_MEM0_BLK_CFG_W5	1_1A54h
MBC0_DOM13_MEM0_BLK_CFG_W6	1_1A58h
MBC0_DOM13_MEM0_BLK_CFG_W7	1_1A5Ch
MBC0_DOM13_MEM0_BLK_CFG_W8	1_1A60h
MBC0_DOM13_MEM0_BLK_CFG_W9	1_1A64h
MBC0_DOM13_MEM0_BLK_CFG_W10	1_1A68h
MBC0_DOM13_MEM0_BLK_CFG_W11	1_1A6Ch
MBC0_DOM13_MEM0_BLK_CFG_W12	1_1A70h
MBC0_DOM13_MEM0_BLK_CFG_W13	1_1A74h
MBC0_DOM13_MEM0_BLK_CFG_W14	1_1A78h
MBC0_DOM13_MEM0_BLK_CFG_W15	1_1A7Ch
MBC0_DOM13_MEM1_BLK_CFG_W0	1_1B80h
MBC0_DOM13_MEM2_BLK_CFG_W0	1_1BA8h
MBC0_DOM13_MEM3_BLK_CFG_W0	1_1BD0h
MBC0_DOM14_MEM0_BLK_CFG_W0	1_1C40h
MBC0_DOM14_MEM0_BLK_CFG_W1	1_1C44h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM14_MEM0_BLK_CFG_W2	1_1C48h
MBC0_DOM14_MEM0_BLK_CFG_W3	1_1C4Ch
MBC0_DOM14_MEM0_BLK_CFG_W4	1_1C50h
MBC0_DOM14_MEM0_BLK_CFG_W5	1_1C54h
MBC0_DOM14_MEM0_BLK_CFG_W6	1_1C58h
MBC0_DOM14_MEM0_BLK_CFG_W7	1_1C5Ch
MBC0_DOM14_MEM0_BLK_CFG_W8	1_1C60h
MBC0_DOM14_MEM0_BLK_CFG_W9	1_1C64h
MBC0_DOM14_MEM0_BLK_CFG_W10	1_1C68h
MBC0_DOM14_MEM0_BLK_CFG_W11	1_1C6Ch
MBC0_DOM14_MEM0_BLK_CFG_W12	1_1C70h
MBC0_DOM14_MEM0_BLK_CFG_W13	1_1C74h
MBC0_DOM14_MEM0_BLK_CFG_W14	1_1C78h
MBC0_DOM14_MEM0_BLK_CFG_W15	1_1C7Ch
MBC0_DOM14_MEM1_BLK_CFG_W0	1_1D80h
MBC0_DOM14_MEM2_BLK_CFG_W0	1_1DA8h
MBC0_DOM14_MEM3_BLK_CFG_W0	1_1DD0h
MBC0_DOM15_MEM0_BLK_CFG_W0	1_1E40h
MBC0_DOM15_MEM0_BLK_CFG_W1	1_1E44h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM15_MEM0_BLK_CFG_W2	1_1E48h
MBC0_DOM15_MEM0_BLK_CFG_W3	1_1E4Ch
MBC0_DOM15_MEM0_BLK_CFG_W4	1_1E50h
MBC0_DOM15_MEM0_BLK_CFG_W5	1_1E54h
MBC0_DOM15_MEM0_BLK_CFG_W6	1_1E58h
MBC0_DOM15_MEM0_BLK_CFG_W7	1_1E5Ch
MBC0_DOM15_MEM0_BLK_CFG_W8	1_1E60h
MBC0_DOM15_MEM0_BLK_CFG_W9	1_1E64h
MBC0_DOM15_MEM0_BLK_CFG_W10	1_1E68h
MBC0_DOM15_MEM0_BLK_CFG_W11	1_1E6Ch
MBC0_DOM15_MEM0_BLK_CFG_W12	1_1E70h
MBC0_DOM15_MEM0_BLK_CFG_W13	1_1E74h
MBC0_DOM15_MEM0_BLK_CFG_W14	1_1E78h
MBC0_DOM15_MEM0_BLK_CFG_W15	1_1E7Ch
MBC0_DOM15_MEM1_BLK_CFG_W0	1_1F80h
MBC0_DOM15_MEM2_BLK_CFG_W0	1_1FA8h
MBC0_DOM15_MEM3_BLK_CFG_W0	1_1FD0h
MBC1_DOM0_MEM0_BLK_CFG_W0	1_2040h
MBC1_DOM0_MEM0_BLK_CFG_W1	1_2044h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM0_MEM0_BLK_CFG_W2	1_2048h
MBC1_DOM0_MEM0_BLK_CFG_W3	1_204Ch
MBC1_DOM0_MEM1_BLK_CFG_W0	1_2180h
MBC1_DOM0_MEM1_BLK_CFG_W1	1_2184h
MBC1_DOM0_MEM1_BLK_CFG_W2	1_2188h
MBC1_DOM0_MEM1_BLK_CFG_W3	1_218Ch
MBC1_DOM1_MEM0_BLK_CFG_W0	1_2240h
MBC1_DOM1_MEM0_BLK_CFG_W1	1_2244h
MBC1_DOM1_MEM0_BLK_CFG_W2	1_2248h
MBC1_DOM1_MEM0_BLK_CFG_W3	1_224Ch
MBC1_DOM1_MEM1_BLK_CFG_W0	1_2380h
MBC1_DOM1_MEM1_BLK_CFG_W1	1_2384h
MBC1_DOM1_MEM1_BLK_CFG_W2	1_2388h
MBC1_DOM1_MEM1_BLK_CFG_W3	1_238Ch
MBC1_DOM2_MEM0_BLK_CFG_W0	1_2440h
MBC1_DOM2_MEM0_BLK_CFG_W1	1_2444h
MBC1_DOM2_MEM0_BLK_CFG_W2	1_2448h
MBC1_DOM2_MEM0_BLK_CFG_W3	1_244Ch
MBC1_DOM2_MEM1_BLK_CFG_W0	1_2580h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM2_MEM1_BLK_CFG_W1	1_2584h
MBC1_DOM2_MEM1_BLK_CFG_W2	1_2588h
MBC1_DOM2_MEM1_BLK_CFG_W3	1_258Ch
MBC1_DOM3_MEM0_BLK_CFG_W0	1_2640h
MBC1_DOM3_MEM0_BLK_CFG_W1	1_2644h
MBC1_DOM3_MEM0_BLK_CFG_W2	1_2648h
MBC1_DOM3_MEM0_BLK_CFG_W3	1_264Ch
MBC1_DOM3_MEM1_BLK_CFG_W0	1_2780h
MBC1_DOM3_MEM1_BLK_CFG_W1	1_2784h
MBC1_DOM3_MEM1_BLK_CFG_W2	1_2788h
MBC1_DOM3_MEM1_BLK_CFG_W3	1_278Ch
MBC1_DOM4_MEM0_BLK_CFG_W0	1_2840h
MBC1_DOM4_MEM0_BLK_CFG_W1	1_2844h
MBC1_DOM4_MEM0_BLK_CFG_W2	1_2848h
MBC1_DOM4_MEM0_BLK_CFG_W3	1_284Ch
MBC1_DOM4_MEM1_BLK_CFG_W0	1_2980h
MBC1_DOM4_MEM1_BLK_CFG_W1	1_2984h
MBC1_DOM4_MEM1_BLK_CFG_W2	1_2988h
MBC1_DOM4_MEM1_BLK_CFG_W3	1_298Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM5_MEM0_BLK_CFG_W0	1_2A40h
MBC1_DOM5_MEM0_BLK_CFG_W1	1_2A44h
MBC1_DOM5_MEM0_BLK_CFG_W2	1_2A48h
MBC1_DOM5_MEM0_BLK_CFG_W3	1_2A4Ch
MBC1_DOM5_MEM1_BLK_CFG_W0	1_2B80h
MBC1_DOM5_MEM1_BLK_CFG_W1	1_2B84h
MBC1_DOM5_MEM1_BLK_CFG_W2	1_2B88h
MBC1_DOM5_MEM1_BLK_CFG_W3	1_2B8Ch
MBC1_DOM6_MEM0_BLK_CFG_W0	1_2C40h
MBC1_DOM6_MEM0_BLK_CFG_W1	1_2C44h
MBC1_DOM6_MEM0_BLK_CFG_W2	1_2C48h
MBC1_DOM6_MEM0_BLK_CFG_W3	1_2C4Ch
MBC1_DOM6_MEM1_BLK_CFG_W0	1_2D80h
MBC1_DOM6_MEM1_BLK_CFG_W1	1_2D84h
MBC1_DOM6_MEM1_BLK_CFG_W2	1_2D88h
MBC1_DOM6_MEM1_BLK_CFG_W3	1_2D8Ch
MBC1_DOM7_MEM0_BLK_CFG_W0	1_2E40h
MBC1_DOM7_MEM0_BLK_CFG_W1	1_2E44h
MBC1_DOM7_MEM0_BLK_CFG_W2	1_2E48h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM7_MEM0_BLK_CFG_W3	1_2E4Ch
MBC1_DOM7_MEM1_BLK_CFG_W0	1_2F80h
MBC1_DOM7_MEM1_BLK_CFG_W1	1_2F84h
MBC1_DOM7_MEM1_BLK_CFG_W2	1_2F88h
MBC1_DOM7_MEM1_BLK_CFG_W3	1_2F8Ch
MBC1_DOM8_MEM0_BLK_CFG_W0	1_3040h
MBC1_DOM8_MEM0_BLK_CFG_W1	1_3044h
MBC1_DOM8_MEM0_BLK_CFG_W2	1_3048h
MBC1_DOM8_MEM0_BLK_CFG_W3	1_304Ch
MBC1_DOM8_MEM1_BLK_CFG_W0	1_3180h
MBC1_DOM8_MEM1_BLK_CFG_W1	1_3184h
MBC1_DOM8_MEM1_BLK_CFG_W2	1_3188h
MBC1_DOM8_MEM1_BLK_CFG_W3	1_318Ch
MBC1_DOM9_MEM0_BLK_CFG_W0	1_3240h
MBC1_DOM9_MEM0_BLK_CFG_W1	1_3244h
MBC1_DOM9_MEM0_BLK_CFG_W2	1_3248h
MBC1_DOM9_MEM0_BLK_CFG_W3	1_324Ch
MBC1_DOM9_MEM1_BLK_CFG_W0	1_3380h
MBC1_DOM9_MEM1_BLK_CFG_W1	1_3384h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM9_MEM1_BLK_CFG_W2	1_3388h
MBC1_DOM9_MEM1_BLK_CFG_W3	1_338Ch
MBC1_DOM10_MEM0_BLK_CFG_W0	1_3440h
MBC1_DOM10_MEM0_BLK_CFG_W1	1_3444h
MBC1_DOM10_MEM0_BLK_CFG_W2	1_3448h
MBC1_DOM10_MEM0_BLK_CFG_W3	1_344Ch
MBC1_DOM10_MEM1_BLK_CFG_W0	1_3580h
MBC1_DOM10_MEM1_BLK_CFG_W1	1_3584h
MBC1_DOM10_MEM1_BLK_CFG_W2	1_3588h
MBC1_DOM10_MEM1_BLK_CFG_W3	1_358Ch
MBC1_DOM11_MEM0_BLK_CFG_W0	1_3640h
MBC1_DOM11_MEM0_BLK_CFG_W1	1_3644h
MBC1_DOM11_MEM0_BLK_CFG_W2	1_3648h
MBC1_DOM11_MEM0_BLK_CFG_W3	1_364Ch
MBC1_DOM11_MEM1_BLK_CFG_W0	1_3780h
MBC1_DOM11_MEM1_BLK_CFG_W1	1_3784h
MBC1_DOM11_MEM1_BLK_CFG_W2	1_3788h
MBC1_DOM11_MEM1_BLK_CFG_W3	1_378Ch
MBC1_DOM12_MEM0_BLK_CFG_W0	1_3840h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM12_MEM0_BLK_CFG_W1	1_3844h
MBC1_DOM12_MEM0_BLK_CFG_W2	1_3848h
MBC1_DOM12_MEM0_BLK_CFG_W3	1_384Ch
MBC1_DOM12_MEM1_BLK_CFG_W0	1_3980h
MBC1_DOM12_MEM1_BLK_CFG_W1	1_3984h
MBC1_DOM12_MEM1_BLK_CFG_W2	1_3988h
MBC1_DOM12_MEM1_BLK_CFG_W3	1_398Ch
MBC1_DOM13_MEM0_BLK_CFG_W0	1_3A40h
MBC1_DOM13_MEM0_BLK_CFG_W1	1_3A44h
MBC1_DOM13_MEM0_BLK_CFG_W2	1_3A48h
MBC1_DOM13_MEM0_BLK_CFG_W3	1_3A4Ch
MBC1_DOM13_MEM1_BLK_CFG_W0	1_3B80h
MBC1_DOM13_MEM1_BLK_CFG_W1	1_3B84h
MBC1_DOM13_MEM1_BLK_CFG_W2	1_3B88h
MBC1_DOM13_MEM1_BLK_CFG_W3	1_3B8Ch
MBC1_DOM14_MEM0_BLK_CFG_W0	1_3C40h
MBC1_DOM14_MEM0_BLK_CFG_W1	1_3C44h
MBC1_DOM14_MEM0_BLK_CFG_W2	1_3C48h
MBC1_DOM14_MEM0_BLK_CFG_W3	1_3C4Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM14_MEM1_BLK_CFG_W0	1_3D80h
MBC1_DOM14_MEM1_BLK_CFG_W1	1_3D84h
MBC1_DOM14_MEM1_BLK_CFG_W2	1_3D88h
MBC1_DOM14_MEM1_BLK_CFG_W3	1_3D8Ch
MBC1_DOM15_MEM0_BLK_CFG_W0	1_3E40h
MBC1_DOM15_MEM0_BLK_CFG_W1	1_3E44h
MBC1_DOM15_MEM0_BLK_CFG_W2	1_3E48h
MBC1_DOM15_MEM0_BLK_CFG_W3	1_3E4Ch
MBC1_DOM15_MEM1_BLK_CFG_W0	1_3F80h
MBC1_DOM15_MEM1_BLK_CFG_W1	1_3F84h
MBC1_DOM15_MEM1_BLK_CFG_W2	1_3F88h
MBC1_DOM15_MEM1_BLK_CFG_W3	1_3F8Ch

Function

MBC[m]_DOM[d]_MEM[s]_BLK_CFG_W[w], where,

- m - mbc index
- d - domain index
- s - memory slave index
- w - word index

These registers are read/write for both the alternate view of the NSE bit and the 3-bit MBACSEL field. This is an array of 4 bit fields defining the block configuration for the given submemory. Each 4-bit field includes an alternative view of the associated NSE bit plus a 3-bit access control select that selects the global access control value that applies to the referenced memory block.

For a given memory block, B, if the value programmed in the MBACSELn field selects a locked MBC_MEMN_GLBACr register, the MBACSEL field is locked until the next reset. Depending on BLK_CFG_W, the NSEn/MBASELn fields correspond to different blocks. The following table describes the relationship between W (word index) and B (block number).

MEM0 supports up to 512 blocks ; W0 - W63

MEM1-3 supports up to 64 block ; W0 - W7

NOTE

This register is shown with fields such that it covers entire 32 bit register. However, actual fields may be less if number of blocks are such that fields don't align to 32 bits. These absent fields may be shown but user can refer to table below to deduce actual fields and treat absent fields as reserved.

Table 306. Block Config Word to Block Number Relationship

Block Config Word	NSE7/ MBACSEL7	NSE6/ MBACSEL6	NSE5/ MBACSEL5	NSE4/ MBACSEL4	NSE3/ MBACSEL3	NSE2/ MBACSEL2	NSE1/ MBACSEL1	NSE0/ MBACSEL0
W0	block 7	block 6	block 5	block 4	block 3	block 2	block 1	block 0
W1	block 15	block 14	block 13	block 12	block 11	block 10	block 9	block 8
W2	block 23	block 22	block 21	block 20	block 19	block 18	block 17	block 16
W3	block 31	block 30	block 29	block 28	block 27	block 26	block 25	block 24
W4	block 39	block 38	block 37	block 36	block 35	block 34	block 33	block 32
W5	block 47	block 46	block 45	block 44	block 43	block 42	block 41	block 40
W6	block 55	block 54	block 53	block 52	block 51	block 50	block 49	block 48
W7	block 63	block 62	block 61	block 60	block 59	block 58	block 57	block 56
For MEM0, block 64 - block 511								
W8	block 71	block 70	block 69	block 68	block 67	block 66	block 65	block 64
. . .								
W63	block 511	block 510	block 509	block 508	block 507	block 506	block 505	block 504

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NSE7	MBACSEL7			NSE6	MBACSEL6			NSE5	MBACSEL5			NSE4	MBACSEL4		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NSE3	MBACSEL3			NSE2	MBACSEL2			NSE1	MBACSEL1			NSE0	MBACSEL0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 NSE7	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
30-28 MBACSEL7	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL7 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
27 NSE6	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
26-24 MBACSEL6	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL6 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
23 NSE5	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
22-20 MBACSEL5	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL5 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
19 NSE4	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
18-16 MBACSEL4	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL4 = r and MBC_MEMN_GLBACr[LK] = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
15 NSE3	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
14-12 MBACSEL3	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL3 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
11 NSE2	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
10-8	Memory Block Access Control Select for block B

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBACSEL2	This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL2 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
7 NSE1	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
6-4 MBACSEL1	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL1 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
3 NSE0	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
2-0 MBACSEL0	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL0 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B

43.8.1.32 MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W0 - MBC1_DOM15_MEM1_BLK_NSE_W0)

Offset

Register	Offset
MBC0_DOM0_MEM0_BLK_NSE_W0	1_0140h
MBC0_DOM0_MEM0_BLK_NSE_W1	1_0144h
MBC0_DOM0_MEM0_BLK_NSE_W2	1_0148h
MBC0_DOM0_MEM0_BLK_NSE_W3	1_014Ch
MBC0_DOM0_MEM1_BLK_NSE_W0	1_01A0h
MBC0_DOM0_MEM2_BLK_NSE_W0	1_01C8h
MBC0_DOM0_MEM3_BLK_NSE_W0	1_01F0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM1_MEM0_BLK_NSE_W0	1_0340h
MBC0_DOM1_MEM0_BLK_NSE_W1	1_0344h
MBC0_DOM1_MEM0_BLK_NSE_W2	1_0348h
MBC0_DOM1_MEM0_BLK_NSE_W3	1_034Ch
MBC0_DOM1_MEM1_BLK_NSE_W0	1_03A0h
MBC0_DOM1_MEM2_BLK_NSE_W0	1_03C8h
MBC0_DOM1_MEM3_BLK_NSE_W0	1_03F0h
MBC0_DOM2_MEM0_BLK_NSE_W0	1_0540h
MBC0_DOM2_MEM0_BLK_NSE_W1	1_0544h
MBC0_DOM2_MEM0_BLK_NSE_W2	1_0548h
MBC0_DOM2_MEM0_BLK_NSE_W3	1_054Ch
MBC0_DOM2_MEM1_BLK_NSE_W0	1_05A0h
MBC0_DOM2_MEM2_BLK_NSE_W0	1_05C8h
MBC0_DOM2_MEM3_BLK_NSE_W0	1_05F0h
MBC0_DOM3_MEM0_BLK_NSE_W0	1_0740h
MBC0_DOM3_MEM0_BLK_NSE_W1	1_0744h
MBC0_DOM3_MEM0_BLK_NSE_W2	1_0748h
MBC0_DOM3_MEM0_BLK_NSE_W3	1_074Ch
MBC0_DOM3_MEM1_BLK_NSE_W0	1_07A0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM3_MEM2_BLK_NSE_W0	1_07C8h
MBC0_DOM3_MEM3_BLK_NSE_W0	1_07F0h
MBC0_DOM4_MEM0_BLK_NSE_W0	1_0940h
MBC0_DOM4_MEM0_BLK_NSE_W1	1_0944h
MBC0_DOM4_MEM0_BLK_NSE_W2	1_0948h
MBC0_DOM4_MEM0_BLK_NSE_W3	1_094Ch
MBC0_DOM4_MEM1_BLK_NSE_W0	1_09A0h
MBC0_DOM4_MEM2_BLK_NSE_W0	1_09C8h
MBC0_DOM4_MEM3_BLK_NSE_W0	1_09F0h
MBC0_DOM5_MEM0_BLK_NSE_W0	1_0B40h
MBC0_DOM5_MEM0_BLK_NSE_W1	1_0B44h
MBC0_DOM5_MEM0_BLK_NSE_W2	1_0B48h
MBC0_DOM5_MEM0_BLK_NSE_W3	1_0B4Ch
MBC0_DOM5_MEM1_BLK_NSE_W0	1_0BA0h
MBC0_DOM5_MEM2_BLK_NSE_W0	1_0BC8h
MBC0_DOM5_MEM3_BLK_NSE_W0	1_0BF0h
MBC0_DOM6_MEM0_BLK_NSE_W0	1_0D40h
MBC0_DOM6_MEM0_BLK_NSE_W1	1_0D44h
MBC0_DOM6_MEM0_BLK_NSE_W2	1_0D48h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM6_MEM0_BLK_NSE_W3	1_0D4Ch
MBC0_DOM6_MEM1_BLK_NSE_W0	1_0DA0h
MBC0_DOM6_MEM2_BLK_NSE_W0	1_0DC8h
MBC0_DOM6_MEM3_BLK_NSE_W0	1_0DF0h
MBC0_DOM7_MEM0_BLK_NSE_W0	1_0F40h
MBC0_DOM7_MEM0_BLK_NSE_W1	1_0F44h
MBC0_DOM7_MEM0_BLK_NSE_W2	1_0F48h
MBC0_DOM7_MEM0_BLK_NSE_W3	1_0F4Ch
MBC0_DOM7_MEM1_BLK_NSE_W0	1_0FA0h
MBC0_DOM7_MEM2_BLK_NSE_W0	1_0FC8h
MBC0_DOM7_MEM3_BLK_NSE_W0	1_0FF0h
MBC0_DOM8_MEM0_BLK_NSE_W0	1_1140h
MBC0_DOM8_MEM0_BLK_NSE_W1	1_1144h
MBC0_DOM8_MEM0_BLK_NSE_W2	1_1148h
MBC0_DOM8_MEM0_BLK_NSE_W3	1_114Ch
MBC0_DOM8_MEM1_BLK_NSE_W0	1_11A0h
MBC0_DOM8_MEM2_BLK_NSE_W0	1_11C8h
MBC0_DOM8_MEM3_BLK_NSE_W0	1_11F0h
MBC0_DOM9_MEM0_BLK_NSE_W0	1_1340h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM9_MEM0_BLK_NSE_W1	1_1344h
MBC0_DOM9_MEM0_BLK_NSE_W2	1_1348h
MBC0_DOM9_MEM0_BLK_NSE_W3	1_134Ch
MBC0_DOM9_MEM1_BLK_NSE_W0	1_13A0h
MBC0_DOM9_MEM2_BLK_NSE_W0	1_13C8h
MBC0_DOM9_MEM3_BLK_NSE_W0	1_13F0h
MBC0_DOM10_MEM0_BLK_NSE_W0	1_1540h
MBC0_DOM10_MEM0_BLK_NSE_W1	1_1544h
MBC0_DOM10_MEM0_BLK_NSE_W2	1_1548h
MBC0_DOM10_MEM0_BLK_NSE_W3	1_154Ch
MBC0_DOM10_MEM1_BLK_NSE_W0	1_15A0h
MBC0_DOM10_MEM2_BLK_NSE_W0	1_15C8h
MBC0_DOM10_MEM3_BLK_NSE_W0	1_15F0h
MBC0_DOM11_MEM0_BLK_NSE_W0	1_1740h
MBC0_DOM11_MEM0_BLK_NSE_W1	1_1744h
MBC0_DOM11_MEM0_BLK_NSE_W2	1_1748h
MBC0_DOM11_MEM0_BLK_NSE_W3	1_174Ch
MBC0_DOM11_MEM1_BLK_NSE_W0	1_17A0h
MBC0_DOM11_MEM2_BLK_NSE_W0	1_17C8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM11_MEM3_BLK_NSE_W0	1_17F0h
MBC0_DOM12_MEM0_BLK_NSE_W0	1_1940h
MBC0_DOM12_MEM0_BLK_NSE_W1	1_1944h
MBC0_DOM12_MEM0_BLK_NSE_W2	1_1948h
MBC0_DOM12_MEM0_BLK_NSE_W3	1_194Ch
MBC0_DOM12_MEM1_BLK_NSE_W0	1_19A0h
MBC0_DOM12_MEM2_BLK_NSE_W0	1_19C8h
MBC0_DOM12_MEM3_BLK_NSE_W0	1_19F0h
MBC0_DOM13_MEM0_BLK_NSE_W0	1_1B40h
MBC0_DOM13_MEM0_BLK_NSE_W1	1_1B44h
MBC0_DOM13_MEM0_BLK_NSE_W2	1_1B48h
MBC0_DOM13_MEM0_BLK_NSE_W3	1_1B4Ch
MBC0_DOM13_MEM1_BLK_NSE_W0	1_1BA0h
MBC0_DOM13_MEM2_BLK_NSE_W0	1_1BC8h
MBC0_DOM13_MEM3_BLK_NSE_W0	1_1BF0h
MBC0_DOM14_MEM0_BLK_NSE_W0	1_1D40h
MBC0_DOM14_MEM0_BLK_NSE_W1	1_1D44h
MBC0_DOM14_MEM0_BLK_NSE_W2	1_1D48h
MBC0_DOM14_MEM0_BLK_NSE_W3	1_1D4Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM14_MEM1_BLK_NSE_W0	1_1DA0h
MBC0_DOM14_MEM2_BLK_NSE_W0	1_1DC8h
MBC0_DOM14_MEM3_BLK_NSE_W0	1_1DF0h
MBC0_DOM15_MEM0_BLK_NSE_W0	1_1F40h
MBC0_DOM15_MEM0_BLK_NSE_W1	1_1F44h
MBC0_DOM15_MEM0_BLK_NSE_W2	1_1F48h
MBC0_DOM15_MEM0_BLK_NSE_W3	1_1F4Ch
MBC0_DOM15_MEM1_BLK_NSE_W0	1_1FA0h
MBC0_DOM15_MEM2_BLK_NSE_W0	1_1FC8h
MBC0_DOM15_MEM3_BLK_NSE_W0	1_1FF0h
MBC1_DOM0_MEM0_BLK_NSE_W0	1_2140h
MBC1_DOM0_MEM1_BLK_NSE_W0	1_21A0h
MBC1_DOM1_MEM0_BLK_NSE_W0	1_2340h
MBC1_DOM1_MEM1_BLK_NSE_W0	1_23A0h
MBC1_DOM2_MEM0_BLK_NSE_W0	1_2540h
MBC1_DOM2_MEM1_BLK_NSE_W0	1_25A0h
MBC1_DOM3_MEM0_BLK_NSE_W0	1_2740h
MBC1_DOM3_MEM1_BLK_NSE_W0	1_27A0h
MBC1_DOM4_MEM0_BLK_NSE_W0	1_2940h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM4_MEM1_BLK_NSE_W0	1_29A0h
MBC1_DOM5_MEM0_BLK_NSE_W0	1_2B40h
MBC1_DOM5_MEM1_BLK_NSE_W0	1_2BA0h
MBC1_DOM6_MEM0_BLK_NSE_W0	1_2D40h
MBC1_DOM6_MEM1_BLK_NSE_W0	1_2DA0h
MBC1_DOM7_MEM0_BLK_NSE_W0	1_2F40h
MBC1_DOM7_MEM1_BLK_NSE_W0	1_2FA0h
MBC1_DOM8_MEM0_BLK_NSE_W0	1_3140h
MBC1_DOM8_MEM1_BLK_NSE_W0	1_31A0h
MBC1_DOM9_MEM0_BLK_NSE_W0	1_3340h
MBC1_DOM9_MEM1_BLK_NSE_W0	1_33A0h
MBC1_DOM10_MEM0_BLK_NSE_W0	1_3540h
MBC1_DOM10_MEM1_BLK_NSE_W0	1_35A0h
MBC1_DOM11_MEM0_BLK_NSE_W0	1_3740h
MBC1_DOM11_MEM1_BLK_NSE_W0	1_37A0h
MBC1_DOM12_MEM0_BLK_NSE_W0	1_3940h
MBC1_DOM12_MEM1_BLK_NSE_W0	1_39A0h
MBC1_DOM13_MEM0_BLK_NSE_W0	1_3B40h
MBC1_DOM13_MEM1_BLK_NSE_W0	1_3BA0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM14_MEM0_BLK_NSE_W0	1_3D40h
MBC1_DOM14_MEM1_BLK_NSE_W0	1_3DA0h
MBC1_DOM15_MEM0_BLK_NSE_W0	1_3F40h
MBC1_DOM15_MEM1_BLK_NSE_W0	1_3FA0h

Function

MBC[m]_DOM[d]_MEM[s]_BLK_NSE_W[w], where,

- m - mbc index
- d - domain index
- s - memory slave index
- w - word index

This is the bitmap of the individual NSE bits for the memory block.

NOTE

This register is shown with fields such that it covers entire 32 bit register. However, actual fields may be less if number of blocks are such that fields don't align to 32 bits. These absent fields may be shown but user can refer to table below to deduce actual fields and treat absent fields as reserved.

The following table describes the relationship between W (word index) and B (block number).

Table 307. Block Config Word to Block Number Relationship

Block NSE Word	BIT31	BIT30-BIT1	BIT0
W0	block 31	block 30 - block 1	block 0
W1	block 63	block 62 - block 33	block 32
For MEM0, block 64 - block 511			
W2	block 95	block 94 - block 65	block 64
. . .			
W15	block 511	block 510 - block 481	block 480

The NSE bitmap can be accessed directly by register reads and writes to these locations, or "indirectly" through writes to NSE_SET, NSE_CLR, NSE_CLR_ALL, and BLK_CFG[NSEn] registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BIT31	BIT30	BIT29	BIT28	BIT27	BIT26	BIT25	BIT24	BIT23	BIT22	BIT21	BIT20	BIT19	BIT18	BIT17	BIT16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BITb	Bit b NonSecure Enable [b = 0 - 31] 0b - Secure accesses to block B are based on corresponding MBACSEL field in register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).

43.8.1.33 MBC Global Access Control (MBC1_MEMN_GLBAC0)

Offset

Register	Offset
MBC1_MEMN_GLBAC0	1_2020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SPR	SPW	SPX	0	SUR	SUW	SUX	0	NPR	NPW	NPX	0	NUR	NUW	NUX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6	NonsecurePriv Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
NPR	NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.1.34 MBC Global Access Control (MBC1_MEMN_GLBAC1 - MBC1_MEMN_GLBAC7)

Offset

Register	Offset
MBC1_MEMN_GLBAC1	1_2024h

Table continues on the next page...

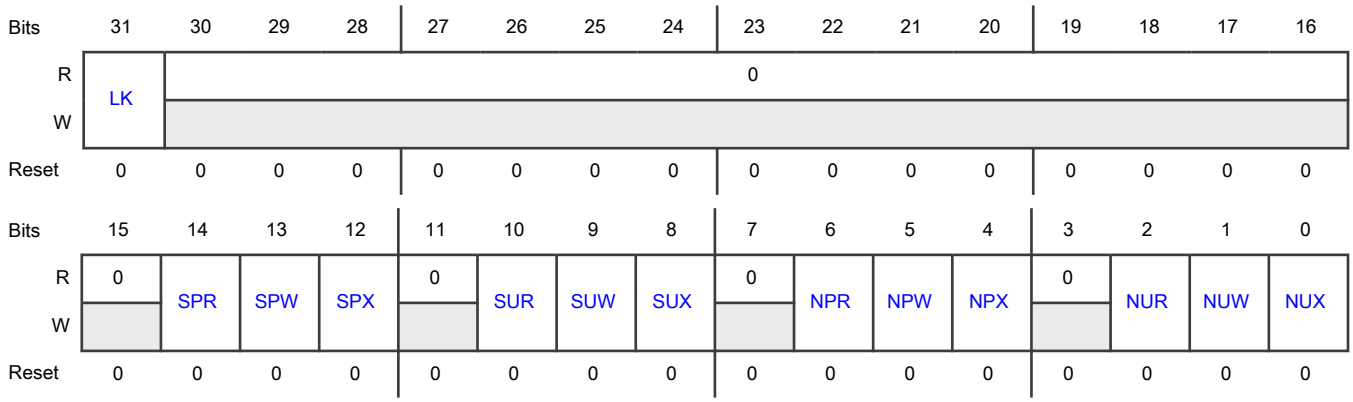
Table continued from the previous page...

Register	Offset
MBC1_MEMN_GLBAC2	1_2028h
MBC1_MEMN_GLBAC3	1_202Ch
MBC1_MEMN_GLBAC4	1_2030h
MBC1_MEMN_GLBAC5	1_2034h
MBC1_MEMN_GLBAC6	1_2038h
MBC1_MEMN_GLBAC7	1_203Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these control access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked and cannot be altered.
30-15 —	Reserved
14	SecurePriv Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
SPR	SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

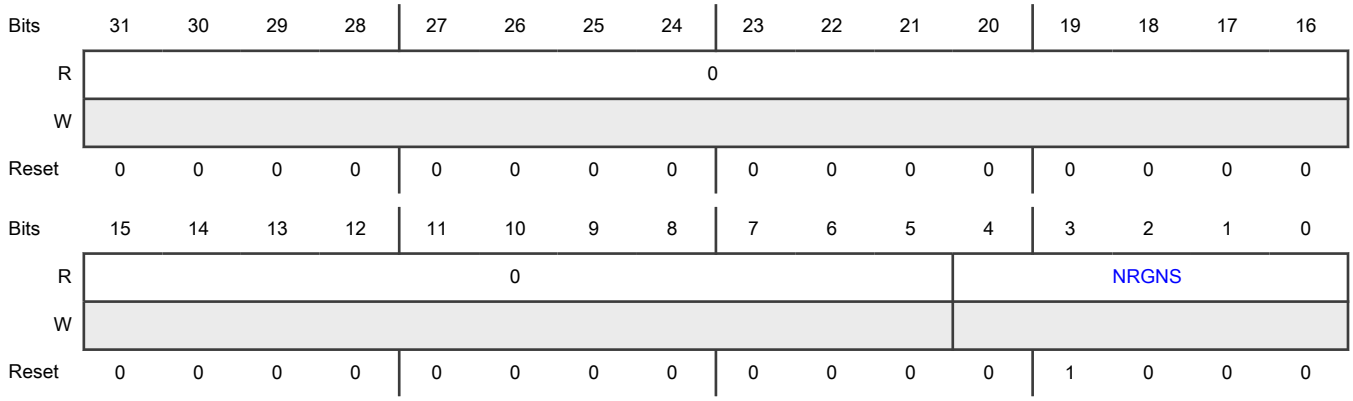
43.8.1.35 MRC Global Configuration Register (MRC0_GLBCFG - MRC1_GLBCFG)

Offset

Register	Offset
MRC0_GLBCFG	1_4000h
MRC1_GLBCFG	1_5000h

Function
This read-only register describes the number of region descriptors for this memory.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 NRGNS	Number of regions [1-16]

43.8.1.36 MRC NonSecure Enable Region Indirect (MRC0_NSE_RGN_INDIRECT - MRC1_NSE_RGN_INDIRECT)

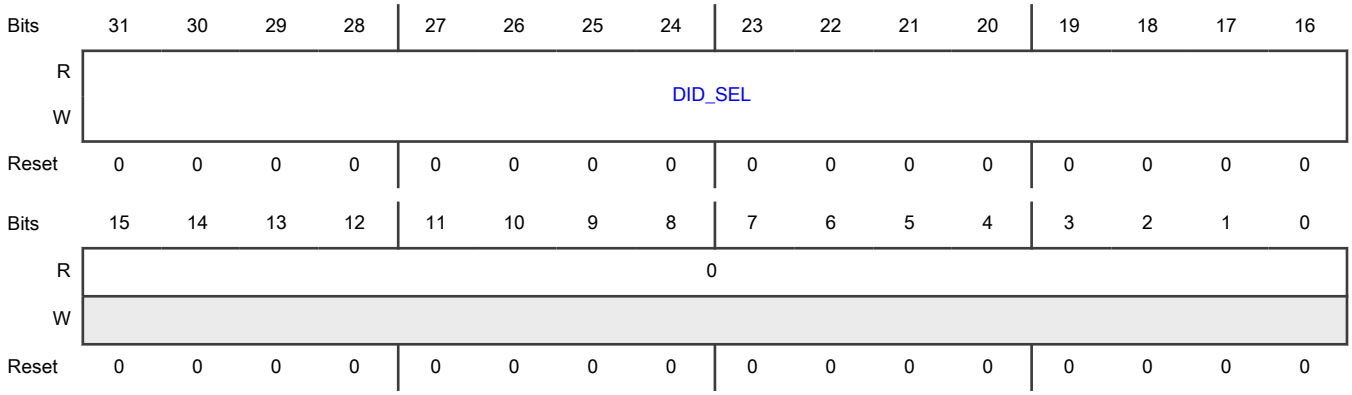
Offset

Register	Offset
MRC0_NSE_RGN_INDIRECT	1_4010h
MRC1_NSE_RGN_INDIRECT	1_5010h

Function

This R/W register defines the selected domains that are affected by writes to the NSE_RGN_SET and NSE_RGN_CLR registers.

Diagram



Fields

Field	Function
31-16 DID_SEL	DID Select Destination domain bitmap select. • Bit [31] = 1, update domain 15 • Bit [30] = 1, update domain 14 • Bit [29] = 1, update domain 13 • Bit [28] = 1, update domain 12 • Bit [27] = 1, update domain 11 • Bit [26] = 1, update domain 10 • Bit [25] = 1, update domain 9 • Bit [24] = 1, update domain 8 • Bit [23] = 1, update domain 7 • Bit [22] = 1, update domain 6 • Bit [21] = 1, update domain 5 • Bit [20] = 1, update domain 4 • Bit [19] = 1, update domain 3 • Bit [18] = 1, update domain 2 • Bit [17] = 1, update domain 1 • Bit [16] = 1, update domain 0
15-0 —	Reserved

43.8.1.37 MRC NonSecure Enable Region Set (MRC0_NSE_RGN_SET - MRC1_NSE_RGN_SET)

Offset

Register	Offset
MRC0_NSE_RGN_SET	1_4014h
MRC1_NSE_RGN_SET	1_5014h

Function

A write to this location sets the appropriate NSE bits in region descriptors for the selected domain in MRC NonSecure Enable Region Indirect register. This register reads as 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	W1SET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 W1SET	Write-1 Set • Bit [15] = Set the NSE bits for RGD15 • Bit [14] = Set the NSE bits for RGD14 • Bit [13] = Set the NSE bits for RGD13 • Bit [12] = Set the NSE bits for RGD12 • Bit [11] = Set the NSE bits for RGD11 • Bit [10] = Set the NSE bits for RGD10 • Bit [9] = Set the NSE bits for RGD9 • Bit [8] = Set the NSE bits for RGD8 • Bit [7] = Set the NSE bits for RGD7

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Bit [6] = Set the NSE bits for RGD6 • Bit [5] = Set the NSE bits for RGD5 • Bit [4] = Set the NSE bits for RGD4 • Bit [3] = Set the NSE bits for RGD3 • Bit [2] = Set the NSE bits for RGD2 • Bit [1] = Set the NSE bits for RGD1 • Bit [0] = Set the NSE bits for RGD0

43.8.1.38 MRC NonSecure Enable Region Clear (MRC0_NSE_RGN_CLR - MRC1_NSE_RGN_CLR)

Offset

Register	Offset
MRC0_NSE_RGN_CLR	1_4018h
MRC1_NSE_RGN_CLR	1_5018h

Function

A write to this location clears the appropriate NSE bits in region descriptors for the selected domain in MRC NonSecure Enable Region Indirect register. This register reads as 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	W1CLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 W1CLR	Write-1 Clear • Bit [15] = Clear the NSE bits for RGD15 • Bit [14] = Clear the NSE bits for RGD14 • Bit [13] = Clear the NSE bits for RGD13 • Bit [12] = Clear the NSE bits for RGD12 • Bit [11] = Clear the NSE bits for RGD11 • Bit [10] = Clear the NSE bits for RGD10 • Bit [9] = Clear the NSE bits for RGD9 • Bit [8] = Clear the NSE bits for RGD8 • Bit [7] = Clear the NSE bits for RGD7 • Bit [6] = Clear the NSE bits for RGD6 • Bit [5] = Clear the NSE bits for RGD5 • Bit [4] = Clear the NSE bits for RGD4 • Bit [3] = Clear the NSE bits for RGD3 • Bit [2] = Clear the NSE bits for RGD2 • Bit [1] = Clear the NSE bits for RGD1 • Bit [0] = Clear the NSE bits for RGD0

43.8.1.39 MRC NonSecure Enable Region Clear All (MRC0_NSE_RGN_CLR_ALL - MRC1_NSE_RGN_CLR_ALL)

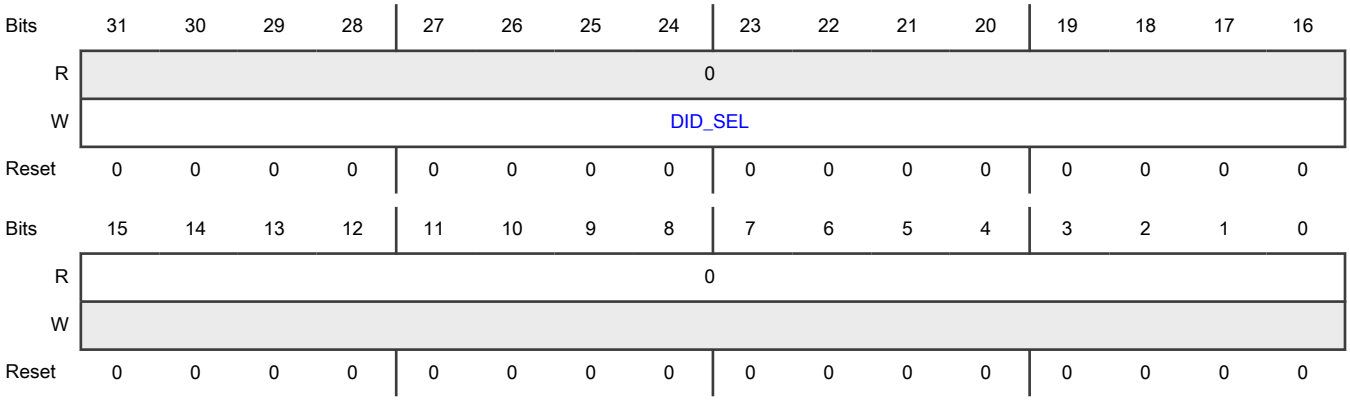
Offset

Register	Offset
MRC0_NSE_RGN_CLR_ALL	1_401Ch
MRC1_NSE_RGN_CLR_ALL	1_501Ch

Function

A write to this register clears all the region NSE bits for the selected domains defined in the write data.

Diagram



Fields

Field	Function
31-16 DID_SEL	DID Select Destination domain bitmap select. • Bit [31] = 1 clear NSE bits for domain 15 • Bit [30] = 1 clear NSE bits for domain 14 • Bit [29] = 1 clear NSE bits for domain 13 • Bit [28] = 1 clear NSE bits for domain 12 • Bit [27] = 1 clear NSE bits for domain 11 • Bit [26] = 1 clear NSE bits for domain 10 • Bit [25] = 1 clear NSE bits for domain 9 • Bit [24] = 1 clear NSE bits for domain 8 • Bit [23] = 1 clear NSE bits for domain 7 • Bit [22] = 1 clear NSE bits for domain 6 • Bit [21] = 1 clear NSE bits for domain 5 • Bit [20] = 1 clear NSE bits for domain 4 • Bit [19] = 1 clear NSE bits for domain 3 • Bit [18] = 1 clear NSE bits for domain 2 • Bit [17] = 1 clear NSE bits for domain 1 • Bit [16] = 1 clear NSE bits for domain 0
15-0 —	Reserved

43.8.1.40 MRC Global Access Control (MRC0_GLBAC0)

Offset

Register	Offset
MRC0_GLBAC0	1_4020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SPR	SPW	SPX	0	SUR	SUW	SUX	0	NPR	NPW	NPX	0	NUR	NUW	NUX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.1.41 MRC Global Access Control (MRC0_GLBAC1 - MRC0_GLBAC7)

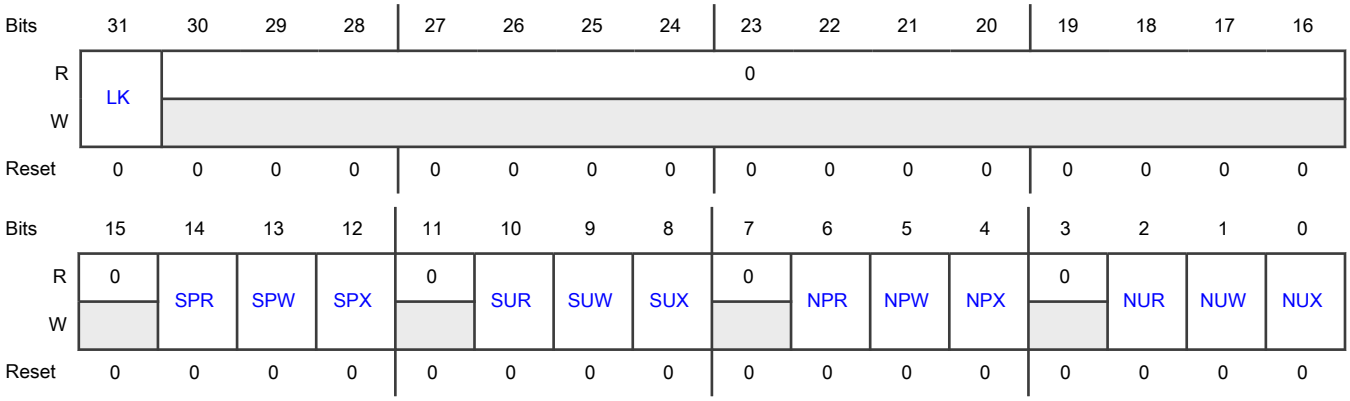
Offset

Register	Offset
MRC0_GLBAC1	1_4024h
MRC0_GLBAC2	1_4028h
MRC0_GLBAC3	1_402Ch
MRC0_GLBAC4	1_4030h
MRC0_GLBAC5	1_4034h
MRC0_GLBAC6	1_4038h
MRC0_GLBAC7	1_403Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked (read-only) and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10	SecureUser Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
SUR	SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.1.42 MRC Region Descriptor Word 0 (MRC0_DOM0_RGD0_W0 - MRC1_DOM15_RGD7_W0)

Offset

For m = 0 to 1; d = 0 to 15; r = 0 to 7:

Register	Offset
MRCm_DOMd_RGDr_W0	1_4040h + (m × 1000h) + (d × 100h) + (r × 8h)

Function

MRC[m]_DOM[d]_RGD[r]_W[w], where,

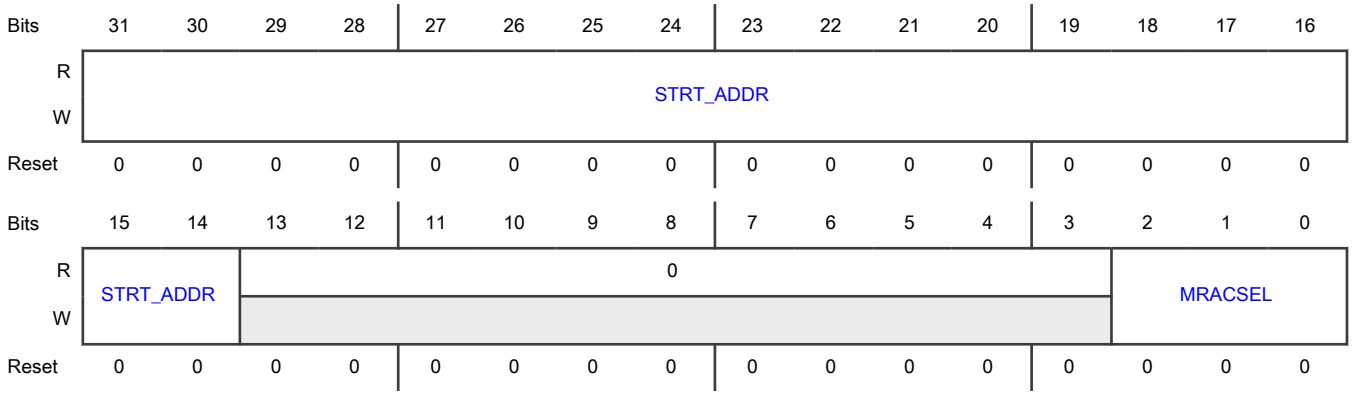
- m - mrc index
- d - domain index
- r - region descriptor index
- w - word index

The MRC[m]_DOM[d]_RGD[r]_W[w] registers provide a 3-dimensional data structure for defining access control policies per domain for each supported memory region.

Memory map/register definition:

There two word-size registers (W[w]) defined in each memory region. There are up to 16 memory region checkers (MRCs), m, each supporting up to 16 memory region descriptors (RGD), r. The number of MRCs is defined by HWCFG0[NMRC].

Diagram



Fields

Field	Function
31-14 STRT_ADDR	Start Address 0-mod-16K physical start address [31:14] of region r. **NOTE** This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1.
13-3 —	Reserved
2-0 MRACSEL	Memory Region Access Control Select This field selects the global access control associated with region r. **NOTE** This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1. 000b - Select MRC_GLBAC0 access control policy 001b - Select MRC_GLBAC1 access control policy 010b - Select MRC_GLBAC2 access control policy 011b - Select MRC_GLBAC3 access control policy 100b - Select MRC_GLBAC4 access control policy 101b - Select MRC_GLBAC5 access control policy 110b - Select MRC_GLBAC6 access control policy 111b - Select MRC_GLBAC7 access control policy

43.8.1.43 MRC Region Descriptor Word 1 (MRC0_DOM0_RGD0_W1 - MRC1_DOM15_RGD7_W1)

Offset

For m = 0 to 1; d = 0 to 15; r = 0 to 7:

Register	Offset
MRCm_DOMd_RGDr_W1	1_4044h + (m × 1000h) + (d × 100h) + (r × 8h)

Function

MRC[m]_DOM[d]_RGD[r]_W[w], where,

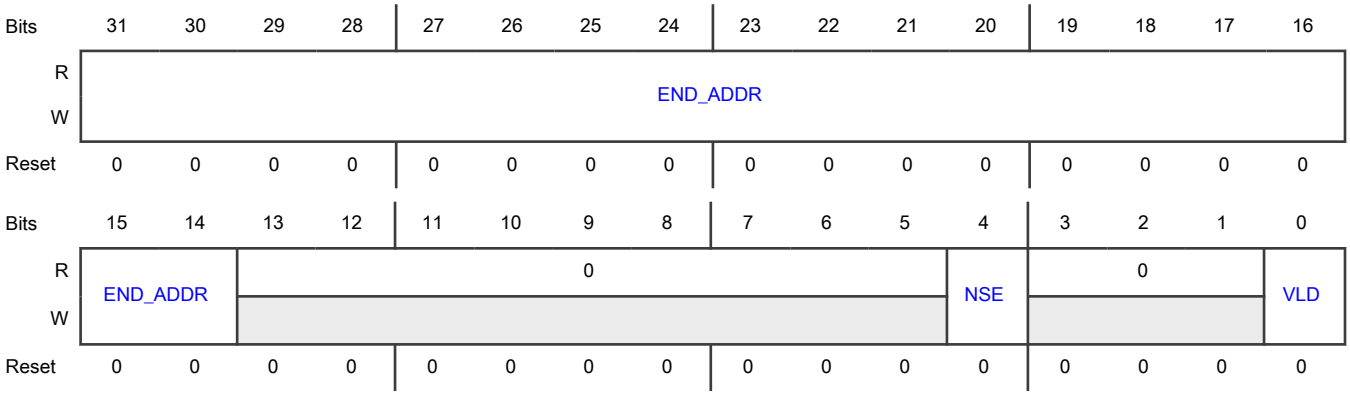
- m - mrc index
- d - domain index
- r - region descriptor index
- w - word index

The MRC[m]_[d]_rgd[r]_w[w] registers provide a 3-dimensional data structure for defining access control policies per domain for each supported memory region.

Memory map/register definition:

There two word-size registers (W[w]) defined in each memory region. There are up to 16 memory region checkers (MRCs), m, each supporting up to 16 region descriptors (RGD), r. The number of MRCs is defined by HWCFG0[NMRC].

Diagram



Fields

Field	Function
31-14 END_ADDR	End Address (16K-1)-mod-16K physical end address [31:14] of region r. **NOTE** This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1.
13-5 —	Reserved
4	NonSecure Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
NSE	Alternate view of NSE flag for Region r. 0b - Secure accesses to region r are based on corresponding MRACSEL field in this register (MRCm_DOMd_RGDr_Ww[MRACSEL]), nonsecure accesses to region r are not allowed. 1b - Secure accesses to region r are not allowed, nonsecure accesses to region r are based on corresponding MRACSEL field in this register (MRCm_DOMd_RGDr_Ww[MRACSEL]).
3-1 —	Reserved
0 VLD	Valid If set, this region is valid and accesses are checked for this region. **NOTE** This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1.

43.8.1.44 MRC Region Descriptor NonSecure Enable (MRC0_DOM0_RGD_NSE - MRC1_DOM15_RGD_NSE)

Offset

For m = 0 to 1; d = 0 to 15:

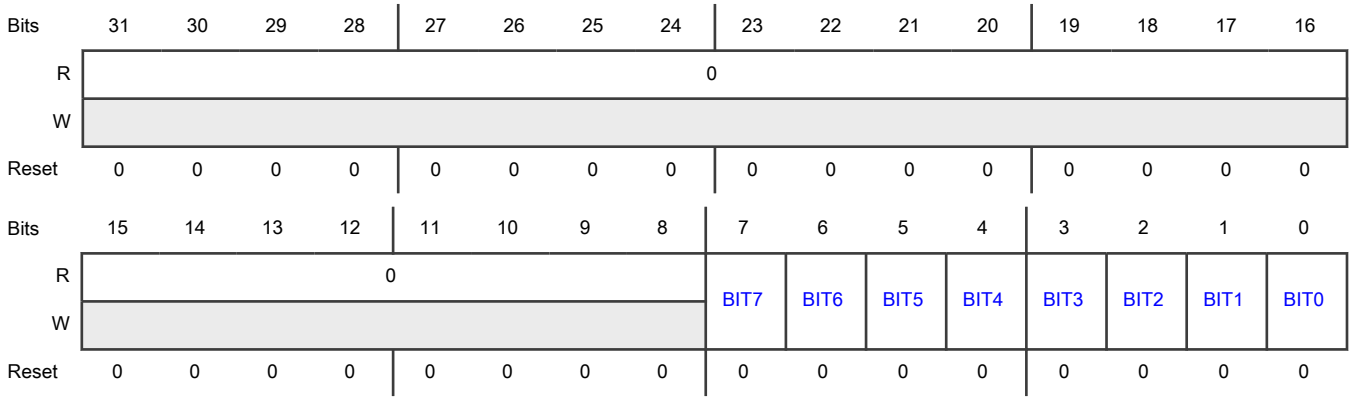
Register	Offset
MRCm_DOMd_RGD_NSE	1_40C0h + (m × 1000h) + (d × 100h)

Function

This register is the bitmap of the individual NSE bits for the region descriptors. Bit n corresponds to region n.

These bits can also be written or "indirectly" through writes to NSE_SET, NSE_CLR, NSE_CLR_ALL, and RGD_W0[NSE] registers.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 BITn	Bit n NonSecure Enable [n = 0 - 15] 0b - Secure accesses to region r are based on corresponding MRACSEL field in this register (MRCm_DOMd_RGDr_Ww[MRACSEL]), nonsecure accesses to region r are not allowed. 1b - Secure accesses to region r are not allowed, nonsecure accesses to region r are based on corresponding MRACSEL field in register (MRCm_DOMd_RGDr_Ww[MRACSEL]).

43.8.1.45 MRC Global Access Control (MRC1_GLBAC0)

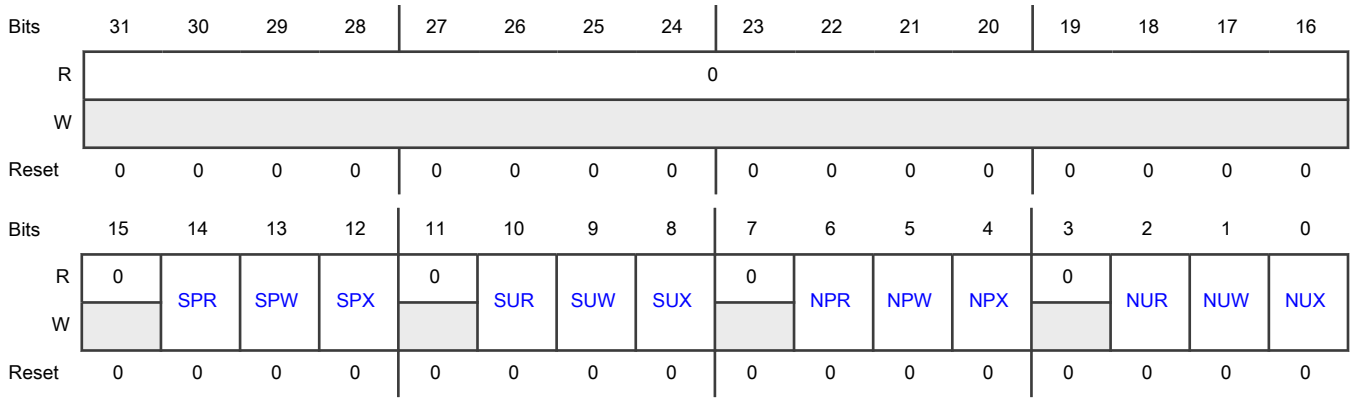
Offset

Register	Offset
MRC1_GLBAC0	1_5020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31-15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.1.46 MRC Global Access Control (MRC1_GLBAC1 - MRC1_GLBAC7)

Offset

Register	Offset
MRC1_GLBAC1	1_5024h
MRC1_GLBAC2	1_5028h

Table continues on the next page...

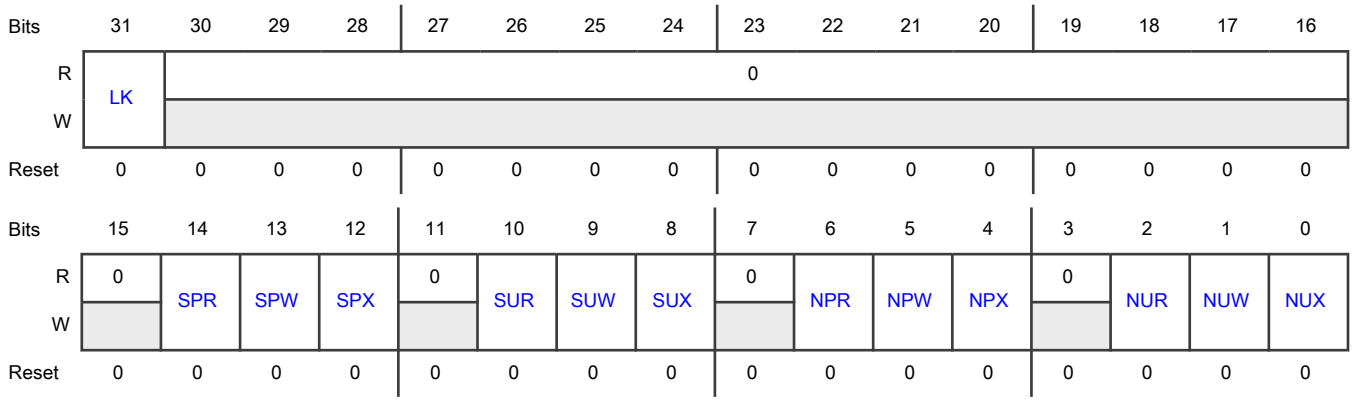
Table continued from the previous page...

Register	Offset
MRC1_GLBAC3	1_502Ch
MRC1_GLBAC4	1_5030h
MRC1_GLBAC5	1_5034h
MRC1_GLBAC6	1_5038h
MRC1_GLBAC7	1_503Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked (read-only) and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5	NonsecurePriv Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
NPW	NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2 TRDC register descriptions

43.8.2.1 TRDC memory map

TRDC2 base address: 4246_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRDC Register (TRDC_CR)	32	RW	0000_0010h
F0h	TRDC Hardware Configuration Register 0 (TRDC_HWCFG0)	32	R	2702_0610h
F4h	TRDC Hardware Configuration Register 1 (TRDC_HWCFG1)	32	R	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
F8h	TRDC Hardware Configuration Register 2 (TRDC_HWCFG2)	32	R	0000_0023h
FCh	TRDC Hardware Configuration Register 3 (TRDC_HWCFG3)	32	R	0000_0000h
100h - 105h	Domain Assignment Configuration Register (DACFG0 - DACFG5)	8	R	See section
1C0h	TRDC IDAU Control Register (TRDC_IDAU_CR)	32	RW	0000_0008h
1E0h	TRDC FLW Control (TRDC_FLW_CTL)	32	RW	0000_0000h
1E4h	TRDC FLW Physical Base (TRDC_FLW_PBASE)	32	R	0100_0000h
1E8h	TRDC FLW Array Base (TRDC_FLW_ABASE)	32	RW	0000_0000h
1ECh	TRDC FLW Block Count (TRDC_FLW_BCNT)	32	RW	0000_0000h
1FCh	TRDC Fault Domain ID (TRDC_FDID)	32	RW	0000_0000h
200h - 23Ch	TRDC Domain Error Location Register (TRDC_DERRLOC0 - TRDC_DERRLOC15)	32	R	0000_0000h
400h	MBC Domain Error Word0 Register (MBC0_DERR_W0)	32	R	0000_0000h
404h	MBC Domain Error Word1 Register (MBC0_DERR_W1)	32	R	0000_0000h
40Ch	MBC Domain Error Word3 Register (MBC0_DERR_W3)	32	RW	0000_0000h
410h	MBC Domain Error Word0 Register (MBC1_DERR_W0)	32	R	0000_0000h
414h	MBC Domain Error Word1 Register (MBC1_DERR_W1)	32	R	0000_0000h
41Ch	MBC Domain Error Word3 Register (MBC1_DERR_W3)	32	RW	0000_0000h
480h	MRC Domain Error Word0 Register (MRC0_DERR_W0)	32	R	0000_0000h
484h	MRC Domain Error Word1 Register (MRC0_DERR_W1)	32	R	0000_0000h
48Ch	MRC Domain Error Word3 Register (MRC0_DERR_W3)	32	RW	0000_0000h
490h	MRC Domain Error Word0 Register (MRC1_DERR_W0)	32	R	0000_0000h
494h	MRC Domain Error Word1 Register (MRC1_DERR_W1)	32	R	0000_0000h
49Ch	MRC Domain Error Word3 Register (MRC1_DERR_W3)	32	RW	0000_0000h
4A0h	MRC Domain Error Word0 Register (MRC2_DERR_W0)	32	R	0000_0000h
4A4h	MRC Domain Error Word1 Register (MRC2_DERR_W1)	32	R	0000_0000h
4ACh	MRC Domain Error Word3 Register (MRC2_DERR_W3)	32	RW	0000_0000h
4B0h	MRC Domain Error Word0 Register (MRC3_DERR_W0)	32	R	0000_0000h
4B4h	MRC Domain Error Word1 Register (MRC3_DERR_W1)	32	R	0000_0000h
4BCh	MRC Domain Error Word3 Register (MRC3_DERR_W3)	32	RW	0000_0000h
4C0h	MRC Domain Error Word0 Register (MRC4_DERR_W0)	32	R	0000_0000h
4C4h	MRC Domain Error Word1 Register (MRC4_DERR_W1)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4CCh	MRC Domain Error Word3 Register (MRC4_DERR_W3)	32	RW	0000_0000h
4D0h	MRC Domain Error Word0 Register (MRC5_DERR_W0)	32	R	0000_0000h
4D4h	MRC Domain Error Word1 Register (MRC5_DERR_W1)	32	R	0000_0000h
4DCh	MRC Domain Error Word3 Register (MRC5_DERR_W3)	32	RW	0000_0000h
4E0h	MRC Domain Error Word0 Register (MRC6_DERR_W0)	32	R	0000_0000h
4E4h	MRC Domain Error Word1 Register (MRC6_DERR_W1)	32	R	0000_0000h
4ECh	MRC Domain Error Word3 Register (MRC6_DERR_W3)	32	RW	0000_0000h
800h	DAC Master Domain Assignment Register (MDA_W0_0_DFMT0)	32	RW	0000_0000h
804h	DAC Master Domain Assignment Register (MDA_W1_0_DFMT0)	32	RW	0000_0000h
820h	DAC Master Domain Assignment Register (MDA_W0_1_DFMT0)	32	RW	0000_0000h
824h	DAC Master Domain Assignment Register (MDA_W1_1_DFMT0)	32	RW	0000_0000h
828h	DAC Master Domain Assignment Register (MDA_W2_1_DFMT0)	32	RW	0000_0000h
840h	DAC Master Domain Assignment Register (MDA_W0_2_DFMT1)	32	RW	2000_0000h
860h	DAC Master Domain Assignment Register (MDA_W0_3_DFMT1)	32	RW	2000_0000h
880h	DAC Master Domain Assignment Register (MDA_W0_4_DFMT1)	32	RW	2000_0000h
8A0h	DAC Master Domain Assignment Register (MDA_W0_5_DFMT0)	32	RW	0000_0000h
8A4h	DAC Master Domain Assignment Register (MDA_W1_5_DFMT0)	32	RW	0000_0000h
8A8h	DAC Master Domain Assignment Register (MDA_W2_5_DFMT0)	32	RW	0000_0000h
8ACh	DAC Master Domain Assignment Register (MDA_W3_5_DFMT0)	32	RW	0000_0000h
8B0h	DAC Master Domain Assignment Register (MDA_W4_5_DFMT0)	32	RW	0000_0000h
8B4h	DAC Master Domain Assignment Register (MDA_W5_5_DFMT0)	32	RW	0000_0000h
8B8h	DAC Master Domain Assignment Register (MDA_W6_5_DFMT0)	32	RW	0000_0000h
1_0000h	MBC Global Configuration Register (MBC0_MEM0_GLBCFG)	32	R	0010_0080h
1_0004h	MBC Global Configuration Register (MBC0_MEM1_GLBCFG)	32	R	0010_0006h
1_0008h	MBC Global Configuration Register (MBC0_MEM2_GLBCFG)	32	R	0010_0005h
1_000Ch	MBC Global Configuration Register (MBC0_MEM3_GLBCFG)	32	R	0014_0011h
1_0010h	MBC NonSecure Enable Block Index (MBC0_NSE_BLK_INDEX)	32	RW	0000_0000h
1_0014h	MBC NonSecure Enable Block Set (MBC0_NSE_BLK_SET)	32	W	0000_0000h
1_0018h	MBC NonSecure Enable Block Clear (MBC0_NSE_BLK_CLR)	32	W	0000_0000h
1_001Ch	MBC NonSecure Enable Block Clear All (MBC0_NSE_BLK_CLR_ALL)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0020h	MBC Global Access Control (MBC0_MEMN_GLBAC0)	32	RW	0000_0000h
1_0024h	MBC Global Access Control (MBC0_MEMN_GLBAC1)	32	RW	0000_0000h
1_0028h	MBC Global Access Control (MBC0_MEMN_GLBAC2)	32	RW	0000_0000h
1_002Ch	MBC Global Access Control (MBC0_MEMN_GLBAC3)	32	RW	0000_0000h
1_0030h	MBC Global Access Control (MBC0_MEMN_GLBAC4)	32	RW	0000_0000h
1_0034h	MBC Global Access Control (MBC0_MEMN_GLBAC5)	32	RW	0000_0000h
1_0038h	MBC Global Access Control (MBC0_MEMN_GLBAC6)	32	RW	0000_0000h
1_003Ch	MBC Global Access Control (MBC0_MEMN_GLBAC7)	32	RW	0000_0000h
1_0040h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0044h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0048h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_004Ch	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0050h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0054h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0058h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_005Ch	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0060h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0064h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0068h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_006Ch	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0070h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0074h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W13)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0078h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_007Ch	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0140h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0144h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0148h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_014Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0180h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_01A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_01A8h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_01C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_01D0h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_01D4h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_01D8h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_01F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0240h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0244h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0248h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_024Ch	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0250h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0254h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0258h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_025Ch	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0260h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0264h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0268h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_026Ch	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0270h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0274h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0278h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_027Ch	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0340h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0344h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0348h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_034Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0380h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_03A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM1_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_03A8h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_03C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_03D0h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_03D4h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_03D8h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_03F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0440h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0444h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0448h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_044Ch	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0450h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0454h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0458h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_045Ch	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0460h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0464h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0468h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_046Ch	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W11)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0470h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0474h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0478h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_047Ch	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0540h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0544h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0548h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_054Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0580h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_05A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_05A8h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_05C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_05D0h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_05D4h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_05D8h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_05F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0640h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0644h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0648h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_064Ch	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0650h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0654h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0658h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_065Ch	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0660h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0664h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0668h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_066Ch	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0670h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0674h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0678h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_067Ch	MBC Memory Block Configuration Word (MBC0_DOM3_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0740h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0744h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0748h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_074Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM0_BLK_NSE_W3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0780h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_07A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_07A8h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_07C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_07D0h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_07D4h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_07D8h	MBC Memory Block Configuration Word (MBC0_DOM3_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_07F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM3_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0840h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0844h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0848h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_084Ch	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0850h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0854h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0858h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_085Ch	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0860h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0864h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W9)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0868h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_086Ch	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0870h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0874h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0878h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_087Ch	MBC Memory Block Configuration Word (MBC0_DOM4_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0940h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0944h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0948h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_094Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0980h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_09A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_09A8h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_09C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_09D0h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_09D4h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_09D8h	MBC Memory Block Configuration Word (MBC0_DOM4_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_09F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM4_MEM3_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0A40h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0A44h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0A48h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_0A4Ch	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0A50h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0A54h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0A58h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_0A5Ch	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0A60h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0A64h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0A68h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_0A6Ch	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0A70h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0A74h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0A78h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_0A7Ch	MBC Memory Block Configuration Word (MBC0_DOM5_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0B40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0B44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM0_BLK_NSE_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0B48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_0B4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0B80h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_0BA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_0BA8h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_0BC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_0BD0h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_0BD4h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_0BD8h	MBC Memory Block Configuration Word (MBC0_DOM5_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_0BF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM5_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0C40h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0C44h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0C48h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_0C4Ch	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0C50h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0C54h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0C58h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_0C5Ch	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W7)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0C60h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0C64h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0C68h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_0C6Ch	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0C70h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0C74h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0C78h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_0C7Ch	MBC Memory Block Configuration Word (MBC0_DOM6_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_0D40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0D44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0D48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_0D4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0D80h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_0DA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_0DA8h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_0DC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_0DD0h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_0DD4h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM3_BLK_CFG_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0DD8h	MBC Memory Block Configuration Word (MBC0_DOM6_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_0DF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM6_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_0E40h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_0E44h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_0E48h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_0E4Ch	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_0E50h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_0E54h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_0E58h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_0E5Ch	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_0E60h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_0E64h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_0E68h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_0E6Ch	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_0E70h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_0E74h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_0E78h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_0E7Ch	MBC Memory Block Configuration Word (MBC0_DOM7_MEM0_BLK_CFG_W15)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_0F40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_0F44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_0F48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_0F4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_0F80h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_0FA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_0FA8h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_0FC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_0FD0h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_0FD4h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_0FD8h	MBC Memory Block Configuration Word (MBC0_DOM7_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_0FF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM7_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1040h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1044h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1048h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_104Ch	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1050h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1054h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W5)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1058h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_105Ch	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1060h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1064h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1068h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_106Ch	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1070h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1074h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1078h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_107Ch	MBC Memory Block Configuration Word (MBC0_DOM8_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1140h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1144h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1148h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_114Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1180h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_11A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_11A8h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_11C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM2_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_11D0h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_11D4h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_11D8h	MBC Memory Block Configuration Word (MBC0_DOM8_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_11F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM8_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1240h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1244h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1248h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_124Ch	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1250h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1254h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1258h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_125Ch	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1260h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1264h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1268h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_126Ch	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1270h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1274h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W13)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1278h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_127Ch	MBC Memory Block Configuration Word (MBC0_DOM9_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1340h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1344h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1348h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_134Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1380h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_13A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_13A8h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_13C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_13D0h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_13D4h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_13D8h	MBC Memory Block Configuration Word (MBC0_DOM9_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_13F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM9_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1440h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1444h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1448h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_144Ch	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1450h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1454h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1458h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_145Ch	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1460h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1464h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1468h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_146Ch	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1470h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1474h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1478h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_147Ch	MBC Memory Block Configuration Word (MBC0_DOM10_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1540h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1544h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1548h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_154Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1580h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_15A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM1_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_15A8h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_15C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_15D0h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_15D4h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_15D8h	MBC Memory Block Configuration Word (MBC0_DOM10_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_15F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM10_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1640h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1644h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1648h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_164Ch	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1650h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1654h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1658h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_165Ch	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1660h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1664h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1668h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_166Ch	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W11)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1670h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1674h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1678h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_167Ch	MBC Memory Block Configuration Word (MBC0_DOM11_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1740h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1744h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1748h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_174Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1780h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_17A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_17A8h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_17C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_17D0h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_17D4h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_17D8h	MBC Memory Block Configuration Word (MBC0_DOM11_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_17F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM11_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1840h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1844h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1848h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_184Ch	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1850h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1854h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1858h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_185Ch	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1860h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1864h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1868h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_186Ch	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1870h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1874h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1878h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_187Ch	MBC Memory Block Configuration Word (MBC0_DOM12_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1940h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1944h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1948h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_194Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM0_BLK_NSE_W3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1980h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_19A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_19A8h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_19C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_19D0h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_19D4h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_19D8h	MBC Memory Block Configuration Word (MBC0_DOM12_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_19F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM12_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1A40h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1A44h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1A48h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_1A4Ch	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1A50h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1A54h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1A58h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_1A5Ch	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1A60h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1A64h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W9)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1A68h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_1A6Ch	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1A70h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1A74h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1A78h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_1A7Ch	MBC Memory Block Configuration Word (MBC0_DOM13_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1B40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1B44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1B48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_1B4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1B80h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_1BA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_1BA8h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_1BC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_1BD0h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_1BD4h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_1BD8h	MBC Memory Block Configuration Word (MBC0_DOM13_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_1BF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM13_MEM3_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1C40h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1C44h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1C48h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_1C4Ch	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1C50h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1C54h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1C58h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_1C5Ch	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_1C60h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1C64h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1C68h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_1C6Ch	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1C70h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1C74h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1C78h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_1C7Ch	MBC Memory Block Configuration Word (MBC0_DOM14_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1D40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1D44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM0_BLK_NSE_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1D48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_1D4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1D80h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_1DA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_1DA8h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_1DC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_1DD0h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_1DD4h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
1_1DD8h	MBC Memory Block Configuration Word (MBC0_DOM14_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_1DF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM14_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_1E40h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_1E44h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_1E48h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_1E4Ch	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_1E50h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_1E54h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_1E58h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_1E5Ch	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W7)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1E60h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_1E64h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_1E68h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_1E6Ch	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_1E70h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_1E74h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_1E78h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_1E7Ch	MBC Memory Block Configuration Word (MBC0_DOM15_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_1F40h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_1F44h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_1F48h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_1F4Ch	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_1F80h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_1FA0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_1FA8h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_1FC8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_1FD0h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_1FD4h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM3_BLK_CFG_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_1FD8h	MBC Memory Block Configuration Word (MBC0_DOM15_MEM3_BLK_CFG_W2)	32	RW	0000_0000h
1_1FF0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM15_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_2000h	MBC Global Configuration Register (MBC1_MEM0_GLBCFG)	32	R	0010_0080h
1_2004h	MBC Global Configuration Register (MBC1_MEM1_GLBCFG)	32	R	0010_0001h
1_2008h	MBC Global Configuration Register (MBC1_MEM2_GLBCFG)	32	R	0014_0001h
1_200Ch	MBC Global Configuration Register (MBC1_MEM3_GLBCFG)	32	R	0013_0001h
1_2010h	MBC NonSecure Enable Block Index (MBC1_NSE_BLK_INDEX)	32	RW	0000_0000h
1_2014h	MBC NonSecure Enable Block Set (MBC1_NSE_BLK_SET)	32	W	0000_0000h
1_2018h	MBC NonSecure Enable Block Clear (MBC1_NSE_BLK_CLR)	32	W	0000_0000h
1_201Ch	MBC NonSecure Enable Block Clear All (MBC1_NSE_BLK_CLR_ALL)	32	RW	0000_0000h
1_2020h	MBC Global Access Control (MBC1_MEMN_GLBAC0)	32	RW	0000_0000h
1_2024h	MBC Global Access Control (MBC1_MEMN_GLBAC1)	32	RW	0000_0000h
1_2028h	MBC Global Access Control (MBC1_MEMN_GLBAC2)	32	RW	0000_0000h
1_202Ch	MBC Global Access Control (MBC1_MEMN_GLBAC3)	32	RW	0000_0000h
1_2030h	MBC Global Access Control (MBC1_MEMN_GLBAC4)	32	RW	0000_0000h
1_2034h	MBC Global Access Control (MBC1_MEMN_GLBAC5)	32	RW	0000_0000h
1_2038h	MBC Global Access Control (MBC1_MEMN_GLBAC6)	32	RW	0000_0000h
1_203Ch	MBC Global Access Control (MBC1_MEMN_GLBAC7)	32	RW	0000_0000h
1_2040h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2044h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2048h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_204Ch	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2050h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_2054h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W5)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2058h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_205Ch	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_2060h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_2064h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_2068h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_206Ch	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_2070h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_2074h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_2078h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_207Ch	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_2140h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2144h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_2148h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_214Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_2180h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_21A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_21A8h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_21C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM2_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_21D0h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_21F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_2240h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2244h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2248h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_224Ch	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2250h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_2254h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_2258h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_225Ch	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_2260h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_2264h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_2268h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_226Ch	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_2270h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_2274h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_2278h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_227Ch	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W15)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2340h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2344h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_2348h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_234Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_2380h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_23A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_23A8h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_23C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_23D0h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_23F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_2440h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2444h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2448h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_244Ch	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2450h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_2454h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_2458h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_245Ch	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W7)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2460h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_2464h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_2468h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_246Ch	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_2470h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_2474h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_2478h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_247Ch	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_2540h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2544h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_2548h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_254Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_2580h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_25A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_25A8h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_25C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_25D0h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_25F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM3_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2640h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2644h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2648h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_264Ch	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2650h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_2654h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_2658h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_265Ch	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_2660h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_2664h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_2668h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_266Ch	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_2670h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_2674h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_2678h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_267Ch	MBC Memory Block Configuration Word (MBC1_DOM3_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_2740h	MBC Memory Block NonSecure Enable Word (MBC1_DOM3_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2744h	MBC Memory Block NonSecure Enable Word (MBC1_DOM3_MEM0_BLK_NSE_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2748h	MBC Memory Block NonSecure Enable Word (MBC1_DOM3_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_274Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM3_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_2780h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_27A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM3_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_27A8h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_27C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM3_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_27D0h	MBC Memory Block Configuration Word (MBC1_DOM3_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_27F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM3_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_2840h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2844h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2848h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_284Ch	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2850h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_2854h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_2858h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_285Ch	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_2860h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_2864h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W9)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2868h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_286Ch	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_2870h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_2874h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_2878h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_287Ch	MBC Memory Block Configuration Word (MBC1_DOM4_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_2940h	MBC Memory Block NonSecure Enable Word (MBC1_DOM4_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2944h	MBC Memory Block NonSecure Enable Word (MBC1_DOM4_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_2948h	MBC Memory Block NonSecure Enable Word (MBC1_DOM4_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_294Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM4_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_2980h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_29A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM4_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_29A8h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_29C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM4_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_29D0h	MBC Memory Block Configuration Word (MBC1_DOM4_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_29F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM4_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_2A40h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2A44h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2A48h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_2A4Ch	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2A50h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_2A54h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_2A58h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_2A5Ch	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_2A60h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_2A64h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_2A68h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_2A6Ch	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_2A70h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_2A74h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_2A78h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_2A7Ch	MBC Memory Block Configuration Word (MBC1_DOM5_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_2B40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM5_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2B44h	MBC Memory Block NonSecure Enable Word (MBC1_DOM5_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_2B48h	MBC Memory Block NonSecure Enable Word (MBC1_DOM5_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_2B4Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM5_MEM0_BLK_NSE_W3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2B80h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_2BA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM5_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_2BA8h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_2BC8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM5_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_2BD0h	MBC Memory Block Configuration Word (MBC1_DOM5_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_2BF0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM5_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_2C40h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2C44h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2C48h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_2C4Ch	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_2C50h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_2C54h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_2C58h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_2C5Ch	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_2C60h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_2C64h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_2C68h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_2C6Ch	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W11)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2C70h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_2C74h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_2C78h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_2C7Ch	MBC Memory Block Configuration Word (MBC1_DOM6_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_2D40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM6_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2D44h	MBC Memory Block NonSecure Enable Word (MBC1_DOM6_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_2D48h	MBC Memory Block NonSecure Enable Word (MBC1_DOM6_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_2D4Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM6_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_2D80h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_2DA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM6_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_2DA8h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_2DC8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM6_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_2DD0h	MBC Memory Block Configuration Word (MBC1_DOM6_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_2DF0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM6_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_2E40h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_2E44h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_2E48h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_2E4Ch	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2E50h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_2E54h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_2E58h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_2E5Ch	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_2E60h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_2E64h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_2E68h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_2E6Ch	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_2E70h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_2E74h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_2E78h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_2E7Ch	MBC Memory Block Configuration Word (MBC1_DOM7_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_2F40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM7_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_2F44h	MBC Memory Block NonSecure Enable Word (MBC1_DOM7_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_2F48h	MBC Memory Block NonSecure Enable Word (MBC1_DOM7_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_2F4Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM7_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_2F80h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_2FA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM7_MEM1_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_2FA8h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_2FC8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM7_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_2FD0h	MBC Memory Block Configuration Word (MBC1_DOM7_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_2FF0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM7_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_3040h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3044h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3048h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_304Ch	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3050h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_3054h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_3058h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_305Ch	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_3060h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_3064h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_3068h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_306Ch	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_3070h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_3074h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W13)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3078h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_307Ch	MBC Memory Block Configuration Word (MBC1_DOM8_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_3140h	MBC Memory Block NonSecure Enable Word (MBC1_DOM8_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3144h	MBC Memory Block NonSecure Enable Word (MBC1_DOM8_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_3148h	MBC Memory Block NonSecure Enable Word (MBC1_DOM8_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_314Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM8_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_3180h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_31A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM8_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_31A8h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_31C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM8_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_31D0h	MBC Memory Block Configuration Word (MBC1_DOM8_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_31F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM8_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_3240h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3244h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3248h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_324Ch	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3250h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_3254h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W5)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3258h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_325Ch	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_3260h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_3264h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_3268h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_326Ch	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_3270h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_3274h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_3278h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_327Ch	MBC Memory Block Configuration Word (MBC1_DOM9_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_3340h	MBC Memory Block NonSecure Enable Word (MBC1_DOM9_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3344h	MBC Memory Block NonSecure Enable Word (MBC1_DOM9_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_3348h	MBC Memory Block NonSecure Enable Word (MBC1_DOM9_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_334Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM9_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_3380h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_33A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM9_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_33A8h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_33C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM9_MEM2_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_33D0h	MBC Memory Block Configuration Word (MBC1_DOM9_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_33F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM9_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_3440h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3444h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3448h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_344Ch	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3450h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_3454h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_3458h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_345Ch	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_3460h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_3464h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_3468h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_346Ch	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_3470h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_3474h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_3478h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_347Ch	MBC Memory Block Configuration Word (MBC1_DOM10_MEM0_BLK_CFG_W15)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3540h	MBC Memory Block NonSecure Enable Word (MBC1_DOM10_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3544h	MBC Memory Block NonSecure Enable Word (MBC1_DOM10_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_3548h	MBC Memory Block NonSecure Enable Word (MBC1_DOM10_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_354Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM10_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_3580h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_35A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM10_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_35A8h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_35C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM10_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_35D0h	MBC Memory Block Configuration Word (MBC1_DOM10_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_35F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM10_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_3640h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3644h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3648h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_364Ch	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3650h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_3654h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_3658h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_365Ch	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W7)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3660h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_3664h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_3668h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_366Ch	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_3670h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_3674h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_3678h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_367Ch	MBC Memory Block Configuration Word (MBC1_DOM11_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_3740h	MBC Memory Block NonSecure Enable Word (MBC1_DOM11_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3744h	MBC Memory Block NonSecure Enable Word (MBC1_DOM11_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_3748h	MBC Memory Block NonSecure Enable Word (MBC1_DOM11_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_374Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM11_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_3780h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_37A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM11_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_37A8h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_37C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM11_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_37D0h	MBC Memory Block Configuration Word (MBC1_DOM11_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_37F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM11_MEM3_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3840h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3844h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3848h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_384Ch	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3850h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_3854h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_3858h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_385Ch	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_3860h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_3864h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_3868h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_386Ch	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_3870h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_3874h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_3878h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_387Ch	MBC Memory Block Configuration Word (MBC1_DOM12_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_3940h	MBC Memory Block NonSecure Enable Word (MBC1_DOM12_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3944h	MBC Memory Block NonSecure Enable Word (MBC1_DOM12_MEM0_BLK_NSE_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3948h	MBC Memory Block NonSecure Enable Word (MBC1_DOM12_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_394Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM12_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_3980h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_39A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM12_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_39A8h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_39C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM12_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_39D0h	MBC Memory Block Configuration Word (MBC1_DOM12_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_39F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM12_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_3A40h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3A44h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3A48h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_3A4Ch	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3A50h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_3A54h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_3A58h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_3A5Ch	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_3A60h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_3A64h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W9)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3A68h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_3A6Ch	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_3A70h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_3A74h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_3A78h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_3A7Ch	MBC Memory Block Configuration Word (MBC1_DOM13_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_3B40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM13_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3B44h	MBC Memory Block NonSecure Enable Word (MBC1_DOM13_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_3B48h	MBC Memory Block NonSecure Enable Word (MBC1_DOM13_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_3B4Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM13_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_3B80h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_3BA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM13_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_3BA8h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_3BC8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM13_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_3BD0h	MBC Memory Block Configuration Word (MBC1_DOM13_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_3BF0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM13_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_3C40h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3C44h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3C48h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_3C4Ch	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3C50h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_3C54h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_3C58h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_3C5Ch	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_3C60h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_3C64h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_3C68h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_3C6Ch	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W11)	32	RW	0000_0000h
1_3C70h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_3C74h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_3C78h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_3C7Ch	MBC Memory Block Configuration Word (MBC1_DOM14_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_3D40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM14_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3D44h	MBC Memory Block NonSecure Enable Word (MBC1_DOM14_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_3D48h	MBC Memory Block NonSecure Enable Word (MBC1_DOM14_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_3D4Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM14_MEM0_BLK_NSE_W3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3D80h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_3DA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM14_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_3DA8h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_3DC8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM14_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_3DD0h	MBC Memory Block Configuration Word (MBC1_DOM14_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_3DF0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM14_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_3E40h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1_3E44h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1_3E48h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
1_3E4Ch	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1_3E50h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
1_3E54h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
1_3E58h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
1_3E5Ch	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
1_3E60h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
1_3E64h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
1_3E68h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W10)	32	RW	0000_0000h
1_3E6Ch	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W11)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_3E70h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W12)	32	RW	0000_0000h
1_3E74h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W13)	32	RW	0000_0000h
1_3E78h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W14)	32	RW	0000_0000h
1_3E7Ch	MBC Memory Block Configuration Word (MBC1_DOM15_MEM0_BLK_CFG_W15)	32	RW	0000_0000h
1_3F40h	MBC Memory Block NonSecure Enable Word (MBC1_DOM15_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1_3F44h	MBC Memory Block NonSecure Enable Word (MBC1_DOM15_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
1_3F48h	MBC Memory Block NonSecure Enable Word (MBC1_DOM15_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
1_3F4Ch	MBC Memory Block NonSecure Enable Word (MBC1_DOM15_MEM0_BLK_NSE_W3)	32	RW	0000_0000h
1_3F80h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1_3FA0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM15_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
1_3FA8h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
1_3FC8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM15_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
1_3FD0h	MBC Memory Block Configuration Word (MBC1_DOM15_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
1_3FF0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM15_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1_4000h	MRC Global Configuration Register (MRC0_GLBCFG)	32	R	0000_0008h
1_4010h	MRC NonSecure Enable Region Indirect (MRC0_NSE_RGN_INDIRECT)	32	RW	0000_0000h
1_4014h	MRC NonSecure Enable Region Set (MRC0_NSE_RGN_SET)	32	RW	0000_0000h
1_4018h	MRC NonSecure Enable Region Clear (MRC0_NSE_RGN_CLR)	32	RW	0000_0000h
1_401Ch	MRC NonSecure Enable Region Clear All (MRC0_NSE_RGN_CLR_ALL)	32	RW	0000_0000h
1_4020h	MRC Global Access Control (MRC0_GLBAC0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4024h	MRC Global Access Control (MRC0_GLBAC1)	32	RW	0000_0000h
1_4028h	MRC Global Access Control (MRC0_GLBAC2)	32	RW	0000_0000h
1_402Ch	MRC Global Access Control (MRC0_GLBAC3)	32	RW	0000_0000h
1_4030h	MRC Global Access Control (MRC0_GLBAC4)	32	RW	0000_0000h
1_4034h	MRC Global Access Control (MRC0_GLBAC5)	32	RW	0000_0000h
1_4038h	MRC Global Access Control (MRC0_GLBAC6)	32	RW	0000_0000h
1_403Ch	MRC Global Access Control (MRC0_GLBAC7)	32	RW	0000_0000h
1_4040h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD0_W0)	32	RW	0000_0000h
1_4044h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD0_W1)	32	RW	0000_0000h
1_4048h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD1_W0)	32	RW	0000_0000h
1_404Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD1_W1)	32	RW	0000_0000h
1_4050h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD2_W0)	32	RW	0000_0000h
1_4054h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD2_W1)	32	RW	0000_0000h
1_4058h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD3_W0)	32	RW	0000_0000h
1_405Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD3_W1)	32	RW	0000_0000h
1_4060h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD4_W0)	32	RW	0000_0000h
1_4064h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD4_W1)	32	RW	0000_0000h
1_4068h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD5_W0)	32	RW	0000_0000h
1_406Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD5_W1)	32	RW	0000_0000h
1_4070h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD6_W0)	32	RW	0000_0000h
1_4074h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD6_W1)	32	RW	0000_0000h
1_4078h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD7_W0)	32	RW	0000_0000h
1_407Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD7_W1)	32	RW	0000_0000h
1_40C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM0_RGD_NSE)	32	RW	0000_0000h
1_4140h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD0_W0)	32	RW	0000_0000h
1_4144h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD0_W1)	32	RW	0000_0000h
1_4148h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD1_W0)	32	RW	0000_0000h
1_414Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD1_W1)	32	RW	0000_0000h
1_4150h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD2_W0)	32	RW	0000_0000h
1_4154h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD2_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4158h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD3_W0)	32	RW	0000_0000h
1_415Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD3_W1)	32	RW	0000_0000h
1_4160h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD4_W0)	32	RW	0000_0000h
1_4164h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD4_W1)	32	RW	0000_0000h
1_4168h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD5_W0)	32	RW	0000_0000h
1_416Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD5_W1)	32	RW	0000_0000h
1_4170h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD6_W0)	32	RW	0000_0000h
1_4174h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD6_W1)	32	RW	0000_0000h
1_4178h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD7_W0)	32	RW	0000_0000h
1_417Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD7_W1)	32	RW	0000_0000h
1_41C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM1_RGD_NSE)	32	RW	0000_0000h
1_4240h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD0_W0)	32	RW	0000_0000h
1_4244h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD0_W1)	32	RW	0000_0000h
1_4248h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD1_W0)	32	RW	0000_0000h
1_424Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD1_W1)	32	RW	0000_0000h
1_4250h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD2_W0)	32	RW	0000_0000h
1_4254h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD2_W1)	32	RW	0000_0000h
1_4258h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD3_W0)	32	RW	0000_0000h
1_425Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD3_W1)	32	RW	0000_0000h
1_4260h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD4_W0)	32	RW	0000_0000h
1_4264h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD4_W1)	32	RW	0000_0000h
1_4268h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD5_W0)	32	RW	0000_0000h
1_426Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD5_W1)	32	RW	0000_0000h
1_4270h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD6_W0)	32	RW	0000_0000h
1_4274h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD6_W1)	32	RW	0000_0000h
1_4278h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD7_W0)	32	RW	0000_0000h
1_427Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD7_W1)	32	RW	0000_0000h
1_42C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM2_RGD_NSE)	32	RW	0000_0000h
1_4340h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD0_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4344h	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD0_W1)	32	RW	0000_0000h
1_4348h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD1_W0)	32	RW	0000_0000h
1_434Ch	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD1_W1)	32	RW	0000_0000h
1_4350h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD2_W0)	32	RW	0000_0000h
1_4354h	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD2_W1)	32	RW	0000_0000h
1_4358h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD3_W0)	32	RW	0000_0000h
1_435Ch	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD3_W1)	32	RW	0000_0000h
1_4360h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD4_W0)	32	RW	0000_0000h
1_4364h	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD4_W1)	32	RW	0000_0000h
1_4368h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD5_W0)	32	RW	0000_0000h
1_436Ch	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD5_W1)	32	RW	0000_0000h
1_4370h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD6_W0)	32	RW	0000_0000h
1_4374h	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD6_W1)	32	RW	0000_0000h
1_4378h	MRC Region Descriptor Word 0 (MRC0_DOM3_RGD7_W0)	32	RW	0000_0000h
1_437Ch	MRC Region Descriptor Word 1 (MRC0_DOM3_RGD7_W1)	32	RW	0000_0000h
1_43C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM3_RGD_NSE)	32	RW	0000_0000h
1_4440h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD0_W0)	32	RW	0000_0000h
1_4444h	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD0_W1)	32	RW	0000_0000h
1_4448h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD1_W0)	32	RW	0000_0000h
1_444Ch	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD1_W1)	32	RW	0000_0000h
1_4450h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD2_W0)	32	RW	0000_0000h
1_4454h	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD2_W1)	32	RW	0000_0000h
1_4458h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD3_W0)	32	RW	0000_0000h
1_445Ch	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD3_W1)	32	RW	0000_0000h
1_4460h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD4_W0)	32	RW	0000_0000h
1_4464h	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD4_W1)	32	RW	0000_0000h
1_4468h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD5_W0)	32	RW	0000_0000h
1_446Ch	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD5_W1)	32	RW	0000_0000h
1_4470h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD6_W0)	32	RW	0000_0000h
1_4474h	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD6_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4478h	MRC Region Descriptor Word 0 (MRC0_DOM4_RGD7_W0)	32	RW	0000_0000h
1_447Ch	MRC Region Descriptor Word 1 (MRC0_DOM4_RGD7_W1)	32	RW	0000_0000h
1_44C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM4_RGD_NSE)	32	RW	0000_0000h
1_4540h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD0_W0)	32	RW	0000_0000h
1_4544h	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD0_W1)	32	RW	0000_0000h
1_4548h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD1_W0)	32	RW	0000_0000h
1_454Ch	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD1_W1)	32	RW	0000_0000h
1_4550h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD2_W0)	32	RW	0000_0000h
1_4554h	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD2_W1)	32	RW	0000_0000h
1_4558h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD3_W0)	32	RW	0000_0000h
1_455Ch	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD3_W1)	32	RW	0000_0000h
1_4560h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD4_W0)	32	RW	0000_0000h
1_4564h	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD4_W1)	32	RW	0000_0000h
1_4568h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD5_W0)	32	RW	0000_0000h
1_456Ch	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD5_W1)	32	RW	0000_0000h
1_4570h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD6_W0)	32	RW	0000_0000h
1_4574h	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD6_W1)	32	RW	0000_0000h
1_4578h	MRC Region Descriptor Word 0 (MRC0_DOM5_RGD7_W0)	32	RW	0000_0000h
1_457Ch	MRC Region Descriptor Word 1 (MRC0_DOM5_RGD7_W1)	32	RW	0000_0000h
1_45C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM5_RGD_NSE)	32	RW	0000_0000h
1_4640h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD0_W0)	32	RW	0000_0000h
1_4644h	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD0_W1)	32	RW	0000_0000h
1_4648h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD1_W0)	32	RW	0000_0000h
1_464Ch	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD1_W1)	32	RW	0000_0000h
1_4650h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD2_W0)	32	RW	0000_0000h
1_4654h	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD2_W1)	32	RW	0000_0000h
1_4658h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD3_W0)	32	RW	0000_0000h
1_465Ch	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD3_W1)	32	RW	0000_0000h
1_4660h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD4_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4664h	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD4_W1)	32	RW	0000_0000h
1_4668h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD5_W0)	32	RW	0000_0000h
1_466Ch	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD5_W1)	32	RW	0000_0000h
1_4670h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD6_W0)	32	RW	0000_0000h
1_4674h	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD6_W1)	32	RW	0000_0000h
1_4678h	MRC Region Descriptor Word 0 (MRC0_DOM6_RGD7_W0)	32	RW	0000_0000h
1_467Ch	MRC Region Descriptor Word 1 (MRC0_DOM6_RGD7_W1)	32	RW	0000_0000h
1_46C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM6_RGD_NSE)	32	RW	0000_0000h
1_4740h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD0_W0)	32	RW	0000_0000h
1_4744h	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD0_W1)	32	RW	0000_0000h
1_4748h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD1_W0)	32	RW	0000_0000h
1_474Ch	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD1_W1)	32	RW	0000_0000h
1_4750h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD2_W0)	32	RW	0000_0000h
1_4754h	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD2_W1)	32	RW	0000_0000h
1_4758h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD3_W0)	32	RW	0000_0000h
1_475Ch	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD3_W1)	32	RW	0000_0000h
1_4760h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD4_W0)	32	RW	0000_0000h
1_4764h	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD4_W1)	32	RW	0000_0000h
1_4768h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD5_W0)	32	RW	0000_0000h
1_476Ch	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD5_W1)	32	RW	0000_0000h
1_4770h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD6_W0)	32	RW	0000_0000h
1_4774h	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD6_W1)	32	RW	0000_0000h
1_4778h	MRC Region Descriptor Word 0 (MRC0_DOM7_RGD7_W0)	32	RW	0000_0000h
1_477Ch	MRC Region Descriptor Word 1 (MRC0_DOM7_RGD7_W1)	32	RW	0000_0000h
1_47C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM7_RGD_NSE)	32	RW	0000_0000h
1_4840h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD0_W0)	32	RW	0000_0000h
1_4844h	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD0_W1)	32	RW	0000_0000h
1_4848h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD1_W0)	32	RW	0000_0000h
1_484Ch	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD1_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4850h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD2_W0)	32	RW	0000_0000h
1_4854h	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD2_W1)	32	RW	0000_0000h
1_4858h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD3_W0)	32	RW	0000_0000h
1_485Ch	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD3_W1)	32	RW	0000_0000h
1_4860h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD4_W0)	32	RW	0000_0000h
1_4864h	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD4_W1)	32	RW	0000_0000h
1_4868h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD5_W0)	32	RW	0000_0000h
1_486Ch	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD5_W1)	32	RW	0000_0000h
1_4870h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD6_W0)	32	RW	0000_0000h
1_4874h	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD6_W1)	32	RW	0000_0000h
1_4878h	MRC Region Descriptor Word 0 (MRC0_DOM8_RGD7_W0)	32	RW	0000_0000h
1_487Ch	MRC Region Descriptor Word 1 (MRC0_DOM8_RGD7_W1)	32	RW	0000_0000h
1_48C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM8_RGD_NSE)	32	RW	0000_0000h
1_4940h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD0_W0)	32	RW	0000_0000h
1_4944h	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD0_W1)	32	RW	0000_0000h
1_4948h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD1_W0)	32	RW	0000_0000h
1_494Ch	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD1_W1)	32	RW	0000_0000h
1_4950h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD2_W0)	32	RW	0000_0000h
1_4954h	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD2_W1)	32	RW	0000_0000h
1_4958h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD3_W0)	32	RW	0000_0000h
1_495Ch	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD3_W1)	32	RW	0000_0000h
1_4960h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD4_W0)	32	RW	0000_0000h
1_4964h	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD4_W1)	32	RW	0000_0000h
1_4968h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD5_W0)	32	RW	0000_0000h
1_496Ch	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD5_W1)	32	RW	0000_0000h
1_4970h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD6_W0)	32	RW	0000_0000h
1_4974h	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD6_W1)	32	RW	0000_0000h
1_4978h	MRC Region Descriptor Word 0 (MRC0_DOM9_RGD7_W0)	32	RW	0000_0000h
1_497Ch	MRC Region Descriptor Word 1 (MRC0_DOM9_RGD7_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_49C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM9_RGD_NSE)	32	RW	0000_0000h
1_4A40h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD0_W0)	32	RW	0000_0000h
1_4A44h	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD0_W1)	32	RW	0000_0000h
1_4A48h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD1_W0)	32	RW	0000_0000h
1_4A4Ch	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD1_W1)	32	RW	0000_0000h
1_4A50h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD2_W0)	32	RW	0000_0000h
1_4A54h	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD2_W1)	32	RW	0000_0000h
1_4A58h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD3_W0)	32	RW	0000_0000h
1_4A5Ch	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD3_W1)	32	RW	0000_0000h
1_4A60h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD4_W0)	32	RW	0000_0000h
1_4A64h	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD4_W1)	32	RW	0000_0000h
1_4A68h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD5_W0)	32	RW	0000_0000h
1_4A6Ch	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD5_W1)	32	RW	0000_0000h
1_4A70h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD6_W0)	32	RW	0000_0000h
1_4A74h	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD6_W1)	32	RW	0000_0000h
1_4A78h	MRC Region Descriptor Word 0 (MRC0_DOM10_RGD7_W0)	32	RW	0000_0000h
1_4A7Ch	MRC Region Descriptor Word 1 (MRC0_DOM10_RGD7_W1)	32	RW	0000_0000h
1_4AC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM10_RGD_NSE)	32	RW	0000_0000h
1_4B40h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD0_W0)	32	RW	0000_0000h
1_4B44h	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD0_W1)	32	RW	0000_0000h
1_4B48h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD1_W0)	32	RW	0000_0000h
1_4B4Ch	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD1_W1)	32	RW	0000_0000h
1_4B50h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD2_W0)	32	RW	0000_0000h
1_4B54h	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD2_W1)	32	RW	0000_0000h
1_4B58h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD3_W0)	32	RW	0000_0000h
1_4B5Ch	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD3_W1)	32	RW	0000_0000h
1_4B60h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD4_W0)	32	RW	0000_0000h
1_4B64h	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD4_W1)	32	RW	0000_0000h
1_4B68h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD5_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4B6Ch	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD5_W1)	32	RW	0000_0000h
1_4B70h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD6_W0)	32	RW	0000_0000h
1_4B74h	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD6_W1)	32	RW	0000_0000h
1_4B78h	MRC Region Descriptor Word 0 (MRC0_DOM11_RGD7_W0)	32	RW	0000_0000h
1_4B7Ch	MRC Region Descriptor Word 1 (MRC0_DOM11_RGD7_W1)	32	RW	0000_0000h
1_4BC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM11_RGD_NSE)	32	RW	0000_0000h
1_4C40h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD0_W0)	32	RW	0000_0000h
1_4C44h	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD0_W1)	32	RW	0000_0000h
1_4C48h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD1_W0)	32	RW	0000_0000h
1_4C4Ch	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD1_W1)	32	RW	0000_0000h
1_4C50h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD2_W0)	32	RW	0000_0000h
1_4C54h	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD2_W1)	32	RW	0000_0000h
1_4C58h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD3_W0)	32	RW	0000_0000h
1_4C5Ch	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD3_W1)	32	RW	0000_0000h
1_4C60h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD4_W0)	32	RW	0000_0000h
1_4C64h	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD4_W1)	32	RW	0000_0000h
1_4C68h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD5_W0)	32	RW	0000_0000h
1_4C6Ch	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD5_W1)	32	RW	0000_0000h
1_4C70h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD6_W0)	32	RW	0000_0000h
1_4C74h	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD6_W1)	32	RW	0000_0000h
1_4C78h	MRC Region Descriptor Word 0 (MRC0_DOM12_RGD7_W0)	32	RW	0000_0000h
1_4C7Ch	MRC Region Descriptor Word 1 (MRC0_DOM12_RGD7_W1)	32	RW	0000_0000h
1_4CC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM12_RGD_NSE)	32	RW	0000_0000h
1_4D40h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD0_W0)	32	RW	0000_0000h
1_4D44h	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD0_W1)	32	RW	0000_0000h
1_4D48h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD1_W0)	32	RW	0000_0000h
1_4D4Ch	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD1_W1)	32	RW	0000_0000h
1_4D50h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD2_W0)	32	RW	0000_0000h
1_4D54h	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD2_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4D58h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD3_W0)	32	RW	0000_0000h
1_4D5Ch	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD3_W1)	32	RW	0000_0000h
1_4D60h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD4_W0)	32	RW	0000_0000h
1_4D64h	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD4_W1)	32	RW	0000_0000h
1_4D68h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD5_W0)	32	RW	0000_0000h
1_4D6Ch	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD5_W1)	32	RW	0000_0000h
1_4D70h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD6_W0)	32	RW	0000_0000h
1_4D74h	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD6_W1)	32	RW	0000_0000h
1_4D78h	MRC Region Descriptor Word 0 (MRC0_DOM13_RGD7_W0)	32	RW	0000_0000h
1_4D7Ch	MRC Region Descriptor Word 1 (MRC0_DOM13_RGD7_W1)	32	RW	0000_0000h
1_4DC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM13_RGD_NSE)	32	RW	0000_0000h
1_4E40h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD0_W0)	32	RW	0000_0000h
1_4E44h	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD0_W1)	32	RW	0000_0000h
1_4E48h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD1_W0)	32	RW	0000_0000h
1_4E4Ch	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD1_W1)	32	RW	0000_0000h
1_4E50h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD2_W0)	32	RW	0000_0000h
1_4E54h	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD2_W1)	32	RW	0000_0000h
1_4E58h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD3_W0)	32	RW	0000_0000h
1_4E5Ch	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD3_W1)	32	RW	0000_0000h
1_4E60h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD4_W0)	32	RW	0000_0000h
1_4E64h	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD4_W1)	32	RW	0000_0000h
1_4E68h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD5_W0)	32	RW	0000_0000h
1_4E6Ch	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD5_W1)	32	RW	0000_0000h
1_4E70h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD6_W0)	32	RW	0000_0000h
1_4E74h	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD6_W1)	32	RW	0000_0000h
1_4E78h	MRC Region Descriptor Word 0 (MRC0_DOM14_RGD7_W0)	32	RW	0000_0000h
1_4E7Ch	MRC Region Descriptor Word 1 (MRC0_DOM14_RGD7_W1)	32	RW	0000_0000h
1_4EC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM14_RGD_NSE)	32	RW	0000_0000h
1_4F40h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD0_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_4F44h	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD0_W1)	32	RW	0000_0000h
1_4F48h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD1_W0)	32	RW	0000_0000h
1_4F4Ch	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD1_W1)	32	RW	0000_0000h
1_4F50h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD2_W0)	32	RW	0000_0000h
1_4F54h	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD2_W1)	32	RW	0000_0000h
1_4F58h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD3_W0)	32	RW	0000_0000h
1_4F5Ch	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD3_W1)	32	RW	0000_0000h
1_4F60h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD4_W0)	32	RW	0000_0000h
1_4F64h	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD4_W1)	32	RW	0000_0000h
1_4F68h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD5_W0)	32	RW	0000_0000h
1_4F6Ch	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD5_W1)	32	RW	0000_0000h
1_4F70h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD6_W0)	32	RW	0000_0000h
1_4F74h	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD6_W1)	32	RW	0000_0000h
1_4F78h	MRC Region Descriptor Word 0 (MRC0_DOM15_RGD7_W0)	32	RW	0000_0000h
1_4F7Ch	MRC Region Descriptor Word 1 (MRC0_DOM15_RGD7_W1)	32	RW	0000_0000h
1_4FC0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM15_RGD_NSE)	32	RW	0000_0000h
1_5000h	MRC Global Configuration Register (MRC1_GLBCFG)	32	R	0000_0008h
1_5010h	MRC NonSecure Enable Region Indirect (MRC1_NSE_RGN_INDIRECT)	32	RW	0000_0000h
1_5014h	MRC NonSecure Enable Region Set (MRC1_NSE_RGN_SET)	32	RW	0000_0000h
1_5018h	MRC NonSecure Enable Region Clear (MRC1_NSE_RGN_CLR)	32	RW	0000_0000h
1_501Ch	MRC NonSecure Enable Region Clear All (MRC1_NSE_RGN_CLR_ALL)	32	RW	0000_0000h
1_5020h	MRC Global Access Control (MRC1_GLBAC0)	32	RW	0000_0000h
1_5024h	MRC Global Access Control (MRC1_GLBAC1)	32	RW	0000_0000h
1_5028h	MRC Global Access Control (MRC1_GLBAC2)	32	RW	0000_0000h
1_502Ch	MRC Global Access Control (MRC1_GLBAC3)	32	RW	0000_0000h
1_5030h	MRC Global Access Control (MRC1_GLBAC4)	32	RW	0000_0000h
1_5034h	MRC Global Access Control (MRC1_GLBAC5)	32	RW	0000_0000h
1_5038h	MRC Global Access Control (MRC1_GLBAC6)	32	RW	0000_0000h
1_503Ch	MRC Global Access Control (MRC1_GLBAC7)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5040h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD0_W0)	32	RW	0000_0000h
1_5044h	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD0_W1)	32	RW	0000_0000h
1_5048h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD1_W0)	32	RW	0000_0000h
1_504Ch	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD1_W1)	32	RW	0000_0000h
1_5050h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD2_W0)	32	RW	0000_0000h
1_5054h	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD2_W1)	32	RW	0000_0000h
1_5058h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD3_W0)	32	RW	0000_0000h
1_505Ch	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD3_W1)	32	RW	0000_0000h
1_5060h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD4_W0)	32	RW	0000_0000h
1_5064h	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD4_W1)	32	RW	0000_0000h
1_5068h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD5_W0)	32	RW	0000_0000h
1_506Ch	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD5_W1)	32	RW	0000_0000h
1_5070h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD6_W0)	32	RW	0000_0000h
1_5074h	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD6_W1)	32	RW	0000_0000h
1_5078h	MRC Region Descriptor Word 0 (MRC1_DOM0_RGD7_W0)	32	RW	0000_0000h
1_507Ch	MRC Region Descriptor Word 1 (MRC1_DOM0_RGD7_W1)	32	RW	0000_0000h
1_50C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM0_RGD_NSE)	32	RW	0000_0000h
1_5140h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD0_W0)	32	RW	0000_0000h
1_5144h	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD0_W1)	32	RW	0000_0000h
1_5148h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD1_W0)	32	RW	0000_0000h
1_514Ch	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD1_W1)	32	RW	0000_0000h
1_5150h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD2_W0)	32	RW	0000_0000h
1_5154h	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD2_W1)	32	RW	0000_0000h
1_5158h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD3_W0)	32	RW	0000_0000h
1_515Ch	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD3_W1)	32	RW	0000_0000h
1_5160h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD4_W0)	32	RW	0000_0000h
1_5164h	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD4_W1)	32	RW	0000_0000h
1_5168h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD5_W0)	32	RW	0000_0000h
1_516Ch	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD5_W1)	32	RW	0000_0000h
1_5170h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD6_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5174h	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD6_W1)	32	RW	0000_0000h
1_5178h	MRC Region Descriptor Word 0 (MRC1_DOM1_RGD7_W0)	32	RW	0000_0000h
1_517Ch	MRC Region Descriptor Word 1 (MRC1_DOM1_RGD7_W1)	32	RW	0000_0000h
1_51C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM1_RGD_NSE)	32	RW	0000_0000h
1_5240h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD0_W0)	32	RW	0000_0000h
1_5244h	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD0_W1)	32	RW	0000_0000h
1_5248h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD1_W0)	32	RW	0000_0000h
1_524Ch	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD1_W1)	32	RW	0000_0000h
1_5250h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD2_W0)	32	RW	0000_0000h
1_5254h	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD2_W1)	32	RW	0000_0000h
1_5258h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD3_W0)	32	RW	0000_0000h
1_525Ch	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD3_W1)	32	RW	0000_0000h
1_5260h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD4_W0)	32	RW	0000_0000h
1_5264h	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD4_W1)	32	RW	0000_0000h
1_5268h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD5_W0)	32	RW	0000_0000h
1_526Ch	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD5_W1)	32	RW	0000_0000h
1_5270h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD6_W0)	32	RW	0000_0000h
1_5274h	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD6_W1)	32	RW	0000_0000h
1_5278h	MRC Region Descriptor Word 0 (MRC1_DOM2_RGD7_W0)	32	RW	0000_0000h
1_527Ch	MRC Region Descriptor Word 1 (MRC1_DOM2_RGD7_W1)	32	RW	0000_0000h
1_52C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM2_RGD_NSE)	32	RW	0000_0000h
1_5340h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD0_W0)	32	RW	0000_0000h
1_5344h	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD0_W1)	32	RW	0000_0000h
1_5348h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD1_W0)	32	RW	0000_0000h
1_534Ch	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD1_W1)	32	RW	0000_0000h
1_5350h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD2_W0)	32	RW	0000_0000h
1_5354h	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD2_W1)	32	RW	0000_0000h
1_5358h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD3_W0)	32	RW	0000_0000h
1_535Ch	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD3_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5360h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD4_W0)	32	RW	0000_0000h
1_5364h	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD4_W1)	32	RW	0000_0000h
1_5368h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD5_W0)	32	RW	0000_0000h
1_536Ch	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD5_W1)	32	RW	0000_0000h
1_5370h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD6_W0)	32	RW	0000_0000h
1_5374h	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD6_W1)	32	RW	0000_0000h
1_5378h	MRC Region Descriptor Word 0 (MRC1_DOM3_RGD7_W0)	32	RW	0000_0000h
1_537Ch	MRC Region Descriptor Word 1 (MRC1_DOM3_RGD7_W1)	32	RW	0000_0000h
1_53C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM3_RGD_NSE)	32	RW	0000_0000h
1_5440h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD0_W0)	32	RW	0000_0000h
1_5444h	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD0_W1)	32	RW	0000_0000h
1_5448h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD1_W0)	32	RW	0000_0000h
1_544Ch	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD1_W1)	32	RW	0000_0000h
1_5450h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD2_W0)	32	RW	0000_0000h
1_5454h	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD2_W1)	32	RW	0000_0000h
1_5458h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD3_W0)	32	RW	0000_0000h
1_545Ch	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD3_W1)	32	RW	0000_0000h
1_5460h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD4_W0)	32	RW	0000_0000h
1_5464h	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD4_W1)	32	RW	0000_0000h
1_5468h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD5_W0)	32	RW	0000_0000h
1_546Ch	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD5_W1)	32	RW	0000_0000h
1_5470h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD6_W0)	32	RW	0000_0000h
1_5474h	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD6_W1)	32	RW	0000_0000h
1_5478h	MRC Region Descriptor Word 0 (MRC1_DOM4_RGD7_W0)	32	RW	0000_0000h
1_547Ch	MRC Region Descriptor Word 1 (MRC1_DOM4_RGD7_W1)	32	RW	0000_0000h
1_54C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM4_RGD_NSE)	32	RW	0000_0000h
1_5540h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD0_W0)	32	RW	0000_0000h
1_5544h	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD0_W1)	32	RW	0000_0000h
1_5548h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD1_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_554Ch	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD1_W1)	32	RW	0000_0000h
1_5550h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD2_W0)	32	RW	0000_0000h
1_5554h	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD2_W1)	32	RW	0000_0000h
1_5558h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD3_W0)	32	RW	0000_0000h
1_555Ch	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD3_W1)	32	RW	0000_0000h
1_5560h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD4_W0)	32	RW	0000_0000h
1_5564h	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD4_W1)	32	RW	0000_0000h
1_5568h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD5_W0)	32	RW	0000_0000h
1_556Ch	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD5_W1)	32	RW	0000_0000h
1_5570h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD6_W0)	32	RW	0000_0000h
1_5574h	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD6_W1)	32	RW	0000_0000h
1_5578h	MRC Region Descriptor Word 0 (MRC1_DOM5_RGD7_W0)	32	RW	0000_0000h
1_557Ch	MRC Region Descriptor Word 1 (MRC1_DOM5_RGD7_W1)	32	RW	0000_0000h
1_55C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM5_RGD_NSE)	32	RW	0000_0000h
1_5640h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD0_W0)	32	RW	0000_0000h
1_5644h	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD0_W1)	32	RW	0000_0000h
1_5648h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD1_W0)	32	RW	0000_0000h
1_564Ch	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD1_W1)	32	RW	0000_0000h
1_5650h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD2_W0)	32	RW	0000_0000h
1_5654h	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD2_W1)	32	RW	0000_0000h
1_5658h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD3_W0)	32	RW	0000_0000h
1_565Ch	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD3_W1)	32	RW	0000_0000h
1_5660h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD4_W0)	32	RW	0000_0000h
1_5664h	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD4_W1)	32	RW	0000_0000h
1_5668h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD5_W0)	32	RW	0000_0000h
1_566Ch	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD5_W1)	32	RW	0000_0000h
1_5670h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD6_W0)	32	RW	0000_0000h
1_5674h	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD6_W1)	32	RW	0000_0000h
1_5678h	MRC Region Descriptor Word 0 (MRC1_DOM6_RGD7_W0)	32	RW	0000_0000h
1_567Ch	MRC Region Descriptor Word 1 (MRC1_DOM6_RGD7_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_56C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM6_RGD_NSE)	32	RW	0000_0000h
1_5740h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD0_W0)	32	RW	0000_0000h
1_5744h	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD0_W1)	32	RW	0000_0000h
1_5748h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD1_W0)	32	RW	0000_0000h
1_574Ch	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD1_W1)	32	RW	0000_0000h
1_5750h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD2_W0)	32	RW	0000_0000h
1_5754h	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD2_W1)	32	RW	0000_0000h
1_5758h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD3_W0)	32	RW	0000_0000h
1_575Ch	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD3_W1)	32	RW	0000_0000h
1_5760h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD4_W0)	32	RW	0000_0000h
1_5764h	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD4_W1)	32	RW	0000_0000h
1_5768h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD5_W0)	32	RW	0000_0000h
1_576Ch	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD5_W1)	32	RW	0000_0000h
1_5770h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD6_W0)	32	RW	0000_0000h
1_5774h	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD6_W1)	32	RW	0000_0000h
1_5778h	MRC Region Descriptor Word 0 (MRC1_DOM7_RGD7_W0)	32	RW	0000_0000h
1_577Ch	MRC Region Descriptor Word 1 (MRC1_DOM7_RGD7_W1)	32	RW	0000_0000h
1_57C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM7_RGD_NSE)	32	RW	0000_0000h
1_5840h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD0_W0)	32	RW	0000_0000h
1_5844h	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD0_W1)	32	RW	0000_0000h
1_5848h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD1_W0)	32	RW	0000_0000h
1_584Ch	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD1_W1)	32	RW	0000_0000h
1_5850h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD2_W0)	32	RW	0000_0000h
1_5854h	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD2_W1)	32	RW	0000_0000h
1_5858h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD3_W0)	32	RW	0000_0000h
1_585Ch	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD3_W1)	32	RW	0000_0000h
1_5860h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD4_W0)	32	RW	0000_0000h
1_5864h	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD4_W1)	32	RW	0000_0000h
1_5868h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD5_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_586Ch	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD5_W1)	32	RW	0000_0000h
1_5870h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD6_W0)	32	RW	0000_0000h
1_5874h	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD6_W1)	32	RW	0000_0000h
1_5878h	MRC Region Descriptor Word 0 (MRC1_DOM8_RGD7_W0)	32	RW	0000_0000h
1_587Ch	MRC Region Descriptor Word 1 (MRC1_DOM8_RGD7_W1)	32	RW	0000_0000h
1_58C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM8_RGD_NSE)	32	RW	0000_0000h
1_5940h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD0_W0)	32	RW	0000_0000h
1_5944h	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD0_W1)	32	RW	0000_0000h
1_5948h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD1_W0)	32	RW	0000_0000h
1_594Ch	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD1_W1)	32	RW	0000_0000h
1_5950h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD2_W0)	32	RW	0000_0000h
1_5954h	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD2_W1)	32	RW	0000_0000h
1_5958h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD3_W0)	32	RW	0000_0000h
1_595Ch	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD3_W1)	32	RW	0000_0000h
1_5960h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD4_W0)	32	RW	0000_0000h
1_5964h	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD4_W1)	32	RW	0000_0000h
1_5968h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD5_W0)	32	RW	0000_0000h
1_596Ch	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD5_W1)	32	RW	0000_0000h
1_5970h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD6_W0)	32	RW	0000_0000h
1_5974h	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD6_W1)	32	RW	0000_0000h
1_5978h	MRC Region Descriptor Word 0 (MRC1_DOM9_RGD7_W0)	32	RW	0000_0000h
1_597Ch	MRC Region Descriptor Word 1 (MRC1_DOM9_RGD7_W1)	32	RW	0000_0000h
1_59C0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM9_RGD_NSE)	32	RW	0000_0000h
1_5A40h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD0_W0)	32	RW	0000_0000h
1_5A44h	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD0_W1)	32	RW	0000_0000h
1_5A48h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD1_W0)	32	RW	0000_0000h
1_5A4Ch	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD1_W1)	32	RW	0000_0000h
1_5A50h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD2_W0)	32	RW	0000_0000h
1_5A54h	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD2_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5A58h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD3_W0)	32	RW	0000_0000h
1_5A5Ch	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD3_W1)	32	RW	0000_0000h
1_5A60h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD4_W0)	32	RW	0000_0000h
1_5A64h	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD4_W1)	32	RW	0000_0000h
1_5A68h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD5_W0)	32	RW	0000_0000h
1_5A6Ch	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD5_W1)	32	RW	0000_0000h
1_5A70h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD6_W0)	32	RW	0000_0000h
1_5A74h	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD6_W1)	32	RW	0000_0000h
1_5A78h	MRC Region Descriptor Word 0 (MRC1_DOM10_RGD7_W0)	32	RW	0000_0000h
1_5A7Ch	MRC Region Descriptor Word 1 (MRC1_DOM10_RGD7_W1)	32	RW	0000_0000h
1_5AC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM10_RGD_NSE)	32	RW	0000_0000h
1_5B40h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD0_W0)	32	RW	0000_0000h
1_5B44h	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD0_W1)	32	RW	0000_0000h
1_5B48h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD1_W0)	32	RW	0000_0000h
1_5B4Ch	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD1_W1)	32	RW	0000_0000h
1_5B50h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD2_W0)	32	RW	0000_0000h
1_5B54h	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD2_W1)	32	RW	0000_0000h
1_5B58h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD3_W0)	32	RW	0000_0000h
1_5B5Ch	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD3_W1)	32	RW	0000_0000h
1_5B60h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD4_W0)	32	RW	0000_0000h
1_5B64h	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD4_W1)	32	RW	0000_0000h
1_5B68h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD5_W0)	32	RW	0000_0000h
1_5B6Ch	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD5_W1)	32	RW	0000_0000h
1_5B70h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD6_W0)	32	RW	0000_0000h
1_5B74h	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD6_W1)	32	RW	0000_0000h
1_5B78h	MRC Region Descriptor Word 0 (MRC1_DOM11_RGD7_W0)	32	RW	0000_0000h
1_5B7Ch	MRC Region Descriptor Word 1 (MRC1_DOM11_RGD7_W1)	32	RW	0000_0000h
1_5BC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM11_RGD_NSE)	32	RW	0000_0000h
1_5C40h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD0_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5C44h	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD0_W1)	32	RW	0000_0000h
1_5C48h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD1_W0)	32	RW	0000_0000h
1_5C4Ch	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD1_W1)	32	RW	0000_0000h
1_5C50h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD2_W0)	32	RW	0000_0000h
1_5C54h	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD2_W1)	32	RW	0000_0000h
1_5C58h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD3_W0)	32	RW	0000_0000h
1_5C5Ch	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD3_W1)	32	RW	0000_0000h
1_5C60h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD4_W0)	32	RW	0000_0000h
1_5C64h	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD4_W1)	32	RW	0000_0000h
1_5C68h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD5_W0)	32	RW	0000_0000h
1_5C6Ch	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD5_W1)	32	RW	0000_0000h
1_5C70h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD6_W0)	32	RW	0000_0000h
1_5C74h	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD6_W1)	32	RW	0000_0000h
1_5C78h	MRC Region Descriptor Word 0 (MRC1_DOM12_RGD7_W0)	32	RW	0000_0000h
1_5C7Ch	MRC Region Descriptor Word 1 (MRC1_DOM12_RGD7_W1)	32	RW	0000_0000h
1_5CC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM12_RGD_NSE)	32	RW	0000_0000h
1_5D40h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD0_W0)	32	RW	0000_0000h
1_5D44h	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD0_W1)	32	RW	0000_0000h
1_5D48h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD1_W0)	32	RW	0000_0000h
1_5D4Ch	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD1_W1)	32	RW	0000_0000h
1_5D50h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD2_W0)	32	RW	0000_0000h
1_5D54h	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD2_W1)	32	RW	0000_0000h
1_5D58h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD3_W0)	32	RW	0000_0000h
1_5D5Ch	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD3_W1)	32	RW	0000_0000h
1_5D60h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD4_W0)	32	RW	0000_0000h
1_5D64h	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD4_W1)	32	RW	0000_0000h
1_5D68h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD5_W0)	32	RW	0000_0000h
1_5D6Ch	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD5_W1)	32	RW	0000_0000h
1_5D70h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD6_W0)	32	RW	0000_0000h
1_5D74h	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD6_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5D78h	MRC Region Descriptor Word 0 (MRC1_DOM13_RGD7_W0)	32	RW	0000_0000h
1_5D7Ch	MRC Region Descriptor Word 1 (MRC1_DOM13_RGD7_W1)	32	RW	0000_0000h
1_5DC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM13_RGD_NSE)	32	RW	0000_0000h
1_5E40h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD0_W0)	32	RW	0000_0000h
1_5E44h	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD0_W1)	32	RW	0000_0000h
1_5E48h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD1_W0)	32	RW	0000_0000h
1_5E4Ch	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD1_W1)	32	RW	0000_0000h
1_5E50h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD2_W0)	32	RW	0000_0000h
1_5E54h	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD2_W1)	32	RW	0000_0000h
1_5E58h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD3_W0)	32	RW	0000_0000h
1_5E5Ch	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD3_W1)	32	RW	0000_0000h
1_5E60h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD4_W0)	32	RW	0000_0000h
1_5E64h	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD4_W1)	32	RW	0000_0000h
1_5E68h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD5_W0)	32	RW	0000_0000h
1_5E6Ch	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD5_W1)	32	RW	0000_0000h
1_5E70h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD6_W0)	32	RW	0000_0000h
1_5E74h	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD6_W1)	32	RW	0000_0000h
1_5E78h	MRC Region Descriptor Word 0 (MRC1_DOM14_RGD7_W0)	32	RW	0000_0000h
1_5E7Ch	MRC Region Descriptor Word 1 (MRC1_DOM14_RGD7_W1)	32	RW	0000_0000h
1_5EC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM14_RGD_NSE)	32	RW	0000_0000h
1_5F40h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD0_W0)	32	RW	0000_0000h
1_5F44h	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD0_W1)	32	RW	0000_0000h
1_5F48h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD1_W0)	32	RW	0000_0000h
1_5F4Ch	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD1_W1)	32	RW	0000_0000h
1_5F50h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD2_W0)	32	RW	0000_0000h
1_5F54h	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD2_W1)	32	RW	0000_0000h
1_5F58h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD3_W0)	32	RW	0000_0000h
1_5F5Ch	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD3_W1)	32	RW	0000_0000h
1_5F60h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD4_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_5F64h	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD4_W1)	32	RW	0000_0000h
1_5F68h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD5_W0)	32	RW	0000_0000h
1_5F6Ch	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD5_W1)	32	RW	0000_0000h
1_5F70h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD6_W0)	32	RW	0000_0000h
1_5F74h	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD6_W1)	32	RW	0000_0000h
1_5F78h	MRC Region Descriptor Word 0 (MRC1_DOM15_RGD7_W0)	32	RW	0000_0000h
1_5F7Ch	MRC Region Descriptor Word 1 (MRC1_DOM15_RGD7_W1)	32	RW	0000_0000h
1_5FC0h	MRC Region Descriptor NonSecure Enable (MRC1_DOM15_RGD_NSE)	32	RW	0000_0000h
1_6000h	MRC Global Configuration Register (MRC2_GLBCFG)	32	R	0000_0010h
1_6010h	MRC NonSecure Enable Region Indirect (MRC2_NSE_RGN_INDIRECT)	32	RW	0000_0000h
1_6014h	MRC NonSecure Enable Region Set (MRC2_NSE_RGN_SET)	32	RW	0000_0000h
1_6018h	MRC NonSecure Enable Region Clear (MRC2_NSE_RGN_CLR)	32	RW	0000_0000h
1_601Ch	MRC NonSecure Enable Region Clear All (MRC2_NSE_RGN_CLR_ALL)	32	RW	0000_0000h
1_6020h	MRC Global Access Control (MRC2_GLBAC0)	32	RW	0000_0000h
1_6024h	MRC Global Access Control (MRC2_GLBAC1)	32	RW	0000_0000h
1_6028h	MRC Global Access Control (MRC2_GLBAC2)	32	RW	0000_0000h
1_602Ch	MRC Global Access Control (MRC2_GLBAC3)	32	RW	0000_0000h
1_6030h	MRC Global Access Control (MRC2_GLBAC4)	32	RW	0000_0000h
1_6034h	MRC Global Access Control (MRC2_GLBAC5)	32	RW	0000_0000h
1_6038h	MRC Global Access Control (MRC2_GLBAC6)	32	RW	0000_0000h
1_603Ch	MRC Global Access Control (MRC2_GLBAC7)	32	RW	0000_0000h
1_6040h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD0_W0)	32	RW	0000_0000h
1_6044h	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD0_W1)	32	RW	0000_0000h
1_6048h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD1_W0)	32	RW	0000_0000h
1_604Ch	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD1_W1)	32	RW	0000_0000h
1_6050h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD2_W0)	32	RW	0000_0000h
1_6054h	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD2_W1)	32	RW	0000_0000h
1_6058h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD3_W0)	32	RW	0000_0000h
1_605Ch	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD3_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6060h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD4_W0)	32	RW	0000_0000h
1_6064h	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD4_W1)	32	RW	0000_0000h
1_6068h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD5_W0)	32	RW	0000_0000h
1_606Ch	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD5_W1)	32	RW	0000_0000h
1_6070h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD6_W0)	32	RW	0000_0000h
1_6074h	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD6_W1)	32	RW	0000_0000h
1_6078h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD7_W0)	32	RW	0000_0000h
1_607Ch	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD7_W1)	32	RW	0000_0000h
1_6080h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD8_W0)	32	RW	0000_0000h
1_6084h	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD8_W1)	32	RW	0000_0000h
1_6088h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD9_W0)	32	RW	0000_0000h
1_608Ch	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD9_W1)	32	RW	0000_0000h
1_6090h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD10_W0)	32	RW	0000_0000h
1_6094h	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD10_W1)	32	RW	0000_0000h
1_6098h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD11_W0)	32	RW	0000_0000h
1_609Ch	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD11_W1)	32	RW	0000_0000h
1_60A0h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD12_W0)	32	RW	0000_0000h
1_60A4h	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD12_W1)	32	RW	0000_0000h
1_60A8h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD13_W0)	32	RW	0000_0000h
1_60ACh	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD13_W1)	32	RW	0000_0000h
1_60B0h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD14_W0)	32	RW	0000_0000h
1_60B4h	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD14_W1)	32	RW	0000_0000h
1_60B8h	MRC Region Descriptor Word 0 (MRC2_DOM0_RGD15_W0)	32	RW	0000_0000h
1_60BCh	MRC Region Descriptor Word 1 (MRC2_DOM0_RGD15_W1)	32	RW	0000_0000h
1_60C0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM0_RGD_NSE)	32	RW	0000_0000h
1_6140h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD0_W0)	32	RW	0000_0000h
1_6144h	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD0_W1)	32	RW	0000_0000h
1_6148h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD1_W0)	32	RW	0000_0000h
1_614Ch	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD1_W1)	32	RW	0000_0000h
1_6150h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD2_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6154h	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD2_W1)	32	RW	0000_0000h
1_6158h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD3_W0)	32	RW	0000_0000h
1_615Ch	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD3_W1)	32	RW	0000_0000h
1_6160h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD4_W0)	32	RW	0000_0000h
1_6164h	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD4_W1)	32	RW	0000_0000h
1_6168h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD5_W0)	32	RW	0000_0000h
1_616Ch	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD5_W1)	32	RW	0000_0000h
1_6170h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD6_W0)	32	RW	0000_0000h
1_6174h	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD6_W1)	32	RW	0000_0000h
1_6178h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD7_W0)	32	RW	0000_0000h
1_617Ch	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD7_W1)	32	RW	0000_0000h
1_6180h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD8_W0)	32	RW	0000_0000h
1_6184h	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD8_W1)	32	RW	0000_0000h
1_6188h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD9_W0)	32	RW	0000_0000h
1_618Ch	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD9_W1)	32	RW	0000_0000h
1_6190h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD10_W0)	32	RW	0000_0000h
1_6194h	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD10_W1)	32	RW	0000_0000h
1_6198h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD11_W0)	32	RW	0000_0000h
1_619Ch	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD11_W1)	32	RW	0000_0000h
1_61A0h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD12_W0)	32	RW	0000_0000h
1_61A4h	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD12_W1)	32	RW	0000_0000h
1_61A8h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD13_W0)	32	RW	0000_0000h
1_61ACh	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD13_W1)	32	RW	0000_0000h
1_61B0h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD14_W0)	32	RW	0000_0000h
1_61B4h	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD14_W1)	32	RW	0000_0000h
1_61B8h	MRC Region Descriptor Word 0 (MRC2_DOM1_RGD15_W0)	32	RW	0000_0000h
1_61BCh	MRC Region Descriptor Word 1 (MRC2_DOM1_RGD15_W1)	32	RW	0000_0000h
1_61C0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM1_RGD_NSE)	32	RW	0000_0000h
1_6240h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD0_W0)	32	RW	0000_0000h
1_6244h	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD0_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6248h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD1_W0)	32	RW	0000_0000h
1_624Ch	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD1_W1)	32	RW	0000_0000h
1_6250h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD2_W0)	32	RW	0000_0000h
1_6254h	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD2_W1)	32	RW	0000_0000h
1_6258h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD3_W0)	32	RW	0000_0000h
1_625Ch	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD3_W1)	32	RW	0000_0000h
1_6260h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD4_W0)	32	RW	0000_0000h
1_6264h	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD4_W1)	32	RW	0000_0000h
1_6268h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD5_W0)	32	RW	0000_0000h
1_626Ch	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD5_W1)	32	RW	0000_0000h
1_6270h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD6_W0)	32	RW	0000_0000h
1_6274h	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD6_W1)	32	RW	0000_0000h
1_6278h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD7_W0)	32	RW	0000_0000h
1_627Ch	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD7_W1)	32	RW	0000_0000h
1_6280h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD8_W0)	32	RW	0000_0000h
1_6284h	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD8_W1)	32	RW	0000_0000h
1_6288h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD9_W0)	32	RW	0000_0000h
1_628Ch	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD9_W1)	32	RW	0000_0000h
1_6290h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD10_W0)	32	RW	0000_0000h
1_6294h	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD10_W1)	32	RW	0000_0000h
1_6298h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD11_W0)	32	RW	0000_0000h
1_629Ch	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD11_W1)	32	RW	0000_0000h
1_62A0h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD12_W0)	32	RW	0000_0000h
1_62A4h	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD12_W1)	32	RW	0000_0000h
1_62A8h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD13_W0)	32	RW	0000_0000h
1_62ACh	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD13_W1)	32	RW	0000_0000h
1_62B0h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD14_W0)	32	RW	0000_0000h
1_62B4h	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD14_W1)	32	RW	0000_0000h
1_62B8h	MRC Region Descriptor Word 0 (MRC2_DOM2_RGD15_W0)	32	RW	0000_0000h
1_62BCh	MRC Region Descriptor Word 1 (MRC2_DOM2_RGD15_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_62C0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM2_RGD_NSE)	32	RW	0000_0000h
1_6340h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD0_W0)	32	RW	0000_0000h
1_6344h	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD0_W1)	32	RW	0000_0000h
1_6348h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD1_W0)	32	RW	0000_0000h
1_634Ch	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD1_W1)	32	RW	0000_0000h
1_6350h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD2_W0)	32	RW	0000_0000h
1_6354h	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD2_W1)	32	RW	0000_0000h
1_6358h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD3_W0)	32	RW	0000_0000h
1_635Ch	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD3_W1)	32	RW	0000_0000h
1_6360h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD4_W0)	32	RW	0000_0000h
1_6364h	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD4_W1)	32	RW	0000_0000h
1_6368h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD5_W0)	32	RW	0000_0000h
1_636Ch	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD5_W1)	32	RW	0000_0000h
1_6370h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD6_W0)	32	RW	0000_0000h
1_6374h	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD6_W1)	32	RW	0000_0000h
1_6378h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD7_W0)	32	RW	0000_0000h
1_637Ch	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD7_W1)	32	RW	0000_0000h
1_6380h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD8_W0)	32	RW	0000_0000h
1_6384h	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD8_W1)	32	RW	0000_0000h
1_6388h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD9_W0)	32	RW	0000_0000h
1_638Ch	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD9_W1)	32	RW	0000_0000h
1_6390h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD10_W0)	32	RW	0000_0000h
1_6394h	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD10_W1)	32	RW	0000_0000h
1_6398h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD11_W0)	32	RW	0000_0000h
1_639Ch	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD11_W1)	32	RW	0000_0000h
1_63A0h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD12_W0)	32	RW	0000_0000h
1_63A4h	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD12_W1)	32	RW	0000_0000h
1_63A8h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD13_W0)	32	RW	0000_0000h
1_63ACh	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD13_W1)	32	RW	0000_0000h
1_63B0h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD14_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_63B4h	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD14_W1)	32	RW	0000_0000h
1_63B8h	MRC Region Descriptor Word 0 (MRC2_DOM3_RGD15_W0)	32	RW	0000_0000h
1_63BCh	MRC Region Descriptor Word 1 (MRC2_DOM3_RGD15_W1)	32	RW	0000_0000h
1_63C0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM3_RGD_NSE)	32	RW	0000_0000h
1_6440h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD0_W0)	32	RW	0000_0000h
1_6444h	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD0_W1)	32	RW	0000_0000h
1_6448h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD1_W0)	32	RW	0000_0000h
1_644Ch	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD1_W1)	32	RW	0000_0000h
1_6450h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD2_W0)	32	RW	0000_0000h
1_6454h	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD2_W1)	32	RW	0000_0000h
1_6458h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD3_W0)	32	RW	0000_0000h
1_645Ch	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD3_W1)	32	RW	0000_0000h
1_6460h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD4_W0)	32	RW	0000_0000h
1_6464h	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD4_W1)	32	RW	0000_0000h
1_6468h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD5_W0)	32	RW	0000_0000h
1_646Ch	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD5_W1)	32	RW	0000_0000h
1_6470h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD6_W0)	32	RW	0000_0000h
1_6474h	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD6_W1)	32	RW	0000_0000h
1_6478h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD7_W0)	32	RW	0000_0000h
1_647Ch	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD7_W1)	32	RW	0000_0000h
1_6480h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD8_W0)	32	RW	0000_0000h
1_6484h	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD8_W1)	32	RW	0000_0000h
1_6488h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD9_W0)	32	RW	0000_0000h
1_648Ch	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD9_W1)	32	RW	0000_0000h
1_6490h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD10_W0)	32	RW	0000_0000h
1_6494h	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD10_W1)	32	RW	0000_0000h
1_6498h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD11_W0)	32	RW	0000_0000h
1_649Ch	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD11_W1)	32	RW	0000_0000h
1_64A0h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD12_W0)	32	RW	0000_0000h
1_64A4h	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD12_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_64A8h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD13_W0)	32	RW	0000_0000h
1_64ACh	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD13_W1)	32	RW	0000_0000h
1_64B0h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD14_W0)	32	RW	0000_0000h
1_64B4h	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD14_W1)	32	RW	0000_0000h
1_64B8h	MRC Region Descriptor Word 0 (MRC2_DOM4_RGD15_W0)	32	RW	0000_0000h
1_64BCh	MRC Region Descriptor Word 1 (MRC2_DOM4_RGD15_W1)	32	RW	0000_0000h
1_64C0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM4_RGD_NSE)	32	RW	0000_0000h
1_6540h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD0_W0)	32	RW	0000_0000h
1_6544h	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD0_W1)	32	RW	0000_0000h
1_6548h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD1_W0)	32	RW	0000_0000h
1_654Ch	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD1_W1)	32	RW	0000_0000h
1_6550h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD2_W0)	32	RW	0000_0000h
1_6554h	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD2_W1)	32	RW	0000_0000h
1_6558h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD3_W0)	32	RW	0000_0000h
1_655Ch	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD3_W1)	32	RW	0000_0000h
1_6560h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD4_W0)	32	RW	0000_0000h
1_6564h	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD4_W1)	32	RW	0000_0000h
1_6568h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD5_W0)	32	RW	0000_0000h
1_656Ch	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD5_W1)	32	RW	0000_0000h
1_6570h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD6_W0)	32	RW	0000_0000h
1_6574h	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD6_W1)	32	RW	0000_0000h
1_6578h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD7_W0)	32	RW	0000_0000h
1_657Ch	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD7_W1)	32	RW	0000_0000h
1_6580h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD8_W0)	32	RW	0000_0000h
1_6584h	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD8_W1)	32	RW	0000_0000h
1_6588h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD9_W0)	32	RW	0000_0000h
1_658Ch	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD9_W1)	32	RW	0000_0000h
1_6590h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD10_W0)	32	RW	0000_0000h
1_6594h	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD10_W1)	32	RW	0000_0000h
1_6598h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD11_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_659Ch	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD11_W1)	32	RW	0000_0000h
1_65A0h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD12_W0)	32	RW	0000_0000h
1_65A4h	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD12_W1)	32	RW	0000_0000h
1_65A8h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD13_W0)	32	RW	0000_0000h
1_65ACh	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD13_W1)	32	RW	0000_0000h
1_65B0h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD14_W0)	32	RW	0000_0000h
1_65B4h	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD14_W1)	32	RW	0000_0000h
1_65B8h	MRC Region Descriptor Word 0 (MRC2_DOM5_RGD15_W0)	32	RW	0000_0000h
1_65BCh	MRC Region Descriptor Word 1 (MRC2_DOM5_RGD15_W1)	32	RW	0000_0000h
1_65C0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM5_RGD_NSE)	32	RW	0000_0000h
1_6640h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD0_W0)	32	RW	0000_0000h
1_6644h	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD0_W1)	32	RW	0000_0000h
1_6648h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD1_W0)	32	RW	0000_0000h
1_664Ch	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD1_W1)	32	RW	0000_0000h
1_6650h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD2_W0)	32	RW	0000_0000h
1_6654h	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD2_W1)	32	RW	0000_0000h
1_6658h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD3_W0)	32	RW	0000_0000h
1_665Ch	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD3_W1)	32	RW	0000_0000h
1_6660h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD4_W0)	32	RW	0000_0000h
1_6664h	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD4_W1)	32	RW	0000_0000h
1_6668h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD5_W0)	32	RW	0000_0000h
1_666Ch	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD5_W1)	32	RW	0000_0000h
1_6670h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD6_W0)	32	RW	0000_0000h
1_6674h	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD6_W1)	32	RW	0000_0000h
1_6678h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD7_W0)	32	RW	0000_0000h
1_667Ch	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD7_W1)	32	RW	0000_0000h
1_6680h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD8_W0)	32	RW	0000_0000h
1_6684h	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD8_W1)	32	RW	0000_0000h
1_6688h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD9_W0)	32	RW	0000_0000h
1_668Ch	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD9_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6690h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD10_W0)	32	RW	0000_0000h
1_6694h	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD10_W1)	32	RW	0000_0000h
1_6698h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD11_W0)	32	RW	0000_0000h
1_669Ch	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD11_W1)	32	RW	0000_0000h
1_66A0h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD12_W0)	32	RW	0000_0000h
1_66A4h	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD12_W1)	32	RW	0000_0000h
1_66A8h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD13_W0)	32	RW	0000_0000h
1_66ACh	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD13_W1)	32	RW	0000_0000h
1_66B0h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD14_W0)	32	RW	0000_0000h
1_66B4h	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD14_W1)	32	RW	0000_0000h
1_66B8h	MRC Region Descriptor Word 0 (MRC2_DOM6_RGD15_W0)	32	RW	0000_0000h
1_66BCh	MRC Region Descriptor Word 1 (MRC2_DOM6_RGD15_W1)	32	RW	0000_0000h
1_66C0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM6_RGD_NSE)	32	RW	0000_0000h
1_6740h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD0_W0)	32	RW	0000_0000h
1_6744h	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD0_W1)	32	RW	0000_0000h
1_6748h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD1_W0)	32	RW	0000_0000h
1_674Ch	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD1_W1)	32	RW	0000_0000h
1_6750h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD2_W0)	32	RW	0000_0000h
1_6754h	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD2_W1)	32	RW	0000_0000h
1_6758h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD3_W0)	32	RW	0000_0000h
1_675Ch	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD3_W1)	32	RW	0000_0000h
1_6760h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD4_W0)	32	RW	0000_0000h
1_6764h	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD4_W1)	32	RW	0000_0000h
1_6768h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD5_W0)	32	RW	0000_0000h
1_676Ch	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD5_W1)	32	RW	0000_0000h
1_6770h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD6_W0)	32	RW	0000_0000h
1_6774h	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD6_W1)	32	RW	0000_0000h
1_6778h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD7_W0)	32	RW	0000_0000h
1_677Ch	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD7_W1)	32	RW	0000_0000h
1_6780h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD8_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6784h	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD8_W1)	32	RW	0000_0000h
1_6788h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD9_W0)	32	RW	0000_0000h
1_678Ch	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD9_W1)	32	RW	0000_0000h
1_6790h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD10_W0)	32	RW	0000_0000h
1_6794h	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD10_W1)	32	RW	0000_0000h
1_6798h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD11_W0)	32	RW	0000_0000h
1_679Ch	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD11_W1)	32	RW	0000_0000h
1_67A0h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD12_W0)	32	RW	0000_0000h
1_67A4h	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD12_W1)	32	RW	0000_0000h
1_67A8h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD13_W0)	32	RW	0000_0000h
1_67ACh	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD13_W1)	32	RW	0000_0000h
1_67B0h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD14_W0)	32	RW	0000_0000h
1_67B4h	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD14_W1)	32	RW	0000_0000h
1_67B8h	MRC Region Descriptor Word 0 (MRC2_DOM7_RGD15_W0)	32	RW	0000_0000h
1_67BCh	MRC Region Descriptor Word 1 (MRC2_DOM7_RGD15_W1)	32	RW	0000_0000h
1_67C0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM7_RGD_NSE)	32	RW	0000_0000h
1_6840h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD0_W0)	32	RW	0000_0000h
1_6844h	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD0_W1)	32	RW	0000_0000h
1_6848h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD1_W0)	32	RW	0000_0000h
1_684Ch	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD1_W1)	32	RW	0000_0000h
1_6850h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD2_W0)	32	RW	0000_0000h
1_6854h	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD2_W1)	32	RW	0000_0000h
1_6858h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD3_W0)	32	RW	0000_0000h
1_685Ch	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD3_W1)	32	RW	0000_0000h
1_6860h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD4_W0)	32	RW	0000_0000h
1_6864h	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD4_W1)	32	RW	0000_0000h
1_6868h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD5_W0)	32	RW	0000_0000h
1_686Ch	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD5_W1)	32	RW	0000_0000h
1_6870h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD6_W0)	32	RW	0000_0000h
1_6874h	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD6_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6878h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD7_W0)	32	RW	0000_0000h
1_687Ch	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD7_W1)	32	RW	0000_0000h
1_6880h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD8_W0)	32	RW	0000_0000h
1_6884h	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD8_W1)	32	RW	0000_0000h
1_6888h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD9_W0)	32	RW	0000_0000h
1_688Ch	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD9_W1)	32	RW	0000_0000h
1_6890h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD10_W0)	32	RW	0000_0000h
1_6894h	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD10_W1)	32	RW	0000_0000h
1_6898h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD11_W0)	32	RW	0000_0000h
1_689Ch	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD11_W1)	32	RW	0000_0000h
1_68A0h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD12_W0)	32	RW	0000_0000h
1_68A4h	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD12_W1)	32	RW	0000_0000h
1_68A8h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD13_W0)	32	RW	0000_0000h
1_68ACh	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD13_W1)	32	RW	0000_0000h
1_68B0h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD14_W0)	32	RW	0000_0000h
1_68B4h	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD14_W1)	32	RW	0000_0000h
1_68B8h	MRC Region Descriptor Word 0 (MRC2_DOM8_RGD15_W0)	32	RW	0000_0000h
1_68BCh	MRC Region Descriptor Word 1 (MRC2_DOM8_RGD15_W1)	32	RW	0000_0000h
1_68C0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM8_RGD_NSE)	32	RW	0000_0000h
1_6940h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD0_W0)	32	RW	0000_0000h
1_6944h	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD0_W1)	32	RW	0000_0000h
1_6948h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD1_W0)	32	RW	0000_0000h
1_694Ch	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD1_W1)	32	RW	0000_0000h
1_6950h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD2_W0)	32	RW	0000_0000h
1_6954h	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD2_W1)	32	RW	0000_0000h
1_6958h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD3_W0)	32	RW	0000_0000h
1_695Ch	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD3_W1)	32	RW	0000_0000h
1_6960h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD4_W0)	32	RW	0000_0000h
1_6964h	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD4_W1)	32	RW	0000_0000h
1_6968h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD5_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_696Ch	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD5_W1)	32	RW	0000_0000h
1_6970h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD6_W0)	32	RW	0000_0000h
1_6974h	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD6_W1)	32	RW	0000_0000h
1_6978h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD7_W0)	32	RW	0000_0000h
1_697Ch	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD7_W1)	32	RW	0000_0000h
1_6980h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD8_W0)	32	RW	0000_0000h
1_6984h	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD8_W1)	32	RW	0000_0000h
1_6988h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD9_W0)	32	RW	0000_0000h
1_698Ch	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD9_W1)	32	RW	0000_0000h
1_6990h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD10_W0)	32	RW	0000_0000h
1_6994h	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD10_W1)	32	RW	0000_0000h
1_6998h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD11_W0)	32	RW	0000_0000h
1_699Ch	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD11_W1)	32	RW	0000_0000h
1_69A0h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD12_W0)	32	RW	0000_0000h
1_69A4h	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD12_W1)	32	RW	0000_0000h
1_69A8h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD13_W0)	32	RW	0000_0000h
1_69ACh	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD13_W1)	32	RW	0000_0000h
1_69B0h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD14_W0)	32	RW	0000_0000h
1_69B4h	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD14_W1)	32	RW	0000_0000h
1_69B8h	MRC Region Descriptor Word 0 (MRC2_DOM9_RGD15_W0)	32	RW	0000_0000h
1_69BCh	MRC Region Descriptor Word 1 (MRC2_DOM9_RGD15_W1)	32	RW	0000_0000h
1_69C0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM9_RGD_NSE)	32	RW	0000_0000h
1_6A40h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD0_W0)	32	RW	0000_0000h
1_6A44h	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD0_W1)	32	RW	0000_0000h
1_6A48h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD1_W0)	32	RW	0000_0000h
1_6A4Ch	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD1_W1)	32	RW	0000_0000h
1_6A50h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD2_W0)	32	RW	0000_0000h
1_6A54h	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD2_W1)	32	RW	0000_0000h
1_6A58h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD3_W0)	32	RW	0000_0000h
1_6A5Ch	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD3_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6A60h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD4_W0)	32	RW	0000_0000h
1_6A64h	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD4_W1)	32	RW	0000_0000h
1_6A68h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD5_W0)	32	RW	0000_0000h
1_6A6Ch	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD5_W1)	32	RW	0000_0000h
1_6A70h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD6_W0)	32	RW	0000_0000h
1_6A74h	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD6_W1)	32	RW	0000_0000h
1_6A78h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD7_W0)	32	RW	0000_0000h
1_6A7Ch	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD7_W1)	32	RW	0000_0000h
1_6A80h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD8_W0)	32	RW	0000_0000h
1_6A84h	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD8_W1)	32	RW	0000_0000h
1_6A88h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD9_W0)	32	RW	0000_0000h
1_6A8Ch	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD9_W1)	32	RW	0000_0000h
1_6A90h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD10_W0)	32	RW	0000_0000h
1_6A94h	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD10_W1)	32	RW	0000_0000h
1_6A98h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD11_W0)	32	RW	0000_0000h
1_6A9Ch	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD11_W1)	32	RW	0000_0000h
1_6AA0h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD12_W0)	32	RW	0000_0000h
1_6AA4h	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD12_W1)	32	RW	0000_0000h
1_6AA8h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD13_W0)	32	RW	0000_0000h
1_6AACh	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD13_W1)	32	RW	0000_0000h
1_6AB0h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD14_W0)	32	RW	0000_0000h
1_6AB4h	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD14_W1)	32	RW	0000_0000h
1_6AB8h	MRC Region Descriptor Word 0 (MRC2_DOM10_RGD15_W0)	32	RW	0000_0000h
1_6ABCh	MRC Region Descriptor Word 1 (MRC2_DOM10_RGD15_W1)	32	RW	0000_0000h
1_6AC0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM10_RGD_NSE)	32	RW	0000_0000h
1_6B40h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD0_W0)	32	RW	0000_0000h
1_6B44h	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD0_W1)	32	RW	0000_0000h
1_6B48h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD1_W0)	32	RW	0000_0000h
1_6B4Ch	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD1_W1)	32	RW	0000_0000h
1_6B50h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD2_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6B54h	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD2_W1)	32	RW	0000_0000h
1_6B58h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD3_W0)	32	RW	0000_0000h
1_6B5Ch	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD3_W1)	32	RW	0000_0000h
1_6B60h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD4_W0)	32	RW	0000_0000h
1_6B64h	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD4_W1)	32	RW	0000_0000h
1_6B68h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD5_W0)	32	RW	0000_0000h
1_6B6Ch	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD5_W1)	32	RW	0000_0000h
1_6B70h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD6_W0)	32	RW	0000_0000h
1_6B74h	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD6_W1)	32	RW	0000_0000h
1_6B78h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD7_W0)	32	RW	0000_0000h
1_6B7Ch	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD7_W1)	32	RW	0000_0000h
1_6B80h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD8_W0)	32	RW	0000_0000h
1_6B84h	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD8_W1)	32	RW	0000_0000h
1_6B88h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD9_W0)	32	RW	0000_0000h
1_6B8Ch	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD9_W1)	32	RW	0000_0000h
1_6B90h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD10_W0)	32	RW	0000_0000h
1_6B94h	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD10_W1)	32	RW	0000_0000h
1_6B98h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD11_W0)	32	RW	0000_0000h
1_6B9Ch	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD11_W1)	32	RW	0000_0000h
1_6BA0h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD12_W0)	32	RW	0000_0000h
1_6BA4h	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD12_W1)	32	RW	0000_0000h
1_6BA8h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD13_W0)	32	RW	0000_0000h
1_6BACh	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD13_W1)	32	RW	0000_0000h
1_6BB0h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD14_W0)	32	RW	0000_0000h
1_6BB4h	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD14_W1)	32	RW	0000_0000h
1_6BB8h	MRC Region Descriptor Word 0 (MRC2_DOM11_RGD15_W0)	32	RW	0000_0000h
1_6BBCh	MRC Region Descriptor Word 1 (MRC2_DOM11_RGD15_W1)	32	RW	0000_0000h
1_6BC0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM11_RGD_NSE)	32	RW	0000_0000h
1_6C40h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD0_W0)	32	RW	0000_0000h
1_6C44h	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD0_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6C48h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD1_W0)	32	RW	0000_0000h
1_6C4Ch	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD1_W1)	32	RW	0000_0000h
1_6C50h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD2_W0)	32	RW	0000_0000h
1_6C54h	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD2_W1)	32	RW	0000_0000h
1_6C58h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD3_W0)	32	RW	0000_0000h
1_6C5Ch	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD3_W1)	32	RW	0000_0000h
1_6C60h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD4_W0)	32	RW	0000_0000h
1_6C64h	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD4_W1)	32	RW	0000_0000h
1_6C68h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD5_W0)	32	RW	0000_0000h
1_6C6Ch	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD5_W1)	32	RW	0000_0000h
1_6C70h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD6_W0)	32	RW	0000_0000h
1_6C74h	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD6_W1)	32	RW	0000_0000h
1_6C78h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD7_W0)	32	RW	0000_0000h
1_6C7Ch	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD7_W1)	32	RW	0000_0000h
1_6C80h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD8_W0)	32	RW	0000_0000h
1_6C84h	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD8_W1)	32	RW	0000_0000h
1_6C88h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD9_W0)	32	RW	0000_0000h
1_6C8Ch	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD9_W1)	32	RW	0000_0000h
1_6C90h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD10_W0)	32	RW	0000_0000h
1_6C94h	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD10_W1)	32	RW	0000_0000h
1_6C98h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD11_W0)	32	RW	0000_0000h
1_6C9Ch	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD11_W1)	32	RW	0000_0000h
1_6CA0h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD12_W0)	32	RW	0000_0000h
1_6CA4h	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD12_W1)	32	RW	0000_0000h
1_6CA8h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD13_W0)	32	RW	0000_0000h
1_6CACh	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD13_W1)	32	RW	0000_0000h
1_6CB0h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD14_W0)	32	RW	0000_0000h
1_6CB4h	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD14_W1)	32	RW	0000_0000h
1_6CB8h	MRC Region Descriptor Word 0 (MRC2_DOM12_RGD15_W0)	32	RW	0000_0000h
1_6CBCh	MRC Region Descriptor Word 1 (MRC2_DOM12_RGD15_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6CC0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM12_RGD_NSE)	32	RW	0000_0000h
1_6D40h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD0_W0)	32	RW	0000_0000h
1_6D44h	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD0_W1)	32	RW	0000_0000h
1_6D48h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD1_W0)	32	RW	0000_0000h
1_6D4Ch	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD1_W1)	32	RW	0000_0000h
1_6D50h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD2_W0)	32	RW	0000_0000h
1_6D54h	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD2_W1)	32	RW	0000_0000h
1_6D58h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD3_W0)	32	RW	0000_0000h
1_6D5Ch	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD3_W1)	32	RW	0000_0000h
1_6D60h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD4_W0)	32	RW	0000_0000h
1_6D64h	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD4_W1)	32	RW	0000_0000h
1_6D68h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD5_W0)	32	RW	0000_0000h
1_6D6Ch	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD5_W1)	32	RW	0000_0000h
1_6D70h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD6_W0)	32	RW	0000_0000h
1_6D74h	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD6_W1)	32	RW	0000_0000h
1_6D78h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD7_W0)	32	RW	0000_0000h
1_6D7Ch	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD7_W1)	32	RW	0000_0000h
1_6D80h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD8_W0)	32	RW	0000_0000h
1_6D84h	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD8_W1)	32	RW	0000_0000h
1_6D88h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD9_W0)	32	RW	0000_0000h
1_6D8Ch	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD9_W1)	32	RW	0000_0000h
1_6D90h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD10_W0)	32	RW	0000_0000h
1_6D94h	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD10_W1)	32	RW	0000_0000h
1_6D98h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD11_W0)	32	RW	0000_0000h
1_6D9Ch	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD11_W1)	32	RW	0000_0000h
1_6DA0h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD12_W0)	32	RW	0000_0000h
1_6DA4h	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD12_W1)	32	RW	0000_0000h
1_6DA8h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD13_W0)	32	RW	0000_0000h
1_6DACh	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD13_W1)	32	RW	0000_0000h
1_6DB0h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD14_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6DB4h	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD14_W1)	32	RW	0000_0000h
1_6DB8h	MRC Region Descriptor Word 0 (MRC2_DOM13_RGD15_W0)	32	RW	0000_0000h
1_6DBCh	MRC Region Descriptor Word 1 (MRC2_DOM13_RGD15_W1)	32	RW	0000_0000h
1_6DC0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM13_RGD_NSE)	32	RW	0000_0000h
1_6E40h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD0_W0)	32	RW	0000_0000h
1_6E44h	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD0_W1)	32	RW	0000_0000h
1_6E48h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD1_W0)	32	RW	0000_0000h
1_6E4Ch	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD1_W1)	32	RW	0000_0000h
1_6E50h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD2_W0)	32	RW	0000_0000h
1_6E54h	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD2_W1)	32	RW	0000_0000h
1_6E58h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD3_W0)	32	RW	0000_0000h
1_6E5Ch	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD3_W1)	32	RW	0000_0000h
1_6E60h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD4_W0)	32	RW	0000_0000h
1_6E64h	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD4_W1)	32	RW	0000_0000h
1_6E68h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD5_W0)	32	RW	0000_0000h
1_6E6Ch	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD5_W1)	32	RW	0000_0000h
1_6E70h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD6_W0)	32	RW	0000_0000h
1_6E74h	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD6_W1)	32	RW	0000_0000h
1_6E78h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD7_W0)	32	RW	0000_0000h
1_6E7Ch	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD7_W1)	32	RW	0000_0000h
1_6E80h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD8_W0)	32	RW	0000_0000h
1_6E84h	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD8_W1)	32	RW	0000_0000h
1_6E88h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD9_W0)	32	RW	0000_0000h
1_6E8Ch	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD9_W1)	32	RW	0000_0000h
1_6E90h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD10_W0)	32	RW	0000_0000h
1_6E94h	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD10_W1)	32	RW	0000_0000h
1_6E98h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD11_W0)	32	RW	0000_0000h
1_6E9Ch	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD11_W1)	32	RW	0000_0000h
1_6EA0h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD12_W0)	32	RW	0000_0000h
1_6EA4h	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD12_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6EA8h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD13_W0)	32	RW	0000_0000h
1_6EACh	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD13_W1)	32	RW	0000_0000h
1_6EB0h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD14_W0)	32	RW	0000_0000h
1_6EB4h	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD14_W1)	32	RW	0000_0000h
1_6EB8h	MRC Region Descriptor Word 0 (MRC2_DOM14_RGD15_W0)	32	RW	0000_0000h
1_6EBCh	MRC Region Descriptor Word 1 (MRC2_DOM14_RGD15_W1)	32	RW	0000_0000h
1_6EC0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM14_RGD_NSE)	32	RW	0000_0000h
1_6F40h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD0_W0)	32	RW	0000_0000h
1_6F44h	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD0_W1)	32	RW	0000_0000h
1_6F48h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD1_W0)	32	RW	0000_0000h
1_6F4Ch	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD1_W1)	32	RW	0000_0000h
1_6F50h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD2_W0)	32	RW	0000_0000h
1_6F54h	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD2_W1)	32	RW	0000_0000h
1_6F58h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD3_W0)	32	RW	0000_0000h
1_6F5Ch	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD3_W1)	32	RW	0000_0000h
1_6F60h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD4_W0)	32	RW	0000_0000h
1_6F64h	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD4_W1)	32	RW	0000_0000h
1_6F68h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD5_W0)	32	RW	0000_0000h
1_6F6Ch	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD5_W1)	32	RW	0000_0000h
1_6F70h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD6_W0)	32	RW	0000_0000h
1_6F74h	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD6_W1)	32	RW	0000_0000h
1_6F78h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD7_W0)	32	RW	0000_0000h
1_6F7Ch	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD7_W1)	32	RW	0000_0000h
1_6F80h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD8_W0)	32	RW	0000_0000h
1_6F84h	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD8_W1)	32	RW	0000_0000h
1_6F88h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD9_W0)	32	RW	0000_0000h
1_6F8Ch	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD9_W1)	32	RW	0000_0000h
1_6F90h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD10_W0)	32	RW	0000_0000h
1_6F94h	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD10_W1)	32	RW	0000_0000h
1_6F98h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD11_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_6F9Ch	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD11_W1)	32	RW	0000_0000h
1_6FA0h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD12_W0)	32	RW	0000_0000h
1_6FA4h	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD12_W1)	32	RW	0000_0000h
1_6FA8h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD13_W0)	32	RW	0000_0000h
1_6FACh	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD13_W1)	32	RW	0000_0000h
1_6FB0h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD14_W0)	32	RW	0000_0000h
1_6FB4h	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD14_W1)	32	RW	0000_0000h
1_6FB8h	MRC Region Descriptor Word 0 (MRC2_DOM15_RGD15_W0)	32	RW	0000_0000h
1_6FBCh	MRC Region Descriptor Word 1 (MRC2_DOM15_RGD15_W1)	32	RW	0000_0000h
1_6FC0h	MRC Region Descriptor NonSecure Enable (MRC2_DOM15_RGD_NSE)	32	RW	0000_0000h
1_7000h	MRC Global Configuration Register (MRC3_GLBCFG)	32	R	0000_0010h
1_7010h	MRC NonSecure Enable Region Indirect (MRC3_NSE_RGN_INDIRECT)	32	RW	0000_0000h
1_7014h	MRC NonSecure Enable Region Set (MRC3_NSE_RGN_SET)	32	RW	0000_0000h
1_7018h	MRC NonSecure Enable Region Clear (MRC3_NSE_RGN_CLR)	32	RW	0000_0000h
1_701Ch	MRC NonSecure Enable Region Clear All (MRC3_NSE_RGN_CLR_ALL)	32	RW	0000_0000h
1_7020h	MRC Global Access Control (MRC3_GLBAC0)	32	RW	0000_0000h
1_7024h	MRC Global Access Control (MRC3_GLBAC1)	32	RW	0000_0000h
1_7028h	MRC Global Access Control (MRC3_GLBAC2)	32	RW	0000_0000h
1_702Ch	MRC Global Access Control (MRC3_GLBAC3)	32	RW	0000_0000h
1_7030h	MRC Global Access Control (MRC3_GLBAC4)	32	RW	0000_0000h
1_7034h	MRC Global Access Control (MRC3_GLBAC5)	32	RW	0000_0000h
1_7038h	MRC Global Access Control (MRC3_GLBAC6)	32	RW	0000_0000h
1_703Ch	MRC Global Access Control (MRC3_GLBAC7)	32	RW	0000_0000h
1_7040h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD0_W0)	32	RW	0000_0000h
1_7044h	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD0_W1)	32	RW	0000_0000h
1_7048h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD1_W0)	32	RW	0000_0000h
1_704Ch	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD1_W1)	32	RW	0000_0000h
1_7050h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD2_W0)	32	RW	0000_0000h
1_7054h	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD2_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7058h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD3_W0)	32	RW	0000_0000h
1_705Ch	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD3_W1)	32	RW	0000_0000h
1_7060h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD4_W0)	32	RW	0000_0000h
1_7064h	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD4_W1)	32	RW	0000_0000h
1_7068h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD5_W0)	32	RW	0000_0000h
1_706Ch	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD5_W1)	32	RW	0000_0000h
1_7070h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD6_W0)	32	RW	0000_0000h
1_7074h	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD6_W1)	32	RW	0000_0000h
1_7078h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD7_W0)	32	RW	0000_0000h
1_707Ch	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD7_W1)	32	RW	0000_0000h
1_7080h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD8_W0)	32	RW	0000_0000h
1_7084h	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD8_W1)	32	RW	0000_0000h
1_7088h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD9_W0)	32	RW	0000_0000h
1_708Ch	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD9_W1)	32	RW	0000_0000h
1_7090h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD10_W0)	32	RW	0000_0000h
1_7094h	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD10_W1)	32	RW	0000_0000h
1_7098h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD11_W0)	32	RW	0000_0000h
1_709Ch	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD11_W1)	32	RW	0000_0000h
1_70A0h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD12_W0)	32	RW	0000_0000h
1_70A4h	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD12_W1)	32	RW	0000_0000h
1_70A8h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD13_W0)	32	RW	0000_0000h
1_70ACh	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD13_W1)	32	RW	0000_0000h
1_70B0h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD14_W0)	32	RW	0000_0000h
1_70B4h	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD14_W1)	32	RW	0000_0000h
1_70B8h	MRC Region Descriptor Word 0 (MRC3_DOM0_RGD15_W0)	32	RW	0000_0000h
1_70BCh	MRC Region Descriptor Word 1 (MRC3_DOM0_RGD15_W1)	32	RW	0000_0000h
1_70C0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM0_RGD_NSE)	32	RW	0000_0000h
1_7140h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD0_W0)	32	RW	0000_0000h
1_7144h	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD0_W1)	32	RW	0000_0000h
1_7148h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD1_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_714Ch	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD1_W1)	32	RW	0000_0000h
1_7150h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD2_W0)	32	RW	0000_0000h
1_7154h	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD2_W1)	32	RW	0000_0000h
1_7158h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD3_W0)	32	RW	0000_0000h
1_715Ch	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD3_W1)	32	RW	0000_0000h
1_7160h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD4_W0)	32	RW	0000_0000h
1_7164h	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD4_W1)	32	RW	0000_0000h
1_7168h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD5_W0)	32	RW	0000_0000h
1_716Ch	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD5_W1)	32	RW	0000_0000h
1_7170h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD6_W0)	32	RW	0000_0000h
1_7174h	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD6_W1)	32	RW	0000_0000h
1_7178h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD7_W0)	32	RW	0000_0000h
1_717Ch	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD7_W1)	32	RW	0000_0000h
1_7180h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD8_W0)	32	RW	0000_0000h
1_7184h	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD8_W1)	32	RW	0000_0000h
1_7188h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD9_W0)	32	RW	0000_0000h
1_718Ch	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD9_W1)	32	RW	0000_0000h
1_7190h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD10_W0)	32	RW	0000_0000h
1_7194h	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD10_W1)	32	RW	0000_0000h
1_7198h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD11_W0)	32	RW	0000_0000h
1_719Ch	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD11_W1)	32	RW	0000_0000h
1_71A0h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD12_W0)	32	RW	0000_0000h
1_71A4h	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD12_W1)	32	RW	0000_0000h
1_71A8h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD13_W0)	32	RW	0000_0000h
1_71ACh	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD13_W1)	32	RW	0000_0000h
1_71B0h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD14_W0)	32	RW	0000_0000h
1_71B4h	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD14_W1)	32	RW	0000_0000h
1_71B8h	MRC Region Descriptor Word 0 (MRC3_DOM1_RGD15_W0)	32	RW	0000_0000h
1_71BCh	MRC Region Descriptor Word 1 (MRC3_DOM1_RGD15_W1)	32	RW	0000_0000h
1_71C0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM1_RGD_NSE)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7240h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD0_W0)	32	RW	0000_0000h
1_7244h	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD0_W1)	32	RW	0000_0000h
1_7248h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD1_W0)	32	RW	0000_0000h
1_724Ch	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD1_W1)	32	RW	0000_0000h
1_7250h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD2_W0)	32	RW	0000_0000h
1_7254h	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD2_W1)	32	RW	0000_0000h
1_7258h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD3_W0)	32	RW	0000_0000h
1_725Ch	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD3_W1)	32	RW	0000_0000h
1_7260h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD4_W0)	32	RW	0000_0000h
1_7264h	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD4_W1)	32	RW	0000_0000h
1_7268h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD5_W0)	32	RW	0000_0000h
1_726Ch	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD5_W1)	32	RW	0000_0000h
1_7270h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD6_W0)	32	RW	0000_0000h
1_7274h	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD6_W1)	32	RW	0000_0000h
1_7278h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD7_W0)	32	RW	0000_0000h
1_727Ch	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD7_W1)	32	RW	0000_0000h
1_7280h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD8_W0)	32	RW	0000_0000h
1_7284h	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD8_W1)	32	RW	0000_0000h
1_7288h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD9_W0)	32	RW	0000_0000h
1_728Ch	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD9_W1)	32	RW	0000_0000h
1_7290h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD10_W0)	32	RW	0000_0000h
1_7294h	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD10_W1)	32	RW	0000_0000h
1_7298h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD11_W0)	32	RW	0000_0000h
1_729Ch	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD11_W1)	32	RW	0000_0000h
1_72A0h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD12_W0)	32	RW	0000_0000h
1_72A4h	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD12_W1)	32	RW	0000_0000h
1_72A8h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD13_W0)	32	RW	0000_0000h
1_72ACh	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD13_W1)	32	RW	0000_0000h
1_72B0h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD14_W0)	32	RW	0000_0000h
1_72B4h	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD14_W1)	32	RW	0000_0000h
1_72B8h	MRC Region Descriptor Word 0 (MRC3_DOM2_RGD15_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_72BCh	MRC Region Descriptor Word 1 (MRC3_DOM2_RGD15_W1)	32	RW	0000_0000h
1_72C0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM2_RGD_NSE)	32	RW	0000_0000h
1_7340h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD0_W0)	32	RW	0000_0000h
1_7344h	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD0_W1)	32	RW	0000_0000h
1_7348h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD1_W0)	32	RW	0000_0000h
1_734Ch	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD1_W1)	32	RW	0000_0000h
1_7350h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD2_W0)	32	RW	0000_0000h
1_7354h	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD2_W1)	32	RW	0000_0000h
1_7358h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD3_W0)	32	RW	0000_0000h
1_735Ch	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD3_W1)	32	RW	0000_0000h
1_7360h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD4_W0)	32	RW	0000_0000h
1_7364h	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD4_W1)	32	RW	0000_0000h
1_7368h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD5_W0)	32	RW	0000_0000h
1_736Ch	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD5_W1)	32	RW	0000_0000h
1_7370h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD6_W0)	32	RW	0000_0000h
1_7374h	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD6_W1)	32	RW	0000_0000h
1_7378h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD7_W0)	32	RW	0000_0000h
1_737Ch	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD7_W1)	32	RW	0000_0000h
1_7380h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD8_W0)	32	RW	0000_0000h
1_7384h	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD8_W1)	32	RW	0000_0000h
1_7388h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD9_W0)	32	RW	0000_0000h
1_738Ch	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD9_W1)	32	RW	0000_0000h
1_7390h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD10_W0)	32	RW	0000_0000h
1_7394h	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD10_W1)	32	RW	0000_0000h
1_7398h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD11_W0)	32	RW	0000_0000h
1_739Ch	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD11_W1)	32	RW	0000_0000h
1_73A0h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD12_W0)	32	RW	0000_0000h
1_73A4h	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD12_W1)	32	RW	0000_0000h
1_73A8h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD13_W0)	32	RW	0000_0000h
1_73ACh	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD13_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_73B0h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD14_W0)	32	RW	0000_0000h
1_73B4h	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD14_W1)	32	RW	0000_0000h
1_73B8h	MRC Region Descriptor Word 0 (MRC3_DOM3_RGD15_W0)	32	RW	0000_0000h
1_73BCh	MRC Region Descriptor Word 1 (MRC3_DOM3_RGD15_W1)	32	RW	0000_0000h
1_73C0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM3_RGD_NSE)	32	RW	0000_0000h
1_7440h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD0_W0)	32	RW	0000_0000h
1_7444h	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD0_W1)	32	RW	0000_0000h
1_7448h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD1_W0)	32	RW	0000_0000h
1_744Ch	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD1_W1)	32	RW	0000_0000h
1_7450h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD2_W0)	32	RW	0000_0000h
1_7454h	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD2_W1)	32	RW	0000_0000h
1_7458h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD3_W0)	32	RW	0000_0000h
1_745Ch	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD3_W1)	32	RW	0000_0000h
1_7460h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD4_W0)	32	RW	0000_0000h
1_7464h	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD4_W1)	32	RW	0000_0000h
1_7468h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD5_W0)	32	RW	0000_0000h
1_746Ch	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD5_W1)	32	RW	0000_0000h
1_7470h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD6_W0)	32	RW	0000_0000h
1_7474h	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD6_W1)	32	RW	0000_0000h
1_7478h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD7_W0)	32	RW	0000_0000h
1_747Ch	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD7_W1)	32	RW	0000_0000h
1_7480h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD8_W0)	32	RW	0000_0000h
1_7484h	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD8_W1)	32	RW	0000_0000h
1_7488h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD9_W0)	32	RW	0000_0000h
1_748Ch	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD9_W1)	32	RW	0000_0000h
1_7490h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD10_W0)	32	RW	0000_0000h
1_7494h	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD10_W1)	32	RW	0000_0000h
1_7498h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD11_W0)	32	RW	0000_0000h
1_749Ch	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD11_W1)	32	RW	0000_0000h
1_74A0h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD12_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_74A4h	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD12_W1)	32	RW	0000_0000h
1_74A8h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD13_W0)	32	RW	0000_0000h
1_74ACh	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD13_W1)	32	RW	0000_0000h
1_74B0h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD14_W0)	32	RW	0000_0000h
1_74B4h	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD14_W1)	32	RW	0000_0000h
1_74B8h	MRC Region Descriptor Word 0 (MRC3_DOM4_RGD15_W0)	32	RW	0000_0000h
1_74BCh	MRC Region Descriptor Word 1 (MRC3_DOM4_RGD15_W1)	32	RW	0000_0000h
1_74C0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM4_RGD_NSE)	32	RW	0000_0000h
1_7540h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD0_W0)	32	RW	0000_0000h
1_7544h	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD0_W1)	32	RW	0000_0000h
1_7548h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD1_W0)	32	RW	0000_0000h
1_754Ch	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD1_W1)	32	RW	0000_0000h
1_7550h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD2_W0)	32	RW	0000_0000h
1_7554h	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD2_W1)	32	RW	0000_0000h
1_7558h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD3_W0)	32	RW	0000_0000h
1_755Ch	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD3_W1)	32	RW	0000_0000h
1_7560h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD4_W0)	32	RW	0000_0000h
1_7564h	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD4_W1)	32	RW	0000_0000h
1_7568h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD5_W0)	32	RW	0000_0000h
1_756Ch	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD5_W1)	32	RW	0000_0000h
1_7570h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD6_W0)	32	RW	0000_0000h
1_7574h	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD6_W1)	32	RW	0000_0000h
1_7578h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD7_W0)	32	RW	0000_0000h
1_757Ch	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD7_W1)	32	RW	0000_0000h
1_7580h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD8_W0)	32	RW	0000_0000h
1_7584h	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD8_W1)	32	RW	0000_0000h
1_7588h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD9_W0)	32	RW	0000_0000h
1_758Ch	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD9_W1)	32	RW	0000_0000h
1_7590h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD10_W0)	32	RW	0000_0000h
1_7594h	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD10_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7598h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD11_W0)	32	RW	0000_0000h
1_759Ch	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD11_W1)	32	RW	0000_0000h
1_75A0h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD12_W0)	32	RW	0000_0000h
1_75A4h	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD12_W1)	32	RW	0000_0000h
1_75A8h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD13_W0)	32	RW	0000_0000h
1_75ACh	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD13_W1)	32	RW	0000_0000h
1_75B0h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD14_W0)	32	RW	0000_0000h
1_75B4h	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD14_W1)	32	RW	0000_0000h
1_75B8h	MRC Region Descriptor Word 0 (MRC3_DOM5_RGD15_W0)	32	RW	0000_0000h
1_75BCh	MRC Region Descriptor Word 1 (MRC3_DOM5_RGD15_W1)	32	RW	0000_0000h
1_75C0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM5_RGD_NSE)	32	RW	0000_0000h
1_7640h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD0_W0)	32	RW	0000_0000h
1_7644h	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD0_W1)	32	RW	0000_0000h
1_7648h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD1_W0)	32	RW	0000_0000h
1_764Ch	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD1_W1)	32	RW	0000_0000h
1_7650h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD2_W0)	32	RW	0000_0000h
1_7654h	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD2_W1)	32	RW	0000_0000h
1_7658h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD3_W0)	32	RW	0000_0000h
1_765Ch	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD3_W1)	32	RW	0000_0000h
1_7660h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD4_W0)	32	RW	0000_0000h
1_7664h	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD4_W1)	32	RW	0000_0000h
1_7668h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD5_W0)	32	RW	0000_0000h
1_766Ch	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD5_W1)	32	RW	0000_0000h
1_7670h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD6_W0)	32	RW	0000_0000h
1_7674h	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD6_W1)	32	RW	0000_0000h
1_7678h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD7_W0)	32	RW	0000_0000h
1_767Ch	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD7_W1)	32	RW	0000_0000h
1_7680h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD8_W0)	32	RW	0000_0000h
1_7684h	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD8_W1)	32	RW	0000_0000h
1_7688h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD9_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_768Ch	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD9_W1)	32	RW	0000_0000h
1_7690h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD10_W0)	32	RW	0000_0000h
1_7694h	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD10_W1)	32	RW	0000_0000h
1_7698h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD11_W0)	32	RW	0000_0000h
1_769Ch	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD11_W1)	32	RW	0000_0000h
1_76A0h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD12_W0)	32	RW	0000_0000h
1_76A4h	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD12_W1)	32	RW	0000_0000h
1_76A8h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD13_W0)	32	RW	0000_0000h
1_76ACh	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD13_W1)	32	RW	0000_0000h
1_76B0h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD14_W0)	32	RW	0000_0000h
1_76B4h	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD14_W1)	32	RW	0000_0000h
1_76B8h	MRC Region Descriptor Word 0 (MRC3_DOM6_RGD15_W0)	32	RW	0000_0000h
1_76BCh	MRC Region Descriptor Word 1 (MRC3_DOM6_RGD15_W1)	32	RW	0000_0000h
1_76C0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM6_RGD_NSE)	32	RW	0000_0000h
1_7740h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD0_W0)	32	RW	0000_0000h
1_7744h	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD0_W1)	32	RW	0000_0000h
1_7748h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD1_W0)	32	RW	0000_0000h
1_774Ch	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD1_W1)	32	RW	0000_0000h
1_7750h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD2_W0)	32	RW	0000_0000h
1_7754h	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD2_W1)	32	RW	0000_0000h
1_7758h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD3_W0)	32	RW	0000_0000h
1_775Ch	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD3_W1)	32	RW	0000_0000h
1_7760h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD4_W0)	32	RW	0000_0000h
1_7764h	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD4_W1)	32	RW	0000_0000h
1_7768h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD5_W0)	32	RW	0000_0000h
1_776Ch	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD5_W1)	32	RW	0000_0000h
1_7770h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD6_W0)	32	RW	0000_0000h
1_7774h	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD6_W1)	32	RW	0000_0000h
1_7778h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD7_W0)	32	RW	0000_0000h
1_777Ch	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD7_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7780h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD8_W0)	32	RW	0000_0000h
1_7784h	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD8_W1)	32	RW	0000_0000h
1_7788h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD9_W0)	32	RW	0000_0000h
1_778Ch	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD9_W1)	32	RW	0000_0000h
1_7790h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD10_W0)	32	RW	0000_0000h
1_7794h	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD10_W1)	32	RW	0000_0000h
1_7798h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD11_W0)	32	RW	0000_0000h
1_779Ch	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD11_W1)	32	RW	0000_0000h
1_77A0h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD12_W0)	32	RW	0000_0000h
1_77A4h	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD12_W1)	32	RW	0000_0000h
1_77A8h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD13_W0)	32	RW	0000_0000h
1_77ACh	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD13_W1)	32	RW	0000_0000h
1_77B0h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD14_W0)	32	RW	0000_0000h
1_77B4h	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD14_W1)	32	RW	0000_0000h
1_77B8h	MRC Region Descriptor Word 0 (MRC3_DOM7_RGD15_W0)	32	RW	0000_0000h
1_77BCh	MRC Region Descriptor Word 1 (MRC3_DOM7_RGD15_W1)	32	RW	0000_0000h
1_77C0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM7_RGD_NSE)	32	RW	0000_0000h
1_7840h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD0_W0)	32	RW	0000_0000h
1_7844h	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD0_W1)	32	RW	0000_0000h
1_7848h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD1_W0)	32	RW	0000_0000h
1_784Ch	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD1_W1)	32	RW	0000_0000h
1_7850h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD2_W0)	32	RW	0000_0000h
1_7854h	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD2_W1)	32	RW	0000_0000h
1_7858h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD3_W0)	32	RW	0000_0000h
1_785Ch	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD3_W1)	32	RW	0000_0000h
1_7860h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD4_W0)	32	RW	0000_0000h
1_7864h	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD4_W1)	32	RW	0000_0000h
1_7868h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD5_W0)	32	RW	0000_0000h
1_786Ch	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD5_W1)	32	RW	0000_0000h
1_7870h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD6_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7874h	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD6_W1)	32	RW	0000_0000h
1_7878h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD7_W0)	32	RW	0000_0000h
1_787Ch	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD7_W1)	32	RW	0000_0000h
1_7880h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD8_W0)	32	RW	0000_0000h
1_7884h	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD8_W1)	32	RW	0000_0000h
1_7888h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD9_W0)	32	RW	0000_0000h
1_788Ch	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD9_W1)	32	RW	0000_0000h
1_7890h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD10_W0)	32	RW	0000_0000h
1_7894h	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD10_W1)	32	RW	0000_0000h
1_7898h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD11_W0)	32	RW	0000_0000h
1_789Ch	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD11_W1)	32	RW	0000_0000h
1_78A0h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD12_W0)	32	RW	0000_0000h
1_78A4h	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD12_W1)	32	RW	0000_0000h
1_78A8h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD13_W0)	32	RW	0000_0000h
1_78ACh	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD13_W1)	32	RW	0000_0000h
1_78B0h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD14_W0)	32	RW	0000_0000h
1_78B4h	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD14_W1)	32	RW	0000_0000h
1_78B8h	MRC Region Descriptor Word 0 (MRC3_DOM8_RGD15_W0)	32	RW	0000_0000h
1_78BCh	MRC Region Descriptor Word 1 (MRC3_DOM8_RGD15_W1)	32	RW	0000_0000h
1_78C0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM8_RGD_NSE)	32	RW	0000_0000h
1_7940h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD0_W0)	32	RW	0000_0000h
1_7944h	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD0_W1)	32	RW	0000_0000h
1_7948h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD1_W0)	32	RW	0000_0000h
1_794Ch	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD1_W1)	32	RW	0000_0000h
1_7950h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD2_W0)	32	RW	0000_0000h
1_7954h	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD2_W1)	32	RW	0000_0000h
1_7958h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD3_W0)	32	RW	0000_0000h
1_795Ch	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD3_W1)	32	RW	0000_0000h
1_7960h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD4_W0)	32	RW	0000_0000h
1_7964h	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD4_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7968h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD5_W0)	32	RW	0000_0000h
1_796Ch	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD5_W1)	32	RW	0000_0000h
1_7970h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD6_W0)	32	RW	0000_0000h
1_7974h	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD6_W1)	32	RW	0000_0000h
1_7978h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD7_W0)	32	RW	0000_0000h
1_797Ch	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD7_W1)	32	RW	0000_0000h
1_7980h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD8_W0)	32	RW	0000_0000h
1_7984h	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD8_W1)	32	RW	0000_0000h
1_7988h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD9_W0)	32	RW	0000_0000h
1_798Ch	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD9_W1)	32	RW	0000_0000h
1_7990h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD10_W0)	32	RW	0000_0000h
1_7994h	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD10_W1)	32	RW	0000_0000h
1_7998h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD11_W0)	32	RW	0000_0000h
1_799Ch	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD11_W1)	32	RW	0000_0000h
1_79A0h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD12_W0)	32	RW	0000_0000h
1_79A4h	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD12_W1)	32	RW	0000_0000h
1_79A8h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD13_W0)	32	RW	0000_0000h
1_79ACh	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD13_W1)	32	RW	0000_0000h
1_79B0h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD14_W0)	32	RW	0000_0000h
1_79B4h	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD14_W1)	32	RW	0000_0000h
1_79B8h	MRC Region Descriptor Word 0 (MRC3_DOM9_RGD15_W0)	32	RW	0000_0000h
1_79BCh	MRC Region Descriptor Word 1 (MRC3_DOM9_RGD15_W1)	32	RW	0000_0000h
1_79C0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM9_RGD_NSE)	32	RW	0000_0000h
1_7A40h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD0_W0)	32	RW	0000_0000h
1_7A44h	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD0_W1)	32	RW	0000_0000h
1_7A48h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD1_W0)	32	RW	0000_0000h
1_7A4Ch	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD1_W1)	32	RW	0000_0000h
1_7A50h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD2_W0)	32	RW	0000_0000h
1_7A54h	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD2_W1)	32	RW	0000_0000h
1_7A58h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD3_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7A5Ch	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD3_W1)	32	RW	0000_0000h
1_7A60h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD4_W0)	32	RW	0000_0000h
1_7A64h	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD4_W1)	32	RW	0000_0000h
1_7A68h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD5_W0)	32	RW	0000_0000h
1_7A6Ch	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD5_W1)	32	RW	0000_0000h
1_7A70h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD6_W0)	32	RW	0000_0000h
1_7A74h	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD6_W1)	32	RW	0000_0000h
1_7A78h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD7_W0)	32	RW	0000_0000h
1_7A7Ch	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD7_W1)	32	RW	0000_0000h
1_7A80h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD8_W0)	32	RW	0000_0000h
1_7A84h	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD8_W1)	32	RW	0000_0000h
1_7A88h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD9_W0)	32	RW	0000_0000h
1_7A8Ch	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD9_W1)	32	RW	0000_0000h
1_7A90h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD10_W0)	32	RW	0000_0000h
1_7A94h	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD10_W1)	32	RW	0000_0000h
1_7A98h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD11_W0)	32	RW	0000_0000h
1_7A9Ch	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD11_W1)	32	RW	0000_0000h
1_7AA0h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD12_W0)	32	RW	0000_0000h
1_7AA4h	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD12_W1)	32	RW	0000_0000h
1_7AA8h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD13_W0)	32	RW	0000_0000h
1_7AACh	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD13_W1)	32	RW	0000_0000h
1_7AB0h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD14_W0)	32	RW	0000_0000h
1_7AB4h	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD14_W1)	32	RW	0000_0000h
1_7AB8h	MRC Region Descriptor Word 0 (MRC3_DOM10_RGD15_W0)	32	RW	0000_0000h
1_7ABCh	MRC Region Descriptor Word 1 (MRC3_DOM10_RGD15_W1)	32	RW	0000_0000h
1_7AC0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM10_RGD_NSE)	32	RW	0000_0000h
1_7B40h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD0_W0)	32	RW	0000_0000h
1_7B44h	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD0_W1)	32	RW	0000_0000h
1_7B48h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD1_W0)	32	RW	0000_0000h
1_7B4Ch	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD1_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7B50h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD2_W0)	32	RW	0000_0000h
1_7B54h	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD2_W1)	32	RW	0000_0000h
1_7B58h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD3_W0)	32	RW	0000_0000h
1_7B5Ch	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD3_W1)	32	RW	0000_0000h
1_7B60h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD4_W0)	32	RW	0000_0000h
1_7B64h	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD4_W1)	32	RW	0000_0000h
1_7B68h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD5_W0)	32	RW	0000_0000h
1_7B6Ch	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD5_W1)	32	RW	0000_0000h
1_7B70h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD6_W0)	32	RW	0000_0000h
1_7B74h	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD6_W1)	32	RW	0000_0000h
1_7B78h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD7_W0)	32	RW	0000_0000h
1_7B7Ch	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD7_W1)	32	RW	0000_0000h
1_7B80h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD8_W0)	32	RW	0000_0000h
1_7B84h	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD8_W1)	32	RW	0000_0000h
1_7B88h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD9_W0)	32	RW	0000_0000h
1_7B8Ch	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD9_W1)	32	RW	0000_0000h
1_7B90h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD10_W0)	32	RW	0000_0000h
1_7B94h	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD10_W1)	32	RW	0000_0000h
1_7B98h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD11_W0)	32	RW	0000_0000h
1_7B9Ch	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD11_W1)	32	RW	0000_0000h
1_7BA0h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD12_W0)	32	RW	0000_0000h
1_7BA4h	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD12_W1)	32	RW	0000_0000h
1_7BA8h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD13_W0)	32	RW	0000_0000h
1_7BACh	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD13_W1)	32	RW	0000_0000h
1_7BB0h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD14_W0)	32	RW	0000_0000h
1_7BB4h	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD14_W1)	32	RW	0000_0000h
1_7BB8h	MRC Region Descriptor Word 0 (MRC3_DOM11_RGD15_W0)	32	RW	0000_0000h
1_7BBCh	MRC Region Descriptor Word 1 (MRC3_DOM11_RGD15_W1)	32	RW	0000_0000h
1_7BC0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM11_RGD_NSE)	32	RW	0000_0000h
1_7C40h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD0_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7C44h	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD0_W1)	32	RW	0000_0000h
1_7C48h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD1_W0)	32	RW	0000_0000h
1_7C4Ch	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD1_W1)	32	RW	0000_0000h
1_7C50h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD2_W0)	32	RW	0000_0000h
1_7C54h	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD2_W1)	32	RW	0000_0000h
1_7C58h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD3_W0)	32	RW	0000_0000h
1_7C5Ch	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD3_W1)	32	RW	0000_0000h
1_7C60h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD4_W0)	32	RW	0000_0000h
1_7C64h	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD4_W1)	32	RW	0000_0000h
1_7C68h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD5_W0)	32	RW	0000_0000h
1_7C6Ch	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD5_W1)	32	RW	0000_0000h
1_7C70h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD6_W0)	32	RW	0000_0000h
1_7C74h	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD6_W1)	32	RW	0000_0000h
1_7C78h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD7_W0)	32	RW	0000_0000h
1_7C7Ch	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD7_W1)	32	RW	0000_0000h
1_7C80h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD8_W0)	32	RW	0000_0000h
1_7C84h	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD8_W1)	32	RW	0000_0000h
1_7C88h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD9_W0)	32	RW	0000_0000h
1_7C8Ch	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD9_W1)	32	RW	0000_0000h
1_7C90h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD10_W0)	32	RW	0000_0000h
1_7C94h	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD10_W1)	32	RW	0000_0000h
1_7C98h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD11_W0)	32	RW	0000_0000h
1_7C9Ch	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD11_W1)	32	RW	0000_0000h
1_7CA0h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD12_W0)	32	RW	0000_0000h
1_7CA4h	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD12_W1)	32	RW	0000_0000h
1_7CA8h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD13_W0)	32	RW	0000_0000h
1_7CACh	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD13_W1)	32	RW	0000_0000h
1_7CB0h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD14_W0)	32	RW	0000_0000h
1_7CB4h	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD14_W1)	32	RW	0000_0000h
1_7CB8h	MRC Region Descriptor Word 0 (MRC3_DOM12_RGD15_W0)	32	RW	0000_0000h
1_7CBCh	MRC Region Descriptor Word 1 (MRC3_DOM12_RGD15_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7CC0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM12_RGD_NSE)	32	RW	0000_0000h
1_7D40h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD0_W0)	32	RW	0000_0000h
1_7D44h	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD0_W1)	32	RW	0000_0000h
1_7D48h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD1_W0)	32	RW	0000_0000h
1_7D4Ch	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD1_W1)	32	RW	0000_0000h
1_7D50h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD2_W0)	32	RW	0000_0000h
1_7D54h	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD2_W1)	32	RW	0000_0000h
1_7D58h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD3_W0)	32	RW	0000_0000h
1_7D5Ch	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD3_W1)	32	RW	0000_0000h
1_7D60h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD4_W0)	32	RW	0000_0000h
1_7D64h	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD4_W1)	32	RW	0000_0000h
1_7D68h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD5_W0)	32	RW	0000_0000h
1_7D6Ch	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD5_W1)	32	RW	0000_0000h
1_7D70h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD6_W0)	32	RW	0000_0000h
1_7D74h	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD6_W1)	32	RW	0000_0000h
1_7D78h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD7_W0)	32	RW	0000_0000h
1_7D7Ch	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD7_W1)	32	RW	0000_0000h
1_7D80h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD8_W0)	32	RW	0000_0000h
1_7D84h	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD8_W1)	32	RW	0000_0000h
1_7D88h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD9_W0)	32	RW	0000_0000h
1_7D8Ch	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD9_W1)	32	RW	0000_0000h
1_7D90h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD10_W0)	32	RW	0000_0000h
1_7D94h	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD10_W1)	32	RW	0000_0000h
1_7D98h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD11_W0)	32	RW	0000_0000h
1_7D9Ch	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD11_W1)	32	RW	0000_0000h
1_7DA0h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD12_W0)	32	RW	0000_0000h
1_7DA4h	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD12_W1)	32	RW	0000_0000h
1_7DA8h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD13_W0)	32	RW	0000_0000h
1_7DACh	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD13_W1)	32	RW	0000_0000h
1_7DB0h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD14_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7DB4h	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD14_W1)	32	RW	0000_0000h
1_7DB8h	MRC Region Descriptor Word 0 (MRC3_DOM13_RGD15_W0)	32	RW	0000_0000h
1_7DBCh	MRC Region Descriptor Word 1 (MRC3_DOM13_RGD15_W1)	32	RW	0000_0000h
1_7DC0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM13_RGD_NSE)	32	RW	0000_0000h
1_7E40h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD0_W0)	32	RW	0000_0000h
1_7E44h	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD0_W1)	32	RW	0000_0000h
1_7E48h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD1_W0)	32	RW	0000_0000h
1_7E4Ch	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD1_W1)	32	RW	0000_0000h
1_7E50h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD2_W0)	32	RW	0000_0000h
1_7E54h	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD2_W1)	32	RW	0000_0000h
1_7E58h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD3_W0)	32	RW	0000_0000h
1_7E5Ch	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD3_W1)	32	RW	0000_0000h
1_7E60h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD4_W0)	32	RW	0000_0000h
1_7E64h	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD4_W1)	32	RW	0000_0000h
1_7E68h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD5_W0)	32	RW	0000_0000h
1_7E6Ch	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD5_W1)	32	RW	0000_0000h
1_7E70h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD6_W0)	32	RW	0000_0000h
1_7E74h	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD6_W1)	32	RW	0000_0000h
1_7E78h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD7_W0)	32	RW	0000_0000h
1_7E7Ch	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD7_W1)	32	RW	0000_0000h
1_7E80h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD8_W0)	32	RW	0000_0000h
1_7E84h	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD8_W1)	32	RW	0000_0000h
1_7E88h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD9_W0)	32	RW	0000_0000h
1_7E8Ch	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD9_W1)	32	RW	0000_0000h
1_7E90h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD10_W0)	32	RW	0000_0000h
1_7E94h	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD10_W1)	32	RW	0000_0000h
1_7E98h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD11_W0)	32	RW	0000_0000h
1_7E9Ch	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD11_W1)	32	RW	0000_0000h
1_7EA0h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD12_W0)	32	RW	0000_0000h
1_7EA4h	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD12_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7EA8h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD13_W0)	32	RW	0000_0000h
1_7EACh	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD13_W1)	32	RW	0000_0000h
1_7EB0h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD14_W0)	32	RW	0000_0000h
1_7EB4h	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD14_W1)	32	RW	0000_0000h
1_7EB8h	MRC Region Descriptor Word 0 (MRC3_DOM14_RGD15_W0)	32	RW	0000_0000h
1_7EBCh	MRC Region Descriptor Word 1 (MRC3_DOM14_RGD15_W1)	32	RW	0000_0000h
1_7EC0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM14_RGD_NSE)	32	RW	0000_0000h
1_7F40h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD0_W0)	32	RW	0000_0000h
1_7F44h	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD0_W1)	32	RW	0000_0000h
1_7F48h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD1_W0)	32	RW	0000_0000h
1_7F4Ch	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD1_W1)	32	RW	0000_0000h
1_7F50h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD2_W0)	32	RW	0000_0000h
1_7F54h	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD2_W1)	32	RW	0000_0000h
1_7F58h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD3_W0)	32	RW	0000_0000h
1_7F5Ch	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD3_W1)	32	RW	0000_0000h
1_7F60h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD4_W0)	32	RW	0000_0000h
1_7F64h	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD4_W1)	32	RW	0000_0000h
1_7F68h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD5_W0)	32	RW	0000_0000h
1_7F6Ch	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD5_W1)	32	RW	0000_0000h
1_7F70h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD6_W0)	32	RW	0000_0000h
1_7F74h	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD6_W1)	32	RW	0000_0000h
1_7F78h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD7_W0)	32	RW	0000_0000h
1_7F7Ch	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD7_W1)	32	RW	0000_0000h
1_7F80h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD8_W0)	32	RW	0000_0000h
1_7F84h	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD8_W1)	32	RW	0000_0000h
1_7F88h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD9_W0)	32	RW	0000_0000h
1_7F8Ch	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD9_W1)	32	RW	0000_0000h
1_7F90h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD10_W0)	32	RW	0000_0000h
1_7F94h	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD10_W1)	32	RW	0000_0000h
1_7F98h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD11_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_7F9Ch	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD11_W1)	32	RW	0000_0000h
1_7FA0h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD12_W0)	32	RW	0000_0000h
1_7FA4h	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD12_W1)	32	RW	0000_0000h
1_7FA8h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD13_W0)	32	RW	0000_0000h
1_7FACh	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD13_W1)	32	RW	0000_0000h
1_7FB0h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD14_W0)	32	RW	0000_0000h
1_7FB4h	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD14_W1)	32	RW	0000_0000h
1_7FB8h	MRC Region Descriptor Word 0 (MRC3_DOM15_RGD15_W0)	32	RW	0000_0000h
1_7FBCh	MRC Region Descriptor Word 1 (MRC3_DOM15_RGD15_W1)	32	RW	0000_0000h
1_7FC0h	MRC Region Descriptor NonSecure Enable (MRC3_DOM15_RGD_NSE)	32	RW	0000_0000h
1_8000h	MRC Global Configuration Register (MRC4_GLBCFG)	32	R	0000_0010h
1_8010h	MRC NonSecure Enable Region Indirect (MRC4_NSE_RGN_INDIRECT)	32	RW	0000_0000h
1_8014h	MRC NonSecure Enable Region Set (MRC4_NSE_RGN_SET)	32	RW	0000_0000h
1_8018h	MRC NonSecure Enable Region Clear (MRC4_NSE_RGN_CLR)	32	RW	0000_0000h
1_801Ch	MRC NonSecure Enable Region Clear All (MRC4_NSE_RGN_CLR_ALL)	32	RW	0000_0000h
1_8020h	MRC Global Access Control (MRC4_GLBAC0)	32	RW	0000_0000h
1_8024h	MRC Global Access Control (MRC4_GLBAC1)	32	RW	0000_0000h
1_8028h	MRC Global Access Control (MRC4_GLBAC2)	32	RW	0000_0000h
1_802Ch	MRC Global Access Control (MRC4_GLBAC3)	32	RW	0000_0000h
1_8030h	MRC Global Access Control (MRC4_GLBAC4)	32	RW	0000_0000h
1_8034h	MRC Global Access Control (MRC4_GLBAC5)	32	RW	0000_0000h
1_8038h	MRC Global Access Control (MRC4_GLBAC6)	32	RW	0000_0000h
1_803Ch	MRC Global Access Control (MRC4_GLBAC7)	32	RW	0000_0000h
1_8040h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD0_W0)	32	RW	0000_0000h
1_8044h	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD0_W1)	32	RW	0000_0000h
1_8048h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD1_W0)	32	RW	0000_0000h
1_804Ch	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD1_W1)	32	RW	0000_0000h
1_8050h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD2_W0)	32	RW	0000_0000h
1_8054h	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD2_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8058h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD3_W0)	32	RW	0000_0000h
1_805Ch	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD3_W1)	32	RW	0000_0000h
1_8060h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD4_W0)	32	RW	0000_0000h
1_8064h	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD4_W1)	32	RW	0000_0000h
1_8068h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD5_W0)	32	RW	0000_0000h
1_806Ch	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD5_W1)	32	RW	0000_0000h
1_8070h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD6_W0)	32	RW	0000_0000h
1_8074h	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD6_W1)	32	RW	0000_0000h
1_8078h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD7_W0)	32	RW	0000_0000h
1_807Ch	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD7_W1)	32	RW	0000_0000h
1_8080h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD8_W0)	32	RW	0000_0000h
1_8084h	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD8_W1)	32	RW	0000_0000h
1_8088h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD9_W0)	32	RW	0000_0000h
1_808Ch	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD9_W1)	32	RW	0000_0000h
1_8090h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD10_W0)	32	RW	0000_0000h
1_8094h	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD10_W1)	32	RW	0000_0000h
1_8098h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD11_W0)	32	RW	0000_0000h
1_809Ch	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD11_W1)	32	RW	0000_0000h
1_80A0h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD12_W0)	32	RW	0000_0000h
1_80A4h	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD12_W1)	32	RW	0000_0000h
1_80A8h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD13_W0)	32	RW	0000_0000h
1_80ACh	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD13_W1)	32	RW	0000_0000h
1_80B0h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD14_W0)	32	RW	0000_0000h
1_80B4h	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD14_W1)	32	RW	0000_0000h
1_80B8h	MRC Region Descriptor Word 0 (MRC4_DOM0_RGD15_W0)	32	RW	0000_0000h
1_80BCh	MRC Region Descriptor Word 1 (MRC4_DOM0_RGD15_W1)	32	RW	0000_0000h
1_80C0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM0_RGD_NSE)	32	RW	0000_0000h
1_8140h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD0_W0)	32	RW	0000_0000h
1_8144h	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD0_W1)	32	RW	0000_0000h
1_8148h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD1_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_814Ch	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD1_W1)	32	RW	0000_0000h
1_8150h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD2_W0)	32	RW	0000_0000h
1_8154h	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD2_W1)	32	RW	0000_0000h
1_8158h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD3_W0)	32	RW	0000_0000h
1_815Ch	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD3_W1)	32	RW	0000_0000h
1_8160h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD4_W0)	32	RW	0000_0000h
1_8164h	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD4_W1)	32	RW	0000_0000h
1_8168h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD5_W0)	32	RW	0000_0000h
1_816Ch	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD5_W1)	32	RW	0000_0000h
1_8170h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD6_W0)	32	RW	0000_0000h
1_8174h	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD6_W1)	32	RW	0000_0000h
1_8178h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD7_W0)	32	RW	0000_0000h
1_817Ch	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD7_W1)	32	RW	0000_0000h
1_8180h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD8_W0)	32	RW	0000_0000h
1_8184h	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD8_W1)	32	RW	0000_0000h
1_8188h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD9_W0)	32	RW	0000_0000h
1_818Ch	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD9_W1)	32	RW	0000_0000h
1_8190h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD10_W0)	32	RW	0000_0000h
1_8194h	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD10_W1)	32	RW	0000_0000h
1_8198h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD11_W0)	32	RW	0000_0000h
1_819Ch	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD11_W1)	32	RW	0000_0000h
1_81A0h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD12_W0)	32	RW	0000_0000h
1_81A4h	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD12_W1)	32	RW	0000_0000h
1_81A8h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD13_W0)	32	RW	0000_0000h
1_81ACh	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD13_W1)	32	RW	0000_0000h
1_81B0h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD14_W0)	32	RW	0000_0000h
1_81B4h	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD14_W1)	32	RW	0000_0000h
1_81B8h	MRC Region Descriptor Word 0 (MRC4_DOM1_RGD15_W0)	32	RW	0000_0000h
1_81BCh	MRC Region Descriptor Word 1 (MRC4_DOM1_RGD15_W1)	32	RW	0000_0000h
1_81C0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM1_RGD_NSE)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8240h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD0_W0)	32	RW	0000_0000h
1_8244h	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD0_W1)	32	RW	0000_0000h
1_8248h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD1_W0)	32	RW	0000_0000h
1_824Ch	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD1_W1)	32	RW	0000_0000h
1_8250h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD2_W0)	32	RW	0000_0000h
1_8254h	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD2_W1)	32	RW	0000_0000h
1_8258h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD3_W0)	32	RW	0000_0000h
1_825Ch	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD3_W1)	32	RW	0000_0000h
1_8260h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD4_W0)	32	RW	0000_0000h
1_8264h	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD4_W1)	32	RW	0000_0000h
1_8268h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD5_W0)	32	RW	0000_0000h
1_826Ch	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD5_W1)	32	RW	0000_0000h
1_8270h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD6_W0)	32	RW	0000_0000h
1_8274h	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD6_W1)	32	RW	0000_0000h
1_8278h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD7_W0)	32	RW	0000_0000h
1_827Ch	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD7_W1)	32	RW	0000_0000h
1_8280h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD8_W0)	32	RW	0000_0000h
1_8284h	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD8_W1)	32	RW	0000_0000h
1_8288h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD9_W0)	32	RW	0000_0000h
1_828Ch	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD9_W1)	32	RW	0000_0000h
1_8290h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD10_W0)	32	RW	0000_0000h
1_8294h	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD10_W1)	32	RW	0000_0000h
1_8298h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD11_W0)	32	RW	0000_0000h
1_829Ch	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD11_W1)	32	RW	0000_0000h
1_82A0h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD12_W0)	32	RW	0000_0000h
1_82A4h	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD12_W1)	32	RW	0000_0000h
1_82A8h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD13_W0)	32	RW	0000_0000h
1_82ACh	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD13_W1)	32	RW	0000_0000h
1_82B0h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD14_W0)	32	RW	0000_0000h
1_82B4h	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD14_W1)	32	RW	0000_0000h
1_82B8h	MRC Region Descriptor Word 0 (MRC4_DOM2_RGD15_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_82BCh	MRC Region Descriptor Word 1 (MRC4_DOM2_RGD15_W1)	32	RW	0000_0000h
1_82C0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM2_RGD_NSE)	32	RW	0000_0000h
1_8340h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD0_W0)	32	RW	0000_0000h
1_8344h	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD0_W1)	32	RW	0000_0000h
1_8348h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD1_W0)	32	RW	0000_0000h
1_834Ch	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD1_W1)	32	RW	0000_0000h
1_8350h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD2_W0)	32	RW	0000_0000h
1_8354h	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD2_W1)	32	RW	0000_0000h
1_8358h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD3_W0)	32	RW	0000_0000h
1_835Ch	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD3_W1)	32	RW	0000_0000h
1_8360h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD4_W0)	32	RW	0000_0000h
1_8364h	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD4_W1)	32	RW	0000_0000h
1_8368h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD5_W0)	32	RW	0000_0000h
1_836Ch	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD5_W1)	32	RW	0000_0000h
1_8370h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD6_W0)	32	RW	0000_0000h
1_8374h	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD6_W1)	32	RW	0000_0000h
1_8378h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD7_W0)	32	RW	0000_0000h
1_837Ch	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD7_W1)	32	RW	0000_0000h
1_8380h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD8_W0)	32	RW	0000_0000h
1_8384h	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD8_W1)	32	RW	0000_0000h
1_8388h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD9_W0)	32	RW	0000_0000h
1_838Ch	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD9_W1)	32	RW	0000_0000h
1_8390h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD10_W0)	32	RW	0000_0000h
1_8394h	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD10_W1)	32	RW	0000_0000h
1_8398h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD11_W0)	32	RW	0000_0000h
1_839Ch	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD11_W1)	32	RW	0000_0000h
1_83A0h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD12_W0)	32	RW	0000_0000h
1_83A4h	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD12_W1)	32	RW	0000_0000h
1_83A8h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD13_W0)	32	RW	0000_0000h
1_83ACh	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD13_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_83B0h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD14_W0)	32	RW	0000_0000h
1_83B4h	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD14_W1)	32	RW	0000_0000h
1_83B8h	MRC Region Descriptor Word 0 (MRC4_DOM3_RGD15_W0)	32	RW	0000_0000h
1_83BCh	MRC Region Descriptor Word 1 (MRC4_DOM3_RGD15_W1)	32	RW	0000_0000h
1_83C0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM3_RGD_NSE)	32	RW	0000_0000h
1_8440h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD0_W0)	32	RW	0000_0000h
1_8444h	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD0_W1)	32	RW	0000_0000h
1_8448h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD1_W0)	32	RW	0000_0000h
1_844Ch	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD1_W1)	32	RW	0000_0000h
1_8450h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD2_W0)	32	RW	0000_0000h
1_8454h	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD2_W1)	32	RW	0000_0000h
1_8458h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD3_W0)	32	RW	0000_0000h
1_845Ch	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD3_W1)	32	RW	0000_0000h
1_8460h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD4_W0)	32	RW	0000_0000h
1_8464h	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD4_W1)	32	RW	0000_0000h
1_8468h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD5_W0)	32	RW	0000_0000h
1_846Ch	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD5_W1)	32	RW	0000_0000h
1_8470h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD6_W0)	32	RW	0000_0000h
1_8474h	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD6_W1)	32	RW	0000_0000h
1_8478h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD7_W0)	32	RW	0000_0000h
1_847Ch	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD7_W1)	32	RW	0000_0000h
1_8480h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD8_W0)	32	RW	0000_0000h
1_8484h	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD8_W1)	32	RW	0000_0000h
1_8488h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD9_W0)	32	RW	0000_0000h
1_848Ch	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD9_W1)	32	RW	0000_0000h
1_8490h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD10_W0)	32	RW	0000_0000h
1_8494h	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD10_W1)	32	RW	0000_0000h
1_8498h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD11_W0)	32	RW	0000_0000h
1_849Ch	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD11_W1)	32	RW	0000_0000h
1_84A0h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD12_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_84A4h	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD12_W1)	32	RW	0000_0000h
1_84A8h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD13_W0)	32	RW	0000_0000h
1_84ACh	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD13_W1)	32	RW	0000_0000h
1_84B0h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD14_W0)	32	RW	0000_0000h
1_84B4h	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD14_W1)	32	RW	0000_0000h
1_84B8h	MRC Region Descriptor Word 0 (MRC4_DOM4_RGD15_W0)	32	RW	0000_0000h
1_84BCh	MRC Region Descriptor Word 1 (MRC4_DOM4_RGD15_W1)	32	RW	0000_0000h
1_84C0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM4_RGD_NSE)	32	RW	0000_0000h
1_8540h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD0_W0)	32	RW	0000_0000h
1_8544h	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD0_W1)	32	RW	0000_0000h
1_8548h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD1_W0)	32	RW	0000_0000h
1_854Ch	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD1_W1)	32	RW	0000_0000h
1_8550h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD2_W0)	32	RW	0000_0000h
1_8554h	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD2_W1)	32	RW	0000_0000h
1_8558h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD3_W0)	32	RW	0000_0000h
1_855Ch	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD3_W1)	32	RW	0000_0000h
1_8560h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD4_W0)	32	RW	0000_0000h
1_8564h	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD4_W1)	32	RW	0000_0000h
1_8568h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD5_W0)	32	RW	0000_0000h
1_856Ch	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD5_W1)	32	RW	0000_0000h
1_8570h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD6_W0)	32	RW	0000_0000h
1_8574h	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD6_W1)	32	RW	0000_0000h
1_8578h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD7_W0)	32	RW	0000_0000h
1_857Ch	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD7_W1)	32	RW	0000_0000h
1_8580h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD8_W0)	32	RW	0000_0000h
1_8584h	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD8_W1)	32	RW	0000_0000h
1_8588h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD9_W0)	32	RW	0000_0000h
1_858Ch	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD9_W1)	32	RW	0000_0000h
1_8590h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD10_W0)	32	RW	0000_0000h
1_8594h	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD10_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8598h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD11_W0)	32	RW	0000_0000h
1_859Ch	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD11_W1)	32	RW	0000_0000h
1_85A0h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD12_W0)	32	RW	0000_0000h
1_85A4h	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD12_W1)	32	RW	0000_0000h
1_85A8h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD13_W0)	32	RW	0000_0000h
1_85ACh	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD13_W1)	32	RW	0000_0000h
1_85B0h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD14_W0)	32	RW	0000_0000h
1_85B4h	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD14_W1)	32	RW	0000_0000h
1_85B8h	MRC Region Descriptor Word 0 (MRC4_DOM5_RGD15_W0)	32	RW	0000_0000h
1_85BCh	MRC Region Descriptor Word 1 (MRC4_DOM5_RGD15_W1)	32	RW	0000_0000h
1_85C0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM5_RGD_NSE)	32	RW	0000_0000h
1_8640h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD0_W0)	32	RW	0000_0000h
1_8644h	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD0_W1)	32	RW	0000_0000h
1_8648h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD1_W0)	32	RW	0000_0000h
1_864Ch	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD1_W1)	32	RW	0000_0000h
1_8650h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD2_W0)	32	RW	0000_0000h
1_8654h	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD2_W1)	32	RW	0000_0000h
1_8658h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD3_W0)	32	RW	0000_0000h
1_865Ch	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD3_W1)	32	RW	0000_0000h
1_8660h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD4_W0)	32	RW	0000_0000h
1_8664h	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD4_W1)	32	RW	0000_0000h
1_8668h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD5_W0)	32	RW	0000_0000h
1_866Ch	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD5_W1)	32	RW	0000_0000h
1_8670h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD6_W0)	32	RW	0000_0000h
1_8674h	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD6_W1)	32	RW	0000_0000h
1_8678h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD7_W0)	32	RW	0000_0000h
1_867Ch	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD7_W1)	32	RW	0000_0000h
1_8680h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD8_W0)	32	RW	0000_0000h
1_8684h	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD8_W1)	32	RW	0000_0000h
1_8688h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD9_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_868Ch	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD9_W1)	32	RW	0000_0000h
1_8690h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD10_W0)	32	RW	0000_0000h
1_8694h	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD10_W1)	32	RW	0000_0000h
1_8698h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD11_W0)	32	RW	0000_0000h
1_869Ch	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD11_W1)	32	RW	0000_0000h
1_86A0h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD12_W0)	32	RW	0000_0000h
1_86A4h	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD12_W1)	32	RW	0000_0000h
1_86A8h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD13_W0)	32	RW	0000_0000h
1_86ACh	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD13_W1)	32	RW	0000_0000h
1_86B0h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD14_W0)	32	RW	0000_0000h
1_86B4h	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD14_W1)	32	RW	0000_0000h
1_86B8h	MRC Region Descriptor Word 0 (MRC4_DOM6_RGD15_W0)	32	RW	0000_0000h
1_86BCh	MRC Region Descriptor Word 1 (MRC4_DOM6_RGD15_W1)	32	RW	0000_0000h
1_86C0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM6_RGD_NSE)	32	RW	0000_0000h
1_8740h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD0_W0)	32	RW	0000_0000h
1_8744h	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD0_W1)	32	RW	0000_0000h
1_8748h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD1_W0)	32	RW	0000_0000h
1_874Ch	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD1_W1)	32	RW	0000_0000h
1_8750h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD2_W0)	32	RW	0000_0000h
1_8754h	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD2_W1)	32	RW	0000_0000h
1_8758h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD3_W0)	32	RW	0000_0000h
1_875Ch	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD3_W1)	32	RW	0000_0000h
1_8760h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD4_W0)	32	RW	0000_0000h
1_8764h	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD4_W1)	32	RW	0000_0000h
1_8768h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD5_W0)	32	RW	0000_0000h
1_876Ch	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD5_W1)	32	RW	0000_0000h
1_8770h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD6_W0)	32	RW	0000_0000h
1_8774h	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD6_W1)	32	RW	0000_0000h
1_8778h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD7_W0)	32	RW	0000_0000h
1_877Ch	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD7_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8780h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD8_W0)	32	RW	0000_0000h
1_8784h	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD8_W1)	32	RW	0000_0000h
1_8788h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD9_W0)	32	RW	0000_0000h
1_878Ch	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD9_W1)	32	RW	0000_0000h
1_8790h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD10_W0)	32	RW	0000_0000h
1_8794h	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD10_W1)	32	RW	0000_0000h
1_8798h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD11_W0)	32	RW	0000_0000h
1_879Ch	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD11_W1)	32	RW	0000_0000h
1_87A0h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD12_W0)	32	RW	0000_0000h
1_87A4h	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD12_W1)	32	RW	0000_0000h
1_87A8h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD13_W0)	32	RW	0000_0000h
1_87ACh	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD13_W1)	32	RW	0000_0000h
1_87B0h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD14_W0)	32	RW	0000_0000h
1_87B4h	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD14_W1)	32	RW	0000_0000h
1_87B8h	MRC Region Descriptor Word 0 (MRC4_DOM7_RGD15_W0)	32	RW	0000_0000h
1_87BCh	MRC Region Descriptor Word 1 (MRC4_DOM7_RGD15_W1)	32	RW	0000_0000h
1_87C0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM7_RGD_NSE)	32	RW	0000_0000h
1_8840h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD0_W0)	32	RW	0000_0000h
1_8844h	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD0_W1)	32	RW	0000_0000h
1_8848h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD1_W0)	32	RW	0000_0000h
1_884Ch	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD1_W1)	32	RW	0000_0000h
1_8850h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD2_W0)	32	RW	0000_0000h
1_8854h	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD2_W1)	32	RW	0000_0000h
1_8858h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD3_W0)	32	RW	0000_0000h
1_885Ch	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD3_W1)	32	RW	0000_0000h
1_8860h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD4_W0)	32	RW	0000_0000h
1_8864h	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD4_W1)	32	RW	0000_0000h
1_8868h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD5_W0)	32	RW	0000_0000h
1_886Ch	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD5_W1)	32	RW	0000_0000h
1_8870h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD6_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8874h	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD6_W1)	32	RW	0000_0000h
1_8878h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD7_W0)	32	RW	0000_0000h
1_887Ch	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD7_W1)	32	RW	0000_0000h
1_8880h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD8_W0)	32	RW	0000_0000h
1_8884h	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD8_W1)	32	RW	0000_0000h
1_8888h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD9_W0)	32	RW	0000_0000h
1_888Ch	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD9_W1)	32	RW	0000_0000h
1_8890h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD10_W0)	32	RW	0000_0000h
1_8894h	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD10_W1)	32	RW	0000_0000h
1_8898h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD11_W0)	32	RW	0000_0000h
1_889Ch	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD11_W1)	32	RW	0000_0000h
1_88A0h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD12_W0)	32	RW	0000_0000h
1_88A4h	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD12_W1)	32	RW	0000_0000h
1_88A8h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD13_W0)	32	RW	0000_0000h
1_88ACh	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD13_W1)	32	RW	0000_0000h
1_88B0h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD14_W0)	32	RW	0000_0000h
1_88B4h	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD14_W1)	32	RW	0000_0000h
1_88B8h	MRC Region Descriptor Word 0 (MRC4_DOM8_RGD15_W0)	32	RW	0000_0000h
1_88BCh	MRC Region Descriptor Word 1 (MRC4_DOM8_RGD15_W1)	32	RW	0000_0000h
1_88C0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM8_RGD_NSE)	32	RW	0000_0000h
1_8940h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD0_W0)	32	RW	0000_0000h
1_8944h	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD0_W1)	32	RW	0000_0000h
1_8948h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD1_W0)	32	RW	0000_0000h
1_894Ch	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD1_W1)	32	RW	0000_0000h
1_8950h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD2_W0)	32	RW	0000_0000h
1_8954h	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD2_W1)	32	RW	0000_0000h
1_8958h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD3_W0)	32	RW	0000_0000h
1_895Ch	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD3_W1)	32	RW	0000_0000h
1_8960h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD4_W0)	32	RW	0000_0000h
1_8964h	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD4_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8968h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD5_W0)	32	RW	0000_0000h
1_896Ch	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD5_W1)	32	RW	0000_0000h
1_8970h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD6_W0)	32	RW	0000_0000h
1_8974h	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD6_W1)	32	RW	0000_0000h
1_8978h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD7_W0)	32	RW	0000_0000h
1_897Ch	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD7_W1)	32	RW	0000_0000h
1_8980h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD8_W0)	32	RW	0000_0000h
1_8984h	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD8_W1)	32	RW	0000_0000h
1_8988h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD9_W0)	32	RW	0000_0000h
1_898Ch	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD9_W1)	32	RW	0000_0000h
1_8990h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD10_W0)	32	RW	0000_0000h
1_8994h	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD10_W1)	32	RW	0000_0000h
1_8998h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD11_W0)	32	RW	0000_0000h
1_899Ch	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD11_W1)	32	RW	0000_0000h
1_89A0h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD12_W0)	32	RW	0000_0000h
1_89A4h	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD12_W1)	32	RW	0000_0000h
1_89A8h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD13_W0)	32	RW	0000_0000h
1_89ACh	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD13_W1)	32	RW	0000_0000h
1_89B0h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD14_W0)	32	RW	0000_0000h
1_89B4h	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD14_W1)	32	RW	0000_0000h
1_89B8h	MRC Region Descriptor Word 0 (MRC4_DOM9_RGD15_W0)	32	RW	0000_0000h
1_89BCh	MRC Region Descriptor Word 1 (MRC4_DOM9_RGD15_W1)	32	RW	0000_0000h
1_89C0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM9_RGD_NSE)	32	RW	0000_0000h
1_8A40h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD0_W0)	32	RW	0000_0000h
1_8A44h	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD0_W1)	32	RW	0000_0000h
1_8A48h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD1_W0)	32	RW	0000_0000h
1_8A4Ch	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD1_W1)	32	RW	0000_0000h
1_8A50h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD2_W0)	32	RW	0000_0000h
1_8A54h	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD2_W1)	32	RW	0000_0000h
1_8A58h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD3_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8A5Ch	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD3_W1)	32	RW	0000_0000h
1_8A60h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD4_W0)	32	RW	0000_0000h
1_8A64h	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD4_W1)	32	RW	0000_0000h
1_8A68h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD5_W0)	32	RW	0000_0000h
1_8A6Ch	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD5_W1)	32	RW	0000_0000h
1_8A70h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD6_W0)	32	RW	0000_0000h
1_8A74h	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD6_W1)	32	RW	0000_0000h
1_8A78h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD7_W0)	32	RW	0000_0000h
1_8A7Ch	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD7_W1)	32	RW	0000_0000h
1_8A80h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD8_W0)	32	RW	0000_0000h
1_8A84h	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD8_W1)	32	RW	0000_0000h
1_8A88h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD9_W0)	32	RW	0000_0000h
1_8A8Ch	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD9_W1)	32	RW	0000_0000h
1_8A90h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD10_W0)	32	RW	0000_0000h
1_8A94h	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD10_W1)	32	RW	0000_0000h
1_8A98h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD11_W0)	32	RW	0000_0000h
1_8A9Ch	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD11_W1)	32	RW	0000_0000h
1_8AA0h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD12_W0)	32	RW	0000_0000h
1_8AA4h	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD12_W1)	32	RW	0000_0000h
1_8AA8h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD13_W0)	32	RW	0000_0000h
1_8AACh	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD13_W1)	32	RW	0000_0000h
1_8AB0h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD14_W0)	32	RW	0000_0000h
1_8AB4h	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD14_W1)	32	RW	0000_0000h
1_8AB8h	MRC Region Descriptor Word 0 (MRC4_DOM10_RGD15_W0)	32	RW	0000_0000h
1_8ABCh	MRC Region Descriptor Word 1 (MRC4_DOM10_RGD15_W1)	32	RW	0000_0000h
1_8AC0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM10_RGD_NSE)	32	RW	0000_0000h
1_8B40h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD0_W0)	32	RW	0000_0000h
1_8B44h	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD0_W1)	32	RW	0000_0000h
1_8B48h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD1_W0)	32	RW	0000_0000h
1_8B4Ch	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD1_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8B50h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD2_W0)	32	RW	0000_0000h
1_8B54h	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD2_W1)	32	RW	0000_0000h
1_8B58h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD3_W0)	32	RW	0000_0000h
1_8B5Ch	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD3_W1)	32	RW	0000_0000h
1_8B60h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD4_W0)	32	RW	0000_0000h
1_8B64h	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD4_W1)	32	RW	0000_0000h
1_8B68h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD5_W0)	32	RW	0000_0000h
1_8B6Ch	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD5_W1)	32	RW	0000_0000h
1_8B70h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD6_W0)	32	RW	0000_0000h
1_8B74h	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD6_W1)	32	RW	0000_0000h
1_8B78h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD7_W0)	32	RW	0000_0000h
1_8B7Ch	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD7_W1)	32	RW	0000_0000h
1_8B80h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD8_W0)	32	RW	0000_0000h
1_8B84h	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD8_W1)	32	RW	0000_0000h
1_8B88h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD9_W0)	32	RW	0000_0000h
1_8B8Ch	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD9_W1)	32	RW	0000_0000h
1_8B90h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD10_W0)	32	RW	0000_0000h
1_8B94h	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD10_W1)	32	RW	0000_0000h
1_8B98h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD11_W0)	32	RW	0000_0000h
1_8B9Ch	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD11_W1)	32	RW	0000_0000h
1_8BA0h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD12_W0)	32	RW	0000_0000h
1_8BA4h	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD12_W1)	32	RW	0000_0000h
1_8BA8h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD13_W0)	32	RW	0000_0000h
1_8BACh	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD13_W1)	32	RW	0000_0000h
1_8BB0h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD14_W0)	32	RW	0000_0000h
1_8BB4h	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD14_W1)	32	RW	0000_0000h
1_8BB8h	MRC Region Descriptor Word 0 (MRC4_DOM11_RGD15_W0)	32	RW	0000_0000h
1_8BBCh	MRC Region Descriptor Word 1 (MRC4_DOM11_RGD15_W1)	32	RW	0000_0000h
1_8BC0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM11_RGD_NSE)	32	RW	0000_0000h
1_8C40h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD0_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8C44h	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD0_W1)	32	RW	0000_0000h
1_8C48h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD1_W0)	32	RW	0000_0000h
1_8C4Ch	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD1_W1)	32	RW	0000_0000h
1_8C50h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD2_W0)	32	RW	0000_0000h
1_8C54h	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD2_W1)	32	RW	0000_0000h
1_8C58h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD3_W0)	32	RW	0000_0000h
1_8C5Ch	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD3_W1)	32	RW	0000_0000h
1_8C60h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD4_W0)	32	RW	0000_0000h
1_8C64h	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD4_W1)	32	RW	0000_0000h
1_8C68h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD5_W0)	32	RW	0000_0000h
1_8C6Ch	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD5_W1)	32	RW	0000_0000h
1_8C70h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD6_W0)	32	RW	0000_0000h
1_8C74h	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD6_W1)	32	RW	0000_0000h
1_8C78h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD7_W0)	32	RW	0000_0000h
1_8C7Ch	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD7_W1)	32	RW	0000_0000h
1_8C80h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD8_W0)	32	RW	0000_0000h
1_8C84h	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD8_W1)	32	RW	0000_0000h
1_8C88h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD9_W0)	32	RW	0000_0000h
1_8C8Ch	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD9_W1)	32	RW	0000_0000h
1_8C90h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD10_W0)	32	RW	0000_0000h
1_8C94h	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD10_W1)	32	RW	0000_0000h
1_8C98h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD11_W0)	32	RW	0000_0000h
1_8C9Ch	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD11_W1)	32	RW	0000_0000h
1_8CA0h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD12_W0)	32	RW	0000_0000h
1_8CA4h	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD12_W1)	32	RW	0000_0000h
1_8CA8h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD13_W0)	32	RW	0000_0000h
1_8CACh	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD13_W1)	32	RW	0000_0000h
1_8CB0h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD14_W0)	32	RW	0000_0000h
1_8CB4h	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD14_W1)	32	RW	0000_0000h
1_8CB8h	MRC Region Descriptor Word 0 (MRC4_DOM12_RGD15_W0)	32	RW	0000_0000h
1_8CBCh	MRC Region Descriptor Word 1 (MRC4_DOM12_RGD15_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8CC0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM12_RGD_NSE)	32	RW	0000_0000h
1_8D40h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD0_W0)	32	RW	0000_0000h
1_8D44h	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD0_W1)	32	RW	0000_0000h
1_8D48h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD1_W0)	32	RW	0000_0000h
1_8D4Ch	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD1_W1)	32	RW	0000_0000h
1_8D50h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD2_W0)	32	RW	0000_0000h
1_8D54h	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD2_W1)	32	RW	0000_0000h
1_8D58h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD3_W0)	32	RW	0000_0000h
1_8D5Ch	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD3_W1)	32	RW	0000_0000h
1_8D60h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD4_W0)	32	RW	0000_0000h
1_8D64h	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD4_W1)	32	RW	0000_0000h
1_8D68h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD5_W0)	32	RW	0000_0000h
1_8D6Ch	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD5_W1)	32	RW	0000_0000h
1_8D70h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD6_W0)	32	RW	0000_0000h
1_8D74h	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD6_W1)	32	RW	0000_0000h
1_8D78h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD7_W0)	32	RW	0000_0000h
1_8D7Ch	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD7_W1)	32	RW	0000_0000h
1_8D80h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD8_W0)	32	RW	0000_0000h
1_8D84h	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD8_W1)	32	RW	0000_0000h
1_8D88h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD9_W0)	32	RW	0000_0000h
1_8D8Ch	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD9_W1)	32	RW	0000_0000h
1_8D90h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD10_W0)	32	RW	0000_0000h
1_8D94h	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD10_W1)	32	RW	0000_0000h
1_8D98h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD11_W0)	32	RW	0000_0000h
1_8D9Ch	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD11_W1)	32	RW	0000_0000h
1_8DA0h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD12_W0)	32	RW	0000_0000h
1_8DA4h	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD12_W1)	32	RW	0000_0000h
1_8DA8h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD13_W0)	32	RW	0000_0000h
1_8DACh	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD13_W1)	32	RW	0000_0000h
1_8DB0h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD14_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8DB4h	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD14_W1)	32	RW	0000_0000h
1_8DB8h	MRC Region Descriptor Word 0 (MRC4_DOM13_RGD15_W0)	32	RW	0000_0000h
1_8DBCh	MRC Region Descriptor Word 1 (MRC4_DOM13_RGD15_W1)	32	RW	0000_0000h
1_8DC0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM13_RGD_NSE)	32	RW	0000_0000h
1_8E40h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD0_W0)	32	RW	0000_0000h
1_8E44h	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD0_W1)	32	RW	0000_0000h
1_8E48h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD1_W0)	32	RW	0000_0000h
1_8E4Ch	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD1_W1)	32	RW	0000_0000h
1_8E50h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD2_W0)	32	RW	0000_0000h
1_8E54h	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD2_W1)	32	RW	0000_0000h
1_8E58h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD3_W0)	32	RW	0000_0000h
1_8E5Ch	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD3_W1)	32	RW	0000_0000h
1_8E60h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD4_W0)	32	RW	0000_0000h
1_8E64h	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD4_W1)	32	RW	0000_0000h
1_8E68h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD5_W0)	32	RW	0000_0000h
1_8E6Ch	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD5_W1)	32	RW	0000_0000h
1_8E70h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD6_W0)	32	RW	0000_0000h
1_8E74h	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD6_W1)	32	RW	0000_0000h
1_8E78h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD7_W0)	32	RW	0000_0000h
1_8E7Ch	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD7_W1)	32	RW	0000_0000h
1_8E80h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD8_W0)	32	RW	0000_0000h
1_8E84h	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD8_W1)	32	RW	0000_0000h
1_8E88h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD9_W0)	32	RW	0000_0000h
1_8E8Ch	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD9_W1)	32	RW	0000_0000h
1_8E90h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD10_W0)	32	RW	0000_0000h
1_8E94h	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD10_W1)	32	RW	0000_0000h
1_8E98h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD11_W0)	32	RW	0000_0000h
1_8E9Ch	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD11_W1)	32	RW	0000_0000h
1_8EA0h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD12_W0)	32	RW	0000_0000h
1_8EA4h	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD12_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8EA8h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD13_W0)	32	RW	0000_0000h
1_8EACh	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD13_W1)	32	RW	0000_0000h
1_8EB0h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD14_W0)	32	RW	0000_0000h
1_8EB4h	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD14_W1)	32	RW	0000_0000h
1_8EB8h	MRC Region Descriptor Word 0 (MRC4_DOM14_RGD15_W0)	32	RW	0000_0000h
1_8EBCh	MRC Region Descriptor Word 1 (MRC4_DOM14_RGD15_W1)	32	RW	0000_0000h
1_8EC0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM14_RGD_NSE)	32	RW	0000_0000h
1_8F40h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD0_W0)	32	RW	0000_0000h
1_8F44h	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD0_W1)	32	RW	0000_0000h
1_8F48h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD1_W0)	32	RW	0000_0000h
1_8F4Ch	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD1_W1)	32	RW	0000_0000h
1_8F50h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD2_W0)	32	RW	0000_0000h
1_8F54h	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD2_W1)	32	RW	0000_0000h
1_8F58h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD3_W0)	32	RW	0000_0000h
1_8F5Ch	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD3_W1)	32	RW	0000_0000h
1_8F60h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD4_W0)	32	RW	0000_0000h
1_8F64h	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD4_W1)	32	RW	0000_0000h
1_8F68h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD5_W0)	32	RW	0000_0000h
1_8F6Ch	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD5_W1)	32	RW	0000_0000h
1_8F70h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD6_W0)	32	RW	0000_0000h
1_8F74h	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD6_W1)	32	RW	0000_0000h
1_8F78h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD7_W0)	32	RW	0000_0000h
1_8F7Ch	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD7_W1)	32	RW	0000_0000h
1_8F80h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD8_W0)	32	RW	0000_0000h
1_8F84h	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD8_W1)	32	RW	0000_0000h
1_8F88h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD9_W0)	32	RW	0000_0000h
1_8F8Ch	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD9_W1)	32	RW	0000_0000h
1_8F90h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD10_W0)	32	RW	0000_0000h
1_8F94h	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD10_W1)	32	RW	0000_0000h
1_8F98h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD11_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_8F9Ch	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD11_W1)	32	RW	0000_0000h
1_8FA0h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD12_W0)	32	RW	0000_0000h
1_8FA4h	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD12_W1)	32	RW	0000_0000h
1_8FA8h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD13_W0)	32	RW	0000_0000h
1_8FACh	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD13_W1)	32	RW	0000_0000h
1_8FB0h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD14_W0)	32	RW	0000_0000h
1_8FB4h	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD14_W1)	32	RW	0000_0000h
1_8FB8h	MRC Region Descriptor Word 0 (MRC4_DOM15_RGD15_W0)	32	RW	0000_0000h
1_8FBCh	MRC Region Descriptor Word 1 (MRC4_DOM15_RGD15_W1)	32	RW	0000_0000h
1_8FC0h	MRC Region Descriptor NonSecure Enable (MRC4_DOM15_RGD_NSE)	32	RW	0000_0000h
1_9000h	MRC Global Configuration Register (MRC5_GLBCFG)	32	R	0000_0010h
1_9010h	MRC NonSecure Enable Region Indirect (MRC5_NSE_RGN_INDIRECT)	32	RW	0000_0000h
1_9014h	MRC NonSecure Enable Region Set (MRC5_NSE_RGN_SET)	32	RW	0000_0000h
1_9018h	MRC NonSecure Enable Region Clear (MRC5_NSE_RGN_CLR)	32	RW	0000_0000h
1_901Ch	MRC NonSecure Enable Region Clear All (MRC5_NSE_RGN_CLR_ALL)	32	RW	0000_0000h
1_9020h	MRC Global Access Control (MRC5_GLBAC0)	32	RW	0000_0000h
1_9024h	MRC Global Access Control (MRC5_GLBAC1)	32	RW	0000_0000h
1_9028h	MRC Global Access Control (MRC5_GLBAC2)	32	RW	0000_0000h
1_902Ch	MRC Global Access Control (MRC5_GLBAC3)	32	RW	0000_0000h
1_9030h	MRC Global Access Control (MRC5_GLBAC4)	32	RW	0000_0000h
1_9034h	MRC Global Access Control (MRC5_GLBAC5)	32	RW	0000_0000h
1_9038h	MRC Global Access Control (MRC5_GLBAC6)	32	RW	0000_0000h
1_903Ch	MRC Global Access Control (MRC5_GLBAC7)	32	RW	0000_0000h
1_9040h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD0_W0)	32	RW	0000_0000h
1_9044h	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD0_W1)	32	RW	0000_0000h
1_9048h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD1_W0)	32	RW	0000_0000h
1_904Ch	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD1_W1)	32	RW	0000_0000h
1_9050h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD2_W0)	32	RW	0000_0000h
1_9054h	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD2_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9058h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD3_W0)	32	RW	0000_0000h
1_905Ch	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD3_W1)	32	RW	0000_0000h
1_9060h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD4_W0)	32	RW	0000_0000h
1_9064h	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD4_W1)	32	RW	0000_0000h
1_9068h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD5_W0)	32	RW	0000_0000h
1_906Ch	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD5_W1)	32	RW	0000_0000h
1_9070h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD6_W0)	32	RW	0000_0000h
1_9074h	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD6_W1)	32	RW	0000_0000h
1_9078h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD7_W0)	32	RW	0000_0000h
1_907Ch	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD7_W1)	32	RW	0000_0000h
1_9080h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD8_W0)	32	RW	0000_0000h
1_9084h	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD8_W1)	32	RW	0000_0000h
1_9088h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD9_W0)	32	RW	0000_0000h
1_908Ch	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD9_W1)	32	RW	0000_0000h
1_9090h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD10_W0)	32	RW	0000_0000h
1_9094h	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD10_W1)	32	RW	0000_0000h
1_9098h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD11_W0)	32	RW	0000_0000h
1_909Ch	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD11_W1)	32	RW	0000_0000h
1_90A0h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD12_W0)	32	RW	0000_0000h
1_90A4h	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD12_W1)	32	RW	0000_0000h
1_90A8h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD13_W0)	32	RW	0000_0000h
1_90ACh	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD13_W1)	32	RW	0000_0000h
1_90B0h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD14_W0)	32	RW	0000_0000h
1_90B4h	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD14_W1)	32	RW	0000_0000h
1_90B8h	MRC Region Descriptor Word 0 (MRC5_DOM0_RGD15_W0)	32	RW	0000_0000h
1_90BCh	MRC Region Descriptor Word 1 (MRC5_DOM0_RGD15_W1)	32	RW	0000_0000h
1_90C0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM0_RGD_NSE)	32	RW	0000_0000h
1_9140h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD0_W0)	32	RW	0000_0000h
1_9144h	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD0_W1)	32	RW	0000_0000h
1_9148h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD1_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_914Ch	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD1_W1)	32	RW	0000_0000h
1_9150h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD2_W0)	32	RW	0000_0000h
1_9154h	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD2_W1)	32	RW	0000_0000h
1_9158h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD3_W0)	32	RW	0000_0000h
1_915Ch	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD3_W1)	32	RW	0000_0000h
1_9160h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD4_W0)	32	RW	0000_0000h
1_9164h	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD4_W1)	32	RW	0000_0000h
1_9168h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD5_W0)	32	RW	0000_0000h
1_916Ch	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD5_W1)	32	RW	0000_0000h
1_9170h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD6_W0)	32	RW	0000_0000h
1_9174h	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD6_W1)	32	RW	0000_0000h
1_9178h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD7_W0)	32	RW	0000_0000h
1_917Ch	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD7_W1)	32	RW	0000_0000h
1_9180h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD8_W0)	32	RW	0000_0000h
1_9184h	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD8_W1)	32	RW	0000_0000h
1_9188h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD9_W0)	32	RW	0000_0000h
1_918Ch	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD9_W1)	32	RW	0000_0000h
1_9190h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD10_W0)	32	RW	0000_0000h
1_9194h	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD10_W1)	32	RW	0000_0000h
1_9198h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD11_W0)	32	RW	0000_0000h
1_919Ch	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD11_W1)	32	RW	0000_0000h
1_91A0h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD12_W0)	32	RW	0000_0000h
1_91A4h	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD12_W1)	32	RW	0000_0000h
1_91A8h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD13_W0)	32	RW	0000_0000h
1_91ACh	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD13_W1)	32	RW	0000_0000h
1_91B0h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD14_W0)	32	RW	0000_0000h
1_91B4h	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD14_W1)	32	RW	0000_0000h
1_91B8h	MRC Region Descriptor Word 0 (MRC5_DOM1_RGD15_W0)	32	RW	0000_0000h
1_91BCh	MRC Region Descriptor Word 1 (MRC5_DOM1_RGD15_W1)	32	RW	0000_0000h
1_91C0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM1_RGD_NSE)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9240h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD0_W0)	32	RW	0000_0000h
1_9244h	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD0_W1)	32	RW	0000_0000h
1_9248h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD1_W0)	32	RW	0000_0000h
1_924Ch	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD1_W1)	32	RW	0000_0000h
1_9250h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD2_W0)	32	RW	0000_0000h
1_9254h	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD2_W1)	32	RW	0000_0000h
1_9258h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD3_W0)	32	RW	0000_0000h
1_925Ch	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD3_W1)	32	RW	0000_0000h
1_9260h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD4_W0)	32	RW	0000_0000h
1_9264h	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD4_W1)	32	RW	0000_0000h
1_9268h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD5_W0)	32	RW	0000_0000h
1_926Ch	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD5_W1)	32	RW	0000_0000h
1_9270h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD6_W0)	32	RW	0000_0000h
1_9274h	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD6_W1)	32	RW	0000_0000h
1_9278h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD7_W0)	32	RW	0000_0000h
1_927Ch	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD7_W1)	32	RW	0000_0000h
1_9280h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD8_W0)	32	RW	0000_0000h
1_9284h	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD8_W1)	32	RW	0000_0000h
1_9288h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD9_W0)	32	RW	0000_0000h
1_928Ch	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD9_W1)	32	RW	0000_0000h
1_9290h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD10_W0)	32	RW	0000_0000h
1_9294h	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD10_W1)	32	RW	0000_0000h
1_9298h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD11_W0)	32	RW	0000_0000h
1_929Ch	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD11_W1)	32	RW	0000_0000h
1_92A0h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD12_W0)	32	RW	0000_0000h
1_92A4h	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD12_W1)	32	RW	0000_0000h
1_92A8h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD13_W0)	32	RW	0000_0000h
1_92ACh	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD13_W1)	32	RW	0000_0000h
1_92B0h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD14_W0)	32	RW	0000_0000h
1_92B4h	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD14_W1)	32	RW	0000_0000h
1_92B8h	MRC Region Descriptor Word 0 (MRC5_DOM2_RGD15_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_92BCh	MRC Region Descriptor Word 1 (MRC5_DOM2_RGD15_W1)	32	RW	0000_0000h
1_92C0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM2_RGD_NSE)	32	RW	0000_0000h
1_9340h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD0_W0)	32	RW	0000_0000h
1_9344h	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD0_W1)	32	RW	0000_0000h
1_9348h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD1_W0)	32	RW	0000_0000h
1_934Ch	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD1_W1)	32	RW	0000_0000h
1_9350h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD2_W0)	32	RW	0000_0000h
1_9354h	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD2_W1)	32	RW	0000_0000h
1_9358h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD3_W0)	32	RW	0000_0000h
1_935Ch	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD3_W1)	32	RW	0000_0000h
1_9360h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD4_W0)	32	RW	0000_0000h
1_9364h	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD4_W1)	32	RW	0000_0000h
1_9368h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD5_W0)	32	RW	0000_0000h
1_936Ch	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD5_W1)	32	RW	0000_0000h
1_9370h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD6_W0)	32	RW	0000_0000h
1_9374h	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD6_W1)	32	RW	0000_0000h
1_9378h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD7_W0)	32	RW	0000_0000h
1_937Ch	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD7_W1)	32	RW	0000_0000h
1_9380h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD8_W0)	32	RW	0000_0000h
1_9384h	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD8_W1)	32	RW	0000_0000h
1_9388h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD9_W0)	32	RW	0000_0000h
1_938Ch	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD9_W1)	32	RW	0000_0000h
1_9390h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD10_W0)	32	RW	0000_0000h
1_9394h	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD10_W1)	32	RW	0000_0000h
1_9398h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD11_W0)	32	RW	0000_0000h
1_939Ch	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD11_W1)	32	RW	0000_0000h
1_93A0h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD12_W0)	32	RW	0000_0000h
1_93A4h	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD12_W1)	32	RW	0000_0000h
1_93A8h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD13_W0)	32	RW	0000_0000h
1_93ACh	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD13_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_93B0h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD14_W0)	32	RW	0000_0000h
1_93B4h	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD14_W1)	32	RW	0000_0000h
1_93B8h	MRC Region Descriptor Word 0 (MRC5_DOM3_RGD15_W0)	32	RW	0000_0000h
1_93BCh	MRC Region Descriptor Word 1 (MRC5_DOM3_RGD15_W1)	32	RW	0000_0000h
1_93C0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM3_RGD_NSE)	32	RW	0000_0000h
1_9440h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD0_W0)	32	RW	0000_0000h
1_9444h	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD0_W1)	32	RW	0000_0000h
1_9448h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD1_W0)	32	RW	0000_0000h
1_944Ch	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD1_W1)	32	RW	0000_0000h
1_9450h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD2_W0)	32	RW	0000_0000h
1_9454h	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD2_W1)	32	RW	0000_0000h
1_9458h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD3_W0)	32	RW	0000_0000h
1_945Ch	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD3_W1)	32	RW	0000_0000h
1_9460h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD4_W0)	32	RW	0000_0000h
1_9464h	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD4_W1)	32	RW	0000_0000h
1_9468h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD5_W0)	32	RW	0000_0000h
1_946Ch	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD5_W1)	32	RW	0000_0000h
1_9470h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD6_W0)	32	RW	0000_0000h
1_9474h	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD6_W1)	32	RW	0000_0000h
1_9478h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD7_W0)	32	RW	0000_0000h
1_947Ch	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD7_W1)	32	RW	0000_0000h
1_9480h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD8_W0)	32	RW	0000_0000h
1_9484h	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD8_W1)	32	RW	0000_0000h
1_9488h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD9_W0)	32	RW	0000_0000h
1_948Ch	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD9_W1)	32	RW	0000_0000h
1_9490h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD10_W0)	32	RW	0000_0000h
1_9494h	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD10_W1)	32	RW	0000_0000h
1_9498h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD11_W0)	32	RW	0000_0000h
1_949Ch	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD11_W1)	32	RW	0000_0000h
1_94A0h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD12_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_94A4h	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD12_W1)	32	RW	0000_0000h
1_94A8h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD13_W0)	32	RW	0000_0000h
1_94ACh	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD13_W1)	32	RW	0000_0000h
1_94B0h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD14_W0)	32	RW	0000_0000h
1_94B4h	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD14_W1)	32	RW	0000_0000h
1_94B8h	MRC Region Descriptor Word 0 (MRC5_DOM4_RGD15_W0)	32	RW	0000_0000h
1_94BCh	MRC Region Descriptor Word 1 (MRC5_DOM4_RGD15_W1)	32	RW	0000_0000h
1_94C0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM4_RGD_NSE)	32	RW	0000_0000h
1_9540h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD0_W0)	32	RW	0000_0000h
1_9544h	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD0_W1)	32	RW	0000_0000h
1_9548h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD1_W0)	32	RW	0000_0000h
1_954Ch	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD1_W1)	32	RW	0000_0000h
1_9550h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD2_W0)	32	RW	0000_0000h
1_9554h	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD2_W1)	32	RW	0000_0000h
1_9558h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD3_W0)	32	RW	0000_0000h
1_955Ch	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD3_W1)	32	RW	0000_0000h
1_9560h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD4_W0)	32	RW	0000_0000h
1_9564h	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD4_W1)	32	RW	0000_0000h
1_9568h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD5_W0)	32	RW	0000_0000h
1_956Ch	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD5_W1)	32	RW	0000_0000h
1_9570h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD6_W0)	32	RW	0000_0000h
1_9574h	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD6_W1)	32	RW	0000_0000h
1_9578h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD7_W0)	32	RW	0000_0000h
1_957Ch	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD7_W1)	32	RW	0000_0000h
1_9580h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD8_W0)	32	RW	0000_0000h
1_9584h	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD8_W1)	32	RW	0000_0000h
1_9588h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD9_W0)	32	RW	0000_0000h
1_958Ch	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD9_W1)	32	RW	0000_0000h
1_9590h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD10_W0)	32	RW	0000_0000h
1_9594h	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD10_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9598h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD11_W0)	32	RW	0000_0000h
1_959Ch	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD11_W1)	32	RW	0000_0000h
1_95A0h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD12_W0)	32	RW	0000_0000h
1_95A4h	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD12_W1)	32	RW	0000_0000h
1_95A8h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD13_W0)	32	RW	0000_0000h
1_95ACh	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD13_W1)	32	RW	0000_0000h
1_95B0h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD14_W0)	32	RW	0000_0000h
1_95B4h	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD14_W1)	32	RW	0000_0000h
1_95B8h	MRC Region Descriptor Word 0 (MRC5_DOM5_RGD15_W0)	32	RW	0000_0000h
1_95BCh	MRC Region Descriptor Word 1 (MRC5_DOM5_RGD15_W1)	32	RW	0000_0000h
1_95C0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM5_RGD_NSE)	32	RW	0000_0000h
1_9640h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD0_W0)	32	RW	0000_0000h
1_9644h	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD0_W1)	32	RW	0000_0000h
1_9648h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD1_W0)	32	RW	0000_0000h
1_964Ch	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD1_W1)	32	RW	0000_0000h
1_9650h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD2_W0)	32	RW	0000_0000h
1_9654h	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD2_W1)	32	RW	0000_0000h
1_9658h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD3_W0)	32	RW	0000_0000h
1_965Ch	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD3_W1)	32	RW	0000_0000h
1_9660h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD4_W0)	32	RW	0000_0000h
1_9664h	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD4_W1)	32	RW	0000_0000h
1_9668h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD5_W0)	32	RW	0000_0000h
1_966Ch	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD5_W1)	32	RW	0000_0000h
1_9670h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD6_W0)	32	RW	0000_0000h
1_9674h	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD6_W1)	32	RW	0000_0000h
1_9678h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD7_W0)	32	RW	0000_0000h
1_967Ch	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD7_W1)	32	RW	0000_0000h
1_9680h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD8_W0)	32	RW	0000_0000h
1_9684h	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD8_W1)	32	RW	0000_0000h
1_9688h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD9_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_968Ch	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD9_W1)	32	RW	0000_0000h
1_9690h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD10_W0)	32	RW	0000_0000h
1_9694h	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD10_W1)	32	RW	0000_0000h
1_9698h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD11_W0)	32	RW	0000_0000h
1_969Ch	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD11_W1)	32	RW	0000_0000h
1_96A0h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD12_W0)	32	RW	0000_0000h
1_96A4h	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD12_W1)	32	RW	0000_0000h
1_96A8h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD13_W0)	32	RW	0000_0000h
1_96ACh	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD13_W1)	32	RW	0000_0000h
1_96B0h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD14_W0)	32	RW	0000_0000h
1_96B4h	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD14_W1)	32	RW	0000_0000h
1_96B8h	MRC Region Descriptor Word 0 (MRC5_DOM6_RGD15_W0)	32	RW	0000_0000h
1_96BCh	MRC Region Descriptor Word 1 (MRC5_DOM6_RGD15_W1)	32	RW	0000_0000h
1_96C0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM6_RGD_NSE)	32	RW	0000_0000h
1_9740h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD0_W0)	32	RW	0000_0000h
1_9744h	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD0_W1)	32	RW	0000_0000h
1_9748h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD1_W0)	32	RW	0000_0000h
1_974Ch	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD1_W1)	32	RW	0000_0000h
1_9750h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD2_W0)	32	RW	0000_0000h
1_9754h	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD2_W1)	32	RW	0000_0000h
1_9758h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD3_W0)	32	RW	0000_0000h
1_975Ch	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD3_W1)	32	RW	0000_0000h
1_9760h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD4_W0)	32	RW	0000_0000h
1_9764h	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD4_W1)	32	RW	0000_0000h
1_9768h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD5_W0)	32	RW	0000_0000h
1_976Ch	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD5_W1)	32	RW	0000_0000h
1_9770h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD6_W0)	32	RW	0000_0000h
1_9774h	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD6_W1)	32	RW	0000_0000h
1_9778h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD7_W0)	32	RW	0000_0000h
1_977Ch	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD7_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9780h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD8_W0)	32	RW	0000_0000h
1_9784h	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD8_W1)	32	RW	0000_0000h
1_9788h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD9_W0)	32	RW	0000_0000h
1_978Ch	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD9_W1)	32	RW	0000_0000h
1_9790h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD10_W0)	32	RW	0000_0000h
1_9794h	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD10_W1)	32	RW	0000_0000h
1_9798h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD11_W0)	32	RW	0000_0000h
1_979Ch	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD11_W1)	32	RW	0000_0000h
1_97A0h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD12_W0)	32	RW	0000_0000h
1_97A4h	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD12_W1)	32	RW	0000_0000h
1_97A8h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD13_W0)	32	RW	0000_0000h
1_97ACh	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD13_W1)	32	RW	0000_0000h
1_97B0h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD14_W0)	32	RW	0000_0000h
1_97B4h	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD14_W1)	32	RW	0000_0000h
1_97B8h	MRC Region Descriptor Word 0 (MRC5_DOM7_RGD15_W0)	32	RW	0000_0000h
1_97BCh	MRC Region Descriptor Word 1 (MRC5_DOM7_RGD15_W1)	32	RW	0000_0000h
1_97C0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM7_RGD_NSE)	32	RW	0000_0000h
1_9840h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD0_W0)	32	RW	0000_0000h
1_9844h	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD0_W1)	32	RW	0000_0000h
1_9848h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD1_W0)	32	RW	0000_0000h
1_984Ch	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD1_W1)	32	RW	0000_0000h
1_9850h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD2_W0)	32	RW	0000_0000h
1_9854h	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD2_W1)	32	RW	0000_0000h
1_9858h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD3_W0)	32	RW	0000_0000h
1_985Ch	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD3_W1)	32	RW	0000_0000h
1_9860h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD4_W0)	32	RW	0000_0000h
1_9864h	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD4_W1)	32	RW	0000_0000h
1_9868h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD5_W0)	32	RW	0000_0000h
1_986Ch	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD5_W1)	32	RW	0000_0000h
1_9870h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD6_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9874h	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD6_W1)	32	RW	0000_0000h
1_9878h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD7_W0)	32	RW	0000_0000h
1_987Ch	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD7_W1)	32	RW	0000_0000h
1_9880h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD8_W0)	32	RW	0000_0000h
1_9884h	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD8_W1)	32	RW	0000_0000h
1_9888h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD9_W0)	32	RW	0000_0000h
1_988Ch	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD9_W1)	32	RW	0000_0000h
1_9890h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD10_W0)	32	RW	0000_0000h
1_9894h	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD10_W1)	32	RW	0000_0000h
1_9898h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD11_W0)	32	RW	0000_0000h
1_989Ch	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD11_W1)	32	RW	0000_0000h
1_98A0h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD12_W0)	32	RW	0000_0000h
1_98A4h	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD12_W1)	32	RW	0000_0000h
1_98A8h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD13_W0)	32	RW	0000_0000h
1_98ACh	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD13_W1)	32	RW	0000_0000h
1_98B0h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD14_W0)	32	RW	0000_0000h
1_98B4h	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD14_W1)	32	RW	0000_0000h
1_98B8h	MRC Region Descriptor Word 0 (MRC5_DOM8_RGD15_W0)	32	RW	0000_0000h
1_98BCh	MRC Region Descriptor Word 1 (MRC5_DOM8_RGD15_W1)	32	RW	0000_0000h
1_98C0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM8_RGD_NSE)	32	RW	0000_0000h
1_9940h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD0_W0)	32	RW	0000_0000h
1_9944h	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD0_W1)	32	RW	0000_0000h
1_9948h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD1_W0)	32	RW	0000_0000h
1_994Ch	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD1_W1)	32	RW	0000_0000h
1_9950h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD2_W0)	32	RW	0000_0000h
1_9954h	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD2_W1)	32	RW	0000_0000h
1_9958h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD3_W0)	32	RW	0000_0000h
1_995Ch	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD3_W1)	32	RW	0000_0000h
1_9960h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD4_W0)	32	RW	0000_0000h
1_9964h	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD4_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9968h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD5_W0)	32	RW	0000_0000h
1_996Ch	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD5_W1)	32	RW	0000_0000h
1_9970h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD6_W0)	32	RW	0000_0000h
1_9974h	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD6_W1)	32	RW	0000_0000h
1_9978h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD7_W0)	32	RW	0000_0000h
1_997Ch	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD7_W1)	32	RW	0000_0000h
1_9980h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD8_W0)	32	RW	0000_0000h
1_9984h	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD8_W1)	32	RW	0000_0000h
1_9988h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD9_W0)	32	RW	0000_0000h
1_998Ch	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD9_W1)	32	RW	0000_0000h
1_9990h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD10_W0)	32	RW	0000_0000h
1_9994h	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD10_W1)	32	RW	0000_0000h
1_9998h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD11_W0)	32	RW	0000_0000h
1_999Ch	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD11_W1)	32	RW	0000_0000h
1_99A0h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD12_W0)	32	RW	0000_0000h
1_99A4h	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD12_W1)	32	RW	0000_0000h
1_99A8h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD13_W0)	32	RW	0000_0000h
1_99ACh	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD13_W1)	32	RW	0000_0000h
1_99B0h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD14_W0)	32	RW	0000_0000h
1_99B4h	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD14_W1)	32	RW	0000_0000h
1_99B8h	MRC Region Descriptor Word 0 (MRC5_DOM9_RGD15_W0)	32	RW	0000_0000h
1_99BCh	MRC Region Descriptor Word 1 (MRC5_DOM9_RGD15_W1)	32	RW	0000_0000h
1_99C0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM9_RGD_NSE)	32	RW	0000_0000h
1_9A40h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD0_W0)	32	RW	0000_0000h
1_9A44h	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD0_W1)	32	RW	0000_0000h
1_9A48h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD1_W0)	32	RW	0000_0000h
1_9A4Ch	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD1_W1)	32	RW	0000_0000h
1_9A50h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD2_W0)	32	RW	0000_0000h
1_9A54h	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD2_W1)	32	RW	0000_0000h
1_9A58h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD3_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9A5Ch	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD3_W1)	32	RW	0000_0000h
1_9A60h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD4_W0)	32	RW	0000_0000h
1_9A64h	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD4_W1)	32	RW	0000_0000h
1_9A68h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD5_W0)	32	RW	0000_0000h
1_9A6Ch	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD5_W1)	32	RW	0000_0000h
1_9A70h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD6_W0)	32	RW	0000_0000h
1_9A74h	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD6_W1)	32	RW	0000_0000h
1_9A78h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD7_W0)	32	RW	0000_0000h
1_9A7Ch	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD7_W1)	32	RW	0000_0000h
1_9A80h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD8_W0)	32	RW	0000_0000h
1_9A84h	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD8_W1)	32	RW	0000_0000h
1_9A88h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD9_W0)	32	RW	0000_0000h
1_9A8Ch	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD9_W1)	32	RW	0000_0000h
1_9A90h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD10_W0)	32	RW	0000_0000h
1_9A94h	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD10_W1)	32	RW	0000_0000h
1_9A98h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD11_W0)	32	RW	0000_0000h
1_9A9Ch	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD11_W1)	32	RW	0000_0000h
1_9AA0h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD12_W0)	32	RW	0000_0000h
1_9AA4h	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD12_W1)	32	RW	0000_0000h
1_9AA8h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD13_W0)	32	RW	0000_0000h
1_9AACh	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD13_W1)	32	RW	0000_0000h
1_9AB0h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD14_W0)	32	RW	0000_0000h
1_9AB4h	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD14_W1)	32	RW	0000_0000h
1_9AB8h	MRC Region Descriptor Word 0 (MRC5_DOM10_RGD15_W0)	32	RW	0000_0000h
1_9ABCh	MRC Region Descriptor Word 1 (MRC5_DOM10_RGD15_W1)	32	RW	0000_0000h
1_9AC0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM10_RGD_NSE)	32	RW	0000_0000h
1_9B40h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD0_W0)	32	RW	0000_0000h
1_9B44h	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD0_W1)	32	RW	0000_0000h
1_9B48h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD1_W0)	32	RW	0000_0000h
1_9B4Ch	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD1_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9B50h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD2_W0)	32	RW	0000_0000h
1_9B54h	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD2_W1)	32	RW	0000_0000h
1_9B58h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD3_W0)	32	RW	0000_0000h
1_9B5Ch	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD3_W1)	32	RW	0000_0000h
1_9B60h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD4_W0)	32	RW	0000_0000h
1_9B64h	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD4_W1)	32	RW	0000_0000h
1_9B68h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD5_W0)	32	RW	0000_0000h
1_9B6Ch	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD5_W1)	32	RW	0000_0000h
1_9B70h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD6_W0)	32	RW	0000_0000h
1_9B74h	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD6_W1)	32	RW	0000_0000h
1_9B78h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD7_W0)	32	RW	0000_0000h
1_9B7Ch	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD7_W1)	32	RW	0000_0000h
1_9B80h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD8_W0)	32	RW	0000_0000h
1_9B84h	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD8_W1)	32	RW	0000_0000h
1_9B88h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD9_W0)	32	RW	0000_0000h
1_9B8Ch	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD9_W1)	32	RW	0000_0000h
1_9B90h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD10_W0)	32	RW	0000_0000h
1_9B94h	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD10_W1)	32	RW	0000_0000h
1_9B98h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD11_W0)	32	RW	0000_0000h
1_9B9Ch	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD11_W1)	32	RW	0000_0000h
1_9BA0h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD12_W0)	32	RW	0000_0000h
1_9BA4h	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD12_W1)	32	RW	0000_0000h
1_9BA8h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD13_W0)	32	RW	0000_0000h
1_9BACh	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD13_W1)	32	RW	0000_0000h
1_9BB0h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD14_W0)	32	RW	0000_0000h
1_9BB4h	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD14_W1)	32	RW	0000_0000h
1_9BB8h	MRC Region Descriptor Word 0 (MRC5_DOM11_RGD15_W0)	32	RW	0000_0000h
1_9BBCh	MRC Region Descriptor Word 1 (MRC5_DOM11_RGD15_W1)	32	RW	0000_0000h
1_9BC0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM11_RGD_NSE)	32	RW	0000_0000h
1_9C40h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD0_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9C44h	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD0_W1)	32	RW	0000_0000h
1_9C48h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD1_W0)	32	RW	0000_0000h
1_9C4Ch	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD1_W1)	32	RW	0000_0000h
1_9C50h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD2_W0)	32	RW	0000_0000h
1_9C54h	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD2_W1)	32	RW	0000_0000h
1_9C58h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD3_W0)	32	RW	0000_0000h
1_9C5Ch	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD3_W1)	32	RW	0000_0000h
1_9C60h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD4_W0)	32	RW	0000_0000h
1_9C64h	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD4_W1)	32	RW	0000_0000h
1_9C68h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD5_W0)	32	RW	0000_0000h
1_9C6Ch	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD5_W1)	32	RW	0000_0000h
1_9C70h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD6_W0)	32	RW	0000_0000h
1_9C74h	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD6_W1)	32	RW	0000_0000h
1_9C78h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD7_W0)	32	RW	0000_0000h
1_9C7Ch	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD7_W1)	32	RW	0000_0000h
1_9C80h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD8_W0)	32	RW	0000_0000h
1_9C84h	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD8_W1)	32	RW	0000_0000h
1_9C88h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD9_W0)	32	RW	0000_0000h
1_9C8Ch	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD9_W1)	32	RW	0000_0000h
1_9C90h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD10_W0)	32	RW	0000_0000h
1_9C94h	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD10_W1)	32	RW	0000_0000h
1_9C98h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD11_W0)	32	RW	0000_0000h
1_9C9Ch	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD11_W1)	32	RW	0000_0000h
1_9CA0h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD12_W0)	32	RW	0000_0000h
1_9CA4h	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD12_W1)	32	RW	0000_0000h
1_9CA8h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD13_W0)	32	RW	0000_0000h
1_9CACh	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD13_W1)	32	RW	0000_0000h
1_9CB0h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD14_W0)	32	RW	0000_0000h
1_9CB4h	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD14_W1)	32	RW	0000_0000h
1_9CB8h	MRC Region Descriptor Word 0 (MRC5_DOM12_RGD15_W0)	32	RW	0000_0000h
1_9CBCh	MRC Region Descriptor Word 1 (MRC5_DOM12_RGD15_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9CC0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM12_RGD_NSE)	32	RW	0000_0000h
1_9D40h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD0_W0)	32	RW	0000_0000h
1_9D44h	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD0_W1)	32	RW	0000_0000h
1_9D48h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD1_W0)	32	RW	0000_0000h
1_9D4Ch	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD1_W1)	32	RW	0000_0000h
1_9D50h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD2_W0)	32	RW	0000_0000h
1_9D54h	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD2_W1)	32	RW	0000_0000h
1_9D58h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD3_W0)	32	RW	0000_0000h
1_9D5Ch	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD3_W1)	32	RW	0000_0000h
1_9D60h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD4_W0)	32	RW	0000_0000h
1_9D64h	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD4_W1)	32	RW	0000_0000h
1_9D68h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD5_W0)	32	RW	0000_0000h
1_9D6Ch	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD5_W1)	32	RW	0000_0000h
1_9D70h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD6_W0)	32	RW	0000_0000h
1_9D74h	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD6_W1)	32	RW	0000_0000h
1_9D78h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD7_W0)	32	RW	0000_0000h
1_9D7Ch	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD7_W1)	32	RW	0000_0000h
1_9D80h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD8_W0)	32	RW	0000_0000h
1_9D84h	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD8_W1)	32	RW	0000_0000h
1_9D88h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD9_W0)	32	RW	0000_0000h
1_9D8Ch	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD9_W1)	32	RW	0000_0000h
1_9D90h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD10_W0)	32	RW	0000_0000h
1_9D94h	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD10_W1)	32	RW	0000_0000h
1_9D98h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD11_W0)	32	RW	0000_0000h
1_9D9Ch	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD11_W1)	32	RW	0000_0000h
1_9DA0h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD12_W0)	32	RW	0000_0000h
1_9DA4h	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD12_W1)	32	RW	0000_0000h
1_9DA8h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD13_W0)	32	RW	0000_0000h
1_9DACh	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD13_W1)	32	RW	0000_0000h
1_9DB0h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD14_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9DB4h	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD14_W1)	32	RW	0000_0000h
1_9DB8h	MRC Region Descriptor Word 0 (MRC5_DOM13_RGD15_W0)	32	RW	0000_0000h
1_9DBCh	MRC Region Descriptor Word 1 (MRC5_DOM13_RGD15_W1)	32	RW	0000_0000h
1_9DC0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM13_RGD_NSE)	32	RW	0000_0000h
1_9E40h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD0_W0)	32	RW	0000_0000h
1_9E44h	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD0_W1)	32	RW	0000_0000h
1_9E48h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD1_W0)	32	RW	0000_0000h
1_9E4Ch	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD1_W1)	32	RW	0000_0000h
1_9E50h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD2_W0)	32	RW	0000_0000h
1_9E54h	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD2_W1)	32	RW	0000_0000h
1_9E58h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD3_W0)	32	RW	0000_0000h
1_9E5Ch	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD3_W1)	32	RW	0000_0000h
1_9E60h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD4_W0)	32	RW	0000_0000h
1_9E64h	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD4_W1)	32	RW	0000_0000h
1_9E68h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD5_W0)	32	RW	0000_0000h
1_9E6Ch	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD5_W1)	32	RW	0000_0000h
1_9E70h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD6_W0)	32	RW	0000_0000h
1_9E74h	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD6_W1)	32	RW	0000_0000h
1_9E78h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD7_W0)	32	RW	0000_0000h
1_9E7Ch	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD7_W1)	32	RW	0000_0000h
1_9E80h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD8_W0)	32	RW	0000_0000h
1_9E84h	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD8_W1)	32	RW	0000_0000h
1_9E88h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD9_W0)	32	RW	0000_0000h
1_9E8Ch	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD9_W1)	32	RW	0000_0000h
1_9E90h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD10_W0)	32	RW	0000_0000h
1_9E94h	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD10_W1)	32	RW	0000_0000h
1_9E98h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD11_W0)	32	RW	0000_0000h
1_9E9Ch	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD11_W1)	32	RW	0000_0000h
1_9EA0h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD12_W0)	32	RW	0000_0000h
1_9EA4h	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD12_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9EA8h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD13_W0)	32	RW	0000_0000h
1_9EACh	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD13_W1)	32	RW	0000_0000h
1_9EB0h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD14_W0)	32	RW	0000_0000h
1_9EB4h	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD14_W1)	32	RW	0000_0000h
1_9EB8h	MRC Region Descriptor Word 0 (MRC5_DOM14_RGD15_W0)	32	RW	0000_0000h
1_9EBCh	MRC Region Descriptor Word 1 (MRC5_DOM14_RGD15_W1)	32	RW	0000_0000h
1_9EC0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM14_RGD_NSE)	32	RW	0000_0000h
1_9F40h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD0_W0)	32	RW	0000_0000h
1_9F44h	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD0_W1)	32	RW	0000_0000h
1_9F48h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD1_W0)	32	RW	0000_0000h
1_9F4Ch	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD1_W1)	32	RW	0000_0000h
1_9F50h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD2_W0)	32	RW	0000_0000h
1_9F54h	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD2_W1)	32	RW	0000_0000h
1_9F58h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD3_W0)	32	RW	0000_0000h
1_9F5Ch	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD3_W1)	32	RW	0000_0000h
1_9F60h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD4_W0)	32	RW	0000_0000h
1_9F64h	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD4_W1)	32	RW	0000_0000h
1_9F68h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD5_W0)	32	RW	0000_0000h
1_9F6Ch	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD5_W1)	32	RW	0000_0000h
1_9F70h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD6_W0)	32	RW	0000_0000h
1_9F74h	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD6_W1)	32	RW	0000_0000h
1_9F78h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD7_W0)	32	RW	0000_0000h
1_9F7Ch	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD7_W1)	32	RW	0000_0000h
1_9F80h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD8_W0)	32	RW	0000_0000h
1_9F84h	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD8_W1)	32	RW	0000_0000h
1_9F88h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD9_W0)	32	RW	0000_0000h
1_9F8Ch	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD9_W1)	32	RW	0000_0000h
1_9F90h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD10_W0)	32	RW	0000_0000h
1_9F94h	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD10_W1)	32	RW	0000_0000h
1_9F98h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD11_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_9F9Ch	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD11_W1)	32	RW	0000_0000h
1_9FA0h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD12_W0)	32	RW	0000_0000h
1_9FA4h	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD12_W1)	32	RW	0000_0000h
1_9FA8h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD13_W0)	32	RW	0000_0000h
1_9FACh	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD13_W1)	32	RW	0000_0000h
1_9FB0h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD14_W0)	32	RW	0000_0000h
1_9FB4h	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD14_W1)	32	RW	0000_0000h
1_9FB8h	MRC Region Descriptor Word 0 (MRC5_DOM15_RGD15_W0)	32	RW	0000_0000h
1_9FBCh	MRC Region Descriptor Word 1 (MRC5_DOM15_RGD15_W1)	32	RW	0000_0000h
1_9FC0h	MRC Region Descriptor NonSecure Enable (MRC5_DOM15_RGD_NSE)	32	RW	0000_0000h
1_A000h	MRC Global Configuration Register (MRC6_GLBCFG)	32	R	0000_0010h
1_A010h	MRC NonSecure Enable Region Indirect (MRC6_NSE_RGN_INDIRECT)	32	RW	0000_0000h
1_A014h	MRC NonSecure Enable Region Set (MRC6_NSE_RGN_SET)	32	RW	0000_0000h
1_A018h	MRC NonSecure Enable Region Clear (MRC6_NSE_RGN_CLR)	32	RW	0000_0000h
1_A01Ch	MRC NonSecure Enable Region Clear All (MRC6_NSE_RGN_CLR_ALL)	32	RW	0000_0000h
1_A020h	MRC Global Access Control (MRC6_GLBAC0)	32	RW	0000_0000h
1_A024h	MRC Global Access Control (MRC6_GLBAC1)	32	RW	0000_0000h
1_A028h	MRC Global Access Control (MRC6_GLBAC2)	32	RW	0000_0000h
1_A02Ch	MRC Global Access Control (MRC6_GLBAC3)	32	RW	0000_0000h
1_A030h	MRC Global Access Control (MRC6_GLBAC4)	32	RW	0000_0000h
1_A034h	MRC Global Access Control (MRC6_GLBAC5)	32	RW	0000_0000h
1_A038h	MRC Global Access Control (MRC6_GLBAC6)	32	RW	0000_0000h
1_A03Ch	MRC Global Access Control (MRC6_GLBAC7)	32	RW	0000_0000h
1_A040h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD0_W0)	32	RW	0000_0000h
1_A044h	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD0_W1)	32	RW	0000_0000h
1_A048h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD1_W0)	32	RW	0000_0000h
1_A04Ch	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD1_W1)	32	RW	0000_0000h
1_A050h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD2_W0)	32	RW	0000_0000h
1_A054h	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD2_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_A058h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD3_W0)	32	RW	0000_0000h
1_A05Ch	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD3_W1)	32	RW	0000_0000h
1_A060h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD4_W0)	32	RW	0000_0000h
1_A064h	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD4_W1)	32	RW	0000_0000h
1_A068h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD5_W0)	32	RW	0000_0000h
1_A06Ch	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD5_W1)	32	RW	0000_0000h
1_A070h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD6_W0)	32	RW	0000_0000h
1_A074h	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD6_W1)	32	RW	0000_0000h
1_A078h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD7_W0)	32	RW	0000_0000h
1_A07Ch	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD7_W1)	32	RW	0000_0000h
1_A080h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD8_W0)	32	RW	0000_0000h
1_A084h	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD8_W1)	32	RW	0000_0000h
1_A088h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD9_W0)	32	RW	0000_0000h
1_A08Ch	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD9_W1)	32	RW	0000_0000h
1_A090h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD10_W0)	32	RW	0000_0000h
1_A094h	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD10_W1)	32	RW	0000_0000h
1_A098h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD11_W0)	32	RW	0000_0000h
1_A09Ch	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD11_W1)	32	RW	0000_0000h
1_A0A0h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD12_W0)	32	RW	0000_0000h
1_A0A4h	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD12_W1)	32	RW	0000_0000h
1_A0A8h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD13_W0)	32	RW	0000_0000h
1_A0ACh	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD13_W1)	32	RW	0000_0000h
1_A0B0h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD14_W0)	32	RW	0000_0000h
1_A0B4h	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD14_W1)	32	RW	0000_0000h
1_A0B8h	MRC Region Descriptor Word 0 (MRC6_DOM0_RGD15_W0)	32	RW	0000_0000h
1_A0BCh	MRC Region Descriptor Word 1 (MRC6_DOM0_RGD15_W1)	32	RW	0000_0000h
1_A0C0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM0_RGD_NSE)	32	RW	0000_0000h
1_A140h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD0_W0)	32	RW	0000_0000h
1_A144h	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD0_W1)	32	RW	0000_0000h
1_A148h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD1_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_A14Ch	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD1_W1)	32	RW	0000_0000h
1_A150h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD2_W0)	32	RW	0000_0000h
1_A154h	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD2_W1)	32	RW	0000_0000h
1_A158h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD3_W0)	32	RW	0000_0000h
1_A15Ch	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD3_W1)	32	RW	0000_0000h
1_A160h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD4_W0)	32	RW	0000_0000h
1_A164h	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD4_W1)	32	RW	0000_0000h
1_A168h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD5_W0)	32	RW	0000_0000h
1_A16Ch	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD5_W1)	32	RW	0000_0000h
1_A170h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD6_W0)	32	RW	0000_0000h
1_A174h	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD6_W1)	32	RW	0000_0000h
1_A178h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD7_W0)	32	RW	0000_0000h
1_A17Ch	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD7_W1)	32	RW	0000_0000h
1_A180h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD8_W0)	32	RW	0000_0000h
1_A184h	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD8_W1)	32	RW	0000_0000h
1_A188h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD9_W0)	32	RW	0000_0000h
1_A18Ch	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD9_W1)	32	RW	0000_0000h
1_A190h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD10_W0)	32	RW	0000_0000h
1_A194h	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD10_W1)	32	RW	0000_0000h
1_A198h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD11_W0)	32	RW	0000_0000h
1_A19Ch	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD11_W1)	32	RW	0000_0000h
1_A1A0h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD12_W0)	32	RW	0000_0000h
1_A1A4h	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD12_W1)	32	RW	0000_0000h
1_A1A8h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD13_W0)	32	RW	0000_0000h
1_A1ACh	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD13_W1)	32	RW	0000_0000h
1_A1B0h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD14_W0)	32	RW	0000_0000h
1_A1B4h	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD14_W1)	32	RW	0000_0000h
1_A1B8h	MRC Region Descriptor Word 0 (MRC6_DOM1_RGD15_W0)	32	RW	0000_0000h
1_A1BCh	MRC Region Descriptor Word 1 (MRC6_DOM1_RGD15_W1)	32	RW	0000_0000h
1_A1C0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM1_RGD_NSE)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_A240h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD0_W0)	32	RW	0000_0000h
1_A244h	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD0_W1)	32	RW	0000_0000h
1_A248h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD1_W0)	32	RW	0000_0000h
1_A24Ch	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD1_W1)	32	RW	0000_0000h
1_A250h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD2_W0)	32	RW	0000_0000h
1_A254h	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD2_W1)	32	RW	0000_0000h
1_A258h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD3_W0)	32	RW	0000_0000h
1_A25Ch	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD3_W1)	32	RW	0000_0000h
1_A260h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD4_W0)	32	RW	0000_0000h
1_A264h	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD4_W1)	32	RW	0000_0000h
1_A268h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD5_W0)	32	RW	0000_0000h
1_A26Ch	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD5_W1)	32	RW	0000_0000h
1_A270h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD6_W0)	32	RW	0000_0000h
1_A274h	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD6_W1)	32	RW	0000_0000h
1_A278h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD7_W0)	32	RW	0000_0000h
1_A27Ch	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD7_W1)	32	RW	0000_0000h
1_A280h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD8_W0)	32	RW	0000_0000h
1_A284h	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD8_W1)	32	RW	0000_0000h
1_A288h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD9_W0)	32	RW	0000_0000h
1_A28Ch	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD9_W1)	32	RW	0000_0000h
1_A290h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD10_W0)	32	RW	0000_0000h
1_A294h	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD10_W1)	32	RW	0000_0000h
1_A298h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD11_W0)	32	RW	0000_0000h
1_A29Ch	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD11_W1)	32	RW	0000_0000h
1_A2A0h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD12_W0)	32	RW	0000_0000h
1_A2A4h	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD12_W1)	32	RW	0000_0000h
1_A2A8h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD13_W0)	32	RW	0000_0000h
1_A2ACh	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD13_W1)	32	RW	0000_0000h
1_A2B0h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD14_W0)	32	RW	0000_0000h
1_A2B4h	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD14_W1)	32	RW	0000_0000h
1_A2B8h	MRC Region Descriptor Word 0 (MRC6_DOM2_RGD15_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_A2BCh	MRC Region Descriptor Word 1 (MRC6_DOM2_RGD15_W1)	32	RW	0000_0000h
1_A2C0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM2_RGD_NSE)	32	RW	0000_0000h
1_A340h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD0_W0)	32	RW	0000_0000h
1_A344h	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD0_W1)	32	RW	0000_0000h
1_A348h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD1_W0)	32	RW	0000_0000h
1_A34Ch	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD1_W1)	32	RW	0000_0000h
1_A350h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD2_W0)	32	RW	0000_0000h
1_A354h	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD2_W1)	32	RW	0000_0000h
1_A358h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD3_W0)	32	RW	0000_0000h
1_A35Ch	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD3_W1)	32	RW	0000_0000h
1_A360h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD4_W0)	32	RW	0000_0000h
1_A364h	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD4_W1)	32	RW	0000_0000h
1_A368h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD5_W0)	32	RW	0000_0000h
1_A36Ch	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD5_W1)	32	RW	0000_0000h
1_A370h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD6_W0)	32	RW	0000_0000h
1_A374h	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD6_W1)	32	RW	0000_0000h
1_A378h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD7_W0)	32	RW	0000_0000h
1_A37Ch	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD7_W1)	32	RW	0000_0000h
1_A380h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD8_W0)	32	RW	0000_0000h
1_A384h	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD8_W1)	32	RW	0000_0000h
1_A388h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD9_W0)	32	RW	0000_0000h
1_A38Ch	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD9_W1)	32	RW	0000_0000h
1_A390h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD10_W0)	32	RW	0000_0000h
1_A394h	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD10_W1)	32	RW	0000_0000h
1_A398h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD11_W0)	32	RW	0000_0000h
1_A39Ch	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD11_W1)	32	RW	0000_0000h
1_A3A0h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD12_W0)	32	RW	0000_0000h
1_A3A4h	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD12_W1)	32	RW	0000_0000h
1_A3A8h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD13_W0)	32	RW	0000_0000h
1_A3ACh	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD13_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_A3B0h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD14_W0)	32	RW	0000_0000h
1_A3B4h	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD14_W1)	32	RW	0000_0000h
1_A3B8h	MRC Region Descriptor Word 0 (MRC6_DOM3_RGD15_W0)	32	RW	0000_0000h
1_A3BCh	MRC Region Descriptor Word 1 (MRC6_DOM3_RGD15_W1)	32	RW	0000_0000h
1_A3C0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM3_RGD_NSE)	32	RW	0000_0000h
1_A440h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD0_W0)	32	RW	0000_0000h
1_A444h	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD0_W1)	32	RW	0000_0000h
1_A448h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD1_W0)	32	RW	0000_0000h
1_A44Ch	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD1_W1)	32	RW	0000_0000h
1_A450h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD2_W0)	32	RW	0000_0000h
1_A454h	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD2_W1)	32	RW	0000_0000h
1_A458h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD3_W0)	32	RW	0000_0000h
1_A45Ch	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD3_W1)	32	RW	0000_0000h
1_A460h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD4_W0)	32	RW	0000_0000h
1_A464h	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD4_W1)	32	RW	0000_0000h
1_A468h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD5_W0)	32	RW	0000_0000h
1_A46Ch	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD5_W1)	32	RW	0000_0000h
1_A470h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD6_W0)	32	RW	0000_0000h
1_A474h	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD6_W1)	32	RW	0000_0000h
1_A478h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD7_W0)	32	RW	0000_0000h
1_A47Ch	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD7_W1)	32	RW	0000_0000h
1_A480h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD8_W0)	32	RW	0000_0000h
1_A484h	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD8_W1)	32	RW	0000_0000h
1_A488h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD9_W0)	32	RW	0000_0000h
1_A48Ch	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD9_W1)	32	RW	0000_0000h
1_A490h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD10_W0)	32	RW	0000_0000h
1_A494h	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD10_W1)	32	RW	0000_0000h
1_A498h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD11_W0)	32	RW	0000_0000h
1_A49Ch	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD11_W1)	32	RW	0000_0000h
1_A4A0h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD12_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_A4A4h	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD12_W1)	32	RW	0000_0000h
1_A4A8h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD13_W0)	32	RW	0000_0000h
1_A4ACh	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD13_W1)	32	RW	0000_0000h
1_A4B0h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD14_W0)	32	RW	0000_0000h
1_A4B4h	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD14_W1)	32	RW	0000_0000h
1_A4B8h	MRC Region Descriptor Word 0 (MRC6_DOM4_RGD15_W0)	32	RW	0000_0000h
1_A4BCh	MRC Region Descriptor Word 1 (MRC6_DOM4_RGD15_W1)	32	RW	0000_0000h
1_A4C0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM4_RGD_NSE)	32	RW	0000_0000h
1_A540h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD0_W0)	32	RW	0000_0000h
1_A544h	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD0_W1)	32	RW	0000_0000h
1_A548h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD1_W0)	32	RW	0000_0000h
1_A54Ch	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD1_W1)	32	RW	0000_0000h
1_A550h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD2_W0)	32	RW	0000_0000h
1_A554h	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD2_W1)	32	RW	0000_0000h
1_A558h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD3_W0)	32	RW	0000_0000h
1_A55Ch	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD3_W1)	32	RW	0000_0000h
1_A560h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD4_W0)	32	RW	0000_0000h
1_A564h	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD4_W1)	32	RW	0000_0000h
1_A568h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD5_W0)	32	RW	0000_0000h
1_A56Ch	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD5_W1)	32	RW	0000_0000h
1_A570h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD6_W0)	32	RW	0000_0000h
1_A574h	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD6_W1)	32	RW	0000_0000h
1_A578h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD7_W0)	32	RW	0000_0000h
1_A57Ch	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD7_W1)	32	RW	0000_0000h
1_A580h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD8_W0)	32	RW	0000_0000h
1_A584h	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD8_W1)	32	RW	0000_0000h
1_A588h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD9_W0)	32	RW	0000_0000h
1_A58Ch	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD9_W1)	32	RW	0000_0000h
1_A590h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD10_W0)	32	RW	0000_0000h
1_A594h	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD10_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_A598h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD11_W0)	32	RW	0000_0000h
1_A59Ch	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD11_W1)	32	RW	0000_0000h
1_A5A0h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD12_W0)	32	RW	0000_0000h
1_A5A4h	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD12_W1)	32	RW	0000_0000h
1_A5A8h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD13_W0)	32	RW	0000_0000h
1_A5ACh	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD13_W1)	32	RW	0000_0000h
1_A5B0h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD14_W0)	32	RW	0000_0000h
1_A5B4h	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD14_W1)	32	RW	0000_0000h
1_A5B8h	MRC Region Descriptor Word 0 (MRC6_DOM5_RGD15_W0)	32	RW	0000_0000h
1_A5BCh	MRC Region Descriptor Word 1 (MRC6_DOM5_RGD15_W1)	32	RW	0000_0000h
1_A5C0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM5_RGD_NSE)	32	RW	0000_0000h
1_A640h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD0_W0)	32	RW	0000_0000h
1_A644h	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD0_W1)	32	RW	0000_0000h
1_A648h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD1_W0)	32	RW	0000_0000h
1_A64Ch	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD1_W1)	32	RW	0000_0000h
1_A650h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD2_W0)	32	RW	0000_0000h
1_A654h	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD2_W1)	32	RW	0000_0000h
1_A658h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD3_W0)	32	RW	0000_0000h
1_A65Ch	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD3_W1)	32	RW	0000_0000h
1_A660h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD4_W0)	32	RW	0000_0000h
1_A664h	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD4_W1)	32	RW	0000_0000h
1_A668h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD5_W0)	32	RW	0000_0000h
1_A66Ch	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD5_W1)	32	RW	0000_0000h
1_A670h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD6_W0)	32	RW	0000_0000h
1_A674h	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD6_W1)	32	RW	0000_0000h
1_A678h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD7_W0)	32	RW	0000_0000h
1_A67Ch	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD7_W1)	32	RW	0000_0000h
1_A680h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD8_W0)	32	RW	0000_0000h
1_A684h	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD8_W1)	32	RW	0000_0000h
1_A688h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD9_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_A68Ch	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD9_W1)	32	RW	0000_0000h
1_A690h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD10_W0)	32	RW	0000_0000h
1_A694h	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD10_W1)	32	RW	0000_0000h
1_A698h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD11_W0)	32	RW	0000_0000h
1_A69Ch	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD11_W1)	32	RW	0000_0000h
1_A6A0h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD12_W0)	32	RW	0000_0000h
1_A6A4h	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD12_W1)	32	RW	0000_0000h
1_A6A8h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD13_W0)	32	RW	0000_0000h
1_A6ACh	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD13_W1)	32	RW	0000_0000h
1_A6B0h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD14_W0)	32	RW	0000_0000h
1_A6B4h	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD14_W1)	32	RW	0000_0000h
1_A6B8h	MRC Region Descriptor Word 0 (MRC6_DOM6_RGD15_W0)	32	RW	0000_0000h
1_A6BCh	MRC Region Descriptor Word 1 (MRC6_DOM6_RGD15_W1)	32	RW	0000_0000h
1_A6C0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM6_RGD_NSE)	32	RW	0000_0000h
1_A740h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD0_W0)	32	RW	0000_0000h
1_A744h	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD0_W1)	32	RW	0000_0000h
1_A748h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD1_W0)	32	RW	0000_0000h
1_A74Ch	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD1_W1)	32	RW	0000_0000h
1_A750h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD2_W0)	32	RW	0000_0000h
1_A754h	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD2_W1)	32	RW	0000_0000h
1_A758h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD3_W0)	32	RW	0000_0000h
1_A75Ch	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD3_W1)	32	RW	0000_0000h
1_A760h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD4_W0)	32	RW	0000_0000h
1_A764h	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD4_W1)	32	RW	0000_0000h
1_A768h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD5_W0)	32	RW	0000_0000h
1_A76Ch	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD5_W1)	32	RW	0000_0000h
1_A770h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD6_W0)	32	RW	0000_0000h
1_A774h	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD6_W1)	32	RW	0000_0000h
1_A778h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD7_W0)	32	RW	0000_0000h
1_A77Ch	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD7_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_A780h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD8_W0)	32	RW	0000_0000h
1_A784h	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD8_W1)	32	RW	0000_0000h
1_A788h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD9_W0)	32	RW	0000_0000h
1_A78Ch	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD9_W1)	32	RW	0000_0000h
1_A790h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD10_W0)	32	RW	0000_0000h
1_A794h	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD10_W1)	32	RW	0000_0000h
1_A798h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD11_W0)	32	RW	0000_0000h
1_A79Ch	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD11_W1)	32	RW	0000_0000h
1_A7A0h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD12_W0)	32	RW	0000_0000h
1_A7A4h	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD12_W1)	32	RW	0000_0000h
1_A7A8h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD13_W0)	32	RW	0000_0000h
1_A7ACh	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD13_W1)	32	RW	0000_0000h
1_A7B0h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD14_W0)	32	RW	0000_0000h
1_A7B4h	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD14_W1)	32	RW	0000_0000h
1_A7B8h	MRC Region Descriptor Word 0 (MRC6_DOM7_RGD15_W0)	32	RW	0000_0000h
1_A7BCh	MRC Region Descriptor Word 1 (MRC6_DOM7_RGD15_W1)	32	RW	0000_0000h
1_A7C0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM7_RGD_NSE)	32	RW	0000_0000h
1_A840h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD0_W0)	32	RW	0000_0000h
1_A844h	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD0_W1)	32	RW	0000_0000h
1_A848h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD1_W0)	32	RW	0000_0000h
1_A84Ch	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD1_W1)	32	RW	0000_0000h
1_A850h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD2_W0)	32	RW	0000_0000h
1_A854h	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD2_W1)	32	RW	0000_0000h
1_A858h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD3_W0)	32	RW	0000_0000h
1_A85Ch	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD3_W1)	32	RW	0000_0000h
1_A860h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD4_W0)	32	RW	0000_0000h
1_A864h	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD4_W1)	32	RW	0000_0000h
1_A868h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD5_W0)	32	RW	0000_0000h
1_A86Ch	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD5_W1)	32	RW	0000_0000h
1_A870h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD6_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_A874h	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD6_W1)	32	RW	0000_0000h
1_A878h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD7_W0)	32	RW	0000_0000h
1_A87Ch	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD7_W1)	32	RW	0000_0000h
1_A880h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD8_W0)	32	RW	0000_0000h
1_A884h	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD8_W1)	32	RW	0000_0000h
1_A888h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD9_W0)	32	RW	0000_0000h
1_A88Ch	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD9_W1)	32	RW	0000_0000h
1_A890h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD10_W0)	32	RW	0000_0000h
1_A894h	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD10_W1)	32	RW	0000_0000h
1_A898h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD11_W0)	32	RW	0000_0000h
1_A89Ch	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD11_W1)	32	RW	0000_0000h
1_A8A0h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD12_W0)	32	RW	0000_0000h
1_A8A4h	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD12_W1)	32	RW	0000_0000h
1_A8A8h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD13_W0)	32	RW	0000_0000h
1_A8ACh	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD13_W1)	32	RW	0000_0000h
1_A8B0h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD14_W0)	32	RW	0000_0000h
1_A8B4h	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD14_W1)	32	RW	0000_0000h
1_A8B8h	MRC Region Descriptor Word 0 (MRC6_DOM8_RGD15_W0)	32	RW	0000_0000h
1_A8BCh	MRC Region Descriptor Word 1 (MRC6_DOM8_RGD15_W1)	32	RW	0000_0000h
1_A8C0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM8_RGD_NSE)	32	RW	0000_0000h
1_A940h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD0_W0)	32	RW	0000_0000h
1_A944h	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD0_W1)	32	RW	0000_0000h
1_A948h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD1_W0)	32	RW	0000_0000h
1_A94Ch	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD1_W1)	32	RW	0000_0000h
1_A950h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD2_W0)	32	RW	0000_0000h
1_A954h	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD2_W1)	32	RW	0000_0000h
1_A958h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD3_W0)	32	RW	0000_0000h
1_A95Ch	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD3_W1)	32	RW	0000_0000h
1_A960h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD4_W0)	32	RW	0000_0000h
1_A964h	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD4_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_A968h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD5_W0)	32	RW	0000_0000h
1_A96Ch	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD5_W1)	32	RW	0000_0000h
1_A970h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD6_W0)	32	RW	0000_0000h
1_A974h	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD6_W1)	32	RW	0000_0000h
1_A978h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD7_W0)	32	RW	0000_0000h
1_A97Ch	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD7_W1)	32	RW	0000_0000h
1_A980h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD8_W0)	32	RW	0000_0000h
1_A984h	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD8_W1)	32	RW	0000_0000h
1_A988h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD9_W0)	32	RW	0000_0000h
1_A98Ch	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD9_W1)	32	RW	0000_0000h
1_A990h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD10_W0)	32	RW	0000_0000h
1_A994h	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD10_W1)	32	RW	0000_0000h
1_A998h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD11_W0)	32	RW	0000_0000h
1_A99Ch	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD11_W1)	32	RW	0000_0000h
1_A9A0h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD12_W0)	32	RW	0000_0000h
1_A9A4h	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD12_W1)	32	RW	0000_0000h
1_A9A8h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD13_W0)	32	RW	0000_0000h
1_A9ACh	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD13_W1)	32	RW	0000_0000h
1_A9B0h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD14_W0)	32	RW	0000_0000h
1_A9B4h	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD14_W1)	32	RW	0000_0000h
1_A9B8h	MRC Region Descriptor Word 0 (MRC6_DOM9_RGD15_W0)	32	RW	0000_0000h
1_A9BCh	MRC Region Descriptor Word 1 (MRC6_DOM9_RGD15_W1)	32	RW	0000_0000h
1_A9C0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM9_RGD_NSE)	32	RW	0000_0000h
1_AA40h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD0_W0)	32	RW	0000_0000h
1_AA44h	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD0_W1)	32	RW	0000_0000h
1_AA48h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD1_W0)	32	RW	0000_0000h
1_AA4Ch	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD1_W1)	32	RW	0000_0000h
1_AA50h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD2_W0)	32	RW	0000_0000h
1_AA54h	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD2_W1)	32	RW	0000_0000h
1_AA58h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD3_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_AA5Ch	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD3_W1)	32	RW	0000_0000h
1_AA60h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD4_W0)	32	RW	0000_0000h
1_AA64h	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD4_W1)	32	RW	0000_0000h
1_AA68h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD5_W0)	32	RW	0000_0000h
1_AA6Ch	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD5_W1)	32	RW	0000_0000h
1_AA70h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD6_W0)	32	RW	0000_0000h
1_AA74h	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD6_W1)	32	RW	0000_0000h
1_AA78h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD7_W0)	32	RW	0000_0000h
1_AA7Ch	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD7_W1)	32	RW	0000_0000h
1_AA80h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD8_W0)	32	RW	0000_0000h
1_AA84h	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD8_W1)	32	RW	0000_0000h
1_AA88h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD9_W0)	32	RW	0000_0000h
1_AA8Ch	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD9_W1)	32	RW	0000_0000h
1_AA90h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD10_W0)	32	RW	0000_0000h
1_AA94h	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD10_W1)	32	RW	0000_0000h
1_AA98h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD11_W0)	32	RW	0000_0000h
1_AA9Ch	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD11_W1)	32	RW	0000_0000h
1_AAA0h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD12_W0)	32	RW	0000_0000h
1_AAA4h	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD12_W1)	32	RW	0000_0000h
1_AAA8h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD13_W0)	32	RW	0000_0000h
1_AAACh	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD13_W1)	32	RW	0000_0000h
1_AAB0h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD14_W0)	32	RW	0000_0000h
1_AAB4h	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD14_W1)	32	RW	0000_0000h
1_AAB8h	MRC Region Descriptor Word 0 (MRC6_DOM10_RGD15_W0)	32	RW	0000_0000h
1_AABCh	MRC Region Descriptor Word 1 (MRC6_DOM10_RGD15_W1)	32	RW	0000_0000h
1_AAC0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM10_RGD_NSE)	32	RW	0000_0000h
1_AB40h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD0_W0)	32	RW	0000_0000h
1_AB44h	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD0_W1)	32	RW	0000_0000h
1_AB48h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD1_W0)	32	RW	0000_0000h
1_AB4Ch	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD1_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_AB50h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD2_W0)	32	RW	0000_0000h
1_AB54h	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD2_W1)	32	RW	0000_0000h
1_AB58h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD3_W0)	32	RW	0000_0000h
1_AB5Ch	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD3_W1)	32	RW	0000_0000h
1_AB60h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD4_W0)	32	RW	0000_0000h
1_AB64h	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD4_W1)	32	RW	0000_0000h
1_AB68h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD5_W0)	32	RW	0000_0000h
1_AB6Ch	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD5_W1)	32	RW	0000_0000h
1_AB70h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD6_W0)	32	RW	0000_0000h
1_AB74h	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD6_W1)	32	RW	0000_0000h
1_AB78h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD7_W0)	32	RW	0000_0000h
1_AB7Ch	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD7_W1)	32	RW	0000_0000h
1_AB80h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD8_W0)	32	RW	0000_0000h
1_AB84h	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD8_W1)	32	RW	0000_0000h
1_AB88h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD9_W0)	32	RW	0000_0000h
1_AB8Ch	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD9_W1)	32	RW	0000_0000h
1_AB90h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD10_W0)	32	RW	0000_0000h
1_AB94h	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD10_W1)	32	RW	0000_0000h
1_AB98h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD11_W0)	32	RW	0000_0000h
1_AB9Ch	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD11_W1)	32	RW	0000_0000h
1_ABA0h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD12_W0)	32	RW	0000_0000h
1_ABA4h	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD12_W1)	32	RW	0000_0000h
1_ABA8h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD13_W0)	32	RW	0000_0000h
1_ABACh	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD13_W1)	32	RW	0000_0000h
1_ABB0h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD14_W0)	32	RW	0000_0000h
1_ABB4h	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD14_W1)	32	RW	0000_0000h
1_ABB8h	MRC Region Descriptor Word 0 (MRC6_DOM11_RGD15_W0)	32	RW	0000_0000h
1_ABBCh	MRC Region Descriptor Word 1 (MRC6_DOM11_RGD15_W1)	32	RW	0000_0000h
1_ABC0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM11_RGD_NSE)	32	RW	0000_0000h
1_AC40h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD0_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_AC44h	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD0_W1)	32	RW	0000_0000h
1_AC48h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD1_W0)	32	RW	0000_0000h
1_AC4Ch	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD1_W1)	32	RW	0000_0000h
1_AC50h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD2_W0)	32	RW	0000_0000h
1_AC54h	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD2_W1)	32	RW	0000_0000h
1_AC58h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD3_W0)	32	RW	0000_0000h
1_AC5Ch	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD3_W1)	32	RW	0000_0000h
1_AC60h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD4_W0)	32	RW	0000_0000h
1_AC64h	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD4_W1)	32	RW	0000_0000h
1_AC68h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD5_W0)	32	RW	0000_0000h
1_AC6Ch	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD5_W1)	32	RW	0000_0000h
1_AC70h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD6_W0)	32	RW	0000_0000h
1_AC74h	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD6_W1)	32	RW	0000_0000h
1_AC78h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD7_W0)	32	RW	0000_0000h
1_AC7Ch	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD7_W1)	32	RW	0000_0000h
1_AC80h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD8_W0)	32	RW	0000_0000h
1_AC84h	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD8_W1)	32	RW	0000_0000h
1_AC88h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD9_W0)	32	RW	0000_0000h
1_AC8Ch	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD9_W1)	32	RW	0000_0000h
1_AC90h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD10_W0)	32	RW	0000_0000h
1_AC94h	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD10_W1)	32	RW	0000_0000h
1_AC98h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD11_W0)	32	RW	0000_0000h
1_AC9Ch	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD11_W1)	32	RW	0000_0000h
1_ACA0h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD12_W0)	32	RW	0000_0000h
1_ACA4h	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD12_W1)	32	RW	0000_0000h
1_ACA8h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD13_W0)	32	RW	0000_0000h
1_ACACh	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD13_W1)	32	RW	0000_0000h
1_ACB0h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD14_W0)	32	RW	0000_0000h
1_ACB4h	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD14_W1)	32	RW	0000_0000h
1_ACB8h	MRC Region Descriptor Word 0 (MRC6_DOM12_RGD15_W0)	32	RW	0000_0000h
1_ACBCh	MRC Region Descriptor Word 1 (MRC6_DOM12_RGD15_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_ACC0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM12_RGD_NSE)	32	RW	0000_0000h
1_AD40h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD0_W0)	32	RW	0000_0000h
1_AD44h	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD0_W1)	32	RW	0000_0000h
1_AD48h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD1_W0)	32	RW	0000_0000h
1_AD4Ch	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD1_W1)	32	RW	0000_0000h
1_AD50h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD2_W0)	32	RW	0000_0000h
1_AD54h	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD2_W1)	32	RW	0000_0000h
1_AD58h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD3_W0)	32	RW	0000_0000h
1_AD5Ch	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD3_W1)	32	RW	0000_0000h
1_AD60h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD4_W0)	32	RW	0000_0000h
1_AD64h	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD4_W1)	32	RW	0000_0000h
1_AD68h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD5_W0)	32	RW	0000_0000h
1_AD6Ch	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD5_W1)	32	RW	0000_0000h
1_AD70h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD6_W0)	32	RW	0000_0000h
1_AD74h	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD6_W1)	32	RW	0000_0000h
1_AD78h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD7_W0)	32	RW	0000_0000h
1_AD7Ch	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD7_W1)	32	RW	0000_0000h
1_AD80h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD8_W0)	32	RW	0000_0000h
1_AD84h	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD8_W1)	32	RW	0000_0000h
1_AD88h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD9_W0)	32	RW	0000_0000h
1_AD8Ch	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD9_W1)	32	RW	0000_0000h
1_AD90h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD10_W0)	32	RW	0000_0000h
1_AD94h	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD10_W1)	32	RW	0000_0000h
1_AD98h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD11_W0)	32	RW	0000_0000h
1_AD9Ch	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD11_W1)	32	RW	0000_0000h
1_ADA0h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD12_W0)	32	RW	0000_0000h
1_ADA4h	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD12_W1)	32	RW	0000_0000h
1_ADA8h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD13_W0)	32	RW	0000_0000h
1_ADACh	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD13_W1)	32	RW	0000_0000h
1_ADB0h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD14_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_ADB4h	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD14_W1)	32	RW	0000_0000h
1_ADB8h	MRC Region Descriptor Word 0 (MRC6_DOM13_RGD15_W0)	32	RW	0000_0000h
1_ADBCh	MRC Region Descriptor Word 1 (MRC6_DOM13_RGD15_W1)	32	RW	0000_0000h
1_ADC0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM13_RGD_NSE)	32	RW	0000_0000h
1_AE40h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD0_W0)	32	RW	0000_0000h
1_AE44h	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD0_W1)	32	RW	0000_0000h
1_AE48h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD1_W0)	32	RW	0000_0000h
1_AE4Ch	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD1_W1)	32	RW	0000_0000h
1_AE50h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD2_W0)	32	RW	0000_0000h
1_AE54h	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD2_W1)	32	RW	0000_0000h
1_AE58h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD3_W0)	32	RW	0000_0000h
1_AE5Ch	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD3_W1)	32	RW	0000_0000h
1_AE60h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD4_W0)	32	RW	0000_0000h
1_AE64h	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD4_W1)	32	RW	0000_0000h
1_AE68h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD5_W0)	32	RW	0000_0000h
1_AE6Ch	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD5_W1)	32	RW	0000_0000h
1_AE70h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD6_W0)	32	RW	0000_0000h
1_AE74h	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD6_W1)	32	RW	0000_0000h
1_AE78h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD7_W0)	32	RW	0000_0000h
1_AE7Ch	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD7_W1)	32	RW	0000_0000h
1_AE80h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD8_W0)	32	RW	0000_0000h
1_AE84h	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD8_W1)	32	RW	0000_0000h
1_AE88h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD9_W0)	32	RW	0000_0000h
1_AE8Ch	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD9_W1)	32	RW	0000_0000h
1_AE90h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD10_W0)	32	RW	0000_0000h
1_AE94h	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD10_W1)	32	RW	0000_0000h
1_AE98h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD11_W0)	32	RW	0000_0000h
1_AE9Ch	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD11_W1)	32	RW	0000_0000h
1_AEA0h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD12_W0)	32	RW	0000_0000h
1_AEA4h	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD12_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_AEA8h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD13_W0)	32	RW	0000_0000h
1_AEACh	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD13_W1)	32	RW	0000_0000h
1_AEB0h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD14_W0)	32	RW	0000_0000h
1_AEB4h	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD14_W1)	32	RW	0000_0000h
1_AEB8h	MRC Region Descriptor Word 0 (MRC6_DOM14_RGD15_W0)	32	RW	0000_0000h
1_AEBCh	MRC Region Descriptor Word 1 (MRC6_DOM14_RGD15_W1)	32	RW	0000_0000h
1_AEC0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM14_RGD_NSE)	32	RW	0000_0000h
1_AF40h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD0_W0)	32	RW	0000_0000h
1_AF44h	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD0_W1)	32	RW	0000_0000h
1_AF48h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD1_W0)	32	RW	0000_0000h
1_AF4Ch	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD1_W1)	32	RW	0000_0000h
1_AF50h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD2_W0)	32	RW	0000_0000h
1_AF54h	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD2_W1)	32	RW	0000_0000h
1_AF58h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD3_W0)	32	RW	0000_0000h
1_AF5Ch	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD3_W1)	32	RW	0000_0000h
1_AF60h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD4_W0)	32	RW	0000_0000h
1_AF64h	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD4_W1)	32	RW	0000_0000h
1_AF68h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD5_W0)	32	RW	0000_0000h
1_AF6Ch	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD5_W1)	32	RW	0000_0000h
1_AF70h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD6_W0)	32	RW	0000_0000h
1_AF74h	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD6_W1)	32	RW	0000_0000h
1_AF78h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD7_W0)	32	RW	0000_0000h
1_AF7Ch	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD7_W1)	32	RW	0000_0000h
1_AF80h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD8_W0)	32	RW	0000_0000h
1_AF84h	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD8_W1)	32	RW	0000_0000h
1_AF88h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD9_W0)	32	RW	0000_0000h
1_AF8Ch	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD9_W1)	32	RW	0000_0000h
1_AF90h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD10_W0)	32	RW	0000_0000h
1_AF94h	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD10_W1)	32	RW	0000_0000h
1_AF98h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD11_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1_AF9Ch	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD11_W1)	32	RW	0000_0000h
1_AFA0h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD12_W0)	32	RW	0000_0000h
1_AFA4h	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD12_W1)	32	RW	0000_0000h
1_AFA8h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD13_W0)	32	RW	0000_0000h
1_AFACh	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD13_W1)	32	RW	0000_0000h
1_AFB0h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD14_W0)	32	RW	0000_0000h
1_AFB4h	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD14_W1)	32	RW	0000_0000h
1_AFB8h	MRC Region Descriptor Word 0 (MRC6_DOM15_RGD15_W0)	32	RW	0000_0000h
1_AFBCh	MRC Region Descriptor Word 1 (MRC6_DOM15_RGD15_W1)	32	RW	0000_0000h
1_AFC0h	MRC Region Descriptor NonSecure Enable (MRC6_DOM15_RGD_NSE)	32	RW	0000_0000h

43.8.2.2 TRDC Register (TRDC_CR)

Offset

Register	Offset
TRDC_CR	0h

Function

This register provides status about the TRDC and global enable bits for the entire module's operation. A fully operational TRDC requires all global bits GVLD {R,B,M} to be asserted. Undefined behavior results if they are not all asserted. If there are no region checkers (MRCs) present in the configuration, GVLDR need not be asserted for a fully operational TRDC.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	LK1	0										0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GVLDR	GVLDB	0						0				HRL			GVLDM
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 LK1	Lock Status This field is the lock status of the TRDC_CR. This field is set when all GVLD bits are set; GVLDR=GVLDB=GVLDM=1b1. Once set, this bit remains asserted until the next reset. 0b - The CR can be written by any secure privileged write. 1b - The CR is locked (read-only) until the next reset.
29-20 —	Reserved
19-16 —	Reserved
15 GVLDR	Global Valid for Memory Region Checkers TRDC global MRC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC MRCs are disabled. 1b - TRDC MRCs are enabled.
14 GVLDB	Global Valid for Memory Block Checkers TRDC global MBC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC MBCs are disabled. 1b - TRDC MBCs are enabled.
13-9 —	Reserved
8-5 —	Reserved
4-1 HRL	Hardware Revision Level This read-only field specifies the TRDC’s hardware and definition revision level. It can be read by software to determine the functional definition of the module.
0 GVLDM	Global Valid for Domain Assignment Controllers TRDC global DAC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC DACs are disabled. 1b - TRDC DACs are enabled.

43.8.2.3 TRDC Hardware Configuration Register 0 (TRDC_HWCFG0)

Offset

Register	Offset
TRDC_HWCFG0	F0h

Function

This read-only register contains information on the TRDC's hardware configuration. Specifically, it defines the number of implemented domains and bus masters along with the number of instances of memory block checkers (MBCs) and memory region checkers (MRCs). The register value at reset is device-specific. Attempted writes are error terminated.

Access: f TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MID				NMRC				0				NMBC			
W																
Reset	0	0	1	0	0	1	1	1	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NMSTR								0				NDID			
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-29 MID	Module ID This field defines major version ID of this module.
28-24 NMRC	Number of MRCs This field defines the number of MRCs on the device [1-16].
23-20 —	Reserved
19-16 NMBC	Number of MBCs This field defines the number of MBCs on the device [1-8].
15-8 NMSTR	Number of bus masters This read-only field defines the number of bus masters.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-5 —	Reserved
4-0 NDID	Number of domains This read-only field defines the number of domains on the device [1-16].

43.8.2.4 TRDC Hardware Configuration Register 1 (TRDC_HWCFG1)

Offset

Register	Offset
TRDC_HWCFG1	F4h

Function

This register contains information on the TRDC's hardware configuration. It provides a mechanism for software to determine its domain number by simply reading the register. See Domain error capture management for more details on typical usage. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												DID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	u	u	u	u

Fields

Field	Function
31-4 —	Reserved
3-0 DID	Domain identifier number This field provides the domain number [0-15] of the requesting bus master.

43.8.2.5 TRDC Hardware Configuration Register 2 (TRDC_HWCFG2)

Offset

Register	Offset
TRDC_HWCFG2	F8h

Function

This register contains information on the TRDC's hardware configuration. It provides a bitmap signaling the presence of a process identifier (PID) register sourced from the given bus master. The HWCFG2 register is associated with bus masters [31-0]. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PIDPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PIDPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1

Fields

Field	Function
31-0 PIDPn	Process identifier present Process identifier present from bus master n, where n = [31-0]. This field provides a bitmap to signal that bus master (where n is defined as bit number) sources a process identifier register to the TRDC_DAC logic. For each bit in this field: • 0 - Bus master n does not source a process identifier register. The TRDC_DAC logic provides the needed PID for processor cores. • 1 - Bus master n sources a process identifier register.

43.8.2.6 TRDC Hardware Configuration Register 3 (TRDC_HWCFG3)

Offset

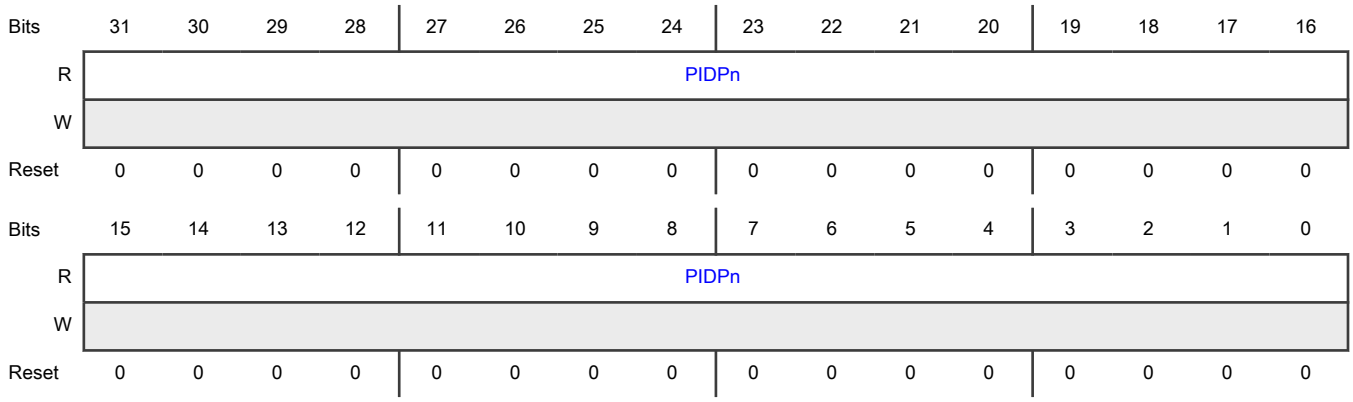
Register	Offset
TRDC_HWCFG3	FCh

Function

This register contains information on the TRDC's hardware configuration. Specifically, it provides a bitmap signaling the presence of a process identifier (PID) register sourced from the given bus master. The HWCFG3 register is associated with bus masters [63-32], while the HWCFG2 register covers bus masters [31-0]. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 PIDPn	Process identifier present Process identifier present from bus master n, where n = [63-32]. This field provides a bitmap to signal that bus master n (where n is defined as 32 + bit number) sources a process identifier register to the TRDC_DAC logic. Process identifier present from bus master n, where n = [31-0]. This field provides a bitmap to signal that bus master (where n is defined as bit number) sources a process identifier register to the TRDC_DAC logic. For each bit in this field: • 0 - Bus master n does not source a process identifier register. The TRDC_DAC logic provides the needed PID for processor cores. • 1 - Bus master n sources a process identifier register.

43.8.2.7 Domain Assignment Configuration Register (DACFG0 - DACFG5)

Offset

Register	Offset
DACFG0	100h
DACFG1	101h
DACFG2	102h
DACFG3	103h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
DACFG4	104h
DACFG5	105h

Function

This register defines the number of implemented domain assignment registers for bus master m, where m+1 can specify from 1 to 64 bus masters. These registers are organized as a byte-sized data array and can be read using 8-, 16- or 32-bit accesses. An all-zero value (NCM = 0, NMDAR = 0) indicates a non-existent bus master. Attempted writes are error terminated.

Register read will not return transfer error when DACFG for a master doesn't exist but it is in the same 32-bit group as DACFG for an existing master. For example, when there are only 2 DACFG0-1 registers for 2 master, and masters for DACFG2-3 don't exist, then access to DACFG2-3 won't return transfer error.

Typically, processor bus masters have one or more domain assignment registers, while non-processor masters have a single domain assignment register.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	7	6	5	4	3	2	1	0
R	NCM	0			NMDAR			
W								
Reset	See Register reset values.							

Register reset values

Register	Reset value
DACFG0	02h
DACFG1	03h
DACFG2–DACFG4	81h
DACFG5	07h

Fields

Field	Function
7 NCM	Non-CPU Master This read-only field signals that bus master m is a non-CPU master. It specifies that the format of the associated MDA_Wr_m register defines a non-processor domain assignment. This field is zero for a non-existent bus master. 0b - Bus master is a processor. 1b - Bus master is a non-processor.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-4 —	Reserved
3-0 NMDAR	Number of master domain assignment registers for bus master m This read-only field specifies the number of registers associated with the master domain assignment register for a given bus master. The value is limited to the range [0-8], where zero indicates a non-existent bus master and non-zero values indicate the number of implemented registers associated with this MDAm.

43.8.2.8 TRDC IDAU Control Register (TRDC_IDAU_CR)

Offset

Register	Offset
TRDC_IDAU_CR	1C0h

Function

This register defines the configuration for the Implementation-Defined Attribution Unit which is tightly-coupled to the TZ-M extensions in the processor core. Many of the fields in this register explicitly control processor TZ-M functions.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PCURRNS	0												0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			LKSAU	LKNSMPU	LKSMPU	LKNSVTOR	LKSVTAI...	0	SAUDIS	MPUNSDIS	MPUSDIS	CFGSECE...	0		VLD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31 PCURRNS	Processor current security Current security state of the processor.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Processor is in Secure state 1b - Processor is in Nonsecure state
30-19 —	Reserved
18-16 —	Reserved
15-13 —	Reserved
12 LKSAU	Lock SAU Asserting this bit prevents changes to the processor's Secure SAU memory regions already programmed. All processor writes to the registers are ignored. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers 1b - Disable writes to the SAU_CTRL, SAU_RNR, SAU_RBAR and SAU_RLAR registers from software or from a debug agent connected to the processor
11 LKNSMPU	Lock Nonsecure MPU Asserting this bit prevents changes to the processor's Nonsecure MPU memory regions already programmed. All processor writes to the registers are ignored. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers 1b - Disable writes to the MPU_CTRL_NS, MPU_RNR_NS, MPU_RBAR_NS, MPU_RLAR_NS, MPU_RBAR_A_NSn and MPU_RLAR_A_NSn from software or from a debug agent connected to the processor
10 LKSMPU	Lock Secure MPU Asserting this signal prevents changes to the processor's programmed Secure MPU memory regions and all processor writes to the registers are ignored. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers 1b - Disable writes to the MPU_CTRL, MPU_RNR, MPU_RBAR, MPU_RLAR, MPU_RBAR_An and MPU_RLAR_An from software or from a debug agent connected to the processor in Secure state
9 LKNSVTOR	Lock Nonsecure Vector Table Offset Register Asserting this signal prevents changes to the processor's Nonsecure vector table base address. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock this register 1b - Disable writes to the VTOR_NS register

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 LKSVTAIRCR	Lock Secure VTOR, Application interrupt and Reset Control Registers This bit is sticky, once set, it can only be cleared with a reset. Asserting this signal prevents processor changes to: • The Secure vector table base address. • Handling of Secure interrupt priority. • BusFault, HardFault, and NMI security target settings in the processor. 0b - Unlock these registers 1b - Disable writes to the VTOR_S, AIRCR[PRIS], and AIRCR[BFHFNMINS] registers
7 —	Reserved
6 SAUDIS	Security Attribution Unit Disable 0b - SAU is enabled 1b - SAU is disabled
5 MPUNSDIS	NonSecure Memory Protection Unit Disabled 0b - Nonsecure MPU is enabled 1b - Nonsecure MPU is disabled
4 MPUSDIS	Secure Memory Protection Unit Disabled 0b - Secure MPU is enabled 1b - Secure MPU is disabled
3 CFGSECEXT	Configure Security Extension 0b - Armv8M Security Extension is disabled 1b - Armv8-M Security Extension is enabled
2-1 —	Reserved
0 VLD	Valid When VLD =0, all address attribute from IDAU is nonSecure. When VLD=1, values in other fields of this register are valid.

43.8.2.9 TRDC FLW Control (TRDC_FLW_CTL)

Offset

Register	Offset
TRDC_FLW_CTL	1E0h

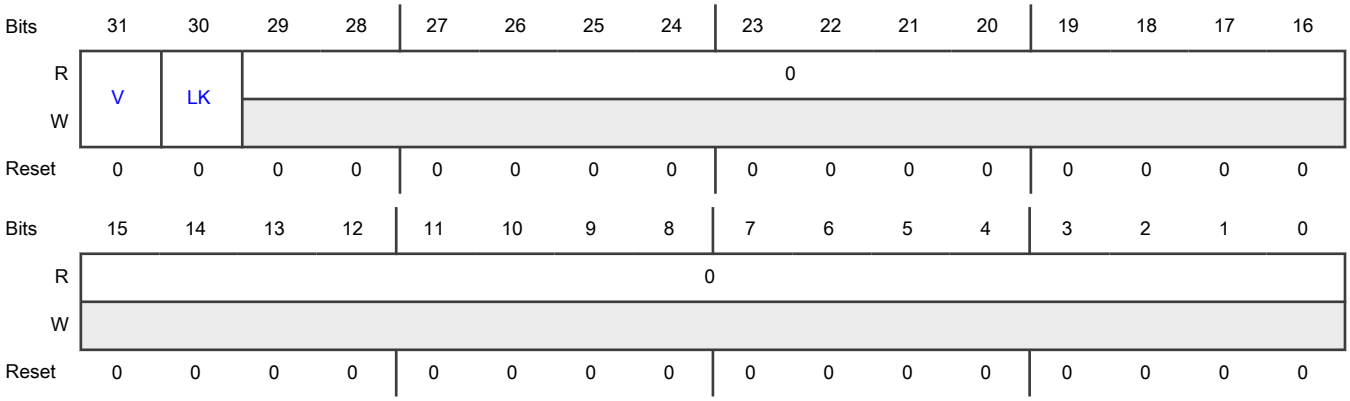
Function

This register provides control of the FLW = Flash Logical Window operation.

This register exists for this device but has no effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31 V	Valid bit 0b - FLW function is disabled. 1b - FLW function is enabled.
30 LK	Lock bit 0b - FLW registers may be modified. 1b - FLW registers are locked until the next reset.
29-0 —	Reserved

43.8.2.10 TRDC FLW Physical Base (TRDC_FLW_PBASE)

Offset

Register	Offset
TRDC_FLW_PBASE	1E4h

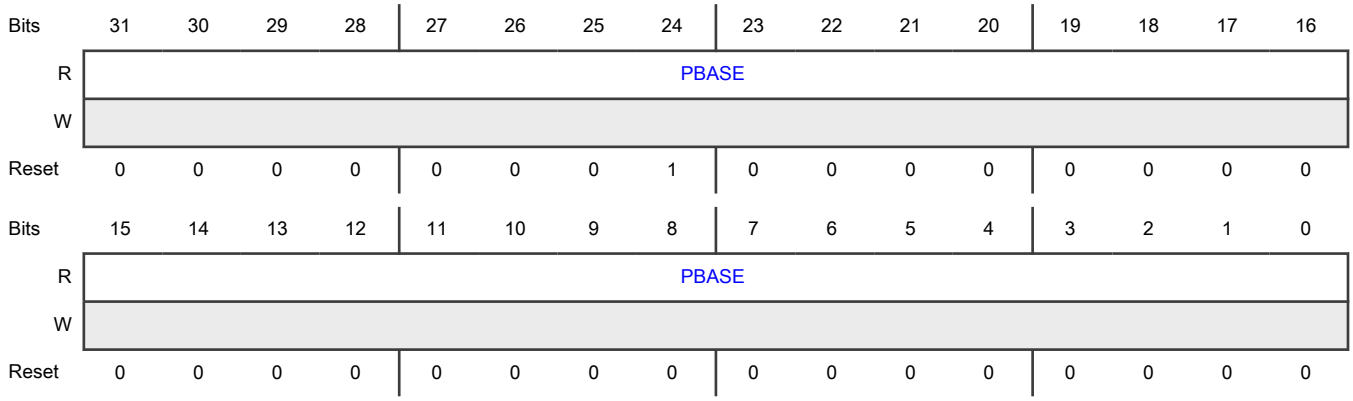
Function

This read-only register gives the physical base address of the Flash Logical Window.

This register exists for this device but has not effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 PBASE	Physical base address Physical address of the base of the FLW (mod 32KBytes).

43.8.2.11 TRDC FLW Array Base (TRDC_FLW_ABASE)

Offset

Register	Offset
TRDC_FLW_ABASE	1E8h

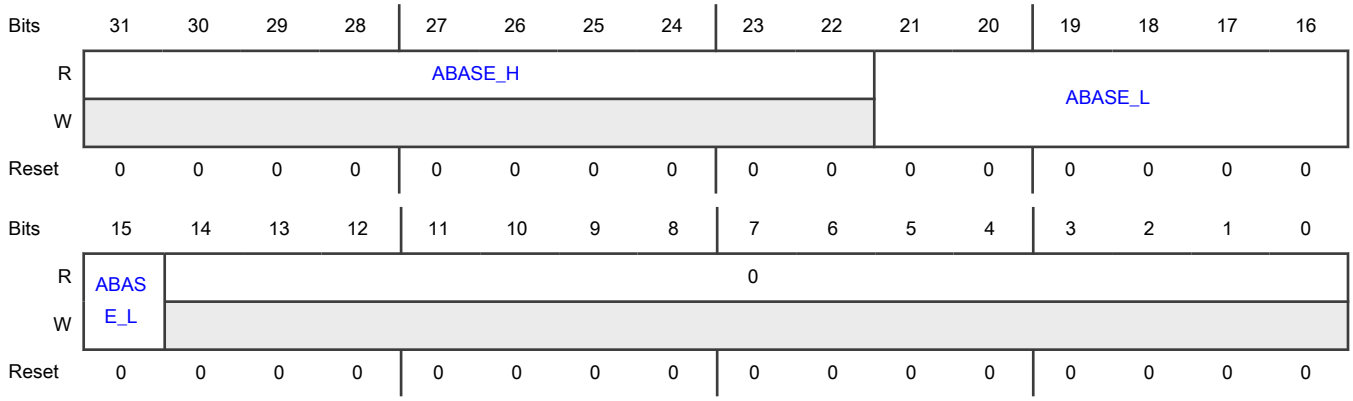
Function

This register gives the flash array base address of the Flash Logical Window.

This register exists for this device but has no effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write. This register is not writable if TRDC_FLW_CTL[LK]=1.

Diagram



Fields

Field	Function
31-22 ABASE_H	Array base address high Flash array address upper bits of the base of the FLW (mod 32KBytes).
21-15 ABASE_L	Array base address low Flash array address lower bits of the base of the FLW (mod 32KBytes).
14-0 —	Reserved

43.8.2.12 TRDC FLW Block Count (TRDC_FLW_BCNT)

Offset

Register	Offset
TRDC_FLW_BCNT	1ECh

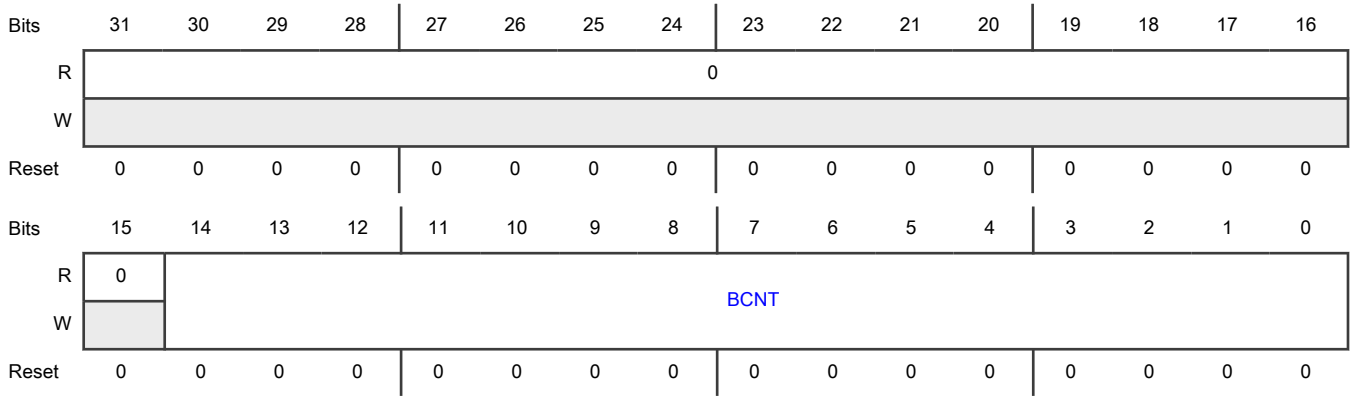
Function

This register gives the size of the Flash Logic Window in 32KByte blocks.

This register exists for this device but has no effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write. This register is not writable if TRDC_FLW_CTL[LK]=1.

Diagram



Fields

Field	Function
31-15	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-0 BCNT	Block Count Size of FLW in number of 32KByte blocks.

43.8.2.13 TRDC Fault Domain ID (TRDC_FDID)

Offset

Register	Offset
TRDC_FDID	1FCh

Function

In the event of an access error, this register is used to specify the domainID of the faulting reference before indexing into the Domain Error registers.

If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												FDID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 FDID	Domain ID of Faulted Access This field indicates the domainID of the fault used to index the array of domain error information. The user queries the DERRLOCx registers to find the DomainID of the faulting access and must write this register with the domain ID before reading the Domain Error registers.

43.8.2.14 TRDC Domain Error Location Register (TRDC_DERRLOC0 - TRDC_DERRLOC15)

Offset

For d = 0 to 15:

Register	Offset
TRDC_DERRLOCd	200h + (d × 4h)

Function

This array of read-only registers provide the instance number of the submodule where (an) access violation(s) occurred. These registers are organized as a word array, indexed by the faulting domain number, d. The two fields of this register provide a bitmap of instances associated with all submodules containing captured error information for that domain. These instance numbers are then used as indices into the DERR_W0_*i*, DERR_W1_*i*, and DERR_W3_*i*register arrays. See Domain error capture management for more details.

When an access violation is detected by either a Memory Region Checker (MRC) or a Memory Block Checker (MBC), address and attribute information of the offending access is captured. Using the faulting domainID number as the index, d, this array of read-only registers provide additional information signaling the instance number of the submodule where the access violation(s) occurred. Since the resulting exception handler needs the submodule instance to retrieve the captured address and attribute information from DERR_W0_*i* and DERR_W1_*i*, these registers provide the instance number details.

Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MRCINST															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								MBCINST							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 MRCINST	MRC instance This field is a bitmap indicating the presence of a detected access violation for domain d in the implemented instances of the MRC. The least-significant bit of this field (bit 16) corresponds to MRC instance 0. The most-significant bit of this field (bit 31) corresponds to MRC instance 15 (MRC instance = i-16 where i is the register bit number), and so on. Multiple bits can be set at any time indicating access violations for this domain have been detected across multiple instances of the MRCs.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For each bit in this field: • 0 - The memory region checker has not detected an access violation or is not physically present. • 1 - The memory region checker has detected one or more access violations for this domain.
15-8 —	Reserved
7-0 MBCINST	MBC instance This field is a bitmap indicating the presence of a detected access violation for domain d in the implemented instances of the MBC. The least-significant bit of this field (bit 0) corresponds to MBC instance 0. The most-significant bit of this field (bit 7) corresponds to MBC instance 7 Multiple bits can be set at any time indicating access violations for this domain have been detected across multiple instances of the MBCs. For each bit in this field: • 0 - The memory block checker has not detected an access violation or is not physically present. • 1 - The memory block checker has detected one or more access violations for this domain.

43.8.2.15 MBC Domain Error Word0 Register (MBC0_DERR_W0 - MBC1_DERR_W0)

Offset

Register	Offset
MBC0_DERR_W0	400h
MBC1_DERR_W0	410h

Function

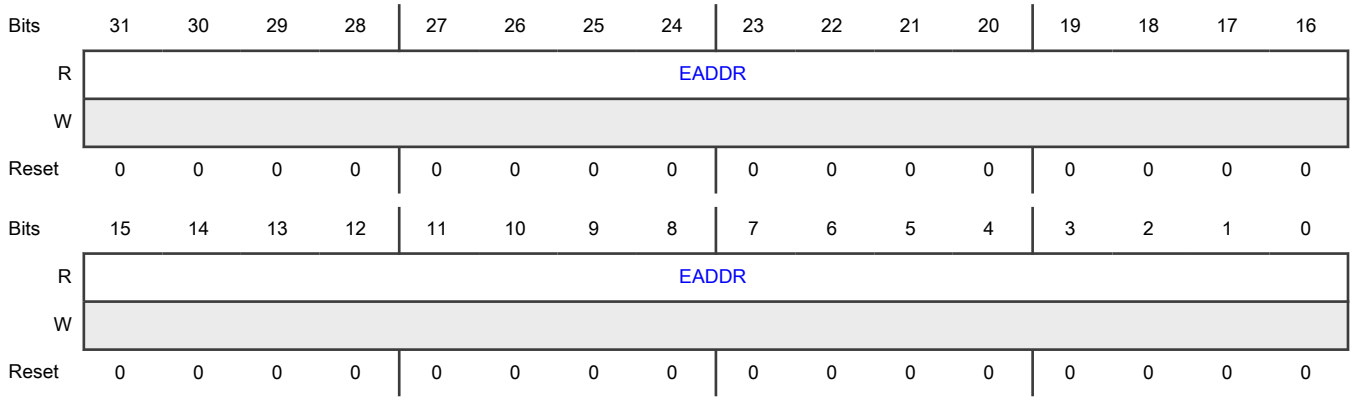
This read-only register array provides the address of an access violation detected by memory block checker (MBC). These registers are organized as a word array, which is indexed by the MBC instance number. That is, the index, i, of this array is the instance number of MBC with the access violation. The submodule instance numbers are provided by the DERRLOC registers. The memory mapped error capture detail registers are organized as 24 sequential 16 byte entries.

When an access violation is detected and the offending information captured, subsequent updates to this register are disabled until the required data pattern is written to the DERR_W3_i register. At that time, this register is cleared and re-enabled to capture the next access violation.

Attempted writes are error terminated as are attempted reads of an MBC instance that is not physically present.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 EADDR	Error address This is the unaliased virtual address of the access that generated the access violation.

43.8.2.16 MBC Domain Error Word1 Register (MBC0_DERR_W1 - MBC1_DERR_W1)

Offset

Register	Offset
MBC0_DERR_W1	404h
MBC1_DERR_W1	414h

Function

This read-only register array provides the attributes of an access violation detected by Memory Block Checker (MBC). These registers are organized as a word array, which is indexed by the violating submodule instance number. Refer to register Domain Error Location(DERRLOCd), Domain Error Word0 Register(DERR_W0_i), and Domain error capture management for more information.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EST		0		EPORT				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				ERW	EATR			0				EDID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 EST	Error state This field signals the state of access violations for this domain in this instance of the block checker or region checker. Once an access violation has been detected and the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. After retrieving the faulting address and attributes, the error capture mechanism must be rearmed by performing a write to DERR_W3_i. 00b - No access violation has been detected. 01b - No access violation has been detected. 10b - A single access violation has been detected. 11b - Multiple access violations for this domain have been detected by this submodule instance. Only the address and attribute information for the first error have been captured in DERR_W0_i and DERR_W1_i.
29-27 —	Reserved
26-24 EPORT	Error port This field identifies the encoded port number of the MBC that detected the access violation. The MBC port number connection is device-specific. See the chip configuration details for more information. This is the MBC slave that has the violation. 000b - mbcxslv0 001b - mbcxslv1 010b - mbcxslv2 011b - mbcxslv3
23-12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 ERW	Error read/write This field signals whether the captured access violation occurred on a read or write reference. 0b - Read access 1b - Write access
10-8 EATR	Error attributes This field captures attributes of the access violation. 000b - Secure user mode, instruction fetch access. 001b - Secure user mode, data access. 010b - Secure privileged mode, instruction fetch access. 011b - Secure privileged mode, data access. 100b - Nonsecure user mode, instruction fetch access. 101b - Nonsecure user mode, data access. 110b - Nonsecure privileged mode, instruction fetch access. 111b - Nonsecure privileged mode, data access.
7-4 —	Reserved
3-0 EDID	Error domain identifier This field captures the domain identifier of the access violation.

43.8.2.17 MBC Domain Error Word3 Register (MBC0_DERR_W3 - MBC1_DERR_W3)

Offset

Register	Offset
MBC0_DERR_W3	40Ch
MBC1_DERR_W3	41Ch

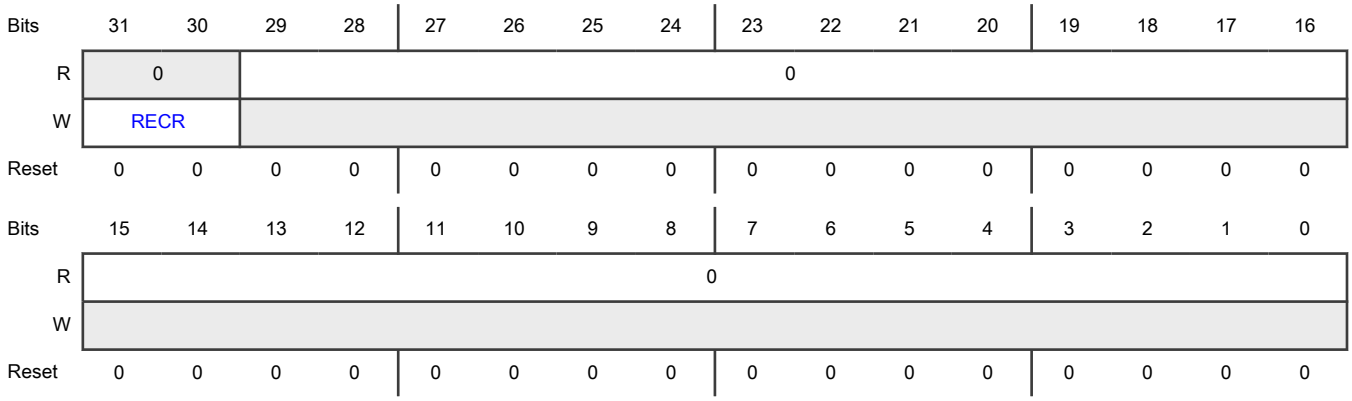
Function

This register is used to rearm the error capture logic and clear the DERR_W0_i and DERR_W1_i registers. After the domain access violation error details have been read, typically in an exception service routine, a 32-bit word write to this register is required to rearm the error capture logic.

A read of this location returns zeroes. Attempted reads of a Memory Region Checker (MRC) or Memory Block Checker (MBC) instance that is not physically present are error terminated.

See Domain error capture management for more details.

Diagram



Fields

Field	Function
31-30 RECR	Rearm Error Capture Registers This 2-bit, write-only field controls the rearming of the domain error capture registers. Once an access violation has been detected with the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. Writing 01b to this field rearms the error capture mechanism and clears the DERR_W0_i and DERR_W1_i registers. A write of any value other than 01b has no effect.
29-0 —	Reserved

43.8.2.18 MRC Domain Error Word0 Register (MRC0_DERR_W0 - MRC6_DERR_W0)

Offset

Register	Offset
MRC0_DERR_W0	480h
MRC1_DERR_W0	490h
MRC2_DERR_W0	4A0h
MRC3_DERR_W0	4B0h
MRC4_DERR_W0	4C0h
MRC5_DERR_W0	4D0h
MRC6_DERR_W0	4E0h

Function

This read-only register array provides the address of an access violation detected by memory region checker (MRC). These registers are organized as a word array, which is indexed by the MRC instance number. That is, the index, i, of this array is the

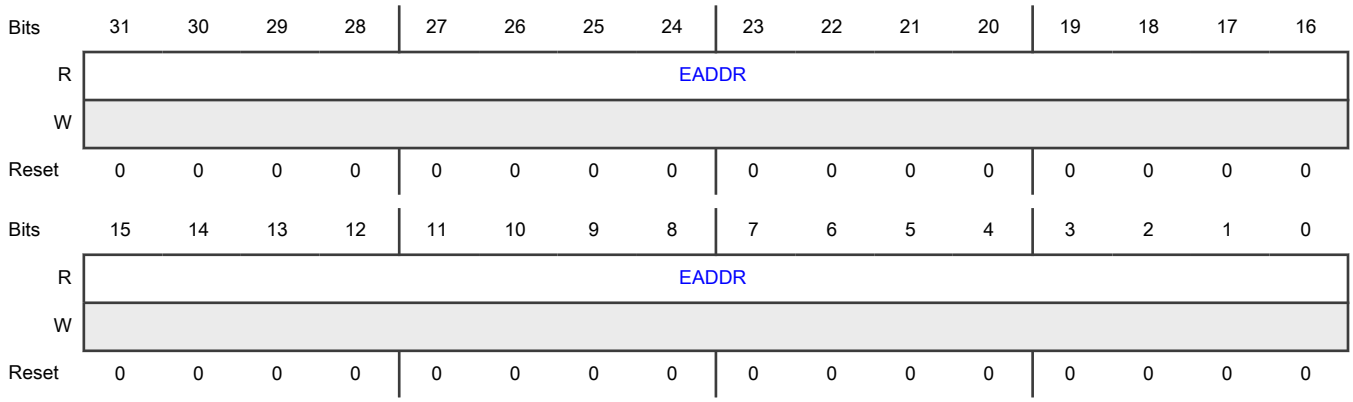
instance number of MRC with the access violation. The submodule instance numbers are provided by the DERRLOC registers. The memory mapped error capture detail registers are organized as 24 sequential 16 byte entries.

When an access violation is detected and the offending information captured, subsequent updates to this register are disabled until the required data pattern is written to the DERR_W3_i register. At that time, this register is cleared and re-enabled to capture the next access violation.

Attempted writes are error terminated as are attempted reads of an MRC instance that is not physically present.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 EADDR	Error address This is the unaliased virtual address of the access that generated the access violation.

43.8.2.19 MRC Domain Error Word1 Register (MRC0_DERR_W1 - MRC6_DERR_W1)

Offset

Register	Offset
MRC0_DERR_W1	484h
MRC1_DERR_W1	494h
MRC2_DERR_W1	4A4h
MRC3_DERR_W1	4B4h
MRC4_DERR_W1	4C4h
MRC5_DERR_W1	4D4h
MRC6_DERR_W1	4E4h

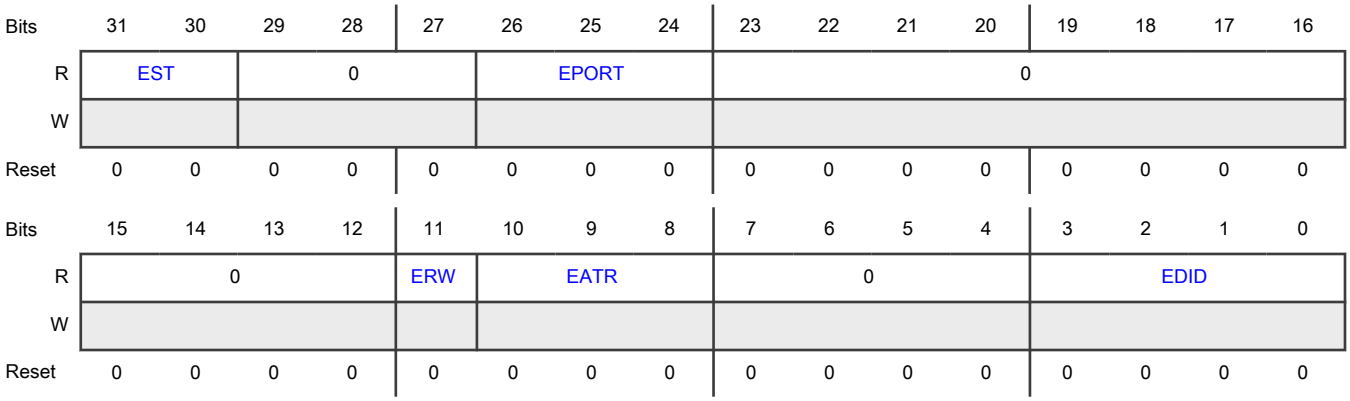
Function

This read-only register array provides the attributes of an access violation detected by memory region checker (MRC). These registers are organized as a word array, which is indexed by the violating submodule instance number. Refer to register

Domain Error Location(DERRLOCd), Domain Error Word0 Register(DERR_W0_i), and Domain error capture management for more information.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-30 EST	Error state This field signals the state of access violations for this domain in this instance of the block checker or region checker. Once an access violation has been detected and the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. After retrieving the faulting address and attributes, the error capture mechanism must be rearmed by performing a write to DERR_W3_i. 00b - No access violation has been detected. 01b - No access violation has been detected. 10b - A single access violation has been detected. 11b - Multiple access violations for this domain have been detected by this submodule instance. Only the address and attribute information for the first error have been captured in DERR_W0_i and DERR_W1_i.
29-27 —	Reserved
26-24 EPORT	Error port This field identifies the encoded port number of the MRC that detected the access violation. The MRC port number connection is device-specific. See the chip configuration details for more information.
23-12 —	Reserved
11	Error read/write This field signals whether the captured access violation occurred on a read or write reference.

Table continues on the next page...

Table continued from the previous page...

Field	Function
ERW	0b - Read access 1b - Write access
10-8 EATR	Error attributes This field captures certain attributes of the access violation. 000b - Secure user mode, instruction fetch access. 001b - Secure user mode, data access. 010b - Secure privileged mode, instruction fetch access. 011b - Secure privileged mode, data access. 100b - Nonsecure user mode, instruction fetch access. 101b - Nonsecure user mode, data access. 110b - Nonsecure privileged mode, instruction fetch access. 111b - Nonsecure privileged mode, data access.
7-4 —	Reserved
3-0 EDID	Error domain identifier This field captures the domain identifier of the access violation.

43.8.2.20 MRC Domain Error Word3 Register (MRC0_DERR_W3 - MRC6_DERR_W3)

Offset

Register	Offset
MRC0_DERR_W3	48Ch
MRC1_DERR_W3	49Ch
MRC2_DERR_W3	4ACh
MRC3_DERR_W3	4BCh
MRC4_DERR_W3	4CCh
MRC5_DERR_W3	4DCh
MRC6_DERR_W3	4ECh

Function

This register is used to rearm the error capture logic and clear the DERR_W0_i and DERR_W1_i registers. After the domain access violation error details have been read, typically in an exception service routine, a 32-bit word write to this register is required to rearm the error capture logic.

A read of this location returns zeroes. Attempted reads of a memory region checker (MRC) or Memory Block Checker (MBC) instance that is not physically present are error terminated.

See Domain error capture management for more details.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0													
W	RECR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 RECR	Rearm Error Capture Registers This 2-bit, write-only field controls the rearming of the domain error capture registers. Once an access violation has been detected with the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. Writing 01b to this field rearms the error capture mechanism and clears the DERR_W0_i and DERR_W1_i registers. A write of any value other than 01b has no effect.
29-0 —	Reserved

43.8.2.21 DAC Master Domain Assignment Register (MDA_W0_0_DFMT0 - MDA_W6_5_DFMT0)

Offset

Register	Offset
MDA_W0_0_DFMT0	800h
MDA_W1_0_DFMT0	804h
MDA_W0_1_DFMT0	820h
MDA_W1_1_DFMT0	824h
MDA_W2_1_DFMT0	828h
MDA_W0_5_DFMT0	8A0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MDA_W1_5_DFMT0	8A4h
MDA_W2_5_DFMT0	8A8h
MDA_W3_5_DFMT0	8ACh
MDA_W4_5_DFMT0	8B0h
MDA_W5_5_DFMT0	8B4h
MDA_W6_5_DFMT0	8B8h

Function

The MDA_Wr_m registers provide a 2-dimensional data structure for assigning bus masters to domains. The number of implemented registers is defined by DACFGm[NMDAR]. This per-master domain assignment is then repeated for each bus master (MDAm). Thus, m specifies the master number and r refers to the specific MDA register for a given bus master.

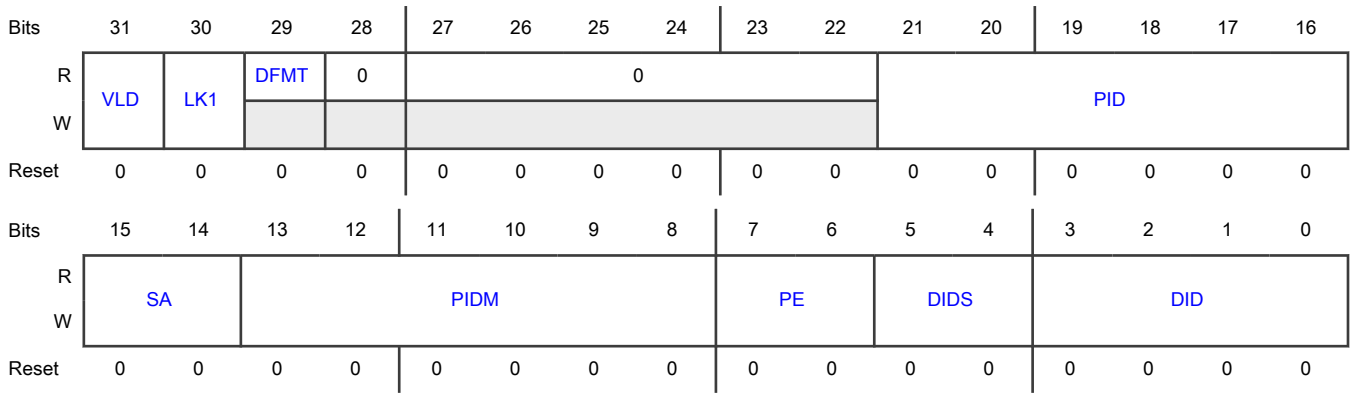
Each Wr within the MDAm structure is a word-sized definition; there are two formats supported, one for processor cores and another for non-processors. Processor masters typically support one or more Wr domain definitions, while non-processor masters support a single Wr. Processor masters Domain Assignment register names use a _DFMT0 suffix. Non-processor masters Domain Assignment register names use a _DFMT1 suffix.

The DAC submodule is responsible for the generation of domain identifiers for every transaction from every bus master. If there is a single Wr for a given master, then the specified domain identifier is used directly. If there are multiple Wr values for a given master, then the DAC evaluates the conditional terms to determine a “hit”. For all Wr hits, their corresponding domain identifiers are simply logically summed together (boolean OR). Use cases are typically expected to *hit in a single Wr* for a processor master. Special care is needed if *none* of the conditional terms hit in any Wr evaluation; for this case, the generated DID = 0 and software needs to be aware of any potential access rights granted for this DID.

Each MDA_Wr_m register has one of two programming models depending on the state of the domain format field, DFMT. The model described in this section is for DFMT = 0. This definition allows three different specifications of the DID for processors. The DFMT = 1 model is described in Master Domain Assignment (MDA_Wr_m_DFMT1) register.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/write.

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates the domain assignment is valid. It is further qualified by CR[GVLDM] = 1. If CR[GVLDM] is cleared, the DID output is defined by the SoC-specific *default DID* value. See the TRDC chip configuration section for more details. If both MDA_Wr_m[VLD] and CR[GVLDM] are asserted, the DID output is defined by the remaining contents of this register. 0b - The Wr domain assignment is invalid. 1b - The Wr domain assignment is valid.
30 LK1	1-bit Lock This field provides a locking mechanism that can be used to limit the ability to write the register. Once set, this bit remains asserted until the next reset. 0b - Register can be written by any secure privileged write. 1b - Register is locked (read-only) until the next reset.
29 DFMT	Domain format Identifies this register's domain assignment format. **NOTE** This bitfield access is ROZ 0b - Processor-core domain assignment 1b - Non-processor domain assignment
28 —	Reserved
27-22 —	Reserved
21-16 PID	Process Identifier This field specifies that the process identifier is to be combined with the PIDM field and included in the domain hit determination. The optional inclusion of the PID and PIDM is controlled by the MDA_Wr_m[PE] field.
15-14 SA	Secure attribute This field defines the secure/nonsecure attribute for processor cores.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The bus master's input secure/nonsecure attribute is used if SA = 1X, or this VLD = 0. If TZM_ENB = 0, the master attribute is forced to nonsecure. A nonsecure write cannot program this field to 2'b00, which is a security level higher than the mode of the process that is writing it. Reset value of SA: • if TZM_ENB=0, SA reset value = 2'b01 • if TZM_ENB=1, SA reset value = 2'b00 00b - Force the bus attribute for this master to secure. 01b - Force the bus attribute for this master to nonsecure. 10b - Use the bus master's secure/nonsecure attribute directly. 11b - Use the bus master's secure/nonsecure attribute directly.
13-8 PIDM	Process Identifier Mask This field provides a masking capability so that multiple process identifiers can be included as part of the domain hit determination. If a bit in the PIDM is set, then the corresponding bit of the PID is ignored in the comparison. The optional inclusion of the PID and PIDM is controlled by the MDA_Wr_m[PE] field.
7-6 PE	Process identifier enable Controls the optional inclusion of the PID, qualified by PIDM, into the domain hit evaluation. It provides the ability to include inclusive or exclusive sets of masked PID values. 00b - No process identifier is included in the domain hit evaluation. 01b - No process identifier is included in the domain hit evaluation. 10b - PE = 2. The process identifier is included in the domain hit evaluation as defined by the expression: `partial_domain_hit = (PE == 2) && ((PID & ~PIDM) == (TRDC_PIDn[PID] & ~PIDM))` 11b - PE = 3. The process identifier is included in the domain hit evaluation as defined by the expression: `partial_domain_hit = (PE == 3) && ~((PID & ~PIDM) == (TRDC_PIDn[PID] & ~PIDM))`
5-4 DIDS	DID Select This field selects the source of the domain identifier. 00b - Use MDAm[3:0] as the domain identifier. 01b - Use the input DID as the domain identifier. 10b - Use MDAm[3:2] concatenated with the low-order 2 bits of the input DID (DID_in[1:0]) as the domain identifier. 11b - Reserved for future use.
3-0	Domain identifier

Table continues on the next page...

Table continued from the previous page...

Field	Function
DID	This 4-bit field is the domain ID attribute that is sent on the accesses from the bus master connected to the DAC when DIDS=00b. DID[3:2] is used when DIDS=10b

43.8.2.22 DAC Master Domain Assignment Register (MDA_W0_2_DFMT1 - MDA_W0_4_DFMT1)

Offset

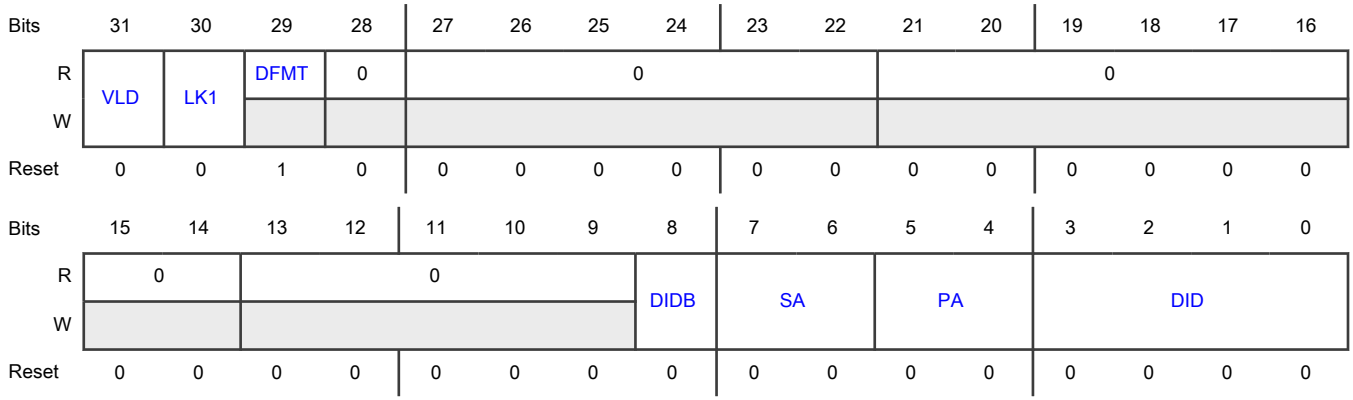
Register	Offset
MDA_W0_2_DFMT1	840h
MDA_W0_3_DFMT1	860h
MDA_W0_4_DFMT1	880h

Function

This register is identical to Master Domain Assignment (MDA_Wr_m_DFMT0) register except that the domain format field, DFMT, is 1 and the PID field is not used. This format supports two different specifications of the DID for non-core bus masters.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/write.

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates the domain assignment is valid. It is further qualified by CR[GVLDM] = 1. If CR[GVLDM] is cleared, the DID output is defined by the SoC-specific *default DID* value. See the TRDC chip configuration section for more details. If both MDA_Wr_m[VLD] and CR[GVLDM] are asserted, the DID output is defined by the remaining contents of this register. 0b - The Wr domain assignment is invalid.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - The Wr domain assignment is valid.
30 LK1	1-bit Lock This field provides a locking mechanism that can be used to limit the ability to write the register. Once set, this bit remains asserted until the next reset. 0b - Register can be written by any secure privileged write. 1b - Register is locked (read-only) until the next reset.
29 DFMT	Domain format Identifies this register's domain assignment format. **NOTE** This bitfield access is ROO 0b - Processor-core domain assignment 1b - Non-processor domain assignment
28 —	Reserved
27-22 —	Reserved
21-16 —	Reserved
15-14 —	Reserved
13-9 —	Reserved
8 DIDB	DID Bypass If asserted, this bit enables the bypassing of an input DID value as the domain identifier for this non-processor bus master. This capability allows non-processor bus masters, for example, a DMA to masquerade as a processor. Once set, this field is “sticky” and remains set until the next reset.

DAC	DID Input
0	0
1	DMA Domain ID

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>DAC</th><th>DID Input</th></tr>
<tr><td>2</td><td>USB Domain ID</td></tr>
</table>
0b - Use MDAn[3:0] as the domain identifier.
1b - Use the DID input as the domain identifier.
</td></tr>
<tr><td>7-6
SA</td><td>Secure attribute
This field defines the secure/nonsecure attribute for non-processor cores.
NOTE
The bus master’s input secure/nonsecure attribute is used if SA = 1X, or this VLD = 0. If TZM_ENB = 0, the master attribute is forced to nonsecure. A nonsecure write cannot program this field to 2'b00, which is a security level higher than the mode of the process that is writing it.
Reset value of SA:
• if TZM_ENB=0, SA reset value = 2'b01
• if TZM_ENB=1, SA reset value = 2'b00
00b - Force the bus attribute for this master to secure.
01b - Force the bus attribute for this master to nonsecure.
10b - Use the bus master’s secure/nonsecure attribute directly.
11b - Use the bus master’s secure/nonsecure attribute directly.</td></tr>
<tr><td>5-4
PA</td><td>Privileged attribute
This field defines the privileged/user attribute for non-processor cores.
NOTE
The bus master’s input privileged/user attribute is used if PA = 1X, or this VLD = 0.
00b - Force the bus attribute for this master to user.
01b - Force the bus attribute for this master to privileged.
10b - Use the bus master’s privileged/user attribute directly.
11b - Use the bus master’s privileged/user attribute directly.</td></tr>
<tr><td>3-0
DID</td><td>Domain identifier
This 4-bit field is the domain ID attribute that is sent on the accesses from the bus master connected to the DAC when DIDB=0.</td></tr>
</table>

43.8.2.23 MBC Global Configuration Register (MBC0_MEM0_GLBCFG - MBC1_MEM3_GLBCFG)

Offset

For m = 0 to 1; r = 0 to 3:

Register	Offset
MBCm_MEMr_GLBCFG	1_0000h + (m × 2000h) + (r × 4h)

Function

These MBC global configuration read-only registers contain information on the MBC's hardware configuration. Specifically, it defines the number of memory blocks and the size of each block in each MBC mem (r).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0										SIZE_LOG2			
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						NBLKS									
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
MBC0_MEM0_GLBCFG	0010_0080h
MBC0_MEM1_GLBCFG	0010_0006h
MBC0_MEM2_GLBCFG	0010_0005h
MBC0_MEM3_GLBCFG	0014_0011h
MBC1_MEM0_GLBCFG	0010_0080h
MBC1_MEM1_GLBCFG	0010_0001h
MBC1_MEM2_GLBCFG	0014_0001h
MBC1_MEM3_GLBCFG	0013_0001h

Fields

Field	Function
31-30 —	Reserved
29-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-16 SIZE_LOG2	Log2 size per block For example SIZE_LOG2=0x0C is 2^12=4 KB blocks.
15-10 —	Reserved
9-0 NBLKS	Number of blocks in this memory

43.8.2.24 MBC NonSecure Enable Block Index (MBC0_NSE_BLK_INDEX - MBC1_NSE_BLK_INDEX)

Offset

Register	Offset
MBC0_NSE_BLK_INDEX	1_0010h
MBC1_NSE_BLK_INDEX	1_2010h

Function

This R/W register defines the selected memories that are affected by the writes to the NSE_BLK_SET and NSE_BLK_CLR registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL9	DID_SEL8	DID_SEL7	DID_SEL6	DID_SEL5	DID_SEL4	DID_SEL3	DID_SEL2	DID_SEL1	DID_SEL0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				MEM_SEL				0		WNDX				0	AI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16	DID Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
DID_SELn	Destination domain bitmap select. 0b - No effect. 1b - Selects NSE bits for this domain.
15-12 —	Reserved
11-8 MEM_SEL	Memory Select Destination memory bitmap select. • Bit [11] - MBC MEM 3 • Bit [10] - MBC MEM 2 • Bit [9] - MBC MEM 1 • Bit [8] - MBC MEM 0
7-6 —	Reserved
5-2 WNDX	Word index into the block NSE bitmap. It selects the BLK_NSE_Wn register, where WNDX determines the value of n.
1 —	Reserved
0 AI	Auto Increment 0b - No effect. 1b - Add 1 to the WNDX field after the register write.

43.8.2.25 MBC NonSecure Enable Block Set (MBC0_NSE_BLK_SET - MBC1_NSE_BLK_SET)

Offset

Register	Offset
MBC0_NSE_BLK_SET	1_0014h
MBC1_NSE_BLK_SET	1_2014h

Function

A write to this register sets the appropriate NSE Bits for the selected domains and memories defined at the word location NSE_BLK_INDEX[WDNX]. If NSE_BLK_INDEX[AI] = 1, then the NSE_BLK_INDEX[WDNX] field is incremented by 1 (modulo-16) after the write is completed. This register reads as zero.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	W1SET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	W1SET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 W1SET	Write-1 Set If set to 1, sets appropriate NSE bit for the selected domains and memories defined at the word location NSE_BLK_INDEX[WDNX]. Write with value 0 has no effect.

43.8.2.26 MBC NonSecure Enable Block Clear (MBC0_NSE_BLK_CLR - MBC1_NSE_BLK_CLR)

Offset

Register	Offset
MBC0_NSE_BLK_CLR	1_0018h
MBC1_NSE_BLK_CLR	1_2018h

Function

A write to this location clears the appropriate NSE bits as defined by the W1CLR[n]=1 for the selected domains and memories Defined at the word location NSE_BLK_INDEX[WDNX]. If NSE_BLK_INDEX[AI] = 1, then the NSE_BLK_INDEX[WDNX] field is incremented by 1 (modulo-16) after the write is completed. This register reads as zero.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	W1CLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	W1CLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 W1CLR	Write-1 Clear If set to 1, Clear the appropriate NSE bit for the selected domains and memories defined at the word location NSE_BLK_INDEX[WDNX]. Write with value 0 has no effect.

43.8.2.27 MBC NonSecure Enable Block Clear All (MBC0_NSE_BLK_CLR_ALL - MBC1_NSE_BLK_CLR_ALL)

Offset

Register	Offset
MBC0_NSE_BLK_CLR_ALL	1_001Ch
MBC1_NSE_BLK_CLR_ALL	1_201Ch

Function
A write to this location clears all the block NSE bits for the selected domains and memories defined in the write data. This register reads as zero.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL...	DID_SEL9	DID_SEL8	DID_SEL7	DID_SEL6	DID_SEL5	DID_SEL4	DID_SEL3	DID_SEL2	DID_SEL1	DID_SEL0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0							
W					MEMSEL											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 DID_SELn	DID Select Destination domain bitmap select. 0b - No effect. 1b - Clear all NSE bits for this domain.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11-8 MEMSEL	Memory Select Destination memory bitmap select. • Bit [11] - clear all MEM 3 NSE bits • Bit [10] - clear all MEM 2 NSE bits • Bit [9] - clear all MEM 1 NSE bits • Bit [8] - clear all MEM 0 NSE bits
7-0 —	Reserved

43.8.2.28 MBC Global Access Control (MBC0_MEMN_GLBAC0)

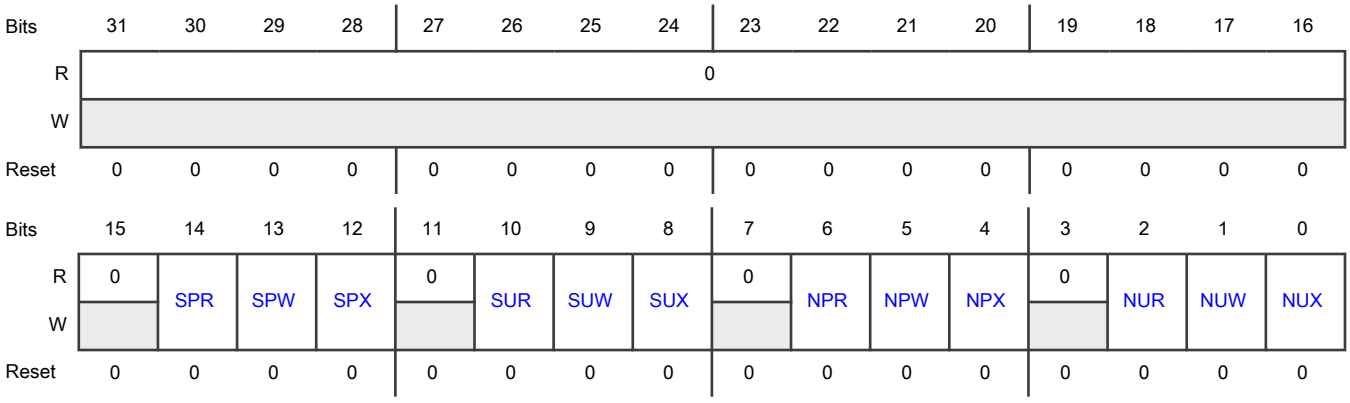
Offset

Register	Offset
MBC0_MEMN_GLBAC0	1_0020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6	NonsecurePriv Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
NPR	NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.29 MBC Global Access Control (MBC0_MEMN_GLBAC1 - MBC0_MEMN_GLBAC7)

Offset

Register	Offset
MBC0_MEMN_GLBAC1	1_0024h

Table continues on the next page...

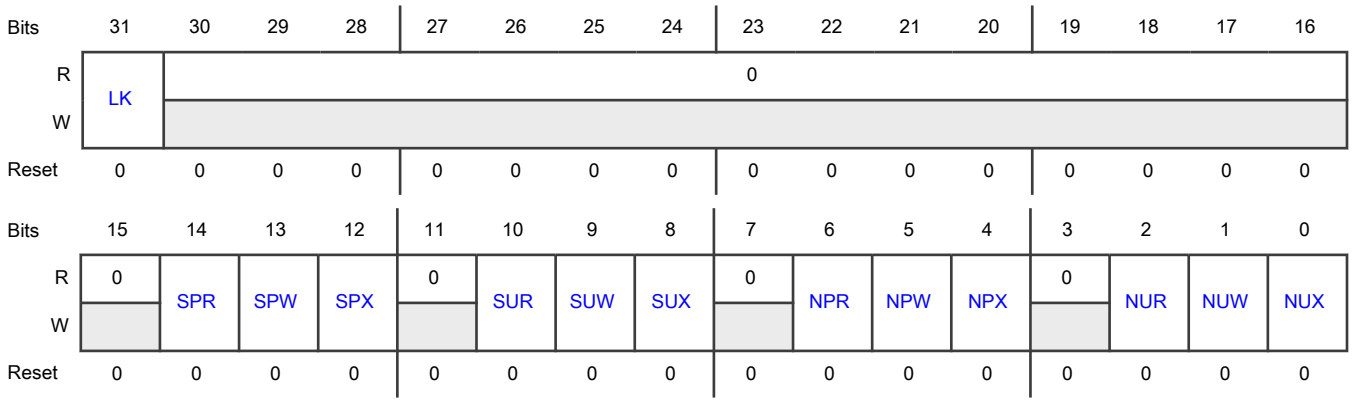
Table continued from the previous page...

Register	Offset
MBC0_MEMN_GLBAC2	1_0028h
MBC0_MEMN_GLBAC3	1_002Ch
MBC0_MEMN_GLBAC4	1_0030h
MBC0_MEMN_GLBAC5	1_0034h
MBC0_MEMN_GLBAC6	1_0038h
MBC0_MEMN_GLBAC7	1_003Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these control access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked and cannot be altered.
30-15 —	Reserved
14	SecurePriv Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
SPR	SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.30 MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W0 - MBC1_DOM15_MEM3_BLK_CFG_W0)

Offset

Register	Offset
MBC0_DOM0_MEM0_BLK_CFG_W0	1_0040h
MBC0_DOM0_MEM0_BLK_CFG_W1	1_0044h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM0_MEM0_BLK_CFG_W2	1_0048h
MBC0_DOM0_MEM0_BLK_CFG_W3	1_004Ch
MBC0_DOM0_MEM0_BLK_CFG_W4	1_0050h
MBC0_DOM0_MEM0_BLK_CFG_W5	1_0054h
MBC0_DOM0_MEM0_BLK_CFG_W6	1_0058h
MBC0_DOM0_MEM0_BLK_CFG_W7	1_005Ch
MBC0_DOM0_MEM0_BLK_CFG_W8	1_0060h
MBC0_DOM0_MEM0_BLK_CFG_W9	1_0064h
MBC0_DOM0_MEM0_BLK_CFG_W10	1_0068h
MBC0_DOM0_MEM0_BLK_CFG_W11	1_006Ch
MBC0_DOM0_MEM0_BLK_CFG_W12	1_0070h
MBC0_DOM0_MEM0_BLK_CFG_W13	1_0074h
MBC0_DOM0_MEM0_BLK_CFG_W14	1_0078h
MBC0_DOM0_MEM0_BLK_CFG_W15	1_007Ch
MBC0_DOM0_MEM1_BLK_CFG_W0	1_0180h
MBC0_DOM0_MEM2_BLK_CFG_W0	1_01A8h
MBC0_DOM0_MEM3_BLK_CFG_W0	1_01D0h
MBC0_DOM0_MEM3_BLK_CFG_W1	1_01D4h
MBC0_DOM0_MEM3_BLK_CFG_W2	1_01D8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM1_MEM0_BLK_CFG_W0	1_0240h
MBC0_DOM1_MEM0_BLK_CFG_W1	1_0244h
MBC0_DOM1_MEM0_BLK_CFG_W2	1_0248h
MBC0_DOM1_MEM0_BLK_CFG_W3	1_024Ch
MBC0_DOM1_MEM0_BLK_CFG_W4	1_0250h
MBC0_DOM1_MEM0_BLK_CFG_W5	1_0254h
MBC0_DOM1_MEM0_BLK_CFG_W6	1_0258h
MBC0_DOM1_MEM0_BLK_CFG_W7	1_025Ch
MBC0_DOM1_MEM0_BLK_CFG_W8	1_0260h
MBC0_DOM1_MEM0_BLK_CFG_W9	1_0264h
MBC0_DOM1_MEM0_BLK_CFG_W10	1_0268h
MBC0_DOM1_MEM0_BLK_CFG_W11	1_026Ch
MBC0_DOM1_MEM0_BLK_CFG_W12	1_0270h
MBC0_DOM1_MEM0_BLK_CFG_W13	1_0274h
MBC0_DOM1_MEM0_BLK_CFG_W14	1_0278h
MBC0_DOM1_MEM0_BLK_CFG_W15	1_027Ch
MBC0_DOM1_MEM1_BLK_CFG_W0	1_0380h
MBC0_DOM1_MEM2_BLK_CFG_W0	1_03A8h
MBC0_DOM1_MEM3_BLK_CFG_W0	1_03D0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM1_MEM3_BLK_CFG_W1	1_03D4h
MBC0_DOM1_MEM3_BLK_CFG_W2	1_03D8h
MBC0_DOM2_MEM0_BLK_CFG_W0	1_0440h
MBC0_DOM2_MEM0_BLK_CFG_W1	1_0444h
MBC0_DOM2_MEM0_BLK_CFG_W2	1_0448h
MBC0_DOM2_MEM0_BLK_CFG_W3	1_044Ch
MBC0_DOM2_MEM0_BLK_CFG_W4	1_0450h
MBC0_DOM2_MEM0_BLK_CFG_W5	1_0454h
MBC0_DOM2_MEM0_BLK_CFG_W6	1_0458h
MBC0_DOM2_MEM0_BLK_CFG_W7	1_045Ch
MBC0_DOM2_MEM0_BLK_CFG_W8	1_0460h
MBC0_DOM2_MEM0_BLK_CFG_W9	1_0464h
MBC0_DOM2_MEM0_BLK_CFG_W10	1_0468h
MBC0_DOM2_MEM0_BLK_CFG_W11	1_046Ch
MBC0_DOM2_MEM0_BLK_CFG_W12	1_0470h
MBC0_DOM2_MEM0_BLK_CFG_W13	1_0474h
MBC0_DOM2_MEM0_BLK_CFG_W14	1_0478h
MBC0_DOM2_MEM0_BLK_CFG_W15	1_047Ch
MBC0_DOM2_MEM1_BLK_CFG_W0	1_0580h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM2_MEM2_BLK_CFG_W0	1_05A8h
MBC0_DOM2_MEM3_BLK_CFG_W0	1_05D0h
MBC0_DOM2_MEM3_BLK_CFG_W1	1_05D4h
MBC0_DOM2_MEM3_BLK_CFG_W2	1_05D8h
MBC0_DOM3_MEM0_BLK_CFG_W0	1_0640h
MBC0_DOM3_MEM0_BLK_CFG_W1	1_0644h
MBC0_DOM3_MEM0_BLK_CFG_W2	1_0648h
MBC0_DOM3_MEM0_BLK_CFG_W3	1_064Ch
MBC0_DOM3_MEM0_BLK_CFG_W4	1_0650h
MBC0_DOM3_MEM0_BLK_CFG_W5	1_0654h
MBC0_DOM3_MEM0_BLK_CFG_W6	1_0658h
MBC0_DOM3_MEM0_BLK_CFG_W7	1_065Ch
MBC0_DOM3_MEM0_BLK_CFG_W8	1_0660h
MBC0_DOM3_MEM0_BLK_CFG_W9	1_0664h
MBC0_DOM3_MEM0_BLK_CFG_W10	1_0668h
MBC0_DOM3_MEM0_BLK_CFG_W11	1_066Ch
MBC0_DOM3_MEM0_BLK_CFG_W12	1_0670h
MBC0_DOM3_MEM0_BLK_CFG_W13	1_0674h
MBC0_DOM3_MEM0_BLK_CFG_W14	1_0678h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM3_MEM0_BLK_CFG_W15	1_067Ch
MBC0_DOM3_MEM1_BLK_CFG_W0	1_0780h
MBC0_DOM3_MEM2_BLK_CFG_W0	1_07A8h
MBC0_DOM3_MEM3_BLK_CFG_W0	1_07D0h
MBC0_DOM3_MEM3_BLK_CFG_W1	1_07D4h
MBC0_DOM3_MEM3_BLK_CFG_W2	1_07D8h
MBC0_DOM4_MEM0_BLK_CFG_W0	1_0840h
MBC0_DOM4_MEM0_BLK_CFG_W1	1_0844h
MBC0_DOM4_MEM0_BLK_CFG_W2	1_0848h
MBC0_DOM4_MEM0_BLK_CFG_W3	1_084Ch
MBC0_DOM4_MEM0_BLK_CFG_W4	1_0850h
MBC0_DOM4_MEM0_BLK_CFG_W5	1_0854h
MBC0_DOM4_MEM0_BLK_CFG_W6	1_0858h
MBC0_DOM4_MEM0_BLK_CFG_W7	1_085Ch
MBC0_DOM4_MEM0_BLK_CFG_W8	1_0860h
MBC0_DOM4_MEM0_BLK_CFG_W9	1_0864h
MBC0_DOM4_MEM0_BLK_CFG_W10	1_0868h
MBC0_DOM4_MEM0_BLK_CFG_W11	1_086Ch
MBC0_DOM4_MEM0_BLK_CFG_W12	1_0870h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM4_MEM0_BLK_CFG_W13	1_0874h
MBC0_DOM4_MEM0_BLK_CFG_W14	1_0878h
MBC0_DOM4_MEM0_BLK_CFG_W15	1_087Ch
MBC0_DOM4_MEM1_BLK_CFG_W0	1_0980h
MBC0_DOM4_MEM2_BLK_CFG_W0	1_09A8h
MBC0_DOM4_MEM3_BLK_CFG_W0	1_09D0h
MBC0_DOM4_MEM3_BLK_CFG_W1	1_09D4h
MBC0_DOM4_MEM3_BLK_CFG_W2	1_09D8h
MBC0_DOM5_MEM0_BLK_CFG_W0	1_0A40h
MBC0_DOM5_MEM0_BLK_CFG_W1	1_0A44h
MBC0_DOM5_MEM0_BLK_CFG_W2	1_0A48h
MBC0_DOM5_MEM0_BLK_CFG_W3	1_0A4Ch
MBC0_DOM5_MEM0_BLK_CFG_W4	1_0A50h
MBC0_DOM5_MEM0_BLK_CFG_W5	1_0A54h
MBC0_DOM5_MEM0_BLK_CFG_W6	1_0A58h
MBC0_DOM5_MEM0_BLK_CFG_W7	1_0A5Ch
MBC0_DOM5_MEM0_BLK_CFG_W8	1_0A60h
MBC0_DOM5_MEM0_BLK_CFG_W9	1_0A64h
MBC0_DOM5_MEM0_BLK_CFG_W10	1_0A68h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM5_MEM0_BLK_CFG_W11	1_0A6Ch
MBC0_DOM5_MEM0_BLK_CFG_W12	1_0A70h
MBC0_DOM5_MEM0_BLK_CFG_W13	1_0A74h
MBC0_DOM5_MEM0_BLK_CFG_W14	1_0A78h
MBC0_DOM5_MEM0_BLK_CFG_W15	1_0A7Ch
MBC0_DOM5_MEM1_BLK_CFG_W0	1_0B80h
MBC0_DOM5_MEM2_BLK_CFG_W0	1_0BA8h
MBC0_DOM5_MEM3_BLK_CFG_W0	1_0BD0h
MBC0_DOM5_MEM3_BLK_CFG_W1	1_0BD4h
MBC0_DOM5_MEM3_BLK_CFG_W2	1_0BD8h
MBC0_DOM6_MEM0_BLK_CFG_W0	1_0C40h
MBC0_DOM6_MEM0_BLK_CFG_W1	1_0C44h
MBC0_DOM6_MEM0_BLK_CFG_W2	1_0C48h
MBC0_DOM6_MEM0_BLK_CFG_W3	1_0C4Ch
MBC0_DOM6_MEM0_BLK_CFG_W4	1_0C50h
MBC0_DOM6_MEM0_BLK_CFG_W5	1_0C54h
MBC0_DOM6_MEM0_BLK_CFG_W6	1_0C58h
MBC0_DOM6_MEM0_BLK_CFG_W7	1_0C5Ch
MBC0_DOM6_MEM0_BLK_CFG_W8	1_0C60h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM6_MEM0_BLK_CFG_W9	1_0C64h
MBC0_DOM6_MEM0_BLK_CFG_W10	1_0C68h
MBC0_DOM6_MEM0_BLK_CFG_W11	1_0C6Ch
MBC0_DOM6_MEM0_BLK_CFG_W12	1_0C70h
MBC0_DOM6_MEM0_BLK_CFG_W13	1_0C74h
MBC0_DOM6_MEM0_BLK_CFG_W14	1_0C78h
MBC0_DOM6_MEM0_BLK_CFG_W15	1_0C7Ch
MBC0_DOM6_MEM1_BLK_CFG_W0	1_0D80h
MBC0_DOM6_MEM2_BLK_CFG_W0	1_0DA8h
MBC0_DOM6_MEM3_BLK_CFG_W0	1_0DD0h
MBC0_DOM6_MEM3_BLK_CFG_W1	1_0DD4h
MBC0_DOM6_MEM3_BLK_CFG_W2	1_0DD8h
MBC0_DOM7_MEM0_BLK_CFG_W0	1_0E40h
MBC0_DOM7_MEM0_BLK_CFG_W1	1_0E44h
MBC0_DOM7_MEM0_BLK_CFG_W2	1_0E48h
MBC0_DOM7_MEM0_BLK_CFG_W3	1_0E4Ch
MBC0_DOM7_MEM0_BLK_CFG_W4	1_0E50h
MBC0_DOM7_MEM0_BLK_CFG_W5	1_0E54h
MBC0_DOM7_MEM0_BLK_CFG_W6	1_0E58h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM7_MEM0_BLK_CFG_W7	1_0E5Ch
MBC0_DOM7_MEM0_BLK_CFG_W8	1_0E60h
MBC0_DOM7_MEM0_BLK_CFG_W9	1_0E64h
MBC0_DOM7_MEM0_BLK_CFG_W10	1_0E68h
MBC0_DOM7_MEM0_BLK_CFG_W11	1_0E6Ch
MBC0_DOM7_MEM0_BLK_CFG_W12	1_0E70h
MBC0_DOM7_MEM0_BLK_CFG_W13	1_0E74h
MBC0_DOM7_MEM0_BLK_CFG_W14	1_0E78h
MBC0_DOM7_MEM0_BLK_CFG_W15	1_0E7Ch
MBC0_DOM7_MEM1_BLK_CFG_W0	1_0F80h
MBC0_DOM7_MEM2_BLK_CFG_W0	1_0FA8h
MBC0_DOM7_MEM3_BLK_CFG_W0	1_0FD0h
MBC0_DOM7_MEM3_BLK_CFG_W1	1_0FD4h
MBC0_DOM7_MEM3_BLK_CFG_W2	1_0FD8h
MBC0_DOM8_MEM0_BLK_CFG_W0	1_1040h
MBC0_DOM8_MEM0_BLK_CFG_W1	1_1044h
MBC0_DOM8_MEM0_BLK_CFG_W2	1_1048h
MBC0_DOM8_MEM0_BLK_CFG_W3	1_104Ch
MBC0_DOM8_MEM0_BLK_CFG_W4	1_1050h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM8_MEM0_BLK_CFG_W5	1_1054h
MBC0_DOM8_MEM0_BLK_CFG_W6	1_1058h
MBC0_DOM8_MEM0_BLK_CFG_W7	1_105Ch
MBC0_DOM8_MEM0_BLK_CFG_W8	1_1060h
MBC0_DOM8_MEM0_BLK_CFG_W9	1_1064h
MBC0_DOM8_MEM0_BLK_CFG_W10	1_1068h
MBC0_DOM8_MEM0_BLK_CFG_W11	1_106Ch
MBC0_DOM8_MEM0_BLK_CFG_W12	1_1070h
MBC0_DOM8_MEM0_BLK_CFG_W13	1_1074h
MBC0_DOM8_MEM0_BLK_CFG_W14	1_1078h
MBC0_DOM8_MEM0_BLK_CFG_W15	1_107Ch
MBC0_DOM8_MEM1_BLK_CFG_W0	1_1180h
MBC0_DOM8_MEM2_BLK_CFG_W0	1_11A8h
MBC0_DOM8_MEM3_BLK_CFG_W0	1_11D0h
MBC0_DOM8_MEM3_BLK_CFG_W1	1_11D4h
MBC0_DOM8_MEM3_BLK_CFG_W2	1_11D8h
MBC0_DOM9_MEM0_BLK_CFG_W0	1_1240h
MBC0_DOM9_MEM0_BLK_CFG_W1	1_1244h
MBC0_DOM9_MEM0_BLK_CFG_W2	1_1248h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM9_MEM0_BLK_CFG_W3	1_124Ch
MBC0_DOM9_MEM0_BLK_CFG_W4	1_1250h
MBC0_DOM9_MEM0_BLK_CFG_W5	1_1254h
MBC0_DOM9_MEM0_BLK_CFG_W6	1_1258h
MBC0_DOM9_MEM0_BLK_CFG_W7	1_125Ch
MBC0_DOM9_MEM0_BLK_CFG_W8	1_1260h
MBC0_DOM9_MEM0_BLK_CFG_W9	1_1264h
MBC0_DOM9_MEM0_BLK_CFG_W10	1_1268h
MBC0_DOM9_MEM0_BLK_CFG_W11	1_126Ch
MBC0_DOM9_MEM0_BLK_CFG_W12	1_1270h
MBC0_DOM9_MEM0_BLK_CFG_W13	1_1274h
MBC0_DOM9_MEM0_BLK_CFG_W14	1_1278h
MBC0_DOM9_MEM0_BLK_CFG_W15	1_127Ch
MBC0_DOM9_MEM1_BLK_CFG_W0	1_1380h
MBC0_DOM9_MEM2_BLK_CFG_W0	1_13A8h
MBC0_DOM9_MEM3_BLK_CFG_W0	1_13D0h
MBC0_DOM9_MEM3_BLK_CFG_W1	1_13D4h
MBC0_DOM9_MEM3_BLK_CFG_W2	1_13D8h
MBC0_DOM10_MEM0_BLK_CFG_W0	1_1440h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM10_MEM0_BLK_CFG_W1	1_1444h
MBC0_DOM10_MEM0_BLK_CFG_W2	1_1448h
MBC0_DOM10_MEM0_BLK_CFG_W3	1_144Ch
MBC0_DOM10_MEM0_BLK_CFG_W4	1_1450h
MBC0_DOM10_MEM0_BLK_CFG_W5	1_1454h
MBC0_DOM10_MEM0_BLK_CFG_W6	1_1458h
MBC0_DOM10_MEM0_BLK_CFG_W7	1_145Ch
MBC0_DOM10_MEM0_BLK_CFG_W8	1_1460h
MBC0_DOM10_MEM0_BLK_CFG_W9	1_1464h
MBC0_DOM10_MEM0_BLK_CFG_W10	1_1468h
MBC0_DOM10_MEM0_BLK_CFG_W11	1_146Ch
MBC0_DOM10_MEM0_BLK_CFG_W12	1_1470h
MBC0_DOM10_MEM0_BLK_CFG_W13	1_1474h
MBC0_DOM10_MEM0_BLK_CFG_W14	1_1478h
MBC0_DOM10_MEM0_BLK_CFG_W15	1_147Ch
MBC0_DOM10_MEM1_BLK_CFG_W0	1_1580h
MBC0_DOM10_MEM2_BLK_CFG_W0	1_15A8h
MBC0_DOM10_MEM3_BLK_CFG_W0	1_15D0h
MBC0_DOM10_MEM3_BLK_CFG_W1	1_15D4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM10_MEM3_BLK_CFG_W2	1_15D8h
MBC0_DOM11_MEM0_BLK_CFG_W0	1_1640h
MBC0_DOM11_MEM0_BLK_CFG_W1	1_1644h
MBC0_DOM11_MEM0_BLK_CFG_W2	1_1648h
MBC0_DOM11_MEM0_BLK_CFG_W3	1_164Ch
MBC0_DOM11_MEM0_BLK_CFG_W4	1_1650h
MBC0_DOM11_MEM0_BLK_CFG_W5	1_1654h
MBC0_DOM11_MEM0_BLK_CFG_W6	1_1658h
MBC0_DOM11_MEM0_BLK_CFG_W7	1_165Ch
MBC0_DOM11_MEM0_BLK_CFG_W8	1_1660h
MBC0_DOM11_MEM0_BLK_CFG_W9	1_1664h
MBC0_DOM11_MEM0_BLK_CFG_W10	1_1668h
MBC0_DOM11_MEM0_BLK_CFG_W11	1_166Ch
MBC0_DOM11_MEM0_BLK_CFG_W12	1_1670h
MBC0_DOM11_MEM0_BLK_CFG_W13	1_1674h
MBC0_DOM11_MEM0_BLK_CFG_W14	1_1678h
MBC0_DOM11_MEM0_BLK_CFG_W15	1_167Ch
MBC0_DOM11_MEM1_BLK_CFG_W0	1_1780h
MBC0_DOM11_MEM2_BLK_CFG_W0	1_17A8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM11_MEM3_BLK_CFG_W0	1_17D0h
MBC0_DOM11_MEM3_BLK_CFG_W1	1_17D4h
MBC0_DOM11_MEM3_BLK_CFG_W2	1_17D8h
MBC0_DOM12_MEM0_BLK_CFG_W0	1_1840h
MBC0_DOM12_MEM0_BLK_CFG_W1	1_1844h
MBC0_DOM12_MEM0_BLK_CFG_W2	1_1848h
MBC0_DOM12_MEM0_BLK_CFG_W3	1_184Ch
MBC0_DOM12_MEM0_BLK_CFG_W4	1_1850h
MBC0_DOM12_MEM0_BLK_CFG_W5	1_1854h
MBC0_DOM12_MEM0_BLK_CFG_W6	1_1858h
MBC0_DOM12_MEM0_BLK_CFG_W7	1_185Ch
MBC0_DOM12_MEM0_BLK_CFG_W8	1_1860h
MBC0_DOM12_MEM0_BLK_CFG_W9	1_1864h
MBC0_DOM12_MEM0_BLK_CFG_W10	1_1868h
MBC0_DOM12_MEM0_BLK_CFG_W11	1_186Ch
MBC0_DOM12_MEM0_BLK_CFG_W12	1_1870h
MBC0_DOM12_MEM0_BLK_CFG_W13	1_1874h
MBC0_DOM12_MEM0_BLK_CFG_W14	1_1878h
MBC0_DOM12_MEM0_BLK_CFG_W15	1_187Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM12_MEM1_BLK_CFG_W0	1_1980h
MBC0_DOM12_MEM2_BLK_CFG_W0	1_19A8h
MBC0_DOM12_MEM3_BLK_CFG_W0	1_19D0h
MBC0_DOM12_MEM3_BLK_CFG_W1	1_19D4h
MBC0_DOM12_MEM3_BLK_CFG_W2	1_19D8h
MBC0_DOM13_MEM0_BLK_CFG_W0	1_1A40h
MBC0_DOM13_MEM0_BLK_CFG_W1	1_1A44h
MBC0_DOM13_MEM0_BLK_CFG_W2	1_1A48h
MBC0_DOM13_MEM0_BLK_CFG_W3	1_1A4Ch
MBC0_DOM13_MEM0_BLK_CFG_W4	1_1A50h
MBC0_DOM13_MEM0_BLK_CFG_W5	1_1A54h
MBC0_DOM13_MEM0_BLK_CFG_W6	1_1A58h
MBC0_DOM13_MEM0_BLK_CFG_W7	1_1A5Ch
MBC0_DOM13_MEM0_BLK_CFG_W8	1_1A60h
MBC0_DOM13_MEM0_BLK_CFG_W9	1_1A64h
MBC0_DOM13_MEM0_BLK_CFG_W10	1_1A68h
MBC0_DOM13_MEM0_BLK_CFG_W11	1_1A6Ch
MBC0_DOM13_MEM0_BLK_CFG_W12	1_1A70h
MBC0_DOM13_MEM0_BLK_CFG_W13	1_1A74h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM13_MEM0_BLK_CFG_W14	1_1A78h
MBC0_DOM13_MEM0_BLK_CFG_W15	1_1A7Ch
MBC0_DOM13_MEM1_BLK_CFG_W0	1_1B80h
MBC0_DOM13_MEM2_BLK_CFG_W0	1_1BA8h
MBC0_DOM13_MEM3_BLK_CFG_W0	1_1BD0h
MBC0_DOM13_MEM3_BLK_CFG_W1	1_1BD4h
MBC0_DOM13_MEM3_BLK_CFG_W2	1_1BD8h
MBC0_DOM14_MEM0_BLK_CFG_W0	1_1C40h
MBC0_DOM14_MEM0_BLK_CFG_W1	1_1C44h
MBC0_DOM14_MEM0_BLK_CFG_W2	1_1C48h
MBC0_DOM14_MEM0_BLK_CFG_W3	1_1C4Ch
MBC0_DOM14_MEM0_BLK_CFG_W4	1_1C50h
MBC0_DOM14_MEM0_BLK_CFG_W5	1_1C54h
MBC0_DOM14_MEM0_BLK_CFG_W6	1_1C58h
MBC0_DOM14_MEM0_BLK_CFG_W7	1_1C5Ch
MBC0_DOM14_MEM0_BLK_CFG_W8	1_1C60h
MBC0_DOM14_MEM0_BLK_CFG_W9	1_1C64h
MBC0_DOM14_MEM0_BLK_CFG_W10	1_1C68h
MBC0_DOM14_MEM0_BLK_CFG_W11	1_1C6Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM14_MEM0_BLK_CFG_W12	1_1C70h
MBC0_DOM14_MEM0_BLK_CFG_W13	1_1C74h
MBC0_DOM14_MEM0_BLK_CFG_W14	1_1C78h
MBC0_DOM14_MEM0_BLK_CFG_W15	1_1C7Ch
MBC0_DOM14_MEM1_BLK_CFG_W0	1_1D80h
MBC0_DOM14_MEM2_BLK_CFG_W0	1_1DA8h
MBC0_DOM14_MEM3_BLK_CFG_W0	1_1DD0h
MBC0_DOM14_MEM3_BLK_CFG_W1	1_1DD4h
MBC0_DOM14_MEM3_BLK_CFG_W2	1_1DD8h
MBC0_DOM15_MEM0_BLK_CFG_W0	1_1E40h
MBC0_DOM15_MEM0_BLK_CFG_W1	1_1E44h
MBC0_DOM15_MEM0_BLK_CFG_W2	1_1E48h
MBC0_DOM15_MEM0_BLK_CFG_W3	1_1E4Ch
MBC0_DOM15_MEM0_BLK_CFG_W4	1_1E50h
MBC0_DOM15_MEM0_BLK_CFG_W5	1_1E54h
MBC0_DOM15_MEM0_BLK_CFG_W6	1_1E58h
MBC0_DOM15_MEM0_BLK_CFG_W7	1_1E5Ch
MBC0_DOM15_MEM0_BLK_CFG_W8	1_1E60h
MBC0_DOM15_MEM0_BLK_CFG_W9	1_1E64h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM15_MEM0_BLK_CFG_W10	1_1E68h
MBC0_DOM15_MEM0_BLK_CFG_W11	1_1E6Ch
MBC0_DOM15_MEM0_BLK_CFG_W12	1_1E70h
MBC0_DOM15_MEM0_BLK_CFG_W13	1_1E74h
MBC0_DOM15_MEM0_BLK_CFG_W14	1_1E78h
MBC0_DOM15_MEM0_BLK_CFG_W15	1_1E7Ch
MBC0_DOM15_MEM1_BLK_CFG_W0	1_1F80h
MBC0_DOM15_MEM2_BLK_CFG_W0	1_1FA8h
MBC0_DOM15_MEM3_BLK_CFG_W0	1_1FD0h
MBC0_DOM15_MEM3_BLK_CFG_W1	1_1FD4h
MBC0_DOM15_MEM3_BLK_CFG_W2	1_1FD8h
MBC1_DOM0_MEM0_BLK_CFG_W0	1_2040h
MBC1_DOM0_MEM0_BLK_CFG_W1	1_2044h
MBC1_DOM0_MEM0_BLK_CFG_W2	1_2048h
MBC1_DOM0_MEM0_BLK_CFG_W3	1_204Ch
MBC1_DOM0_MEM0_BLK_CFG_W4	1_2050h
MBC1_DOM0_MEM0_BLK_CFG_W5	1_2054h
MBC1_DOM0_MEM0_BLK_CFG_W6	1_2058h
MBC1_DOM0_MEM0_BLK_CFG_W7	1_205Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM0_MEM0_BLK_CFG_W8	1_2060h
MBC1_DOM0_MEM0_BLK_CFG_W9	1_2064h
MBC1_DOM0_MEM0_BLK_CFG_W10	1_2068h
MBC1_DOM0_MEM0_BLK_CFG_W11	1_206Ch
MBC1_DOM0_MEM0_BLK_CFG_W12	1_2070h
MBC1_DOM0_MEM0_BLK_CFG_W13	1_2074h
MBC1_DOM0_MEM0_BLK_CFG_W14	1_2078h
MBC1_DOM0_MEM0_BLK_CFG_W15	1_207Ch
MBC1_DOM0_MEM1_BLK_CFG_W0	1_2180h
MBC1_DOM0_MEM2_BLK_CFG_W0	1_21A8h
MBC1_DOM0_MEM3_BLK_CFG_W0	1_21D0h
MBC1_DOM1_MEM0_BLK_CFG_W0	1_2240h
MBC1_DOM1_MEM0_BLK_CFG_W1	1_2244h
MBC1_DOM1_MEM0_BLK_CFG_W2	1_2248h
MBC1_DOM1_MEM0_BLK_CFG_W3	1_224Ch
MBC1_DOM1_MEM0_BLK_CFG_W4	1_2250h
MBC1_DOM1_MEM0_BLK_CFG_W5	1_2254h
MBC1_DOM1_MEM0_BLK_CFG_W6	1_2258h
MBC1_DOM1_MEM0_BLK_CFG_W7	1_225Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM1_MEM0_BLK_CFG_W8	1_2260h
MBC1_DOM1_MEM0_BLK_CFG_W9	1_2264h
MBC1_DOM1_MEM0_BLK_CFG_W10	1_2268h
MBC1_DOM1_MEM0_BLK_CFG_W11	1_226Ch
MBC1_DOM1_MEM0_BLK_CFG_W12	1_2270h
MBC1_DOM1_MEM0_BLK_CFG_W13	1_2274h
MBC1_DOM1_MEM0_BLK_CFG_W14	1_2278h
MBC1_DOM1_MEM0_BLK_CFG_W15	1_227Ch
MBC1_DOM1_MEM1_BLK_CFG_W0	1_2380h
MBC1_DOM1_MEM2_BLK_CFG_W0	1_23A8h
MBC1_DOM1_MEM3_BLK_CFG_W0	1_23D0h
MBC1_DOM2_MEM0_BLK_CFG_W0	1_2440h
MBC1_DOM2_MEM0_BLK_CFG_W1	1_2444h
MBC1_DOM2_MEM0_BLK_CFG_W2	1_2448h
MBC1_DOM2_MEM0_BLK_CFG_W3	1_244Ch
MBC1_DOM2_MEM0_BLK_CFG_W4	1_2450h
MBC1_DOM2_MEM0_BLK_CFG_W5	1_2454h
MBC1_DOM2_MEM0_BLK_CFG_W6	1_2458h
MBC1_DOM2_MEM0_BLK_CFG_W7	1_245Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM2_MEM0_BLK_CFG_W8	1_2460h
MBC1_DOM2_MEM0_BLK_CFG_W9	1_2464h
MBC1_DOM2_MEM0_BLK_CFG_W10	1_2468h
MBC1_DOM2_MEM0_BLK_CFG_W11	1_246Ch
MBC1_DOM2_MEM0_BLK_CFG_W12	1_2470h
MBC1_DOM2_MEM0_BLK_CFG_W13	1_2474h
MBC1_DOM2_MEM0_BLK_CFG_W14	1_2478h
MBC1_DOM2_MEM0_BLK_CFG_W15	1_247Ch
MBC1_DOM2_MEM1_BLK_CFG_W0	1_2580h
MBC1_DOM2_MEM2_BLK_CFG_W0	1_25A8h
MBC1_DOM2_MEM3_BLK_CFG_W0	1_25D0h
MBC1_DOM3_MEM0_BLK_CFG_W0	1_2640h
MBC1_DOM3_MEM0_BLK_CFG_W1	1_2644h
MBC1_DOM3_MEM0_BLK_CFG_W2	1_2648h
MBC1_DOM3_MEM0_BLK_CFG_W3	1_264Ch
MBC1_DOM3_MEM0_BLK_CFG_W4	1_2650h
MBC1_DOM3_MEM0_BLK_CFG_W5	1_2654h
MBC1_DOM3_MEM0_BLK_CFG_W6	1_2658h
MBC1_DOM3_MEM0_BLK_CFG_W7	1_265Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM3_MEM0_BLK_CFG_W8	1_2660h
MBC1_DOM3_MEM0_BLK_CFG_W9	1_2664h
MBC1_DOM3_MEM0_BLK_CFG_W10	1_2668h
MBC1_DOM3_MEM0_BLK_CFG_W11	1_266Ch
MBC1_DOM3_MEM0_BLK_CFG_W12	1_2670h
MBC1_DOM3_MEM0_BLK_CFG_W13	1_2674h
MBC1_DOM3_MEM0_BLK_CFG_W14	1_2678h
MBC1_DOM3_MEM0_BLK_CFG_W15	1_267Ch
MBC1_DOM3_MEM1_BLK_CFG_W0	1_2780h
MBC1_DOM3_MEM2_BLK_CFG_W0	1_27A8h
MBC1_DOM3_MEM3_BLK_CFG_W0	1_27D0h
MBC1_DOM4_MEM0_BLK_CFG_W0	1_2840h
MBC1_DOM4_MEM0_BLK_CFG_W1	1_2844h
MBC1_DOM4_MEM0_BLK_CFG_W2	1_2848h
MBC1_DOM4_MEM0_BLK_CFG_W3	1_284Ch
MBC1_DOM4_MEM0_BLK_CFG_W4	1_2850h
MBC1_DOM4_MEM0_BLK_CFG_W5	1_2854h
MBC1_DOM4_MEM0_BLK_CFG_W6	1_2858h
MBC1_DOM4_MEM0_BLK_CFG_W7	1_285Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM4_MEM0_BLK_CFG_W8	1_2860h
MBC1_DOM4_MEM0_BLK_CFG_W9	1_2864h
MBC1_DOM4_MEM0_BLK_CFG_W10	1_2868h
MBC1_DOM4_MEM0_BLK_CFG_W11	1_286Ch
MBC1_DOM4_MEM0_BLK_CFG_W12	1_2870h
MBC1_DOM4_MEM0_BLK_CFG_W13	1_2874h
MBC1_DOM4_MEM0_BLK_CFG_W14	1_2878h
MBC1_DOM4_MEM0_BLK_CFG_W15	1_287Ch
MBC1_DOM4_MEM1_BLK_CFG_W0	1_2980h
MBC1_DOM4_MEM2_BLK_CFG_W0	1_29A8h
MBC1_DOM4_MEM3_BLK_CFG_W0	1_29D0h
MBC1_DOM5_MEM0_BLK_CFG_W0	1_2A40h
MBC1_DOM5_MEM0_BLK_CFG_W1	1_2A44h
MBC1_DOM5_MEM0_BLK_CFG_W2	1_2A48h
MBC1_DOM5_MEM0_BLK_CFG_W3	1_2A4Ch
MBC1_DOM5_MEM0_BLK_CFG_W4	1_2A50h
MBC1_DOM5_MEM0_BLK_CFG_W5	1_2A54h
MBC1_DOM5_MEM0_BLK_CFG_W6	1_2A58h
MBC1_DOM5_MEM0_BLK_CFG_W7	1_2A5Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM5_MEM0_BLK_CFG_W8	1_2A60h
MBC1_DOM5_MEM0_BLK_CFG_W9	1_2A64h
MBC1_DOM5_MEM0_BLK_CFG_W10	1_2A68h
MBC1_DOM5_MEM0_BLK_CFG_W11	1_2A6Ch
MBC1_DOM5_MEM0_BLK_CFG_W12	1_2A70h
MBC1_DOM5_MEM0_BLK_CFG_W13	1_2A74h
MBC1_DOM5_MEM0_BLK_CFG_W14	1_2A78h
MBC1_DOM5_MEM0_BLK_CFG_W15	1_2A7Ch
MBC1_DOM5_MEM1_BLK_CFG_W0	1_2B80h
MBC1_DOM5_MEM2_BLK_CFG_W0	1_2BA8h
MBC1_DOM5_MEM3_BLK_CFG_W0	1_2BD0h
MBC1_DOM6_MEM0_BLK_CFG_W0	1_2C40h
MBC1_DOM6_MEM0_BLK_CFG_W1	1_2C44h
MBC1_DOM6_MEM0_BLK_CFG_W2	1_2C48h
MBC1_DOM6_MEM0_BLK_CFG_W3	1_2C4Ch
MBC1_DOM6_MEM0_BLK_CFG_W4	1_2C50h
MBC1_DOM6_MEM0_BLK_CFG_W5	1_2C54h
MBC1_DOM6_MEM0_BLK_CFG_W6	1_2C58h
MBC1_DOM6_MEM0_BLK_CFG_W7	1_2C5Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM6_MEM0_BLK_CFG_W8	1_2C60h
MBC1_DOM6_MEM0_BLK_CFG_W9	1_2C64h
MBC1_DOM6_MEM0_BLK_CFG_W10	1_2C68h
MBC1_DOM6_MEM0_BLK_CFG_W11	1_2C6Ch
MBC1_DOM6_MEM0_BLK_CFG_W12	1_2C70h
MBC1_DOM6_MEM0_BLK_CFG_W13	1_2C74h
MBC1_DOM6_MEM0_BLK_CFG_W14	1_2C78h
MBC1_DOM6_MEM0_BLK_CFG_W15	1_2C7Ch
MBC1_DOM6_MEM1_BLK_CFG_W0	1_2D80h
MBC1_DOM6_MEM2_BLK_CFG_W0	1_2DA8h
MBC1_DOM6_MEM3_BLK_CFG_W0	1_2DD0h
MBC1_DOM7_MEM0_BLK_CFG_W0	1_2E40h
MBC1_DOM7_MEM0_BLK_CFG_W1	1_2E44h
MBC1_DOM7_MEM0_BLK_CFG_W2	1_2E48h
MBC1_DOM7_MEM0_BLK_CFG_W3	1_2E4Ch
MBC1_DOM7_MEM0_BLK_CFG_W4	1_2E50h
MBC1_DOM7_MEM0_BLK_CFG_W5	1_2E54h
MBC1_DOM7_MEM0_BLK_CFG_W6	1_2E58h
MBC1_DOM7_MEM0_BLK_CFG_W7	1_2E5Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM7_MEM0_BLK_CFG_W8	1_2E60h
MBC1_DOM7_MEM0_BLK_CFG_W9	1_2E64h
MBC1_DOM7_MEM0_BLK_CFG_W10	1_2E68h
MBC1_DOM7_MEM0_BLK_CFG_W11	1_2E6Ch
MBC1_DOM7_MEM0_BLK_CFG_W12	1_2E70h
MBC1_DOM7_MEM0_BLK_CFG_W13	1_2E74h
MBC1_DOM7_MEM0_BLK_CFG_W14	1_2E78h
MBC1_DOM7_MEM0_BLK_CFG_W15	1_2E7Ch
MBC1_DOM7_MEM1_BLK_CFG_W0	1_2F80h
MBC1_DOM7_MEM2_BLK_CFG_W0	1_2FA8h
MBC1_DOM7_MEM3_BLK_CFG_W0	1_2FD0h
MBC1_DOM8_MEM0_BLK_CFG_W0	1_3040h
MBC1_DOM8_MEM0_BLK_CFG_W1	1_3044h
MBC1_DOM8_MEM0_BLK_CFG_W2	1_3048h
MBC1_DOM8_MEM0_BLK_CFG_W3	1_304Ch
MBC1_DOM8_MEM0_BLK_CFG_W4	1_3050h
MBC1_DOM8_MEM0_BLK_CFG_W5	1_3054h
MBC1_DOM8_MEM0_BLK_CFG_W6	1_3058h
MBC1_DOM8_MEM0_BLK_CFG_W7	1_305Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM8_MEM0_BLK_CFG_W8	1_3060h
MBC1_DOM8_MEM0_BLK_CFG_W9	1_3064h
MBC1_DOM8_MEM0_BLK_CFG_W10	1_3068h
MBC1_DOM8_MEM0_BLK_CFG_W11	1_306Ch
MBC1_DOM8_MEM0_BLK_CFG_W12	1_3070h
MBC1_DOM8_MEM0_BLK_CFG_W13	1_3074h
MBC1_DOM8_MEM0_BLK_CFG_W14	1_3078h
MBC1_DOM8_MEM0_BLK_CFG_W15	1_307Ch
MBC1_DOM8_MEM1_BLK_CFG_W0	1_3180h
MBC1_DOM8_MEM2_BLK_CFG_W0	1_31A8h
MBC1_DOM8_MEM3_BLK_CFG_W0	1_31D0h
MBC1_DOM9_MEM0_BLK_CFG_W0	1_3240h
MBC1_DOM9_MEM0_BLK_CFG_W1	1_3244h
MBC1_DOM9_MEM0_BLK_CFG_W2	1_3248h
MBC1_DOM9_MEM0_BLK_CFG_W3	1_324Ch
MBC1_DOM9_MEM0_BLK_CFG_W4	1_3250h
MBC1_DOM9_MEM0_BLK_CFG_W5	1_3254h
MBC1_DOM9_MEM0_BLK_CFG_W6	1_3258h
MBC1_DOM9_MEM0_BLK_CFG_W7	1_325Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM9_MEM0_BLK_CFG_W8	1_3260h
MBC1_DOM9_MEM0_BLK_CFG_W9	1_3264h
MBC1_DOM9_MEM0_BLK_CFG_W10	1_3268h
MBC1_DOM9_MEM0_BLK_CFG_W11	1_326Ch
MBC1_DOM9_MEM0_BLK_CFG_W12	1_3270h
MBC1_DOM9_MEM0_BLK_CFG_W13	1_3274h
MBC1_DOM9_MEM0_BLK_CFG_W14	1_3278h
MBC1_DOM9_MEM0_BLK_CFG_W15	1_327Ch
MBC1_DOM9_MEM1_BLK_CFG_W0	1_3380h
MBC1_DOM9_MEM2_BLK_CFG_W0	1_33A8h
MBC1_DOM9_MEM3_BLK_CFG_W0	1_33D0h
MBC1_DOM10_MEM0_BLK_CFG_W0	1_3440h
MBC1_DOM10_MEM0_BLK_CFG_W1	1_3444h
MBC1_DOM10_MEM0_BLK_CFG_W2	1_3448h
MBC1_DOM10_MEM0_BLK_CFG_W3	1_344Ch
MBC1_DOM10_MEM0_BLK_CFG_W4	1_3450h
MBC1_DOM10_MEM0_BLK_CFG_W5	1_3454h
MBC1_DOM10_MEM0_BLK_CFG_W6	1_3458h
MBC1_DOM10_MEM0_BLK_CFG_W7	1_345Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM10_MEM0_BLK_CFG_W8	1_3460h
MBC1_DOM10_MEM0_BLK_CFG_W9	1_3464h
MBC1_DOM10_MEM0_BLK_CFG_W10	1_3468h
MBC1_DOM10_MEM0_BLK_CFG_W11	1_346Ch
MBC1_DOM10_MEM0_BLK_CFG_W12	1_3470h
MBC1_DOM10_MEM0_BLK_CFG_W13	1_3474h
MBC1_DOM10_MEM0_BLK_CFG_W14	1_3478h
MBC1_DOM10_MEM0_BLK_CFG_W15	1_347Ch
MBC1_DOM10_MEM1_BLK_CFG_W0	1_3580h
MBC1_DOM10_MEM2_BLK_CFG_W0	1_35A8h
MBC1_DOM10_MEM3_BLK_CFG_W0	1_35D0h
MBC1_DOM11_MEM0_BLK_CFG_W0	1_3640h
MBC1_DOM11_MEM0_BLK_CFG_W1	1_3644h
MBC1_DOM11_MEM0_BLK_CFG_W2	1_3648h
MBC1_DOM11_MEM0_BLK_CFG_W3	1_364Ch
MBC1_DOM11_MEM0_BLK_CFG_W4	1_3650h
MBC1_DOM11_MEM0_BLK_CFG_W5	1_3654h
MBC1_DOM11_MEM0_BLK_CFG_W6	1_3658h
MBC1_DOM11_MEM0_BLK_CFG_W7	1_365Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM11_MEM0_BLK_CFG_W8	1_3660h
MBC1_DOM11_MEM0_BLK_CFG_W9	1_3664h
MBC1_DOM11_MEM0_BLK_CFG_W10	1_3668h
MBC1_DOM11_MEM0_BLK_CFG_W11	1_366Ch
MBC1_DOM11_MEM0_BLK_CFG_W12	1_3670h
MBC1_DOM11_MEM0_BLK_CFG_W13	1_3674h
MBC1_DOM11_MEM0_BLK_CFG_W14	1_3678h
MBC1_DOM11_MEM0_BLK_CFG_W15	1_367Ch
MBC1_DOM11_MEM1_BLK_CFG_W0	1_3780h
MBC1_DOM11_MEM2_BLK_CFG_W0	1_37A8h
MBC1_DOM11_MEM3_BLK_CFG_W0	1_37D0h
MBC1_DOM12_MEM0_BLK_CFG_W0	1_3840h
MBC1_DOM12_MEM0_BLK_CFG_W1	1_3844h
MBC1_DOM12_MEM0_BLK_CFG_W2	1_3848h
MBC1_DOM12_MEM0_BLK_CFG_W3	1_384Ch
MBC1_DOM12_MEM0_BLK_CFG_W4	1_3850h
MBC1_DOM12_MEM0_BLK_CFG_W5	1_3854h
MBC1_DOM12_MEM0_BLK_CFG_W6	1_3858h
MBC1_DOM12_MEM0_BLK_CFG_W7	1_385Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM12_MEM0_BLK_CFG_W8	1_3860h
MBC1_DOM12_MEM0_BLK_CFG_W9	1_3864h
MBC1_DOM12_MEM0_BLK_CFG_W10	1_3868h
MBC1_DOM12_MEM0_BLK_CFG_W11	1_386Ch
MBC1_DOM12_MEM0_BLK_CFG_W12	1_3870h
MBC1_DOM12_MEM0_BLK_CFG_W13	1_3874h
MBC1_DOM12_MEM0_BLK_CFG_W14	1_3878h
MBC1_DOM12_MEM0_BLK_CFG_W15	1_387Ch
MBC1_DOM12_MEM1_BLK_CFG_W0	1_3980h
MBC1_DOM12_MEM2_BLK_CFG_W0	1_39A8h
MBC1_DOM12_MEM3_BLK_CFG_W0	1_39D0h
MBC1_DOM13_MEM0_BLK_CFG_W0	1_3A40h
MBC1_DOM13_MEM0_BLK_CFG_W1	1_3A44h
MBC1_DOM13_MEM0_BLK_CFG_W2	1_3A48h
MBC1_DOM13_MEM0_BLK_CFG_W3	1_3A4Ch
MBC1_DOM13_MEM0_BLK_CFG_W4	1_3A50h
MBC1_DOM13_MEM0_BLK_CFG_W5	1_3A54h
MBC1_DOM13_MEM0_BLK_CFG_W6	1_3A58h
MBC1_DOM13_MEM0_BLK_CFG_W7	1_3A5Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM13_MEM0_BLK_CFG_W8	1_3A60h
MBC1_DOM13_MEM0_BLK_CFG_W9	1_3A64h
MBC1_DOM13_MEM0_BLK_CFG_W10	1_3A68h
MBC1_DOM13_MEM0_BLK_CFG_W11	1_3A6Ch
MBC1_DOM13_MEM0_BLK_CFG_W12	1_3A70h
MBC1_DOM13_MEM0_BLK_CFG_W13	1_3A74h
MBC1_DOM13_MEM0_BLK_CFG_W14	1_3A78h
MBC1_DOM13_MEM0_BLK_CFG_W15	1_3A7Ch
MBC1_DOM13_MEM1_BLK_CFG_W0	1_3B80h
MBC1_DOM13_MEM2_BLK_CFG_W0	1_3BA8h
MBC1_DOM13_MEM3_BLK_CFG_W0	1_3BD0h
MBC1_DOM14_MEM0_BLK_CFG_W0	1_3C40h
MBC1_DOM14_MEM0_BLK_CFG_W1	1_3C44h
MBC1_DOM14_MEM0_BLK_CFG_W2	1_3C48h
MBC1_DOM14_MEM0_BLK_CFG_W3	1_3C4Ch
MBC1_DOM14_MEM0_BLK_CFG_W4	1_3C50h
MBC1_DOM14_MEM0_BLK_CFG_W5	1_3C54h
MBC1_DOM14_MEM0_BLK_CFG_W6	1_3C58h
MBC1_DOM14_MEM0_BLK_CFG_W7	1_3C5Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM14_MEM0_BLK_CFG_W8	1_3C60h
MBC1_DOM14_MEM0_BLK_CFG_W9	1_3C64h
MBC1_DOM14_MEM0_BLK_CFG_W10	1_3C68h
MBC1_DOM14_MEM0_BLK_CFG_W11	1_3C6Ch
MBC1_DOM14_MEM0_BLK_CFG_W12	1_3C70h
MBC1_DOM14_MEM0_BLK_CFG_W13	1_3C74h
MBC1_DOM14_MEM0_BLK_CFG_W14	1_3C78h
MBC1_DOM14_MEM0_BLK_CFG_W15	1_3C7Ch
MBC1_DOM14_MEM1_BLK_CFG_W0	1_3D80h
MBC1_DOM14_MEM2_BLK_CFG_W0	1_3DA8h
MBC1_DOM14_MEM3_BLK_CFG_W0	1_3DD0h
MBC1_DOM15_MEM0_BLK_CFG_W0	1_3E40h
MBC1_DOM15_MEM0_BLK_CFG_W1	1_3E44h
MBC1_DOM15_MEM0_BLK_CFG_W2	1_3E48h
MBC1_DOM15_MEM0_BLK_CFG_W3	1_3E4Ch
MBC1_DOM15_MEM0_BLK_CFG_W4	1_3E50h
MBC1_DOM15_MEM0_BLK_CFG_W5	1_3E54h
MBC1_DOM15_MEM0_BLK_CFG_W6	1_3E58h
MBC1_DOM15_MEM0_BLK_CFG_W7	1_3E5Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM15_MEM0_BLK_CFG_W8	1_3E60h
MBC1_DOM15_MEM0_BLK_CFG_W9	1_3E64h
MBC1_DOM15_MEM0_BLK_CFG_W10	1_3E68h
MBC1_DOM15_MEM0_BLK_CFG_W11	1_3E6Ch
MBC1_DOM15_MEM0_BLK_CFG_W12	1_3E70h
MBC1_DOM15_MEM0_BLK_CFG_W13	1_3E74h
MBC1_DOM15_MEM0_BLK_CFG_W14	1_3E78h
MBC1_DOM15_MEM0_BLK_CFG_W15	1_3E7Ch
MBC1_DOM15_MEM1_BLK_CFG_W0	1_3F80h
MBC1_DOM15_MEM2_BLK_CFG_W0	1_3FA8h
MBC1_DOM15_MEM3_BLK_CFG_W0	1_3FD0h

Function

MBC[m]_DOM[d]_MEM[s]_BLK_CFG_W[w], where,

- m - mbc index
- d - domain index
- s - memory slave index
- w - word index

These registers are read/write for both the alternate view of the NSE bit and the 3-bit MBACSEL field. This is an array of 4 bit fields defining the block configuration for the given submemory. Each 4-bit field includes an alternative view of the associated NSE bit plus a 3-bit access control select that selects the global access control value that applies to the referenced memory block.

For a given memory block, B, if the value programmed in the MBACSELn field selects a locked MBC_MEMN_GLBACr register, the MBACSEL field is locked until the next reset. Depending on BLK_CFG_W, the NSEn/MBASELn fields correspond to different blocks. The following table describes the relationship between W (word index) and B (block number).

MEM0 supports up to 512 blocks ; W0 - W63

MEM1-3 supports up to 64 block ; W0 - W7

NOTE

This register is shown with fields such that it covers entire 32 bit register. However, actual fields may be less if number of blocks are such that fields don't align to 32 bits. These absent fields may be shown but user can refer to table below to deduce actual fields and treat absent fields as reserved.

Table 308. Block Config Word to Block Number Relationship

Block Config Word	NSE7/ MBACSEL7	NSE6/ MBACSEL6	NSE5/ MBACSEL5	NSE4/ MBACSEL4	NSE3/ MBACSEL3	NSE2/ MBACSEL2	NSE1/ MBACSEL1	NSE0/ MBACSEL0
W0	block 7	block 6	block 5	block 4	block 3	block 2	block 1	block 0
W1	block 15	block 14	block 13	block 12	block 11	block 10	block 9	block 8
W2	block 23	block 22	block 21	block 20	block 19	block 18	block 17	block 16
W3	block 31	block 30	block 29	block 28	block 27	block 26	block 25	block 24
W4	block 39	block 38	block 37	block 36	block 35	block 34	block 33	block 32
W5	block 47	block 46	block 45	block 44	block 43	block 42	block 41	block 40
W6	block 55	block 54	block 53	block 52	block 51	block 50	block 49	block 48
W7	block 63	block 62	block 61	block 60	block 59	block 58	block 57	block 56
For MEM0, block 64 - block 511								
W8	block 71	block 70	block 69	block 68	block 67	block 66	block 65	block 64
. . .								
W63	block 511	block 510	block 509	block 508	block 507	block 506	block 505	block 504

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NSE7	MBACSEL7			NSE6	MBACSEL6			NSE5	MBACSEL5			NSE4	MBACSEL4		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NSE3	MBACSEL3			NSE2	MBACSEL2			NSE1	MBACSEL1			NSE0	MBACSEL0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 NSE7	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
30-28 MBACSEL7	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL7 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
27 NSE6	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
26-24 MBACSEL6	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL6 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
23 NSE5	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
22-20 MBACSEL5	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL5 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
19 NSE4	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
18-16 MBACSEL4	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL4 = r and MBC_MEMN_GLBACr[LK] = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
15 NSE3	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
14-12 MBACSEL3	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL3 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
11 NSE2	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
10-8	Memory Block Access Control Select for block B

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBACSEL2	This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL2 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
7 NSE1	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
6-4 MBACSEL1	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL1 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
3 NSE0	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
2-0 MBACSEL0	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL0 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B

43.8.2.31 MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W0 - MBC1_DOM15_MEM3_BLK_NSE_W0)

Offset

Register	Offset
MBC0_DOM0_MEM0_BLK_NSE_W0	1_0140h
MBC0_DOM0_MEM0_BLK_NSE_W1	1_0144h
MBC0_DOM0_MEM0_BLK_NSE_W2	1_0148h
MBC0_DOM0_MEM0_BLK_NSE_W3	1_014Ch
MBC0_DOM0_MEM1_BLK_NSE_W0	1_01A0h
MBC0_DOM0_MEM2_BLK_NSE_W0	1_01C8h
MBC0_DOM0_MEM3_BLK_NSE_W0	1_01F0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM1_MEM0_BLK_NSE_W0	1_0340h
MBC0_DOM1_MEM0_BLK_NSE_W1	1_0344h
MBC0_DOM1_MEM0_BLK_NSE_W2	1_0348h
MBC0_DOM1_MEM0_BLK_NSE_W3	1_034Ch
MBC0_DOM1_MEM1_BLK_NSE_W0	1_03A0h
MBC0_DOM1_MEM2_BLK_NSE_W0	1_03C8h
MBC0_DOM1_MEM3_BLK_NSE_W0	1_03F0h
MBC0_DOM2_MEM0_BLK_NSE_W0	1_0540h
MBC0_DOM2_MEM0_BLK_NSE_W1	1_0544h
MBC0_DOM2_MEM0_BLK_NSE_W2	1_0548h
MBC0_DOM2_MEM0_BLK_NSE_W3	1_054Ch
MBC0_DOM2_MEM1_BLK_NSE_W0	1_05A0h
MBC0_DOM2_MEM2_BLK_NSE_W0	1_05C8h
MBC0_DOM2_MEM3_BLK_NSE_W0	1_05F0h
MBC0_DOM3_MEM0_BLK_NSE_W0	1_0740h
MBC0_DOM3_MEM0_BLK_NSE_W1	1_0744h
MBC0_DOM3_MEM0_BLK_NSE_W2	1_0748h
MBC0_DOM3_MEM0_BLK_NSE_W3	1_074Ch
MBC0_DOM3_MEM1_BLK_NSE_W0	1_07A0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM3_MEM2_BLK_NSE_W0	1_07C8h
MBC0_DOM3_MEM3_BLK_NSE_W0	1_07F0h
MBC0_DOM4_MEM0_BLK_NSE_W0	1_0940h
MBC0_DOM4_MEM0_BLK_NSE_W1	1_0944h
MBC0_DOM4_MEM0_BLK_NSE_W2	1_0948h
MBC0_DOM4_MEM0_BLK_NSE_W3	1_094Ch
MBC0_DOM4_MEM1_BLK_NSE_W0	1_09A0h
MBC0_DOM4_MEM2_BLK_NSE_W0	1_09C8h
MBC0_DOM4_MEM3_BLK_NSE_W0	1_09F0h
MBC0_DOM5_MEM0_BLK_NSE_W0	1_0B40h
MBC0_DOM5_MEM0_BLK_NSE_W1	1_0B44h
MBC0_DOM5_MEM0_BLK_NSE_W2	1_0B48h
MBC0_DOM5_MEM0_BLK_NSE_W3	1_0B4Ch
MBC0_DOM5_MEM1_BLK_NSE_W0	1_0BA0h
MBC0_DOM5_MEM2_BLK_NSE_W0	1_0BC8h
MBC0_DOM5_MEM3_BLK_NSE_W0	1_0BF0h
MBC0_DOM6_MEM0_BLK_NSE_W0	1_0D40h
MBC0_DOM6_MEM0_BLK_NSE_W1	1_0D44h
MBC0_DOM6_MEM0_BLK_NSE_W2	1_0D48h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM6_MEM0_BLK_NSE_W3	1_0D4Ch
MBC0_DOM6_MEM1_BLK_NSE_W0	1_0DA0h
MBC0_DOM6_MEM2_BLK_NSE_W0	1_0DC8h
MBC0_DOM6_MEM3_BLK_NSE_W0	1_0DF0h
MBC0_DOM7_MEM0_BLK_NSE_W0	1_0F40h
MBC0_DOM7_MEM0_BLK_NSE_W1	1_0F44h
MBC0_DOM7_MEM0_BLK_NSE_W2	1_0F48h
MBC0_DOM7_MEM0_BLK_NSE_W3	1_0F4Ch
MBC0_DOM7_MEM1_BLK_NSE_W0	1_0FA0h
MBC0_DOM7_MEM2_BLK_NSE_W0	1_0FC8h
MBC0_DOM7_MEM3_BLK_NSE_W0	1_0FF0h
MBC0_DOM8_MEM0_BLK_NSE_W0	1_1140h
MBC0_DOM8_MEM0_BLK_NSE_W1	1_1144h
MBC0_DOM8_MEM0_BLK_NSE_W2	1_1148h
MBC0_DOM8_MEM0_BLK_NSE_W3	1_114Ch
MBC0_DOM8_MEM1_BLK_NSE_W0	1_11A0h
MBC0_DOM8_MEM2_BLK_NSE_W0	1_11C8h
MBC0_DOM8_MEM3_BLK_NSE_W0	1_11F0h
MBC0_DOM9_MEM0_BLK_NSE_W0	1_1340h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM9_MEM0_BLK_NSE_W1	1_1344h
MBC0_DOM9_MEM0_BLK_NSE_W2	1_1348h
MBC0_DOM9_MEM0_BLK_NSE_W3	1_134Ch
MBC0_DOM9_MEM1_BLK_NSE_W0	1_13A0h
MBC0_DOM9_MEM2_BLK_NSE_W0	1_13C8h
MBC0_DOM9_MEM3_BLK_NSE_W0	1_13F0h
MBC0_DOM10_MEM0_BLK_NSE_W0	1_1540h
MBC0_DOM10_MEM0_BLK_NSE_W1	1_1544h
MBC0_DOM10_MEM0_BLK_NSE_W2	1_1548h
MBC0_DOM10_MEM0_BLK_NSE_W3	1_154Ch
MBC0_DOM10_MEM1_BLK_NSE_W0	1_15A0h
MBC0_DOM10_MEM2_BLK_NSE_W0	1_15C8h
MBC0_DOM10_MEM3_BLK_NSE_W0	1_15F0h
MBC0_DOM11_MEM0_BLK_NSE_W0	1_1740h
MBC0_DOM11_MEM0_BLK_NSE_W1	1_1744h
MBC0_DOM11_MEM0_BLK_NSE_W2	1_1748h
MBC0_DOM11_MEM0_BLK_NSE_W3	1_174Ch
MBC0_DOM11_MEM1_BLK_NSE_W0	1_17A0h
MBC0_DOM11_MEM2_BLK_NSE_W0	1_17C8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM11_MEM3_BLK_NSE_W0	1_17F0h
MBC0_DOM12_MEM0_BLK_NSE_W0	1_1940h
MBC0_DOM12_MEM0_BLK_NSE_W1	1_1944h
MBC0_DOM12_MEM0_BLK_NSE_W2	1_1948h
MBC0_DOM12_MEM0_BLK_NSE_W3	1_194Ch
MBC0_DOM12_MEM1_BLK_NSE_W0	1_19A0h
MBC0_DOM12_MEM2_BLK_NSE_W0	1_19C8h
MBC0_DOM12_MEM3_BLK_NSE_W0	1_19F0h
MBC0_DOM13_MEM0_BLK_NSE_W0	1_1B40h
MBC0_DOM13_MEM0_BLK_NSE_W1	1_1B44h
MBC0_DOM13_MEM0_BLK_NSE_W2	1_1B48h
MBC0_DOM13_MEM0_BLK_NSE_W3	1_1B4Ch
MBC0_DOM13_MEM1_BLK_NSE_W0	1_1BA0h
MBC0_DOM13_MEM2_BLK_NSE_W0	1_1BC8h
MBC0_DOM13_MEM3_BLK_NSE_W0	1_1BF0h
MBC0_DOM14_MEM0_BLK_NSE_W0	1_1D40h
MBC0_DOM14_MEM0_BLK_NSE_W1	1_1D44h
MBC0_DOM14_MEM0_BLK_NSE_W2	1_1D48h
MBC0_DOM14_MEM0_BLK_NSE_W3	1_1D4Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM14_MEM1_BLK_NSE_W0	1_1DA0h
MBC0_DOM14_MEM2_BLK_NSE_W0	1_1DC8h
MBC0_DOM14_MEM3_BLK_NSE_W0	1_1DF0h
MBC0_DOM15_MEM0_BLK_NSE_W0	1_1F40h
MBC0_DOM15_MEM0_BLK_NSE_W1	1_1F44h
MBC0_DOM15_MEM0_BLK_NSE_W2	1_1F48h
MBC0_DOM15_MEM0_BLK_NSE_W3	1_1F4Ch
MBC0_DOM15_MEM1_BLK_NSE_W0	1_1FA0h
MBC0_DOM15_MEM2_BLK_NSE_W0	1_1FC8h
MBC0_DOM15_MEM3_BLK_NSE_W0	1_1FF0h
MBC1_DOM0_MEM0_BLK_NSE_W0	1_2140h
MBC1_DOM0_MEM0_BLK_NSE_W1	1_2144h
MBC1_DOM0_MEM0_BLK_NSE_W2	1_2148h
MBC1_DOM0_MEM0_BLK_NSE_W3	1_214Ch
MBC1_DOM0_MEM1_BLK_NSE_W0	1_21A0h
MBC1_DOM0_MEM2_BLK_NSE_W0	1_21C8h
MBC1_DOM0_MEM3_BLK_NSE_W0	1_21F0h
MBC1_DOM1_MEM0_BLK_NSE_W0	1_2340h
MBC1_DOM1_MEM0_BLK_NSE_W1	1_2344h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM1_MEM0_BLK_NSE_W2	1_2348h
MBC1_DOM1_MEM0_BLK_NSE_W3	1_234Ch
MBC1_DOM1_MEM1_BLK_NSE_W0	1_23A0h
MBC1_DOM1_MEM2_BLK_NSE_W0	1_23C8h
MBC1_DOM1_MEM3_BLK_NSE_W0	1_23F0h
MBC1_DOM2_MEM0_BLK_NSE_W0	1_2540h
MBC1_DOM2_MEM0_BLK_NSE_W1	1_2544h
MBC1_DOM2_MEM0_BLK_NSE_W2	1_2548h
MBC1_DOM2_MEM0_BLK_NSE_W3	1_254Ch
MBC1_DOM2_MEM1_BLK_NSE_W0	1_25A0h
MBC1_DOM2_MEM2_BLK_NSE_W0	1_25C8h
MBC1_DOM2_MEM3_BLK_NSE_W0	1_25F0h
MBC1_DOM3_MEM0_BLK_NSE_W0	1_2740h
MBC1_DOM3_MEM0_BLK_NSE_W1	1_2744h
MBC1_DOM3_MEM0_BLK_NSE_W2	1_2748h
MBC1_DOM3_MEM0_BLK_NSE_W3	1_274Ch
MBC1_DOM3_MEM1_BLK_NSE_W0	1_27A0h
MBC1_DOM3_MEM2_BLK_NSE_W0	1_27C8h
MBC1_DOM3_MEM3_BLK_NSE_W0	1_27F0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM4_MEM0_BLK_NSE_W0	1_2940h
MBC1_DOM4_MEM0_BLK_NSE_W1	1_2944h
MBC1_DOM4_MEM0_BLK_NSE_W2	1_2948h
MBC1_DOM4_MEM0_BLK_NSE_W3	1_294Ch
MBC1_DOM4_MEM1_BLK_NSE_W0	1_29A0h
MBC1_DOM4_MEM2_BLK_NSE_W0	1_29C8h
MBC1_DOM4_MEM3_BLK_NSE_W0	1_29F0h
MBC1_DOM5_MEM0_BLK_NSE_W0	1_2B40h
MBC1_DOM5_MEM0_BLK_NSE_W1	1_2B44h
MBC1_DOM5_MEM0_BLK_NSE_W2	1_2B48h
MBC1_DOM5_MEM0_BLK_NSE_W3	1_2B4Ch
MBC1_DOM5_MEM1_BLK_NSE_W0	1_2BA0h
MBC1_DOM5_MEM2_BLK_NSE_W0	1_2BC8h
MBC1_DOM5_MEM3_BLK_NSE_W0	1_2BF0h
MBC1_DOM6_MEM0_BLK_NSE_W0	1_2D40h
MBC1_DOM6_MEM0_BLK_NSE_W1	1_2D44h
MBC1_DOM6_MEM0_BLK_NSE_W2	1_2D48h
MBC1_DOM6_MEM0_BLK_NSE_W3	1_2D4Ch
MBC1_DOM6_MEM1_BLK_NSE_W0	1_2DA0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM6_MEM2_BLK_NSE_W0	1_2DC8h
MBC1_DOM6_MEM3_BLK_NSE_W0	1_2DF0h
MBC1_DOM7_MEM0_BLK_NSE_W0	1_2F40h
MBC1_DOM7_MEM0_BLK_NSE_W1	1_2F44h
MBC1_DOM7_MEM0_BLK_NSE_W2	1_2F48h
MBC1_DOM7_MEM0_BLK_NSE_W3	1_2F4Ch
MBC1_DOM7_MEM1_BLK_NSE_W0	1_2FA0h
MBC1_DOM7_MEM2_BLK_NSE_W0	1_2FC8h
MBC1_DOM7_MEM3_BLK_NSE_W0	1_2FF0h
MBC1_DOM8_MEM0_BLK_NSE_W0	1_3140h
MBC1_DOM8_MEM0_BLK_NSE_W1	1_3144h
MBC1_DOM8_MEM0_BLK_NSE_W2	1_3148h
MBC1_DOM8_MEM0_BLK_NSE_W3	1_314Ch
MBC1_DOM8_MEM1_BLK_NSE_W0	1_31A0h
MBC1_DOM8_MEM2_BLK_NSE_W0	1_31C8h
MBC1_DOM8_MEM3_BLK_NSE_W0	1_31F0h
MBC1_DOM9_MEM0_BLK_NSE_W0	1_3340h
MBC1_DOM9_MEM0_BLK_NSE_W1	1_3344h
MBC1_DOM9_MEM0_BLK_NSE_W2	1_3348h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM9_MEM0_BLK_NSE_W3	1_334Ch
MBC1_DOM9_MEM1_BLK_NSE_W0	1_33A0h
MBC1_DOM9_MEM2_BLK_NSE_W0	1_33C8h
MBC1_DOM9_MEM3_BLK_NSE_W0	1_33F0h
MBC1_DOM10_MEM0_BLK_NSE_W0	1_3540h
MBC1_DOM10_MEM0_BLK_NSE_W1	1_3544h
MBC1_DOM10_MEM0_BLK_NSE_W2	1_3548h
MBC1_DOM10_MEM0_BLK_NSE_W3	1_354Ch
MBC1_DOM10_MEM1_BLK_NSE_W0	1_35A0h
MBC1_DOM10_MEM2_BLK_NSE_W0	1_35C8h
MBC1_DOM10_MEM3_BLK_NSE_W0	1_35F0h
MBC1_DOM11_MEM0_BLK_NSE_W0	1_3740h
MBC1_DOM11_MEM0_BLK_NSE_W1	1_3744h
MBC1_DOM11_MEM0_BLK_NSE_W2	1_3748h
MBC1_DOM11_MEM0_BLK_NSE_W3	1_374Ch
MBC1_DOM11_MEM1_BLK_NSE_W0	1_37A0h
MBC1_DOM11_MEM2_BLK_NSE_W0	1_37C8h
MBC1_DOM11_MEM3_BLK_NSE_W0	1_37F0h
MBC1_DOM12_MEM0_BLK_NSE_W0	1_3940h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM12_MEM0_BLK_NSE_W1	1_3944h
MBC1_DOM12_MEM0_BLK_NSE_W2	1_3948h
MBC1_DOM12_MEM0_BLK_NSE_W3	1_394Ch
MBC1_DOM12_MEM1_BLK_NSE_W0	1_39A0h
MBC1_DOM12_MEM2_BLK_NSE_W0	1_39C8h
MBC1_DOM12_MEM3_BLK_NSE_W0	1_39F0h
MBC1_DOM13_MEM0_BLK_NSE_W0	1_3B40h
MBC1_DOM13_MEM0_BLK_NSE_W1	1_3B44h
MBC1_DOM13_MEM0_BLK_NSE_W2	1_3B48h
MBC1_DOM13_MEM0_BLK_NSE_W3	1_3B4Ch
MBC1_DOM13_MEM1_BLK_NSE_W0	1_3BA0h
MBC1_DOM13_MEM2_BLK_NSE_W0	1_3BC8h
MBC1_DOM13_MEM3_BLK_NSE_W0	1_3BF0h
MBC1_DOM14_MEM0_BLK_NSE_W0	1_3D40h
MBC1_DOM14_MEM0_BLK_NSE_W1	1_3D44h
MBC1_DOM14_MEM0_BLK_NSE_W2	1_3D48h
MBC1_DOM14_MEM0_BLK_NSE_W3	1_3D4Ch
MBC1_DOM14_MEM1_BLK_NSE_W0	1_3DA0h
MBC1_DOM14_MEM2_BLK_NSE_W0	1_3DC8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM14_MEM3_BLK_NSE_W0	1_3DF0h
MBC1_DOM15_MEM0_BLK_NSE_W0	1_3F40h
MBC1_DOM15_MEM0_BLK_NSE_W1	1_3F44h
MBC1_DOM15_MEM0_BLK_NSE_W2	1_3F48h
MBC1_DOM15_MEM0_BLK_NSE_W3	1_3F4Ch
MBC1_DOM15_MEM1_BLK_NSE_W0	1_3FA0h
MBC1_DOM15_MEM2_BLK_NSE_W0	1_3FC8h
MBC1_DOM15_MEM3_BLK_NSE_W0	1_3FF0h

Function

MBC[m]_DOM[d]_MEM[s]_BLK_NSE_W[w], where,

- m - mbc index
- d - domain index
- s - memory slave index
- w - word index

This is the bitmap of the individual NSE bits for the memory block.

NOTE

This register is shown with fields such that it covers entire 32 bit register. However, actual fields may be less if number of blocks are such that fields don't align to 32 bits. These absent fields may be shown but user can refer to table below to deduce actual fields and treat absent fields as reserved.

The following table describes the relationship between W (word index) and B (block number).

Table 309. Block Config Word to Block Number Relationship

Block NSE Word	BIT31	BIT30-BIT1	BIT0
W0	block 31	block 30 - block 1	block 0
W1	block 63	block 62 - block 33	block 32
For MEM0, block 64 - block 511			
W2	block 95	block 94 - block 65	block 64

Table continues on the next page...

Table 309. Block Config Word to Block Number Relationship (continued)

Block NSE Word	BIT31	BIT30-BIT1	BIT0
		. . .	
W15	block 511	block 510 - block 481	block 480

The NSE bitmap can be accessed directly by register reads and writes to these locations, or "indirectly" through writes to NSE_SET, NSE_CLR, NSE_CLR_ALL, and BLK_CFG[NSEn] registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BIT31	BIT30	BIT29	BIT28	BIT27	BIT26	BIT25	BIT24	BIT23	BIT22	BIT21	BIT20	BIT19	BIT18	BIT17	BIT16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BITb	Bit b NonSecure Enable [b = 0 - 31] 0b - Secure accesses to block B are based on corresponding MBACSEL field in register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).

43.8.2.32 MBC Global Access Control (MBC1_MEMN_GLBAC0)

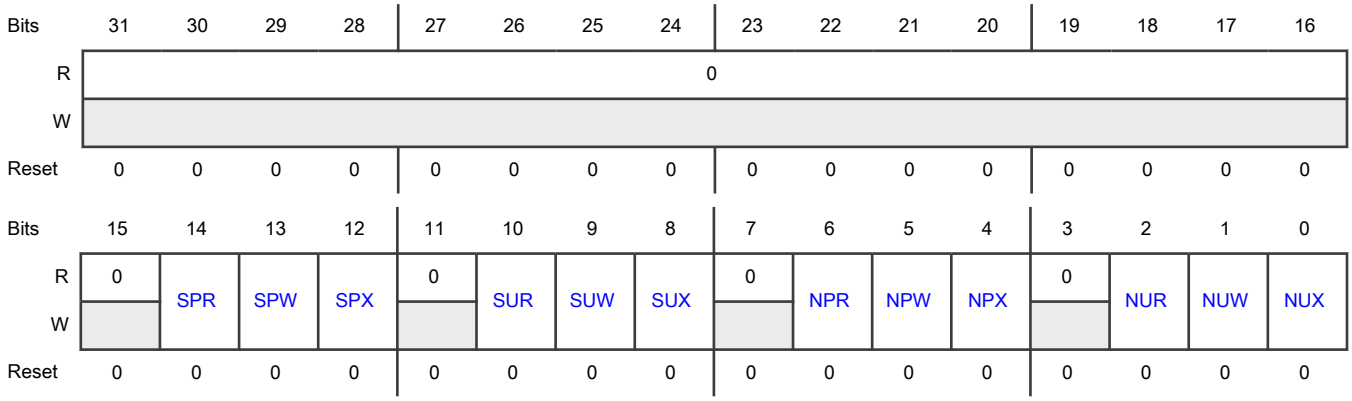
Offset

Register	Offset
MBC1_MEMN_GLBAC0	1_2020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9	SecureUser Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
SUW	SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.33 MBC Global Access Control (MBC1_MEMN_GLBAC1 - MBC1_MEMN_GLBAC7)

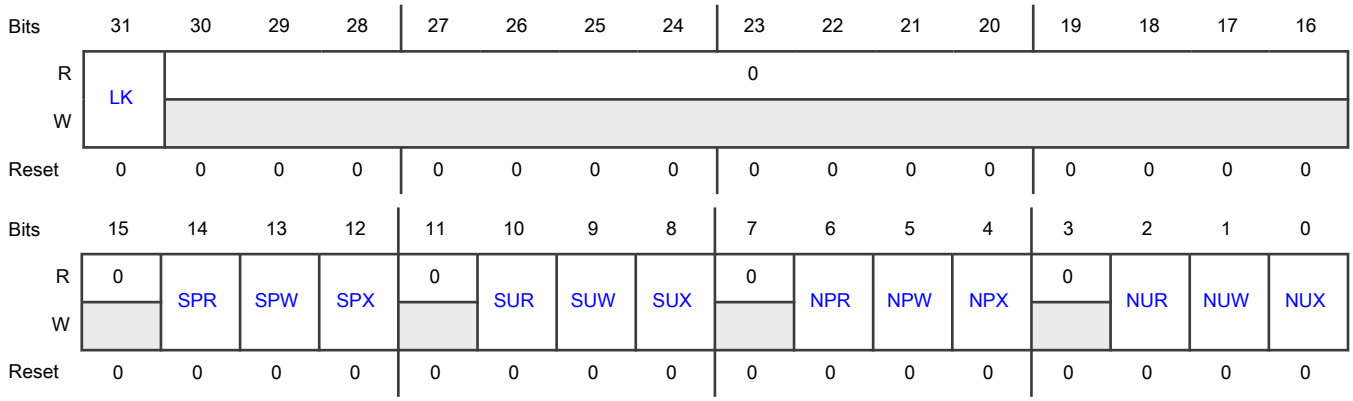
Offset

Register	Offset
MBC1_MEMN_GLBAC1	1_2024h
MBC1_MEMN_GLBAC2	1_2028h
MBC1_MEMN_GLBAC3	1_202Ch
MBC1_MEMN_GLBAC4	1_2030h
MBC1_MEMN_GLBAC5	1_2034h
MBC1_MEMN_GLBAC6	1_2038h
MBC1_MEMN_GLBAC7	1_203Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these control access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

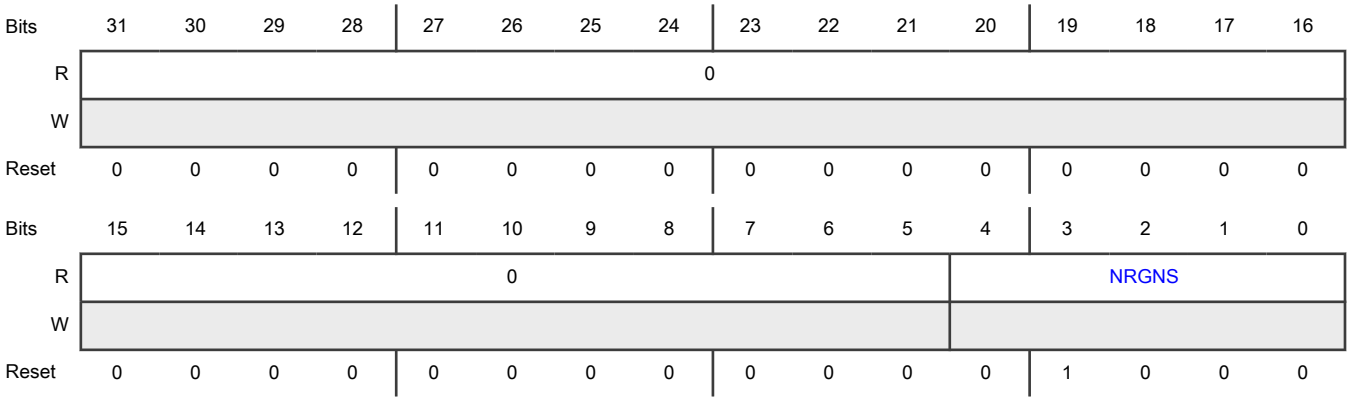
43.8.2.34 MRC Global Configuration Register (MRC0_GLBCFG - MRC1_GLBCFG)

Offset

Register	Offset
MRC0_GLBCFG	1_4000h
MRC1_GLBCFG	1_5000h

Function
This read-only register describes the number of region descriptors for this memory.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 NRGNS	Number of regions [1-16]

43.8.2.35 MRC NonSecure Enable Region Indirect (MRC0_NSE_RGN_INDIRECT - MRC6_NSE_RGN_INDIRECT)

Offset

Register	Offset
MRC0_NSE_RGN_INDIRECT	1_4010h
MRC1_NSE_RGN_INDIRECT	1_5010h

Table continues on the next page...

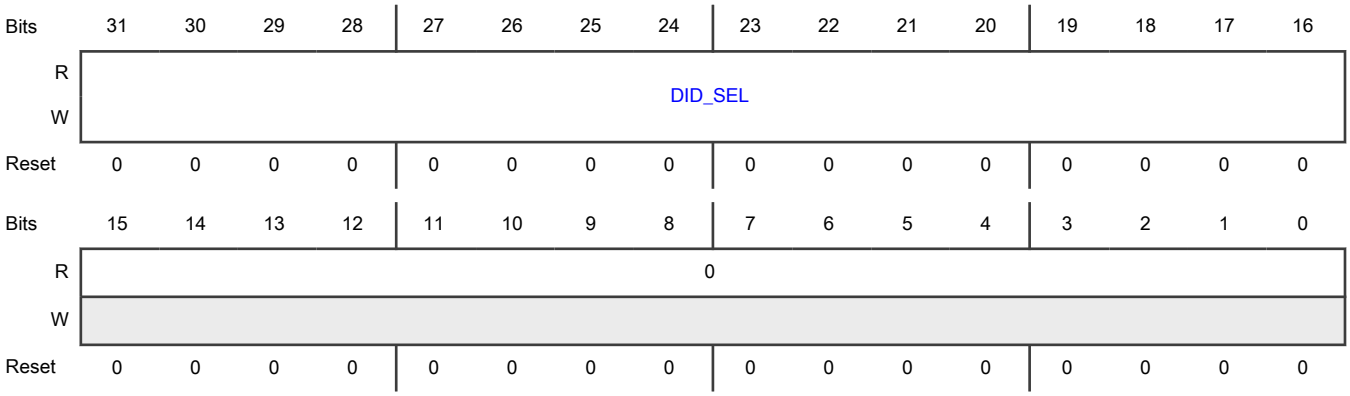
Table continued from the previous page...

Register	Offset
MRC2_NSE_RGN_INDIRECT	1_6010h
MRC3_NSE_RGN_INDIRECT	1_7010h
MRC4_NSE_RGN_INDIRECT	1_8010h
MRC5_NSE_RGN_INDIRECT	1_9010h
MRC6_NSE_RGN_INDIRECT	1_A010h

Function

This R/W register defines the selected domains that are affected by writes to the NSE_RGN_SET and NSE_RGN_CLR registers.

Diagram



Fields

Field	Function
31-16 DID_SEL	DID Select Destination domain bitmap select. • Bit [31] = 1, update domain 15 • Bit [30] = 1, update domain 14 • Bit [29] = 1, update domain 13 • Bit [28] = 1, update domain 12 • Bit [27] = 1, update domain 11

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Bit [26] = 1, update domain 10 • Bit [25] = 1, update domain 9 • Bit [24] = 1, update domain 8 • Bit [23] = 1, update domain 7 • Bit [22] = 1, update domain 6 • Bit [21] = 1, update domain 5 • Bit [20] = 1, update domain 4 • Bit [19] = 1, update domain 3 • Bit [18] = 1, update domain 2 • Bit [17] = 1, update domain 1 • Bit [16] = 1, update domain 0
15-0 —	Reserved

43.8.2.36 MRC NonSecure Enable Region Set (MRC0_NSE_RGN_SET - MRC6_NSE_RGN_SET)

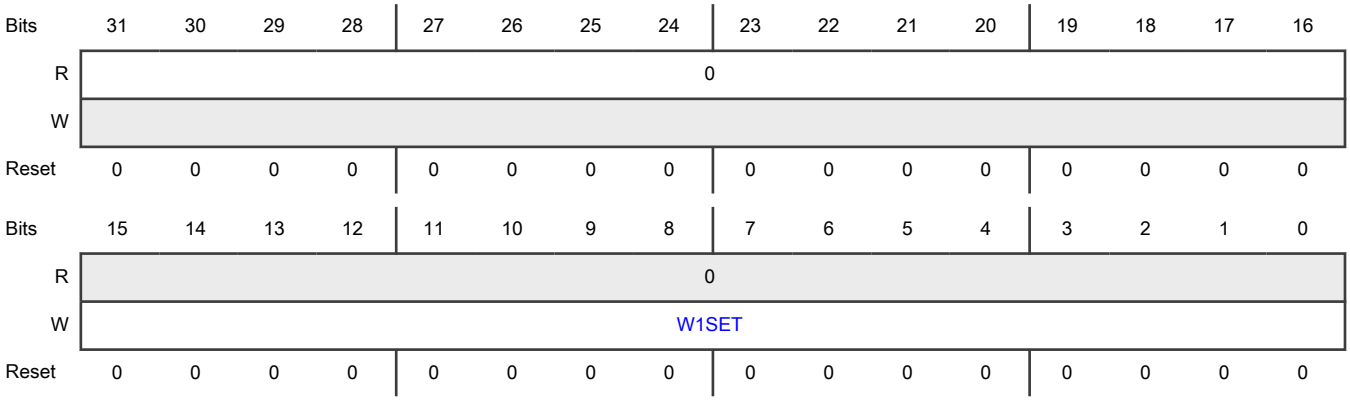
Offset

Register	Offset
MRC0_NSE_RGN_SET	1_4014h
MRC1_NSE_RGN_SET	1_5014h
MRC2_NSE_RGN_SET	1_6014h
MRC3_NSE_RGN_SET	1_7014h
MRC4_NSE_RGN_SET	1_8014h
MRC5_NSE_RGN_SET	1_9014h
MRC6_NSE_RGN_SET	1_A014h

Function

A write to this location sets the appropriate NSE bits in region descriptors for the selected domain in MRC NonSecure Enable Region Indirect register. This register reads as 0.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 W1SET	Write-1 Set • Bit [15] = Set the NSE bits for RGD15 • Bit [14] = Set the NSE bits for RGD14 • Bit [13] = Set the NSE bits for RGD13 • Bit [12] = Set the NSE bits for RGD12 • Bit [11] = Set the NSE bits for RGD11 • Bit [10] = Set the NSE bits for RGD10 • Bit [9] = Set the NSE bits for RGD9 • Bit [8] = Set the NSE bits for RGD8 • Bit [7] = Set the NSE bits for RGD7 • Bit [6] = Set the NSE bits for RGD6 • Bit [5] = Set the NSE bits for RGD5 • Bit [4] = Set the NSE bits for RGD4 • Bit [3] = Set the NSE bits for RGD3 • Bit [2] = Set the NSE bits for RGD2 • Bit [1] = Set the NSE bits for RGD1 • Bit [0] = Set the NSE bits for RGD0

43.8.2.37 MRC NonSecure Enable Region Clear (MRC0_NSE_RGN_CLR - MRC6_NSE_RGN_CLR)

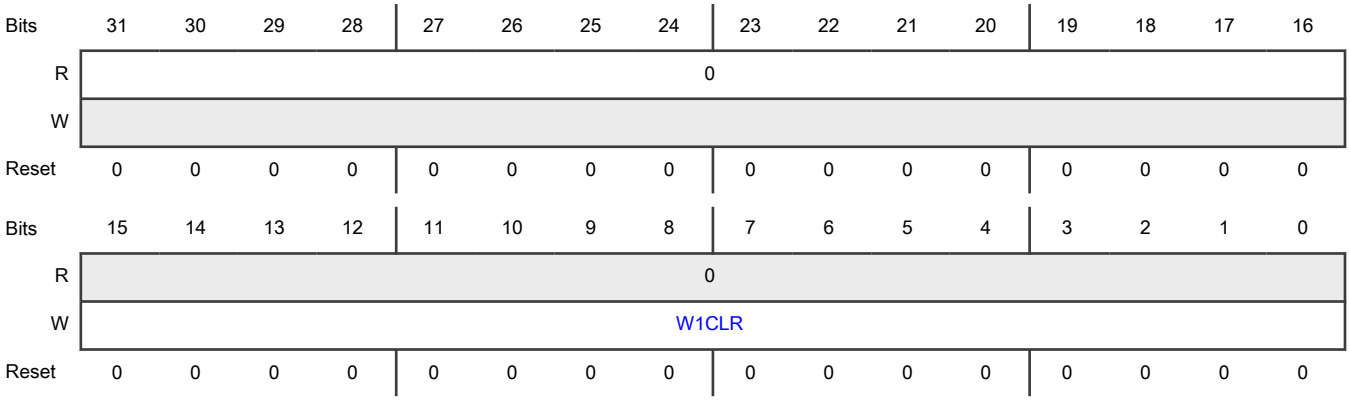
Offset

Register	Offset
MRC0_NSE_RGN_CLR	1_4018h
MRC1_NSE_RGN_CLR	1_5018h
MRC2_NSE_RGN_CLR	1_6018h
MRC3_NSE_RGN_CLR	1_7018h
MRC4_NSE_RGN_CLR	1_8018h
MRC5_NSE_RGN_CLR	1_9018h
MRC6_NSE_RGN_CLR	1_A018h

Function

A write to this location clears the appropriate NSE bits in region descriptors for the selected domain in MRC NonSecure Enable Region Indirect register. This register reads as 0.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 W1CLR	Write-1 Clear • Bit [15] = Clear the NSE bits for RGD15 • Bit [14] = Clear the NSE bits for RGD14 • Bit [13] = Clear the NSE bits for RGD13 • Bit [12] = Clear the NSE bits for RGD12

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Bit [11] = Clear the NSE bits for RGD11 • Bit [10] = Clear the NSE bits for RGD10 • Bit [9] = Clear the NSE bits for RGD9 • Bit [8] = Clear the NSE bits for RGD8 • Bit [7] = Clear the NSE bits for RGD7 • Bit [6] = Clear the NSE bits for RGD6 • Bit [5] = Clear the NSE bits for RGD5 • Bit [4] = Clear the NSE bits for RGD4 • Bit [3] = Clear the NSE bits for RGD3 • Bit [2] = Clear the NSE bits for RGD2 • Bit [1] = Clear the NSE bits for RGD1 • Bit [0] = Clear the NSE bits for RGD0

43.8.2.38 MRC NonSecure Enable Region Clear All (MRC0_NSE_RGN_CLR_ALL - MRC6_NSE_RGN_CLR_ALL)

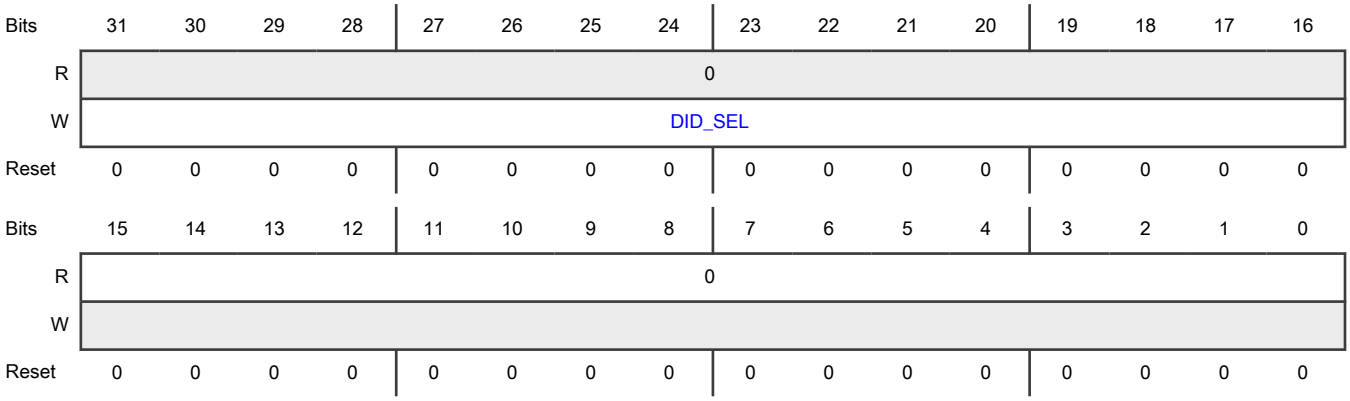
Offset

Register	Offset
MRC0_NSE_RGN_CLR_ALL	1_401Ch
MRC1_NSE_RGN_CLR_ALL	1_501Ch
MRC2_NSE_RGN_CLR_ALL	1_601Ch
MRC3_NSE_RGN_CLR_ALL	1_701Ch
MRC4_NSE_RGN_CLR_ALL	1_801Ch
MRC5_NSE_RGN_CLR_ALL	1_901Ch
MRC6_NSE_RGN_CLR_ALL	1_A01Ch

Function

A write to this register clears all the region NSE bits for the selected domains defined in the write data.

Diagram



Fields

Field	Function
31-16 DID_SEL	DID Select Destination domain bitmap select. • Bit [31] = 1 clear NSE bits for domain 15 • Bit [30] = 1 clear NSE bits for domain 14 • Bit [29] = 1 clear NSE bits for domain 13 • Bit [28] = 1 clear NSE bits for domain 12 • Bit [27] = 1 clear NSE bits for domain 11 • Bit [26] = 1 clear NSE bits for domain 10 • Bit [25] = 1 clear NSE bits for domain 9 • Bit [24] = 1 clear NSE bits for domain 8 • Bit [23] = 1 clear NSE bits for domain 7 • Bit [22] = 1 clear NSE bits for domain 6 • Bit [21] = 1 clear NSE bits for domain 5 • Bit [20] = 1 clear NSE bits for domain 4 • Bit [19] = 1 clear NSE bits for domain 3 • Bit [18] = 1 clear NSE bits for domain 2 • Bit [17] = 1 clear NSE bits for domain 1 • Bit [16] = 1 clear NSE bits for domain 0
15-0 —	Reserved

43.8.2.39 MRC Global Access Control (MRC0_GLBAC0)

Offset

Register	Offset
MRC0_GLBAC0	1_4020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SPR	SPW	SPX	0	SUR	SUW	SUX	0	NPR	NPW	NPX	0	NUR	NUW	NUX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.40 MRC Global Access Control (MRC0_GLBAC1 - MRC0_GLBAC7)

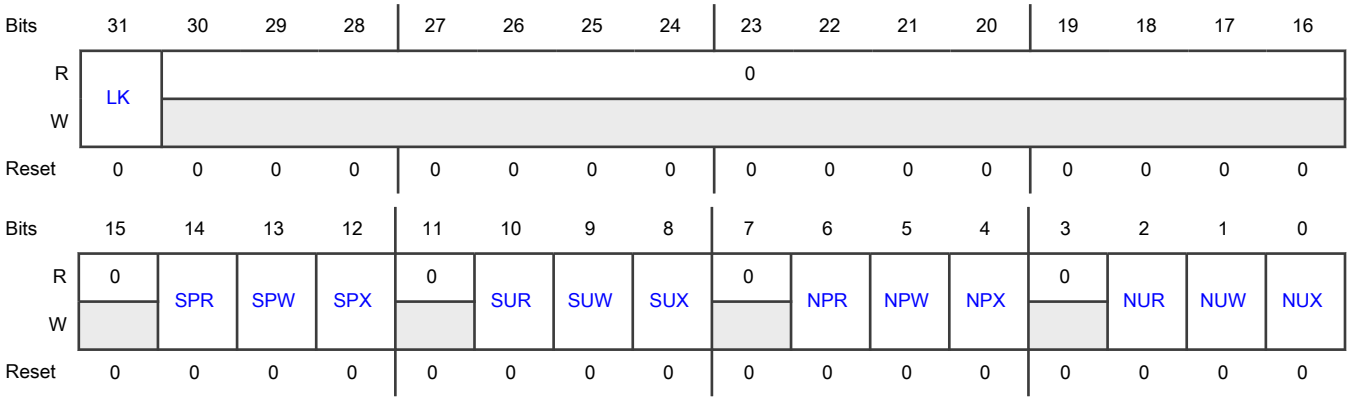
Offset

Register	Offset
MRC0_GLBAC1	1_4024h
MRC0_GLBAC2	1_4028h
MRC0_GLBAC3	1_402Ch
MRC0_GLBAC4	1_4030h
MRC0_GLBAC5	1_4034h
MRC0_GLBAC6	1_4038h
MRC0_GLBAC7	1_403Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked (read-only) and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10	SecureUser Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
SUR	SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.41 MRC Region Descriptor Word 0 (MRC0_DOM0_RGD0_W0 - MRC6_DOM15_RGD15_W0)

Offset

Registers in this array exist only for the following combinations of index values.

Index *m*	Index *d*	Index *r*
0–1	0–15	0–7
2–6	0–15	0–15

Register	Offset
MRCm_DOMd_RGDr_W0	1_4040h + (m × 1000h) + (d × 100h) + (r × 8h)

Function

MRC[m]_DOM[d]_RGD[r]_W[w], where,

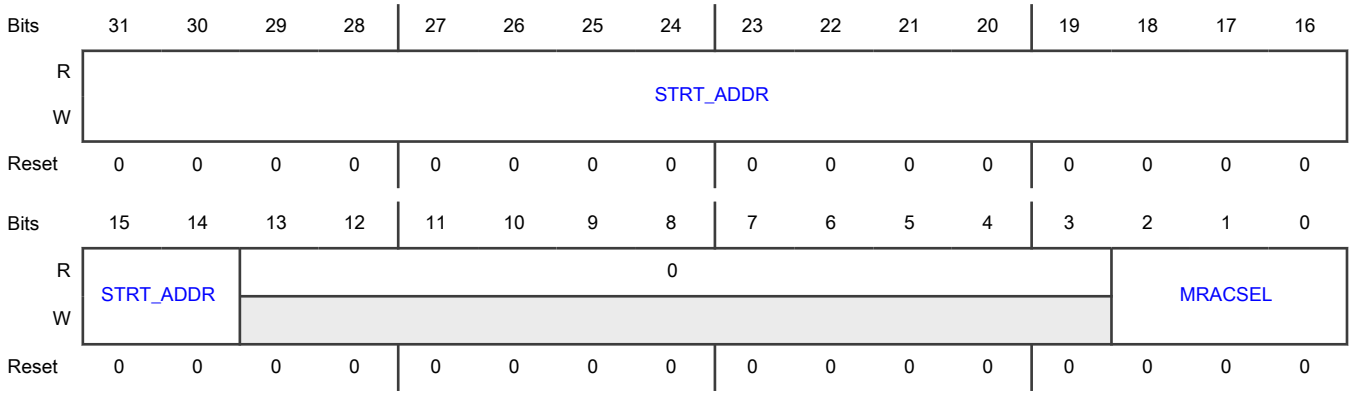
- m - mrc index
- d - domain index
- r - region descriptor index
- w - word index

The MRC[m]_DOM[d]_RGD[r]_W[w] registers provide a 3-dimensional data structure for defining access control policies per domain for each supported memory region.

Memory map/register definition:

There two word-size registers (W[w]) defined in each memory region. There are up to 16 memory region checkers (MRCs), m, each supporting up to 16 memory region descriptors (RGD), r. The number of MRCs is defined by HWCFG0[NMRC].

Diagram



Fields

Field	Function
31-14 STRT_ADDR	Start Address 0-mod-16K physical start address [31:14] of region r. **NOTE** This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1.
13-3 —	Reserved
2-0 MRACSEL	Memory Region Access Control Select This field selects the global access control associated with region r. **NOTE** This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1. 000b - Select MRC_GLBAC0 access control policy 001b - Select MRC_GLBAC1 access control policy 010b - Select MRC_GLBAC2 access control policy 011b - Select MRC_GLBAC3 access control policy 100b - Select MRC_GLBAC4 access control policy 101b - Select MRC_GLBAC5 access control policy 110b - Select MRC_GLBAC6 access control policy 111b - Select MRC_GLBAC7 access control policy

43.8.2.42 MRC Region Descriptor Word 1 (MRC0_DOM0_RGD0_W1 - MRC6_DOM15_RGD15_W1)

Offset

Registers in this array exist only for the following combinations of index values.

Index *m*	Index *d*	Index *r*
0–1	0–15	0–7
2–6	0–15	0–15

Register	Offset
MRCm_DOMd_RGDr_W1	1_4044h + (m × 1000h) + (d × 100h) + (r × 8h)

Function

MRC[m]_DOM[d]_RGD[r]_W[w], where,

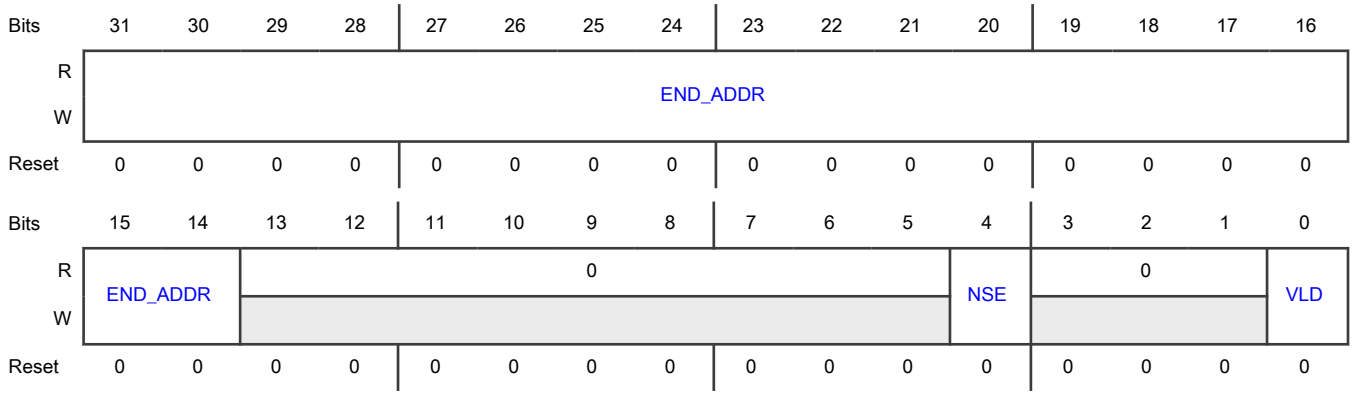
- m - mrc index
- d - domain index
- r - region descriptor index
- w - word index

The MRC[m]_[d]_rgd[r]_w[w] registers provide a 3-dimensional data structure for defining access control policies per domain for each supported memory region.

Memory map/register definition:

There two word-size registers (W[w]) defined in each memory region. There are up to 16 memory region checkers (MRCs), m, each supporting up to 16 region descriptors (RGD), r. The number of MRCs is defined by HWCFG0[NMRC].

Diagram



Fields

Field	Function
31-14 END_ADDR	End Address (16K-1)-mod-16K physical end address [31:14] of region r.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1.
13-5 —	Reserved
4 NSE	NonSecure Enable Alternate view of NSE flag for Region r. 0b - Secure accesses to region r are based on corresponding MRACSEL field in this register (MRCm_DOMd_RGDr_Ww[MRACSEL]), nonsecure accesses to region r are not allowed. 1b - Secure accesses to region r are not allowed, nonsecure accesses to region r are based on corresponding MRACSEL field in this register (MRCm_DOMd_RGDr_Ww[MRACSEL]).
3-1 —	Reserved
0 VLD	Valid If set, this region is valid and accesses are checked for this region. **NOTE** This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1.

43.8.2.43 MRC Region Descriptor NonSecure Enable (MRC0_DOM0_RGD_NSE - MRC1_DOM15_RGD_NSE)

Offset

For m = 0 to 1; d = 0 to 15:

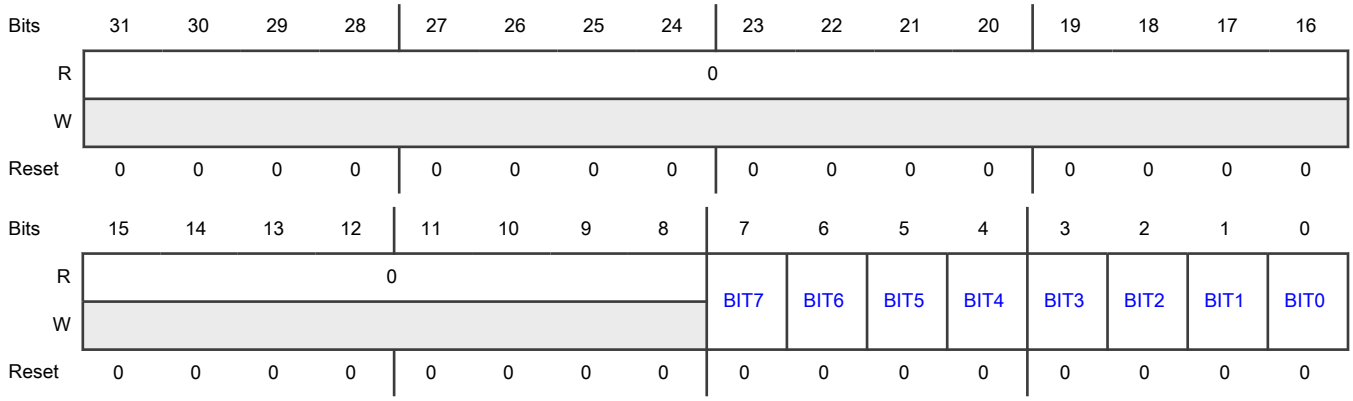
Register	Offset
MRCm_DOMd_RGD_NSE	1_40C0h + (m × 1000h) + (d × 100h)

Function

This register is the bitmap of the individual NSE bits for the region descriptors. Bit n corresponds to region n.

These bits can also be written or "indirectly" through writes to NSE_SET, NSE_CLR, NSE_CLR_ALL, and RGD_W0[NSE] registers.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 BITn	Bit n NonSecure Enable [n = 0 - 15] 0b - Secure accesses to region r are based on corresponding MRACSEL field in this register (MRCm_DOMd_RGDr_Ww[MRACSEL]), nonsecure accesses to region r are not allowed. 1b - Secure accesses to region r are not allowed, nonsecure accesses to region r are based on corresponding MRACSEL field in register (MRCm_DOMd_RGDr_Ww[MRACSEL]).

43.8.2.44 MRC Global Access Control (MRC1_GLBAC0)

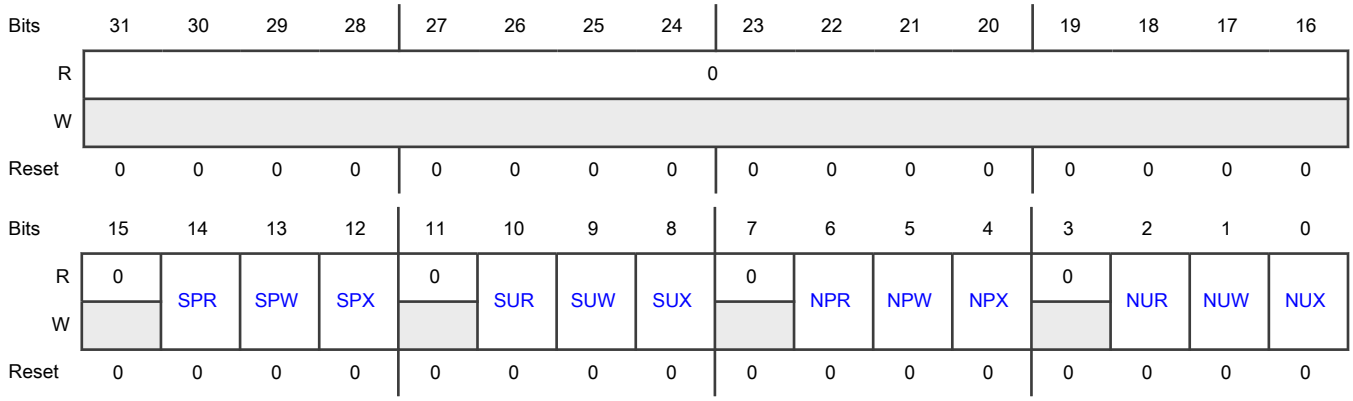
Offset

Register	Offset
MRC1_GLBAC0	1_5020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9	SecureUser Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
SUW	SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.45 MRC Global Access Control (MRC1_GLBAC1 - MRC1_GLBAC7)

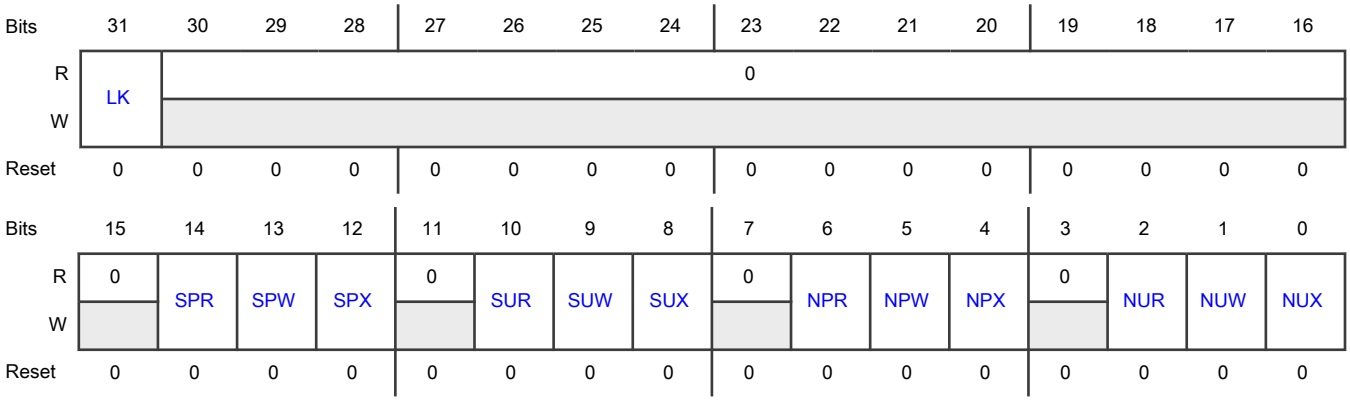
Offset

Register	Offset
MRC1_GLBAC1	1_5024h
MRC1_GLBAC2	1_5028h
MRC1_GLBAC3	1_502Ch
MRC1_GLBAC4	1_5030h
MRC1_GLBAC5	1_5034h
MRC1_GLBAC6	1_5038h
MRC1_GLBAC7	1_503Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked (read-only) and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.46 MRC Global Configuration Register (MRC2_GLBCFG - MRC6_GLBCFG)

Offset

Register	Offset
MRC2_GLBCFG	1_6000h
MRC3_GLBCFG	1_7000h
MRC4_GLBCFG	1_8000h
MRC5_GLBCFG	1_9000h
MRC6_GLBCFG	1_A000h

Function
This read-only register describes the number of region descriptors for this memory.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											NRGNS				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-5 —	Reserved
4-0 NRGNS	Number of regions [1-16]

43.8.2.47 MRC Global Access Control (MRC2_GLBAC0)

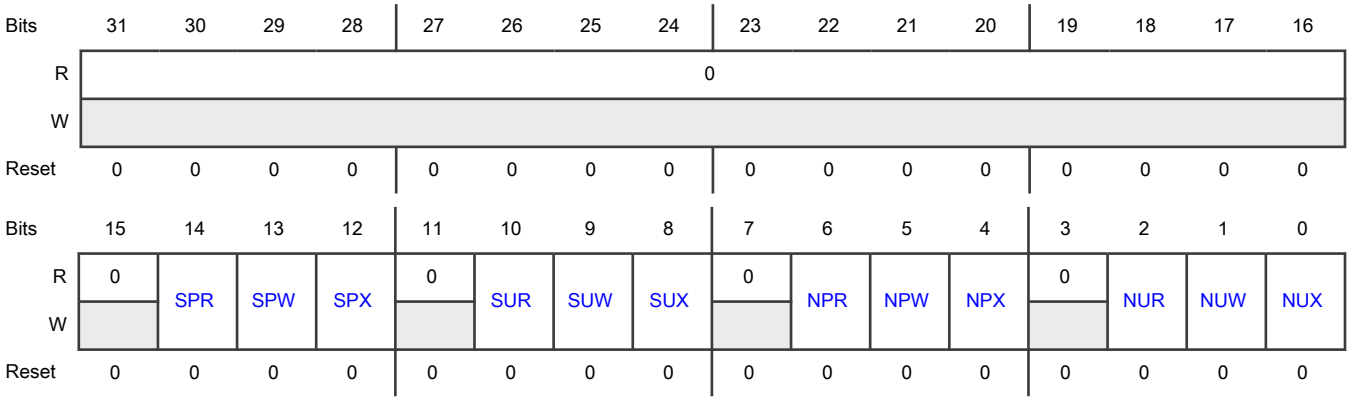
Offset

Register	Offset
MRC2_GLBAC0	1_6020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10	SecureUser Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
SUR	SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.48 MRC Global Access Control (MRC2_GLBAC1 - MRC2_GLBAC7)

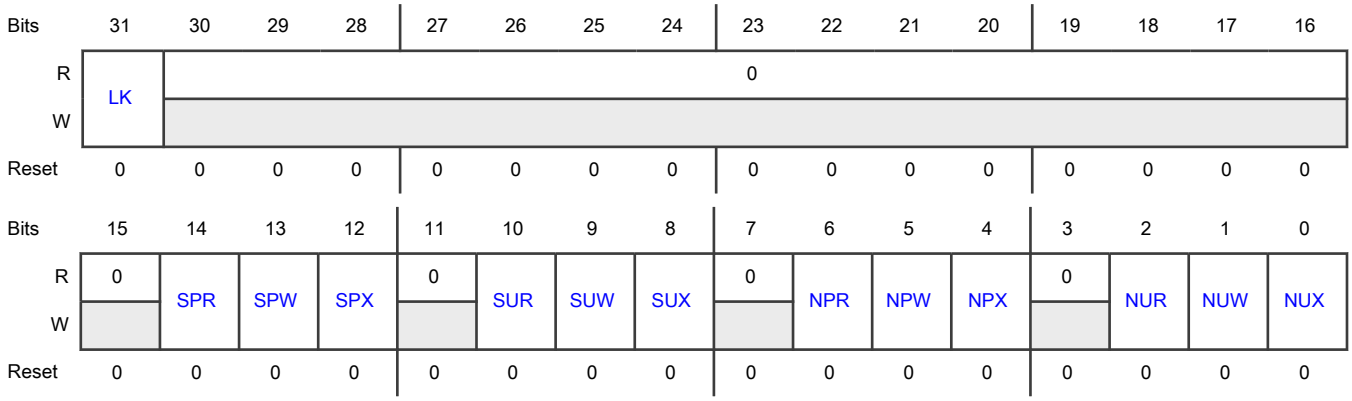
Offset

Register	Offset
MRC2_GLBAC1	1_6024h
MRC2_GLBAC2	1_6028h
MRC2_GLBAC3	1_602Ch
MRC2_GLBAC4	1_6030h
MRC2_GLBAC5	1_6034h
MRC2_GLBAC6	1_6038h
MRC2_GLBAC7	1_603Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked (read-only) and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10	SecureUser Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
SUR	SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.49 MRC Region Descriptor NonSecure Enable (MRC2_DOM0_RGD_NSE - MRC6_DOM15_RGD_NSE)

Offset

For m = 2 to 6; d = 0 to 15:

Register	Offset
MRCm_DOMd_RGD_NSE	1_40C0h + (m × 1000h) + (d × 100h)

Function

This register is the bitmap of the individual NSE bits for the region descriptors. Bit n corresponds to region n.

These bits can also be written or "indirectly" through writes to NSE_SET, NSE_CLR, NSE_CLR_ALL, and RGD_W0[NSE] registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 BITn	Bit n NonSecure Enable [n = 0 - 15] 0b - Secure accesses to region r are based on corresponding MRACSEL field in this register (MRCm_DOMd_RGDr_Ww[MRACSEL]), nonsecure accesses to region r are not allowed. 1b - Secure accesses to region r are not allowed, nonsecure accesses to region r are based on corresponding MRACSEL field in register (MRCm_DOMd_RGDr_Ww[MRACSEL]).

43.8.2.50 MRC Global Access Control (MRC3_GLBAC0)

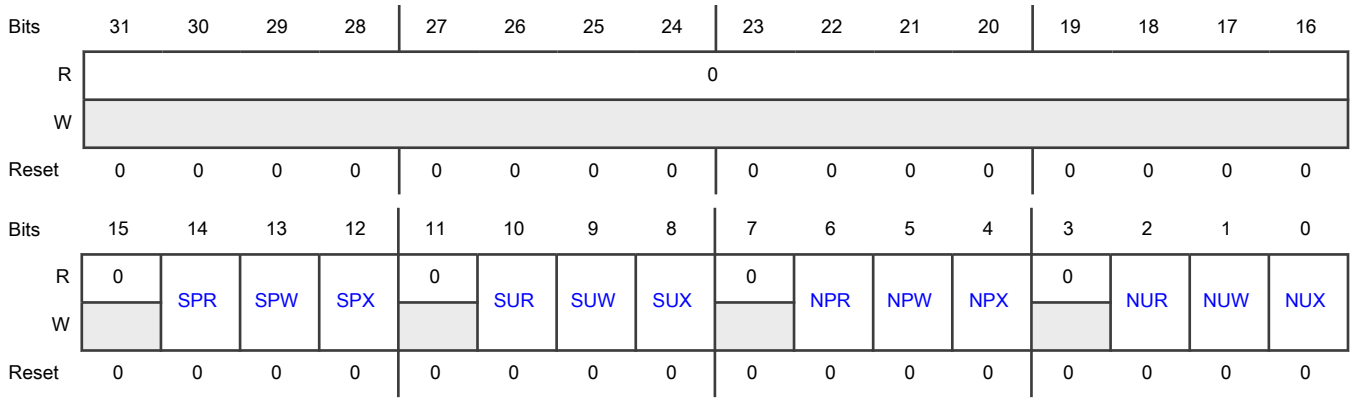
Offset

Register	Offset
MRC3_GLBAC0	1_7020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31-15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.51 MRC Global Access Control (MRC3_GLBAC1 - MRC3_GLBAC7)

Offset

Register	Offset
MRC3_GLBAC1	1_7024h
MRC3_GLBAC2	1_7028h

Table continues on the next page...

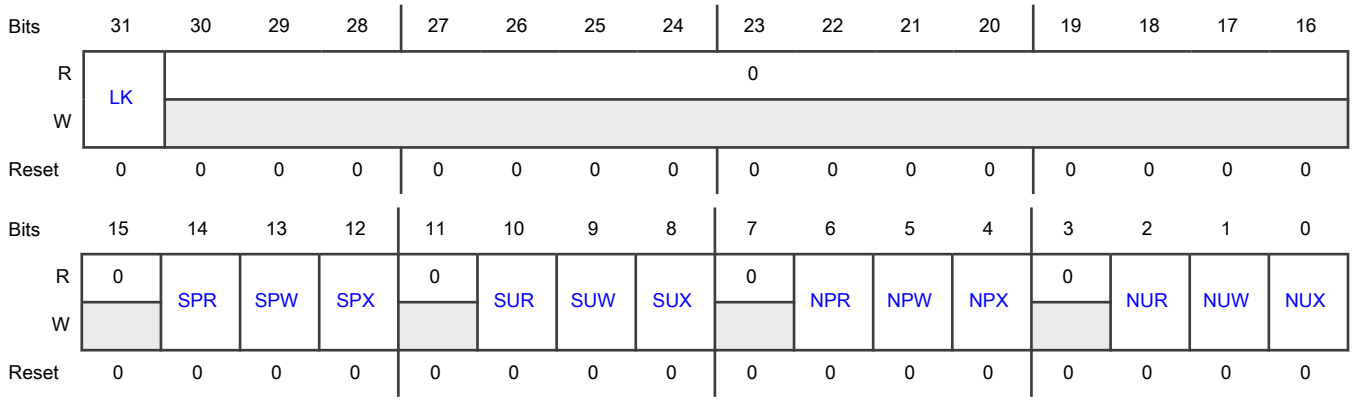
Table continued from the previous page...

Register	Offset
MRC3_GLBAC3	1_702Ch
MRC3_GLBAC4	1_7030h
MRC3_GLBAC5	1_7034h
MRC3_GLBAC6	1_7038h
MRC3_GLBAC7	1_703Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked (read-only) and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5	NonsecurePriv Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
NPW	NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.52 MRC Global Access Control (MRC4_GLBAC0)

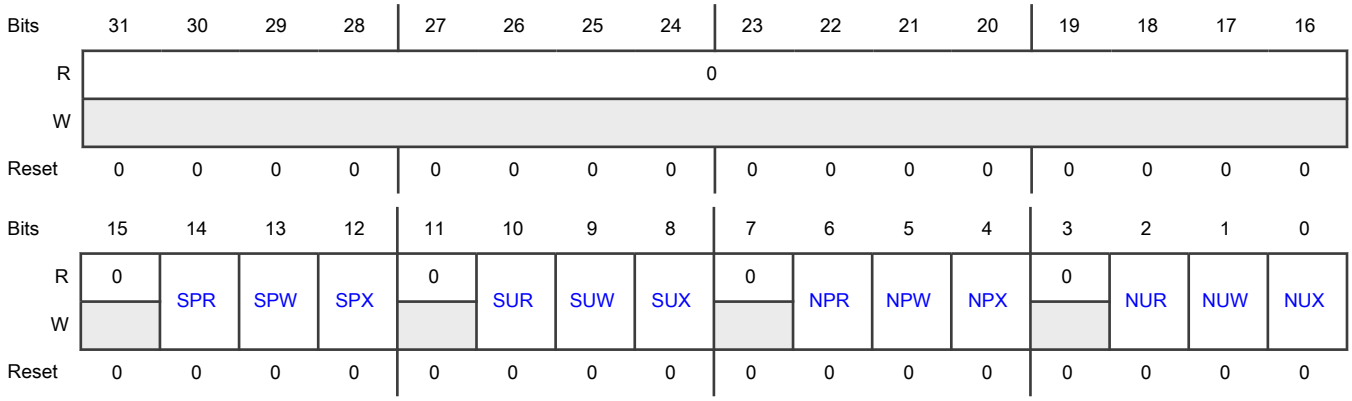
Offset

Register	Offset
MRC4_GLBAC0	1_8020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9	SecureUser Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
SUW	SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.53 MRC Global Access Control (MRC4_GLBAC1 - MRC4_GLBAC7)

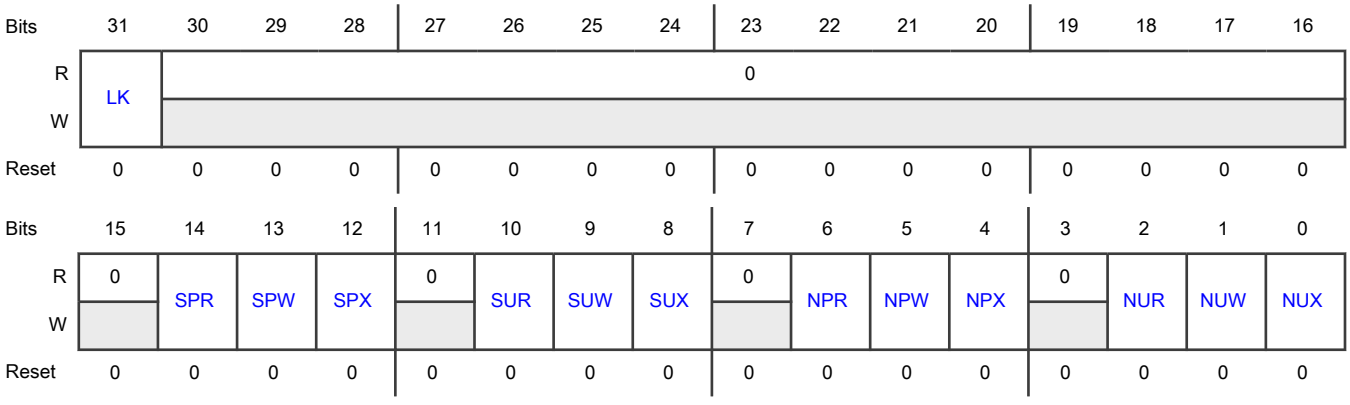
Offset

Register	Offset
MRC4_GLBAC1	1_8024h
MRC4_GLBAC2	1_8028h
MRC4_GLBAC3	1_802Ch
MRC4_GLBAC4	1_8030h
MRC4_GLBAC5	1_8034h
MRC4_GLBAC6	1_8038h
MRC4_GLBAC7	1_803Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked (read-only) and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.54 MRC Global Access Control (MRC5_GLBAC0)

Offset

Register	Offset
MRC5_GLBAC0	1_9020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SPR	SPW	SPX	0	SUR	SUW	SUX	0	NPR	NPW	NPX	0	NUR	NUW	NUX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.55 MRC Global Access Control (MRC5_GLBAC1 - MRC5_GLBAC7)

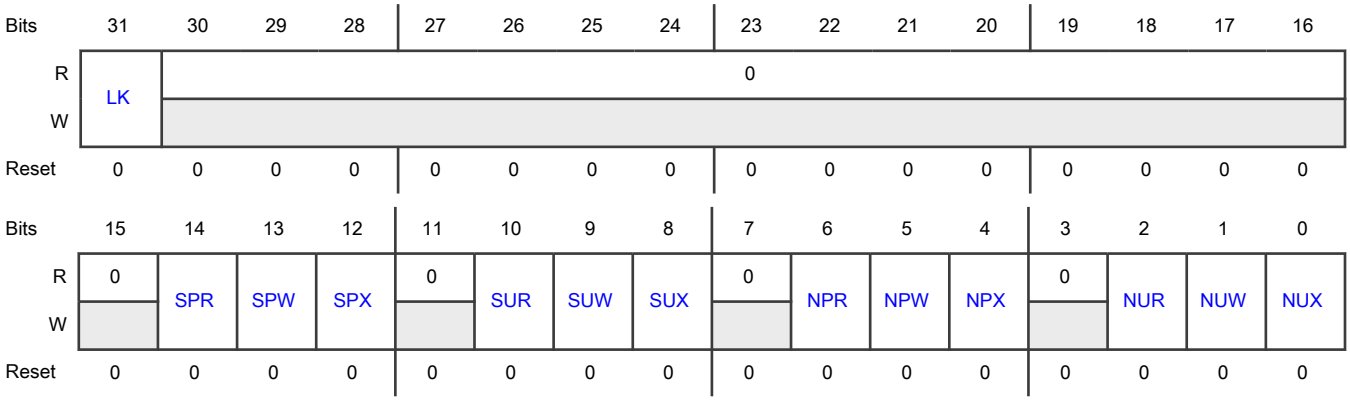
Offset

Register	Offset
MRC5_GLBAC1	1_9024h
MRC5_GLBAC2	1_9028h
MRC5_GLBAC3	1_902Ch
MRC5_GLBAC4	1_9030h
MRC5_GLBAC5	1_9034h
MRC5_GLBAC6	1_9038h
MRC5_GLBAC7	1_903Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked (read-only) and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10	SecureUser Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
SUR	SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.56 MRC Global Access Control (MRC6_GLBAC0)

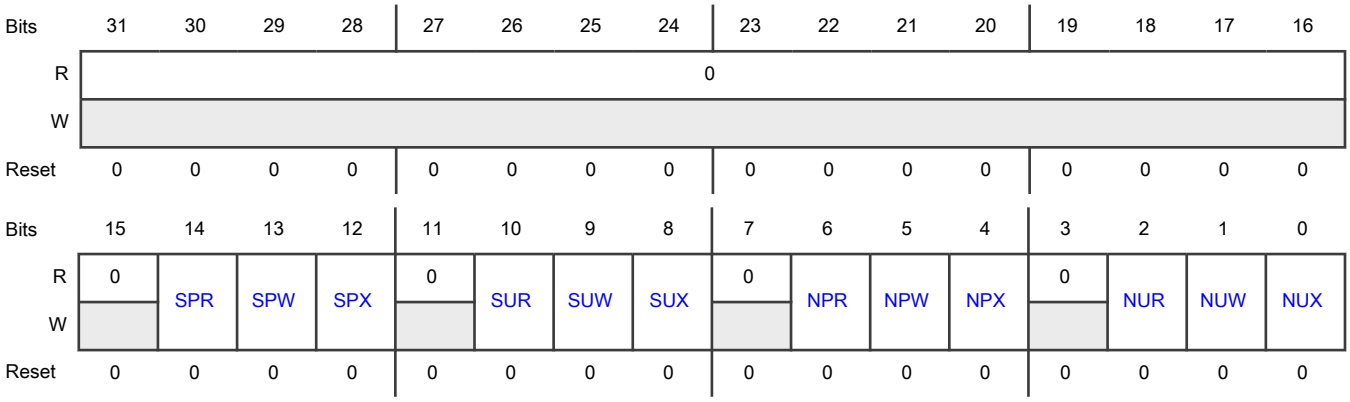
Offset

Register	Offset
MRC6_GLBAC0	1_A020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6	NonsecurePriv Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
NPR	NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.2.57 MRC Global Access Control (MRC6_GLBAC1 - MRC6_GLBAC7)

Offset

Register	Offset
MRC6_GLBAC1	1_A024h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MRC6_GLBAC2	1_A028h
MRC6_GLBAC3	1_A02Ch
MRC6_GLBAC4	1_A030h
MRC6_GLBAC5	1_A034h
MRC6_GLBAC6	1_A038h
MRC6_GLBAC7	1_A03Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these control access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SPR	SPW	SPX	0	SUR	SUW	SUX	0	NPR	NPW	NPX	0	NUR	NUW	NUX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked (read-only) and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5	NonsecurePriv Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
NPW	NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

43.8.3 TRDC register descriptions

43.8.3.1 TRDC memory map

TRDC3 base address: 4281_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRDC Register (TRDC_CR)	32	RW	0000_0010h
F0h	TRDC Hardware Configuration Register 0 (TRDC_HWCFG0)	32	R	2000_0510h
F4h	TRDC Hardware Configuration Register 1 (TRDC_HWCFG1)	32	R	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
F8h	TRDC Hardware Configuration Register 2 (TRDC_HWCFG2)	32	R	0000_0000h
FCh	TRDC Hardware Configuration Register 3 (TRDC_HWCFG3)	32	R	0000_0000h
100h - 104h	Domain Assignment Configuration Register (DACFG0 - DACFG4)	8	R	81h
1C0h	TRDC IDAU Control Register (TRDC_IDAU_CR)	32	RW	0000_0008h
1E0h	TRDC FLW Control (TRDC_FLW_CTL)	32	RW	0000_0000h
1E4h	TRDC FLW Physical Base (TRDC_FLW_PBASE)	32	R	0100_0000h
1E8h	TRDC FLW Array Base (TRDC_FLW_ABASE)	32	RW	0000_0000h
1ECh	TRDC FLW Block Count (TRDC_FLW_BCNT)	32	RW	0000_0000h
1FCh	TRDC Fault Domain ID (TRDC_FDID)	32	RW	0000_0000h
200h - 23Ch	TRDC Domain Error Location Register (TRDC_DERRLOC0 - TRDC_DERRLOC15)	32	R	0000_0000h
800h	DAC Master Domain Assignment Register (MDA_W0_0_DFMT1)	32	RW	2000_0000h
820h	DAC Master Domain Assignment Register (MDA_W0_1_DFMT1)	32	RW	2000_0000h
840h	DAC Master Domain Assignment Register (MDA_W0_2_DFMT1)	32	RW	2000_0000h
860h	DAC Master Domain Assignment Register (MDA_W0_3_DFMT1)	32	RW	2000_0000h
880h	DAC Master Domain Assignment Register (MDA_W0_4_DFMT1)	32	RW	2000_0000h

43.8.3.2 TRDC Register (TRDC_CR)

Offset

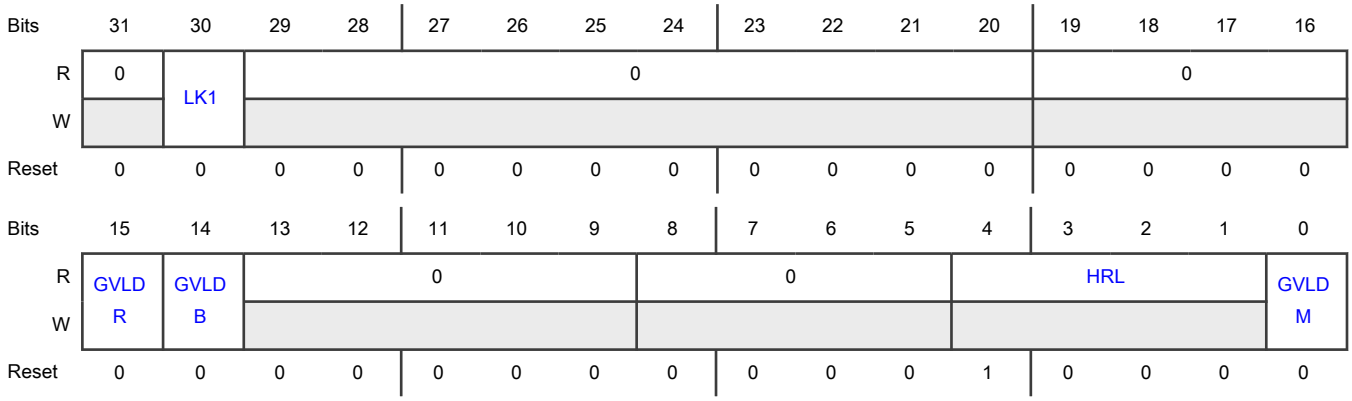
Register	Offset
TRDC_CR	0h

Function

This register provides status about the TRDC and global enable bits for the entire module's operation. A fully operational TRDC requires all global bits GVLD {R,B,M} to be asserted. Undefined behavior results if they are not all asserted. If there are no region checkers (MRCs) present in the configuration, GVLDR need not be asserted for a fully operational TRDC.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31 —	Reserved
30 LK1	Lock Status This field is the lock status of the TRDC_CR. This field is set when all GVLD bits are set; GVLDR=GVLDB=GVLDM=1b1. Once set, this bit remains asserted until the next reset. 0b - The CR can be written by any secure privileged write. 1b - The CR is locked (read-only) until the next reset.
29-20 —	Reserved
19-16 —	Reserved
15 GVLDR	Global Valid for Memory Region Checkers TRDC global MRC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC MRCs are disabled. 1b - TRDC MRCs are enabled.
14 GVLDB	Global Valid for Memory Block Checkers TRDC global MBC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC MBCs are disabled. 1b - TRDC MBCs are enabled.
13-9 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
8-5 —	Reserved
4-1 HRL	Hardware Revision Level This read-only field specifies the TRDC's hardware and definition revision level. It can be read by software to determine the functional definition of the module.
0 GVLDM	Global Valid for Domain Assignment Controllers TRDC global DAC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC DACs are disabled. 1b - TRDC DACs are enabled.

43.8.3.3 TRDC Hardware Configuration Register 0 (TRDC_HWCFG0)

Offset

Register	Offset
TRDC_HWCFG0	F0h

Function

This read-only register contains information on the TRDC's hardware configuration. Specifically, it defines the number of implemented domains and bus masters along with the number of instances of memory block checkers (MBCs) and memory region checkers (MRCs). The register value at reset is device-specific. Attempted writes are error terminated.

Access: f TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MID				NMRC				0				NMBC			
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NMSTR								0				NDID			
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0

Fields

Field	Function
31-29 MID	Module ID This field defines major version ID of this module.
28-24 NMRC	Number of MRCs This field defines the number of MRCs on the device [1-16].
23-20 —	Reserved
19-16 NMBC	Number of MBCs This field defines the number of MBCs on the device [1-8].
15-8 NMSTR	Number of bus masters This read-only field defines the number of bus masters.
7-5 —	Reserved
4-0 NDID	Number of domains This read-only field defines the number of domains on the device [1-16].

43.8.3.4 TRDC Hardware Configuration Register 1 (TRDC_HWCFG1)

Offset

Register	Offset
TRDC_HWCFG1	F4h

Function

This register contains information on the TRDC's hardware configuration. It provides a mechanism for software to determine its domain number by simply reading the register. See Domain error capture management for more details on typical usage. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												DID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	u	u	u	u

Fields

Field	Function
31-4 —	Reserved
3-0 DID	Domain identifier number This field provides the domain number [0-15] of the requesting bus master.

43.8.3.5 TRDC Hardware Configuration Register 2 (TRDC_HWCFG2)

Offset

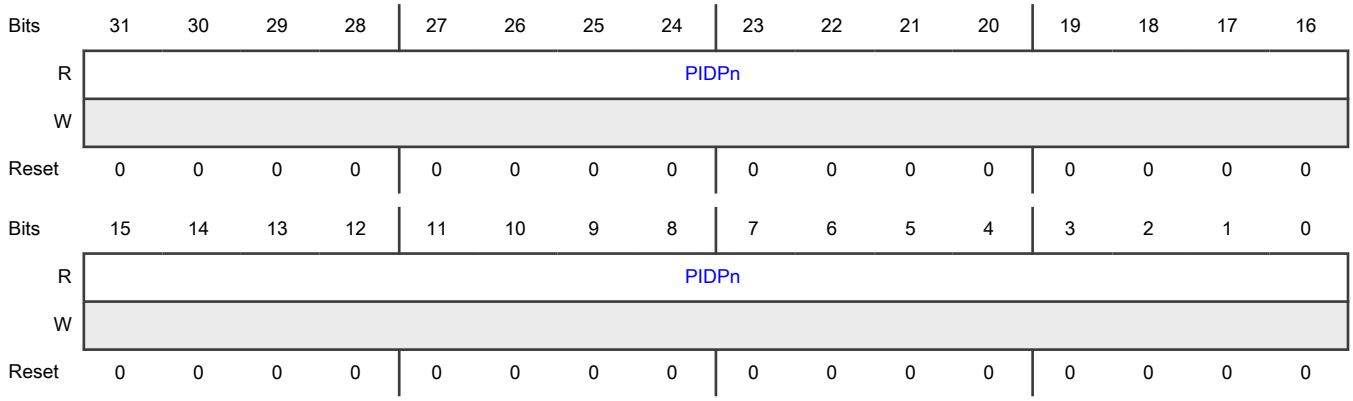
Register	Offset
TRDC_HWCFG2	F8h

Function

This register contains information on the TRDC's hardware configuration. It provides a bitmap signaling the presence of a process identifier (PID) register sourced from the given bus master. The HWCFG2 register is associated with bus masters [31-0]. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 PIDPn	Process identifier present Process identifier present from bus master n, where n = [31-0]. This field provides a bitmap to signal that bus master (where n is defined as bit number) sources a process identifier register to the TRDC_DAC logic. For each bit in this field: • 0 - Bus master n does not source a process identifier register. The TRDC_DAC logic provides the needed PID for processor cores. • 1 - Bus master n sources a process identifier register.

43.8.3.6 TRDC Hardware Configuration Register 3 (TRDC_HWCFG3)

Offset

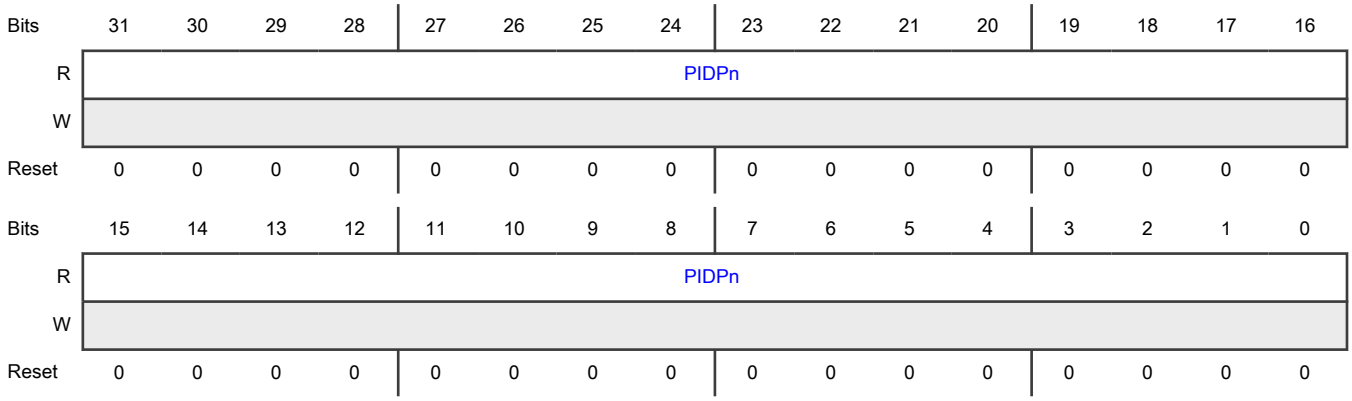
Register	Offset
TRDC_HWCFG3	FCh

Function

This register contains information on the TRDC's hardware configuration. Specifically, it provides a bitmap signaling the presence of a process identifier (PID) register sourced from the given bus master. The HWCFG3 register is associated with bus masters [63-32], while the HWCFG2 register covers bus masters [31-0]. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 PIDPn	Process identifier present Process identifier present from bus master n, where n = [63-32]. This field provides a bitmap to signal that bus master n (where n is defined as 32 + bit number) sources a process identifier register to the TRDC_DAC logic. Process identifier present from bus master n, where n = [31-0]. This field provides a bitmap to signal that bus master (where n is defined as bit number) sources a process identifier register to the TRDC_DAC logic. For each bit in this field: • 0 - Bus master n does not source a process identifier register. The TRDC_DAC logic provides the needed PID for processor cores. • 1 - Bus master n sources a process identifier register.

43.8.3.7 Domain Assignment Configuration Register (DACFG0 - DACFG4)

Offset

Register	Offset
DACFG0	100h
DACFG1	101h
DACFG2	102h
DACFG3	103h
DACFG4	104h

Function

This register defines the number of implemented domain assignment registers for bus master m, where m+1 can specify from 1 to 64 bus masters. These registers are organized as a byte-sized data array and can be read using 8-, 16- or 32-bit accesses. An all-zero value (NCM = 0, NMDAR = 0) indicates a non-existent bus master. Attempted writes are error terminated.

Register read will not return transfer error when DACFG for a master doesn't exist but it is in the same 32-bit group as DACFG for an existing master. For example, when there are only 2 DACFG0-1 registers for 2 master, and masters for DACFG2-3 don't exist, then access to DACFG2-3 won't return transfer error.

Typically, processor bus masters have one or more domain assignment registers, while non-processor masters have a single domain assignment register.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	7	6	5	4	3	2	1	0
R	NCM	0			NMDAR			
W								
Reset	1	0	0	0	0	0	0	1

Fields

Field	Function
7 NCM	Non-CPU Master This read-only field signals that bus master m is a non-CPU master. It specifies that the format of the associated MDA_Wr_m register defines a non-processor domain assignment. This field is zero for a non-existent bus master. 0b - Bus master is a processor. 1b - Bus master is a non-processor.
6-4 —	Reserved
3-0 NMDAR	Number of master domain assignment registers for bus master m This read-only field specifies the number of registers associated with the master domain assignment register for a given bus master. The value is limited to the range [0-8], where zero indicates a non-existent bus master and non-zero values indicate the number of implemented registers associated with this MDAm.

43.8.3.8 TRDC IDAU Control Register (TRDC_IDAU_CR)

Offset

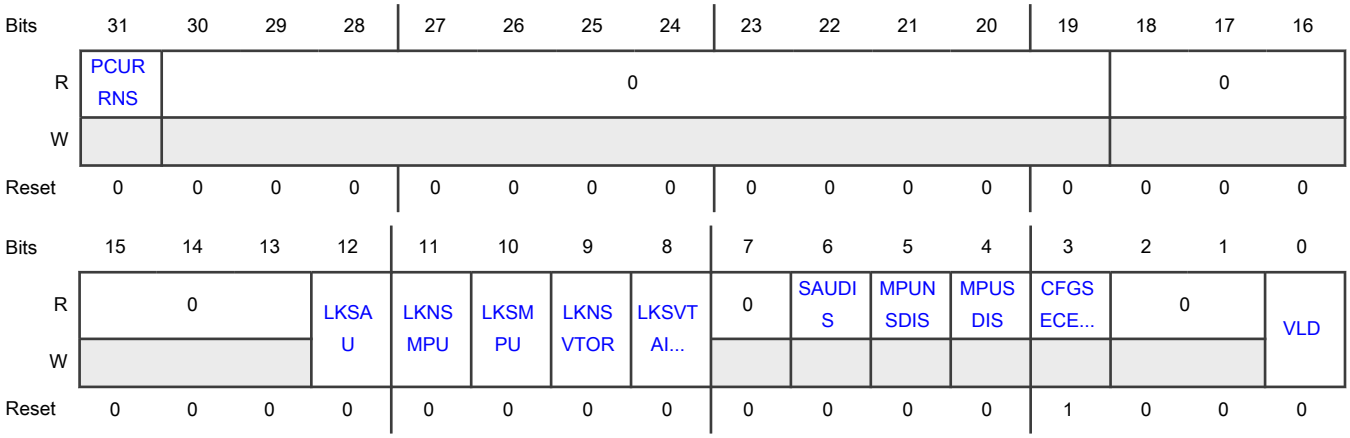
Register	Offset
TRDC_IDAU_CR	1C0h

Function

This register defines the configuration for the Implementation-Defined Attribution Unit which is tightly-coupled to the TZ-M extensions in the processor core. Many of the fields in this register explicitly control processor TZ-M functions.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31 PCURRNS	Processor current security Current security state of the processor. 0b - Processor is in Secure state 1b - Processor is in Nonsecure state
30-19 —	Reserved
18-16 —	Reserved
15-13 —	Reserved
12 LKSAU	Lock SAU Asserting this bit prevents changes to the processor's Secure SAU memory regions already programmed. All processor writes to the registers are ignored. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers 1b - Disable writes to the SAU_CTRL, SAU_RNR, SAU_RBAR and SAU_RLAR registers from software or from a debug agent connected to the processor
11 LKNSMPU	Lock Nonsecure MPU Asserting this bit prevents changes to the processor's Nonsecure MPU memory regions already programmed. All processor writes to the registers are ignored. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Disable writes to the MPU_CTRL_NS, MPU_RNR_NS, MPU_RBAR_NS, MPU_RLAR_NS, MPU_RBAR_A_NSn and MPU_RLAR_A_NSn from software or from a debug agent connected to the processor
10 LKSMPU	Lock Secure MPU Asserting this signal prevents changes to the processor's programmed Secure MPU memory regions and all processor writes to the registers are ignored. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers 1b - Disable writes to the MPU_CTRL, MPU_RNR, MPU_RBAR, MPU_RLAR, MPU_RBAR_An and MPU_RLAR_An from software or from a debug agent connected to the processor in Secure state
9 LKNSVTOR	Lock Nonsecure Vector Table Offset Register Asserting this signal prevents changes to the processor's Nonsecure vector table base address. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock this register 1b - Disable writes to the VTOR_NS register
8 LKSVTAIRCR	Lock Secure VTOR, Application interrupt and Reset Control Registers This bit is sticky, once set, it can only be cleared with a reset. Asserting this signal prevents processor changes to: • The Secure vector table base address. • Handling of Secure interrupt priority. • BusFault, HardFault, and NMI security target settings in the processor. 0b - Unlock these registers 1b - Disable writes to the VTOR_S, AIRCR[PRIS], and AIRCR[BFHFNMINS] registers
7 —	Reserved
6 SAUDIS	Security Attribution Unit Disable 0b - SAU is enabled 1b - SAU is disabled
5 MPUNSDIS	NonSecure Memory Protection Unit Disabled 0b - Nonsecure MPU is enabled 1b - Nonsecure MPU is disabled
4 MPUSDIS	Secure Memory Protection Unit Disabled 0b - Secure MPU is enabled 1b - Secure MPU is disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 CFGSECEXT	Configure Security Extension 0b - Armv8M Security Extension is disabled 1b - Armv8-M Security Extension is enabled
2-1 —	Reserved
0 VLD	Valid When VLD =0, all address attribute from IDAU is nonSecure. When VLD=1, values in other fields of this register are valid.

43.8.3.9 TRDC FLW Control (TRDC_FLW_CTL)

Offset

Register	Offset
TRDC_FLW_CTL	1E0h

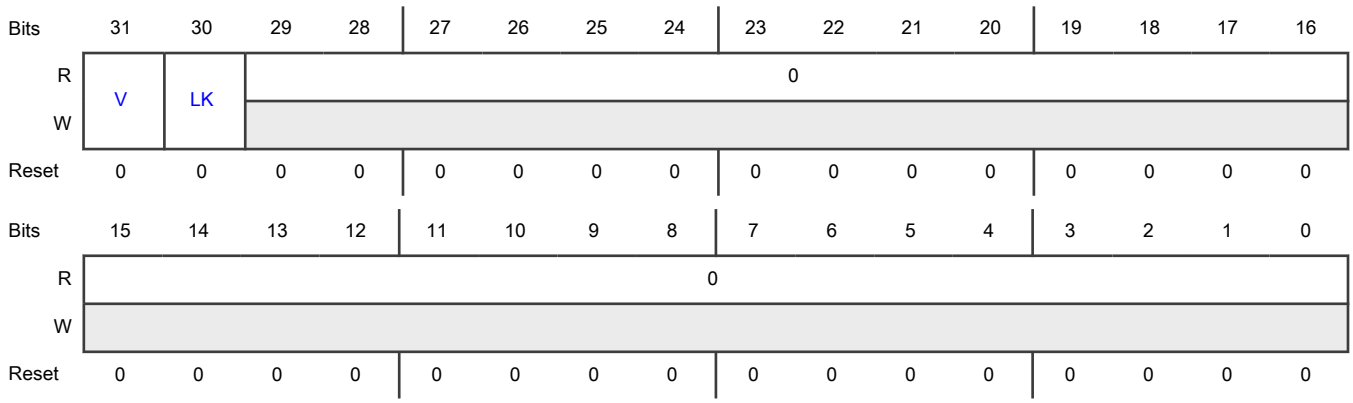
Function

This register provides control of the FLW = Flash Logical Window operation.

This register exists for this device but has no effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31 V	Valid bit 0b - FLW function is disabled. 1b - FLW function is enabled.
30 LK	Lock bit 0b - FLW registers may be modified. 1b - FLW registers are locked until the next reset.
29-0 —	Reserved

43.8.3.10 TRDC FLW Physical Base (TRDC_FLW_PBASE)

Offset

Register	Offset
TRDC_FLW_PBASE	1E4h

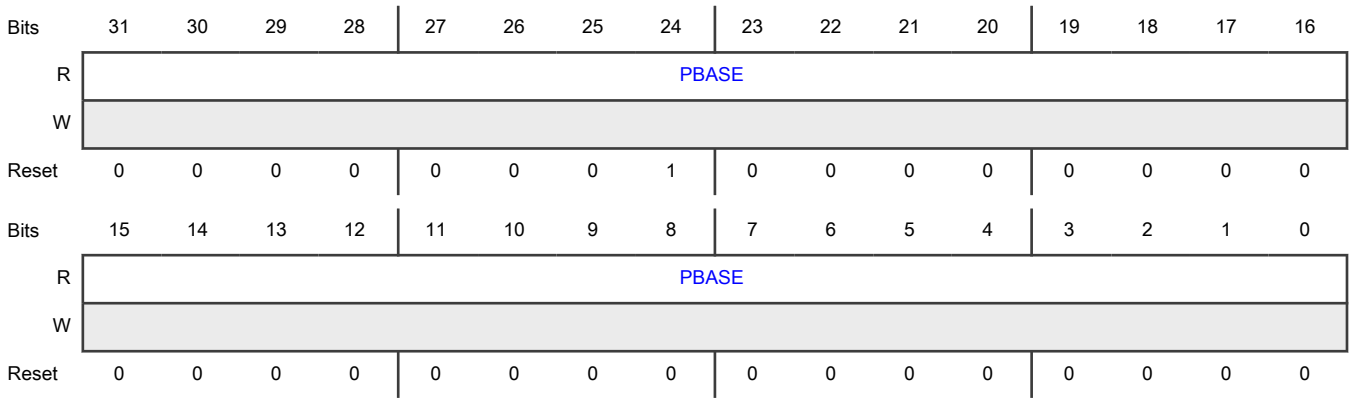
Function

This read-only register gives the physical base address of the Flash Logical Window.

This register exists for this device but has not effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 PBASE	Physical base address Physical address of the base of the FLW (mod 32KBytes).

43.8.3.11 TRDC FLW Array Base (TRDC_FLW_ABASE)

Offset

Register	Offset
TRDC_FLW_ABASE	1E8h

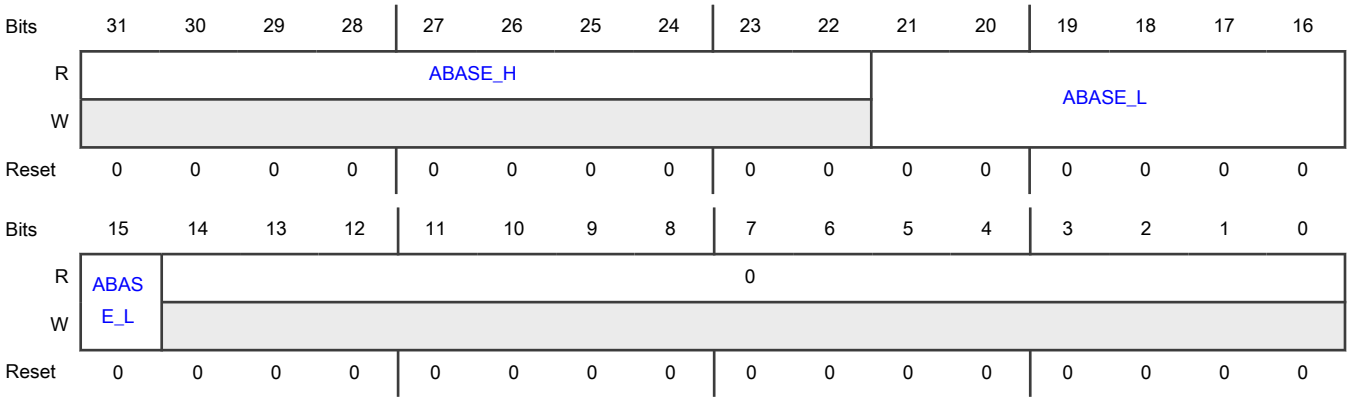
Function

This register gives the flash array base address of the Flash Logical Window.

This register exists for this device but has no effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write. This register is not writable if TRDC_FLW_CTL[LK]=1.

Diagram



Fields

Field	Function
31-22 ABASE_H	Array base address high Flash array address upper bits of the base of the FLW (mod 32KBytes).
21-15 ABASE_L	Array base address low Flash array address lower bits of the base of the FLW (mod 32KBytes).
14-0 —	Reserved

43.8.3.12 TRDC FLW Block Count (TRDC_FLW_BCNT)

Offset

Register	Offset
TRDC_FLW_BCNT	1ECh

Function

This register gives the size of the Flash Logic Window in 32KByte blocks.

This register exists for this device but has no effect on device operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write. This register is not writable if TRDC_FLW_CTL[LK]=1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	BCNT														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14-0 BCNT	Block Count Size of FLW in number of 32KByte blocks.

43.8.3.13 TRDC Fault Domain ID (TRDC_FDID)

Offset

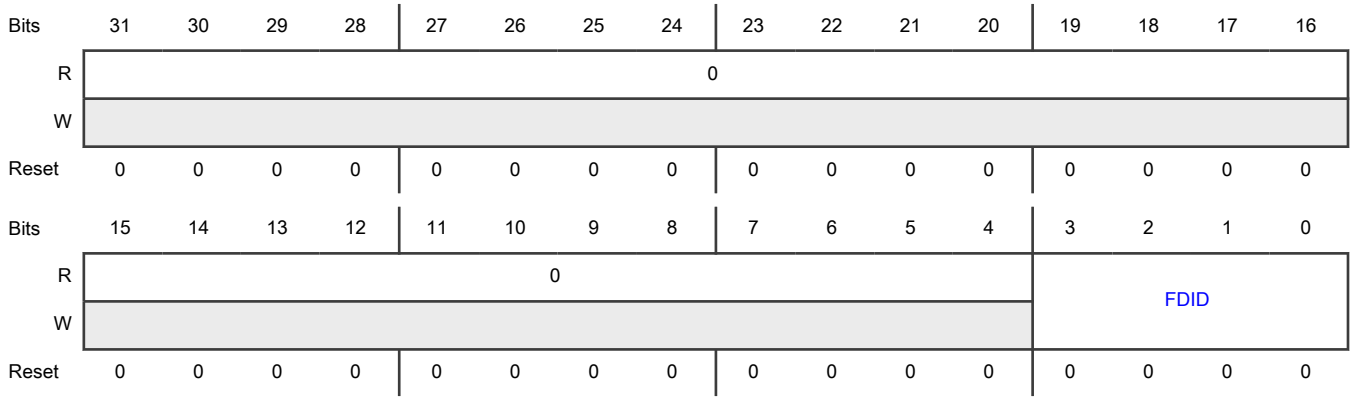
Register	Offset
TRDC_FDID	1FCh

Function

In the event of an access error, this register is used to specify the domainID of the faulting reference before indexing into the Domain Error registers.

If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 FDID	Domain ID of Faulted Access This field indicates the domainID of the fault used to index the array of domain error information. The user queries the DERRLOCx registers to find the DomainID of the faulting access and must write this register with the domain ID before reading the Domain Error registers.

43.8.3.14 TRDC Domain Error Location Register (TRDC_DERRLOC0 - TRDC_DERRLOC15)

Offset

For d = 0 to 15:

Register	Offset
TRDC_DERRLOCd	200h + (d × 4h)

Function

This array of read-only registers provide the instance number of the submodule where (an) access violation(s) occurred. These registers are organized as a word array, indexed by the faulting domain number, d. The two fields of this register provide a bitmap of instances associated with all submodules containing captured error information for that domain. These instance numbers are then used as indices into the DERR_W0_*i*, DERR_W1_*i*, and DERR_W3_*i*register arrays. See Domain error capture management for more details.

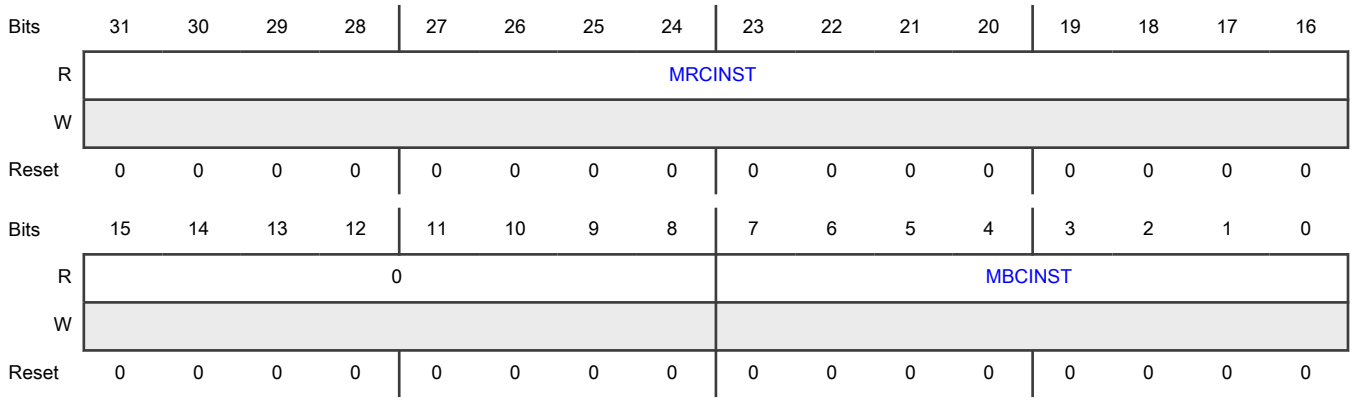
When an access violation is detected by either a Memory Region Checker (MRC) or a Memory Block Checker (MBC), address and attribute information of the offending access is captured. Using the faulting domainID number as the index, d, this array of

read-only registers provide additional information signaling the instance number of the submodule where the access violation(s) occurred. Since the resulting exception handler needs the submodule instance to retrieve the captured address and attribute information from DERR_W0_*i* and DERR_W1_*i*, these registers provide the instance number details.

Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-16 MRCINST	MRC instance This field is a bitmap indicating the presence of a detected access violation for domain d in the implemented instances of the MRC. The least-significant bit of this field (bit 16) corresponds to MRC instance 0. The most-significant bit of this field (bit 31) corresponds to MRC instance 15 (MRC instance = i-16 where i is the register bit number), and so on. Multiple bits can be set at any time indicating access violations for this domain have been detected across multiple instances of the MRCs. For each bit in this field: • 0 - The memory region checker has not detected an access violation or is not physically present. • 1 - The memory region checker has detected one or more access violations for this domain.
15-8 —	Reserved
7-0 MBCINST	MBC instance This field is a bitmap indicating the presence of a detected access violation for domain d in the implemented instances of the MBC. The least-significant bit of this field (bit 0) corresponds to MBC instance 0. The most-significant bit of this field (bit 7) corresponds to MBC instance 7 Multiple bits can be set at any time indicating access violations for this domain have been detected across multiple instances of the MBCs. For each bit in this field: • 0 - The memory block checker has not detected an access violation or is not physically present. • 1 - The memory block checker has detected one or more access violations for this domain.

43.8.3.15 DAC Master Domain Assignment Register (MDA_W0_0_DFMT1 - MDA_W0_4_DFMT1)

Offset

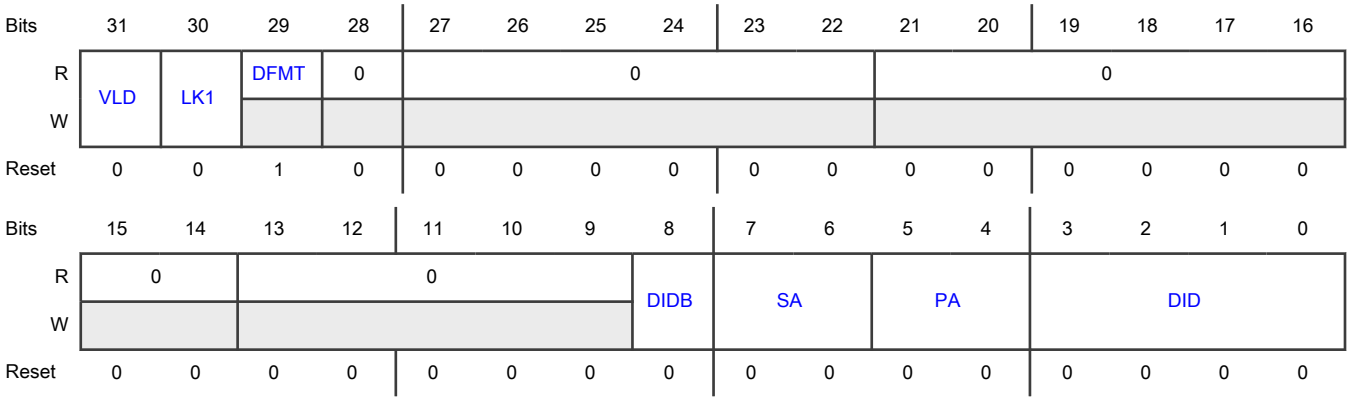
Register	Offset
MDA_W0_0_DFMT1	800h
MDA_W0_1_DFMT1	820h
MDA_W0_2_DFMT1	840h
MDA_W0_3_DFMT1	860h
MDA_W0_4_DFMT1	880h

Function

This register is identical to Master Domain Assignment (MDA_Wr_m_DFMT0) register except that the domain format field, DFMT, is 1 and the PID field is not used. This format supports two different specifications of the DID for non-core bus masters.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/write.

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates the domain assignment is valid. It is further qualified by CR[GVLDM] = 1. If CR[GVLDM] is cleared, the DID output is defined by the SoC-specific *default DID* value. See the TRDC chip configuration section for more details. If both MDA_Wr_m[VLD] and CR[GVLDM] are asserted, the DID output is defined by the remaining contents of this register. 0b - The Wr domain assignment is invalid. 1b - The Wr domain assignment is valid.
30 LK1	1-bit Lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field provides a locking mechanism that can be used to limit the ability to write the register. Once set, this bit remains asserted until the next reset. 0b - Register can be written by any secure privileged write. 1b - Register is locked (read-only) until the next reset.
29 DFMT	Domain format Identifies this register's domain assignment format. **NOTE** This bitfield access is ROO 0b - Processor-core domain assignment 1b - Non-processor domain assignment
28 —	Reserved
27-22 —	Reserved
21-16 —	Reserved
15-14 —	Reserved
13-9 —	Reserved
8 DIDB	DID Bypass If asserted, this bit enables the bypassing of an input DID value as the domain identifier for this non-processor bus master. This capability allows non-processor bus masters, for example, a DMA to masquerade as a processor. Once set, this field is “sticky” and remains set until the next reset. **DAC** / **DID Input** 0 / 0 1 / DMA Domain ID 2 / USB Domain ID 0b - Use MDAn[3:0] as the domain identifier.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Use the DID input as the domain identifier.
7-6 SA	Secure attribute This field defines the secure/nonsecure attribute for non-processor cores. **NOTE** The bus master's input secure/nonsecure attribute is used if SA = 1X, or this VLD = 0. If TZM_ENB = 0, the master attribute is forced to nonsecure. A nonsecure write cannot program this field to 2'b00, which is a security level higher than the mode of the process that is writing it. Reset value of SA: • if TZM_ENB=0, SA reset value = 2'b01 • if TZM_ENB=1, SA reset value = 2'b00 00b - Force the bus attribute for this master to secure. 01b - Force the bus attribute for this master to nonsecure. 10b - Use the bus master's secure/nonsecure attribute directly. 11b - Use the bus master's secure/nonsecure attribute directly.
5-4 PA	Privileged attribute This field defines the privileged/user attribute for non-processor cores. **NOTE** The bus master's input privileged/user attribute is used if PA = 1X, or this VLD = 0. 00b - Force the bus attribute for this master to user. 01b - Force the bus attribute for this master to privileged. 10b - Use the bus master's privileged/user attribute directly. 11b - Use the bus master's privileged/user attribute directly.
3-0 DID	Domain identifier This 4-bit field is the domain ID attribute that is sent on the accesses from the bus master connected to the DAC when DIDB=0.

Chapter 44
Error Injection Module (EIM)

44.1 Chip-specific EIM information

Table 310. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

This device has one instance of the EIM module.

The following table shows the EIM channel assignments:

Table 311. EIM Channel Assignments

Channel	Target for Error Injection
0	CM7 Instruction Cache Tag Array
1	CM7 Instruction Cache Data Array
2	CM7 Data Cache Tag Array
3	CM7 Data Cache Data Array0
4	CM7 Data Cache Data Array1

44.2 Overview

The Error Injection Module (EIM) is mainly used for diagnostic purposes. It provides a method for diagnostic coverage of internal memories (for example, system RAM, cache RAMs, and peripheral memories). See the chip-specific EIM information to determine which functional safety features are supported by this method.

EIM enables you to induce artificial errors on error-checking mechanisms of a system, such as ECC for RAM read data and parity bits. For each such mechanism that EIM supports on the chip, EIM can inject single-bit and multi-bit inversions on data in the applicable target bus. Injecting faults on memory accesses can be used to exercise the SEC-DED ECC function of the related system.

44.2.1 Features

The EIM includes these features:

- Supports 5 error injection channels. See the chip-specific EIM information for channel assignment details.
- Protection against accidental enable and reconfiguration error injection function via two-stage enable mechanism

44.2.2 Block diagram

The following diagram shows an example of EIM implementation with a 64-bit read data bus and an 8-bit checkbit bus.

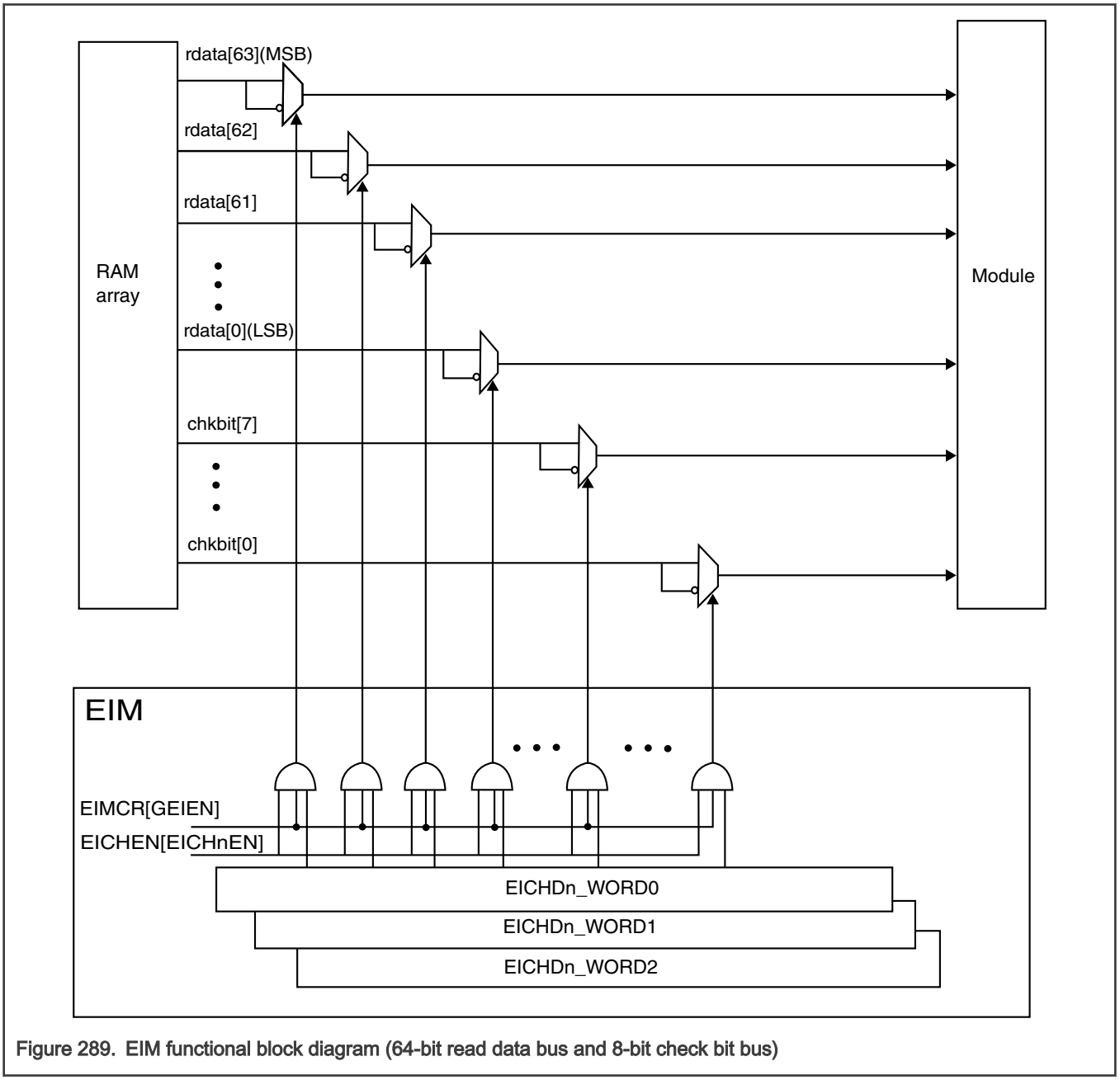


Figure 289. EIM functional block diagram (64-bit read data bus and 8-bit check bit bus)

Several memory elements are implemented within a device, which may not only be the large memory blocks (Flash and SRAM) but also smaller memories like caches, the TCD blocks, and the embedded peripheral memories. Some larger memories may actually be built from multiple memory elements, dependent on their size or function. Each of these memory elements implements its own control logic, the memory controller, that performs the accesses to the actual memory, the memory array. An EIM channel

is associated with a memory controller and provides the capability to alter one or multiple signals in the read access path from the corresponding memory array(s). Only memory controllers controlling a safety related memory may be associated with an EIM channel.

44.3 Functional description

The EIM provides protection against accidental enabling and reconfiguration of the error injection function by enforcing a two-stage enablement mechanism. To properly enable the error injection mechanism for a channel:

- Write 1 to the EICHEN[EICH*n*EN] field, where *n* denotes the channel number.
- Write 1 to EIMCR[GEIEN].

NOTE

When the use case for a channel requires writing any EICHD*n*_WORD register, write the EICHD*n*_WORD register before executing the two-stage enablement mechanism. A successful write to any EICHD*n*_WORD register clears the corresponding EICHEN[EICH*n*EN] field.

The EIM supports 5 error injection channels. See the chip-specific EIM information for channel assignment details. Each channel:

- Can be assigned to a single memory array interface by intercepting the assigned memory read data bus and checkbit bus, and injects errors by inverting the value transmitted for selected bits on each bus line.
- Can be assigned to a redundant comparison unit by intercepting the signals being compared, and injecting errors by inverting the value transmitted for selected bits on each bus line.

On a memory read access, the applicable EICHD*n*_WORD registers define which bits of the read data and/or checkbit bus to invert.

Figure 289 depicts the interception and override of a 64-bit read data bus and an 8-bit checkbit data bus for an example memory array.

Error injection scenarios

The EIM supports these cases of error injection:

- To generate a single-bit error, invert only 1 bit of the CHKBIT_MASK or DATA_MASK in the EICHD*n*_WORD registers.
- To generate a multi-bit error, invert only 2 bits of the CHKBIT_MASK or DATA_MASK in the EICHD*n*_WORD registers.

NOTE

An attempt to invert more than 2 bits in one operation might result in undefined behavior.

To enable error injection:

1. Set the EICHD*n*_WORD*m*[CHKBIT_MASK] and EICHD*n*_WORD*m*[B*a*_*b*DATA_MASK] fields for each channel that will be driving an injection.
2. Program the EICHEN register to enable the channels that will be injecting errors.
3. Set the EIMCR[GEIEN] field to globally allow all enabled channels to actively inject errors.

To disable error injection, either disable the EIMCR[GEIEN] field or disable the individual channel enable fields of the EICHEN register.

44.4 Initialization

This module does not require initialization.

44.6 EIM register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 312. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	44	43-32	31-0	—
1	128	127-96	95-64	63-32
2	104	103-96	95-64	63-32
3	128	127-96	95-64	63-32
4	128	127-96	95-64	63-32

Table 313. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
1	31-0	—	—	—	—
2	31-0	—	—	—	—
3	31-0	—	—	—	—
4	31-0	—	—	—	—

44.6.1 EIM memory map

EIM base address: 4B86_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
14Ch	Error Injection Channel Descriptor 1, Word3 (EICHD1_WORD3)	32	RW	0000_0000h
150h	Error Injection Channel Descriptor 1, Word4 (EICHD1_WORD4)	32	RW	0000_0000h
180h	Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
18Ch	Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)	32	RW	0000_0000h
190h	Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
1CCh	Error Injection Channel Descriptor 3, Word3 (EICHD3_WORD3)	32	RW	0000_0000h
1D0h	Error Injection Channel Descriptor 3, Word4 (EICHD3_WORD4)	32	RW	0000_0000h
200h	Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
20Ch	Error Injection Channel Descriptor 4, Word3 (EICHD4_WORD3)	32	RW	0000_0000h
210h	Error Injection Channel Descriptor 4, Word4 (EICHD4_WORD4)	32	RW	0000_0000h

44.6.2 Error Injection Module Configuration Register (EIMCR)

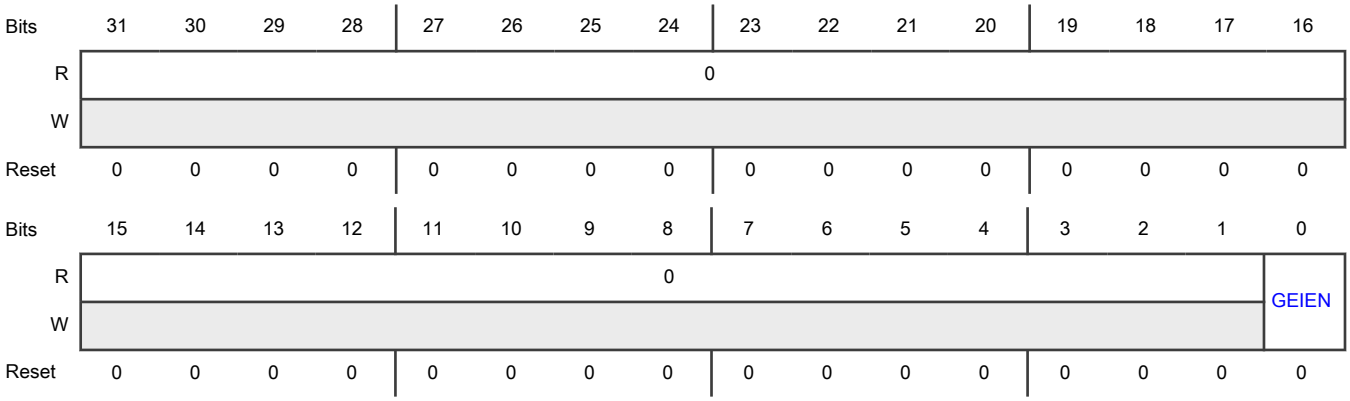
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled

44.6.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 —	Reserved

44.6.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)

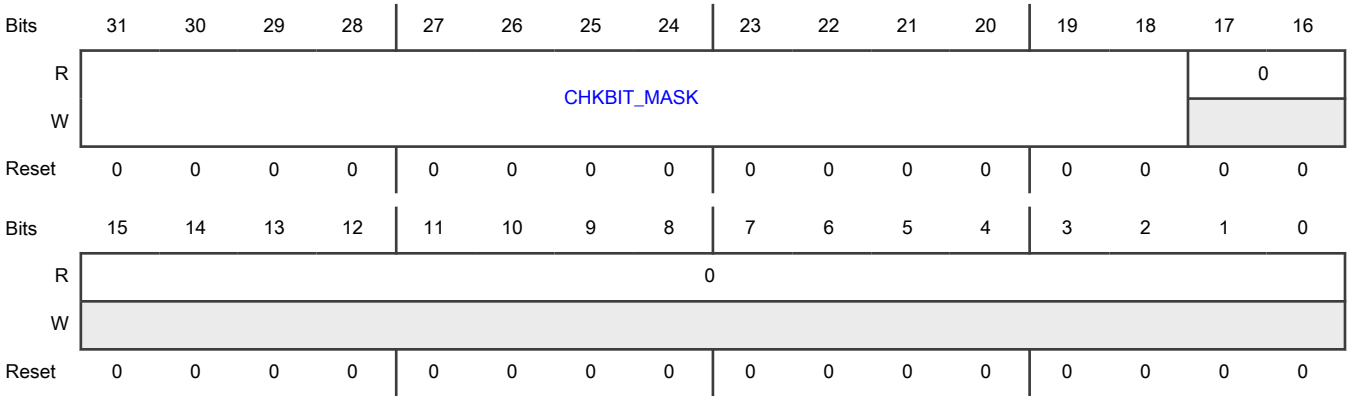
Offset

Register	Offset
EICHD0_WORD0	100h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-18 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[13:0] (14 bits wide), CHKBIT_MASK[13] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
17-0 —	Reserved

44.6.5 Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)

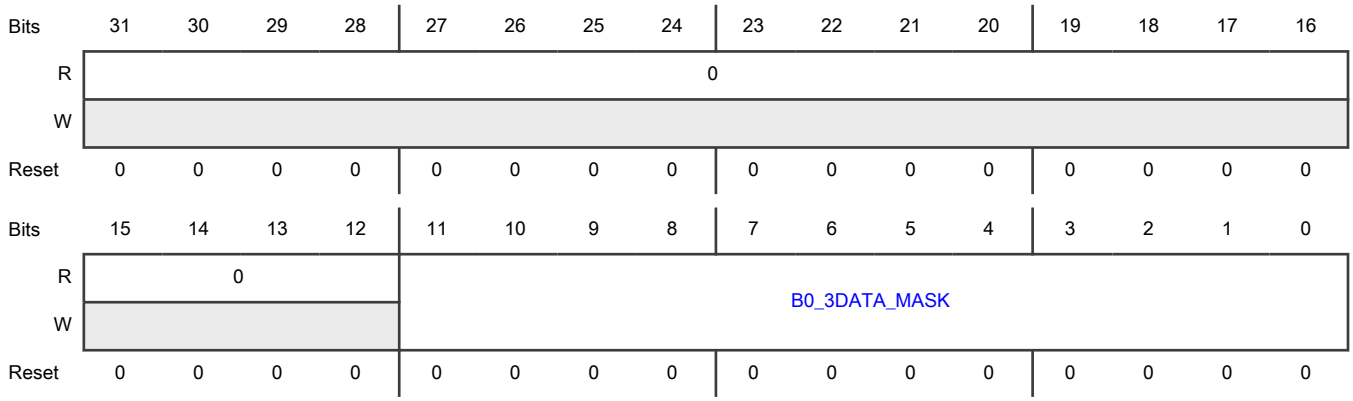
Offset

Register	Offset
EICHD0_WORD1	104h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

44.6.6 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD4_WORD2)

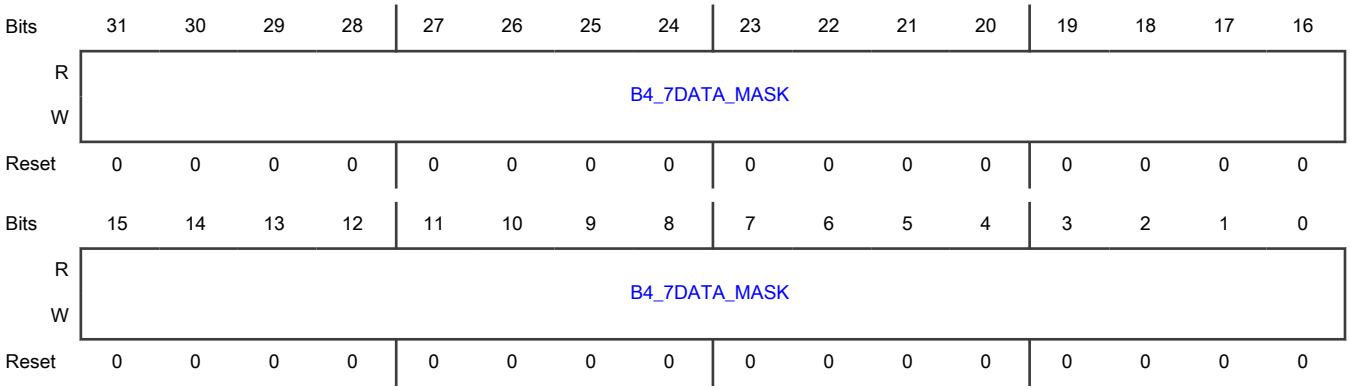
Offset

Register	Offset
EICHD0_WORD2	108h
EICHD1_WORD2	148h
EICHD2_WORD2	188h
EICHD3_WORD2	1C8h
EICHD4_WORD2	208h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

44.6.7 Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)

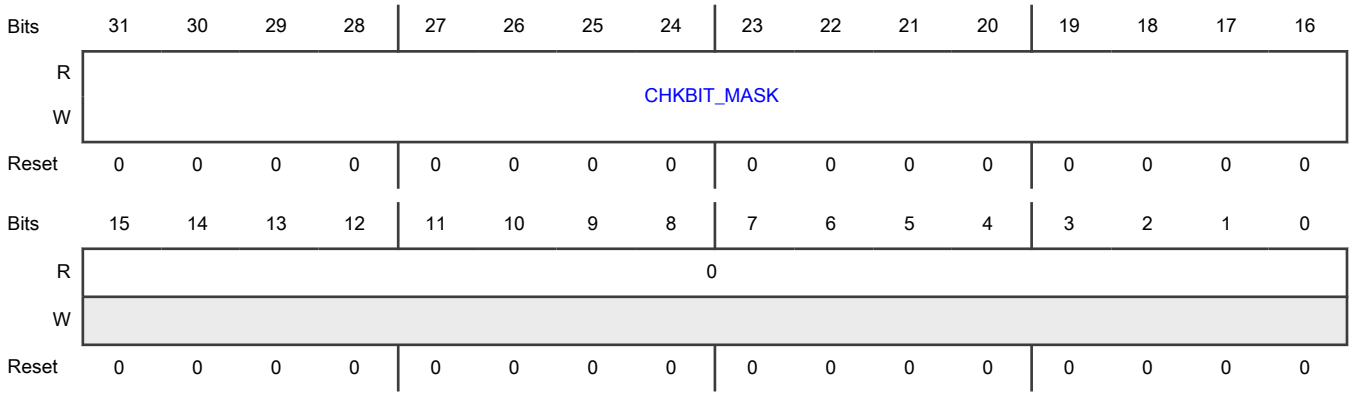
Offset

Register	Offset
EICHD1_WORD0	140h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16	Checkbit Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
CHKBIT_MASK	This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

44.6.8 Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)

Offset

Register	Offset
EICHD1_WORD1	144h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

44.6.9 Error Injection Channel Descriptor n, Word3 (EICHD1_WORD3 - EICHD4_WORD3)

Offset

Register	Offset
EICHD1_WORD3	14Ch
EICHD2_WORD3	18Ch
EICHD3_WORD3	1CCh
EICHD4_WORD3	20Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B8_11DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B8_11DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

44.6.10 Error Injection Channel Descriptor n, Word4 (EICHD1_WORD4 - EICHD4_WORD4)

Offset

Register	Offset
EICHD1_WORD4	150h
EICHD2_WORD4	190h
EICHD3_WORD4	1D0h
EICHD4_WORD4	210h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B12_15DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B12_15DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

44.6.11 Error Injection Channel Descriptor n, Word0 (EICHD2_WORD0 - EICHD4_WORD0)

Offset

Register	Offset
EICHD2_WORD0	180h
EICHD3_WORD0	1C0h
EICHD4_WORD0	200h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CHKBIT_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHKBIT_MASK												0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[27:0] (28 bits wide), CHKBIT_MASK[27] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
3-0 —	Reserved

44.6.12 Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)

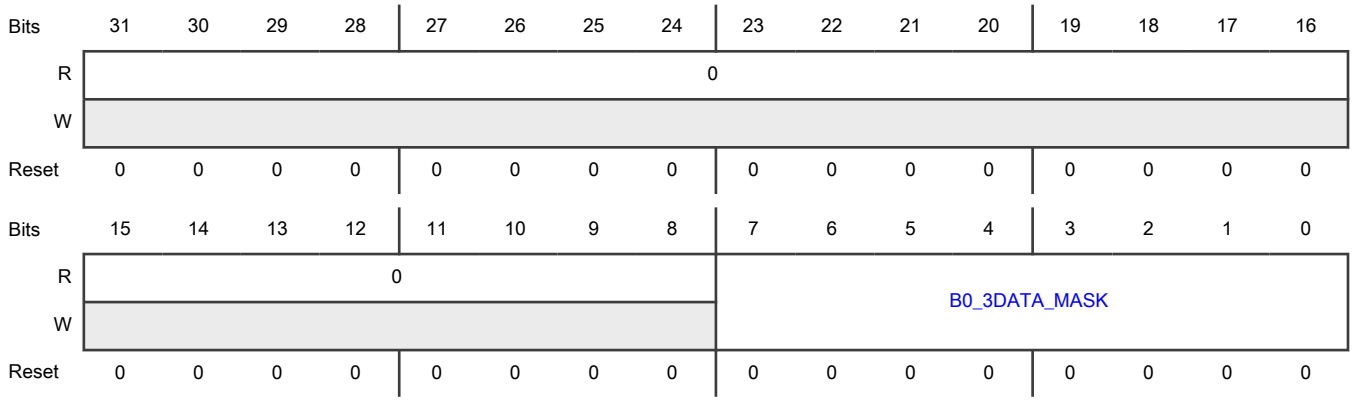
Offset

Register	Offset
EICHD2_WORD1	184h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

44.6.13 Error Injection Channel Descriptor n, Word1 (EICHD3_WORD1 - EICHD4_WORD1)

Offset

Register	Offset
EICHD3_WORD1	1C4h
EICHD4_WORD1	204h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Chapter 45
Error Reporting Module (ERM)

45.1 Chip-specific ERM information

Table 314. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

This device has one instance of the ERM module.

The following table shows the ERM channel assignments and which channel is defined in ERM Configuration Register 0. Unlisted channels are not used.

Table 315. ERM Channel Assignments

Channel	CR0 Bits	Target for Error Report Configuration
0	CR0[31:30]	CM7 Instruction Cache Tag
1	CR0[27:26]	CM7 Instruction Cache Data
2	CR0[23:22]	CM7 Data Cache Tag
3	CR0[19:18]	CM7 Data Cache Data

45.2 Overview

The Error Reporting Module (ERM) provides information and optional interrupt notification on memory error events associated with ECC and parity. The ERM collects error events on memory accesses for memory arrays, such as flash memory, system RAM, or peripheral RAMs. ERM supports various channels for memory sources where each ERM channel is associated with a different memory module. See the chip-specific ERM information for details about supported memory sources and specific memory channel assignments. If the memory supports ECC, then ERM syndrome and error address information is captured along with the error event. ERM does not capture syndrome or error address for cache memories or memory with parity instead of ECC.

45.2.1 Features

The ERM includes these features:

- Optional interrupt notification on captured error events
- Support for error event capturing for memory sources, with individual reporting fields and interrupt configuration per memory channel
- Recording the count value of the number of corrected error events

45.3 Functional description

45.3.1 Single-bit correction events

When a single-bit correction event on Memory *n* is detected, the ERM:

- Records the event by changing the value of the applicable Status Register bit SR*x*[SBC*n*] to 1.
- Increments the correctable error count value (until the counter reaches its maximum value): CORR_ERR_CNT*n*[COUNT].

The ERM holds event information only for the last reported event.

To clear the record of an event, write 1 to SR*x*[SBC*n*] to change its value to 0.

To reset the correctable error count value, write all zeros to CORR_ERR_CNT*n*[COUNT].

Optional interrupt notification for single-bit correction events

The ERM provides an option to generate an interrupt notification upon the report of a single-bit correction event. To enable single-bit correction interrupts for a channel:

1. To enable interrupt notification for single-bit correction events on Memory *n*, set CR*x*[ESCIE*n*] to 1.
2. Subsequently, when a single-bit correction event on Memory *n* is detected, the ERM:
   - Records the event as usual.
   - Additionally sends an interrupt notification corresponding to the event.
3. To clear both the record of an event and the corresponding interrupt notification, write 1 to SR*x*[SBC*n*] to change its value to 0.

45.3.2 Non-correctable error events

When a non-correctable ECC error event on Memory *n* is detected, the ERM:

- Records the event by changing the value of the applicable Status Register bit: SR*x*[NCE*n*] to 1.

The ERM holds event information only for the last reported event.

To clear the record of an event, write 1 to SR*x*[NCE*n*] to change its value to 0.

Optional interrupt notification for non-correctable error events

The ERM provides an option to generate an interrupt notification upon the report of a non-correctable ECC event. To enable non-correctable error interrupts for a channel:

1. To enable interrupt notifications for non-correctable error events on Memory *n*, set CR*x*[ENCIE*n*] to 1.
2. Subsequently, when a non-correctable error event on Memory *n* is detected, the ERM:
   - Records the event as usual.
   - Additionally sends an interrupt notification corresponding to the event.
3. To clear both the record of an event and the corresponding interrupt notification, write 1 to SR*x*[NCE*n*] to change its value to 0.

NOTE

Parity errors can be mapped to non-correctable errors where error attributes like SYNDROME, ADDRESS are not provided.

45.4 Initialization

For each ERM channel supporting memory with ECC, prepare the corresponding memory array before enabling ERM interrupts about errors for that memory.

1. Initialize the memory to a known value so that the correct corresponding ECC codeword is stored.
2. During the memory's initialization, if the ERM captures information about any ECC error event, clear the corresponding SR*x*[SBC*n*] or SR*x*[NCE*n*] field that stores the record of the event.
3. Program the applicable CR*x*[ESCIE*n*] and CR*x*[ENCIE*n*] fields to enable ERM interrupts as desired.

45.5 ERM register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

NOTE

- See the chip-specific ERM information for details on Memory channel mapping.
- To access the channel registers, corresponding memory channel clock must be enabled.

45.5.1 ERM memory map

ERM base address: 4B86_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	RW	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)	32	RW	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT1)	32	RW	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT2)	32	RW	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT3)	32	RW	0000_0000h

45.5.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 1 single-bit correction events is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15-14 —	Reserved
13-12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

45.5.3 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register reports error events for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

45.5.4 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT3)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h

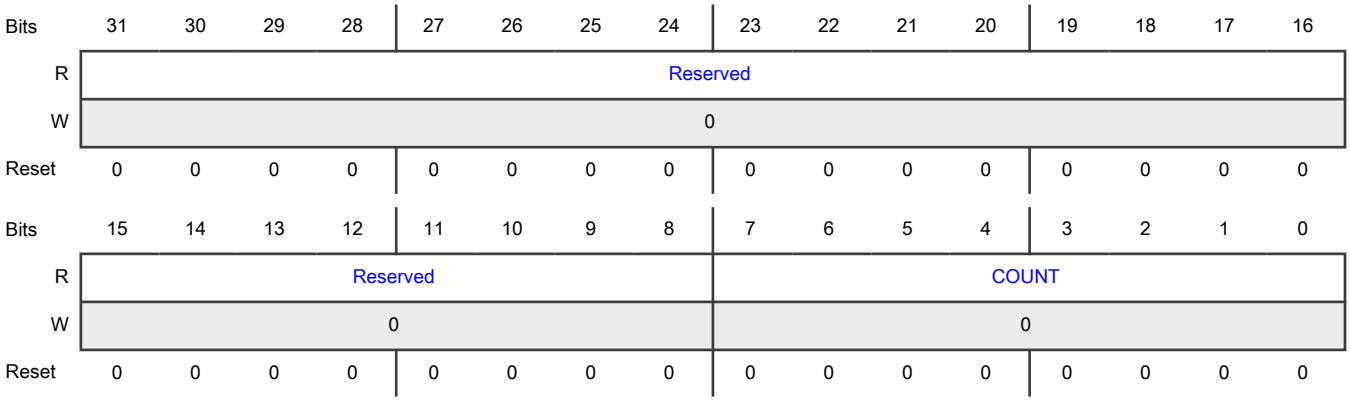
Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory*n*, where*n*denotes the memory channel.

NOTE

Non-correctable errors are considered a serious fault, so the ERM does not provide any mechanism to count non-correctable errors. Only correctable errors are counted.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh.COUNT value will stop when it reaches maximum value FFh and will not wrap even if additional errors occur. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing a non-zero value has no effect.

Chapter 46
Crossbar Switch (AXBS)

46.1 Overview

This section provides information on the layout, configuration, and programming of the crossbar switch.

The crossbar switch connects bus masters and bus slaves using a crossbar switch structure. This structure allows all bus masters to access different bus slaves simultaneously, while providing arbitration among the bus masters when they access the same slave. A variety of bus arbitration methods and attributes may be programmed on a slave-by-slave basis.

46.1.1 Features

- Symmetric crossbar bus switch implementation
  - Allows concurrent access from different masters to different slaves
  - Slave arbitration attributes configured on a slave-by-slave basis
- Single-clock 32-bit transfer
- Support for burst transfers of 64 bits of data
- Support for low-power park mode
- Master high-priority elevation
- 32-bit AHB crossbar bus switch compatible with ARM's Advanced Microcontroller Bus Architecture (AMBA) Specification v2.0

46.2 Functional description

Information about general operation and arbitration are provided in this section.

46.2.1 General operation

When a master accesses the crossbar switch, the access is immediately taken. If the targeted slave port of the access is available, then the access is immediately presented on the slave port. Single-clock or zero-wait-state accesses are possible through the crossbar. If the targeted slave port of the access is busy or parked on a different master port, the requesting master sees wait states inserted until the targeted slave port can service the master's request. The latency in servicing the request depends on each master's priority level and the responding slave's access time.

Because the crossbar switch appears to be just another slave to the master device, the master device does not know whether it owns the slave port it is targeting. The master waits while it does not have control of the slave port it is targeting.

After the master acquires control of the slave port, it controls the port until it relinquishes the port by running an IDLE cycle or by targeting a different slave port for its next access.

The master can also lose control of the slave port if another higher-priority master makes a request to the slave port. However, if the master is running a fixed-length burst transfer, it retains control of the slave port until that transfer completes.

The crossbar terminates all master IDLE transfers, as opposed to allowing the termination to come from one of the slave buses. Additionally, when no master is requesting access to a slave port, the crossbar drives IDLE transfers onto the slave bus, even though a default master may be granted access to the slave port.

When a slave bus is being idled by the crossbar, it can park the slave port on the master port indicated by CRS*n*[PARK]. This is done to save the initial clock of arbitration delay that otherwise would be seen if the same master had to arbitrate to gain control of the slave port. The slave port can also be put into low-power park mode to save power, by using CRS*n*[PCTL].

46.2.2 Register coherency

The operation of the crossbar is affected as soon as a register is written. The values of the registers do not track with slave-port-related master accesses, but instead track only with slave accesses.

46.2.3 Arbitration

The crossbar switch supports the following arbitration algorithms:

- Round-robin

The arbitration scheme is independently programmable for each slave port.

46.2.3.1 Fixed-priority operation

When operating in fixed-priority mode, each master is assigned a unique priority level in the priority registers (PRS*n*). If two masters request access to the same slave port, the master with the highest priority in the selected priority register gains control over the slave port.

NOTE

In this arbitration mode, a higher-priority master can monopolize a slave port, preventing access from any lower-priority master to the port.

When a master makes a request to a slave port, the slave port checks whether the new requesting master's priority level is higher than that of the master that currently has control over the slave port, unless the slave port is in a parked state. The slave port performs an arbitration check at every clock edge to ensure that the master, if any, has control of the slave port.

The following table describes possible scenarios based on the requesting master port.

Table 316. Methods of how the crossbar switch grants control of a slave port to a master

When	Then the crossbar switch grants control to the requesting master
Both of the following are true: • The current master is not running a transfer. • The new requesting master's priority level is higher than that of the current master.	At the next clock edge
Both of the following are true: • The current master is running a fixed-length burst transfer or a locked transfer. • The requesting master's priority level is higher than that of the current master.	At the end of the burst transfer or locked transfer
The requesting master's priority level is lower than the current master.	At the conclusion of one of the following cycles: • An IDLE cycle • A non-IDLE cycle to a location other than the current slave port

46.2.3.2 Round-robin priority operation

When operating in round-robin mode, each master is assigned a relative priority based on the master port number. This relative priority is compared to the master port number (ID) of the last master to perform a transfer on the slave bus. The highest priority requests the master owns the slave bus at the next transfer boundary, accounting for locked and fixed-length burst transfers. Priority is based on how far ahead the ID of the requesting master is of the ID of the last master.

After a master is granted access to a slave port, a master may perform as many transfers as desired to that port until another master requests the same slave port. The next master in line is granted access to the slave port at the next transfer boundary, or possibly on the next clock cycle, if the current master has no pending access request.

As an example of arbitration in round-robin mode, assume that the crossbar is implemented with master ports 0, 1, 4, and 5. If the last master of the slave port was master 1, and masters 0, 4, and 5 make simultaneous requests, they are serviced in this order: 4,5, and then 0.

Parking may continue to be used in a round-robin mode, but does not affect the round-robin pointer unless the parked master performs a transfer. Handoff occurs to the next master in line after one cycle of arbitration. If the slave port is put into low-power park mode, the round-robin pointer is reset to point at master port 0, giving it the highest priority.

46.2.3.3 Clocking

This module has no clocking considerations.

46.2.3.4 Interrupts

This module has no interrupts.

46.2.3.5 Priority assignment

Each master port must be assigned a unique 3-bit priority level. If an attempt is made to program multiple master ports with the same priority level within the priority registers (PRS*n*), the crossbar switch responds with a bus error and the registers are not updated.

46.3 External signals

This module has no external signals.

46.4 Initialization/application information

No initialization is required for the crossbar switch.

Hardware reset ensures that all the register bits used by the crossbar switch are properly initialized to a valid state. However, the following settings and priorities may be programmed to achieve the maximum system performance:

- During the configuration of the crossbar switch, all other masters must be IDLE.
- To prevent reconfiguration of the crossbar switch, write 1 to CRS*n*[RO].

46.5 Memory map and register definition

Each slave port of the crossbar switch contains configuration registers. Read- and write transfers require two bus clock cycles. The registers can be read from and written to only in supervisor mode. Additionally, these registers can be read from or written to only by 32-bit accesses.

A bus error response is returned if an unimplemented location is accessed within the crossbar switch.

The CRS*n* and PRS*n* registers can be programmed as read-only to prevent changes to their configuration. After being read-only protected, future writes to them terminate with a data storage error.

NOTE

This section shows the registers for all eight master and slave ports. If a master or slave is not used on this particular chip, then unexpected results occur when writing to its registers. See the chip configuration details for the exact master and slave assignments for your chip.

All references to the crossbar switch registers are based on the physical port connections.

46.5.1 AXBS register descriptions

46.5.1.1 AXBS memory map

AXBS base address: 4451_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Priority Slave Registers (PRS0)	32	RW	0013_4605h
10h	Control Register (CRS0)	32	RW	0002_0000h
100h	Priority Slave Registers (PRS1)	32	RW	0013_4605h
110h	Control Register (CRS1)	32	RW	0002_0000h
200h	Priority Slave Registers (PRS2)	32	RW	0013_4605h
210h	Control Register (CRS2)	32	RW	0002_0000h
300h	Priority Slave Registers (PRS3)	32	RW	0013_4605h
310h	Control Register (CRS3)	32	RW	0002_0000h
400h	Priority Slave Registers (PRS4)	32	RW	0013_4605h
410h	Control Register (CRS4)	32	RW	0002_0000h
500h	Priority Slave Registers (PRS5)	32	RW	0013_4605h
510h	Control Register (CRS5)	32	RW	0002_0000h
600h	Priority Slave Registers (PRS6)	32	RW	0013_4605h
610h	Control Register (CRS6)	32	RW	0002_0000h
700h	Priority Slave Registers (PRS7)	32	RW	0013_4605h
710h	Control Register (CRS7)	32	RW	0002_0000h

46.5.1.2 Priority Slave Registers (PRS0 - PRS7)

Offset

Register	Offset
PRS0	0h
PRS1	100h
PRS2	200h
PRS3	300h
PRS4	400h
PRS5	500h
PRS6	600h
PRS7	700h

Function

The priority slave registers(PRS*n*) set the priority of each master port on a per slave port basis and reside in each slave port. The priority register can be accessed only with 32-bit access. After the CRS*n*[RO] bit is set, the PRS*n* register can only be read; attempts to write to it have no effect on PRS*n* and result in a bus-error response to the master initiating the write.

Two available masters must not be programmed with the same priority level. Attempts to program two or more masters with the same priority level result in a bus-error response and the PRS*n* is not updated.

NOTE

Valid values for the M*n* priority fields depend on which masters are available on the chip. This information can be found in the chip-specific information for the crossbar.

- If the chip contains fewer than three masters, only one bit is valid.
- If the chip contains fewer than five masters, only two bits are valid.
- If five or more masters are present, all three bits of the priority field are used.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	Reserved			0	Reserved			0	M5			0	M4		
W		-				-										
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	M3			0	M2			0	M1			0	M0		
W																
Reset	0	1	0	0	0	1	1	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31 —	Reserved
30-28 —	Reserved
27 —	Reserved
26-24 —	Reserved
23 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
22-20 M5	Master 5 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or lowest priority when accessing the slave port.
19 —	Reserved
18-16 M4	Master 4 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or lowest priority when accessing the slave port.
15 —	Reserved
14-12 M3	Master 3 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8the or lowest priority when accessing the slave port.
11 —	Reserved
10-8 M2	Master 2 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8the or lowest priority when accessing the slave port.
7 —	Reserved
6-4 M1	Master 1 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or lowest priority when accessing the slave port.
3 —	Reserved
2-0 M0	Master 0 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or the lowest priority when accessing the slave port.

46.5.1.3 Control Register (CRS0 - CRS7)

Offset

Register	Offset
CRS0	10h
CRS1	110h
CRS2	210h
CRS3	310h
CRS4	410h
CRS5	510h
CRS6	610h
CRS7	710h

Function

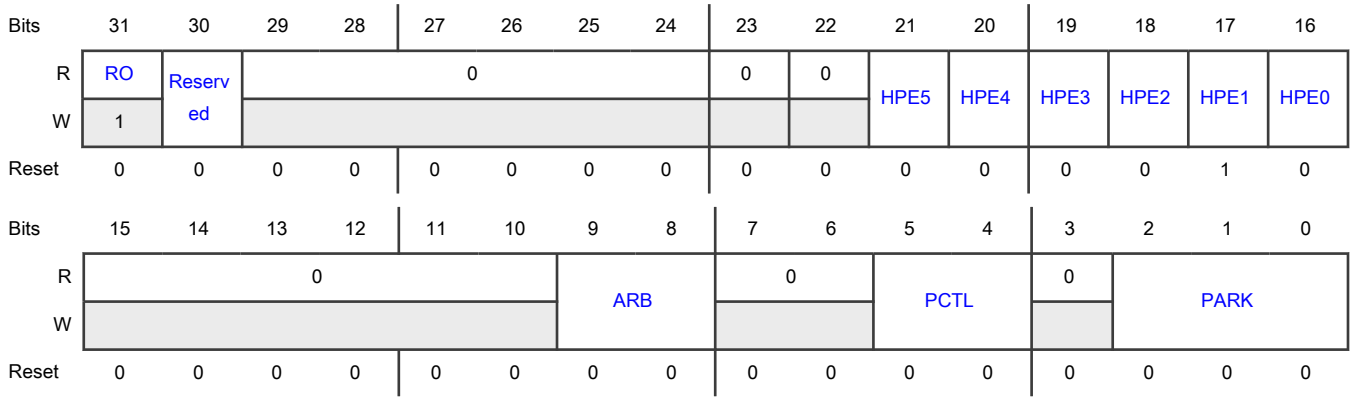
These registers control several features of each slave port and must be accessed using 32-bit accesses. After CRS*n*[RO] is set, the PRS*n* can only be read; attempts to write to it have no effect and results in an error response.

NOTE

See the chip-specific crossbar information for the reset value of this register.

Not all HPE *n* fields may be active. See the chip-specific crossbar information for which masters support master high-priority elevation. Setting a field corresponding to a master that does not support master, high-priority elevation has no effect.

Diagram



Fields

Field	Function
31 RO	Read Only Forces the PRS*n* and CRS*n* registers to be read-only. After being set, only a hardware reset clears this field. 0b - The CRS*n* and PRS*n* registers are writeable 1b - The CRS*n* and PRS*n* registers are read-only and cannot be written (attempted writes have no effect on the registers and result in a bus error response).
30 —	Reserved
29-24 —	Reserved
23 —	Reserved
22 —	Reserved
21 HPE5	High Priority Elevation 5 This field enables master high-priority elevation for master 5,on this slave port. If enabled, the master is able to elevate its priority to the highest. 0b - Master high-priority elevation for master 5. is disabled on this slave port. 1b - Master high-priority elevation for master 5. is enabled on this slave port.
20 HPE4	High Priority Elevation 4 This field enables master high-priority elevation for master 4,on this slave port. If enabled, the master can elevate its priority to the highest. 0b - Master high-priority elevation for master 4. is disabled on this slave port.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Master high-priority elevation for master 4. is enabled on this slave port.
19 HPE3	High Priority Elevation 3 This field enables master high-priority elevation for master 3,on this slave port. If enabled, the master can elevate its priority to the highest. 0b - Master high-priority elevation for master 3. is disabled on this slave port. 1b - Master high-priority elevation for master 3. is enabled on this slave port.
18 HPE2	High Priority Elevation 2 This field enables master high-priority elevation for master 2, on this slave port. If enabled, the master can elevate its priority to the highest. 0b - Master high-priority elevation for master 2. is disabled on this slave port. 1b - Master high-priority elevation for master 2. is enabled on this slave port.
17 HPE1	High Priority Elevation 1 This field enables master high-priority elevation for master 1 on this slave port. If enabled, the master can elevate its priority to the highest. 0b - Master high-priority elevation for master 1. is disabled on this slave port. 1b - Master high-priority elevation for master 1. is enabled on this slave port.
16 HPE0	High Priority Elevation 0 This field enables master high-priority elevation for master 0,on this slave port. If enabled, the master can elevate its priority to the highest. 0b - Master high-priority elevation for master 0. is disabled on this slave port. 1b - Master high-priority elevation for master 0. is enabled on this slave port.
15-10 —	Reserved
9-8 ARB	Arbitration Mode This field selects the arbitration policy for the slave port. 00b - Fixed priority 01b - Round-robin (rotating) priority 10b - Reserved 11b - Reserved
7-6 —	Reserved
5-4	Parking Control

Table continues on the next page...

Table continued from the previous page...

Field	Function
PCTL	This field determines the slave port's parking control. The low-power park feature results in an overall power savings if the slave port is not saturated; however, this forces an extra latency clock when any master tries to access the slave port when not in use because it is not parked on any master. 00b - When no master makes a request, the arbiter parks the slave port on the master port defined by the PARK bit field. 01b - When no master makes a request, the arbiter parks the slave port on the last master to be in control of the slave port. 10b - Low-power park. When no master makes a request, the slave port is not parked on a master and the arbiter drives all outputs to a constant safe state. 11b - Reserved
3 —	Reserved
2-0 PARK	Park This field determines which master port the current slave port parks on when no masters are actively making requests and the PCTL bits are cleared. **NOTE** Select only master ports that are present on the chip. Otherwise, undefined behavior might occur. 000b - Park on master port M0 001b - Park on master port M1 010b - Park on master port M2 011b - Park on master port M3 100b - Park on master port M4 101b - Park on master port M5 110b - Park on master port M6 111b - Park on master port M7

Chapter 47
Network Interconnect Bus System (NIC-400)

47.1 Overview

This section provides an overview of the NIC-400 (Network Inter-Connect) AXI arbiter IP. The NIC-400 (by Arm Ltd.) is a configurable AXI arbiter between several masters and slaves. The NIC-400 IP is designed so that many configuration options are selected at the hardware design stage, determined by SoC characteristics and needs, while several other configuration options are software-controlled.

This chapter covers in brief the NIC-400 while providing configuration details on the NIC-400 instances used in the chip. For complete details on the NIC-400 design, see the Arm specification, AMBA® Network Interconnect (NIC-400) Technical Reference Manual, version r2p3.

47.1.1 NIC-400 main features

Key features of the NIC-400 module include the following:

- Address space memory mapping.
- Programmer's view, for software-configured parameters, via internal "GPV" ports.
- Support for cross-clock domain synchronization.

It supports Arm CoreLink QoS-400 Network Interconnect Advanced Quality of Service (QoS) to arbitrate and regulate the transactions from various SoC masters.

See the following documentation on Arm website for more details:

- ARM CoreLink QoS-400 Network Interconnect Advanced Quality of Service Supplement to ARM CoreLink NIC-400 Network Interconnect Technical Reference Manual

47.1.2 Modes and operations

The NIC-400 supports a normal functional mode only.

47.2 External signals

There are no external I/O interfaces for NIC-400.

47.3 Module instances

This chip contains two instances of the NIC-400 module: NIC_MAIN and NIC_MEGA.

47.4 Memory map and register definition

For detailed descriptions of these registers, see the Arm document AMBA® Network Interconnect (NIC-400) Technical Reference Manual, version r2p3, and ARM CoreLink QoS-400 Network Interconnect Advanced Quality of Service Supplement to ARM CoreLink NIC-400 Network Interconnect Technical Reference Manual.

47.4.1 NIC_MAIN registers

The NIC_MAIN GPV base address is GPV1_BASE = 0x43900000 (Non-Seure space) or 0x53900000 (Secure space). The following registers are implemented in this NIC:

Table 317. NIC_MAIN registers

Register name	Port name	IP name	Absolute address	Reset value	Notes
fn_mod_ahb	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x028	0	
wr_tidemark	m_a_2	ETR	GPV2_BASE + 0x44000 + 0x040	4	The WFIFO depth is 8.
wr_tidemark	m_a_5	DAP	GPV2_BASE + 0x47000 + 0x040	4	The WFIFO depth is 8.
read_qos	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x100	0	
read_qos	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x100	0	
read_qos	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x100	0	
read_qos	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x100	0	
read_qos	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x100	0	
read_qos	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x100	0	
write_qos	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x104	0	
write_qos	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x104	0	
write_qos	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x104	0	
write_qos	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x104	0	
write_qos	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x104	0	
write_qos	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x104	0	
fn_mod	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x108	0	

Table continues on the next page...

Table 317. NIC_MAIN registers (continued)

Register name	Port name	IP name	Absolute address	Reset value	Notes
fn_mod	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x108	0	
fn_mod	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x108	0	
fn_mod	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x108	0	
fn_mod	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x108	0	
fn_mod	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x108	0	
fn_mod	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x108	0	
fn_mod	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x108	0	
qos_cntl	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x10C	0	
qos_cntl	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x10C	0	
qos_cntl	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x10C	0	
qos_cntl	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x10C	0	
qos_cntl	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x10C	0	
qos_cntl	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x10C	0	
qos_cntl	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x10C	0	
qos_cntl	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x10C	0	
max_ot	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x110	0	

Table continues on the next page...

Table 317. NIC_MAIN registers (continued)

Register name	Port name	IP name	Absolute address	Reset value	Notes
max_ot	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x110	0	
max_ot	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x110	0	
max_ot	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x110	0	
max_ot	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x110	0	
max_ot	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x110	0	
max_ot	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x110	0	
max_ot	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x110	0	
max_comb_ot	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x114	0	
max_comb_ot	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x114	0	
max_comb_ot	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x114	0	
max_comb_ot	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x114	0	
max_comb_ot	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x114	0	
max_comb_ot	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x114	0	
max_comb_ot	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x114	0	
max_comb_ot	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x114	0	
aw_p	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x118	0	

Table continues on the next page...

Table 317. NIC_MAIN registers (continued)

Register name	Port name	IP name	Absolute address	Reset value	Notes
aw_p	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x118	0	
aw_p	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x118	0	
aw_p	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x118	0	
aw_p	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x118	0	
aw_p	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x118	0	
aw_p	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x118	0	
aw_p	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x118	0	
aw_b	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x11C	0	
aw_b	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x11C	0	
aw_b	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x11C	0	
aw_b	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x11C	0	
aw_b	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x11C	0	
aw_b	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x11C	0	
aw_b	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x11C	0	
aw_b	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x11C	0	
aw_r	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x120	0	

Table continues on the next page...

Table 317. NIC_MAIN registers (continued)

Register name	Port name	IP name	Absolute address	Reset value	Notes
aw_r	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x120	0	
aw_r	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x120	0	
aw_r	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x120	0	
aw_r	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x120	0	
aw_r	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x120	0	
aw_r	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x120	0	
aw_r	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x120	0	
ar_p	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x124	0	
ar_p	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x124	0	
ar_p	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x124	0	
ar_p	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x124	0	
ar_p	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x124	0	
ar_p	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x124	0	
ar_p	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x124	0	
ar_p	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x124	0	
ar_b	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x128	0	

Table continues on the next page...

Table 317. NIC_MAIN registers (continued)

Register name	Port name	IP name	Absolute address	Reset value	Notes
ar_b	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x128	0	
ar_b	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x128	0	
ar_b	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x128	0	
ar_b	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x128	0	
ar_b	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x128	0	
ar_b	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x128	0	
ar_b	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x128	0	
ar_r	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x12C	0	
ar_r	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x12C	0	
ar_r	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x12C	0	
ar_r	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x12C	0	
ar_r	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x12C	0	
ar_r	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x12C	0	
ar_r	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x12C	0	
ar_r	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x12C	0	
target_fc	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x130	0	

Table continues on the next page...

Table 317. NIC_MAIN registers (continued)

Register name	Port name	IP name	Absolute address	Reset value	Notes
target_fc	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x130	0	
target_fc	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x130	0	
target_fc	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x130	0	
target_fc	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x130	0	
target_fc	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x130	0	
target_fc	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x130	0	
target_fc	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x130	0	
ki_fc	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x134	0	
ki_fc	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x134	0	
ki_fc	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x134	0	
ki_fc	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x134	0	
ki_fc	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x134	0	
ki_fc	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x134	0	
ki_fc	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x134	0	
ki_fc	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x134	0	
qos_range	m_a_0	NETC	GPV1_BASE + 0x42000 + 0x138	0	

Table continues on the next page...

Table 317. NIC_MAIN registers (continued)

Register name	Port name	IP name	Absolute address	Reset value	Notes
qos_range	m_a_1	CM7	GPV1_BASE + 0x43000 + 0x138	0	
qos_range	m_a_2	ETR	GPV1_BASE + 0x44000 + 0x138	0	
qos_range	m_a_3	MEGAMIX	GPV1_BASE + 0x45000 + 0x138	0	
qos_range	m_a_4	AONMIX	GPV1_BASE + 0x46000 + 0x138	0	
qos_range	m_a_5	DAP	GPV1_BASE + 0x47000 + 0x138	0	
qos_range	m_a_6	eDMA4	GPV1_BASE + 0x48000 + 0x138	0	
qos_range	m_a_7	IEE	GPV1_BASE + 0x49000 + 0x138	0	

47.4.2 NIC_MEGA registers

The NIC_MEGA GPV base address is GPV2_BASE = 0x42830000 (Non-Seure space) or 0x52830000 (Secure space). The following registers are implemented in this NIC:

Table 318. NIC_MEGA registers

Register name	Port name	IP name	Absolute address	Reset value	Notes
read_qos	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x100	0	
read_qos	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x100	0	
read_qos	m_d_3	USB	GPV2_BASE + 0x15000 + 0x100	0	
read_qos	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x100	0	
write_qos	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x104	0	
write_qos	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x104	0	

Table continues on the next page...

Table 318. NIC_MEGA registers (continued)

Register name	Port name	IP name	Absolute address	Reset value	Notes
write_qos	m_d_3	USB	GPV2_BASE + 0x15000 + 0x104	0	
write_qos	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x104	0	
fn_mod	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x108	0	
fn_mod	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x108	0	
fn_mod	m_d_3	USB	GPV2_BASE + 0x15000 + 0x108	0	
fn_mod	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x108	0	
qos_cntl	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x10C	0	
qos_cntl	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x10C	0	
qos_cntl	m_d_3	USB	GPV2_BASE + 0x15000 + 0x10C	0	
qos_cntl	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x10C	0	
max_ot	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x110	0	
max_ot	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x110	0	
max_ot	m_d_3	USB	GPV2_BASE + 0x15000 + 0x110	0	
max_ot	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x110	0	
max_comb_ot	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x114	0	
max_comb_ot	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x114	0	

Table continues on the next page...

Table 318. NIC_MEGA registers (continued)

Register name	Port name	IP name	Absolute address	Reset value	Notes
max_comb_ot	m_d_3	USB	GPV2_BASE + 0x15000 + 0x114	0	
max_comb_ot	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x114	0	
aw_p	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x118	0	
aw_p	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x118	0	
aw_p	m_d_3	USB	GPV2_BASE + 0x15000 + 0x118	0	
aw_p	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x118	0	
aw_b	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x11C	0	
aw_b	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x11C	0	
aw_b	m_d_3	USB	GPV2_BASE + 0x15000 + 0x11C	0	
aw_b	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x11C	0	
aw_r	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x120	0	
aw_r	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x120	0	
aw_r	m_d_3	USB	GPV2_BASE + 0x15000 + 0x120	0	
aw_r	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x120	0	
ar_p	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x124	0	
ar_p	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x124	0	

Table continues on the next page...

Table 318. NIC_MEGA registers (continued)

Register name	Port name	IP name	Absolute address	Reset value	Notes
ar_p	m_d_3	USB	GPV2_BASE + 0x15000 + 0x124	0	
ar_p	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x124	0	
ar_b	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x128	0	
ar_b	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x128	0	
ar_b	m_d_3	USB	GPV2_BASE + 0x15000 + 0x128	0	
ar_b	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x128	0	
ar_r	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x12C	0	
ar_r	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x12C	0	
ar_r	m_d_3	USB	GPV2_BASE + 0x15000 + 0x12C	0	
ar_r	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x12C	0	
target_fc	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x130	0	
target_fc	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x130	0	
target_fc	m_d_3	USB	GPV2_BASE + 0x15000 + 0x130	0	
target_fc	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x130	0	
ki_fc	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x134	0	
ki_fc	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x134	0	

Table continues on the next page...

Table 318. NIC_MEGA registers (continued)

Register name	Port name	IP name	Absolute address	Reset value	Notes
ki_fc	m_d_3	USB	GPV2_BASE + 0x15000 + 0x134	0	
ki_fc	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x134	0	
qos_range	m_d_0	uSDHC1	GPV2_BASE + 0x12000 + 0x138	0	
qos_range	m_d_1	uSDHC2	GPV2_BASE + 0x13000 + 0x138	0	
qos_range	m_d_3	USB	GPV2_BASE + 0x15000 + 0x138	0	
qos_range	m_d_4	FLexSPI_FLR	GPV2_BASE + 0x16000 + 0x138	0	

47.4.3 IB registers

There are no IB registers implemented in this chip.

47.4.4 Address region control registers

There are no registers of this type implemented in this chip.

47.4.5 Peripheral ID registers

The peripheral ID registers are implemented in NIC_MAIN and NIC_MEGA. For more details, see the Arm document AMBA® Network Interconnect (NIC-400) Technical Reference Manual, version r2p3.

Chapter 48
Audio Overview

48.1 Audio Overview

The audio subsystem consists of the following modules: SAI-1, SAI-2, SAI-3, SAI-4, PDM, ASRC, and SPDIF. In addition, the IOMUX must be appropriately configured to get signals in and out of the chip.

Audio Module Overview provides an overview of each of the audio modules, followed by a module-specific section.

48.1.1 Audio Module Overview

SAI*n* are synchronous serial interfaces used to transfer audio data. They can be accessed by both the eDMA and Arm CPUs. Their input/output are connected to the pads through IOMUX.

The PDM Microphone Interface (PDM) is used to receive audio from microphone. It supports 8-channel input with individual enable control for full or partial set of channels.

Asynchronous Sample Rate Converter (ASRC) is used for PCM sample rate conversion from one audio sampling clock domain to another.

The Sony/Philips digital interface (SPDIF) audio module is a stereo transceiver that allows the processor to receive and transmit digital audio over it. The SPDIF receiver section includes a frequency measurement block that allows the precise measurement of an incoming sampling frequency. A recovered clock is provided by the SPDIF receiver section and may be used to drive both internal and external components in the system.

The audio interfaces are summarized as the table below.

Table 319. Audio Interface Summary

Interface	Function	RX Data Line	TX Data Line
SAI-1	External audio	2	2
SAI-2	External audio	1	1
SAI-3	External audio	1	1
SAI-4	External audio	4	4
SPDIF	External audio	1	1
PDM (8 channel)	External microphone		
ASRC	Asynchronous Sample Rate Converter		

SAI-4 is used for multi-channel audio interface, it supports up to 8-channels audio input and 8-channels audio output at 384kHz/32-bit. SAI-1, SAI-2, and SAI-3 is used for stereo audio input and output up to 384KHz/32-bit.

To reduce IO count and keep the flexibilities of supporting multiple RX and TX data lines application, SAI-4 has following options on data pin multiplexing. The multiplexing is done in the IOMUX.

Table 320. SAI-4 TX/RX Data Pin Multiplexing

Options	Number of RX Data Line	Number of TX Data Line
0	1	4
1	2	3

Table continues on the next page...

Table 320. SAI-4 TX/RX Data Pin Multiplexing (continued)

Options	Number of RX Data Line	Number of TX Data Line
2	3	2
3	4	1

Detailed clock multiplexing scheme is shown in the following figure.

Please note that SAI-1 is a standalone SAI module and is not shown in the Audio subsystem clocking diagram below.

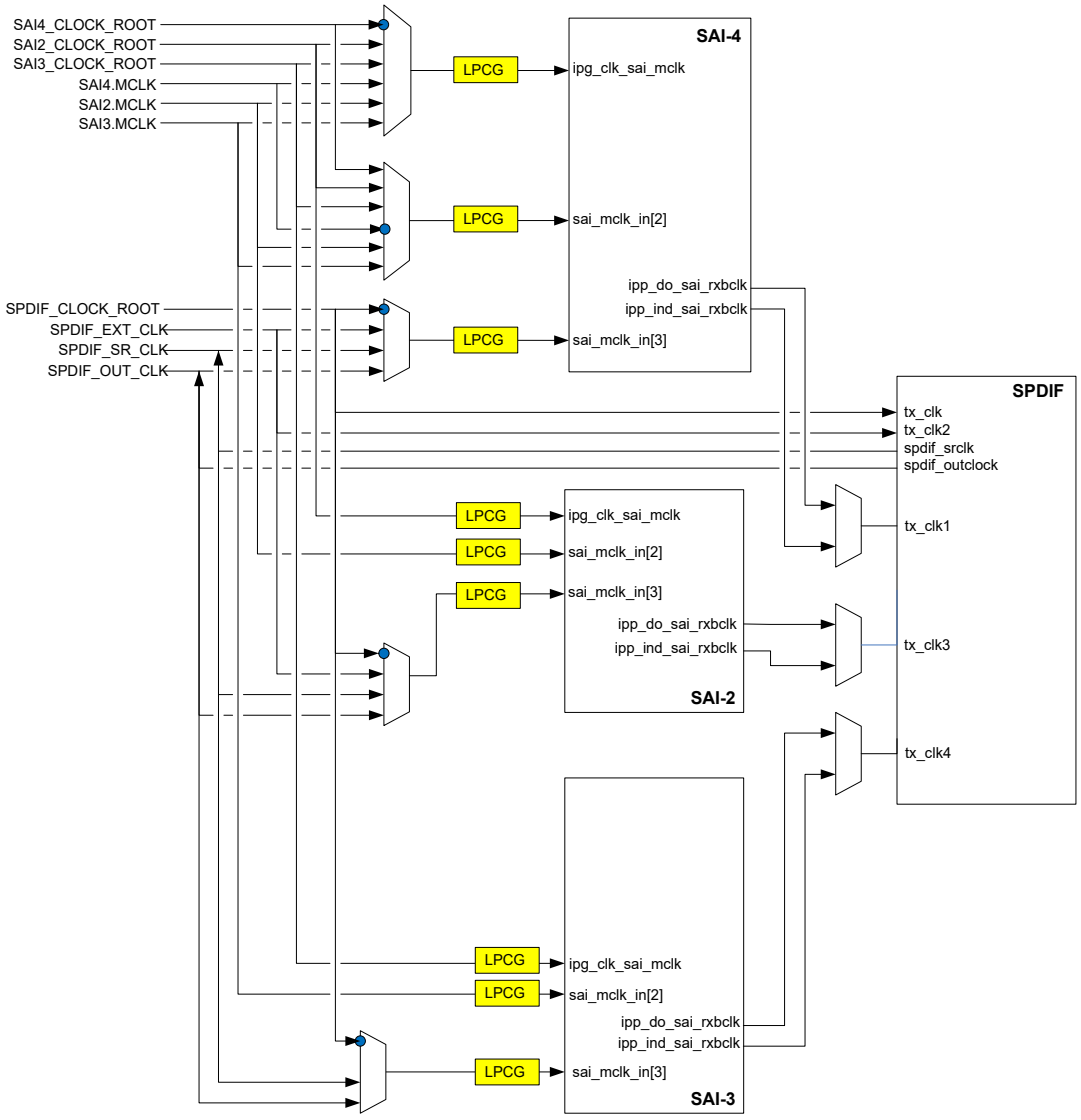


Figure 290. Audio subsystem clocking diagram

NOTE

SPDIF tx_clk1, tx_clk3, and tx_clk4 inputs are selected inside the SAI-4, SAI-2, and SAI-3, respectively.

48.1.2 Synchronous Audio Interface (SAI)

- Transmitter with independent Bit Clock and Frame Sync supporting a maximum of 4 data lines
- Receiver with independent Bit Clock and Frame Sync supporting a maximum of 4 data lines
- Maximum Frame Size of 32 Words
- Word size programmable from 8-bits to 32-bits
- Word size configured separately for first word and remaining words in frame
- Asynchronous FIFO for each Transmit and Receive data line
- Graceful restart after FIFO Error

48.1.3 Sony/Philips Digital Interface (SPDIF)

The Sony/Philips Digital Interface (SPDIF) module is a stereo transceiver that allows the processor to receive and transmit digital audio over it using the IEC60958 standard, consumer format. The chip provides a single SPDIF receiver with one input, and one SPDIF transmitter with one output. The SPDIF transceiver allows the handling of both SPDIF channel status (CS) and User (U) data and features a frequency measurement block that allows the precise measurement of the incoming sampling frequency.

The clock recovered by the SPDIF receiver is provided to drive both internal and external components in the system such as the SPDIF transmitter, SAI ports, as well as external A/Ds or D/As, with clocking control provided via related registers.

The SPDIF is composed of two parts: SPDIF Receiver and SPDIF Transmitter. The SPDIF receiver extracts the audio data from each SPDIF frame and places the data in two 16-word-deep FIFOs, one FIFO for the left channel, the other FIFO for the right channel. The FIFOs support programmable watermark levels so that FIFO Full service request can be triggered when the combined number of data words stored in both FIFOs is 2, 8, 16 or 32 words. It is recommended to program the watermark level to trigger a FIFO Full service request when 16 word locations are filled. For optimal performance when servicing the FIFO Full service request, the FIFOs should be read alternately, starting with the left channel FIFO. The Channel Status and User Bits are also extracted from each frame and placed in the corresponding registers. The SPDIF receiver provides a bypass option for direct transfer of the SPDIF input signal to the SPDIF transmitter.

For the SPDIF transmitter, the audio data is provided by the processor via the SPDIFTxLeft and SPDIFTxRight registers, and the data is stored in two 16-word-deep FIFOs, one for the right channel, the other for the left channel. The FIFOs support programmable watermark levels so that FIFO Empty service request can be triggered when the combined number of empty data words locations in both FIFOs is 8, 16, 24 or 32 words. It is recommended to program the watermark level to trigger a FIFO Empty service request when 16 word locations are empty. For optimal performance when servicing the FIFO Empty service request, the FIFOs should be written alternately, starting with the left channel FIFO. The Channel Status bits are also provided via the corresponding registers. The SPDIF transmitter generates an SPDIF output bitstream in the biphase mark format (IEC 60958), which consists of audio data, channel status and user bits.

The data handled by the SPDIF module is 24-bit wide. The 24-bit SPDIF data is aligned in the 24 least significant bits of the 32-bit shared peripheral bus data word. The 8 most significant bits of the 32-bit word are ignored by the SPDIF Transmitter when data is being stored in the Transmit FIFOs from the peripheral bus. The 8 most significant bits of the 32-bit word are zeroed by the SPDIF Receiver module when the data is being read from the Receiver FIFOs to the peripheral bus.

Note that 16-bit data is left-aligned in the 24-bit word format of the SPDIF. This means that when receiving 16-bit data, it will be located in the middle two bytes of the 32-bit peripheral bus data word, while the 8 bits of the MSB and the 8 bits of the LSB will be zero. When 16-bit data is to be transmitted, the 32-bit word to be written to the SPDIF Transmit FIFOs should be created as follows: the 16-bit data should be located in the middle two bytes of the 32-bit data word and the 8 bits of the LSB must be set to zero, while the 8 bits of the MSB will be ignored.

48.1.4 PDM Microphone Interface (PDM)

The PDM module receives audio from a microphone. It includes the following features:

- Fixed filtering characteristics for audio application

- Full or partial set of channels operation with individual enable control
- Programmable PDM clock generator
- Programmable decimation rate
- 16-bit signed output result
- Overall stopband attenuation more than 80dB
- Overall passband ripple less than 0.2dB
- CIC filter
- DC remover
- Half Band filter
- FIR filter
- FIFOs with DMA capability
- Hardware Voice Activity Detector (HWVAD)
- 4 lines, 8 channels

Chapter 49
Asynchronous Sample Rate Converter (ASRC)

49.1 Chip-specific ASRC information

Table 321. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

NOTE

MQS is not supported in this chip.

This device has one instance of the ASRC module.

The below figure shows the ASRC System Overview for this chip.

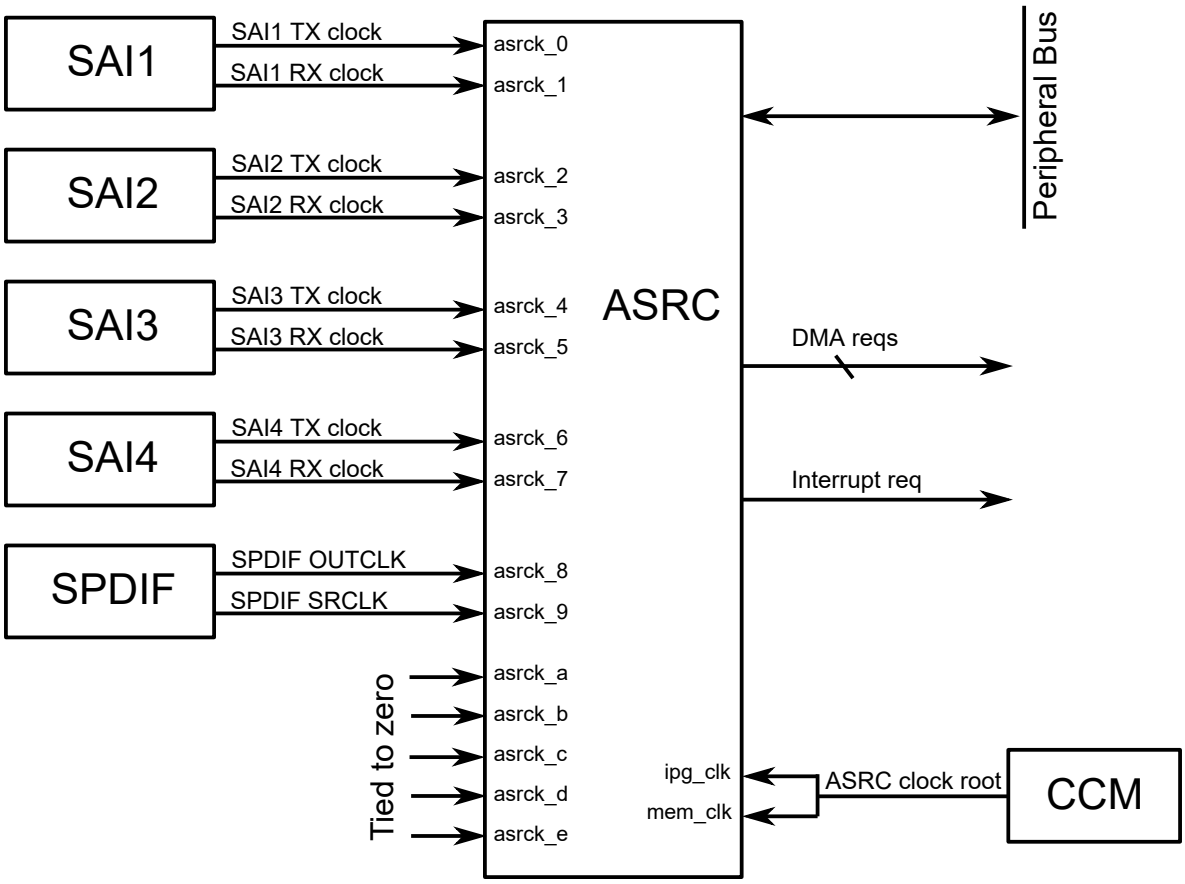


Figure 291. ASRC System Overview

NOTE

Each chip select signal cannot be arbitrarily configured and has its own fixed address range.

49.2 Overview

The Asynchronous Sample Rate Converter (ASRC) converts the sampling rate of a signal associated with an input clock into a signal associated with a different output clock.

The ASRC supports concurrent sample rate conversion of up to 10 channels of about -120 dB THD+N. The ASRC supports up to three sampling rate pairs.

The incoming audio data to this chip may be received from various sources at different sampling rates. The outgoing audio data of this chip may have different sampling rates and it can also be associated with output clocks that are asynchronous to the input clocks.

The ASRC is implemented as a co-processor in hardware, with minimal ARM Platform intervention required.

The ASRC Filter Processor (FP) handles the main signal processing algorithm. It has its own simplified instructions. The FP resources (MIPS,RAM,ROM ALU etc.) are shared among the different conversation pairs.

49.2.1 Block diagram

The following figure is the ASRC block diagram.

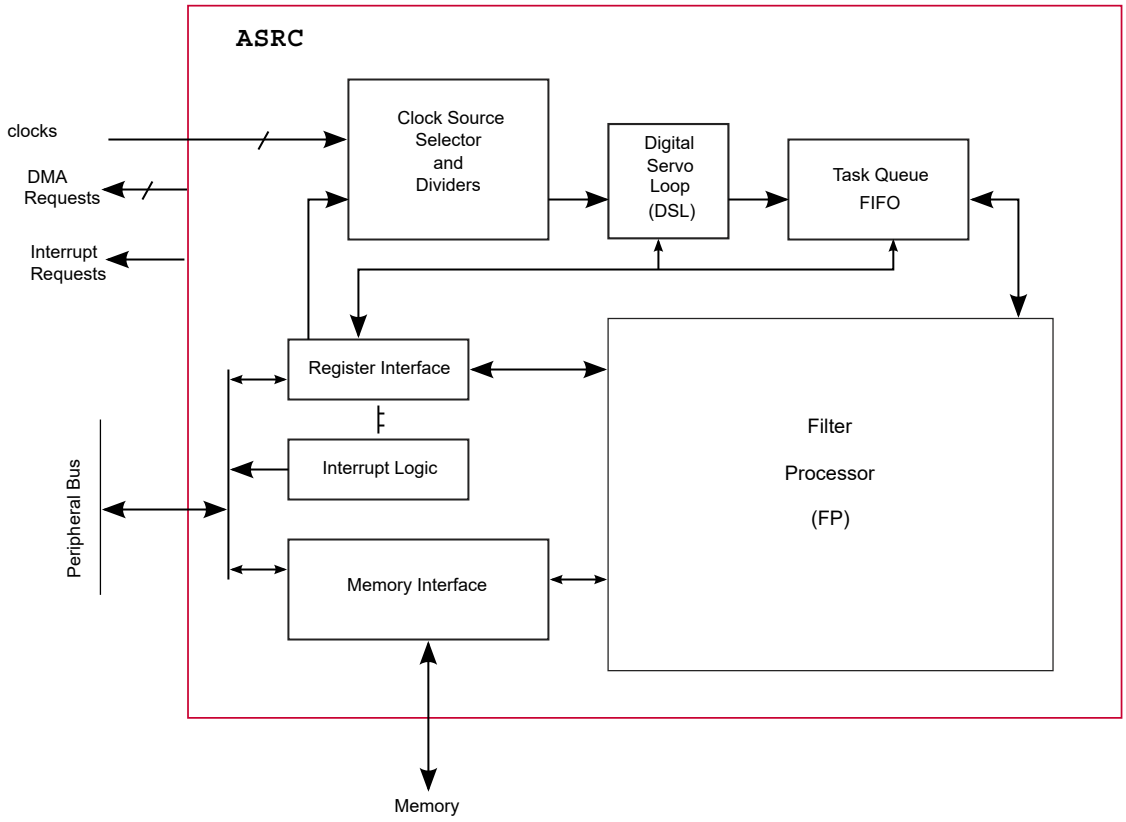


Figure 292. ASRC block diagram

49.2.2 Features

The ASRC features are listed below:

- Any number (0-10) of contiguous channels can be associated to one of the sampling rate pairs
- Support user-programmable threshold for the input/output FIFOs
- Support flexible 8/16/24-bit width of input data, and 16/24-bit width of output data
- Designed for rate conversion between 32 kHz, 44.1 kHz, 48 kHz, 96 kHz, and 192 kHz. The useful signal bandwidth is below 24 kHz
- Other input sampling rates in the range of 8 kHz to 200 kHz is also supported[4]
- Other output sampling rates in the range of 30 kHz to 200 kHz is also supported[4]
  - The limitation is the supported ratio (Fsin/Fsout) range (between 1/24 to 8)
- Automatic accommodation to slow variations in the incoming and outgoing sampling rates.
- Linear phase
- Tolerant to sample clock jitter

[4] The ASRC designed input and output sampling rate are between 44.1 kHz, 32 kHz, 48 kHz, 96 kHz, and 192 kHz. Useful signal bandwidth is below 24 kHz.

- Designed for real-time streaming audio usage. The output sampling clock must be always physically available in the system.

Clock/Data Connections

- The sampling rate clocks are directly connected to the ASRC block. The ratio estimation of the input clocks with output clocks are done in ASRC hardware when both input/output sampling clocks are physically available.
- When both the input sampling clock and the output sampling clock are physically available, the rate conversion can work by configuring the physical clocks.
- When the input sampling clock is not physically available, the rate conversion can still work by setting ideal-ratio values into ASRC interface registers.
- The clock signals come from the following blocks:
  - SAI, receiving bit clock and transmitting bit clock
  - SPDIF, receiving bit clock and transmitting bit clock
  - other audio peripherals etc.
- The Digital Servo Loop (DSL) is a digital PLL for tracking the relationship between input/output sampling clocks. It is shared among all pairs in a first-come-first-serve form
- The exchange of audio data is done by the processor accessing ASRC block through registers defined on shared peripheral bus

49.3 Functional description

This section provides a complete functional description of the block.

49.3.1 Operating modes

The ASRC supports modes described in the following sections:

- Data transfer schemes
- Word alignment supported

49.3.1.1 Data transfer schemes

49.3.1.1.1 Data input modes

The input mode for each of the three channel sets may be set independently. Three modes of supplying data to the ASRC input FIFOs are available:

- Polling
- Interrupt
- DMA

In all input-data transfer schemes, the ASRC fetches data from each enabled FIFO and processes the data sample-by-sample after each rising edge of the associated input sampling clock until the FIFO level reaches a threshold.

After the threshold is reached, the ASRC requests data. The FIFO size for each channel set is 64 samples and the threshold is set at 32 samples.

The threshold can be defined by interface registers ASRMCRx, x = A, B or C.

If the ASRC attempts to fetch data from an empty FIFO, an error is generated and the ASRSTR[AOLE] bit is set. If the ASRC overload interrupt is enabled (ASRIER[AOLIE] bit is set), an interrupt is generated.

When writing data to an input FIFO, you must ensure that it is in a predefined sequence. For example, when writing to an input FIFO, the sequence should be: channel_0, channel_1, channel_2,..., channel_n, channel_0, channel_1, channel_2, etc. Here

channel_n stands for the data intended for the n^{th} channel. The hardware re-allocates each data to its corresponding channel FIFO. The channel being re-allocated is shown by ASRCCR[ACIx], x=A,B or C.

49.3.1.1.1.1 Data input polling mode

Polling mode is the default mode following power-on or individual reset, and is selected by clearing the associated channel set A, B, or C data-input interrupt enable bit (ASRIER[ADIEx], where x=A, B or C). In this mode, data-input interrupts are disabled. When the FIFO level is below the threshold, the associated status bit (ASRSTR[AIDEx], where x=A, B, or C) is set. To clear the status bit, the FIFO must be written with enough data to raise the level above the threshold.

49.3.1.1.1.2 Data input interrupt mode

The ASRC input FIFOs can also be serviced by interrupts. To enable interrupts, the corresponding data-input interrupt enable bits (ASRIER[ADIEx], where x=A, B, or C) should be set. An interrupt is automatically generated any time the input FIFO level is below the threshold. The interrupt is cleared when enough data is written to the FIFO to raise the level above the threshold.

49.3.1.1.1.3 Data input DMA mode

The ASRC input FIFOs can also be filled using DMA. In this mode, the data-input interrupt-enable bits (ASRIER[ADIEx], where x=A, B, or C) should be cleared and the DMA controller should be configured to use the ASRC as a request source.

49.3.1.1.2 Data output modes

The output mode for each of the 3 channel sets (A, B, and C) may be set independently.

Three modes of retrieving data from the ASRC output FIFOs are available:

- Polling
- Interrupt
- DMA

In all output-data transfer schemes, the ASRC places a processed sample into the associated output FIFO. After a threshold is reached, the ASRC requests that data be transferred out of the FIFO.

The FIFO size for each channel set is 64 samples and the threshold is set at 32 samples. The threshold can be defined by interface registers ASRMCRx, x = A, B or C.

If the ASRC attempts to place data into a FIFO that is already full, an error is generated and the ASRSTR[AOLE] bit is set. If the ASRC overload interrupt is enabled (ASRIER[AOLIE] bit is set), an interrupt is generated.

Each output FIFO is organized in the same channel order in which the associated input FIFO was written.

Three transfer modes are supported by Interface Block.

49.3.1.1.2.1 Data output polling mode

The ASRC output FIFOs can be serviced by polling. In this mode, ensure the associated output-data interrupt enable bit (ASRIER[ADOEx], where x=A, B, or C) is cleared. In this mode, all output-data interrupts are disabled. Any time the output FIFO exceeds the threshold the associated status bit (ASRSTR[AODFx], where x=A, B, or C) is set. To clear the status bit, enough data must be read from the associated output FIFO to lower the level below the threshold.

49.3.1.1.2.2 Data output interrupt mode

The ASRC output FIFOs may also be serviced using interrupts. To enable this mode, the corresponding output-data interrupt-enable bits (ASRIER[ADOEx], where x=A, B, or C) should be set. Any time the output FIFO level exceeds the threshold, an interrupt is automatically generated. The interrupt is cleared when enough data is read from the FIFO to lower the level below the threshold.

49.3.1.1.2.3 Data output DMA mode

The ASRC output FIFOs can also be read using DMA. In this mode, the output-data interrupt-enable bits (ASRIER[ADOEx], where x=A, B, or C) should be cleared and the DMA controller should be configured to use the ASRC as a request source.

49.3.1.2 Word alignment supported

49.3.1.2.1 Input data alignment modes

The position and length of input data word to the input data FIFOs ASRDIA, ASRDIB, ASRDIC are programmable. The control bits are defined in ASRMCR1x {x=A, B, or C}. It supports the following modes:

Table 322. Input data alignment

Format	Bit number																															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
8-bit LSB Aligned																									7	6	5	4	3	2	1	0
8-bit MSB Aligned																	7	6	5	4	3	2	1	0								
16-bit LSB Aligned																	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
16-bit MSB Aligned	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																
24-bit LSB Aligned									23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
24-bit MSB Aligned	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								

49.3.1.2.2 Output data alignment modes

The position and length of output data word from the output data FIFOs ASRDOA, ASRDOB, ASRDOC are programmable. The control bits are defined in ASRMCR1x {x=A, B, or C}. It supports the following modes:

Table 323. Output data alignment

Format	Bit number																															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
16-bit LSB Aligned																	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
16-bit LSB Aligned with Sign Extension	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Table continues on the next page...

Table 323. Output data alignment (continued)

Format	Bit number																															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
16-bit MSB Aligned	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																
24-bit LSB Aligned									23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
24-bit LSB Aligned with Sign Extension	23	23	23	23	23	23	23	23	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
24-bit MSB Aligned	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								

49.3.2 Algorithm description

49.3.2.1 Signal processing flow

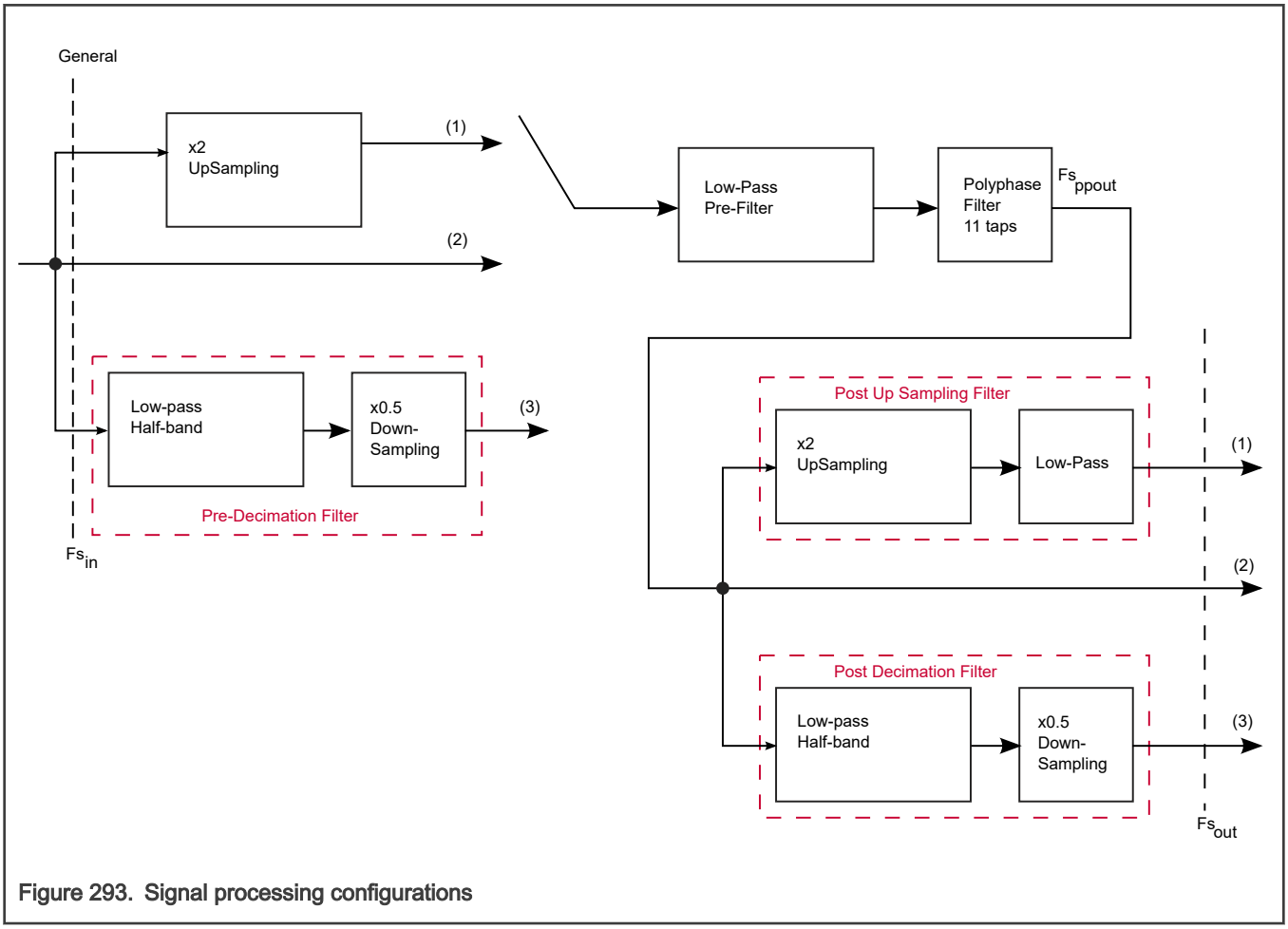


Figure 293. Signal processing configurations

The figure above shows the possible configurations of the ASRC. Each configuration consists of 2 to 4 stages.

- x2 up-sampling rate expander (zero insertion only) (input branch 1), direct connection (input branch 2), or low-pass pre decimation filter (consisting of a low-pass half-band FIR filter with x0.5 downsampling rate decimator) (input branch 3),
- low-pass pre-filter, the low-pass bandwidth is at most 0.25 x Fs, where Fs is the sampling rate of the input signal to this low-pass pre-filter,
- polyphase filter,
- x2 post upsampling filter (consisting of a x2 up-sampling rate expander (zero insertion only) with low-pass half-band FIR filter) (output branch 1), direct connection (output branch 2), or low-pass post decimation filter (consisting of a low-pass half-band FIR filter with x0.5 downsampling rate decimator) (output branch 3).

By flowing through different processing branches and different setups of the pre-filter, this ASRC scheme can be used to handle different rate conversion requirements.

Table 324. ASRC rate conversion configurations

Configuration	Max signal bandwidth before polyphase filter	Signal sampling rate of polyphase filter output
a: Input Branch 1 + Output Branch 1	$BW_{in} = Fs_{in}/2$	$Fs_{ppout} = Fs_{out}/2$
b: Input Branch 1 + Output Branch 2	$BW_{in} = Fs_{in}/2$	$Fs_{ppout} = Fs_{out}$
c: Input Branch 1 + Output Branch 3	$BW_{in} = Fs_{in}/2$	$Fs_{ppout} = 2Fs_{out}$
d: Input Branch 2 + Output Branch 1	$BW_{in} = Fs_{in}/4$	$Fs_{ppout} = Fs_{out}/2$
e: Input Branch 2 + Output Branch 2	$BW_{in} = Fs_{in}/4$	$Fs_{ppout} = Fs_{out}$
f: Input Branch 2 + Output Branch 3	$BW_{in} = Fs_{in}/4$	$Fs_{ppout} = 2Fs_{out}$
g: Input Branch 3 + Output Branch 1	$BW_{in} = Fs_{in}/8$	$Fs_{ppout} = Fs_{out}/2$
h: Input Branch 3 + Output Branch 2	$BW_{in} = Fs_{in}/8$	$Fs_{ppout} = Fs_{out}$
i: Input Branch 3 + Output Branch 3	$BW_{in} = Fs_{in}/8$	$Fs_{ppout} = 2Fs_{out}$

Table 325 lists some typical settings for typical sample rate conversation pairs to achieve less MIPS. ASRC can be set to automatically fill these settings depending on input/output sample rate or users can configure the fields manually to achieve better filter bandwidth performance.

Table 325. Pre-processing, post-processing options

{Pre_Proc, Post_Proc}		Fsout (kHz)											
		8	12	16	24	32	44.1	48	64	88.2	96	128	192
Fsin (kHz)	8	{0,1}	{0,1}	{0,1}	{0,0}	{0,0}	{0,0}	{0,0}	{0,0}	{0,0}	{0,0}	{0,0}	{0,0}
	12	{0,2}	{0,1}	{0,1}	{0,1}	{0,1}	{0,0}	{0,0}	{0,0}	{0,0}	{0,0}	{0,0}	{0,0}
	16	{1,2}	{0,2}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0,0}	{0,0}	{0,0}	{0,0}	{0,0}
	24	{1,2}	{1,2}	{0,2}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0,0}	{0,0}	{0,0}	{0,0}

Table continues on the next page...

Table 325. Pre-processing, post-processing options (continued)

{Pre_Proc, Post_Proc}		Fsout (kHz)											
		8	12	16	24	32	44.1	48	64	88.2	96	128	192
	32	{1,2}	{1,2}	{1,2}	{0,2}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0,0}	{0,0}	{0,0}
	44.1	{2,2}	{1,2}	{1,2}	{0,2}	{0,2}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0,0}	{0,0}
	48	{2,2}	{1,2}	{1,2}	{1,2}	{0,2}	{0,2}	{0,1}	{0,1}	{0,1}	{0,1}	{0,1}	{0,0}
	64	{2,2}	{2,2}	{1,2}	{1,2}	{0,2}	{0,2}	{0,2}	{0,1}	{0,1}	{0,1}	{0,1}	{0,0}
	88.2	NA	{2,2}	{2,2}	{1,2}	{1,2}	{1,2}	{1,2}	{1,1}	{1,1}	{1,1}	{1,1}	{1,1}
	96	NA	{2,2}	{2,2}	{1,2}	{1,2}	{1,2}	{1,2}	{1,1}	{1,1}	{1,1}	{1,1}	{1,1}
	128	NA	NA	{2,2}	{2,2}	{1,2}	{1,2}	{1,2}	{1,1}	{1,1}	{1,1}	{1,1}	{1,1}
	192	NA	NA	NA	{2,2}	{2,2}	{2,2}	{2,2}	{2,1}	{2,1}	{2,1}	{2,1}	{2,1}

NOTE

In the {Pre_Proc, Post_Proc} pair, the meaning of the values are:

Pre_Proc:

- 0 --- Pre-processing Branch 1 as shown in Figure 293
- 1 --- Pre-processing Branch 2 as shown in Figure 293
- 2 --- Pre-processing Branch 3 as shown in Figure 293, decimation-by-2

Post_Proc:

- 0 --- Post-processing Branch 1 as shown in Figure 293
- 1 --- Post-processing Branch 2 as shown in Figure 293
- 2 --- Post-processing Branch 3 as shown in Figure 293

The latencies of the different option can be roughly calculated as follows:

- For PreProc = 0, PostProc = 1 : min latency = constant_A / input-sample-rate + constant_B / output-sample-rate
- For PreProc = 0, PostProc = 0 : min latency = constant_A / input-sample-rate + constant_C / output-sample-rate
- For PreProc = 1, PostProc = 1 : min latency = constant_D / input-sample-rate + constant_B / output-sample-rate

The constants above (e.g., constant_A means the Constant for Preproc = 0, constant_B means the Constant for Postproc = 1, ...) are only influenced by the PreProc/PostProc and (input/output) sampling rate to which they are connected. Input latencies have no relationship with the output latencies, but both elements add together to form the total latencies.

For a rough estimation, the constants can be set as:

- Constant for Preproc = 0: 39
- Constant for Preproc = 1: 78.5
- Constant for Preproc = 2: 235
- Constant for Postproc = 0: 42.5
- Constant for Postproc = 1: 8.5

- Constant for Postproc = 2: 172

The max latency can be derived from this value by using the following formula (where 32 means the input/output FIFO depth that initiates data transfer):

- max latency = min latency + 32 / input-sample-rate + 32 / output-sample-rate

49.3.2.2 Operation of the filter

49.3.2.2.1 Support of physical clocks

This design supports physical sampling clocks. The clocks can be provided by:

- Sony/Phillips digital interface (SPDIF)
- Serial Audio Interface (SAI)
- Core master clock derivative as ASRCK1

NOTE

Your specific device may not support all of the clock sources. See the chip-specific section for this module, at the beginning of this chapter.

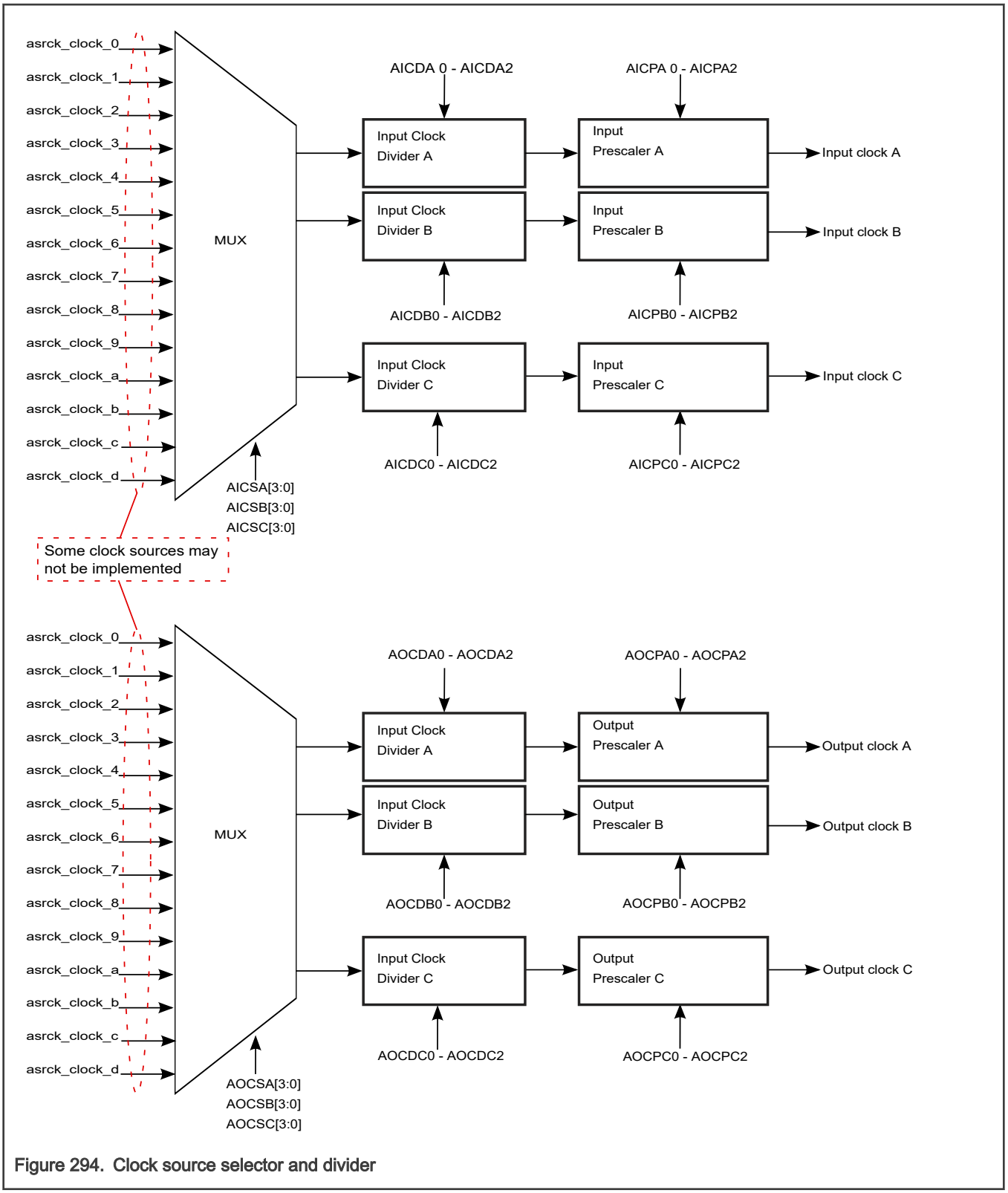


Figure 294. Clock source selector and divider

Software can set the ASRC Clock Source Register (ASRCSR) and the Clock Divider Register to select the desired clock source and divide it to the needed sample rate clock for use by the ASRC.

The following is the description of the Clock Source Selector and Divider figure above (x refers to A, B or C):

- If the input clock is available, Pair x divider and prescaler depends on the input clock and input sample rate, which can be calculated by:
  - Input Clock x = (Input Sample Rate x) * AICDx * (2^(AICPx))
  - Where the Input Sample Clock Source x is determined by ASRCSR[AICSx]
- If the output clock is available, Pair x divider and prescaler depends on the output clock and output sample rate, which can be calculated by:
  - Output Clock x = (Output Sample Rate x) * AOCDx * (2^(AOCPx))
  - Where the Output Sample Clock Source x is determined by ASRCSR[AOCSx]

NOTE

The clocks have the following restriction: If the prescaler is set to 1, the clock divider can only be set to 1 and the clock source must have a 50% duty cycle.

NOTE

For normal setup procedure for the ASRC block, see section Application information.

49.3.3 Clocks

The table below describes the clock sources for ASRC. Please see clock control block for clock setting, configuration and gating information.

Table 326. ASRC clocks

Clock name	Description
asrck_clock_d	ASRC module clock
ipg_clk	Peripheral clock
mem_clk	Peripheral access clock

49.3.4 Interrupts

ASRC has several interrupts events.

The priorities are shown in the following table.

Table 327. Interrupt priorities/vector

Priority	Offset	Description
lowest	0x0	ASRC Pair A input data needed
	0x2	ASRC Pair B input data needed
	0x4	ASRC Pair C input data needed
	0x6	ASRC Pair A output data ready
	0x8	ASRC Pair B output data ready
	0xA	ASRC Pair C output data ready
highest	0xC	ASRC Overload

49.3.5 DMA requests

ASRC has six DMA requests. They are directly connected to the lowest six status bits in the ASRSTR register. For some ARM platforms, the six status bits are directly connected to the DMA peripheral as DMA event signals.

Table 328. DMA requests

Type	Description
0	ASRC Pair A input data needed
1	ASRC Pair B input data needed
2	ASRC Pair C input data needed
3	ASRC Pair A output data ready
4	ASRC Pair B output data ready
5	ASRC Pair C output data ready

49.4 External signals

The ASRC does not have external signals pinned out.

The following table shows the bit definitions that are used by the ASRCSR register to control the sources of the input and output clocks of the ASRC.

NOTE

Your specific device may not support all of the clock sources. See the chip-specific section for this module, at the beginning of this chapter.

Table 329. ASRC bit clock definitions

Bit Clk Name	Definition	Description
bit clock 0	SAI1_TX_BCLK	SAI1 Transmitter serial bit clock
bit clock 1	SAI1_RX_BCLK	SAI1 Receiver serial bit clock
bit clock 2	SAI2_TX_BCLK	SAI2 Transmitter serial bit clock
bit clock 3	SAI2_RX_BCLK	SAI2 Receiver serial bit clock
bit clock 4	SAI3_TX_BCLK	SAI3 Transmitter serial bit clock
bit clock 5	SAI3_RX_BCLK	SAI3 Receiver serial bit clock
bit clock 6	SAI4_TX_BCLK	SAI4 Transmitter serial bit clock
bit clock 7	SAI4_RX_BCLK	SAI4 Receiver serial bit clock
bit clock 8	SPDIF_TX_BCLK	SPDIF Transmitter serial bit clock
bit clock 9	SPDIF_RX_BCLK	SPDIF Receiver serial bit clock

Table continues on the next page...

Table 329. ASRC bit clock definitions (continued)

Bit Clk Name	Definition	Description
bit clock A	SAI2_CLK_ROOT	SAI2 Clock Root
bit clock B	SAI3_CLK_ROOT	SAI3 Clock Root
bit clock C	SAI4_CLK_ROOT	SAI4 Clock Root
bit clock D	MIC_CLK_ROOT	MIC Clock Root
bit clock E	MQS_CLK_ROOT	MQS Clock Root

49.5 Application information

The following outlines an **example** setup procedure for the ASRC block:

1. Configure and enable the ASRC clock source. See this device's clocking chapter for more information
2. Disable the ASRC (write 0x00000000 to ASRCTR)
3. Enable the peripheral that is connected to the ASRC. See this device's clocking chapter and the ASRC chip specific section for more information
4. Configure the following:
    a. Enable ASRC function(ASRCTR[ASRCEN]=1)
    b. Enable Use of Ideal Ratio for Pair A (ASRCTR[IDRA]=1)
    c. Enable Use of Ratio for Pair A (ASRCTR[URSA]=1)
    d. Assign 2 channels to Pair A (ASRCNCR[ANCA]=2)
    e. Configure Post-Processing Configuration and Pre-Processing Configuration for Conversion Pair A. See Signal processing flow
    f. Set ASRC Clock Divider 1 (ASRCDR1) and ASRC Clock Divider 2 (ASRCDR2. See Support of physical clocks for more information
    g. Set the Input Clock Source A (ASRCSR[AICSA]) and Output Clock Source A (ASRCSR[AOCSA]) both to Bit Clock n

    NOTE

    Select one of the bit clocks supported by your specific device. See the chip-specific section for this module, at the beginning of this chapter.

    h. Set the Ideal Ratio for Pair A. The ratio depends on input sample rate and output sample rate. See the ASRC Ideal Ratio for Pair A-High Part (ASRIDRHA) and ASRC Ideal Ratio for Pair A -Low Part (ASRIDRLA) register descriptions. If the input sample rate is equal to the output sample rate, Ideal Ratio is 1:
        - Ideal Ratio for Pair A-High Part (ASRIDRHA = 0x4)
        - Ideal Ratio for Pair A-Low Part (ASRIDRLA = 0x0)
5. For Pair A - Set input threshold to 32 and output threshold to 8 (ASRMCRA[INFIFO_THRESHOLDA], ASRMCRA[OUTFIFO_THRESHOLDA])
6. Zeroize Pair A buffers(ASRMCRA[ZEROBUFA]=1)
7. Do not bypass polyA filter (ASRMCRA[BYPASSPOLYA]=0)
8. Set Pair A data width (ASRMCR1A[IWD]) and data alignment(ASRMCR1A[IMSB] and ASRC_ASRMCR1A[OMSB]) depends on the audio data format

9. Disable interrupt handler (ASRIER = 0x00000000)
10. After configuration is done, enable the ASRC pair(ASRCTR[ASREA]=1)
11. Poll until the initialization is done (ASRCFG and ASRCFG[INIRQA]) == 0)
12. Feed Input Data (ASRDIA)
13. Get Output Data (ASRDOA)
14. Disable ASRC (ASRCTR[ASRCEN] = 0)

49.6 ASRC memory map/register definition

49.6.1 ASRC register descriptions

All useful registers are listed in the memory map below. The access of undefined registers behaves as normal registers.

All the interface registers are LSB aligned except the input FIFOs and the output FIFOs, and each register has only 24 effective bits.

The input FIFO and output FIFO word alignment can be defined using ASRMCR1{A,B,C} registers in 32-bit interface system.

49.6.1.1 ASRC memory map

ASRC base address: 429A_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ASRC Control (ASRCTR)	32	RW	0000_0000h
4h	ASRC Interrupt Enable (ASRIER)	32	RW	0000_0000h
Ch	ASRC Channel Number Configuration (ASRCNCR)	32	RW	0000_0000h
10h	ASRC Filter Configuration Status (ASRCFG)	32	RW	0000_0000h
14h	ASRC Clock Source (ASRCSR)	32	RW	0000_0000h
18h	ASRC Clock Divider 1 (ASRCDR1)	32	RW	0000_0000h
1Ch	ASRC Clock Divider 2 (ASRCDR2)	32	RW	0000_0000h
20h	ASRC Status (ASRSTR)	32	R	0000_0000h
40h - 50h	ASRC Parameter x (ASRPM1 - ASRPM5)	32	RW	0000_0000h
54h	ASRC Task Queue FIFO 1 (ASRTFR1)	32	RW	0000_0000h
5Ch	ASRC Channel Counter (ASRCCR)	32	RW	0000_0000h
60h	ASRC Data Input for Pair x (ASRDIA)	32	RW	0000_0000h
64h	ASRC Data Output for Pair x (ASRDOA)	32	R	0000_0000h
68h	ASRC Data Input for Pair x (ASRDIB)	32	RW	0000_0000h
6Ch	ASRC Data Output for Pair x (ASRDOB)	32	R	0000_0000h
70h	ASRC Data Input for Pair x (ASRDIC)	32	RW	0000_0000h
74h	ASRC Data Output for Pair x (ASRDOC)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
80h	ASRC Ideal Ratio for Pair A-High Part (ASRIDRHA)	32	RW	0000_0000h
84h	ASRC Ideal Ratio for Pair A -Low Part (ASRIDRLA)	32	RW	0000_0000h
88h	ASRC Ideal Ratio for Pair B-High Part (ASRIDRHB)	32	RW	0000_0000h
8Ch	ASRC Ideal Ratio for Pair B-Low Part (ASRIDRLB)	32	RW	0000_0000h
90h	ASRC Ideal Ratio for Pair C-High Part (ASRIDRHC)	32	RW	0000_0000h
94h	ASRC Ideal Ratio for Pair C-Low Part (ASRIDRLC)	32	RW	0000_0000h
98h	ASRC 76 kHz Period (ASR76K)	32	RW	0000_0A47h
9Ch	ASRC 56 kHz Period (ASR56K)	32	RW	0000_0DF3h
A0h	ASRC Misc Control for Pair A (ASRMCRA)	32	RW	0000_0000h
A4h	ASRC FIFO Status for Pair A (ASRFSTA)	32	R	0000_0000h
A8h	ASRC Misc Control for Pair B (ASRMCRB)	32	RW	0000_0000h
ACh	ASRC FIFO Status for Pair B (ASRFSTB)	32	R	0000_0000h
B0h	ASRC Misc Control for Pair C (ASRMCRC)	32	RW	0000_0000h
B4h	ASRC FIFO Status for Pair C (ASRFSTC)	32	R	0000_0000h
C0h - C8h	ASRC Misc Control 1 for Pair X (ASRMCR1A - ASRMCR1C)	32	RW	0000_0000h

49.6.1.2 ASRC Control (ASRCTR)

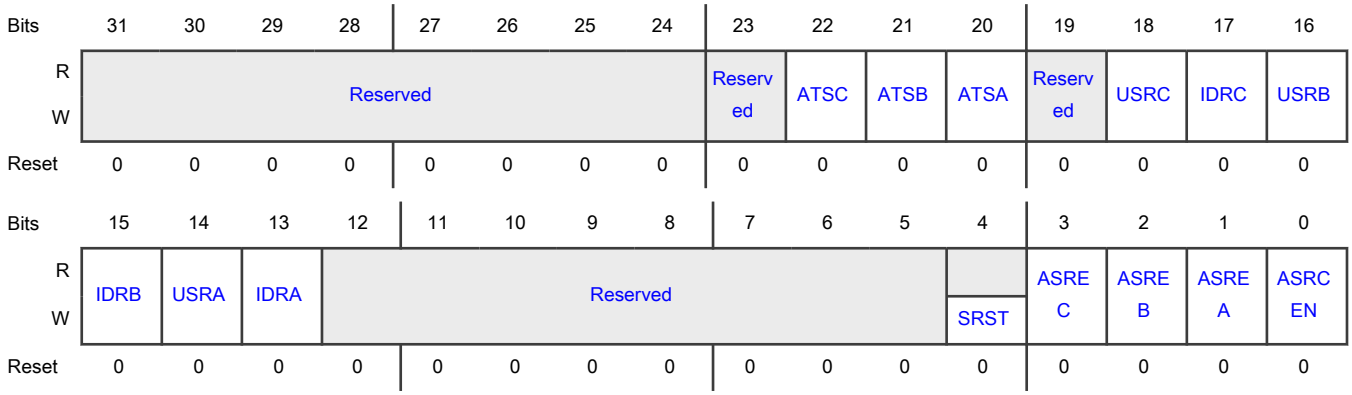
Offset

Register	Offset
ASRCTR	0h

Function

Controls the ASRC operations.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 —	Should be written as zero for compatibility.
22 ATSC	ASRC Pair C Automatic Selection For Processing Options When this bit is 1, pair C automatically updates its pre-processing and post-processing options (ASRCFG[PREMODC], ASRCFG[POSTMODC] see ASRC Filter Configuration Status Register for Pair C) based on the frequencies it detected. To use this option, the two parameter registers(ASR76K and ASR56K) should be set correctly (see ASRC 76 kHz Period in terms of ASRC processing clock and ASRC 56 kHz Period in terms of ASRC processing clock). When this bit is 0, the user is responsible for choosing the proper processing options for pair C. This bit should be disabled when {USRC, IDRC}={1,1}. 0b - Pair C does not automatically update its pre-processing and post-processing options 1b - Pair C automatically updates its pre-processing and post-processing options
21 ATSB	ASRC Pair B Automatic Selection For Processing Options When this bit is 1, pair B automatically updates its pre-processing and post-processing options (ASRCFG[PREMODB], ASRCFG[POSTMODB] see ASRC Filter Configuration Status Register for Pair B) based on the frequencies it detected. To use this option, the two parameter registers(ASR76K and ASR56K) should be set correctly (see ASRC 76 kHz Period in terms of ASRC processing clock and ASRC 56 kHz Period in terms of ASRC processing clock). When this bit is 0, the user is responsible for choosing the proper processing options for pair B. This bit should be disabled when {USRB, IDRB}={1,1}. 0b - Pair B does not automatically update its pre-processing and post-processing options 1b - Pair B automatically updates its pre-processing and post-processing options
20	ASRC Pair A Automatic Selection For Processing Options

Table continues on the next page...

Table continued from the previous page...

Field	Function
ATSA	When this bit is 1, pair A automatically updates its pre-processing and post-processing options (ASRCFG[PREMODA], ASRCFG[POSTMODA] see ASRC Filter Configuration Status Register for Pair A) based on the frequencies it detected. To use this option, the two parameter registers(ASR76K and ASR56K) should be set correctly (see ASRC 76 kHz Period in terms of ASRC processing clock and ASRC 56 kHz Period in terms of ASRC processing clock). When this bit is 0, the user is responsible for choosing the proper processing options for pair A. This bit should be disabled when {USRA, IDRA}={1,1}. 0b - Pair A does not automatically update its pre-processing and post-processing options 1b - Pair A automatically updates its pre-processing and post-processing options
19 —	Should be written as zero for compatibility.
18 USRC	Use Ratio for Pair C Use ratio as the input to ASRC. This bit is used in conjunction with IDRC control bit. 0b - Do not use ratio as the input to ASRC for pair C 1b - Use ratio as the input to ASRC for pair C
17 IDRC	Use Ideal Ratio for Pair C When USRC=0, this bit has no usage. When USRC=1 and IDRC=0, ASRC internal measured ratio is used. When USRC=1 and IDRC=1, the idea ratio from the interface register ASRIDRHC, ASRIDRLC is used. It is suggested to manually set ASRCFG[POSTMODC], ASRCFG[PREMODC] according to Pre-processing, post-processing options in this case. 0b - ASRC internal measured ratio is used 1b - Ideal ratio from the interface register ASRIDRHC, ASRIDRLC is used
16 USRB	Use Ratio for Pair B Use ratio as the input to ASRC. This bit is used in conjunction with IDRB control bit. 0b - Do not use ratio as the input to ASRC for pair B 1b - Use ratio as the input to ASRC for pair B
15 IDRB	Use Ideal Ratio for Pair B When USRB=0, this bit has no usage. When USRB=1 and IDRB=0, ASRC internal measured ratio is used. When USRB=1 and IDRB=1, the idea ratio from the interface register ASRIDRHB, ASRIDRLB is used. It is suggested to manually set ASRCFG[POSTMODB], ASRCFG[PREMODB] according to Pre-processing, post-processing options in this case. 0b - ASRC internal measured ratio is used

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Ideal ratio from the interface register ASRIDRHB, ASRIDRLB is used
14 USRA	Use Ratio for Pair A Use ratio as the input to ASRC. This bit is used in conjunction with IDRA control bit. 0b - Do not use ratio as the input to ASRC for pair A 1b - Use ratio as the input to ASRC for pair A
13 IDRA	Use Ideal Ratio for Pair A When USRA=0, this bit has no usage. When USRA=1 and IDRA=0, ASRC internal measured ratio is used. When USRA=1 and IDRA=1, the ideal ratio from the interface register ASRIDRHA, ASRIDRLA is used. It is suggested to manually set ASRCFG[POSTMODA], ASRCFG[PREMODA] according to Pre-processing, post-processing options in this case. 0b - ASRC internal measured ratio is used 1b - Ideal ratio from the interface register ASRIDRHA, ASRIDRLA is used
12-5 —	Should be written as zero for compatibility.
4 SRST	Software Reset This bit is self-clear bit. Once it has been written as 1, it generates a software reset signal inside ASRC. After 9 cycles of the ASRC processing clock, this reset process stops, and this bit is cleared automatically. 0b - ASRC Software reset cleared 1b - ASRC Software reset generated. NOTE: This is a self-clear bit
3 ASREC	ASRC Enable C Enable the operation of the conversion C of ASRC. 0b - Disabled 1b - Enabled
2 ASREB	ASRC Enable B Enables the operation of the conversion B of ASRC. 0b - Disabled 1b - Enabled
1 ASREA	ASRC Enable A Enables the operation of the conversion A of ASRC. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
0 ASRCEN	ASRC Enable Enables the operation of ASRC. 0b - Disabled 1b - Enabled

49.6.1.3 ASRC Interrupt Enable (ASRIER)

Offset

Register	Offset
ASRIER	4h

Function

Enables ASRC data input, data output, overload, and wait state interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								AFPWE	AOLIE	ADOEC	ADOEB	ADOEA	ADIEC	ADIEB	ADIEA
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-8 —	Should be written as zero for compatibility.
7	FP in Wait State Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
AFPWE	Enables the FP in wait state interrupt. 0b - Disabled 1b - Enabled
6 AOLIE	Overload Interrupt Enable Enables the overload interrupt. 0b - Disabled 1b - Enabled
5 ADOEC	Pair C Data Output Interrupt Enable Enables the data output C interrupt. 0b - Disabled 1b - Enabled
4 ADOEB	Pair B Data Output Interrupt Enable Enables the data output B interrupt. 0b - Disabled 1b - Enabled
3 ADOEA	Pair A Data Output Interrupt Enable Enables the data output A interrupt. 0b - Disabled 1b - Enabled
2 ADIEC	Pair C Data Input Interrupt Enable Enables the data input C interrupt. 0b - Disabled 1b - Enabled
1 ADIEB	Pair B Data Input Interrupt Enable Enables the data input B interrupt. 0b - Disabled 1b - Enabled
0 ADIEA	Pair A Data Input Interrupt Enable Enables the data input A Interrupt. 0b - Disabled 1b - Enabled

49.6.1.4 ASRC Channel Number Configuration (ASRCNCR)

Offset

Register	Offset
ASRCNCR	Ch

Function

Sets the number of channels used by each ASRC conversion pair.

There are 10 channels available for distribution among 3 conversion pairs, they are ordered as 0,1,...,9. The bottom [0, ANCA-1] channels are used for pair A, the top [10-ANCC, 9] channels are used for pair C, and the [ANCA, ANCA+ANCB-1] channels are allocated for pair B. In case that ANCA=0, then the [0, ANCB-1] channels are assigned for pair B.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				ANCC				ANCB				ANCA			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-12 —	Should be written as zero for compatibility.
11-8 ANCC	Number of C Channels ANCC+ANCB+ANCA <= 10. Hardware is not checking the constraint. Programmer should take the responsibility to ensure the constraint is satisfied. 0000b - 0 channels in C (Pair C is disabled) 0001b - 1 channel in C 0010b - 2 channels in C 0011b - 3 channels in C

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - 4 channels in C 0101b - 5 channels in C 0110b - 6 channels in C 0111b - 7 channels in C 1000b - 8 channels in C 1001b - 9 channels in C 1010b - 10 channels in C 1011b-1111b - Should not be used.
7-4 ANCB	Number of B Channels 0000b - 0 channels in B (Pair B is disabled) 0001b - 1 channel in B 0010b - 2 channels in B 0011b - 3 channels in B 0100b - 4 channels in B 0101b - 5 channels in B 0110b - 6 channels in B 0111b - 7 channels in B 1000b - 8 channels in B 1001b - 9 channels in B 1010b - 10 channels in B 1011b-1111b - Should not be used.
3-0 ANCA	Number of A Channels 0000b - 0 channels in A (Pair A is disabled) 0001b - 1 channel in A 0010b - 2 channels in A 0011b - 3 channels in A 0100b - 4 channels in A 0101b - 5 channels in A 0110b - 6 channels in A 0111b - 7 channels in A 1000b - 8 channels in A 1001b - 9 channels in A 1010b - 10 channels in A

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1011b-1111b - Should not be used.

49.6.1.5 ASRC Filter Configuration Status (ASRCFG)

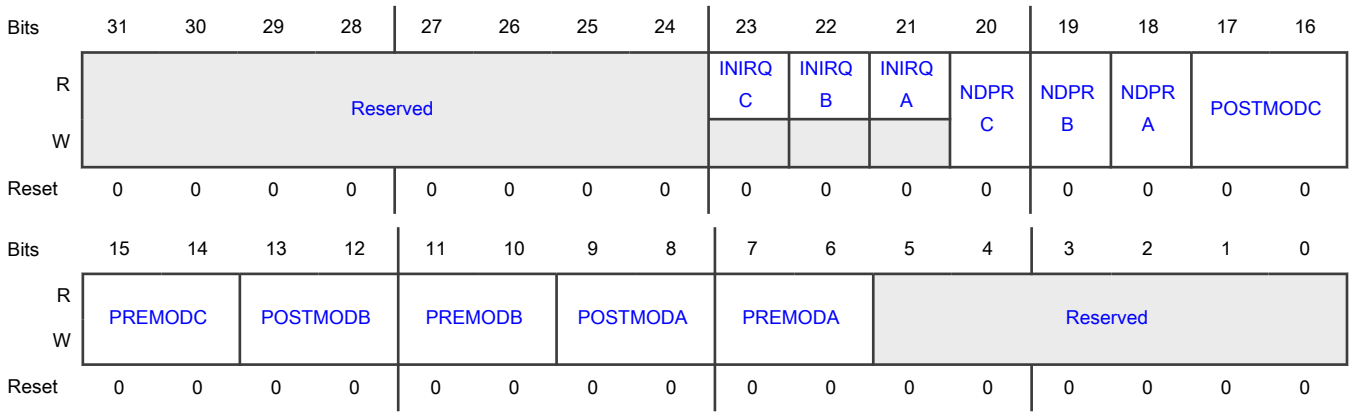
Offset

Register	Offset
ASRCFG	10h

Function

Sets and/or automatically senses the ASRC operations.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 INIRQC	Initialization for Conversion Pair C is Served When this bit is 1, it means the initialization for conversion pair C is served. This bit is cleared by disabling the ASRC conversion pair (ASRCTR[ASREC]=0 or ASRCTR[ASRCEN]=0). 0b - Initialization for Conversion Pair C not served 1b - Initialization for Conversion Pair C served
22 INIRQB	Initialization for Conversion Pair B is Served

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this bit is 1, it means the initialization for conversion pair B is served. This bit is cleared by disabling the ASRC conversion pair (ASRCTR[ASREB]=0 or ASRCTR[ASRCEN]=0). 0b - Initialization for Conversion Pair B not served 1b - Initialization for Conversion Pair B served
21 INIRQA	Initialization for Conversion Pair A is served When this bit is 1, it means the initialization for conversion pair A is served. This bit is cleared by disabling the ASRC conversion pair (ASRCTR[ASREA]=0 or ASRCTR[ASRCEN]=0). 0b - Initialization for Conversion Pair A not served 1b - Initialization for Conversion Pair A served
20 NDPRC	Not Use Default Parameters for RAM-Stored Parameters For Conversion Pair C 0b - Use default parameters for RAM-stored parameters. Override any parameters already in RAM. 1b - Don't use default parameters for RAM-stored parameters. Use the parameters already stored in RAM.
19 NDPRB	Not Use Default Parameters for RAM-Stored Parameters For Conversion Pair B 0b - Use default parameters for RAM-stored parameters. Override any parameters already in RAM. 1b - Don't use default parameters for RAM-stored parameter. Use the parameters already stored in RAM.
18 NDPRA	Not Use Default Parameters for RAM-stored Parameters For Conversion Pair A 0b - Use default parameters for RAM-stored parameters. Override any parameters already in RAM. 1b - Don't use default parameters for RAM-stored parameters. Use the parameters already stored in RAM.
17-16 POSTMODC	Post-Processing Configuration for Conversion Pair C These bits are read/written by the user if ASRCTR[ATSC]=0, and can also be automatically updated by the ASRC internal logic if ASRCTR[ATSC]=1 (see ASRC Control Register for Pair C). These bits set the selection of the post-processing configuration. 00b - Select Upsampling-by-2 as defined in Signal Processing Flow. 01b - Select Direct-Connection as defined in Signal Processing Flow. 10b - Select Downsampling-by-2 as defined in Signal Processing Flow. 11b - Reserved.
15-14 PREMODC	Pre-Processing Configuration for Conversion Pair C These bits are read/written by the user if ASRCTR[ATSC]=0, and can also be automatically updated by the ASRC internal logic if ASRCTR[ATSC]=1 (see ASRC Control Register for Pair C). These bits set the selection of the pre-processing configuration.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Select Upsampling-by-2. This is defined in Signal processing flow 01b - Select Direct-Connection. This is defined in Signal processing flow 10b - Select Downsampling-by-2. as defined in Signal processing flow 11b - Select passthrough mode. In this case, POSTMODC[1:0] have no use.
13-12 POSTMODB	Post-Processing Configuration for Conversion Pair B These bits are read/written by the user if ASRCTR[ATSB]=0, and can also be automatically updated by the ASRC internal logic if ASRCTR[ATSB]=1 (see ASRC Control Register for Pair B). These bits set the selection of the post-processing configuration. 00b - Select Upsampling-by-2. This is defined in Signal processing flow 01b - Select Direct-Connection. This is defined in Signal processing flow 10b - Select Downsampling-by-2. This is defined in Signal processing flow 11b - Reserved.
11-10 PREMODB	Pre-Processing Configuration for Conversion Pair B These bits are read/written by the user if ASRCTR[ATSB]=0, and can also be automatically updated by the ASRC internal logic if ASRCTR[ATSB]=1 (see ASRC Control Register for Pair B). These bits set the selection of the pre-processing configuration. 00b - Select Upsampling-by-2. This is defined in Signal processing flow 01b - Select Direct-Connection. This is defined in Signal processing flow 10b - Select Downsampling-by-2. This is defined in Signal processing flow 11b - Select passthrough mode. In this case, POSTMODB[1:0] have no use.
9-8 POSTMODA	Post-Processing Configuration for Conversion Pair A These bits are read/written by the user if ASRCTR[ATSA]=0, and can also be automatically updated by the ASRC internal logic if ASRCTR[ATSA]=1 (see ASRC Control Register for Pair A). These bits set the selection of the post-processing configuration. 00b - Select Upsampling-by-2. This is defined in Signal processing flow 01b - Select Direct-Connection. This is defined in Signal processing flow 10b - Select Downsampling-by-2. This is defined in Signal processing flow 11b - Reserved.
7-6 PREMODA	Pre-Processing Configuration for Conversion Pair A These bits are read/written by the user if ASRCTR[ATSA]=0, and can also be automatically updated by the ASRC internal logic if ASRCTR[ATSA]=1 (see ASRC Control Register for Pair A). These bits set the selection of the pre-processing configuration. 00b - Select Upsampling-by-2. This is defined in Signal processing flow 01b - Select Direct-Connection. This is defined in Signal processing flow

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Select Downsampling-by-2. This is defined in Signal processing flow 11b - Select passthrough mode. In this case, POSTMODA[1:0] have no use.
5-0 —	Should be written as zero for compatibility.

49.6.1.6 ASRC Clock Source (ASRCSR)

Offset

Register	Offset
ASRCSR	14h

Function

Controls the sources of the input and output clocks of the ASRC.

See External signals.

NOTE

Your specific device may not support all of the clock sources. See the chip-specific section for this module, at the beginning of this chapter.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								AOCSC				AOCSB			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	AOCSA				AICSC				AICSB				AICSA			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-20 AOCSC	Output Clock Source C 0000b - Bit clock 0 0001b - Bit clock 1 0010b - Bit clock 2 0011b - Bit clock 3 0100b - Bit clock 4 0101b - Bit clock 5 0110b - Bit clock 6 0111b - Bit clock 7 1000b - Bit clock 8 1001b - Bit clock 9 1010b - Bit clock A 1011b - Bit clock B 1100b - Bit clock C 1101b - Bit clock D 1110b - Bit clock E 1111b - Clock disabled, connected to zero
19-16 AOCSB	Output Clock Source B 0000b - Bit clock 0 0001b - Bit clock 1 0010b - Bit clock 2 0011b - Bit clock 3 0100b - Bit clock 4 0101b - Bit clock 5 0110b - Bit clock 6 0111b - Bit clock 7 1000b - Bit clock 8 1001b - Bit clock 9 1010b - Bit clock A 1011b - Bit clock B 1100b - Bit clock C 1101b - Bit clock D 1110b - Bit clock E

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1111b - Clock disabled, connected to zero
15-12 AOCSA	Output Clock Source A 0000b - Bit clock 0 0001b - Bit clock 1 0010b - Bit clock 2 0011b - Bit clock 3 0100b - Bit clock 4 0101b - Bit clock 5 0110b - Bit clock 6 0111b - Bit clock 7 1000b - Bit clock 8 1001b - Bit clock 9 1010b - Bit clock A 1011b - Bit clock B 1100b - Bit clock C 1101b - Bit clock D 1110b - Bit clock E 1111b - Clock disabled, connected to zero
11-8 AICSC	Input Clock Source C 0000b - Bit clock 0 0001b - Bit clock 1 0010b - Bit clock 2 0011b - Bit clock 3 0100b - Bit clock 4 0101b - Bit clock 5 0110b - Bit clock 6 0111b - Bit clock 7 1000b - Bit clock 8 1001b - Bit clock 9 1010b - Bit clock A 1011b - Bit clock B 1100b - Bit clock C 1101b - Bit clock D

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1110b - Bit clock E 1111b - Clock disabled, connected to zero
7-4 AICSB	Input Clock Source B 0000b - Bit clock 0 0001b - Bit clock 1 0010b - Bit clock 2 0011b - Bit clock 3 0100b - Bit clock 4 0101b - Bit clock 5 0110b - Bit clock 6 0111b - Bit clock 7 1000b - Bit clock 8 1001b - Bit clock 9 1010b - Bit clock A 1011b - Bit clock B 1100b - Bit clock C 1101b - Bit clock D 1110b - Bit clock E 1111b - Clock disabled, connected to zero
3-0 AICSA	Input Clock Source A 0000b - Bit clock 0 0001b - Bit clock 1 0010b - Bit clock 2 0011b - Bit clock 3 0100b - Bit clock 4 0101b - Bit clock 5 0110b - Bit clock 6 0111b - Bit clock 7 1000b - Bit clock 8 1001b - Bit clock 9 1010b - Bit clock A 1011b - Bit clock B 1100b - Bit clock C

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1101b - Bit clock D 1110b - Bit clock E 1111b - Clock disabled, connected to zero

49.6.1.7 ASRC Clock Divider 1 (ASRCDR1)

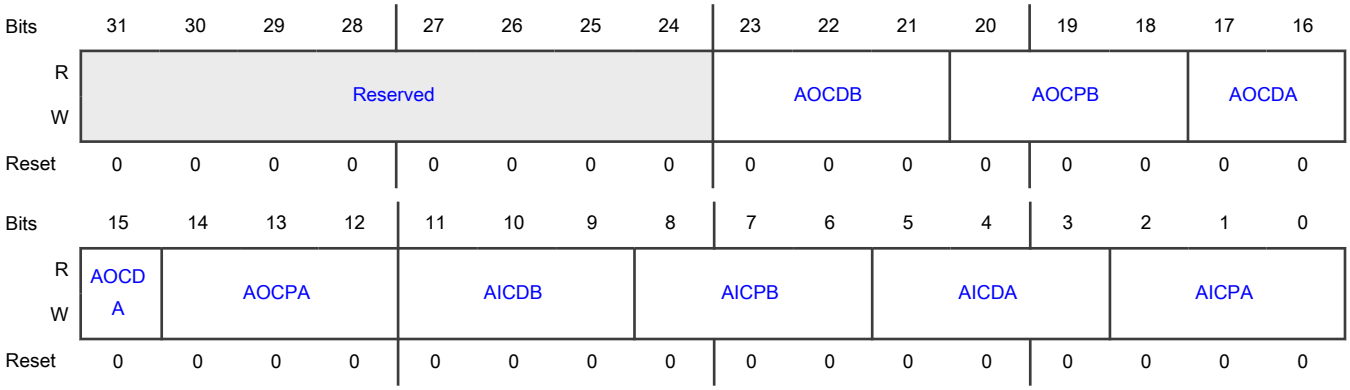
Offset

Register	Offset
ASRCDR1	18h

Function

Controls the division factors of the ASRC input and output clock sources.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-21 AOCDB	Output Clock Divider B Specifies the divide ratio of the output clock divider B. The divide ratio may range from 1 to 8 (AOCDB[2:0] = 000 to 111).
20-18 AOCPB	Output Clock Prescaler B Specifies the prescaling factor of the output prescaler B. The prescaling ratio may be any power of 2 from 1 to 128.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17-15 AOCDA	Output Clock Divider A Specifies the divide ratio of the output clock divider A. The divide ratio may range from 1 to 8 (AOCDA[2:0] = 000 to 111).
14-12 AOCPA	Output Clock Prescaler A Specifies the prescaling factor of the output prescaler A. The prescaling ratio may be any power of 2 from 1 to 128.
11-9 AICDB	Input Clock Divider B Specifies the divide ratio of the input clock divider B. The divide ratio may range from 1 to 8 (AICDB[2:0] = 000 to 111).
8-6 AICPB	Input Clock Prescaler B Specifies the prescaling factor of the input prescaler B. The prescaling ratio may be any power of 2 from 1 to 128.
5-3 AICDA	Input Clock Divider A Specifies the divide ratio of the input clock divider A. The divide ratio may range from 1 to 8 (AICDA[2:0] = 000 to 111).
2-0 AICPA	Input Clock Prescaler A Specifies the prescaling factor of the input prescaler A. The prescaling ratio may be any power of 2 from 1 to 128.

49.6.1.8 ASRC Clock Divider 2 (ASRCDR2)

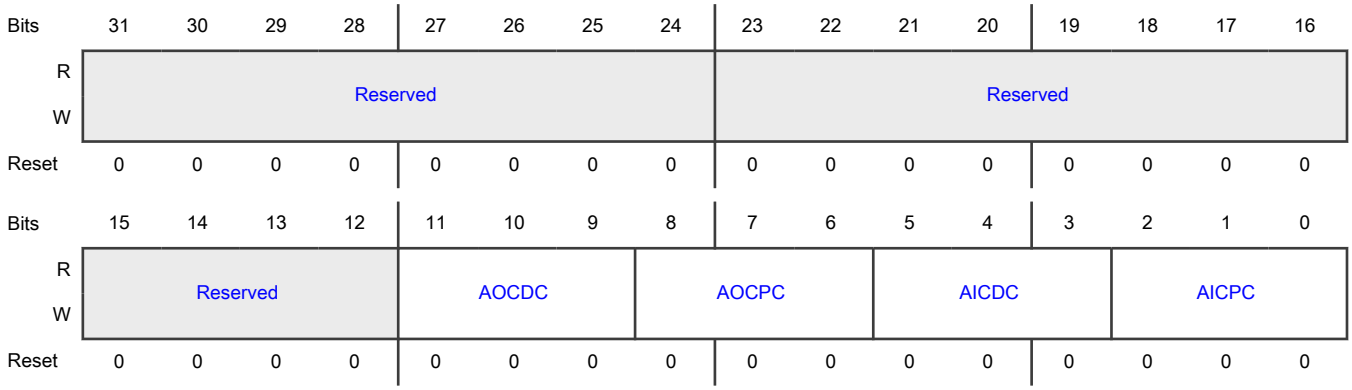
Offset

Register	Offset
ASRCDR2	1Ch

Function

Controls the division factors of the ASRC input and output clock sources.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-12 —	Should be written as zero for compatibility.
11-9 AOCDC	Output Clock Divider C Specifies the divide ratio of the output clock divider C. The divide ratio may range from 1 to 8 (AOCDC[2:0] = 000 to 111).
8-6 AOCPC	Output Clock Prescaler C Specifies the prescaling factor of the output prescaler C. The prescaling ratio may be any power of 2 from 1 to 128.
5-3 AICDC	Input Clock Divider C Specifies the divide ratio of the input clock divider C. The divide ratio may range from 1 to 8 (AICDC[2:0] = 000 to 111).
2-0 AICPC	Input Clock Prescaler C Specifies the prescaling factor of the input prescaler C. The prescaling ratio may be any power of 2 from 1 to 128.

49.6.1.9 ASRC Status (ASRSTR)

Offset

Register	Offset
ASRSTR	20h

Function

Used by the processor core to examine the status of the ASRC block and clear the overload interrupt request and AOLE flag bit. Read of the status register returns the current state of ASRC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved		DSLCNT	ATQOL	AOOLC	AOOLB	AOOLA	AIOLC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AIOLB	AIOLA	AODOC	AODOB	AODOA	AIDUC	AIDUB	AIDUA	FPWT	AOLE	AODFC	AODFB	AODFA	AIDEC	AIDEB	AIDEA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-22 —	Should be written as zero for compatibility.
21 DSLCNT	Digital Servo Loop (DSL) Counter Input to FIFO Ready When set, this bit indicates that new DSL counter information is stored in the internal ASRC FIFO. When clear, this bit indicates that new DSL counter information is in the process of storage into the internal ASRC FIFO. When ASRIER[AFPWE]=1, the rising edge of this signal proposes an interrupt request. Writing any value with this bit set clears the interrupt request proposed by the rising edge of this bit. 0b - New DSL counter information is in the process of storage into the internal ASRC FIFO 1b - New DSL counter information is stored in the internal ASRC FIFO
20 ATQOL	Task Queue FIFO overload When set, this bit indicates that task queue FIFO logic is overloaded. This may help to check the reason why overload interrupt happens. The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - Task queue FIFO logic is not overloaded 1b - Task queue FIFO logic is overloaded
19	Pair C Output Task Overload

Table continues on the next page...

Table continued from the previous page...

Field	Function
AOOLC	When set, this bit indicates that pair C output task is overloaded. This may help to check the reason why overload interrupt happens.> The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - Pair C output task is not overloaded 1b - Pair C output task is overloaded
18 AOOLB	Pair B Output Task Overload When set, this bit indicates that pair B output task is overloaded. This may help to check the reason why overload interrupt happens. The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - Pair B output task is not overloaded 1b - Pair B output task is overloaded
17 AOOLA	Pair A Output Task Overload When set, this bit indicates that pair A output task is overloaded. This may help to check the reason why overload interrupt happens. The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - Pair A output task is not overloaded 1b - Pair A output task is overloaded
16 AIOLC	Pair C Input Task Overload When set, this bit indicates that pair C input task is overloaded. This may help to check the reason why overload interrupt happens. The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - Pair C input task is not overloaded 1b - Pair C input task is overloaded
15 AIOLB	Pair B Input Task Overload When set, this bit indicates that pair B input task is overloaded. This may help to check the reason why overload interrupt happens. The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - Pair B input task is not overloaded 1b - Pair B input task is overloaded
14 AIOLA	Pair A Input Task Overload When set, this bit indicates that pair A input task is overloaded. This may help to check the reason why overload interrupt happens. The bit is cleared when writing ASRSTR[AOLE] as 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Pair A input task is not overloaded 1b - Pair A input task is overloaded
13 AODOC	Output Data Buffer C has Overflowed When set, this bit indicates that output data buffer C has overflowed. When clear, this bit indicates that output data buffer C has not overflowed The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - No Overflow in Output data buffer C 1b - Overflow in Output data buffer C
12 AODOB	Output Data Buffer B has Overflowed When set, this bit indicates that output data buffer B has overflowed. When clear, this bit indicates that output data buffer B has not overflowed The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - No Overflow in Output data buffer B 1b - Overflow in Output data buffer B
11 AODOA	Output Data Buffer A has Overflowed When set, this bit indicates that output data buffer A has overflowed. When clear, this bit indicates that output data buffer A has not overflowed. The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - No Overflow in Output data buffer A 1b - Overflow in Output data buffer A
10 AIDUC	Input Data Buffer C has Underflowed When set, this bit indicates that input data buffer C has underflowed. When clear, this bit indicates that input data buffer C has not underflowed. The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - No Underflow in Input data buffer C 1b - Underflow in Input data buffer C
9 AIDUB	Input Data Buffer B has Underflowed When set, this bit indicates that input data buffer B has underflowed. When clear, this bit indicates that input data buffer B has not underflowed. The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - No Underflow in Input data buffer B 1b - Underflow in Input data buffer B
8	Input Data Buffer A has Underflowed

Table continues on the next page...

Table continued from the previous page...

Field	Function
AIDUA	When set, this bit indicates that input data buffer A has underflowed. When clear, this bit indicates that input data buffer A has not underflowed. The bit is cleared when writing ASRSTR[AOLE] as 1. 0b - No Underflow in Input data buffer A 1b - Underflow in Input data buffer A
7 FPWT	FP is in Wait States This bit is for debug only. When set, this bit indicates that ASRC is in wait states. When clear, this bit indicates that ASRC is not in wait states. 0b - ASRC is not in wait state 1b - ASRC is in wait state
6 AOLE	Overload Error Flag When set, this bit indicates that the task rate is too high for the ASRC to handle. The reasons for overload may be: • too high input clock frequency, • too high output clock frequency, • incorrect selection of the pre-filter, • low ASRC processing clock, • too many channels, • underrun, • or any combination of the reasons above. Since the ASRC uses the same hardware resources to perform various tasks, the real reason for the overload is not straight forward, and it should be carefully analyzed by the programmer. If ASRIER[AOLIE]=1, an interrupt is proposed when this bit is set. Write any value with this bit set as one into the status register clears this bit and the interrupt request proposed by this bit. 0b - No overload 1b - Task rate is too high
5 AODFC	Number of data in Output Data Buffer C is Greater than Threshold When set, this bit indicates that number of data already existing in ASRDOC is greater than threshold and the processor can read data from ASRDOC. A DMA request is always generated when the AODFC bit is set, but a DMA transfer takes place only if a DMA channel is active and triggered by this event. 0b - The threshold has not yet been met and no data output C interrupt is generated 1b - When AODFC is set, the ASRC generates data output C interrupt request to the processor if ASRIER[ADOEC] = 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 AODFB	Number of data in Output Data Buffer B is Greater than Threshold When set, this bit indicates that number of data already existing in ASRDOB is greater than threshold and the processor can read data from ASRDOB. A DMA request is always generated when the AODFB bit is set, but a DMA transfer takes place only if a DMA channel is active and triggered by this event. 0b - The threshold has not yet been met and no data output B interrupt is generated 1b - When AODFB is set, the ASRC generates data output B interrupt request to the processor if ASRIER[ADOEB] = 1
3 AODFA	Number of Data in Output Data Buffer A is Greater than Threshold When set, this bit indicates that number of data already existing in ASRDOA is greater than threshold and the processor can read data from ASRDOA. A DMA request is always generated when the AODFA bit is set, but a DMA transfer takes place only if a DMA channel is active and triggered by this event. 0b - The threshold has not yet been met and no data output A interrupt is generated 1b - When AODFA is set, the ASRC generates data output A interrupt request to the processor if ASRIER[ADOEA] = 1
2 AIDEC	Number of Data in Input Data Buffer C is Less than Threshold When set, this bit indicates that the number of data still available in ASRDIC is less than the threshold and the processor can write data to ASRDIC. A DMA request is always generated when the AIDEC bit is set, but a DMA transfer takes place only if a DMA channel is active and triggered by this event. 0b - The threshold has been met and no data input C interrupt is generated 1b - When AIDEC is set, the ASRC generates data input C interrupt request to the processor if ASRIER[AIDEC] = 1
1 AIDEB	Number of Data in Input Data Buffer B is Less than Threshold When set, this bit indicates that number of data still available in ASRDIB is less than threshold and the processor can write data to ASRDIB. A DMA request is always generated when the AIDEB bit is set, but a DMA transfer takes place only if a DMA channel is active and triggered by this event. 0b - The threshold has been met and no data input B interrupt is generated 1b - When AIDEB is set, the ASRC generates data input B interrupt request to the processor if ASRIER[AIDEB] = 1
0 AIDEA	Number of Data in Input Data Buffer A is Less than Threshold When set, this bit indicates that number of data still available in ASRDIA is less than the threshold and the processor can write data to ASRDIA. A DMA request is always generated when the AIDEA bit is set, but a DMA transfer takes place only if a DMA channel is active and triggered by this event. 0b - The threshold has been met and no data input A interrupt is generated 1b - When AIDEA is set, the ASRC generates data input A interrupt request to the processor if ASRIER[AIDEA] = 1

49.6.1.10 ASRC Parameter x (ASRPM1 - ASRPM5)

Offset

Register	Offset
ASRPM1	40h
ASRPM2	44h
ASRPM3	48h
ASRPM4	4Ch
ASRPM5	50h

Function

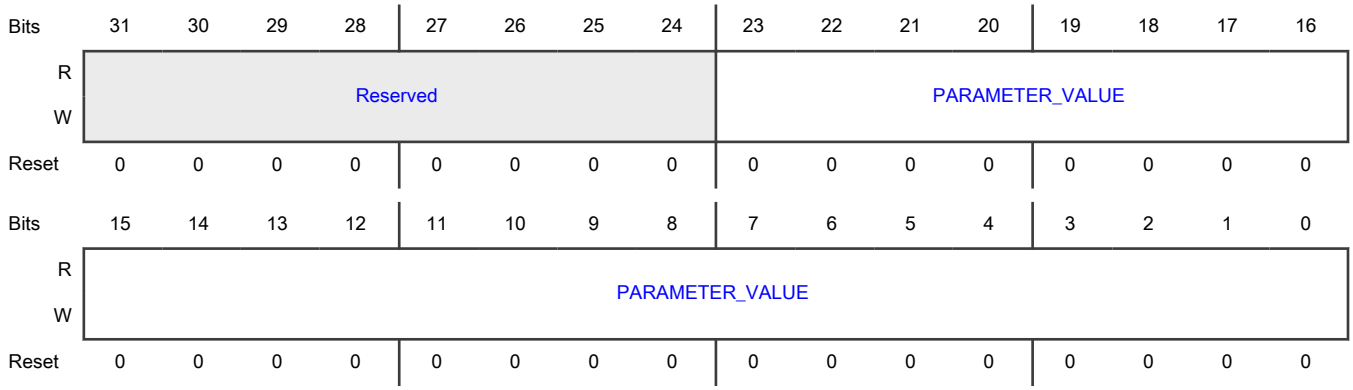
Parameter registers determine the performance of ASRC.

The parameter registers must be initialized by software before ASRC is enabled. Recommended values are given in the table below:

Table 330. ASRC Parameter Registers (ASRPM1~ASRPM5)

Register	Offset	Access	Reset Value	Recommend Value
ASRPM1	0x40	R/W	0x00_0000	0x7fffff
ASRPM2	0x44	R/W	0x00_0000	0x255555
ASRPM3	0x48	R/W	0x00_0000	0xff7280
ASRPM4	0x4C	R/W	0x00_0000	0xff7280
ASRPM5	0x50	R/W	0x00_0000	0xff7280

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 PARAMETER_VALUE	Parameter Value See recommended values table.

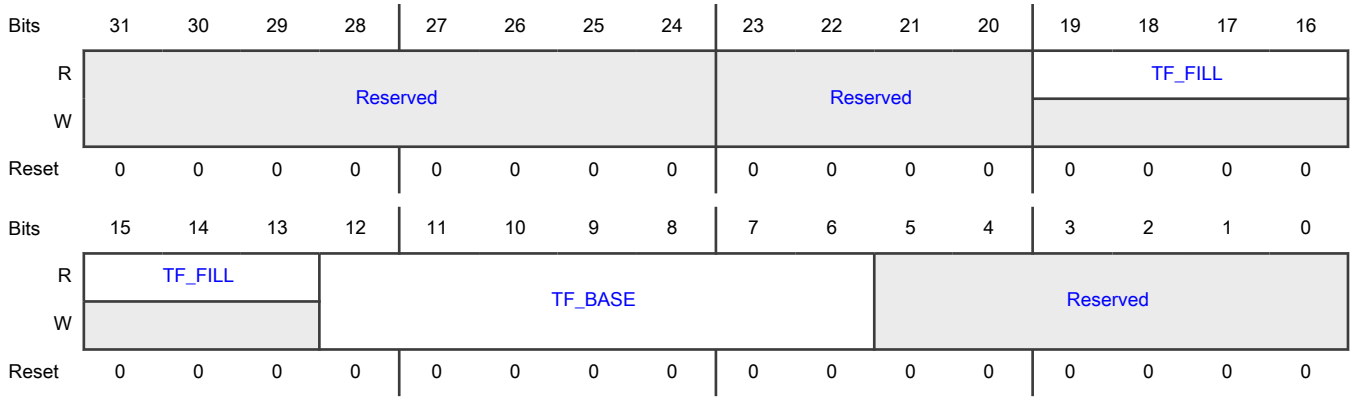
49.6.1.11 ASRC Task Queue FIFO 1 (ASRTFR1)

Offset

Register	Offset
ASRTFR1	54h

Function
Defines and shows the parameters for ASRC inner task queue FIFOs.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 —	Should be written as zero for compatibility.
19-13	Current Number of Entries in Task Queue FIFO

Table continues on the next page...

Table continued from the previous page...

Field	Function
TF_FILL	
12-6 TF_BASE	Base Address for Task Queue FIFO Set to 0x7C.
5-0 —	Should be written as zero for compatibility.

49.6.1.12 ASRC Channel Counter (ASRCCR)

Offset

Register	Offset
ASRCCR	5Ch

Function

Sets and reflects the current specific input/output FIFO being accessed through shared peripheral bus for each ASRC conversion pair.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								ACOC				ACOB			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ACOA				ACIC				ACIB				ACIA			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-20 ACOC	Channel Counter for Pair C's Output FIFO These bits stand for the current channel being accessed through shared peripheral bus for Pair C's output FIFO's usage. The value can be any value between [0, ANCC-1]

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-16 ACOB	Channel Counter for Pair B's Output FIFO These bits stand for the current channel being accessed through shared peripheral bus for Pair B's output FIFO's usage. The value can be any value between [0, ANCB-1]
15-12 ACOA	Channel Counter for Pair A's Output FIFO These bits stand for the current channel being accessed through shared peripheral bus for Pair A's output FIFO's usage. The value can be any value between [0, ANCA-1]
11-8 ACIC	Channel Counter for Pair C's Input FIFO These bits stand for the current channel being accessed through shared peripheral bus for Pair C's input FIFO's usage. The value can be any value between [0, ANCC-1]
7-4 ACIB	Channel Counter for Pair B's Input FIFO These bits stand for the current channel being accessed through shared peripheral bus for Pair B's input FIFO's usage. The value can be any value between [0, ANCB-1]
3-0 ACIA	Channel Counter for Pair A's Input FIFO These bits stand for the current channel being accessed through shared peripheral bus for Pair A's input FIFO's usage. The value can be any value between [0, ANCA-1]

49.6.1.13 ASRC Data Input for Pair x (ASRDIA - ASRDIC)

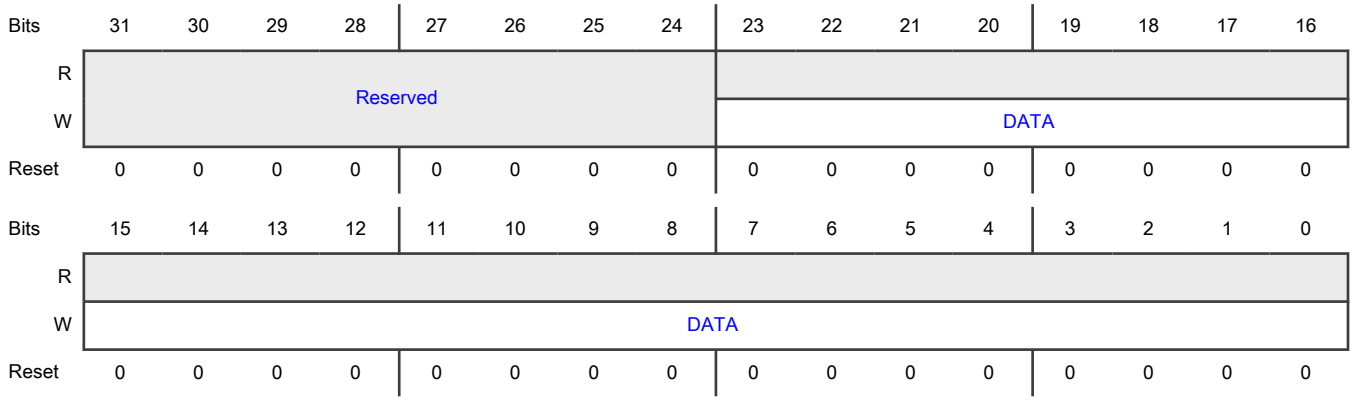
Offset

Register	Offset
ASRDIA	60h
ASRDIB	68h
ASRDIC	70h

Function

Interface registers for the audio data input of pair A,B,C respectively. They are backed by FIFOs.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 DATA	Data Audio data input.

49.6.1.14 ASRC Data Output for Pair x (ASRDOA - ASRDOC)

Offset

Register	Offset
ASRDOA	64h
ASRDOB	6Ch
ASRDOC	74h

Function

Interface registers for the audio data output of pair A,B,C respectively. They are backed by FIFOs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								DATA							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 DATA	Data Audio data output.

49.6.1.15 ASRC Ideal Ratio for Pair A-High Part (ASRIDRHA)

Offset

Register	Offset
ASRIDRHA	80h

Function

The ideal ratio registers (ASRIDRHA, ASRIDRLA) hold the ratio value IDRATIOA. IDRATIOA = $Fs_{inA}/Fs_{outA} = Ts_{outA}/Ts_{inA}$ is a 32-bit fixed point value with 26 fractional bits. This value is only useful when ASRCTR[USRA, IDRA]=2'b11.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								IDRATIOA_H							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-8 —	Reserved
7-0 IDRATIOA_H	Ideal Ratio A High High part of ideal ratio value for pair A that constitutes bits [31:24] of the ideal ratio IDRATIOA. See also ASRIDRLA[IDRATIOA_L] field.

49.6.1.16 ASRC Ideal Ratio for Pair A -Low Part (ASRIDRLA)

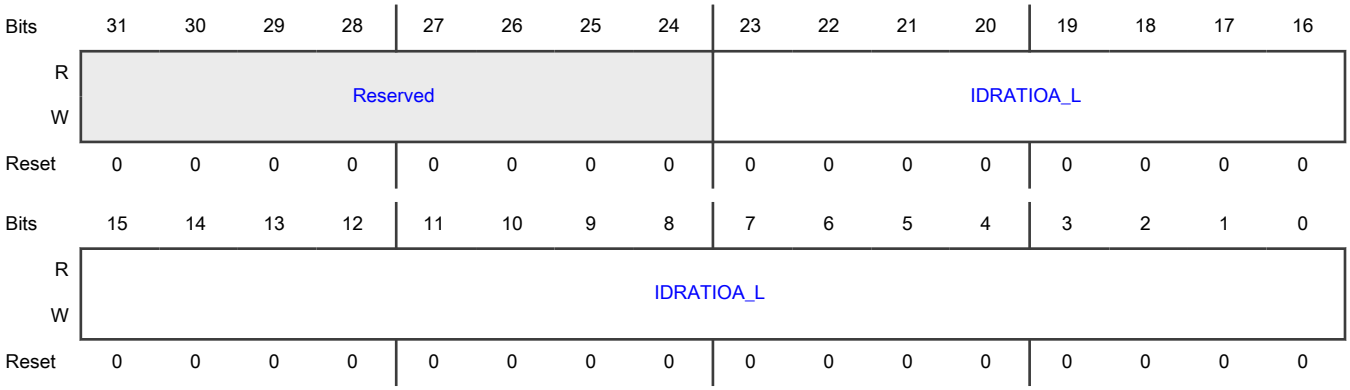
Offset

Register	Offset
ASRIDRLA	84h

Function

The ideal ratio registers (ASRIDRHA, ASRIDRLA) hold the ratio value IDRATIOA. IDRATIOA = Fs_{inA}/Fs_{outA} = Ts_{outA}/Ts_{inA} is a 32-bit fixed point value with 26 fractional bits. This value is only useful when ASRCTR[USRA, IDRA]=2'b11.

Diagram



Fields

Field	Function
31-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-0 IDRATIOA_L	Ideal Ratio A Low Low part of ideal ratio value for pair A that constitutes bits [23:0] of the ideal ratio IDRATIOA. See also ASRIDRHA[IDRATIOA_H] field.

49.6.1.17 ASRC Ideal Ratio for Pair B-High Part (ASRIDRHB)

Offset

Register	Offset
ASRIDRHB	88h

Function

The ideal ratio registers (ASRIDRHB, ASRIDRLB) hold the ratio value IDRATIOB. IDRATIOB = Fs_{inB}/Fs_{outB} = Ts_{outB}/Ts_{inB} is a 32-bit fixed point value with 26 fractional bits. This value is only useful when ASRCTR[USRB, IDRB]=2'b11.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IDRATIOB_H							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-8 —	Reserved
7-0 IDRATIOB_H	Ideal Ratio B High High part of ideal ratio value for pair B that constitutes bits [31:24] of the ideal ratio IDRATIOB. See also ASRIDRLB[IDRATIOB_L] field.

49.6.1.18 ASRC Ideal Ratio for Pair B-Low Part (ASRIDRLB)

Offset

Register	Offset
ASRIDRLB	8Ch

Function

The ideal ratio registers (ASRIDRHB, ASRIDRLB) hold the ratio value IDRATIOB. IDRATIOB = $Fs_{inB}/Fs_{outB} = Ts_{outB}/Ts_{inB}$ is a 32-bit fixed point value with 26 fractional bits. This value is only useful when ASRCTR[USRB, IDRB]=2'b11.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								IDRATIOB_L							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDRATIOB_L															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 IDRATIOB_L	Ideal Ratio B Low Low part of ideal ratio value for pair B that constitutes bits [23:0] of the ideal ratio IDRATIOB. See also ASRIDRHB[IDRATIOB_H] field.

49.6.1.19 ASRC Ideal Ratio for Pair C-High Part (ASRIDRHC)

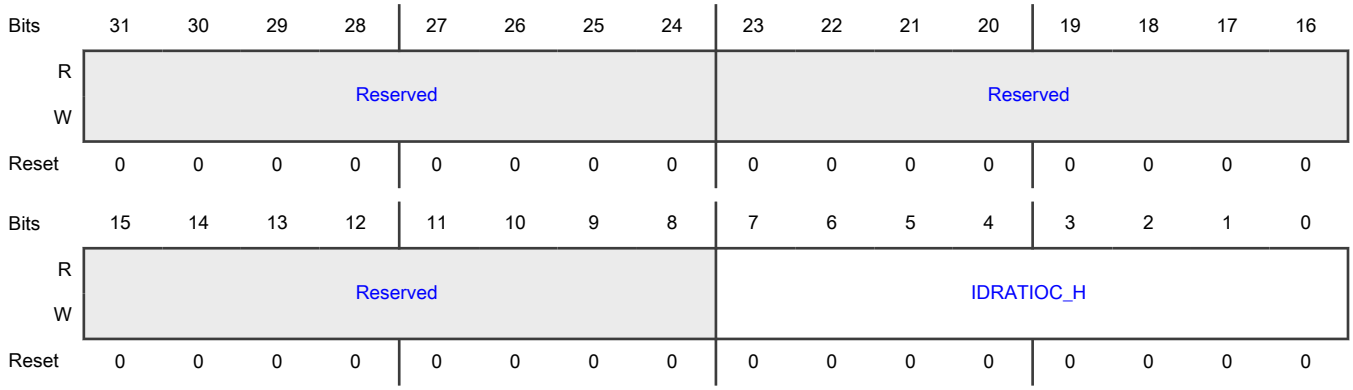
Offset

Register	Offset
ASRIDRHC	90h

Function

The ideal ratio registers (ASRIDRHC, ASRIDRLC) hold the ratio value IDRATIOC. IDRATIOC = $Fs_{inC}/Fs_{outC} = Ts_{outC}/Ts_{inC}$ is a 32-bit fixed point value with 26 fractional bits. This value is only useful when ASRCTR[USRC, IDRC]=2'b11.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-8 —	Reserved
7-0 IDRATIOC_H	Ideal Ratio C High High part of ideal ratio value for pair C that constitutes bits [31:24] of the ideal ratio IDRATIOC. See also ASRIDRLC[IDRATIOC_L] field.

49.6.1.20 ASRC Ideal Ratio for Pair C-Low Part (ASRIDRLC)

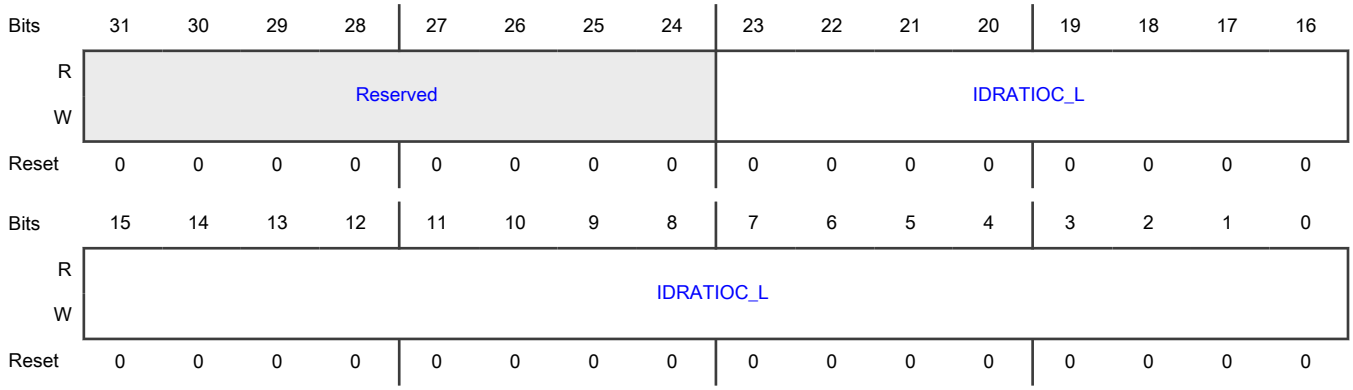
Offset

Register	Offset
ASRIDRLC	94h

Function

The ideal ratio registers (ASRIDRHC, ASRIDRLC) hold the ratio value IDRATIOC. IDRATIOC = Fs_{inC}/Fs_{outC} = Ts_{outC}/Ts_{inC} is a 32-bit fixed point value with 26 fractional bits. This value is only useful when ASRCTR[USRC, IDRC]=2'b11.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 IDRATIOC_L	Ideal Ratio C Low Low part of ideal ratio value for pair C that constitutes bits [23:0] of the ideal ratio IDRATIOC. See also ASRIDRHC[IDRATIOC_H] field.

49.6.1.21 ASRC 76 kHz Period (ASR76K)

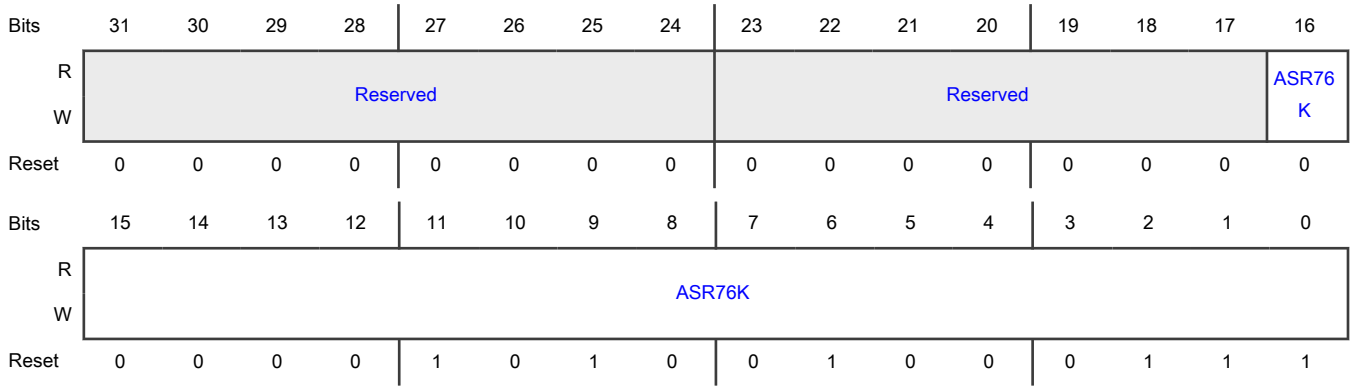
Offset

Register	Offset
ASR76K	98h

Function

Holds the period of the 76 kHz sampling clock in terms of the ASRC processing clock with frequency Fs_{ASRC}. ASR76K = Fs_{ASRC}/Fs_{76k}. Reset value is 0x0A47 which assumes that Fs_{ASRC}=200 MHz. This register is used to help the ASRC internal logic to decide the pre-processing and the post-processing options automatically (see ASRC Control Register and ASRC Filter Configuration Status Register). In a system when Fs_{ASRC} = 133 MHz, the value should be assigned explicitly as 0x06D6 in user application code.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-17 —	Reserved
16-0 ASR76K	Value for the Period of the 76 kHz Sampling Clock

49.6.1.22 ASRC 56 kHz Period (ASR56K)

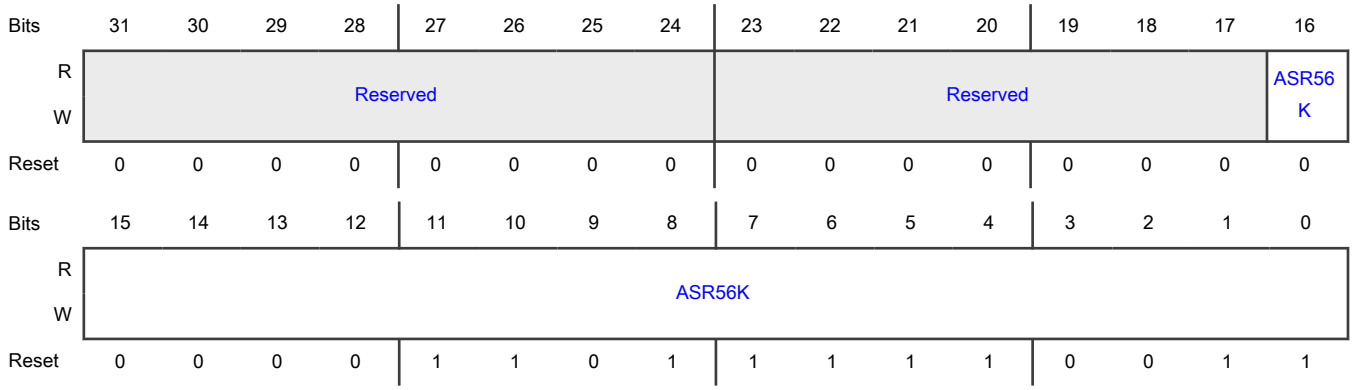
Offset

Register	Offset
ASR56K	9Ch

Function

Holds the period of the 56 kHz sampling clock in terms of the ASRC processing clock with frequency Fs_{ASRC}. ASR56K = Fs_{ASRC}/Fs_{56k}. Reset value is 0x0DF3 which assumes that Fs_{ASRC} = 200 MHz. This register is used to help the ASRC internal logic to decide the pre-processing and the post-processing options automatically (see ASRC Control Register and ASRC Misc Control Register 1 for Pair C). In a system when Fs_{ASRC} = 133 MHz, the value should be assigned explicitly as 0x0947 in user application code.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-17 —	Reserved
16-0 ASR56K	Value for the Period of the 56 kHz Sampling Clock

49.6.1.23 ASRC Misc Control for Pair A (ASRMCRA)

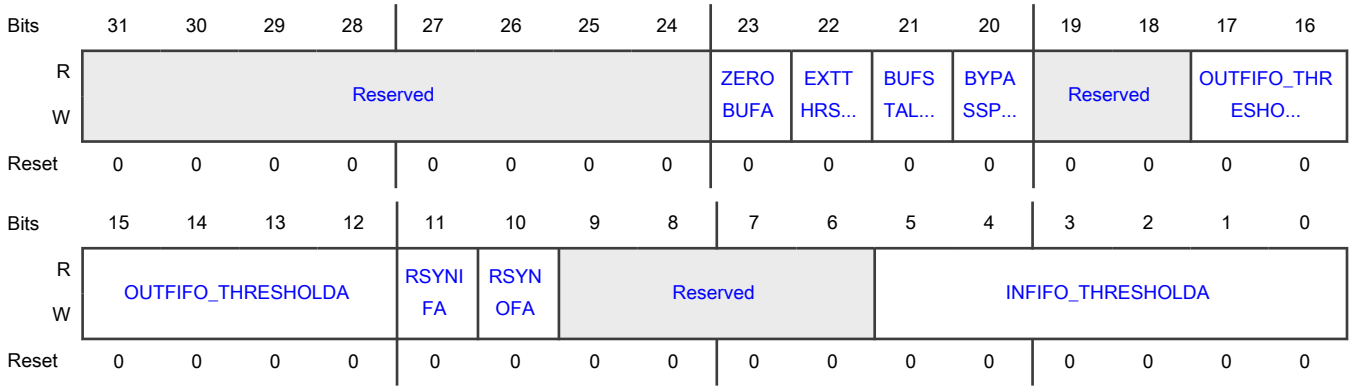
Offset

Register	Offset
ASRMCRA	A0h

Function

Used to control Pair A internal logic.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 ZEROBUFA	Zero Buffer A Initialize buffer of Pair A when pair A is enabled. Always clear option. This bit is used to control whether the buffer is to be zeroized when pair A is enabled. 0b - Zeroize the buffer 1b - Don't zeroize the buffer
22 EXTTHRSHA	Use External Thresholds for FIFO Control of Pair A Determines whether the FIFO thresholds externally defined in this register is used to control ASRC internal FIFO logic for pair A. 0b - Use default thresholds. 1b - Use external defined thresholds.
21 BUFSTALLA	Stall Pair A Conversion in Case of Buffer Near Empty/Full Condition Determines whether the near empty/full FIFO condition stalls the rate conversion for pair A. This option can only work when external ratio is used. Near empty condition is the condition when input FIFO has less than 4 useful samples per channel. Near full condition is the condition when the output FIFO has less than 4 vacant sample words to fill per channel. 0b - Don't stall Pair A conversion even in case of near empty/full FIFO conditions. 1b - Stall Pair A conversion in case of near empty/full FIFO conditions.
20 BYPASSPOLYA	Bypass Polyphase Filtering for Pair A Determines whether the polyphase filtering part of Pair A conversion is bypassed. 0b - Don't bypass polyphase filtering.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Bypass polyphase filtering.
19-18 —	Should be written as zero for future compatibility.
17-12 OUTFIFO_THRESHOLDA	Threshold for Pair A's Output FIFO per Channel These bits stand for the threshold for Pair A's output FIFO per channel. Possible range is [0,63]. When the value is n, it means that: when the number of output FIFO fillings of the pair is greater than n samples per channel, the output data ready flag is set; when the number of output FIFO fillings of the pair is less than or equal to n samples per channel, the output data ready flag is automatically cleared.
11 RSYNIFA	Re-sync Input FIFO Channel Counter If bit set, force ASRCCR[ACIA]=0. If bit clear, do not touch ASRCCR[ACIA]. 0b - Do not touch ASRCCR[ACIA] 1b - Force ASRCCR[ACIA]=0
10 RSYNOFA	Re-sync Output FIFO Channel Counter If bit set, force ASRCCR[ACOA]=0. If bit clear, do not touch ASRCCR[ACOA]. 0b - Do not touch ASRCCR[ACOA] 1b - Force ASRCCR[ACOA]=0
9-6 —	Should be written as zero for future compatibility.
5-0 INFIFO_THRESHOLDA	Threshold for Pair A's Input FIFO per Channel These bits stand for the threshold for Pair A's input FIFO per channel. Possible range is [0,63]. When the value is n, it means that: when the number of input FIFO fillings of the pair is less than n samples per channel, the input data needed flag is set; when the number of input FIFO fillings of the pair is greater than or equal to n samples per channel, the input data needed flag is automatically cleared.

49.6.1.24 ASRC FIFO Status for Pair A (ASRFSTA)

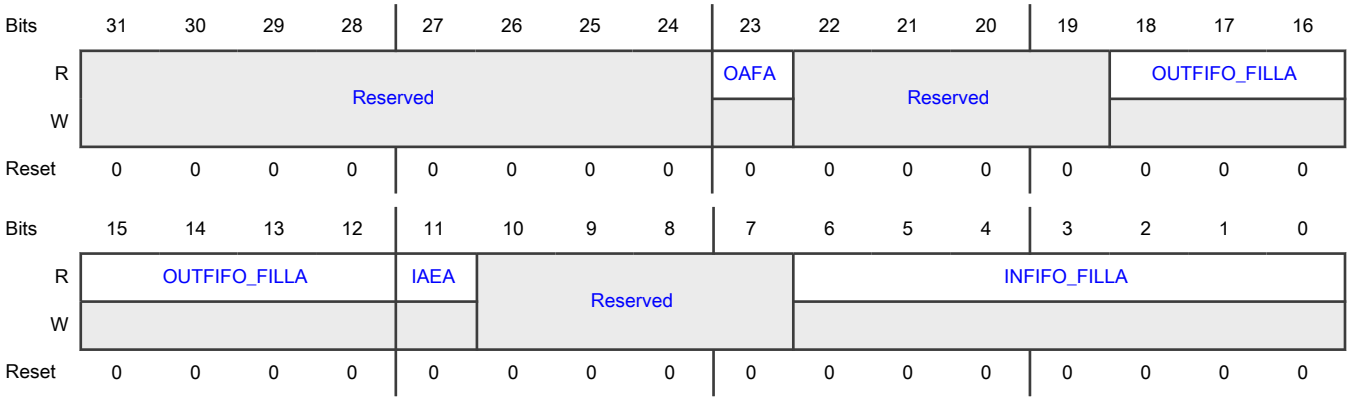
Offset

Register	Offset
ASRFSTA	A4h

Function

Used to show Pair A internal FIFO conditions.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 OAFA	Output FIFO is Near Full for Pair A This bit is to indicate whether the output FIFO of Pair A is near full. 0b - Output FIFO is not near full for Pair A 1b - Output FIFO is near full for Pair A
22-19 —	Should be written as zero for future compatibility.
18-12 OUTFIFO_FILLA	Fillings for Pair A's Output FIFO per Channel These bits stand for the fillings for Pair A's output FIFO per channel. Possible range is [0,64].
11 IAEA	Input FIFO is Near Empty for Pair A This bit is to indicate whether the input FIFO of Pair A is near empty. 0b - Input FIFO is not near empty for Pair A 1b - Input FIFO is near empty for Pair A
10-7 —	Should be written as zero for future compatibility.
6-0 INFIFO_FILLA	Fillings for Pair A's Input FIFO per Channel These bits stand for the fillings for Pair A's input FIFO per channel. Possible range is [0,64].

49.6.1.25 ASRC Misc Control for Pair B (ASRMCRB)

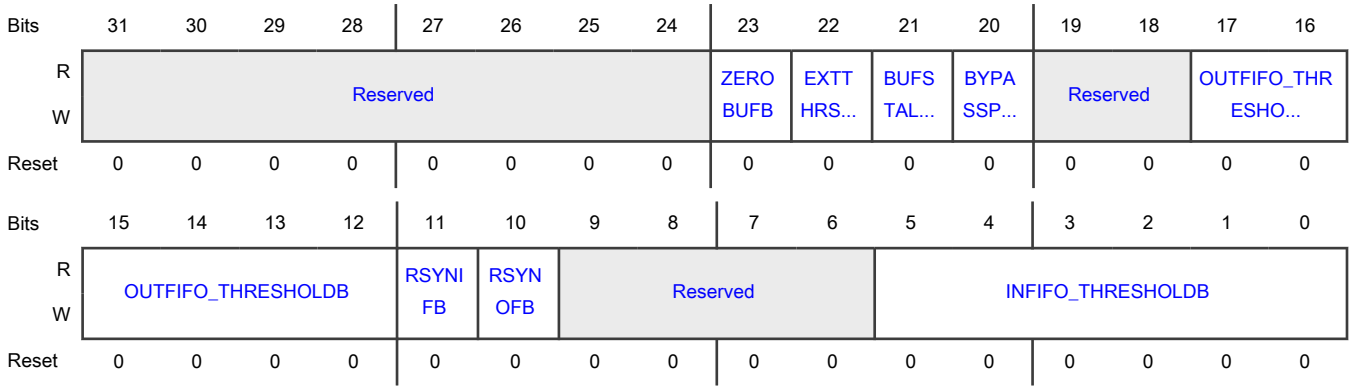
Offset

Register	Offset
ASRMCRB	A8h

Function

Used to control Pair B internal logic.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 ZEROBUFB	Zero Buffer B Initialize buffer of Pair B when pair B is enabled. This bit is used to control whether the buffer is to be zeroized when pair B is enabled. 0b - Zeroize the buffer 1b - Don't zeroize the buffer
22 EXTTHRSHB	Use External Thresholds for FIFO Control of Pair B Determines whether the FIFO thresholds externally defined in this register is used to control ASRC internal FIFO logic for pair B. 0b - Use default thresholds. 1b - Use external defined thresholds.
21 BUFSTALLB	Stall Pair B Conversion in Case of Buffer Near Empty/Full Condition Determines whether the near empty/full FIFO condition stalls the rate conversion for pair B. This option can only work when external ratio is used.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Near empty condition is the condition when input FIFO has less than 4 useful samples per channel. Near full condition is the condition when the output FIFO has less than 4 vacant sample words to fill per channel. 0b - Don't stall Pair B conversion even in case of near empty/full FIFO conditions. 1b - Stall Pair B conversion in case of near empty/full FIFO conditions.
20 BYPASSPOLYB	Bypass Polyphase Filtering for Pair B Determines whether the polyphase filtering part of Pair B conversion is bypassed. 0b - Don't bypass polyphase filtering. 1b - Bypass polyphase filtering.
19-18 —	Should be written as zero for future compatibility.
17-12 OUTFIFO_THRESHOLDB	Threshold for Pair B's Output FIFO per Channel These bits stand for the threshold for Pair B's output FIFO per channel. Possible range is [0,63]. When the value is n, it means that: when the number of output FIFO fillings of the pair is greater than n samples per channel, the output data ready flag is set; when the number of output FIFO fillings of the pair is less than or equal to n samples per channel, the output data ready flag is automatically cleared.
11 RSYNIFB	Re-sync Input FIFO Channel Counter If bit set, force ASRCCR[ACIB]=0. If bit clear, do not touch ASRCCR[ACIB]. 0b - Do not touch ASRCCR[ACIB] 1b - Force ASRCCR[ACIB]=0
10 RSYNOFB	Re-sync Output FIFO Channel Counter If bit set, force ASRCCR[ACOB]=0. If bit clear, do not touch ASRCCR[ACOB]. 0b - Do not touch ASRCCR[ACOB] 1b - Force ASRCCR[ACOB]=0
9-6 —	Should be written as zero for future compatibility.
5-0 INFIFO_THRESHOLDB	Threshold for Pair B's Input FIFO per Channel These bits stand for the threshold for Pair B's input FIFO per channel. Possible range is [0,63]. When the value is n, it means that:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	when the number of input FIFO fillings of the pair is less than n samples per channel, the input data needed flag is set; when the number of input FIFO fillings of the pair is greater than or equal to n samples per channel, the input data needed flag is automatically cleared.

49.6.1.26 ASRC FIFO Status for Pair B (ASRFSTB)

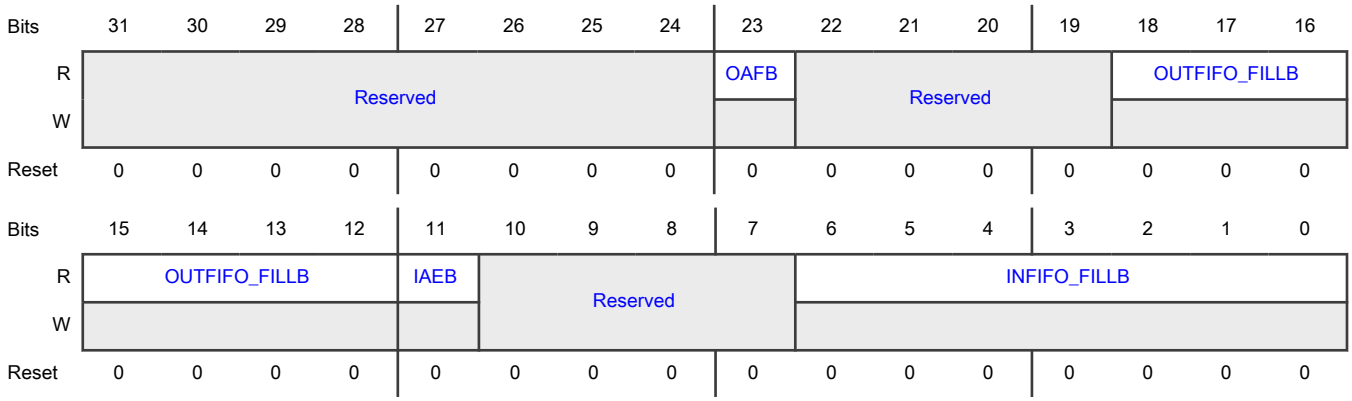
Offset

Register	Offset
ASRFSTB	ACh

Function

Used to show Pair B internal FIFO conditions.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 OAFB	Output FIFO is Near Full for Pair B This bit is to indicate whether the output FIFO of Pair B is near full. 0b - Output FIFO is not near full for Pair B 1b - Output FIFO is near full for Pair B
22-19	Should be written as zero for future compatibility.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
18-12 OUTFIFO_FILLB	Fillings for Pair B's Output FIFO per Channel These bits stand for the fillings for Pair B's output FIFO per channel. Possible range is [0,64].
11 IAEB	Input FIFO is Near Empty for Pair B This bit is to indicate whether the input FIFO of Pair B is near empty. 0b - Input FIFO is not near empty for Pair B 1b - Input FIFO is near empty for Pair B
10-7 —	Should be written as zero for future compatibility.
6-0 INFIFO_FILLB	Fillings for Pair B's Input FIFO per Channel These bits stand for the fillings for Pair B's input FIFO per channel. Possible range is [0,64].

49.6.1.27 ASRC Misc Control for Pair C (ASRMCRC)

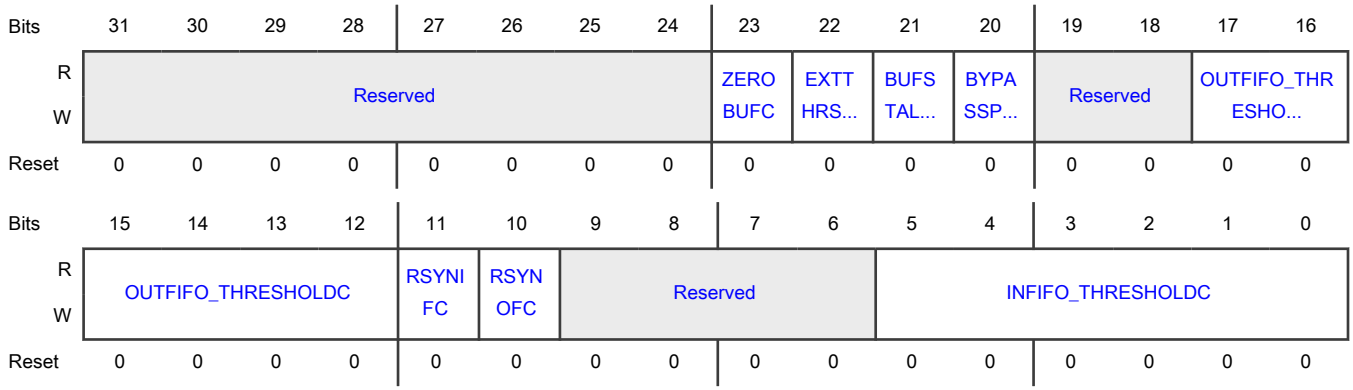
Offset

Register	Offset
ASRMCRC	B0h

Function

Used to control Pair C internal logic.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 ZEROBUFC	Zero Buffer C Initialize buffer of Pair C when pair C is enabled. This bit is used to control whether the buffer is to be zeroized when pair C is enabled. 0b - Zeroize the buffer 1b - Don't zeroize the buffer
22 EXTTHRSHC	Use External Thresholds for FIFO Control of Pair C Determines whether the FIFO thresholds externally defined in this register is used to control ASRC internal FIFO logic for pair C. 0b - Use default thresholds. 1b - Use external defined thresholds.
21 BUFSTALLC	Stall Pair C Conversion in Case of Buffer Near Empty/Full Condition Determines whether the near empty/full FIFO condition stalls the rate conversion for pair C. This option can only work when external ratio is used. Near empty condition is the condition when input FIFO has less than 4 useful samples per channel. Near full condition is the condition when the output FIFO has less than 4 vacant sample words to fill per channel. 0b - Don't stall Pair C conversion even in case of near empty/full FIFO conditions. 1b - Stall Pair C conversion in case of near empty/full FIFO conditions.
20 BYPASSPOLYC	Bypass Polyphase Filtering for Pair C Determines whether the polyphase filtering part of Pair C conversion is bypassed. 0b - Don't bypass polyphase filtering. 1b - Bypass polyphase filtering.
19-18 —	Should be written as zero for future compatibility.
17-12 OUTFIFO_THRESHOLDC	Threshold for Pair C's Output FIFO per Channel These bits stand for the threshold for Pair C's output FIFO per channel. Possible range is [0,63]. When the value is n, it means that: when the number of output FIFO fillings of the pair is greater than n samples per channel, the output data ready flag is set; when the number of output FIFO fillings of the pair is less than or equal to n samples per channel, the output data ready flag is automatically cleared.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 RSYNIFC	Re-sync Input FIFO Channel Counter If bit set, force ASRCCR[ACIC]=0. If bit clear, do not touch ASRCCR[ACIC]. 0b - Do not touch ASRCCR[ACIC] 1b - Force ASRCCR[ACIC]=0
10 RSYNOFC	Re-sync Output FIFO Channel Counter If bit set, force ASRCCR[ACOC]=0. If bit clear, do not touch ASRCCR[ACOC]. 0b - Do not touch ASRCCR[ACOC] 1b - Force ASRCCR[ACOC]=0
9-6 —	Should be written as zero for future compatibility.
5-0 INFIFO_THRESHOLDC	Threshold for Pair C's Input FIFO per Channel These bits stand for the threshold for Pair C's input FIFO per channel. Possible range is [0,63]. When the value is n, it means that: when the number of input FIFO fillings of the pair is less than n samples per channel, the input data needed flag is set; when the number of input FIFO fillings of the pair is greater than or equal to n samples per channel, the input data needed flag is automatically cleared.

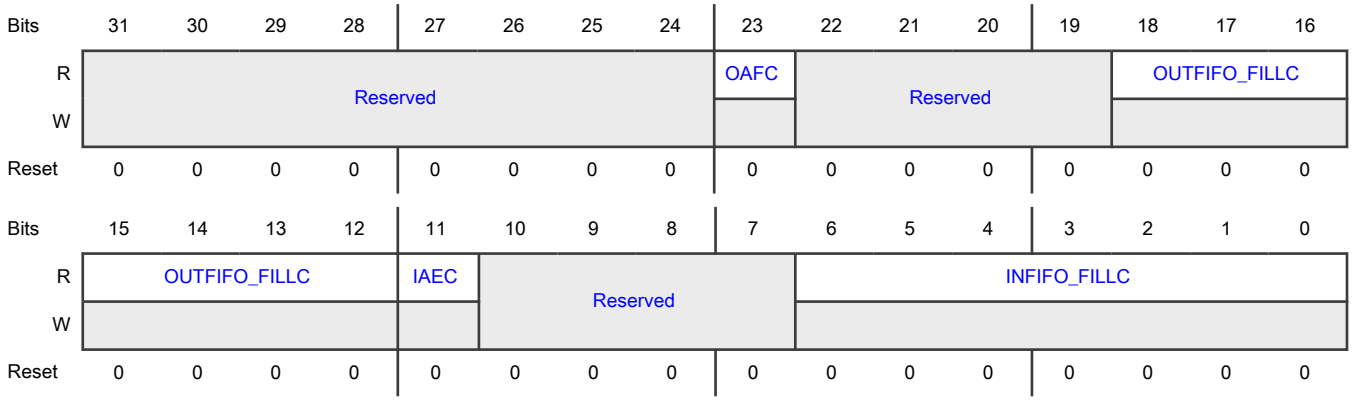
49.6.1.28 ASRC FIFO Status for Pair C (ASRFSTC)

Offset

Register	Offset
ASRFSTC	B4h

Function
Used to show Pair C internal FIFO conditions.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 OAFC	Output FIFO is Near Full for Pair C This bit is to indicate whether the output FIFO of Pair C is near full. 0b - Output FIFO is not near full for Pair C 1b - Output FIFO is near full for Pair C
22-19 —	Should be written as zero for future compatibility.
18-12 OUTFIFO_FILLC	Fillings for Pair C's Output FIFO per Channel These bits stand for the fillings for Pair C's output FIFO per channel. Possible range is [0,64].
11 IAEC	Input FIFO is Near Empty for Pair C This bit is to indicate whether the input FIFO of Pair C is near empty. 0b - Input FIFO is not near empty for Pair C 1b - Input FIFO is near empty for Pair C
10-7 —	Should be written as zero for future compatibility.
6-0 INFIFO_FILLC	Fillings for Pair C's Input FIFO per Channel These bits stand for the fillings for Pair C's input FIFO per channel. Possible range is [0,64].

49.6.1.29 ASRC Misc Control 1 for Pair X (ASRMCR1A - ASRMCR1C)

Offset

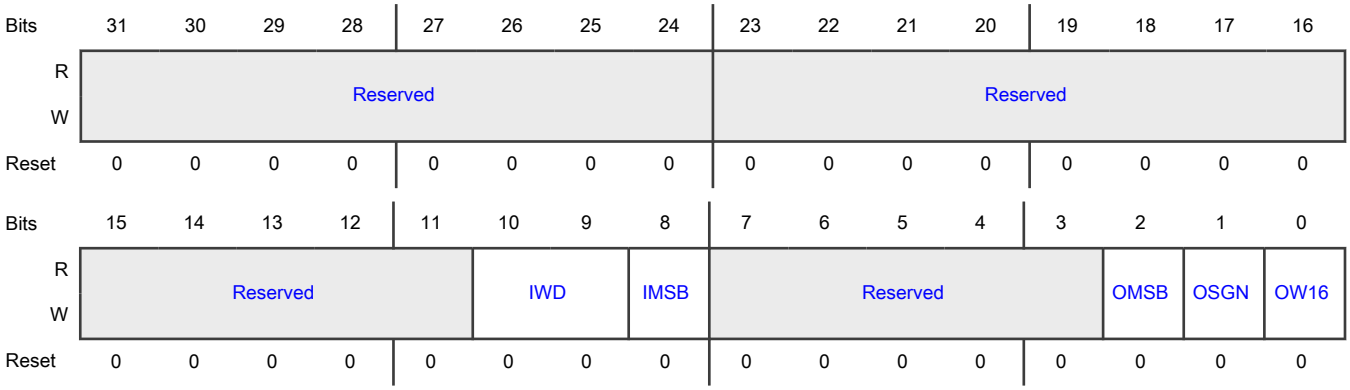
Register	Offset
ASRMCR1A	C0h
ASRMCR1B	C4h
ASRMCR1C	C8h

Function

Used to control Pair *x* internal logic (for data alignment etc.).

The bit assignment for all the input data formats is the same as that supported by the SAI.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-11 —	Should be written as zero for future compatibility.
10-9 IWD	Data Width of the Input FIFO These three bits determines the bitwidth for the audio data into ASRC. 00b - 24-bit audio data. 01b - 16-bit audio data. 10b - 8-bit audio data. 11b - Reserved.
8	Data Alignment of the Input FIFO

Table continues on the next page...

Table continued from the previous page...

Field	Function
IMSB	Determines the data alignment of the input FIFO. 0b - LSB aligned 1b - MSB aligned
7-3 —	Should be written as zero for future compatibility.
2 OMSB	Data Alignment of the Output FIFO Determines the data alignment of the output FIFO. 0b - LSB aligned 1b - MSB aligned
1 OSGN	Sign Extension Option of the Output FIFO Determines the sign extension option of the output FIFO. 0b - No sign extension 1b - Sign extension
0 OW16	Bit Width Option of the Output FIFO Determines the bit width option of the output FIFO. 0b - 24-bit output data 1b - 16-bit output data

Chapter 50
PDM Microphone Interface (PDM)

50.1 Chip-specific PDM information

Table 331. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

50.2 Overview

MICFIL delivers audio from microphones to the processor in several applications, such as mobile telephones. As current digital audio systems use a multi-bit audio signal (also known as multi-bit PCM) to represent the signal, this module implements the required digital interface (a series of filters) to transform a pulse density modulated (PDM) microphone bitstream into a 24-bit PCM signal in the audio band, at a configurable output sampling rate.

The implementation of this digital interface is based on the application of digital signal processing techniques used in hardware. The MICFIL architecture is designed to save gate count and minimize power consumption.

You can implement MICFIL to work in a multi-channel mode. All channels have the same configuration, but you could independently turn on or turn off each input channel.

50.2.1 Block diagram

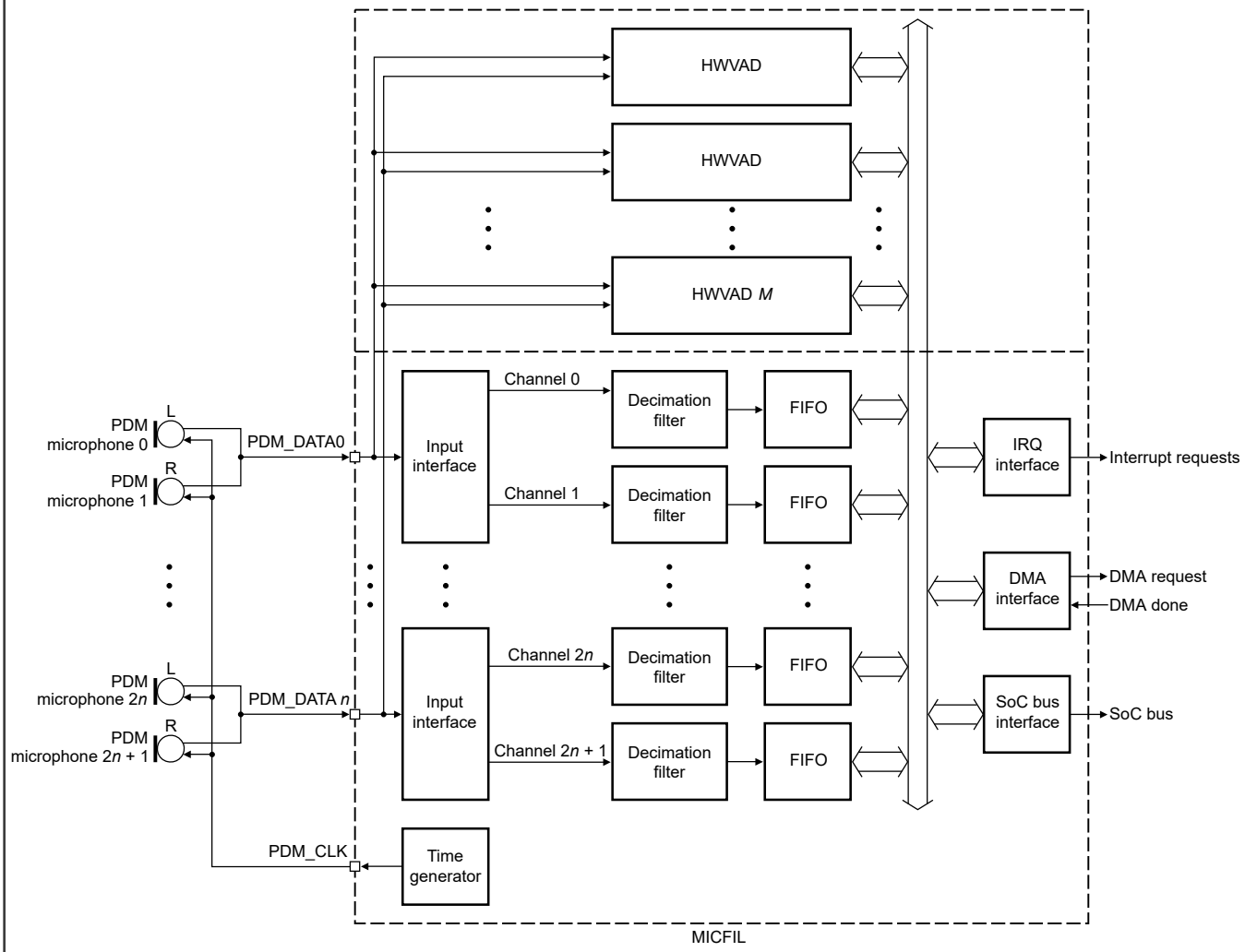


Figure 295. MICFIL block diagram

50.2.2 Features

- Decimation filters:
  - Fixed-point filtering
  - 24-bit PCM audio output
  - Internal clock divider for a programmable PDM clock generation:
  - Full or partial set of channel operations with individual enable controls
  - Programmable decimation rate
  - Programmable DC remover
  - Programmable DC remover at output
  - Range adjustment capability
- FIFOs with interrupt and DMA capability: each FIFO having a length of 8 entries
- HWVAD, equipped with:

— Interrupt capability

— Zero-Crossing Detection (ZCD) option

50.3 Functional description

50.3.1 Time generator

The time generator unit:

- Generates the PDM_CLK to microphones. This clock is the same and is active for all the PDM microphones, which means, it is not possible to turn off the PDM_CLK only for one single microphone.
- Delivers the PDM_CLK to all microphones that must operate at the same clock frequency. Each input interface receives a time multiplexed PDM bitstream from two PDM microphones and it separates audio information in two channels: left (0) and right (1). Every decimation filter, corresponding to its channel, does this processing.

NOTE

A soft reset affects the time generator unit (see CTRL_1[SRES]). The PDM_CLK starts with a rising edge after a soft reset is issued.

50.3.1.1 Clock divider

The sampling rate of the microphone input depends on the PDM_CLK rate, which you can trim using an internal clock divider, configuring the quality mode (see Quality modes) or trimming the MICFIL's root clock frequency (MICFIL_CLK_ROOT rate).

The internal clock divider frequency. You must configure CTRL_2[CLKDIV] by using Equation 4.

$$\text{CLKDIV} = \frac{\text{MICFIL\_CLK\_ROOT rate}}{8 \times \text{OSR} \times (\text{output rate})}$$

Equation 4. CLKDIV value calculation

After calculating the value of CTRL_2[CLKDIV], you can estimate the generated PDM_CLK rate based on Equation 5, where the "K" factor value depends on the quality mode shown in Table 332 and floor(x) is the function that takes a real number x as input and gives the greatest integer less than or equal to x as output.

$$\text{PDM\_CLK rate} = \frac{\text{MICFIL\_CLK\_ROOT}}{2 \times \text{floor} \times (\text{K} \times \text{CLKDIV})}$$

Equation 5. PDM_CLK rate calculation

Table 332. K factor value

Quality mode	K factor
High Quality	1/2
Medium Quality, Very-Low Quality 0	1
Low Quality, Very-Low Quality 1	2
Very-Low Quality 2	4

50.3.1.1.1 Minimum required CLKDIV

NOTE

This minimum CLKDIV condition is required only if any decimation filter channel is enabled and the clock divider is enabled, otherwise this condition should be ignored.

To work properly the CLKDIV value must have a minimum value according to the quality mode configured (see Quality modes), the number of enabled channels (EC), OSR value and K factor (see I/O timing requirements). When this condition is not met, the LOWFREQF bit is asserted and an error interrupt can be issued (in case ERREN = 1).

The equations to estimate the minimum required CLKDIV value depending on the configured quality mode are shown in Equation 6 (for medium, high and low quality modes) and Equation 7 (for very low quality modes), on these equations floor(x) is the function that takes as input a real number x and gives as output the greatest integer less than or equal to x.

$$\text{floor}(\text{K x CLKDIV}) \geq \text{K x } \frac{10 + 93EC}{8\text{xOSR}}$$

Equation 6. Minimum required CLKDIV value in Medium, High and Low Quality modes

$$\text{floor}(\text{K x CLKDIV}) \geq \text{K x } \frac{10 + 43EC}{8\text{xOSR}}$$

Equation 7. Minimum required CLKDIV value in Very Low Quality modes

50.3.1.2 MICFIL_CLK_ROOT rate and CLKDIV calculation examples

Example 1:

If it is required an output rate = 16 kHz and enables 4 channels (EC=4). A valid configuration is following:

1. Configure High Quality mode and CIC decimation by 32 (CICOSR = 0, then OSR = 16).
2. Check the PDM_CLK rate for High Quality mode in Table 336. PDM_CLK rate = 16 kHz x 8 x OSR = 16 kHz x 8 x 16 = 2.048 MHz.
3. Check the K factor value for High Quality mode in Table 332. K factor is 1/2.
4. Calculate minimum CLKDIV value (see Equation 6): floor(K x CLKDIV) ≥ K x (10 + 93EC)/(8 x OSR) then floor(K x CLKDIV) ≥ 1.4921875. Finally, CLKDIV ≥ 4.
5. Calculate the minimum MICFIL_CLK_ROOT rate:
   - From Equation 4: MICFIL_CLK_ROOT rate = 8 x OSR x CLKDIV x (Output rate)
   - From step 4 result: CLKDIV ≥ 4, then MICFIL_CLK_ROOT rate ≥ 8 x OSR x 4 x 16 kHz.
   - Finally: MICFIL_CLK_ROOT rate ≥ 8.192 MHz
6. Check if selected MICFIL_CLK_ROOT rate and CLKDIV value satisfies the IO timing requirements.
7. Finally, check again if the selected MICFIL_CLK_ROOT rate satisfies Equation 4. If not, adjust CLKDIV or change MICFIL_CLK_ROOT rate accordingly.

Example 2:

If it is required an output rate = 48 kHz and enables 4 channels (EC=4). A valid configuration is following:

1. Configure Medium Quality mode and CIC decimation by 16 (CICOSR = 0, then OSR = 16).
2. Check the PDM_CLK rate for Medium Quality mode in Table 336. PDM_CLK rate = 48 kHz x 4 x OSR = 48 kHz x 4 x 16 = 3.072 MHz.
3. Check the K factor value for Medium Quality mode in Table 332. K factor is 1.
4. Calculate minimum CLKDIV value (see Equation 6): floor(K x CLKDIV) ≥ K x (10 + 93EC)/(8 x OSR) then floor(K x CLKDIV) ≥ 2.984375. Finally, CLKDIV ≥ 3.
5. Calculate the minimum MICFIL_CLK_ROOT rate:
   - From Equation 4: MICFIL_CLK_ROOT rate = 8 x OSR x CLKDIV x (Output rate)
   - From step 4 result: CLKDIV ≥ 3, then MICFIL_CLK_ROOT rate ≥ 8 x OSR x 3 x 48 kHz.
   - Finally: MICFIL_CLK_ROOT rate ≥ 18.432 MHz

- A different value of CLKDIV may be preferable, CLKDIV = 4 results in MICFIL_CLK_ROOT = 24.576 MHz, a standard audio frequency.

6. Check if selected MICFIL_CLK_ROOT rate and CLKDIV value satisfies the IO timing requirements.
7. Finally, check again if the selected MICFIL_CLK_ROOT rate satisfies Equation 4. If not, adjust CLKDIV or change MICFIL_CLK_ROOT rate accordingly.

50.3.1.3 I/O timing requirements

The PDM microphones must meet the setup and hold timing requirements shown in Table 333 and Figure 296. The "K" factor value in Table 333 depends on the quality mode you select based on Table 332.

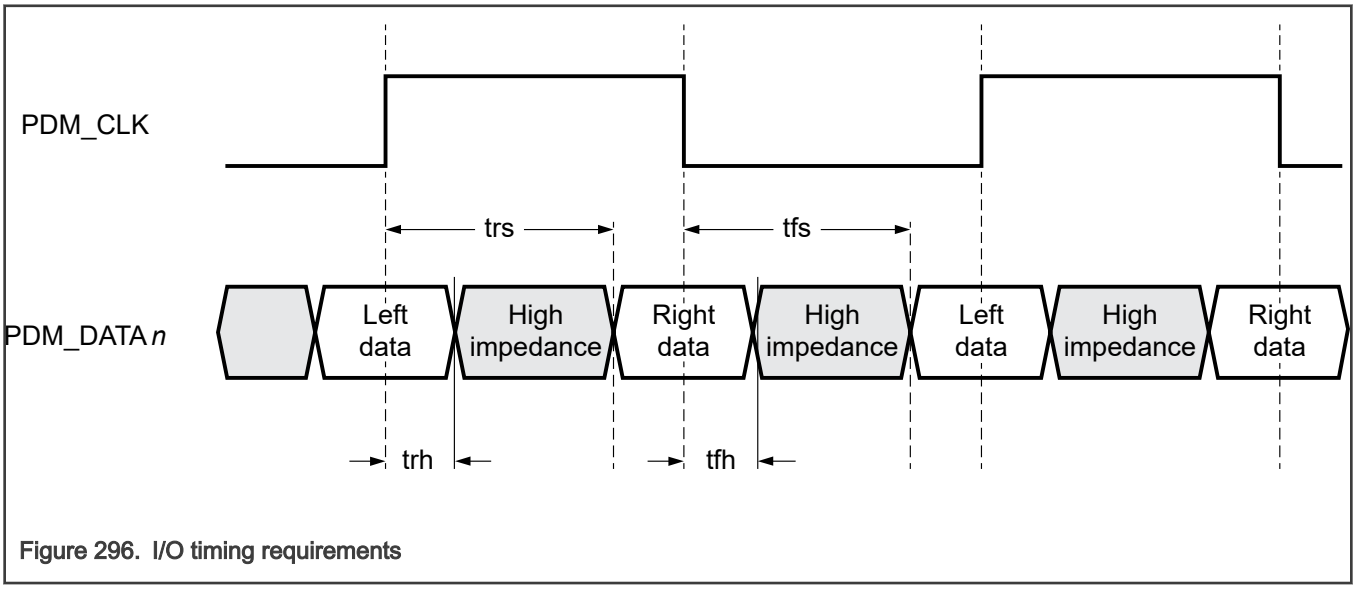


Figure 296. I/O timing requirements

Table 333. Timing parameters

Parameter	Value
trs, tfs	$\leq \frac{\text{floor}(\text{K x } CLKDIV) - 1}{\text{MICFIL\_CLK\_ROOT rate}}$ Depending on the value of "K," you must ensure that "floor(K x CTRL_2[CLKDIV])" > 1 to avoid timing problems.
trh, tfh	≥ 0

50.3.2 Input interface

Figure 297 shows the timing diagram of the input interface signals . The bitstream incoming from the microphone data input "n" (PDM_DATAn) in the first half (right microphone) of the PDM_CLK is directed to channel "2n+1" and the data generated during the last half (left microphone) is directed to channel "2n".

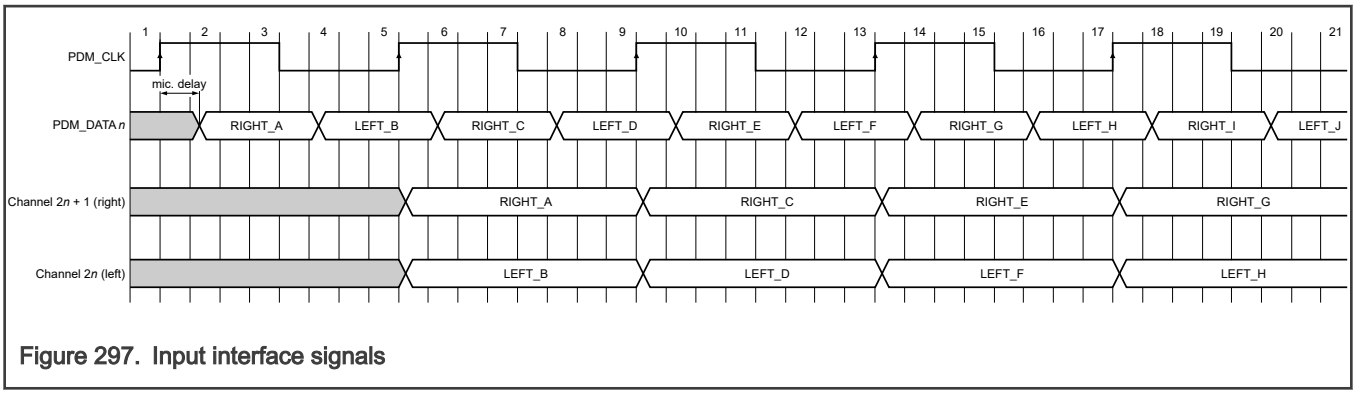


Figure 297. Input interface signals

50.3.3 Decimation filter

A decimation filter facilitates PDM to PCM conversion.

Figure 298 shows the block diagram of the decimation filter. The block consists of:

- A cascade integrator comb (CIC) filter, which is a low-pass filter that you commonly use to filter sigma-delta modulator output signals. The filter is implemented in the audio band (20 Hz–22.5 kHz @48 kHz output sampling rate by default) with a configurable decimation rate. You can implement it using a series of CIC, compensation, and DC remover filters.
- A DC remover, which is a high-pass filter that you use to remove the DC component of the processed signal with a configurable cut-off frequency. Two DC removers exist: one in the input of the decimation filter, embedded in the CIC filter, and another in the last stage of the decimation filter.
- Two compensation filters for each channel, which implement a low-pass digital filter with decimation by 2. You can use them to compensate the high CIC drop in passband. These filters help the FIR filter block to flatten the overall passband response.

You activate the "Sampler" block depending on the quality mode selected (using CTRL_2[QSEL]). In Low-Quality and Very-Low-Quality 2 modes, the sampling rate is doubled.

The decimation filter output depends on the quality mode that you select by using CTRL[QSEL], based on different compensation filter output. You can supply the DC remover with the 2 FS signal, instead of the 1 FS signal, to achieve lower-power consumption, bypassing the second compensation decimation filter. The second compensation filter is bypassed when CTRL_2[QSEL] = 110b, 101b, or 100b (in Very-Low Quality modes).

You can adjust the range of each decimation filter by using a range adjustment block so that you can raise an error interrupt in case an adjustment overflow or underflow is detected.

The output of a decimation filter is stored into a FIFO buffer, and each FIFO is mapped to MICFIL Output Result (DATACH0 - DATACH7). It is possible to generate either an interrupt or a DMA request when, in each FIFO of all enabled channels, the number of data stored surpasses a configured watermark.

Also, independently on decimation filters, there is a Hardware Voice Activity Detector (HWVAD) which implements voice-detection algorithms to generate wake-up interrupts. See Hardware voice activity detector.

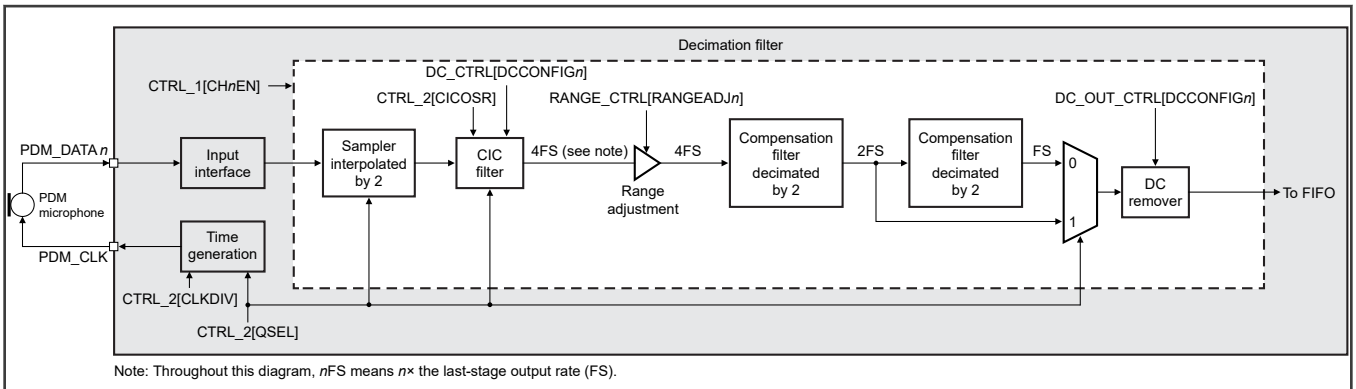


Figure 298. Decimation filter block diagram

50.3.3.1 CIC filter

The CIC filter is an optimized implementation of the low-pass SINC filter, with multiple stages. It is widely used because it requires just adders and subtracters. The CIC filter comprises an integrator and a comb as its sub-blocks. As shown in the following figure, CTRL_2[CICOSR] and the quality mode you select (see Quality modes) define the CIC decimation rate.

$$OSR = 16 - CICOSR$$

$$\text{CIC decimation rate} = \begin{cases} 2 \times OSR & \text{; If HQ, VLQ0} \\ OSR & \text{; others} \end{cases}$$

Equation 8. CIC decimation rate

50.3.3.2 DC remover

Two DC removers exist in the filter chain, one in the input and another in the output. The former is integrated in the CIC filter, and the latter is placed just before the FIFO. Both can be bypassed independently.

The DC remover eliminates the DC level component of the signal using a high-pass filter with a very low cut-off frequency that you select using DC_CTRL[DCCONFIGn]. This filter is bypassed when the value of this field is 11b.

50.3.3.3 Compensation filters

A compensation filter divides the baseband signal into two equal subbands before it could to be decimated by 2. It also compensates the CIC filter drop in passband and helps to flatten the overall passband response. MICFIL includes two compensation filter stages, with the possibility of bypassing the last one.

50.3.3.4 Filter performance

50.3.3.4.1 Overall filter gain

As shown in Equation 9, the overall filter gain depends on:

- The quality mode you select (see Quality modes).
- The CIC decimation rate (see Equation 8).
- Dynamic range adjustment of the CIC filter (see RANGE_CTRL[RANGEADJn]).

NOTE

To get a better audio performance, you must configure and adjust RANGE_CTRL[RANGEADJn] depending on the quality mode selected, as pointed out in the register description.

$$\text{Overall filter gain (dB)} = \begin{cases} 100 \times \log_{10}(32 - 2 \times \text{CICOSR}) + 6.02 \times \text{RANGE\_CTRL[RANGEADJ}n\text{]} - 150.50 & ; \text{if QSEL} \in [\text{HQ,VLQ0}] \\ 100 \times \log_{10}(16 - \text{CICOSR}) + 6.02 \times \text{RANGE\_CTRL[RANGEADJ}n\text{]} - 150.50 & ; \text{others} \end{cases}$$

Equation 9. Overall filter gain

Finally, the input and output dBFS levels relate to each other, as shown in Equation 10.

$$\text{Output (dBFS)} = \text{input (dBFS)} + \text{overall filter gain (dB)}$$

Equation 10. Filter input and output relationship

50.3.3.4.2 Overall passband

The decimation filter passband depends on the quality mode and output rate (FS). For instance, the overall filter passband for different output data rates when MICFIL is in High (HQ), Medium (MQ), and Low-Quality (LQ) modes is shown in Table 334.

Table 334. Filter passband in HQ, MQ, and LQ modes (DC_CTRL[DCCONFIGn] = 0)

Output data rate	Overall filter passband
FS	**0.0004FS - 0.4688FS**
16 kHz	6.67 Hz - 7.50 kHz
48 kHz	20 Hz - 22.5 kHz

The overall filter passband for different output data rates when MICFIL is in Very-Low Quality (VLQ) modes is shown in Table 335.

Table 335. Filter passband in VLQn modes (DC_CTRL[DCCONFIGn] = 0)

Output data rate	Overall filter passband
FS	**0.0002FS - 0.2344FS**
16 kHz	3.33 Hz - 3.75 kHz
48 kHz	10.00 Hz - 11.25 kHz
96 kHz	20 Hz - 22.5 kHz

50.3.3.5 Quality modes

If the decimation filter is running independently on the selected power mode, MICFIL can operate in multiple different quality modes that you can select by using CTRL_2[QSEL]. The decimation and interpolation rates in internal blocks are shown in Table 336, where the oversampling rate (OSR) depends on the value of CTRL_2[CICOSR], as shown in the following equation.

$$\text{OSR} = 16 - \text{CICOSR}$$

Equation 11. OSR

You can configure the selected quality dynamically when the decimation filter is running; the output rate is maintained in every quality selected.

The passband of the decimation filter depends on the quality mode (see Overall passband).

To maintain the same output rate in the decimation filter when the quality is changed, the PDM_CLK rate changes depending on the quality selected. For more on PDM_CLK, see Time generator.

You can enable or disable the sampler and compensation filters according to the selected quality. When they are disabled, you can achieve a low-power consumption. The selected quality also determines whether the CIC filter decimation rate is doubled. For more, see Decimation filter.

With an increase of the clock rate and OSR, you can expect a better signal-to-noise ratio (SNR) performance, which in turn leads to a higher audio quality.

Table 336. Quality modes

Quality mode	CTRL_2[QSEL]	Sampler interpolation	CIC filter decimation	First-compensation filter decimation	Second-compensation filter decimation	PDM_CLK rate	Passband
High Quality	001	—	:(2OSR)	:2	:2	Output rate × 8 × OSR	To ~0.5 × output rate
Medium Quality	000	—	:OSR	:2	:2	Output rate × 4 × OSR	To ~0.5 × output rate
Low Quality	111	× 2	:OSR	:2	:2	Output rate × 2 × OSR	To ~0.5 × output rate
Very-Low Quality 0	110	—	:(2OSR)	:2	—	Output rate × 4 × OSR	To ~0.25 × output rate
Very-Low Quality 1	101	—	:OSR	:2	—	Output rate × 2 × OSR	To ~0.25 × output rate
Very-Low Quality 2	100	× 2	:OSR	:2	—	Output rate × OSR	To ~0.25 × output rate

50.3.3.6 Output rate

The output rate is the sampling rate at the decimation filter output. It depends on the sampling rate of the microphone input and the configured quality mode (see Quality modes). Common audio sampling rates are 48 kHz and 16 kHz. In this chapter, the output rate is sometimes referred to as FS.

50.3.4 Hardware voice activity detector

The Hardware Voice Activity Detector (HWVAD) is a block responsible for detect voice activity in a channel selected by the user as shown in the Figure 299 and Figure 300. The HWVAD takes data from the input of a selected PDM microphone to detect if there is any voice activity. If a voice activity is detected, an interrupt could be delivered to the system. It can be configured in Envelope-based or Energy-based mode and it should be configured as described in Initialization.

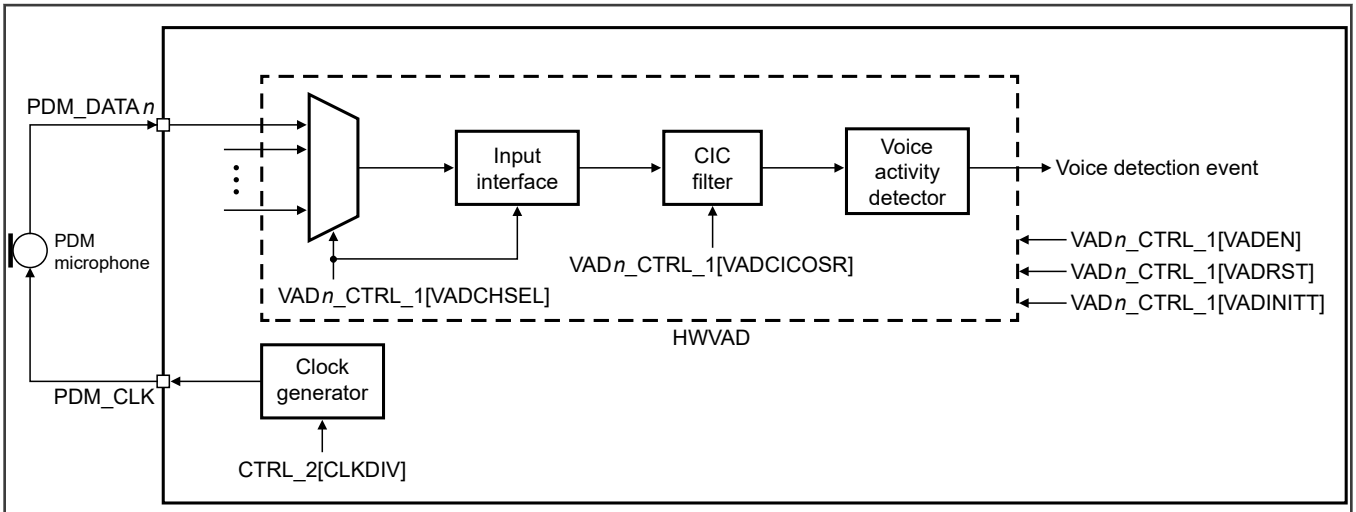


Figure 299. HWVAD block diagram

The HWVAD can divide the incoming PCM signal from its internal CIC in frames, the frame energy is used to estimate the background noise. Both frame energy and noise energy are examined, a voice activity is detected when a specific delta between them is detected, then a interrupt will be issued. Optionally a Zero-Crossing Detection block (ZCD) could be enabled to avoid low energy voiced speech be missed, improving the voice detection performance.

The HWVAD detector can be enabled by the VADEN bit (VAD_CTRL_1 register) and could be reset by software through the VADRST bit (VAD_CTRL_1 register) . It is important to remark that the HWVAD detector could be enabled and/or reset only when the MICFIL isn't running i.e. when the BSY_FIL bit in STAT register is cleared.

To perform voice detection a Noise Filter and Signal Filter are used as shown Figure 300. In addition, a minimum block in the output of the Noise Filter and a maximum block in the output of the Signal Filter can be used. The result of both blocks can be scaled by gain blocks. Finally, the signal and noise estimated are compared, voice is detected when signal is greater than noise.

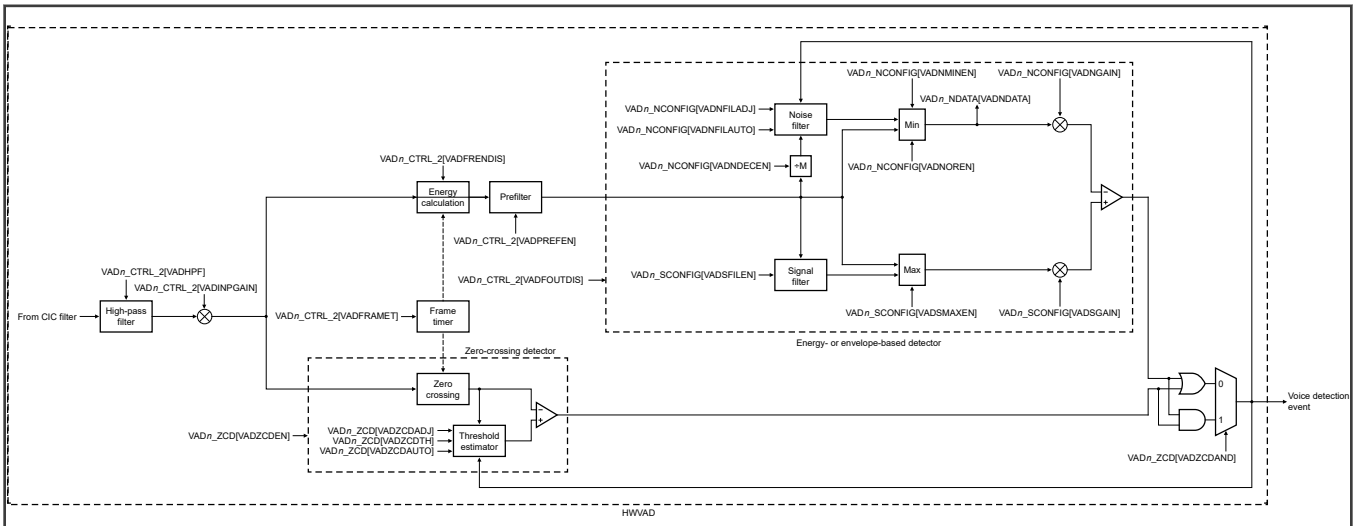


Figure 300. HWVAD block diagram (continued)

50.3.4.1 High-pass filter

The purpose of the High-Pass Filter is to remove the low frequency components. The cut-off frequency is selected by the VADHPF field. This filter is bypassed when VADHPF=00.

50.3.4.2 Input setting

The HWVAD input has internally a user configurable High-Pass filter (see VADHPF bit description) and a user configurable gain (see VADINPGAIN). Both controls could be used to configure the module based on external noise conditions.

50.3.4.3 Frame time setting

In order to perform voice detection the signal frames incoming from the CIC filter can be divided in frames whose duration can be configured. The frame time will depend on the CIC output rate, the value of VADFRAMET and a constant as is defined in Equation 12.

$$frame\ \ time = \frac{VADFRAMET \times 192}{CIC\ \ output\ \ rate}$$

Equation 12. Frame time

For instance, in the typical case of the CIC filter output rate would be 192 kHz, the frame time would be 10 ms when VADFRAMET is 10.

50.3.4.4 Initialization time

An initialization time can be configured by the VADINITT field. During this initialization time no voice detection event is issued, all detectors are kept disabled.

For instance, in case that frame time is 10ms and VADINITT = 10, the initialization time would be 110ms.

$$initialization\ \ time = (\mathrm{VADINITT}+1) \times frame\ \ time$$

Equation 13. Initialization time

50.3.4.5 Energy calculation

In case VADFRENDIS = 0, the sum of the absolute values of all the samples of each frame are used to detect voice. Otherwise, if VADFRENDIS = 1, the absolute value of the samples are used to detect voice i.e. the data incoming from CIC is not divided in frames.

50.3.4.6 Pre-filter

The Pre-Filter block is a low-pass filter which can be enabled when VADPREFEN = 1.

50.3.4.7 Energy/Envelope-based detector

To perform voice detection a Noise Filter and Signal Filter are used as shown Figure 300. In addition, a minimum block in the output of the Noise Filter and a maximum block in the output of the Signal Filter can be used. The result of both blocks can be scaled by gain blocks. Finally, the signal and noise estimated are compared, voice is detected when signal is greater than noise.

50.3.4.7.1 Noise filter

The Noise Filter is a low-pass filter . This filter could be adjusted by the VADNFILADJ field . In case VADNFILAUTO = 1 the noise filter is enabled only when voice is not detected, when voice is detected the filter output does not change. In case VADNFILAUTO = 0 the filter is always enabled.

A decimation stage in the input of the Noise Filter can be enabled by the VADNDECEN field.

50.3.4.7.2 Minimum noise

In order to estimate the noise, a block which calculates the minimum value between the input and output of the noise filter is used. This block can be enabled by VADNMINEN, otherwise the output of the noise filter is passed directly.

The estimated noise of the output of this block is stored in the VADNDATA register for post-processing via software.

50.3.4.7.3 Signal filter

The Signal Filter is a low-pass filter . This filter will be enabled when VADSFILEN=1, otherwise the filter will be bypassed.

50.3.4.7.4 Maximum signal

In order to estimate the signal, a block which calculates the maximum value between the input and output of the signal filter can be used. This block can be enabled by VADSMAXEN.

50.3.4.7.5 Filter result gain setting

There are two gain adjustment controls, one for the estimated noise (VADNGAIN) and another for signal estimated (VADSGAIN). Both gain adjustments outputs are compared, the comparison result determines whether or not there is voice in the channel.

50.3.4.8 Zero-crossing detector

Optionally, the HWVAD performance can be increased by the use of a Zero-Crossing Detection (ZCD) circuit. It is enabled by a VADZCDEN bit.

The threshold of the ZCD could be static or calculated dynamically depending on the VADZCDAUTO bit. When it is dynamically determined (VADZCDAUTO = 1) the threshold is calculated automatically taking VADZCDTH as initial value, otherwise (VADZCDAUTO = 0) the threshold is static in the VADZCDTH user configurable value.

You can use AND or OR to combine the ZCD result with the energy-based or envelope-based detection, it is selected by the VADZCDAND bit. The filter coefficient is equal to (VADZCDADJ + 1) × 2^{-4}.

50.3.4.9 Interrupt

The HWVAD issues an interrupt when voice activity is detected and the VADIE bit in the VAD_CTRL_1 register is enabled. The voice activity event also will set the VADIF flag in the VAD_STAT register, this bit will be cleared when it is written 1.

50.3.4.10 Extended operation

The VADIF bit could be used for a HWVAD extended operation i.e. using the VADNDATA value to determine a more accurate voice detection event by software. The procedure to follow will be:

1. An interrupt is issued by the HWVAD, the VADIF is set.
2. Leave the VADIF asserted while software is processing the VADNDATA field content.
3. When software finishes to process the VADNDATA and detects voice activity:
   - If Envelope-based Detection is used, pulse VADST10 by more than 2-cycles and clear VADIF flag.
   - If Energy-based Detection is used, clear VADIF flag.

50.3.4.11 VAD soft-reset

The VAD soft-reset is requested by writing 1 to the VADRST bit. It resets the following blocks and/or registers:

- VADIF, VADINSATF fields.
- All internal VAD buffers.

50.3.5 Bus interface

The bus interface manages DMA transactions, interrupt requests, and user register interface with the chip.

MICFIL can deliver an interrupt request or DMA request. Then, the chip or DMA, respectively, could access the filter results stored in FIFOs via the internal bus interface.

For detailed information on DMA transactions and interrupt requests, see DMA and Interrupts.

50.3.5.1 FIFO

This is the first-in, first-out (FIFO) buffer that saves the overall filtering results. The length of each FIFO is 8 entries, and there exists one FIFO for each channel. A push writes data into the FIFO, and a read pops data from FIFO. The push and pop operations from FIFO are in different clock domains. The push operation uses the MICFIL clock while the pop operation uses the CPU clock. The FIFO has a configurable watermark (see FIFO_CTRL[FIFOWMK]). When the amount of data in all FIFOs of the enabled channels is greater than the watermark value (FIFO_CTRL[FIFOWMK]), a DMA or IRQ is requested, depending on the value of CTRL_1[DISEL]. After CTRL_1[PDMIEN] enables the filters, the initial FIR results are discarded and not written into the FIFOs. Approximately 68 initial FIR results are not stored in FIFOs (FIR_RDY=0). After the filter generates valid data, the results are pushed into the FIFO.

A new filtered result is discarded (not stored) when the FIFO is full and CTRL_1[FIFOOVFn] becomes 1. Similarly, when trying a pop operation, if the FIFO is empty, no valid data is read and CTRL_1[FIFOUNDn] becomes 1. In both cases, when overflow or underflow occurs, they are flagged in MICFIL FIFO Status (FIFO_STAT) to the respective channel, and an error interrupt is generated if CTRL_1[ERREN] is 1.

When you reset the FIFO by writing 1 to CTRL_1[SRES], zero values fill all FIFO positions, as well as push and pop pointers are set to their initial positions.

50.3.6 Modes of operation

This section describes the MICFIL operating modes that depend on:

- The chip's power mode
- The chip's Doze mode .
- The chip's Debug mode .
- Configuration of CTRL_1[MDIS], CTRL_1[DBGE], CTRL_1[DBG], and CTRL_1[DOZEN], as summarized in the following table.

Table 337. Mode selection

<table>
<tr><th>Mode</th><th>CTRL_1[MDIS]</th><th>Chip Doze and DOZEN = 1</th><th>Chip Power mode</th><th>CTRL_1[DOZEN]</th><th>Chip Debug | CTRL_1[DBG]</th><th>CTRL_1[DBGE]</th><th>CPU clock</th><th>MICFIL internal clock</th><th>Decimation Filter state</th><th>HWVAD[n] state</th></tr>
<tr><td>Normal</td><td rowspan="6">0</td><td rowspan="4">0</td><td rowspan="2">RUN</td><td rowspan="2">Any</td><td>0</td><td>Any</td><td rowspan="3">Enabled</td><td rowspan="4">Enabled</td><td>Run (depends on PDMIEN and CHnEN)</td><td>Run (depends on PDMIEN and VADEN)</td></tr>
<tr><td>Debug</td><td rowspan="2">1</td><td>1</td></tr>
<tr><td>Low-Power</td><td rowspan="2">SLEEP</td><td rowspan="2">0</td><td>0</td><td rowspan="3">Any</td><td rowspan="3">Any</td><td>Stop</td><td>Stop</td></tr>
<tr><td>DEEPSLEEP</td><td rowspan="3">Gated (externally)</td><td>Run (depends on PDMIEN and CHnEN)</td><td rowspan="2">Run (depends on PDMIEN and VADEN)[1]</td></tr>
<tr><td>Disable/ Low-Leakage</td><td>SLEEP/ DEEPSLEEP</td><td>1</td><td rowspan="2">Gated</td><td rowspan="2">Stop</td><td rowspan="2">Stop</td></tr>
<tr><td>1</td><td>Any</td><td>Any</td><td>Enabled</td></tr>
</table>

Table continues on the next page...

Table 337. Mode selection (continued)

Mode	CTRL_1[MDIS]	Chip Doze and DOZEN = 1	Chip Power mode	CTRL_1[DOZEN]	Chip Debug | CTRL_1[DBG]	CTRL_1[DBGE]	CPU clock	MICFIL internal clock	Decimation Filter state	HWVAD[n] state
	1	Any					Gated (internally)			

1. HWVAD[n] will run in DEEPSLEEP only if it is placed on an always-on power domain in the chip.

50.3.6.1 Normal mode

This is the default operational mode for MICFIL. In this mode, you can enable the channel functionality to perform the filtering operation.

50.3.6.2 Debug mode

You could use this mode to check the status of the module, when requested. In this mode, you can stop MICFIL after the remaining data processing is complete, depending on the value of CTRL_1[DBGE]. You can activate this mode by writing 1 to CTRL_1[DBG].

If CTRL_1[DBGE] = 0, and CTRL_1[DBG] = 1 when MICFIL is running, MICFIL stops after processing the remaining data. Otherwise, if DBGE = 1, MICFIL continues to run.

MICFIL cannot enter Debug mode after entering DLL or Low-Power mode.

The Debug mode affects all channels in MICFIL. Access to output buffers, as well as their related flags, remains operational.

50.3.6.3 Low-Power mode

MICFIL enters In Low-Power mode, MICFIL is able to filter the PDM microphone data and perform voice detection such as in Normal mode, and can generate a wake-up event through asynchronous DM or IRQ requests, depending on the values of CTRL_1[CHnEN], VADEN, and CTRL_1[PDMIEN].

The Low-Power mode is entered when the SoC enters to SLEEP or DEEPSLEEP modes and DOZEN=0. In this mode the CPU clock is gated externally and MICFIL could be kept running.

In case a SoC SLEEP mode is requested, the decimation filter is kept running such as Normal mode and the SoC SLEEP mode is acknowledged immediately.

In case a SoC DEEPSLEEP mode is requested it is waited until the last input samples are processed to acknowledge the SoC mode and stop the decimation filter.

During this mode, VADEN will continue running if it was previously enabled (VADEN=1) during Normal mode.

You can generate DMA and IRQ requests asynchronously to wake up the CPU.

NOTE

The asynchronous DMA or interrupt requests are generated only in this mode.

50.3.6.4 DLL mode

In this mode, MICFIL stops after the remaining data processing is complete. MICFIL is in DLL mode when:

- The chip is in SLEEP and DEEPSLEEP modes and CTRL_1[DOZEN] = 1.
- The chip is in Doze mode and CTRL_1[DOZEN] = 1.
- CTRL_1[MDIS] = 1.

If you request DLL mode when MICFIL is running, MICFIL stops after completion to process the remaining data.

You can write only to CTRL_1 and CTRL_2, with the exception of writes to CTRL_1[DBG] and CTRL_1[SRES], which are ignored. However, you can still read these fields.

50.3.7 Clocking

Table 338. MICFIL clocks

Clock name	Description
MICFIL_CLK, MICFIL_CLK_S	Clock that controls the registers
MICFIL_CLK_APP	Application clock that operates the filter

Besides the above clock signals, MICFIL generates PDM_CLK for PDM microphones based on MICFIL_CLK_APP frequency and the value in CLKDIV.

50.3.8 Interrupts

If CTRL_1[DISEL] = 10b, an interrupt request indicates that filtering results are stored in the FIFOs to be read by the CPU. In this case, you can read the data in DATACHn[DATA], and STAT[CHnF] becomes 1 for the enabled channels when the CHn FIFO surpasses the watermark level.

When MICFIL is in Low-Power mode, the CPU clock is disabled, but the MICFIL clock remains enabled. In this case, an asynchronous interrupt is delivered to the system. This wakes up the CPU clock, which returns to Normal mode. The system could read the FIFO data, after which it must clear STAT[CHnF] to get a new interrupt request.

An error interrupt request can also be generated when an exceptional condition happens, such as an overflow or underflow in a channel output, or an overflow or underflow of a FIFO buffer, or the MICFIL clock rate is not the minimum value required to the MICFIL working (see Minimum required CLKDIV).

CTRL_1[ERREN] enables error interruption and the respective status flag signalizes the type of error:

- OUTOVFn and OUTUNFn for channel outputs errors
- FIFO_STAT[FIFOOVFn] and FIFO_STAT[FIFOUNDn] for FIFO errors
- LOWFREQF for MICFIL clock rate

50.3.9 DMA

If CTRL_1[DISEL] = 1, the FIFO stored samples are read through DMA transactions. Then if there is at least one enabled channel and all FIFOs of all enabled channels reach their watermark levels, the output interface makes a request for a DMA transaction.

If CTRL_1[PDMIEN] = 1 and the filters start to operate, you must wait until the filter results stabilize and no filter results are stored in the FIFOs. Therefore, the first DMA request takes longer to occur than the ones that follow.

STAT[CHnF] becomes 1 after a DMA transaction is complete.

If MICFIL is in Low-Power mode and the watermark level is surpassed, an asynchronous DMA request is delivered to the system. Then, the system wakes up the CPU clock, returning to Normal mode to read the FIFO data.

50.4 External signals

Table 339. PDM microphone interface external signals

Signal	Description	Direction
PDM_DATAn	Data bitstream received from the n^{th} PDM microphone pair throughout the input interface (see Input interface)	I
PDM_CLK	Clock that is delivered to the PDM microphones from the clock generator (see Time generator)	O

NOTE

MICFIL does not provide metastability protection or filtering for input data. You must generate the data using the provided PDM_CLK.

50.5 Initialization

50.5.1 Procedure without HWVAD

Perform this procedure for MICFIL initialization without enabling voice activity detector, i.e. (by using only decimation filters):

1. Program CTRL_2[CLKDIV], CTRL_1[CHnEN], FIFO_CTRL[FIFOWMK], CTRL_1[DISEL], CTRL_1[ERREN], and so on, as required. For more details see MICFIL_CLK_ROOT rate and CLKDIV calculation examples.
2. Start the module operation by writing 1 to CTRL_1[PDMIEN].

The module is ready to receive data from the input ports of the enabled channels.

50.5.2 Initialization procedure with HWVAD

In this section is described the initialization procedures to initialize MICFIL with the voice activity detector.

The voice activity detector can work in two modes: Energy-based mode or Envelope-based mode. Optionally, a Zero-Crossing Detector can be enabled to improve the voice detection.

The Energy-based method separates the audio signal in frames and estimates the energy of these frames to detect voice activity. In the other hand, the Envelope-based method calculates the audio signal envelope to detect voice.

Both methods have the same detection performance, but the Energy-based is recommended for low power applications where is expected that the chip is asleep for long periods. Envelope-based detection is recommended for applications where the device require to wake faster and it is not asleep for long periods.

The Zero-Crossing Detector is recommended to improve detection with fricatives phonemes (/e.g. /f/ or /s/) in noisy environments.

50.5.2.1 HWVAD in Energy-based mode

The sequence of steps to initialize the voice activity detector in Energy-based mode is as follows:

1. Program the control bitfields as desired for your application (CLKDIV, CHnEN, FIFOWMK, DISEL, ERREN etc.). For more details see MICFIL_CLK_ROOT rate and CLKDIV calculation examples.
2. Configure general HWVAD parameters:
    a. CIC oversampling rate (VADCICOSR).
    b. Configure the voice source channel (VADCHSEL).
    c. Configure the internal initialization timer (VADINITT) if required. More details in Initialization time.
    d. Assert VADEN.
3. Configure input, noise and signal gains according to the environmental noise conditions:
    a. Configure the input gain (VADINPGAIN).
    b. Configure the signal gain (VADSGAIN).
    c. Configure the noise gain (VADNGAIN).
4. Enable the high-pass filter if required (VADHPF).
5. Configure the VADNFILADJ field according to environmental noise conditions (VADNFILADJ = 16 is recommended).
6. Put HWVAD in energy-based mode (default values):
    a. Keep the VADFRENDIS field cleared.
    b. Keep the VADPREFEN field cleared.

c. Keep the VADSFILEN field cleared.

d. Keep the VADSMAXEN field cleared.

e. Keep the VADNFILAUTO field asserted.

f. Keep the VADNMINEN field cleared.

g. Keep the VADNDECEN field cleared.

h. Keep the VADNOREN field cleared.

7. Initialize the Zero-Crossing Detector (Zero-crossing detector initialization) if required.
8. Enable interrupts if required (VADIE and VADERIE).
9. Start the overall module asserting the PDMIEN field.
10. The module is ready to receive data from the input ports of the enabled channels and to trigger interrupts when voice is detected.

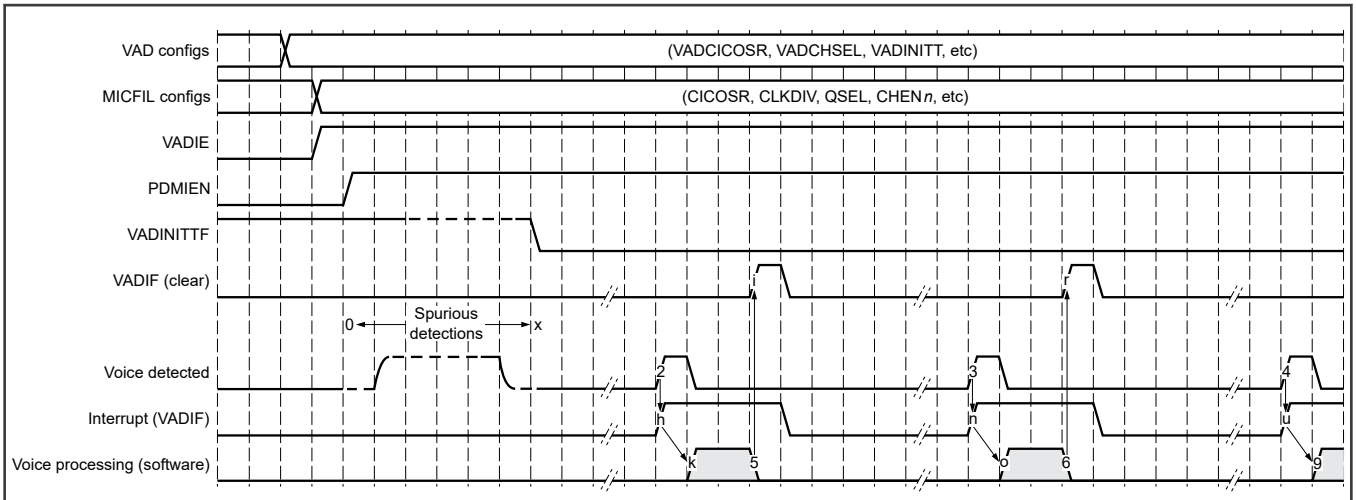


Figure 301. Energy-based initialization sequence

50.5.2.2 HWVAD in Envelope-based mode

The sequence of steps to initialize MICIL with the voice activity detector in Envelope-based mode is as follows:

1. Program the control bitfields as desired for your application (CLKDIV, CHnEN, FIFOWMK, DISEL, ERREN etc.). For more details see MICFIL_CLK_ROOT rate and CLKDIV calculation examples.
2. Configure general HWVAD parameters:
   a. CIC oversampling rate (VADCICOSR).
   b. Configure the voice source channel (VADCHSEL).
   c. Configure the internal initialization timer (VADINITT) if required. More details in Initialization time.
   d. Assert the VADEN field.
3. Configure input, noise and signal gains according the environment noise conditions:
   a. Configure the input gain (VADINPGAIN).
   b. Configure the signal gain (VADSGAIN).
   c. Configure the noise gain (VADNGAIN).
4. Enable the high-pass filter if required (VADHPF).

5. Put HWVAD in envelope-based mode:
    a. Keep VADNFILADJ field cleared.
    b. Assert the VADFRENDISfield.
    c. Assert the VADPREFEN field.
    d. Assert the VADSFILEN field.
    e. Assert the VADSMAXEN field.
    f. Clear the VADNFILAUTO field.
    g. Assert the VADNMINEN field.
    h. Assert the VADNDECEN field.
    i. Assert the VADNOREN field.
6. Initialize the Zero-Crossing Detector (Zero-crossing detector initialization) if required.
7. Start the overall module asserting the PDMIEN bit.
8. Wait for VADINITF to be cleared.
9. Write VADST10 field three times.
10. Enable interrupts if required (VADIE and VADERIE).
11. The module is ready to receive data from the input ports of the enabled channels and to trigger interrupts when voice is detected.

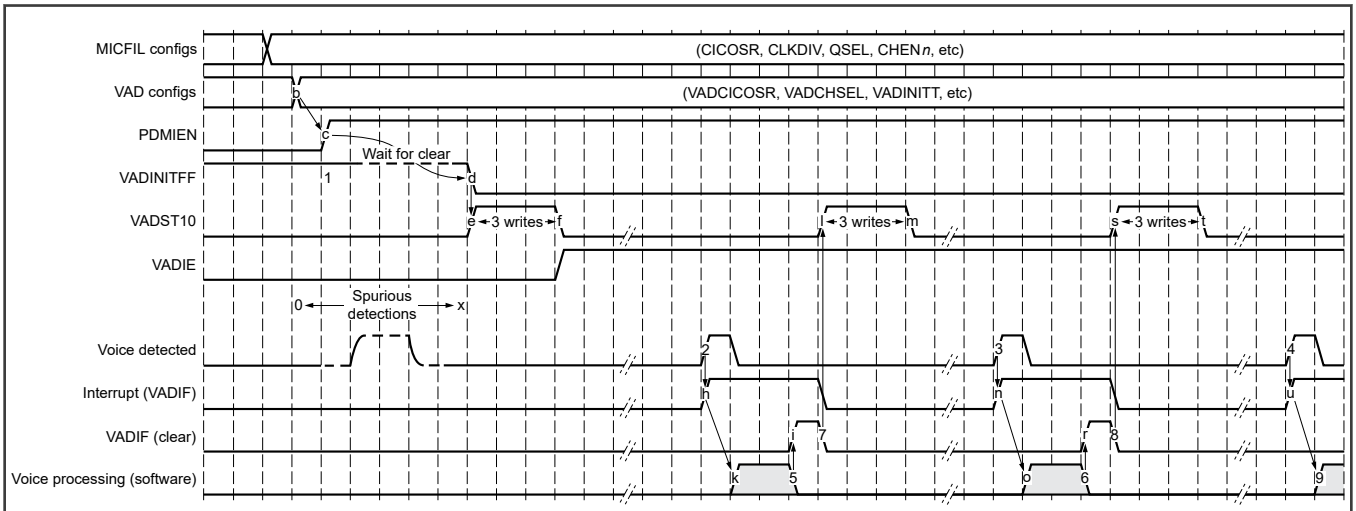


Figure 302. Envelope-based initialization sequence

50.5.2.3 Zero-crossing detector initialization

The sequence of steps to initialize the Zero-Crossing Detector in Automatic mode is as follows:

1. Configure VADZCDADJ field.
2. Configure an initial threshold in VADZCDTH field according to the environmental noise conditions if required.
3. Configure VADZCDAND field.
4. Keep VADZCDAUTO field asserted.
5. Assert VADZCDEN field.

The sequence of steps to initialize the Zero-Crossing Detector in Manual mode is as follows:

1. Configure a threshold in VADZCDTH field according to the environmental noise conditions if required.
2. Configure VADZCDAND field.
3. Clear VADZCDAUTO field.
4. Assert VADZCDEN field.

50.6 Application information

50.6.1 Reconfiguration procedure

Perform this procedure to reconfigure MICFIL:

1. Write 0 to CTRL_1[PDMIEN].
2. Wait for STAT[BSY_FIL] to clear.
3. Run a soft-reset cycle.
4. Run an initialization procedure as described in Procedure without HWVAD or Initialization procedure with HWVAD.

50.7 MICFIL register descriptions

This section provides the memory map and detailed description of all MICFIL registers. Any access to reserved areas can issue a slave bus error indication.

You must implement the registers allocated in this memory map for up to 4 pairs of microphones, indicating that 8 channels are available.

50.7.1 MICFIL memory map

PDM base address: 42BE_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	MICFIL Control 1 (CTRL_1)	32	RW	0000_0000h
4h	MICFIL Control 2 (CTRL_2)	32	RW	0000_0000h
8h	MICFIL Status (STAT)	32	RW	0000_0000h
10h	MICFIL FIFO Control (FIFO_CTRL)	32	RW	0000_0007h
14h	MICFIL FIFO Status (FIFO_STAT)	32	RW	0000_0000h
24h - 40h	MICFIL Output Result (DATACH0 - DATACH7)	32	R	0000_0000h
64h	MICFIL DC Remover Control (DC_CTRL)	32	RW	0000_0000h
74h	MICFIL Range Control (RANGE_CTRL)	32	RW	0000_0000h
7Ch	MICFIL Range Status (RANGE_STAT)	32	RW	0000_0000h
90h	Voice Activity Detector 0 Control (VAD0_CTRL_1)	32	RW	0000_0000h
94h	Voice Activity Detector 0 Control (VAD0_CTRL_2)	32	RW	000A_0000h
98h	Voice Activity Detector 0 Status (VAD0_STAT)	32	RW	8000_0000h
9Ch	Voice Activity Detector 0 Signal Configuration (VAD0_SCONFIG)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A0h	Voice Activity Detector 0 Noise Configuration (VAD0_NCONFIG)	32	RW	8000_0000h
A4h	Voice Activity Detector 0 Noise Data (VAD0_NDATA)	32	R	0000_0000h
A8h	Voice Activity Detector 0 Zero-Crossing Detector (VAD0_ZCD)	32	RW	0000_0004h

50.7.2 MICFIL Control 1 (CTRL_1)

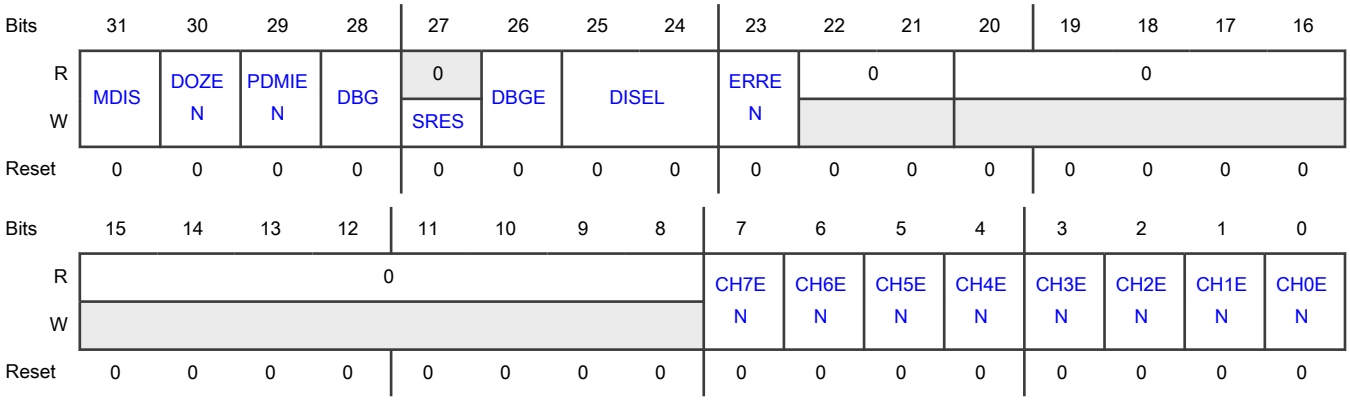
Offset

Register	Offset
CTRL_1	0h

Function

Contains control fields for MICFIL.

Diagram



Fields

Field	Function
31 MDIS	Module Disable Places MICFIL in DLL mode (see DLL mode). The generated PDM_CLK stops, and consequently, the microphone generates no data. **NOTE** When this field = 1, only writes to MICFIL Control 1 (CTRL_1) and MICFIL Control 2 (CTRL_2) are allowed with the exception of writes to CTRL_1[DBG] and CTRL_1[SRES]. A slave bus error indication is issued if you write to any other register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Normal mode 1b - DLL mode
30 DOZEN	Doze Enable Enables Doze mode. When the chip is in Doze mode and this field is active, MICFIL enters Stop mode (see DLL mode). 0b - Disables 1b - Enables
29 PDMIEN	MICFIL Enable Starts the module operation. Only the channels that CTRL_1[CHnEN] enables or VAD enabled by VADEN start operating. When this field becomes 0, MICFIL stops after it completes the remaining filter processing. **NOTE** You must enable this field as the last step when initializing the module (see Initialization). 0b - Stops MICFIL operation 1b - Starts MICFIL operation
28 DBG	Debug Mode Specifies the mode (Debug or Normal) that MICFIL operates in. MICFIL operates in Debug mode, depending on the value of CTRL_1[DBGE]. When Debug mode is disabled, MICFIL operates in Normal mode. See Debug mode for more on Debug mode. **NOTE** You can write to this field only when CTRL_1[MDIS] = 0. 0b - Normal 1b - Debug
27 SRES	Software Reset Provides the CPU with the capability to initialize MICFIL through the slave-bus interface. The field always reads 0, and is effective only when CTRL_1[MDIS] = 0. It is a self-clearing field. **NOTE** You must write 1 to this field only when STAT[BSY_FIL] = 0. You can use this field to clear data in all filters. It resets the following blocks and registers: • All internal buffers in decimation filters and voice activity detectors • All FIFOs • STAT[CHnF] and LOWFREQF flag and MICFIL FIFO Status (FIFO_STAT)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Time generator 0b - No action 1b - Software reset
26 DBGE	Module Enable in Debug Enables operation in Debug mode (see Debug mode). 0b - Disables after completing the current frame 1b - Enables operation
25-24 DISEL	DMA Interrupt Selection Specifies the type of interrupt requests (DMA or filter) that MICFIL performs when the number of FIFO elements surpasses the configured watermark. Note that requests are only generated from the channels for which CTRL_1[CHnEN] = 1. **NOTE** You must not change the value of this field when STAT[BSY_FIL] = 1. 00b - Disables DMA and interrupt requests 01b - Enables DMA requests 10b - Enables interrupt requests 11b - Reserved
23 ERREN	Error Interruption Enable Enables error interrupts (exceptions). The following conditions can lead to errors: • An overflow or underflow in the FIFOs (indicated by FIFO_STAT[FIFOOVFn] or FIFO_STAT[FIFOUNDn], respectively) • An overflow or underflow in the decimations filters (indicated by RANGE_STAT[RANGEOVFn] or RANGE_STAT[RANGEUNFn], respectively) • not enough MICFIL internal clock frequency (LOWFREQF bit). **NOTE** You must not change the value of this field when STAT[BSY_FIL] = 1. 0b - Disables 1b - Enables
22-21 —	Reserved
20-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 —	Reserved
7-0 CHnEN	Channel n Enable Enables the decimation filter channel n (see Initialization for more information). **NOTE** You must not change the value of this field when STAT[BSY_FIL] = 1.

50.7.3 MICFIL Control 2 (CTRL_2)

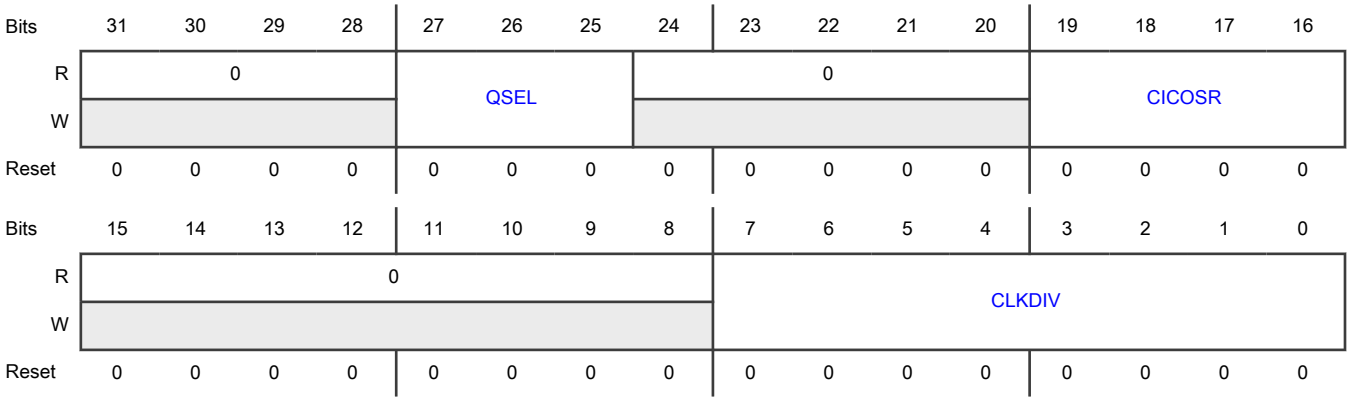
Offset

Register	Offset
CTRL_2	4h

Function

Contains control fields for MICFIL.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-25 QSEL	Quality Mode Defines the actual quality mode (see Quality modes) of the decimation filter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE You must not change the value of this field when STAT[BSY_FIL] = 1. 000b - Medium-Quality mode 001b - High-Quality mode 100b - Very-Low-Quality 2 mode 101b - Very-Low-Quality 1 mode 110b - Very-Low-Quality 0 mode 111b - Low-Quality mode
24-20 —	Reserved
19-16 CICOSR	CIC Decimation Rate Helps you configure the CIC filter decimation rate (see Equation 8 for more information). **NOTE** You must not change the value of this field when STAT[BSY_FIL] = 1. 0000b - CIC oversampling rate = 0 0001b - CIC oversampling rate = 1 0010b-1110b - ... 1111b - CIC oversampling rate = 15
15-8 —	Reserved
7-0 CLKDIV	Clock Divider Configures the internal clock divider value (see Clock divider for more information) using the following equation: $\text{PDM\_CLK rate} = \frac{\text{MICFIL\_CLK\_ROOT}}{2 \times \text{floor} \times (\text{K} \times \text{CLKDIV})}$ **NOTE** • You must not change the value of this field when STAT[BSY_FIL] = 1. • CLKDIV = 0 has the same effect as CLKDIV = 1. In Medium Quality mode, CLKDIV = 0 or 1 has the same effect as CLKDIV = 2. In High Quality mode, CLKDIV = 0 or 1 is not allowed. 0000_0000b - Internal clock divider value = 0 0000_0001b - Internal clock divider value = 1 0000_0010b-1111_1110b - ...

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1111_1111b - Internal clock divider value = 255

50.7.4 MICFIL Status (STAT)

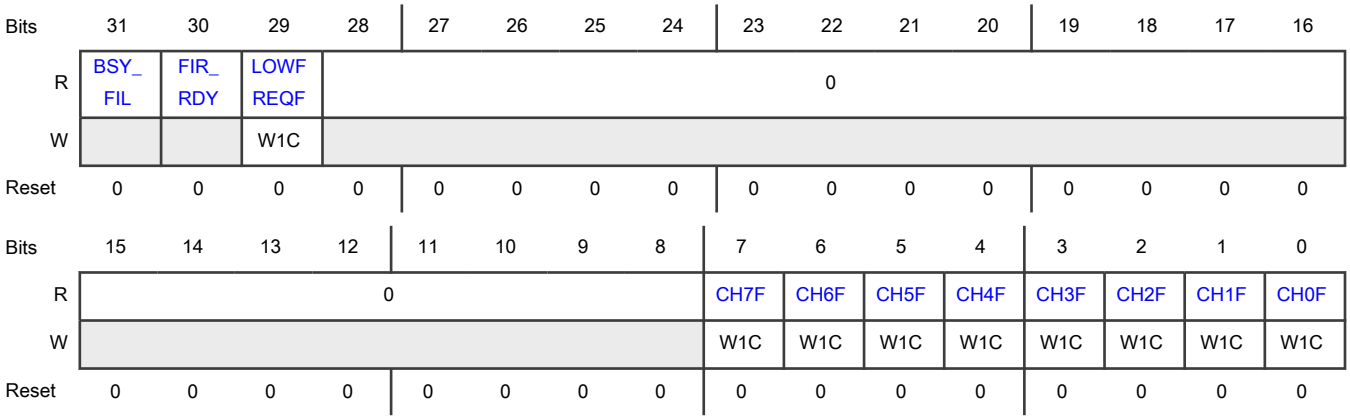
Offset

Register	Offset
STAT	8h

Function

Contains status flags for MICFIL.

Diagram



Fields

Field	Function
31 BSY_FIL	Busy Flag Signalizes when at least a decimation filter channel or at least a HWVAD is running and the PDM_CLK is being generated. You could use this field as a confirmation that all decimation filters are stopped in a transition to Debug mode (when CTRL_1[DBGE] = 0) or DLL mode. 0b - MICFIL is stopped 1b - MICFIL is running
30 FIR_RDY	Filter Data Ready Indicates whether all the decimation filter buffers are updated, and all filter data provided from now on is reliable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not reliable 1b - Reliable
29 LOWFREQF	Low Frequency Flag Indicates that the CLKDIV field value is not high enough to filter all enabled channels , and the compensation filters didn't finish their operations before the next sample arrived. See Minimum required CLKDIV. Write a logic 1 to this field to clear this flag. **NOTE** The filtering results in this condition are unpredictable. 0b - CLKDIV value is OK 1b - CLKDIV value is too low
28-16 —	Reserved
15-8 —	Reserved
7-0 CHnF	Channel n Output Data Flag Indicates whether this channel's FIFO has surpassed its watermark value and the data is available in MICFIL Output Result (DATACH0 - DATACH7). This flag contributes to generate an interrupt or DMA request if CTRL_1[DISEL] enables it. Write 1 to CTRL_1[DISEL] to clear this flag. If the DMA request is set (CTRL_1[DISEL] = 1), CHnF is set according to the watermark level . In other cases (IRQ included), CHnF is set according to the watermark level and is cleared when you write 1 to it. **NOTE** If DMA is enabled, this flag clears itself after the DMA transaction is complete. 0b - Not surpassed 1b - Surpassed

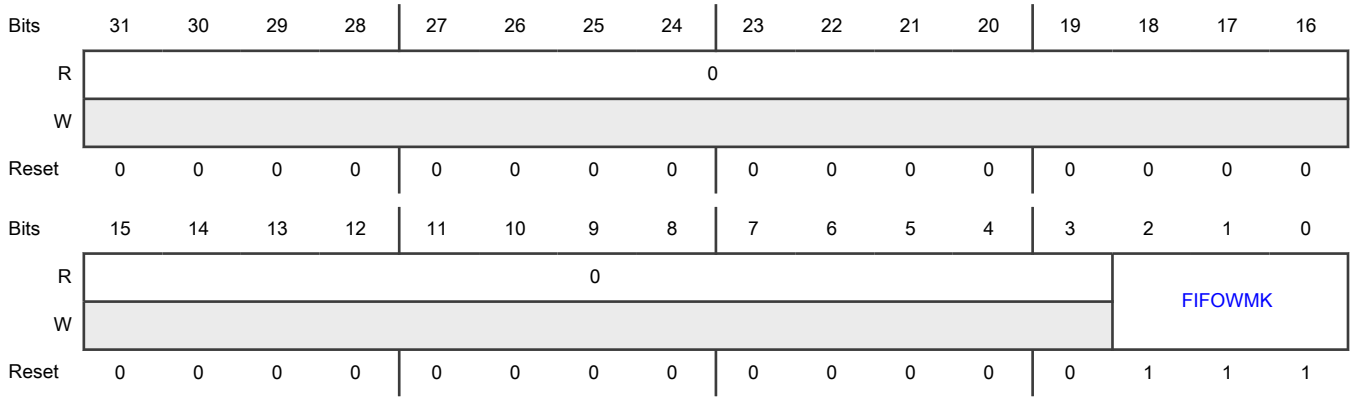
50.7.5 MICFIL FIFO Control (FIFO_CTRL)

Offset

Register	Offset
FIFO_CTRL	10h

Function
Contains MICFIL FIFO control fields.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 FIFOWMK	FIFO Watermark Control Controls the watermark of the FIFO used to set STAT[CHnF]. If the number of results in the FIFO is greater than the value of this field, STAT[CHnF] is set. A DMA request or interrupt can be also generated according to the configuration of CTRL_1[DISEL]. **NOTE** You must not change the value of this field when STAT[BSY_FIL] = 1. Before changing the watermark value, you must service all filtered data and clear channel flags.

50.7.6 MICFIL FIFO Status (FIFO_STAT)

Offset

Register	Offset
FIFO_STAT	14h

Function

Contains MICFIL FIFO status flags.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIFOUND7	FIFOUND6	FIFOUND5	FIFOUND4	FIFOUND3	FIFOUND2	FIFOUND1	FIFOUND0	FIFOOVF7	FIFOOVF6	FIFOOVF5	FIFOOVF4	FIFOOVF3	FIFOOVF2	FIFOOVF1	FIFOOVF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 FIFOUNDn	FIFO Underflow Exception Flag for Channel n Indicates an exceptional underflow condition in the FIFO. If the FIFO is empty and a read operation is requested, then this flag is set. 0b - No exception by FIFO underflow 1b - Exception by FIFO underflow
7-0 FIFOOVFn	FIFO Overflow Exception Flag for Channel n Indicates an exceptional overflow condition in the FIFO. If the FIFO is full and you receive an additional result to be written, this flag is set. 0b - No exception by FIFO overflow 1b - Exception by FIFO overflow

50.7.7 MICFIL Output Result (DATACH0 - DATACH7)

Offset

For a = 0 to 7:

Register	Offset
DATACHa	24h + (a × 4h)

Function

Contains the word value stored at the top of the channel's n FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	Channel n Data Reflects the word value stored at the top of the channel's n FIFO. The samples are integer-signed numbers in the two’s-complement format. **NOTE** The even FIFO data width is 32-bits; only the 24 more significant bits have information, and the other bits are always 0.

50.7.8 MICFIL DC Remover Control (DC_CTRL)

Offset

Register	Offset
DC_CTRL	64h

Function

Contains DC remover control fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DCCONFIG7		DCCONFIG6		DCCONFIG5		DCCONFIG4		DCCONFIG3		DCCONFIG2		DCCONFIG1		DCCONFIG0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-14: DCCONFIG7 13-12: DCCONFIG6 11-10: DCCONFIG5 9-8: DCCONFIG4 7-6: DCCONFIG3 5-4: DCCONFIG2 3-2: DCCONFIG1 1-0: DCCONFIG0	Channel n DC Remover Configuration Defines the value of the cut-off frequency of the DC remover. 00b - 21 Hz 01b - 83 Hz 10b - 152 Hz 11b - DC remover is bypassed

50.7.9 MICFIL Range Control (RANGE_CTRL)

Offset

Register	Offset
RANGE_CTRL	74h

Function
Contains range configuration information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RANGEADJ7				RANGEADJ6				RANGEADJ5				RANGEADJ4			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RANGEADJ3				RANGEADJ2				RANGEADJ1				RANGEADJ0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28: RANGEADJ7 27-24: RANGEADJ6 23-20: RANGEADJ5 19-16: RANGEADJ4 15-12: RANGEADJ3 11-8: RANGEADJ2 7-4: RANGEADJ1 3-0: RANGEADJ0	Channel n Range Adjustment Adjusts the dynamic range of the CIC filter. It is an unsigned positive value. You must configure this field within the following ranges: • If CTRL_2[QSEL] is in High-Quality and Very-Low-Quality 0 modes (HQ and VLQ0): RANGEADJn ≤ 37 - ceil($7\log_2(2OSR)$) • If CTRL_2[QSEL] is in Medium-Quality and Very-Low Quality 1 modes (MQ and VLQ1): RANGEADJn ≤ 37 - ceil($7\log_2(OSR)$) • If CTRL_2[QSEL] is in Low Quality and Very-Low-Quality 2 modes (LQ and VLQ2): RANGEADJn ≤ 36 - ceil($7\log_2(OSR)$) where OSR = 16 - CICOSR, which is the CIC oversampling rate (see Equation 11), and the ceiling function, ceil(x), maps to the least integer greater than or equal to x.

50.7.10 MICFIL Range Status (RANGE_STAT)

Offset

Register	Offset
RANGE_STAT	7Ch

Function

Contains range status information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								RANGEUN...	RANGEUN...	RANGEUN...	RANGEUN...	RANGEUN...	RANGEUN...	RANGEUN...	RANGEUN...
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RANGEOV...	RANGEOV...	RANGEOV...	RANGEOV...	RANGEOV...	RANGEOV...	RANGEOV...	RANGEOV...
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 RANGEUNFn	Channel n Range Underflow Error Flag Indicates an exceptional underflow condition in MICFIL range. **NOTE** If this error flag is asserted, channel n data is not reliable. You must adjust the value of RANGE_CTRL[RANGEADJn] until this flag is not asserted anymore. 0b - No exception by range underflow 1b - Exception by range underflow
15-8 —	Reserved
7-0 RANGEOVFn	Channel n Range Overflow Error Flag Indicates an exceptional overflow condition in MICFIL range. **NOTE** If this error flag is asserted, channel n data is not reliable. You must adjust the value of RANGE_CTRL[RANGEADJn] until this flag is not asserted anymore. 0b - No exception by range overflow 1b - Exception by range overflow

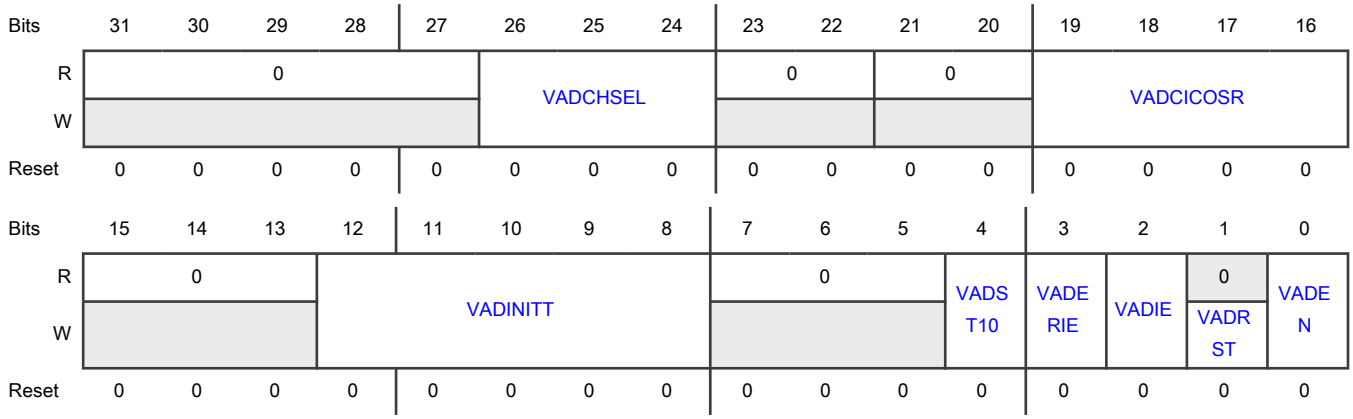
50.7.11 Voice Activity Detector 0 Control (VAD0_CTRL_1)

Offset

Register	Offset
VAD0_CTRL_1	90h

Function
Contains HWVAD configuration bits.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 VADCHSEL	Voice Activity Detector Channel Selector Selects the number of channel which the hardware voice activity detector will work. **NOTE** This field mustn't change when BSY_FIL is asserted. 000b - PDM Microphone 0 Left 001b - PDM Microphone 0 Right 010b - PDM Microphone 1 Left 011b-101b - ... 110b - PDM Microphone 3 Left 111b - PDM Microphone 3 Right
23-22 —	Reserved
21-20 —	Reserved
19-16 VADCICOSR	Voice Activity Detector CIC Oversampling Rate Defines the oversampling rate of the CIC filter. The oversampling rate is (16-CICOSR). **NOTE** This field mustn't change when BSY_FIL is asserted.
15-13	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
12-8 VADINITT	Voice Activity Detector Initialization Time Selects the number of frames to be used to initialize the voice detection. During this period the output of the voice activity detector is forced to inactive. For more information, see Initialization time. **NOTE** The initialization time will be the time taken by [VADINITT+1] frames. **NOTE** This field mustn't change when BSY_FIL is asserted. 0_0000b - VADINITT = 0 0_0001b - VADINITT = 1 0_0010b-1_1110b - ... 1_1111b - VADINITT = 31
7-5 —	Reserved
4 VADST10	Voice Activity Detector Internal Filters Initialization Initializes the signal and noise filter with pre-filter value. **NOTE** If the use of this bit is required, it should be stay pulsed by more than 2 cycles of system clock . **NOTE** There is not voice activity detection when VADST10 is asserted. 0b - Normal operation 1b - Filters initialized
3 VADERIE	Voice Activity Detector Error Interruption Enable Enables Error Interrupts in MICFIL. This error could be caused by an overflow in the input of the HWVAD and it is signalized by the VADINSATF flag. 0b - Disables 1b - Enables
2 VADIE	Voice Activity Detector Interruption Enable Enables interrupts in MICFIL when voice activity event has been detected by the Hardware Voice Activity Detector (HWVAD). See the VADIF for more information. 0b - Disables

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enables
1 VADRST	Voice Activity Detector Reset Provides the CPU with the capability to initialize the VAD through the slave-bus interface. This bit always reads as zero. This bit is self-cleared. See VAD soft-reset.
0 VADEN	Voice Activity Detector Enable Enables the use of the Hardware Voice Activity Detector. **NOTE** This field mustn't change when BSY_FIL is asserted. 0b - Disables 1b - Enables

50.7.12 Voice Activity Detector 0 Control (VAD0_CTRL_2)

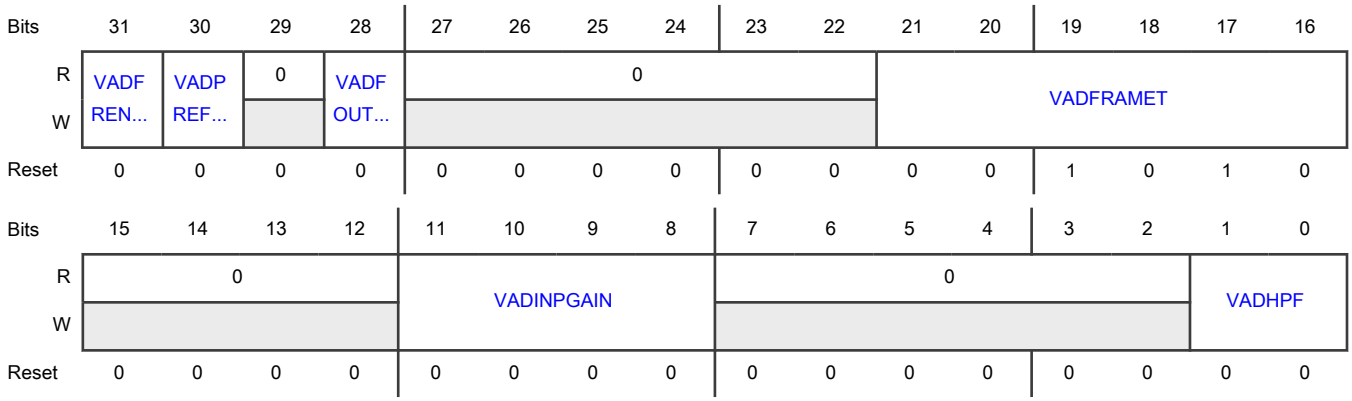
Offset

Register	Offset
VAD0_CTRL_2	94h

Function

NOTE
This register mustn't change when BSY_FIL is asserted.

Diagram



Fields

Field	Function
31 VADFRENDIS	Voice Activity Detector Frame Energy Disable Disables the calculus of energy in a frame. For more information, see Noise filter. 0b - Enables 1b - Disables
30 VADPREFEN	Voice Activity Detector Pre Filter Enable Enables a pre-filter in the input of Noise and Signal filters. For more information, see Pre-filter. 0b - Pre-filter bypassed 1b - Pre-filter enabled
29 —	Reserved
28 VADFOUTDIS	Voice Activity Detector Force Output Disable Disables the output of the Energy/Envelope-based Detector. 0b - Enables 1b - Disables
27-22 —	Reserved
21-16 VADFRAMET	Voice Activity Detector Frame Time Selects the scale value to control the frame time duration as is described in Frame time setting. **NOTE** VADFRAMET = 0 is not allowed. 00_0000b - VADFRAMET = 1 00_0001b - VADFRAMET = 2 00_0010b-11_1110b - ... 11_1111b - VADFRAMET = 63
15-12 —	Reserved
11-8 VADINPGAIN	Voice Activity Detector Input Gain Configures the gain value in the HWVAD input as a positive or negative (two's complement) number of bits to shift. 0000b - No shift 0001b - Shift 1 bit to the left

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0010b - Shift 2 bits to the left 0011b-0110b - ... 0111b - Shift 7 bits to the left 1000b - Shift 8 bits to the right 1001b - Shift 7 bits to the right 1010b-1110b - ... 1111b - Shift 1 bits to the right
7-2 —	Reserved
1-0 VADHPF	Voice Activity Detector High-Pass Filter Configures the internal high-pass filter. 00b - Filter bypassed 01b - Cut-off frequency at 1750 Hz 10b - Cut-off frequency at 215 Hz 11b - Cut-off frequency at 102 Hz

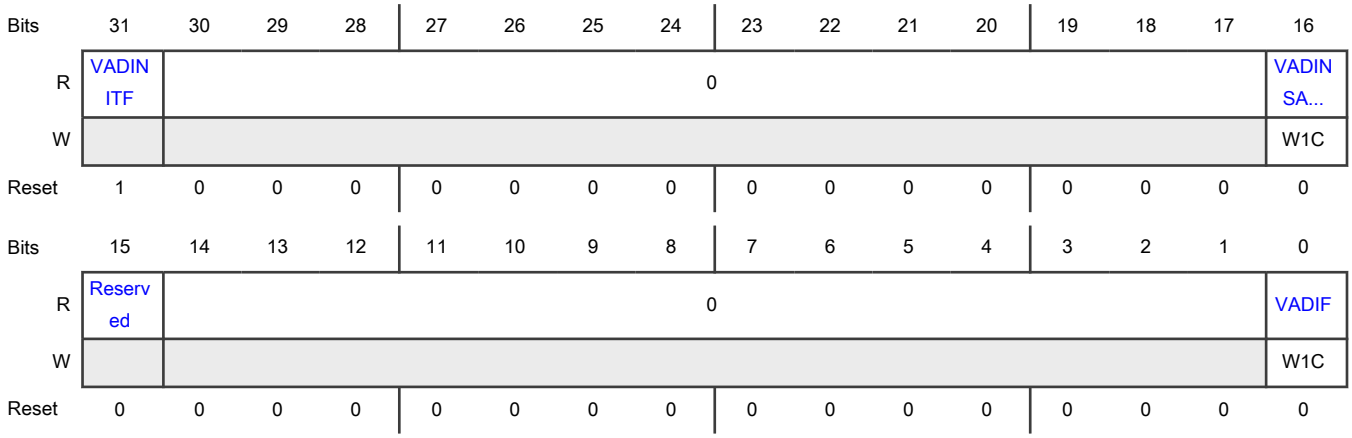
50.7.13 Voice Activity Detector 0 Status (VAD0_STAT)

Offset

Register	Offset
VAD0_STAT	98h

Function
Contains HWVAD status bits.

Diagram



Fields

Field	Function
31 VADINITF	Voice Activity Detector Initialization Flag Signalizes when the HWVAD is being initialized according to the VADINITT field value. 0b - Not being initialized 1b - Being initialized
30-17 —	Reserved
16 VADINSATF	Voice Activity Detector Input Saturation Flag Indicates an exceptional saturation condition by an overflow or underflow in the HWVAD input. 0b - No exception 1b - Exception
15 —	Reserved
14-1 —	Reserved
0 VADIF	Voice Activity Detector Interrupt Flag Indicates when voice activity has been detected by the HWVAD. Write a logic 1 to this field to clear this flag. 0b - Not detected 1b - Detected

50.7.14 Voice Activity Detector 0 Signal Configuration (VAD0_SCONFIG)

Offset

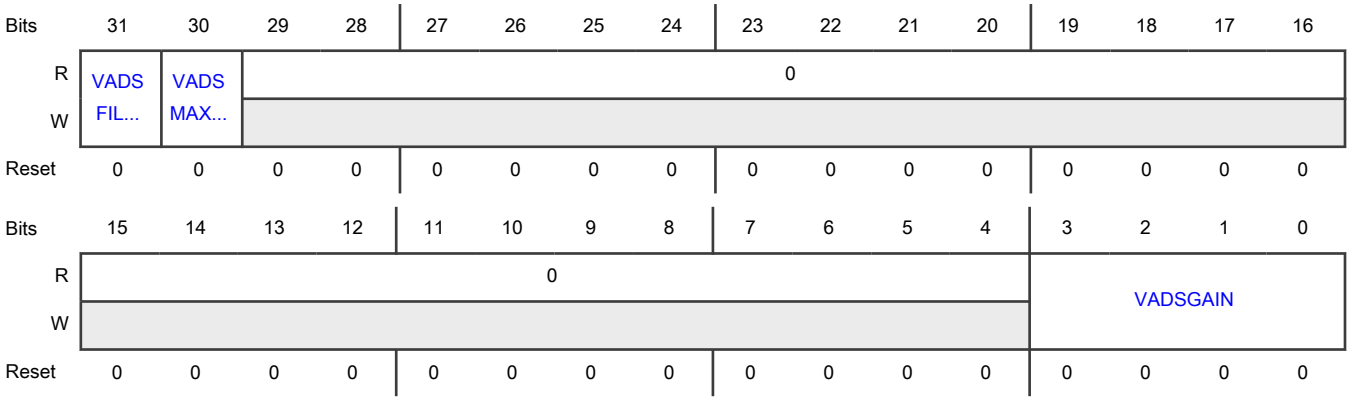
Register	Offset
VAD0_SCONFIG	9Ch

Function

NOTE

These bit-fields must be configured when MICFIL is not running.

Diagram



Fields

Field	Function
31 VADSFILEN	Voice Activity Detector Signal Filter Enable Enables the signal filter. For more information, see Signal filter. 0b - Disables 1b - Enables
30 VADSMAXEN	Voice Activity Detector Signal Maximum Enable Selects whether or not a block to calculate the maximum value between the input and output of signal filter is used as signal estimation. For more information, see Signal filter. 0b - Maximum block bypassed 1b - Maximum block enabled
29-4 —	Reserved
3-0	Voice Activity Detector Signal Gain

Table continues on the next page...

Table continued from the previous page...

Field	Function
VADSGAIN	Configures the gain value for the signal energy or envelope estimated. It is a unsigned value which is multiplied with the estimated signal data. **NOTE** VADSGAIN = 0 gain corresponds to 1. 0000b,0001b - Multiplier = 1 0010b - Multiplier = 2 0011b-1110b - ... 1111b - Multiplier = 15

50.7.15 Voice Activity Detector 0 Noise Configuration (VAD0_NCONFIG)

Offset

Register	Offset
VAD0_NCONFIG	A0h

Function

NOTE

These bit-fields must be configured when MICFIL is not running.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VADNFIL...	VADNMIN...	VADNDEC...	VADNOREN	0											
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				VADNFILADJ				0				VADNGAIN			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Voice Activity Detector Noise Filter Auto

Table continues on the next page...

Table continued from the previous page...

Field	Function
VADNFILAUTO	Selects whether the noise filter is activated automatically based on voice activity information. For more information, see Noise filter . 0b - Noise filter always enabled 1b - Noise filter enabled/disabled based on voice activity information
30 VADNMINEN	Voice Activity Detector Noise Minimum Enable Selects whether or not a block to calculate the minimum value between the input and output of noise filter is used as noise estimation. For more information, see Noise filter . 0b - Minimum block bypassed 1b - Minimum block enabled
29 VADNDECEN	Voice Activity Detector Noise Decimation Enable Selects whether or not the input of the noise filter is decimated. For more information, see Noise filter . 0b - Not decimated 1b - Decimated
28 VADNOREN	Voice Activity Detector Noise OR Enable Enables a OR logic in the output of minimum noise estimator block. For more information, see Filter result gain setting . 0b - Not decimated 1b - Decimated
27-13 —	Reserved
12-8 VADNFILADJ	Voice Activity Detector Noise Filter Adjustment Selects the adjustment value of the noise filter. For more information, see Noise filter. 0_0000b - Adjustment value = 0 0_0001b - Adjustment value = 1 0_0010b-1_1110b - ... 1_1111b - Adjustment value = 31
7-4 —	Reserved
3-0 VADNGAIN	Voice Activity Detector Noise Gain Configures the gain value for the noise energy or envelope estimated. It is an unsigned value which is multiplied with the estimated noise data.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE VADNGAIN = 0 gain corresponds to 1. 0000b,0001b - Multiplier = 1 0010b - Multiplier = 2 0011b-1110b - ... 1111b - Multiplier = 15

50.7.16 Voice Activity Detector 0 Noise Data (VAD0_NDATA)

Offset

Register	Offset
VAD0_NDATA	A4h

Function

Stores the noise envelope calculated by the HWVAD.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VADNDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 VADNDATA	Voice Activity Detector Noise Data Stores the noise energy or noise envelope calculated by the HWVAD. It can be used by software for a further voice activity detection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field value should be ignored when VADFOUTDIS = 1 or VADST10 = 1.

50.7.17 Voice Activity Detector 0 Zero-Crossing Detector (VAD0_ZCD)

Offset

Register	Offset
VAD0_ZCD	A8h

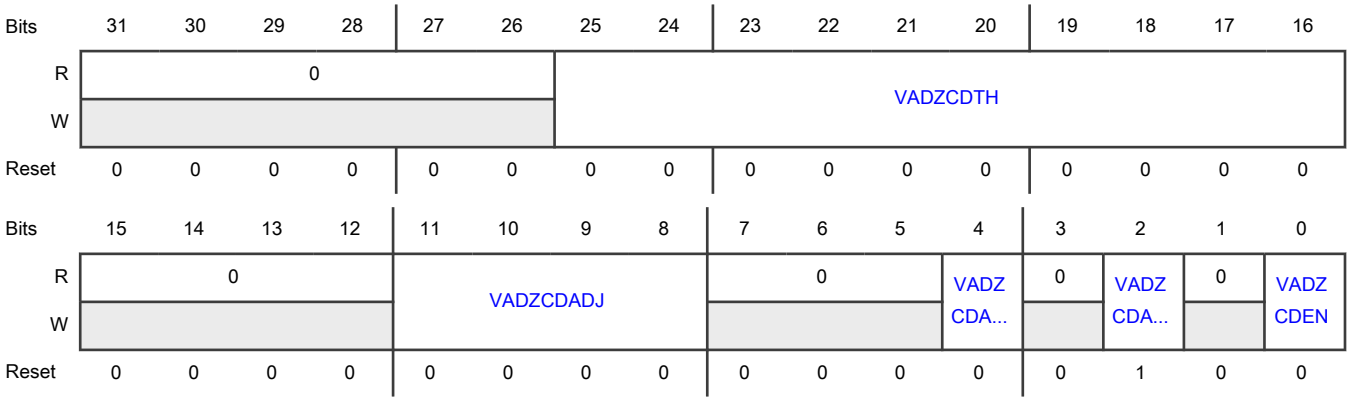
Function

Stores the zero-crossing detector configuration bits.

NOTE

These bit-fields must be configured when MICFIL is not running.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 VADZCDTH	Zero-Crossing Detector Threshold Configures the initial value taken to estimate the ZCD threshold when it is estimated automatically (VASDZCDAUTO = 1), otherwise (VADZCDAUTO = 0) this bit-field is the static value of the ZCD threshold.
15-12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11-8 VADZCDADJ	Zero-Crossing Detector Adjustment Selects the adjustment value of the threshold estimator. For more information, see Zero-crossing detector. 0000b - Adjustment value = 0 0001b - Adjustment value = 1 0010b-1110b - ... 1111b - Adjustment value = 15
7-5 —	Reserved
4 VADZCDAND	Zero-Crossing Detector AND Behavior Selects whether the Zero-Crossing detector results will be combined with the energy-based detection using AND or OR to form the voice activity detection event. 0b - OR 1b - AND
3 —	Reserved
2 VADZCDAUTO	Zero-Crossing Detector Automatic Threshold Disables the automatic estimation of the ZCD threshold. 0b - Disables 1b - Enables
1 —	Reserved
0 VADZCDEN	Zero-Crossing Detector Enable Enables the Zero-Crossing Detector (ZCD) in the Hardware Voice Activity Detector (HWVAD). 0b - Disables 1b - Enables

Chapter 51
Synchronous Audio Interface (SAI)

51.1 Chip-specific SAI information

Table 340. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

51.2 Overview

SAI provides an interface that supports full-duplex serial digital audio interfaces with frame synchronization formats such as I^2S, AC97, TDM, and Codec/DSP interfaces.

51.2.1 Block diagram

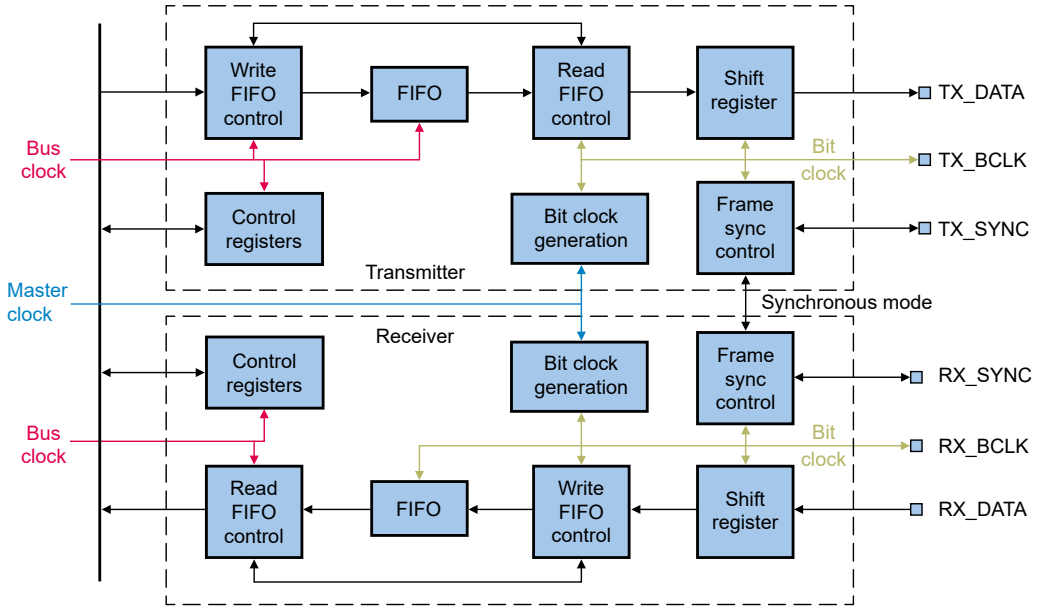


Figure 303. Block diagram

51.2.2 Features

Features can vary among chips and also among SAI modules on the same chip. See the chip-specific SAI information at the beginning of this chapter.

- Transmitter with independent bit clock and frame sync supporting 1 data line
- Receiver with independent bit clock and frame sync supporting 1 data line
- Each data line supports a maximum frame size of 32 words
- Word length of 8 to 32 bits
- Word length configured separately for the first word and remaining words in a frame
- Asynchronous 32 × 32-bit FIFO for each transmit and receive data line supports:
  - Graceful restart after FIFO error
  - Automatic restart after FIFO error without software intervention
  - 8- and 16-bit data packing into each 32-bit FIFO word

51.3 Functional description

This section provides a complete functional description of SAI.

51.3.1 Modes of operation

Module power modes include:

- Run mode
- Stop mode
- Debug mode

51.3.1.1 Run mode

In Run mode, the SAI transmitter and receiver operate normally.

51.3.1.2 Stop mode

There are two situations in which the SAI transmitter or receiver can be configured to continue operating in Stop mode:

- The SAI transmitter or receiver is using an externally generated bit clock.
- The SAI transmitter or receiver is using an audio master clock that operates when the chip is in Stop mode.

Write 1 to TCSR[STOPE] to configure the transmitter to run in Stop mode. Write 1 to RCSR[STOPE] to configure the receiver to run in Stop mode.

The SAI transmitter and receiver can generate an asynchronous interrupt to wake the CPU from Stop mode.

When the chip enters Stop mode, if TCSR[STOPE] = 0, the transmitter is disabled after completing the current transmit frame. Similarly, when the chip enters Stop mode, if RCSR[STOPE] = 0, the receiver is disabled after completing the current receive frame. Entry into Stop mode is blocked (not acknowledged) while waiting for the transmitter and receiver to be disabled at the end of the current frame.

51.3.1.3 Debug mode

You can configure either or both the SAI transmitter and receiver to continue operating in Debug mode.

- Write 1 to TCSR[DBGE] to configure the transmitter to run in Debug mode.
- Write 1 to RCSR[DBGE] to configure the receiver to run in Debug mode.

When TCSR[DBGE] = 0 and RCSR[DBGE] = 0 and the chip enters Debug mode, SAI is disabled after completing the current transmit or receive frame. Debug mode does not affect the transmitter and receiver bit clocks.

51.3.2 Synchronous modes

The SAI transmitter and receiver can operate synchronously to each other. You can configure the SAI transmitter and receiver to operate with synchronous bit clock and frame sync (see TCR2[SYNC] and RCR2[SYNC]).

If both the transmitter and receiver use the transmitter bit clock and frame sync:

- Configure the transmitter for asynchronous operation and the receiver for synchronous operation.
- Enable the receiver in Synchronous mode only after enabling both the transmitter and receiver.
- Enable the transmitter last and disable the transmitter first.

If both the transmitter and receiver use the receiver bit clock and frame sync:

- Configure the receiver for asynchronous operation and the transmitter for synchronous operation.
- Enable the transmitter in Synchronous mode only after enabling both the receiver and transmitter.
- Enable the receiver last and disable the receiver first.

When operating in Synchronous mode, the transmitter and receiver share only the bit clock, frame sync, and transmitter/receiver enable. Otherwise, the transmitter and receiver operate independently, although the configuration register settings must be consistent across both the transmitter and receiver.

51.3.3 Frame sync configuration

When enabled, SAI continuously transmits and receives frames of data. Each frame consists of a fixed number of words and each word consists of a fixed number of bits. Within each frame, you can mask any word, causing the receiver to ignore that word and the transmitter to 3-state during that word.

The frame sync signal indicates the start of a frame. A valid frame sync requires a rising edge (if active high) or falling edge (if active low) to be detected. The transmitter or receiver cannot be busy with a previous frame. A valid frame sync is ignored (Slave mode) or not generated (Master mode) for the first four bit-clock cycles after enabling the transmitter or receiver.

You can configure the transmitter and receiver frame sync independently with any of the following options:

- Can be generated externally or internally.
- Can be active high or active low.
- Can assert with the first bit in frame or assert one bit early.
- Can assert for a duration of between one bit-clock cycle and the first word length.
- Frame length can be from 1 to 32 words.
- Word length of 8 to 32 bits
  - First word length and remaining word lengths can be configured separately.
- Transmit/Receive words MSB first or LSB first

These configuration options cannot be changed after enabling the SAI transmitter or receiver.

51.3.4 Data FIFO

Each transmit and receive channel includes a 32 × 32-bit FIFO. Use the Transmit (TDR*n*) and Receive (RDR*n*) Data registers to access FIFO data.

51.3.4.1 Data alignment

Data in the FIFO can be aligned anywhere within the 32-bit-wide register via the First Bit Shifted configuration field. This field selects the bit index (between 31 and 0) of the first bit shifted.

Examples of supported data alignment and the required First Bit Shifted configuration are shown in Figure 304 for LSB-first configurations and Figure 305 for MSB-first configurations.

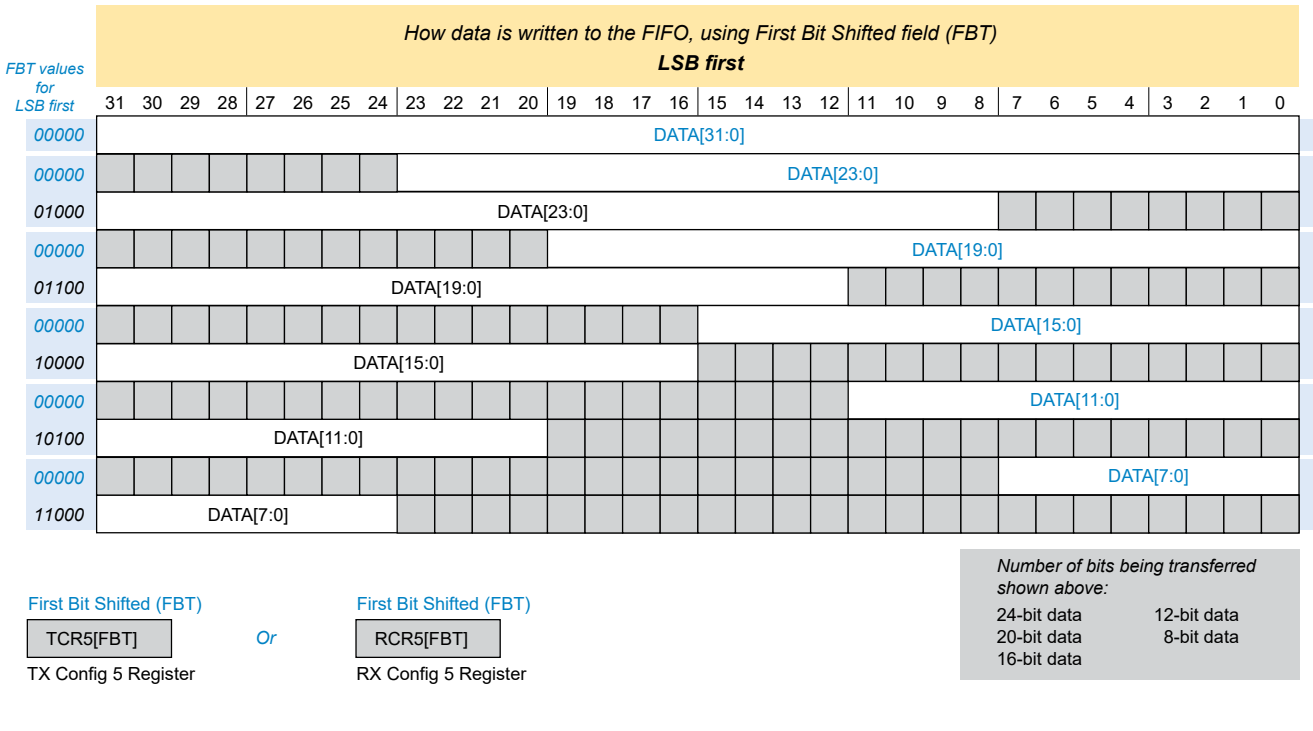


Figure 304. SAI first bit shifted, LSB-first

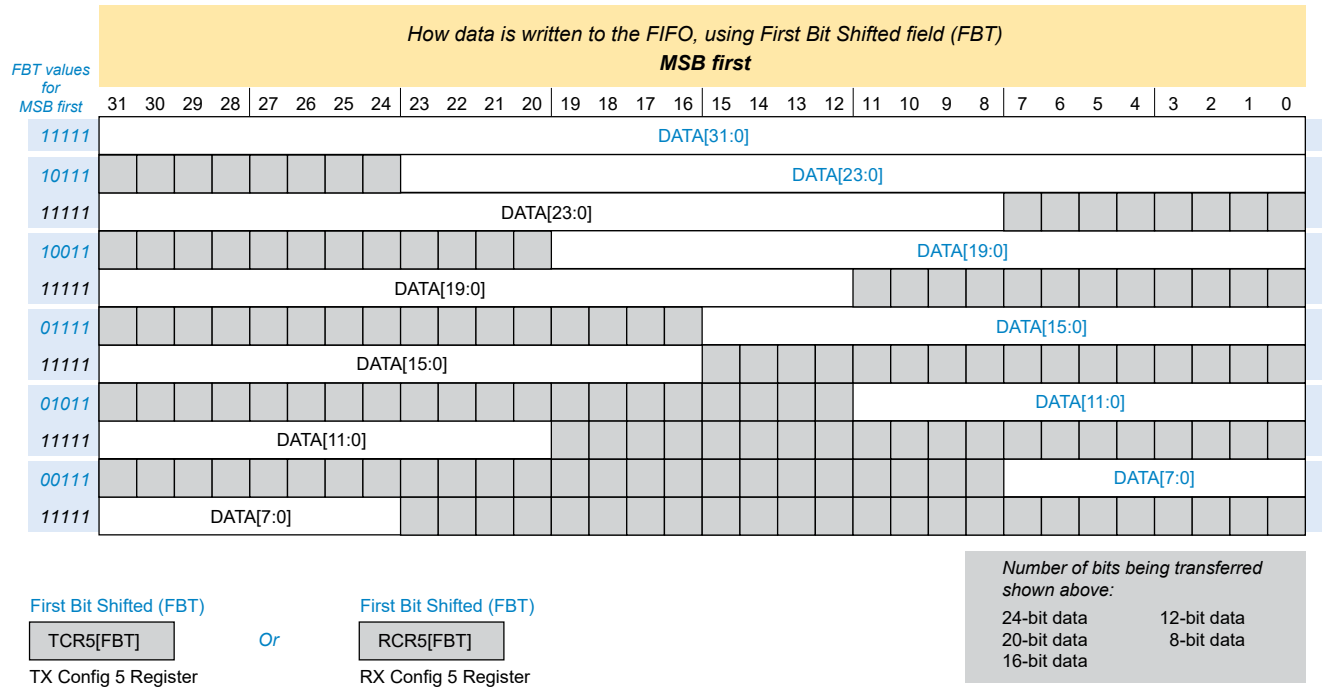


Figure 305. SAI first bit shifted, MSB-first

51.3.4.2 FIFO pointers

When writing to a Transmit Data register (TDR*n*), the Write FIFO Pointer (WFP) of the corresponding Transmit FIFO register (TFR*n*) increments after each valid write. SAI supports 8-, 16-, and 32-bit writes to TDR*n* and the FIFO pointer increments after each individual write. Only use 8-bit writes when transmitting up to 8-bit data; only use 16-bit writes when transmitting up to 16-bit data.

- If the transmit FIFO is full, writes to TDR*n* are ignored.
- If the transmit FIFO is empty, write to TDR*n* at least three bit-clock cycles before the start of the next unmasked word. This write avoids a FIFO underrun.
- Before enabling the transmitter, initialize the transmit FIFO with data. The transmitter starts a new frame after enabling the transmitter. If no data is in the FIFO, the transmitter immediately generates an error.

When reading a Receive Data register (RDR*n*), the Read FIFO Pointer (RFP) of the corresponding Receive FIFO register (RFR*n*) increments after each valid read. SAI supports 8-, 16-, and 32-bit reads from RDR*n* and the FIFO pointer increments after each individual read. Only use 8-bit reads when receiving up to 8-bit data; only use 16-bit reads when receiving up to 16-bit data.

- If the receive FIFO is empty, reads to RDR*n* are ignored.
- If the receive FIFO is full, read RDR*n* at least three bit-clock cycles before the end of an unmasked word. This read avoids a FIFO overrun.

51.3.4.3 FIFO packing

Using FIFO packing, you can store multiple 8- or 16-bit data words in one 32-bit FIFO word for the transmitter or receiver. You could emulate this feature by adjusting the number of bits per word and number of words per frame. FIFO packing, however, does not require even multiples of words per frame, and it fully supports word masking.

When FIFO packing is enabled, the FIFO pointers only increment when the full 32-bit FIFO word has been written (transmit) or read (receive). In this way, FIFO packing supports scenarios where different words within each frame are stored in different areas of memory.

Using 16-bit FIFO packing for transmitting, the transmit shift register loads at the start of each frame and after every second unmasked transmit word. The first transmitted word is taken from 16-bit word at byte offset 0h. (The first bit is selected by TCR5[FBT] and must be configured within this 16-bit word.) The second transmitted word is taken from the 16-bit word at byte offset 2h (The first bit is selected by TCR5[FBT][3:0].) The transmitter transmits logic zeroes until the start of the next word after the 16-bit word has been transmitted.

Using 16-bit FIFO packing for receiving, the receive shift register is stored after every second unmasked received word. If there is an odd number of unmasked-received words in each frame, the receive register is also stored and at the end of each frame. The first received word is stored in the 16-bit word at byte offset 0h. (The first bit is selected by RCR5[FBT] and must be configured within this 16-bit word.) The second received word is stored in the 16-bit word at byte offset 2h. (The first bit is selected by RCR5[FBT][3:0].) The receiver ignores received data until the start of the next word after the 16-bit word has been received.

8-bit FIFO packing is similar to 16-bit packing, except four words are loaded or stored into each 32-bit FIFO word. The first word is stored in byte offset 0h, the second word in byte offset 1h, and so on. TCR5[FBT] and/or RCR5[FBT] must be configured within byte offset 0h.

51.3.5 Word mask register

The SAI transmitter and receiver each have a word mask register (Transmit Mask (TMR) and Receive Mask (RMR)) that you can use to mask any word in the frame. The word mask register is double buffered. You can update it before the end of each frame to mask a particular word in the next frame.

TMR causes the Transmit Data pin to be 3-stated for the length of each selected word and the transmit FIFO is not read for masked words.

RMR causes the received data for each selected word to be discarded and not written to the receive FIFO.

51.3.6 Clocking

SAI clocks include:

- Audio master clock
- Bit clock
- Bus clock

51.3.6.1 Audio master clock

The audio master clock generates the bit clock when the receiver or transmitter is configured for an internally generated bit clock. The transmitter and receiver can independently select between the bus clock and up to three audio master clocks to generate the bit clock.

The audio master clock generation and selection is chip-specific. See chip-specific clocking information about how the audio master clocks are generated.

51.3.6.2 Bit clock

The SAI transmitter and receiver support asynchronous free-running bit clocks that can be generated internally from an audio master clock or supplied externally. There is also the option for synchronous bit clock and frame sync operation between the receiver and transmitter or between multiple SAI peripherals.

- If both transmitter and receiver are configured for asynchronous operation, the transmitter and receiver each use their own bit clock and frame sync.
- If the transmitter is configured for asynchronous operation and the receiver is configured for synchronous operation, both use the transmitter bit clock and frame sync.
- If the receiver is configured for asynchronous operation and the transmitter is configured for synchronous operation, both use the receiver bit clock and frame sync.

The software configures synchronous or asynchronous operation, and that choice selects the bit clock and frame sync used.

Externally generated bit clocks must be:

- Enabled before the SAI transmitter or receiver is enabled.
- Disabled after the SAI transmitter or receiver is disabled and completes its current frames.

In asynchronous operation, an SAI transmitter or receiver may use a bit clock externally generated by an SAI that is disabled in Stop mode. In this case, software should disable the transmitter or receiver before entering Stop mode. This issue does not apply when the transmitter or receiver is in synchronous operation because all synchronous SAI modules are enabled and disabled simultaneously.

51.3.6.3 Bus clock

The bus clock is used by the control and configuration registers and to generate synchronous interrupts and DMA requests.

NOTE

Although no minimum bus clock frequency is specified, the frequency must be fast enough (relative to the bit clock frequency) to serve the FIFOs. This requirement must be met without generating a transmitter FIFO underrun or receiver FIFO overflow condition.

51.3.7 Reset

SAI is asynchronously reset on system reset. SAI has a Software reset and a FIFO reset.

51.3.7.1 Software reset

The SAI transmitter includes a software reset that resets all transmitter internal logic, including the bit clock generation, status flags, and FIFO pointers. The SAI receiver includes a software reset that resets all receiver internal logic, including the bit clock generation, status flags, and FIFO pointers.

These software resets do not reset the configuration registers. The software resets remain asserted until cleared by software.

51.3.7.2 FIFO reset

The SAI transmitter includes a FIFO reset that synchronizes the FIFO write pointer to the value of the FIFO read pointer. This FIFO reset empties the FIFO contents and is used after TCSR[FEF] becomes 1, and before the FIFO is reinitialized and TCSR[FEF] becomes 0. The FIFO reset is asserted for one cycle only.

The SAI receiver includes a FIFO reset that synchronizes the FIFO read pointer to the value of the FIFO write pointer. This FIFO reset empties the FIFO contents. Use after the RCSR[FEF] becomes 1 and any remaining data has been read from the FIFO, and before RCSR[FEF] becomes 0. The FIFO reset is asserted for one cycle only.

51.3.8 Interrupts and DMA requests

In SAI, the transmitter and receiver generate separate interrupts and separate DMA requests, but use the same status flags. SAI generates asynchronous versions of the transmitter and receiver interrupts to wake up the CPU from Stop mode. Asynchronous interrupts are only generated when the system clock is gated but the corresponding BCLK is active.

51.3.8.1 FIFO request flag

The transmit FIFO request flag (TCSR[FRF]) becomes 1 when the number of entries in any enabled transmit FIFO is less than or equal to the transmit FIFO watermark configuration (TCR1[TFW]). This flag becomes 0 when the number of entries in each enabled transmit FIFO is greater than the transmit FIFO watermark configuration.

The receive FIFO request flag (RCSR[FRF]) becomes 1 when the number of entries in any enabled receive FIFO is greater than the receive FIFO watermark configuration (RCR1[RFW]). This flag becomes 0 when the number of entries in each enabled receive FIFO is less than or equal to the receive FIFO watermark configuration.

The FIFO request flag can generate an interrupt (TCSR[FRIE] = 1) or a DMA request (TCSR[FRDE] = 1).

51.3.8.2 FIFO warning flag

The transmit FIFO warning flag (TCSR[FWF]) becomes 1 when the number of entries in any of the enabled transmit FIFOs is empty. This flag becomes 0 when the number of entries in each enabled transmit FIFO is not empty.

The receive FIFO warning flag (RCSR[FWF]) becomes 1 when the number of entries in any of the enabled receive FIFOs is full. This flag becomes 0 when the number of entries in each enabled receive FIFO is not full.

The FIFO warning flag can generate an interrupt (TCSR[FWIE] = 1) or a DMA request (TCSR[FWDE] = 1).

51.3.8.3 FIFO error flag

The Transmit FIFO Error Flag (TCSR[FEF]) becomes 1 when any of the enabled transmit FIFOs underrun. After the error flag becomes 1, all enabled transmit channels transmit zero data before TCSR[FEF] becomes 0.

When FIFO Continue on Error is enabled (TCR4[FCONT] = 1), the FIFO continues transmitting data following an underrun without software intervention. To ensure that data transmits in the correct order, the transmitter continues from the same word number in the frame that caused the FIFO to underrun. It continues only after new data writes to the transmit FIFO. Software should still write 0 to the TCSR[FEF] flag, but without reinitializing the transmit FIFOs.

The Receive FIFO Error Flag (RCSR[FEF]) becomes 1 when any enabled receive FIFO overflows. After the flag becomes 1, all enabled receive channels discard received data until RCSR[FEF] becomes 0 and the next receive frame starts. Empty all enabled receive FIFOs before RCSR[FEF] becomes 0.

When Receive FIFO Continue on Error is enabled (RCR4[FCONT] = 1), the FIFO continues receiving data following an overflow without software intervention. To ensure that data is received in the correct order, the receiver continues from the same word number in the frame that caused the FIFO to overflow. It continues only after data has been read from the receive FIFO. Software should still write 0 to the RCSR[FEF] flag, but without emptying the receive FIFOs.

The FIFO Error Flag can generate only an interrupt.

51.3.8.4 Sync error flag

The Sync Error Flag (TCSR[SEF] or RCSR[SEF]) becomes 1 when both of these conditions are true:

- The frame sync is generated externally.
- The external frame sync asserts when the transmitter or receiver is busy with the previous frame.

The external frame sync assertion is ignored and the Sync Error Flag becomes 1. When the Sync Error Flag becomes 1, the transmitter or receiver continues checking for frame sync assertion when idle or at the end of each frame.

The Sync Error Flag can only generate an interrupt.

51.3.8.5 Word start flag

The Word Start Flag (TCSR[WSF]) becomes 1 at the start of the second bit-clock cycle for the selected word. This word is selected via the Word Flag Configuration field (TCR3[WDFL]).

The Word Start Flag can generate an interrupt only.

51.4 External signals

Name	Function	I/O
TX_BCLK	Transmit Bit Clock. The bit clock is an input when externally generated and an output when internally generated.	I/O

Table continues on the next page...

Table continued from the previous page...

Name	Function	I/O
TX_SYNC	Transmit Frame Sync. The frame sync is an input sampled synchronously by the bit clock when externally generated. It is an output generated synchronously by the bit clock when internally generated.	I/O
TX_DATA	Transmit Data. The transmit data is generated synchronously by the bit clock and is 3-stated whenever not transmitting a word.	O
RX_BCLK	Receive Bit Clock. The bit clock is an input when externally generated and an output when internally generated.	I/O
RX_SYNC	Receive Frame Sync. The frame sync is an input sampled synchronously by the bit clock when externally generated. It is an output generated synchronously by the bit clock when internally generated.	I/O
RX_DATA	Receive Data. The receive data is sampled synchronously by the bit clock.	I

51.5 Initialization

To initialize the SAI:

- Enable the clock to SAI.
- Reset the internal transmitter logic and receiver logic by TCSR[SR] and RCSR[SR].

51.6 Memory map and register definition

A read or write access to an address from offset 100h and above will result in a bus error.

51.6.1 SAI register descriptions

51.6.1.1 SAI memory map

SAI1 base address: 443B_0000h

SAI2 base address: 42BB_0000h

SAI3 base address: 42BC_0000h

SAI4 base address: 42BD_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0301_0000h
4h	Parameter (PARAM)	32	R	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8h	Transmit Control (TCSR)	32	RW	0000_0000h
Ch	Transmit Configuration 1 (TCR1)	32	RW	0000_0000h
10h	Transmit Configuration 2 (TCR2)	32	RW	0000_0000h
14h	Transmit Configuration 3 (TCR3)	32	RW	0000_0000h
18h	Transmit Configuration 4 (TCR4)	32	RW	0000_0000h
1Ch	Transmit Configuration 5 (TCR5)	32	RW	0000_0000h
20h - 2Ch	Transmit Data (TDR0 - TDR3)	32	W	See section
40h - 4Ch	Transmit FIFO (TFR0 - TFR3)	32	R	See section
60h	Transmit Mask (TMR)	32	RW	0000_0000h
88h	Receive Control (RCSR)	32	RW	0000_0000h
8Ch	Receive Configuration 1 (RCR1)	32	RW	0000_0000h
90h	Receive Configuration 2 (RCR2)	32	RW	0000_0000h
94h	Receive Configuration 3 (RCR3)	32	RW	0000_0000h
98h	Receive Configuration 4 (RCR4)	32	RW	0000_0000h
9Ch	Receive Configuration 5 (RCR5)	32	RW	0000_0000h
A0h - ACh	Receive Data (RDR0 - RDR3)	32	R	See section
C0h - CCh	Receive FIFO (RFR0 - RFR3)	32	R	See section
E0h	Receive Mask (RMR)	32	RW	0000_0000h

51.6.1.2 Version ID (VERID)

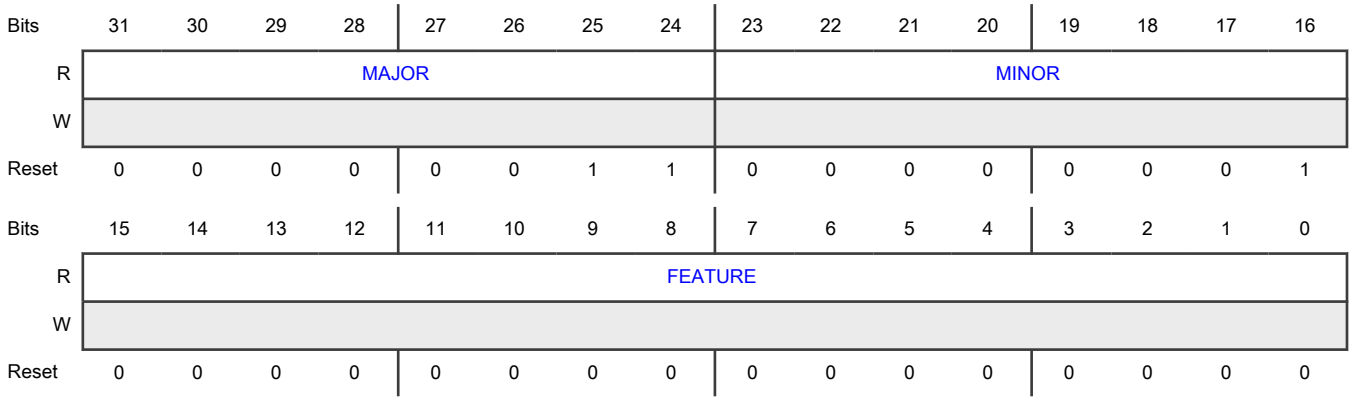
Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the specification.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000_0000_0000_0000b - Standard feature set.

51.6.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains parameter values implemented in the module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												FRAME			
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FIFO				0				DATALINE			
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PARAM	SAI1: 0005_0402h SAI2,SAI3: 0005_0501h SAI4: 0005_0504h

Fields

Field	Function
31-20 —	Reserved
19-16 FRAME	Frame Size The maximum number of slots per frame is 2^FRAME.
15-12 —	Reserved
11-8 FIFO	FIFO Size The number of words in each FIFO is 2^FIFO.
7-4 —	Reserved
3-0 DATALINE	Number of Datalinks The number of datalines implemented.

51.6.1.4 Transmit Control (TCSR)

Offset

Register	Offset
TCSR	8h

Function

Contains transmitter enable fields including resets, error and interrupt enable fields, and error flag fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TE	STOPE	DBGE	BCE	0		0	SR	0			WSF	SEF	FEF	FWF	FRF
W							FR					W1C	W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			WSIE	SEIE	FEIE	FWIE	FRIE	0			0			FWDE	FRDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TE	Transmitter Enable Enables and disables the transmitter. When software clears this field, the transmitter remains enabled, and this bit remains 1, until the end of the current frame. 0b - Transmitter is disabled. 1b - Transmitter is enabled, or transmitter has been disabled and has not yet reached end of frame.
30 STOPE	Stop Enable Configures transmitter operation in Stop mode. 0b - Transmitter disabled in Stop mode. 1b - Transmitter enabled in Stop mode.
29 DBGE	Debug Enable Enables and disables transmitter operation in Debug mode. The transmit bit clock is not affected by Debug mode. 0b - Transmitter is disabled in Debug mode, after completing the current frame. 1b - Transmitter is enabled in Debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 BCE	Bit Clock Enable Enables the transmit bit clock, separately from TE. This field is automatically set whenever TE is set. When software clears this field, the transmit bit clock remains enabled, and this bit remains set, until the end of the current frame. 0b - Transmit bit clock is disabled. 1b - Transmit bit clock is enabled.
27-26 —	Reserved
25 FR	FIFO Reset Empties the FIFO, and sets the FIFO read and write pointers to the same value, which may or may not be zero. **NOTE** The FIFO reset is asserted for one cycle only. Reading this field always returns zero. Reset FIFO pointers when the transmitter is disabled or the FIFO error flag is 1. 0b - No effect. 1b - FIFO reset.
24 SR	Software Reset When 1, resets the internal transmitter logic including the FIFO read and write pointers. Software-visible registers are not affected, except for the status registers. **NOTE** The software reset remains asserted until software clears the bit by writing 0. 0b - No effect 1b - Software reset
23-21 —	Reserved
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0b - Start of word not detected. 1b - Start of word detected.
19 SEF	Sync Error Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0b - Sync error not detected. 1b - Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled transmit FIFO has underrun. Write a logic 1 to this field to clear this flag. 0b - Transmit underrun not detected. 1b - Transmit underrun detected.
17 FWF	FIFO Warning Flag Indicates that an enabled transmit FIFO is empty. 0b - No enabled transmit FIFO is empty. 1b - Enabled transmit FIFO is empty.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled transmit channel FIFO is less than or equal to the transmit FIFO watermark. 0b - Transmit FIFO watermark has not been reached. 1b - Transmit FIFO watermark has been reached.
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables and disables word start interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables and disables sync error interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
10 FEIE	FIFO Error Interrupt Enable Enables and disables FIFO error interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
9	FIFO Warning Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
FWIE	Enables and disables FIFO warning interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
8 FRIE	FIFO Request Interrupt Enable Enables and disables FIFO request interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables and disables DMA warning. 0b - Disables the DMA warning. 1b - Enables the DMA warning.
0 FRDE	FIFO Request DMA Enable Enables and disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.

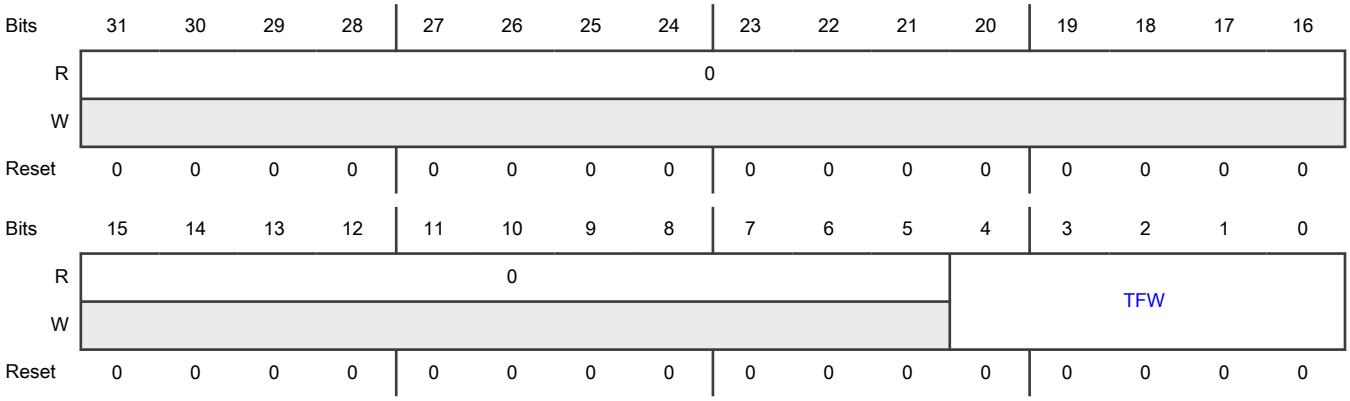
51.6.1.5 Transmit Configuration 1 (TCR1)

Offset

Register	Offset
TCR1	Ch

Function
Configures the watermark level for all enabled transmit channels.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 TFW	Transmit FIFO Watermark Number of 32-bit FIFO words **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
SAI1	TCR1[3–0]	TCR1[4]
SAI2	TCR1	—
SAI3	TCR1	—
SAI4	TCR1	—

NOTE
The descriptions of the field settings vary by module instance.

Instance	Field value and description
SAI1	0000b - 1 FIFO word 0001b - 2 FIFO words 0010b-1110b - (TFW +1) FIFO words 1111b - 16 FIFO words

Field	Function

Instance	Field value and description
SAI2	0_0000b - 1 FIFO word 0_0001b - 2 FIFO words 0_0010b-1_1110b - (TFW +1) FIFO words 1_1111b - 32 FIFO words
SAI3	0_0000b - 1 FIFO word 0_0001b - 2 FIFO words 0_0010b-1_1110b - (TFW +1) FIFO words 1_1111b - 32 FIFO words
SAI4	0_0000b - 1 FIFO word 0_0001b - 2 FIFO words 0_0010b-1_1110b - (TFW +1) FIFO words 1_1111b - 32 FIFO words

51.6.1.6 Transmit Configuration 2 (TCR2)

Offset

Register	Offset
TCR2	10h

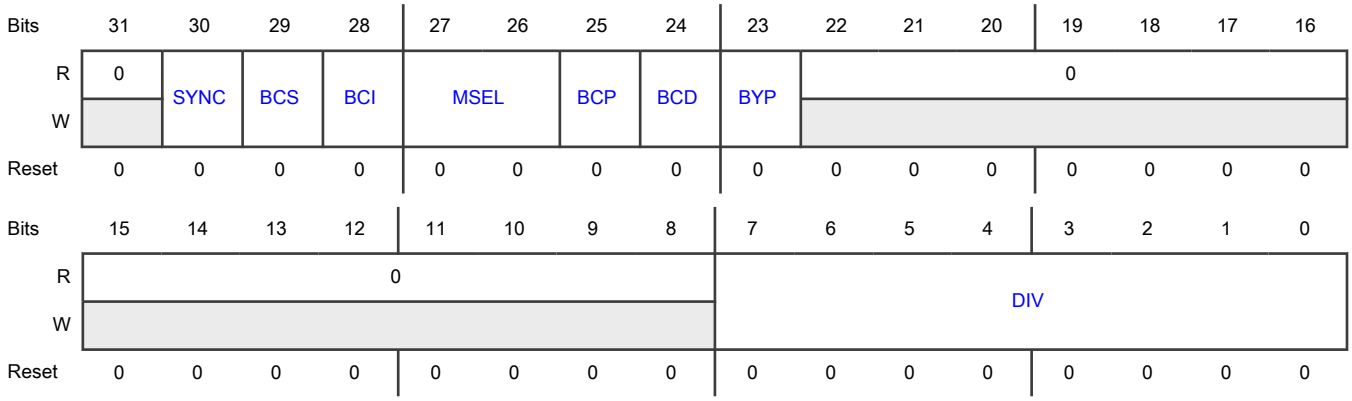
Function

Contains the SYNC mode and clock setting fields.

NOTE

This register must not be altered when TCSR[TE] is 1.

Diagram



Fields

Field	Function
31 —	Reserved
30 SYNC	Synchronous Mode Configures Asynchronous and Synchronous modes of operation. When configured for Synchronous mode, the receiver must be configured for Asynchronous operation. 0b - Asynchronous mode 1b - Synchronous with receiver
29 BCS	Bit Clock Swap Swaps the bit clock used by the transmitter. When the transmitter is configured in asynchronous mode and this bit is set, the transmitter is clocked by the receiver bit clock (RX_BCLK). This allows the transmitter and receiver to share the same bit clock, but the transmitter continues to use the transmit frame sync (TX_SYNC). When the transmitter is configured in Synchronous mode, the transmitter BCS field and receiver BCS field must be set to the same value. When both are set, the transmitter and receiver are both clocked by the transmitter bit clock (TX_BCLK) but use the receiver frame sync (RX_SYNC). 0b - Use the normal bit clock source. 1b - Swap the bit clock source.
28 BCI	Bit Clock Input When this field is 1 and when using an internally generated bit clock in Synchronous or Asynchronous mode, the bit clock used by the transmitter is delayed by the pad output delay. (The transmitter is clocked by the pad input as if the clock was externally generated.) This setting decreases the data input setup time, but increases the data output valid time. The Slave mode timing from the datasheet should be used for the transmitter when this bit is 1. In Synchronous mode, this bit allows the transmitter to use the Slave mode timing from the datasheet, while the receiver uses the Master mode timing. This field has no effect when configured for an externally generated bit clock .

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE When BCI = 1, both the input buffer and output buffer must be enabled for the BCLK pad. 0b - No effect. 1b - Internal logic is clocked as if bit clock was externally generated.
27-26 MSEL	MCLK Select Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the meaning of each option. 00b - Bus Clock selected. 01b - Master Clock (MCLK) 1 option selected. 10b - Master Clock (MCLK) 2 option selected. 11b - Master Clock (MCLK) 3 option selected.
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. 0b - Bit clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1b - Bit clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Bit clock is generated externally in Slave mode. 1b - Bit clock is generated internally in Master mode.
23 BYP	Bit Clock Bypass Bypasses the bit clock divider. Internal bit clock is divide-by-one of the audio master clock. 0b - Internal bit clock is generated from bit clock divider. 1b - Internal bit clock is divide-by-one of the audio master clock.
22-8 —	Reserved
7-0 DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock when configured for an internal bit clock. The division value is (DIV + 1) * 2.

51.6.1.7 Transmit Configuration 3 (TCR3)

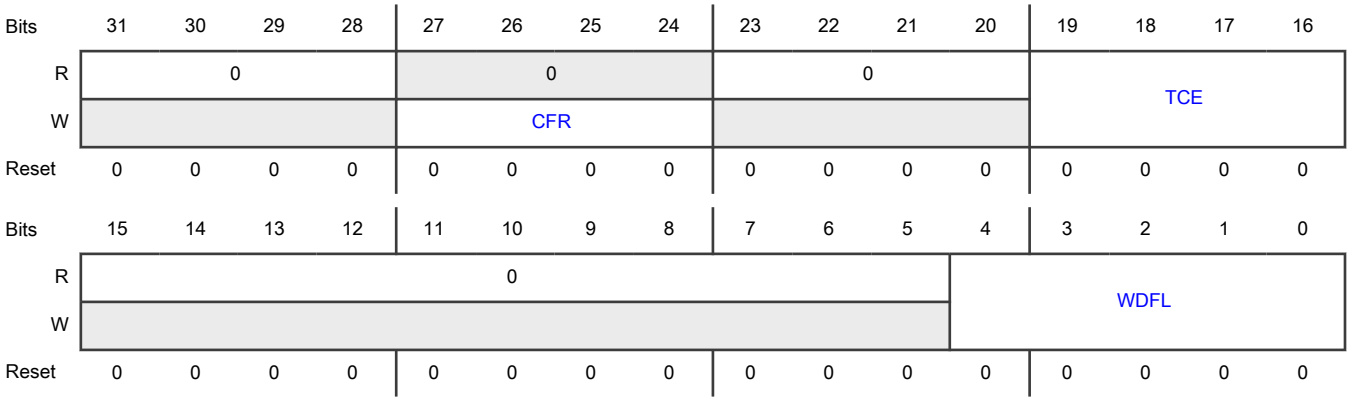
Offset

Register	Offset
TCR3	14h

Function

Contains the transmit channel settings.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 CFR	Channel FIFO Reset Resets the FIFO pointers for a specific channel. Reading this field always returns zero. FIFO pointers should only be reset when a channel is disabled or the FIFO error flag is set. The width of CFR field = the number of transmit channels (call it N). For example, if CFR is 2 bits wide, then bit position 24 refers to transmit channel 1 FIFO pointer and bit position 25 refers to transmit channel 2 FIFO pointer. Writing 1 to bit 24 resets transmit channel 1 FIFO pointer, and writing 1 to bit 25 enables transmit channel 2 FIFO pointer. Writing 1 to bit N resets transmit channel N FIFO pointer. **NOTE** When there is only a single channel, there is no need for individual channel FIFO reset (TCR3[CFR]). In the case of a single channel, use the global FIFO reset (TCSR[FR]). 0b - No effect. 1b - Transmit data channel N FIFO is reset.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI1</td><td>TCR3[25–24]</td><td>TCR3[27–26]</td></tr>
<tr><td>SAI2</td><td>—</td><td>TCR3</td></tr>
<tr><td>SAI3</td><td>—</td><td>TCR3</td></tr>
<tr><td>SAI4</td><td>TCR3</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>23-20
—</td><td>Reserved</td></tr>
<tr><td>19-16
TCE</td><td>
Transmit Channel Enable
Enables the corresponding data channel for transmit operation. Changing TCE field takes effect immediately for generating the FIFO request and warning flags. It takes effect at the end of each frame for transmit operations.
The width of TCE field = the number of transmit channels (call it N). For example, if TCE field is two bits wide, then bit position 16 refers to transmit channel 1 and bit position 17 refers to transmit channel 2. Writing 1 to bit 16 enables transmit channel 1, and setting bit 17 enables transmit channel 2. Writing 1 to bit N enables transmit channel N.
0b - Transmit data channel N is disabled.
1b - Transmit data channel N is enabled.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI1</td><td>TCR3[17–16]</td><td>TCR3[19–18]</td></tr>
<tr><td>SAI2</td><td>TCR3[16]</td><td>TCR3[19–17]</td></tr>
<tr><td>SAI3</td><td>TCR3[16]</td><td>TCR3[19–17]</td></tr>
<tr><td>SAI4</td><td>TCR3</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>15-5</td><td>Reserved</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4-0 WDFL	Word Flag Configuration Configures which word causes the word start flag (TCSR[WSF]) to become 1. The value written must be one less than the word number. For example, writing 0 selects the first word in the frame. When configured to a value greater than TCR4[FRSZ], TCSR[WSF] is never 1.

51.6.1.8 Transmit Configuration 4 (TCR4)

Offset

Register	Offset
TCR4	18h

Function

Contains the transmit fields for FIFO Combine mode, FIFO Packing mode, and frame sync settings.

NOTE

This register must not be altered when TCSR[TE] is 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			FCONT	FCOMB		FPACK		0				FRSZ			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			SYWD					0		CHMOD	MF	FSE	ONDEM	FSP	FSD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28	FIFO Continue on Error

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>FCONT</td><td>Configures when SAI continues transmitting after a FIFO error has been detected.
0b - On FIFO error, SAI continues from the start of the next frame after the FIFO error flag has been cleared.
1b - On FIFO error, SAI continues from the same word that caused the FIFO error to become 1 after the FIFO warning flag returns to 0.</td></tr>
<tr><td>27-26
FCOMB</td><td>FIFO Combine Mode
When FIFO Combine mode is enabled for FIFO writes, software writing to any FIFO data register alternates the write among the enabled data channel FIFOs. For example, if two data channels are enabled then the first write is performed to the first enabled data channel FIFO and the second write is performed to the second enabled data channel FIFO. Resetting the FIFO or disabling FIFO Combine mode for FIFO writes resets the pointer back to the first enabled data channel.
When FIFO Combine mode is enabled for FIFO reads from the transmit shift registers, the transmit data channel output alternated between the enabled data channel FIFOs. For example, if two data channels are enabled then the first unmasked word is transmitted from the first enabled data channel FIFO and the second unmasked word is transmitted from the second enabled data channel FIFO. Since the first word of the frame is always transmitted from the first enabled data channel FIFO, it is recommended that the number of unmasked words per frame is evenly divisible by the number of enabled data channels.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI1</td><td>TCR4</td><td>—</td></tr><tr><td>SAI2</td><td>—</td><td>TCR4</td></tr><tr><td>SAI3</td><td>—</td><td>TCR4</td></tr><tr><td>SAI4</td><td>TCR4</td><td>—</td></tr></table>
00b - FIFO Combine mode disabled.
01b - FIFO Combine mode enabled on FIFO reads (from transmit shift registers).
10b - FIFO Combine mode enabled on FIFO writes (by software).
11b - FIFO Combine mode enabled on FIFO reads (from transmit shift registers) and writes (by software).</td></tr>
<tr><td>25-24
FPACK</td><td>FIFO Packing Mode
Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8 or 16 bits then only the first 8 or 16 bits are loaded from the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	FIFO packing is enabled, the FIFO write pointer only increments when the full 32-bit FIFO word has been written by software. 00b - FIFO packing is disabled. 01b - Reserved 10b - 8-bit FIFO packing is enabled. 11b - 16-bit FIFO packing is enabled.
23-21 —	Reserved
20-16 FRSZ	Frame size Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 32 words.
15-13 —	Reserved
12-8 SYWD	Sync Width Configures the length of the frame sync in number of bit-clock cycles. The value written must be one less than the number of bit-clock cycles. For example, write 0 for the frame sync to assert for one bit-clock cycle only. The sync width cannot be configured longer than the first word of the frame.
7-6 —	Reserved
5 CHMOD	Channel Mode Configures whether transmit data pins are configured for TDM mode or Output mode. 0b - TDM mode, transmit data pins are 3-stated when slots are masked or channels are disabled. 1b - Output mode, transmit data pins are never 3-stated and output zero when slots are masked or channels are disabled.
4 MF	MSB First Configures whether the LSB or the MSB is transmitted first. 0b - LSB is transmitted first. 1b - MSB is transmitted first.
3 FSE	Frame Sync Early 0b - Frame sync asserts with the first bit of the frame. 1b - Frame sync asserts one bit before the first bit of the frame.
2	On Demand Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
ONDEM	When 1, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is 0. 0b - Internal frame sync is generated continuously. 1b - Internal frame sync is generated when the FIFO warning flag is 0.
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0b - Frame sync is active high. 1b - Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0b - Frame sync is generated externally in Slave mode. 1b - Frame sync is generated internally in Master mode.

51.6.1.9 Transmit Configuration 5 (TCR5)

Offset

Register	Offset
TCR5	1Ch

Function

Contains transmit word width and bit index settings.

NOTE

This register must not be altered when TCSR[TE] is 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				WNW				0				W0W			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			FBT					0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 WNW	Word N Width Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported. Bit settings 0 - 6 (0b - 0_0110b) not supported. 0_0111b - 8 bits per word 0_1000b - 9 bits per word 0_1001b-1_1110b - (WNW value + 1) bits per word 1_1111b - 32 bits per word
23-21 —	Reserved
20-16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame. Bit settings 0 - 6 (0b - 0_0110b) not supported. 0_0111b - 8 bits per word 0_1000b - 9 bits per word 0_1001b-1_1110b - (W0W value + 1) bits per word 1_1111b - 32 bits per word
15-13 —	Reserved
12-8 FBT	First Bit Shifted Configures the bit index for the first bit transmitted for each word in the frame. If configured for MSB-first, the index of the next bit transmitted is one less than the current bit transmitted. If configured for LSB-first, the index of the next bit transmitted is one more than the current bit transmitted. The value written must be greater than or equal to the word width when configured for MSB-first. The value written must be less than or equal to 31-word width when configured for LSB-first. See Data alignment. 0_0000b - Bit index is 0. 0_0001b-1_1110b - Bit index is FBT value. 1_1111b - Bit index is 31.
7-0 —	Reserved

51.6.1.10 Transmit Data (TDR0 - TDR3)

Offset

Register	Offset
TDR0	20h
TDR1	24h
TDR2	28h
TDR3	2Ch

Function

When the transmit FIFO is not full, writes to this register push the data written into the transmit data FIFO. When the transmit FIFO is full, writes to this register are ignored.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI1	TDR0–TDR1	TDR2–TDR3
SAI2	TDR0	TDR1–TDR3
SAI3	TDR0	TDR1–TDR3
SAI4	TDR0–TDR3	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	TDR															
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	TDR															
Reset	See Register reset values.															

Register reset values

Register	Reset value
TDR0	SAI1–SAI4: 0000_0000h

Table continues on the next page...

Table continued from the previous page...

Register	Reset value
TDR1	SAI1: 0000_0000h SAI2,SAI3: Register not supported SAI4: 0000_0000h
TDR2–TDR3	0000_0000h

Fields

Field	Function
31-0 TDR	Transmit Data Register

51.6.1.11 Transmit FIFO (TFR0 - TFR3)

Offset

Register	Offset
TFR0	40h
TFR1	44h
TFR2	48h
TFR3	4Ch

Function

The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI1	TFR0–TFR1	TFR2–TFR3
SAI2	TFR0	TFR1–TFR3
SAI3	TFR0	TFR1–TFR3
SAI4	TFR0–TFR3	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WCP	0											WFP			
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										RFP					
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
TFR0	SAI1–SAI4: 0000_0000h
TFR1	SAI1: 0000_0000h SAI2,SAI3: Register not supported SAI4: 0000_0000h
TFR2–TFR3	0000_0000h

Fields

Field	Function
31 WCP	Write Channel Pointer When FIFO Combine mode is enabled for writes, indicates that this data channel is the next FIFO to be written. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
SAI1	TFR0–TFR1	—
SAI2	—	TFR0
SAI3	—	TFR0
SAI4	TFR0–TFR3	—

0b - No effect

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>1b - FIFO Combine mode is enabled for FIFO writes and this FIFO will be written on the next FIFO write.</td></tr>
<tr><td>30-22
—</td><td>Reserved</td></tr>
<tr><td>21-16
WFP</td><td>Write FIFO Pointer
FIFO write pointer for transmit data channel
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI1</td><td>TFR0–TFR1[20–16]</td><td>TFR0–TFR1[21]</td></tr><tr><td>SAI2</td><td>TFR0</td><td>—</td></tr><tr><td>SAI3</td><td>TFR0</td><td>—</td></tr><tr><td>SAI4</td><td>TFR0–TFR3</td><td>—</td></tr></table></td></tr>
<tr><td>15-6
—</td><td>Reserved</td></tr>
<tr><td>5-0
RFP</td><td>Read FIFO Pointer
FIFO read pointer for transmit data channel
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI1</td><td>TFR0–TFR1[4–0]</td><td>TFR0–TFR1[5]</td></tr><tr><td>SAI2</td><td>TFR0</td><td>—</td></tr><tr><td>SAI3</td><td>TFR0</td><td>—</td></tr><tr><td>SAI4</td><td>TFR0–TFR3</td><td>—</td></tr></table></td></tr>
</table>

51.6.1.12 Transmit Mask (TMR)

Offset

Register	Offset
TMR	60h

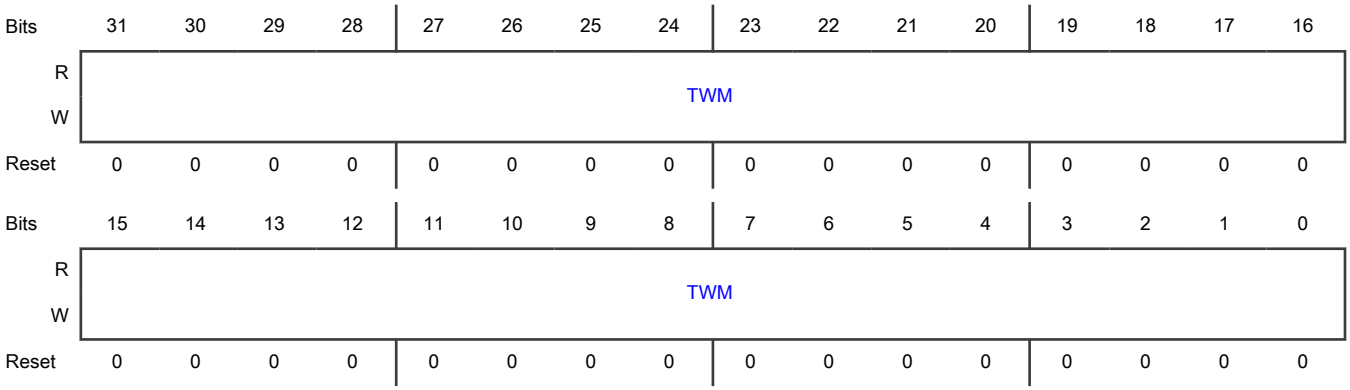
Function

This register is double-buffered and updates:

1. When TCSR[TE] first becomes 1.
2. At the end of each frame.

This setup allows the masked words in each frame to change from frame to frame.

Diagram



Fields

Field	Function
31-0 TWM	Transmit Word Mask Configures whether the transmit word is masked for the corresponding word in the frame. In this case, being masked means that transmit data pins are 3-stated or drive zero and transmit data is not read from the FIFO. 0000_0000_0000_0000_0000_0000_0000_0000b - Word N is enabled. 0000_0000_0000_0000_0000_0000_0000_0001b - Word N is masked. The transmit data pins are 3-stated or drive zero when masked.

51.6.1.13 Receive Control (RCSR)

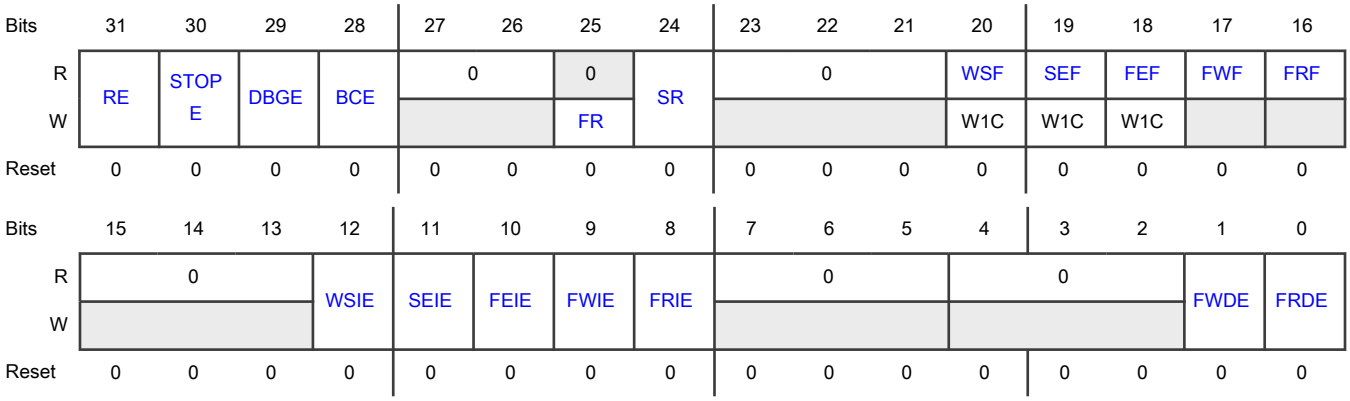
Offset

Register	Offset
RCSR	88h

Function

Contains receiver enable fields including resets, error and interrupt enable fields, and error flag fields.

Diagram



Fields

Field	Function
31 RE	Receiver Enable Enables and disables the receiver. When software clears this field, the receiver remains enabled, and this bit remains 1, until the end of the current frame. 0b - Receiver is disabled. 1b - Receiver is enabled, or receiver has been disabled and has not yet reached end of frame.
30 STOPE	Stop Enable Configures receiver operation in Stop mode. 0b - Receiver disabled in Stop mode. 1b - Receiver enabled in Stop mode.
29 DBGE	Debug Enable Enables and disables receiver operation in Debug mode. The receive bit clock is not affected by Debug mode. 0b - Receiver is disabled in Debug mode, after completing the current frame. 1b - Receiver is enabled in Debug mode.
28 BCE	Bit Clock Enable Enables the receive bit clock, separately from RE. This field is automatically set whenever RE is set. When software clears this field, the receive bit clock remains enabled, and this field remains set, until the end of the current frame. 0b - Receive bit clock is disabled. 1b - Receive bit clock is enabled.
27-26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25 FR	FIFO Reset Empties the FIFO, and sets the FIFO read and write pointers to the same value, which may or may not be zero. Reading this field always returns zero. **NOTE** The FIFO reset is asserted for one cycle only. FIFO pointers should only be reset when the receiver is disabled or the FIFO error flag is 1. 0b - No effect. 1b - FIFO reset.
24 SR	Software Reset Resets the internal receiver logic including the FIFO pointers. Software-visible registers are not affected, except for the status registers. **NOTE** The software reset remains asserted until cleared by software. Software clears the bit by writing 0. 0b - No effect. 1b - Software reset.
23-21 —	Reserved
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0b - Start of word not detected. 1b - Start of word detected.
19 SEF	Sync Error Flag Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0b - Sync error not detected. 1b - Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled receive FIFO has overflowed. Write a logic 1 to this field to clear this flag. 0b - Receive overflow not detected. 1b - Receive overflow detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 FWF	FIFO Warning Flag Indicates that an enabled receive FIFO is full. 0b - No enabled receive FIFO is full. 1b - Enabled receive FIFO is full.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled receive channel FIFO is greater than the receive FIFO watermark. 0b - Receive FIFO watermark not reached. 1b - Receive FIFO watermark has been reached.
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables and disables word start interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables and disables sync error interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
10 FEIE	FIFO Error Interrupt Enable Enables and disables FIFO error interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
9 FWIE	FIFO Warning Interrupt Enable Enables and disables FIFO warning interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
8 FRIE	FIFO Request Interrupt Enable Enables and disables FIFO request interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables and disables DMA warnings. 0b - Disables DMA warnings. 1b - Enables DMA warnings.
0 FRDE	FIFO Request DMA Enable Enables and disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.

51.6.1.14 Receive Configuration 1 (RCR1)

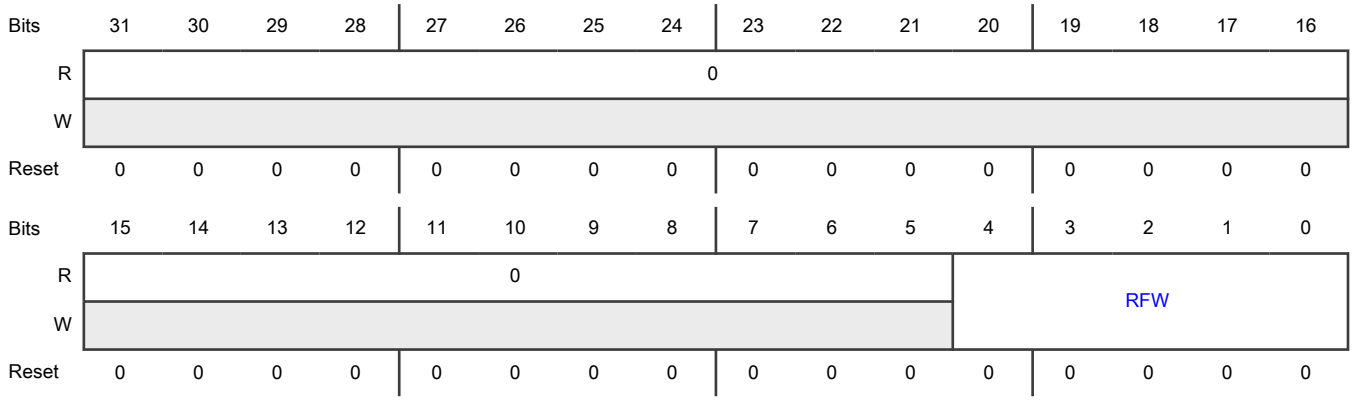
Offset

Register	Offset
RCR1	8Ch

Function

Configures the watermark level for all enabled receiver channels.

Diagram



Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31-5
—</td><td>Reserved</td></tr>
<tr><td>4-0
RFW</td><td>Receive FIFO Watermark
Number of 32-bit FIFO words
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI1</td><td>RCR1[3–0]</td><td>RCR1[4]</td></tr>
<tr><td>SAI2</td><td>RCR1</td><td>—</td></tr>
<tr><td>SAI3</td><td>RCR1</td><td>—</td></tr>
<tr><td>SAI4</td><td>RCR1</td><td>—</td></tr>
</table>
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>SAI1</td><td>0000b - 1 FIFO word
0001b - 2 FIFO words
0010b-1110b - (RFW value + 1) FIFO words
1111b - 16 FIFO words</td></tr>
<tr><td>SAI2</td><td>0_0000b - 1 FIFO word
0_0001b - 2 FIFO words
0_0010b-1_1110b - (RFW value + 1) FIFO words
1_1111b - 32 FIFO words</td></tr>
<tr><td>SAI3</td><td>0_0000b - 1 FIFO word
0_0001b - 2 FIFO words
0_0010b-1_1110b - (RFW value + 1) FIFO words
1_1111b - 32 FIFO words</td></tr>
<tr><td>SAI4</td><td>0_0000b - 1 FIFO word
0_0001b - 2 FIFO words</td></tr>
</table></td></tr>
</table>

Table continued from the previous page...

Field	Function

Instance	Field value and description
	0_0010b-1_1110b - (RFW value + 1) FIFO words 1_1111b - 32 FIFO words

51.6.1.15 Receive Configuration 2 (RCR2)

Offset

Register	Offset
RCR2	90h

Function

Contains the SYNC mode and clock setting fields.

NOTE

This register must not be altered when RCSR[RE] is 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SYNC	BCS	BCI	MSEL		BCP	BCD	BYP	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								DIV							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30	Synchronous Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYNC	Configures Asynchronous and Synchronous modes of operation. When configured for a Synchronous mode of operation, the transmitter must be configured for asynchronous operation. 0b - Asynchronous mode 1b - Synchronous with transmitter
29 BCS	Bit Clock Swap Swaps the bit clock used by the receiver. When the receiver is configured in Asynchronous mode and this bit is 1, the receiver is clocked by the transmitter bit clock (TX_BCLK). This setting allows the transmitter and receiver to share the same bit clock, but the receiver continues to use the receiver frame sync (RX_SYNC). When the receiver is configured in Synchronous mode, the transmitter BCS field and receiver BCS field must be set to the same value. When both are set, the transmitter and receiver are both clocked by the receiver bit clock (RX_BCLK) but use the transmitter frame sync (TX_SYNC). 0b - Use the normal bit clock source. 1b - Swap the bit clock source.
28 BCI	Bit Clock Input When this field is set and using an internally generated bit clock in either Synchronous or Asynchronous mode, the bit clock used by the receiver is delayed by the pad output delay (the receiver is clocked by the pad input as if the clock was externally generated). This setting decreases the data input setup time, but increases the data output valid time. The Slave mode timing from the datasheet should be used for the receiver when this bit is 1. In Synchronous mode, this bit allows the receiver to use the Slave mode timing from the datasheet, while the transmitter uses the Master mode timing. This field has no effect when configured for an externally generated bit clock. **NOTE** When BCI = 1 both the input buffer and output buffer must be enabled for the BCLK pad. 0b - No effect. 1b - Internal logic is clocked as if bit clock was externally generated.
27-26 MSEL	MCLK Select Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the availability and chip-specific meaning of each option. 00b - Bus Clock selected. 01b - Master Clock (MCLK) 1 option selected. 10b - Master Clock (MCLK) 2 option selected. 11b - Master Clock (MCLK) 3 option selected.
25	Bit Clock Polarity

Table continues on the next page...

Table continued from the previous page...

Field	Function
BCP	Configures the polarity of the bit clock. 0b - Bit clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1b - Bit clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Bit clock is generated externally in Slave mode. 1b - Bit clock is generated internally in Master mode.
23 BYP	Bit Clock Bypass Bypasses the bit clock divider, internal bit clock is divide-by-one of the audio master clock. 0b - Internal bit clock is generated from bit clock divider. 1b - Internal bit clock is divide-by-one of the audio master clock.
22-8 —	Reserved
7-0 DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock, when configured for an internal bit clock. The division value is (DIV + 1) * 2.

51.6.1.16 Receive Configuration 3 (RCR3)

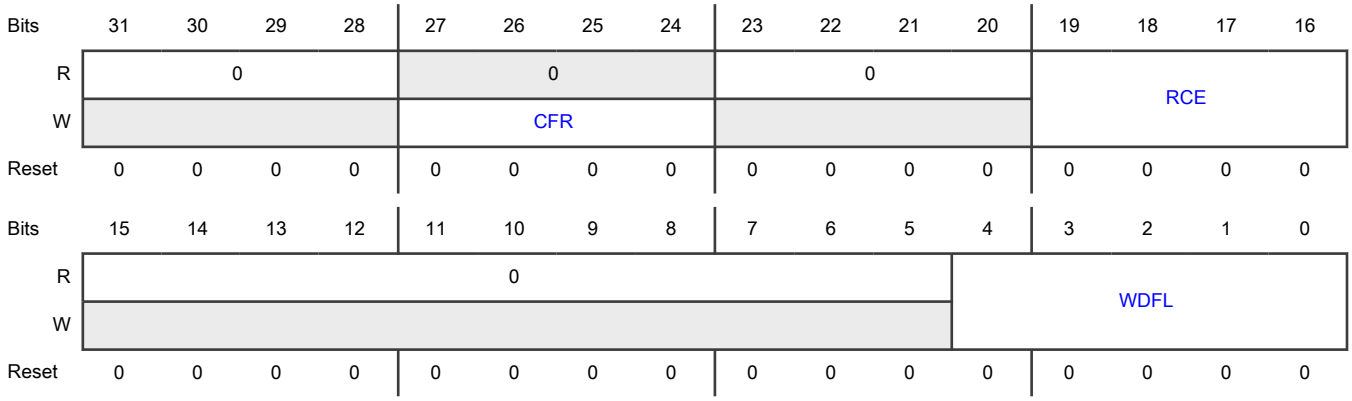
Offset

Register	Offset
RCR3	94h

Function

Contains the receive channel settings.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 CFR	Channel FIFO Reset Resets the FIFO pointers for a specific channel. Reading this field always returns zero. FIFO pointers should only be reset when a channel is disabled or the FIFO error flag is 1. The width of CFR field = the number of receive channels (call it N). For example, if CFR is two bits wide, then bit position 24 refers to receive channel 1 FIFO pointer and bit position 25 refers to receive channel 2 FIFO pointer. Writing 1 to bit 24 resets receive channel 1 FIFO pointer, and writing 1 to bit 25 enables receive channel 2 FIFO pointer. Writing 1 to bit N resets receive channel N FIFO pointer. 0b - No effect. 1b - Receive data channel N FIFO is reset. **NOTE** This field is not supported in every instance. The following table includes only supported registers.
23-20 —	Reserved

Instance	Field supported in	Field not supported in
SAI1	RCR3[25–24]	RCR3[27–26]
SAI2	—	RCR3
SAI3	—	RCR3
SAI4	RCR3	—

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>19-16
RCE</td><td>Receive Channel Enable
Enables the corresponding data channel for receive operation. Changing this field takes effect immediately for generating the FIFO request and warning flags. It takes effect at the end of each frame for receive operations.
The width of RCE = the number of receive channels (call it N). For example, if RCE is two bits wide, then bit position 16 refers to receive channel 1 and bit position 17 refers to receive channel 2. Writing 1 to bit 16 enables receive channel 1, and writing 1 to bit 17 enables receive channel 2. Writing 1 to bit N enables receive channel N.
NOTE
When there is only a single channel, there is no need for individual channel FIFO reset. In the case of a single channel, use the global FIFO reset (RCSR[FR]).
0b - Receive data channel N is disabled.
1b - Receive data channel N is enabled.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI1</td><td>RCR3[17–16]</td><td>RCR3[19–18]</td></tr>
<tr><td>SAI2</td><td>RCR3[16]</td><td>RCR3[19–17]</td></tr>
<tr><td>SAI3</td><td>RCR3[16]</td><td>RCR3[19–17]</td></tr>
<tr><td>SAI4</td><td>RCR3</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>15-5
—</td><td>Reserved</td></tr>
<tr><td>4-0
WDFL</td><td>Word Flag Configuration
Configures which word causes the word start flag (RCSR[WSF]) to become 1. The value written should be one less than the word number (for example, write zero to select the first word in the frame). When configured to a value greater than RCR4[FRSZ], then RCSR[WSF] is never 1.
0_0000b - Word 1
0_0001b - Word 2
0_0010b-1_1110b - Word (WDFL value + 1)
1_1111b - Word 32</td></tr>
</table>

51.6.1.17 Receive Configuration 4 (RCR4)

Offset

Register	Offset
RCR4	98h

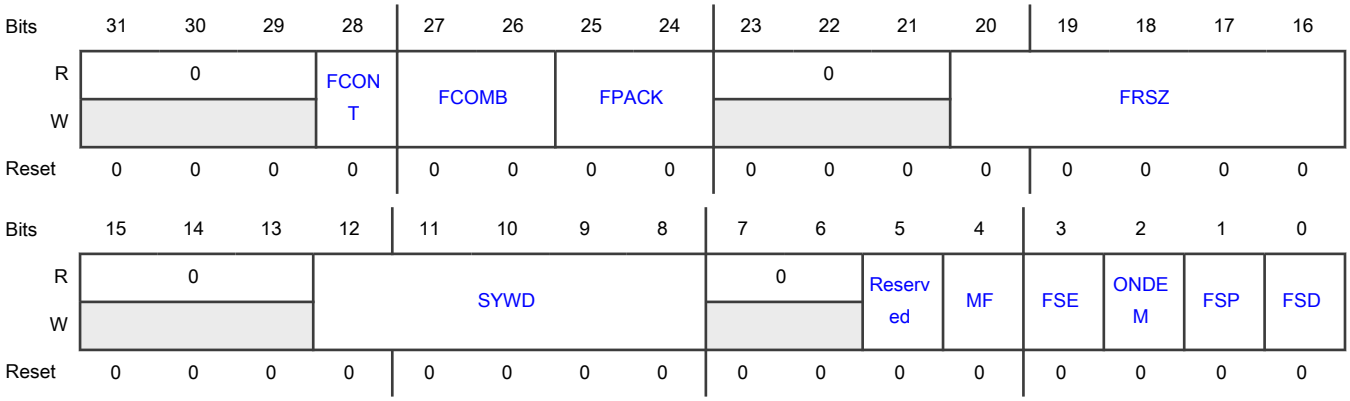
Function

Contains the receive fields for FIFO Combine mode, FIFO Packing mode, and frame sync settings.

NOTE

This register must not be altered when RCSR[RE] is 1.

Diagram



Fields

Field	Function
31-29 —	Reserved
28 FCONT	FIFO Continue on Error Configures when SAI continues receiving after a FIFO error has been detected. 0b - On FIFO error, SAI continues from the start of the next frame after the FIFO error flag returns to 0. 1b - On FIFO error, SAI continues from the same word that caused the FIFO error to become 1 after the FIFO warning flag returns to 0.
27-26 FCOMB	FIFO Combine Mode When FIFO Combine mode is enabled for FIFO reads, software reading any FIFO data register alternates the read among the enabled data channel FIFOs. For example, if two data channels are enabled then the first read is performed to the first enabled data channel FIFO and the second read is performed to the second

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>enabled data channel FIFO. Resetting the FIFO or disabling FIFO Combine mode for FIFO reads resets the pointer back to the first enabled data channel.

When FIFO Combine mode is enabled for FIFO writes from the receive shift registers, the first enabled data channel input alternates between the enabled data channel FIFOs. For example, if two data channels are enabled, then the first unmasked received word is stored in the first enabled data channel FIFO and the second unmasked received word is stored in the second enabled data channel FIFO. Since the first word of the frame is always stored in the first enabled data channel FIFO, it is recommended that the number of unmasked words per frame is evenly divisible by the number of enabled data channels.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SAI1</td><td>RCR4</td><td>—</td></tr>
<tr><td>SAI2</td><td>—</td><td>RCR4</td></tr>
<tr><td>SAI3</td><td>—</td><td>RCR4</td></tr>
<tr><td>SAI4</td><td>RCR4</td><td>—</td></tr>
</table>

00b - FIFO Combine mode disabled.

01b - FIFO Combine mode enabled on FIFO writes (from receive shift registers).

10b - FIFO Combine mode enabled on FIFO reads (by software).

11b - FIFO Combine mode enabled on FIFO writes (from receive shift registers) and reads (by software).</td></tr>
<tr><td>25-24
FPACK</td><td>FIFO Packing Mode

Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8 or 16 bits, only the first 8 or 16 bits are stored in the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO read pointer only increments when the full 32-bit FIFO word has been read by software.

00b - FIFO packing is disabled

01b - Reserved

10b - 8-bit FIFO packing is enabled

11b - 16-bit FIFO packing is enabled</td></tr>
<tr><td>23-21
—</td><td>Reserved</td></tr>
<tr><td>20-16</td><td>Frame Size</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
FRSZ	Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 32 words. 0_0000b - 1 word per frame 0_0001b - 2 words per frame 0_0010b-1_1110b - (FRSZ value + 1) words per frame 1_1111b - 32 words per frame
15-13 —	Reserved
12-8 SYWD	Sync Width Configures the length of the frame sync in number of bit-clock cycles. The value written must be one less than the number of bit-clock cycles. For example, write 0 for the frame sync to assert for one bit-clock cycle only. The sync width cannot be configured longer than the first word of the frame. 0_0000b - 1 bit-clock cycle 0_0001b - 2 bit-clock cycle 0_0010b-1_1110b - (SYWD value + 1) bit-clock cycle 1_1111b - 32 bit-clock cycle
7-6 —	Reserved
5 —	Reserved Software should only write zero to this bit.
4 MF	MSB First Configures whether the LSB or the MSB is received first. 0b - LSB is received first. 1b - MSB is received first.
3 FSE	Frame Sync Early 0b - Frame sync asserts with the first bit of the frame. 1b - Frame sync asserts one bit before the first bit of the frame.
2 ONDEM	On Demand Mode When 1, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is 0. 0b - Internal frame sync is generated continuously. 1b - Internal frame sync is generated when the FIFO warning flag is 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0b - Frame sync is active high. 1b - Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0b - Frame Sync is generated externally in Slave mode. 1b - Frame Sync is generated internally in Master mode.

51.6.1.18 Receive Configuration 5 (RCR5)

Offset

Register	Offset
RCR5	9Ch

Function

Contains receive word and bit index settings.

NOTE

This register must not be altered when RCSR[RE] is 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				WNW				0				W0W			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FBT				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 WNW	Word N Width Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported. Bit settings 0 - 6 (0b - 0_0110b) are not supported. 0_0111b - 8 bits per word 0_1000b - 9 bits per word 0_1001b-1_1110b - (WNW value + 1) bits per word 1_1111b - 32 bits per word
23-21 —	Reserved
20-16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame. 0_0000b - 1 bit per word 0_0001b - 2 bits per word 0_0010b-1_1110b - (W0W value + 1) bits per word 1_1111b - 32 bits per word
15-13 —	Reserved
12-8 FBT	First Bit Shifted Configures the bit index for the first bit received for each word in the frame. If configured for MSB-first, the index of the next bit received is one less than the current bit received. If configured for LSB-first, the index of the next bit received is one more than the current bit received. The value written must be greater than or equal to the word width when configured for MSB-first. The value written must be less than or equal to 31-word width when configured for LSB-first. See Data alignment. 0_0000b - Bit index is 0. 0_0001b-1_1110b - Bit index is FBT value. 1_1111b - Bit index is 31.
7-0 —	Reserved

51.6.1.19 Receive Data (RDR0 - RDR3)

Offset

Register	Offset
RDR0	A0h
RDR1	A4h
RDR2	A8h
RDR3	ACh

Function

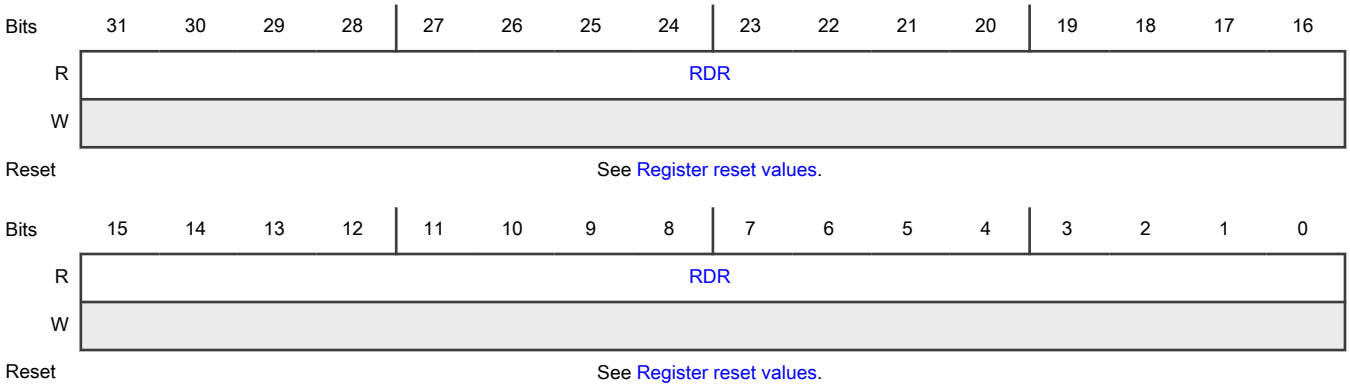
When the receive FIFO is not empty, reads from this register return the data from the top of the receive FIFO. When the receive FIFO is empty, reads from this register are ignored.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI1	RDR0–RDR1	RDR2–RDR3
SAI2	RDR0	RDR1–RDR3
SAI3	RDR0	RDR1–RDR3
SAI4	RDR0–RDR3	—

Diagram



Register reset values

Register	Reset value
RDR0	SAI1–SAI4: 0000_0000h

Table continues on the next page...

Table continued from the previous page...

Register	Reset value
RDR1	SAI1: 0000_0000h SAI2,SAI3: Register not supported SAI4: 0000_0000h
RDR2–RDR3	0000_0000h

Fields

Field	Function
31-0 RDR	Receive Data Register

51.6.1.20 Receive FIFO (RFR0 - RFR3)

Offset

Register	Offset
RFR0	C0h
RFR1	C4h
RFR2	C8h
RFR3	CCh

Function

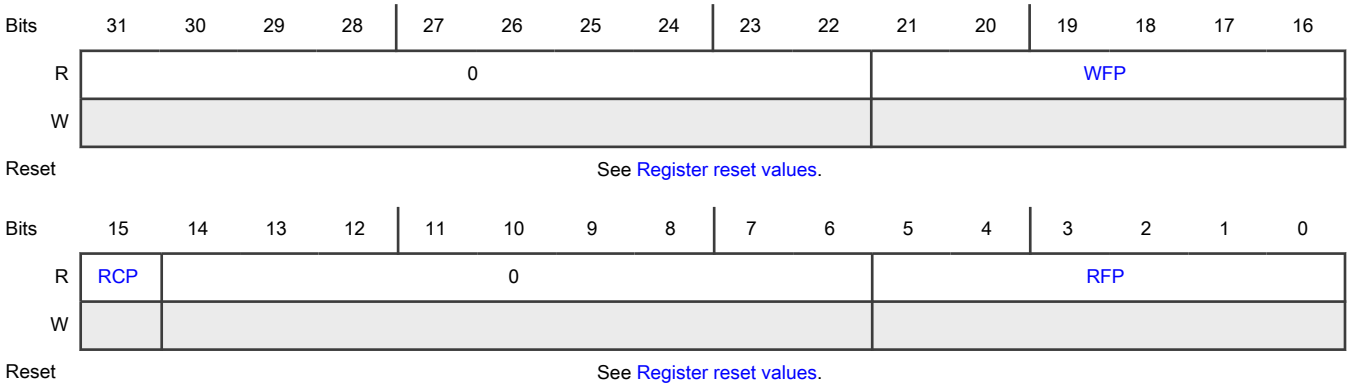
The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
SAI1	RFR0–RFR1	RFR2–RFR3
SAI2	RFR0	RFR1–RFR3
SAI3	RFR0	RFR1–RFR3
SAI4	RFR0–RFR3	—

Diagram



Register reset values

Register	Reset value
RFR0	SAI1–SAI4: 0000_0000h
RFR1	SAI1: 0000_0000h SAI2,SAI3: Register not supported SAI4: 0000_0000h
RFR2–RFR3	0000_0000h

Fields

Field	Function
31-22 —	Reserved
21-16 WFP	Write FIFO Pointer FIFO write pointer for receive data channel. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
SAI1	RFR0–RFR1[20–16]	RFR0–RFR1[21]
SAI2	RFR0	—
SAI3	RFR0	—
SAI4	RFR0–RFR3	—

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>15
RCP</td><td>Receive Channel Pointer
When FIFO Combine mode is enabled for reads, indicates that this data channel is the next FIFO to be read.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI1</td><td>RFR0–RFR1</td><td>—</td></tr><tr><td>SAI2</td><td>—</td><td>RFR0</td></tr><tr><td>SAI3</td><td>—</td><td>RFR0</td></tr><tr><td>SAI4</td><td>RFR0–RFR3</td><td>—</td></tr></table>
0b - No effect.
1b - FIFO Combine mode is enabled for FIFO reads and this FIFO will be read on the next FIFO read.</td></tr>
<tr><td>14-6
—</td><td>Reserved</td></tr>
<tr><td>5-0
RFP</td><td>Read FIFO Pointer
FIFO read pointer for receive data channel.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SAI1</td><td>RFR0–RFR1[4–0]</td><td>RFR0–RFR1[5]</td></tr><tr><td>SAI2</td><td>RFR0</td><td>—</td></tr><tr><td>SAI3</td><td>RFR0</td><td>—</td></tr><tr><td>SAI4</td><td>RFR0–RFR3</td><td>—</td></tr></table></td></tr>
</table>

51.6.1.21 Receive Mask (RMR)

Offset

Register	Offset
RMR	E0h

Function

This register is double-buffered and updates:

1. When RCSR[RE] first becomes 1.
2. At the end of each frame.

This setup allows the masked words in each frame to change from frame to frame.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RWM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RWM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RWM	Receive Word Mask Configures whether the receive word is masked (received data ignored and not written to receive FIFO) for the corresponding word in the frame. 0000_0000_0000_0000_0000_0000_0000_0000b - Word N is enabled. 0000_0000_0000_0000_0000_0000_0000_0001b - Word N is masked.

Chapter 52
Sony/Philips Digital Interface (SPDIF)

52.1 Chip-specific SPDIF information

Table 341. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

Available clock sources in SPDIFTxClk[TxClk_Source]:

Table 342. Clock sources

Bitfield	Description
10-8 TxClk_Source	TxClk_Source 000b - extal_clk (from OSC24M) 001b - tx_clk 010b - tx_clk1 011b - tx_clk2 100b - tx_clk3 101b - ipg_clk input (frequency divided) 110b - tx_clk4 111b - from clk_root_mic

52.2 Overview

The Sony/Philips Digital Interface (SPDIF) audio block is a stereo transceiver that allows the processor to receive and transmit digital audio.

52.2.1 Block diagram

The figure below shows a block diagram of the SPDIF transceiver data paths (receiver and transmitter) and its interface.

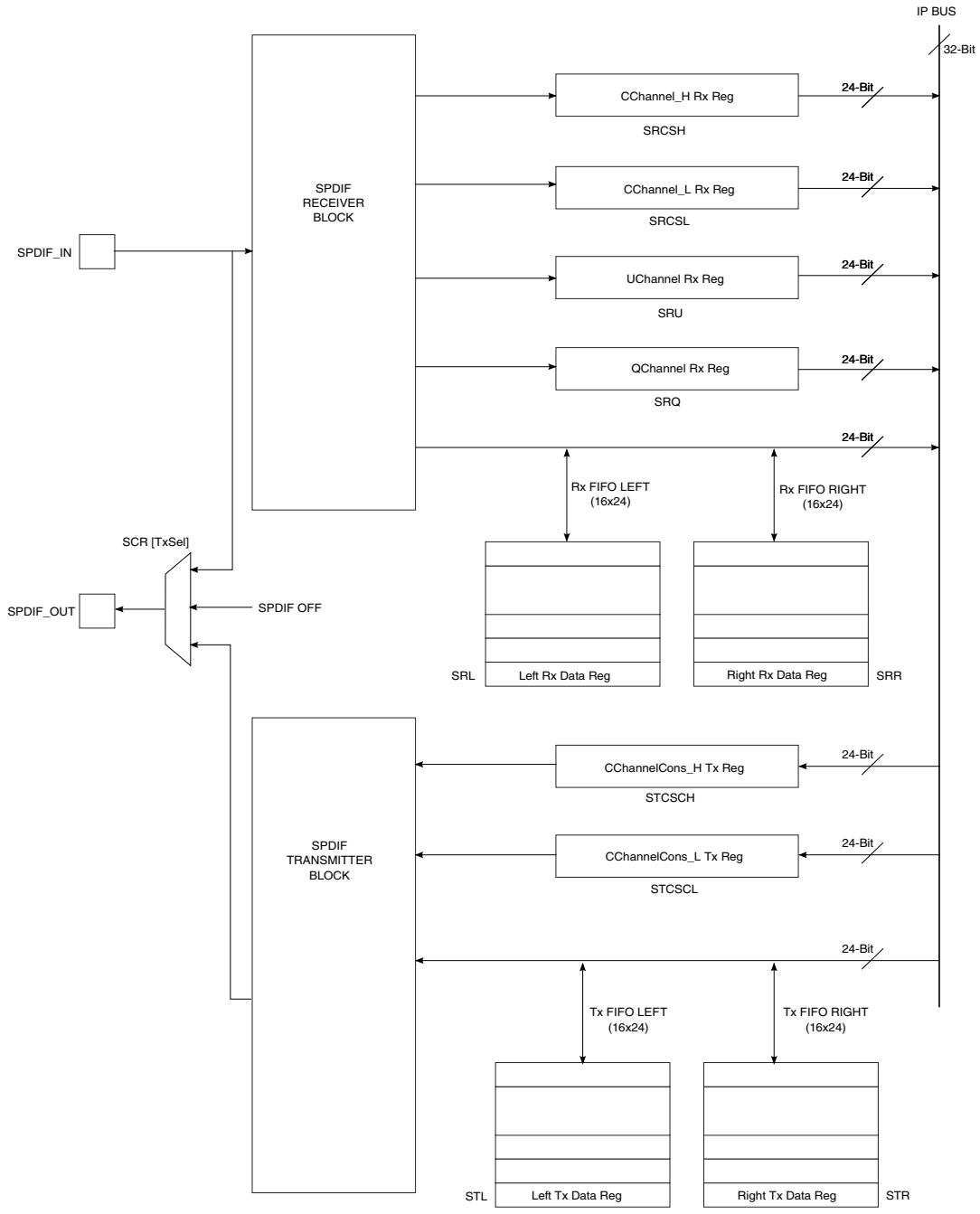


Figure 306. SPDIF transceiver data interface block diagram

52.2.2 Features

- Allows the handling of both SPDIF channel status (CS) and User (U) data
- Includes a frequency measurement block that allows the precise measurement of an incoming sampling frequency

- Provides a recovered clock to drive both internal components in the system, such as SAI ports, and external components, such as A/Ds or D/As, with clocking control provided via related registers

52.3 Functional description

The SPDIF is composed of two parts: SPDIF Receiver and SPDIF Transmitter.

The SPDIF receiver extracts the audio data from each SPDIF frame and places the data in the SPDIF Rx left and right FIFOs. The Channel Status and User Bits are also extracted from each frame and placed in the corresponding registers. The SPDIF receiver also provides a bypass option for direct transfer of the SPDIF input signal to the SPDIF transmitter.

For the SPDIF transmitter, the audio data is provided by the processor via the SPDIFTxLeft and SPDIFTxRight registers. The Channel Status bits are also provided via the corresponding registers. The SPDIF transmitter generates a SPDIF output bitstream in the biphase mark format (IEC60958), which consists of audio data, channel status and user bits.

In the SPDIF transmitter, the IEC60958 biphase bit stream is generated on both edges of the SPDIF Transmit clock. The SPDIF Transmit clock is generated by the SPDIF internal clock generate block and the sources are from outside of the SPDIF block. For the SPDIF receiver, it can recover the SPDIF Rx clock.

The Tx clock is sent to the ASRC.

The Rx clock is sent to the ASRC.

NOTE

ASRC (Asynchronous Sample Rate Converter) is not part of SPDIF. See the dedicated ASRC module for more information.

Figure 307 shows the clock structure of the SPDIF transceiver.

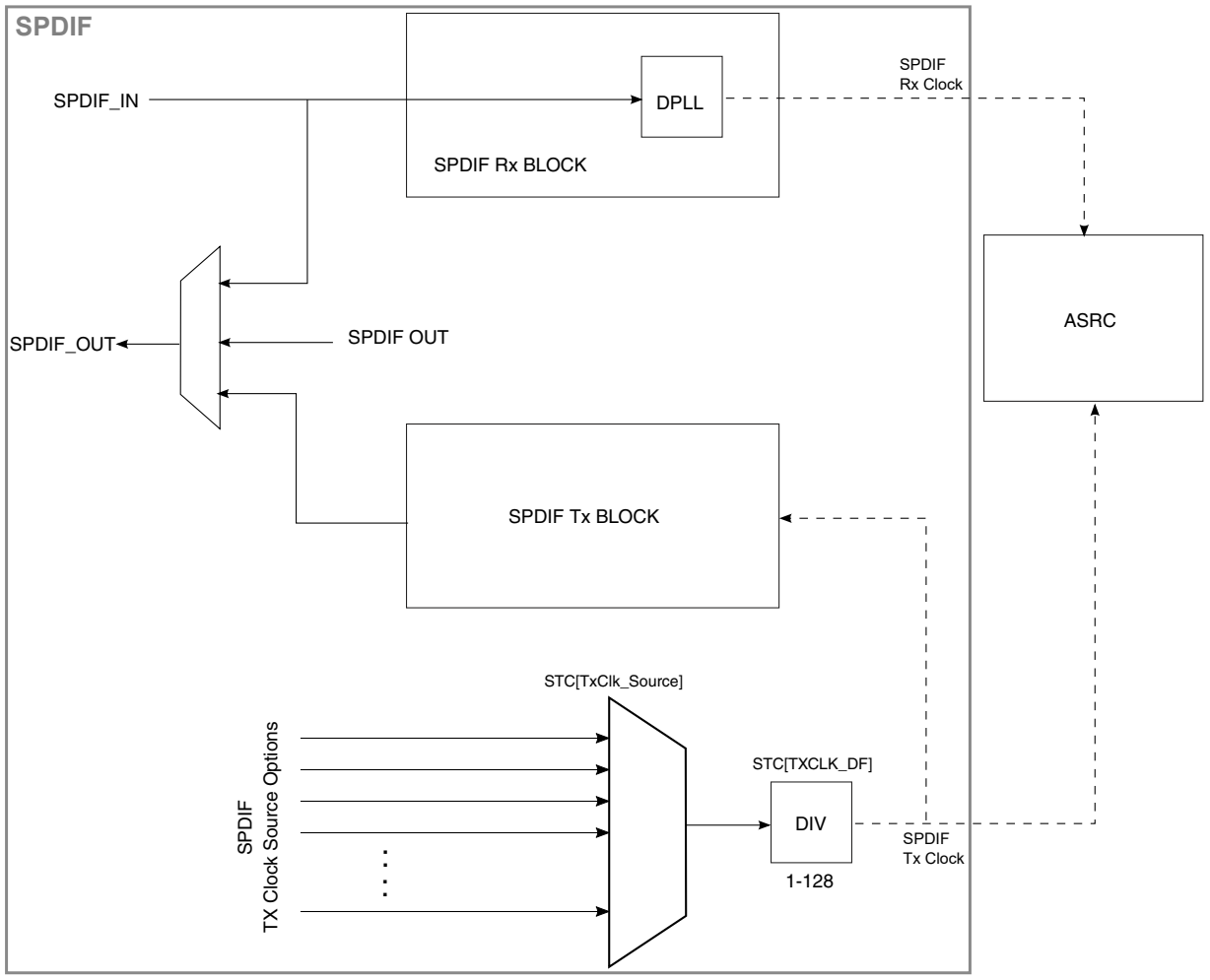


Figure 307. SPDIF transceiver clock diagram

52.3.1 SPDIF receiver

The SPDIF receiver extracts the audio data from each SPDIF frame and places the data in Rx left and right FIFOs.

The Tx left and right FIFOs are 16-deep and 24-bit-wide (equal to the audio data width). The Channel Status and User Bits are also extracted from each frame and placed in corresponding registers. The SPDIF receiver also provides a bypass option for direct transfer of the SPDIF input signal to the SPDIF transmitter.

The SPDIF receiver handles the main data audio stream and recovers the bit clock from the SPDIF input signal. The sample rate can be determined from the frequency measuring block. Additionally, the receiver supports the SPDIF C and U channels. The SPDIF C and U channel data is interfaced directly to memory-mapped registers.

All the data registers are controlled by the Interrupt Control Block and transferred to the memory-mapped IP bus.

52.3.1.1 Audio data reception

The SPDIF Receiver block extracts the audio data from the IEC60958 stream, and outputs this via Rx left and right FIFOs to the memory-mapped registers SPDIFRxLeft and SPDIFRxRight.

Data from the SPDIF receiver is buffered in receive FIFO, and can be read by the processor from the memory-mapped registers.

SPDIF receiver data registers - Behavior on overrun, underrun

The SPDIF Data Receive registers (SPDIFRxLeft and SPDIFRxRight) have individual FIFOs for left and right channel. As a result, there is always the possibility that left and right FIFOs may go out of sync due to FIFO underruns and FIFO overruns that affect only one part (left or right) of any FIFOs. To prevent this from happening, hardware has been added to the device. Two mechanisms to prevent mismatch between the FIFOs are available.

If a SPDIF Data Rx FIFO overrun occurs on e.g. the right half of the FIFO, the sample that caused the overrun is not written to the right half (due to overrun). Special hardware will make sure the next sample is not written to the left half of the FIFO. If the overrun occurs on the left half of the FIFO, the next sample is not written to the right half of the FIFO.

SPDIF receiver data registers - Automatic resynchronization of FIFOs

An automatic FIFO resynchronization feature is available. It can be enabled and disabled separately for every FIFO. If it is enabled, the hardware will check to see if the left and right FIFOs are in sync. If that is not the case, it will set the filling pointer of the right FIFO to be equal to the filling pointer of the left FIFO.

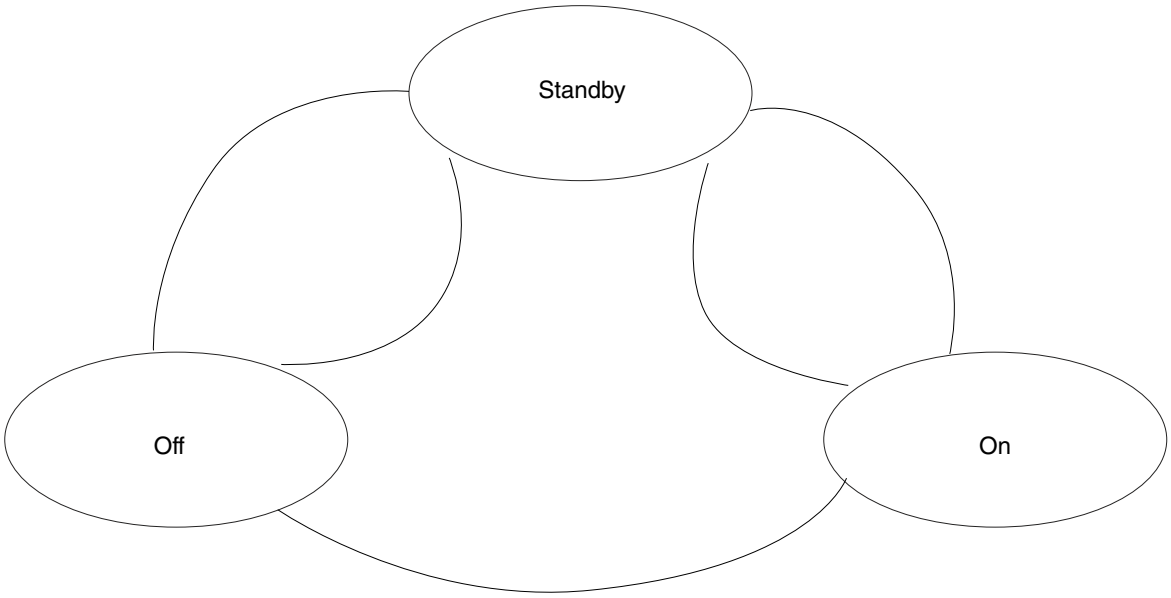


Figure 308. FIFO auto-resync controller state machine

The operation is explained from the state diagram shown above. Every FIFO auto-resync controller has a state machine with 3 states: Off, StandBy and On. In the On state, the filling of the left FIFO is compared with the filling of right, and if they are not equal, right is made equal to left, and an interrupt is generated.

The controller will stay in Off state when the feature is disabled. When not disabled, the state machine will go to Off state on any processor read or write to the FIFO. It will go from On or Off to Standby on any left sample read from SPDIF Tx FIFOs, or on any left sample write to SPDIF Rx FIFOs. The controller will go from Standby to On on any right sample read from SPDIF Tx FIFO, or on any right sample write to SPDIF Rx FIFO. There is a control bit in the SPDIF Configuration Register (SCR) to enable/disable the feature for the SPDIF Rx FIFO and SPDIF Tx FIFO.

See Application information for additional details.

52.3.1.2 Channel status reception

There are 2 modes of Channel Status Reception: 48-bit and 192-bit.

48-bit mode

In 48-bit mode, a total of 48 channel status bits are received in two registers. No interpretation is performed by the SPDIF receiver block.

Channel Status Bits are ordered first bit left. CS-channel MSB bit "0" is located in bit position 23 in the memory-mapped register SPDIFRxCChannel_h. CS-channel bit "23" is considered the LSB bit 0 in the register. C-channel bit 24 to 47 is seen as [23:0] bits of register SPDIFRxCChannel_l.

192-bit mode

In 192-bit mode, set RXCChannel_192b_en located in the SPDIF Configuration Register bit[25] to "1". A total of 192 channel status bits are received in six registers.

Channel Status Bits are ordered first bit left.

- CS-channel MSB bit "0" is located in bit position 31 in the memory-mapped register SPIDFRxCChannel_Addr_31_0, CS-channel bit "31" is the LSB bit 0 in this register
- C-channel bit 32 to 63 is seen as [31:0] bits of register SPDIFRxCChannel_Addr_63_32
- C-channel bit 64 to 95 is seen as [31:0] bits of register SPDIFRxCChannel_Addr_95_64
- C-channel bit 96 to 127 is seen as [31:0] bits of register SPDIFRxCChannel_Addr_127_96
- C-channel 128 to 159 is seen as [31:0] bits of register SPDIFRxCChannel_Addr_159_128
- C-channel 160 to 191 is seen as [31:0] bits of register SPDIFRxCChannel_Addr_191_160

52.3.1.2.1 Channel status interrupt

When the value of a new SPDIF "CS" channel status frame is loaded in the register, an interrupt is generated. The interrupt is cleared when the processor writes the corresponding bit in the InterruptStat register.

52.3.1.3 User bit reception

There are two modes for U Channel reception, CD and non-CD. As is decided by USyncMode (bit 1 of CDText_Control register).

Behavior of U Channel receive interface on incoming CD U Channel Sub-code in SPDIF receiver

This mode is selected if UsyncMode, bit 1 in register CD Text control is set "1".

The CD sub-code stream embedded into the SPDIF U channel consists of a sequence of packets. Every packet is made up 98 "symbols". The first two symbols of every packet are "sync symbols", the other 96 symbols are "data symbols".

Any sequence found in the SPDIF U channel stream starting with a leading one, followed by 7 information bits, is recognized as a "data symbol". Subsequent data symbols are separated by "pauses". During the "pause", "zero bits" are seen on the SPDIF U channel.

Data symbols are coming in MSB first. The MSB is the leading one.

When a "long pause" is seen between 2 subsequent "data symbols", the SPDIF receiver will assume the reception of one or more "sync symbols". Table below gives details.

Table 343. Sync control bits

Number of U Channel zero bits	Corresponding number of sync symbols
0-1	Unpredictable, not allowed
2-10	0

Table continues on the next page...

Table 343. Sync control bits (continued)

Number of U Channel zero bits	Corresponding number of sync symbols
11-22	1
23-34	2
35-46	3
>45	Unpredictable, not allowed

The recognition of the number of sync symbols derives from the fact that the U channel transmitter in the CD channel decoder will transmit one symbol on average every 12 SPDIF channel bits. On this average rate, there is a maximum tolerance of 5%.

The SPDIF receiver is tolerant of symbol errors. Due to the physical nature of the transmission of the data over the CD disc, not more than 1 out of any 5 consecutive user channel symbols may be in error. The error may cause a change in data value, which is not detected by this interface, or it may cause a data symbol to be seen as a sync symbol, or a sync symbol to be seen as a data symbol. However, not more than 1 out of any 5 consecutive user channel symbols should be affected in this way.

The SPDIF U channel circuitry recognizes the 98-symbol packet structure, and sends the 96 symbol payload to the processor application. The 96 symbol payload is transmitted to the processor via 2 registers:

- The SPDIFRxUChannel register. In this register, data is presented 3 symbols at the time to the processor. Every time 3 new valid symbols, received on the SPDIF U Channel are present, the UChannelRxFull interrupt is asserted. For one 98-symbol packet, 96 symbols are carried across SPDIFRxUChannel. To transfer all this data, 32 UChannelRxFull interrupts are generated.
- The QChannelReceive register. In this register, only the Q bit of the packet is accumulated. Operation is similar to UChannelReceive. Because only Q-bit is transferred, only 96 Q-bits are transferred for any 98-symbol packet. To transfer this data, 4 QChannelRxFull interrupts are generated. When QChannelRxFull occurs, it is coincident with UChannelRxFull. There is only one QChannelRxFull for every 8 UChannelRxFull. The convention is that most significant data is transmitted first, and is left-aligned in the registers.
- Timing regarding packet boundary is extracted by hardware. The last UChannelRxFull corresponding to a given packet should be coincident with the last QChannelRxFull. In this last U, Q channel interrupt, symbols 95-98 are received, Q channel bits 67-98. The interrupts are coincident with UQSyncFound, flagging last symbols of the current frame.
- When the start of the new packet is found before the current packet is complete (less than 98 symbols in the packet), the UQFrameError interrupt is set. The application software should read out UChannelReceive and QchannelReceive registers, discard the value, and assume the start of a new packet.
- As already said, packet sync extraction is tolerant for single-symbol errors. Packet sync detection is based on the recognition of the sequence data-sync-sync-data in the symbol stream, because this is the only syncing sequence that is not affected by single errors. If the sync symbols are not found 98 symbols after the previous occurrence, it is assumed to be destroyed by channel error, and a new sync symbols is interpolated.
- Normally, only data bytes are passed to the application software. Every databyte will have its most significant bit set. If sync symbols are passed to the application software, they are seen as all-zero symbols. Sync symbols can only end up in the data stream due to channel error.

Behavior of U Channel receive interface on incoming non-CD data

This mode is selected if UsyncMode, bit 1 in register CD Text control is set'0'.

In non-CD mode, the SPDIF U channel stream is recognized as a sequence of "data symbols". No packet recognition is done.

Any sequence found in the SPDIF U channel stream starting with a leading one, followed by 7 information bits, is recognized as a "data symbol". Subsequent data symbols are separated by "pauses". During the "pause", "zero bits" are seen on the SPDIF U channel.

3 consecutive data symbols seen in the SPDIF U Channel stream are grouped together into the SPDIFRxUChannel register. First symbol is left, last symbol is right aligned. When SPDIFRxUChannel contains 3 new data symbols, UChannelRxFull is asserted.

In this mode, the operation of QchannelRx and associated interrupt QchannelRxFull is reserved, undefined. And the operation of UQFrameError and UQSyncFound is also reserved, undefined.

The U channel is extracted, and output by the SPDIF Rx on SPDIFRxUChannel-Stream.

When incoming SPDIF data parity error or bit error is detected, and if the next SPDIF word for that channel is error-free, the SPDIF word in error is replaced with the average of the previous word and next word. When incoming SPDIF data parity error or bit error is detected, and the next SPDIF word is in error, the previous SPDIF word is repeated.

52.3.1.4 Validity flag reception

An interrupt is associated with the Validity flag. (interrupt 16 - SPDIFValNoGood). This interrupt is set every time a frame is seen on the SPDIF interface with the validity bit set to "invalid".

52.3.1.5 SPDIF receiver interrupt exception definition

Several SPDIF exceptions can trigger an interrupt. They are:

- Control Status channel change. Set when SPDIFRxCChannel_l register is updated. The register is updated for every new C-Channel received. The exception is reset on write to SIS register.
- SPDIF Illegal Symbol. Set on reception of illegal symbol during SPDIF receive. Reset by writing register InterruptClear.[5]
- SPDIF bit error. Set on reception of bit error. (Parity bit does not match). Reset on write to InterruptClear register.
- Receive data FIFO full. Set when SPDIF receive data FIFO is full.
- Receive data FIFO underrun/overrun. Set when there is a underrun/overrun on the SPDIF receive data FIFO.
- Receive data FIFO resynchronization. Set when a resynchronization event occurs on the SPDIF receive data FIFO.
- Receive U Channel buffer full. Set when next 24 bits of U channel code are available.
- Receive Q Channel buffer overrun. Set when Q channel buffer overrun.
- Receive U Channel buffer overrun. Set on U channel buffer overrun.
- Receive Q Channel buffer full. Set when next 24 bits of Q channel code are available.
- Receive UQ sync found. Set when UQ channel sync found.
- Receive UQ frame error. Set when UQ frame error found.

52.3.1.6 Standards compliance

The SPDIF interface is compatible with the Tech 3250-E standard of the European Broadcasting Union, except clause 6.3.3 and the IEC60958-3 Ed2 for relevant topics.

Supported input frequency range is 12 kHz up to 96 kHz. (fully compliant) and 96 kHz up to 176 kHz (Can interface with compliant SPDIF transmitter within same cabinet, making reasonable assumptions on jitter added due to interconnecting wire.)

Tolerated jitter on SPDIF input signals are 0.25 bit peak-peak for high frequencies. There is no jitter limit for low frequencies. The user channel extraction in CD mode is capable of coping with single-symbol errors, and still retrieve U channel frames on correct boundaries. This capability is required for reliable reception of CD-Text from some Philips CD channel decoders. This capability was deemed more important than compliance with the IEC60958 annex A.3 standard, and for this reason user channel reception

[5] The SPDIF input is a biphase/mark modulated signal. The time between any two successive transitions of the SPDIF signal is always 1, 2 or 3 SPDIF symbol periods long. The SPDIF receiver will parse the stream, and split it in so-called symbols. It recognizes s1, s2 and s3 symbols, depending on the length of the symbols. Not all sequences of these symbols are allowed. To give an example, a sequence s2-s1-s1-s1-s2 cannot occur in a no-error SPDIF signal. If the receiver finds such an illegal sequence, the illegal symbol interrupt is set. No corrective action is undertaken. When the interrupt occurs, this means that(a) The SPDIF signal is destroyed by noise (b) The SPDIF frequency changed.

is not compliant with IEC60958 annex A.3. However, the interface is capable to receive U channel inserted by a typical CD channel decoder. Also, in this case, it is more robust and tolerant for channel error than what is required by IEC60958 annex A.3.

52.3.1.7 SPDIF PLOCK detection and Rxclk output

Using the high speed system clock, the internal DPLL can extract the bit clock (advanced pulse) from the input bitstream. When this internal DPLL is locked, the LOCK bit of PhaseConfig Register will be set, and the SPDIF Lock output pin SPDIF_LOCK will be asserted.

After DPLL has locked, the pulses are generated, and the average pulse rate is 128 x the sampling frequency.(For a 44.1 kHz input sampling frequency, the average pulse rate = 128 x 44.1 kHz.) The pulse signal is used in the FreqMeas circuit to generate the frequency measurement result.

52.3.1.8 Measuring frequency of SPDIF_RxClk

The internal DPLL can extract the bit clock (advanced plus) from the input bitstream. To do that, it is necessary to measure the frequency of the incoming signal in relationship with the system clock (BUS_CLK).

Associated with it are two registers, PhaseConfig and FreqMeas. The circuit will measure the frequency of the incoming clock as a function of the BUS_CLK. The circuit is a second-order filter. The output is a value represented by an unsigned number stored in the 24-bit FreqMeas register, giving the frequency of the source as a function of the BUS_CLK.

FreqMeas[23:0] = FreqMeas_CLK / BUS_CLK * 2^{10} * GAIN.

For example, if the GAIN is selected as 8*(2**10) (PhaseConfig[5:3] = 3'b011), the actual result

FreqMeas_CLK / BUS_CLK is equal to FreqMeas[23:0] / 2^{23}.

52.3.2 SPDIF transmitter

Audio data for the SPDIF transmitter is provided by processor via the SPDIFTxLeft and SPDIFTxRight registers.

Clocking for SPDIF transmitter is selected through a multiplexer from several clock sources (see register bitfield description STC[TxClk_Source] for transmit clock source inputs). The SPDIF transmitter clock source can be divided down as needed using Txclk_DF. The SPDIF transmitter output can be chosen from either the SPDIF transmitter block, directly from the SPDIF receiver (via the output multiplexer), or disabled.

The SPDIF transmitter generates a SPDIF output bitstream in IEC60958 biphase mark format, consisting of audio data, channel status.

52.3.2.1 Audio data transmission

Audio data for the SPDIF transmitter is provided by the processor via SPDIFTxLeft and SPDIFTxRight registers. They send audio data to Tx left and right FIFOs. The Tx left and right FIFOs are also 16-deep and 24-width (equal to the audio data width).

SPDIF transmitter data registers - Behavior on overrun, underrun

The SPDIF Data Transmit registers (SPDIFTxLeft and SPDIFTxRight) have individual FIFOs for left and right channel. As a result, there is always the possibility that left and right FIFOs may go out of sync due to FIFO underruns and FIFO overruns that affect only one part (left or right) of any FIFO. To prevent this from happening, hardware has been added on the device. Two mechanisms to prevent mismatch between the FIFOs are available.

If SPDIF Tx FIFO underruns on the right half of the FIFO, no sample leaves that FIFO (because it was already empty). Special hardware will make sure that the next sample read from the left FIFO will not leave the FIFO (no read strobe is generated). If the underrun occurs on the left half of the FIFO, next read strobe to the right FIFO is blocked.

SPDIF transmitter data registers - Automatic resynchronization of FIFOs

See Audio data reception and Application information.

52.3.2.2 Channel status transmission

There are 2 modes of Channel Status Transmission: 48-bit and 192-bit.

48-bit mode

In 48-bit mode, a total of 48 Consumer channel status bits are transmitted from two registers. Channel Status Bits are ordered first bit left.

CS-channel MSB bit "0" is located in bit position 23 in the memory-mapped register SPDIFTxCChannelCons_h. CS-channel bit "23" is considered bit 0 in the register. C-channel bits 24-47 are seen as MSB-LSB bits of register SPDIFTxCChannelCons_l.

192-bit mode

In 192-bit mode, a total of 192 Consumer channel status bits are transmitted from six registers. Channel Status Bits are ordered first bit left.

Channel Status Bits are ordered first bit left.

- CS-channel MSB bit "0" is located in bit position 31 in the memory-mapped register SPDIFTxCChannel_Addr_31_0, CS-channel bit "31" is bit 0 in this register
- CS-channel bit 32 to 63 is seen as [31:0] bits of register SPDIFTxCChannel_Addr_63_32
- C-channel bit 64 to 95 is seen as [31:0] bits of register SPDIFTxCChannel_Addr_95_64
- C-channel bit 96 to 127 is seen as [31:0] bits of register SPDIFTxCChannel_Addr_127_96
- C-channel 128 to 159 is seen as [31:0] bits of register SPDIFTxCChannel_Addr_159_128
- C-channel 160 to 191 is seen as [31:0] bits of register SPDIFTxCChannel_Addr_191_160

52.3.2.3 Validity flag transmission

The validity bit setting is performed via the ValCtrl bit of the SPDIF_SCR register.

52.3.3 Clocking

The table found here describes the clock sources for SPDIF.

Please see clock control block for clock setting, configuration and gating information.

Table 344. SPDIF clocks

Clock name	Description
gclkw_t0	Global clock
ipg_clk_s	Peripheral access clock
tx_clk	Module Tx clock

52.4 External signals

The following table describes the external signals of SPDIF:

Table 345. SPDIF external signals

Signal	Description	Direction
SPDIF_EXT_CLK	External clock signal	I
SPDIF_IN	Input line	I
SPDIF_LOCK	Lock signal	O
SPDIF_OUT	Output line signal	O
SPDIF_SR_CLK	RX Clock	O
SPDIF_OUT_CLK	TX Clock	O

52.5 Application information

The automatic FIFO resynchronization can be switched on, and will avoid all mismatches between left and right FIFOs, if the software obeys the following rules:

1. When the left data is read or written to the left FIFO, in the same place of the program, data must be read or written to the right FIFO. Maximum time difference between left and right is 1/2 sample clock. (E.g. if sample frequency is 44 kHz, approximately 10 micro-seconds. For 88 kHz, approximately 5 micro-seconds.)
2. Write/read data to FIFOs at least 2 samples at the time. If there is a mismatch Left-Right, the resync logic may go on only 1 sample clock after last data is read/written to the FIFO. Also acceptable is polling the FIFO, if at least part of the time 2 samples will be read/written to it.
3. After DPLL lock, the user must bypass the first 2 CNEW interrupt/events, and start sampling data from the 3rd CNEW event. The user must do the following:
   - Sample data must be discarded for a minimum of 128 to a maximum of 384 samples.
   - Clear all error flags after the 3rd CNEW event flag.

SPDIF receiver details

There are three exceptions associated with the SPDIF Receiver FIFOs

- full
- under/overrun
- resync

When the "full" condition is set for processor data input registers, the processor should read data from the FIFO, before overrun occurs. When "full" is set, and the FIFO contains e.g. 6 samples, it is acceptable for the software to read first 6 samples from the LEFT address, followed by 6 samples from the RIGHT address, or 6 samples from the RIGHT address, followed by 6 samples from the LEFT address, or 1 sample LEFT, followed by 1 sample RIGHT repeated 6 times. There is no order specified.

The implementation for SPDIF Rx is a double FIFO, one for left and one for right. "full" is set when both FIFOs are full. "underrun, overrun" are set when one of the FIFOs do underrun or do overrun. The resync interrupt means hardware took special action to resynchronize left and right FIFOs.

The FIFO level at which the "full" interrupt is generated, is programmable via the Full Select field in the SPDIFConfigReg register.

Rx FIFO on and Rx FIFO reset

Two additional control fields of the SPDIF Rx FIFO are the on/off select and FIFO reset fields.

If on/off select is set to off, all-zero will be read from the FIFO, irrespective of the data received over the SPDIF interface.

If FIFO reset is set, the FIFO is blocked at "1 sample in FIFO". In this, the full interrupt will be on if FullSelect is set to "00". If FullSelect is set to any other value, interrupt will be off. The other interrupts are always off.

SPDIFTxLeft, SPDIFTxRight details

With SPDIF Tx FIFOs three exceptions are associated:

- empty
- under/overrun
- resync

When the empty condition is set for processor data output registers, the processor should write data to the FIFO, before underrun occurs. When empty is set and, for instance, 6 samples need to be written, it is acceptable for the software to write first 6 samples from the LEFT address, followed by 6 samples from the RIGHT address, or 1 sample LEFT, followed by 1 sample RIGHT repeated 6 times. Left should be written before right. The implementation of all data out FIFOs is a double FIFO, one for left and one for right. Empty is set when both FIFOs are empty. Underrun, overrun are set when one of the FIFOs do underrun or do overrun. Resync is set when the hardware resynchronizes left and right FIFOs.

On receiving underrun, overrun interrupt, synchronization between Left and Right words in the FIFOs may be lost. Synchronization will not be lost when the underrun or overrun comes from the IEC60958 side of the FIFO. If the processor reads or writes more data from, for example, left than from right, synchronization will be lost. If automatic resynchronization is enabled, and if the software obeys the rules to let this work, resynchronization will be automatic.

52.6 SPDIF memory map/register definition

52.6.1 SPDIF register descriptions

52.6.1.1 SPDIF memory map

SPDIF base address: 42BA_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	SPDIF Configuration Register (SCR)	32	RW	0000_0400h
4h	CDText Control Register (SRCD)	32	RW	0000_0000h
8h	PhaseConfig Register (SRPC)	32	RW	0000_0000h
Ch	InterruptEn Register (SIE)	32	RW	0000_0000h
10h	InterruptStat Register (SIS)	32	RW	0000_0002h
14h	SPDIFRxLeft Register (SRL)	32	R	0000_0000h
18h	SPDIFRxRight Register (SRR)	32	R	0000_0000h
1Ch	SPDIFRxCChannel_h Register (SRCSH)	32	R	0000_0000h
20h	SPDIFRxCChannel_l Register (SRCSL)	32	R	0000_0000h
24h	UchannelRx Register (SRU)	32	R	0000_0000h
28h	QchannelRx Register (SRQ)	32	R	0000_0000h
2Ch	SPDIFTxLeft Register (STL)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
30h	SPDIFTxRight Register (STR)	32	RW	0000_0000h
34h	SPDIFTxCChannelCons_h Register (STCSCH)	32	RW	0000_0000h
38h	SPDIFTxCChannelCons_l Register (STCSCL)	32	RW	0000_0000h
44h	FreqMeas Register (SRFM)	32	R	See section
50h	SPDIFTxClk Register (STC)	32	RW	0002_0F00h
60h	SPDIF receive C channel register, bits 31-0 (SPDIFRxCChannel_Addr_31_0)	32	R	0000_0000h
64h	SPDIF receive C channel register, bits 63-32 (SPDIFRxCChannel_Addr_63_32)	32	R	0000_0000h
68h	SPDIF receive C channel register, bits 95-64 (SPDIFRxCChannel_Addr_95_64)	32	R	0000_0000h
6Ch	SPDIF receive C channel register, bits 127-96 (SPDIFRxCChannel_Addr_127_96)	32	R	0000_0000h
70h	SPDIF receive C channel register, bits 159-128 (SPDIFRxCChannel_Addr_159_128)	32	R	0000_0000h
74h	SPDIF receive C channel register, bits 191-160 (SPDIFRxCChannel_Addr_191_160)	32	R	0000_0000h
78h	SPDIF transmit C channel register, bits 31-0 (SPDIFTxCChannel_Addr_31_0)	32	RW	0000_0000h
7Ch	SPDIF transmit C channel register, bits 63-32 (SPDIFTxCChannel_Addr_63_32)	32	RW	0000_0000h
80h	SPDIF transmit C channel register, bits 95-64 (SPDIFTxCChannel_Addr_95_64)	32	RW	0000_0000h
84h	SPDIF transmit C channel register, bits 127-96 (SPDIFTxCChannel_Addr_127_96)	32	RW	0000_0000h
88h	SPDIF transmit C channel register, bits 159-128 (SPDIFTxCChannel_Addr_159_128)	32	RW	0000_0000h
8Ch	SPDIF transmit C channel register, bits 191-160 (SPDIFTxCChannel_Addr_191_160)	32	RW	0000_0000h

52.6.1.2 SPDIF Configuration Register (SCR)

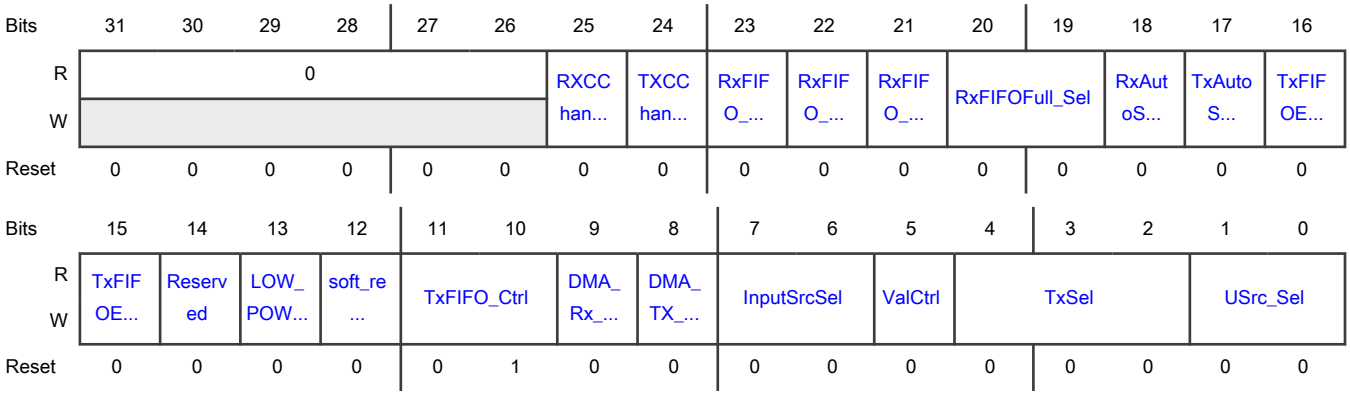
Offset

Register	Offset
SCR	0h

Function

Contains SPDIF configuration selections.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 RXCChannel_192b_en	RXCChannel_192b_en RX CChannel 192bits Enable 0b - SPDIF receives only 48 bits of 192 C bits from input audio stream 1b - SPDIF receives 192 bits of C in audio stream
24 TXCChannel_192b_en	TXCChannel_192b_en TX CChannel 192bits Enable 0b - SPDIF transmits 48 bits of C in audio stream. Other C bits in 49 to 192 frames are 0 1b - SPDIF transmits 192 bits of C in audio stream
23 RxFIFO_Ctrl	RxFIFO_Ctrl 0b - Normal operation 1b - Always read zero from Rx data register
22 RxFIFO_Off_On	RxFIFO_Off_On 0b - SPDIF Rx FIFO is on 1b - SPDIF Rx FIFO is off. Does not accept data from interface
21 RxFIFO_Rst	RxFIFO_Rst 0b - Normal operation 1b - Reset register to 1 sample remaining
20-19	RxFIFOFull_Sel

Table continues on the next page...

Table continued from the previous page...

Field	Function
RxFIFOFull_Sel	00b - Full interrupt if at least 1 sample in Rx left and right FIFOs 01b - Full interrupt if at least 4 sample in Rx left and right FIFOs 10b - Full interrupt if at least 8 sample in Rx left and right FIFOs 11b - Full interrupt if at least 16 sample in Rx left and right FIFO
18 RxAutoSync	RxAutoSync 0b - Rx FIFO auto sync off 1b - RxFIFO auto sync on
17 TxAutoSync	TxAutoSync 0b - Tx FIFO auto sync off 1b - Tx FIFO auto sync on
16-15 TxFIFOEmpty_Sel	TxFIFOEmpty_Sel 00b - Empty interrupt if 0 sample in Tx left and right FIFOs 01b - Empty interrupt if at most 4 sample in Tx left and right FIFOs 10b - Empty interrupt if at most 8 sample in Tx left and right FIFOs 11b - Empty interrupt if at most 12 sample in Tx left and right FIFOs
14 —	- Reserved
13 LOW_POWER	LOW_POWER When write 1 to this bit, it will cause SPDIF enter low-power mode. return 1 when SPDIF in Low-Power mode.
12 soft_reset	soft_reset When write 1 to this bit, it will cause SPDIF software reset. The software reset will last 8 cycles. When in the reset process, return 1 when read. else return 0 when read.
11-10 TxFIFO_Ctrl	TxFIFO_Ctrl 00b - Send out digital zero on SPDIF Tx 01b - Tx Normal operation 10b - Reset to 1 sample remaining 11b - Reserved
9	DMA_Rx_En DMA Receive Request Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
DMA_Rx_En	(RX FIFO full)
8 DMA_TX_En	DMA_TX_En DMA Transmit Request Enable (Tx FIFO empty)
7-6 InputSrcSel	InputSrcSel Input Source Selector 00b - SPDIF_IN 01b-11b - None
5 ValCtrl	ValCtrl 0b - Outgoing Validity always set 1b - Outgoing Validity always clear
4-2 TxSel	TxSel 000b - Off and output 0 001b - Feed-through SPDIF_IN 010b-100b - Reserved 101b - Tx Normal operation - From SPDIF Tx Block 110b,111b - Reserved
1-0 USrc_Sel	USrc_Sel U channel source select 00b - No embedded U channel 01b - U channel from SPDIF receive block (CD mode) 10b - Reserved 11b - U channel from on chip transmitter

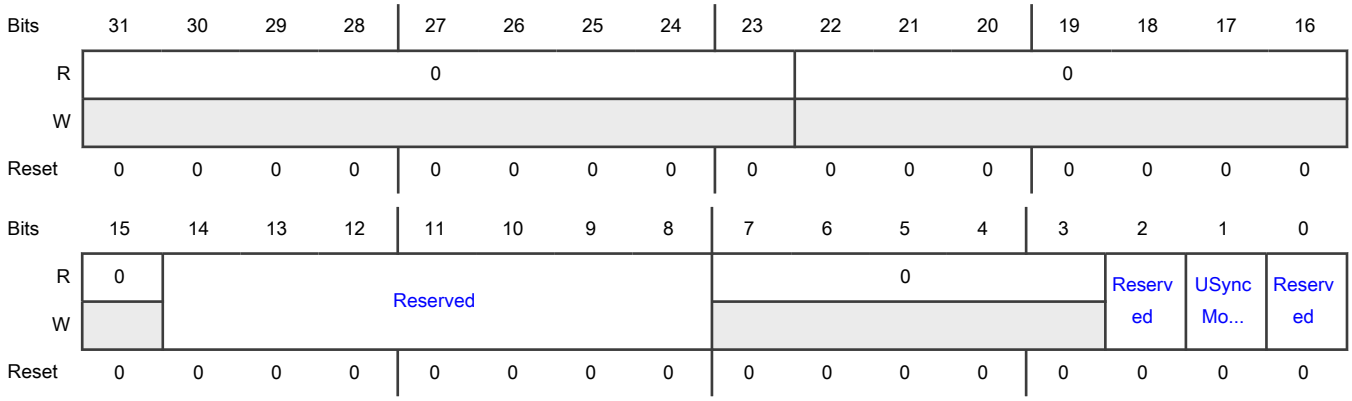
52.6.1.3 CDText Control Register (SRCD)

Offset

Register	Offset
SRCD	4h

Function
Contains CD text controls

Diagram



Fields

Field	Function
31-23 —	This is a 24-bit register the upper byte is unimplemented.
22-15 —	- Return zeros when read
14-8 —	- Reserved. Set to zero.
7-3 —	- Return zeros when read
2 —	- Reserved
1 USyncMode	USyncMode 0b - Non-CD data 1b - CD user channel subcode
0 —	- Reserved

52.6.1.4 PhaseConfig Register (SRPC)

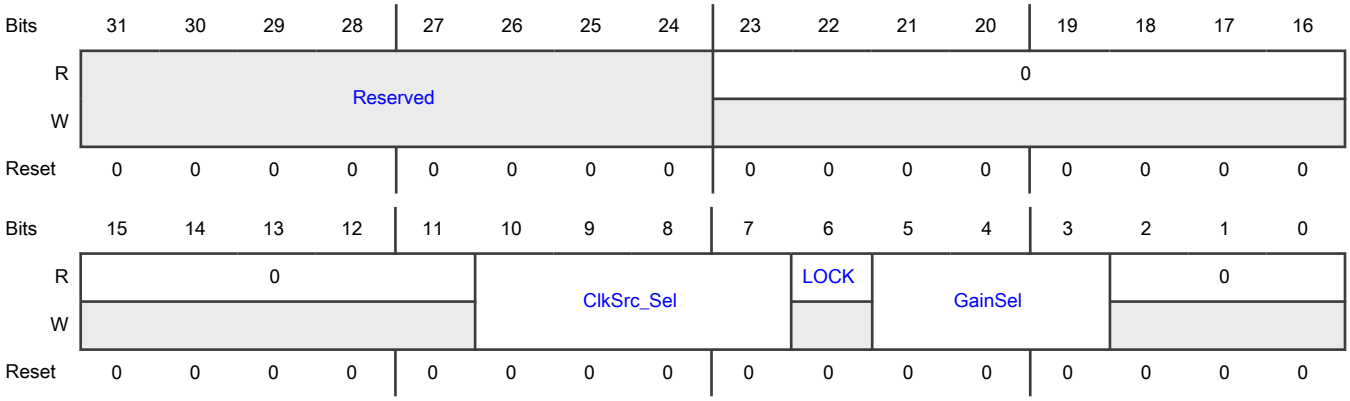
Offset

Register	Offset
SRPC	8h

Function

This register contains clocking and DPLL configuration and status details.

Diagram



Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-11 —	- Reserved, return zeros when read
10-7 ClkSrc_Sel	ClkSrc_Sel Clock source selection, all other settings not shown are reserved: 0000b - if (DPLL Locked) SPDIF_RxClk else REF_CLK_32K (XTALOSC) 0001b - if (DPLL Locked) SPDIF_RxClk else tx_clk (SPDIF0_CLK_ROOT) 0011b - if (DPLL Locked) SPDIF_RxClk else SPDIF_EXT_CLK 0101b - REF_CLK_32K (XTALOSC) 0110b - tx_clk (SPDIF0_CLK_ROOT) 1000b - SPDIF_EXT_CLK
6 LOCK	LOCK LOCK bit to show that the internal DPLL is locked, read only
5-3 GainSel	GainSel Gain selection: 000b - 24*(2**10) 001b - 16*(2**10) 010b - 12*(2**10)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - 8*(2**10) 100b - 6*(2**10) 101b - 4*(2**10) 110b - 3*(2**10) 111b - Reserved
2-0 —	- Reserved, return zeros when read

52.6.1.5 InterruptEn Register (SIE)

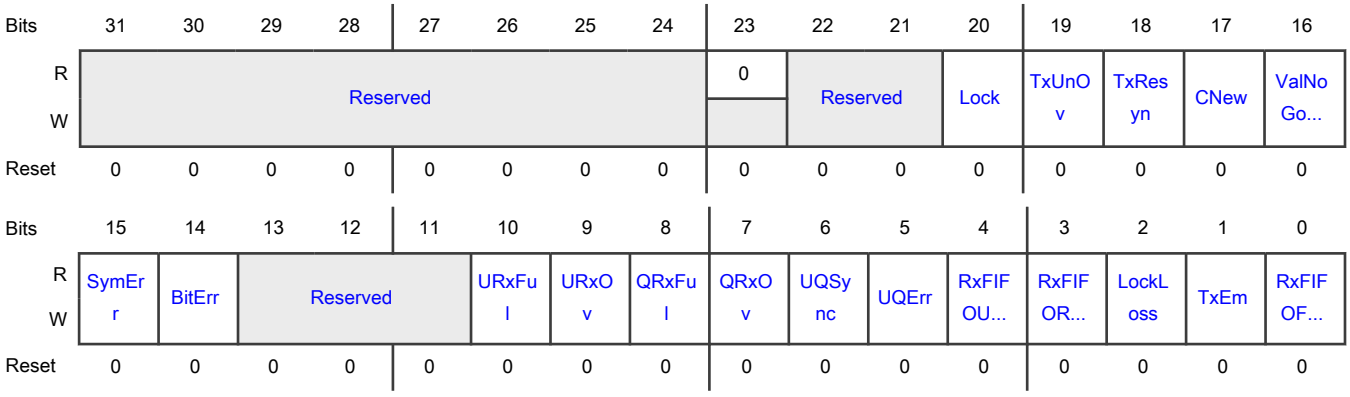
Offset

Register	Offset
SIE	Ch

Function

The InterruptEn register (SPDIF_SIE) provides control over the enabling of interrupts.

Diagram



Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
22-21 —	- Reserved. Set to zero.
20 Lock	Lock SPDIF receiver's DPLL is locked
19 TxUnOv	TxUnOv SPDIF Tx FIFO under/overrun
18 TxResyn	TxResyn SPDIF Tx FIFO resync
17 CNew	CNew SPDIF receive change in value of control channel
16 ValNoGood	ValNoGood SPDIF validity flag no good
15 SymErr	SymErr SPDIF receiver found illegal symbol
14 BitErr	BitErr SPDIF receiver found parity bit error
13-11 —	- Reserved. Set to zero.
10 URxFul	URxFul U Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from U Rx reg.
9 URxOv	URxOv U Channel receive register overrun
8 QRxFul	QRxFul Q Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from Q Rx reg.
7 QRxOv	QRxOv Q Channel receive register overrun
6 UQSync	UQSync U/Q Channel sync found

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 UQErr	UQErr U/Q Channel framing error
4 RxFIFOUnOv	RxFIFOUnOv Rx FIFO underrun/overrun
3 RxFIFOResyn	RxFIFOResyn Rx FIFO resync
2 LockLoss	LockLoss SPDIF receiver loss of lock
1 TxEm	TxEm SPDIF Tx FIFO empty, can't be cleared with reg. IntClear. To clear it, write to Tx FIFO.
0 RxFIFOFul	RxFIFOFul SPDIF Rx FIFO full, can't be cleared with reg. IntClear. To clear it, read from Rx FIFO.

52.6.1.6 InterruptStat Register (SIS)

Offset

Register	Offset
SIS	10h

Function

The InterruptStat (SPDIF_SIS) register is a read only register that provides the status on interrupt operations.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								0			Lock	TxUnOv	TxResyn	CNew	ValNoGo...
W												W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SymErr	BitErr	0			URxFul	URxOv	QRxFul	QRxOv	UQSync	UQErr	RxFIFOU...	RxFIFOR...	LockLoss	TxEm	RxFIFOF...
W	W1C	W1C				W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-21 —	- Reserved, for InterruptStat/Clear return zeros when read.
20 Lock	Lock SPDIF receiver's DPLL is locked
19 TxUnOv	TxUnOv SPDIF Tx FIFO under/overrun
18 TxResyn	TxResyn SPDIF Tx FIFO resync
17 CNew	CNew SPDIF receive change in value of control channel
16 ValNoGood	ValNoGood SPDIF validity flag no good
15 SymErr	SymErr SPDIF receiver found illegal symbol
14 BitErr	BitErr SPDIF receiver found parity bit error
13-11 —	- Reserved. Return zeros when read
10 URxFul	URxFul U Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from U Rx reg.
9 URxOv	URxOv U Channel receive register overrun
8 QRxFul	QRxFul Q Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from Q Rx reg.
7 QRxOv	QRxOv Q Channel receive register overrun
6	UQSync

Table continues on the next page...

Table continued from the previous page...

Field	Function
UQSync	U/Q Channel sync found
5 UQErr	UQErr U/Q Channel framing error
4 RxFIFOUnOv	RxFIFOUnOv Rx FIFO underrun/overrun
3 RxFIFOResyn	RxFIFOResyn Rx FIFO resync
2 LockLoss	LockLoss SPDIF receiver loss of lock
1 TxEm	TxEm SPDIF Tx FIFO empty, can't be cleared with reg. IntClear. To clear it, write to Tx FIFO.
0 RxFIFOFul	RxFIFOFul SPDIF Rx FIFO full, can't be cleared with reg. IntClear. To clear it, read from Rx FIFO.

52.6.1.7 SPDIFRxLeft Register (SRL)

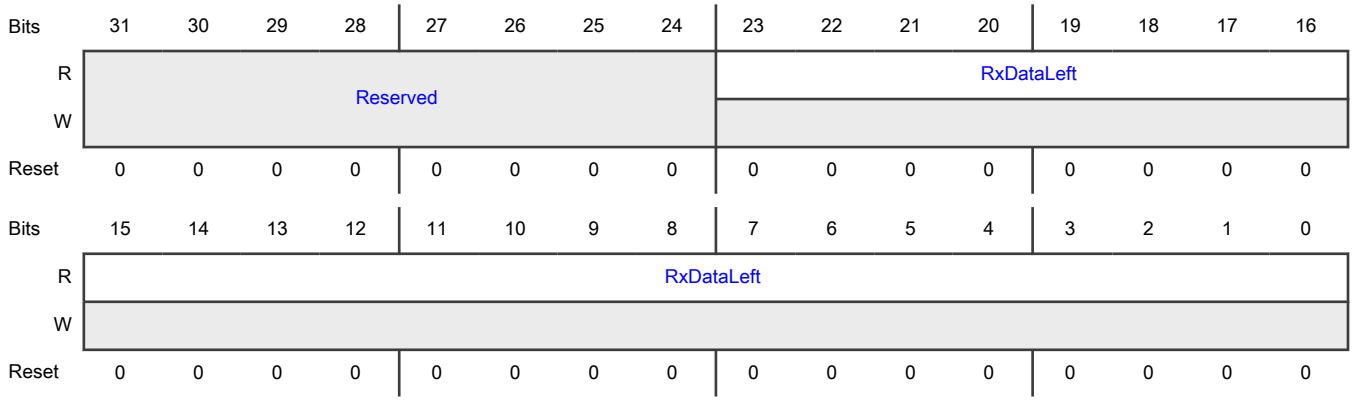
Offset

Register	Offset
SRL	14h

Function

SPDIFRxLeft register is an audio data reception register.

Diagram



Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 RxDataLeft	RxDataLeft Processor receive SPDIF data left

52.6.1.8 SPDIFRxRight Register (SRR)

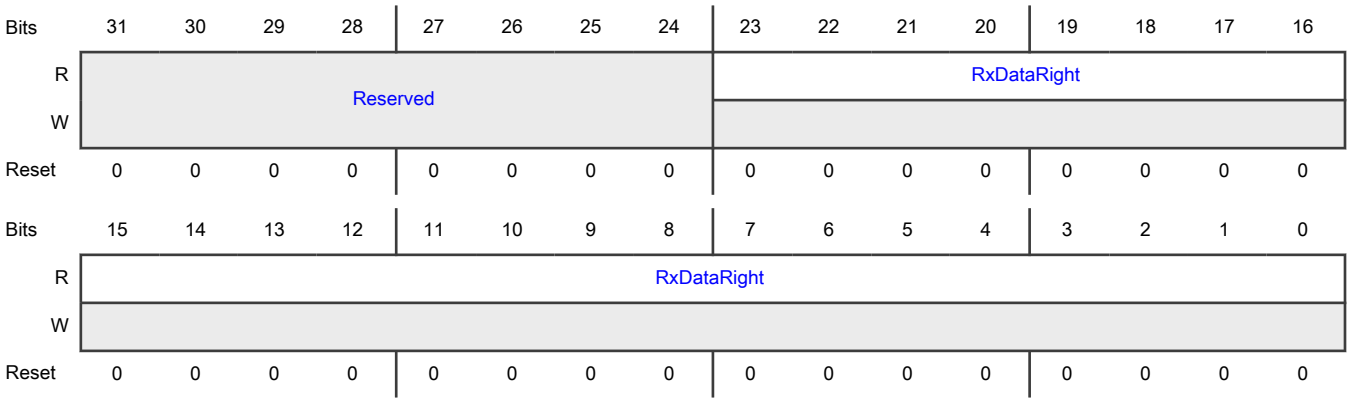
Offset

Register	Offset
SRR	18h

Function

SPDIFRxRight register is an audio data reception register.

Diagram



Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 RxDataRight	RxDataRight Processor receive SPDIF data right

52.6.1.9 SPDIFRxCChannel_h Register (SRCSH)

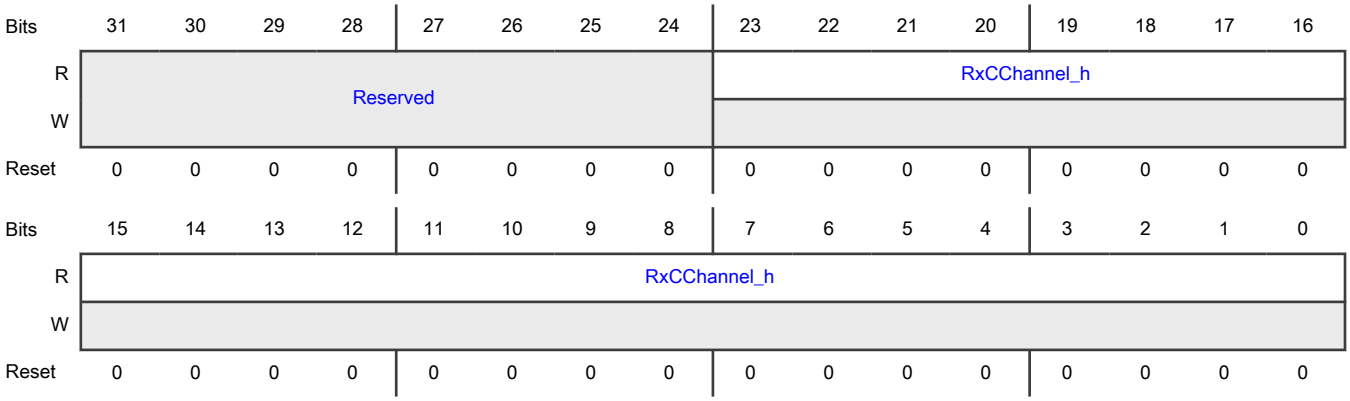
Offset

Register	Offset
SRCSH	1Ch

Function

SPDIFRxCChannel_h register is a channel status reception register.

Diagram



Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 RxCChannel_h	RxCChannel_h SPDIF receive C channel register, contains first 24 bits of C channel without interpretation

52.6.1.10 SPDIFRxCChannel_l Register (SRCSL)

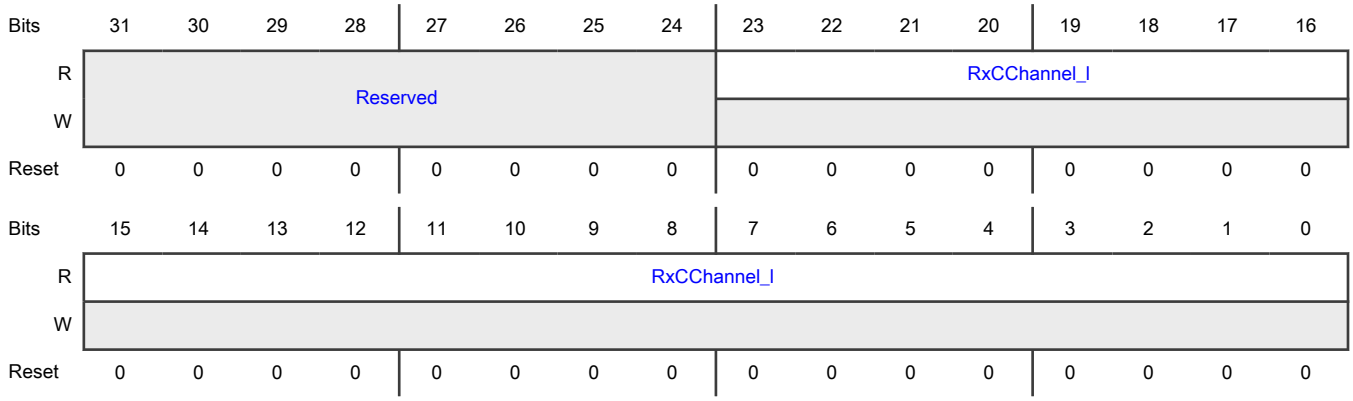
Offset

Register	Offset
SRCSL	20h

Function

SPDIFRxCChannel_l register is a channel status reception register.

Diagram



Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 RxCChannel_l	RxCChannel_l SPDIF receive C channel register, contains next 24 bits of C channel without interpretation

52.6.1.11 UchannelRx Register (SRU)

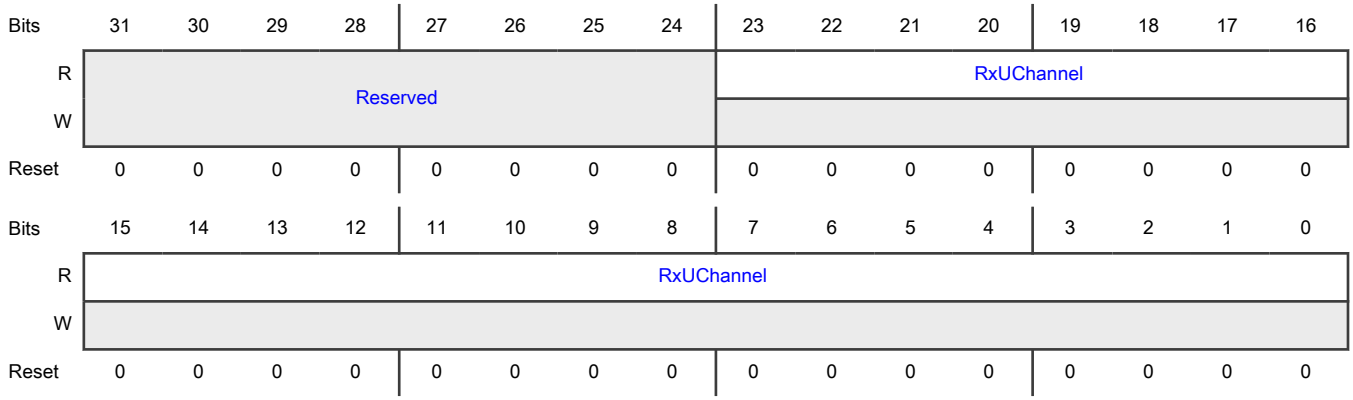
Offset

Register	Offset
SRU	24h

Function

UChannelRx register is a user bits reception register.

Diagram



Fields

Field	Function
31-24 —	[unimplemented] This is a 24-bit register the upper byte is unimplemented.
23-0 RxUChannel	RxUChannel SPDIF receive U channel register, contains next 3 U channel bytes

52.6.1.12 QchannelRx Register (SRQ)

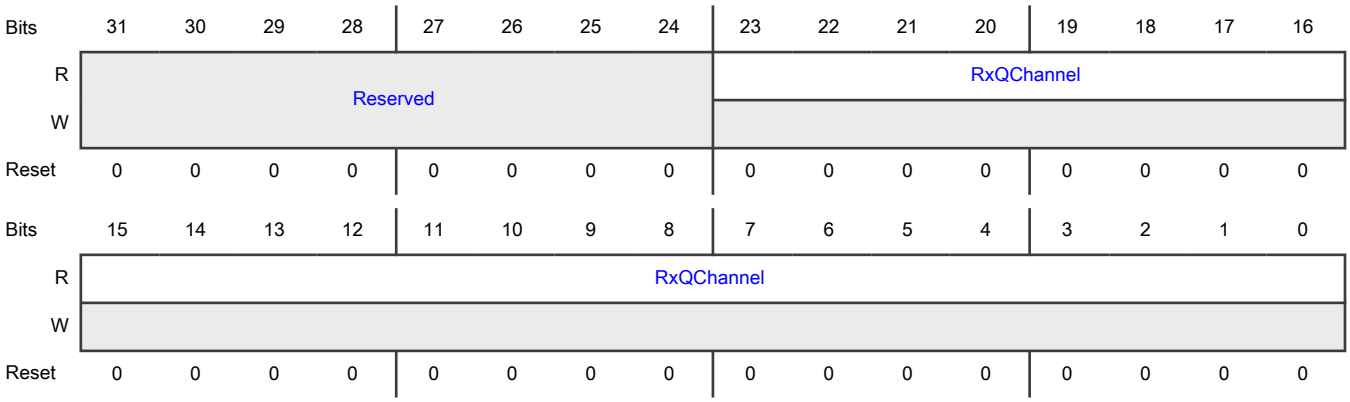
Offset

Register	Offset
SRQ	28h

Function

QChannelRx register is a user bits reception register.

Diagram



Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 RxQChannel	RxQChannel SPDIF receive Q channel register, contains next 3 Q channel bytes

52.6.1.13 SPDIFTxLeft Register (STL)

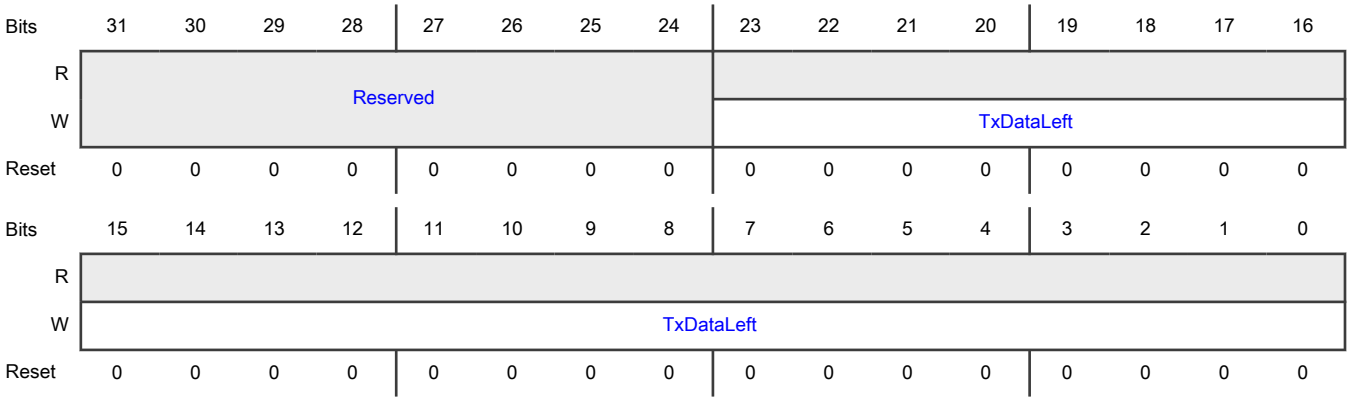
Offset

Register	Offset
STL	2Ch

Function

SPDIFTxLeft register is an audio data transmission register.

Diagram



Fields

Field	Function
31-24 —	[unimplemented] This is a 24-bit register the upper byte is unimplemented.
23-0 TxDataLeft	TxDataLeft SPDIF transmit left channel data. It is write-only, and always returns zeros when read

52.6.1.14 SPDIFTxRight Register (STR)

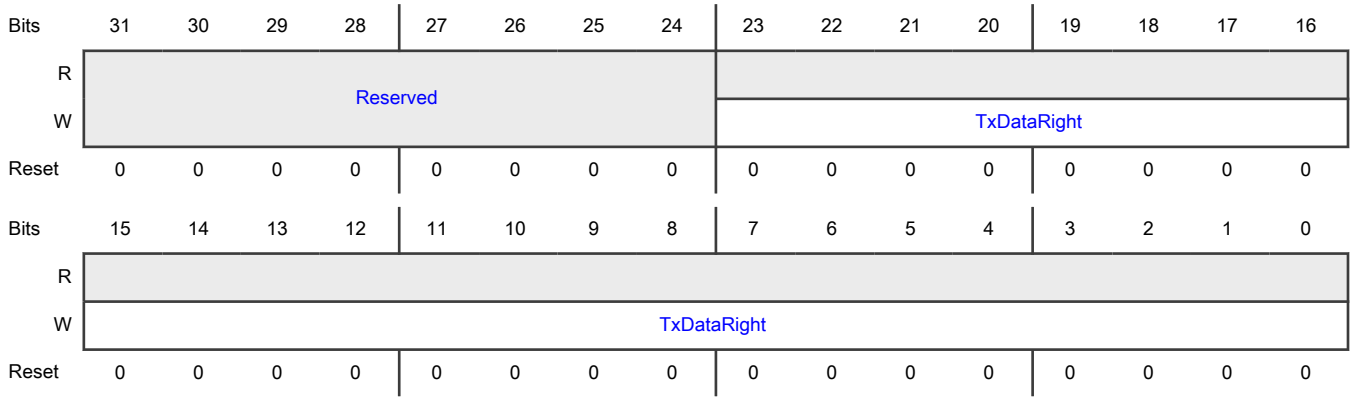
Offset

Register	Offset
STR	30h

Function

SPDIFTxRight register is an audio data transmission register.

Diagram



Fields

Field	Function
31-24 —	[unimplemented] This is a 24-bit register the upper byte is unimplemented.
23-0 TxDataRight	TxDataRight SPDIF transmit right channel data. It is write-only, and always returns zeros when read

52.6.1.15 SPDIFTxCChannelCons_h Register (STCSCH)

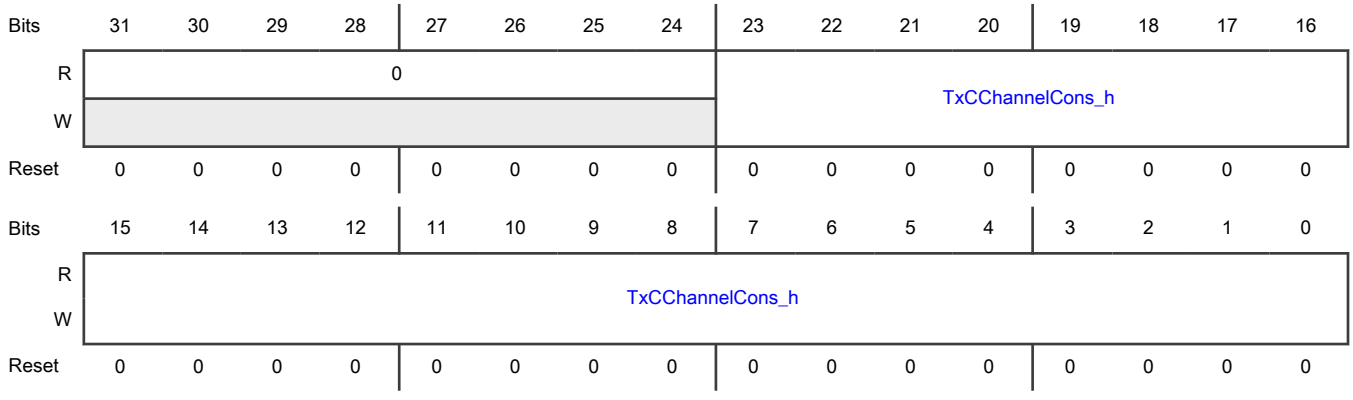
Offset

Register	Offset
STCSCH	34h

Function

SPDIFTxCChannelCons_h register is a channel status transmission register.

Diagram



Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 TxCChannelCons_h	TxCChannelCons_h SPDIF transmit Cons. C channel data, contains first 24 bits without interpretation. When read, it returns the latest data written by the processor

52.6.1.16 SPDIFTxCChannelCons_l Register (STCSCL)

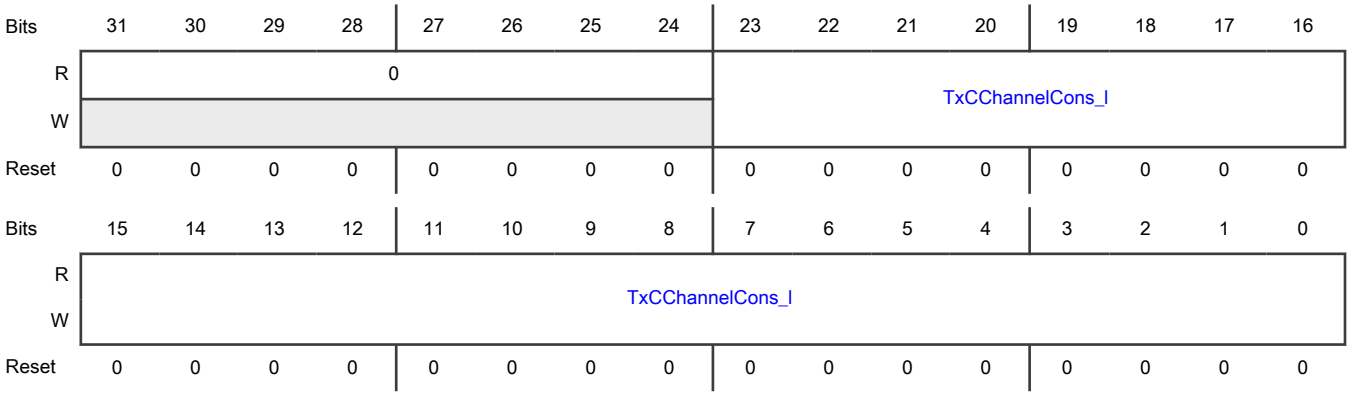
Offset

Register	Offset
STCSCL	38h

Function

SPDIFTxCChannelCons_l register is a channel status transmission register.

Diagram



Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 TxCChannelCons_l	TxCChannelCons_l SPDIF transmit Cons. C channel data, contains next 24 bits without interpretation. When read, it returns the latest data written by the processor

52.6.1.17 FreqMeas Register (SRFM)

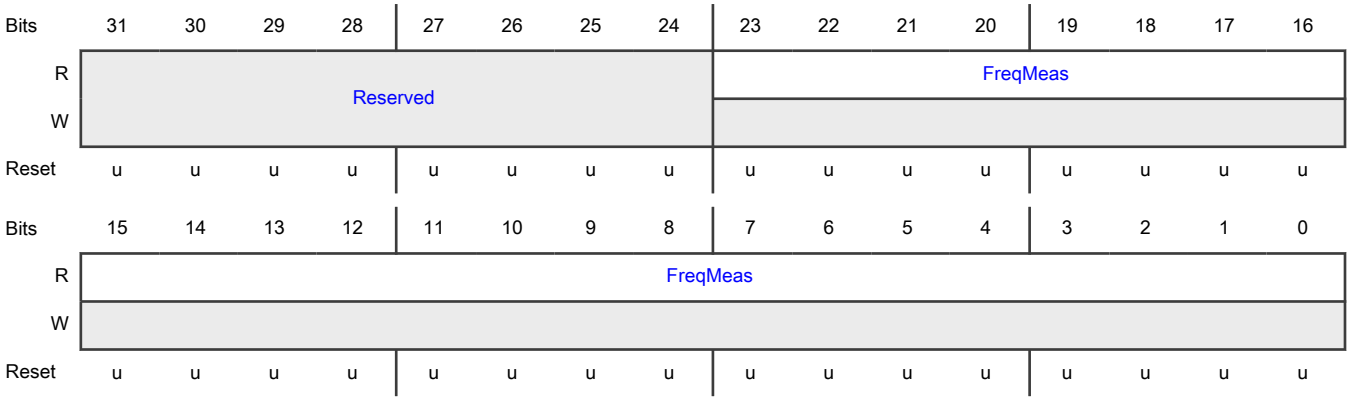
Offset

Register	Offset
SRFM	44h

Function

Contains Frequency measurement data.

Diagram



Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-0 FreqMeas	FreqMeas Frequency measurement data

52.6.1.18 SPDIFTxClk Register (STC)

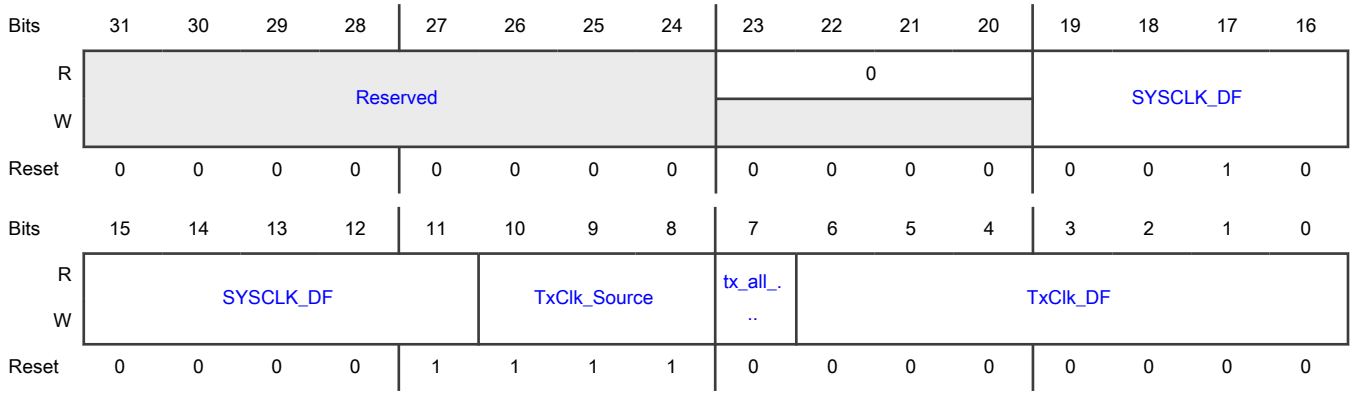
Offset

Register	Offset
STC	50h

Function

The SPDIFTxClk Control register includes the means to select the transmit clock and frequency division.

Diagram



Fields

Field	Function
31-24 —	This is a 24-bit register the upper byte is unimplemented.
23-20 —	- Reserved, return zeros when read
19-11 SYSCLK_DF	SYSCLK_DF system clock divider factor, 2~512. 0_0000_0000b - no clock signal 0_0000_0001b - divider factor is 2 1_1111_1111b - divider factor is 512
10-8 TxClk_Source	TxClk_Source **NOTE** See device-specific Clocking chapter to know the clock sources used. **NOTE** Bit values of this field are device-specific. See the chip-specific section for this module, at the beginning of this chapter.
7 tx_all_clk_en	tx_all_clk_en Spdif transfer clock enable. When data is going to be transferred, this bit should be set to1. 0b - disable transfer clock. 1b - enable transfer clock.
6-0 TxClk_DF	TxClk_DF Divider factor (1-128)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000_0000b - divider factor is 1 000_0001b - divider factor is 2 111_1111b - divider factor is 128

52.6.1.19 SPDIF receive C channel register, bits 31-0 (SPDIFRxCChannel_Addr_31_0)

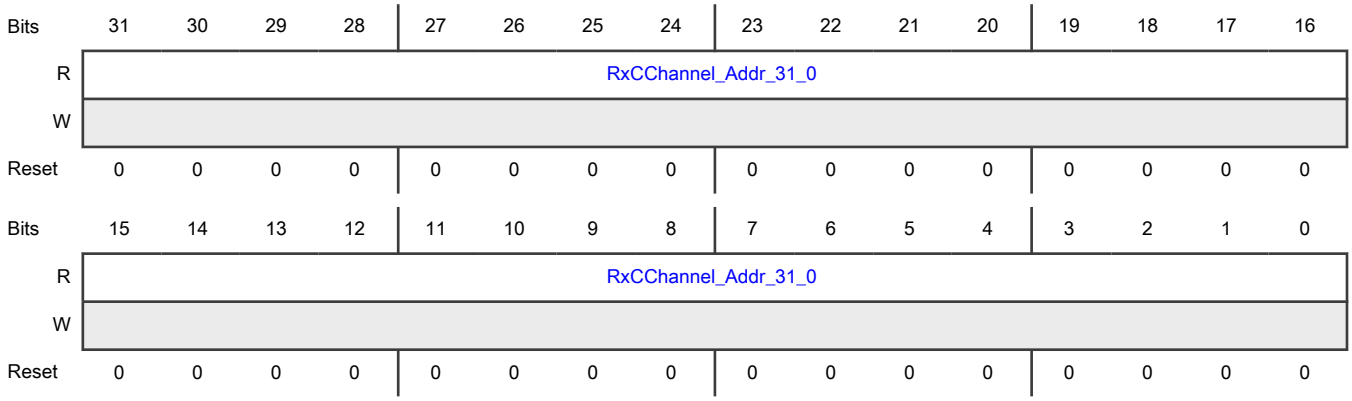
Offset

Register	Offset
SPDIFRxCChannel_Addr_31_0	60h

Function

Contains bits 31-0 of C Channel (Receive)

Diagram



Fields

Field	Function
31-0 RxCChannel_Addr_31_0	RxCChannel_Addr_31_0 SPDIF receive C channel register, contains 0 to 31 bits of C channel without interpretation.

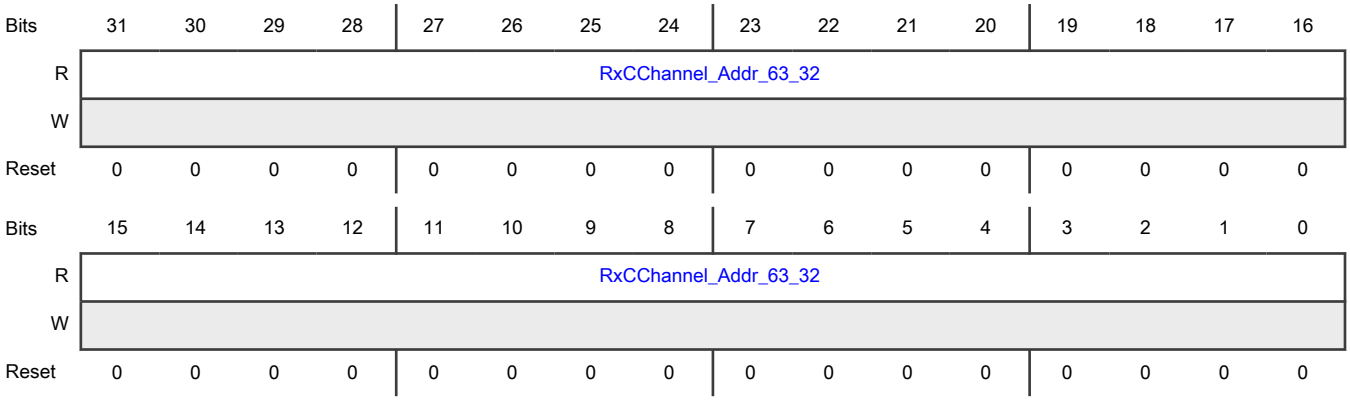
52.6.1.20 SPDIF receive C channel register, bits 63-32 (SPDIFRxCChannel_Addr_63_32)

Offset

Register	Offset
SPDIFRxCChannel_Addr_63_32	64h

Function
Contains bits 63-32 of C Channel (Receive)

Diagram



Fields

Field	Function
31-0 RxCChannel_Addr_63_32	RxCChannel_Addr_63_32 SPDIF receive C channel register, contains 32 to 63 bits of C channel without interpretation.

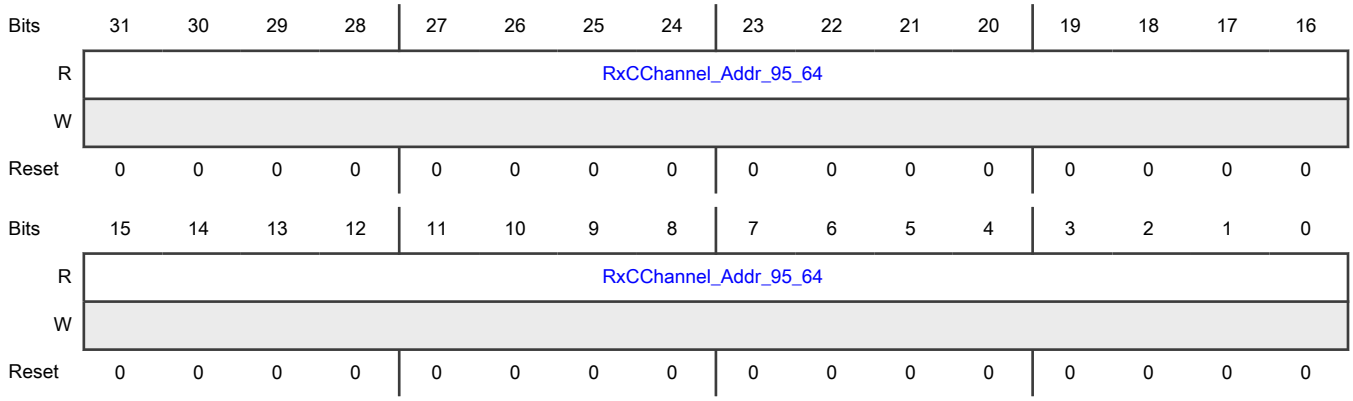
52.6.1.21 SPDIF receive C channel register, bits 95-64 (SPDIFRxCChannel_Addr_95_64)

Offset

Register	Offset
SPDIFRxCChannel_Addr_95_64	68h

Function
Contains bits 95-64 of C Channel (Receive)

Diagram



Fields

Field	Function
31-0 RxCChannel_Addr_95_64	RxCChannel_Addr_95_64 SPDIF receive C channel register, contains 64 to 95 bits of C channel without interpretation.

52.6.1.22 SPDIF receive C channel register, bits 127-96 (SPDIFRxCChannel_Addr_127_96)

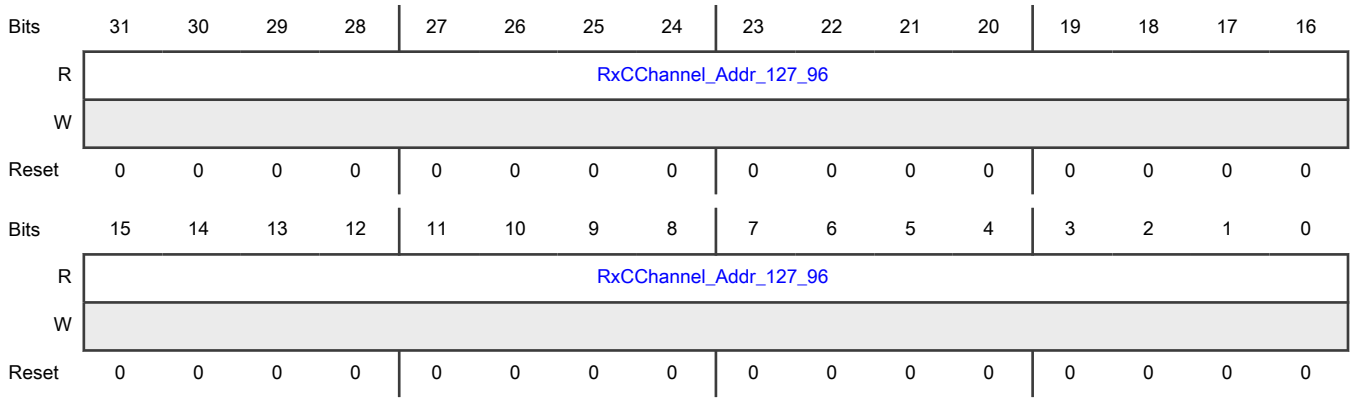
Offset

Register	Offset
SPDIFRxCChannel_Addr_127_96	6Ch

Function

Contains bits 127-96 of C Channel (Receive)

Diagram



Fields

Field	Function
31-0 RxCChannel_Addr_127_96	RxCChannel_Addr_127_96 SPDIF receive C channel register, contains 96 to 127 bits of C channel without interpretation.

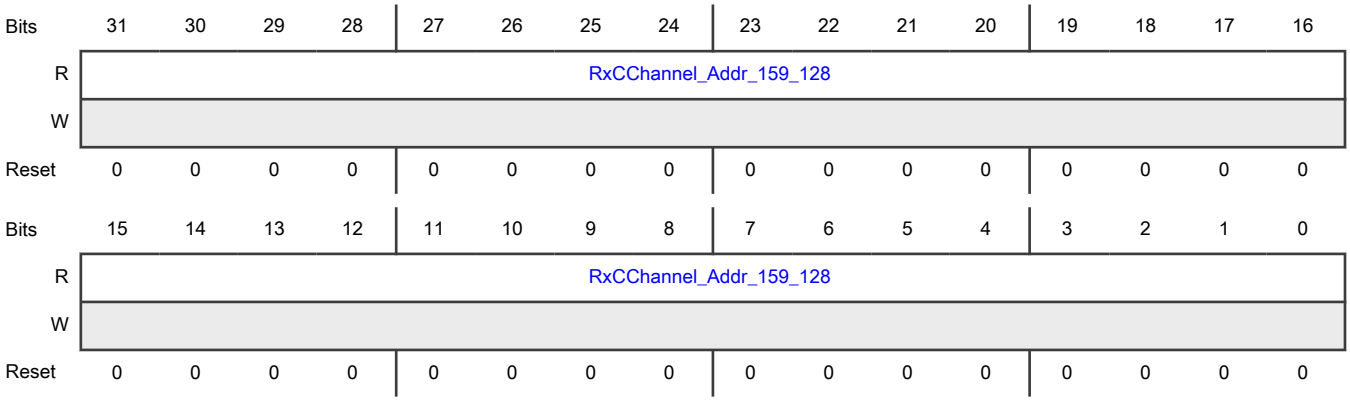
52.6.1.23 SPDIF receive C channel register, bits 159-128 (SPDIFRxCChannel_Addr_159_128)

Offset

Register	Offset
SPDIFRxCChannel_Addr_159_128	70h

Function
Contains bits 159-128 of C Channel (Receive)

Diagram



Fields

Field	Function
31-0 RxCChannel_Addr_159_128	RxCChannel_Addr_159_128 SPDIF receive C channel register, contains 128 to 159 bits of C channel without interpretation.

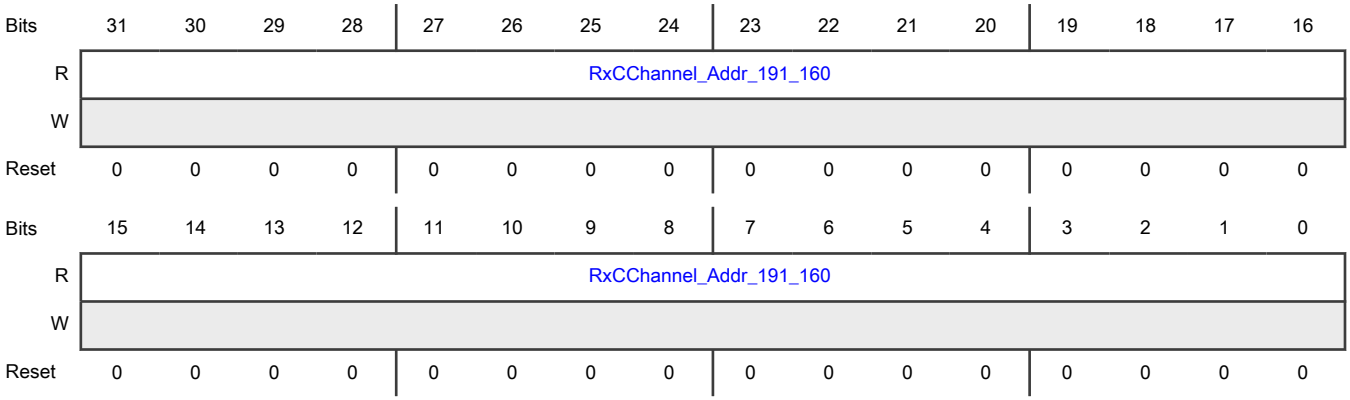
52.6.1.24 SPDIF receive C channel register, bits 191-160 (SPDIFRxCChannel_Addr_191_160)

Offset

Register	Offset
SPDIFRxCChannel_Addr_191_160	74h

Function
Contains bits 191-160 of C Channel (Receive)

Diagram



Fields

Field	Function
31-0 RxCChannel_Addr_191_160	RxCChannel_Addr_191_160 SPDIF receive C channel register, contains 160 to 191 bits of C channel without interpretation.

52.6.1.25 SPDIF transmit C channel register, bits 31-0 (SPDIFTxCChannel_Addr_31_0)

Offset

Register	Offset
SPDIFTxCChannel_Addr_31_0	78h

Function
Contains bits 31-0 of C Channel (Transmit)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TxCChannel_Addr_31_0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TxCChannel_Addr_31_0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TxCChannel_Addr_31_0	TxCChannel_Addr_31_0 SPDIF transmit C channel register, contains 0 to 31 bits of C channel without interpretation.

52.6.1.26 SPDIF transmit C channel register, bits 63-32 (SPDIFTxCChannel_Addr_63_32)

Offset

Register	Offset
SPDIFTxCChannel_Addr_63_32	7Ch

Function

Contains bits 63-32 of C Channel (Transmit)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TxCChannel_Addr_63_32															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TxCChannel_Addr_63_32															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TxCChannel_Addr_63_32	TxCChannel_Addr_63_32 SPDIF transmit C channel register, contains 32 to 63 bits of C channel without interpretation.

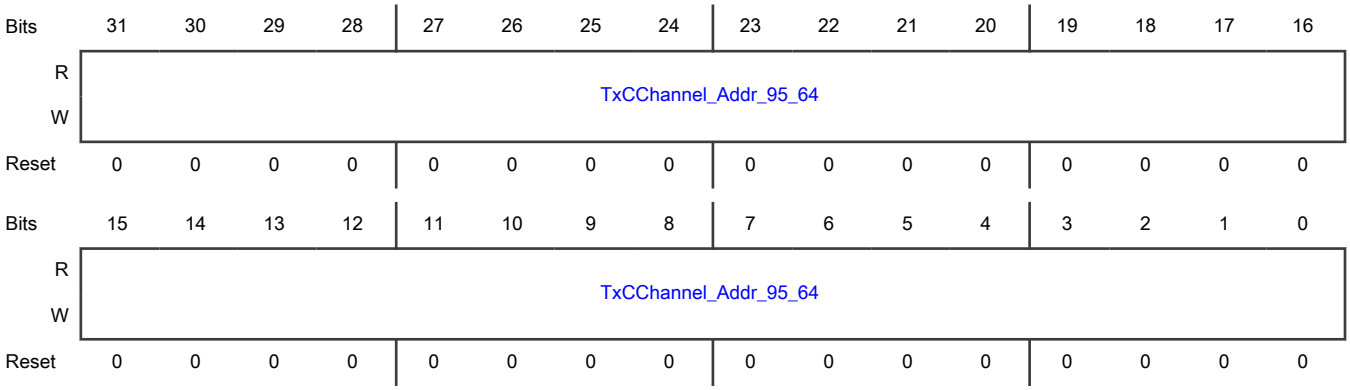
52.6.1.27 SPDIF transmit C channel register, bits 95-64 (SPDIFTxCChannel_Addr_95_64)

Offset

Register	Offset
SPDIFTxCChannel_Addr_95_64	80h

Function
Contains bits 95-64 of C Channel (Transmit)

Diagram



Fields

Field	Function
31-0 TxCChannel_Addr_95_64	TxCChannel_Addr_95_64 SPDIF transmit C channel register, contains 64 to 95 bits of C channel without interpretation.

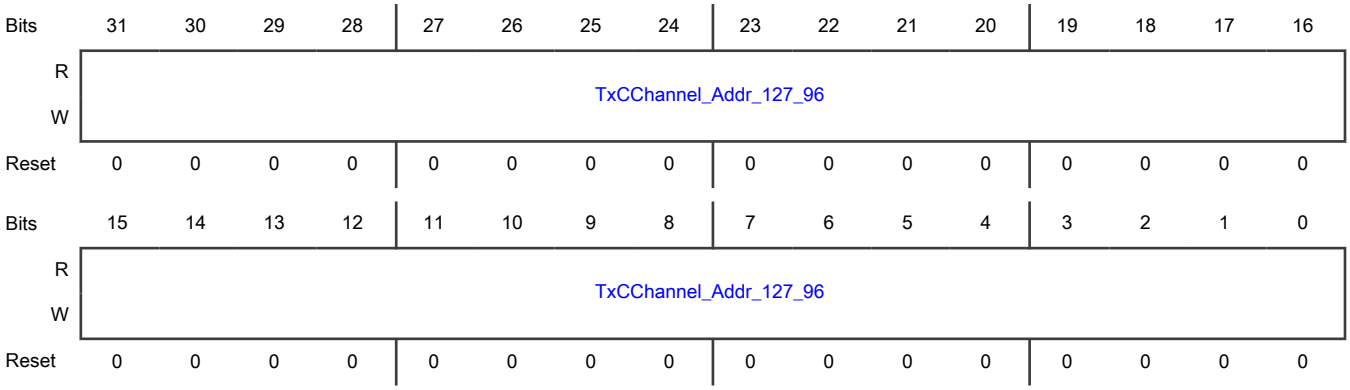
52.6.1.28 SPDIF transmit C channel register, bits 127-96 (SPDIFTxCChannel_Addr_127_96)

Offset

Register	Offset
SPDIFTxCChannel_Addr_127_96	84h

Function
Contains bits 127-96 of C Channel (Transmit)

Diagram



Fields

Field	Function
31-0 TxCChannel_Addr_127_96	TxCChannel_Addr_127_96 SPDIF transmit C channel register, contains 96 to 127 bits of C channel without interpretation.

52.6.1.29 SPDIF transmit C channel register, bits 159-128 (SPDIFTxCChannel_Addr_159_128)

Offset

Register	Offset
SPDIFTxCChannel_Addr_159_128	88h

Function
Contains bits 159-128 of C Channel (Transmit)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TxCChannel_Addr_159_128															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TxCChannel_Addr_159_128															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TxCChannel_Addr_159_128	TxCChannel_Addr_159_128 SPDIF transmit C channel register, contains 128 to 159 bits of C channel without interpretation.

52.6.1.30 SPDIF transmit C channel register, bits 191-160 (SPDIFTxCChannel_Addr_191_160)

Offset

Register	Offset
SPDIFTxCChannel_Addr_191_160	8Ch

Function

Contains bits 191-160 of C Channel (Transmit)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TxCChannel_Addr_191_160															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TxCChannel_Addr_191_160															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TxCChannel_Addr_191_160	TxCChannel_Addr_191_160 SPDIF transmit C channel register, contains 160 to 191 bits of C channel without interpretation.

Chapter 53
Ethernet Controller (NETC)

53.1 Chip-specific NETC Information

Table 346. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

53.2 Introduction

The NETC complex is comprised of two main components:

- Ethernet switch and controller (NETC) Integrated Endpoint
- PCIe Integrated Endpoint Root Complex (iEPRC)

NETC presents itself as a multi-function PCIe Root Complex Integrated Endpoint (RCiEP or simply iEP). As such, it contains multiple PCIe functions related to Ethernet functionality such as Ethernet switching and Network Interface Controller (NIC) functionality. As an iEP device, NETC does not integrate real PCIe hardware; there are no PCIe physical, data link or transaction layers involved.

The PCIe compliance brings the major benefit of each NETC function being able to be isolated, initialized, quiesced and reset independently of each other which is clearly a benefit for any device that runs multiple software partitions. Another benefit is the ability to use standard PCIe device drivers to optionally discover what features/capabilities exists on the device.

The implications of using PCIe compliant programming model are:

- The main programming model constraint that results from NETC PCIe compliance is that some of the NETC registers are structured and laid out in the address space according to the PCIe specification.
- Another programming model implication of NETC PCIe compliance is usage of Message Signal Interrupts (MSI-X) replacing interrupt wires.
- Although NETC is presented as a PCIe iEP it is possible to use it as a set of more traditional peripheral devices with driver software which does not have support for PCIe and its discovery processes. This requires some configuration during

the boot and initialization of the device. For more information, see Use of Integrated End-Points by Non-PCIe Aware Operating Systems.

The iEPRC provides support for PCIe enumeration and the interface for the host to access the NETC PCIe configuration registers (standardized set of registers) and NETC specific registers. The iEPRC also includes a component called event collector, which allows NETC to report, for instance, power management events and certain errors to the host. The iEPRC is described in the following sections in general terms as a host for one or more iEPs with its own address map.

53.3 PCIe Integrated End-Point Root Complex (iEPRC)

53.3.1 Overview

The iEP-RC provides the necessary infrastructure to present NETC as a PCIe Integrated Endpoint so that it appears to be a PCIe device to system software. That is the device makes use of the support provided by system software such as boot firmware, operating systems, and Virtual Machine Managers (VMM, such as, Xen or KVM) for PCIe devices. This despite the fact that the functionality does not use PCIe serial links but is integrated into the SoC.

By PCIe definition, a Root Complex includes at least one Host Bridge, Root Port, or Root Complex integrated End-Point (RCiEP). As there are no PCIe physical links, the iEP-RC contains no Host Bridge and no Root Ports. In this instance of the iEP-RC, it hosts the NETC iEP along with the support of one Root Complex Event Collector (RCEC).

Use of the PCIe infrastructure simplifies adding support for the integrated devices to this system software and facilitates:

- Device discovery and location
- Resource requirement discovery and allocation (such as interrupt assignment, device register address)
- Event reporting (such as catastrophic internal errors, power management events) for events which are system level and outside of the scope of the devices' normal operation

In the case of VMMs it allows integration with no changes to the VMM software (for example, VFIO). Boot firmware and OSes can discover and support many different physical configurations of integrated devices with at most the addition of a driver for a pseudo "PCI controller" (the iEP-RC).

This simplifies software support and "up streaming" of device specific software by limiting the changes to the device specific drivers.

Figure 309 shows a simplified block diagram of the iEP-RC architecture. This consists of the iEP-RC, one RCEC and support for the NETC iEP.

NOTE

This diagram shows logical connections to the SoC Interconnect based on the different types of access required to the various blocks. This is not intended to dictate the number of actual connections as a physical connection to the interconnect can be used for multiple purposes. The ability to combine uses of a connection usually depends on factors such as whether the address ranges assigned for the different types of accesses can be adjacent in the SoC address map to facilitate address decode in the interconnect.

53.3.1.1 Block diagram

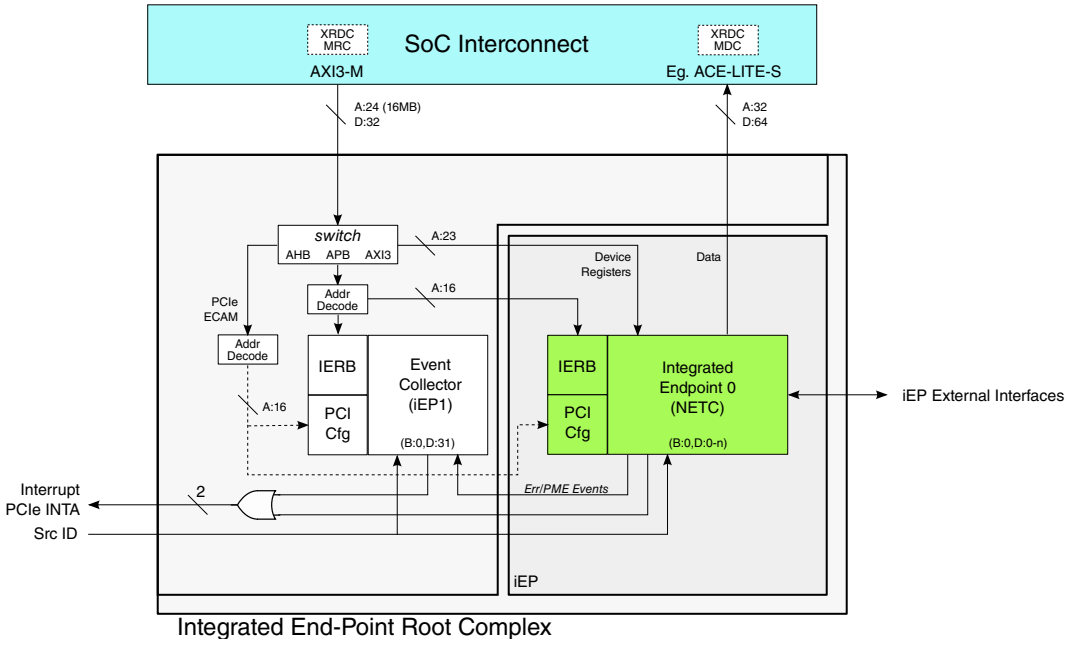


Figure 309. iEP Architecture

53.3.2 Functional Description

53.3.2.1 PCI/PCIe Background

It is useful to have an understanding of PCI/PCIe to be able to understand PCI Root Complex Integrated Endpoints. This section is intended to give an overview for those not familiar with PCI, not to provide a detailed explanation.

53.3.2.1.1 Overview

PCI and its successor PCI Express (PCIe) provide a framework for connecting peripheral devices to a computer consisting of one or more CPU(s) and memory. PCI defines how such devices are discovered and integrated into the system. This includes:

- Assignment of addresses for registers and memory in the device, determining how the "host system" accesses these
- Assignment, configuration, and control of interrupts including various types of message signaled interrupts (MSI)
- Reporting and handling certain errors
- Handling power management including events that will cause the system to exit low power modes
- Adding and removing such devices to a “live” system (hotplug)

PCI is defined for a multi-connect, parallel bus and PCIe for a serial interconnect. Devices can be arranged into a tree topology through the use of bridges or switches. Both PCI and PCIe are intended to be used with pluggable cards and are often used with peripherals on the same circuit board. None the less the use of PCI with peripheral devices which are integrated into an SoC is becoming common as it allows using the extensive standardized approaches and software infrastructure created for PCI with these integrated peripherals simplifying software development and portability across different SoCs.

With PCI, system software (boot firmware, OS, …) is initially only aware of a PCI controller -- a device which acts as the bridge between the host and the PCI/PCIe. It is not aware of any devices connected to this controller. Software identifies the PCI connected devices through a process known as enumeration.

53.3.2.1.2 Routing/Requester ID (RID)

PCI is organized in a tree like hierarchy with levels in the tree identified by Bus number and components at each level identified by Device and Function within a Device. A specific Function is identified by its Routing (or Requester) ID (RI or RID) consisting of Bus, Device, and Function numbers. Bus, Device, and Function numbers are 8, 5, and 3 bits respectively so RIDs are 16 bits allowing for 256 buses, 32 Devices per Bus, and 8 Functions per Device. Since PCIe uses serial links a Bus will often have only a single Device (although that Device may support multiple diverse functions). PCIe therefore supports Alternative Routing ID (ARI) which assumes that only Device 0 is present on a Bus and allows the Device ID bits to be used as additional Function IDs on that Bus. This allows a single multi-function PCIe Device to have up to 256 Functions.

Buses are connected to each other using components that bridge or switch between the buses. These components are configured by system software as they are discovered. Bus numbers are assigned during the enumeration process as the process proceeds. Each bridge or switch is assigned a range of bus numbers that depend on the needs of the devices and topology downstream of it.

A PCI Function can be considered to be the basic set functionality that a peripheral device would present to software. A network interface (Ethernet controller) would be a Function. A Device is a collection of (usually related) Functions. Typically PCI plugin card would be a single Device although it is also possible for a card to represent multiple Devices.

53.3.2.1.3 Config Space

On PCI each Function has a 256 B space of configuration registers. For PCIe this is expanded to 4 KB which incorporates the 256 B set of registers defined for PCI at offset 0 so that PCIe maintains compatibility with PCI.

The format of the first 64B of the configuration space (called the header) is standardized. Two different types of headers are defined: Type 0, used for port/bridge/switch functions, and Type 1, used for all others.

Sets of registers in the remaining Config space are defined using a tag-length-value like structure. These sets of registers are known as capability structures. Several capability structures are standardized and provision exists for a vendor to define their own capability structures. The TLV format of the capability structures allows system software to step over structures that it doesn't understand and find the standard capability structures it needs to configure the Function in the system.

Registers defined within the header and in standardized capability structures allow system software to assign or discover system resources (such as interrupts and address space for access to the PCI Device from the host system) and to configure and control functions such as power management, device hot plug, and reporting/handling certain errors.

Switch or bridge devices have registers within their headers which control the range of buses that are allocated to their downstream devices and the address ranges assigned to downstream devices thus describing the tree topology.

53.3.2.1.4 Config Space Access

Configuration space is addressed using a combination of the 16b RID in the upper bits and the offset within the Config space, 8b in the PCI standard 256B space or 12b in the 4KB PCIe extended Config space.

PCIe defines the Enhanced Configuration Access Mechanism (ECAM). ECAM uses a range of addresses in the host systems memory space large enough to allow direct, linear addressing of Config spaces of all Buses, Devices, and Functions in the hierarchy. Software simply performs a read or write within this range using the RID plus offset within Config space as the offset inside the ECAM range.

The full range of potential RIDs for a PCIe (Bus/Device/Function) with a 4KB Config space requires an ECAM space of 256MB.

53.3.2.1.5 Enumeration

Enumeration is the process performed by software to discover all Devices connected to a given PCIe hierarchy and to configure the bridge/switch functions appropriately. Enumeration is performed by reading a specific 16b configuration register (the Vendor ID register) for Function 0 of each potential Device using a process specific to the PCI controller. This will return 0xFFFF for any device which does not exist and a valid vendor ID for those that do. If Function 0 of a Device is discovered then the software can probe for other functions in that Device.

In addition to the Vendor ID another register provides a 16b Device ID which is assigned by the vendor. The Device ID identifies the specific device so that the appropriate driver can be loaded and used to interact with the device. Drivers register with the OS indicating what Device ID/Vendor IDs they support using an OS specific mechanism.

53.3.2.1.6 Function Address Space Allocation or Discovery

RIDs are used to access a Function's Config space and, for instance, by host system software to identify the source of an error or event interrupt. Normal driver software does not use the RID to access a peripheral's functionality. It does this using registers within an address space assigned to the function within the system/host address space. PCI/PCIe inherently provides memory mapped access.

Base Address Registers (BARs) in the header of Config space support up to six 32b addressed memory spaces or three 64b addressed spaces per Function (by combining two 32b BARs). Functions indicate the amount of address space required in the system's address map to access various components in the Function. This can include device registers or memory. PCI defines a standard way for a Function to indicate the amount of memory space required as well as certain attributes such as whether the host can prefetch data from that space (e.g. whether reads have side effects) and for system software to set the address range allocated.

The details of use of these address spaces is not defined by the standard. A typically use is to allow software to access operational registers required by the Function. A specific driver for the device, determined by the Vendor ID and Device ID, must be loaded which understands the use of the address space(s) assigned using the BARs. Standard system software can, however, can allocate address space without knowing the details of its use.

A PCI Root Complex (controller) has a range or window of addresses assigned to it within the host or systems address map. Accesses from the host (e.g. by the host GPPs) are directed to the RC. During enumeration system software assigns ranges of addresses to the Functions it discovers, as needed by those functions.

A Function may alternatively make use of an Enhanced Allocation capability structure in its Config space to inform the system of fixed addresses within the SoC's memory map at which its registers or memory can be accessed. This Enhanced Allocation mechanism can be used instead of using the BARs found in the Config header. Enhanced Allocation is especially useful for iEPs since they do not need to connect to the SoC/system interconnect through the Root Complex. They can use addresses which are not allocated within the RC's address window. Using Enhanced Allocation not only simplifies verification of the iEP but can greatly simplify integration of the iEP IP into an SoC.

53.3.2.1.7 Interrupts

PCI supports up to 4 interrupt signals known as INTx (x = A..D). PCI also supports 2 types of message signaled interrupts (MSI and MSI-X).

Wired interrupts (INTx) are configured and controlled through registers in the header of Config space. MSI are configured and controlled through registers in an MSI capability structure. MSI-X are configured and controlled through registers within a BAR specified address space with a reference (BAR id and offset within the BAR) provided in an MSI-X capability structure. The format of the MSI/MSI-X configuration registers is specified by the PCI standards.

Functions indicate their requirements and capabilities to support the various types of interrupts through settings in defined registers (in the header) and registers in the MSI and MSI-X capability structures. System software can examine these registers and allocate interrupts accordingly. Similar to address assignment, system software can do this allocation without any detailed knowledge of the intended use of the interrupts thus separating the process of allocating of resource (interrupts) from their use.

PCI Functions can make use of one of the 4 (potentially) available wire signaled interrupts and this is selected using a register in the Config header. This requires multiplexing all interrupt sources onto a single interrupt signal, and then the interrupt service routine must determine the actual source of the interrupt before calling the appropriate handler. In addition, only a single core in a multi-core system can handle the interrupts from a Function which uses wire signaled interrupts. Different MSI(-X)s can be allocated and used for different interrupt sources and these can be serviced by different cores thus spreading the load of responding to these interrupts. As a result MSI(-X) are usually preferred over wire signaled interrupts.

53.3.2.1.8 Virtualization

The Single Root I/O Virtualization and Sharing Specification (SR-IOV) is an extension to PCIe that defines how PCIe devices support virtualization. It does so by defining a special type of Function known as a Virtual Function (VF). A VF is always associated with a Physical Function (PF) and a PF may have multiple VFs.

A PF is a PCIe Function and represents a set of functionality presented by a peripheral device to software. A VF typically supports a subset of the PF's functionality which can be provided to software for use by a Virtual Machine, container, or other software entity.

The PF has some management and control capability over its VFs and over the underlying device functionality. Devices will often provide a means for the VF to request configuration and control changes, mediated by the PF and its driver software. For instance, a messaging system may be provided so that VF drivers can request that a PF driver make changes on its behalf.

Functions are isolated from each other so VFs are isolated from other VFs and from their PF. Because of its reduced capabilities a VF is restricted, to some extent, from interfering with other VFs. VFs can be directly assigned to a Virtual Machine, container, or user space process with direct access to the hardware.

Taking an Ethernet controller (ENETC) as an example, an Ethernet port would have a PF and some number of VFs. The controller would provide classification mechanisms to send different frames to specific Functions, for instance based on different destination MAC addresses. It might also have the capability to replicate some frames, such as broadcast frames, for all Functions. (Alternatively, this function could be performed by the PF driver software.) The PF would have the ability to control this selection mechanism. VFs would not. If the system using this controller ran several Virtual Machines it could assign a VF to each VM. The host OS would use the PF. Each VM would, in some sense, see the Ethernet port as belonging to it.

As PCI Functions, VFs have RIDs and Config space. Address assignments for VFs are not handled through the normal PCI BAR mechanism and they support a subset of the Config registers that their PFs support. Certain settings in the VF Config registers are hardwired to match their PFs settings.

53.3.2.1.9 PCIe Topologies

In PCIe, the host PCIe controller (Root Complex) contains Bus 0 for that PCIe hierarchy. There are typically one or more bridge or switch components, known as Root Ports (RP), on that Bus which connect to serial PCIe links. Each of these links then connects to either a PCIe Endpoint (EP) and/or a PCI(e) switch.

It is the PCIe EPs which provides the actual peripheral device functionality.

The RC may also include integrated components such as RCiEP (or simply iEP) and RCEC.

The RCEC is a component provided to allow iEPs to report, for instance, power management events and certain errors to system software. With non-integrated devices these are reported with messages across a PCIe serial link and it is the responsibility of upstream infrastructure components such as an RP or a switch or bridge port to take this messages and turn them into interrupts to the system. Since such upstream components do not exist for an iEP the RCEC provides this capability. Note that having the iEP generate these interrupts directly violates the concept that an iEP should appear like a regular PCIe EP to system software since regular EPs do not themselves generate these interrupts.

Since we are only dealing with iEPs and ECs in this Root Complex, examples of this topology will only be discussed further.

53.3.2.2 Integrated Endpoint Architecture

The iEPRC is architected to support one or more integrated End-Points (iEP) and zero or more RCEC. This section describes in general terms, the overall architecture of the Integrated Endpoint Root Complex IP and the Integrated Endpoint.

NOTE

This instance of the iEP-RC, includes a single iEP (that is NETC) and a single RCEC.

53.3.2.2.1 Root Complex Functionality

The iEP-RC implements ECAM to provide access to the PCIe Config space of the iEPs and the main function of the Root Complex logic is to route accesses from software to the PCI Config space to the appropriate destination, either an iEP, RCEC, or the Root Complexes' own Config space.

In addition to routing Config accesses to the correct iEP or RCEC. the Root Complex routes accesses to a block of registers in each iEP and RCEC, the iEP Register Block (iERB). There is an iERB for each Device on Bus 0 for a total of 32 possible iERBs for an iEP-RC.

The Root Complex also routes legacy wired interrupt requests to the system from iEPs that require the ability to use PCIe INTx interrupts. The spec allows a Root Complex to have up to 4 wired interrupts corresponding to the 4 PCI interrupts (INTA thru INTD) but normally only a single wired interrupt is supported per RC. If multiple iEPs require use of wired interrupts then the RC is responsible for logically OR'ing those requests onto the interrupt signals as required.

53.3.2.2.2 RCEC Functionality

The RCEC provides functionality for iEPs which would otherwise be provided by PCIe components downstream of the iEP, if the iEP were an Endpoint on an actual PCIe topology. This includes error reporting for certain catastrophic errors in an iEP (e.g. fatal memory error in an internal memory, PCIe Advanced Error Reporting) and reporting and managing power management events (e.g. requests to exit low power states, PCIe Power Management Events). These are reported as in-band messages on a PCIe topology. These in-band messages are converted to interrupts which can be understood by the host by the downstream component such as a Root Port.

A given RCEC can support either wired or MSI-X interrupts for reporting these events and errors to software.

The iEP-RC includes zero or more RCECs. Normally at least one RCEC is required in an iEP-RC. Legacy (reused) IP converted to an iEP may not make use of RCEC for error and power management event reporting. If all iEPs are legacy logic which do not support using an RCEC then no RCEC may exist in the iEP-RC.

In many cases a single RCEC can be used by all iEPs connected to an iEP-RC provided all iEPs have the same requirements for how their events/errors are signaled to software. A scenario where multiple RCECs would be required is when one of the iEPs has a need to cause the system to exit from a low power mode (for wake-on-LAN). This would normally require the use of a wired interrupt on the part of the RCEC to cause an exit from the low power mode since MSI(-X) will not work when significant portions of the SoC are powered down. The PCI programming model for event reporting does not allow a single RCEC to use both wire signaled interrupts and MSI at the same time for different iEPs. To allow the use of a wired interrupt for the iEP requiring wakeup and MSI for the remainder of the iEPs two RCEC can be used one of which uses an INTx and the other which uses MSI.

NOTE

This instance of the iEP-RC IP includes a single RCEC, which supports only wire signaled interrupts. These are used to report both errors and power management events.

53.3.2.2.3 Integrated Endpoint Functionality

iEPs present their native functionality, such as NETC, to the system as multiple PCI Functions. In addition to whatever native functionality they provide iEPs must provide functionality as part of their role as PCI integrated Endpoints.

Primarily, this consists of accepting and responding to accesses to the PCI Configuration space of each of the Functions they support. Registers in the Config space are used to inform software of the capabilities and status of the Function as well as configure and control its functionality.

Specifically, this includes:

- Function reset
- Enabling/disabling of the Function as a whole
- Enabling/disabling of some types of error reporting
- Enabling/disabling isolation functionality
- Enabling/disabling interrupts

53.3.2.2.4 SR-IOV Support

iEPs such as NETC, which support virtualized or isolated access to their functionality, for instance by containers or Virtual Machines running on the SoC, do so using PCI SR-IOV. iEPs can implement from one to 1000’s of Virtual Functions.

An iEP which implements SR-IOV must support the SR-IOV Extended Capabilities structure. VF Migration is used only with MR-IOV and is not supported by iEPs. VFs of an SR-IOV RCiEP device are associated with the same Root Complex Event Collector (if any) as their PF.

53.3.2.2.5 Addressing

53.3.2.2.5.1 Page Size and Alignment

iEPs uses a minimum page size and alignment of 64KB. Register allocations are rounded up to the next 64KB regardless of actual requirements. This ensures proper alignment and address space allocation when system software requests a page size of less than 64KB (e.g. 4KB). iEPs can claim support for smaller page sizes with minimal complexity at the expense of potentially using more address space than actually required.

53.3.2.2.5.2 PCI BAR Usage

PCI supports both 32 bit and 64 bit Base Address Registers (BARs). iEPs uses 64b BARs.

53.3.2.2.5.3 PCI Enhanced Addressing

iEPs use Enhanced Addressing to communicate their addresses to software. Software is informed of base addresses and apertures rather than assigning them using writable BARs in the Config header. The addresses assigned to iEPs are not within address windows in the SoC's address map assigned to the iEP-RC or within the address windows assigned to any PCIe controller.

iEPs connect directly to the SoC Interconnect to accept transactions targeting their registers or embedded memories. The iEP-RC is not involved in routing such accesses to the iEPs.

53.3.2.2.6 Interrupts

iEPs may support MSI-X or INTx for functional interrupts. If INTx is used it is routed through the iEP-RC and follows the interrupt standard described by the RCEC. It should be noted that, if wired interrupts are used by an iEP, all interrupt sources will share a single interrupt to the cores. The iEP must provide mechanisms for software to distinguish between different interrupt sources within its own programming model.

53.3.2.2.7 Device and Function Number Assignment

iEPs claim an integer number of contiguous Devices in the RID space, to hold non-virtual and physical function. This does not mean that an iEP implements all Functions within that space. Holes may be left in the RID space claimed by an iEP including the end of the RID space it occupies.

If an iEP (for example, NETC) has PFs that can implement a large number of VFs, then it is assigned a secondary Bus number and secondary base Device number. It uses this secondary Bus and Device as the starting point for the RIDs of any sets of VFs which meet this criteria. The goal is to minimize consumption of Device numbers on Bus 0 to allow other iEPs use of this space as well.

The RID of the first VF for this iEP is:

- *secondary-Bus:secondary-base-Device:0*

and the remaining iEP's VFs following this occupies as much Device and Bus space as required.

All (non-Virtual, Physical) iEP Functions are in Bus 0. Buses 1 through 7 are used for Virtual Functions only.

RCECs also occupy RID and Config space. RCECs are implemented as Functions in Bus 0, Device 31. The first RCEC is Function 0 of Device 31 and any other RCEC's take sequential Function numbers in that Device.

53.3.2.2.8 Device ID Assignment

The Device ID identifies a particular PCIe Function and is assigned by the vendor. It is used in conjunction with the Vendor ID and Revision ID for software to determine which driver should be loaded.

The table below indicates the defined values for Device ID (0xE*xxx* family) as it relates to Integrated End Points. All other values are reserved.

Table 347. Device ID Assignment

Vendor ID	Device ID	
Freescale (0x1957)	E001	RCEC which support only INTx
	E002	RCEC which support MSI/MSI-X + INTx
	E100	ENETC rev 1
	E101	ENETC rev 2
	E110	ENETC rev 2 acting as a NETC switch management port
	E111	TSN Ethernet switch management port
	E200	PON Port
	EE00	EMDIO controller rev 2
	EE01	EMDIO controller rev 1
	EE02	IEEE 1588/1722
	EEF0	TSN Ethernet Switch
	EEF2	Switch config and control
	EF00	ENETC VSI

53.3.2.2.9 Power Management and Power Management Events (PME)

Power management is used to place portions of an SoC into low power states, possibly removing their clock and power. This may be done if those portions of the SoC are not required by the system or if the system itself is entering a low power state. Power management of the iEP-RC and iEPs is performed by software using the registers in the Power Management Capability structure.

PCI defines a number of power states. Software can determine the power state of a Function and can transition it between states to either reduce the power required by a Function when it is not required or to increase its power in preparation for using it's functionality. It also defines capabilities for a Function to request a change from a low power state to a higher power state if, for instance, an external event is detected which requires bringing the system out of a low power state.

iEP Functions which require the ability to wake the system from a low power state do so using PCI Power Management Events (PME). iEPs signal PMEs to their RCEC using a wired signal. RCECs in turn indicate that a PME has occurred using an interrupt. If the intent is to cause the system to exit a low power state (e.g. for wake-on-LAN) then a wire signaled interrupt must be used.

53.3.2.2.10 Error Reporting

iEPs report certain errors using PCI error reporting mechanisms. Most of the PCI and PCIe specific errors do not apply to an iEP due to the fact that iEPs do not have serial links and the majority of the error types supported are related failures on these links.

53.3.2.2.10.1 PCIe Advanced Error Reporting Support

iEPs that support AER must support reporting of Internal errors RCEC. Support of other AER errors is not required.

This type of error is used to report unexpected errors and faults such as bit errors in internal memories in an iEP. Only one Function within an iEP should report any single error which occurs in shared resources in the iEP.

AER does not distinguish between different types of Internal errors. AER allows these errors to be classified as "recoverable" or unrecoverable" but all errors of a given type (e.g. Internal) are treated in the same way.

iEPs do not use this mechanism to report normal operational errors such as CRC errors in packets received over an external Ethernet port.

53.3.2.2.11 Reset

PCIe defines a means for software to initiate a soft reset of individual Functions called Function Level Reset (FLR). Requirements for FLR are described in the section PCI Function Level Reset Support.

53.3.2.2.12 Pre-boot Initialization

PCI devices commonly support a mechanism to perform some type of pre-boot initialization by reading configuration from nonvolatile memory at boot-up, before enumeration can occur. This can include allocating internal resources between Functions, and simply setting initial defaults for some user visible registers. This configuration is changeable only by rewriting the configuration data in nonvolatile memory and requires a full (such as, Conventional) reset to cause this to take effect. FLR does not cause this initialization to take place.

53.3.2.2.13 iEP Register Block

An iEP Register Block (iERB) is a 64 KB size page containing registers that are used for pre-boot initialization, debug, and non-customer configuration. There is a an iERB per Device.

The lower 32 KB of the iERB are reserved for architected registers required for iEPs. The upper 32 KB are available for an iEP to use as it sees fit.

53.3.2.3 INTx Signaling

iEPs which require the use of wire signaled (SPI) interrupts do so through the iEP-RC, not directly to the SoC interrupt controller. Each such iEP has a single INTx request line to the iEP-RC.

The iEP-RC supports one wire signaled interrupt. All INTx interrupt sources are muxed (logically ORed) onto this single line to the host/SoC by the iEP-RC. Similarly, multiple Functions within an iEP may use wire signaled interrupts and the iEP is responsible for muxing (logically ORing) these requests. The INTx request to iEP-RC remains asserted as along as any source in the iEP is asserting its request.

If PM-PCI power management is used, any request for D3 power state or while in D3 power state will deassert INTx.

53.3.2.4 Power Management Signaling

To reduce power consumption portions of an SoC can be placed in low power modes and possibly have their power removed. This includes iEPs. When in this low power mode some iEPs may wish to signal the system requesting that it be brought out of this state to a more fully functional state, for instance, an iEP implementing an Ethernet controller may provide wake-on-LAN functionality. This capability is provided in PCI using Power Management Event (PME) signaling.

A PME message is delivered to the Root Complex Event Collector which is configured to wake the system by generating a wire signaled interrupt (i.e. INTx) by setting the PME Interrupt Enable bit in the Root Control register of its PCI Express Capabilities structure.

Functions indicate PME to the RCEC using their own signal. A Function which asserts this signal continues to do so until the condition is cleared by software by writing the PME Status bit in the Function’s PMCSR register in its PCI Power Management Capability structure.

The RCEC is aware of the RID associated with each PME signal it receives from iEPs and reports that RID in the Root Status Register of its PCI Express Capabilities structure. This RID is valid and the interrupt to the host asserted as long as the PME Status bit in the Root Status register is set.

Software must clear the RCEC’s Root Status register’s PME Status bit when it has finished bringing the associated Function out of its low power state. If any PME request signal is asserted the RCEC will continue to assert the interrupt if enabled and the PME Status bit will remain set.

If multiple Functions have their PME signal asserted then the RCEC can select the Function to report using any means. It is not required to report PMEs in order of assertion. Software should take note of this especially if the RCEC’s Root Status PME Status is cleared before the Function’s PMCSR PME Status is cleared. The RCEC may choose to report a different Function’s PME rather than continue to report the PME from that Function.

53.3.2.5 Advanced Error Reporting Signaling

Advanced Error Reporting is used to report PCIe related errors such as problems with the link, Access Control errors and internal errors such as memory failures. These errors are typically not normal operational errors of the Function reporting the error and can not be addressed by its specific driver.

Each iEP must report the severity of the error and the RID of the Function reporting the error. Each Function may independently report errors and the iEP is responsible for collecting and correctly reporting error severities and RIDs sequentially. If multiple Functions report AER errors then the iEP is responsible for selecting what order to report the errors in. It may do so in any order, not necessarily in the order of assertion. Once it has started reporting a given error the iEP continues to report that error until the source of the request no longer requests it. The iEP may then start reporting a different error by deasserting it's AER error indication, updating the severity and RID, and reasserting the error indication.

The RCEC is configured to signal the host/SoC when an AER error is reported using an interrupt (i.e. INTx) through the various Error Reporting Enable bits in the Root Error Command register in their AER Capabilities structure. The RCEC asserts the interrupt as long as any iEP's AER indication signal is asserted.

If multiple AER errors are reported simultaneously the RCEC will select which to report and in what order. It may do so in any order not necessarily in the order that the signals are asserted. Once the RCEC has chosen to report a given error it continues to do so until that AER error indication is deasserted.

NOTE

Note that only non-virtual (Physical) Functions are required to implement AER.

53.3.2.6 Reset

RCECs do not support reset.

53.3.3 Use of Integrated End-Points by Non-PCIe Aware Operating Systems

53.3.3.1 Overview

Many real-time environments and operating systems (RTOS) do not have support for PCIe. Although NETC is presented as a PCIe RCiEP it is not necessary to implement full PCIe support to make use of NETC's functionality.

NETC is divided into several PCIe Functions. Each Function is a logically distinct peripheral device. These include a PTP/IEEE 1588 timer device, a shared access MDIO interface for configuring and controlling Ethernet PHYs, optionally one or more Ethernet Network Interfaces and optionally a switch management Function.

With some minimal configuration during the boot and initialization of the SoC it is possible to use NETC Functions with more traditional device driver software without relying on any PCIe aware software. The appropriate registers and settings are described here. In the case that some Functions are Virtual Functions (VFs), when Single Root I/O Virtualization (SR-IOV) is supported, some run time support must also be implemented in the driver for the Physical Function (PF) to support the VFs.

53.3.3.2 Enhanced Configuration Access Mechanism (ECAM)

PCIe has the concept of a Configuration Mechanism as a way to discover PCIe devices and perform basic configuration and control such as enable/disable and reset. See Config Space.

PCIe has defined the Enhanced Configuration Access Mechanism (ECAM) which provides a memory-mapped, common set of registers for each Function, occupying up to 4KB per Function, and following a specified format. To allow NETC to appear to be a PCIe End Point (that is be an RCiEP) the iEP-RC supports ECAM. The address of the ECAM can be found by referring to Memory Map and Table 4. NETC PCI Express ECAM Memory Space.

The offset within the ECAM to a specific Function's 4KB space is determined by an offset from this base using the Function's RID. Refer to Table 353 to find the RID of the various Functions.

This section lists the ECAM registers which need to change during boot/initialization of NETC.

53.3.3.3 Integrated Endpoint Register Block (IERB)

The Integrated Endpoint Register Block (IERB) is a set of registers provided to allow configuring the RCiEPs to set, for instance, default values or allocate resources between RCiEPs. Conceptually it provides the same functionality for the RCiEPs that PCIe peripheral's nonvolatile memory would provide. See Root Complex Event Collector iERB Memory Map.

IERB is not a PCIe standard concept, however its use can be important when initializing NETC. Registers in IERB are used to determine allocation of resources such as buffer and table memory.

53.3.3.4 Privileged Register Block (PRB)

The Privileged Register Block (PRB) is a set of registers provided to allow operational control of RCiEP for reset and global error management.

PRB is not a PCIe standard concept, however its use can be important during operation of NETC. Refer to the privileged register section of each iEP's register descriptions for more details.

53.3.3.5 Root Complex Event Collector (RCEC)

The iEP-RC includes a Root Complex Event Collector (RCEC). This functionality is included for use with a full PCIe stack and can generally be ignored if such a stack is not being implemented.

53.3.3.6 Peripheral Device Register Addresses

Each Function's registers are located in one (or more) 64KB, aligned, page(s) in the address map of the SoC. PCIe generally uses Base Address Registers (BARs), in ECAM, to assign addresses to a Function's registers in the SoC address map during the PCIe enumeration (discovery) process. NETC makes use of a PCIe concept known as Enhanced Allocation (EA), which is only permitted for iEPs, and which allows the peripheral device registers to be at fixed location within an SoC's address map.

These addresses are then listed in PCIe Capability structures for each Function in ECAM and can be discovered during the enumeration process. However, it is not necessary to discover these addresses through PCIe enumeration because these addresses are fixed and known for a given SoC address map. The BARs in ECAM are not used.

Refer to Memory Map and the iEP-RC memory map tables (Table 348 and Table 349) to determine the location of the peripheral device registers for each NETC Function in the SoC address map. The registers are then laid out as described in the address maps for the individual Functions. These address ranges can then be communicated to the device drivers using the mechanism supported by the RTOS or run time environment in use, e.g. device tree entries or XML or json config files.

53.3.3.7 Steps to Initialize RCiEP Peripherals for Direct Use

After initializing any static configuration values in the Integrated Endpoint Register Block (IERB) the following are the high level steps for initializing an RCiEP Function for use:

1. Enable the Function
2. Initialize Message Signaled Interrupts (MSI) for use with the Function

Each of these steps may involve multiple sub-steps and are outlined in following sections. These steps will need to be repeated following a Function Level Reset, for that Function.

53.3.3.8 Enabling Peripheral Devices

Each Function must be enabled through configuration registers in it's ECAM space. These registers are described here. If SR-IOV is supported, then so-called Virtual Functions are handled differently than non-virtual (Physical) Functions. All non-virtual Functions require the same steps for enabling.

53.3.3.8.1 Enabling Non-virtual Functions

The Bus Master Enable and Memory Access enable bits in the PCI command register in a Function's ECAM must be set (to 1) in order to enable the Function.

53.3.3.8.2 Enabling Virtual Functions (VFs)

VFs must be enabled in their corresponding PF's ECAM registers. The number of VFs must be set in the PF's PCIE_CFC_SRIOV_NUM_VFS register and the VFs must be enabled by setting VF_ENABLE and VF_MSE in the PFs PCIE_CFC_SRIOV_CTL register in ECAM.

Each VF must then be enabled by setting the Bus Master Enable in PCI_CFH_CMD register in the VF's ECAM.

53.3.3.9 Interrupts

NETC peripheral devices make use of Message Signaled Interrupts (MSI's) specifically the "eXtended" PCIe version (MSI-X). When using the MSI paradigm a device signals an interrupt by writing a "message" to a specified address. This write is then translated into an interrupt at a core associated with the address/message pair. See Message Signaled Interrupt (MSI-X) for more information on this.

Each Function has a (possibly single entry) table of MSI {address, message} tuples which must be populated for it during initialization. The driver then selects which interrupt to associate with a specific event (e.g. transmit completion) by configuring an index into this table in a device register.

The MSI table for each Function is located in a separate 64KB, aligned, page for each Function and, as for device registers, the address of these pages can be found by referring to the iEP-RC memory map tables in Memory Map (Table 348 and Table 349). Note that PCIe allows for this table to be at an offset within this page, as indicated by the Function's PCI_CFC_MSIX_TABLE_OFF_BIR, however this offset is normally 0 for NETC Functions.

See Message Signaled Interrupt (MSI-X) for more information on MSI-X and the format of the MSI-X vector table.

53.3.3.9.1 Setting the Size of the MSI Tables

During initialization the sizes of the various Function's MSI-X vector tables can be changed by setting values in the IERB, within the supported ranges.

For ENETC the number of vectors assigned to each SI (PF and any VFs) must be set using the PSI*a*CFGR2 register (a is the SI number) in the PF's register space.

Number of MSI-X entries allocated should at least meet the minimum expectations of the drivers for each device.

53.3.3.9.2 Populating the MSI-X Vector Table and Enabling MSI-X

Each Function's MSI-X vector table must then be populated with entries appropriate for the SoC MSI-X translation mechanism.

Finally, MSI-X must be enabled by setting MSIX_EN and clearing FUNC_MASK in the Function's PCI_CFC_MSIX_MSG_CTL register in ECAM.

53.3.3.10 Functionality Provided by PFs for Their VFs

This section is just a brief summary of the functionality provided by a PF for its VFs. For a more comprehensive understanding of the full set of functionalities, refer to the following sections of this document.

The intent for VFs in SR-IOV is that they be assigned to Virtual Machines. As a result, they do not have full, direct control over some of the capabilities available to them. The entity controlling the PF controls these capabilities and the entity using the VF must ask the entity controlling the PF to make changes on the VF's behalf.

A mailbox style communications mechanism is provided in hardware to allow the driver software on a VF to send a message to the driver software on the PF requesting the PF driver to make changes for the VF driver. See PSI-VSI Messaging for more information about this mechanism.

53.3.4 Memory Map

NOTE

Systems may have software which runs at different privilege levels. In this document the term privileged or privileged software refers to the most privileged software layer, which could be Virtual Memory Manager (VMM) an operating system or software hypervisor.

This section provides a detailed description of all accessible iEP-RC memory and registers. Reserved bits should be written with 0 and ignored on read. Unused registers read zero and a write has no effect. The mapping of iEPs within the PCI Express ECAM (Enhanced Configuration Access Mechanism) memory allocated for iEP-RC, can be seen in table Table 348.

The Root Complex memory map is 16MB, of which ECAM space accounts for 8MB. The ECAM size limits the PCIe hierarchy an 8 bus topology. The remaining Root Complex memory space is reserved for iEP device registers.

The base address location of this memory map in SoC is 0x60000000.

Table 348. Root Complex Memory Space (16MB)

Address Offset	Registers
0x00_0000-0x7F_FFFF (8MB)	iEP PCIe ECAM Config (4KB per function, 32KB per device, 1MB per bus, supporting an 8 bus PCIe topology)
iEP 0: 0x00_0000 - 0x00_7FFF(8 x 4KB)	NETC: (device 0)
iEP 1: 0x0F_8000 - 0x0F_FFFF (8 x 4KB)	RCEC: (device 31)
0x80_0000 - 0x8F_FFFF (1MB)	iEP Register Blocks (64KB per endpoint)
iEP 0: 0x80_0000 - 0x80_FFFF (64KB)	NETC
iEP 1: 0x81_0000 - 0x81_FFFF (64KB)	RCEC
0x90_0000 - 0x9F_FFFF (1MB)	Privileged Register Blocks (64KB per endpoint)
iEP 0: 0x90_0000 - 0x90_FFFF (64KB)	NETC
0xA0_0000 - 0xFF_FFFF (6MB)	iEP 0: NETC device registers

Table 349. NETC Root Complex Memory Map Layout

Aperture Size of BAR	Function	EA Address Offset in NETC Root Complex space	PCIe EA BAR Equivalent Index
8MB	PCIe Config (ECAM)	0x00_0000	n/a
1MB	IERB	0x80_0000	n/a
1MB	Privileged	0x90_0000	n/a
1MB	F2 Switch	0xA0_0000	0
256KB	PF3 ENETC instance 0	0xB0_0000	0
256KB	PF4 ENETC instance 1	0xB4_0000	0
128KB	F0 Timer	0xB8_0000	0
128KB	F1 EMDIO	0xBA_0000	0
64KB	F0 MSI-X	0xBC_0000	2
64KB	F1 MSI-X	0xBD_0000	2

Table continues on the next page...

Table 349. NETC Root Complex Memory Map Layout (continued)

Aperture Size of BAR	Function	EA Address Offset in NETC Root Complex space	PCIe EA BAR Equivalent Index
64KB	F2 MSI-X	0xBE_0000	2
64KB	PF3 MSI-X	0xBF_0000	2
64KB	PF4 MSI-X	0xC0_0000	2
64KB	PF4 VF1	0xC1_0000	9
64KB	PF4 VF1 MSI-X	0xC2_0000	11

53.3.4.1 Root Complex Event Collector iERB Memory Map

Within the 8MB ECAM address space allocated to the Integrated Endpoint Root Complex by the SoC, a 64KB Integrated Endpoint Register Block (iERB) exists per iEP. The RCEC is an iEP on bus 0 and hence has its own iERB space. The iERB is used for pre-boot initialization, debug and non-customer configuration. The lower 32KB of the iERB are reserved for architected registers required for Integrated Endpoints. The upper 32KB is available for use by RC(EC). The table below shows how the iERB memory space is divided for the Event Collector.

Table 350. Event Collector Integrated Endpoint Register Block (iERB) Memory Map

Address Offset	Registers
0x0000-0x0FFF	Function 0 configuration (RCEC 0)
0x1000-0x7FFF	Reserved (architected for expansion of additional functions)
0x8000-0xFFFF	Reserved (user space)

53.3.4.2 PCI Express ECAM Event Collector config register descriptions

This section describes the PCI Express Type0 config header and capabilities as they relate to the Event Collector of the iEPRC. The Event Collector uses a single function and PCIe function that is not present will return a Vendor ID of 0xFFFF and 0's for all other register accesses within that function.

53.3.4.2.1 PCI_Config_Header_Type0 memory map

IERC_F0_PCI_HDR_TYPE0 base address: 600F_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PCI device ID and vendor ID register (PCI_CFH_DID_VID)	32	R	E001_1957h
4h	PCI command register (PCI_CFH_CMD)	16	RW	0000h
6h	PCI status register (PCI_CFH_STAT)	16	R	0010h
8h	PCI revision ID and classcode register (PCI_CFH_REVID_CLASSCODE)	32	R	0807_0002h
Ch	PCI cache line size register (PCI_CFH_CL_SIZE)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
Eh	PCI header type register (PCI_CFH_HDR_TYPE)	8	R	00h
2Ch	PCI subsystem vendor ID register (PCI_CFH_SUBSYS_VID)	16	R	1957h
2Eh	PCI subsystem ID register (PCI_CFH_SUBSYS_ID)	16	R	E001h
34h	PCI capabilities pointer register (PCI_CFH_CAP_PTR)	8	R	40h
3Ch	PCI interrupt line register (PCI_CFH_INT_LINE)	8	RW	00h
3Dh	PCI interrupt pin register (PCI_CFH_INT_PIN)	8	R	01h
40h	PCI PCIe capabilities list register (PCI_CFC_PCIE_CAP_LIST)	16	R	8010h
42h	PCI PCIe capabilities register (PCI_CFC_PCIE_CAP)	16	R	00A2h
44h	PCI PCIe device capabilities register (PCI_CFC_PCIE_DEV_CAP)	32	R	0000_0000h
4Ah	PCI PCIe device status register (PCI_CFC_PCIE_DEV_STAT)	16	R	0000h
5Ch	PCI PCIe root control register (PCI_CFC_PCIE_ROOT_CTL)	16	RW	0000h
60h	PCI PCIe root status register (PCI_CFC_PCIE_ROOT_STAT)	32	RW	0000_0000h
80h	PCI PCI-PM capabilities list register (PCI_CFC_PCIPM_CAP_LIST)	16	R	0001h
82h	PCI PCI-PM capabilities register (PCI_CFC_PCIPM_CAP)	16	R	0003h
84h	PCI PCI-PM control and status register (PCI_CFC_PCIPM_CTL_STAT)	16	RW	0008h
87h	PCI PCI-PM capabilities data register (PCI_CFC_PCIPM_DATA)	8	R	00h
100h	PCIe AER extended capability header (PCIE_CFC_AER_EXT_CAP_HDR)	32	R	1381_0001h
12Ch	PCIe AER root error command register (PCIE_CFC_AER_ROOT_ERR_CMD)	32	RW	0000_0000h
130h	PCIe AER root error status register (PCIE_CFC_AER_ROOT_ERR_STAT)	32	RW	0000_0000h
134h	PCIe AER error source identification register (PCIE_CFC_AER_ERR_SRC_ID)	32	R	0000_0000h
138h	PCIe RCEC Endpoint association extended capability header (PCIE_CFC_RCEC_EPA_EXT_CAP_HDR)	32	R	0001_0007h
13Ch	PCIe RCEC Endpoint association bitmap registerr (PCIE_CFC_RCEC_EPA_BITMAP)	32	R	0000_0001h

53.3.4.2.2 PCI device ID and vendor ID register (PCI_CFH_DID_VID)

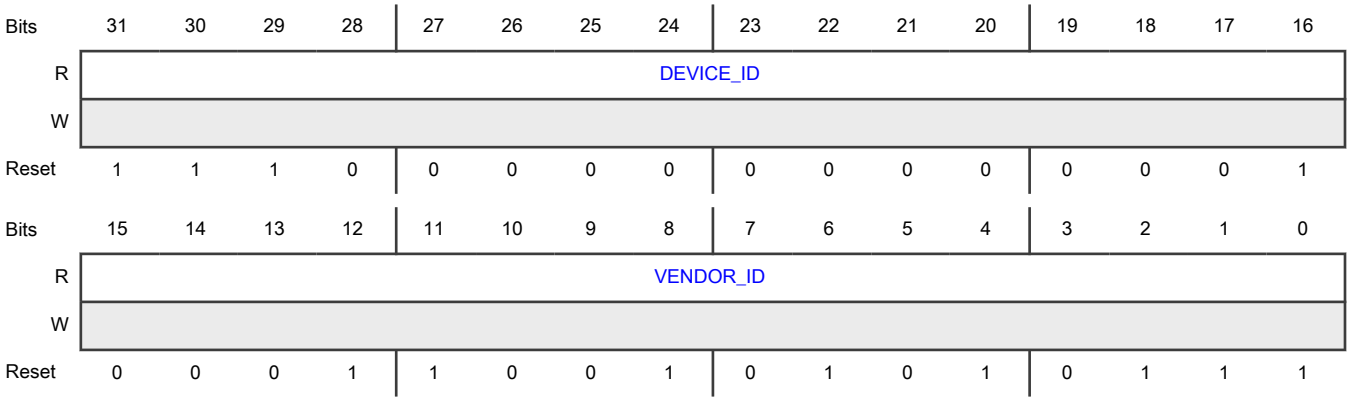
Offset

Register	Offset
PCI_CFH_DID_VID	0h

Function

This is the PCI config header device and vendor ID register.

Diagram



Fields

Field	Function
31-16 DEVICE_ID	Device ID This field identifies the device ID of the device. The reset value is determined from iERB F0_EC_CFH_DIDVID[DEVICE_ID]
15-0 VENDOR_ID	Vendor ID This field identifies the manufacturer of the device. The reset value is determined from iERB F0_EC_CFH_DIDVID[VENDOR_ID].

53.3.4.2.3 PCI command register (PCI_CFH_CMD)

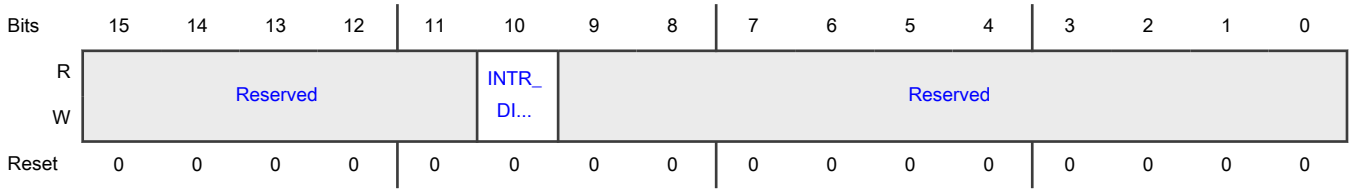
Offset

Register	Offset
PCI_CFH_CMD	4h

Function

This is the PCI config header command register.

Diagram



Fields

Field	Function
15-11 —	Reserved
10 INTR_DISABLE	Interrupt disable This field controls the ability of an Event Collect to assert the INTx wired interrupts. 0: Assertion of the INTx wired interrupts are enabled. 1: Assertion of the INTx wired interrupts are disabled.
9-0 —	Reserved

53.3.4.2.4 PCI status register (PCI_CFH_STAT)

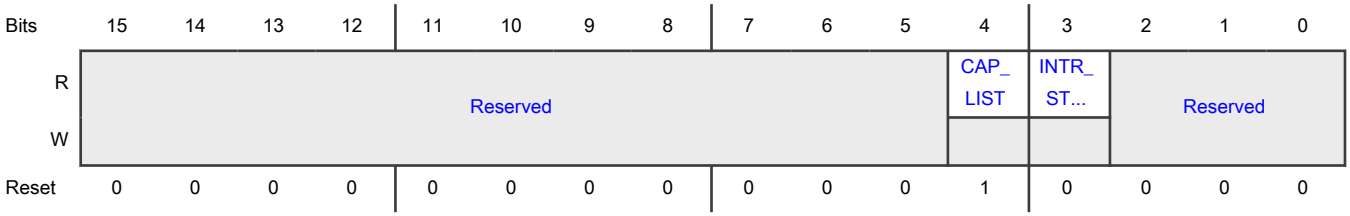
Offset

Register	Offset
PCI_CFH_STAT	6h

Function

This is the PCI config header status register.

Diagram



Fields

Field	Function
15-5 —	Reserved
4 CAP_LIST	Capabilities List Indicates the presence of an Extended Capability list item. Since all PCI Express device Functions are required to implement the PCI Express Capability structure, this bit must be hardwired to 1b.
3 INTR_STATUS	Interrupt Status When set, indicates that an INTx interrupt is pending for the Event Collector. This bit will clear automatically if interrupts are disabled (INTR_DISABLE=0b1) or the interrupt source is cleared.
2-0 —	Reserved

53.3.4.2.5 PCI revision ID and classcode register (PCI_CFH_REVID_CLASSCODE)

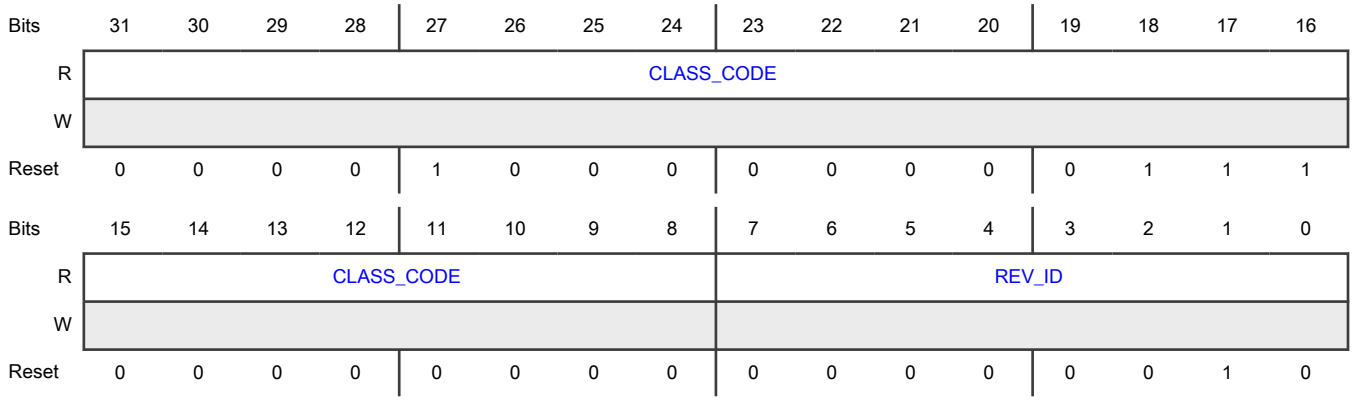
Offset

Register	Offset
PCI_CFH_REVID_CLASSCODE	8h

Function

This is the PCI config header revision ID and classcode register.

Diagram



Fields

Field	Function
31-8 CLASS_CODE	Class code The Class Code register is read-only and is used to identify the generic function of the device and, in some cases, a specific register level programming interface. The register is broken into three byte size fields. The upper byte (at offset 0Bh) is a base class code which broadly classifies the type of function the device performs. The middle byte (at offset 0Ah) is a sub-class code which identifies more specifically the function of the device. The lower byte (at offset 09h) identifies a specific register-level programming interface (if any) so that device independent software can interact with the device. Hardcoded to 0x080700 to indicate Event Collector.
7-0 REV_ID	Revision ID This register specifies a device specific revision identifier and is a vendor defined extension to the Device ID.

53.3.4.2.6 PCI cache line size register (PCI_CFH_CL_SIZE)

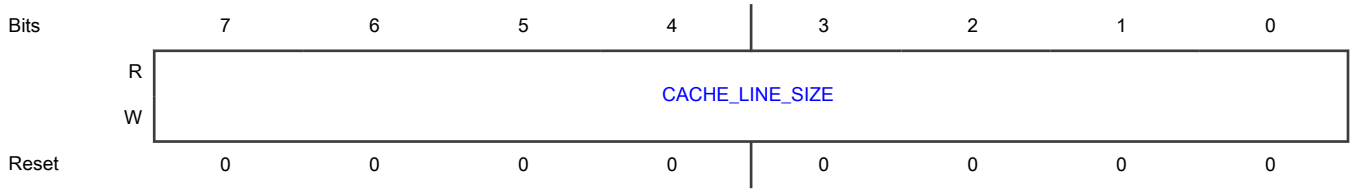
Offset

Register	Offset
PCI_CFH_CL_SIZE	Ch

Function

This is the PCI config header cache line size register.

Diagram



Fields

Field	Function
7-0 CACHE_LINE_SIZE	Cache line size The Cache Line Size register is set by the system firmware or the operating system to system cache line size. This field is implemented by PCI Express devices as a readwrite field for legacy compatibility purposes but has no effect on any PCI Express device behavior.

53.3.4.2.7 PCI header type register (PCI_CFH_HDR_TYPE)

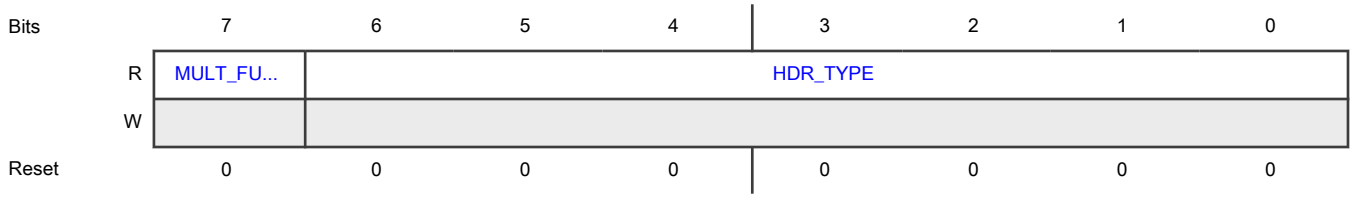
Offset

Register	Offset
PCI_CFH_HDR_TYPE	Eh

Function

This is the PCI config header header type register.

Diagram



Fields

Field	Function
7 MULT_FUNC_DEV	Multi-function device When set, indicates that the Root Complex contains multiple Event Collectors.
6-0 HDR_TYPE	Header type This field identifies the layout of the second part of the predefined header (beginning at byte 10h in Configuration Space). This register is hardwired to 00h (Type 0 header for Event Collector).

53.3.4.2.8 PCI subsystem vendor ID register (PCI_CFH_SUBSYS_VID)

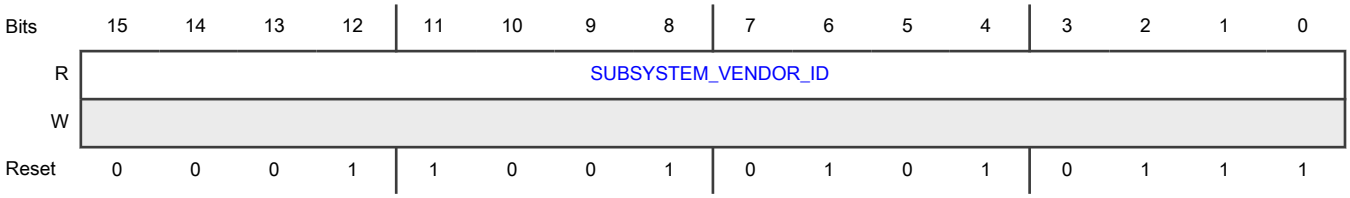
Offset

Register	Offset
PCI_CFH_SUBSYS_VID	2Ch

Function

This is the PCI config header subsystem vendor ID register.

Diagram



Fields

Field	Function
15-0 SUBSYSTEM_VENDOR_ID	This read only field identifies the manufacturer of the subsystem. The reset value is determine from iERB F0_EC_CFH_SIDSVID[SUBSYSTEM_VENDOR_ID].

53.3.4.2.9 PCI subsystem ID register (PCI_CFH_SUBSYS_ID)

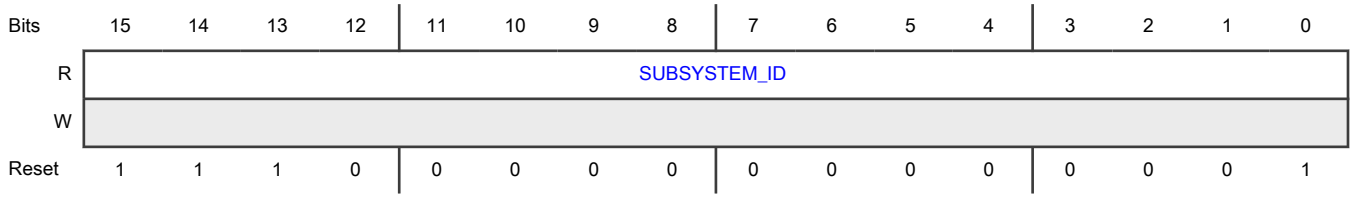
Offset

Register	Offset
PCI_CFH_SUBSYS_ID	2Eh

Function

This is the PCI config header subsystem ID register.

Diagram



Fields

Field	Function
15-0 SUBSYSTEM_ID	This read only field identifies the particular subsystem. The reset value is determine from iERB F0_EC_CFH_SIDSVID[SUBSYSTEM_ID].

53.3.4.2.10 PCI capabilities pointer register (PCI_CFH_CAP_PTR)

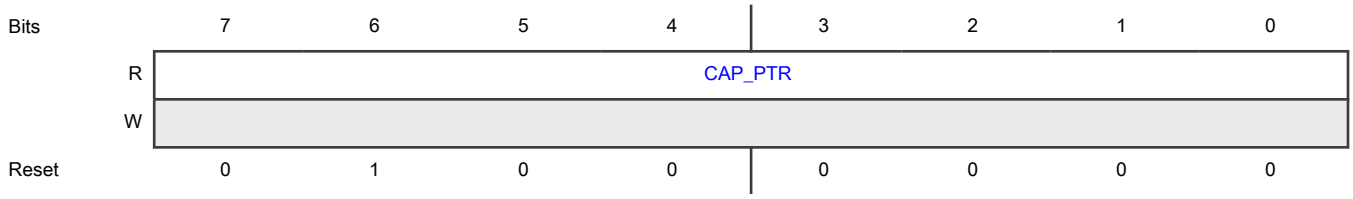
Offset

Register	Offset
PCI_CFH_CAP_PTR	34h

Function

This is the PCI config header capabilities pointer register.

Diagram



Fields

Field	Function
7-0 CAP_PTR	This register is used to point to a linked list of new capabilities implemented by this device. This register is only valid if the “Capabilities List” bit in the Status Register is set. If implemented, the bottom two bits are reserved and should be set to 00b. Software should mask these bits off before using this register as a pointer in Configuration Space to the first entry of a linked list of new capabilities. Hardwired to 40h.

53.3.4.2.11 PCI interrupt line register (PCI_CFH_INT_LINE)

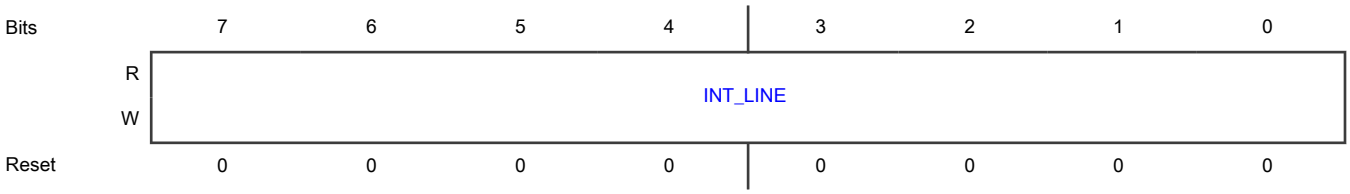
Offset

Register	Offset
PCI_CFH_INT_LINE	3Ch

Function

This is the PCI config header interrupt line register.

Diagram



Fields

Field	Function
7-0 INT_LINE	Interrupt Line register communicates interrupt line routing information. The register is read/ write and must be implemented by any Function that uses an interrupt pin. Values in this register are programmed by system software and are system architecture specific. The Function itself does not use this value; rather the value in this register is used by device drivers and operating systems.

53.3.4.2.12 PCI interrupt pin register (PCI_CFH_INT_PIN)

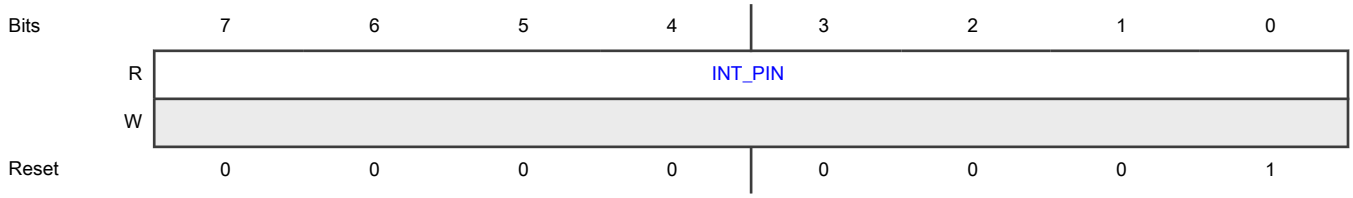
Offset

Register	Offset
PCI_CFH_INT_PIN	3Dh

Function

This is the PCI config header interrupt pin register.

Diagram



Fields

Field	Function
7-0 INT_PIN	The Interrupt Pin register is a read-only register that identifies the legacy interrupt Message(s) the function uses. Valid values are 01h, 02h, 03h, and 04h that map to legacy interrupt Messages for INTA, INTB, INTC, and INTD respectively. A value of 00h indicates that the Function uses no legacy interrupt Message(s).

53.3.4.2.13 PCI PCIe capabilities list register (PCI_CFC_PCIE_CAP_LIST)

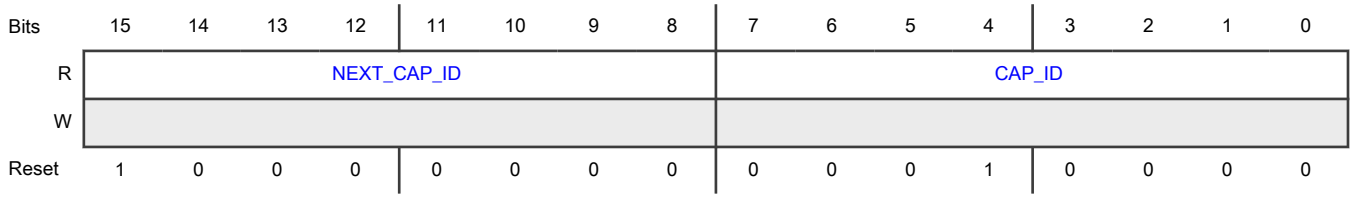
Offset

Register	Offset
PCI_CFC_PCIE_CAP_LIST	40h

Function

This is the PCI config capability PCIe capabilities list register.

Diagram



Fields

Field	Function
15-8 NEXT_CAP_ID	This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of capabilities. Hardwired to 80h.
7-0 CAP_ID	Indicates the PCI Express Capability structure. Hardwired to 10h.

53.3.4.2.14 PCI PCIe capabilities register (PCI_CFC_PCIE_CAP)

Offset

Register	Offset
PCI_CFC_PCIE_CAP	42h

Function

This is the PCI config capability PCIe capabilities register.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		INT_MSG_NUM					Reserved	DEV_PORT_TYPE				CAP_VER			
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0

Fields

Field	Function
15-14 —	Reserved
13-9 INT_MSG_NUM	Interrupt message number This field indicates which MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of this capability structure. For MSI-X, the value in this field indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. If both MSI and MSI-X are implemented, they are permitted to use different vectors, though software is permitted to enable only one mechanism at a time. If MSI-X is enabled, the value in this field must indicate the vector for MSI-X. If MSI is enabled or neither is enabled, the value in this field must indicate the vector for MSI. If software enables both MSI and MSI-X at the same time, the value in this field is undefined. Hardwired to 00h, this Event Collector does not support MSI or MSI-X.
8 —	Reserved
7-4 DEV_PORT_TYPE	Device/Port type Indicates the specific type of this PCI Express Function. Hardwired to Ah to indicate Root Complex Event Collector.
3-0 CAP_VER	Capability Version Indicates PCI-SIG defined PCI Express Capability structure version number. Hardwired to 2h.

53.3.4.2.15 PCI PCIe device capabilities register (PCI_CFC_PCIE_DEV_CAP)

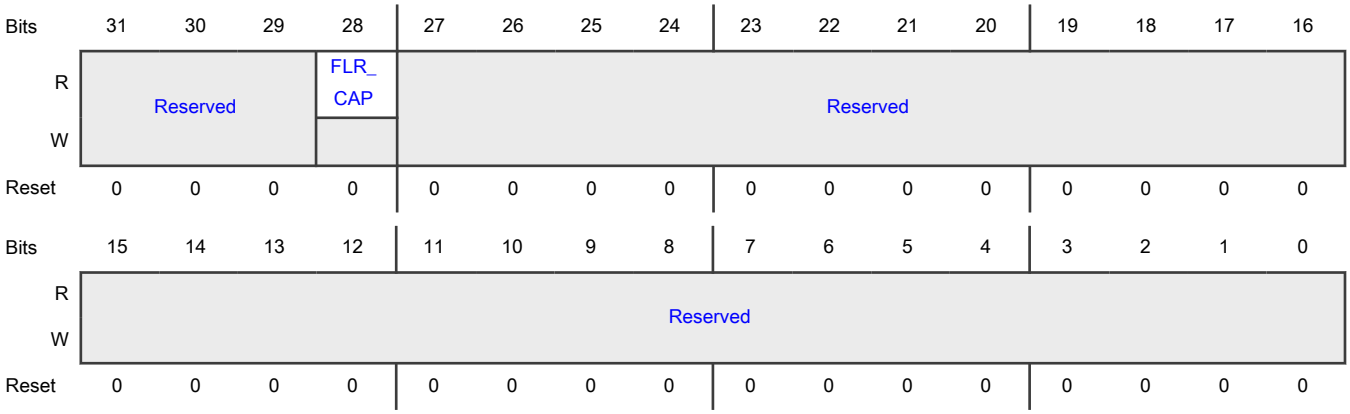
Offset

Register	Offset
PCI_CFC_PCIE_DEV_CAP	44h

Function

This is the PCI config capability PCIe device capabilities register.

Diagram



Fields

Field	Function
31-29 —	Reserved
28 FLR_CAP	Function level reset capability This bit applies to Endpoints only, hardwired to 0b.
27-0 —	Reserved

53.3.4.2.16 PCI PCIe device status register (PCI_CFC_PCIE_DEV_STAT)

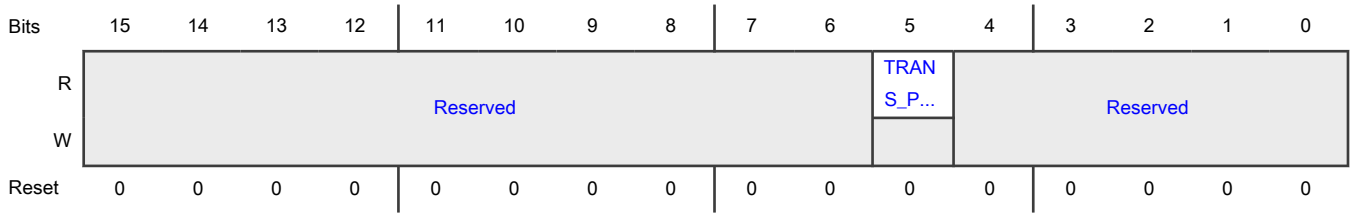
Offset

Register	Offset
PCI_CFC_PCIE_DEV_STAT	4Ah

Function

This is the PCI config capability PCIe device status register.

Diagram



Fields

Field	Function
15-6 —	Reserved
5 TRANS_PEND	Transaction pending When set, this bit indicates that the Event Collector has issued Non-Posted Requests that have not been completed. The event collector reports this bit cleared only when all outstanding Non-Posted Requests have completed or have been terminated by the Completion Timeout mechanism. Will only be non-zero if MSI-X is supported.
4-0 —	Reserved

53.3.4.2.17 PCI PCIe root control register (PCI_CFC_PCIE_ROOT_CTL)

Offset

Register	Offset
PCI_CFC_PCIE_ROOT_CTL	5Ch

Function

This is the PCI config capability PCIe root control register.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												PME_INT...	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-4 —	Reserved
3 PME_INT_EN	PME interrupt enable When Set, this bit enables PME interrupt generation upon receipt of a PME Message as reflected in the PME Status bit. A PME interrupt is also generated if the PME Status bit is Set when this bit is changed from Clear to Set.
2-0 —	Reserved

53.3.4.2.18 PCI PCIe root status register (PCI_CFC_PCIE_ROOT_STAT)

Offset

Register	Offset
PCI_CFC_PCIE_ROOT_STAT	60h

Function

This is the PCI config capability PCIe root status register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														PME_PEND	PME_STA...
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PME_REQ_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved
17 PME_PEND	PME pending This bit indicates that another PME is pending when the PME Status bit is Set. When the PME Status bit is cleared by software; the PME is delivered by hardware by setting the PME Status bit again and updating the PME Requester ID field appropriately. The PME Pending bit is cleared by hardware if no more PMEs are pending.
16 PME_STATUS	PME status This bit indicates that PME was asserted by the PME Requester indicated in the PME Requester ID field. Subsequent PMEs are kept pending until the status register is cleared by software by writing a 1b. Default value of this bit is 0b.
15-0 PME_REQ_ID	PME requester ID This field indicates the PCI Requester ID of the last PME Requester. This field is only valid when the PME Status bit is Set.

53.3.4.2.19 PCI PCI-PM capabilities list register (PCI_CFC_PCIPM_CAP_LIST)

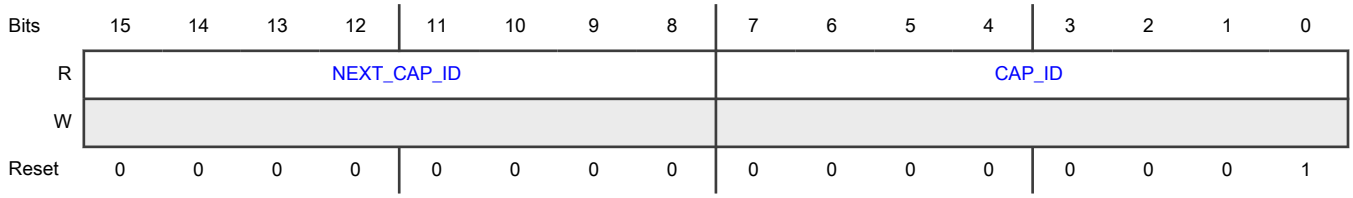
Offset

Register	Offset
PCI_CFC_PCIPM_CAP_LIST	80h

Function

This is the PCI config capability PCI-PM capabilities list register.

Diagram



Fields

Field	Function
15-8 NEXT_CAP_ID	This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of capabilities. Hardwired to 00h.
7-0 CAP_ID	Indicates the PCI-PM Capability structure. Hardwired to 01h.

53.3.4.2.20 PCI PCI-PM capabilities register (PCI_CFC_PCIPM_CAP)

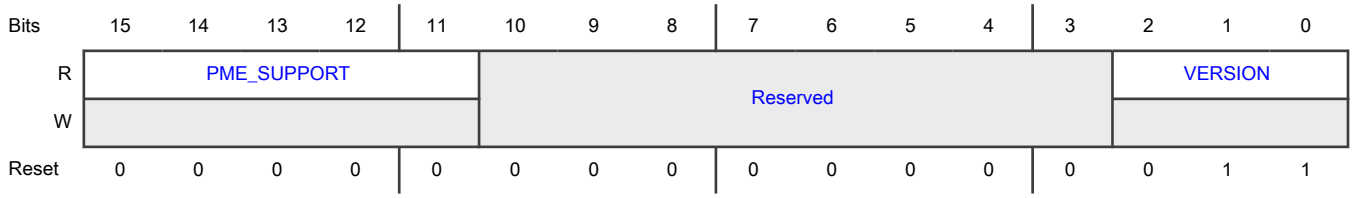
Offset

Register	Offset
PCI_CFC_PCIPM_CAP	82h

Function

This is the PCI config capability PCI-PM capabilities register.

Diagram



Fields

Field	Function
15-11 PME_SUPPORT	PME support Event Collector does not support generating PM_PME notifications, hardwired to 00h.
10-3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2-0 VERSION	Version RCEC complies with the PCI PM specification, rev 1.2.

53.3.4.2.21 PCI PCI-PM control and status register (PCI_CFC_PCIPM_CTL_STAT)

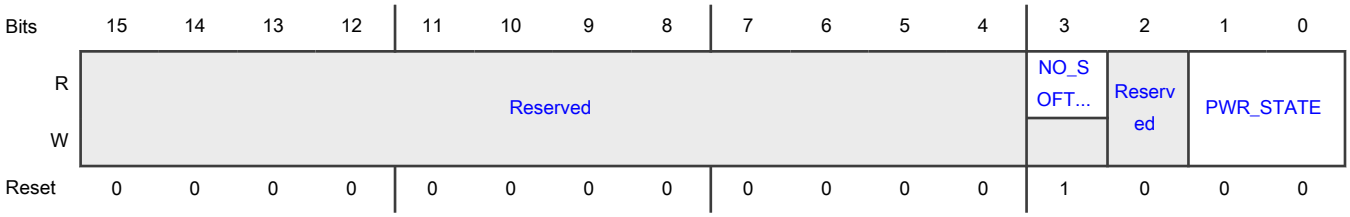
Offset

Register	Offset
PCI_CFC_PCIPM_CTL_STAT	84h

Function

This is the PCI config capability PCI-PM control and status register.

Diagram



Fields

Field	Function
15-4 —	Reserved
3 NO_SOFT_RST	No soft reset When set ("1"), this bit indicates that when RCEC transitions from $D3_{hot}$ to $D0_{active}$ because of modifying Power State bits in the PCI_CFC_PCIPM_CTL_STAT register, no internal reset is issued and Configuration Context is preserved. Upon transition from the $D3_{hot}$ to the $D0_{active}$ state, no additional operating system intervention is required to preserve PCIe Configuration Context beyond writing the Power State bits. When clear ("0"), RCEC performs an internal reset upon transitioning from $D3_{hot}$ to $D0_{uninitialized}$ via software control of the Power State bits in the PCI_CFC_PCIPM_CTL_STAT register. Configuration Context is lost when performing the soft reset. Upon transition from the $D3_{hot}$ to the $D0_{uninitialized}$ state, full re-initialization sequence is needed to return the device to $D0_{active}$. Regardless of this bit, EC preserves the PME context when PME is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 —	Reserved
1-0 PWR_STATE	Power state This field is used to set and report the power state of a function as follows: 00b = D0 01b = D1 (not supported by EC) 10b = D2 (not supported by EC) 11b = D3 If, for any reason, the operating system software attempts to put a function into a power management state that the function does not support, the function should handle this gracefully and remain in whatever state it was in before the request. The PWR_STATE bits will reflect the current state of the function not the intended invalid state which was written. If INTx is asserted when D3 is requested, it will deassert. If MSI-X is supported, any actively pending transactions will be completed before D3 state is entered.

53.3.4.2.22 PCI PCI-PM capabilities data register (PCI_CFC_PCIPM_DATA)

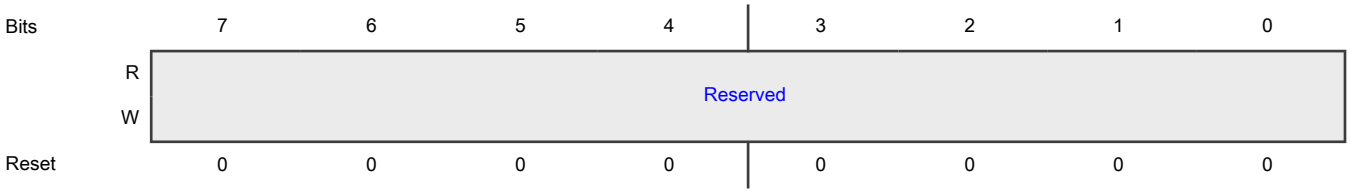
Offset

Register	Offset
PCI_CFC_PCIPM_DATA	87h

Function

This is the PCI config capability PCI-PM data register. Not supported by RCEC since AUX power not supported in $D3_{cold}$.

Diagram



Fields

Field	Function
7-0 —	Reserved

53.3.4.2.23 PCIe AER extended capability header (PCIE_CFC_AER_EXT_CAP_HDR)

Offset

Register	Offset
PCIE_CFC_AER_EXT_CAP_HDR	100h

Function

This is the PCIe config capability Advanced Error Reporting extended capability header.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NEXT_CAP_OFF												CAP_VER			
W																
Reset	0	0	0	1	0	0	1	1	1	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCIE_EXT_CAP_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-20 NEXT_CAP_OFF	Next capability offset This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of capabilities.
19-16 CAP_VER	Capability version Must be 1h for this version of the specification.
15-0 PCIE_EXT_CAP_ID	The Extended Capability ID for the AER Extended Capability is 0001h.

53.3.4.2.24 PCIe AER root error command register (PCIE_CFC_AER_ROOT_ERR_CMD)

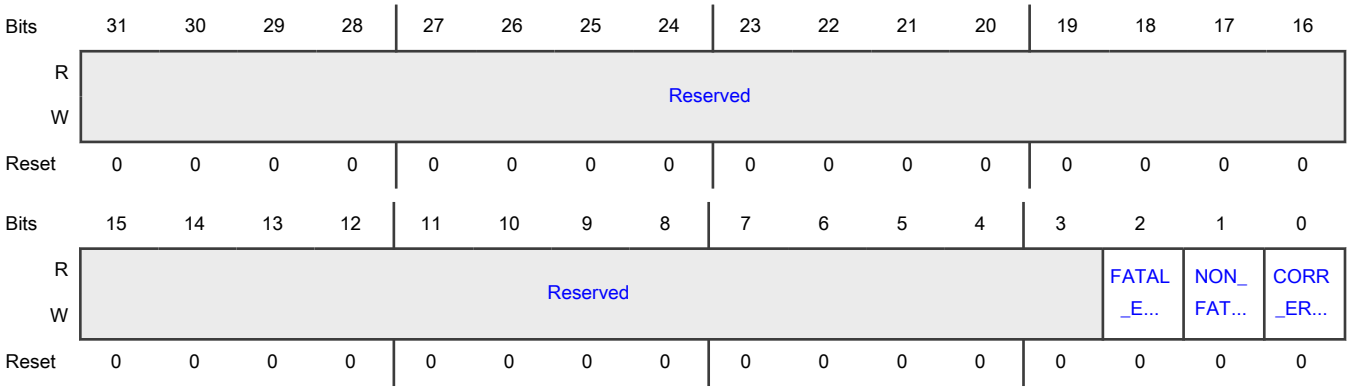
Offset

Register	Offset
PCIE_CFC_AER_ROOT_ERR_CMD	12Ch

Function

This is the PCIe config capability Advanced Error Reporting root error command register. The Root Error Command register allows further control of Root Complex Event Collector response to correctable, non-fatal, and fatal error messages reported by integrated Endpoints or the Event Collector itself.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 FATAL_ERR_RPT_EN	Fatal error reporting enable When set, this bit enables the generation of an interrupt when a fatal error is reported by any of the Integrated Endpoints (iEPs) or the Event Collector itself. Fatal uncorrectable errors for an iEP means it has been compromised and further operation is unreliable. Examples of this include multi-bit ECC errors in internal memory or other memories where the source cannot be isolated. To resume operation of the iEP, a device or function level reset is required.
1 NON_FATAL_ERR_RPT_EN	Non-fatal error reporting enable When set, this bit enables the generation of an interrupt when a non-fatal error is reported by any of the Integrated Endpoints (iEPs) or the Event Collector itself. Non-fatal uncorrectable errors require software intervention to resume operation and may need to reconfigure an iEP or a function therein. A function level reset of the iEP should not be required to continue operation.
0 CORR_ERR_RPT_EN	Correctable error reporting enable When set, this bit enables the generation of an interrupt when a correctable error is reported by any of the Integrated Endpoints (iEPs) or the Event Collector itself.

53.3.4.2.25 PCIe AER root error status register (PCIE_CFC_AER_ROOT_ERR_STAT)

Offset

Register	Offset
PCIE_CFC_AER_ROOT_ERR_STAT	130h

Function

This is the PCIe config capability Advanced Error Reporting root status command register. The Root Error Status register reports status of error messages (ERR_COR, ERR_NONFATAL, and ERR_FATAL) received by the Event Collector, and of errors detected by the Event Collector itself (which are treated conceptually as if the EC had sent an error message to itself). Each correctable and uncorrectable (Non-fatal and Fatal) error source has a first error bit and a next error bit associated with it respectively. When an error is received by the Event Collector, the respective first error bit is Set and the Requester ID is logged in the Error Source Identification register. A set individual error status bit indicates that a particular error category occurred; software may clear an error status by writing a 1b to the respective bit. If software does not clear the first reported error before another error message is received of the same category (correctable or uncorrectable), the corresponding next error status bit will be set but the Requester ID of the subsequent error Message is discarded. The next error status bits may be cleared by software by writing a 1b to the respective bit as well.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved									ERR_FAT...	ERR_NON...	FIRST_U...	MULT_ER...	ERR_FAT...	MULT_ER...	ERR_CORR
W										W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved
6 ERR_FATAL	Fatal error received Set when one or more fatal uncorrectable error messages have been received. **NOTE** This bit is "sticky" and is not cleared by PCI-PM power state change to uninitialized state.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 ERR_NON_FATAL	Non-fatal error received Set when one or more non-fatal uncorrectable error messages have been received. **NOTE** This bit is "sticky" and is not cleared by PCI-PM power state change to uninitialized state.
4 FIRST_UCORR_FATAL	First uncorrectable error fatal Set when the first uncorrectable error message received is for a fatal error. **NOTE** This bit is "sticky" and is not cleared by PCI-PM power state change to uninitialized state.
3 MULT_ERR_FATAL_NON_FATAL	Multiple fatal/non-fatal errors received Set when either a fatal or a non-fatal error is received and ERR_FATAL_NON_FATAL received is already set. **NOTE** This bit is "sticky" and is not cleared by PCI-PM power state change to uninitialized state.
2 ERR_FATAL_NON_FATAL	Fatal/Non-fatal error received Set when either a fatal or a non-fatal error message is received and this bit is not already set. **NOTE** This bit is "sticky" and is not cleared by PCI-PM power state change to uninitialized state.
1 MULT_ERR_CORR	Multiple correctable errors received Set when a correctable error message is received and ERR_CORR received is already set. **NOTE** This bit is "sticky" and is not cleared by PCI-PM power state change to uninitialized state.
0 ERR_CORR	Correctable error received Set when a correctable error message is received and this bit is not already set. **NOTE** This bit is "sticky" and is not cleared by PCI-PM power state change to uninitialized state.

53.3.4.2.26 PCIe AER error source identification register (PCIE_CFC_AER_ERR_SRC_ID)

Offset

Register	Offset
PCIE_CFC_AER_ERR_SRC_ID	134h

Function

This is the PCIe config capability Advanced Error Reporting error source identification register. The Error Source Identification register identifies the source, i.e. Requester ID (see Routing/Requester ID (RID)), of first correctable and uncorrectable (Non-fatal/ Fatal) errors reported in the Root Error Status register.

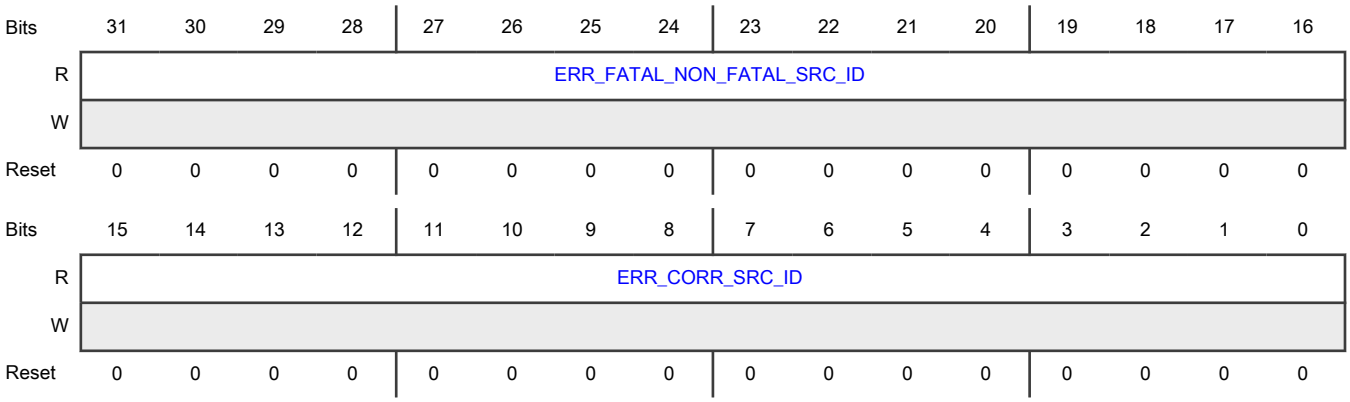
It should be noted that in the event of a global error reported by NETC, all functions will capture the error and report to RCEC, thus source ID may reflect the first function to report the error, but not necessarily the first function detecting the error.

This register is updated regardless of the settings of the Root Control register and the Root Error Command register.

NOTE

This register is "sticky" and is not cleared by PCI-PM power state change to uninitialized state.

Diagram



Fields

Field	Function
31-16 ERR_FATAL_NON_FATAL_SRC_ID	ERR_FATAL_NON_FATAL Source Identification Loaded with the Requester ID indicated in the received ERR_FATAL_NON_FATAL message when the ERR_FATAL_NON_FATAL received bit is not already set.
15-0 ERR_CORR_SRC_ID	ERR_CORR Source Identification Loaded with the Requester ID indicated in the received ERR_CORR message when the ERR_CORR received bit is not already set.

53.3.4.2.27 PCIe RCEC Endpoint association extended capability header (PCIE_CFC_RCEC_EPA_EXT_CAP_HDR)

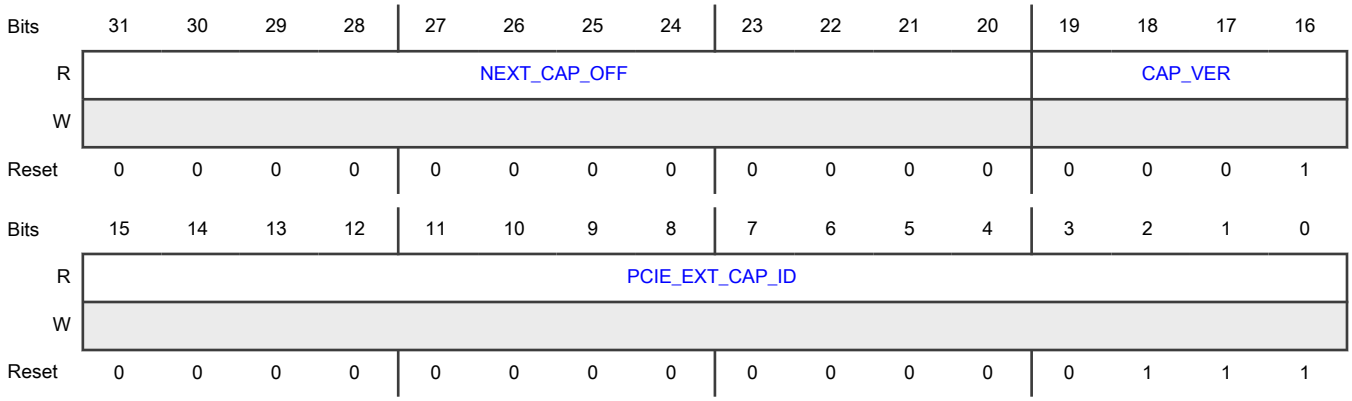
Offset

Register	Offset
PCIE_CFC_RCEC_EPA_EXT_CAP_HDR	138h

Function

This is the PCIe Root Complex Event Collector Endpoint association extended capability header register.

Diagram



Fields

Field	Function
31-20 NEXT_CAP_OFF	Next capability offset This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of capabilities.
19-16 CAP_VER	Capability version Must be 1h for this version of the specification.
15-0 PCIE_EXT_CAP_ID	The Extended Capability ID for the Event Collector Endpoint Association Capability is 0007h.

53.3.4.2.28 PCIe RCEC Endpoint association bitmap registerr (PCIE_CFC_RCEC_EPA_BITMAP)

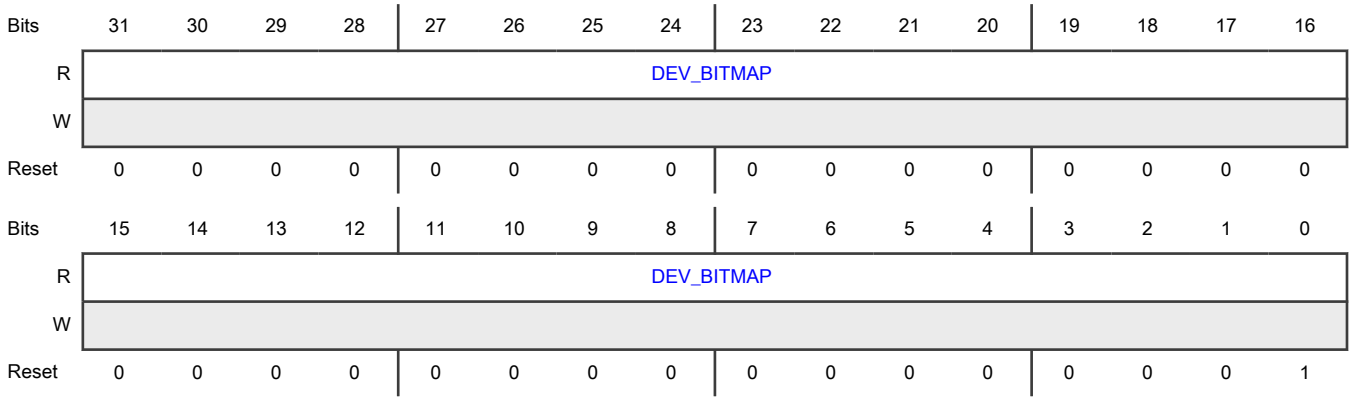
Offset

Register	Offset
PCIE_CFC_RCEC_EPA_BITMAP	13Ch

Function

This is the PCIe Root Complex Event Collector Endpoint association bitmap register. The Association Bitmap for Root Complex Integrated Endpoints is a read-only register that sets the bits corresponding to the Device Numbers of Root Complex Integrated Endpoints supported by the Root Complex Event Collector on the same Logical Bus as the Event Collector itself. The bit corresponding to the Device Number of the Root Complex Event Collector must always be set.

Diagram



Fields

Field	Function
31-0 DEV_BITMAP	Associated Endpoint device bitmap 31-0.

53.3.4.3 Event Collector Integrated Endpoint Register Block register descriptions

This section describes the Integrated Endpoint Register Block (iERB) registers. The iERB is a 64 KB size page containing registers that are used for pre-boot initialization, debug, and non-customer configuration.

The lower 32 KB of the iERB are reserved for architected registers required for Event Collector. The upper 32 KB are used for Event Collector specific resources.

53.3.4.3.1 IERB memory map

IERC_IERB base address: 6081_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Function 0 EC config header device ID and vendor ID register (F0_EC_CFH_DIDVID)	32	RW	E001_1957h
4h	Function 0 EC config header subsystem ID and subsystem vendor ID register (F0_EC_CFH_SIDSVID)	32	RW	E001_1957h

53.3.4.3.2 Function 0 EC config header device ID and vendor ID register (F0_EC_CFH_DIDVID)

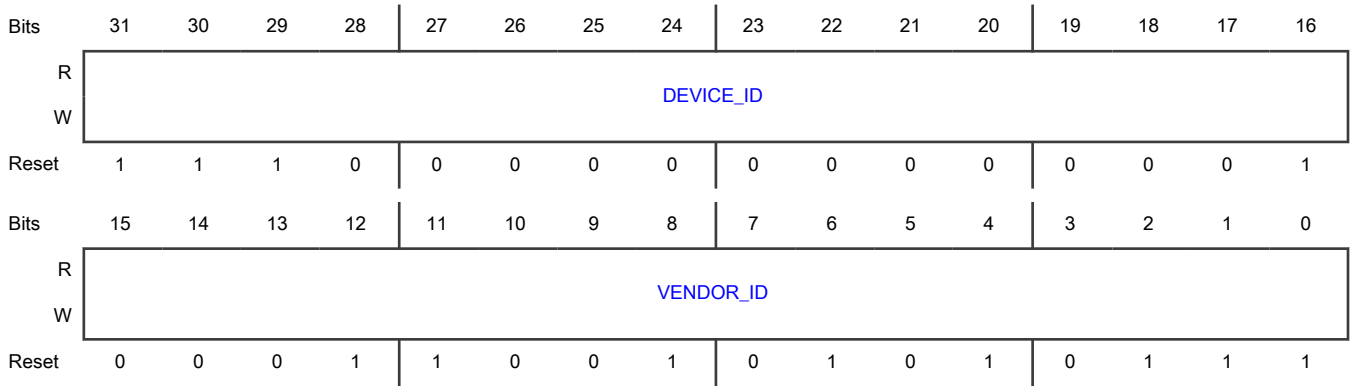
Offset

Register	Offset
F0_EC_CFH_DIDVID	0h

Function

This is the PCI Device and Vendor ID register. The values set here are reflected in the PCIe Type 0 header.

Diagram



Fields

Field	Function
31-16 DEVICE_ID	Device ID This field identifies the device ID of the device shown in the PCIe Device ID Register (02h). Default POR value is determined from the function's Device ID strap pins.
15-0 VENDOR_ID	Vendor ID This field identifies the manufacturer of the device as shown in the PCIe Vendor ID Register (00h). The default value will indicate Freescale (0x1957).

53.3.4.3.3 Function 0 EC config header subsystem ID and subsystem vendor ID register (F0_EC_CFH_SIDSVID)

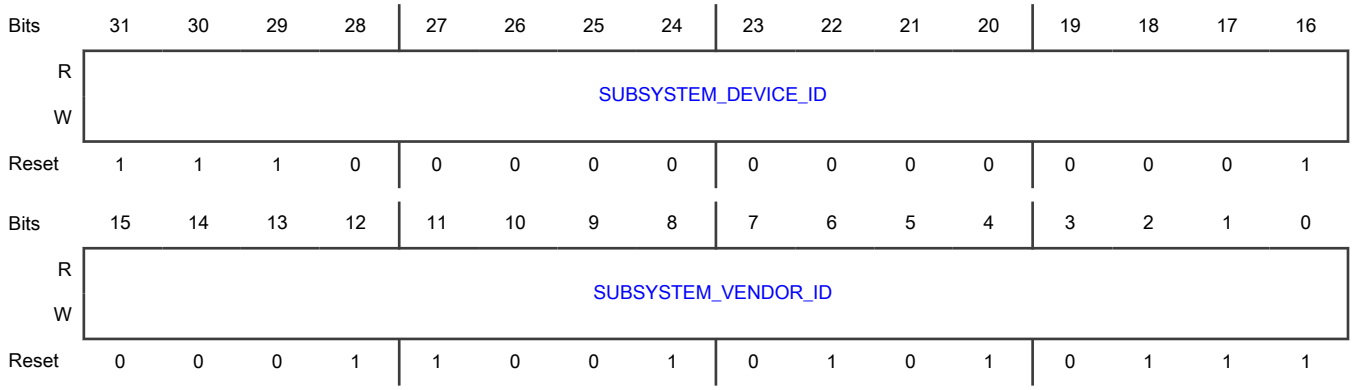
Offset

Register	Offset
F0_EC_CFH_SIDSVID	4h

Function

This is the PCI Subsystem ID and Subsystem Vendor ID register. The values set here are reflected in the PCIe Type 0 header.

Diagram



Fields

Field	Function
31-16 SUBSYSTEM_DEVICE_ID	Subsystem Device ID This field identifies the particular subsystem as shown in the PCIe Subsystem ID Register (2Eh). Default POR value is determined from the function's Device ID strap pins.
15-0 SUBSYSTEM_VENDOR_ID	Subsystem Vendor ID This field identifies the manufacturer of the subsystem as shown in the PCIe Subsystem Vendor ID Register (2Ch). The default value is the same as F0_EC_CFH_DIDVID[VENDOR_ID].

53.4 Ethernet Switch and Controller (NETC)

53.4.1 Overview

NETC is a TSN capable Ethernet IP which enables a comprehensive portfolio of Ethernet solutions. One of the major capabilities offered by NETC is the support of Time Sensitive Networking (TSN). TSN is a set of standards developed by IEEE that enables Ethernet to handle time sensitive traffic in addition to best effort traffic. TSN allows time sensitive traffic and best effort traffic to co-exist on the same network, makes Ethernet deterministic with bounded low latency, and with improved reliability and resiliency.

As shown in the NETC block diagram (see Block diagram), NETC provides a full range of Ethernet capabilities including Ethernet Controller / Network Interface Controller (NIC) and switching functionality.

NETC provides full 802.1Q Ethernet switch functionality, advanced QoS with 8 traffic classes and 4 discard resilience levels, and a full range of TSN standards capabilities.

The NIC functionality in NETC is known as ENETC (EtherNET Controller). ENETC supports virtualization/isolation based on PCIe Single Root IO Virtualization (SR-IOV), advanced QoS with 8 traffic classes and 4 discard resilience levels, and a full range of TSN standards and NIC offload capabilities. Switch CPU/host ENETC is fully integrated with the switch and does not require a back-to-back MAC, instead a light weight "pseudo MAC" provides the delineation between switch and Ethernet Controller. This translates to lower power (less logic and memory) and lower delay (as there is no serialization delay across this link).

NETC presents itself as a multi-function PCIe Root Complex Integrated Endpoint (RCiEP) for easy software discovery of peripheral functions. As such, it contains multiple PCIe functions. NETC also hosts PCIe functions for other related Ethernet functions, such as the Timer (IEEE 1588 and 802.1AS-2020) and MDIO (for managing external Ethernet PHY devices via an MDIO interface). PCIe RCiEP allows for easy software integration into OSes which support PCIe but can be easily integrated as a simple platform device for RTOSes or bare metal implementations which do not support it.

NETC connects to the Root Complex Event Collector to convey power management and error messages.

Software interacts with Ethernet through ENETC. It uses descriptor rings in memory managed by software external to NETC to transfer packets to and from ENETC. ENETC implements a DMA which is used to access these data structures and buffers.

Configuration and control of ENETC(s), as well as the switch, is implemented using a combination of registers and a command message interface implemented using descriptor rings in memory. Again, these are distinct, separate PCIe functions for ENETC(s) and the switch.

53.4.1.1 Block diagram

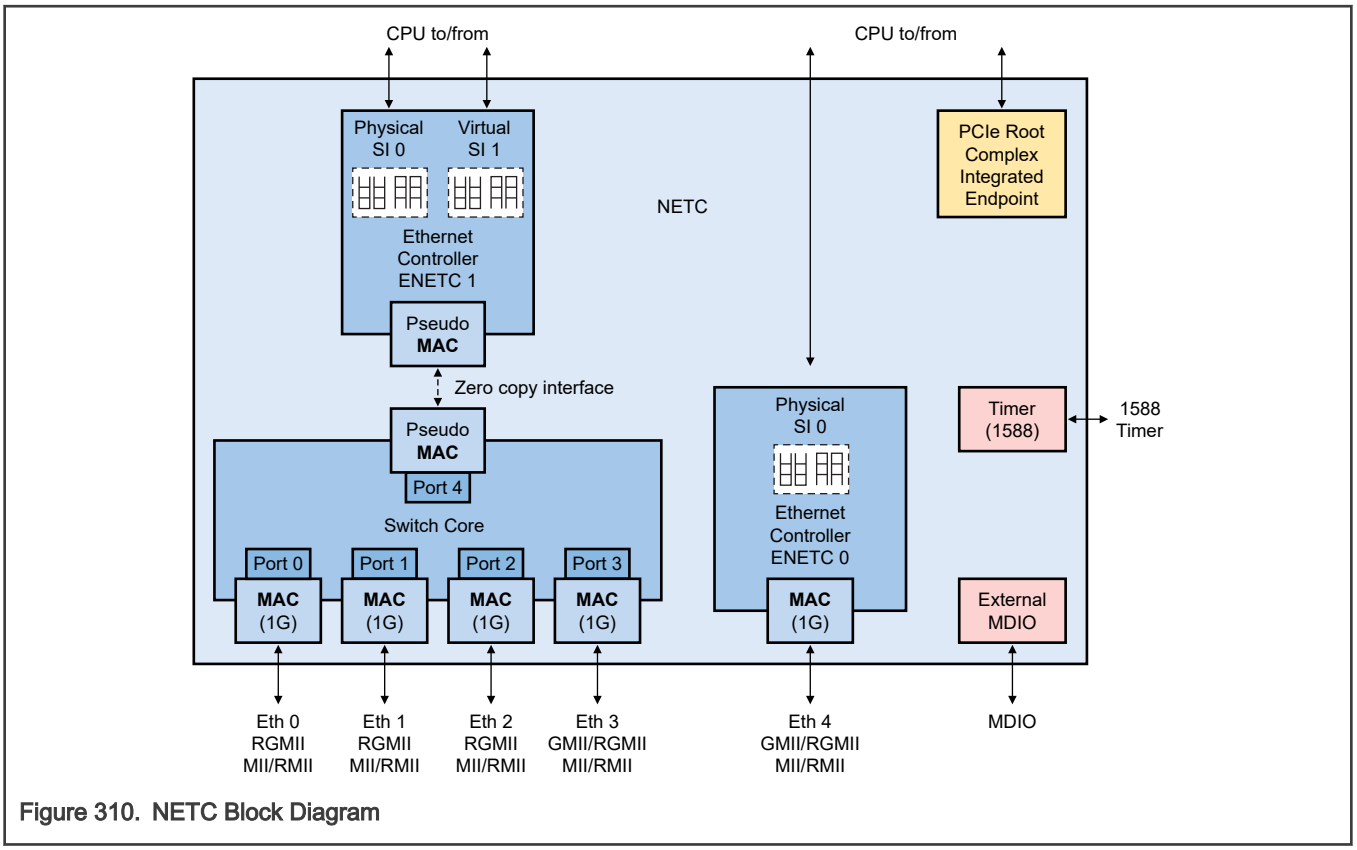


Figure 310. NETC Block Diagram

53.4.1.2 Features

Below is a list of features supported on NETC. Take note that not all of these features may be available on a SoC. See Table 351 for more details.

53.4.1.2.1 System features

- **Presents as a PCIe Root Complex Integrated End Point (RCiEP)** - For easy software discovery of peripheral Functions; leverage the infrastructure built into OSes and RTOS for PCIe.
- **Discoverable capabilities at a granular level** - Software can discover what features/capabilities exists; allow an easy upgrade path.
- **Power management** - Low power support based on PCIe Power Management. Includes wake-on-LAN on stand-alone ENETC.
- **Multifunction support per block** - Multiple functions as a single IP block using shared resources; reduced size and power.

53.4.1.2.2 802.1Q switch features

- **Wire-rate performance** - Non-blocking wire-rate switching from any port to any port at any Ethernet frame size (minimum 64 bytes) including multicast.

- **Optional cut-through for Ethernet ports** - Start transmitting frame out destination switch port before frame entirely received, to reduce latency.
- **MAC Filtering or Forwarding Database (FDB) table with guarantee minimum of entries** - Guarantee entries that can be added without table address conflict.
- **IP multicast Layer-2 forwarding** - Layer-2 forwarding based on IP multicast addresses.
- **IGMP/MLD snooping** - Capability to capture IGMP/MLD packets passing through, and have multicasting configured accordingly.
- **Hardware based address learning** - Hardware can add MAC addresses to the FDB autonomously with independent and shared VLAN learning.
- **Spanning Tree Protocols** - Spanning Tree and Multiple Spanning Tree Protocol (STP, MSTP) (and variants such as RTSP), provides basic interconnecting of multiple Ethernet switches with redundant paths, without forwarding loops.
- **Ingress filtering** - ACL style ternary match filtering on a wide set of protocol header fields (up to Layer-4) and user defined payload fields, to specify ingress packet treatments.
- **Stream identification** - Exact and ternary match stream identification. Ternary matching (implemented though ingress filtering) on a wide set of protocol header fields (up to Layer-4) and user defined payload fields. Exact matching on a wide set of Layer-2 protocol header fields and user defined payload fields.
- **Layer-2 header modification** - Comprehensive range of packet modifications: push/pop/translate of one VLAN (TPID, PCP, DEI, and VID) on ingress and/or on egress, sequence generation tag (R-TAG or HSR) modification.
- **Packet payload modification** - Layer-2 and Layer-3 payload modifications with arbitrary, dynamic content.
- **Ingress mirroring** - Ability to "mirror" or copy specific packets to another destination.
- **Copy/redirect to host** - Ability to copy or redirect specific packets to the host.
- **8 priorities (traffic classes) and 4 drop resilience levels** - Comprehensive internal QoS support with 8 priority egress queues per port scheduled using strict and/or weighted fair queuing.
- **802.1p Class of Service prioritization and tagging** - Flexible mapping of PCP on ingress and/or egress.
- **Storm control** - Controls for reducing replicated traffic of broadcast, multicast and flooded unicast/multicast traffic.
- **Time Capture** - Provides the ability to measure latency of a given frame through the switch.
- **TSN capabilities** - Full range of TSN standards, see TSN features .

53.4.1.2.3 Host interface (NIC) features

- **MAC and VLAN filtering** - Filtering for multiple MAC addresses and VLANs per port; steering to SIs (Station Interfaces) for VEPA support.
- **Virtualization** - Hardware supported isolation/virtualization through SR-IOV and Virtual Edge Port Aggregator (VEPA); support of multiple SIs (Station Interfaces) where there is one mandatory SI (SI 0) which is the Physical Station Interface (PSI) while the other optional SIs are VSI (Virtual Station Interface).
  - Hardware supported messaging between PSI and VSIs for resource and configuration requests and notifications.
  - Functional and recoverable error interrupt per SI.
  - Software controlled device, port (PF) and per interface (VF) reset capability; functionally equivalent to supported PCI function level reset (FLR)
- **Transmit and receive buffer descriptor rings to transfer packets to and from host** - Assignment of rings to any SI with isolation support. Buffer descriptor chaining support to allow frames to span multiple buffers.
- **Ingress filtering** - ACL style ternary match filtering on a wide set of protocol header fields (up to Layer-4) and user defined payload fields, to specify ingress packet treatments.

- **Stream identification** - Exact and ternary match stream identification. Ternary matching (implemented though ingress filtering) on a wide set of protocol header fields (up to Layer-4) and user defined payload fields. Exact matching on a wide set of Layer-2 protocol header fields and user defined payload fields.
- **VLAN tag extraction/insertion** - Insert predetermined tag on Tx and remove expected tag on Rx as seen by the SI. Note that this may be the second VLAN tag in the frame if the outer VLAN tag is the SI-based VLAN.
- **SI-Based VLAN** - Removal and addition of SI-based VLAN.
- **Checksum offload** - IP and TCP/UDP checksum offload for receive.
- **8 traffic classes and 4 drop resilience levels** - Comprehensive internal QoS support.
- **Interrupt coalescing control** - Interrupt coalescing can be set for each interrupt source.
- **CRC-32 generation** - CRC-32 generation and append on transmit or forwarding of user application provided FCS selectable on a per Transmit BD ring basis
- **TSN capabilities** - Full range of TSN standards, see TSN features .

53.4.1.2.4 Ethernet MAC features

- **10M/100M/GE port speeds** - Full range of standard 802.3 Ethernet speeds.
- **Half duplex support at 10M and 100M speeds** - Enables use of various low power PHYs.
- **EEE support** - Energy Efficient Ethernet.
- **Pause support** - Recognize and generate PAUSE frames with timing support for both receive and transmit.
- **One-step and two-step timestamping support for PTP/IEEE1588/IEEE802.1AS-2020** - One-step: update PTP Correction field based on current time and passed in timestamp value. Two-step: capture and report time of SFD.
- **if/ether/RMON MIB stats** - Comprehensive statistics support enables system management and debugging.

53.4.1.2.5 TSN features

- **802.1Qav-2009** - Forwarding and Queueing Enhancements for Time Sensitive Streams (FQTSS); provides credit-based shaper supports audio/video bridging (AVB).
- **802.1Qci-2017** - Per Stream Filtering and Policing (PSFP); receive side checking to ensure that the transmitter is following expected time schedule/rate profile.
- **802.1CBdb D0.5 (switch only)** - Frame Replication and Elimination for Reliability (FRER); packets are replicated at source and sent over redundant links with duplicates eliminated at destination. Often used in a loop with 2 port switches.
- **802.1Qbv-2015** - Enhancements for Scheduled Traffic (EST); time gated scheduling which provides time "slots" for specific classes of traffic in a manner similar to TDM. Host (beyond standard) can select time/time gate during which a packet will be sent; granular control by software of packet transmission.
- **802.1Qch-2017** - Cyclic Queuing and Forwarding (CQF); standardized forwarding in time slots across connected systems; Bucket brigade style forwarding provides TDM-like slots across multiple cascaded sub-systems to deliver deterministic latency.
- **802.1Qbu-2016/802.1br-2016** - Preemption; allows some traffic class(es) to interrupt the transmission of other classes.

53.4.1.2.6 Time Synchronization features

Supports IEEE 1588-2019 and PTP/802.1AS-2020

- Synchronized 64-bit nanosecond resolution time
- Common time base for all ports/switch
- Free running uncompensated time (in NETC clock ticks)
- Periodic pulse generation - Generate periodic (e.g. 1PPS) sync pulses.

- External input time capture - Capture time of 1PPS or similar pulses for sync with local systems.
- Non-periodic pulse generation at a specific future time
- Peer delay ratio support via software use of two-step timestamping

53.4.1.2.7 NETC SoC specific configurations

The table below provides a capability overview of the NETC.

Table 351. NETC SoC specific configurations

Feature	Configuration
PCIe Functions	
Timer Instances	1
EMDIO Instances	1
Switch Instances	1 (5 ports)
ENETC Instances	2
VSI Instances	1
Links	6
PCIe MSI-X Vectors	53
MACs	
Ethernet MAC	MAC 0/1/2/3/4
	10M, 100M, 1G operation
	MII/RMII/RGMII modes,
	supports half duplex,
	supports preemption
Pseudo MAC	MAC 5 - 1G pseudo MAC @ 240 MHz This figure includes the Ethernet Layer 1 overhead of 20 bytes per frame to reflect the total bits on a Ethernet physical wire.
Switch	
Aggregate bandwidth	5G at 240 MHz
Ports	5 ports - bound to links 0/1/2/3/5
ETM QOS	8
Buffer pools	8
Shared buffer pools	2
Cut-through supported	Yes
802.1Qci-2017 - Per Stream Filtering and Policing (PSFP)	Yes
802.1CBdb D0.5 - Frame Replication and Elimination for Reliability (FRER)	Yes

Table continues on the next page...

Table 351. NETC SoC specific configurations (continued)

Feature	Configuration
802.1Qav-2009 - Forwarding and Queueing Enhancements for Time Sensitive Streams (FQTSS)	Yes
802.1Qbv-2015 - Enhancements for Scheduled Traffic (EST)	Yes
802.1Qbu-2016/802.1br-2016 - Preemption	Yes
802.1Qch-2017 - Cyclic Queuing and Forwarding (CQF)	Yes
Guaranteed FDB entries	0
Egress Treatment table entries	384
Egress Count table entries	384
Egress Sequence Recovery table entries	384
Number of IPV	8
ENETC	
Ports	ENETC instance 0 - bound to link 4 (standalone)
	ENETC instance 1 - bound to link 5 (bound to switch)
Number VSI	ENETC instance 0 - 0
	ENETC instance 1 - 1
Number of Rx BD rings	14
Number of Tx BD rings	14
Number of command BD rings	5
ICM priorities	2
RSS (Receive Side Scaling) enabled	0
RFS (Receive Flow Steering) enabled	0
802.1Qci-2017 - Per Stream Filtering and Policing (PSFP)	Yes
LSO (Large Send Offload)	No
RSC (Receive Segment Coalesce)	No
Tx Checksum offload	No
Wake On LAN	Yes (ENETC instance 0 only)
802.1Qav-2009 - Forwarding and Queueing Enhancements for Time Sensitive Streams (FQTSS)	Yes
802.1Qbv-2015 - Enhancements for Scheduled Traffic (EST)	Yes
802.1Qbu-2016/802.1br-2016 - Preemption	Yes
Number of MAC Address Filter table entries	4 per ENETC
Number of VLAN Address Filter table entries	4 per ENETC
Number of RFS entries	0

Table continues on the next page...

Table 351. NETC SoC specific configurations (continued)

Feature	Configuration
Number of IPV	8
Shared between Switch, ENETC - TCAM	
Ingress Port Filter words (key can range from 2 to 14 words) - switch default	140
Ingress Port Filter words (key can range from 2 to 14 words) - ENETC 0 default	28
Ingress Port Filter words (key can range from 2 to 14 words) - ENETC 1 default	28
Shared between Switch, ENETC - common memory	
Rate Policer table entries - switch default	32
Rate Policer table entries - ENETC 0 default	2
Rate Policer table entries - ENETC 1 default	4
Ingress Stream table entries - switch default	384
Ingress Stream table entries - ENETC 0 default	8
Ingress Stream table entries - ENETC 1 default	16
Ingress Stream Count table entries - switch default	384
Ingress Stream Count table entries - ENETC 0 default	8
Ingress Stream Count table entries - ENETC 1 default	16
Ingress Sequence Generation table entries - switch default	384
Ingress Sequence Generation table entries - ENETC 0/1 default	N/A
Stream Gate Instance table entries - switch default	32
Stream Gate Instance table entries - ENETC 0 default	8
Stream Gate Instance table entries - ENETC 1 default	0
Stream Gate Control List entries per Stream Gate Instance - switch default	4 (2 admin + 2 operational)
Stream Gate Control List entries per Stream Gate Instance - ENETC 0 default	4 (2 admin + 2 operational)
Stream Gate Control List entries per Stream Gate Instance - ENETC 1 default	0
Frame Modification table entries - switch default	64
Frame Modification table entries - ENETC 0/1 default	N/A
Frame Modification Data table entries - switch default	256
Frame Modification Data table entries - ENETC 0/1 default	0

Table continues on the next page...

Table 351. NETC SoC specific configurations (continued)

Feature	Configuration
Exact match hash table entries - switch default; shared between FDB, L2 IPv4 Multicast filter, VLAN Filter, Ingress Stream Identification, Ingress Stream Filter tables	2448
Exact match hash table entries - ENETC 0 default; shared between Ingress Stream Identification, Ingress Stream Filter tables	8
Exact match hash table entries - ENETC 1 default; shared between Ingress Stream Identification, Ingress Stream Filter tables	16
Switch frame buffering - default	3138 words (20 bytes each) for total of 62,760 bytes
ENETC receive frame buffering - default	1490 words (20 bytes each) for total of 29,800 bytes
Shared between Switch, ENETC - Time Gate Scheduling memory	
Gate Control List entries - switch default	1280
Gate Control List entries - ENETC 0 default	256
Gate Control List entries - ENETC 1 default	256
Timer	
1722 enabled	No
1722 number of channels	0
Supports 802.1AS time	Yes
Number of alarms	2
Number of triggers	2
Number of fipers	3
MDIO	
Supports external emdioc (for PHYs)	Yes
Supports internal mdio (for RevMII or internal PCS)	Yes

The table below states the performance of NETC.

Table 352. NETC Performance - Throughput

Elements	Throughput
Switch - Non-blocking wire-rate switching from any port to any port at any Ethernet frame size (minimum 64 bytes) including multicast.	• 5 Gbit/s @ 240 MHz; 4 x 1 Gbit/s Ethernet port + 1 x 1 Gbit/s pseudo Ethernet port (see Note 1). Assumes standalone ENETC instance is not enabled/used. If the standalone ENETC instance is enabled (1G bit/s), wire-rate throughput (64 bytes frame size) on the Ethernet switch ports is achieved with a minimum frame size of 128 bytes on both ENETC instances.

Table continues on the next page...

Table 352. NETC Performance - Throughput (continued)

Elements	Throughput
ENETC port - Wire-rate termination at any Ethernet frame size (minimum 64 bytes).	• 2 Gbit/s @ 240 MHz; 1 x 1 Gbit/s Ethernet port + 1 x 1 Gbit/s pseudo Ethernet port (see Note 1). Assumes a 4 Gbit/s switch.
ENETC host interface - Aggregate maximum receive and transmit bit rate to/from host system memory. • Includes the Ethernet Layer 1 overhead of 20 bytes per frame to reflect the total bits on the Ethernet wire. • From a host interface perspective, each frame replicated by ENETC, is considered as a separate frame on the host Rx interface; for instance, a frame that is replicated twice, will therefore consume twice the bandwidth on the Rx host interface.	• 2 Gbit/s receive and 2 Gbit/s transmit @ 240 MHz (aggregate of 4 Gbit/s); across all ENETC instances.
ENETC host interface - Aggregate maximum receive and transmit frame rate to/from host system memory. • From a host interface perspective, each frame replicated by ENETC, is considered as a separate frame on the host Rx interface; for instance, a frame that is replicated twice, translates to 2 frames on the Rx host interface.	• 3 million frames per second receive and 3 million frames per second transmit @ 240 MHz (aggregate of 6 million frames per second); across all ENETC instances.

NOTE

Figures achievable under the conditions where all supported functions are performed, with the exception of the storm control function being disabled on the switch and the stream gating function being disabled on the ENETC instance that is attached to the switch. Pseudo link bandwidth figure includes the Ethernet Layer 1 overhead of 20 bytes per frame to reflect the total bits on an Ethernet wire.

53.4.2 Functional Description

53.4.2.1 Interface Modes of Operation

NETC has the following modes of operation.

- Interface Mode
  - MII mode
    - The port is configured for MII if PMCAPR[MII_PROT]=0b000. The MAC register IF_MODE[IFMODE] must be set to 0b001 for MII mode.
    - Link speed is controlled by IF_MODE[M10].
    - Duplex mode is controlled by IF_MODE[HD].
  - RMII mode
    - The port is configured for RMII if PMCAPR[MII_PROT]=0b001. The MAC register IF_MODE[IFMODE] must be set to 0b011 for RMII mode.
    - Link speed is controlled by IF_MODE[M10].
    - Duplex mode is controlled by IF_MODE[HD].
  - RGMII mode

- The port is configured for RGMII if PMCAPR[MII_PROT]=0b010. The MAC register IF_MODE[IFMODE] must be set to 0b100 for RGMII mode.
- Link speed is manually configured by IF_MODE[SSP].
- Note that for RGMII 10/100 Mbps modes (IF_MODE[SSP]=00 or 01), the device is dependent on receiving Rx clock (RXC) clock pulses in order to change TXC from the default 125 MHz to 25 or 2.5 MHz.
- Duplex mode for 10/100 Mbps speeds is manually configured by IF_MODE[SFD].

— GMII mode

- The port is configured for GMII if PMCAPR[MII_PROT]=0b011. The MAC register IF_MODE[IFMODE] must be set to 0b010 for GMII mode.
- Link speed mode for GMII is always 1 Gbps.
- Half duplex is not supported for GMII mode.

• Interface Options

— Half duplex

- Half duplex is supported for MII, RMII, and RGMII 10/100 Mbps
- Half duplex is not supported for speeds faster than 100 Mbps
- When half duplex is enabled with 1588 timestamping, only two-step timestamping is supported.

— Reverse Mode

- The port is configured for Reverse Mode if PIOCAPR[REVMII]=1. The MAC register IF_MODE[REVMII] must be set to 0b1 for reverse mode. When in reverse mode, the port acts as a PHY rather than as a MAC. See "Reverse Mode (RevMII) section" for details on Reverse Mode.
- Reverse mode is supported for all interface modes.

— Frame preemption

- Frame preemption is supported on a link if LaMCAPR[FP]=1
- Frame preemption is enabled on supported ports via MAC_MERGE_MMCSR[ME] = 1 or 2.
- When preemption is enabled, NETC uses both the express (0) and preemptable (1) MAC for operation.

— Internal Loopback

- The port is configured for internal loopback mode via PM_COMMAND_CONFIG[LOOP_ENA].

53.4.2.2 Common Resources

53.4.2.2.1 PCI Express Layer

NETC is operating as a multi-function RCiEP. The PCIe layer allows a PCIe device driver to discover and configure the device. NETC hosts multiple functions, including the EMDIO controller, one or more timers, one or more Ethernet controllers, zero or more switches and zero or more Virtual Station Interfaces (VSIs).

NETC connects to the Root Complex Event Collector to convey power management and error messages. It supports generation of MSI-X messages representing functional interrupts towards the system.

An ENETC that is assigned one or more VSIs operates as an SR-IOV capable physical function with one or more virtual functions.

The figure below shows a conceptual layout of the PCI functions hosted by NETC integrated endpoint.

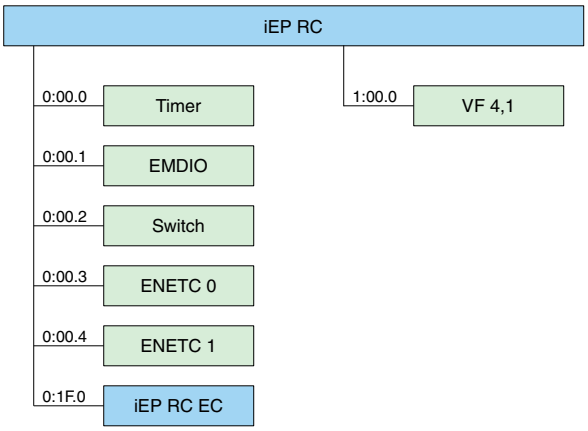


Figure 311. Root Complex Integrated Endpoint Block Diagram

53.4.2.2.1.1 Capability Discovery

NETC is discoverable as a PCIe device which has multiple PCIe functions. Each of these PCIe functions is either a physical port or a supporting function for NETC. Each physical port has one or more station interfaces associated with them. Both the port and station interfaces have capability registers to determine the overall supporting features, such as transmit/receive rings, traffic classes and offload functions.

The PCI Express device driver performs enumeration and discovers basic configuration as follows:

1. Determines available devices on the bus and number of supported functions by reading the Vendor ID field. NETC is a multi-function device.
2. Reads the Device ID from each PCIe function to determine type of function; eg. Ethernet port, IEEE 1588 etc.
3. Discovers the PCI and PCIe extended capabilities.
4. If the PCIe SR-IOV capability is present for an Ethernet port, this physical function (or physical station interface) may have additional virtual functions associated with it. Each virtual function represents a virtual station interface.
5. The PCIe Enhanced Allocation capability contains the base address of the device registers.
6. For each type of function, PCIe calls the appropriate device driver and passes the base address of the device registers.

The complete discovery sequence of physical and virtual functions for an SR-IOV multi-PF multi-function capable device is defined in PCI-SIG® Single Root I/O Virtualization and Sharing (SR-IOV) specification v1.2.

The following steps describe the basic discovery and configuration sequence by the NETC Ethernet port device driver:

1. Device driver receives the base address of the physical station interface, port and global registers.
2. Device driver receives the number of virtual station interfaces supported.
3. Device driver reads the port capability registers (PCAPRn). These registers will contain the following:
   - Support for offload functions and generic port capabilities
   - Number of transmit and receive buffer descriptor rings supported
   - Traffic classes supported
4. Device driver configures the capabilities of each virtual station interface.
   - Assign transmit and receive rings using PSI*a*CFGR0[NUM_TX_BDR] and PSI*a*CFGR0[NUM_RX_BDR].
   - Restrict transmit traffic classes using PSI*a*CFGR1[TC*b*_MAP].
   - Other capabilities may be available and can be configured as needed during run-time.

5. Software may configure the transmit and receive rings now or anytime later during run-time. See sections Transmit Buffer Descriptor Rings and Receive Buffer Descriptor Ring.
6. The port frame preemption register (MAC_MERGE_MMCSR[ME], PFPCR[FPE_TC*a*]) allow software to enable preemption as a result of further network discovery of link partners' capabilities.

53.4.2.2.1.2 PCI Express Root Complex Integrated Endpoint

NETC is operating as an SR-IOV multi-Function capable RCiEP. As such, it may span multiple devices and contain multiple functions, including the EMDIO controller, IEEE 1588 timer, one or more Ethernet host controllers, zero or more switches and zero or more Virtual Station Interfaces (VSIs). The PCIe layer allows a PCIe device driver to discover and configure the device.

It should be noted that a PCIe function that is not present will return a Vendor ID of 0xFFFF and 0's for all other register accesses within that function.

The table below shows all the functions of NETC operating as an integrated endpoint. Virtual functions are placed starting at bus 1 and are spaced 16 functions apart, allowing for a 64KB memory isolation per VSI.

Table 353. PCI Device and Function List

Bus Number	Device Number	Function Number	Description
0	0	0	**Timer** Base Class Code: 08h (Base System Peripheral) Sub-Class Code: 80h (System Peripheral) Device ID: 0xEE02
		1	**Central EMDIO** Base Class Code: 08h (Base System Peripheral) Sub-Class Code: 80h (System Peripheral) Device ID: 0xEE00
		2	**Ethernet Switch Config and Control (instance 0)** • 4 Ethernet ports (0-3) • 1 pseudo port (4) bound to ENETC instance 1 Base Class Code: 02h (Network Controller) Sub-Class Code: 00h (Ethernet Controller) Device ID: 0xEEF2
		3	**Ethernet host controller (ENETC - instance 0)** • 1 Ethernet port Base Class Code: 02h (Network Controller) Sub-Class Code: 00h (Ethernet Controller) Device ID: 0xE101
		4	**Ethernet host controller (ENETC - instance 1)** • 1 pseudo port bound to switch instance 0 - port 4. Base Class Code: 02h (Network Controller)

Table continues on the next page...

Table 353. PCI Device and Function List (continued)

Bus Number	Device Number	Function Number	Description
			Sub-Class Code: 00h (Ethernet Controller) Device ID: 0xE101
		5 - 7	Reserved
0	31	0	**Root Complex Event Collector** Base Class Code: 08h (Base System Peripheral) Sub-Class Code: 07h Device ID: 0xE001
1	0	0	VF 3,1 - VF1 for PF3 (VSI 0)

The table below lists capabilities associated with the (physical) functions that support SR-IOV.

Table 354. NETC SR-IOV PCIe Functions

Field	Physical Function 3	Description
Name	Ethernet Network Controller	Generic description
SR-IOV Cap Version	1	Corresponds to the Capability Version of SR-IOV Extended Capability Header.
Number of PF VFs	0	Corresponds to the Initial VFs (0Ch) and the Total VFs field (0Eh) of the SR-IOV capability structure. This number must be set to assign VFs to the PF.
Number of Initial VFs	0	Corresponds to the Num VFs (0Ch) of the SR-IOV capability structure. All VFs are initially visible.
PF Dependency Link	0	Corresponds to the Function Dependency Link (12h) of the SR-IOV capability structure. There is no dependency between PFs.
First VF Offset	253	Corresponds to First VF Offset field (14h). The field is fixed and cannot be modified.
VF Stride	16	VF Stride defines the Routing ID offset from one VF to the next.
VF Device ID	0xEF00	VF device ID is configured through IERB register PF*n*_IE_CFH_VFDID.
Supported Page Sizes	n/a	NETC uses PCI Enhanced Allocation with a page size of 64KB.

NETC supports the PCI Express Enhanced Configuration Mechanism (ECAM), which allows for a 4KB PCI Configuration space with a Type 0 header. The following PCI and PCI Express Extended capabilities are supported and their location within the ECAM is shown. The offset within ECAM shown in this table are for reference purposes only. Standard PCIe enumeration and discovery techniques should be used to find these structures.

Table 355. PCI and PCIe Capabilities Supported

Offset in Configuration Space	Type of Capability	PF/VF	Capability	Capability ID
0x40	PCI	Both	PCI Express Capability	10h
0x80	PCI	Both	Extended Message Signaled Interrupts (MSI-X)	05h
0x90	PCI	PF only	PCI Power Management Interface (PMI)	01h
0x9C	PCI	PF only	Enhanced Allocation (EA)	14h
0x100	PCI Express Extended	Both	Advanced Error Reporting (AER)	0001h
0x130	PCI Express Extended	Both	Access Control Services (ACS)	000Dh
0x140	PCI Express Extended	Both	Readiness Time Reporting (RTR)	0022h
0x150	PCI Express Extended	PF only	SR-IOV Capabilities	0010h

53.4.2.2.1.2.1 Enhanced Allocation

NETC does not support the standard PCI BARs (which are tied to 0). Instead the PCI Enhanced Allocation (EA) capability is used to fix the base addresses in the system. See the table below for the base addresses of the EA BEIs and the aperture size. Note that both the MSI-X table and Pending Bit Array (PBA) for a given function are located within a single 64KB page, see Message Signaled Interrupt (MSI-X) for more information about MSI-X. The offsets shown in this table are for reference only. Standard PCIe enumeration and discovery techniques should be used to determine these addresses from the EA Capability structures in the ECAM space.

Table 356. Enhanced Allocation Memory Map

F	EA BEI	Aperture Size (MaxOffset)	Offset	Description	Notes
0	0	128KB	0x0_0000	Timer base registers	Embedded function.
			0x1_0000	NETC global registers	
	2	64KB	0x0000	Timer MSI-X Table	
			0x0800	Timer MSI-X Pending Bit Array (PBA)	
1	0	128KB	0x0_0000	EMDIO base registers	Embedded function.
			0x1_0000	NETC global registers	
	2	64KB	0x0000	EMDIO MSI-X Table	
			0x0800	EMDIO MSI-X Pending Bit Array (PBA)	
2	0	1MB	0x0_0000	Switch base registers	Ethernet switch instance 0
			0x8_0000	NETC global registers	
	2	64KB	0x0000	Switch MSI-X Table	
			0x0800	Switch MSI-X Pending Bit Array (PBA)	
3	0	256KB	0x0_0000	ENETC physical station interface registers	Ethernet network controller instance 0
			0x1_0000	ENETC base registers	

Table continues on the next page...

Table 356. Enhanced Allocation Memory Map (continued)

F	EABEI	Aperture Size (MaxOffset)	Offset	Description	Notes
			0x2_0000	NETC global registers	
	2	64KB	0x0000	ENETC MSI-X Table	
			0x0800	ENETC MSI-X Pending Bit Array (PBA)	
	9	64KB	0x1_0800 x *n*	ENETC virtual station interface *n* registers, where n={0-0}, separated by aperture size bytes	
	11	64KB	0x1_0000 x *n*	ENETC virtual station interface *n* MSI-X Table, separated by aperture size bytes	
			0x1_0800 x *n*	ENETC virtual station interface *n* MSI-X Pending Bit Array (PBA), separated by aperture size bytes	
4	0	256KB	0x0_0000	ENETC physical station interface registers	Ethernet network controller instance 1
			0x1_0000	ENETC base registers	
			0x2_0000	NETC global registers	
	2	64KB	0x0000	ENETC MSI-X Table	
			0x0800	ENETC MSI-X Pending Bit Array (PBA)	

53.4.2.2.1.2.2 PCIe Function Number Assignment

The NETC PCIe device will allocate and assign PCIe function numbers starting at 0. The automatically assigned function number is shown in IERB binding register field T/EMDIO/S/E*a*BCR[FN]. Assignment is based on the following formula:

1. Only valid NETC instances, as defined by the IERB binding register field T/EMDIO/S/E*a*BCR[VALID], will be considered.
2. Valid instances will be assigned PCIe function numbers in increasing order, starting with function 0.
3. The NETC functions will be evaluated and assigned function numbers in the following order:
    a. Timer instances
    b. EMDIO instances
    c. Switch instances
    d. ENETC instances

53.4.2.2.2 NETC Table Management Protocol (NTMP)

Some NETC functionality is controlled using control messages sent to the hardware using BD ring interface with 32B descriptors similar to the packet Transmit BD ring used on ENETC. This BD ring interface is referred to as the command BD ring. This is used to configure functionality where the underlying resources may be shared between different entities or being too large to configure using direct registers (for example, lookup tables).

All of the functions (adding and removing table entries, installing schedules and so on) will be in effect by the time the command buffer descriptor is returned, assuming success except those which have a specified activation time, as noted in their description. If software is relying on a filter or rule being present or removed, it should wait for the buffer descriptor to return before taking any action based on it. Query of a function will return the state at the time it was processed by hardware. Different state may be reported if multiple requests are in-flight or manipulated by multiple entities concurrently. Ordering is guaranteed within a command BD ring that targets the same class of command (NTMP version 1.0), or the same table (NTMP version 2.0). Command BD ring selection amongst NETC functions is performed in a round robin manner.

A messaging protocol called the NETC Table Management Protocol (NTMP) is provided for exchanging configuration and management information between the software and the hardware using the command BD ring interface.

Two versions of the protocol are supported; NTMP version 1.0 and NTMP version 2.0. Following sections provide more details about the definitions of the message headers and the general message structures of each version of the messaging protocols.

53.4.2.2.2.1 NTMP Version 1.0

A command BD ring uses 32B descriptors and is similar in programming to the transmit ring operation. The generic format for the command descriptor is shown below. There is a short format where all information is provided (set command) or received (query command) within the command buffer descriptor. The long format is always accompanied by an address to a data buffer which provides (set command) or receives (query command) fields required to carry out the operation. Commands are identified by their 8-bit Class code and 8-bit Command code. Upon completion of a set command, the first 24 bytes of the set command CBD are not preserved. Upon completion of a query command, all fields in the first 24 bytes which are specific to the query command (ADDR for example) will not be preserved when the command returns; only query response specific field(s) are populated by the hardware.

A query command issued on an entry without a previous set command will have response returned with EN (enable bit) set to 0 but the rest of response fields will have undefined values. The EN field is located in the data portion of the command The format of the data portion depends on the class and command type.

Table 357. CBD Generic Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ADDR[31:0] / DATA[31:0]																																0x00
ADDR[63:32] / DATA[63:32]																																0x04
DATA[95:64]																																0x08
DATA[127:96]																																0x0C
DATA[159:128]																																0x10
DATA[191:160]																																0x14
LENGTH (SF=0) / Reserved (SF=1)																INDEX																0x18
SF	CI	STATUS							—							CLASS						CMD										0x1C

Table 358. CBD Field Descriptions

Bits	Name	Description
255	SF	Short format Determines if address field is present. 0 Long format. Address is used to indicate data buffer. 1 Short format. No data buffer, all information contained in the descriptor.
254	CI	Completion interrupt. 0 Disabled. 1 Enabled. When command BD completed for the ring, SICBDRIDR[CBDC] will be set and an interrupt will be generated if SICBDRIER[CBDCIE]=1.

Table continues on the next page...

Table 358. CBD Field Descriptions (continued)

<table>
<tr><th>Bits</th><th>Name</th><th>Description</th></tr>
<tr><td>253-248</td><td>STATUS</td><td>Status. Updated by NETC on completion of the configuration/query command. A non-zero value may indicate error or a class specific event. Note: if multiple errors are detected, only one bit in the STATUS field will be set. Unless specified, configuring reserved values of any kind, be it reserved field or reserved code point, results in undefined behavior.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>253</td><td>Reserved; set to 0</td></tr>
<tr><td>252</td><td>Integrity error
• The command did not execute due to a data integrity error (ECC on internal memory or AXI read/write error)</td></tr>
<tr><td>251</td><td>Class specific</td></tr>
<tr><td>250</td><td>Access violation error
• The entity is not allowed to perform the task requested</td></tr>
<tr><td>249</td><td>Size error
• Invalid table index, out of range.
• Table overflow, no additional entries available.</td></tr>
<tr><td>248</td><td>Format error
Buffer descriptor format is invalid, including:
• Illegal class or command.
• Invalid SF bit setting
• LENGTH is zero for long format.
• LENGTH is too small for buffer size.</td></tr>
</table></td></tr>
<tr><td>247-240</td><td>—</td><td>Reserved</td></tr>
<tr><td>239-232</td><td>CLASS</td><td>Class of command. See CMD field to determine operation.</td></tr>
<tr><td>231-224</td><td>CMD</td><td>Command. Used with class to determine operation. Setting a command value that is not supported, will result in undefined behavior.</td></tr>
<tr><td>223-208</td><td>LENGTH</td><td>Length((SF=0)
The length field specifies the effective number of bytes in the data buffer pointed to by ADDR field.</td></tr>
<tr><td>207-192</td><td>INDEX</td><td>Index
The index refers to an entry location within a table.</td></tr>
</table>

Table continues on the next page...

Table 358. CBD Field Descriptions (continued)

Bits	Name	Description
223-0	DATA[223:0]	Data. The content of this data depends on the class and command type.
63-0	ADDR	Address to data. The length of the data is defined by the LEN field. There is an alignment restrictions for the data address to be 16-byte aligned; hardware ignores the least significant 4 bits of the ADDR field.

53.4.2.2.2.2 NTMP Version 2.0

Note that the NETC IP block version used in this chip uses 32-byte long BD descriptors in the message control rings.

In the context of NTMP version 2.0 (thereafter referred to as NTMP only), a hardware table is the term used to refer to a hardware data structure, consisting of a set of similar entities, known as table entries, that share common characteristics. In other words, a table could be thought of as a set or a container of similar entries where each entry has the same structure as the other entries in the table. The figure below illustrates graphically the concept of a hardware table.

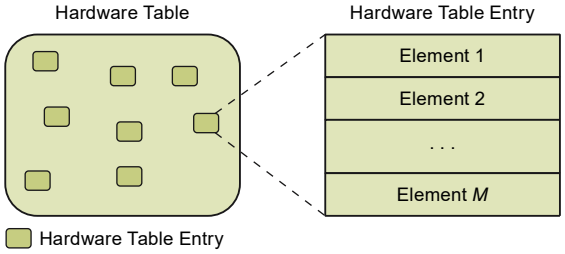


Figure 312. Hardware Tables, Entries and Elements

Because of practical considerations a table can accommodate a finite number of entries. As shown in the figure, the entries in the table do not have to be necessarily stored in a sequenced way.

Following are the ways to identify the location of an entry in a table:

- Index - an identifier that can be used to uniquely identify the location of an entry in the table in a direct manner. If a table is capable of storing up to N entries, the index value ranges from 0 to N - 1 where index 0 indicates the first entry in the table and index N - 1 indicates the last entry in the table. Accessing a table entry using an index is a fast and efficient operation.
- Opaque Entry ID - this is an implementation specific identifier, which is meaningful to the hardware but is of little or no significance to the external user of the table (that is, software). Because the value of the identifier has little or no meaning to the user, the user may not infer the location of the element in the table. An example of an opaque entry ID could be a pointer, an address or an offset identifying the location of the element in hardware managed memory. Accessing a table entry using an opaque entry ID is normally a fast and efficient operation. When an unassigned opaque entry ID is used to access an entry, hardware needs to process the command as if the entry were not found.
- Key - this is a set of attribute values stored in the entry that are used jointly to identify an entry in a table. Except in the case of ternary match tables, the NTMP protocol requires all the keys in a given table to be unique, that is, no two entries in the same table may have the same key.

NOTE

Depending on the table, the entries in the table could be accessed using more than one of the methods listed above.

In the next sections we will use the term Entry ID to identify either the index or the opaque entry ID identifiers and the term Key to identify the key entry identifier. Whenever necessary, it should be clear from the context if the Index or the Opaque Entry ID identifier is used.

Each hardware table entry contains one or many entry elements. The elements in an entry simply correspond to different parts of the data that must be stored and maintained to implement the required functionality.

53.4.2.2.2.2.1 Entry Elements

Entry elements store data needed by hardware and/or software to implement the required functionality.

The following three types of entry elements are supported:

- Key Elements - a key element contains the key data which, as explained before, is used to identify an entry in a table. Typical key data components are fields found in a frame and used to perform table lookups during frame processing, for example, MAC address, VLAN ID, and so on. Other key data components may be used to locate the key element in a table or to identify a table instance to use, for example, entry precedence or hardware resource identifier.

  Two types of Key Element exist within the NTMP protocol: the Ternary Match Key Element and the Exact Match Key Element:

  — Exact Match Key Element: used within the context of exact match tables. This key element can be used to perform any operation. When querying the Exact Match Key Element, the queried result will match exactly the data which was supplied when performing the add operation on the entry.

  — Ternary Match Key Element: used in the context of ternary match tables. Generally, this type of Key Element is composed of three pieces of information: a precedence, data and a data mask. The precedence portion of the Key Element will be used to determine the matching entry in the event that several entries' data/mask combinations match a lookup. This key element can be used to perform the add operation only. The precise data stored in the hardware will be compressed based on the mask portion of the Key Element. Thus, when querying the Ternary Match Key Element, the queried result will not match precisely the value of the Key Element which was supplied when adding the entry. When querying, the precedence portion of the key and the data mask portion of the key will always be returned as it was set by an add or update operation, The data portion, however, will be returned as an expansion of the masked data stored in the hardware; that is to say that any bit in the data whose corresponding data mask bit was a "don't care" will return as '0' when querying, regardless of the state of this bit in the original data portion of the key.

- Configuration Elements - a configuration element contains data that originates from the software control plane. The data stored in the configuration elements is used by the hardware to implement the required datapath processing.
- Status Elements - a status element contains data that typically originates from the hardware. The data stored in the status elements is used by the hardware to inform the software about the state of its resources. Software can read status elements from the hardware and in some cases software can update the element, however, hardware can change the value of the element at any time.

A table entry contains one or more elements. A typical entry will contain, likely no more than one configuration element and may contain zero, one or a number of status elements. The actual number and types of the elements present in an entry depend on the functionality supported by the given table.

53.4.2.2.2.2.2 Table Types

The following table types are supported by the NTMP protocol:

- Indexed table - linear arrays of homogeneous entries accessed using an index (0, 1, 2, ..., N-1) that uniquely identifies an entry within the arrays. The entries are typically stored in contiguous manner, meaning that entries are adjacent to one another in memory with no gaps between them. When accessing the table, the actual memory position of an entry is calculated by hardware from the index based on the size of the entry and the base memory address of the table. Given the index value of an entry, no processing or minimal processing of the index value is needed to access the table entry. Accessing an indexed table is a fast and efficient operation and, if the index value is not out of range for the table, it guarantees locating a unique entry in the table. Normally, there is no need to store the index value in the table entry. There are two types of indexed tables supported:

  — **Static bounded index table** - this table type is for the classic or traditional indexed table(s). Each of these tables has a well-defined, that is, bounded, size and memory is allocated for all the entries in the table. In these tables all the entries are pre-initialized by hardware and the entries can be accessed with the Update and Query operation at any time. The Add and Delete operations do not apply to these tables.

  — **Dynamic bounded index table** - this table type is for the tables that qualify the entries in the tables as present (or active) and absent (or inactive). Each of these tables has a well-defined, that is, bounded, size and memory is

allocated for all the entries in the table. In these tables all the entries are initially considered absent (or inactive). To add an entry to the table, the Add operation must be used. Once an entry is present in a table, it can be updated, queried, or deleted. The Delete operation removes the entry from the table. An attempt to update, query or delete an absent (or inactive) entry results in "entry not found" code being returned. Note that the word "dynamic" in the table name indicates that entries can be added to and deleted from the table in a dynamic fashion; it does not imply anything about the table memory management.

- Exact match table using hash function, also referred to as hash table - the search key must exactly match the key values stored in the table entry. All the key values must be unique. A hash function transforms the, possibly long, key into a shorter identifier which, like the index in an indexed table, is used as a direct index into the table. However, since there is the possibility of hash collisions, an additional mechanism is required to resolve the possible hash collisions. Once a new entry has been added to a hash table, a hardware-generated Entry ID (opaque entry ID) is provided to uniquely identify the location of an entry in a direct manner.
- Ternary match table using TCAM - all entry key matching fields are composed of pairs of data and mask values, to support a per-bit ternary match capability. To look for a match on a particular bit, set its mask bit to 1, and its corresponding data bit to the desired matching value, 0 or 1. A mask value of all 1's for a particular field means that field is required. To make a bit don't care, set its mask bit to 0. In a ternary match table, a single lookup key can be matched by multiple table entries, and thus, every entry must have a priority/precedent assigned by the software. When multiple entries match, the entry with the highest priority/precedence, is returned as the matching entry. Ternary match tables are typically implemented using a Ternary Content Addressable Memory (TCAM). TCAM provides deterministic and high-speed lookups. The following two types of ternary match tables are supported:
  - Hardware managed ternary match table – this table type is such that the precise location of an entry within the TCAM is managed by hardware. When adding an entry, software supplies the key and a specific precedence for this entry. The hardware will then find the correct location for the entry within the TCAM table and relocate existing entries if required based on the precedence of the newly added entry. Note that when adding multiple entries with the same Key Element, each entry will be added to the table with a new and unique Entry ID.
  - Software managed ternary match table – this table type is such that the precise location of an entry within the TCAM is managed by software. When adding an entry, software supplies the key and a specific location (Entry ID) where the entry is to be added. The hardware will then add the entry at the exact location specified by the software command. Entries at the lowest Entry ID take the highest precedence.

53.4.2.2.2.2.3 NTMP Request and Responses Messages

In the NTMP protocol, software sends request messages to hardware and receives response messages from hardware. Messages directed to hardware carry commands to be executed by hardware on a table entry or multiple entries and the responses carry the status information and response data if applicable.

53.4.2.2.2.2.3.1 Commands and Operation Types

Each NTMP command can perform one or a few of the following operations:

- Add Operation - this operation is used to add a single entry to a table. If an entry with the specified Entry ID or Key already exists in the table, that is, it has been added before, the add operation is not performed and the response message indicates that the entry has been found (NUM_MATCHED in Response Message Header is set to 1). There is one exception to this, and that is the case where an entry is added to a hardware managed ternary match table, which already exists in the table. When adding multiple entries with the same Key Element in a hardware managed ternary match table, each entry will be added to the table with a new and unique Entry ID.
- Query Operation - this operation is used to retrieve information from an existing entry or entries. If an entry with the specified Entry ID or Key does not exist in the table that is, no such entry has been added before or it has been deleted, the query operation is not performed, and the response message indicates that no entry has been found (NUM_MATCHED in Response Message Header is set to 0).
- Update Operation - this operation is used to modify an existing entry or entries. If an entry with the specified Entry ID or Key does not exist in the table, that is, no such entry has been added before or it has been deleted, the update operation is not performed, and the response message indicates that no entry has been found (NUM_MATCHED in Response Message Header is set to 0). Note that if a single update command targets multiple entries, all these entries are modified

using the same request message data. If a request message command specifies both the add and query operations, then NUM_MATCHED in the Response Message Header indicates the number of entries matched by the add operation. If NUM_MATCHED is set to 0 (no entry found), an add operation has been performed whereby if set to 1 (one entry found), the entry has been added before, and as a result, the add operation has not been performed. For the query operation in this combination command, it can be assumed that the query operation has been executed, as there will always be a match for the query operation unless an error is detected.

- Delete Operation - this operation is used to delete an existing entry or entries. If an entry with the specified Entry ID or Key does not exist in the table, that is, no such entry has been added before or has been already deleted, the delete operation is not performed, and the response message indicates that no entry has been found (NUM_MATCHED in Response Message Header is set to 0).

The NTMP protocol allows software to request multiple operation types in a single request message command. For example, the add and query operations could both be requested in one request message command to allow software to add a new entry using a key and query the Entry ID of the newly added entry. Not all the combinations of the operation types are supported in the NTMP command messages.

53.4.2.2.2.2.3.2 Operation Type Precedence

When multiple operation types are included in a request message they are always executed in the following order:

1. Add Operation first,
2. Query Operation next,
3. Update Operation next,
4. Delete Operation last.

53.4.2.2.2.2.3.3 Access Methods

Every NTMP protocol request message must specify the method to identify the table entry or entries to execute the command on. Four such access methods are supported:

- Entry ID Match - this access method uses the provided Entry ID, either an index or an opaque identifier, to identify the table entry. The Entry ID is used to access the entry in a direct way, for example, the Entry ID is used as an index or an offset in the table. In each request message, only a single entry in a table can be accessed with this access method. The Entry ID Match access method may be used on all the table types and with all the operation types except for the Add operations on tables whose Entry IDs are generated by hardware (such as exact match or ternary match tables).
- Exact Match Key Element Match - this access method uses the provided Exact Match Key Element to identify the table entry that matches the Key Element. This involves comparing the provided Key Element data with the Key Element stored in the entry. In each request message, only a single entry in a table can be accessed with this access method. When adding a new entry, the Key Element is used to calculate the unique entry ID of the entry within the table. The Exact Match Key Element Match access method may be used on Exact Match Tables.
- Ternary Match Key Element Match - this access method can only be used when performing add operations on ternary match tables. When adding a new entry using the Ternary Match Key Element, a unique entry ID for that entry is assigned by the hardware. That entry ID will remain assigned to that entry until it is deleted from the table and may be used for subsequent queries, deletion, or other table commands. The Ternary Match Key Element Match may be used on all ternary match tables, and as stated above, only the add operation is supported when using this access method.
- Search - this access method uses the provided search criteria data to identify and access the table entry or entries of interest. One or many entries in a table can be accessed with this access method, that is, all the entries that match the search criteria are accessed. The Search access method may be used on all the table types and with all the operation types except for the add operations.

53.4.2.2.2.2.3.4 Search Access Method

The Search access methods offers software a very flexible and powerful way to manage hardware. With a single search request message, software can issue a multi-operation command and execute such a command on all the entries in a given table that match flexibly defined search criteria.

The search access method has several characteristics:

- A single search message can be sent and executed on one table only. This is true for all access methods as an NTMP request message can be destined to a single table only.
- Some tables do not support the search access method.
- While executing a search operation, hardware needs to examine each entry present in the table. For each entry found, hardware checks if the search criteria match the data in the entry and, if so, performs the requested operation or operations on the entry.
- Software must allocate response data buffers that are large enough to store responses of at least one entry and, in cases of search commands, possibly multiple entries. Note that the search query operations tend to return more data than other operations and, therefore may require larger response buffers.
- For more optimal response buffer usage in search commands software can use response buffer sizes that accommodate an integral multiple of entry responses. For example, if an entry response occupies *K* bytes, the response buffer size could be set to (*F* + *n***K*) bytes, where *F* accounts for the possibly present fixed part of the response and *n* is the number of entries that can be returned in a single response.
- In each search response message hardware provides the resume entry ID which indicates if the search is finished or not. If a search is not finished, software is expected to send the provided resume entry ID back to hardware in the subsequent search continuation request message to let hardware know where in the table to resume the search.
- The, so called, NULL Entry ID is used in the request message to indicate the start of a new search. The same NULL Entry ID is used in the response message to indicate the end of a search.
- A table could be modified during a multi-message search, i.e., new entries could be added and/or existing entries could be deleted. However, depending on the table implementation, such modifications could result in duplicate table entries being found and returned to software and/or some entries not being found at all.

53.4.2.2.2.2.4 Entry ID Management

As explained at the beginning of NTMP Version 2.0, the tables rely on the Entry IDs to identify the entries in the tables. An Entry ID can be either an index or an opaque identifier. It is important to identify which entity in the system, namely software or hardware, manages the Entry IDs for each table. Entry ID management refers to allocating, de-allocating and, in general, keeping track of the availability and usage of entries in a table.

Whenever an index form of an Entry ID is required in an Add operation, software must allocate and supply the Entry ID value. Whenever an opaque Entry ID is returned in a query response, hardware must supply the Entry ID value. An entry ID is allocated through the Add operation and it is deallocated through the Delete operation.

53.4.2.2.2.2.5 NTMP Request Message Header Format

Table 359. Request Header Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ADDR																																0x0
ADDR																																0x4
REQUEST_LENGTH																RESPONSE_LENGTH																0x8

Table continues on the next page...

Table 359. Request Header Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RR=0	CCI	HEADER_VERSION						TABLE_ID								ACCESS_METHOD								CMD								0xC
																												ADD	QUERY	UPDATE	DELETE	
																																0x10
																																0x14
																																0x18
																NPF=1																0x1C

Table 360. Request Header Description

Bits	Width	Name	Description
63-0	64	ADDR	Address This field contains the address at which both the request and response data buffers are stored. Both buffers are stored at the same physical location which means that the same memory buffer is used for both of them. The memory buffer is first used as the request data buffer and then it is reused as the response data buffer. The address value stored in the ADDR field must be 16-byte aligned; hardware ignores the least significant 4 bits of the ADDR field.
83-64	20	RESPONSE_LENGTH	Response Buffer Length This field contains the length of the Response Data Buffer (data structure). Software must specify a Response Data Buffer length that is large enough to store the STATUS field if it is applicable and, in case of a query command, the response of at least one entry or, in cases of a query search command, possibly multiple entries. For more optimal memory buffer usage in search commands software can specify Response Data Buffer length that can accommodate an integral multiple of entry responses. For example, if an entry response occupies K bytes, the Response Data Buffer length could be set to (F + n*K) bytes, where F accounts for the possibly present fixed part of the response and n is the number of entries that can be returned in a single response. If this field is set to a value smaller than the size needed to store the minimum response, the command will fail with the error code “0x083” returned in the response message.

Table continues on the next page...

Table 360. Request Header Description (continued)

Bits	Width	Name	Description
			If this field is set to a value larger than the size of the allocated memory buffer used to store the Response Data Buffer, the command execution outcome is undefined. The maximum value of this field is 0xFFFF.
95-84	12	REQUEST_LENGTH	Request Buffer Length This field contains the length of the Request Data Buffer (data structure). The field must be set to the exact size of Request Data Buffer (data structure). If this field is set to a value smaller than the size of the Request Data Buffer, the command will fail with the error code “0x083” returned in the response message. If this field is set to a value larger than the size of the Request Data Buffer, the command execution outcome is undefined. The size of the allocated memory buffer that is used to store Request Data Buffer, can be larger than the REQUEST_LENGTH. For example, this can occur if the Response Data Buffer size is larger than the Request Data Buffer size, since the same memory buffer is used to store both buffers.
103-96	8	CMD	Command This field indicates the command to be issued to the hardware. A command may specify one or many operations to be executed in a well defined sequence. The supported operations are Add, Update, Delete and Query. The details of these operation are described below: Delete Operation: If the entry exists, it is deleted. If the entry does not exist, no table delete operation is performed. Update Operation: If the entry exists, the update actions specified in the UPDATE_ACTIONS field are performed and the table entry is updated with the data passed in the Request Data Buffer. If the entry does not exist, no table update operation is performed. Query Operation: If the entry exists, the table entry elements indicated by the QUERY_ACTIONS field in the request data buffer are returned in the response message. If the entry does not exist, no table query operation is performed. Add Operation: Add a new table entry if one does not exist already. If the entry already exists, no table add operation is performed. There is one exception to this, and that is the case where an entry is added to a hardware managed ternary match table, which already exists in the table. When adding multiple entries with the same Key Element in a hardware managed ternary match table, each entry will be added to the table with a new and unique Entry ID. Add is not supported when using the Search Access Method. Add is not supported when using

Table continues on the next page...

Table 360. Request Header Description (continued)

Bits	Width	Name	Description
			the Entry ID Access Method on tables whose Entry IDs are generated by hardware. 0x01 = Delete. The delete operation is performed. 0x02 = Update. The update operation is performed. 0x04 = Query. The query operation is performed. 0x05 = Query, followed by Delete. The query operation is performed first, followed by a delete operation. 0x06 = Query, followed by Update. The query operation is performed first, followed by an update operation. 0x08 = Add. The add operation is performed. 0x0A = Add or Update. If the entry exists, the Update operation is performed. If the entry does not exist, then the Add operation is performed instead. 0x0C = Add, followed by a Query. The Add operation is performed first, followed by a query operation. 0x0E = Add, followed by a Query, followed by an Update. The Add operation is performed. Then, the Query operation is performed. Then, if the entry existed prior to the Add operation of this command, the Update operation will be performed. All other field values are reserved and, if set, will result in an error code being returned in the response message.
107-104	4	--	Reserved
111-108	4	ACCESS_METHOD	Access Method This field specifies the format of the ACCESS_KEY field within the Request Data Buffer as one of the following: 0x0 = Entry ID Match. This access method indicates the Entry ID Match access method. ENTRY_ID is used as the ACCESS_KEY to access the table entry in a direct way. The Entry ID Match may be used on all tables for all operations except for the Add operations on exact match tables. 0x1 = Exact Match Key Element Match. This access method indicates the Exact Match Key Element Match access method. KEYE_DATA is used as the ACCESS_KEY and is matched against the Key Element stored in the exact match Table Entry. In general, the Exact Match Key Element Match access method may be used on all the applicable exact match tables and for all operations

Table continues on the next page...

Table 360. Request Header Description (continued)

Bits	Width	Name	Description
			0x2 = Search. This access method indicates the Search access method. The SEARCH_CRITERIA field is used as the ACCESS_KEY. The search criteria are a set of rules that are used to search for and identify all the entries that match these criteria. In general, the Search may be used on all tables for all operations except for: o Add operations 0x3 = Ternary Match Key Element Match. This access method indicates the Ternary Match Key Element Match. KEYE_DATA is used as the ACCESS_KEY. KEYE_DATA is the combination of a precedence, data, and mask value and is used when inserting entries into a Ternary Match Table. The Ternary Match Key Element Match is applicable only to ternary match tables, and is applicable only to the add operation. The NTMP protocol defines the following pieces of information for each table: - The length of the ENTRY_ID field, e.g., 16-bit or 32-bit or other and for which operations the Entry ID Match is supported. - Whether the Key Element Match access method is supported or not and if so, the specific format or formats of the Key Element for that Table ID and which operations may use the Key Element Match. Note that any reference to the Key Element Match will refer to the type of Key Element Match which is applicable to this table. For example, if a ternary match table indicates that Key Element Match is supported, it is referring to the support of the Ternary Match Key Element Match. - Whether the Search access method is supported or not and if so, the specific format of the SEARCH_CRITERIA field for that Table ID.
119-112	8	TABLE_ID	Table ID This field identifies to hardware which hardware table is the target of the request message.
125-120	6	HEADER_VERSION	Header Version This field specifies the format of the NTMP request message header 0x0 = Reserved. 0x1 = Reserved. 0x2 = version 2.0. 0x3-0x3F = Reserved.
126	1	CCI	Command Completion Interrupt

Table continues on the next page...

Table 360. Request Header Description (continued)

Bits	Width	Name	Description
			0x0 = Disabled. 0x1 = Enabled. When the processing of the request message has been completed the command completion interrupt can be generated. Whether the interrupt is generated or not depends on the settings of the message ring interrupt control register.
127	1	RR	Response Ready This flag must be set to zero by software in each request message and it is set to one by hardware in each response messages once hardware has finished processing the request message. This field is located in the same place in the request and response message headers.
238-128	111	--	Reserved
239	1	NPF	NTMP Protocol Format 0b = NTMP version 1.0 protocol is used 1b = NTMP version 2.0 or later protocol is used.
255-240	16	--	Reserved

53.4.2.2.2.2.6 NTMP Response Message Header Format

Table 361. Response Header Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
																																0x0
																																0x4
																																0x8
RR								ERROR								NUM_MATCHED																0xC
																																0x10
																																0x14
																																0x18
																																0x1C

Table 362. Response Header Description

Bits	Width	Name	Description
95-0	96	--	Reserved

Table continues on the next page...

Table 362. Response Header Description (continued)

Bits	Width	Name	Description
111-96	16	NUM_MATCHED	Number of Entries Matched This field indicates the number of entries that were matched and, if applicable returned, by the command. For example, in a search query response this field indicates the number of the matched entries returned in the response. Note also that an attempt to add an already existing entry results in this field being set to 1 which means that the entry already exists and the requested add operation was not performed. If more than 65535 matches were found then this field pegs at 0xFFFF.
123-112	12	ERROR	Error A zero value of this field indicates that the command has been executed successfully. A non-zero value of this field indicates that an error condition has occurred while executing the command and that no other fields in the response can be considered valid. Many error codes describe conditions triggered by malformed or unsupported commands. Receiving this kind of an error code means that the command has not been executed and that the hardware remains in a coherent state. These error conditions do not result in sending error notifications to the system-level error control unit. Some error codes, however, indicate more severe error conditions. Receiving this kind of an error code means that the command has not executed properly, the hardware may or may not be in a coherent state and that the system-level error control unit may have been notified about the error. When the system-level error control unit is notified about the error it also identifies the corrective action or actions to be taken. All the error codes that may result in error notifications being sent to the system-level error control unit are clearly identified as such. 0x000 = No error. 0x080 = Invalid table ID. 0x081 = Access method specified is not supported. 0x082 = Entry ID value is out of range. 0x083 = REQUEST_LENGTH or RESPONSE_LENGTH specified in the Request Header, is not sufficient. Note that in the case of a Response Data Buffer, this error code is returned only when the RESPONSE_LENGTH specified in the Request Header size is too small to accommodate the minimum response. 0x084 = Invalid command. 0x085 = Reserved.

Table continues on the next page...

Table 362. Response Header Description (continued)

Bits	Width	Name	Description
			0x086 = Multi-bit ECC or parity error observed during command processing. This error condition may trigger a notification to the system-level error control unit. 0x087 = Exceeded hash entry limit. 0x088 = Exceeded maximum hash collision chain limit and the CAM (guaranteed FDB entries) if present is full. 0x089 = Invalid ENTRY_ID for ENTRY_ID generated by hardware (hash, TCAM). 0x08B = Command for index table before OSR[ITM_STATE]=0. 0x08C = Query action specified is invalid. 0x08D = Invalid table access privilege. 0x08E = System Bus Read Error encountered while processing the command. This error condition may trigger a notification to the system-level error control unit. 0x08F = System Bus Write Error encountered while processing the command. This error condition may trigger a notification to the system-level error control unit. 0x090 = Client encountered a fault while processing the command. This error condition may trigger a notification to the system-level error control unit. 0x091 to 0x0FF = reserved. 0x100 to 0xFFF = Table specific error codes.
126-124	3	--	--
127	1	RR	Response Ready See Request Header Description - RR
255-128	128	--	Reserved

53.4.2.2.2.2.7 NTMP Request Message Data Buffer

Request and response data buffers contain information that is, generally table specific. However, the building blocks of the data buffers and how these blocks are incorporated into the data buffers are similar for all the tables.

In general, a request data buffer starts with some fields that are common for all the tables and are followed by the table specific entry element data blocks. The table below shows the structure of a typical request data buffer.

Table 363. Generic NTMP Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--												zEU		yEU		xEU	
ACCESS_KEY																																0x4
...																																...
ACCESS_KEY																																...
xE_DATA																																...

Table continues on the next page...

Table 363. Generic NTMP Request Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
...																																...
xE_DATA																																...
zE_DATA																																...
...																																...
zE_DATA																																...

The three fields, namely the UPDATE_ACTIONS, QUERY_ACTIONS and ACCESS_KEY fields are always present, regardless of the type of the table. The TABLE_VERSION field is used to version the formats of the request/response data buffers on the table ID level. In other words, each table ID value may support different formats of the request/response data buffers and these formats can be versioned using this table version field. What follows are blocks of element data; in this example. there are two such blocks for elements with names X and Z.

The following table provides the descriptions of the fields present in the generic NTMP request data buffer format.

Table 364. Generic NTMP Request Data Buffer Description

Bits	Width	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions This field contains table specific update action code-points. See section Update Actions for more details.
27-24	4	QUERY_ACTIONS	Query Actions This field contains table specific query action code-points. See section Query Actions for more details.
31-28	4	TABLE_VERSION	Table Version This field specifies the table specific formats of the NTMP request and response data buffers. See section Table Version for more details.
variable -32	variable	ACCESS_KEY	Access Key This field contains table specific access key. See section Access Key for more details.
variable-variable	variable	xE_DATA, zE_DATA	Element Data Blocks These fields contain table specific element data blocks. See section Element Data Blocks for more details.

53.4.2.2.2.2.7.1 Update Actions

The UPDATE_ACTIONS field in the request data buffer contains a set of code-points. Each of the code-points is associated with a specific table entry element and it indicates what kind of operation is to be performed on that element. Each of the code-points

may also be associated with an element data block. In the example, the X Element Update (xEU) code-point is associated with the xE_DATA element data block and the z Element Update (zEU) code-point is associated with the zE_DATA element data block. Note that in this example the yEU code-point is not associated with any element data block which means that the yEU update action does not require any element data to perform the update.

The zero value in each of the UPDATE_ACTIONS code-points means that the specific update is not to be performed. A non-zero value means that an update is to be performed by hardware. Note that depending on the value of the code-point, a different operation may be performed on the same element. In our example, the xEU and zEU update actions are both 1-bit code-points which means that only one action can be performed on the x and z elements, as the zero value indicate that no action is performed. The yEU update is a two-bit code-point which indicates that up to 3 different actions could be performed on the y element in the entry. Note also, that the yEU update action does not require any data to perform the update and this is why there is no yE_DATA element data block present in the request message. In an actual system, the yEU-like update action could be suitable for updating a statistics element where there is no need for any element data and the update actions could be: reset statistics, pause collecting statistics and re-start collecting statistics.

It is important to know that regardless of the UPDATE_ACTIONS code-point values, all the element data blocks are always present. In other words, regardless if the code-point values are zero or not, all the defined element data blocks are always present in the request data buffer, even if they are not used.

The exact number and meaning of the update actions supported is table specific.

53.4.2.2.2.2.7.2 Query Actions

The QUERY_ACTIONS field in the request data buffer defines a set of code-points that are used to indicate which elements should be returned in the response to a query request. Each of the query action code-points defines a different set of elements to be returned in a query response message. This can be used to selectively query elements in a table entry or entries. If a query action code point indicates that an entry element or some elements are not to be queried, the data for these elements will not be present in the response message data buffer; the response message data buffer contains data only for the queried elements.

The exact number and meaning of the query action code-points is table specific.

53.4.2.2.2.2.7.3 Table Version

The TABLE_VERSION field in the request data buffer defines the pair of the request and response data buffer formats used by the table specified in the TABLE_ID field of the NTMP message header. The table version allows each supported NTMP table, and identified by the table ID, to have different versions of the request and response data buffer formats. This, in turn, facilitates adding new features or modifying existing features in NTMP tables while developing new versions of IP blocks. The table version field makes it easy to version a table without having to increment the NTMP header version or having to introduce a new table which could be identical in nature to an existing table and only differ in the supported feature set.

53.4.2.2.2.2.7.4 Access Key

The ACCESS_KEY field in the request data buffer is associated with the ACCESS_METHOD field in the request message header; the interpretation of the data stored in the ACCESS_KEY field depends on the value stored in ACCESS_METHOD field in the request header. In the add operation, the ACCESS_KEY data is used by hardware to determine where in the table to add the entry. In the update or query operations, the ACCESS_KEY data is used by hardware to determine which of the existing entry or entries the request message is destined for.

The exact meaning of the data stored in the ACCESS_KEY is table specific.

53.4.2.2.2.2.7.5 Element Data Blocks

The remaining fields in the request data buffer are the different element data blocks used in the add and update operations. Each element data block contains element specific data needed to perform the add operation or the update action requested in the corresponding codepoint of the UPDATE_ACTIONS field.

In our example there are two element data blocks: xE_DATA and zE_DATA. They contain data needed to perform the update actions that can be requested through the xEU and zEU code-points, respectively. Note that, as mentioned before, the yEU update action does not require any data and therefore does not have an element data block present in the data buffer.

An important characteristic of the request messages is that while they can act on multiple entries in the table, e.g., multiple entries could be updated in a single request operation, information from the same element data blocks is applied to all the entries addressed by the request message.

As mentioned before, regardless of the UPDATE_ACTIONS code-point values, all the element data blocks are always present. In other words, regardless if the code-point values are zero or not, all the defined element data blocks are always present in the request data buffer, even if they are not used.

The exact number of the element data blocks and the meaning of the fields within them are table specific.

The request data buffer format shown in Table 363 is generic and as such it does not apply exactly as shown to all the different operation types. The following section explains how the format changes for the different operation types.

53.4.2.2.2.2.7.6 Operation Specific Request Data Buffer Formats

The request data buffer format shown in Table 363 applies to the Add and Update operations. The Query and Delete operations use a slightly different data buffer format. This is because the Query and Delete operations do not require any element data to be present in the request. A generic request data buffer format used by the Query and Delete operations is shown in the table below.

Table 365. Generic NTMP Request Data Buffer Format for Query and Delete Operations

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--													zEU	yEU	xEU	
ACCESS_KEY																																0x4
...																																...
ACCESS_KEY																																...

Note that the UPDATE_ACTIONS and QUERY_ACTIONS fields are present in the formats for all the operations, however, they are ignored by hardware if they do not apply. More accurately, the UPDATE_ACTIONS field is ignored in the Query operation and both of the fields are ignored in the Delete operation.

The comparison of the formats for the Add and Update operations (Table 363) and for the Query and Delete operations (Table 365) makes it clear that the Add/Update operation format is a superset of the Query/Delete operation format. Also, the UPDATE_ACTIONS, QUERY_ACTIONS and ACCESS_KEY fields are present in all the operation formats and are always at the same offsets in the request data buffer. This is illustrated graphically in the figure below.

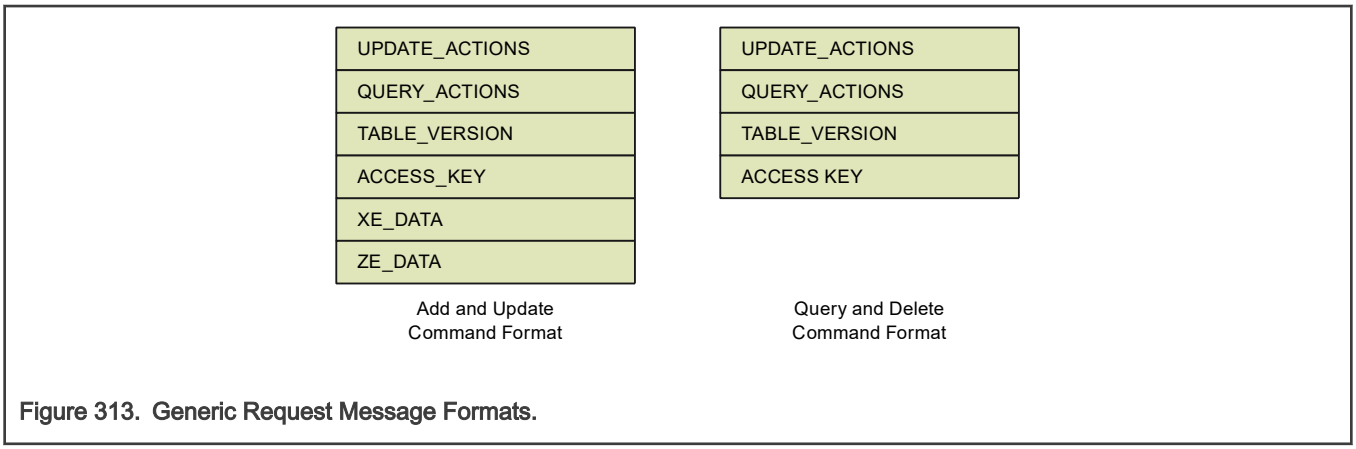


Figure 313. Generic Request Message Formats.

53.4.2.2.2.2.8 NTMP Response Message Data Buffer

Like with the request data buffer, the response data buffers use a common format. Consider Table below which shows the structure of a typical response data buffer.

Table 366. Generic NTMP Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS																																0x0
...																																...
STATUS																																...
ENTRY_1_RESPONSE_ENTRY_ID																																...
ENTRY_1_RESPONSE_xE_DATA																																...
ENTRY_1_RESPONSE_yE_DATA																																...
ENTRY_1_RESPONSE_zE_DATA																																...
...																																...
ENTRY_*i*_RESPONSE_ENTRY_ID																																...
ENTRY_*i*_RESPONSE_xE_DATA																																...
ENTRY_*i*_RESPONSE_yE_DATA																																...
ENTRY_*i*_RESPONSE_zE_DATA																																...
...																																...
ENTRY_*N*_RESPONSE_ENTRY_ID																																...
ENTRY_*N*_RESPONSE_xE_DATA																																...
ENTRY_*N*_RESPONSE_yE_DATA																																...
ENTRY_*N*_RESPONSE_zE_DATA																																...
...																																...

The response data buffer contains the STATUS field followed by the response data for, in this example, N table entries.

53.4.2.2.2.2.8.1 Status

The presence or absence of the STATUS field in the response message depends on the table, i.e., some tables may have the STATUS field and others may not. The size and content of the STATUS field is also table specific. The response status field is intended to return status information that is not stored in hardware, i.e., it is not stored in any of the entry elements. Typically, the status information is available to hardware during the processing of a request message and once the response messages have been generated and sent to software the status information is no longer available to hardware. A good example of status data is the resume entry ID. Resume entry ID is used with the Search access method and it indicates to software at which entry in the table to resume the search.

In addition to being table specific, the status information may also be different for the different operation types. For example, the Add operation could return different status information than the Delete operation for the same table. In order to simplify the protocol, the amount of response data buffer space reserved for the STATUS field must be large enough to accommodate the largest status variant in a given table.

If a given table does not return any status-like information, the STATUS field is not present in the response data buffer for this table. However, all the tables that support the Search access method do return the resume entry ID and therefore have the STATUS field present in their responses.

53.4.2.2.2.8.2 Entry Response(s)

Entry responses are returned only in the case of the Query operation. They are not returned in the case of the Add, Update or Delete operations.

Generically, a response data buffer may contain responses for none, one or many entries. The number of entry responses depends on the number of matching entries found. If the access method used was not Search, there will be a maximum of one entry response returned, depending if the entry exists or not. If the access method used was Search, responses for many entries could be returned. The number of entry responses returned in the response data buffer is stored in the NUM_MATCHED field of the response header.

In the example in Table 366, N entry responses are returned, and each entry response contains an Entry ID as well as the data stored in the xE, yE and zE entry elements.

The response data buffer format shown in Table 366 is generic and as such it does not apply exactly as shown to all the different operation types. The following section explains how the format changes for the different operation types.

53.4.2.2.2.8.3 Operation Specific Response Data Buffer Formats

The response data buffer format shown in Table 366 applies to the Query operation only. The Add, Update and Delete operations use somewhat different data buffer format. This is because the Add, Update and Delete operations do not require any element data to be present in the response. A generic response data buffer format used by the Add, Update and Delete operations is shown in table below

Table 367. Generic NTMP Response Data Buffer Format for Add, Update and Delete Operations

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS																																0x0
...																																...
STATUS																																...

The only field present in the response data buffer for the Add, Update and Delete operations is the STATUS field. Because, in general, the STATUS field may not apply to a specific table, it may not be present, meaning that response data buffer could contain no valid data.

The comparison of the formats for the query operation (Table 366) and for the Add, Update and Delete operations (Table 367) makes it clear that the Query operation format is a superset of the Add/Update/Delete format. This is illustrated graphically in the figure below.

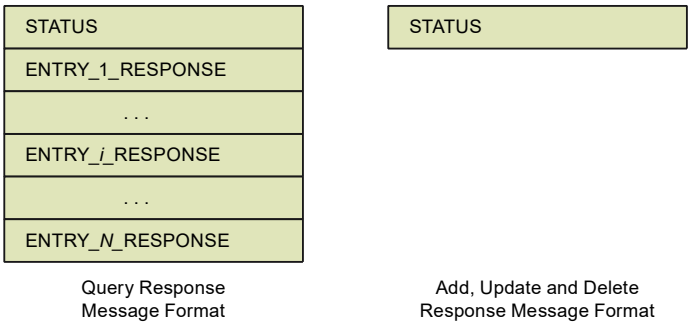


Figure 314. Generic Response Message Formats.

53.4.2.2.2.2.9 Data Buffer Alignment Considerations

In the NTMP protocol the alignment requirements specify the restrictions of the relative positions of the various data elements in the request and response data buffers. The following two alignment types are considered:

- Data component alignment.
- Entry response alignment.

Data Component Alignment

Data component alignment refers to the relative position of a data component in a request or response data buffer in relation to the preceding data components. NTMP data components are structures such as an access key, entry element, status, etc.

Some of the data components are always present in a data buffer while others may or may not be present depending on the command type. Different data components may have different alignment requirements.

The data component alignment for each data component in each NTMP table is well defined and documented. The specified data component alignments must be respected to ensure message format coherency.

Entry Response Alignment

An entry response is a collection of data components associated with a single table entry. NTMP commands utilizing the search access method may produce response data buffers containing multiple entry responses. Entry response alignment refers to the relative position of an entry response in the response data buffer in relation to the preceding data component or entry response.

The entry response alignment for each NTMP table is well defined and documented. The specified entry response alignments must be respected to ensure message format coherency.

Example

Consider the example shown in the figure below. The example shows a query response data buffer with two data components followed by *N* entry responses. "Data Component 2 Padding" is needed because of the required data component 2 alignment. All the entry response paddings are needed because of the required entry response alignment. Note that "Entry Response 1 Padding", may be of a different size than all the following entry response paddings. This is because "Data Component 2" may require a different padding than an entry response. Entry response paddings 2, ..., *N*-1 have the same size. The figure also shows *M* data components present in the entry response. Except for the first data component all others require a data component padding.

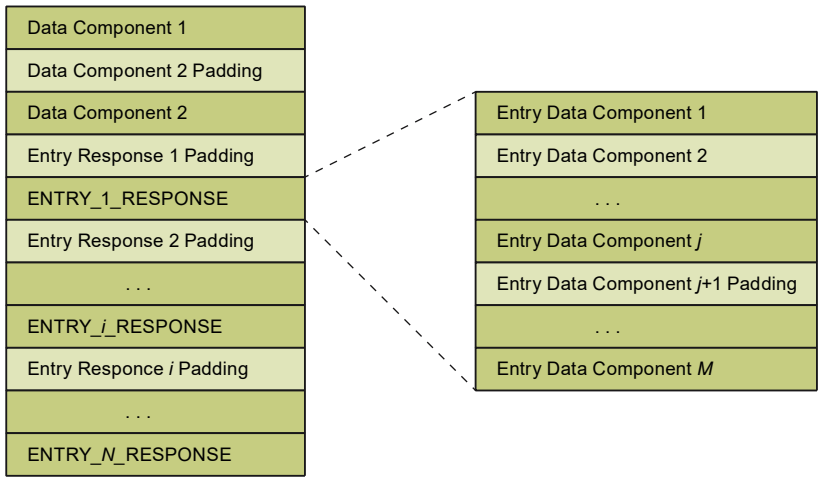


Figure 315. Data Buffer Alignment Considerations - Example.

53.4.2.2.3 Transmit and Receive Buffer Descriptor Format

53.4.2.2.3.1 Transmit Buffer Descriptor Format

This section describes the format of the buffer descriptors used by software to pass packets to hardware for transmission. The basic or standard descriptor is 16B and provides the ability to send a packet, possibly using multiple buffers, as well as optionally supporting one or two offloads. In addition, this descriptor can be extended by adding another 16B which supports more offloads. Since all transmit rings use 16B buffer descriptors, the extended version will always require a minimum of two chained BDs. The shaded fields in the descriptors indicate that the field is only applicable in the first BD. Reserved fields should be written with 0's.

The feature(s) supported by the (basic) descriptor format are:

- Time specific departure
- Switch management

Table 368. Transmit Buffer Descriptor Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ADDR[31:0]																																0x00
ADDR[63:0]																																0x04
FRM_LEN																BUF_LEN																0x08
F	E	FI	FL		W	Depends on Flags (FL) setting																										0x0C
			0	0	0	FLQ		Depends on Flags Qualifier (FLQ) setting																								0x0C
						0	0	--																								0x0C
						0	1	--																								0x0C
						1	0	SMSO	--			INGR_PORT				--																0x0C
									TSR	--		EGR_PORT				--	IPV		DR	--												0x0C
						1	1	--																								0x0C
			0	1	0	--																										0x0C
			1	0	0	TSE	TX_START																									0x0C
			1	1	0	--																										0x0C

Table 369. Transmit Buffer Descriptor Field Descriptions

Bits	Name	Description
127	F	Final Determines if this is the last buffer descriptor for the frame. This field is always valid. 0 Not final. Next buffer descriptor is part of a chain to create a list of buffers, or is an extension buffer descriptor.

Table continues on the next page...

Table 369. Transmit Buffer Descriptor Field Descriptions (continued)

Bits	Name	Description
		1 Final. This is the last buffer descriptor. Last buffer descriptor always needs to have the F bit set to 1 regardless of whether it is part of a chain or not, unless the next buffer descriptor is an extension buffer descriptor.
126	E	Extended Indicated that this descriptor is extended by the next descriptor in a chain. Valid only in the first descriptor in a chain. 0 Standard. 1 Extended. The next chained BD has the extended format.
125	FI	Frame interrupt. Valid only in the first descriptor in a chain. 0 Disabled. 1 Enabled. Transmit frame event for the ring, TB*a*IDR[TXF], will be set after completion of the frame and an interrupt will be generated if TB*a*IER[TXFIE]=1.
124-123	FL	Flags Indicate the presence of additional fields in bit offset 96-122. Valid only in the first descriptor in a chain 00b Definition depends on the Flags Qualifier (FLQ) setting. 01b Reserved 10b Indicates that the time specific departure fields are present 11b Reserved
121-120	FLQ	Flags Qualifier Indicates the presence of additional fields in bit offset 96-119. Valid only in the first descriptor in a chain. 00b No optional fields present, remaining bits at bit offset 96-119 are reserved (must be set to 0) 01b Reserved 10b Indicates that switch management sending options fields are present 11b Reserved
119	SMSO	Switch Management Sending Options This field specifies the sending options (optional) host originated switch management traffic. Applicable only if the *ENETC* port is connected internally to a switch port, designated as a switch management port. SMCAPR[SM] set to 1b indicates that ENETC is connected to a switch port designated as the switch management port. **0**: Switch Port Masquerading Host originated switch management traffic forwarded by the switch as it was received from another switch port. The switch port number is specified in the INGR_PORT field of the transmit BD.

Table continues on the next page...

Table 369. Transmit Buffer Descriptor Field Descriptions (continued)

Bits	Name	Description
		This option uses the regular ENETC transmit datapath processing and the switch ingress datapath processing, which in turn relies on the pseudo link to transfer frames between ENETC and the switch. **1**: Direct Switch Enqueue This option is used to direct host originated switch management traffic out a specific switch egress queue. It by-passes most of the ENETC transmit datapath processing and the switch ingress datapath processing. A dedicated transmit BD ring must be used for this option, and all frames put on that transmit BD ring will be directed to the specified switch egress queue, subject to the switch egress congestion management policy (possibility of having frames dropped if the egress queue is congested). Only SI 0 can make use of this option, with transmit BD ring number 0 permanently assigned to support this option. This transmit BD ring will not be assigned to a traffic class, meaning that this transmit BD ring will not be considered by the HTA transmit scheduling mechanism. For each frame put on this transmit BD ring, its transmit buffer descriptor (BD) must have the FL field set to 00b, the FLQ field set to 10b and the SMSO field set to 1b. If these fields are set otherwise, the frame is not transmitted and the STATUS field of the transmit BD is updated with the "Programming Error in Buffer Descriptor Used for Direct Switch Enqueue" code point. The output switch port and IPV need to be specified in the transmit BD (EGR_PORT and IPV fields respectively) in order for the switch to determine the switch egress queue. The DR needs also to be specified (in the DR field of transmit BD) for congestion management handling within the switch. The switch buffer pool ID will be determined by the switch using the IPV provided in transmit BD and the input switch port number to which the ENETC is internally connected. When sending the frame out from a specific NETC switch external port, there is the option to request timestamping of the frame using the one-step timestamping method or the two-step timestamping method (that is, returning the transmit timestamp to the host), or both. Request to perform one-step timestamping, is specified by setting the IEEE 1588 PTP one-step timestamp offload flag to 1 in the E_FLAGS field, and by writing to the TIMESTAMP field, the timestamp value to be used by the one-step timestamp function to calculate the residence time to be added to the Correction field of the frame being transmitted. Request to perform two-step timestamping, is specified by setting the Timestamp Reference Request (TSR) field to 1. If the frame is enqueued successfully onto the switch egress queue (no errors reported in the STATUS field), hardware writes back a timestamp identifier in the TXTSID field of the transmit buffer descriptor. This identifier is used by software to correlate the transmit timestamp response with the transmit frame.
118	TSR	Timestamp Reference Request Request to capture (and return to the host) the timestamp when the frame's SFD is transmitted by the switch's Ethernet MAC. After the switch dequeues and transmits the frame, it sends a transmit timestamp reference response message back to the host (as a Host Reason of 0x3) to the receive BD ring specified in the switch management host reason a receive BD ring mapping register (SMHR3BDRMR) with the receive buffer descriptor containing the TXTSID from the writeback, and the timestamp value returned from the switch Ethernet MAC.

Table continues on the next page...

Table 369. Transmit Buffer Descriptor Field Descriptions (continued)

Bits	Name	Description
		Valid if SMSO=1b
116-112	INGR_PORT	Ingress switch port number this frame is to be injected towards. Applicable only if the *ENETC* port is connected internally to a switch port, designated as a switch management port Valid values: 0..SCAPR0[NUM_PORT]-1 Valid if SMSO=0b
116-112	EGR_PORT	Egress switch port number this frame is to be transmitted. Applicable only if the *ENETC* port is connected internally to a switch port, designated as a switch management port Valid values: 0..SCAPR0[NUM_PORT]-1 Valid if SMSO=1b
110-108	IPV	Internal Priority Value Applicable only if the ENETC port is connected internally to a switch port, designated as a switch management port, and the frame is being transmitted with the Switch Management Sending Options (SMSO) set to 1 (directing a frame out a specific switch egress queue). The IPV specified in this field is be used by the switch to determine the switch egress queue and buffer pool. Valid if SMSO=1b
107-106	DR	Discard Resilience Applicable only if the ENETC port is connected internally to a switch port, designated as a switch management port, and the frame is being transmitted with the Switch Management Sending Options (SMSO) set to 1 (directing a frame out a specific switch egress queue). This field is used for congestion management handling within the switch. Valid if SMSO=1b
122	W	Written Used to determine if writeback occurred after completion of this frame. Must be initialized to 0b by software, hardware will change this to 1b if it writes back to the descriptor. Valid only in the first descriptor in a chain.
121	TSE	Transmit start enable Determines if the TX_START field is valid, and to be used for time specific departure. Valid only in the first descriptor in a chain. Present only if FL is 10b.
120-96	TX_START	Transmit start time Required start time for transmission of this frame that is time specific departure. This value is expressed as bit [29:5] of the IEEE 1588 synchronized nanosecond time. Since bit 31 and 30 are always 0, the time window is approximately 1 second and then wraps around. As a result, a departure time of up to approximately 0.5 s in the future from the current time, may be specified

Table continues on the next page...

Table 369. Transmit Buffer Descriptor Field Descriptions (continued)

Bits	Name	Description
		so that when hardware schedules the frame, it can determine whether the departure time is in the past or in the future. Note that the unit is 32 ns, since bit 0 to 4 are not included. Valid only in the first descriptor in a chain. Present only if FL is 10b.
95-80	FRM_LEN	Frame Length This field specifies the total effective frame length including data in the buffer referenced by this descriptor and those referenced by any chained descriptors. FRM_LEN must be same or less than all BUF_LEN combined. FRM_LEN is valid and required for both single and multi transmit buffer descriptors frames. Frames must not have a FRM_LEN that is less than 16 bytes. Frames of 0-15 bytes are not supported. The maximum recommended setting of FRM_LEN is 2000 bytes. If FRM_LEN exceeds 9600 bytes, the frame is not transmitted and the STATUS field of the transmit buffer descriptor is updated with the error code 0x00A (invalid frame/buffer/chain length).
79-64	BUF_LEN	Buffer Length The buffer length field specifies the effective number of bytes in the data buffer pointed to by ADDR field. This field must be a non-zero value. There must be no extra buffer descriptors (of any BUF_LEN value) beyond the one that contains the last byte of frame data. First descriptor in a chain must not have a BUFF_LEN that is less than 16 bytes.
63-0	ADDR	Address The memory address of a buffer containing the full or partial Ethernet packet. This address may be a virtual or a physical address as defined by software. When chaining is defined (F=0), multiple addresses are used to create the full packet.

The figure below shows the layout of the extension descriptor which can follow a standard descriptor and allows transmission of packets using more offloads than the standard descriptor alone. This descriptor serves as an extension of the standard transmit descriptor and is present if the E (“Extended”) bit of the standard descriptor is set (1b). Note that descriptor extensions can only follow the initial standard descriptor in a chain for a multi-buffer frame. The added offload features provided by the extended buffer descriptor format include:

- VLAN insert offload
- Timestamp offload

Table 370. Extension Transmit Buffer Descriptor Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TIMESTAMP																																0x00
PCP			DEI	VID												TPID		--														0x04
--																																0x08
F	--		FMT		--		E_FLAGS									--																0x0C

Table 371. Extension Transmit Buffer Descriptor Field Descriptions

Bits	Name	Description
127	F	Final Determines if this is the last buffer descriptor for the frame. This field is always valid. 0 Not final. Next buffer descriptor is part of a chain to create a list of buffers. 1 Final. This is the last buffer descriptor, Last buffer descriptor always needs to have the F bit set to 1 regardless of whether it is part of a chain or not.
126-125	--	Reserved
124-122	FMT	Format 000b The format described here All other values reserved
121-120	--	Reserved
119-112	E_FLAGS	Extension flags Indicates what offloads are enabled for this frame, see table below.

Bits	Description
119-115	Reserved
114	IEEE 1588 PTP two-step timestamp offload When set, timestamp information is updated in TIMESTAMP field after successful transmit.
113	IEEE 1588 PTP one-step timestamp offload The hardware subtracts the timestamp value provided in the TIMESTAMP field from the timestamp value captured when the frame transmission started. The hardware then adds the results of the subtraction (also referred to as the residence time) “on-the-fly” to the Correction field of the frame being transmitted. The Correction field location in the frame is specified in the register PMa_SINGLE_STEP[OFFSET]. **NOTE** The one-step timestamp offload is not supported on the pseudo MAC.
112	VLAN insert offload When set, indicates that the packet is a VLAN packet and hardware must add the VLAN tag header to the packet on transmit. The VLAN tag to be inserted by hardware is specified in the extension transmit buffer descriptor fields TPID, PCP, DEI and VID.

63-61	PCP	Priority code point

Table continues on the next page...

Table 371. Extension Transmit Buffer Descriptor Field Descriptions (continued)

Bits	Name	Description
		This field contains the PCP value of the VLAN tag to be inserted in the packet during transmission.
60	DEI	Drop eligible indicator This field contains the DEI value of the VLAN tag to be inserted in the packet during transmission.
59-48	VID	VLAN identifier This field contains the VLAN identifier of the VLAN tag to be inserted in the packet during transmission.
47-46	TPID	Tag protocol identifier This field contains the TPID of the VLAN tag to be inserted in the packet during transmission. The TPID is specified as a 2-bit code value. 2 standard and 2 custom tag identifiers are supported. 00b Standard C-VLAN 0x8100 01b Standard S-VLAN 0x88A8 10b Custom C-VLAN as defined by CVLANR1[ETYPE] 11b Custom S-VLAN as defined by CVLANR2[ETYPE]
45-32	--	Reserved
31-0	TIMESTAMP	Timestamp The timestamp value provided in this field is subtracted from the SFD transmit time of the frame. The result of the subtraction (also referred as to the residence time) is then added to the value read in the Correction field of the frame. Finally, the result of the addition is writing back to the Correction field of the frame. The timestamp value is expressed in units of nanoseconds, with the upper 2 bits of the timestamp value (bits 31-30) not being used. The value of this field is in the range 0 to 0x3FFFFFFF. This field is valid when IEEE 1588 PTP one-step timestamp offload flag is set.

Following transmission hardware may write data back into the transmit descriptor, depending on the requested offloads and whether any errors occurred in transmit processing. To allow a blind write instead of a read-modify-write the fields of the descriptor are overwritten. The figure below shows the descriptor format.

Writeback always occurs if a direct switch enqueue is requested. Otherwise writeback only occurs if required either due to a requested offload or an error. The W (Written) bit at bit position 122 in the transmit descriptor must be initialized to a 0 by software when the descriptor is passed to hardware and is written with a 1 during writeback to allow detection of the writeback. Writeback only occurs in the first descriptor of a chain of descriptors for a frame.

Table 372. Transmit Buffer Writeback Descriptor Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TIMESTAMP																																0x00

Table continues on the next page...

Table 372. Transmit Buffer Writeback Descriptor Format (continued)

--								TXTSID	0x04
--									0x08
F = 0	E = 0	0	0	0	W = 1	--	STATUS	--	0x0C

Table 373. Transmit Buffer Writeback Descriptor Field Descriptions

Bits	Name	Description	
120-112	STATUS	Status	
		Code Points	Description
		0x001:	Programming error An error exists in either the Tx BD, the Tx ring registers, or some combination of these which meant that this frame could not be transmitted.
		0x002:	Time specific departure drop The time defined in TX_START expired before frame could be processed for transmission.
		0x008:	Frame size error One of the following conditions occurred: • Frame length exceeds PTCaTMSDUR[MAXSDU] (where a is the traffic class number) for a non-segmented frame. • Frame has exceeded 63 chained BDs in length. Frames provided in chained BDs must not exceed 63 BDs. • FRM_LEN is larger than (combined) buffer lengths in a multi-BD frame.
		0x009:	Null address A null address 0xFFFF_FFFF_FFFF_FFFF detected in the transmit buffer descriptor, or entire transmit descriptor is read in as 0s, or an all 0s BD is encountered in the middle of a mulit-BD frame.
		0x00A:	Invalid frame/buffer/chain length • A frame length of less than 16 bytes is detected in the transmit buffer descriptor. • A buffer length of less than 16 bytes is detected in the first descriptor of a chain. • A single-BD frame's FRM_LEN value exceeds the BUF_LEN. Note that this condition occurring in a multi-BD frame is reported as a frame size error. • A FRM_LEN exceeding 9600 bytes has been provided. • The W bit in a provided BD is not 0. • A single-BD frame (FRM_LEN <= BUF_LEN) does not have F=1 in the BD or its extension.

Table continues on the next page...

Table 373. Transmit Buffer Writeback Descriptor Field Descriptions (continued)

Bits	Name	Description	
		0x010:	Source MAC address spoofing detected An attempt was made to transmit a frame with a source MAC address not matching that of the SI's primary MAC address when PSI*a*CFGR0[ASE]=1.
		0x020:	Frame dropped due to port reset. Timestamp value is not valid if this bit is set to 1.
		0x021:	Frame dropped due to port disabled. Frame not transmitted (dropped) due to port disabled (POR[TXDIS]= 0b1).
		0x040:	VLAN TPID not allowed Frame not transmitted (dropped) because the TPID in it's VLAN tag (either in the frame itself or through the transmit BD) is not allowed.
		0x060:	Programming error in buffer descriptor used for direct switch enqueue Buffer descriptor in the transmit BD ring for direct switch enqueue, doesn't have the FLQ field set to 10b and the SMSO field set to 1, or a buffer descriptor has the FLQ field set to 10b and the SMSO field set to 1, in a non "direct switch enqueue" transmit BD ring. This error code is also returned if a direct switch enqueue is attempted within a transmit BD ring assigned to an ENETC which is not designated as a switch management ENETC.
		0x080:	Frame too large for time gating window The frame was determined to be too large to transmit during the open phase of the traffic class transmit gate. Adjust the window size or the frame size.
		0x090:	AXI read error Frame not transmitted (dropped) due to an AXI read error detected.
		0x091:	AXI write error Frame not transmitted (dropped) due to an AXI write error detected.
		0x0A0:	Multi-bit ECC error Frame not transmitted (dropped) due to a multi-bit ECC error detected.
		0x0F0:	Parity error Frame not transmitted (dropped) due to a parity write error detected.
		0x100:	Frame dropped due to switch congestion Frame drops due to congestion on the switch buffer pool or class queue, in the case where the option to direct a frame out a specific switch egress queue is used.
111-96	--	Reserved	

Table continues on the next page...

Table 373. Transmit Buffer Writeback Descriptor Field Descriptions (continued)

Bits	Name	Description
31-0	TIMESTAMP	Timestamp Timestamp value represents the transmission time of the frame. The timestamp reported in this field can be either a synchronized timestamp or a free running timestamp. For the Ethernet MAC, the PMa_COMMAND_CONFIG[TS_MODE] register determines which of the two clock sources (synchronized clock or free running clock) is used for sampling the timestamp. For the pseudo MAC, the PCR[TIMER_CS] register determines which of the two clock sources (synchronized or free running) is used for sampling the timestamp. If a synchronized timestamp is reported, the timestamp in this field is the lower 32 bits of the 64-bit synchronized timestamp. The value in this field is in units of nanoseconds with the range 0 to 0xFFFFFFFF. To reconstruct the 64-bit synchronized timestamp, perform the following steps: 1. Get the current 64-bit synchronized time. It can be obtained from the following registers: • **ENETC SI register address space** - Read to the SICTR0 register followed immediately by a read to the SICTR1 register. • **Timer register space** - Read to the TMR_FRT_L register, followed immediately by a read to the TMR_SRT_L register and to the TMR_SRT_H register. TMR_SRT_L/H indicates the current synchronized time. 2. Check if the lower 32-bits of the current synchronized time has wrapped around since the timestamp was captured, by comparing it to the timestamp value reported in this field. If the reported timestamp value in this field is higher than the lower 32-bits of the current synchronized time, the time has wrapped around resulting the need to adjust the upper 32-bits of the current synchronized time by subtracting 0x00000001 from it. This value is then used to set the upper 32 bits of the synchronized timestamp. If the time has not wrapped around, the upper 32 bits of the current synchronized time read from the register, is used to set the upper 32 bits of the synchronized timestamp. 3. The upper 32 bits value determined in step 2, and the lower 32 bits reported in this field are combined to create the 64-bit synchronized timestamp. If a free-running timestamp is reported, the timestamp in this field is the lower 32 bits of the 64-bit free running timestamp. The value in this field is in units of NETC clock ticks with the range 0 to 0xFFFFFFFF. To reconstruct the 64-bit free running timestamp, perform the following steps: 1. Get the current free-running time by issuing a read to the TMR_FRT_L register followed immediately by a read to the TMR_FRT_H register. 2. Check if the lower 32-bits of the current free-running time has wrapped around since the timestamp was captured, by comparing it to the timestamp value reported in this field. If the timestamp value reported in this field is higher than the lower 32-bits of the current free-running time, the time has wrapped around resulting the need to adjust the upper 32-bits of the current free-running time by subtracting 0x00000001 from it. This value is then used to set the upper 32 bits of the free-running timestamp. If the time has not wrapped around, the upper 32 bits of the current running time read from the register, is used to set the upper 32 bits of the free-running timestamp. 3. The upper 32 bits value determined in step 2, and the lower 32 bits reported in this field are combined to create the 64-bit free running timestamp. This field is updated after successful transmission when IEEE 1588 PTP two-step timestamp offload is enabled for the frame transmitted.

Table continues on the next page...

Table 373. Transmit Buffer Writeback Descriptor Field Descriptions (continued)

Bits	Name	Description
15-0	TXTSID	Transmit Timestamp Identifier This field contains an incrementing value that is updated with every writeback. It is only meaningful when the transmit buffer descriptor has specified that a timestamp be captured (and be returned to the host) when the frame's SFD is transmitted by the switch's Ethernet MAC. This timestamp request is specified by setting the Timestamp Request Reference (TSR) field to 1, in the transmit buffer descriptor. When TSR is set 1, the TXTSID value from the writeback must be used to associate the received TSR response with the transmit frame.

The figure below illustrates the four possible ways to represent a frame in memory using the 16B standard and extended BD formats.

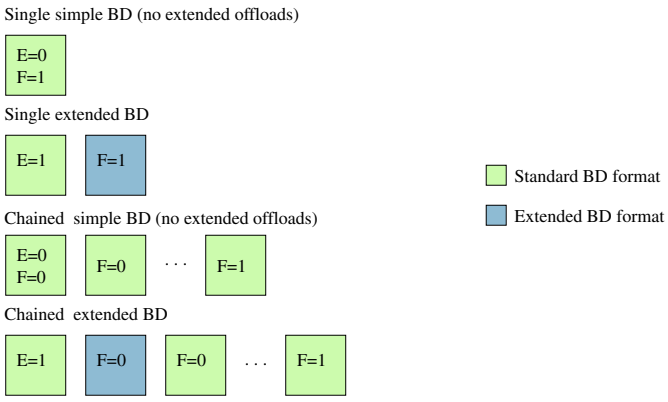


Figure 316. BD Frame Representation

Transmit Frame Restrictions and Limitations

This section describes unsupported transmit frame configurations and limitations on supported configurations. Software must ensure that frames which violate the following conditions are not given to the ENETC for processing. Erroneous hardware behavior will result; i.e. frames with error, dropped frames.

1. Frames must not have a FRM_LEN that is less than 16 bytes. Frames of 0-15 bytes are not supported.
2. Chained transmit frames must not contain any BDs beyond the number required to hold the transmitted frame; i.e. the F=1 BD must contain the final frame byte. One or more "empty" BDs of any BUF_LEN >=0 must not be present following the BD(s) that contain frame data.
3. Chained transmit frames must not contain any BDs that have BUF_LEN = 0. This restriction is already stated in the definition of the BUF_LEN field.
4. Chained transmit frames must not have a FRM_LEN which is greater than the sum of its BDs BUF_LEN fields.

53.4.2.2.3.2 Receive Buffer Descriptor Format

The BD formats for receive rings which operate with standard 16B descriptors are shown below. Software includes the buffer address to NETC in the provided BD. NETC overwrites the address with information extracted from the received frame. The shaded fields in the descriptors indicate that the field is only applicable for unchained BDs or the first descriptor in a chain.

Table 374. Software Provided Receive Buffer Descriptor Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ADDR[31:0]																																0x00
ADDR[63:0]																																0x04
--																																0x08
--																																0x0C

Table 375. Standard Receive Writeback Buffer Descriptor Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
PARSER_SUMMARY																INET_CSUM																0x00
RSS_HASH																																0x04
RSS_HASH_P																						--								SRC_PORT		0x04
PCP			DEI		VID											BUF_LEN																0x08
F	R	--							ERROR							FLAGS						--			HR					TPID		0x0C

The BD formats for receive rings which operate with extended 32B descriptors are shown below. Additional fields support:

- Timestamping for PTP (IEEE 1588) and gPTP (802.1AS-2020).

Software must configure the BD rings and the steering of frames accordingly. Software includes the buffer address to NETC in the provided BD. NETC overwrites the address field from the provided BD with information extracted from the received frame. The shaded fields in the descriptors indicate that the field is only applicable for unchained BDs or the first descriptor in a chain.

Table 376. Software Provided Extended Receive Buffer Descriptor Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ADDR[31:0]																																0x00
ADDR[63:0]																																0x04
--																																0x08
--																																0x0C
--																																0x10
--																																0x14
--																																0x18
--																																0x1C

Table 377. Extended Receive Writeback Buffer Descriptor Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
PARSER_SUMMARY																INET_CSUM																0x00
RSS_HASH																																0x04
RSS_HASH_P																						--						SRC_PORT				0x04
PCP			DEI	VID												BUF_LEN																0x08
F	R	--							ERROR							FLAGS						--					HR			TPID		0x0C
TIMESTAMP																																0x10
--																																0x14
--																																0x18
--																																0x1C

Rx Buffer Descriptors containing transmit timestamp reference response messages are formatted as follows. These buffer descriptors are received in response to a successfully processed Tx buffer descriptor which had a set TSR bit (Timestamp Reference Request). Received transmit timestamp reference response BDs are identified by their Host Reason code (0x3 Timestamp Response), which is unique to these messages and is used to determine which Rx ring within the designated switch management ENETC they are received in. Mapping of Host Reason to receive BD ring is via the receive BD ring mapping register SMHRaBDRMR.

No DMA operation is performed when NETC produces transmit timestamp reference response messages. So, if a BD ring is used exclusively for handling these messages it is not necessary for software to provide a valid ADDR in the ring entries.

The standard ERROR code field is present but only provides an indication of internal logic parity or SRAM ECC mismatch for transmit timestamp reference response message BD entries.

Table 378. Transmit Timestamp Reference Response Buffer Descriptor Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TIMESTAMP																																0x00
--																																0x04
--																TXTSID																0x08
F	R	--							ERROR							--									HR					--			0x0C

Table 379. Software Provided Receive Buffer Descriptor Field Descriptions

Bits	Name	Description
255-128	--	Reserved (32B format only)
127-64	--	Reserved

Table continues on the next page...

Table 379. Software Provided Receive Buffer Descriptor Field Descriptions (continued)

Bits	Name	Description
63-0	ADDR	Address Memory address of a buffer containing the full or partial Ethernet packet. NETC will ignore bits 5-0 of the address field. May be a virtual or a physical address as defined by software. When chaining is defined (F=0), multiple addresses are used to create the full packet. Minimum buffer size is 128 bytes. For better performance, it is recommended not to use a buffer size smaller than 256 bytes. The memory address of the buffers must be 64B aligned. If the SBCR[WBS] register is configured for 128B maximum burst size (WBS=3) then the memory address of the receive buffers must be 128B aligned. WBS=2 is the recommended setting (default; 64B maximum burst size).

Table 380. Receive Writeback Buffer Descriptor Field Descriptions

Bits	Name	Description
255-168	--	Reserved (extended format only)
159-128	TIMESTAMP	Timestamp (extended format only) Timestamp value that was applied by MAC at time of ingress . For the Ethernet MAC, the timestamp is captured when the MAC detects the one-octet start frame delimiter (SFD) of the frame. For the pseudo MAC, the timestamp is captured at the time the frame was transferred across the pseudo link, as there is no frame serialization or copy across a pseudo link. Valid when timestamp flag is set. The timestamp reported in this field can be either the synchronized timestamp or the free running timestamp. The PCR[TIMER_CS] register determines which of the two timestamps (synchronized or free running) is reported. If the synchronized timestamp is reported, the timestamp in this field is the lower 32 bits of the 64-bit synchronized timestamp. The value in this field is in units of nanoseconds with the range 0 to 0xFFFFFFFF. To reconstruct the 64-bit synchronized timestamp, perform the following steps: 1. Get the current 64-bit synchronized time. It can be obtained from the following registers: • **ENETC SI register address space** - Read to the SICTR0 register followed immediately by a read to the SICTR1 register. • **Timer register space** - Read to the TMR_FRT_L register, followed immediately by a read to the TMR_SRT_L register and to the TMR_SRT_H register. TMR_SRT_L/H indicates the current synchronized time. 2. Check if the lower 32-bits of the current synchronized time has wrapped around since the timestamp was captured, by comparing it to the timestamp value reported in this field. If the reported timestamp value in this field is higher than the lower 32-bits of the current synchronized time, the time has wrapped around resulting the need to adjust the upper 32-bits of the current synchronized time by subtracting 0x00000001 from it. This value is then used to set the upper 32 bits of the synchronized timestamp. If the time has

Table continues on the next page...

Table 380. Receive Writeback Buffer Descriptor Field Descriptions (continued)

Bits	Name	Description
		not wrapped around, the upper 32 bits of the current synchronized time read from the register, is used to set the upper 32 bits of the synchronized timestamp. 3. The upper 32 bits value determined in step 2, and the lower 32 bits reported in this field are combined to create the 64-bit synchronized timestamp. If the free-running timestamp is reported, the timestamp in this field is the lower 32 bits of the 64-bit free running timestamp. The value in this field is in units of NETC clock ticks with the range 0 to 0xFFFFFFFF. To reconstruct the 64-bit free running timestamp, perform the following steps: 1. Get the current free-running time by issuing a read to the TMR_FRT_L register followed immediately by a read to the TMR_FRT_H register. 2. Check if the lower 32-bits of the current free-running time has wrapped around since the timestamp was captured, by comparing it to the timestamp value reported in this field. If the timestamp value reported in this field is higher than the lower 32-bits of the current free-running time, the time has wrapped around resulting the need to adjust the upper 32-bits of the current free-running time by subtracting 0x00000001 from it. This value is then used to set the upper 32 bits of the free-running timestamp. If the time has not wrapped around, the upper 32 bits of the current running time read from the register, is used to set the upper 32 bits of the free-running timestamp. 3. The upper 32 bits value determined in step 2, and the lower 32 bits reported in this field are combined to create the 64-bit free running timestamp. When the extended 32B descriptors are used, every received frame, has its captured timestamp reported in this field. If the ENETC port is connected internally to a switch port, designated as a switch management port, and the Host Reason is set to a non-zero value, this field contains the Rx timestamp captured at the switch port where the frame was received.
127	F	Final Determines if multiple descriptors are chained to form a list of buffer addresses. 0 Not final. Next buffer descriptor is part of a chain to create a list of buffers. 1 Final. This is the last buffer descriptor in a chain.
126	R	Ready Indicates that the BD has been written. If the ready bit is set for the first BD, it also indicates the entire chain of BDs is complete. NETC will not set the ready bit for the first BD in a chain unless all other BDs associated with it are also complete. Note that although software can use the R bit as an efficient means to qualify the BD contents and also read frame data from the associated buffers (i.e. avoid the need to poll ring PI), it is still necessary to check the ring's RBaPIR[BDR_INDEX] prior to incrementing RBaCIR[BDR_INDEX]. Set R bits will appear in the ring ahead of the corresponding RBaPIR update. Software must not advance the ring consumer index before hardware has updated the ring producer index.
125-120	--	Reserved

Table continues on the next page...

Table 380. Receive Writeback Buffer Descriptor Field Descriptions (continued)

Bits	Name	Description
119-112	ERROR	Error status code for the received frame. The listed codes indicate the presence of a detected error condition or event. **Bits** / **Description** 119-112 / 0x00: No error detected. 0x40: L3 (IPv4) Header Checksum validation failed. 0x41: L4 (TCP or UDP) Checksum validation failed. 0x80: Internal logic parity or SRAM (ECC) mismatch detected. 0x88: System Bus Read Error detected during processing of the frame. 0x89: System Bus Write Error detected during processing of the frame. All unlisted code points are reserved and will not appear.
111-104	FLAGS	Flags for receive information. **Bits** / **Description** 111-109 / Reserved 108 / L4 (TCP or UDP) Checksum was validated and found to be correct. 107 / L3 (IPv4) Header Checksum was validated and found to be correct. 106 / Timestamp received When set, timestamp information is available in TIMESTAMP field. 105 / VLAN header extracted When set, VLAN header information is available in TPID, PCP,DEI and VID fields. 104 / RSSV - RSS hash valid 0 = RSS_HASH/RSS_HASH_P field is not valid 1 = RSS_HASH/RSS_HASH_P field is valid The value in the RSS_HASH/RSS_HASH_P field is not valid if RSS is not supported, RSS is disabled, or no RSS rules were matched.
103-102	--	Reserved
101-98	HR	Host Reason 0: Regular forwarded frame 1: Ingress mirroring 2: MAC learning

Table continues on the next page...

Table 380. Receive Writeback Buffer Descriptor Field Descriptions (continued)

Bits	Name	Description
		3: Timestamp Response 4-7: Reserved 8-15: Software defined Host Reason
97-96	TPID	Extracted VLAN tag protocol identifier. A 2-bit TPID value identifying which known value matched. Valid if VLAN Extracted flag (bit 105) is set to 1. 00 0x8100 01 0x88A8 10 Custom tag 1 (as defined in CVLANR1[ETYPE]) 11 Custom tag 2 (as defined in CVLANR2[ETYPE])
95-93	PCP	Priority code point A 3-bit field which refers to the IEEE 802.1p class of service and maps to the frame priority level. Valid if VLAN Extracted flag (bit 105) is set to 1.
92	DEI	Drop eligible indicator May be used separately or in conjunction with PCP to indicate frames eligible to be dropped in the presence of congestion. Valid if VLAN Extracted flag (bit 105) is set to 1.
91-80	VID	VLAN identifier The VLAN identifier is a 12-bit field specifying the VLAN to which the frame belongs. Valid if VLAN Extracted flag (bit 105) is set to 1.
79-64	BUF_LEN	Buffer length The length field specifies the effective number of bytes in the data buffer pointed to by ADDR field. It does not include any size adjustments done by NETC for alignment purposes to the head or tail end BD.
63-32	RSS_HASH	Receive side scaling 32-bit hash value. Valid only if the RSSV bit is set to 1.
63-40	RSS_HASH_P	Receive side scaling 24-bit hash value. Valid only if the RSSV bit is set to 1, and the ENETC port is connected internally to a switch port, designated as a switch management port.
36-32	SRC_PORT	Switch Port ID of received frame. Valid only if the ENETC port is connected internally to a switch port, designated as a switch management port.
31-16	PARSER_SUMMARY	Parser summary, see Parse Summary.
15-0	INET_CSUM	Internet checksum. Calculated over the L2 payload, i.e. does not include the L2 header (first 14 bytes consisting of DMAC, SMAC and EtherType) nor FCS. The Internet checksum is used in various Internet protocols to check for data corruption in headers.

Table 381. Transmit Timestamp Reference Response Writeback Buffer Descriptor Field Descriptions

Bits	Name	Description
127	F	Final

Table continues on the next page...

Table 381. Transmit Timestamp Reference Response Writeback Buffer Descriptor Field Descriptions (continued)

Bits	Name	Description
		Determines if multiple descriptors are chained to form a list of buffer addresses. 0 Reserved. 1 Final. BDs containing a transmit timestamp reference response are always unchained.
126	R	Ready Indicates that the BD has been written. Note that although software can use the R bit as an efficient means to qualify the BD contents (that is, avoid the need to poll ring PI), it is still necessary to check the ring's RBaPIR[BDR_INDEX] prior to incrementing RBaCIR[BDR_INDEX]. Set R bits will appear in the ring ahead of the corresponding RBaPIR update. Software must not advance the ring consumer index before hardware has updated the ring producer index.
125-120	--	Reserved
119-112	ERROR	Error status code The listed codes indicate the presence of a detected error condition or event. 0x00: No error detected. 0x80: Internal logic parity or SRAM (ECC) mismatch detected. All unlisted code points are reserved and will not appear.
111-102	--	Reserved
101-98	HR	Host Reason 0-2: Reserved 3: Timestamp Response 4-15: Reserved
97-96	--	Reserved
95-80	--	Reserved
79-64	TXTSID	Transmit Timestamp Identifier Transmit timestamp identifier (or correlation identifier) that was written back by hardware in the TXTSID field of the transmit buffer descriptor, in response to a successfully processed transmit buffer descriptor which had a set TSR bit (Timestamp Reference Request). The transmit timestamp identifier is used by software to correlate the transmit timestamp reference response with the request (or transmit frame).
63-32	--	Reserved
31-0	TIMESTAMP	Timestamp Timestamp value represents the transmission time of the frame on a switch port.

Table continues on the next page...

Table 381. Transmit Timestamp Reference Response Writeback Buffer Descriptor Field Descriptions (continued)

Bits	Name	Description
		The transmit timestamp can be either a synchronized timestamp or a free running timestamp. The PMa_COMMAND_CONFIG[TS_MODE] register determines which of the two clock sources (synchronized clock or free running clock) is used for sampling the timestamp. If the synchronized timestamp is reported, the timestamp in this field is the lower 32 bits of the 64-bit synchronized timestamp. The value in this field is in units of nanoseconds with the range 0 to 0xFFFFFFFF. To reconstruct the 64-bit synchronized timestamp, perform the following steps: 1. Get the current 64-bit synchronized time. It can be obtained from the following registers: • **ENETC SI register address space** - Read to the SICTR0 register followed immediately by a read to the SICTR1 register. • **Timer register space** - Read to the TMR_FRT_L register, followed immediately by a read to the TMR_SRT_L register and to the TMR_SRT_H register. TMR_SRT_L/H indicates the current synchronized time. 2. Check if the lower 32-bits of the current synchronized time has wrapped around since the timestamp was captured, by comparing it to the timestamp value reported in this field. If the reported timestamp value in this field is higher than the lower 32-bits of the current synchronized time, the time has wrapped around resulting the need to adjust the upper 32-bits of the current synchronized time by subtracting 0x00000001 from it. This value is then used to set the upper 32 bits of the synchronized timestamp. If the time has not wrapped around, the upper 32 bits of the current synchronized time read from the register, is used to set the upper 32 bits of the synchronized timestamp. 3. The upper 32 bits value determined in step 2, and the lower 32 bits reported in this field are combined to create the 64-bit synchronized timestamp. If the free-running timestamp is reported, the timestamp in this field is the lower 32 bits of the 64-bit free running timestamp. The value in this field is in units of NETC clock ticks with the range 0 to 0xFFFFFFFF. To reconstruct the 64-bit free running timestamp, perform the following steps: 1. Get the current free-running by issuing a read to the TMR_FRT_L register followed immediately by a read to the TMR_FRT_H register. 2. Check if the lower 32-bits of the current free-running time has wrapped around since the timestamp was captured, by comparing it to the timestamp value reported in this field. If the timestamp value reported in this field is higher than the lower 32-bits of the current free-running time, the time has wrapped around resulting the need to adjust the upper 32-bits of the current free-running time by subtracting 0x00000001 from it. This value is then used to set the upper 32 bits of the free-running timestamp. If the time has not wrapped around, the upper 32 bits of the current running time read from the register, is used to set the upper 32 bits of the free-running timestamp. 3. The upper 32 bits value determined in step 2, and the lower 32 bits reported in this field are combined to create the 64-bit free running timestamp.

53.4.2.2.4 MAC Layer

53.4.2.2.4.1 Ethernet MAC

Following are the major functions provided by the Ethernet MAC:

- Full MAC layer implementation compliant with IEEE802.3 specification.
- 10M/100M/GE port speeds.
- Half duplex support at 10M and 100M speeds.
- EEE (Energy Efficient Ethernet).
- CRC-32 checking with forwarding of the FCS field to upper layer processing functions.
- CRC-32 generation and append on transmit or forwarding of provided FCS.
- Ethernet PAUSE frame (802.3 Annex 31B) termination.
- One-step and two-step timestamping support for PTP/IEEE1588/IEEE802.1AS-2020.
- 802.3 basic and mandatory managed Objects statistic counters and IETF Management Information Database. (MIB) package (RFC2665) and Remote Network Monitoring (RMON) counters.
- IEEE 802.3 physical-layer fields encapsulation and decapsulation such as the Preamble, Start of Frame Delimiter and Inter-Frame Gap. The Preamble must be an integral number of octets (octet-aligned).

53.4.2.2.4.1.1 MAC Flow Control

The Ethernet MAC supports one mode of flow control:

- Link Pause Flow Control: This is the standard IEEE 802.3 defined PAUSE frame to allow pausing the remote device connected to the link (link local pause). Upon reception (if enabled by configuration) the transmitter is paused for the time received in the PAUSE frame.

If a frame of another mode is received it is treated as a regular command frame. Depending on PM*a*_COMMAND_CONFIG[CNT_FRM_EN], it is either discarded or forwarded to the user application but has no effect within the MAC. See PM*a*_COMMAND_CONFIG register for configuration options.

53.4.2.2.4.1.1.1 MAC Transmit Pause

In transmit direction, MAC receives flow control requests from upper layer processing functions, and determines based on the request and current state whether to schedule transmitting a link PAUSE control frame. The following conditions trigger the transmission of a link PAUSE control frame:

- A Tx pause is not in progress and there is a request for pausing the transmission of the other end of the link – the MAC schedules transmitting a new link PAUSE control frame with the programmed quanta value (PMn_PAUSE_QUANTA[PQNT]), enters 'Tx pause in progress' state, and starts counting down the quanta threshold (PM1_PAUSE_THRESH[QTH]).
- A Tx pause is already in progress, but the upper layer function dropped the pause request (request link partner resume transmission) – the MAC schedules transmitting a new link PAUSE control frame with QUANTA = 0, stops the quanta threshold countdown, and exits 'Tx pause in progress' state
- A Tx pause is already in progress, the request is still outstanding, and the quanta threshold countdown counter expires – the MAC schedules transmitting a refresh link PAUSE control frame with QUANTA=PQNT, stays in 'Tx pause in progress' state, and restarts counting down the quanta threshold with the counter reset to QTH.

The MAC completes transmitting any application frame that has already started transmission before it schedules transmitting a new link PAUSE control frame. Transmit of link PAUSE control frames takes priority over pending application frames that have not started transmission.

When preemption is enabled, upper layer processing functions must be configured in such a way that they won't generate requests to generate PAUSE frame. See Preemption for more details.

53.4.2.2.4.1.1.2 MAC Receive Pause Operation

PAUSE frames are processed in the MAC receive engine if PM*a*_COMMAND_CONFIG[PAUSE_IGN] is set to 0. In that case, the quanta is extracted from the terminated PAUSE frame and sent to the MAC transmit control. The quanta extracted is loaded into an internal timer to pause the transmitter. The transmitter continues to complete any ongoing frame transmission and then enters a pause state where it does not read any frames from the transmit FIFO. When the transmitter has reached its pause state, the timer starts to decrement. When the timer reaches 0 the transmitter resumes to normal transmission of frames.

If a CRC or a length error is detected, the quanta is ignored. If the PM*a*_COMMAND_CONFIG[PAUSE_IGN] is set to 1, the quanta is not extracted from the received PAUSE frame. In either case, transmit continues normal transmission without interruption.

PAUSE frames are also optionally forwarded to the MAC Client if PM*a*_COMMAND_CONFIG[PAUSE_FWD] is set to 1.

Table 382. PAUSE Frame Processing Options

PAUSE_IGN	PAUSE_FWD	Description
0	0	• PAUSE frames terminated • Quanta extracted and sent to MAC transmit path • PAUSE frames not transferred to MAC FIFO interface
0	1	• PAUSE frames terminated • Quanta extracted and sent to MAC transmit path • PAUSE frames transferred to MAC FIFO interface
1	0	• PAUSE frames not terminated • Quanta not extracted and not sent to MAC transmit path • PAUSE frames not transferred to MAC FIFO interface
1	1	• PAUSE frames not terminated • Quanta not extracted and not sent to MAC transmit path • PAUSE frames transferred to MAC FIFO interface

A PAUSE frame is valid only if all of the following conditions are true:

- Ethertype is set to 0x8808
- Opcode immediately following Ethertype is 0x0001
- Destination MAC address matches configured primary MAC address (registers PM*a*_MAC_ADDR_0/1) or the control frame multicast address 01-80-C2-00-00-01
- Valid CRC
- Length of 64 bytes
- Received by the express MAC

When a PAUSE frame is received, the statistics counter aRxPAUSEMACCtrlFrames is always incremented.

53.4.2.2.4.1.2 Preemption

NETC supports IEEE 802.1Qbu preemption. A physical port which supports preemption contains two MACs; i.e. MAC 0 and 1. When frame preemption is enabled, MAC 0 handles express traffic, and MAC 1 handles preemptive traffic. Preemption is enabled on a per port basis by setting MAC_MERGE_MMCSR[ME] to 1 or 2. The two MACs are merged on one port as defined in 802.3 Clause 99.When frame preemption is disabled, all traffic goes through MAC 0. The ability to designate one or more priorities as express (and the remaining priorities as preemptable) means that, when an express frame is ready for transmission and a preemptable frame is already in the process of being transmitted, the delay before the express frame transmission starts is at worst

the sum of the minimum final (64B) and non-final (MAC_MERGE_MMCSR[RAFS]) fragment sizes minus four octet times and will often be 64 octet times or less.

The traffic class assignment to either the express MAC or the preemptable MAC is specified by setting PFPCR[FPE_TCa] (where a corresponds to the traffic class number). By default all traffic classes are assigned to the express MAC. Preemption, is therefore a useful tool for reducing the jitter experienced by time-sensitive frames. These frames could be transmitted using any of the available transmission selection algorithms, but the likely choice would be an algorithm that is intended for use with time sensitive data, such as the time-gating, time specific departure scheduling, or credit-based shaping. In common use cases, it would be expected that express traffic would have a higher priority than preemptable traffic.

When preemption is enabled, generation of PAUSE frames must be disabled, as stated in the IEEE 802.3 standard.

When preemption is disabled, PAUSE frames can be transmitted and received on the express MAC only.

When preemption is enabled on a port, IEEE 1588 PTP one-step timestamping is not supported.

For statistics related to preemption, see section Ethernet MAC Statistics Counters.

53.4.2.2.4.1.3 Ethernet MAC Statistics Counters

NETC offers a set of statistics counters required in IEEE 802.3 basic, mandatory and recommended management information packages. In addition, these counters can be used to generate applicable objects of the Management Information Base (MIB, MIB-II) according to IETF RFC2665 for SNMP managed environments. For monitoring applications, RMON counters are available according to IETF RFC2819.

In NETC, a physical port which supports preemption will contain two MACs - MAC 0 and 1. When frame preemption is enabled, MAC 0 handles express traffic, and MAC 1 handles preemptive traffic. As a result, the user needs to read the counters from both MACs and add them together to get the correct count values. Preemption Verify and Respond mPackets are not included in MAC statistics. When frame preemption is disabled, all traffic goes through MAC 0. In this case, the user only needs to look at the counters in MAC 0.

Following tables list the available Ethernet MAC statistics counters.

Table 383. Ethernet MAC Traffic Statistics Counters

Direction	Counter	Description
Rx	PM*a*_REOCTn	Received Ethernet octets counter (etherStatsOctets); good and errored
Rx	PM*a*_ROCTn	Received octets counter (ifInOctets); good
Rx	PM*a*_RXPFn	Received valid PAUSE frames counter
Rx	PM*a*_RFRMn	Received frames counter; good
Rx	PM*a*_RVLANn	Received VLAN tagged frames counter; good
Rx	PM*a*_RUCAn	Received unicast frames counter; good
Rx	PM*a*_RMCAn	Received multicast frames counter; good
Rx	PM*a*_RBCAn	Received broadcast frames counter; good
Rx	PM*a*_RPKTn	Received packets counter (etherStatsPkts); good and errored
Rx	PM*a*_RMIN63n	Received Min to 63-octet packets counter; good only
Rx	PM*a*_R64n	Received 64-octet packets counter (etherStatsPkts64Octets); good and errored
Rx	PM*a*_R127n	Received 65 to 127-octet packets counter (etherStatsPkts65to127Octets); good and errored

Table continues on the next page...

Table 383. Ethernet MAC Traffic Statistics Counters (continued)

Direction	Counter	Description
Rx	PM*a*_R255n	Received 128 to 255-octet packets counter (etherStatsPkts128to255Octets); good and errored
Rx	PM*a*_R511n	Received 256 to 511-octet packets counter (etherStatsPkts256to511Octets); good and errored
Rx	PM*a*_R1023n	Received 512 to 1023-octet packets counter (etherStatsPkts512to1023Octets); good and errored
Rx	PM*a*_R1522n	Received 1024 to 1522-octet packets counter (etherStatsPkts1024to1522Octets); good and errored
Rx	PM*a*_R1523Xn	Received 1523 to Max-octet packets counter (etherStatsPkts1523toMaxOctets); good and errored
Rx	PM*a*_RCNPn	Received control packets counter (type 0x8808 but not PASUE packets)
Rx	MAC_MERGE_MMFAOCR	Received valid reassembled frames counter
Rx	MAC_MERGE_MMFCRXR	Received additional mPackets counter
Tx	PM*a*_TEOCTn	Transmitted Ethernet octets counter (etherStatsOctets); good and errored
Tx	PM*a*_TOCTn	Transmitted octets counter (ifOutOctets); good
Tx	PM*a*_TXPFn	Transmitted valid PAUSE frames counter
Tx	PM*a*_TFRMn	Transmitted frames counter; good
Tx	PM*a*_TVLANn	Transmitted VLAN tagged frames counter; good
Tx	PM*a*_TUCAn	Transmitted unicast frames counter; good
Tx	PM*a*_TMCAn	Transmitted multicast frames counter; good
Tx	PM*a*_TBCAn	Transmitted broadcast frames counter; good
Tx	PM*a*_TPKTn	Transmitted packets counter (etherStatsPkts); good and errored
Tx	PM*a*_T64n	Transmitted 64-octet packets counter (etherStatsPkts64Octets); good and errored
Tx	PM*a*_T127n	Transmitted 65 to 127-octet packets counter (etherStatsPkts65to127Octets); good and errored
Tx	PM*a*_T255n	Transmitted 128 to 255-octet packets counter (etherStatsPkts128to255Octets); good and errored
Tx	PM*a*_T511n	Transmitted 256 to 511-octet packets counter (etherStatsPkts256to511Octets); good and errored
Tx	PM*a*_T1023n	Transmitted 512 to 1023-octet packets counter (etherStatsPkts512to1023Octets); good and errored
Tx	PM*a*_T1522n	Transmitted 1024 to 1522-octet packets counter (etherStatsPkts1024to1522Octets); good and errored

Table continues on the next page...

Table 383. Ethernet MAC Traffic Statistics Counters (continued)

Direction	Counter	Description
Tx	PM*a*_T1523Xn	Transmitted 1523 to Max-octet packets counter (etherStatsPkts1523toMaxOctets); good and errored
Tx	PM*a*_TCNPn	Transmitted control packets counter (type 0x8808 but not PAUSE packets)
Tx	MAC_MERGE_MMFCTXR	Transmitted additional mPackets counter
Tx	MAC_MERGE_MMHCR	Hold transitions from false to true counter

Table 384. Ethernet MAC Frame Drops/Errors Counters

Direction	Scope	Counter	Description
Rx	MAC	PM*a*_RERRn	MAC received frame error counter; incremented for each frame received with an error (except for undersized frames)
Rx	MAC	PM*a*_RUNDn	MAC received undersized frames counter
Rx	MAC	PM*a*_ROVRn	MAC received oversized frames counter
Rx	MAC	PM*a*_RFCSn	MAC received frame check sequence (FCS) error counter
Rx	MAC	PM*a*_RFRGn	MAC received undersized frames with FCS error counter
Rx	MAC	PM*a*_RJBRn	MAC received oversized frames with FCS error counter
Rx	MAC	PM*a*_RDRPn	MAC received frame drops counter; full or truncated frames dropped due to receive FIFO overflows
Rx	MAC	PM*a*_RDRNTPn	MAC received non-truncated frame drops count; full frame dropped due to receive FIFO overflows
Rx	MAC	PM*a*_RFRGn	Receive fragment packet counter; incremented for each packet which is shorter than the length programmed in PM*a*_MINFRM register received with a bad CRC.
Rx	MAC	PM*a*_RJBRn	Receive jabber packet counter; incremented for each packet which is larger than the maximum frame length specified in register PM*a*_MAXFRM[MAXFRM] register received with a bad CRC.
Rx	MAC Merge	MAC_MERGE_MMFAECR	MAC Merge frame reassembly error counter
Rx	MAC Merge	MAC_MERGE_MMFSECR	MAC Merge frame SMD error counter
Tx	MAC	PM*a*_TERRn	MAC transmitted frames error counter
Tx	MAC	PM*a*_TUNDn	MAC transmitted frames counter less than 64B with good FCS
Tx	MAC	PM*a*_TDFRn	MAC transmitted frames counter with deferral (half duplex only)
Tx	MAC	PM*a*_TSCOLn	MAC transmitted frames counter with single collision (half duplex only)
Tx	MAC	PM*a*_TMCOLn	MAC transmitted frames counter with multiple collisions (half duplex only)

Table continues on the next page...

Table 384. Ethernet MAC Frame Drops/Errors Counters (continued)

Direction	Scope	Counter	Description
Tx	MAC	PM*a*_TECOLn	MAC dropped frames counter with excess collisions (half duplex only)
Tx	MAC	PM*a*_TLCOLn	MAC dropped frames counter with late collision (half duplex only)

53.4.2.2.4.1.4 Ethernet MAC Graceful Stop

In order to stop receive:

1. Write PM1_COMMAND_CONFIG[RX_EN]=0b; this step is required only if frame preemption is enabled.
2. Wait for PM1_IEVENT[RX_EMPTY] to be set; this step is required only if frame preemption is enabled.
3. Write PM0_COMMAND_CONFIG[RX_EN]=0b.
4. Wait for PM0_IEVENT[RX_EMPTY] to be set.
5. Wait for PSR[RX_BUSY] to clear.
6. Stop the upper layer functions from receiving any frames by setting POR[RXDIS] to 1b.

In order to stop transmit:

1. Perform the graceful stop of traffic from upper layer processing functions; set POR[TXDIS] to 1b, for the upper layer processing functions to stop enqueuing frames for transmission and to flush any internal Tx queues.
2. Wait for PM0_IEVENT[TX_EMPTY] to be set. If frame preemption is enabled, wait for TX_EMPTY from both PM0_IEVENT and PM1_IEVENT registers.
3. Wait 64 byte time for Tx packet transmission to complete.
4. Write PM0_COMMAND_CONFIG[TX_EN]=0b.
5. Write PM1_COMMAND_CONFIG[TX_EN]=0b; this step is required only if frame preemption is enabled.

53.4.2.2.4.2 Pseudo MAC

A switch port connects internally to an ENETC port using a pseudo link. Pseudo MACs are used at both ends of a pseudo link to transfer frames across a pseudo link.

Functions provided by the pseudo MAC are:

- **Frame transfer across a pseudo link** - Main tasks are:
  - Pseudo links are implemented without a physical transmission medium, and thus frames are not encapsulated with IEEE 802.3 physical-layer fields such as the Preamble, Start of Frame Delimiter and Inter-Frame Gap, when transferred across a pseudo link. However when scheduling a frame, the PTXSDUOR register can be configured with the appropriate per frame overhead; overhead (number of bytes) to be added to the actual length of each frame, to reflect the PPDU (Physical Layer PDU) frame length on a physical Ethernet link.
  - Only frame descriptors are transferred across a pseudo link; actual frames continue to remain in the same memory location as they are transferred across the pseudo link. Thus there is no need to store or copy these frames as they have already been stored in the shared internal buffer memory.
  - Out of band metadata to carry switch management port related info (source port, Host Reason). There is also an annotated frame header carried along with the frame when performing frame modifications (only added on transmit, by the switch).
  - Rate limiting function if 802.1Qbv operation is not configured. The rate limiting function is achieved by enabling time gate scheduling (PTGSCR[TGE] set to 1) with no administrative gate control list configured, and for ENETC with EaTGSLR[ZERO_LOOKAHEAD] set to 1. There is no equivalent "ZERO_LOOKAHEAD" field for the switch as there is no need to compensate for any DMA delay; default value for the S0TGSLR[MIN_LOOKAHEAD] field is suitable

to rate limit a switch port bound to pseudo MAC. The bit rate is specified in PCR[PSPEED]. This also requires the timer function to be configured and enabled. Either the default nanosecond timer or the 1588 timer can be used to generate the required nanosecond resolution time. The default nanosecond timer is configured and enabled by default out of reset. The pseudo MAC for both ENETC and the switch will have this rate limiting function configured and enabled by default.

- **Statistics** - Subset of RMON counters provided, no frame length related counts provided.
- **Timestamp** - Support consists of:
  - Receive (switch and ENETC): Two timestamps (one sampling the synchronized clock whereby the other sampling the free running clock) at the time the frame was transferred across the pseudo link.
  - Transmit (ENETC only): On any given transmit frame, the pseudo MAC returns the timestamp at the moment where the frame was transferred across the pseudo link. The Ethernet Tx I/F returns this value to HTA which in turn uses this as the indication that the corresponding transmit BD descriptors have been transmitted, and if the IEEE 1588 PTP two-step timestamp offload is enabled, writes back the returned timestamp in the metadata of the host transmit descriptor. Note that IEEE 1588 PTP one-step timestamp offload is not supported on the pseudo MAC.
- **Pseudo link control and configuration.**
  - Transmit link speed configuration (specified in PCR[PSPEED]); used by the egress port scheduler to determine the transmit link speed on a port.
  - Padding configuration; indicates whether padding is to be applied; only used by HTA Transmit to determine if a frame needs to be padded to a minimum frame length of 64 bytes.
  - Pseudo MAC enable/disable (POR[TXDIS/RXDIS]).
- **Flow control** - Flow control on per port basis to mimic Ethernet PAUSE functionality (no Pause frame generation).
  - Internal flow control signals are carried across the pseudo link in both directions, to request the other end of the link to pause transmission for all traffic classes.
  - These flow control signals are directly connected to the respective pseudo link end point transmit scheduler. When these flow control signals are asserted, the transmit scheduler enters a pause state where it stops scheduling frames for all traffic classes.
- **FCS** - The pseudo MAC does not remove the FCS(s). The FCS(s) are carried in the frame, and remain as the frame is processed by the ENETC receive datapath. No FCS checking is performed on receive as no transmission of bits occurs on pseudo link.
- **Preemption** - Not applicable.

No minimum and maximum frame length checking is performed on both receive and transmit.

The only frame drop that can occur is, if there is no available internal memory for creating the annotated frame header when performing frame modifications (on transmit, and only applicable to the switch). These frame discards are counted against the port's Tx discard count register (PTXDCR) along with the setting of the Shared Memory Resource Exhaustion Discard Reason (SMREDR) flag to 1 in the port's Tx discard count reason register 0 (PTXDCRR0).

53.4.2.2.4.2.1 Pseudo MAC Statistics Counters

Following table lists the available pseudo MAC statistics counters.

Table 385. Pseudo MAC Traffic Statistics Counters

Direction	Counter	Description
Rx	PPMROCR0/1	Received octets counter (ifInOctets)
Rx	PPMRUFCR0/1	Received unicast frames counter
Rx	PPMRMFCR0/1	Received multicast frames counter

Table continues on the next page...

Table 385. Pseudo MAC Traffic Statistics Counters (continued)

Direction	Counter	Description
Rx	PPMRBFCR0/1	Received broadcast frames counter
Tx	PPMTOCR0/1	Transmitted octets counter (ifOutOctets)
Tx	PPMTUFCR0/1	Transmitted unicast frames counter
Tx	PPMTMFCR0/1	Transmitted multicast frames counter
Tx	PPMTBFCR0/1	Transmitted broadcast frames counter

NOTE

Pseudo MAC statistics counters are not updated for frames sent towards the switch management port with a Host Reason set to a codepoint other than 0 (not regular forwarded frame).

53.4.2.2.5 Common Internal Memory

The common internal memory (also referred in the document as "common memory") is used by NETC functions for frame buffering and for the storage of exact match lookup tables. The common internal memory is divided into a set of fixed (static) non-overlapping memory partitions, namely:

- Shared memory partition.
- NETC function partitions used for the storage of indexed tables; one partition for each of the NETC functions.
- Hash table bucket array partition. The hash table bucket array partition is a global hash table bucket array memory that is shared by all hash tables from the switch and ENETC functions.
- Guaranteed hash table entry partition. Each entry in this memory partition is currently used to store the content of a switch FDB entry, when the FDB table entry is added as a Content-Addressable-Memory (CAM) entry, as opposed to an hash entry (hash space), due to collision chain length having reached its maximum length.

The layout of the common internal memory is shown in the figure below:

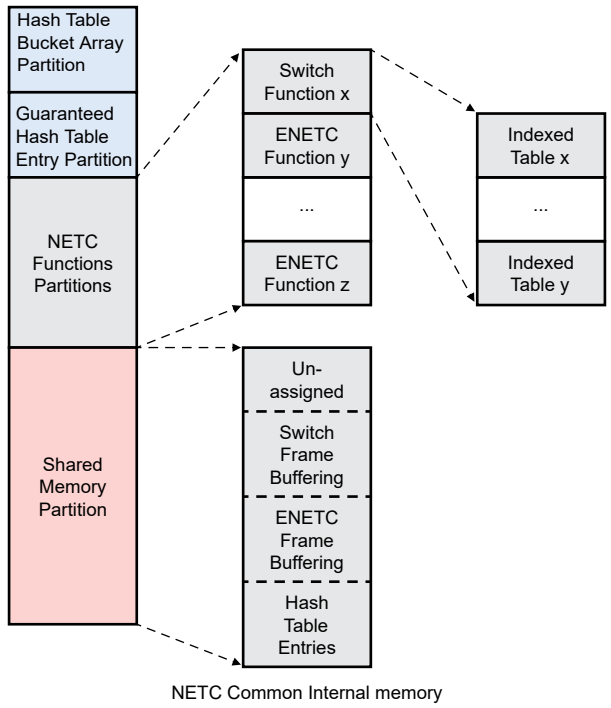


Figure 317. NETC Common Internal Memory Layout

Common internal memory allocation for the above memory partitions is performed at pre-boot initialization based on the setting of Integrated Endpoint Register Block (IERB) registers with the exception of the guaranteed hash table entry partition whose allocation is fixed (or "hardwired"). The number of guaranteed hash table entries is indicated in the Guaranteed hash table entry memory capability register (GHTEMCAPR).

- Hash table bucket array partition allocation is set to the value configured in the Hash bucket table memory allocation register (HBTMAR).
- Switch function indexed table partition is set to the value configured in the Switch 0 index table memory allocation register (S0ITMAR).
- Size of each ENETC function indexed table partition is set to the value configured in a corresponding ENETC index table memory allocation register (EaITMAR).
- Shared memory partition is set the total size of the common internal memory (CMCAPR) minus the values in HBTMAR, GHTEMCAPR, S0ITMAR and EaITMAR(s).

The allocation is performed in a contiguous fashion, allocating first the common internal memory for the hash table bucket array partition followed by the guaranteed hash table entry partition, by the switch function indexed table partition and ENETC function indexed table partition(s), with the remaining space allocated to the shared memory partition.

Shared memory partition

The shared memory partition is managed using a single free list of fixed equal size units of memory. The shared memory partition is used for the following storage functions:

- Frame buffering for the switch; this is referred to as the switch shared internal buffer memory
- Receive frame buffering for all ENETC ports; this is referred to as the ENETC receive shared internal buffer memory
- Storage for table lookup entries; allocation/deallocation of those entries to tables is managed by the hardware

An allotment is configured at pre-boot initialization time for each of the above storage functions, by configuring the following IERB registers:

- Switch 0 shared memory buffer allotment register (S0SMBAR) for the switch shared internal buffer memory.

- ENETC receive shared memory buffer allotment register (ERSMBAR) for the ENETC receive shared internal buffer memory.
- Switch 0 hash table memory allotment register (S0HTMAR) for the hash table lookup entries used by the switch.
- ENETC hash table memory allotment register (EaHTMAR) for the hash table lookup entries used by ENETC, where a represents an ENETC function.

The sum of all of individual allotments must not exceed the total size of the shared common memory region minus the amount of memory required by the switch and ENETC functions for buffering frames that haven't been assigned to a specific memory allotment (frames in flight). The latter is referred to as the "unassigned" frame buffering memory (memory acquired but not assigned to a memory allotment). The amount of unassigned buffering memory is determined by adding the following memory amounts (20 bytes of frame buffering per memory word/unit):

- Sum of all physical ports usage (switch and ENETC) for storing frames being received from the MAC but not yet accounted against the switch frame buffering or ENETC frame buffering. For a given port, this amount corresponds to the maximum frame size that can be received from the express MAC plus the maximum frame size that can be received from the preemptable MAC (the later applies only if preemption is enabled/used).
- Sum of all ENETC instances usage for transmit frame buffering. For a given ENETC instance, the recommended amount is 2 times the maximum frame size for the express MAC or the pseudo MAC plus 2 times the maximum frame size for the preemptable MAC (the later applies only if preemption is enabled/used).
- One additional maximum frame size for the switch, as the switch frame buffering usage can go over by one maximum frame size above the amount of the switch buffering memory configured in S0SMBAR.
- 64 memory words/units of 24 bytes for internal hardware usage.

Free memory units from the free list are available to be dynamically allocated to any storage function subject to the allotment constraint of that function.

When the amount of free memory units falls below the threshold indicated in SMDTR[THRESH], a depletion indication is asserted internally, which may trigger "intelligent drop" of frames from the ingress queues in the ENETC Ingress Congestion Manager (ICM).

NETC function partitions

NETC function partitions are used for storing specific switch and ENETC indexed tables. Specifically, there is one partition for the switch function and one partition for each ENETC function, where each partition may hold one or more indexed tables. These tables are implemented as linear arrays of words, stored in contiguous manner, meaning that words are adjacent to one another in memory with no gaps between them. Indexed table entries can occupy one or more words, and within a given table, entries can be either of fixed size or variable size.

In the case of fixed size entries, the table is accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the table. When accessing a table with fixed size entries, the actual memory position of an entry is calculated by hardware by multiplying the entry index by the number of words occupied by an entry, and then adding the result to the base memory address of the table. The base memory addresses of the indexed tables in the common internal memory, are determined by hardware when the sizes of the indexed tables are configured.

In the case of variable size entries, the table is accessed using an index (0, 1, 2, ..., n) expressed in units of words relative to the start of the table, that points to the first word of the entry to be accessed.

Note that not all indexed tables are stored in common memory. Some of the indexed tables are stored in dedicated internal memories, and thus there sizes cannot be changed. Switch and ENETC indexed tables that are stored in the common internal memory are listed respectively in Table 386 and Table 387.

Within each function (switch and ENETC(s)), memory allocation is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) registers. These allocations can also be changed when the functions are initialized through their respective base function registers. The IERB and function base registers used to configure the size of each indexed table, are shown below.

Table 386. Switch Indexed Tables Allocation Registers

Table Name	IERB Registers	Switch Base Registers
Rate Policer table	S0RPITMAR[NUM_WORDS]	RPITMAR[NUM_WORDS]
Ingress Stream Count table	S0ISCITMAR[NUM_WORDS]	ISCITMAR[NUM_WORDS]
Ingress Stream table	S0ISITMAR[NUM_WORDS]	ISITMAR[NUM_WORDS]
Ingress Sequence Generation table	S0ISQGITMAR[NUM_WORDS]	ISQGITMAR[NUM_WORDS]
Stream Gate Instance table	S0SGIITMAR[NUM_WORDS]	SGIITMAR[NUM_WORDS]
Stream Gate Control List table	S0SGCLITMAR[NUM_WORDS]	SGCLITMAR[NUM_WORDS]
Frame Modification table	S0FMITMAR[NUM_WORDS]	FMITMAR[NUM_WORDS]
Frame Modification Data table	S0FMDITMAR[NUM_WORDS]	FMDITMAR[NUM_WORDS]

Table 387. ENETC Indexed Tables Allocation Registers

Table Name	IERB Registers	ENETC a Base Registers
Rate Policer table	EaRPITMAR[NUM_WORDS]	RPITMAR[NUM_WORDS]
Ingress Stream Count table	EaISCITMAR[NUM_WORDS]	ISCITMAR[NUM_WORDS]
Ingress Stream table	EaISITMAR[NUM_WORDS]	ISITMAR[NUM_WORDS]
Stream Gate Instance table	EaSGIITMAR[NUM_WORDS]	SGIITMAR[NUM_WORDS]
Stream Gate Control List table	EaSGCLITMAR[NUM_WORDS]	SGCLITMAR[NUM_WORDS]

The sum of all of the table sizes within a NETC function must not exceed the size of the NETC function memory partition. Within a function, the memory allocation is performed in a contiguous fashion, assigning in the order listed in the table above; starting first with the Rate Policer table, followed by the Ingress Stream Count table, and so on.

53.4.2.2.6 Time Gate Scheduling Memory

The Time Gate Scheduling internal memory is used to store all of the administrative gate control lists and the operational gate control lists information, for time gate scheduling. The memory is shared between the switch and all of the ENETC instances supporting time gate scheduling. Each gate control list is comprised of a sequence of gate operation entries where each entry occupies one word of memory. The total number of words in the Time Gate Scheduling internal memory is indicated in TGSMCAPR[NUM_WORDS].

The allocation to the switch and the ENETC instances, is based on the setting of the IERB registers S0TGSTAR[NUM_WORDS] and E*a*TGSTAR[NUM_WORDS], respectively. The sum of all of the individual allocations must not exceed the total size of the Time Gate Scheduling internal memory. The values configured in the IERB registers S0TGSTAR[NUM_WORDS] and E*a*TGSTAR[NUM_WORDS], are then reflected in the respective switch and ENETC instance registers TGSTCAPR[NUM_WORDS].

53.4.2.2.7 Ingress Port Filter TCAM

The Ingress Port Filter Ternary-Content-Addressable Memory (TCAM) is used to implement the Ingress Port Filter tables for the switch and for each ENETC instance. The total number of TCAM words/lines available to all NETC functions, is indicated in IPFTCAPR[NUM_WORDS]. Each TCAM word (or line) is 48-bit wide.

The allocation to the switch and the ENETC instances, is based on the setting of the IERB registers S0IPFTMAR[ALLLOC] and E*a*IPFTMAR[ALLOC], respectively. The sum of all of the individual allocations must not exceed the total size of the Ingress Port Filter TCAM. The values configured in the IERB registers S0IPFTMAR[ALLLOC] and E*a*IPFTMAR[ALLOC], are then reflected in the respective switch and ENETC instance registers IPFTCAPR[NUM_WORDS].

53.4.2.2.8 Error Management

NETC supports three different types of errors which are capable of generating an interrupt through a process described in Error Reporting. The following error categories exist:

- Correctable errors

  Correctable errors include those error conditions where hardware can recover without any loss of information. Hardware corrects these errors and software intervention is not required. Examples of correctable errors include:

  — single-bit ECC errors

- Uncorrectable errors.

  Uncorrectable errors may impact the functionality of the interface. Depending on the severity and type of error, software intervention may be required. The detection of an uncorrectable fatal error will result in the device/function entering a safe state, see Error Reaction.

  — Non-fatal errors

    Non-fatal uncorrectable errors can be global or per function. To resume operation software may be required to reconfigure the entire function or part of it. A Function Level Reset (FLR) should not be required to continue operation.

  — Fatal errors

    Fatal uncorrectable errors can be global or per function. When an uncorrectable fatal error occurs it means NETC or a function has been compromised and is unable to perform a hardware task. Examples of this include multi-bit ECC errors in internal memory or other memories where the source cannot be isolated and stop operation (safe state reaction). To resume operation, a soft reset of NETC for global events or an FLR for function level events is required.

53.4.2.2.8.1 Error Reporting

NETC reports errors to the SoC using the INTA pin, see Wired Interrupt. For this to occur the PCIe layer must be properly configured.

Hardware detects an error and reports it directly to a device error register. There may be additional information captured at the device level depending on the error type. Once the device error has been detected it can be reported to the PCIe iEP function's Advanced Error Reporting capability structure. This is done by clearing the associated device Report Disable bit, see Error Detection and Handling for a full description of the involved device registers.

A device error reported to the PCIe function will set either of the bits:

- PCIE_CFC_AER_CORR_ERR_STAT[CORR_INT_ERR]
- PCIE_CFC_AER_UCORR_ERR_STAT[UCORR_INT_ERR]

For INTA to be asserted, the function must report the PCIe error to the Root Complex Event Collector. This is accomplished by clearing the appropriate function mask bit:

- PCIE_CFC_AER_CORR_ERR_MASK[CORR_INT_MASK]=b0
- PCIE_CFC_AER_UCORR_ERR_MASK[UCORR_INT_ERR_MASK]=b0.

It should be noted that the distinction between the fatal and non-fatal (severity) uncorrectable error is defined in the PCIe AER uncorrectable error severity register, PCIE_CFC_AER_UCORR_ERR_SEV[UCORR_INT_SEV]. It cannot be set on a per event basis for a function. Software may have to rely on the device registers to determine what action.

The Event Collector receiving an error from one of the associated functions, as defined by PCIE_CFC_RCEC_EPA_BITMAP[DEV_BITMAP], will set one of the following detect bits:

- PCIE_CFC_AER_ROOT_ERR_STAT[ERR_FATAL]
- PCIE_CFC_AER_ROOT_ERR_STAT[ERR_NON_FATAL]
- PCIE_CFC_AER_ROOT_ERR_STAT[ERR_CORR]

For the Event Collector to assert INTA for a reported error, it must enable the reporting of the error and clear the interrupt disable bit:

- PCIE_CFC_AER_ROOT_ERR_CMD[FATAL_ERR_RPT_EN]=b1
- PCIE_CFC_AER_ROOT_ERR_CMD[NON_FATAL_ERR_RPT_EN]=b1
- PCIE_CFC_AER_ROOT_ERR_CMD[CORR_ERR_RPT_EN]=b1
- PCI_CFH_CMD[INTR_DISABLE]=b0

53.4.2.2.8.2 Error Detection and Handling

The figure below shows a simplified flow diagram to determine which device error registers to read based on what has been reported in the PCIe Event Collector by one or more functions. Here it is assumed that the most critical error is handled first. Omitted from the flow is clearing of errors and repeating the process if additional errors are pending.

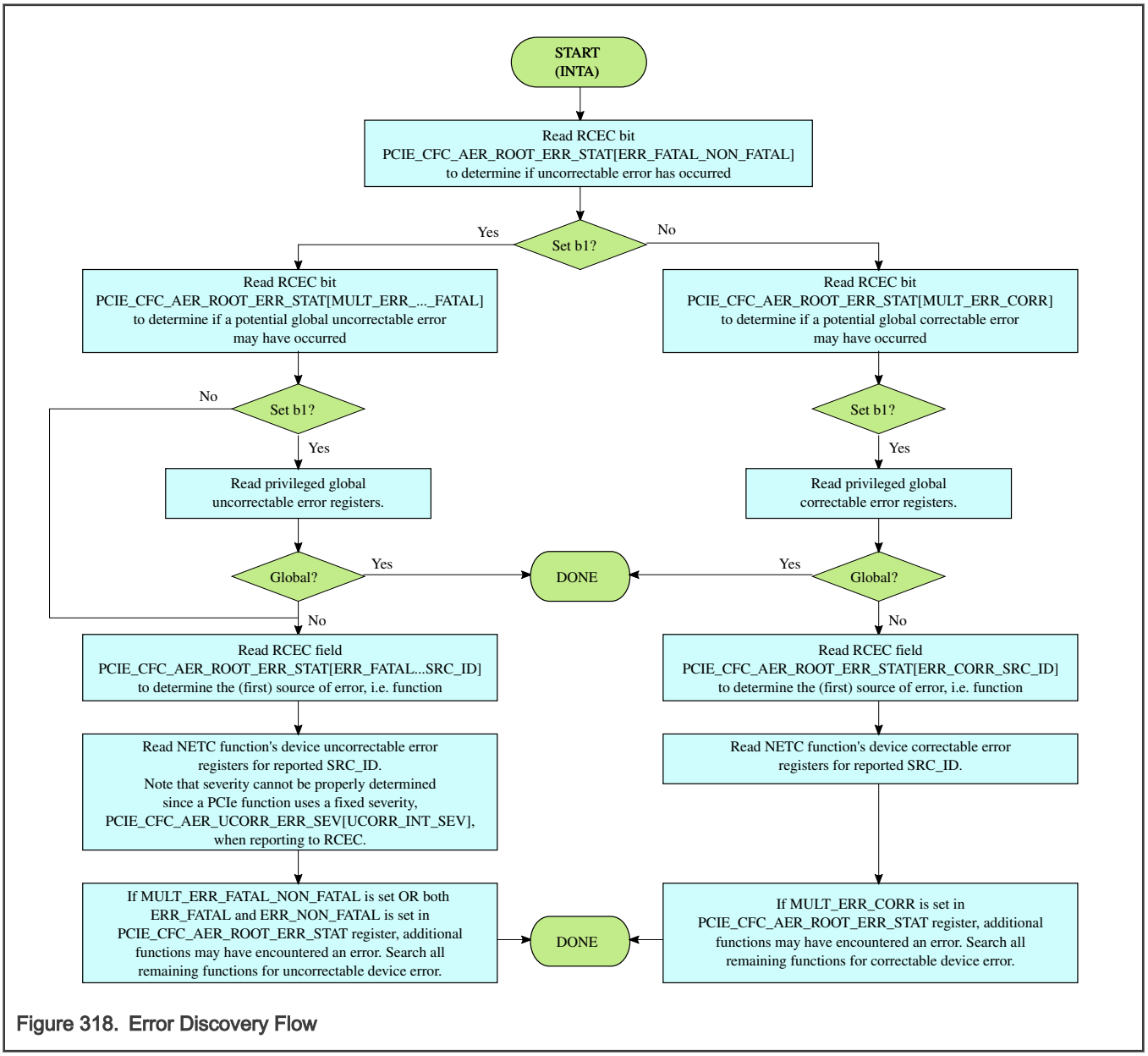


Figure 318. Error Discovery Flow

The tables below describe error handling in more detail; how each error is detected, enabled and where additional error capture information can be found.

Table 388. Uncorrectable Non-Fatal Software Errors

Error Description	Error Handling	
Received frame steered to a disabled station interface.	**Device Error Detect**[1]	SIUPESR[PE], SIUPESR[DROP_SI_EN]
	PCIe Error Report Disable	SIUPECR[RD]
	Error Capture[2]	SIUPESR[M] , SIUPECTR[COUNT]
	Target Function(s)	ENETC PF (SI0), ENETC VF (SI1+)
	Comments Programming error. The station interface is not enabled. The frame has been dropped.	
Received frame steered to a BD ring which has not been enabled by the SI.	**Device Error Detect**[1]	SIUPESR[PE], SIUPESR[DROP_RING_EN]
	PCIe Error Report Disable	SIUPECR[RD]
	Error Capture[2]	SIUPESR[M], SIUPECTR[COUNT]
	Target Function(s)	ENETC PF (SI0), ENETC VF (SI1+)
	Comments Programming error. The default receive BD ring 0 is not enabled or a ring is targeted by RFS/RSS that has not been enabled. The frame has been dropped and counted.	
Receive frame steered to a non existing ring (within a group) that results in a ring number outside the total number of rings assigned.	**Device Error Detect**[1]	SIUPESR[PE], SIUPESR[DROP_RING_SEL]
	PCIe Error Report Disable	SIUPECR[RD]
	Error Capture[2]	SIUPESR[M], SIUPECTR[COUNT]
	Target Function(s)	ENETC PF (SI0), ENETC VF (SI1+)
	Comments Programming error. If a selected BD ring is outside the range of rings assigned for a group, a ring for a higher numbered group may be selected and no error reported. In the case the ring (number) selected exceeds the total number of rings assigned to the SI, an error will occur. The frame has been dropped and counted.	
Illegal receive BD ring mapping (e.g. IPV to BDR, RFS BD ring selection out of range, no receive rings mapped).	**Device Error Detect**[1]	SIUPESR[PE], SIUPESR[DROP_RING_SEL]
	PCIe Error Enable/Disable	SIUPECR[RD]
	Error Capture[2]	SIUPESR[M], SIUPECTR[COUNT]
	Target Function(s)	ENETC PF (SI0), ENETC VF (SI1+)
	Comments Programming error. There is an illegal mapping from Internal Priority Value to BD ring number. The BD ring number exceeds the assigned number of rings for the station interface. The frame has been dropped and counted.	
Switch / ENETC misconfiguration.	**Device Error Detect**[1]	n/a
	PCIe Error Enable/Disable	n/a

Table continues on the next page...

Table 388. Uncorrectable Non-Fatal Software Errors (continued)

Error Description	Error Handling	
	Error Capture[2]	ENETC: PRXDCRR0[ITEDR] Switch: PRXDCRR0[ITEDR/FMMEDR], PTXDCRR0[ITEDR/FMMDR]
	Target Function(s)	Switch, ENETC
	Comments These Switch and ENETC programming errors do not have the capability to generate interrupt or faults to the FCCU. Software must read these bits to check the configuration for correctness.	

Table 389. Correctable Hardware Errors

Error Description	Error Handling	
Internal memory single bit ECC error	**Device Error Detect**[1]	Switch/ENETC: CMESR[SBEE] SI: SICMESR[SBEE] Global: CMESR[SBEE]
	PCIe Error Report Disable/Threshold	Switch/ENETC: CMECR[THRESHOLD] SI: SICMECR[THRESHOLD] Global: CMECR[THRESHOLD]
	Error Capture[2]	Switch/ENETC: CMESR[MEM_ID] SI: SICMESR[MEM_ID] Global: CMESR[MEM_ID]
	Target Function(s)	Switch, ENETC, ENETC PF (SI0), ENETC VF (SI1+), Global
	Comments Single bit ECC errors are flagged internally by NETC and are detected at a function or global level, see Table 392. This type of error is correctable by hardware.	

Table 390. Uncorrectable Non-Fatal Hardware Errors

Error Description	Error Handling	
Switch system bus error.	**Device Error Detect**[1]	UNSBESR[SBE], CBDR*a*SR[SBE]
	PCIe Error Report Disable	UNSBECR[THRESHOLD]
	Error Capture[2]	UNSBESR[SB_ID], UNSBECTR[COUNT]
	Target Function(s)	Switch
	Comments See Table 394 for more information.	
ENETC station interface receive system bus error.	**Device Error Detect**[1]	SIUNSBESR[SBE], RB*a*SR[SBE]

Table continues on the next page...

Table 390. Uncorrectable Non-Fatal Hardware Errors (continued)

Error Description	Error Handling	
	PCIe Error Report Disable	SIUNSBECR[THRESHOLD]
	Error Capture[2]	SIUNSBESR[SB_ID], SIUNSBECTR[COUNT]
	Target Function(s)	ENETC PF (SI0), VF (SI1+)
	Comments See Table 394 for more information.	
ENETC station interface transmit system bus error.	**Device Error Detect**[1]	SIUNSBESR[SBE], TB*a*SR[SBE]
	PCIe Error Report Disable	SIUNSBECR[THRESHOLD]
	Error Capture[2]	SIUNSBESR[SB_ID], SIUNSBECTR[COUNT], TB*a*CIR[BDR_INDEX]
	Target Function(s)	ENETC PF (SI0), VF (SI1+)
	Comments See Table 394 for more information.	
Ethernet MAC error.	**Device Error Detect**[1]	UNMACESR[PORT*n*], PM*a*_IEVENT[LOC_FAULT, REM_FAULT, LI_FAULT, TX_OVFL, TX_UNFL, RX_OVFL, MGR_SERR, MGR_AERR]
	Device Error Mask	PM*n*_IMASK[...]
	PCIe Error Report Disable	UNMACECR[PORT*n*]
	Error Capture[2]	PM*a*_RERRn, MAC_MERGE_MMFSECR, MAC_MERGE_MMFAECR
	Target Function(s)	Switch/ENETC w/ Ethernet Link
	Comments Ethernet port MAC error is uncorrectable and non-fatal. The error status register provides a summary of all MAC errors detected by Ethernet links bound to the instance. PCIe reporting can be disabled by the error control register. The event mask and clearing is done through the MAC registers.	
Internal memory multi bit ECC error	**Device Error Detect**[1]	Switch/ENETC: UNMESR0[MBEE] SI: SIUNMESR0[MBEE] Global: UNMESR0[MBEE]
	PCIe Error Report Disable	Switch/ENETC: UNMECR[THRESHOLD/RD] SI: SIUNMECR[THRESHOLD/RD] Global: UNMECR[THRESHOLD/RD]
	Error Capture[2]	Switch/ENETC: UNMESR0[MEM_ID/SYNDROME], UMESR1[ADDR], UNMECTR[COUNT] SI: SIUNMESR0[MEM_ID/SYNDROME], SIUMESR1[ADDR], SIUNMECTR[COUNT] Global: UNMESR0[MEM_ID/SYNDROME], UMESR1[ADDR], UNME[COUNT]

Table continues on the next page...

Table 390. Uncorrectable Non-Fatal Hardware Errors (continued)

Error Description	Error Handling	
	Target Function(s)	Switch, ENETC, ENETC PF (SI0), ENETC VF (SI1+), Global
	Comments Non-fatal multi-bit ECC errors are detected at a function or global level, see Table 392.	

Table 391. Uncorrectable Fatal Hardware Errors

Error Description	Error Handling	
Timer system bus error.	**Device Error Detect**[1]	TUFSBESR[SBE]
	Device Error Report Disable	TUFSBECR[RD]
	Error Capture[2]	TUFSBESR[M/SB_ID]
	Target Function(s)	Timer
	Comments See Table 394 for more information. A Function Level Reset may be required.	
EMDIO system bus error.	**Device Error Detect**[1]	EMDIOUFSBESR[SBE]
	Device Error Report Disable	EMDIOUFSBECR[RD]
	Error Capture[2]	EMDIOUFSBESR[M/SB_ID]
	Target Function(s)	EMDIO
	Comments See Table 394 for more information. A Function Level Reset may be required.	
Switch system bus error.	**Device Error Detect**[1]	UFSBESR[SBE]
	PCIe Error Report Disable	UFSBECR[RD]
	Error Capture[2]	UFSBESR[M/SB_ID]
	Target Function(s)	Switch
	Comments See Table 394 for more information. A Function Level Reset may be required.	
ENETC station interface system bus error.	**Device Error Detect**[1]	SIUFSBESR[SBE], RB*a*SR[SBE], TB*a*SR[SBE]
	PCIe Error Report Disable	SIUFSBECR[RD]
	Error Capture[2]	SIUFSBESR[M/SB_ID], RB*a*CIR[BDR_INDEX], TB*a*CIR[BDR_INDEX]
	Target Function(s)	ENETC PF (SI0), ENETC VF (SI1+)
	Comments	

Table continues on the next page...

Table 391. Uncorrectable Fatal Hardware Errors (continued)

Error Description	Error Handling	
	See Table 394 for more information. A Function Level Reset may be required.	
Internal memory multi bit ECC error	**Device Error Detect**[1]	Switch/ENETC/Global: UFMESR0[MBEE] SI: SIUFMESR0[MBEE]
	PCIe Error Report Disable	Switch/ENETC/Global: UFMECR[RD] SI: SIUFMECR[RD]
	Error Capture[2]	Switch/ENETC/Global: UFMESR0[M/MEM_ID/SYNDROME], UFMESR1[ADDR] SI: SIUFMESR0[M/MEM_ID/SYNDROME], SIUFMESR1[ADDR]
	Target Function(s)	Switch, ENETC, ENETC PF (SI0), ENETC VF (SI1+), Global
	Comments Fatal multi-bit ECC errors are detected at a function or global level, see Table 392. A Function Level Reset or device reset is required.	

The error memory identifier (MEM_ID) is part of the single-bit and multi-bit ECC error reporting and can be found in the CMESR/(UN/UF)MESR0 status registers.

Table 392. Error Memory ID (MEM_ID)

Source ID	Impact Level	Memory Function
0	Global	Ingress Port Filter table memory
1	Global	Receive Flow Steering (RFS) memory
2	Switch/ENETC Function	Frame Header memory
3	Switch Function	Replication and Egress Packet memory
4	Switch/ENETC Function	Queue Descriptor Head memory
5	Switch/ENETC Function	Queue Descriptor memory
6	Switch/ENETC Function	Congestion Group memory
7	Switch/ENETC Function	Class Scheduler memory
8	Switch Function	Egress Count memory
9	Switch Function	Sequence Recovery memory
10	Switch Function	Buffer Pool memory
11	Switch Function	Flow Control memory
12	Switch/ENETC Function	Gate Manager memory
13	Switch/ENETC Function	Host Transfer Agent Thread Context memory
14	Global	AXI Read Box memory
15	Global	AXI Write Box memory
16	Switch/ENETC/SI Function	Prefetcher memory

Table continues on the next page...

Table 392. Error Memory ID (MEM_ID) (continued)

Source ID	Impact Level	Memory Function
17	Global / Function	Common memory. NOTE: End-clients that report global multi-bit ECC errors, using this memory identifier, will report the syndrome as 0; the syndrome identifies the first pertinent bit position (column) in memory that caused the error.

Table 393. Error Block ID (BLOCK_ID)

Block ID	Impact Level	Description
0	Switch / ENETC	Ethernet Interface(s) (includes MAC, pseudo MAC and MAC client)
1	Global	Internal Memory Unit Manager (IMUM) block; manages the shared memory partition in the common memory.
2	Switch / ENETC	Parse Classify Engine (PCE) block; realizes the Ingress Packet Processing (Switch) and Ingress Port Processing (ENETC) functional blocks.
3	Switch / ENETC	Replication Forwarding Engine (RFE); realizes the Replication and Egress Packet Processing functional block.
4	Global	Egress Traffic Management (ETM) block; realizes the Egress Queuing and Traffic Management (Switch), Ingress Congestion Manager (ENETC) and Egress Port Processing (ENETC) functional blocks.
5	ENETC / SI	Host Transfer Agent
6	Switch / ENETC / SI	Prefetcher
7	Global / All functions	MSI-X
8	EMDIO	EMDIO Controller
9	Global / Switch / ENETC	CSI
10	Global	Multi Target Arbiter (MTA)
11	Global	ETM enqueue crossbar
12	Global	IPF TCAM
13	Global	RFS TCAM
14	Global / ENETC / SI	AXI Read
15	Global / ENETC / SI	AXI Write

Table 394. System Bus ID (SB_ID)

Source ID	Agent	Scope and Severity	Description
0	Prefetcher (SI)	Function Fatal	A system bus error occurred during prefetch of a receive BD for the station interface. Frame BD prefetching for the station interface ring is stalled. The frame BD producer index will indicate the last successfully updated BD. Function reset is required.
1	Prefetcher (SI)	Function Fatal	A system bus error occurred during prefetch of a transmit BD for the station interface. Frame BD prefetching for the station interface ring is stalled. The frame BD producer index will indicate the last successfully updated BD. Function reset is required.
2	Prefetcher (SI)	Function Fatal	A system bus error occurred during prefetch of a command BD for the station interface. All command BD prefetching for the station interface is stalled. Function reset is required.
3	HTA receive/ transmit	Function Non-Fatal	A system bus error occurred during a data access (read or write) as a result of processing a command BD for the station interface. Ring writeback status code indicates the nature of the error. Software should clear the error detect events from the relevant error status register (e.g. SIUNSBESR) prior to re-using the command BD ring.
4	HTA receive/ transmit	Function Non-Fatal	A system bus error occurred while writing status information to command BD in memory. The RR bit in the response message indicates the presence of a response command BD entry in the ring. Software should disable the ring and clear the error detect events from the relevant error status register (e.g. SIUNSBESR) prior to re-enabling the command BD ring. This error is primarily useful as a debug aid when the location provided to NETC for the command BD ring is not writeable by the HTA; i.e. expected responses are not received in the ring.
5	HTA receive	Function Non-Fatal	A system bus error occurred while writing status information to receive frame BD in memory. Frame BD prefetching for the station interface ring is stalled until the BD ring is re-enabled. The ring producer index and R bit in the receive writeback BD indicate the presence of a writeback response. RBSR[SBE] is also set and identifies the affected ring. Software should disable the ring and clear the error detect events from the relevant error status register (e.g. SIUNSBESR) prior to re-enabling the BD ring. This error is primarily useful as a debug aid when the location provided to NETC for the receive BD ring is not writeable by the HTA; i.e. expected responses are not received in the ring. This code is returned for the first BD of a frame only. Due to transaction pipelining, write errors on accesses to non-first BDs within a multiple-BD frame are reported as Source ID 6.
6	HTA receive	Function Non-Fatal	A system bus error occurred while writing receive frame data in memory. Frame BD prefetching for the station interface ring will be stalled until the BD ring is re-enabled. RBSR[SBE] is set and identifies the affected ring. The frame BD ring producer index will indicate the last successfully updated BD. BD status code indicates the affected entry. Data for the affected frame is not reliable. Software should disable the ring and clear the error detect events from the relevant error status register (e.g. SIUNSBESR) prior to re-enabling the BD ring. Due to transaction pipelining, write errors on accesses to non-first BDs within a multiple-BD frame will also be reported as Source ID 6.
7	HTA receive	Function Fatal	A system bus error occurred while processing the VSI to PSI message partition copy. The message failed and must be retried by the virtual station interface. Function reset is required.

Table continues on the next page...

Table 394. System Bus ID (SB_ID) (continued)

Source ID	Agent	Scope and Severity	Description
8	HTA transmit	Function Non-Fatal	A system bus error occurred while writing status information to transmit frame BD in memory. The ring consumer index and W bit in the transmit writeback BD indicate the presence of a writeback response. TBSR[SBE] is set and identifies the affected ring. Software should disable the ring and clear the error detect events from the relevant error status register (e.g. SIUNSBESR) prior to re-enabling the BD ring. This error is primarily useful as a debug aid when the location provided to NETC for the transmit BD ring is not writeable by the HTA; i.e. expected responses are not received in the ring.
9	HTA transmit	Function Non-Fatal	A system bus error occurred while reading frame data from memory. Current frame was not transmitted, BD writeback status code identifies the affected frame. TBSR[SBE] is set and identifies the affected ring.
10	Prefetcher (Switch)	Function Fatal	A system bus error occurred during prefetch of a command BD for the switch. All command BD prefetching for the switch is stalled. Function reset is required.
11	Switch	Function Non-Fatal	A system bus error occurred during a data access (read or write) as a result of processing a command/query BD for the switch. Ring writeback status code indicates the nature of the error. Software should clear the error detect events from the relevant error status register (e.g. SIUNSBESR) prior to re-using the command BD ring.
12	Switch	Function Non-Fatal	A system bus error occurred while writing status information to command BD in memory. The RR bit in the response message indicates the presence of a response command BD entry in the ring. Software should disable the ring and clear the error detect events from the relevant error status register (e.g. SIUNSBESR) prior to re-enabling the command BD ring. This error is primarily useful as a debug aid when the location provided to NETC for the command BD ring is not writeable by the Switch; i.e. expected responses are not received in the ring.
13	MSI-X	Function Fatal	A system bus error occurred while writing message signaled interrupt to interrupt controller. Messaged failed to be delivered and interrupt is lost. Function reset is not required, but MSI-X table address(es) may have to be reconfigured.

53.4.2.2.8.3 Error Reaction

Following are the possible error reactions in NETC:

- Correctable errors (ECC) will correct any single event, count the event, and continue processing normally on all functions.
- Uncorrectable non-fatal errors on a frame context will cause the frame to be discarded or aborted (cut-through) and an event count to be updated. Processing continues normally on all functions.
- Uncorrectable fatal errors in a specific function (timer, EMDIO, ENETC, switch) will cause that function to stop processing. The function itself will be disabled, while processing continues normally on all other functions.
  - **Timer fault**
    - PCIe bus master enabled cleared (PCIe PCI_CFH_CMD[BUS_MASTER_EN]=0b) for the PCIe function.
  - **EMDIO fault**
    - PCIe bus master enabled cleared (PCIe PCI_CFH_CMD[BUS_MASTER_EN]=0b) for the PCIe function.
  - **Switch fault**
    - PCIe bus master enabled cleared (PCIe PCI_CFH_CMD[BUS_MASTER_EN]=0b) for the PCIe function.

- Ethernet MAC ports disabled (PM0_COMMAND_CONFIG[RX_EN/TX_EN]=0b).
- Pseudo MAC ports disabled (POR[TXDIS/RXDIS]=1b).

— **ENETC/PSI fault**

- PCIe bus master enabled cleared (PCIe PCI_CFH_CMD[BUS_MASTER_EN]=0b) for the PCIe function. All VSIs bound to the instance are also affected in the same way.
- PSI disabled (SIMR[EN]=0b). All VSIs bound to the instance are also affected in the same way.
- Ethernet/Pseudo MAC port disabled (PM0_COMMAND_CONFIG[RX_EN/TX_EN]=0b / POR[TXDIS/RXDIS]=1b).

— **VSI fault**

- PCIe bus master enabled cleared (PCIe PCI_CFH_CMD[BUS_MASTER_EN]=0b) for the PCIe function.
- VSI disabled (SIMR[EN]=0b).

- Global uncorrectable non-fatal error is most commonly reported by a service block (e.g. Multi-target Arbiter, AXI resources, Prefetcher). These service blocks may return an error indication back to the client who has issued the command, so the client may enter the appropriate safe state. See the description of the safety mechanism for more information. NETC processing continues without interruption.
- Global uncorrectable fatal error on a common structure will cause NETC processing to stop, and all of the NETC functions to enter safe state as described above for each individual function. Processing will come to a stop implying that the error does not get propagated, that processing thread stops in its tracks, but other threads may limp along if not negatively hindered. The disabling of functions will ensure this skid comes to a full stop.

Software must monitor the single bit error recoverable error counts or frame discards/bad frames due to error counts in order to detect errors of a more permanent nature.

53.4.2.2.9 Address Alignment

NETC supports byte alignment for transmit buffer descriptor addresses and 64B alignment for receive buffer descriptor addresses. Transmit and receive buffer descriptor rings need to be 128B aligned. PSI-VSI messages needs to be 64B aligned. Data buffers and messages that cross a 4KB page boundaries may incur a small performance penalty. Apart from these restrictions it is generally recommended to align buffers to 64B whenever possible, to improve performance.

The receive ring mode register, RB*a*MR, provides an additional field for alignment of the frame header in the 1st frame descriptor buffer.

- Setting RB*a*MR[AL]=1 allows for a +2B data (0s) alignment which is written by NETC

The last buffer used will align the tail end of data written to a 64B alignment. The figure below shows the alignment of a frame using the settings described above. Included is an example of how a 60B frame can be made into an optimal 64B aligned data transfer to memory.

NOTE

Additional data written for alignment or extension to optimize the memory write to 64B should not be assumed to be 0 or any other value.

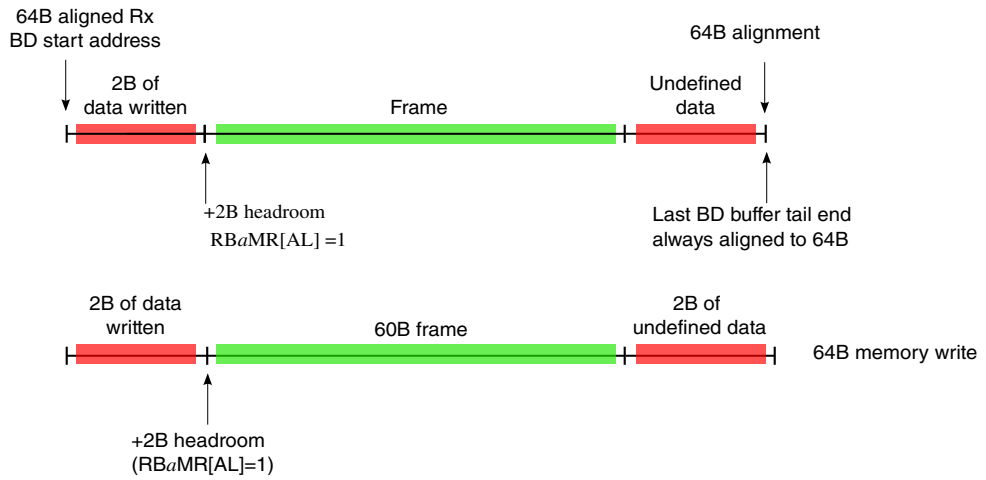


Figure 319. Frame alignment in receive buffer

53.4.2.3 ENETC

53.4.2.3.1 ENETC Datapath

The ENETC datapath consists of the receive datapath blocks, the transmit datapath blocks and a shared internal buffer memory used for internal receive and transmit frame buffering.

Figure 320 shows the functional block diagram of an externally accessible ENETC port (stand-alone ENETC) configuration.

When NETC includes the switch function, ENETC connects to the switch without the need of a back-to-back MAC connection. Instead a light weight back-to-back "pseudo MAC connection" provides the delineation between the switch and ENETC. The back-to-back connection is referred to as a pseudo link. Only frame descriptors are transferred across a pseudo link, resulting in no serialization delay. There is no need to store or copy frames received on a pseudo MAC as they have already been stored in the shared internal buffer memory by the switch. Similarly, there is no need to retrieve or copy frames when sending frames to pseudo MAC. Figure 321 shows the functional block diagram of an ENETC port connected internally to the switch.

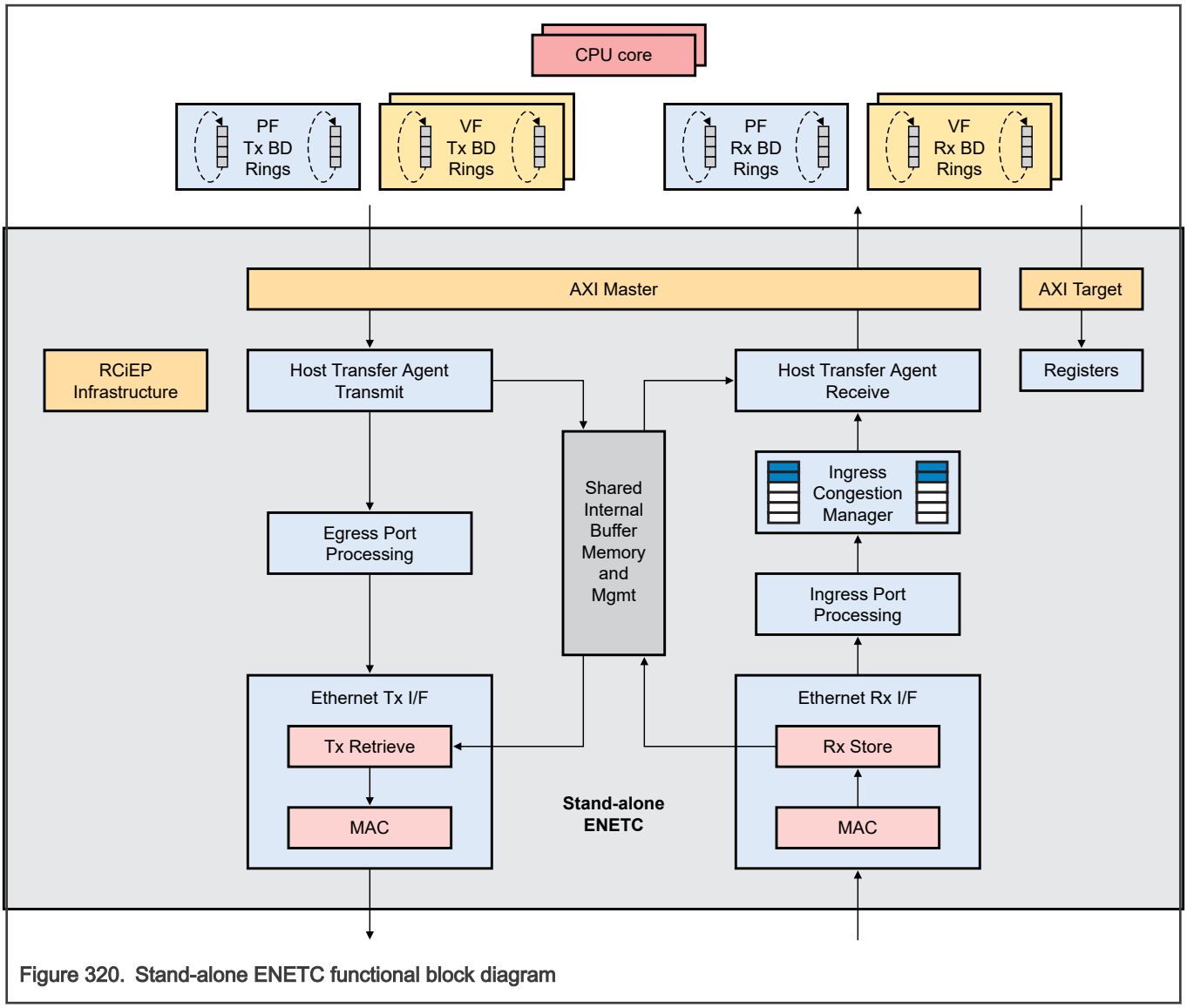


Figure 320. Stand-alone ENETC functional block diagram

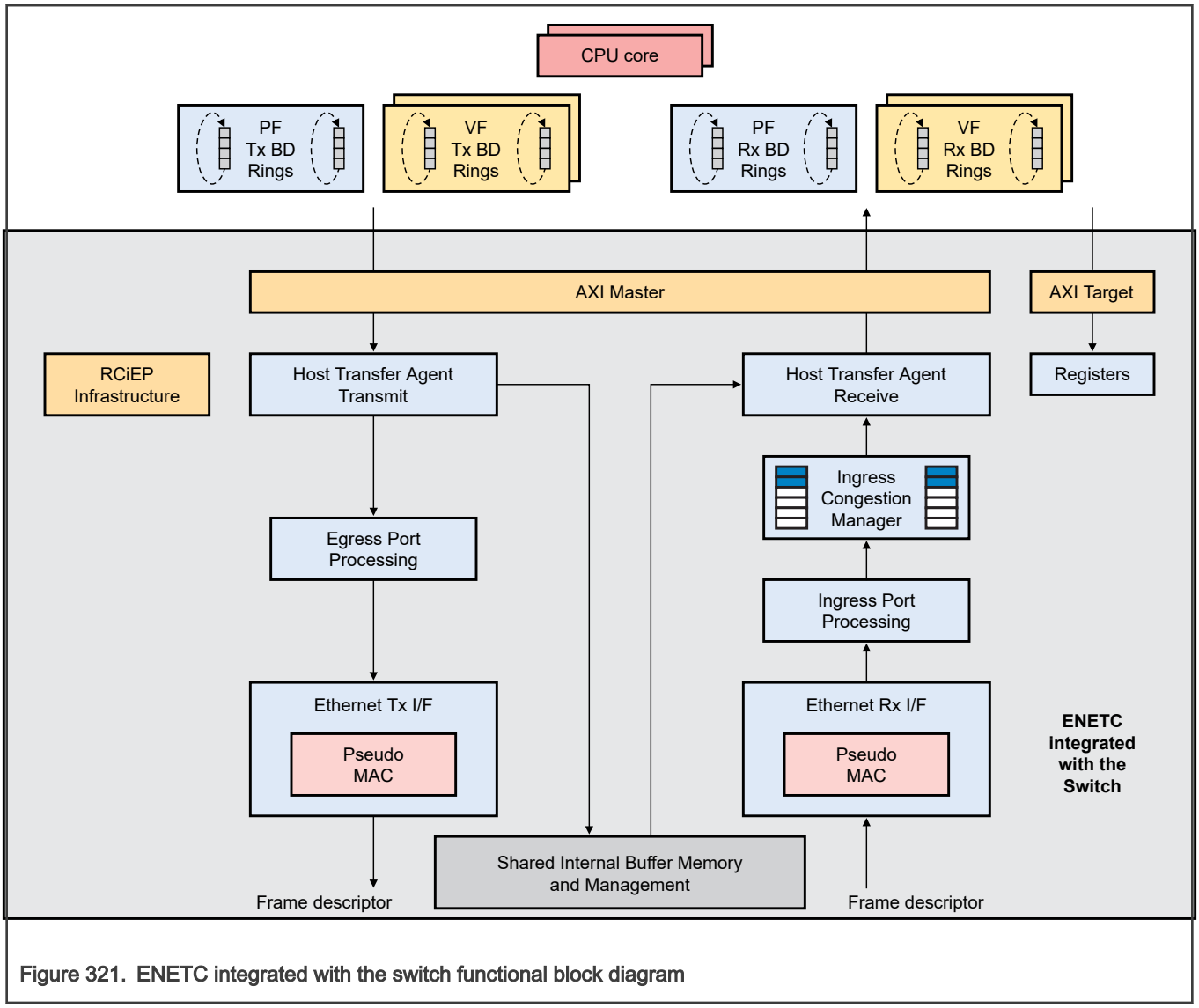


Figure 321. ENETC integrated with the switch functional block diagram

The receive datapath is implemented as a series of functional processing blocks pipelined together. This includes the Ethernet Rx I/F, Ingress Port Processing, Ingress Congestion Manager and the Host Transfer Agent (Receive).

The Ingress Port Processing block implements the ingress parsing, classification (ingress table lookups) and policing functions.

The Ingress Congestion Manager is responsible for implementing the ingress congestion management policies (e.g. drop decisions or back-pressure) when received frames are starting to consume too much of the shared internal buffer memory as a result of downstream resources congestion; e.g. system memory access congestion at the Host Transfer Agent when transferring a frame to the host.

The Host Transfer Agent (receive) implements the receive interface functionality to the host. It includes the necessary DMA engines to efficiently transfer frames from the shared internal buffer memory to host system memory. It also provides multi-receive BD rings support towards the host, for load spreading, virtualization support and QoS.

The transmit datapath is implemented as a series of functional processing blocks pipelined together. This includes the Host Transfer Agent (transmit), Egress Port Processing, and Ethernet Tx I/F.

The Host Transfer Agent (transmit) implements the transmit interface functionality from the host. This includes determining which frame (transmit buffer descriptor(s)) from the transmit BD rings assigned to a port to send next when bandwidth on the outgoing port is available. Once a frame has been selected for transmission, it is loaded from the host system memory into the shared internal buffer memory. The device transmits the frame only after it has completely loaded all frame data from host memory and has performed the required header modifications (i.e. insertion of up to 2 VLAN tags). Note that there is no memory copy when

transferring a frame across a pseudo link (ENETC port connected internally to the switch). Instead a frame descriptor (which includes a pointer to the frame), is passed to the other end of the link (i.e. in this case the switch) when sending a frame across a pseudo link.

The Egress Port Processing block implements the internal transmit queues. The internal transmit queues are used as short staging queues to provide a place to hold frames ready to be transmitted.

ENETC supports multiple Station Interfaces (SIs) for PCIe SR-IOV support (I/O isolation/virtualization) where each SI can be assigned a programmable number of transmit and receive buffer descriptor rings (BD Rings). The SI 0 interface is called the Physical Station Interface (PSI), while SI 1, 2, ...n interface is called Virtual Station Interface n (VSIn). A PSI per port is always present. VSIs are present only if I/O virtualization is supported on the port. A PSI is mapped to a PCIe SR-IOV Physical Function (PF) whereby a VSI is mapped to a PCIe SR-IOV Virtual Function (VF).

A single 64-bit ACE-Lite compliant master interface is used is to move data between ENETC and host system memory. The maximum number of bytes to transfer in a read transaction and an in a write transaction is user configurable. See register SBCR, for more details.

53.4.2.3.1.1 Shared Internal Buffer Memory

As frames are received from the Ethernet MACs, they are stored immediately in the shared internal buffer memory, and remain stored in this memory as they move through the ENETC receive frame processing pipeline up to the point where frames are ready to be transferred into receive data buffers in the host system memory. For frames received from the pseudo MACs, they are already stored in the shared internal buffer memory by the switch and remain stored in the same memory location as they move through the ENETC receive frame processing pipeline; there is no memory copy when receiving a frame from a pseudo MAC. When received on ENETC, non-replicated frames are no longer accounted against switch shared internal buffer memory allotment (and buffer pool), instead their usage then gets accounted against the ENETC receive shared internal buffer memory allotment. For replicated frames, they remain accounted in the switch shared internal buffer memory allotment till the last copy of the frame is transmitted out the switch. There may be frame copies still in ENETC at that point, in which case the buffers of those frames are still in use and still accounted in ENETC (and thus not released), but these buffers are no longer accounted in the switch (buffer pool and switch shared internal buffer memory allotment). Note that from an ENETC congestion management accounting perspective, each replicated frame received from the switch (pseudo MAC), is considered as a separate frame, meaning that the amount of ENETC buffering consumed by each frame reference is equivalent to hold the entire frame.

For transmit frame buffering, a port-per-HTA Transmit priority (2 priorities) byte credit-based flow control mechanism is used to control the amount of internal memory used for transmit buffering. There is a separate byte credit count maintained for each of the two priority tiers within a given port. The ENETC *a* transmit high priority tier byte credit register (E*a*TXHPTBCR) is used to configure the maximum number of byte credits for the high priority tier whereby the ENETC *a* transmit low priority tier byte credit register (E*a*TXLPTBCR) is used to configure the maximum number of byte credits for the low priority tier. Here, *a* indicates the ENETC instance number. The high priority tier is used for traffic directed to the express MAC or pseudo MAC. The low priority tier only applies when IEEE 802.1Qbu frame preemption is enabled, and specifically used for traffic directed to the preemptable MAC. For more details on frame preemption, see Preemption. Note that there are no frame drops when credits are exhausted, instead back-pressure is applied by stopping scheduling host transmit frames.

Once a transmit BD ring is selected for transmission, the frame at the head of the transmit BD ring is loaded from the host system memory into the shared internal buffer memory, and the frame remains stored in the internal memory until it has been transmitted to the Ethernet MAC. For ENETC ports connected internally to the switch, the frame continues to remain in the same memory location as it is transferred to the switch across the pseudo link. Once the frame is received by the switch, its frame buffer usage gets accounted against the switch shared internal buffer memory allotment.

The ENETC shared internal buffer memory partitioning is shown in the following figure:

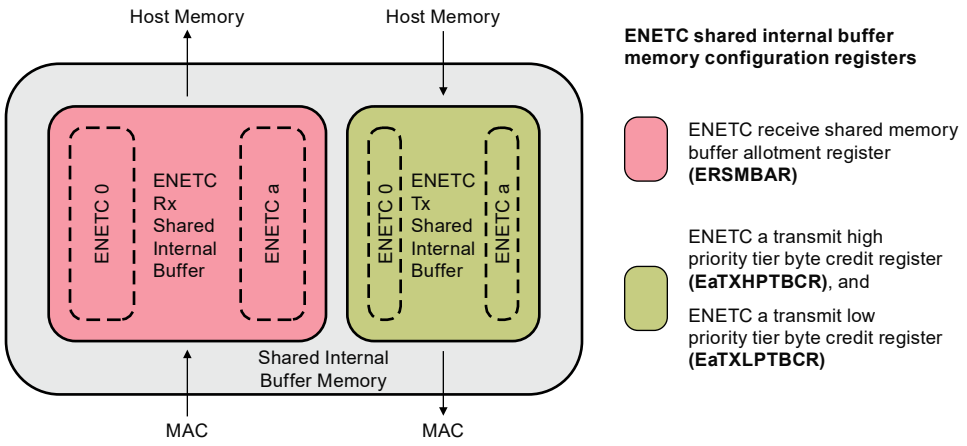


Figure 322. ENETC shared internal buffer memory

53.4.2.3.1.2 Transmit Datapath

The transmit datapath is implemented as a series of functional processing blocks pipelined together. This includes the Host Transfer Agent (transmit), Egress Port Processing, and Ethernet Tx I/F. Details of these functional blocks are given in the following sections.

53.4.2.3.1.2.1 Host Transfer Agent Transmit

The figure below shows the processing through the Host Transfer Agent Transmit.

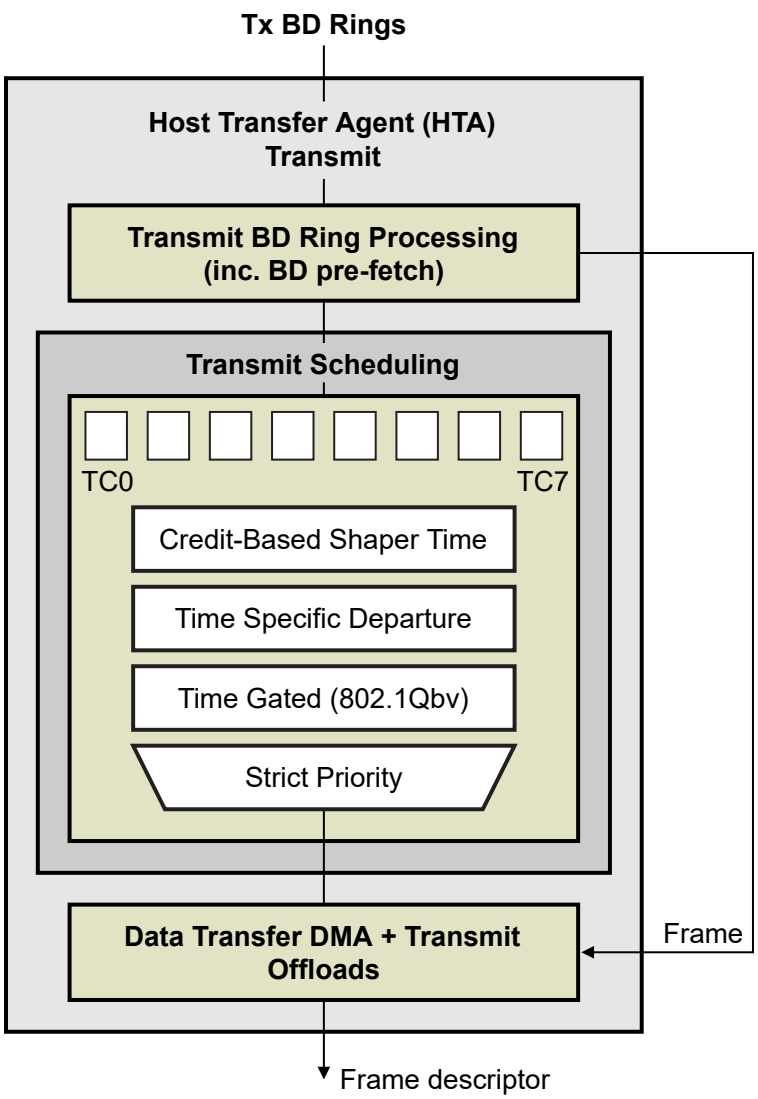


Figure 323. Host Transfer Agent Transmit processing diagram

The Host Transfer Agent (HTA) Transmit block implements the transmit interface functionality from the host. This includes selecting a transmit BD ring from which a frame (transmit buffer descriptor(s)) is to be transmitted when bandwidth on the outgoing port is available. Once a transmit BD ring has been selected, the frame from the head of this transmit BD ring is loaded from the host system memory into the shared internal buffer memory, which in turn serves as a staging memory for getting the frame ready for transmission to the outgoing port.

A Station Interface (SI) will normally be assigned one or more transmit buffer descriptor rings (BDRs) during initialization, these will be numbered 0, 1, 2, …, n, in the SI address map.

The number of transmit rings used per SI depends on the priorities and number of cores that need to be supported. It is allowed to have multiple transmit rings with the same priority if for example an SI services multiple cores. Same priority rings may have different weights.

53.4.2.3.1.2.1.1 Transmit Buffer Descriptor Rings

A Station Interface (SI) will normally be assigned one or more transmit buffer descriptor rings (BDRs) during configuration, see PSI*a*CFGR0[NUM_TX_BDR], these will be numbered 0-*n* in the SI address map. When the SI has been assigned transmit rings, each ring is configured as follows:

1. Configure the transmit ring base address registers TB*a*BAR0/1. The address must be 128 byte aligned.

2. Configure the transmit ring producer index register TB*a*PIR. Software should initialize the index value at this address to 0 to reflect an empty ring. Optionally if the ring is pre-initialized with BDs the index values should reflect the number of BDs added.
3. Configure the transmit ring consumer index register TB*a*CIR. This index value indicates the next BD to be processed by HTA Transmit. When the ring is re-initialized this register should be set to 0. The consumer index is later updated by HTA Transmit during processing of the ring.
4. Configure the transmit ring size TB*a*LENR[LENGTH] to indicate how many 16B entries, are allowed in the ring. Note that when the BD extension bit is set (BD[E]=1), each frame uses two or more entries in the ring. There is no wrap bit to indicate full/empty; one entry is used to differentiate between full and empty. When producer index + 1 equals consumer index, the ring is considered full. When the producer/consumer indices match, the ring is always considered empty.
5. Configure the priority value of a transmit ring via the register TB*a*MR[PRIO]. Also configure the mapping of priority value to traffic class. PRIO2TCMR0 register is used to define the mapping of internal priority value to traffic class. Register PSI*a*CFGR1 (where *a* is the VSI number) is used to remap a VSI traffic class to a different transmit traffic class to restrict use.
6. For an ENETC port connected internally to a switch port, designated as a switch management port, there is the option to direct a frame out a specific switch egress queue. To support this option, a dedicated transmit BD ring must be used, and all frames put on that transmit BD ring will be directed to the specified switch egress queue, subject to the switch egress congestion management policy (possibility of having frames dropped if the egress queue is congested). Only SI 0 can make use of this option, with transmit BD ring number 0 permanently assigned to support this option. This transmit BD ring will not be assigned to a traffic class, meaning that this transmit BD ring will not be considered by the HTA transmit scheduling mechanism. For each frame put on this transmit BD ring, its transmit buffer descriptor (BD) must have the FLQ field set to 10b and the SMSO field set to 1. The output switch port and IPV need also to be specified (in the transmit BD) in order for the switch to derive the switch egress queue. The DR needs also to be specified (in the transmit BD) for congestion management handling within the switch. The switch buffer pool ID will be derived by the switch using the IPV and the switch port number to which the ENETC is internally connected. If the frame is enqueued successfully onto the switch egress queue, hardware may write back a timestamp identifier in the metadata of the transmit buffer descriptor if the transmission time of the frame on the switch port has been requested to be returned to the host. If the frame enqueue is rejected by the switch (due to congestion), an error is reported in the transmit buffer descriptor, indicating that frame wasn't transmitted out the specified switch egress queue due to congestion.
7. Enable the transmit ring TB*a*MR[EN]=1.

53.4.2.3.1.2.1.1.1 Operation of the Transmit Buffer Descriptor Ring

To add one or more buffer descriptors to an enabled transmit ring, follow the procedure below.

1. Software checks to make sure the ring is not full by examining the consumer index (CI) register, TB*a*CIR. Note that maximum number of BDs allowed on a ring is one less than ring size.
2. Software adds one or more BDs to the ring. A memory barrier must be used to guarantee update to memory. If software requires an interrupt when a BD has been transmitted, set BD[FI]=1 and enable the interrupt through TB*a*IER[TXFIE]
3. Software updates producer index register TB*a*PIR to match number of BDs added to the ring. The size of the BD ring index depends on the BD ring size. For 64K-1 entries a 16-bit index value is used.
4. When TB*a*CIR[BDR_INDEX] match TB*a*PIR[BDR_INDEX], the ring is empty. After one or more BDs have been transmitted, the consumer index will be updated by HTA Transmit. The status information is updated in memory for each transmitted BD if required, e.g. timestamp or error information. TB*a*CIR[STAT_ID] is incremented by HTA Transmit if one or more transmitted BDs encountered an error in the last batch of BDs processed. Reading TB*a*CIR[STAT_ID] will reset the field to 0. If software reads a non-zero value for TB*a*CIR[STAT_ID], errors were encountered in the last batch of processed BDs, which can be useful to avoid memory reads with the intent to only examine the status of the transfer.

Table 395. Consumer Index Register (TB*a*CIR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STAT_ID																BDR_INDEX															

5. After one or more BDs have been transmitted, the consumer index as defined by registers TB*a*CIR, will be updated by the device.
    a. Status information is updated in memory for each transmitted BD if required, e.g. timestamp or error information.

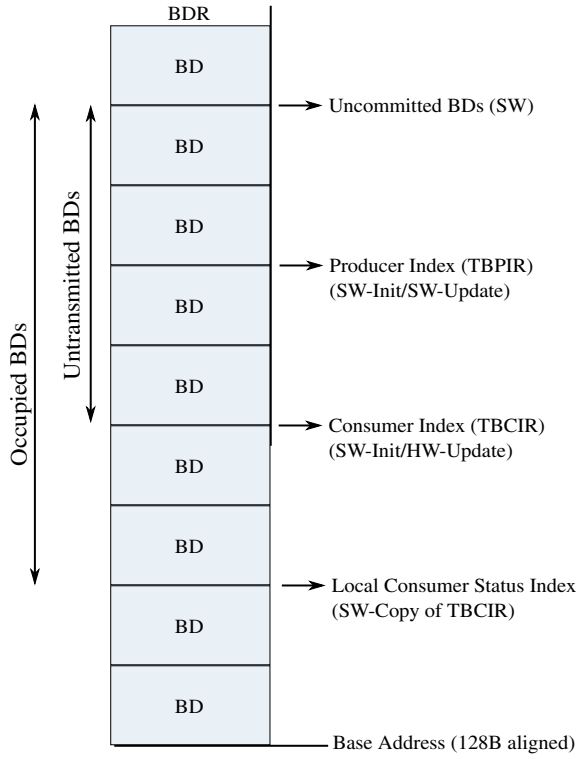


Figure 324. BDR Transmit Components

53.4.2.3.1.2.1.1.2 Transmit Buffer Descriptor Fetching

A pre-fetch mechanism is used to load buffer descriptors from the transmit BD rings in host memory (e.g. DDR) to a cache internal to the pre-fetcher. As BDs are loaded into the internal memory, they are put onto an internal BD queue that is associated with the transmit BD ring from which the BDs were fetched. There is one queue for each possible transmit BD ring. Each internal BD queue can hold up to 8 transmit BDs. When an internal BD queue has 3 or less transmit BDs, a fetch is initiated as soon as there are any transmit BDs ready for transmission in the corresponding transmit BD ring. When an internal BD queue has 4 transmit BDs, the pre-fetch mechanism waits until there are at least 4 transmit BDs in the corresponding transmit ring before loading the next BDs. This results in fetching BDs aligned on a cache line boundary. Transmit BD ring selection for the purpose of fetching BDs is performed in a round robin manner. The BD transfers have higher priority than the frame data transfers on the system memory interface.

53.4.2.3.1.2.1.1.2 Transmit Scheduling

Traffic can be prioritized into a number of traffic classes (up to a maximum of 8). The number of available traffic classes is SoC specific, see Capability Discovery. Traffic classes are numbered from zero to N-1, where N is the number of traffic classes. Numerically higher traffic classes have higher priority. Each transmit buffer descriptor ring (BDR) is assigned a priority, which maps to one of the available traffic classes using register PRIO2TCMR0.

The PSI has the option to remap a VSI traffic class to another traffic class for transmission using register PSI*a*CFGR1 (for a = 1 to 2). This could be a case of restricting the use of certain traffic classes.

As a result, a traffic class may have multiple transmit BD rings assigned to it, where each transmit BD ring either attached to the same SI or different SIs.

Transmit scheduling refers to the procedures used to determine from which of the transmit BD rings assigned to a port to send next when an outgoing port is free.

Each port contains a 3-level hierarchical scheduler.

At the root node (port level), the scheduler selects traffic based on traffic classes according to the following scheduling algorithms:

- Time specific departure (TSD) - Enables the user to specify the time when a frame is to be transmitted. When this capability is enabled, the port will delay the transmission of the frame so that it can be transmitted at the precisely specified time.
- IEEE 802.1Qav - Credit-based shaping (CBS), also referred to as Forwarding and Queueing Enhancements for Time Sensitive Streams (FQTSS). A maximum of 7 traffic classes is supported with credit-based shaping (CBS) enabled. CBS should be enabled on numerically highest traffic class(es).
- IEEE 802.1Qbv - Enhancements for Scheduled Traffic (EST), provides time "slots" for specific classes of traffic in a manner similar to time division multiplexing. EST is also referred in the document as to 'time gate scheduling'.
- Strict priority - Strict priority scheduling amongst eligible traffic classes; i.e. traffic classes with one or more frames available for transmission.

The above algorithms/mechanisms can be used together. Time specific departure and CBS require that IEEE 802.1Qbv be enabled, with or without a gate control list configured.

When preemption is enabled, the express traffic classes tier and the preemptable traffic classes tier are scheduled independently based on the next transmission opportunity from their respective link tier.

When the port is attached to a pseudo MAC, the internal flow control signals from the pseudo link are connected directly to the scheduler. When asserted, the scheduler enters a pause state where it stops scheduling frames for all traffic classes. When de-asserted, the scheduler resumes to normal transmission of frames.

At the next level of the hierarchy, traffic is selected on the basis of SIs where a proprietary weighted fair scheduling algorithm named Weighted Bandwidth Fair Scheduling (WBFS) is used. WBFS provides work conserving fair bandwidth allocation among SIs. The WBFS algorithm is described in Weighted Bandwidth Fair Scheduling (WBFS) Algorithm. Only SIs which have an available frame that map to the selected TC are considered. Weight value for an SI is configured via the PSIaCFGR0[SIBW] register. A higher weight number indicates less bandwidth. SIs assigned with a weight of 0 are treated as strict priority and serviced ahead of SIs assigned with a non-zero weight. SIs assigned with the same weight, are serviced using the WBFS algorithm in which the bandwidth to those SIs is equally shared among them.

At the last level, called leaves, the scheduler selects a transmit BD ring from which a frame (transmit buffer descriptor(s)) is to be transmitted using a combination of strict priority and frame-based weighted round robin algorithms. Only BDRs which have an available frame that map to the selected SI and whose IPV maps to the selected TC, are considered. Transmit BD rings with larger numeric priority value are treated as strict priority and are serviced ahead of transmit BD rings with a smaller numeric priority value. Transmit BD rings with the same priority value are scheduled using a frame-based weighted round robin algorithm. The weight of each transmit BD ring is configured via the TBaMR[WRR] register. The weight corresponds to how many times a transmit BD ring will be selected consecutively. For example, if a transmit BD ring has a weight of 5 it will be selected 5 times before the next transmit BD ring.

IEEE 802.1Qbv and time specific departure (TSD) require that a traffic class to be transmitted at a precisely specific time. For ENETC, transmit scheduling must be performed in advance to ensure there is sufficient time to load the frame from host memory to the shared internal buffer memory prior to the frame departure time (in the case of TSD) or prior to the frame open time gate window (in the case of IEEE 802.1Qbv). For IEEE 802.1Qbv, the dequeue rate is performed at the rate of the link speed (rate configured

on the Ethernet MAC or pseudo MAC (PCR[PSPEED])) instead of pushing data as fast as possible to the Egress Port Processing block until it applies back pressure.

In order to support 802.1Qbv, the scheduling structure supports the concept of ‘transmission gates’. A transmission gate is a construct that connects or disconnects the scheduler (or the transmission selection function) from a traffic class (or Tx BD ring(s)), allowing or preventing the scheduler from selecting frames from that traffic class. A transmission gate has two states: open and closed. Only frames in a traffic class with an open transmission gate are eligible for scheduling.

When 802.1Qbv is disabled (or enabled but no gate control list is operational), the states of the transmission gates are set according to the state values configured in PDGSR[DGS_TC*n*], where ‘*n*’ is the traffic class number. The power-on-reset settings for PDGSR[DGS_TC*n*], are gate open. The PDGSR[DGS_TC*n*] settings can be written at any time and take effect immediately if no gate control list is operational or if 802.1Qbv is disabled.

The figure below shows the HTA transmit scheduling structure when time gate scheduling is enabled, and with 2 traffic classes configured with CBS and TC0 and TC7 configured with multiple SIs and multiple BDRs at the same IPV for each SI.

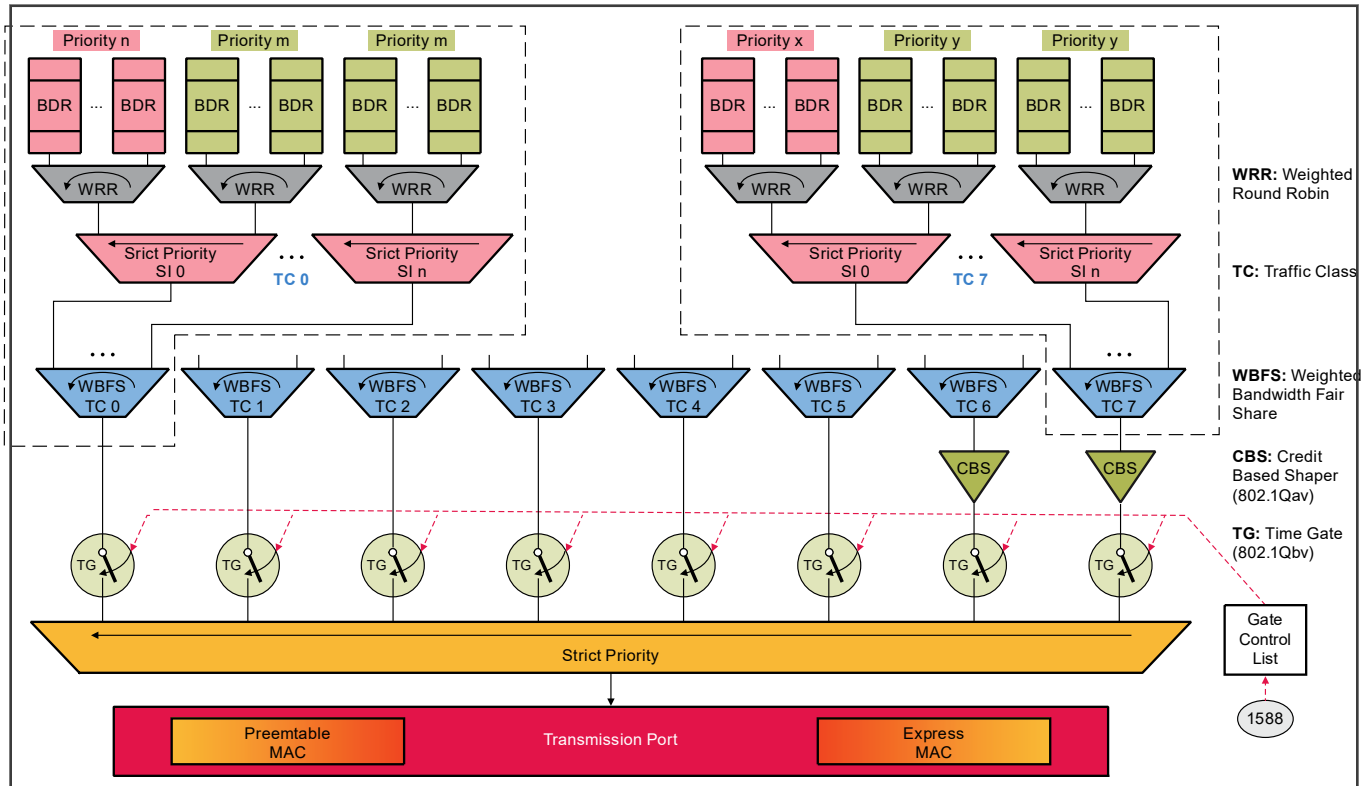


Figure 325. HTA transmit scheduling structure with time gate scheduling enabled

The figure below shows the HTA transmit scheduling structure when time specific departure is enabled.

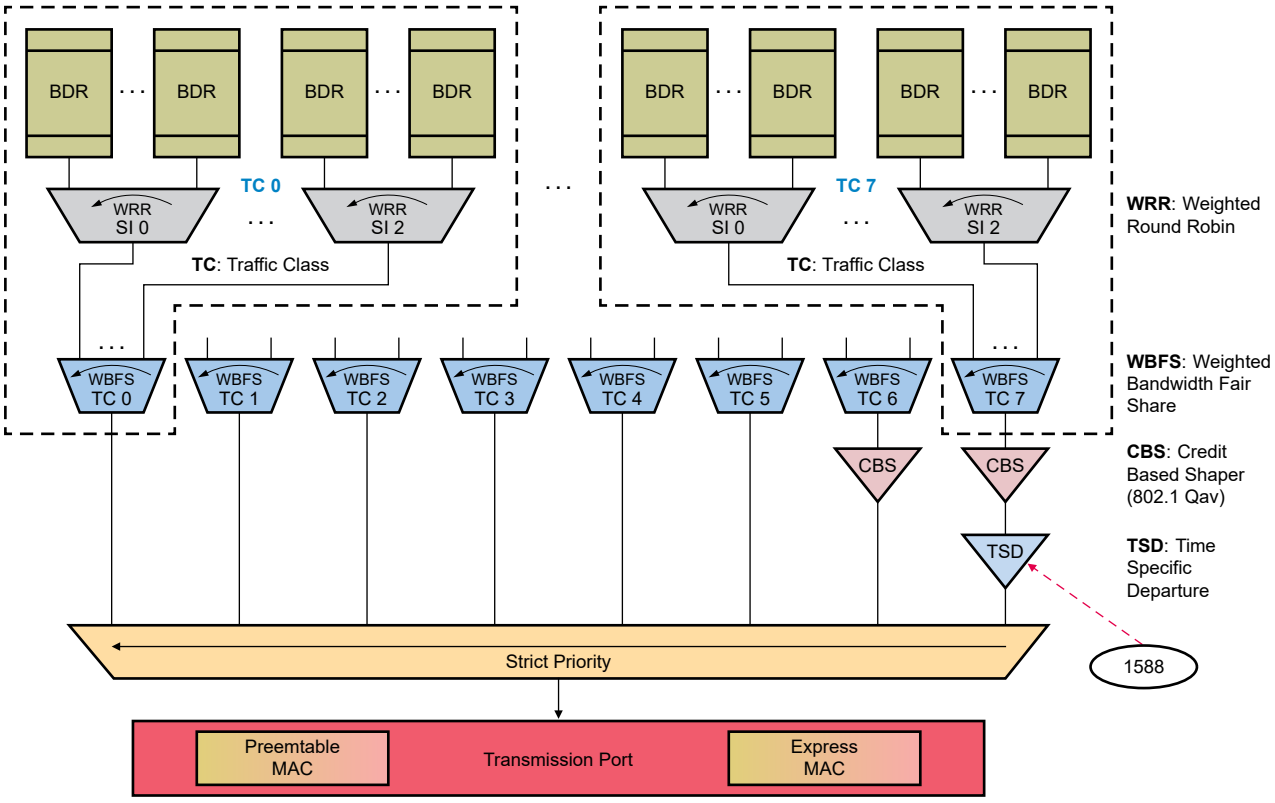


Figure 326. HTA transmit scheduling structure with time specific departure

An HTA transmit thread is dispatched to load the frame from the host memory to the shared internal buffer memory along with executing any required transmit offloads (e.g. MAC anti spoofing, VLAN insert). These transmit threads are executed on the HTA multi-threaded hardware engine(s) which are shared engine(s) between HTA Receive and HTA Transmit for executing hardware threads. Each engine implements a finite state machine (FSM) which operates on a plurality of receive and transmit threads with each thread handling a single frame. One or more engines is instantiated inside the HTA hardware block depending on the requirements e.g. number of ports, port speeds. Multiple transmit threads can be dispatched for a port. More threads can be dispatched than the total number of possible active threads (i.e. the threads currently being handled by the engines) since some threads may need to suspend (e.g. to wait for an earlier larger frame directed to the same internal queue to finish). These in-active or suspended threads therefore do not occupy a thread in an engine.

Once all frame data from host memory has been loaded into the internal memory with all required transmit offloads completed, the frame is sent towards the Egress Port Processing block where it will be queued internally waiting to be transmitted to the MAC.

53.4.2.3.1.2.1.2.1 Credit-Based Shaper - IEEE 802.1Qav

A credit-based shaper is implemented, as per IEEE 802.1Qav (Forwarding and Queueing Enhancements for Time Sensitive Streams (FQTSS)), which is needed for applications requiring bandwidth allocation. The credit-based shaper acts independently, per traffic class, to control the bandwidth distribution between normal traffic and time-sensitive traffic with respect to the total link bandwidth available. In order for the credit-based shaper algorithm to operate as intended, it is necessary for all traffic classes that support the algorithm to be numerically higher than any traffic classes that support the strict priority algorithm.

The Audio Video Bridging (AVB) profile specification (IEEE 802.1BA) states that the sum of AVB Class A and Class B traffic may not exceed 75% of the port transmit rate. The following example uses 70% for AVB frames and 30% for non-AVB frames. AVB class A and B uses one credit-based shaper each, while non AVB classes do not use any credit-based shapers.

Table 396. Bandwidth allocation example for AVB

Class	Percent Bandwidth (total 100%)
AVB Class A	50%
AVB Class B	20%
Non-AVB classes	30%

The following bandwidth availability parameters exist for each port, and for each traffic class that supports the credit-based shaper algorithm.

- *idleSlope*

  The actual bandwidth that is currently reserved for use by the queue(s) associated with a traffic class. Once the frame transmission is complete, *credit* increases back to zero at the rate of *idleSlope*, at which point, a further frame can be selected for transmission. Credits also increase at the rate of idleSlope up to *hiCredit* when there are frames available to send waiting for conflicting traffic to complete. If there is an operational gate control list that is active, then credits are accumulated only while the gate for the credit-based shaper traffic class, is open.

- *sendSlope*

  Credit consumption (per byte) while traffic class is transmitting is calculated as: sendSlope = idleSlope - portTxRate. If a class had 100% bandwidth, sendSlope would be 0, allowing for continuous transfers.

- *hiCredit*

  hiCredit is the maximum number of credits that can be accumulated by the credit-based shaper. The hiCredit is dependent on the maximum interference time for the traffic class.

  For the AVB Class A (highest priority), hiCredit is calculated as follows:

  hiCredit = maxSizedFrame * (idleSlope/portTxRate)

  For the AVB Class B (second highest priority), the maximum interference size is equal to the maximum burst size for the AVB Class A plus the maximum interference size for SR class A (which is equal to the maximum frame size for the port). Maximum burst size for the AVB Class A is calculated as follows:

  maxAVBClassABurst = (RA * M0)/(R0 - RA) + MA (equation (L-16) in IEEE Std 802.1Q-2018), where

  - RA: idleSlope for AVB Class A
  - R0: port transmit rate (specified in PCR[PSPEED])
  - M0: maximum sized frame for the port
  - MA: maximum sized frame for AVB Class A

  hiCredit = (maxAVBClassABurst + maxSizedFrame) * (idleSlope/portTxRate)

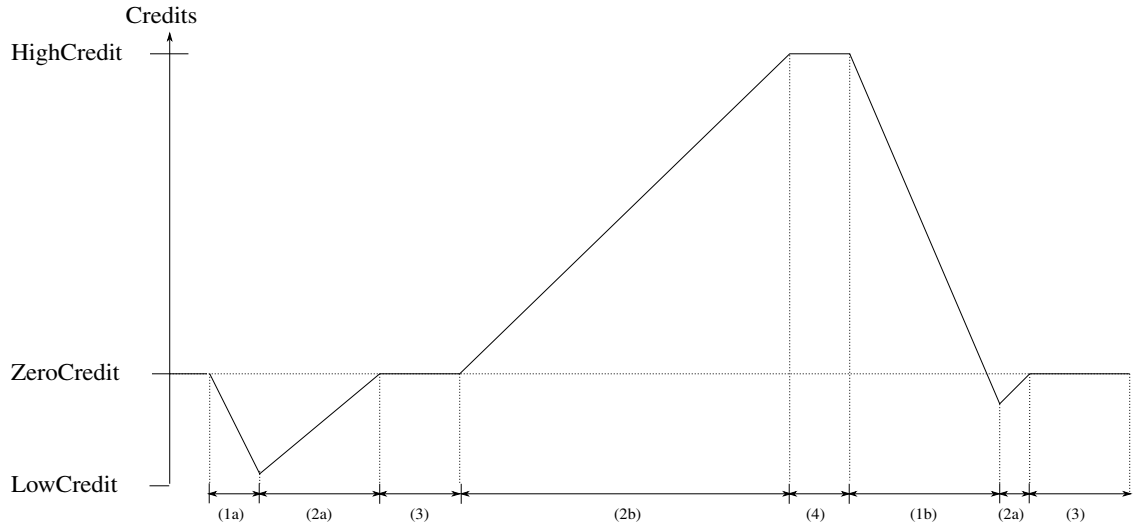
  The maximum sized frame for a traffic class is determines by PTCaTMSDUR[MAXSDU] (where a is the traffic class number)

- *loCredit*

  loCredit is the minimum number of credits that can be accumulated by the credit-based shaper. The loCredit is computed by hardware as follows:

  loCredit = (idleSlope - portTxRate) * maxSizedFrame/portTxRate

The figure below shows the different stages of credit changes for a traffic class as it transmits, waits or is idle.



1a) Send slope. Credits decrement for each byte transmitted assuming BW<100%.
1b) Send slope. Multiple frames may be sent when credits>=0. Bursting would allows class to catch up if conflict occurs.
2a) Idle slope. Increment back to zero credits at the rate of allocated bandwidth for each byte sent.
2b) Idle slope. Class has packet ready for transmit while there are conflicting frames using the port. Allows for >0 credits.
3) No frame ready for transmit.
4) High credit watermark. AVB cannot accumulate more credits.

Figure 327. Stages of credit changes for shaper

To program the credit-based shaper for a class, follow these steps:

1. Enable time gate scheduling for the port by setting PTGSCR[TGE] to 1. An administrative gate control list does not need to be configured, for the credit-based shaper to operate. This also requires the timer function to be configured and enabled. Either the default nanosecond timer or the 1588 timer can be used to generate the required nanosecond resolution time.
2. Enable credit-based shaper for the port traffic class *a* by setting PTC*a*CBSR0[CBSE]=1.
3. Determine *hiCredit* from the formula described above and set PTC*a*CBSR1[HI_CREDIT].
4. Set the allocated bandwidth for the traffic class credit-based shaper which ranges from 1-100 (%), in register PTC*a*CBSR0[BW]. The total bandwidth for all enabled shapers must not exceed 100% or the allotted bandwidth.
5. Set PTXSDUOR register with the appropriate per frame overhead; that is number of bytes to be added to the actual length of each frame, to determine the Physical Layer PDU (PPDU) frame length on the link.

53.4.2.3.1.2.1.2.2 Time Specific Departure

A time specific departure capability is implemented, which enables the user to specify when a frame can be transmitted. When this capability is enabled, the device will delay the transmission of the frame so that it can be transmitted at the precisely specified time.

Time specific departure operations is enabled by performing the following steps:

1. Enable time gate scheduling for the port by setting PTGSCR[TGE] set to 1. An administrative gate control list does not need to be configured, for time specific departure to operate.
2. Enable time specific departure on the traffic class(es) that are to support time specific departure operations, by setting PTC*a*TSDR[TSDE] to 1, where *a* corresponds to the traffic class number.

NOTE

The 1588 timer must be initialized and configured before enabling time specific departure on any traffic class. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer is in normal operation, any change to the 1588 timer configuration (for example, in TMROFF_H/L), except for TMR_ADD updates, requires that time specific departure be disabled on all traffic classes (PTCaTSDR[TSDE]=0).

The departure time of a frame is specified in the transmit BD[TX_START] field. The value of this field is expressed as bit [29:5] of the 1588 synchronized nanosecond time. Since bit 31 and 30 are always 0, the time window is approximately 1 s and then wraps around. As a result, a departure time of up to approximately 0.5 s in the future from the current time, may be specified so that when hardware schedules the frame, it can determine whether the departure time is in the past or in the future. Note that the unit is 32 ns, since bit 0 to 4 are not included. If the departure time in the transmit BD has not yet been reached, based on the current time, the packet will not be transmitted. To indicate the presence of a departure time in the transmit BD, the BD[FL] must be set to b10.

Transmit BD rings transmission selection process is performed in advance to ensure there is sufficient time to load the frame from host memory to internal buffer prior to the departure time of the frame. The time specific departure function relies on the use of the time gate scheduler for executing the transmission selection process ahead of time; that is, 1588 time + look ahead time where the look ahead time is equal to the sum of the time values in the registers EaTGSLR[MIN_LOOKAHEAD] and PTGSATOR[ADV_TIME_OFFSET]. A time specific departure frame becomes available for scheduling when the 1588 time + look ahead time reaches the frame scheduled departure time. See Time Gate Scheduling - IEEE 802.1Qbv, for more details on the operation of the time gate scheduler. Note that the time gate scheduler can operate with or without an administrative gate control list.

Software must guarantee that frames with a departure time in a transmit ring are in increasing order or additional delay may be imposed upon a frame. Packets in the ring are processed in order and the packet at the head of the ring can delay sending of packets later in the ring. Any frame with no departure (or start) time information will be considered eligible to be scheduled, according to its traffic class transmission selection algorithm. To reduce the packet delay variation due to interference from non-time scheduled traffic, consider the following:

1. Time specific departure operation should only be used on the highest priority traffic class. All traffic on that traffic class should use time specific departure operation. Note that within a traffic class, any subsequent frames with or without a departure time, will be held behind if the frame ahead has a departure time that has not been reached.
2. The traffic class being used for time specific departure operation, can use frame preemption by setting MAC_MERGE_MMCSR[ME] to 1 or 2, to enable frame preemption on the port and by assigning the time sensitive traffic class (used for time specific departure) to the express MAC (PFPCR[FPE_TCa] set to 0 (where a corresponds to the traffic class number)) and non time sensitive traffic classes to the preemptable MAC (PFPCR[FPE_TCa] set to 1 (where a corresponds to the traffic class number)). Frame preemption allows interrupt transmission of an ongoing frame. Worst case scenario is the time scheduled frame transfer being delayed by at worst the sum of the minimum final (64B) and non-final (MAC_MERGE_MMCSR[RAFS]) fragment sizes minus four octet times, which cannot be preempted.

53.4.2.3.1.2.1.2.3 Time Gate Scheduling - IEEE 802.1Qbv

IEEE 802.1Qbv, also referred in this document as time gate scheduling, introduces a time based transmission mechanism operating on a global (offline) configured periodic schedule called the gate control list. IEEE 802.1Qbv amendment has been incorporated in the IEEE standard 802.1Q 2018; the mechanism itself is referred in the standard as "enhancements for scheduled traffic."

The mechanism is shown in the diagram below.

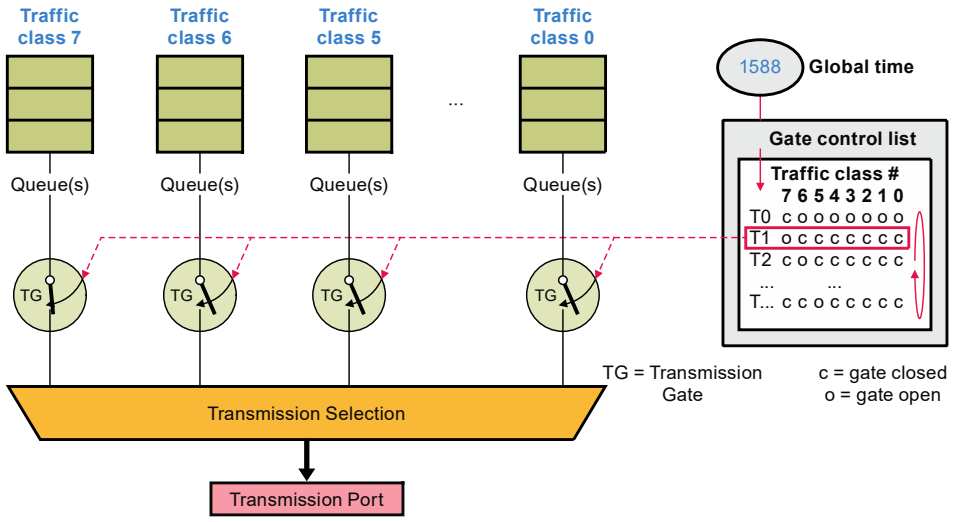


Figure 328. Time Gate Scheduling - IEEE 802.1Qbv

Each traffic class has a gate, which controls if the traffic class is open or closed. Only frames in traffic classes with open gate are eligible for transmission. For the switch, each traffic class has a single queue, and for ENETC, it is possible that a traffic class may have more than one queue (or Tx BD ring). In this case, all queues (or Tx BD rings) in a traffic class have/shared the same gate state.

The gate state in each traffic class is controlled by the gate control list. A gate control list consists of an ordered list of gate control entries, where each entry specifies an 8 bit gate state vector where each bit (one per gate), indicates whether the gate is open or closed. The gate control entry also specifies the time duration for the states.

The gate control lists are typically generated offline, and then timely configured on each network device. To support no-downtime gate control list configuration changes, the standard requires that a network device supports two sets of gate control lists:

- **Operational gate control list** – Represents the currently active gate control list.
- **Administrative gate control list** - Provides a means of configuring a new gate control list prior to its installation in a running system.

The hardware executes the (operational) gate control list cyclically based on a global time. The hardware starts executing the gate control list from the head, and then goes through the gate control entries as time goes on, and after the cycle time is passed, the hardware jumps back to the head of the gate control list.

The transmission selection process (or the scheduler) which executes at the port level, may only select frames from traffic classes that have their gates in the open state. The transmission selection uses a strict priority algorithm to select the next frame for transmission by querying/checking the open traffic classes in descending order from the highest priority traffic class to the lowest priority traffic class. If the open traffic class being queried has one or more frames waiting to be transmitted, then the scheduler determines if the frame at the head of the queue can be transmitted entirely before its traffic class gate closes. If it is determined that there is insufficient time available to transmit the entirety of that frame (transmission of the frame does not fit in the current open window), that frame will remain queued, and an attempt to schedule it will be made in the next open window associated with the frame traffic class. The transmission selection process then moves to the next lowest priority traffic class that has its gate open.

To set up time gate scheduling, follow these steps:

1. The 1588 timer must be initialized and configured before enabling time gate scheduling. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer is in normal operation, any change to the 1588 timer configuration (for example, in TMROFF_H/L), except for TMR_ADD updates, requires that time gate scheduling be disabled ((PTGSCR[TGE]=0). The state of each gate while time gate scheduling is disabled, can be controlled via register PDGSR[DGS_TCn], where n is the traffic class number.
2. Enable time gate scheduling for the port by setting PTGSCR[TGE] to 1.

3. The initial state of each gate, when no gate control list is operational, is determined by the setting in PDGSR[DGS_TCn], where n is the traffic class number. The power-on-reset setting for PDGSR[DGS_TCn] is gate open. The initial state of each gate can be changed by configuring PDGSR[DGS_TCn]. Note that the PDGSR[DGS_TCn] settings can be written at any time and take effect immediately if no gate control list is operational or if time gate scheduling is disabled. Thus, it may be required to re-configure PDGSR[DGS_TCn], before disabling time gate scheduling, if the state of the gates when time gate scheduling is disabled, differ from their initial state when time gate scheduling is enabled.
4. Set the maximum frame size for each traffic class in PTC*a*TMSDUR[MAXSDU]. Frames that exceed the limit are discarded.
5. Configure the (administrative) gate control list through the use of the Time Gate Scheduling table; for more details, see Time Gate Scheduling Table for the ENETC and Time Gate Scheduling Table for the switch. The Time Gate Scheduling table contains time gate scheduling configuration and operational information for a given NETC function. There is one Time Gate Scheduling table for each ENETC instance and one Time Gate Scheduling table for the entire switch. For the switch, there is one entry for each switch port, whereby for an ENETC instance, there is a single entry for the entire ENETC instance. Each entry includes both an administrative gate control list and an operational gate control list, along with the related configuration and status parameters. The gate control list includes the following parameters:
   - Size of the gate control list.
   - Base time (start time) of the schedule. Base time is expressed in units of nanoseconds.
   - Cycle time of gate control list. Cycle time is expressed in units of nanoseconds. The cycle time cannot be shorter than the sum of all gate control list entry durations.
   - Cycle extension time of gate control list. Cycle extension time is expressed in units of nanoseconds.
   - Gate control list.
6. When hardware processes a Time Gate Scheduling table command to configure a new (administrative) gate control list, it calculates the time at which the administrative gate control list is to become active (config change time), based upon the values of the base time and cycle time of the administrative gate control list. When the config change time of the administrative gate control list is reached, the hardware installs the gate control list (the gate control list becomes operational) and starts the execution of the gate control list. This function is performed in advance by having its time reference set to the 1588 time + E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET], to adjust for latency encountered in the transmit processing pipeline. Register PTGSATOR[ADV_TIME_OFFSET] is used to specify the latency across the MAC plus if needed, delays outside of NETC (for example, PHY delay). Register E*a*TGSLR[MIN_LOOKAHEAD] is used only for ENETC, to specify the latency encountered for scheduling, dequeuing, and loading frames from the host memory to NETC internal memory. Register S0TGSLR[MIN_LOOKAHEAD] is used only for the switch, to specify the latency encountered for scheduling and dequeuing frames. Note that E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] are IERB registers, and thus can only be setup at pre-boot initialization time.
7. Similarly, time gate scheduling of all frames is also done in the future to adjust for latency encountered in the transmit processing pipeline. As in the previous step, this is achieved by advancing the 1588 time reference by the amount of time (*LookAheadTime*) configured in E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET].
8. Set PTXSDUOR register with the appropriate per frame overhead; that is number of bytes to be added to the actual length of each frame, to determine the PPDU (Physical Layer PDU) frame length on the link.

Time Gate Scheduling Operation

The setting of the transmission gate state for the gate associated with each of the port's traffic classes and the execution of transmission selection function are performed in advance to mitigate against any long latency (e.g. DMA latency in the case of ENETC) and latency variation encountered in the transmit processing pipeline. This in turn leads to better timing accuracy and reduction in jitter. The time used to perform these function, is set to the 1588 time + E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]. Since the transmission selection function is performed in advance, it no longer relies on the MAC/link being idle for the indication that there is a transmission opportunity to transmit a frame. Instead the link is modeled by the transmission selection function to determine when frames are selected for transmission. Once the transmit transmission function selects a frame for transmission, next transmission opportunity provided by the link model, will occur after the time interval corresponding to the transmission time of the current selected frame at the rate configured in PCR[SPEED], with physical frame

overhead (IFG, preamble, SFD) accounted. The frame overhead is specified in PTXSDUOR. Time remaining for the transmission of a frame is tracked on every clock cycle. Note that the modeling of link also models the preemption function if enabled (such as stopping transmission of preemptible frame with added overhead of non-final fragments (mCRC, IFG, preamble, SFD) being accounted for, in the link model).

Each frame selected for transmission is tagged with a timestamp corresponding to the advance time that the time gate scheduler is operating on (1588 time + E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]). The timestamp of a frame determines its departure time, that is the time at which the frame should be transmitted to the link. Then at the point where frames are transferred to the MAC, a timestamp-based shaper is implemented where the frames are released only when their transmission time (timestamp) is reached. The timestamp-based shaper function can also be performed in advance by having the time used to perform this function advanced by the amount of time specified in PTGSATOR[ADV_TIME_OFFSET], to adjust for the latency encountered through the Ethernet interface logic (MAC + physical layer latency).

Time gate scheduling proceeds as follows when there is transmission opportunity on the link to transmit a frame. It first selects the highest priority traffic class for which there is a frame waiting to be transmitted at the head of the traffic class. It then determines if that frame can be transmitted. A frame for a given traffic class can only be transmitted if the transmission gate associated with that traffic class is open and there is sufficient time for the entire frame (length of the frame is known) to be transmitted before the next gate-closed operation associated with that gate is executed. If both conditions are not met, then the next highest priority traffic class for which there is a frame waiting to be transmitted at the head of the traffic class, is selected to determine if that frame can be transmitted based on the criteria described above.

To determine if a frame can be transmitted, the scheduler first checks if there is sufficient time left in the current gate control entry interval, to transmit the entirety of the frame. If there is insufficient time, then the pre-processed gate control list data is accessed to extract how many bytes (up to the maximum SDU length) that can be transmitted in the next gate operations for this traffic class. This gate control list pre-processing, which is performed when the gate control list is configured, consists of performing the following computation for each entry in the gate control list.

- For each traffic class that has its gate open, the next gate operations in the gate control list (next gate control entries) are inspected to accumulate the number of bytes that can be transmitted in each gate control entry, until the accumulated number of bytes reaches the configured maximum SDU frame size, or a gate-closed operation is found.
- If the end of the gate control list is reached and the gate control cycle time is longer than the total execution time of the sequence of gate operations, the last gate control entry is extended to the end of the gate control list cycle. Once the end of the gate control list cycle is reached (no gate-closed operation found), and the accumulated the number of bytes that can be transmitted has not reached the size of a maximum SDU frame, inspection continues from the beginning of the same gate control list.
- This accumulated number of bytes that can be transmitted, is then stored in the gate control list entry, to be used later by the scheduler to determine for a given traffic class, how many bytes that can be transmitted in the next gate operations. This pre-processed data avoids having the scheduler at run-time, examining the next entries in the gate control list.

If it is determined that there is insufficient time available to transmit the entirety of that frame (transmission of the frame does not fit in the current open window), that frame will remain queued, and an attempt to schedule it will be made in the next open window associated with the frame traffic class.

In the case where the existing gate control list cycle is extended so that it ends at the base time of the new gate control list to be installed, any inspection from previous gate control entries will not continue into this extension time interval. Frame scheduling during the extension time internal is permitted with the gate states set to the values of the last gate control entry of the existing gate control list.

If there is a new gate control list scheduled to be installed in the next gate control list cycle, the scheduler cannot take into consideration the new configuration as the scheduler is using pre-processed gate control list data to determine how many bytes that can be transmitted in the next gate operations. Furthermore, any of the next gate operations accumulated byte fields in the current operational gate control list, that resulted from inspecting back to the beginning of the gate control list, are no longer accurate. As a result, all of the next gate operations accumulated byte fields in the last cycle are overwritten to 0.

Note that once the time gate scheduler has cycled through the entire operational gate control list and has detected a condition where a frame at the head of a traffic class will never be able to be transmitted, that frame will be discarded in the case of switch. These frame discards are counted against the port's Tx discard count register (PTXDCR) along with the setting of the Time Gate Scheduling Frame Too Large Discard Reason (TGSFTLDR) flag to 1 in the port's Tx discard count reason register 0 (PTXDCRR0).

In the case of ENETC, the frame is not transmitted and the STATUS field of the transmit buffer descriptor is updated with the error code 0x080 (frame too large for time gating window). The conditions where a frame at the head of a traffic class will never be able to be transmitted are:

- A frame at the head of a traffic class, although shorter than the value configured in PTC*a*TMSDUR[MAXSDU] (where *a* is the traffic class number), cannot be transmitted because there is no time interval between any gate-open event and a subsequent gate-close event, sufficient, long enough to transmit the entirety of the frame.
- There are no gate-open events for that traffic class in the entire operational gate control list; its gate is closed in every entry of the operational gate control list.

In the absence of any gate control list configuration, the time gate scheduler if enabled, still executes the transmission selection function ahead of time based on the setting of E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD], E*a*TGSLR[ZERO_LOOKAHEAD] and PTGSATOR[ADV_TIME_OFFSET].

Gate Control List Installation

The time when a new gate control list takes affect is calculated based on the rules described below (where *LookAheadTime* = E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET])

If the operational gate control list is not active, PTGAGLSR[TG]=0, the time (i.e. *ConfigChangeTime*) at which the next gate control list configuration change is scheduled to occur is determined as follows (where *CurrentTime* is at the 1588 time reference):

- if *AdminBaseTime* >= *CurrentTime* + *LookAheadTime*, then *ConfigChangeTime* = *AdminBaseTime*
- if *AdminBaseTime* < *CurrentTime* + *LookAheadTime*, then *ConfigChangeTime* = (*AdminBaseTime* + N**AdminCycleTime*) where N is the smallest integer for which the relation (*ConfigChangeTime* >= *CurrentTime* + *LookAheadTime*) would be TRUE.

If the operational gate control list is active, PTGAGLSR[TG]=1, the time (i.e. *ConfigChangeTime*) at which the next gate control list configuration change is scheduled to occur is determined as follows (where the end of the current operational cycle (*EndCurrentCycle*), is the absolute end time of the currently operational cycle without the cycle extension time, at the time reference of 1588 time + EaTGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]):

- if *AdminBaseTime* < *EndCurrentCycle* + *OperCycleTime*, then *ConfigChangeTime* = *AdminBaseTime* + N**AdminCycleTime*) where N is the smallest integer for which the relation (*ConfigChangeTime* >= *EndCurrentCycle* + *OperCycleTime*) would be TRUE.
- if *AdminBaseTime* >= *EndCurrentCycle* + *OperCycleTime*, then *ConfigChangeTime* = *AdminBaseTime*

The *ConfigChangeTime* can be retrieved by issuing a query command against the Time Gate Scheduling table entry associated with the port.

When the administrative gate control is about to be installed, and operational gate control list is not active, PTGAGLSR[TG]=0, then *CycleStartTime* is set to *ConfigChangeTime*. If the operational gate control list is active, PTGAGLSR[TG]=1, *CycleStartTime*, is determined according to the following rules:

- If *ConfigChangeTime* > *EndCurrentCycle* + *OperCycleTimeExtension*, then *CycleStartTime* = *EndCurrentCycle*
- If *ConfigChangeTime* <= *EndCurrentCycle* + *OperCycleTimeExtension*, then *CycleStartTime* = *ConfigChangeTime*

If a new gate control list configuration is installed and starts to execute in the current cycle, the current cycle is truncated in order to start the first new cycle and any inspection in the old cycle would not continue into the new cycle in order to start the first new cycle. This could result for the time interval of the last entry of the old gate control list to be very short. This is turn could cause overrunning the next gate-close event, or cause not being able to transmit a frame because the time interval was shortened, and that frame could then get discarded if it had been waiting to be transmitted for a full cycle interval.

Time Gate Scheduling Example

Time gating may be better understood with an example gate control list showing how scheduling will be performed. The example below, Table 397, shows a non-overlapping open gate case, where two traffic classes are never allowed to transmit at the same time. The interval normally defines a multiple of time ticks, in this example assume that the interval and number of bytes allowed to be transferred are the same. The number of bytes per interval would normally be calculated based on the interval length, transmit rate, and time tick granularity.

The total length of the gate control list is 3000B when adding up each individual time interval. The (admin) cycle time provided must be 3000B or greater. If the cycle time is 3000B, then when entry 15 completes, entry 0 would start immediately. Additional time added to the cycle time will extend the entry 15 interval. Assume here that the cycle time is set to 3100.

Table 397. Example: Non-Overlapping Gate Control List

Entry	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Interval (bytes)	100	200	100	300	100	200	100	300	500	100	200	200	300	100	100	100
TC 0	0	0	**1**	**1**	0	0	0	0	**1**	**1**	0	0	0	0	0	0
TC 3	0	0	0	0	0	**1**	0	0	0	0	0	**1**	0	0	0	0
TC 7	**1**	**1**	0	0	0	0	**1**	**1**	0	0	0	0	**1**	**1**	**1**	**1**

In this example, traffic class 7 gate has an open window (gate in the open state) of 1000 bytes starting at the beginning of the 4th last entry and ending at the end of the 2nd entry of the next cycle (if no new gate control list is to be installed).

Lets look at a first scheduling example where two frames are about to be scheduled. Both frames are available at least *LookAheadTime* before they are required to be transmitted in the open time gate window (Interval 1). This time is needed for the device to fetch the frame from memory and prepare it for transmission.

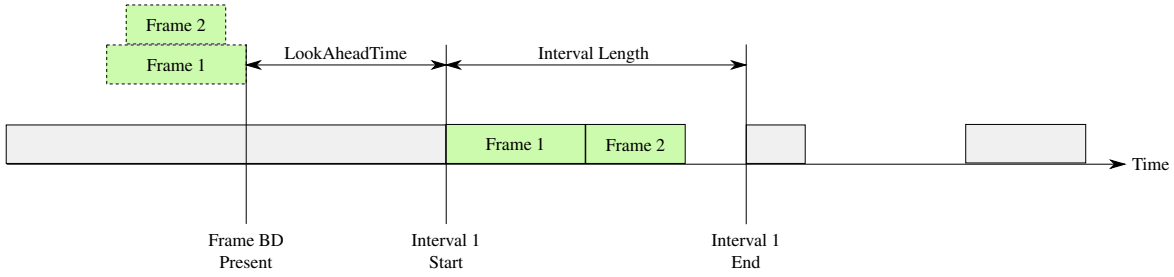


Figure 329. Scheduling of Frames

In this next example, consider the case where a best-effort first frame arrives "late" for scheduling within the next open gate window. Since the scheduler is always looking at LookAheadTime in the future it will calculate the amount of time left in a window at LookAheadTime from the current time to the end of the window. If the frame can still fit it will be scheduled for that window.

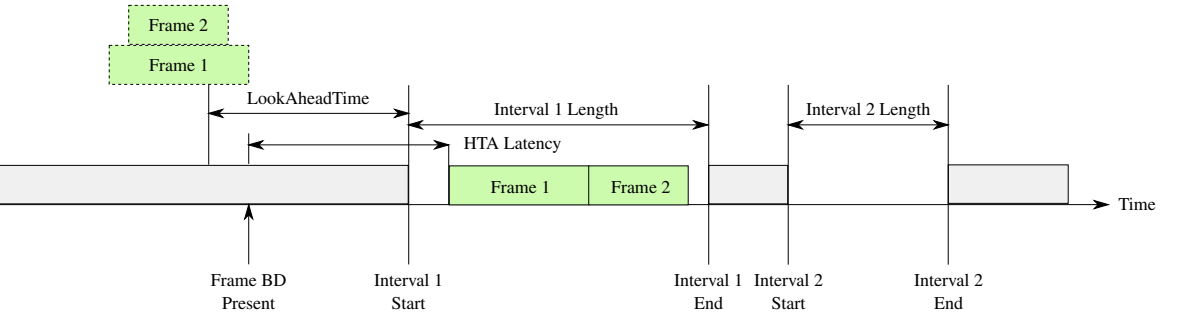


Figure 330. Scheduling of Frames - Late Arrival

Time Gate Scheduling in Combination with Preemption

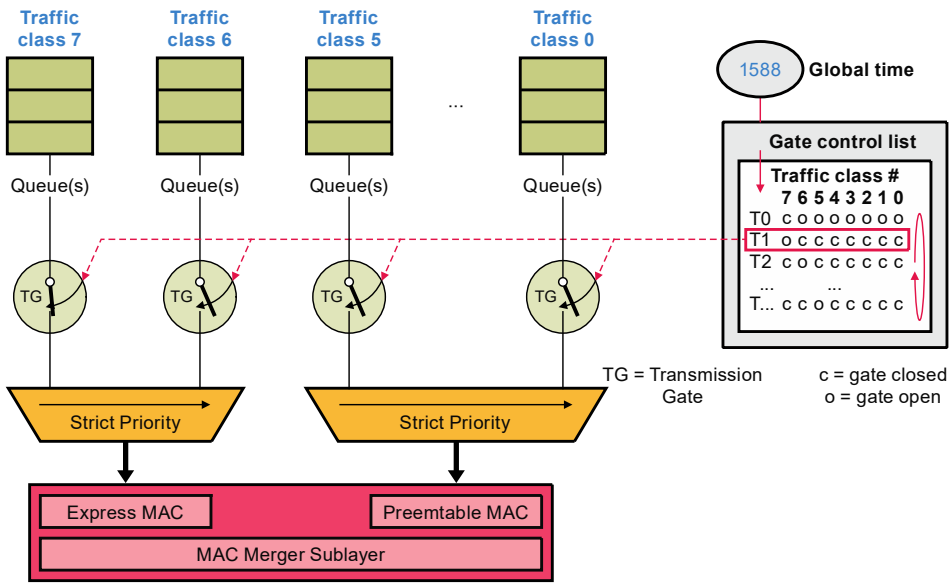


Figure 331. Time gate scheduling in combination with frame preemption

When time gate scheduling is used in combination with frame preemption, the "Hold and Release" mechanism can be used to protect the express traffic class window from interference from preemptable traffic, while at the same time minimizing the impact of the protected window on the amount of bandwidth that is available to preemptable traffic. The "Set-And-Hold-MAC" and "Set-And-Release-MAC" gate operations are used to specify the start and the end of a protected window.

When the time gate scheduler is processing a "Set-And-Hold-MAC" gate operation, asserting the Hold signal to the MAC merge sublayer is not always performed. Specifically, at the point of time where an Hold signal needs to be asserted (start of the hold advance interval), the time gate scheduler will not assert an Hold signal to the MAC merge sublayer under the following conditions:

- The preemptible frame transmission will complete its transmission before the Set-And-Hold-MAC time slot begins.
- There is no preemptible frame transmission.

The time gate scheduler will not schedule a preemptible frame during the Set-And-Hold-MAC time slot, regardless of whether or not an Hold signal was asserted. Furthermore, the time gate scheduler will not schedule a preemptible frame during the hold advance interval, if the preemptible frame transmission would continue into the Set-And-Hold-MAC time slot.

Delaying the scheduling of preemptable traffic classes to the start of the "Set-And-Release-MAC" time slot, has the benefit of selecting a newly arrived higher-priority preemptible frame, instead of the lower priority preemptible frames that were hold off by the scheduler prior and during the "Set-And-Hold-MAC" time slot. Note that once the time gate scheduling decides to not schedule a preemptible frame, any subsequent preemptible frames received will not be scheduled till the "Set-And-Release-MAC" time slot begins even if they there were of higher-priority.

In all other cases, the time gate scheduler will assert the Hold signal, in advance to the Set-And-Hold-MAC time slot by the time interval corresponding to the transmission of the worst case (longest) fragment length that cannot be preempted plus associated physical layer overheads. IEEE Std 802.3-2018 Clause 99 defines a minimum final fragment size of 64 bytes and allows the minimum non-final fragment size to be 64, 128, 192, or 256 bytes. For example, with a non-final fragment size of 64 bytes, a fragment that is 123 or fewer bytes in length cannot be preempted, because the non-final fragment would be less than 64 bytes (60 bytes + 4 bytes for mCRC). The minimum non-final fragment size is configured in MAC_MERGE_MMCSR[RAFS].

Note that time gate scheduler operates at the time reference of 1588 time + E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET] and assumes that there is no delay from asserting the Hold/Release signal to the MAC merge

sublayer and the MAC merge sublayer ceasing/resuming to transmit any preemptible frame. Since it operates in advance, instead of immediately asserting the Hold/Release signal, an Hold/Release message is generated, and tagged with a timestamp set to the advance time (1588 time + E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]), at which the Hold/Release signal must be asserted assuming no internal pipeline delay from the point where frames are transferred to the MAC to the point where frame preemption occurs. Then at the point where frames are transferred to the MAC, a timestamp-based shaper is implemented where the Hold/Release messages are released, which trigger immediate assertion of the Hold/Release signal to the MAC merge sublayer when their time (timestamp) is reached. This timestamp-based shaper function is performed in advance by the amount of time specified in PTGSHCR [HOLD_SKEW] (that is, 1588 time + PTGSHCR [HOLD_SKEW]), to adjust for the Hold/Release latency encountered through the Ethernet interface logic (Hold Release delay to the MAC merge sublayer + any external delay (for example, PHY delay)).

Take for example, where MAC_MERGE_MMCSR[RAFS] is set to 64 bytes, meaning that the minimum fragment size that can be preempted is 124 bytes. If the time gate schedule determines that the Hold must be asserted, the time for asserting the hold would be set to 163 bytes transmission time prior to the start of the Set-And-Hold-MAC time slot; 20 bytes (IFG, preamble, SFD) + 123 bytes + 20 bytes (IFG, preamble, SFD).

An increased IFG length and any other overhead increases in frame transmission time are accounted for by setting the PTXSDUOR register appropriately.

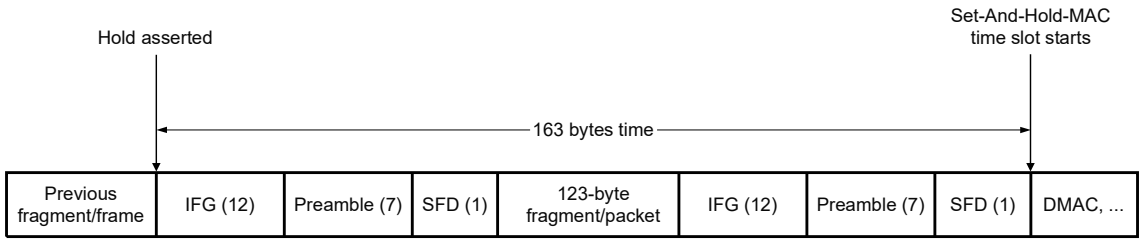


Figure 332. Hold Assertion Timing

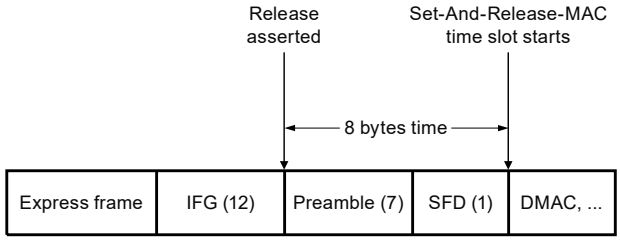


Figure 333. Release Assertion Timing

In the case where the "Hold and Release" mechanism is not used to protect the express traffic class window, and a preemptible frame has been scheduled and its transmission cannot be stopped by the time the express traffic gate opens, the scheduling of the express traffic class is delayed up to the point where the transmission of the preemptible frame has ceased.

If the schedule that applies to the preemptible traffic classes is constructed to disallow preemptible traffic to be transmitted at certain time (that is, there are "gate close" events for the preemptible traffic classes), then there is no way of telling in advance how many times a given frame will be preempted before its transmission is complete, and thus this configuration may result in overrunning the next gate-close event for the preemptible traffic classes concerned.

Note that the when gate control list preprocessing is inspecting the next gate operations in the gate control list (next gate control entries) for a preemptible traffic class, and encounters a gate control list entry that contains an express traffic class with its gate open, the accumulated number of bytes that can be transmitted is set to the configured maximum SDU frame size for the preemptible traffic class being inspected.

Time Gate Scheduling in Combination with Half Duplex Transmission

When time gate scheduling (802.1Qbv) is used in combination with half duplex transmission, the time gate scheduler compensates for deferred transmissions. Accurate transmission scheduling (that is, transmitting only within open gate time intervals) will be recovered for future frames in the event that one or more scheduled frame transmissions are deferred. These deferred frame(s) may violate their classes' open gate times as a result of the deferral, but it will not create a frame scheduling backlog that results in persistent gate violations once the deferred transmission has completed.

The time gate scheduler does not compensate for transmission back off delays due to collisions. A back off delay that follows a collision event may result in gate violations by the re-transmitted frames. In addition, collision back off delays will result in persistently late frame transmissions (potential gate violations) so long as there are transmit frames available for selection by the time gate scheduler; that is as long as transmit traffic is continuously flowing.

The time gate scheduler will recover accurate transmission scheduling during any idle time periods when there are no transmit frames available for selection. Once the sum of the idle time periods equals the total collision delays encountered, accurate transmission scheduling is reestablished.

53.4.2.3.1.2.1.2.4 Transmit Scheduling Timing

The figure below shows the moments in time related to current time a frame has been scheduled for IEEE 802.1Qbv. This is to emphasize the role of EaTGSLR[MIN_LOOKAHEAD], PTGSATOR[ADV_TIME_OFFSET] and PTGSHCR[HOLD_SKEW] values.

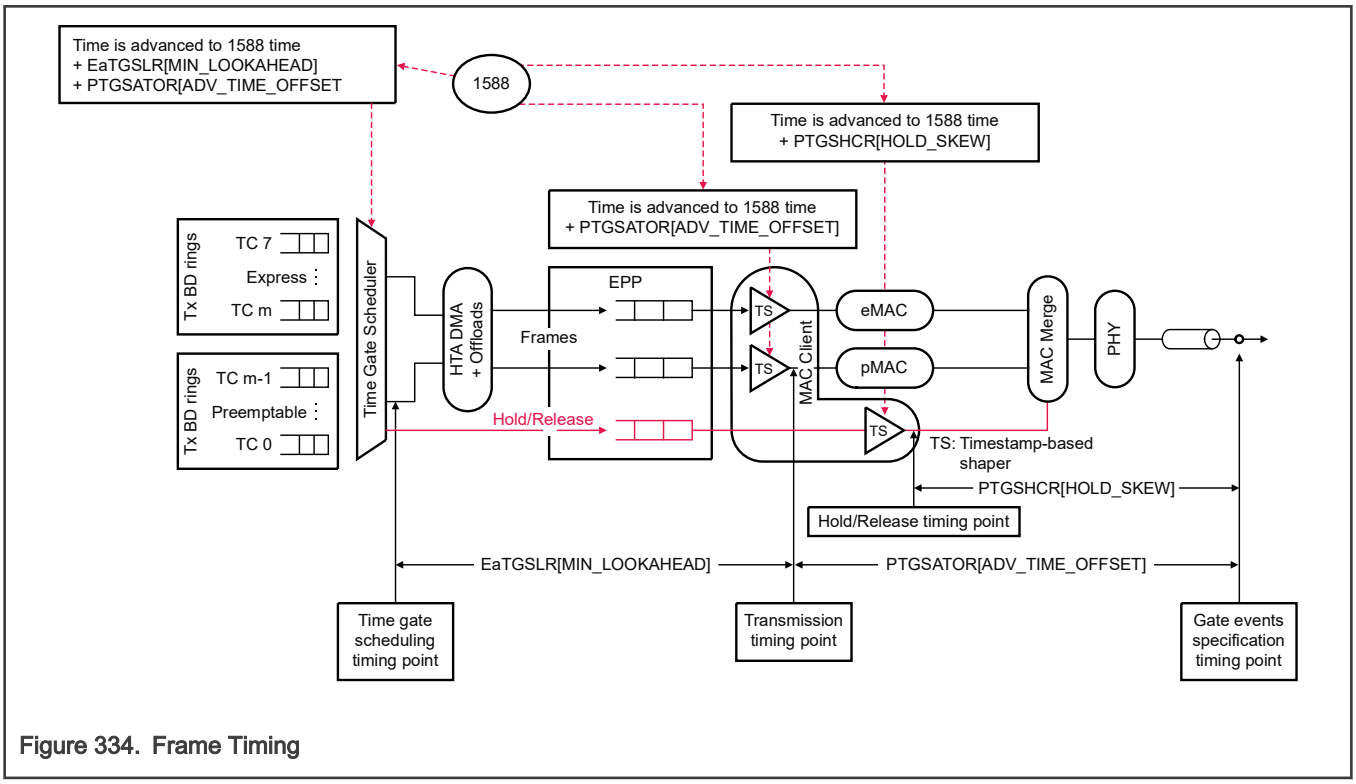


Figure 334. Frame Timing

For time gate scheduling, a frame that appears at the time gate schedule timing point (reference point at which the time of the gate events are specified (gate control list), would have been scheduled at 1588 time - EaTGSLR[MIN_LOOKAHEAD] - PTGSATOR[ADV_TIME_OFFSET] inside the HTA. After which the frame would then have been assigned a departure time and DMA'd to the Egress Port Processing (EPP). The EPP will then have forwarded the frame to the MAC client which in turn will sent to the MAC at 1588 time - PTGSATOR[ADV_TIME_OFFSET] . The added latency through the MAC will have caused the frame to be sent out of ENETC at 1588 time. For example, with a EaTGSLR[MIN_LOOKAHEAD] time of 800 ns, PTGSATOR[ADV_TIME_OFFSET] of 200 ns, and a gate opening of 50000 ns the HTA will schedule the first frame of the gate at 49000 ns. The frame will then be forwarded to the MAC at 49800 ns and will appear on the output (time gate schedule timing point) at the gate opening of 50000 ns.

53.4.2.3.1.2.1.3 VLAN Insert

The device supports the offloading of VLAN tag insertion. VLAN tag can be inserted in the following ways:

- **SI-Based VLAN insertion** – The PF driver through the PSI can request that hardware inserts the SI-based VLAN into the outgoing frame. This is inserted “outside” of any software tag i.e. this tag is inserted closest to the Ethernet header with its Ethertype immediately following the SMAC address. SI-based VLAN insertion is enabled by setting PSI*a*CFGR0[SIVIE] to 1 and PSI*a*VLANR[E] to 1 along with the appropriate VLAN tag configuration. (i.e. VLAN ID, PCP, DEI and TPID settings in the PSI*a*VLANR register).
- **Descriptor-based VLAN insertion** – Software can request hardware to insert a VLAN tag by putting the VLAN tag to be inserted in the transmit BD ring. TB*a*MR[VIH] is set to 1 to indicate that software intends to use descriptor-based VLAN insertion offload. When this flag is set, calculations performed by hardware in the transmit scheduler involving the size of the frame (e.g. credit-based shaper) will account for the extra 4 bytes of VLAN information in their calculations regardless of whether descriptor-based VLAN insertion offload is used or not used.

The following processing is applied to each frame as it is taken from a transmit BD ring for transmit:

1. Firstly, if the frame from the host has a VLAN tag (either in the frame itself or in the BD descriptor (descriptor-based VLAN insertion), a check is made to determine if the Ethertype of the VLAN tag is "acceptable". If no descriptor-based insertion is requested then the outer Ethertype in the frame (16b immediately following the SMAC address) is compared against the VLAN Ethertypes (0x8100, 0x88A8, CustTag1 (CVLANR1[ETYPE] if valid), CustTag2 (CVLANR2[ETYPE] if valid)) in order and matching stops on the first match found , and a 2-bit TPID value is assigned identifying which known value matched (00: 0x8100, 01: 0x88A8, 10: CustTag1, 11: CustTag2). If a match is found, the 2-bit TPID is checked against the 4-bit map of allowed VLAN types for this SI (PSI*a*CFGR0[SIVC]). By using the 2-bit TPID value as an index into the SIVC bitmap. If the corresponding bit is set then the VLAN tag has an "acceptable" Ethertype. If the corresponding bit is not set, then this VLAN tag not allowed and the frame is not transmitted and an appropriate error is reported in the BD. If no match is found when the outer Ethertype in the frame is compared against the VLAN Ethertypes, the frame is considered untagged, and thus accepted for transmission.
2. If a VLAN is provided in the Tx BD then the 2-bit TPID of the VLAN tag to be inserted (taken from the Tx BD) is checked to determine if the Ethertype of the VLAN tag is "acceptable". If the Ethertype to be used to construct the VLAN tag is a custom VLAN Ethertype (i.e. CVLANR1/CVLANR2[ETYPE]), the CVLANR1/CVLANR2[V] field is checked to see if it is set to 1 (i.e. valid). If set to 0, the VLAN Ethertype specified in the BD to be inserted is not valid, and as a result the frame is not transmitted and an appropriate error is reported in the BD. If set to 1, the 2-bit TPID is then checked against the 4-bit map of allowed VLAN types for this SI (PSI*a*CFGR0[SIVC]) by using the 2-bit TPID as an index into the SIVC bitmap. If the corresponding bit is set then the VLAN tag has an "acceptable" Ethertype. If the corresponding bit is not set, then this VLAN tag is not allowed and the frame is not transmitted and an appropriate error is reported in the BD.
3. Hardware VLAN insertion is implemented as required. Logically this can be described as:
   - Any BD supplied VLAN is inserted in the outer position immediately following the SMAC address.
   - If configured the SI-based VLAN is inserted in the outer position immediately following the SMAC address. If both descriptor-based VLAN insertion and Si-based VLAN are enabled, the SI-based VLAN tag will appear as the outer most tag inserted, immediately after the Ethernet header (i.e. immediately following the SMAC address) and the descriptor-based VLAN tag will appear as the second outermost VLAN tag in the frame. Note that there is no checking needed for the SI based VLAN insertion. That is controlled by the PSI and, if enabled, is performed unknown to the driver on the SI itself and is inherently allowed.
4. Any bandwidth calculations used in transmit will account for the extra 4/8 bytes of VLAN information in their calculations.

53.4.2.3.1.2.1.4 Source MAC anti spoofing

The hardware supports anti spoofing functionality. This feature ensures that software is sending frames with a source MAC address that is associated with the SI.

If anti spoofing is enabled for an SI (PSI*a*CFGR0[ASE] set to 1), the source MAC address of each frame being sent by this SI is compared against its SI Primary MAC address register PSI*a*PMAR0/1 (where a is the SI number). If it matches, then the frame is allowed to be transmitted. Otherwise, a lookup of the source MAC address is performed in the exact match MAC address table. If this does not result in a match with the SI’s bit set in the result bitmap then the frame is not transmitted.

The Source MAC anti spoofing feature is not applicable to frames that are enqueue directly to egress switch ports using the direct switch enqueue capability (SMSO=1).

53.4.2.3.1.2.2 Egress Port Processing (EPP)

The Egress Port Processing block acts as a staging resource to mediate the transfer of transmit data to the MACs. Two priority queues per port are used to hold frames waiting for transmission to the MAC.

A physical port supporting IEEE 802.1Qbu frame preemption contains two MACs; i.e. MAC 0 and MAC 1. MAC 0 is called the express MAC and is used to handle the express traffic, whereby MAC 1 is called the preemptable MAC and is used to handle the preemptive traffic. Within the Egress Port Processing block, there is an internal queue for holding frames waiting to be transmitted on the express MAC and another internal queue for holding frames waiting to be transmitted on the preemptable MAC. If preemption is disabled, only MAC 0 (express MAC) is used. MAC 1 (preemptable MAC) and its corresponding internal queue are not used. The traffic class assignment to either the express MAC or the preemptable MAC is specified by setting PFPCR[FPE_TCa] (where a corresponds to the traffic class number). For more details on frame preemption, see Preemption.

A physical port that doesn't support IEEE 802.1Qbu frame preemption contains one MAC and one internal queue within the Egress Port Processing block. Similarly, a port attached to a pseudo MAC contains one internal queue within the Egress Port Processing block.

A port-per-HTA Transmit priority byte credit-based flow control mechanism is used to limit the amount of frame bytes in the ENETC transmit datapath that have been scheduled for transmission but not yet transferred to a given port and MAC ((express MAC or preemptable MAC (physical port), or pseudo MAC). There is a separate byte credit count maintained for each MAC, each initialized to the value configured in the ENETC *a* transmit high priority tier byte credit register (E*a*TXHPTBCR) for the express MAC or pseudo MAC and in the ENETC *a* transmit low priority tier byte credit register (E*a*TXLPTBCR) for the preemptable MAC. Here *a* is the ENETC instance number.

Credits are decremented when a frame is scheduled by HTA Transmit and incremented as the frame is being sent to the MAC. HTA is allowed to schedule a frame as long as credits are greater than 0. This will limit the number of frame bytes in the ENETC transmit path to "E*a*TXHPTBCR[BYTE_CREDIT] + maximum frame size" for traffic destined to the express MAC or pseudo MAC and to "E*a*TXLPTBCR[BYTE_CREDIT] + maximum frame size" for traffic destined to the preemptable MAC.

When IEEE 802.1Qbv or time specific departure (TSD) is enabled, the HTA transmit scheduler will indicate the departure time at which the frame is to be transmitted to MAC.

53.4.2.3.1.2.3 Ethernet Tx I/F

The Ethernet Tx I/F block integrates the Tx MAC functionality and implements the interface functions specific to the MAC from the transmit path.

53.4.2.3.1.2.3.1 Ethernet MAC

In the case where frames are transmitted to an Ethernet MAC, the Ethernet Tx I/F retrieves the frame from the shared internal buffer memory and transfers it to the Ethernet MAC.

If the FCS is not being provided by software, then it is calculated by HTA Transmit as it is performing frame modifications if any, and loading the frame from host memory to the shared internal buffer memory. Whether or not software provides the FCS, is specified through the register TB*a*MR[CRC], where a is the BD ring number. Once the frame has been loaded into the shared internal buffer memory, the calculated 4 bytes FCS is appended to the frame. If the port is configured to pad a frame to the minimum length of 64 bytes (see PMa_COMMAND_CONFIG[TXP] register for more details), HTA transmit will pad the frame with 0s to extend the frame to 60 bytes after performing frame modifications, if any. It will then append the calculated 4 bytes FCS to the frame.

The Ethernet MAC transmit function manages inter-packet gap and generates preamble and SFD or SMD.

The MAC supports configurable minimum frame size from 18 to 64B, and configurable maximum frame size up to 2000B. See Port MAC a Minimum Frame Length Register (PM0_MINFRM - PM1_MINFRM) and Port MAC a Maximum Frame Length Register (PM0_MAXFRM - PM1_MAXFRM) registers for more details.

After a frame has been sent to the Ethernet MAC, the Ethernet Tx I/F performs the following actions:

- For any given transmit frame, the Ethernet MAC returns the timestamp at the moment where the one-octet start frame delimiter (SFD) of the frame was transmitted. PMa_COMMAND_CONFIG[TS_MODE] determines which clock source

(synchronized 1588 clock or 802.1AS free running clock) is used for sampling the timestamp. The Ethernet Tx I/F returns this value to HTA which in turn uses this as the indication that the corresponding transmit BD descriptors have been transmitted, and if the IEEE 1588 PTP two-step timestamp offload is enabled, writes back the returned timestamp in the metadata of the host transmit descriptor.

- IEEE 1588 PTP one-step timestamp offload is also supported. When set, the Ethernet MAC is instructed to insert the timestamp information into the frame when transmitting the frame.
- Ethernet Tx I/F increments in chunks of 72 bytes, the high priority byte credit count for the port-per-HTA Transmit priority (2 priorities) byte credit-based flow control mechanism, for traffic directed to the express MAC. Similarly, it increments in chunks of 72 bytes, the low priority byte credit count for the port-per-HTA Transmit priority (2 priorities) byte credit-based flow control mechanism, for traffic directed to the preemptable MAC.
- Internal buffers holding the frame are released.

For any uncorrectable errors detected (such as multi-bit ECC errors), the frame is discarded if the frame transfer to the MAC has not started, and an appropriate error is reported in the transmit buffer descriptor. If the frame transmission has started, the frame can no longer be discarded at this point. Instead the MAC is instructed to set the frame FCS field to a bad CRC value to ensure that the next hop identifies the frame as being bad. The error (no longer a frame discard) is counted against the port's MAC FCS counters (PMa_TERRn and PMa_TFCSn).

53.4.2.3.1.2.3.2 Pseudo MAC

For frames transmitted to a pseudo MAC, there is no need to retrieve or copy these frames, as only a frame descriptor is being transferred across that interface.

Note that IEEE 1588 PTP one-step timestamp offload is not supported on the pseudo MAC.

If the FCS is not being provided by software, then it is calculated by HTA Transmit as it is loading the frame from host memory to the shared internal buffer memory. Whether or not software provides the FCS, is specified through the register TB*a*MR[CRC], where a is the BD ring number. Once the frame has been loaded into the shared internal buffer memory, the calculated 4 bytes FCS is appended to the frame. If the port is configured to pad a frame to the minimum length of 64 bytes (see PPMCR[TXPAD], for more details), HTA transmit will pad the frame with 0s to extend the frame to 60 bytes after performing frame modifications. It will then append the calculated 4 bytes FCS to the frame.

After a frame has been passed to the pseudo MAC, the Ethernet Tx I/F performs the following actions:

- Increments in chunks of 72 bytes, the high priority byte credit count for the port-per-HTA Transmit priority (2 priorities) byte credit-based flow control mechanism.
- For any given transmit frame, the pseudo MAC returns the timestamp at the moment where the frame was transferred across the pseudo link. PCR[TIMER_CS] determines which clock source (synchronized clock or free running clock) is used for sampling the timestamp. The Ethernet Tx I/F returns this value to HTA which in turn uses this as the indication that the corresponding transmit BD descriptors have been transmitted, and if the IEEE 1588 PTP two-step timestamp offload is enabled, writes back the returned timestamp in the metadata of the host transmit descriptor.

If the ENETC port is connected internally to a switch port, designated as a switch management port, an out of band metadata indicating whether to forward as it was received from the switch management port itself or from another switch port, is passed along with the frame to the pseudo MAC. This out of band metadata is taken from the host transmit descriptor of the frame.

53.4.2.3.1.3 Receive Datapath

The receive data path is implemented as a series of functional processing blocks pipelined together. This includes the Ethernet Rx I/F, Ingress Port Processing, Ingress Congestion Manager, and the Host Transfer Agent (receive). Details of these blocks are given in the following sections.

53.4.2.3.1.3.1 Ethernet Rx I/F

The Ethernet Rx I/F block integrates the Rx MAC functionality and implements the interface functions specific from the MAC to the ENETC receive path.

53.4.2.3.1.3.1.1 Ethernet MAC

In the case where frames are received from an Ethernet MAC, the Ethernet Rx I/F stores them immediately in the shared internal buffer memory.

The Ethernet MAC detects the SFD or SMD, receives the frame, and checks for layer 2 data integrity (FCS errors) while forwarding data to MAC client. All frames >= 18B with valid SFD are forwarded to MAC client.

The last 4 bytes of the frame which represents the frame check sequence (FCS), are also stored in the shared internal buffer memory. The Ethernet MAC does not remove the FCS from the frame. The FCS carried in the frame remains as the frame is processed by the ENETC receive datapath. The FCS is not updated if frame modifications are performed on the frame by the ENETC receive datapath (i.e. VLAN removal offloads). When frames are transferred into receive data buffers in the host system memory, there is the option to preserve or remove the FCS. Whether to preserve or remove the FCS is specified through the register RB*a*MR[CRC], where *a* is the BD ring number.

For any internal error detected, the ERROR field of the receive descriptor for the frame is updated to indicate that the frame is bad (internal fault code point (0x80) set).

The first 144 bytes of the received frame in addition of being stored in the shared internal buffer memory, are also stored in a frame header memory which is accessed by the Ingress Port Processing (IPP) for processing the received frame. Once the first 24 bytes of data is received and presented to the IPP, it starts parsing the frame header (first 24 bytes). If the end of the parse graph has not been reached while parsing the first 24 bytes of the frame, the IPP waits for the next 24 bytes to be received so that it can resume parsing, and so on up to 144 bytes. Once parsing is completed and although the entire frame may not have been entirely received, the IPP proceeds with executing the required lookup-ups and resulting packet processing actions that aren't dependent on at the end of frame status (good/bad) and/or frame length. The IPP will then wait for the entire frame to be received before executing actions that are dependent on the end of frame status and/or frame length (for example, policing, byte counters updates).

No drops are expected when delivering frames from the Ethernet Rx I/F to the Ingress Port Processing (IPP) as the IPP is engineered to operate at wire-rate (assuming a minimum frame length of 64 bytes); no back-pressure is expected from the IPP. If an unforeseen overload condition of the IPP is detected, received frames from the MAC will be discarded. These frame discards are counted against the port's Rx discard count register (PRXDCR) along with the setting of the Pre-Classification Discard Reason (PCDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0).

The following metadata is passed along with the frame:

- Two timestamps (one sampling the synchronized clock and the other sampling the free running clock); for all receive frames, the timestamps are captured when MAC detects the one-octet start frame delimiter (SFD) of the frame. The receive timestamp is reported in the receive BD ring descriptor if the receive BD ring is configured with the extended 32B buffer descriptor format. PCR[TIMER_CS] determines which of the two timestamps (synchronized or free running) is to be reported.
- Indication that the frame was received from an Ethernet MAC (not a pseudo MAC) along with the indication whether it was received from the Express MAC or the Preemptable MAC.
- ENETC instance number; acquired from the register LaBCR[SW_PORT_ENETC_INST], where a corresponds to the link number from which the frame was received.
- Frame length; calculated by the Ethernet RX I/F as it transfers the frame from the MAC Rx FIFO to internal memory.
- Indicator if MAC detected an error (for example, CRC error, invalid frame length, and so on). Errored frames are passed to the Ingress Port Processing where they will be discarded.
- Indicator of whether the received frame is a magic packet. On ports supporting the power management Wake-on-LAN feature, every frame is scanned by the MAC for occurrences of 6 bytes of 0xFF, followed by sixteen repetitions of the endpoint MAC address. The MAC address in the magic packet pattern, must match the MAC address configured in the PM*a*_MAC_ADDR_0/1 registers. No check is performed beyond the magic packet pattern; e.g. destination MAC address in the Ethernet header of a magic packet can be a broadcast, multicast, or unicast MAC address.
- Indicator that the received frame was truncated due to the lack of free shared internal buffer memory available to store the entire frame. These frame are passed to the Ingress Packet Processing where they will be discarded. These frame discards are counted against the port's Rx discard count register (PRXDCR) along with the setting of the Shared Memory Resource Exhaustion Discard Reason (SMREDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0).

For ports supporting IEEE 802.1Qbu frame preemption, there are two MACs used per port (express MAC and preemptable MAC). In this case, the Ethernet RX I/F implements the interface functions to both MACs.

53.4.2.3.1.3.1.2 Pseudo MAC

For frames received from a pseudo MAC, only frame descriptors are transferred across a pseudo link. There is no need to store or copy these frames as they have already been stored in the shared internal buffer memory by the switch. Frame descriptors received on a pseudo link from a switch port include references to the original frame (including FCS) and annotated frame header if present as a result of the switch having to perform frame modifications. This annotated frame header represents the first n bytes of the frame, along with an indication of how many bytes from the start of the original frame are no longer valid. There is an FCS covering the annotated frame header, which is distinct from the FCS covering the entire original frame.

The pseudo MAC doesn't remove the FCS(s). The FCS or two FCSs carried in the frame remain as the frame is processed by the ENETC receive datapath. The FCS(s) are not updated if frame modifications are performed on the frame by ENETC receive datapath (i.e. VLAN removal offloads).

When a frame is received, the first 144 bytes of the received frame are stored in a frame header memory which is accessed by the Ingress Port Processing (IPP) for processing the received frame.

No drops are expected when delivering frames from the Ethernet Rx I/F to the IPP as the IPP is engineered to operate at wire-rate (assuming a minimum frame length of 64 bytes); no back-pressure is expected from the IPP. If an unforeseen overload condition of the IPP is detected, received frames from the pseudo MAC are no longer accepted; frames are not dropped, instead back-pressure is applied towards the switch.

The following metadata is passed along with the frame:

- Two timestamps (one sampling the synchronized clock and the other sampling the free running clock) at the time the frame was transferred across the pseudo link. For the receive timestamp reported in the receive BD ring descriptor (receive BD ring must be configured with the extended 32B buffer descriptor format), PCR[TIMER_CS] determines which of the two timestamps (synchronized or free running) is to be reported.
- Indication that the frame was received from a pseudo MAC.
- ENETC instance number; derived from the register LaBCR[SW_PORT_ENETC_INST]
- Frame length.
- If the ENETC port is connected internally to a switch port, designated as a switch management port, an indication of the source port ID from which the frame was received is provided along with the reason why this frame was forwarded to the switch management port. Once the frame has been transferred to the host system memory, the source port ID and reason to forward to the switch management port, are placed in the metadata of the receive descriptor for the frame. Only frames with the Host Reason set to "regular forwarded frame" can be received from a pseudo MAC. Frames with a Host Reason set to a codepoint other than a "regular forwarded frame" do not go through a pseudo link, instead they are sent directly from the switch to the ENETC Ingress Congestion Manager (ICM), where they will be queued based on internal QoS set by the switch, for delivery to the HTA receive; note that all of the Ingress Port Processing (IPP) functions are by-passed with bit 0 of the SI bitmap set to 1 (indicates to HTA that the frame is to be delivered to SI numbered 0).

No errors are expected; FCS is not checked as there are no transmission of bits on pseudo link interface as only a frame descriptor is being transferred across that interface.

53.4.2.3.1.3.2 Ingress Port Processing (IPP)

The figure below shows the processing through the Ingress Port Processing.

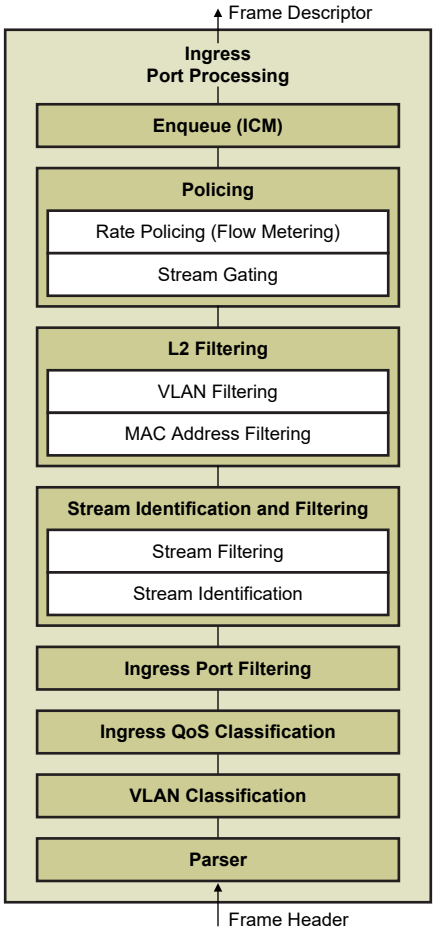


Figure 335. Ingress Port Processing diagram

The Ingress Port Processing block responsibilities are to parse the frame headers and to do the necessary table lookups (or classification) to determine the internal QoS attributes (Internal Priority Value/IPV and Discard Resiliency/DR), to decide whether to accept the frame and to determine the set of Station Interfaces (SIs) where to forward the frame next.

After it is determined that a frame can be delivered to a least one SI, a policing function may be applied. Policing involves determining whether a frame is in or out of some defined time window and bandwidth profile. If out of profile, the DR is marked down (more discardable) or the frame is dropped as per profile. Frames rejected by the prior classification stage are not policed.

The Ingress Port Processing block performs the following ingress port processing functions in a pipeline manner:

- **Parser** - Parses the frames and identifies the locations of the required packet header fields in preparation for classification. Supports major protocols such as Ethernet, IPv4, IPv6, TCP and UDP.
- **VLAN classification** - Determine what VLAN tag data will be passed with the frame for later ingress processing. This includes optional assignment of configured port tag data if the frame is untagged.
- **Ingress QoS classification** - Provides internal identification of the QoS attributes that are used for subsequent QoS processing as frames go through the device. As an example, the IPV and DR can be set based on the pbits and DE bit in a VLAN tag. Internal QoS attributes indicate the given QoS (queueing priority and discard resiliency) as the frame moves through the device. At any queuing point, frames can be placed in the appropriate queue based on the internal queuing priority and can be dropped based on the internal drop priority.
- **Ingress port filtering** - Can deny or allow a frame. It may also override internal QoS attributes associated with the frame and set the stream ID or rate policer instance ID. It can also forward a frame to a particular set of SIs.

- **Stream identification and filtering** - Implements stream identification and filtering as per IEEE 802.1Qci (Per-Stream Filtering and Policing).
- **L2 filtering** - Performs a series of lookups based on SMAC, DMAC and VLAN, to decide whether to accept the frame and to provide the set of Station Interfaces (SIs) eligible to receive the frame.
- **Policing** - Implements traffic policing as per IEEE 802.1Qci (Per-Stream Filtering and Policing) to protect against time window and bandwidth violations.
- **Enqueue** - Sends frames towards the HTA block by passing the frames to the Ingress Congestion Manager (ICM) where they will be queued internally (inside ICM); i.e. frames are placed in the appropriate ICM queue based on their incoming port and internal Priority Value (IPV). There are 2 priority queues per port. Within a port the IPV is used to select one of the priority queues based on the configured IPV to ICM receive port priority queue mapping (IPV2ICMPMR0) register. There is no drop at this point (i.e. no enqueue reject) as ICM ensures that an arriving frame is always accepted by having the ability to drop some already stored and enqueued frames, in the case where the ENETC receive shared internal buffer memory becomes full.

Frames received from the Ethernet Rx I/F with the indication that an error was detected are dropped. No drop count is incremented at this point as the appropriate error count has already been incremented by the Ethernet Rx I/F.

53.4.2.3.1.3.3 Parser

The Parser parses up to L4 protocol headers in up to the first 144 bytes of the frame. Extracted field values and locations of relevant L2, L3 and L4 headers values are then used to perform the various table lookups and hash functions in subsequent processing stages of the IPP.

During parsing, the parser examines and validates headers.

The parsing process can be expressed as a directed graph with nodes representing header types and edges (arrows) representing the sequence of headers. Parsing starts at the root node then walks from node to node using the next-header field values. Specifically, each node works within a protocol layer presuming a certain type of header, confirms the validity of the header, identifies boundaries of the header, extracts attributes and offsets, and determines the next header to examine beyond the current header.

Figure below shows the parse graph supported in the IPP. The parse graph is structured by layers (L2, L3, and L4), with the details per each layer provided in the next sections.

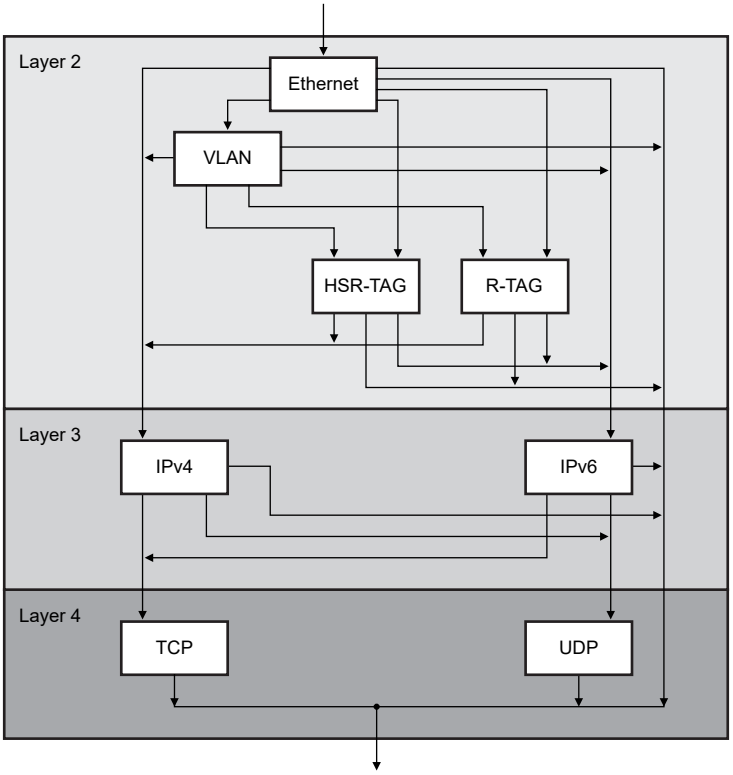


Figure 336. IPP Parse Graph

By default, the parser will parse and extract fields from each layer. There is the option to configure the parser to end parsing at L2 or L3 regardless of the content of the frame (for more details, see the port parser configuration register (PPCR)). The parser will also extract the payload (up to 24 bytes by default) of the last layer parsed (L2, L3 or L4). The maximum amount of payload to extract is configurable for each layer via the port parser configuration register (PPCR).

Parsing is terminated via one of the following:

- Upon reaching the natural end of a branch in the parse graph
- Upon reaching a layer (L3 or L4) for which the parser has been configured not to parse.
- A parsing error is encountered; note that parse information from headers that have been successfully parsed is still used by table lookups and hash functions in subsequent processing stages of the IPP.

53.4.2.3.1.3.3.1 Layer 2 Headers

As the first header in a frame, the parser supports Ethernet frames of the IEEE 802.3 (Ethernet II) format. IEEE 802.3 (Ethernet II) frames may be tagged with the IEEE 802.1Q VLAN tagging method. The 4-byte IEEE 802.1Q VLAN tag has two 2-byte components: Tag Protocol Identifier (TPID, residing within the type/length field) and the Tag Control Information (TCI). The following 4 TPID values are recognized as VLAN tags:

- The standard types 0x8100 and 0x88A8
- Two optional user programmable EtherType values known as CustTag1 and CustTag2; configured through CVLANR1 and CVLANR2 registers.

The TCI includes the following fields:

- PCP - Priority Code Point, the first three bits of the TCI define user priority.
- DEI - Drop Eligibility/ (single-bit field)
- VID - The 12-bit VLAN ID (VID) field identifies the VLAN the frame belongs to. The 12 bits allow for 4096 different VLANs. Out of the 4096 VLANs, the VID of zero is used to identify priority tagged frames and the value of 4095 is reserved.

Parser can parse frames with up to 2 VLAN tags. VLAN tags are named based on their position in the frame relative to other VLAN tags and the beginning of the frame i.e. the 802.3/Ethernet header consisting of Destination followed by Source MAC addresses.

The VLAN tag or header nearest the 802.3/Ethernet header is called the outer (or first) tag. The VLAN tag which is next closest to the 802.3 header is the inner (or second) tag. VLAN tags are expected to be contiguous, following one after the other, with no intervening non-VLAN tags.

Note that any VLAN tags which are within encapsulated frames and after any Upper Layer Protocol (ULP) header are not recognized. If only one VLAN tag is present in the frame, the inner VLAN tag is considered not present.

The outer and inner VLAN tag data (PCP, DEI and VID), if present, are extracted from the frame. The TPID from each VLAN tag is matched against the 4 known VLAN EtherTypes (2 standard, up to 2 custom) and a 2-bit TPID value is assigned identifying which known value matched (00: 0x8100, 01: 0x88A8, 10: CustTag1, 11: CustTag2). Matching is done against the list {0x8100, 0x88A8, CustTag1, CustTag2} in order and matching stops on the first match found. A 4-bit vector where each bit represents a TPID code point (bit 0 -> 00, bit 1 -> 01, bit 2 -> 10 and Bit 3 -> 11) is used by subsequent processing stages within the IPP to perform non-contiguous masked matching operation against the TPID.

The information found in the next EtherType field (after VLAN parsing) is used to determine the next ULP header to parse. The EtherType field is located following the Source MAC address field if no VLAN tag present, or following the last VLAN tag. The ULP header is determined according to the “Ethernet - EtherType to Next ULP” mapping described in the table below.

EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends. If parsing reached end of frame, while more octets are expected to extract the 2-byte EtherType, the extracted EtherType is set to 0.

Table 398. Ethernet - EtherType to Next ULP Mapping

EtherType	Recognized Protocol	Next ULP to be parsed
x0800	IPv4	IPv4
x86DD	IPv6	IPv6
x892F	HSR Tag	HSR Tag
xF1C1	IEEE 802.1CB R-TAG	Redundancy tag
Pre-standard R-TAG EtherType register	IEEE 802.1CB draft 2.0 R-TAG	Redundancy tag

For High Availability Seamless Redundancy (HSR) tag, the parser locates the EtherType field following the 6-byte HSR tag. The next ULP header is determined according to the “HSR tag - EtherType to Next ULP” mapping described in the table below. If the EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 399. HSR tag - EtherType to Next ULP Mapping

EtherType	Recognized Protocol	Next ULP to be parsed
x0800	IPv4	IPv4
x86DD	IPv6	IPv6

For the 802.1CB R-tag, the parser locates the EtherType field following the 6-byte CB Redundancy tag. The next ULP header is determined according to the “CB Redundancy tag - EtherType to Next ULP” mapping described in the table below. If the EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

The 4-byte pre-standard (802.1CB draft 2.0) R-TAG is also supported, The format of this tag is 2 bytes for the EtherType followed by 2 bytes for the sequence number. The recognized value for the Ethertype value of the pre-standard R-TAG is configured through the pre-standard R-TAG Ethertype register (PSRTAGETR). For the 4-byte pre-standard R-TAG, the parser locates the EtherType field following the 4-byte pre-standard R-TAG. The next ULP header is determined according to the "Redundancy tag -

EtherType to Next ULP" mapping described in the table below. If the EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 400. CB Redundancy tag - EtherType to Next ULP Mapping

EtherType	Recognized Protocol	Next ULP to be parsed
x0800	IPv4	IPv4
x86DD	IPv6	IPv6

Following information is provided along with the frame, by the parser for subsequent processing within the IPP:

- Source and Destination MAC addresses are extracted.
- Indication of whether the outer VLAN tag header is present. If the header is present and valid (no parsing error), extracted VLAN tag data along with a 2-bit code identifying the TPID is also provided.
- Indication of whether the inner VLAN tag header is present. If the header is present and valid (no parsing error), extracted VLAN tag data along with a 2-bit code identifying the TPID is also provided. If only one VLAN tag is present in the frame, the inner VLAN tag is considered not present.
- Whether destination MAC address is unicast, multicast or broadcast.
- A 3-bit redundancy tag type codepoint
  000b = No redundancy tag present
  001b = 802.1CB draft 2.0 R-TAG.
  010b = 802.1CB R-TAG.
  011b = HSR Tag
  100b ... 111b = Reserved.
- The sequence number is extracted if a redundancy tag is present and valid (no parsing error)
- The last parsed EtherType is extracted; i.e. 2 bytes immediately after source MAC address or VLAN tag(s)/R-TAG/HSR (if any). If the EtherType wasn't parsed successfully (e.g. missing one byte due to frame too short), the EtherType is set to all 0s.
- Extracting up to 24 bytes located immediately after payload EtherType. The payload EtherType is located immediately after the source MAC address or VLAN tag(s)/R-TAG/HSR (if any) if parsing is ending at this layer. The maximum amount of payload to extract is configurable via the port parser configuration register (PPCR).

Possible parsing errors for L2 headers are:

- Frame parsing reached end of frame while parsing a VLAN tag or R-TAG/HSR that expects more data. Frames shorter than 14 bytes excluding 4-byte FCS will never reach the parser as they will be discarded by the Ethernet Rx I/F.

There are no validation checks performed on the Source and Destination MAC addresses, and EtherType fields.

Note that if an L2 header parsing error is detected, the frame is not discarded. Instead, only successfully parsed L2 header fields are extracted, but no payload data is extracted. The parse information is then passed along with the frame for subsequent processing within the IPP(e.g. table lookups and hash functions).

Note that any sequence tag information extracted by the parser for usage by subsequent processing stages, is overwritten if FRER sequence generation is enabled for the frame. NETC only supports ONE CB redundancy tag, one added as part of sequence generation if enabled, or a parsed redundancy tag if present.

53.4.2.3.1.3.3.2 Layer 3 Headers

The parser supports parsing of both IPv4 and IPv6 packets. Only the first IPv4 or IPv6 header encountered in a packet is parsed.

IPv4 options are not parsed. They are also not skipped meaning that the next ULP header will not be parsed if the packet has IPv4 options.

If a non-fragmented IPv4 packet (Fragment Offset is set to zero and the More Fragments flag is set to zero) or an initial fragment packet ((Fragment Offset is set to zero and the More Fragments flag is set to one) is parsed, then the parser uses the information stored in the Protocol field to determine the next ULP header to be parsed.

For IPv4 header, the next ULP header is determined according to the "IPv4 - Protocol to Next ULP" mapping described in the table below. If the Protocol field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 401. IPv4 - Protocol to Next ULP Mapping

Protocol (decimal)	Recognized Protocol	Next ULP to be parsed
6	TCP	TCP
17	UDP	UDP

For IPv6 header, the IPv6 main header may be followed by an arbitrary number (0, 1 or multiple) of extension headers chained together through the Next Header field. IPv6 extension headers are not parsed. They are also not skipped meaning that the next ULP header will not be parsed if the packet has extension headers.

The parser uses the information stored in the Next Header field of the IPv6 main header to determine the next ULP header. The next ULP header is determined according to the “IPv6 – Next Header to Next ULP” mapping described in the table below. If the Next Header is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 402. IPv6 - Next Header to Next ULP Mapping

Next Header (decimal)	Recognized Protocol	Next ULP to be parsed
6	TCP	TCP
17	UDP	UDP

Following information is provided along with the frame, by the parser for subsequent processing within the IPP:

- Indication of whether the outer IP header is present and valid (no parsing error). If the header is present and valid, indication of whether IP header is v6 or v4, is also provided.
- Indication of whether IPv4 options / IPv6 extensions are present. Note that value 59 (No Next Header) in the Next Header field of the IPv6 main header, is not considered an IPv6 extension header. See table below for the recognized IPv6 extension headers.
- IP source address , IP destination address, DSCP and Protocol fields are extracted.
- Extracting up to 24 bytes located immediately after the IP header if parsing is ending at this layer. The maximum amount of payload to extract is configurable via the port parser configuration register (PPCR). Note that if IPv4 options or IPv6 extensions are present, payload extraction starts at the beginning of the IPv4 Options/IPv6 Extensions header.

Possible parsing errors for L3 headers are:

- Frame parsing reached end of frame while parsing a L3 header that expects more data.
- For each IPv4 header parsed, if any of the following conditions is detected:
  - Version number in IPv4 header is not equal to 4.
  - IPv4 Header Length field value is less than 20 bytes, or exceeds the received packet length (excludes L2 header), or exceeds the IPv4 header Total Packet Length field value
- For each IPv6 header parsed, if any of the following conditions is detected:
  - Version number in IPv6 header packet is not equal to 6,

Note that if an L3 header parsing error is detected, the frame is not discarded. Instead, the L3 header is considered not present, and parsing ends. The parse information from the L2 headers (including L2 payload if extracted) is passed along with the frame for subsequent processing within the IPP (e.g. table lookups and hash functions).

Table 403. IPv6 extension headers

Protocol Number (decimal)	Description
0	IPv6 Hop-by-Hop Option
43	Routing Header for IPv6
44	Fragment Header for IPv6
50	Encapsulating Security Payload
51	Authentication Header
60	Destination Options for IPv6
135	Mobility Header
139	Host Identity Protocol
140	Shim6 Protocol
253	Use for experimentation and testing
254	Use for experimentation and testing

53.4.2.3.1.3.3.3 Layer 4 Headers

The parser supports parsing of both TCP and UDP packets. Only the first TCP or UDP header encountered in a packet is parsed.

The TCP header may have options but these aren't examined.

Following information is provided along with the frame, by the parser for subsequent processing within the IPP:

- If the L4 header parsed is TCP, an indication that TCP is present (L4 codepoint set 01b).
- If the L4 header parsed is UDP, an indication that UDP is present (L4 codepoint set 10b).
- Source and destination port numbers are extracted.
- If no TCP nor UDP header is detected, an indication that other L4 is present (L4 codepoint set 00b).
- Extracting up to 24-bytes located immediately after the first 4 bytes of the TCP/UDP header. The maximum amount of bytes to extract is configurable via the port parser configuration register (PPCR).

Possible parsing errors for L4 headers are:

- Frame parsing reached end of frame while extracting source and destination port numbers (L4 header shorter than 4 bytes).

Note that if an L4 header parsing error is detected, the frame is not discarded. Instead, the L4 header is considered other L4, and parsing ends. The parse information from the L3 headers (including L3 payload if extracted) is passed along with the frame for subsequent processing within the IPP (e.g. table lookups and hash functions).

53.4.2.3.1.3.4 VLAN classification

VLAN classification processing stage determines what VLAN tag data will be passed with the frame for later ingress processing.

Following VLAN tag metadata provided by the parser is used by VLAN classification:

- Indication of whether the outer VLAN tag header is present (no parsing error)
- Indication of whether the inner VLAN tag header is present (no parsing error)

- Extracted outer VLAN tag data from frame
  - PCP, DEI, VID
  - TPID expressed as 2-bit code point
- Extracted inner VLAN tag data from frame
  - PCP, DEI, VID
  - TPID expressed as 2-bit code point

Output of the VLAN classification is:

- VLAN tag attribute indicators
  - Indication of whether the frame is untagged
  - Indication of whether outer VLAN tag is valid, if valid then the following attributes further qualifies it
    - VLAN tag data taken from port outer VLAN configuration register
    - VID is set 0 in the frame outer VLAN tag; i.e. priority tag
  - Indication of whether inner VLAN tag is valid, if valid then the following attributes further qualifies it
    - VLAN tag data taken from port inner VLAN configuration register
    - VID is set to 0 in the frame inner VLAN tag; i.e. priority tag
- Outer VLAN tag data (PCP, DEI, VID) if outer VLAN tag is valid
- Inner VLAN tag data (PCP, DEI, VID) if inner VLAN tag is valid

The setting of the above VLAN classification output attribute indicators (along with appropriate VLAN tag data) is provided in the following steps:

1. All VLAN classification output indicators are cleared at the start of VLAN classification processing except for the outer/inner tag presence indicators which are taken from the parser. Outer/inner tag data and 2-bit TPID code point are also taken from the parser
2. If no tags are present then frame metadata is marked as untagged.
3. If only the outer tag is present (i. e. there is only one VLAN tag recognized in the frame) and the setting requires that it be treated as the inner tag (PVCLCTR[OAI] bit set to 1) then the outer tag data is used as the inner tag data for further processing. The outer tag is deemed not present and the inner tag is deemed present.
4. For each of the inner/outer tags which are (deemed) present the appropriate acceptance bitmap (PTAR[IVTPIDL/OVTPIDL]) is checked. If the tag's TPID's bit is set then the tag is acceptable otherwise it is deemed not present.
5. For each of the inner/outer tags which are (deemed) present if the tag has a 0 VID and port VLAN VID 0 usage is enabled (PINVLANR/PONVLANR[VZE] bit is set to 1) then insert the port VLAN VID (PINVLANR/PONVLANR[VID]) into the frame's metadata VLAN tag data. Frame metadata is marked as having VID 0 in that tag for upstream use.
6. For each of the inner/outer tags which are (deemed) not present if the port VLAN is enabled (PINVLANR/PONVLANR[PNE] bit is set) then the appropriate configured (inner/outer) port VLAN tag data ((PINVLANR/PONVLANR[PCP], (PINVLANR/PONVLANR[DEI] and PINVLANR/PONVLANR[VID]) is used as the VLAN tag data in the frame metadata. The tag is now deemed present. Frame metadata is marked as using port VLAN data in that tag for upstream use.
7. For each of the inner/outer tags if the tag is (deemed) present then frame metadata is marked as tag data being valid.

The resulting VLAN tag data and attribute indicators are passed along for upstream use.

53.4.2.3.1.3.5 QoS classification

QoS classification involves setting the internal QoS. The internal QoS indicates the given QoS (Internal Priority Value and Drop Resilience) as the frame moves through the device, and encounters resource contention and possibly be conditioned (e.g.

policing). Internally at any queuing point, frames can be placed in the appropriate queue based on the Internal Priority Value and can be dropped based on the Drop Resilience.

Internal Priority Value (IPV) is a 3-bit attribute assigned to a frame which is used for certain scheduling decisions in subsequent receive processing stages. A higher IPV value generally indicates higher priority, assuming strict priority scheduling is applied. The lowest priority is 0 and the highest is 7.

Drop Resilience (DR) is a 2-bit attribute assigned to a frame which is used to determine the likelihood of dropping the frame if there is congestion. A higher DR value generally indicates being more discardable. For example:

- Drop resilience 0

  Committed frames. With proper allocation of buffer and bandwidth resources within the network, it is possible to guarantee a lossless and timely delivery of a committed frames. In a "colored" mapping scheme such as IEEE 802.1Qci flow metering, this corresponds to Green.

- Drop resilience 1

  Normal frames. In a "colored" mapping scheme such as IEEE 802.1Qci flow metering, this corresponds to Green.

- Drop resilience 2

  Non-committed frames. At sign of congestion, these frames can be dropped but should otherwise be delivered. In a "colored" mapping scheme such as IEEE 802.1Qci flow metering, this corresponds to Yellow.

- Drop resilience 3

  Violating frames. These frames are delivered on a "best effort" basis. In a "colored" mapping scheme such as IEEE 802.1Qci flow metering, this corresponds to Red.

Received frames are initially assigned an IPV and DR based on the setting for the port (PQOSMR[DIPV] and PQOSMR[DDR]).

The PCP and DEI from one of the sets of VLAN tag data may be used to override the default settings. This is achieved by setting PQOSMR[VE] to 1 to enable the use of VLAN tag data to determine IPV and DR and configuring PQOSMR[VS] to determine which VLAN tag (inner/outer) to be used.

If the selected VLAN tag is valid (VLAN classification output), the PCP and DEI from the selected VLAN tag data are used to determine IPV and DR based on the configured per port mapping VLANIPVMP*a*R0/1 and VLANDRMP*a*R, where *a* identifies a mapping profile. The profile to use (i.e *a*) is specified in PQOSMR[VQMP].

IPV and DR values for the frame may also be changed by subsequent processing stages.

53.4.2.3.1.3.6 Ingress Port Filtering

This stage implements the ingress port filtering functions. There are two functions being performed; denial of service checks function and the ingress port filtering lookup function.

The denial of service (DOS) checks function is first executed to determine whether admit or drop incoming frames. The denial of service checks to be performed are specified in DoS L2 configuration register (DOSL2CR). Whether to perform the L2 DOS checks is specified in register PCR[L2DOSE].

If the frame hasn't been discarded by the previous function, the ingress port filtering lookup function is then executed by examining the frame header information to determine on whether to deny or allow a frame. It may also be used to override internal QoS attributes associated with the frame and set parameters (e.g. Rate Policer Entry ID, stream ID, SI bitmap) for the subsequent frame processing functions.

The ingress port filtering lookup function must be enabled to be executed (i.e. PIPFCR[EN] set to 1). If the ingress port filtering lookup function is disabled, the ingress port filtering lookup function is by-passed, and frames are passed to the next frame processing function.

The ingress port filtering lookup function involves performing a lookup against the ingress port filter table. The ingress port filter table is defined as a ternary match table. In a ternary match table, all entry key matching fields are composed of pairs of data and mask values, to support a per-bit ternary match capability. For each {data, mask} bit pair, the following combinations are supported:

- 00 = always match (X)
- 01 = match on 0
- 11 = match on 1
- 10 = always match (X); note that the table management query operation will always be returning 0 for any data bits which have their corresponding mask bit set to 0, regardless of whether they are set to 0 or 1 by the table management add or update operations.

In other words, to look for a match on a particular bit, set its mask bit to 1, and its corresponding data bit to the desired matching value, 0 or 1. A mask value of all 1's for a particular field means that field is required. To make a bit don't care, set its mask bit to 0.

In a ternary match table, a single lookup key can be matched by multiple table entries, and thus every entry must have a priority (or precedence) assigned by the software. When multiple entries match, the entry with the highest priority, is returned as the matching entry.

If no entry matches the frame, the frame is allowed in, and passed to the next frame processing function.

The ingress port filter table is implemented using a Ternary Content Addressable Memory (TCAM).

For programming of the table entries, see Ingress Port Filter Table

The ingress port filter table entry includes the following matching key fields (each field is bit-wise maskable):

- Source and destination MAC address
- Outer VLAN tag PCP, DEI and VID
- Inner VLAN tag PCP, DEI and VID
- Payload EtherType, located immediately after the source MAC address or VLAN tag(s)/R-TAG/HSR (if any).
- IPv4 source address and destination IPv4 address
- IPv6 source address and destination IPv6 address
- Protocol field present; for IPv6 packets the Protocol key (called the "Next Header" in IPv6) is not extracted if the packet has IPv6 extension headers.
- Protocol
- IP DSCP
- TCP/UDP header source and destination port
- Payload (up to 24-bytes)
  - When no IP header is present, the payload is matched against the frame bytes located immediately after the payload EtherType. The payload EtherType is located immediately after the source MAC address or VLAN tag(s)/R-TAG/HSR (if any).
  - When IP header is present and no TCP or UDP header is present, the payload is matched against the frame bytes located immediately after the IP header except when IPv4 options or IPv6 extensions are present. For the latter, payload matching starts at the beginning of the IPv4 Options/IPv6 Extensions header.
  - When TCP or UDP header is present, the payload is matched against the frame bytes located immediately after the first 4 bytes of the TCP/UDP header.
- Frame attributes fields
  - o Outer VLAN tag present
  - o Inner VLAN tag present
  - o IP header present
  - o Indication of whether IP header is v6 or v4
  - o TCP header present

- UDP header present
- Other L4 present (not TCP or UDP)
- IPv4 option / IPv6 extension present
- Sequence Tag present
- Magic packet

Note that when hardware constructs a table lookup key for a frame header field for which the frame header is not present in the frame, it sets the key field to zero. If zero corresponds to a valid value in a frame header field, this could lead to a false positive match. To avoid these false positive matches, the present frame header (or field) 1-bit key matching fields are defined in the table entry to explicitly specify whether the presence of a frame header (or field) must be present or not present in a frame header, for an entry to match.

Following are the possible actions when an ingress port filter entry is matched:

- Filtering action, one of the following:
  - Discard
  - Permit
- Modifying internal QoS attributes (IPV and DR)
- Indicating that the frame should be treated as a Wake-on-LAN frame (valid if IPFCAPR[WOL]=1)
- Setting internal parameter, one of the following
  - Enabling policing with a specified Rate Policer Entry ID
  - Specifying a Ingress Stream Entry ID along with lookup precedence
  - Setting a pre L2 filtering SI bitmap that will be used by the L2 filtering function to determine the final SI bitmap.

53.4.2.3.1.3.7 Stream Identification and Filtering

This stage implements the stream identification and filtering functions. Steam identification mainly involves classifying incoming traffic into a locally-significant identifier (global to the device), that identifies a stream within the device. This identifier is referred to as the Ingress Stream Entry ID (IS_EID). The stream identification function is implemented as per IEEE 802.1CB (Frame Replication and Elimination for Reliability (FREF)). Note that IEEE 802.1Qci (Per-Stream Filtering and Policing) exploits the stream identification functions specified in IEEE 802.1CB. The stream filtering function identifies the actions/functions to be applied to frames belonging to the identified stream. Namely, the main actions/functions are:

- Whether a frame is accepted or discarded.
- Maximum service data unit size defined for the stream as per IEEE 802.1Qci (Per-Stream Filtering and Policing).
- Stream gate instance and rate policer instance to be used to police the stream as per IEEE 802.1Qci (Per-Stream Filtering and Policing). The stream gating and rate policer functions are applied later in the ingress processing pipeline after it has been determined where to forward the frame.
- FREF (IEEE 802.1CB) sequence generation function. This function applies only to the switch.
- Stream forwarding function; the port(s) to which the frame should be output are provided in the stream context along with the egress packet processing actions to be applied to packets for each of the outgoing port(s); when the stream forwarding function is selected, the 802.1Q bridge forwarding function is by-passed. This function applies only to the switch.
- Ingress frame modifications such as ingress VLAN translation and IEEE 802.1CB active stream identification (replacing DMAC and/or VLAN). This function applies only to the switch.
- The Ingress Stream Counter ID that indicates the Ingress Stream Count table entry to be used to count matched frames and count discarded frames.

53.4.2.3.1.3.7.1 Stream Identification

Stream identification function mainly involves performing a series of table lookups to determine the Ingress Stream Entry ID (IS_EID), which in turn is used to access the Ingress Stream table to retrieve the action(s) to be applied to the frame. A default IS_EID, can be specified on a per port basis in PISIDCR register. Following are the lookup tables that can yield the IS_EID.

- **Ingress Port Filter table** - IS_EID can be specified in an Ingress Port Filter entry along with its relative precedent resolution value. Specifying IS_EID in ingress port filter entry is optional. If specified, its relative precedent resolution value determines if further lookups are required for stream identification.
- **Ingress Stream Identification table** - The Ingress Port Filter table lookup (prior stage) is followed up by a series of up to 2 hash based exact match table lookups into the Ingress Stream Identification table. The key used for stream identification exact match lookup is specified through a "key construction rule". A key construction rule specifies how the lookup key should be constructed. Specifically, it specifies metadata, specific frame header fields, and payload data that should be concatenated to form a key for the table lookup. A key construction rule is specified through a set of registers. For the switch, there are 4 key construction rules available for the entire switch, and are identified by the decimal number (0-3). For ENETC, there are 2 key construction rules per ENETC instance, and are identified by the decimal number (0-1). Note that the key construction register names imbeds the ID of the key construction rule that they are associated with. For example, ISIDKCaCR0, "a" represents the rule ID (decimal) that the register is associated with. Assignment of key construct rules to Ingress Stream Identification table lookups is achieved through key construction rule profiles. A key construction rule profile contains 2 construction key rules (one for the first exact match lookup and one for the second exact match lookup). Key construction rule profiles are derived through a static hard-wired assignment; profile 0 includes rules ID 0 and 1, profile 1 includes rules ID 2 and 3, and so on if it needs to be expanded in the future. For the switch, which of the two key construction rule profiles to use, is configurable on per switch port basis (register PISIDCR[KCPAIR]). For ENETC, key construction rule profile 0 is always used.

The following details the sequence of table lookups for determining the IS_EID (also illustrated in Figure 337).

1. If the Ingress Port Filtering function has yield an IS_EID and its relative precedent resolution value indicates that it has the highest precedence level (set to 0), then this IS_EID is selected, else go to step #2.
2. The first of the two hash based exact match lookups is performed (if enabled (PISIDCR[KC0EN]=1)) based on the key construction rule specified for this lookup. If a matching entry is found, the IS_EID specified in the entry is then selected. If this lookup is not enabled or a matching entry is not found and if the Ingress Port Filtering function has yield an IS_EID with a relative precedent resolution value falling between first and second exact match lookup (i.e. value set to 1), then this IS_EID is selected. Otherwise go to step #3.
3. The second of the two hash based exact match lookups is performed (if enabled (PISIDCR[KC1EN]=1)) based on the key construction rule specified for this lookup. If a matching entry is found then the IS_EID specified in the entry is selected. If this lookup is not enabled or a matching entry is not found and the Ingress Port Filtering function had yield an IS_EID then this IS_EID is selected. Else the default IS_EID is selected.

Note that if the selected IS_EID is NULL, the ingress stream frame processing will not be performed and the frame is passed on to the next stage of processing.

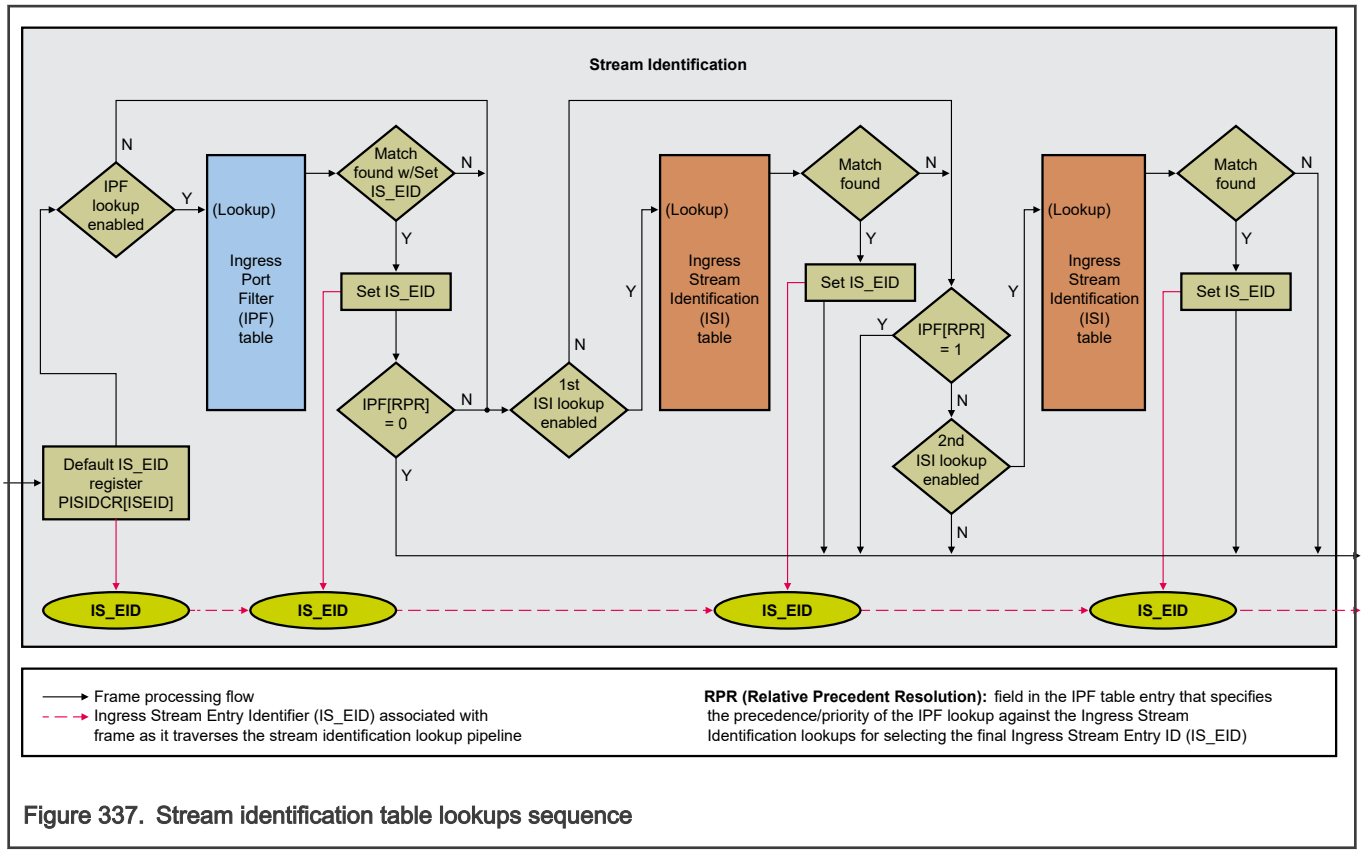


Figure 337. Stream identification table lookups sequence

Following registers are associated with lookups in the Ingress Stream Identification table.

- Ingress Stream Identification Capability Register (ISIDCAPR); indicates number of key construction rules, maximum key size, and so on.
- Ingress Stream Identification Hash Table Capability Register (ISIDHTCAPR); indicates which configuration access methods are supported.
- Switch registers for ingress stream identification key construction rules are noted below.
  - Ingress Stream Identification Key Construction a Operational register (ISIDKC*a*OR); indicates the number of entries in the Ingress Stream Identification table for particular key construction rule, where *a* =0..3 represents the key construction rule ID.
  - Ingress Stream Identification Key Construction Configuration Register 0 (ISIDKC*a*CR0) , where *a*=(0..3) represents the key construction rule ID.
  - Ingress Stream Identification Key Construction a Payload Field 0 Configuration Register (ISIDKC*a*PF0CR), where *a*=(0..3) represents the key construction rule ID.
  - Ingress Stream Identification Key Construction a Payload Field 1 Configuration Register (ISIDKC*a*PF1CR), where *a*=(0..3) represents the key construction rule ID.
  - Ingress Stream Identification Key Construction a Payload Field 2 Configuration Register (ISIDKC*a*PF2CR), where *a*=(0..3) represents the key construction rule ID.
  - Ingress Stream Identification Key Construction a Payload Field 3 Configuration Register (ISIDKC*a*PF3CR), where *a*=(0..3) represents the key construction rule ID.
  - Port Ingress Stream Identification Configuration Register (PISIDCR); used to enable/disable exact match ingress stream identification lookups and to specify which key construction rule profile to use.
- ENETC registers for ingress stream identification key construction are noted below.

- Ingress Stream Identification Key Construction a Operational register (ISIDKC*a*OR); indicates the number of entries in the Ingress Stream Identification table for particular key construction rule, where *a* =0..1 represents the key construction rule ID.
- Ingress Stream Identification Key Construction a Configuration Register 0 (ISIDKC*a*CR0), where *a*=(0..1) represents the key construction rule ID.
- Ingress Stream Identification Key Construction a Payload Field 0 Configuration Register (ISIDKC*a*PF0CR), where *a*=(0..1) represents the key construction rule ID.
- Ingress Stream Identification Key Construction a Payload Field 1 Configuration Register (ISIDKC*a*PF1CR), where *a*=(0..1) represents the key construction rule ID.
- Ingress Stream Identification Key Construction a Payload Field 2 Configuration Register (ISIDKC*a*PF2CR), where *a*=(0..1) represents the key construction rule ID.
- Ingress Stream Identification Key Construction a Payload Field 3 Configuration Register (ISIDKC*a*PF3CR), where *a*=(0..1) represents the key construction rule ID.
- Port Ingress Stream Identification Configuration Register (PISIDCR); used to enable/disable exact match ingress stream identification lookups.

The Ingress Stream Identification table is managed using command messages. See Ingress Stream Identification Table for the switch and Ingress Stream Identification Table for ENETC.The Ingress Stream Identification table entry includes the following matching key fields:

- Key construction rule ID (referred to as KEY_TYPE in the Ingress Stream Identification table)
- Source port ID; applicable to the switch only
- Source Port Masquerading (SPM); applicable to the switch only
- Frame portion of the key (up to 16 bytes) frame header key construction is specified in ISIDKC*a*CR0 and frame payload key construction is specified in ISIDKC*a*PF*b*CR , where *a* represents the rule ID and *b* represents the payload field.

The frame portion of the key format (frame key) is specified by the aforementioned key construction rule registers. The frame key entered in the Ingress Stream Identification table entry has to be in packed form in the order they appear in the frame with each field formatted in big endian format (order they are received from the link). If the constructed key is less than 16 bytes then the frame key entered in the Ingress Stream Identification table entry, must be padded with zeros for the rest of the bytes.

Protocol header data or payload data has to be added to the key in the Ingress Stream Identification entry if indicated as 'present' in key construction register. Format of protocol header data or payload data in 16 byte key (referred to as Key) is specified below.

- Source port and SPM are included in the key if respective 'present' bit is set in the key construction register. These values are part of the reserved area in the hash key and do not add bytes to the frame portion of the key.
- Following are the options for the Ethernet header:
  - If SMAC or DMAC is present in key construction register then each of these adds 6 bytes of data to the key. If both are present then DMAC has to follow SMAC.
  - If VID or PCP are indicated as present in key construction register then the following is added to the key: VID only, PCP only, VID and PCP, or 'no VLAN tag present in the frame'. VID and PCP can be specified for both inner and outer VLANs.
  - A one-byte frame attribute code point is included in the key indicating whether there is no 802.1CB R-TAG/HSR tag, 802.1CB draft 2.0 R-TAG present, 802.1CB R-TAG present or HSR tag present.
  - If EtherType is present in key construction register then an EtherType is added to the key. The EtherType extracted is located immediately after source MAC address or VLAN tag(s)/R-TAG/HSR (if any).
  - ISIDKCaPFbCR registers specify the key format for the payload data (where 'a' indicates the key construction rule used for the exact match lookups and 'b' indicates the payload field). The key format is big endian (network byte order).

53.4.2.3.1.3.7.2 Per-Stream Filtering

Based on the selected IS_EID, the Ingress Stream table entry is read from Ingress Stream table to obtain the actions to be performed for the received frame:

- Forwarding options for switch function - Discard frame, redirect/copy frame to switch management port, use 802.1Q bridge forwarding function or use stream forwarding which by-passes the 802.1Q bridge forwarding function.
- Forwarding options for ENETC function - Discard frame, forward with SI bitmap or forward without SI bitmap set.
- Specify whether to perform the maximum Service Data Unit (SDU) check. See Maximum SDU Check for more details.
- Override IPV, DR
- The frame is to be mirrored to the mirror destination specified in the IMDCR0 register (applicable to the switch only).
- Enable timestamp capture if not already enabled (applicable to the switch only).
- Enable/disable Ingress Sequence Generation (applicable to the switch only).
- Enable/disable Ingress Stream Gating operation. If enable, the stream gate instance identifier is provided to indicate the stream gate instance to be used.
- Option to override Rate Policer Entry ID.
- Option to disable cut through frame transfer on a specific destination port or on all destination ports (applicable to the switch only).
- Specify egress packet processing actions (applicable to the switch only)
- Specify the Ingress Stream Counter ID to count matched frames and count discarded frames. Discard counts are maintained for maximum SDU drops, Stream Gating frame drops and Rate Policing frame drops. If Ingress Stream Counter ID is set to NULL, no Ingress Stream Counter table entry is updated. If the entry referred to by the Ingress Stream Count ID is not found in the Ingress Stream Count table (entry has not been added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0).
- Option to specify performing an Ingress Stream Filter table lookup; to support a PSFP (IEEE 802.1Qci) stream filtering function that uses the 3-bit PCP value (priority) of the received frame and the stream handle (i.e. Ingress Stream Entry ID) to select a stream filter. Note that the Ingress Stream table is used to support stream filters that require an exact match only on the stream Id (PCP is wild-carded). Note that to support stream filters that requires exact match only on the PCP (stream Id is wild-carded), multiple Ingress Stream table entries must be created for a given stream, one for each of the PCP value that requires matching. For the latter, the PCP must be included as one of the parameter/key field to determine the Ingress Stream Entry ID (IS_EID).
- Option to specify Ingress Frame Modification Entry ID (IFM_EID) to perform ingress frame modification operation (applicable to the switch only).

If a frame is discarded, in addition to updating the relevant discard counts in the Ingress Stream Count table, discard count and discard reasons are updated in PRXDCR, PRXDCRRR and in PRXDCRR0/1 registers.

The Ingress Stream table is an index table. There is a single instance of this table for the switch and an instance per ENETC function. The IS_EID is used as the index to this table. For more details, see Ingress Stream Table for the switch and Ingress Stream Table for ENETC. Ingress stream table capabilities and configuration are specified in following registers:

- Ingress Stream Index Table Capability Register (ISITCAPR)
- Ingress Stream Index Table Memory Allocation Register (ISITMAR)
- Ingress Stream Index Table Operational Register (ISITOR)

If the entry referred to by the selected IS_EID (non-NULL value) is not found in the Ingress Stream table (entry has not been added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0). If the selected IS_EID is the default IS_EID specified in PISIDCR[ISEID], and is set to a value which indexes outside of the available Ingress Stream table entries, the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along

with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0). Note that IS_EIDs configured in tables (Ingress Port Filter table, Ingress Stream Identification table), are checked at configuration time to ensure that the ID falls within the range of valid indexes. If the IS_EID is out of range and is not set to NULL, the table configuration operation is not performed. If the IS_EID is set to NULL, the stream filtering function is by-passed.

53.4.2.3.1.3.7.3 Per-Stream, Per-Priority Filtering

To support a PSFP (IEEE 802.1Qci) stream filtering function that uses the 3-bit PCP value (priority) of the received frame and the stream handle (i.e. Ingress Stream Entry ID) to select a stream filter, an additional lookup (i.e. against the Ingress Stream Filter table) must be performed to obtain the PSFP (IEEE 802.1Qci) filtering and policing actions. If a lookup is performed against the Ingress Stream Filter table, the PSFP (IEEE 802.1Qci) filtering and policing actions from the Ingress Stream table entry are overridden by the actions from the matched Ingress Stream Filter table entry. If no match is found in the Ingress Stream Filter table, processing moves on with the actions from the matched Ingress Stream table entry.

As stated earlier on, in the case where the stream handle (i.e. Ingress Stream Entry ID) is used to select a stream filter, the Ingress Stream table entry contains the PSFP (IEEE 802.1Qci) filtering and policing actions that are to be applied to frames received on this stream. Furthermore to support stream filters that requires exact match only on the PCP (stream Id is wild-carded), multiple Ingress Stream table entries must be created for a given stream, one for each of the PCP that requires matching. Additionally, the PCP must be included as one of the parameter/key field to determine the Ingress Stream Entry ID (IS_EID).

The Ingress Stream Filter table is a hash table accessed using the IS_EID specified in the Ingress Stream table entry and PCP in the VLAN tag of the received frame. For more details, see Ingress Stream Filter Table for the switch and Ingress Stream Filter Table for ENETC.

If a match is found, the following actions can be specified:

- Options to override IPV, DR.
- Option to mirror frame to the mirror destination specified in the IMDCR0 register (applicable to the switch only).
- Option to disable cut through forwarding of frame if not already disabled (applicable to the switch only).
- Enable timestamp capture if not already enabled (applicable to the switch only).
- Option to enable/disable Ingress Stream Gating. If enable, the stream gate instance identifier is provided to indicate the stream gate instance to be used.
- Option to override Rate Policer Entry ID.
- Specify the Ingress Stream Counter ID to count matched frames and count discarded frames. Discard counts are maintained for maximum SDU drops, Stream Gating frame drops and Rate Policing frame drops. If Ingress Stream Counter ID is set to NULL, no Ingress Stream Counter table entry is updated. This field overrides the Ingress Stream Counter ID field specified in the previous matched Ingress Stream table entry. If the entry referred to by the Ingress Stream Count ID is not found in the Ingress Stream Count table (entry has not been added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0).
- Specify the maximum SDU check. See the following section for more details.

If a frame is discarded, in addition to updating the relevant discard counts in the Ingress Stream Count table, discard count and discard reasons are updated in PRXDCR, PRXDCRRR and in PRXDCRR0/1 registers.

53.4.2.3.1.3.7.4 Maximum SDU Check

Maximum SDU check is performed prior to policing where frame will be dropped if frame size exceeds maximum SDU. Note that the maximum SDU check is not performed, if the frame is being forwarded in the cut-through mode of operation, to at least one egress port. This means that the maximum SDU check is not performed for a frame that is replicated, where some copies are forwarded as cut-through frames while others are forwarded as store-and-forward frames. A value of zero disables the maximum SDU check.

53.4.2.3.1.3.8 L2 filtering

L2 filtering consists of performing a series of lookups based on SMAC, DMAC and VLAN, to decide whether to accept the frame and to provide the set of Station Interfaces (SIs) eligible to receive the frame.

L2 filtering is performed in two stages:

1. MAC Address filtering
2. VLAN filtering

Each received frame is associated with an SI bitmap which is used to provide to HTA the set of Station Interfaces (SIs) eligible to receive the frame within a port. Each bit of the bitmap corresponds to an SI on a given port. Least significant bit of the bitmap corresponds to the smallest SI number; i.e. bit offset 0 of the bitmap corresponds to SI numbered 0, bit offset 1 to SI numbered 1, and so on. The SIs to which the frame is to be delivered, are identified by having their corresponding bit set to 1 in the bitmap. The SI bitmap is initially set to 0 and updated while the frame is processed by the various L2 filters.

When operating in the Wake-on-LAN (WoL) mode, all frames are first filtered according to its WoL frame type. The MAC Address filtering and VLAN filtering stages are by-passed. See L2 Filtering with Wake-on-LAN, for more details.

53.4.2.3.1.3.8.1 MAC address filtering

Following registers and lookup tables are used in MAC address filtering:

- **SI Primary MAC Address registers (per SI)**

  Each SI has a primary MAC address configured in registers PSI*a*PMAR0/PSI*a*PMAR1 (where a is the SI number).

- **Exact Match MAC address table (per port)**

  An exact match table that is shared between SIs. Each table entry includes a MAC address that may be unicast or multicast and a result containing an SI bitmap identifying the associated SI(s) with the MAC address. If more than one entry are present with the same MAC address, the matched entry with the lowest table index is returned.

  This table is configured using command messages (see MAC Address Filter Table). Take note that the Primary MAC addresses don't need to be added into this table.

- **Unicast MAC hash table (per SI)**

  A 64 entry (1-bit wide entry) hash table used to match against the hashed unicast MAC addresses.

  The unicast MAC address to be matched is hashed into a 6-bit index which in turn is used to access a bit in the unicast MAC hash table. If the corresponding bit is set to one, then this is considered a successful match.

  The unicast MAC hash table is defined on per SI basis, via PSI*a*UMHFR0/1 registers, where a is the SI number.

- **Multicast MAC hash table (per SI)**

  A 64 entry (1-bit wide entry) hash table used to match against the hashed multicast MAC addresses.

  The multicast MAC address to be matched is hashed into a 6-bit index which in turn is used to access a bit in the multicast MAC hash table. If the corresponding bit is set to one, then this is considered a successful match.

  The multicast MAC hash table is defined on per SI basis via the PSI*a*MMHFR0/1 registers, where a is the SI number.

For the hash tables, as there is no subsequent match against the key used to generate the (hash) index into the table these are “imperfect” filters and have false positives. Software drivers relying on these filters must be aware of this and do further filtering to ensure that only frames of interest are accepted. This filtering is an assist only provided to improve performance.

Software which configures these hash tables, inserting and removing entries, must track the actual addresses or IDs that are inserted so that removals can be handled properly. Multiple addresses or IDs may require the same bit to be set in the table. On removal of an address/ID the corresponding bit should be cleared only if no other address/ID requires that bit to be set.

See Hash Functions for Imperfect Match (“hash”) Filtering for more details on the hash functions.

L2 filtering is performed according to the following steps:

1. **Initialization** - The frame metadata SI bitmap is initially set to 0 and updated while the frame is processed by the various L2 filters.

2. **Destination unicast or multicast MAC address exact match lookup** – Destination MAC address extracted from the frame is matched against each SI Primary MAC address register PSI*a*PMAR0/PSI*a*PMAR1 (where a is the SI number). If a match is found, the corresponding bit in the SI bitmap attribute of the frame metadata is set. In parallel, an exact match lookup is performed on the Exact Match MAC address table using the Destination MAC address (unicast or multicast) extracted from the frame. If a match is found, then for every bit set in the SI bitmap of the matched entry result, corresponding bit in the SI bitmap attribute of the frame metadata is set
3. **Broadcast** - If the frame is a broadcast frame then the frame metadata SI bitmap is updated with the bits set for each SI that does not have broadcast receipt disabled (broadcast receipt is disabled if SIMR[RNBM] bit set to 1).
4. **Destination unicast MAC address hash match lookup** – If the frame is a unicast frame and the prior steps yielded no SIs, for every SI, the Destination unicast MAC address extracted from the frame is hashed into an index to access the per SI Unicast MAC hash table ([PSI*a*UMHFR0/1] registers). If the accessed bit is set to 1, then the corresponding bit in the SI bitmap attribute of the frame metadata is set to 1.
5. **Destination multicast MAC address hash match lookup** – If the frame is a multicast frame and the prior steps yielded no SIs, for every SI, the Destination multicast MAC address extracted from the frame is hashed into an index to access the per SI Multicast MAC hash table ([PSI*a*MMHFR0/1] registers). If the accessed bit is set to 1, then the corresponding bit in the SI bitmap attribute of the frame metadata is set to 1.
6. **Unicast Promiscuous** - If the frame is a unicast frame then the frame metadata SI bitmap is updated with the bits set for each SI that has unicast promiscuous enabled (PSIPMMR[SI*a*_MAC_UP] bit is set to 1).
7. **Multicast promiscuous** - If the frame is a multicast frame then the frame metadata SI bitmap is updated with the bits set for each SI that has multicast promiscuous enabled (PSIPMMR[SI*a*_MAC_MP] bit is set to 1).
8. **Source MAC address pruning** - Source MAC address pruning is used to discard frames that originated from an SI and got forwarded back to this device by an external switch. Source MAC address extracted from the frame is matched against each SI Primary MAC address register PSI*a*PMAR0/PSI*a*PMAR1 (where a is the SI number). If a match is found the corresponding bit in the SI bitmap attribute of the frame metadata is set to 0. In parallel, an exact match lookup is performed on the Exact Match MAC address table using the Source MAC address. If a match is found, then for every bit set in the SI bitmap of the matched entry result, corresponding bit in the SI bitmap attribute of the frame metadata is set to 0. Source MAC address pruning is enabled by setting the PSI*a*CFGR0[SPE] bit to 1 (where a is the SI number).
9. **Unicast MAC address reject** – If the frame is a unicast frame then the frame metadata SI bitmap is updated with the bits reset for each SI that does have unicast receipt disabled (unicast receipt is disabled if SIMR[RNUM] bit set to 1).
10. **Multicast MAC address reject** - If the frame is a multicast frame then the frame metadata SI bitmap is updated with the bits reset for each SI that does have multicast receipt disabled (multicast receipt is disabled if SIMR[RNMM] bit set to 1).
11. **Next Stage - VLAN filtering** - The frame is passed to the next filtering stage (VLAN).

53.4.2.3.1.3.8.2 VLAN filtering

Following lookup tables are used in VLAN filtering:

- **Exact Match VLAN table (per port)**

  An exact match VLAN table shared between SIs. Each table entry includes a VLAN ID and 2-bit TPID, and a result containing an SI bitmap identifying the associated SI(s) with that VLAN (VLAN ID and TPID).

  This table is configured using command messages (see VLAN Address Filter Table) with some entries intended to be used for entries matching the SI-based VLANs. For VLAN removal and insertion, the SI-based VLANs are configured through register PSIaVLANR (where a is the SI number). For VLAN filtering, the SI-based VLANs must be explicitly inserted in the exact match VLAN table. If more than one entry are present with the same VLAN ID, the matched entry with the lowest table index is returned.

  Software is responsible for inserting entries for the SI-based VLANs in the exact match VLAN table. Hardware does not automatically insert these entries. Entries must be reserved for the SI-based VLANs for the various SIs but hardware does

not restrict the use of the table or enforce these "reservations". Software is responsible for ensuring the entries are available if required.

- **VLAN hash table (per SI)**

  A 64 entry (1-bit wide entry) hash table used to match against hashed VLAN ID.

  The VLAN ID to be matched is hashed into a 6-bit index which in turn is used to access a bit in the VLAN hash table. If the corresponding bit is set to one, then this is considered a successful match.

  The VLAN hash table is defined on per SI basis via PSI*a*VHFR0/1 registers, where a is the SI number.

  For the hash table, as there is no subsequent match against the key used to generate the (hash) index into the table these are "imperfect" filters and have false positives. Software drivers relying on these filters must be aware of this and do further filtering to ensure that only frames of interest are accepted. This filtering is an assist only provided to improve performance.

  Software which configures these hash tables, inserting and removing entries, must track the actual addresses or IDs that are inserted so that removals can be handled properly. Multiple addresses or IDs may require the same bit to be set in the table. On removal of an address/ID the corresponding bit should be cleared only if no other address/ID requires that bit to be set.

  See Hash Functions for Imperfect Match (“hash”) Filtering for more details on the hash functions.

VLAN filtering is performed according to the following steps:

1. **Initialization** - The frame metadata SI bit mask is initially set to all 0s. Each bit of the bit mask corresponds to an SI on a given port. Least significant bit of the bit mask corresponds to the smallest SI number; i.e. bit offset 0 of the bit mask corresponds to SI numbered 0, bit offset 1 to SI numbered 1, and so on. VLAN filtering is always on.
2. **VLAN selection** - There is a per port setting (PSIVLANFMR[VS]) selecting inner or outer VLAN tag data derived by the VLAN classification. If the selected VLAN tag (out of VLAN classification) is not valid, then steps 3, 4 and 5 are skipped.
3. **VLAN exact match lookup** - An exact match lookup is performed on the Exact Match VLAN table using the VLAN ID and TPID from the VLAN tag selected in step 1. If a match is found, then for every bit set in the SI bitmap of the matched entry result, corresponding bit in the SI bit mask attribute of the frame metadata is set.
4. **VLAN hash match lookup** - If the prior step yielded no SIs, for each SI, the VLAN ID from the VLAN tag selected in step 1 is hashed into an index to access the per SI VLAN hash table (PSI*a*VHFR0/1 registers). If the bit is set, then the corresponding bit in the SI bit mask attribute of the frame metadata is set.
5. **VLAN Promiscuous** - The frame metadata SI bit mask is updated with the bits set for each SI that has VLAN promiscuous enabled (PSIPVMR[SI*a*_VLAN_P] bit enabled).
6. **Untagged** - If the frame metadata untagged indicator is set or if the selected VLAN tag (out of VLAN classification) is not valid then the frame metadata SI bit mask is updated with the bits set for each SI that does have untagged enabled (PSIPVMR[SI*a*_VLAN_UTA] set to 1).

Result from MAC address filtering (frame metadata SI bitmap) is masked (bitwise "AND" operation) with the result from VLAN filtering (frame metadata SI bit mask).

The resulting frame metadata SI bitmap is "ORed" (bitwise OR operation) with the pre L2 filtering bitmap set by an Ingress Port Filtering table lookup match or an Ingress Stream table lookup match. If the resulting frame metadata SI bitmap is non-zero, the frame is passed to HTA, otherwise the frame is discarded.

53.4.2.3.1.3.8.3 L2 Filtering with Wake-on-LAN

When operating in the Wake-on-LAN (WoL) mode, all frames are first filtered based on whether the frame has been identified as a WoL trigger frame. Any frame not matching the supported WoL frame types is discarded. These frame discards are counted against the port's Rx discard count register (PRXDCR) along with the setting of the Station Interface Filter Discard Reason (SIFDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0).

WoL trigger frames are identified though the Ingress Port Filtering function by adding one or more Ingress Port Filter entries with the "Wake-on-LAN Trigger" action set to "enabled". The Ingress Port Filtering function can match WoL trigger frames based on a mix of L2, L3, and L4 header fields including arbitrary fields (e.g. fields not identified by parsing). It can also match frames that have been identified by the MAC, as "magic packets".

If a WoL trigger frame is detected, a WoL event is generated followed by sending the WoL trigger frame towards HTA by having it queued internally (inside ICM) with the SI bitmap set to the pre L2 filtering bitmap stored in the Ingress Port Filter entry that matched the WoL trigger frame. Subsequent frames resume normal L2 filtering processing.

53.4.2.3.1.3.8.4 Hash Functions for Imperfect Match (“hash”) Filtering

There are 2 different types of imperfect match filtering performed: MAC address filtering and VLAN ID filtering. Both of these will use a simple and relatively small 64 entry tables to determine a “match”. In hardware, these entries will be single bit essentially indicating (potential) match or not. As there is no subsequent match against the key used to generate the (hash) index into the table these are “imperfect” filters and may generate false positive matches.

53.4.2.3.1.3.8.4.1 MAC Address Hash Functions

The figure below shows the format of the Ethernet MAC address. On the wire, transmission order for Ethernet is from the most significant byte to the least significant byte and within a byte the transmission order is from the least significant bit to the most significant bit. The MAC address bits are labeled as they appear on the wire, with bit 0 identifying the first bit of the MAC address on the wire and bit 47 identifying the last bit of the MAC address on the wire.

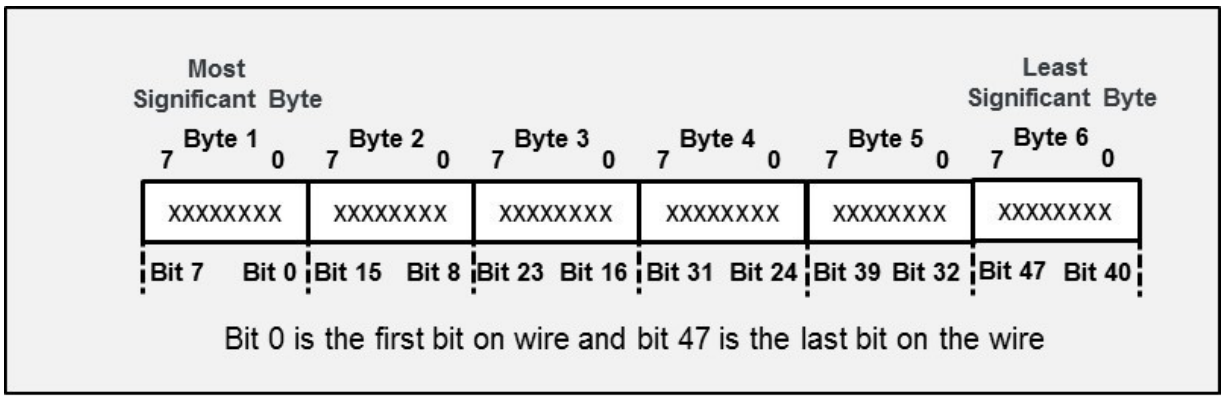


Figure 338. Ethernet MAC Address Format

The upper 24 bits of all MAC addresses (except the broadcast address) consist of an Organizationally Unique Identifier (OUI). The lower 24 bits are then assigned by that organization.

For unicast addresses, these are often assigned sequentially, or near sequentially, to systems such as switches or network adapters as they are manufactured. It is common to see a network with unicast MAC addresses with a small number of OUIs and in some case with changes only in the lower 8 or 16 bits for some addresses if the systems were purchased at or near the same time.

A similar situation can exist for multicast addresses. The OUIs commonly seen include those used for L2 management traffic, IPv6 management traffic, and IPv4 multicast. In some cases, the lower 24 bits are used to represent small numbers (i.e. the upper bits of the 24b are 0). In others, the lower 24b may actually be derived from the unicast MAC address (e.g. for some IPv6 addresses) and thus have the same characteristics as the unicast MAC addresses.

Given the relatively small distribution of “entropy” in these addresses it is important to not include too much input into a single bit of the hash result from a single byte or set of contiguous bits in the address.

The hash function used for imperfect MAC address filtering will use the exclusive-or (XOR) of every 6th bit in the 48b address to generate one bit of the required 6-bit (hash) index. If we number the bits in the MAC address b0 .. b47 then our 6b hash index (h) is:

h0 = b0 XOR b6 … XOR b42

h1 = b1 XOR b7 … XOR b43

h2 = b2 XOR b8 … XOR b44

h3 = b3 XOR b9 … XOR b45

h4 = b4 XOR b10 … XOR b46

h5 = b5 XOR b11 … XOR b47

The hash index h0 is the least significant bit of the index, h5 is the most significant bit of the index.

53.4.2.3.1.3.8.4.2 VLAN ID Hash Function

VLAN IDs are often assigned with relatively small values and over a relatively small portion of the 12b space.

The hash function used for imperfect VLAN ID filtering will XOR every 6th bit in the 12b VLAN ID to generate 1 bit of the required 6 bit hash result. If a VLAN ID is b11 .. b0 where bit 11 is the most significant bit and bit 0 is the least significant bit. Then the hash result (h) is:

h0 = b0 XOR b6

h1 = b1 XOR b7

h2 = b2 XOR b8

h3 = b3 XOR b9

h4 = b4 XOR b10

h5 = b5 XOR b11

The hash index h0 is the least significant bit of the index, h5 is the most significant bit of the index.

53.4.2.3.1.3.9 Policing

This function implements traffic policing as per IEEE 802.1Qci (Per-Stream Filtering and Policing) to protect against time window (stream gating) and bandwidth violations (rate policing).

53.4.2.3.1.3.9.1 Stream Gating

Whether to perform stream gating is determined earlier by the stream identification and filtering function. If stream gating is to be performed, a Stream Gate Instance Entry ID (SGI_EID) is provided to access the Stream Gate Instance table.

The Stream Gate Instance table contains the stream gate configuration and operational information for the switch or an ENETC instance. Each entry contains a set of parameters that defines a single stream gate instance. A stream gate instance includes a gate control list, which specifies the time gates or time slots during which incoming frames from one or more streams will be accepted.

The gate control lists are typically generated offline, and then timely configured on each network device. To support no-downtime gate control list configuration changes, the standard requires that a network device supports two sets of gate control lists:

- **Operational gate control list** – Represents the currently active gate control list.
- **Administrative gate control list** - Provides a means of configuring a new gate control list prior to its installation in a running system. An administrative gate control list becomes operational at Config Change Time as defined in the IEEE 802.1 Qci specification and remains operational till another administrative stream gate control list is installed.

The hardware executes the gate control list cyclically based on a global time. The hardware starts executing the gate control list from the head, and then goes through the gate control entries as time goes on, and after the cycle time is passed, the hardware jumps back to the head of the gate control list.

A stream gate in a gate control list entry can be in one of the two states:

- **Open**: Frames pass through the gate
- **Closed**: Frames do not pass through the gate; frames are discarded.

There are two "Open Gate Check" modes (PSGCR[OGC]) available for determining if a frame can pass through an open gate:

- Only the start of frame needs to be within the open gate interval.
- Both the start and end of the frame need to be within the open gate interval.

For more details on managing this table, see Stream Gate Instance Table for ENETC and switch. If the entry referred to by SGI_EID is not found in the Stream Gate Instance table (entry has not been added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0). If the SGI_EID is set to NULL, the stream gating function is by-passed.

Note that a Stream Gate Instance has to be associated with streams received from the same port. Streams received from different ports cannot use the same Stream Gate Instance. This is not enforced by hardware; considered a software configuration error.

The actual stream gate control lists are stored in separate table called the Stream Gate Control List table, with their entry ID being specified in the Stream Gate Instance table. For a given Stream Gate Instance, only two stream gate control lists may be allocated to the instance. It is recommended that the entry ID for each of these two stream gate control lists be predetermined and fixed by software, and persist till the Gate Stream Instance is no longer in use.

Each Stream Gate Control List entry is of variable size, and is used to hold one stream gate control list. One or up to 256 gates can be specified for a gate control list. A list consists of one configuration 24-byte word plus a variable number of words to configure stream gates. Up to two stream gates can be specified in a word. The Stream Gate Control Entry ID (SGCL_EID) has to be specified in units of words such that it points to the first word in a stream gate control list.

The configuration word and the gate configurations for a gate control list has to be in consecutive words in Stream Gate Control List table. For example, if a gate control list starts at address offset 0x100 in SGCL table and has 9 gates specified, then, the gate list SGCL_EID is 0x100 and the stream gate control list configuration will be at address offset 0x100, configuration for gates 0 and 1 will be in 0x101, configuration for gates 2 and 3 are in 0x102, and so on, and configuration for gate 8 is in 0x105. Hence the gate control list starts at address offset 0x100 and ends at offset 0x105. As such the next gate control list can start at address offset 0x106.

For more details on managing the SGCL table, see Stream Gate Control List for ENETC and the switch.

For stream gating, frame reception timestamp can be aligned to an earlier reference timing point (e.g. at the remote port's transmit timing point). This is achieved by having the frame reception timestamp(s) adjusted by subtracting the delay specified in PSGCR[PDELAY], from the Start of Frame receive timestamp and from the End of Frame receive timestamp (in the case where PSGCR[OGC] is set to 1).

A stream gate control list contains an ordered list of gate operations. Each gate operation specifies the following:

- Gate state (open/close) that determines whether a frame is allowed to pass through the gate or not.
- Internal priority value (IPV).
- Time interval in nanoseconds.
- Interval maximum number of PDU/SDU (Protocol/Service Data Unit) bytes. An optional feature to limit the number of frame bytes that can pass an open gate during its time interval.
- Option to disable cut-through forwarding of frame if not already disabled (applicable to the switch only).

If the frame is being forwarded in the cut-through mode of operation, to at least one port, the initial segment size of the frame is used for checking interval maximum number of bytes. Frame is accepted if the accumulated interval byte count in a gate plus the number of bytes received in the first segment of the frame is less than or equal to the interval maximum number of PDU/SDU bytes, else the frame is discarded. If frame is accepted, accumulated interval byte count gets updated later when the entire frame is received. Note that in cut-through operation, it is possible that the interval maximum number of PDU/SDU bytes can be exceeded (at most by an PDU/SDU size minus the initial segment size).

If the frame is being forwarded in the store-and-forward mode of operation, to all ports identified as the destination for the frame, the frame is accepted if accumulated interval byte count in a gate plus the frame size is less than or equal to the interval maximum number of PDU/SDU bytes. Accumulated interval byte count in a gate is not updated if frame is discarded.

Statistics for stream gating related frame drops are kept in the Ingress Stream Count table.

Gate control list installation is carried on, as defined by IEEE 802.1Qci (Per-Stream Filtering and Policing)

NOTE

The 1588 timer must be initialized and configured before adding any administrative gate control list to a Stream Gate Instance. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer is in normal operation, any change to the 1588 timer configuration (for example, TMROFF_H/L), except for TMR_ADD updates, requires that all Stream Gate Instances with an operation gate control list be deleted, and for Stream Gate Instances that have no operation gate control list, that the administrative gate control list if present, be removed from the Stream Gate instance.

IERB registers:

- Stream Gating Lag Time For Timestamp Refresh Register (SGLTTR). Common for both ENETC and switch. Lag time is in units of nanoseconds. The refresh timestamp timer is used to trigger the hardware to traverse the stream gate control list up to the entry corresponding to the current time minus a lag time (value in SGLTTR) when there is no traffic being received on the port.
- Switch Stream Gate Instance Index Table Memory Allocation Register (S0SGIITMAR)
- Switch Stream Gate Control List Index Table Memory Allocation Register (S0SGCLITMAR)
- ENETC a Stream Gate Instance Index Table Memory Allocation Register (EaSGIITMAR)
- ENETC a Stream Gate Control List Index Table Memory Allocation Register (EaSGCLITMAR)

Switch / ENETC function registers:

- Stream Gate Capability Register (SGCAPR)
- Stream Gate Instance Index Table Capability Register (SGIITCAPR)
- Stream Gate Instance Index Table Memory Allocation Register (SGIITMAR)
- Stream Gate Instance Index Table Operational Register (SGIITOR)
- Stream Gate Control List Index Table Capability Register (SGCLITCAPR)
- Stream Gate Control List Index Table Memory Allocation Register (SGCLITMAR)
- Stream Gate Control List Table Memory Operational Register (SGCLTMOR) Port Rx Discard Count related Registers (PRXDCR, PRXDCRR0/1). When a discard is due to stream gate function the discard count will be updated in this register and following reasons will be specified.
  - [SGCDR] - Stream Gate Closed discard reason.
  - [MSDUEDR] Maximum SDU Exceed discard reason

53.4.2.3.1.3.9.2 Rate Policing

Whether to perform rate policing is determined in prior ingress processing functions. A rate policer instance (or more specifically a Rate Policer Entry ID (RP_EID)) can be specified in the Ingress Port Filter table, Ingress Stream table, and/or Ingress Stream Filter table. When more than one source specifies a rate policer instance, only one instance can be applied to a frame with the following precedent (with 1 having higher precedence):

1. Ingress Stream Filter table
2. Ingress Stream table
3. Ingress Port Filter table

By default (prior to any lookups), the RP_EID is set to NULL, meaning that no rate policing is to be applied to the frame.

Rate policing (or flow metering) is compliant to IEEE 802.1Qci specification.

If rate policing is to be performed, the Rate Policer Entry ID (RP_EID) specified by one of the above table lookups, is used to access the Rate Policer Instance table. For more details on managing this table , see Rate Policer Table for ENETC and Rate Policer Table for the switch. If the entry referred to by RP_EID (non-NULL value) is not found in the Rate Policer table (entry has not been

added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0). If the RP_EID is set to NULL, the rate policing function is by-passed.

The Rate Policer table contains the rate policer configuration and operational information for the entire function (switch or an ENETC function). Each entry contains a set of parameters that defines a single rate policer instance. A rate policer instance controls the amount of traffic (or bytes) that is allowed to go through for a particular received flow of frames.

The MEF (Metro Ethernet Forum) 10.3 technical specification "bandwidth profile algorithm for one flow per envelope processing without token sharing", is used.

When the rate policer instance is configured to operate in the color-aware mode, the incoming frame "color" is indicated through the Drop Resilience (DR) associated with the frame. When the rate policer instance is configured to operate in the color-blind mode, the incoming frame is assumed to always be green for the rate policer. The figure below shows the default decision making process. Some of the default decisions can be modified by configuration, for example setting MR, DOY or NDOR in a rate policer instance will change the default packet marking and drop decisions. Dropped frames by a rate policer instance are counted in against the port's Rx discard count register (PRXDCR), the Rate Policer table entry drop counter and the Ingress Stream Count table entry counter.

If the frame is being forwarded in the cut-through mode of operation (switch only), to at least one port, the initial segment size of the cut-through frame is used by the rate policer for making the decision to declare an arriving frame as Green, Yellow, or Red and then based on the color assigned to the frame, whether the frame is dropped or marked (overwriting the Drop Resilience (DR) assigned to the frame). If the frame is accepted, the rate policer byte credits get updated as data is received (cut-through segment transfers). If the total frame length turns out to be greater than the number of tokens in the bucket that was used to declare the color of the frame, then the bucket token count will go negative (effectively borrowing future tokens). This has a side effect of increasing the effective committed or excess burst size by up to a maximum frame size minus the size of the initial cut-through frame segment.

NOTE

The timer function must be configured and enabled before any rate policer instance is enabled (see Rate Policer table entry Functional Enable Element Data). Either the default nanosecond timer or the 1588 timer can be used to generate the required nanosecond resolution time. The default nanosecond timer is configured and enabled by default out of reset. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer is in normal operation, any change to the 1588 timer configuration (for example, TMROFF_H/L), except for TMR_ADD updates, requires that all rate policer instances be disabled (see Rate Policer table entry Functional Enable Element Data).

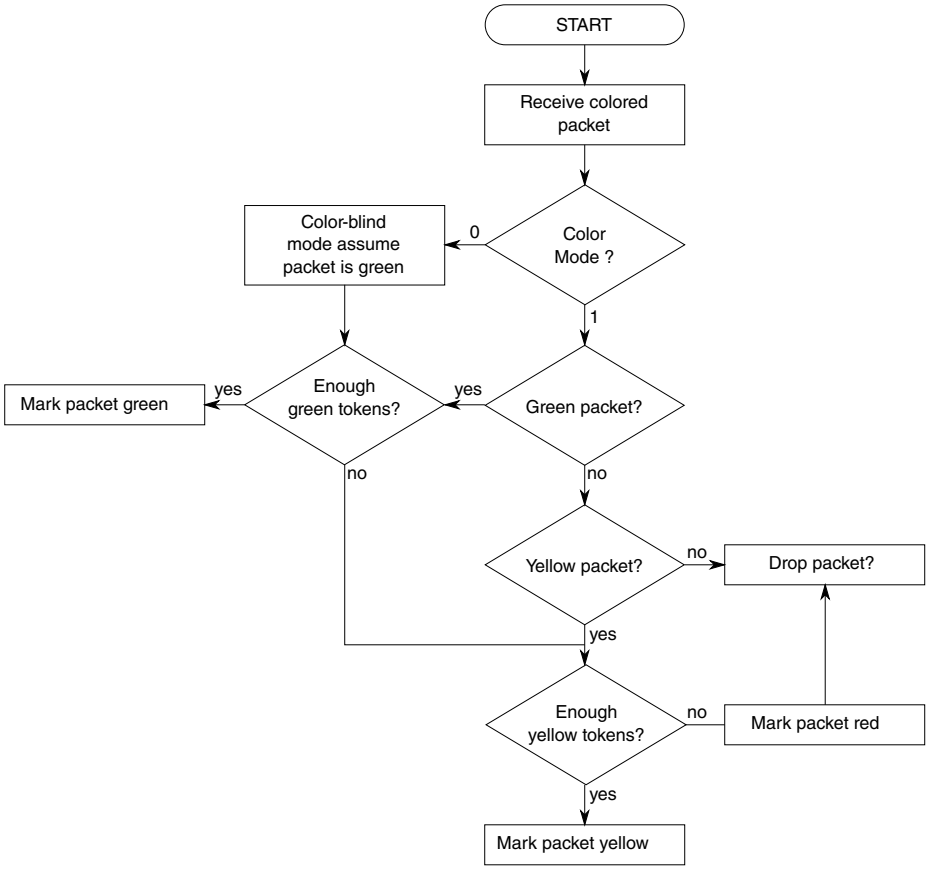


Figure 339. Bandwidth Profile Coloring Flow Diagram

53.4.2.3.1.3.10 Ingress Congestion Manager (ICM)

The Ingress Congestion Manager (ICM) block provides the main buffering/queuing for frames waiting to be scheduled for processing by the HTA Receive. ICM is also responsible for implementing drop decisions and when to start pausing the other end of link when received frames are starting to consume too much of the ENETC receive shared internal buffer memory due to downstream congestion, for example endpoint traffic encountering congestion at the Host Transfer Agent due to slow host system memory access.

The congestion control discipline employed by the ICM is to drop a received frame already queued inside ICM so that when a frame arrives from the MAC, it is always accepted (that is never dropped). This approach provides optimal usage/efficiency of the ENETC receive shared internal buffer memory where no memory space is ever held idle. Ports that need to use more shared internal buffer memory for receive can use more than their minimal allocation, borrowing from other quiet ports. When the ENETC receive shared internal buffer memory consumed by a port is reaching its maximum port usage level or that the ENETC receive shared internal buffer memory is becoming full, internal memory is reclaimed by dropping frames already queued based on the port receive internal memory usage and congestion resiliency.

By default, the amount of internal memory available for all ports is equally shared between all ports. There is an option to explicitly configure a guaranteed amount of internal memory for receive buffering on per port basis. This is achieved by specifying a receive entitlement for each port via the IERB ENETC *a* receive memory buffer entitlement register (E*a*RXMBER). The value configured in E*a*RXMBER is reflected in the ENETC instance base port receive memory buffer entitlement register (PRXMBER). To ensure that no drop occurs when port usage is below its receive entitlement, the sum of all receive entitlements must not exceed the total size of the ENETC receive shared internal buffer

memory is configured via the IERB ENETC receive shared memory buffer allotment register (ERSMBAR). The value configured in ERSMBAR is reflected in the NETC global ENETC shared memory ENETC receive buffer capability register (SMERBCAPR).

When a port exceeds its configured receive entitlement, it becomes eligible for dropping when the ENETC receive shared internal buffer memory becomes full. The ENETC received shared internal buffer memory usage is indicated in the shared memory ENETC receive buffer operational register 0 (SMERBOR0). Note that if there is no receive entitlement configured for a port, then this port is always considered eligible for dropping. Reclaimed memory is obtained by dropping frames first from the port that exceeds the most its receive entitlement. Within a port, frames with a lower congestion resiliency are dropped first. If more than one scheduling priority queue within a port have frames at the same lower congestion resiliency, frames are dropped from the lower scheduling priority queue first. When the port receive usage exceeds its configured limit, frames are dropped from the port till its usage is no longer exceeding its configured limit. Frames with a lower congestion resiliency are dropped first. If more than one scheduling priority queue have frames at the same lower congestion resiliency, frames are dropped from the lower scheduling priority queue first. This limit is configured via the IERB ENETC a receive memory buffer limit register (E*a*RXMBLR). The value configured in E*a*RXMBLR is reflected in the ENETC instance base port receive memory buffer limit register (PRXMBLR). This limit defines an upper-bound limit for a given port receive internal memory usage. The ENETC port received shared internal buffer memory usage is indicated in the ENETC instance base port receive buffer count register (PRXBCR).

Dequeue operations are performed by ICM when the downstream (i.e. HTA receive) indicates its readiness to accept incoming frames.

Dequeue scheduling first selects which port to service next. Scheduling among ports is performed using a byte-based weighted fair scheduling algorithm. A proprietary algorithm, Weighted Bandwidth Fair Scheduling (WBFS) is used to implement the weighted fair scheduling function. The WBFS algorithm is described in Weighted Bandwidth Fair Scheduling (WBFS) Algorithm. Once a port is selected, queue selection within a port is performed using a strict priority algorithm. There are 2 priority queues per port. The higher priority queue is selected for dequeue if it is not empty otherwise the lower priority queue will be selected. As long as there are frames in the higher priority queue, no frames from the lower priority queue will be serviced. A single frame is dequeued from the selected queue

A byte credit-based flow control mechanism on a per priority tier (2 tiers) basis is provided to control the rate at which the consumer (i.e. HTA receive) can accept data from ICM. A byte credit counter is maintained for each priority tier, for all ENETC instances. This credit counter represents the number of bytes that can be in flight inside the HTA for a given priority tier. The byte credit counter must be non-zero for a priority to be considered eligible by the dequeue scheduling mechanism. A priority is no longer eligible for dequeues when its byte credit counter has reached a value 0 or lower (negative); after a dequeue it is possible that the byte credit count becomes a negative value if the accumulated byte credit count value was lower than the actual dequeued frame size. HTA 0 high priority configuration register (HTA0HPCR) is used to configure the maximum number of byte credits for the high priority tier whereby the HTA 0 low priority configuration register (HTA0LPCR) is used to configure the maximum number of byte credits for the low priority tier.

ICM can request the MAC to signal the other end of the link, to pause its data transmission on the link, based on shared internal receive buffer memory consumption. For each port, ICM monitors the amount of data accumulated in the shared internal receive buffer memory and if this amount exceeds the Pause ON threshold (PPAUONTR register), it then sends a request to the MAC to signal the other end of the link, to pause its data transmission on the link. The pausing of the other end of the link is stopped when the amount of data accumulated in the internal receive buffer goes below the Pause OFF threshold (PPAUOFFTR register).

53.4.2.3.1.3.11 Host Transfer Agent (HTA) Receive

The figure below shows the processing through the Host Transfer Agent Receive.

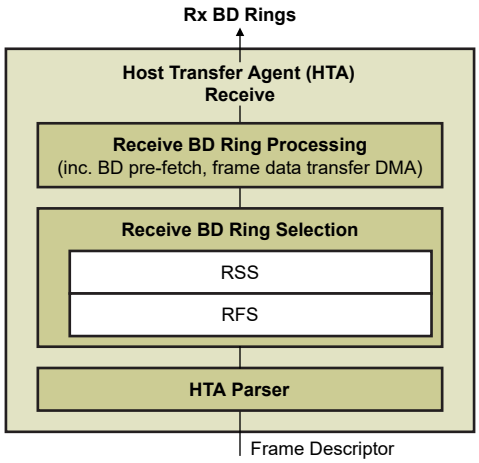


Figure 340. Host Transfer Agent Receive processing diagram

The HTA Receive block implements the receive interface functionality to the host. It includes the necessary DMA engines to efficiently transfer received frames to the host. It also provides multi-queue (buffer descriptor ring) support towards the host, for load spreading, virtualization support and QoS.

A Station Interface (SI) will normally be assigned one or more receive buffer descriptor rings (BDRs) during initialization, these will be numbered 0, 1, 2, …, n, in the SI address map.

Once receive BD rings have been assigned to an SI, they can further be assigned to a group within a SI. A group is an abstract entity (e.g. representing a core for the purpose of distribution), which will resolve into a specific receive BD ring that is a member of that group based on a configurable Internal Priority Value (IPV) to Receive BD ring mapping. This mapping allows each IPV (or set of IPV's) to be mapped to a specific receive ring for QoS distinction. The grouping should be done evenly within a SI for IPV to map to an existing ring. Note that there are at most 8 IPV values so there should not be more than eight receive rings per group.

ICM can push one frame per scheduling opportunity per input port towards HTA Receive. Associated with each frame, is the following metadata:

- SI bitmap; provides the set of Station Interfaces (SIs) to receive the frame
- IPV and DR
- IPV qualifier indicator; whether IPV was set as a result of match in the following tables: Ingress Port Filter, Ingress Stream, ingress Stream Filter or in Stream Gate Instance tables.
- If the ENETC port is connected internally to a switch port, designated as a switch management port, an indication of the source port ID from which the frame was received is provided along with the reason why this frame was forwarded to the switch management port.

The HTA Transmit functional block and the HTA Receive functional block are realized by the HTA hardware block which for the for the most part, is implemented using multi-threaded hardware engine(s) where each engine implements a finite state machine (FSM) which operates on a plurality of receive and transmit threads with each thread handling a single frame. One or more engines is instantiated depending on the requirements e.g. number of ports, port speeds. Multiple threads can be dispatched for a port. More threads can be dispatched than the total number of possible active threads (i.e. the threads currently being handled by the engines) since some threads may need to suspend (e.g. to wait for an earlier larger frame directed to the same BD ring to finish). These in-active or suspended threads therefore do not occupy a thread in an engine.

For a given frame for each of the SIs marked as eligible to receive that frame, HTA dispatches a hardware receive thread to deliver the frame to the SI. When a frame is to be delivered to more than one SI, a single copy of the frame is kept in internal memory with multiple copies written to system memory (i.e. one copy per SI). Once the processing on the last SI is completed, the internal memory that was used to store the frame is released.

Each receive thread performs the following processing:

- **Parsing** - Parses the frames and identifies the locations of the required packet header fields in preparation for classification. Supports major protocols such as Ethernet, IPv4, IPv6, TCP, UDP. It also implements the receive IP, TCP and UDP checksum offloads.
- **Receive BD ring selection** - Within an SI, determines which receive BD ring is used for the frame received. Receive BD ring selection is based on QoS and Receive Flow Steering (RFS).
- **VLAN removal** - If configured, removes VLAN tag(s) from the frame and optionally reports it in the receive BD ring descriptor.
- **Receive BD processing** - Interface functionality to the hosts including transferring frame from internal memory into buffers in system memory.

53.4.2.3.1.3.12 HTA Parser

The Parser parses up to L4 protocol headers in up to the first 256 bytes of the frame. It identifies the locations of the required packet header fields for processing by subsequent receive stages of the device. For VLAN headers, the actual field values extraction is performed by the parser. Extracted VLAN header field values and locations of relevant L2, L3 and L4 headers values are then used to perform the various table lookups and hash functions in subsequent receive stages of the device. The parser also generates a 16-bit Parse Summary of the header types identified. This summary is returned in the meta data of the receive descriptor for the frame.

During parsing, the parser examines and validates headers and validates the Internet Checksum for IPv4, TCP, and UDP. The parser is invoked with the frame fully received from the MAC and stored in internal memory.

The parsing process can be expressed as a directed graph with nodes representing header types and edges (arrows) representing the sequence of headers. Parsing starts at the root node then walks from node to node using the next-header field values. Specifically, each node works within a protocol layer presuming a certain type of header, confirms the validity of the header, identifies boundaries of the header, extracts attributes and offsets, and determines the next header to examine beyond the current header.

Figure below shows the parse graph supported by the device. The parse graph is structured by layers (L2, L3, and L4), with the details per each layer provided in the next sections. Parsing normally progresses from the lower layer to the upper layers. In the case where a node is to be re-entered, the previous parse data from that node is typically overwritten. Exceptions are stacked protocols. Special provisions are made for these protocols and will be discussed in the section pertaining to those headers.

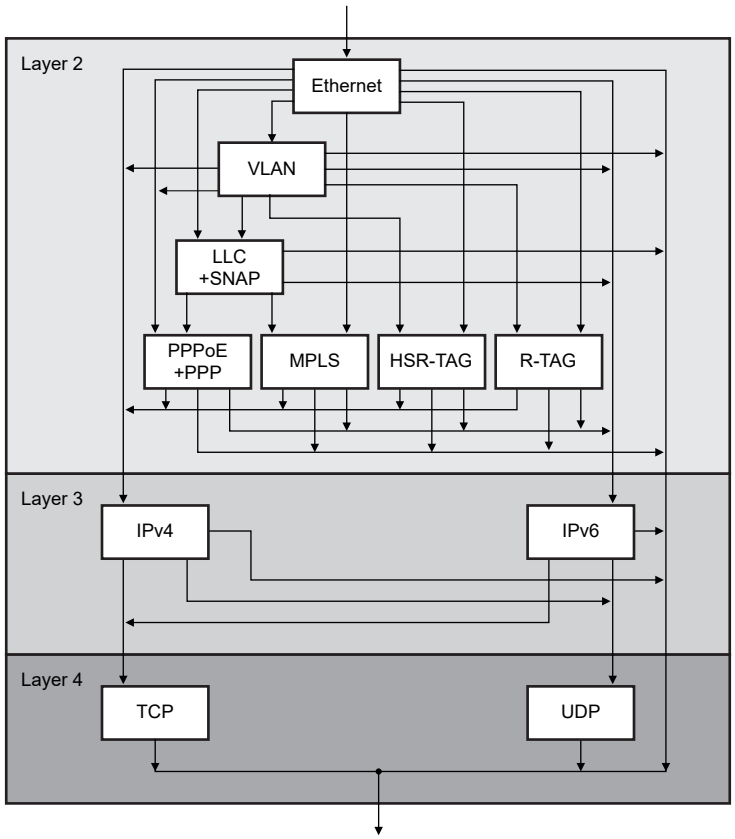


Figure 341. HTA Parse Graph

Parsing is terminated via one of the following:

- Upon reaching the natural end of a branch in the parse graph
- A parsing error is encountered; note that parse information from headers that have been successfully parsed is still used by table lookups and hash functions in subsequent receive stages of the device

If the ENETC port is connected internally to a switch port, designated as a switch management port, frames are parsed regardless of their Host Reason metadata value.

53.4.2.3.1.3.12.1 Layer 2 Headers

As the first header in a frame, the parser supports Ethernet frames of the following formats: IEEE 802.3 (Ethernet II) and 802.3 SNAP frame formats. Both Ethernet II and 802.3/SNAP frames may be tagged with the IEEE 802.1Q VLAN tagging method. The 4-byte IEEE 802.1Q VLAN tag has two 2-byte components: Tag Protocol Identifier (TPID, residing within the type/length field) and the Tag Control Information (TCI). The following 4 TPID values are recognized as VLAN tags:

- The standard types 0x8100 and 0x88A8
- Two optional user programmable EtherType values known as CustTag1 and CustTag2; configured through CVLANR1 and CVLANR2 registers.

The TCI includes the following fields:

- PCP - Priority Code Point, the first three bits of the TCI define user priority.
- DEI - Drop Eligibility/ (single-bit field)
- VID - The 12-bit VLAN ID (VID) field identifies the VLAN the frame belongs to. The 12 bits allow for 4096 different VLANs. Out of the 4096 VLANs, the VID of zero is used to identify priority tagged frames and the value of 4095 is reserved.

Parser supports frames with multiple VLAN tags. VLAN tags are named based on their position in the frame relative to other VLAN tags and the beginning of the frame i.e. the 802.3/Ethernet header consisting of Destination followed by Source MAC addresses.

The VLAN tag or header nearest the 802.3/Ethernet header is called the outer (or first) tag. The VLAN tag which is next closest to the 802.3 header is the inner (or second) tag. VLAN tags are expected to be contiguous, following one after the other, with no intervening non-VLAN tags.

Note that any VLAN tags which are within encapsulated frames and after any ULP header are not recognized. If only one VLAN tag is present in the frame, the inner VLAN tag is considered not present.

The outer and inner VLAN tag data (PCP, DEI and VID), if present, are extracted from the frame. The TPID from each VLAN tag is matched against the 4 known VLAN EtherTypes (2 standard, up to 2 custom) and a 2-bit TPID value is assigned identifying which known value matched (00: 0x8100, 01: 0x88A8, 10: CustTag1, 11: CustTag2). If the inner VLAN tag is present, the parser also indicates whether 3 or more VLAN tags are present. Matching is done against the list {0x8100, 0x88A8, CustTag1, CustTag2} in order and matching stops on the first match found. A 4-bit vector where each bit represents a TPID code point (bit 0 -> 00, bit 1 -> 01, bit 2 -> 10 and Bit 3 -> 11) is used by subsequent HTA receive processing stages to perform non-contiguous masked matching operation against the TPID.

The information found in the next EtherType field (after VLAN parsing) is used to determine the next ULP header to parse. The EtherType field is located following the Source MAC address field if no VLAN tag present, or following the last VLAN tag. The ULP header is determined according to the “Ethernet - EtherType to Next ULP” mapping described in the table below.

EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends. If parsing reached end of frame, while more octets are expected to extract the 2-byte EtherType, the extracted EtherType is set to 0.

Table 404. HTA Parser - Ethernet - EtherType to Next ULP Mapping

EtherType	Recognized Protocol	Next ULP to be parsed
x0806	ARP	Ends parsing, ARP reported in the Parse Summary
x05DC or less	LLC+SNAP	LLC+SNAP
x0800	IPv4	IPv4
x22F0	IEEE 1722	Ends parsing
x86DD	IPv6	IPv6
x8847	MPLS Unicast	MPLS
x8848	MPLS Multicast	MPLS
x8864	PPPoE+PPP Session Stage	PPPoE+PPP
x8892	Profinet RT Unicast	Ends parsing
x88A4	EtherCat	Ends parsing
x88d7	TT-Ethernet	Ends parsing
x88F7	PTP	Ends parsing
x88FB	PRP/HSR Supervision	Ends parsing
x8906	FCoE	Ends parsing
x8914	FCoE Initialize Protocol (FIP)	Ends parsing
x891D	TT-Ethernet	Ends parsing
x892F	HSR Tag	HSR Tag

Table continues on the next page...

Table 404. HTA Parser - Ethernet - EtherType to Next ULP Mapping (continued)

EtherType	Recognized Protocol	Next ULP to be parsed
xF1C1	CB Redundancy tag	CB Redundancy tag
CustEtype1	Configurable EtherType 1 (PARCE0CR)	Ends parsing
CustEtype2	Configurable EtherType 2 (PARCE1CR)	Ends parsing
CustEtype3	Configurable EtherType 3 (PARCE2CR)	Ends parsing
CustEtype4	Configurable EtherType 4 (PARCE3CR)	Ends parsing

For an LLC+SNAP header, the parser checks that the first 3 bytes (the LLC header) is 0xAAAA03 and the next 3 bytes (OUI) are 0. If both conditions aren't true, the header is not considered an LLC+SNAP header and parsing ends. If both condition are met, then the parser uses the value in the EtherType field to determine the next ULP header to parse. The next ULP header is determined according to the "HTA Parser - 802.3/SNAP - EtherType to Next ULP" mapping described in the table below. If the EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 405. HTA Parser - 802.3/SNAP - EtherType to Next ULP Mapping

EtherType	Recognized Protocol	Next ULP to be parsed
x0800	IPv4	IPv4
x86dd	IPv6	IPv6
x8847	MPLS	MPLS
x8848	MPLS	MPLS
x8864 PPPoE+PPP	PPPoE+PPP	PPPoE+PPP

For an MPLS header, the parser traverses the stack of MPLS labels looking for a label with the S bit set, identifying the last label. When the last label has been reached, the next ULP header is determined based on the interpretation of the last label. If the label is set to 0, then the next header is IPv4. If the label is set to 2, then the next header is IPv6. Otherwise parsing ends.

For PPPoE+PPP header, the parser locates the PPP protocol field. If the protocol field is set to 0x0021, then the next header is IPv4. If the protocol is set to 0x0057, then the next header is IPv6. Otherwise parsing ends.

For High Availability Seamless Redundancy (HSR) tag, the parser locates the EtherType field following the 6-byte HSR tag. The next ULP header is determined according to the "HTA Parser - HSR tag - EtherType to Next ULP" mapping described in the table below. If the EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 406. HTA Parser - HSR tag - EtherType to Next ULP Mapping

EtherType	Recognized Protocol	Next ULP to be parsed
x0800	IPv4	IPv4
x86dd	IPv6	IPv6

For the 802.1CB R-TAG (or CB Redundancy tag), the parser locates the EtherType field following the 6-byte 802.1CB R-TAG. The next ULP header is determined according to the "HTA Parser - CB Redundancy tag - EtherType to Next ULP" mapping described in the table below. If the EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 407. HTA Parser - CB Redundancy tag - EtherType to Next ULP Mapping

EtherType	Recognized Protocol	Next ULP to be parsed
x0800	IPv4	IPv4
x86dd	IPv6	IPv6

Four custom Ethertype values can be configured, and if any of the custom EtherType is detected, parsing ends with the Parse Summary set to the "Non-IP" (6) code point in bits 6-4 and to the L2 Only "Unknown" code point in bits 3-0.

Following information is captured by the parser for updating the Parse Summary and/or for usage by subsequent HTA receive processing stages:

- Indication of whether the outer VLAN tag header is present is passed as frame metadata to subsequent HTA receive processing stages. If header is present and valid (no parsing error), extracted VLAN tag data along with a 2-bit code identifying the TPID is also passed as frame metadata.
- Indication of whether the inner VLAN tag header is present and valid (no parsing error) is passed as frame metadata to subsequent HTA receive processing stages. If header is present and valid, extracted VLAN tag data along with a 2-bit code identifying the TPID is also passed as frame metadata. Additionally, an indication of whether 3 or more VLAN tags are present in the frame are also passed as frame metadata. If only one VLAN tag is present in the frame, the inner VLAN tag is considered not present.
- Whether destination MAC address is unicast, multicast or broadcast; this is passed as frame metadata to subsequent HTA receive processing stages.
- Whether the frame is a BPDU frame: MAC DA is 01:80:C2:00:00:00; this is reported in the Parse Summary.
- Whether the frame is an ARP frame (can be of any type/opcode); this is reported in the Parse Summary
- Whether the frame is a Control Protocol : MAC DA is 01:80:C2:00:00:00-01:80:C2:00:00:00:FF; this is reported in the Parse Summary.
- For each parsed L2 header (e.g. Ethernet, VLAN, MPLS, etc.), the L2 header type is reported in the Parse Summary.
- EtherTypes that are recognized but not parsed (e.g. custom EtherType, FCoE, etc.) are reported in the Parse Summary.

There are no validation checks performed on the Source and Destination MAC addresses, and EtherType fields. Subsequent HTA receive processing stages can assume the Ethernet header to be always present, that the Ethernet header starts at byte position zero within the frame and that there are no parsing errors for the Ethernet header.

Possible parsing errors for L2 headers are:

- Frame parsing reached end of frame while parsing a header that expects more data.
- For LLC+SNAP header, if 802.3 length is larger than the frame received.
- For PPPoE+PPP header, if any of the following conditions are detected
  - VER field is not set to 0x1 (RFC2516 section 4),
  - TYPE field is not to set to 0x1 (RFC2516 section 4),
  - Code field on PPP Session Stage frames is not set to 0 (RFC2516 section 4),
  - Session ID is set to 0xFFFF which is reserved and thus must not be used. (RFC2516 section 4)

53.4.2.3.1.3.12.2 Layer 3 Headers

The parser supports parsing of both IPv4 and IPv6 packets.

Only the first IPv4 or IPv6 header encountered in a packet is parsed.

If a non-fragmented IPv4 packet (Fragment Offset is set to zero and the More Fragments flag is set to zero) or an initial fragment packet ((Fragment Offset is set to zero and the More Fragments flag is set to one) is parsed, then the parser uses the information stored in the Protocol field to determine the next ULP header to be parsed.

Packets with IPv4 options are supported, thus enabling Layer 4 header recognition although no examination of the IPv4 option, unless it is an initial fragment packet. For initial fragment packets, IPv4 options are not parsed. They are also not skipped meaning that the next ULP header will not be parsed if the initial fragment packet has IPv4 options.

For IPv4 header, the next ULP header is determined according to the "HTA Parser - IPv4 - Protocol to Next ULP" mapping described in the table below. If the Protocol field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 408. HTA Parser - IPv4 - Protocol to Next ULP Mapping

Protocol (decimal)	Recognized Protocol	Next ULP to be parsed
1	ICMP	Ends parsing, ICMP reported in the Parse Summary
2	IGMP	Ends parsing, IGMP reported in the Parse Summary
6	TCP	TCP
17	UDP	UDP
33	DCCP	Ends parsing, DCCP reported in the Parse Summary
50	ESP	Ends parsing, IPsec ESP reported in the Parse Summary
132	SCTP	Ends parsing, SCTP reported in the Parse Summary
136	UDP Lite	Ends parsing, UDP-Lite reported in the Parse Summary

For IPv6 header, the IPv6 main header may be followed by an arbitrary number (0, 1 or multiple) of extension headers chained together through the Next Header field. The extension header chain typically ends with the last header corresponding to the upper-layer protocol carrying the packet's payload (for example, TCP, UDP). In the presence of extension headers, the parser traverses the extension header chain for the main purpose of finding the upper-layer protocol header (for example, TCP or UDP).

Parsing of the IPv6 extension headers is described in the table below. If an unrecognized extension header is detected (not present in the table below), parsing ends but the IPv6 header is considered successfully parsed (with relevant information captured) if no other errors are detected.

Table 409. IPv6 Extension Headers Parsing

Extension Header (decimal)	Extension Header Processing
0 (Hop-by-Hop Options)	Hop-by-Hop Options Extension header is parsed without being examined (values not checked nor captured). If present, it must immediately proceed the IPv6 main header and there can only be one of these in a packet
43 (Routing)	Routing Extension header is parsed without being examined (values not checked nor captured) The pseudo header generation for the L4 checksum considers the IP addresses of the main IPv6 header ignoring any IP address in the Routing Extension header. Therefore, checksum calculation for packets with the Routing Extension header may result in erroneous value.

Table continues on the next page...

Table 409. IPv6 Extension Headers Parsing (continued)

Extension Header (decimal)	Extension Header Processing
44 (Fragment)	If it is a non-initial fragment, then the parser does not process the next header (if any) and ends parsing
50 (ESP)	Ends parsing, IPsec ESP reported in the Parse Summary
51 (AH)	Ends parsing
60 (Destination Options)	Destination Options Extension header is parsed without being examined (values not checked nor captured).
135 (mobility)	Mobility Extension header is parsed without being examined (values not checked nor captured).
139 (Host Identity Protocol)	Host Identity Protocol Extension header is parsed without being examined (values not checked nor captured).
140 (Shim6 Protocol)	Shim6 Protocol Extension header is parsed without being examined (values not checked nor captured).
253 (Reserved)	Use for experimentation and testing. This extension header is parsed without being examined (values not checked nor captured).
254 (Reserved)	Use for experimentation and testing. This extension header is parsed without being examined (values not checked nor captured).

The parser uses the information stored in the IPv6 Next Header to determine the next ULP header. The next ULP header is determined according to the “HTA Parser - IPv6 – Next Header to Next ULP” mapping described in the table below. If the Next Header is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 410. HTA Parser - IPv6 - Next Header to Next ULP Mapping

Next Header (decimal)	Recognized Protocol	Next ULP to be parsed
6	TCP	TCP
17	UDP	UDP
33	DCCP	Ends parsing, DCCP reported in the Parse Summary
58	ICMPv6	Ends parsing, ICMPv6 reported in the Parse Summary
59	No Next Header	Indicates no more data to follow. Ends parsing
132	SCTP	Ends parsing, SCTP reported in the Parse Summary
136	UDP Lite	Ends parsing, UDP-Lite reported in the Parse Summary

Following information is captured by the parser for updating the Parse Summary and/or for usage by subsequent HTA receive processing stages.

- Indication of whether the outer IP header is present and valid (no parsing error) is passed as frame metadata to subsequent HTA receive processing stages. If the header is present and valid, its location in the frame is also passed as frame metadata.
- For each parsed L3 header (IPv4 and IPv6), the L3 header type is reported in the Parse Summary. If the outer IP header is a fragment (IPv4 "more fragments" flag is set or the IPv4 "fragment offset" field is non-zero or IPv6 Fragment Extension Header is present), the parse summary L4 field is set to the code point "1" (fragment)
- Protocols (or next ULP headers) that are recognized but not parsed (e.g. ICMP, UDP lite, etc.) are reported in the Parse Summary.
- L4 Protocol Present indicator will be set to 1, along with extracting the IPv4 Protocol field, If the outer IPv4 header is present and valid.
- L4 Protocol Present indicator will be set to 1, along with extracting the IPv6 Next Header field , if the outer IPv6 header is present and valid, and if any of the conditions below are encountered:
  - Upper-layer protocol header is found (doesn't end with an extension header) and is not set to the "No Next Header".
  - The AH and ESP extension headers are encountered.

Possible parsing errors for L3 headers are:

- Frame parsing reached end of frame while parsing a header that expects more data.
- For each IPv4 header parsed, if any of the following conditions is detected:
  - IP Checksum is incorrect (if checking is enabled)
  - Version number in IPv4 header is not equal to 4
  - IPv4 fragment does not contain at least eight data bytes and this is not the last fragment
  - IPv4 Header Length field value is less than 20 bytes, or exceeds the received packet length (excludes L2 header), or exceeds the IPv4 header Total Packet Length field value
- For each IPv6 header parsed, if any of the following conditions is detected:
  - Version number in IPv6 header packet is not equal to 6,
  - Hop by Hop extension header did not immediately follow the IPv6 main header or there is more than one Hop by Hop header.

53.4.2.3.1.3.12.3 Layer 4 Headers

The parser supports parsing of both TCP and UDP packets. Only the first TCP or UDP header encountered in a packet is parsed.

For the TCP or UDP header, the hardware calculates the expected checksum including the pseudo IP header. For IPv6 headers, the IP addresses are taken from the main IPV6 header ignoring the optional extension headers. Therefore, checksums for TCP/UDP packets encapsulated in IPv6 with extension headers that contain IP addresses may results in erroneous value.

TCP and UPD checksums are not validated for IPv4 and IPv6 fragmented packets.

For both TCP and UDP checksum offloads, hardware excludes any padding bytes on received frame with length that is shorted than 81 bytes (length of the entire received frame from the MAC). Checksum calculation is not performed for received frames with padding that have a length larger than 80 bytes.

The TCP header may have any options but these aren’t examined.

Take note that the actual calculation of the expected TCP and UDP checksums of a frame is performed later on when the hardware transfers the frame into receive data buffers in the host system memory.

For the TCP header, the next ULP header is determined according to the Destination Port field. If the Destination Port field is set 3386 (decimal), then the next header to parse is the GTP' header (i.e. header type captured in Parse Summary but no GTP' fields are extracted/validated). Otherwise parsing ends.

For the UDP header, the next ULP header is determined according to the “HTA Parser - UDP - Destination Port to Next ULP” mapping described in the table below. If the Protocol field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

There two configurable UDP Destination Port values to indicate an encapsulated VxLAN header in addition to the to the common value 4789 (decimal).

Table 411. HTA Parser - UDP - Destination Port to Next ULP Mapping

Destination Port (decimal)	Recognized Protocol	Next ULP to be parsed
319	PTP	Ends parsing, UTP PTP reported in the Parse Summary
320	PTP	Ends parsing, UTP PTP reported in the Parse Summary
2123	GTP-C	Ends parsing, UTP GTP reported in the Parse Summary
2152	GTP-U	Ends parsing, UDP GTP reported in the Parse Summary
3386	GTP'	Ends parsing, UDP GTP' reported in the Parse Summary
4500	ESP	Ends parsing, UDP ESP reported in the Parse Summary
4789	VxLAN	Ends parsing, UDP VxLAN reported in the Parse Summary
5246	CapWap-ctrl	Ends parsing, UDP CapWap-ctrl reported in the Parse Summary
5247	CapWap	Ends parsing, UDP CapWap reported in the Parse Summary
Configured VxLAN DP1	VxLAN	Ends parsing, UDP VxLAN reported in the Parse Summary
Configured VxLAN DP2	VxLAN	Ends parsing, UDP VxLAN reported in the Parse Summary

IPSec, SCTP, DCCP, GTP, ESP and VxLAN headers are recognized with their header type captured in the Parse Summary. However, parser doesn't parse these headers; i.e. no fields are extracted, no header checks are performed and the next ULP header for these headers is not located.

Following information is captured by the parser for updating the Parse Summary and/or for usage by subsequent HTA receive processing functions

- Indication of whether an L4 header (i.e. TCP or UDP) is present and valid (no parsing error) is passed as frame metadata to subsequent HTA receive processing stages. If an L4 header is present and valid, its location in frame is also passed in the metadata.
- If an L4 header is present, an indication of whether L4 header is TCP(0) or UDP(1) is passed as frame metadata for subsequent HTA receive processing.
- For each parsed L4 header (TCP, UDP), the L4 header type is reported in the Parse Summary.
- Protocols (or next ULP headers) that are recognized but not parsed (e.g. PTP, CapWap, etc.) are reported in the Parse Summary.

Possible parsing errors for L4 headers are:

- Frame parsing reached end of frame while parsing a header that expects more data.
- For TCP header, if any of the following conditions are detected:
  - TCP Checksum is incorrect (checking is enabled and hardware was able to calculate the checksum) This checks is performed later on when the hardware completed the transfer of the frame into receive data buffers in the host system memory.
  - SYN and FIN flags set
  - SYN and RST flags set
  - FIN and RST flags set
  - SYN, FIN and RST flags set
  - Node of the flags are set
- For UDP header, if any of the following conditions are detected:
  - UDP Checksum is incorrect (if checking is enabled and hardware was able to calculate the checksum); if a UDP header that has a checksum of zero is encapsulated in IPv4, no validation is carried out, however if the UDP header is encapsulated in a IPv6 packet, this is considered an error. This checks is performed later on when the hardware completed the transfer of the frame into receive data buffers in the host system memory.
  - UDP length is smaller than 8, or larger than the IP length when the latter only if it is not a fragmented packet.
  - UDP length is equal to 8 bytes and Destination Port is set to 2123 (GTP-C) , 2152 (GTP-U), or 3386 GTP'.
  - GTP (0 bytes).

53.4.2.3.1.3.12.4 Parse Summary

The parser generates a 16-bit Parse Summary of the header types identified. This summary is returned in the metadata of the receive descriptor for the frame. See table below for the parse summary fields.

Table 412. Parse Summary Fields

<table>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td rowspan="3">Parsing Error:
0 None
1 Parsing error detected</td><td colspan="2" rowspan="3">Reserved</td><td colspan="3" rowspan="3">L2:
0 Eth-II
1 LLC-SNAP
2 MPLS
3 PPPoE+PPP
4 HSR Tag
5 R-TAG</td><td colspan="2" rowspan="3">VLAN:
0 None
1 1 tag
2 2 or more tags</td><td rowspan="3">IPv4 Opt / IPv6 Ext</td><td colspan="2">L3:
0 IPv4
1 IPv6</td><td colspan="6">L4: 0-27</td></tr>
<tr><td colspan="2">2 -</td><td colspan="6"></td></tr>
<tr><td colspan="4">Non-IP: 6</td><td colspan="4">L2 Only: 0-11</td></tr>
</table>

Parsing Error: This field is set to 1 if any parsing error is detected. When a parsing error is detected, the header at which the error is detected, is identified in the following Parse Summary header type identification fields unless the errors was detected in an upper layer header not identified in the Parse Summary; in this case it would be the last header identified in the Parse Summary.

IPv4 Opt / IPv6 Ext: For IPv4, this bit is set if the outer IPv4 header has options. For IPv6, this is bit is set if the outer IPv6 header has one or more Extension headers. Note that value 59 (No Next Header) in the Next Header field of the IPv6 main header, is not considered an IPv6 extension header.

L3: When this field holds either 0 or 1

L4: When a fragment is detected, no upper layer beyond the current IP header is reported in the Parse Summary.

Non-IP: These code points are used to indicate the detection of protocols other than IP.

Table 413. Parse Summary L3 and L4 Field Descriptions

L3 (6:5)	L4 (4:0)		
0 IPv4 1 IPv6	0 Undetermined 1 Fragment 2 - 3 ICMP/ICMPv6 4 IGMP 5 IPsec ESP 6 - 7 - 8 - 9 IPv4	10 IPv4-UDP/TCP 11 IPv6 12 IPv6-UDP/TCP 13 - 14 TCP GTP' 15 DCCP 16 TCP 17 UDP 18 UDP-Lite 19 SCTP	20 UDP PTP 21 UDP ESP/IKE 22 UDP GTP 23 UDP GTP' 24 UDP CapWap-ctrl 25 UDP CapWap-data 26 - 27 UDP VxLAN

Non-IP (6:4)	L2 Only (4:0)		
6 Non-IP related protocols	0 Unknown 1 BPDU 2 Eth Ctrl MAC 3 ARP	4 PTP 5 FCoE 6 MPLS 7 IEEE 1722	8 EtherCat 9 Profinet RT unic 10 PRP/HSR Super 11 TT-Ethernet

53.4.2.3.1.3.12.5 Receive Checksum offloads

The device supports the offloading of the receive checksum validation for IPv4, TCP and UDP headers.

Checksum validation for IPv4, TCP and UDP headers in the presence of IPv4 options are supported.

For IPv6 headers, the IP addresses are taken from the main IPV6 header ignoring the optional extension headers. Therefore, checksums for TCP/UDP packets encapsulated in IPv6 with extension headers that contain IP addresses may results in erroneous value. TCP and UPD checksums are not validated for IPv4 and IPv6 fragmented packets.

Take note that the actual calculation of the expected TCP and UDP checksums of a frame is performed when the hardware transfers the frame into receive data buffers in the host system memory.

Receive checksum validation for IPv4, TCP and UDP headers can be independently enabled/disabled on per port basis (see Parser checksum configuration register (PARCSCR)). If a UDP header that has a checksum of zero, is encapsulated in IPv4, no validation is carried out, however if the UDP header is encapsulated in a IPv6 packet, this is considered an error.

Failure of the L3 (IPv4) header checksum validation or the L4 (TCP or UDP) checksum validation, is reported as an error in the ERROR field of the receive buffer descriptor. Whether the L3 (IPv4) header checksum and/or L4 (TCP or UDP) checksums were validated and found correct are also reported in the received buffer descriptor.

Calculation of a bulk checksum (1's complement checksum) is performed by the hardware when transferring a frame into receive data buffers in the host system memory. The bulk sum is calculated over the frame from the 1st byte to the last byte, excluding the Ethernet CRC if it is transferred by the MAC (i.e. present in the frame). The bulk sum is then updated to exclude the first 14 bytes (Ethernet header). Once the entire frame has been transferred from the device to the host, the bulk checksum is inverted (in order to be Internet protocol standard compliant) and then placed in the metadata of the receive descriptor for the frame.

53.4.2.3.1.3.12.6 Parsing support for Wake-on-LAN filtering

To support Wake-on-LAN filtering, the parser provides the identification of the following frames:

- TCP SYN set in the TCP header
- ICMPv6 header detected with the Type Identifier set to MLD/Query (ICMPv6 Type 130 decimal)
- ICMPv6 header detected with the Type Identifier set to ND/NS (ICMPv6 Type 135 decimal).

In addition to the above Wake-on-LAN frame types, magic packets are also supported as a Wake-on-LAN frame type. Magic packet detection is performed by the Ethernet I/F block.

53.4.2.3.1.3.13 Receive Buffer Descriptor Ring Selection

This processing stage determines which receive BD ring is used for the frame received.

To determine the receive descriptor ring for the ingress frame, a multi step classification/mapping is performed by the SI. The receive BD rings assignment to group is performed by setting the SIRBGCR register according to the rules defined in the register definition. Once the configuration of groups and ring allocation to SIs is complete, the receive BD rings can be properly initialized and setup. See Receive Buffer Descriptor Ring for more details. The steps to determine the BD ring are described below:

1. Firstly, an SI default group is selected using SIMR[DEFAULT_RX_GROUP]. If groups are not enabled/used (i.e. SIRBGCR[NUM_GROUPS]=0), then BD ring numbered 0 is selected as the default. If the ENETC port is connected internally to a switch port, designated as a switch management port, and the frame metadata field Host Reason is set to non-zero (i.e. not a regular forwarded frame), this step is not applicable, go to step #2. Otherwise go to step #3.
2. If the ENETC port is connected internally to a switch port, designated as a switch management port, and the frame metadata field Host Reason is set to non-zero (i.e. not a regular forwarded frame), the BD ring is selected by mapping the Host Reason to a receive BD ring using the switch management Host Reason a receive BD ring mapping register SMHRaBDRMR, where a is the Host Reason codepoint number. There are no more steps in classification afterwards. Note that frames with the metadata field Host Reason set to non-zero can only be received from SI 0.
3. Receive Flow Steering (RFS) is used first to determine whether to discard the frame, direct to a BD ring group or direct to a BD ring. If RFS is disabled (PRFSMR[RFSE]=0) or not supported, this step is by-passed. If there is a match and the BD ring is found directly, there are no more steps in classification. If there is a match and the BD group is found directly, step #4 is by-passed. As stated in step #2, if the ENETC port is connected internally to a switch port, designated as a switch management port, and the frame metadata field Host Reason is set to non-zero (i.e. not a regular forwarded frame), the RFS function is not performed (by-passed).
4. If RFS is disabled or did not find a match, Receive Side Scaling (RSS) is used to find the BD ring group. If RSS is disabled or not supported, the SI default group BD ring is selected as determined in step #1. As stated in step #2, if the ENETC port is connected internally to a switch port, designated as a switch management port, and the frame metadata field Host Reason is set to non-zero (i.e. not a regular forwarded frame), the RSS function is not performed (by-passed).
5. If groups are enabled, IPV to be used for mapping to a BD ring within the selected BD ring group is determined according to the following steps:
    a. Use the IPV received from ICM if it was set as a result of an Ingress Port Filtering table lookup match or was set by PSFP. Otherwise go to the next step.
    b. If SIMR[V2IPVE]=1 and VLAN tag is present, VLAN tag PCP+DEI field is used to map to an IPV value by the SI (SIVLANIPVMR0/1 registers). The VLAN tag selection is described in HTA - VLAN Selection.
    c. If SIMR[V2IPVE]=0, the port default IPV is used (PQOSMR[DIPV]).
    d. If SIMR[V2IPVE]=1 and no VLAN tag present, the port default IPV is used (PQOSMR[DIPV])
6. IPV maps to a ring within the selected BD ring group (SIIPVBDRMR register).

To assign and configure rings to a SI, the following steps are required:

1. Assign transmit rings to SIs by setting the SI configuration register field PSI*a*CFGR0[NUM_TX_BDR].
2. Assign receive rings to SIs by setting the SI configuration register field PSI*a*CFGR0[NUM_RX_BDR].
3. Configure the receive BD rings assignment to group by setting SIRBGCR register according to the rules defined in the register definition.

4. Define the default BDR group for each SI by setting register SIMR[DEFAULT_RX_GROUP] unless the maximum number of groups supported is 1, in which case, the SI default group is set to group number 0 (note that SIMR[DEFAULT_RX_GROUP] is not provided when the maximum number of groups supported is 1).
5. Define the VLAN PCP+DEI to IPV mapping by setting register SIVLANIPVMR0/1.
6. Define the SI IPV to receive BDR mapping by setting register SIIPVBDRMR.
7. Program receive ring registers as desired.

For example:

- SI 0 and SI 1 are assigned 8 receive rings each. SI 0 configures 2 groups (0-1) with 4 rings in each. SI 1 configures 1 group (0) with 8 rings in the group.
- SI 0 is assigned 7 rings and SI 1 assigned 1 ring. SI 0 has an odd number of rings but can still define any number of groups (0-2), but may end up with unassigned rings since number of rings per group has to be the same, see SIRBGCR description.

53.4.2.3.1.3.13.1 HTA - VLAN Selection

VLAN tag selection for HTA is as follows:

1. a. If the outer VLAN tag is not valid, then the steps below are skipped and no field is present for the VLAN tag
   b. If removal of SI-based VLAN tag is disabled for this SI (PSI*a*CFGR0[VTE] set to 0), go to step #4.
   c. If removal of SI-based VLAN tag is enabled for this SI (PSI*a*CFGR0[VTE] set to 1), go to step #2.
2. The outer extracted VID and outer 2-bit TPID from the frame are matched against the SI-based VLAN (VLAN ID and TPID from PSI*a*VLANR register) if PSI*a*VLANR[E] is set to 1. If the SI-based VLAN tag removal is enabled and the outer VLAN tag matched the configured SI-based VLAN, this VLAN tag is skipped for the purpose of VLAN tag selection as the SI-Based VLAN tag is hidden from the SI driver. If there is no match against the SI-based VLAN, go to step #4.
3. If the inner VLAN is present and valid, the inner 2-bit TPID is checked against the SI's VLAN extraction bit map (PSI*a*CFGR0[SIVC]) by using the 2-bit TPID value as an index into the SIVC bitmap. If the appropriate bit is set then the VLAN tag has an "acceptable" Ethertype, and the inner VLAN tag extracted from the frame is used as the key for the VLAN tag. Otherwise no field is present for the VLAN tag. The following step is skipped.
4. The outer 2-bit TPID is checked against the SI's VLAN extraction bit map (PSIaCFGR0[SIVC]). By using the 2-bit TPID value as an index into the SIVC bitmap. If the appropriate bit is set then the VLAN tag has an "acceptable" Ethertype, and, the outer VLAN tag extracted from the frame is used as the key for the VLAN tag. Otherwise no field is present for the VLAN tag.

53.4.2.3.1.3.14 VLAN Removal

HTA Receive provides offload support for the removal of 802.1Q VLAN information from single or double tagged frames.

For the purposes of removal, the inner/outer VLAN identifications are taken from the HTA parser. The removal logic is applied on per SI.

Any outer VLAN tag (starting immediately after the SMAC address) may be removed if it matches the SI-based VLAN for the SI. A remaining VLAN tag, which is now the outer tag on the frame, may also be removed if the SI RX BD ring is configured for removal. This removed tag may optionally be presented to software in the Rx BD.

The following VLAN processing is applied for each SI when of a frame is copied from internal memory to system memory buffers.

1. If the outer VLAN is present and valid, and if removal of SI-based VLAN tag is enable for this SI (PSI*a*CFGR0[VTE] set to 1), go to step #2. If the outer VLAN is not valid, then the steps below are skipped. If the outer VLAN is present and valid, and if removal of SI-based VLAN tag is disable for this SI (PSI*a*CFGR0[VTE] set to 0), go to step #5.
2. The outer extracted VID and outer 2-bit TPID from the frame are matched against the SI-based VLAN (VLAN ID and TPID from PSI*a*VLANR register) if PSI*a*VLANR[E] is set to 1. If there is a match against the SI-based VLAN, then the tag is removed for this SI. The parse summary VLAN code point is also updated; i.e., if parse summary VLAN code

point is 1, then it is set to 0. If parse summary VLAN code point is 2, then it is set to 1 only if the "3 or more VLAN tags present" indicator is not set. If there is no match against the SI-based VLAN, go to step # 5.

3. VLAN removal configuration (RB*a*MR[VTE]) is checked to determine whether the VLAN tag is to be removed (in this case the second tag). If VLAN removal is disabled, the next steps are skipped.
4. If the inner VLAN is present, the inner 2-bit TPID is checked against the SI's VLAN extraction bitmap (PSI*a*CFGR0[SIVC]) by using the 2-bit TPID value as an index into the SIVC bitmap. If the appropriate bit is set then the VLAN header is removed. Note that the parse summary is also not updated. If the SI's VLAN presentation is enabled (RBaMR[VTPD] bit set 0), then the VLAN information from the removed VLAN header is provided to software in the Rx BD. Next step is skipped.
5. VLAN removal configuration for this SI (RB*a*MR[VTE]) is checked to determine whether the VLAN tag is to be removed (in this case the first tag). If enable, the outer 2-bit TPID is checked against the SI's VLAN extraction bitmap (PSI*a*CFGR0[SIVC]) by using the 2-bit TPID value as an index into the SIVC bitmap. If the appropriate bit is set then the VLAN header is removed. Note that the parse summary VLAN code point is not updated. If the VLAN presentation is enable (RBaMR[VTPD] bit set 0), then the VLAN information from the removed VLAN header is provided to software in the Rx BD.

53.4.2.3.1.3.15 Receive Buffer Descriptor Ring

A Station Interface (SI) will normally be assigned one or more receive buffer descriptor rings (BDRs) during initialization, see PSI*a*CFGR0[NUM_RX_BDR], these will be numbered 0-*n* in the SI address map. It is therefore important to initialize the ring groups before receive ring programming. When the SI has been assigned receive ring, each rings is configured as follows:

1. Ensure group/ring assignment is completed first. Refer to section Receive Buffer Descriptor Ring Selection.
2. Configure the receive ring base address registers RB*a*BAR0/1. The address must be 128B aligned.
3. Configure the receive ring producer index register RB*a*PIR with a value of 0. The producer index is initially configured by software but owned by hardware after the ring has been enabled. Hardware increments the index when a frame is received which may consume one or more BDs. Hardware is not allowed to increment the producer index to match the consumer index since it is used to indicate an empty condition. The ring can hold at most RB*a*LENR[LENGTH]*8-1 received BDs.
4. Configure the receive ring consumer index register RB*a*CIR. The consumer index is owned by software and updated during operation of the of the BD ring by software, to indicate that any receive data occupied in the BD has been processed and it has been prepared for new data.
   - If consumer index and producer index are initialized to the same value, it indicates that all BDs in the ring have been prepared and hardware owns all of the entries.
   - If consumer index is initialized to producer index plus N, it would indicate N BDs have been prepared. Note that hardware cannot start if only a single buffer is prepared due to the restrictions described in (2).
   - Software may write consumer index to match producer index anytime while the ring is operational to indicate all received BDs prior have been processed and new BDs prepared for hardware.
5. Configure the receive ring size RB*a*LENR[LENGTH]. Note that the maximum number of received BDs allowed on the ring is one less than the ring size.
6. Configure the receive rings buffer descriptor format, standard 16B or extended 32B. The extended format supports timestamping (PTP).
7. Configure the grouping of rings using SIRBGCR for load spreading.
8. Configure the receive ring MSI-X vector number SIMSIRR*a*VR[VECTOR]. Interrupt grouping can be accomplished by assigning two or more rings to the same vector.
9. Configure the receive ring buffer size RB*a*BSR[BSIZE]. If buffer chaining is not desirable, buffer size must be large enough to hold the maximum frame size.
10. Before enabling the receive ring at least two BDs should be prepared in memory by software, as indicated by the consumer index, since there is no condition for zero prepared BDs allowed.

11. Enable the receive ring RB*a*MR[EN]=1. If a frame is steered to a receive ring which is not enabled, a programming error will occur and the frame will be dropped.

53.4.2.3.1.3.15.1 Operation of the Receive Buffer Descriptor Ring

The operation of the receive BD ring is as follows:

1. SW initializes the receive ring as described above.
2. A prefetch mechanism is used to load buffer descriptors from the receive BD rings in host memory (e.g. DDR) to an internal cache in the prefetcher. As BDs are loaded into the internal memory, they are put onto an internal BD queue that is associated with the receive BD ring from which the BDs were fetched. There is one queue for each possible receive BD ring. Each internal BD queue can hold up to 8 receive BDs. When an internal BD queue has 3 or less receive BDs, a fetch is initiated as soon as there are any receive BDs ready for reception in the corresponding receive BD ring. When an internal BD queue has 4 receive BDs, the prefetch mechanism waits until there are at least 4 receive BDs in the corresponding receive ring before loading the next BDs. This results in fetching BDs aligned on a 64B cache line boundary. Receive BD ring selection for the purpose of fetching BDs is performed in a round robin manner. The BD transfers have higher priority than the frame data transfers on the system memory interface.
3. If the ring is not full (i.e. BDs/buffers are available), HTA Receive will write data to the buffer referenced by the next prefetched BD and then update the BD by writing either 16B (standard format) or 32B (extended format) to the BD memory location. The final bit BD[F] is set if this is the last BD in a chain. The ready bit, BD[R], will be set for the (first) BD only after the frame has been fully committed to memory. Note that a full condition occurs when the producer index trails the consumer index by one.
4. HTA Receive will update the producer index register when the last BD of a chain has been written to memory to reflect that one or more BDs were added.
5. Software may use the producer index register (RB*a*PIR) or polling of BD[R] bit to determine when the next frame is available.
   - If software relies mainly on the ready bit, BD[R], for processing of received BDs, it should clear the ready bit after processing the BD. There may be a performance benefit of using the ready bit since the producer index register (RB*a*PIR) update has to wait for BDs to be committed to memory. This introduces a small window between when BD[R] is asserted and the producer index register (RB*a*PIR) is updated. Normal operation of clearing BD[R] as indicated followed by updating the consumer index register (RB*a*CIR) will avoid violating this window. The memory location of the BD ring versus reading producer index, i.e. cached or not, should be considered as well.
   - If software relies solely on the producer index register (RB*a*PIR) for processing of frames, it does not need to poll the BD for readiness and does not need to clear the BD ready bit after processing a BD. Software may use interrupt coalescing to determine when new BDs have been added to a ring to avoid polling the producer index register (RB*a*PIR).
6. Software increments the consumer index register (RB*a*CIR) after processing the received frame and adding new buffers as needed. Software must not advance the consumer index (RB*a*CIR) before hardware has updated the producer index (RB*a*PIR).

53.4.2.3.1.3.15.2 FCS Processing

An FCS is an error-detecting code added to a frame to verify the integrity of the frame as it traverses the network. On NETC, the FCS is used to check frame integrity when a frame is entering NETC (that is received from an Ethernet link) and when a frame is exiting NETC. The later check is referred to as an internal FCS check. On an ENETC function, the internal FCS check is performed as frames are transferred from shared internal buffer memory into receive data buffers in the host system memory. As frames are read out from the shared internal buffer memory, the FCS is calculated. When the end of the frame is reached, the calculated FCS is compared against the FCS associated with the frame. If there is a mismatch, the ERROR field of the receive descriptor for the frame is updated to indicate that the frame is bad (code point set to 0x20 (Frame CRC (FCS) validation failed)). Note that the internal FCS check does not include any frame modifications performed by HTA Receive, as these frame modifications are applied to the frame after the internal FCS check is performed that is frame modifications are applied when the frame is loaded into receive data buffers in the host system memory.

For frames received from a pseudo MAC, only frame descriptors are transferred across a pseudo link. Frame descriptors received on a pseudo link include references to the original frame (including FCS) and annotated frame header if present as a result of the switch performing frame modifications. This annotated frame header represents the first n bytes of the frame, along with an indication of how many bytes from the start of the original frame are no longer valid. There is an FCS covering the annotated frame header, which is distinct from the FCS covering the entire original frame. As part of the internal FCS check, the FCS of the annotated frame header and the FCS of the entire original frame are recalculated, prior to re-assembling the frame and transferring it to the receive data buffers in the host system memory. If any of the two recalculated FCSs do not match their original FCS, an internal FCS error is reported for the frame. If the FCS is to be preserved when delivering the frame to software (specified through register RBaMR[CRC], where 'a' is the BD ring number), HTA receive will regenerate a new FCS. The regenerated FCS does not include any frame modifications performed by HTA Receive, as these frame modifications are applied afterwards, when the frame is loaded into receive data buffers in the host system memory.

53.4.2.3.1.3.15.3 Receive Buffer Descriptor Ring Congestion Management

A receive BD ring will experience congestion (instantaneously (versus average) BD ring depth growing) whenever the processing core is not able to keep up with the traffic steered to the receive BD ring. This could be due to poor usage of load spreading or BD ring too small to handle temporal processing delays by the core. There are two modes of congestion management for managing receive BD ring congestion; lossless and lossy.

In the lossless mode, pausing the other end of the link is the main method of dealing with congestion. If a receive BD ring reaches full level (i.e. no BDs/buffers are available), HTA will wait for more BDs to be added to the ring by software. This will cause back-pressure to the Ingress Congestion Manager (ICM) which in turn will request the MAC to signal the other end of the link, to pause its data transmission on the link, if congestion persists (it starts running out of internal memory). The PAUSE On/Off thresholds (PPAUONTR and PPAUOFFTR) must be set for ICM to request the MAC to pause the other end of the link. It is recommended that pausing the other end of the link, should not be used with time-sensitive networks, since the transmission of all traffic classes from the other end of the link would be paused when the port is flow controlled.

In the lossy mode, tail drop is the main method for dealing with congestion (i.e. when receive BD ring reaches full level meaning that no BDs/buffers are available). This means that frames are simply dropped if the destination BD ring is enabled but has no BDs/buffers at the time when HTA needs to store that frame to the host memory.

The congestion management mode (lossless or lossy) is configurable on per BD ring basis by setting the RBaMR[CM] register.

Note that hardware does not account for FCS removal (see RBaMR[CRC], for more details) or VLAN removal (see VLAN Removal, for more details) when determining if there are sufficient BDs/buffers available to store the frame to the host memory.

The congestion management mode (lossless or lossy) is not applicable to transmit timestamp reference responses which are identified by their Host Reason code (0x3 Timestamp Response). If a receive BD ring reaches full level (i.e. no BDs/buffers are available when a transmit timestamp reference response is ready to be delivered to the host, HTA will wait for more BDs to be added to the ring by software; the transmit timestamp reference response will not be dropped.

53.4.2.3.2 Table Management Commands

Table management commands use the NTMP protocol. Familiarity with the basic NTMP concepts, described in the NETC Table Management Protocol (NTMP) section of this document, is needed to understand the material documented in this section.

Some functionality is configured and controlled using control messages sent to ENETC using a BD ring interface like the transmit BD ring. This BD ring interface is referred to as the command BD ring.

Control messages are primarily used when the underlying functionality being configured may be too large to configure using direct registers (e.g. large tables). Each SI (PSI and VSI(s) if present) has a command BD ring.

Command BD ring selection amongst all NETC functions' command BD rings (i.e. the pair of switch rings plus the per-SI ENETC rings) is performed in a round robin manner. Each command BD ring is serviced (one command per round) once in each round of the robin. Each round considers all command BD rings defined amongst all NETC functions.

The NETC serializes commands within a single ring. It is safe to pipeline commands A and B within a ring where command A must complete before command B begins. Any ordered processing must be done either using a single ring, or waiting until command A has completed before issuing command B.

Each SI command ring is configured as follows:

1. Configure the command ring base address registers SICBDRBAR0 and SICBDRBAR0. The address must be 128 byte aligned.
2. Configure the command ring producer index register SICBDRPIR. Software should initialize the index value at this address to 0 to reflect an empty ring. Optionally if the ring is pre-initialized with BDs the index values should reflect the number of BDs added.
3. Configure the command ring consumer index register SICBDRCIR. This index value indicates the next BD to be processed by the hardware. When the ring is re-initialized this register should be set to 0. The consumer index is later updated by the hardware during processing of the ring.
4. Configure the command ring size SICBDRLENR to indicate how many 32B entries, are allowed in the ring. There is no wrap bit to indicate full/empty; one entry is used to differentiate between full and empty. When producer index + 1 equals consumer index, the ring is considered full. When the producer/consumer indices match, the ring is always considered empty.
5. Enable the command ring SICBDRMR [EN]=1.

To add one or more buffer descriptors (or commands) to an enabled transmit ring, follow the procedure below. To add one or more buffer descriptors (or commands) to an enabled transmit ring, follow the procedure below:

1. Software checks to make sure the ring is not full by examining the consumer index (CI) register, SICBDRCIR. Note that maximum number of BDs allowed on a ring is one less than ring size.
2. Software adds one or more BDs to the ring. A memory barrier must be used to guarantee update to memory. If software requires an interrupt when a BD has been transmitted, set BD[CI/CCI] =1 and enable the interrupt through SICBDRIER[CBDCIE].
3. When SICBDRCIR [BDR_INDEX] match SICBDRPIR [BDR_INDEX], the ring is empty. After one or more BDs have been processed, the consumer index will be updated by hardware.

53.4.2.3.2.1 Table Management Protocol Version 1.0 Commands

Commands supported by ENETC are described in the table below. The format of the descriptor is detailed in NTMP Version 1.0.

Table 414. Supported Commands

Class	Command	Description
0	-	Reserved
1	0	Set entry in MAC address filter table
1	1	Query entry in MAC address filter table
2	0	set entry in VLAN filter table
2	1	query entry in VLAN filter table

Some command classes can only be issued by a PSI. The table below indicates for each command class, the type of SIs that can issue the command class.

Table 415. Functions of the Command BD Ring

Function	Owner	Access From	Register Support	BD Format	Notes
MAC address filtering	Port	PSI command BD ring	Capability only	short	VSI access through messaging to request PSI to modify
VLAN filtering	Port	PSI command BD ring	Capability only	short	VSI access through messaging

53.4.2.3.2.1.1 MAC Address Filter Table

Table 416. CBD Class 1: MAC Address Filter Table Entry Set Descriptor - Short Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
MAC_ADDR[31:0]																																0x00
—																MAC_ADDR[47:32]																0x04
EN	—																															0x08
—																																0x0C
—																																0x10
—																SI_BITMAP																0x14
—																INDEX																0x18
SF=1	CI	—	STATUS						—									CLASS=1														CMD=0		0x1C

Table 417. CBD Class 1: MAC Address Filter Table Entry Set Descriptor Field Descriptions

Bits	Name	Description
207-192	INDEX	Index within the MAC address filter table
175-160	SI_BITMAP	Station interfaces 15-0 for which this filter applies. Specifying unimplemented SIs will be ignored.
95	EN	Enable entry
47-0	MAC_ADDR	This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset. For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then MAC_ADDR[7:0] equals 11h and MAC_ADDR[47:40] equals 16h.

Software can query the current contents of a MAC address filter table entry using a message command, specifying the index into the MAC address filter table. The response is the same format as the set descriptor.

Table 418. CBD Class 1: MAC Address Filter Table Entry Query Descriptor - Short Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
—																																0x00
—																																0x04
—																																0x08
—																																0x0C

Table continues on the next page...

Table 418. CBD Class 1: MAC Address Filter Table Entry Query Descriptor - Short Format (continued)

—							0x10
—							0x14
—					INDEX		0x18
SF=1	CI	—	STATUS	—	CLASS=1	CMD=1	0x1C

53.4.2.3.2.1.2 VLAN Address Filter Table

Table 419. CBD Class 2: VLAN Address Filter Table Entry Set Descriptor - Short Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
—														TPID		—				VLAN_ID[11:0]												0x00
—																																0x04
EN	—																															0x08
—																																0x0C
—																																0x10
—																SI_BITMAP																0x14
—																INDEX																0x18
SF=1	CI	—	STATUS						—							CLASS=2				CMD=0												0x1C

Table 420. CBD Class 2: VLAN Address Filter Table Entry Set Descriptor Field Descriptions

Bits	Name	Description
207-192	INDEX	Absolute index position within the VLAN address filter table
175-160	SI_BITMAP	Station interfaces 15-0 for which this filter applies. Specifying unimplemented SIs will be ignored.
95	EN	Enable entry
17-16	TPID	2 bit TPID code
11-0	VLAN_ID	12-bit VLAN identifier

Software can query the current contents of a VLAN filter table entry using a message command, specifying the index into the VLAN filter table. The response is the same format as the set descriptor.

Table 421. CBD Class 2: VLAN Filter Table Entry Query Descriptor - Short Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
—																																0x00
—																																0x04
—																																0x08
—																																0x0C
—																																0x10
—																																0x14
—																INDEX																0x18
SF=1	CI	—	STATUS						—							CLASS=2						CMD=1										0x1C

53.4.2.3.2.2 Table Management Protocol Version 2.0 Commands

Commands supported by the ENETC are described in the table below. The format of the descriptor is detailed in NTMP Request Message Data Buffer and NTMP Response Message Data Buffer.

Table 422. Supported ENETC Commands

Table ID	Table Name	Table Type	Access From
5	Time Gate Scheduling table	Indexed table	PSI command BD ring
10	Rate Policer Table	Indexed table	PSI command BD ring
13	Ingress Port Filter table	Ternary match table	PSI command BD ring
30	Ingress Stream Identification table	Hash table	PSI command BD ring
31	Ingress Stream table	Indexed table	PSI command BD ring
32	Ingress Stream Filter table	Hash table	PSI command BD ring
36	Stream Gate Instance table	Indexed table	PSI command BD ring
37	Stream Gate Control List table	Indexed table	PSI command BD ring
38	Ingress Stream Count table	Indexed table	PSI command BD ring

53.4.2.3.2.2.1 Time Gate Scheduling Table

Table 423. Table ID 5 - Time Gate Scheduling Table Common Attributes

TABLE_ID	5
TABLE_VERSION	0
Table Type	Static bounded index table

Table continues on the next page...

Table 423. Table ID 5 - Time Gate Scheduling Table Common Attributes (continued)

Table Description	The Time Gate Scheduling table contains time gate scheduling configuration and operational information for a NETC function. There is one Time Gate Scheduling table for each ENETC instance and one Time Gate Scheduling table for the entire switch. For the switch, there is one entry for each switch port, whereby for an ENETC instance, there is a single entry for the entire ENETC instance. Each entry includes both an administrative gate control list and an operational gate control list, along with related configuration and status parameters. There is an internal memory that is used to store the administrative gate control list and the operational gate control list of each table entry. The number of words available for all entries in the Time Gate Scheduling table is specified in register TGSTCAPR[NUM_WORDS]. The memory is managed using a single free list of fixed equal size units (or words) of memory. Free memory units from the free list are available to be dynamically allocated to any gate control lists subject to the total memory available and maximum gate control list length (TGSTCAPR[MAX_GCL_LEN]. Each gate control list is comprised of a sequence of gate operation entries where each entry occupies one word of memory. Table management command operations Update and Query are supported. After reset, each entry is present with all fields cleared to zero; there is no administrative gate control list and operational gate control list, indicated by their gate control list length being set to 0. An administrative gate control list can only be added by an Update operation, on a entry that contains no administrative gate control list (length is set to 0). An existing administrative gate control list cannot be directly updated/modified. The administrative gate control list contained in the entry must be first removed by issuing an Update operation that sets the administrative gate control list length to zero, followed by another Update operation that adds the new administrative gate control list. An administrative gate control list cannot be removed if its config change time is 1 ms away. Therefore, before attempting to remove an administrative gate control list, a check must be performed to determine if the config change time minus the current time is less than 1 ms. If so, then the administrative gate control list can no longer be removed as it will become the operational gate control list. There are two ways to obtain the config change time: • config change time is returned in the STATUS field of a command response to a command configuring an administrative gate control list. • config change time can be retrieved by issuing a query operation against the Time Gate scheduling table entry. Once an operational gate control list is installed, it cannot be removed by a table management command. Disabling time gate scheduling on a port (PTGSCR[TGE]=0), will remove both the operational gate control list and the administrative gate control list from the corresponding table entry. When adding an administrative gate control list while there is a currently running gate control list (that is operational gate control list), the base time of the administrative gate control list (ADMIN_BASE_TIME value) must be set to a time in the future. The minimum future time is equal to: *CurrentTime* + *LookAheadTime* + *TableCmdProcessingTime* + (2 × *OperCycleTime*) Where: *CurrentTime* = 1588 (synchronized) time at which software is to issue the Time Gate Scheduling table update command *LookAheadTime* = E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET] *TableCmdProcessingTime* = Hardware processing time of the Time Gate Scheduling table update command 21,200 NETC clock cycles; for example, @240Mhz ~ 90µs *OperCycleTime* = Cycle time of the operational gate control list Thus, if there is a currently running gate control list and the base time of the administrative gate control list to be added is within the following range: *AdminBaseTime* < *CurrentTime* + *LookAheadTime* + *TableCmdProcessingTime* + (2 × *OperCycleTime*)

Table continues on the next page...

Table 423. Table ID 5 - Time Gate Scheduling Table Common Attributes (continued)

	Where: *AdminBaseTime* = Base time of the administrative gate control list Then software must adjust the base time of the administrative gate control list (ADMIN_BASE_TIME value) as follows: *Adjusted AdminBaseTime* = *AdminBaseTime* + (N × *AdminCycleTime*) where N is the smallest integer for which the relation *Adjusted AdminBaseTime* ≥ *CurrentTime* + *LookAheadTime* + *TableCmdProcessingTime* + (2 × *OperCycleTime*) would be TRUE. When the administrative gate control list is installed (by the hardware) as the operational gate control list, the gate control list itself is not copied, instead the gate control list is assigned as the operational gate control list. A query to the administrative gate control list will then return 0 for the length indicating that there is no administrative gate control list present in the table entry. However, note that ADMIN_ BASE_TIME, ADMIN_CYCLE_TIME and ADMIN_CYCLE_TIME_EXT values are preserved, until a new administrative gate control list is configured.			
Number of Entries	For the switch, the number of entries is equal to the number of switch ports; number of switch ports is indicated in register SCAPR0[NUM_PORT]. For ENETC, there is a one table entry per table.			
Default Reset Behavior	After reset, each entry is present with all fields cleared to zero; there is no administrative gate control list and operational gate control list, indicated by their gate control list length being set to 0			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x2 = Update. 0x4 = Query.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	Variable	4
	OLSE_DATA	Operational List State Element	Variable	4

Table continues on the next page...

Table 423. Table ID 5 - Time Gate Scheduling Table Common Attributes (continued)

Entry Alignment [bytes]	4
Entry ID Management	Software assigns entry IDs.
Response STATUS Applicable	Yes
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = Reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x0D0 = Command issued when time gating function is disabled for the port. 0x0D1 = Update action attempted on an existing admin gate control. An existing admin gate control list cannot be modified, Delete admin gate control list first before creating a new admin list. (Use update action with ADMIN_CONTROL_LIST_LENGTH =0 to perform delete). 0x0D2 = Update action attempted exceeds TGSTCAPR[MAX_GCL_LEN]. 0x0D3 = Update action attempted exceeds TGSTCAPR[NUM_WORDS]. 0x0D4 = Insufficient resources to perform the requested operation (not enough free time gate list entries) 0x0D5 = Update action attempted with ADMIN_CYCLE_TIME, ADMIN_TIME_INTERVAL_GE_i or truncated ADMIN_TIME_INTERVAL_GE_n due ADMIN_CYCLE_TIME specified is not sufficient to transmit 64 byte of frame data + header overhead, or if a gate time interval specified is zero. Header overhead is specified in the PTXSDUOR register. 0x0D6 = Update action attempted with ADMIN_BASE_TIME specified that is smaller than *CurrentTime* + *LookAheadTime* + *TableCmdProcessingTime* + (2 × *OperCycleTime*). 0x0D7 = Update action attempted with ADMIN_CYCLE_TIME + ADMIN_CYCLE_TIME_EXT is greater than 2^{32}-1. 0x0D8 = Query action issued when config change occurred. Retry query. 0x0D9 = Update action attempted with ADMIN_HR_CB_GE_i set to an invalid value.

Table 424. Table ID 5 - Time Gate Scheduling Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																...
CFGE_DATA																																Variable

Table 425. Table ID 5 - Time Gate Scheduling Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 5 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
Variable-64	Variable	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 426. Table ID 5 - Time Gate Scheduling Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS[31:0]																																0x0
STATUS[63:32]																																0x4
STATUS[95:64]																																0x8
ENTRY_ID																																0xC

Table continues on the next page...

Table 426. Table ID 5 - Time Gate Scheduling Table Response Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CFGE_DATA																																0x10
...																																...
CFGE_DATA																																Variable
OLSE_DATA																																4-Byte Aligned
...																																...
OLSE_DATA																																Variable

Table 427. Table ID 5 - Time Gate Scheduling Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
95-0	96	STATUS	Status Bits [63:0] contain the value of the ConfigChangeTime variable as computed by the SetConfigChangeTime() procedure during execution of the update action. The value is expressed in nanoseconds. Bits [95:64] are reserved for future use. For generic details on the STATUS field, see Status.
127-96	32	ENTRY_ID [1]	Entry ID Note that the Entry ID value which is supplied corresponds to the Port ID with which this time gate control list is associated. For generic details on the Entry ID field, see Entry ID Management.
Variable-128	Variable	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.
Variable-Variable	Variable	OLSE_DATA[1]	Operational List State Element Data See OLSE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 428. Table ID 5 - Time Gate Scheduling Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ADMIN_BASE_TIME																																0x0
ADMIN_BASE_TIME																																0x4
ADMIN_CYCLE_TIME																																0x8
ADMIN_CYCLE_TIME_EXT																																0xC
--																						ADMIN_CONTROL_LIST_LENGTH										0x10

Table continues on the next page...

Table 428. Table ID 5 - Time Gate Scheduling Table CFGE_DATA Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ADMIN_TIME_INTERVAL_GE_0																																0x14
--												HR_CB_GE_0				--								ADMIN_TC_STATES_GE_0								0x18
...																																...
ADMIN_TIME_INTERVAL_GE_N																																0x14+8N
--												HR_CB_GE_N				--								ADMIN_TC_STATES_GE_N								0x18+8N

Table 429. Table ID 5 - Time Gate Scheduling Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	ADMIN_BASE_TIME	Administrative Base Time This field sets the base time, i.e., the start time, of the administrative gate control sequence. The time is expressed in units of nanoseconds. The field implements the AdminBaseTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 AdminBaseTime parameter, which is the PTP time format, consisting of a 48-bit integer number of seconds and a 32-bit integer number of nanoseconds. Instead of using the PTP time format, the field expects the time to be specified as a 64-bit integer number of nanoseconds. Therefore, if this field is to be set to a value which was obtained from a managed object using the IEEE 802.1Q-2018 PTP time format, the value must be converted by software from the IEEE 802.1Q-2018 format to the 64-bit integer number of nanoseconds.
95-64	32	ADMIN_CYCLE_TIME	Administrative Cycle Time Sets the cycle time of the administrative gate control list. This is the period of time over which the sequence of operations in a gate control list repeats. Command will be rejected if ADMIN_CYCLE_TIME=0. The time is expressed as an integer number of nanoseconds. The field implements the AdminCycleTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 AdminCycleTime parameter, which is a rational number in units of seconds (numerator and denominator). Instead of using the IEEE 802.1Q-2018 AdminCycleTime parameter format, the field expects the time to be specified as a 32-bit integer number of nanoseconds. Therefore, if this field is to be set to a value which was obtained from a IEEE 802.1Q-2018 managed object, the value must be converted by software from the IEEE 802.1Q-2018 format to the 32-bit integer number of nanoseconds. Note that if the cycle time is shorter than the sum of the time intervals (see ADMIN_TIME_INTERVAL_GE_i), the sequence of operations is truncated, which in turn could cause the undesirable effect of overrunning a next gate-close event.
127-96	32	ADMIN_CYCLE_TIME_EXT	Administrative Cycle Time Extension This is the maximum amount of time by which the ADMIN_CYCLE_TIME is permitted to be extended when a new gate control sequence is being installed. The cycle time extension is used to extend the last complete cycle so that it ends at the new base time,

Table continues on the next page...

Table 429. Table ID 5 - Time Gate Scheduling Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			or in other words, that the old and new cycles line up. Setting the cycle time extension to a value that extends the last complete cycle to complete before the new base time (too short), will result in undefined behavior of the time gate operation. The time is expressed as an integer number of nanoseconds.
143-128	16	ADMIN_CONTROL_LIST_LENGTH	Administrative Control List Length This field indicates the number of entries in the administrative gate control list. Valid values for N are 1 to TGSTCAPR[MAC_GCL_LEN] for creating a new administrative gate control list and 0 to remove the administrative gate control list. The command will be rejected if list length is invalid.
159-144	16	--	Reserved
(223 + 64i) - (160 + 64i)	32	ADMIN_TIME_INTERVAL_GE_i	Administrative Time Interval for Gate Entry i, where i = 0, ..., N-1 Specified in nanoseconds. A minimum of 2 gate entries must be specified. Restrictions (applies to any time interval entries): - Minimum gate time interval specified must not be less than (((TGP +2) * 4) + 20)/NETC clock frequency, where TGP is the number of NETC ports (ENETC and switch) where time gating is enabled.
	8	ADMIN_TC_STATES_GE_i	Administrative Traffic Class Gate States for Gate Entry i, where i = 0, ..., N-1. Bit 0: Gate state for traffic class 0. Bit 1: Gate state for traffic class 1. Bit 2: Gate state for traffic class 2. Bit 3: Gate state for traffic class 3. Bit 4: Gate state for traffic class 4. Bit 5: Gate state for traffic class 5. Bit 6: Gate state for traffic class 6. Bit 7: Gate state for traffic class 7. For each of the traffic classes above, the following values apply: 0b = Gate closed. 1b = Gate open.
	8	--	Reserved
	4	ADMIN_HR_CB_GE_i	Administrative gate operation type (as per IEEE 802.1Q-2018) field for gate control list entry i. 0x0: SetGateStates. HoldRequest is unchanged. 0x1: Set-And-Hold-MAC. HoldRequest is set to value hold (1). 0x2: Set-And-Release-MAC. HoldRequest is set to value release (0).
	12	--	Reserved

Table 430. Table ID 5 - Time Gate Scheduling Table OLSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CONFIG_CHANGE_TIME																																0x0
CONFIG_CHANGE_TIME																																0x4
CONFIG_CHANGE_ERROR																																0x8
CONFIG_CHANGE_ERROR																																0xC
OPER_BASE_TIME																																0x10
OPER_BASE_TIME																																0x14
OPER_CYCLE_TIME																																0x18
OPER_CYCLE_TIME_EXT																																0x1C
--																OPER_CONTROL_LIST_LENGTH																0x20
OPER_TIME_INTERVAL_GE_0																																0x24
--														HR_CB_GE_0		--								OPER_TC_STATES_GE_0							0x28
...																																...
OPER_TIME_INTERVAL_GE_M																																0x24+8M
--														HR_CB_GE_M		--								OPER_TC_STATES_GE_M							0x28+8M

Table 431. Table ID 5 - Time Gate Scheduling Table OLSE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	CONFIG_CHANGE_TIME	This field specifies the time at which the administrative gate control list is to become active. The time is expressed in units of nanoseconds. The field implements the ConfigChangeTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 ConfigChangeTime parameter, which is the PTP time format, consisting of a 48-bit integer number of seconds and a 32-bit integer number of nanoseconds. Instead of using the PTP time format, the field expresses the time as a 64-bit integer number of nanoseconds. Therefore, if this field is to be queried from an IEEE 802.1Q-2018 managed object, the value must be converted by software from the 64-bit integer number of nanoseconds to the IEEE 802.1Q-2018 PTP time format.
127-64	64	CONFIG_CHANGE_ERROR	Count of the number of times that a configuration change (new schedule) has been requested with the old schedule still running and the requested base time was in the past.
191-128	64	OPER_BASE_TIME	Operational Base Time Indicates the base time, i.e., the start time, of the operational gate control sequence.

Table continues on the next page...

Table 431. Table ID 5 - Time Gate Scheduling Table OLSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			The time is expressed in units of nanoseconds. The time indicated in the field is advanced by the amount of time specified in EaTGSLR/ S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]. The field implements the OperBaseTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 OperBaseTime parameter, which is the PTP time format, consisting of a 48-bit integer number of seconds and a 32-bit integer number of nanoseconds. Instead of using the PTP time format, the field expresses the time as a 64-bit integer number of nanoseconds. Therefore, if this field is to be queried from an IEEE 802.1Q-2018 managed object, the value must be converted by software from the 64-bit integer number of nanoseconds to the IEEE 802.1Q-2018 PTP time format.
223-192	32	OPER_CYCLE_TIME	Operational Cycle Time Indicates the cycle time of the operational gate control list. This is the period of time over which the sequence of operations in a gate control list repeats. Command will be rejected if ADMIN_CYCLE_TIME=0. The time is expressed as an integer number of nanoseconds.
255-224	32	OPER_CYCLE_TIME_EXT	Operational Cycle Time Extension This is the maximum amount of time by which the OPER_CYCLE_TIME is permitted to be extended when a new gate control sequence is being installed. The time is expressed as an integer number of nanoseconds.
271-256	16	OPER_CONTROL_LIST_LENGTH	Operational Control List Length This field indicates the number of entries in the operational gate control list.
287-272	16	--	Reserved
(351 + 64i) - (288 + 64i)	32	OPER_TIME_INTERVAL_GE_i	Operational Time Interval for Gate Entry i, where i = 0, ..., M-1 Specified in nanoseconds. Restrictions (applies to any time interval entries): - Minimum time interval time is the time required to transmit 128 bytes. - Time interval needs to be greater than the advance offset time configured in PTGATOR[ADV_TIME_OFFSET]. - The sum of any 15 consecutive time intervals must be greater than the advance offset time configured in EaTGSLR[MIN_LOOKAHEAD] / S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]. - An entries' time interval should be able to transmit the Traffic Classes' MAX_SDU when fusing.
	8	OPER_TC_STATES_GE_i	Operational Traffic Class Gate States for Gate Entry i, where i = 0, ..., M-1 Bit 0: Gate state for traffic class 0. Bit 1: Gate state for traffic class 1. Bit 2: Gate state for traffic class 2. Bit 3: Gate state for traffic class 3. Bit 4: Gate state for traffic class 4. Bit 5: Gate state for traffic class 5.

Table continues on the next page...

Table 431. Table ID 5 - Time Gate Scheduling Table OLSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Bit 6: Gate state for traffic class 6. Bit 7: Gate state for traffic class 7. For each of the traffic classes above, the following values apply: 0b = Gate closed. 1b = Gate open.
	8	--	Reserved
	4	OPER_HR_CB_GE_i	Operational gate operation type (as per 802.1Q) field for gate control list entry i. 0x0: SetGateStates. HoldRequest is unchanged. 0x1: Set-And-Hold-MAC. HoldRequest is set to value hold (1). 0x2: Set-And-Release-MAC. HoldRequest is set to value release (0).
	12	--	Reserved

53.4.2.3.2.2.2 Rate Policer Table

Table 432. Table ID 10 - Rate Policer Table Common Attributes

TABLE_ID	10
TABLE_VERSION	0
Table Type	Dynamic bounded index table
Table Description	The Rate Policer table contains the rate policer configuration and operational information for a NETC function. There is one Rate Policer table for each ENETC instance and one Rate Policer table for the entire switch. Each table entry contains a set of parameters that defines a single rate policer instance. A rate policer instance controls the amount of traffic that is allowed to go through for a particular received flow of frames. A rate policer instance can be pointed to in the Ingress Port Filter, Ingress Stream, Ingress Stream Filter tables and storm control registers. Table entries used for storm control and table entries used for rate policing are mutual exclusive; these entries cannot be shared between these two distinct functions. Rate policers can be used across switch ports, and streams. In the case of bandwidth violations, a rate policer instance can change the drop resilience parameter associated with each frame and can discard frames on various basis such as the drop resilience of the frame. Table management command operations Add, Delete, Update and Query are supported. This table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. The entries are stored in contiguous manner in a common memory. For this table, each table entry occupies four words. Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) registers; S0RPITMAR[NUM_WORDS] for switch and EaRPITMAR[NUM_WORDS] for ENETC. These allocations can also be changed when the functions are initialized through their respective base function register RPITMAR[NUM_WORDS].
Number of Entries	Maximum number of entries is indicated in RPITCAPR[NUM_ENTRIES]. Number of entries in-use is indicated in RPITOR[NUM_ENTRIES].

Table continues on the next page...

Table 432. Table ID 10 - Rate Policer Table Common Attributes (continued)

Default Reset Behavior	No entries in the table by default.			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update. 0x8 = Add.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	18	4
	FEE_DATA	Functional Enable Element	1	1
	PSE_DATA	Policer State Element	1	1
	STSE_DATA	Statistics Element	84	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x120 = SDU_TYPE specified in CFGE_DATA is out of range.			

Table 433. Table ID 10 - Rate Policer Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--												STSEU	PSEU	FEEU	CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																...
xxx NOT PART OF DATA BUFFER xxx								FEE_DATA								CFGE_DATA																0x18

Table 434. Table ID 10 - Rate Policer Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The FEN bit is cleared and the rate policer instance is disabled. Then, the data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. The user must re-enable the rate policer instance manually after a configuration update by means of a Functional Enable Element update. This Functional Enable update may be done at the same time as the Configuration Element update. Bit 1: FEEU - Functional Enable Element Update. 0b = No update performed to the Functional Enable Element. 1b = The data specified in the FEE_DATA field is written to the Functional Enable Element of the table entry. Bit 2: PSEU - Policer State Element Update. 0b = No update performed to the Policer State Element. 1b = The MR state is reset to 0. Bit 3: STSEU - Statistics Element Update. 0b = No update performed to the Statistics Element. 1b = All counters within the Statistics Element are reset. Bits 15-4: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query.

Table continues on the next page...

Table 434. Table ID 10 - Rate Policer Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 10 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
207-64	144	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.
215-208	8	FEE_DATA[1]	Functional Enable Element Data See FEE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 435. Table ID 10 - Rate Policer Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
STSE_DATA																																0x4
...																																...
STSE_DATA																																0x54
CFGE_DATA																																0x58
...																																...
PSE_DATA																FEE_DATA						CFGE_DATA										0x68

Table 436. Table ID 10 - Rate Policer Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID The Entry ID represents the rate policer instance index with a range of 0 to <RPITCAPR[NUM_ENTRIES]>-1. Values which index outside of the available table entries are reserved and will result in an error. For generic details on the Entry ID field, see Entry ID Management.
703-32	672	STSE_DATA[1]	Statistics Element Data See STSE_DATA_Format.
847-704	144	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.
855-848	8	FEE_DATA[1]	Functional Enable Element Data See FEE_DATA_Format.
863-856	8	PSE_DATA[1]	Policer State Element Data See PSE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 437. Table ID 10 - Rate Policer Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CIR																																0x0
CBS																																0x4
EIR																																0x8
EBS																																0xC
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																--									SDU_TYPE	NDOR	CF	CM	DOY	MREN		0x10

Table 438. Table ID 10 - Rate Policer Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	CIR	Committed Information Rate Expressed in units of 3.725 bits per second, this parameter limits the average rate of frames declared green by the rate policer instance.

Table continues on the next page...

Table 438. Table ID 10 - Rate Policer Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
63-32	32	CBS	Committed Burst Size Expressed in bytes, this is the maximum fill (number of bytes) allowed in the Committed (C) token bucket. This parameter limits the number of bytes allowed in a burst of frames that are declared green by the rate policer instance.
95-64	32	EIR	Excess Information Rate Expressed in units of 3.725 bits per second, this parameter limits the average rate of frames declared yellow by the rate policer instance.
127-96	32	EBS	Excess Burst Size Expressed in bytes, this is the maximum fill (number of bytes) allowed in the Excess (E) token bucket. This parameter limits the number of bytes allowed in a burst of frames that are declared yellow by the rate policer instance.
128	1	MREN	Mark All Frames Red Enable This field enables the rate policer blocking function called "mark all frames red". When the "mark all frames red" feature is enabled, any frames dropped because they were marked red or yellow (if drop on yellow is enabled) will cause all subsequent frames to be marked red and the MR bit to be set to 1 within the Policer State Element. Frames marked "red" are always dropped unless the NDOR field is set to 1, in which case the red frames are not dropped, but their DR is set to 3. The hardware will continue to mark red all frames arriving at this rate policer instance until the MR bit is cleared via an update command to the Policer State Element. 0b = The rate policer blocking function "mark all frames red" feature is disabled. 1b = The rate policer blocking function "mark all frames red" feature is enabled. Not valid when NDOR=1.
129	1	DOY	Drop on Yellow 0b = Frames marked as yellow by the rate policer instance are not dropped. 1b = Frames marked as yellow by the rate policer instance are dropped.
130	1	CM	Color mode 0b = Color blind; pre-color information (encoded in the Drop Resilience (DR)) is not taken into account by the rate policer instance. 1b = Color aware; pre-color information (encoded in the Drop Resilience (DR)) is taken into account by the rate policer instance.
131	1	CF	Coupling flag This field enables the feature of coupling the Committed (C) bucket and Excess (E) bucket. When the buckets are coupled, and the C bucket becomes full (C tokens = CBS), any overflow C tokens will be added to the E bucket, up to the max fill of the E bucket (EBS). 0b = C and E token buckets are not coupled. 1b = C and E token buckets are coupled.

Table continues on the next page...

Table 438. Table ID 10 - Rate Policer Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
132	1	NDOR	No Drop on Red. When this field is set to 1, the red frames from this rate policer instance, are not dropped, but their DR is set to 3. Frames marked red by the flow meter blocking feature "mark all frames red" (enable/disable via the MREN) will also not be dropped. When this field is set to 0, frames marked "red" are always dropped.
134-133	2	SDU_TYPE	Service Data Unit Type This field specifies the type of PDU/SDU (Protocol/Service Data Unit) for a frame received by this rate policer instance when performing flow meter calculations. Overhead values are specified by register PRXSDUOR. 00b = PPDU (Physical Layer PDU). 01b = MPDU (MAC PDU). 10b = MSDU (MAC SDU). All other values reserved.
143-135	9	--	Reserved

Table 439. Table ID 10 - Rate Policer Table FEE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						--									FEN	0x0

Table 440. Table ID 10 - Rate Policer Table FEE_DATA Description

Bits	Width [bits]	Name	Description
0	1	FEN	Functional Enable This field is used to enable/disable a rate policer instance. 0b = The rate policer instance is disabled, but the token buckets and statistics counters and other configuration information are left intact, to allow reading them after a policer instance has been disabled. Frames directed to a policer instance that is disabled are passed through (not discarded). 1b = The rate policer instance is enabled based on the configuration information within the Configuration Element. Enabling a rate policer instance consists of setting the committed (C) token bucket to its full value (which is CBS), setting the excess (E) token bucket to its full value (which is EBS), and clearing all of the statistics counters to 0. Note that the timer function must be configured and enabled before any rate policer instance is enabled. Either the default nanosecond timer or the 1588 timer can be used to generate the required nanosecond resolution time. The default nanosecond timer is configured and enabled by default out of reset. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer

Table continues on the next page...

Table 440. Table ID 10 - Rate Policer Table FEE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			is in normal operation any change to the 1588 timer configuration (for example, TMROFF_H/L), except for TMR_ADD updates, requires that all rate policer instances be disabled.
7-1	7	--	Reserved.

Table 441. Table ID 10 - Rate Policer Table PSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						--									MR	0x0

Table 442. Table ID 10 - Rate Policer Table PSE_DATA Description

Bits	Width [bits]	Name	Description
0	1	MR	Mark Red Flag 0b = Indicates that the rate policer blocking "mark all frames red" function has not been triggered. 1b = Indicates that all frames arriving at this rate policer are marked red by the rate policer blocking "mark all frames red" function.
7-1	7	--	Reserved.

Table 443. Table ID 10 - Rate Policer Table STSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
BYTE_COUNT																																0x0
BYTE_COUNT																																0x4
DROP_FRAMES																																0x8
--																																0xC
DR0_GRN_FRAMES																																0x10
--																																0x14
DR1_GRN_FRAMES																																0x18
--																																0x1C
DR2_YLW_FRAMES																																0x20
--																																0x24
REMARK_YLW_FRAMES																																0x28

Table continues on the next page...

Table 443. Table ID 10 - Rate Policer Table STSE_DATA Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--																																0x2C
DR3_RED_FRAMES																																0x30
--																																0x34
REMARK_RED_FRAMES																																0x38
--																																0x3C
LTS																																0x40
BCI																																0x44
BCS	BCF																															0x48
BEI																																0x4C
BES	BEF																															0x50

Table 444. Table ID 10 - Rate Policer Table STSE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	BYTE_COUNT	Number of bytes received by the rate policer instance. Refer to SDU_TYPE for the SDU type used for byte count updates..
95-64	32	DROP_FRAMES	Number of frames dropped by the rate policer instance.
159-128	32	DR0_GRN_FRAMES	Number of frames marked green with DR=0 by the rate policer instance.
223-192	32	DR1_GRN_FRAMES	Number of frames marked green with DR=1 by the rate policer instance.
287-256	32	DR2_YLW_FRAMES	Number of frames marked yellow with DR=2 by the rate policer instance.
351-320	32	REMARK_YLW_FRAMES	Number of frames re-marked from green to yellow by the rate policer instance. If CM=(color-blind mode), this will equal DR2_YLW_FRAMES.
415-384	32	DR3_RED_FRAMES	Number of frames marked red (DR=3) by the rate policer instance.
479-448	32	REMARK_RED_FRAMES	Number of frames re-marked from green or yellow to red by the rate policer instance. If CM=0 (color-blind mode), this will equal DR3_RED_FRAMES.

Table continues on the next page...

Table 444. Table ID 10 - Rate Policer Table STSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
543-512	32	LTS	Last timestamp. This is the timestamp of the last frame (or periodic update) received by this rate policer instance. Frame timestamps in the receive datapath are applied at the MAC, and use the 32 lsb of the 64 bit 1588 time base, see also the TIMESTAMP field in the receive buffer descriptor.
575-544	32	BCI	Bucket Committed Integer (part) Indicates the integer part (32 bits) of the committed token bucket fill level, expressed in units of bytes.
606-576	31	BCF	Bucket Committed Fractional (part) Indicates the fractional part (31 bits) of the committed token bucket fill level, expressed in units of bytes.
607	1	BCS	Bucket Committed Sign-bit. Indicates the sign-bit value of the committed token bucket fill level. The committed token bucket fill level, expressed in units of bytes, is represented as a 64-bit 2’s complement value (BCI) with a 32-bits integer part and a 31-bits fractional part (BCF). The 2’s complement notation is used as the amount of tokens in the bucket can go negative (effectively borrowing future tokens) when forwarding cut-through frames. In cut-through mode, the amount of tokens required (and consumed initially) in the bucket for considering a frame ‘conforming’, is the initial segment size of the cut-through frame. Tokens then continue to get consumed as the rest of the cut-through frame is received, and then possibly resulting in the bucket going negative. The fractional part is mainly due to the replenishment of the bucket, as the replenishment rate is in units of 3.725 bits per second. Note that a fraction of a token (or byte) means from an algorithm perspective that the token is not considered available in the bucket.
639-608	32	BEI	Bucket Excess Integer (part) Indicates the integer part (32 bits) of the excess token bucket fill level, expressed in units of bytes.
670-640	31	BEF	Bucket Excess Fractional (part) Indicates the fractional part (31 bits) of the excess token bucket fill level, expressed in units of bytes.
671	1	BES	Bucket Excess Sign-bit. Indicates the sign-bit value of the excess token bucket fill level. The excess token bucket fill level, expressed in units of bytes, is represented as a 64-bit 2’s complement value with a 32-bits integer part (BEI) and a 31-bits fractional part (BEF). The 2’s complement notation is used as the amount of tokens in the bucket can go negative (effectively borrowing future tokens) when forwarding cut-through frames. In cut-through mode, the amount of tokens required (and consumed initially) in the bucket for considering a frame ‘conforming’, is the initial segment size of the cut-through frame. Tokens then continue to get consumed as the rest of the cut-through frame is received, and then possibly resulting in the bucket going negative. The

Table continues on the next page...

Table 444. Table ID 10 - Rate Policer Table STSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			fractional part is mainly due to the replenishment of the bucket, as the replenishment rate is in units of 3.725 bits per second. Note that a fraction of a token (or byte) means from an algorithm perspective that the token is not considered available in the bucket.

53.4.2.3.2.2.3 Ingress Port Filter Table

Table 445. Table ID 13 - Ingress Port Filter Table Common Attributes

TABLE_ID	13
TABLE_VERSION	0
Table Type	Hardware Managed Ternary Match table
Table Description	The Ingress Port Filter table contains a set of filters each capable of classifying incoming traffic using a mix of L2, L3, and L4 parsed and arbitrary field data. As a result of a filter match, several actions can be specified such as on whether to deny or allow a frame, overriding internal QoS attributes associated with the frame and setting parameters for the subsequent frame processing functions (e.g. stream identification, policing, ingress mirroring). Each entry corresponds to a filter. The Ingress Port Filter table is enabled per-port via PIPFCR[EN], when there is at least one entry added for that port. If disabled, or if no match is found, the frame is allowed and passed to the next frame processing function. The Ingress Port Filter entries are added using a precedence value. If a frame matches multiple entries, the entry with the higher precedence is used. The Ingress Port Filter table is implemented using a Ternary Content Addressable Memory (TCAM). TCAM provides deterministic and high-speed lookups, however they consume significant amount of die size area compared to other types of memories, and thus the need to optimize storage is important. As a result, a narrow (48-bit wide), but deep TCAM organization is used, where a table entry can occupy multiple sequential lines of the TCAM. The maximum number of TCAM lines occupied by an Ingress Port Filter entry is 14 lines. With the exception of the first TCAM line, TCAM lines for which all fields are masked out (their corresponding mask field set to all 0s) will not actually exist in the TCAM. However, if only the first line is not completely masked out, TCAM line 4 is added, where the reserved portion of the TCAM line (bits 47:40) are not masked out. As a result, an Ingress Port Filter entry may occupy from 2 lines to 14 lines, depending on how many TCAM lines are entirely don't cares. See IPF Entry Format for the Ingress Port Filter entry implementation format (that is, Ingress Port Filter table entry fields contained in each TCAM line). A maximum of 24 consecutive bytes can be used as payload data for a Ingress Port Filter entry. The per-port parser register PPCR is used to limit the amount of payload data needed, if no entries are added attempting to match the full 24B of payload. If the end of frame is received before the desired amount of payload has been extracted, then the payload data will be padded with zeros up to the next 6B multiple (TCAM line). There is a small opportunity for false matches, where the zero-padded payload matches an entry with actual payload data of zeros. However, since the payload is only padded to the next 6B boundary, the likelihood of false matches is reduced, since it only can false match an entry with zeros as the last few payload bytes, and the payload data ends at that 6B boundary. For example, if 16B of payload is required, and the end of frame happens after extracting 12B of payload, there will be no zero padding, since the payload data extracted ended on a 6B boundary. If the end of frame occurs after extracting 14B, then there will be 4B of zero padding, to complete that TCAM line. In this case, only the first 2B of zero padding will have a valid mask, the last 2B of padding will have a mask of "don't care" since we only care about 16B of payload data.

Table continues on the next page...

Table 445. Table ID 13 - Ingress Port Filter Table Common Attributes (continued)

	Software adds entries using the "Add followed by Query" operation and the Ternary Match Key Element Match Access Method. The PRECEDENCE value is used to insert the entry. Hardware automatically moves entries with lower PRECEDENCE values when adding a new entry. The ENTRY_ID assigned to the entry is returned as part of the response to the Query operation. Software can issue a Search operation to dump entries that have been added along with their ENTRY_ID values. A query operation cannot be performed using the Key Element (KEYE_DATA). If the entry being added is too large for the available memory, an error will be returned. The IPFTMOR per-function register can be used to help determine the current usage. When deleting entries, Software provides the ENTRY_ID of the entry to be deleted. Hardware will find the entry specified, delete all lines associated to the entry, and shift entries up to close the gap (if any). When software performs a Query operation, and requests the Key Element of a ternary match (TCAM) table to be returned, the Key Element information in the Response Data Buffer may not precisely match the data which was used when an Add operation was issued. Ternary match tables perform a lossy compression when storing the key element data within the key element and thus, when querying a ternary match table's key element, software must be aware of the following behavior: for any data bits which had their associated mask bit set to 0 (don't care), the resulting data bit in the query response will be 0 regardless of how that bit was set in the key element data when adding or updating the key element. The precedence value within a Key Element will be returned precisely upon query. The Search Access Method can only be used to dump the Ingress Port Filter table entries for a given function (i.e. match all entries for a given ENETC, or the Switch as a whole). Table management operations Add, Delete, Update and Query are supported.		
Number of Entries	Maximum number of words is indicated in IPFTCAPR[NUM_WORDS]. Number of words in-use is indicated in IPFTMOR[NUM_WORDS]. Note: Port Filter entries can vary in size, from 2 to 14 words.		
Default Reset Behavior	No entries in the table by default.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update.
	Exact Match Key Element Match (KEY_ELEMENT)	No	--
	Ternary Match Key Element Match (KEY_ELEMENT)	Yes	Following table management command operation is supported: 0xC = Add followed by a Query. When adding multiple entries with the same Key Element,

Table continues on the next page...

Table 445. Table ID 13 - Ingress Port Filter Table Common Attributes (continued)

			each entry will be added to the table with a new and unique Entry ID.	
	Search (SEARCH_CRITERIA)	Yes	Only supported when command operation = 0x4 (Query).	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (IPF_EID)	4	4
	ACCESS_KEY	Access Key	212	4
	CFGE_DATA	Configuration Element	8	4
	STSE_DATA	Statistics Element	8	4
	KEYE_DATA	Key Element	212	4
Entry Alignment [bytes]	4			
Entry ID Management	Hardware assigns entry IDs.			
Response STATUS Applicable	Yes			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x150 = HR value not valid. Only checked if command issued from the Switch and FLTFA=0x2 or FLTFA=0x3 0x151 = Entry being added does not fit in table. 0x152 = CFGE_DATA update without STSE_DATA update. 0x154 = RPR set to a reserved value. Only checked if FLTA=0x2. 0x155 = FLTA_TGT is outside valid range and not NULL. Only checked if FLTA>0x0. 0x156 = FLTA=0x3 when command issued from the Switch. 0x157 = FLTFA>0x1 when command issued from an ENETC PF.			

Table 446. Table ID 13 - Ingress Port Filter Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS														STSEU	CFGEU	0x0
																--																
ACCESS_KEY																																0x4
...																																...
ACCESS_KEY																																0xD4

Table continues on the next page...

Table 446. Table ID 13 - Ingress Port Filter Table Request Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CFGE_DATA																																0xD8
CFGE_DATA																																0xDC

Table 447. Table ID 13 - Ingress Port Filter Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Whenever the configuration element is updated, the statistic counter must be reset. Note: If CFGEU=1 and STSEU=0, the command will be returned with an error. Bit 1: STSEU - Statistics Element Update. 0b = No update performed to the Statistics Element. 1b = All counters within the Statistics Element are reset. Note: if STSEU=1, CFGEU can be 0 or 1. Bits 15-2: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. 0x1 = Query ENTRY_ID only. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 13 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
1727-32	1696	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. Bits 1695-31: Reserved. If ACCESS_METHOD = 0x2 (Search): Bits 31-0: SEARCH_CRITERIA. See SEARCH_CRITERIA Format. Bits 1695-31: Reserved. If ACCESS_METHOD = 0x3 (Ternary Match Key Element Match): See Request

Table continues on the next page...

Table 447. Table ID 13 - Ingress Port Filter Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			Header Description. Bits 1695-0: KEYE_DATA All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
1791-1728	64	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 448. Table ID 13 - Ingress Port Filter Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS																																0x0
ENTRY_ID																																0x4
KEYE_DATA																																0x8
...																																...
KEYE_DATA																																0xD8
STSE_DATA																																0xDC
STSE_DATA																																0xE0
CFGE_DATA																																0xE4
CFGE_DATA																																0xE8

Table 449. Table ID 13 - Ingress Port Filter Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	STATUS	Status Bits 31-0: RESUME_ENTRY_ID - An Entry ID which can be used as the RESUME_ENTRY_ID value within the ACCESS_KEY of a subsequent search to continue the search. When RESUME_ENTRY_ID is returned as the NULL Entry ID, this indicates that the search was completed successfully and no further iterations are required. Note: A NULL Entry ID is 0xFFFF_FFFF. Note: This field is valid only in responses for commands which use the Search as the Access Method. For all other Access Methods, this field is reserved.

Table continues on the next page...

Table 449. Table ID 13 - Ingress Port Filter Table Response Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			For generic details on the STATUS field, see Status. For generic details on the Search Access Method, see Search Access Method.
63-32	32	ENTRY_ID [1]	Entry ID Entry ID value generated by Hardware, used to access existing entries. Note: A NULL Entry ID is 0xFFFF_FFFF. For generic details on the Entry ID field, see Entry ID Management.
1759-64	1632	KEYE_DATA[1]	Key Element Data See KEYE_DATA Format.
1723-1760	64	STSE_DATA[1]	Statistics Element Data See STSE_DATA Format.
1887-1824	64	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 450. Table ID 13 - Ingress Port Filter Table KEYE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--																PRECEDENCE																0x0
--																																0x04
FRM_ATTR_FLAGS_MASK																FRM_ATTR_FLAGS																0x8
--								SRC_PORT_MASK						SRC_PORT		--				DSCP_MASK						DSCP					0xC
OUTER_VLAN_TCI_MASK																OUTER_VLAN_TCI																0x10
DMAC																																0x14
DMAC_MASK																DMAC																0x18
DMAC_MASK																																0x1C
SMAC																																0x20
SMAC_MASK																SMAC																0x24
SMAC_MASK																																0x28
INNER_VLAN_TCI_MASK																INNER_VLAN_TCI																0x2C
ETHERTYPE_MASK																ETHERTYPE																0x30
--																IP_PROTOCOL_MASK																0x34

Table continues on the next page...

Table 450. Table ID 13 - Ingress Port Filter Table KEYE_DATA Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--																																0x38
--																																0x3C
--																																0x40
IP_SRC_ADDR																																0x44
...																																...
IP_SRC_ADDR																																0x50
--																																0x54
--																																0x58
IP_SRC_ADDR_MASK																																0x5C
...																																...
IP_SRC_ADDR_MASK																																0x68
L4_SRC_PORT_MASK																L4_SRC_PORT																0x6C
--																																0x70
IP_DEST_ADDR																																0x74
...																																...
IP_DEST_ADDR																																0x80
--																																0x84
--																																0x88
IP_DEST_ADDR_MASK																																0x8C
...																																...
IP_DEST_ADDR_MASK																																0x98
L4_DEST_PORT_MASK																L4_DEST_PORT																0x9C
--																																0xA0
PLD_MASK_BYTE_1								PLD_DATA_BYTE_1								PLD_MASK_BYTE_0								PLD_DATA_BYTE_0							0xA4
...																																...
PLD_MASK_BYTE_23								PLD_DATA_BYTE_23								PLD_MASK_BYTE_22								PLD_DATA_BYTE_22							0xD0

Table 451. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description

Bits	Width [bits]	Name	Description
15-0	16	PRECEDENCE	Precedence

Table continues on the next page...

Table 451. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Precedence value of an entry. When adding an entry, the existing entries are walked until an entry is found with a lower or equal Relative Priority value. The new entry will be inserted at that point. In the case of an equal Relative Priority value, the new entry will have precedence over the existing entry.
63-16	48	--	Reserved.
79-64	16	FRM_ATTR_FLAGS	Frame Attribute Flags The frame attribute flags to match. Bit 0: Switch port masquerading (Switch function) 0b = Switch port is not masqueraded. 1b = Switch port is masqueraded. Applicable only if the incoming port is designated as a switch management port. There is the option on per frame basis, to indicate to the switch to process the frame as it was received from a specified switch port other than the switch management port. This option is referred to as the switch port masquerading option. Bit 0: Reserved (ENETC function) Bit 1: Reserved. Bit 2: Outer VLAN Present. The outer VLAN tag is considered present if VLAN classification has deemed the outer VLAN tag valid and the VLAN tag wasn't taken from the port VLAN configuration register. 0b = Outer VLAN is not present. 1b = Outer VLAN is present. Bit 3: Inner VLAN Present. The inner VLAN tag is considered present if VLAN classification has deemed the inner VLAN tag valid and the VLAN tag wasn't taken from the port VLAN configuration register. 0b = Inner VLAN is not present. 1b = Inner VLAN is present. Bits 6-4: Sequence Tag Code. 000b = R-TAG/HSR tag is not present. 001b = 802.1CB draft 2.0 R-TAG is present. 010b = 802.1CB R-TAG is present. 011b = HSR Tag is present. 100b ... 111b = Reserved. Bit 7: IP Header Present. 0b = IP header is not present. 1b = IP header is present.

Table continues on the next page...

Table 451. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Bit 8: IP Version. 0b = IPv4. 1b = IPv6. Bit 9: IPv4 option / IPv6 extension present. 0b = IPv4 option / IPv6 extension is not present. 1b = IPv4 option / IPv6 extension is present. Bits 11-10: L4 Code. 00b = Other L4. The L4 Header is considered as other L4 if it is not one of the following L4 Headers. 01b = TCP header is present. 10b = UDP header is present. 11b = SCTP header is present (not supported for NETC 3.0 and 3.1). Bit 12: Wake-on-LAN Magic Packet Present. 0b = WoL magic packet is not present. 1b = WoL magic packet is present. Note: Valid if IPFCAPR[WOL]=1. Reserved for Switch function. Bits 15-13: Reserved. Note: When adding an entry, all fields and masks must be correctly set, regardless of the value of this field. i.e. if Bit 8 indicates this entry is for IPv4, the IP address masks must be set for the IPv4 bits only.
95-80	16	FRM_ATTR_FLAGS_MASK	Frame Attribute Flags Mask This field is used to mask the FRM_ATR_FLAGS field when determining matches.
101-96	6	DSCP	Differentiated Services Code Point This field is matched against the differentiated services code point (DSCP) of the received frame.
107-102	6	DSCP_MASK	Differentiated Services Code Point Mask This field is used to mask the DSCP field when hardware is matching this entry.
111-108	4	--	Reserved.
116-112	5	SRC_PORT	Source Port ID This field is matched against the Source port ID. Note: this field is reserved for ENETC.
121-117	5	SRC_PORT_MASK	Source Port ID Mask This field is used to mask the SRC_PORT field when hardware is matching this entry. Note: this field is reserved for ENETC.

Table continues on the next page...

Table 451. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
127-122	6	--	Reserved.
143-128	16	OUTER_VLAN_TCI	Outer VLAN Tag Control Information This field is matched against the outer VLAN tag TCI of the received frame. TCI includes 3-bit Priority Code Point (PCP), 1-bit Drop Eligibility (DEI) and 12-bit VLAN ID (VID). This field is defined in network byte order (big-endian). Most significant byte of the TCI is stored at the lowest byte offset of this field.
159-144	16	OUTER_VLAN_TCI_MASK	Outer VLAN Tag Control Information Mask This field is used to mask the OUTER_VLAN_TCI field when hardware is matching this entry.
207-160	48	DMAC	Destination MAC Address This field is matched against the destination MAC address of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the destination MAC address is stored at the lowest byte offset of this field.
255-208	48	DMAC_MASK	Destination MAC Address Mask This field is used to mask the DMAC field when hardware is matching this entry.
303-256	48	SMAC	Source MAC Address This field is matched against the source MAC address of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the source MAC address is stored at the lowest byte offset of this field.
351-304	48	SMAC_MASK	Source MAC Address Mask This field is used to mask the SMAC field when hardware is matching this entry.
367-352	16	INNER_VLAN_TCI	Inner VLAN Tag Control Information This field is matched against the inner VLAN tag TCI of the received frame. TCI includes 3-bit Priority Code Point (PCP), 1-bit Drop Eligibility (DEI) and 12-bit VLAN ID (VID). This field is defined in network byte order (big-endian). Most significant byte of the TCI is stored at the lowest byte offset of this field.
383-368	16	INNER_VLAN_TCI_MASK	Inner VLAN Tag Control Information Mask This field is used to mask the INNER_VLAN_TCI field when hardware is matching this entry.
399-384	16	ETHERTYPE	EtherType This field is matched against the 2-byte EtherType after VLAN tag(s)/R-TAG/HSR tag (if any), of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the EtherType is stored at the lowest byte offset of this field.

Table continues on the next page...

Table 451. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
415-400	16	ETHERTYPE_MASK	EtherType Mask This field is used to mask the ETHERTYPE field when hardware is matching this entry.
423-416	8	IP_PROTOCOL	IP Protocol Indicates which L4 protocol is encapsulated. This field is matched against the IPv4 Protocol or the IPv6 Next Header (upper layer) field of the received frame.
431-424	8	IP_PROTOCOL_MASK	IP Protocol Mask This field is used to mask the IP_PROTOCOL field when hardware is matching this entry.
543-432	112	--	Reserved.
671-544	128	IP_SRC_ADDR	IP Source Address This field is matched against the IP source address of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the IP address is stored at the lowest byte offset of this field. If this IP address defines an IPv6 address, then: Bits 127-0: IPv6 source address. If this IP address defines an IPv4 address, then: Bits 95-0: Reserved. Bits 127-96: IPv4 source address .
735-672	64	--	Reserved.
863-736	128	IP_SRC_ADDR_MASK	IP Source Address Mask This field is used to mask the IP_SRC_ADDR field when hardware is matching this entry. Note: If the IP_SRC_ADDR is intended to match an IPv4 frame, then bits 95-0 of this field must be 0.
879-864	16	L4_SRC_PORT	L4 Source Port This field is matched against the L4 destination port of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the port number is stored at the lowest byte offset of this field.
895-880	16	L4_SRC_PORT_MASK	L4 Source Port Mask This field is used to mask the L4_SRC_PORT field when hardware is matching this entry.

Table continues on the next page...

Table 451. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
927-896	32	--	Reserved.
1055-928	128	IP_DEST_ADDR	IP Destination Address This field is matched against the IP destination address of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the IP address is stored at the lowest byte offset of this field. If this IP address defines an IPv6 address, then: Bits 127-0: IPv6 destination address. If this IP address defines an IPv4 address, then: Bits 95-0: Reserved. Bits 127-96: IPv4 destination address .
1119-1056	64	--	Reserved.
1247-1120	128	IP_DEST_ADDR_MASK	IP Destination Address Mask This field is used to mask the IP_DEST_ADDR field when hardware is matching this entry. Note: If the IP_DEST_ADDR is intended to match an IPv4 frame, then bits 95-0 of this field must be 0.
1263-1248	16	L4_DEST_PORT	L4 Destination Port This field is matched against the L4 destination port of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the port number is stored at the lowest byte offset of this field.
1279-1264	16	L4_DEST_PORT_MASK	L4 Destination Port Mask This field is used to mask the L4_DEST_PORT field when hardware is matching this entry.
1311-1280	32	--	Reserved.
1319-1312	8	PLD_DATA_BYTE_0	Payload Byte 0 This field is matched against the most significant byte (or first byte) of the payload data of the received frame. Payload data starting position in the received frame is dependent at which point of the frame the parser has ended. - When no IP header is present, the payload data starts immediately after the payload EtherType. The payload EtherType is located immediately after the source MAC address or VLAN tag(s)/R-TAG/HSR tag (if any).

Table continues on the next page...

Table 451. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			- When IP header is present and no TCP or UDP header is present, payload data starts immediately after the IP header except when IPv4 options or IPv6 extensions are present. For the latter, payload data starts at the beginning of the IPv4 Options/IPv6 Extensions header. - When TCP or UDP header is present, the payload data starts immediately after the TCP/UDP Ports.
1327-1320	8	PLD_MASK_BYTE_0	Payload Mask for Byte 0 This field is used to mask the PLD_DATA_BYTE_0 field when hardware is matching this entry.
1335-1328	8	PLD_DATA_BYTE_1	Payload Byte 1 This field is matched against the second-most significant byte (or second byte) of the payload data of the received frame.
1343-1336	8	PLD_MASK_BYTE_1	Payload Mask for Byte 1 This field is used to mask the PLD_DATA_BYTE_1 field when hardware is matching this entry.
...	...	...	...
1671-1664	8	PLD_DATA_BYTE_22	Payload Byte 22 This field is matched against the second-least significant byte (or second last byte) of the payload data of the received frame.
1679-1672	8	PLD_MASK_BYTE_22	Payload Mask for Byte 22 This field is used to mask the PLD_DATA_BYTE_22 field when hardware is matching this entry.
1687-1680	8	PLD_DATA_BYTE_23	Payload Byte 23 This field is matched against the least significant byte (or last byte) of the payload data of the received frame.
1695-1688	8	PLD_MASK_BYTE_23	Payload Mask for Byte 23 This field is used to mask the PLD_DATA_BYTE_23 field when hardware is matching this entry.

Table 452. Table ID 13 - Ingress Port Filter Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--									TIMECAPE	HR				CTD	RPR		FLTA		WOLTE	IMIRE	--	FLTFA		ODR	DR		OIPV	IPV					0x0
FLTA_TGT																																0x4

Table 453. Table ID 13 - Ingress Port Filter Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
3-0	4	IPV	Internal Priority Value New IPV to be assigned to frame, if OIPV is set. The least significant 3 bits are used, most significant bit is reserved.
4	1	OIPV	Override Internal Priority Value (IPV) 0b = Don't override frame IPV. 1b = Override frame IPV with value in IPV field of this entry.
6-5	2	DR	Drop Resilience New DR to be assigned to frame, if ODR is set.
7	1	ODR	Override Drop Resilience (DR) 0b = Don't override frame DR. 1b = Override frame DR with value in DR field of this entry.
9-8	2	FLTFA	Filter Forwarding Action (Switch function) 00b = Discard. 01b = Permit. 10b = Redirect frame to switch management port without any frame modification, along with Host Reason metadata set to the value configured in the HR field of this entry. Rate policing function will be applied to the frame if enabled (i.e. FLTA = 01b). Ingress Stream and 802.1Q bridge forwarding functions are by-passed. 11b = Copy frame to switch management port without any frame modification, along with Host Reason metadata set to the value configured in the HR field of this entry. Frame forwarding action not impacted. Filter Forwarding Action (ENETC function) 00b = Discard. 01b = Permit. 10b, 11b = Reserved.
10	1	--	Reserved.
11	1	IMIRE	Ingress Mirroring Enable 0b = No ingress mirroring action specified in this entry. 1b = The frame is mirrored to the mirror destination specified in the IMDCR0 register. Note: Not applicable to ENETC function.
12	1	WOLTE	Wake-on-LAN Trigger Enable 0b = WoL trigger is disabled. 1b = WoL trigger is enabled. When this field is set to 1b (WoL trigger is enabled), FLTA should be set to 11b (setting a pre L2 filtering SI bitmap) and optionally Override Internal Priority Value (IPV) and Override Drop Resilience (DR). Other actions should not be specified. Note: Valid if IPFCAPR[WOL]=1. Reserved for the Switch function.

Table continues on the next page...

Table 453. Table ID 13 - Ingress Port Filter Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
14-13	2	FLTA	Filter Action (Switch function) 00b = No action. 01b = Rate action with the Rate Policer Entry ID (RP_EID) set to the value configured in the FLTA_TGT field of this entry. 10b = Ingress stream identification action where the Ingress Stream Entry ID (IS_EID) is set to the value configured in the FLTA_TGT field of this entry. 11b = Reserved. Filter Action (ENETC function) 00b = No action. 01b = Rate policing action with the Rate Policer Entry ID (RP_EID) set to the value configured in the FLTA_TGT field of this entry. 10b = Ingress stream identification action where the Ingress Stream Entry ID (IS_EID) is set to the value configured in the FLTA_TGT field of this entry. 11b = Setting a pre L2 filtering SI bitmap that will be used by the L2 filtering function to determine the final SI bitmap. The value of pre L2 filtering SI bitmap is set to the value configured in the FLTA_TGT field of this entry.
16-15	2	RPR	Relative Precedent Resolution This field is used to select the final Ingress Stream Entry ID (IS_EID) when more than one table lookup can yield an IS_EID. This field specifies the precedence/priority of this table lookup entry relative to other table lookups that can yield the IS_EID. (IS_EID) Precedence Value for this entry. 00b = Highest precedence. 01b = Precedence is between ISIDKC0CR0 & ISIDKC1CR0 Ingress Stream Identification exact match lookup. Note that ISIDKC0CR0 lookup match has higher precedence value than ISIDKC1CR0's lookup match. 10b = Lowest precedence. 11b = Reserved. Valid if FLTA = 10b.
17	1	CTD	Cut through disable. 0b = No action. 1b = Cut-through is disabled for a frame matching this entry with this bit set. Valid when FLTFA = 01b. Note: Not applicable to ENETC function.
21-18	4	HR	Host Reason. This field specifies the Host Reason metadata when frame is redirected/copied to the switch management port (i.e. FLTFA = 10b or 11b). Value specified has to be a software defined Host Reason (8-15). See Switch Management Port for details. Note: Not applicable to ENETC function.
22	1	TIMECAPE	Timestamp Capture Enable 0b = No change to the timestamp.

Table continues on the next page...

Table 453. Table ID 13 - Ingress Port Filter Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			1b = Timestamp is to be captured; valid if FLTFA = 01b. Note: Not applicable to ENETC function.
31-23	9	--	Reserved.
63-32	32	FLTA_TGT	Target For Selected Filter Action (Switch function) If FLTA = 01b, this is a RP_EID (Rate Policer Table Entry ID). 0x0..RPITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); rate policing function is by-passed Other values are reserved If FLTA = 10b, this is an IS_EID (Ingress Stream Table Entry ID). 0x0..ISITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); stream filtering function is by-passed Other values are reserved Target For Selected Filter Action (ENETC Function) If FLTA = 01b, this is a RP_EID (Rate Policer Table Entry ID). 0x0..RPITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); rate policing function is by-passed Other values are reserved If FLTA = 10b, this is an IS_EID (Ingress Stream Table Entry ID). 0x0..ISITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); stream filtering function is by-passed Other values are reserved If FLTA = 11b, this is a pre L2 filtering SI bitmap. Each bit of the bitmap corresponds to an SI on a given port. Least significant bit of the bitmap corresponds to the smallest SI number. The SIs to which the frame is to be delivered, are identified by having their corresponding bit set to 1 in the bitmap. Bits 0-15: Bit wise setting of bitmap. LSB is the lowest order of SI. Bits 16-31: Reserved.

Table 454. Table ID 13 - Ingress Port Filter Table STSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
MATCH_COUNT																																0x0
MATCH_COUNT																																0x4

Table 455. Table ID 13 - Ingress Port Filter Table STSE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	MATCH_COUNT	Match Count A count of how many times this entry has been matched. In the event where a frame matches multiple entries, only the count of the entry with the highest relative priority will be incremented.

Table 456. Table ID 13 - Ingress Port Filter Table SEARCH_CRITERIA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RESUME_ENTRY_ID																																0x0

Table 457. Table ID 13 - Ingress Port Filter Table SEARCH_CRITERIA Description

Bits	Width [bits]	Name	Description
31-0	32	RESUME_ENTRY_ID	Resume Entry ID When starting a search, pass the NULL Entry ID. To continue a search, pass the RESUME_ENTRY_ID value returned in the CBD Response of the previous search command. Note: A NULL Entry ID is 0xFFFF_FFFF

Table 458. Table ID 13 - Ingress Port Filter Table Entry TCAM Layout

TCAM Line	Ingress Port FIlter Table Entry Fields (6 bytes per line)
1	OUTER_VLAN_TCI, FRM_ATTR_FLAGS, DSCP, SRC_PORT
2	DMAC
3	SMAC
4	IP_PROTOCOL, ETHERTYPE, INNER_VLAN_TCI
5-6	IP_SRC_ADDR (IPv6 address most significant 12 bytes)
7	IP_SRC_ADDR (IPv6 address least significant 4 bytes or IPv4 address), L4_SRC_PORT
8-9	IP_DEST_ADDR (IPv6 address most significant 12 bytes
10	IP_DEST_ADDR (IPv6 address least significant 4 bytes or IPv4 address), L4_DEST_PORT
11	PLD_DATA_BYTE_0, ..., PLD_DATA_BYTE_5
12	PLD_DATA_BYTE_6, ..., PLD_DATA_BYTE_11
13	PLD_DATA_BYTE_12, ..., PLD_DATA_BYTE_17
14	PLD_DATA_BYTE_18, ..., PLD_DATA_BYTE_23

See Table 450 for the description of the fields in the above table.

53.4.2.3.2.2.4 Ingress Stream Identification Table

Table 459. Table ID 30 - Ingress Stream Identification Table Common Attributes

TABLE_ID	30		
TABLE_VERSION	0		
Table Type	Hash table		
Table Description	The Ingress Stream Identifier table contains a set of entries each capable of classifying incoming traffic using a mix of L2 parsed and arbitrary data fields, into streams. As a result of an entry match, an Ingress Stream Entry ID is provided, which is a locally significant identifier (global to the switch), that identifies a stream within an ENETC instance or the switch. The Ingress Stream Identification table can be used to support the IEEE 802.CB stream identification functions. The Ingress Stream Identifier table is an exact match table where four logical tables can be defined using four different L2 parsed and arbitrary data fields key (key construction types). The key construction type (defined in registers) is included in the key element of an entry, to identify to which of the four logical tables the entry belongs to. The datapath can perform two exact match lookups, each against different Ingres Stream Identifier logical table. The two logical tables (or key construction type) to be used is configurable on per port basis. Table management command operations Add, Delete, Update and Query are supported.		
Number of Entries	The memory allotment to store the hash entries is shared across all of the (ENETC instance/ switch) function hash tables. The maximum amount of memory words available to store all of the function hash tables entries, is indicated in the register HTMCAPR[NUM_WORDS]). If the number of memory words in-use to store hash table entries for the function has reached this limit, the Add operation will fail. The number of memory words in-use by the function for hash table entries is indicated in the register HTMOR[AMOUNT].		
Default Reset Behavior	No entries in the table by default.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query.
	Exact Match Key Element Match (KEY_ELEMENT)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x8 = Add. 0xC = Add, followed by a Query.
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--

Table continues on the next page...

Table 459. Table ID 30 - Ingress Stream Identification Table Common Attributes (continued)

	Search (SEARCH_CRITERIA)	Yes	Only supported when command operation = 0x4 (Query). In the event of a search command requiring multiple iterations (see RESUME_ENTRY_ID), the user must not delete any entries in the table between iterations of that search.	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (ISI_EID)	4	4
	ACCESS_KEY	Access Key	20	4
	CFGE_DATA	Configuration Element	4	4
	KEYE_DATA	Key Element	20	4
Entry Alignment [bytes]	4			
Entry ID Management	Hardware assigns entry IDs.			
Response STATUS Applicable	Yes			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x260 = Port ID or specified in KEYE_DATA is out of range. Valid range of values for Port ID are {0..CAPR0[NUM_LINKS]-1}. KEY_TYPE for ENETC is not valid. 0x261 = IS_EID in invalid. Valid range of values for IS_EID are {0.. ISITCAPR[NUM_ENTRIES]-1} and not NULL IS_EID.			

Table 460. Table ID 30 - Ingress Stream Identification Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGEU	
KEYE_DATA																																0x4
...																																...
KEYE_DATA																																0x14
CFGE_DATA																																0x18

Table 461. Table ID 30 - Ingress Stream Identification Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. 0x1 = Query ENTRY_ID only. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 30 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
191-32	160	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. Bits 159-32: Reserved If ACCESS_METHOD = 0x1 (Key Element Match): Bits 159-0: KEYE_DATA. See KEYE_DATA Format. If ACCESS_METHOD = 0x2 (Search): Bits 31-0: SEARCH_CRITERIA. SEARCH_CRITERIA Format. Bits 159-32: Reserved All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
223-192	32	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 462. Table ID 30 - Ingress Stream Identification Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS																																0x0

Table continues on the next page...

Table 462. Table ID 30 - Ingress Stream Identification Table Response Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x4
KEYE_DATA																																0x8
...																																...
KEYE_DATA																																0x18
CFGE_DATA																																0x1C

Table 463. Table ID 30 - Ingress Stream Identification Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	STATUS	STATUS Bits 31-0: RESUME_ENTRY_ID - An Entry ID which can be used as the RESUME_ENTRY_ID value within the ACCESS_KEY of a subsequent search to continue the search. When RESUME_ENTRY_ID is returned as the NULL Entry ID, this indicates that the search was completed successfully and no further iterations are required. This field is valid only in responses for operations which use the Search as the Access Method. For all other Access Methods, this field is reserved. Note: A NULL Entry ID is 0xFFFF_FFFF For generic details on the STATUS field, see Status. For generic details on the Search Access Method, see Search Access Method.
63-32	32	ENTRY_ID [1]	Entry ID When an unassigned entry ID is used to access an entry, hardware needs to process the command as if the entry were not found. For generic details on the Entry ID field, see Entry ID Management.
223-64	160	KEYE_DATA[1]	Key Element Data See KEYE_DATA_Format.
255-224	32	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 464. Table ID 30 - Ingress Stream Identification Table KEYE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--																								SPM	SRC_PORT_ID						KEY_TYPE		0x0
FRAME_KEY																																0x4
...																																...
FRAME_KEY																																0x10

Table 465. Table ID 30 - Ingress Stream Identification Table KEYE_DATA Description

Bits	Width [bits]	Name	Description
1-0	2	KEY_TYPE	Key Type Switch Function 00b = Key construction is specified in ISIDKC0CR0 (header) and ISIDKC0PFbCR (payload), where 'b' represents the payload field. 01b = Key construction is specified in ISIDKC1CR0 (header) and ISIDKC1PFbCR (payload), where 'b' represents the payload field. 10b = Key construction is specified in ISIDKC2CR0 (header) and ISIDKC2PFbCR (payload), where 'b' represents the payload field. 11b = Key construction is specified in ISIDKC3CR0 (header) and ISIDKC3PFbCR (payload), where 'b' represents the payload field. ENETC Function 00b = Key construction is specified in ISIDKC0CR0 (header) and ISIDKC0PFbCR (payload), where 'b' represents the payload field. 01b = Key construction is specified in ISIDKC1CR0 (header) and ISIDKC1PFbCR (payload), where 'b' represents the payload field. All other values reserved.
6-2	5	SRC_PORT_ID	Source Port ID Switch Function Switch port ID expressed as an integer value to be used in key construction if PORTP is set to 1 in key construction register. ENETC Function Reserved. This field is not used for ENETC function.
7	1	SPM	Source Port Masquerading Switch port masquerading to be used in key construction if SPMP is set to 1 in key construction register. Switch Function 0b = Match frames from switch port(s). 1b = Match frame from switch management port(s) that has switch port masquerading. ENETC Function Reserved. This field is not used for ENETC function.
31-8	24	--	Reserved.
159-32	128	FRAME_KEY	Frame portion of the Key Key format is specified by the key construction registers ISIDKCaCR0 (frame header) and ISIDKCaPFbCR (frame payload), where ‘a’ represents the key construction rule ID and ‘b’ represents the payload field. The key entered has to be in packed form in the order they appear in the frame with each field formatted in big endian format (order they are received from the link). Maximum key size for the frame portion is 16 bytes. If the constructed key is less than 16 bytes then rest of the bytes have to padded with zeros (i.e. zeros must be entered in this field for the remaining bytes). If the key constructed is larger than 16 bytes, then this will result in no match.

Table 466. Table ID 30 - Ingress Stream Identification Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
IS_EID																																0x0

Table 467. Table ID 30 - Ingress Stream Identification Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	IS_EID	Ingress Stream Entry ID (IS_SID) This field specifies the index to use for accessing the Ingress Stream table. 0x0..ISITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); stream filtering function is by-passed Other values are reserved

Table 468. Table ID 30 - Ingress Stream Identification Table SEARCH_CRITERIA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RESUME_ENTRY_ID																																0x0

Table 469. Table ID 30 - Ingress Stream Identification Table SEARCH_CRITERIA Description

Bits	Width [bits]	Name	Description
31-0	32	RESUME_ENTRY_ID	Resume Entry ID. When starting a search, pass the NULL Entry ID. To continue a search, pass the RESUME_ENTRY_ID value returned in the CBD Response of the previous search operation. Note: A NULL Entry ID is 0xFFFF_FFFF

53.4.2.3.2.2.5 Ingress Stream Table

Table 470. Table ID 31 - Ingress Stream Table Common Attributes

TABLE_ID	31
TABLE_VERSION	0
Table Type	Dynamic bounded index table
Table Description	The Ingress Stream table contains configuration and operational information about streams identified by the stream identification function. Each entry includes the actions (or functions) that are to be applied to frames belonging to a particular stream, such as: • Whether a frame is accepted or discarded • Specifies the PSFP (IEEE 802.1Qci) filtering and policing actions. To support a PSFP (IEEE 802.1Qci) stream filtering function that uses the 3-bit PCP value (priority) of the received frame and the stream handle (i.e. Ingress Stream Entry ID) to select a stream filter, an

Table continues on the next page...

Table 470. Table ID 31 - Ingress Stream Table Common Attributes (continued)

<table>
<tr><td></td><td colspan="4">additional lookup (i.e. Ingress Stream Filter table lookup) can be specified in the Ingress Stream table entry.
• FREF (IEEE 802.1CB) sequence generation function (switch only)
• Stream forwarding function; the 802.1Q bridge forwarding function is by-passed, instead the Ingress Stream table provides the port(s) to which the frame should be output to (switch only)
• Ingress frame modification in addition to adding a sequence tag (switch only)
• Egress packet processing actions to be applied to packets received on this specific stream; typically used along with the stream forwarding function. (switch only)
The Ingress Stream table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. The entries are stored in contiguous manner in the common memory. Ingress Stream Entry ID (IS_EID) is used as an index to an entry in this table. For this table, each table entry occupies one word. Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) registers; S0ISITMAR[NUM_WORDS] for switch and EaISITMAR[NUM_WORDS] for ENETC. These allocations can also be changed when the functions are initialized through their respective base function registers ISITMAR[NUM_WORDS]. Table management command operations Add, Delete, Update and Query are supported.</td></tr>
<tr><td>Number of Entries</td><td colspan="4">Maximum number of entries is indicated in ISITCAPR[NUM_ENTRIES]. Number of entries in-use is indicated in ISITOR[NUM_ENTRIES].</td></tr>
<tr><td>Default Reset Behavior</td><td colspan="4">No entries in the table by default.</td></tr>
<tr><td rowspan="5">Access Method (ACCESS_METHOD)</td><td>Access Method</td><td>Supported</td><td colspan="2">Restrictions</td></tr>
<tr><td>Entry ID Match (ENTRY_ID)</td><td>Yes</td><td colspan="2">Following table management command operations are supported.

0x1 = Delete.
0x2 = Update.
0x4 = Query.
0x8 = Add.</td></tr>
<tr><td>Exact Match Key Element Match (KEY_ELEMENT)</td><td>No</td><td colspan="2">--</td></tr>
<tr><td>Ternary Match Key Element Match (KEY_ELEMENT)</td><td>No</td><td colspan="2">--</td></tr>
<tr><td>Search (SEARCH_CRITERIA)</td><td>No</td><td colspan="2">--</td></tr>
<tr><td rowspan="3">Data Buffer Component Attributes</td><td>Name</td><td>Description</td><td>Size [bytes]</td><td>Alignment [bytes]</td></tr>
<tr><td>ENTRY_ID</td><td>Entry Id (IS_EID)</td><td>4</td><td>4</td></tr>
<tr><td>ACCESS_KEY</td><td>Access Key</td><td>4</td><td>4</td></tr>
</table>

Table continues on the next page...

Table 470. Table ID 31 - Ingress Stream Table Common Attributes (continued)

	CFGE_DATA	Configuration Element	38	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x270 = Option specified in one or more of the following fields is not valid – FA, CTD or ISQA, SDU_TYPE. Note that CTD and ISQA are checked for Switch function only. 0x271 = One or more of following: - Entry IDs are not in valid range or Entry ID is not NULL. - Check valid ranges specified for these Entry IDs in Ingress Stream table entry – RP_EID, SGI_EID, ISQ_EID, ET_EID or EPORT. Note that ISQ_EID, ET_EID are checked for Switch function only. - ET_EID is checked if (FA =010b .. 101b) & (OETEID!=0), - EPORT is checked if (FA = 010b .. 101b) & (OETEID= 0x1 OR CTD= 0x1) - HR is checked if FA = 001b, 100b, or 101b. HR specified cannot be 0x0. 0x272 = FM_EID format or index is out of range. Check performed is as follows: – FM_EID format option type is invalid. - FM_EID format is option 1 and the Index is out of range and not NULL, or FM_EID format is option 2 and VUDA or SQTA is out of range.			

Table 471. Table ID 31 - Ingress Stream Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																...
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																CFGE_DATA																0x2C

Table 472. Table ID 31 - Ingress Stream Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element.

Table continues on the next page...

Table 472. Table ID 31 - Ingress Stream Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved.
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 31 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Table ID 31 Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
367-64	304	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA_Format

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 473. Table ID 31 - Ingress Stream Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
CFGE_DATA																																0x4
...																																...
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																CFGE_DATA																0x28

Table 474. Table ID 31 - Ingress Stream Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Valid range of IS_EID:{0.. ISITCAPR[NUM_ENTRIES]-1}. For generic details on the Entry ID field, see Entry ID Management.
335-32	304	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 475. Table ID 31 - Ingress Stream Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--			SDU_TYPE		FA			HR				OSGI	ORP	ISQA		SPPD	--	TIMECAPE	IMIRE	ODR	DR		OIPV	IPV			--				SFE	0x0
CTD		OET EID		EPORT				IFME_LEN_CHANGE								MSDU																0x4
ISQ_EID																																0x8
RP_EID																																0xC
SGI_EID																																0x10
IFM_EID																																0x14
ET_EID																																0x18
ISC_EID																																0x1C
--																EGRESS_PORT_BITMAP																0x20
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																SI_MAP																0x24

Table 476. Table ID 31 - Ingress Stream Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
0	1	SFE	Stream Filtering Enable To support a PSFP (IEEE 802.1Qci) stream filtering function that uses the 3-bit PCP value (priority) of the received frame and the stream handle (i.e. Ingress Stream Entry ID) to select a stream filter, an additional lookup (i.e. Ingress Stream Filter table lookup) must be performed to obtain the PSFP stream filter information. 0b = The Ingress Stream Filter table lookup is by-passed (or not required). 1b = The Ingress Stream Filter table lookup is performed using as a key, the Ingress Stream Entry ID and PCP (extracted from the received frame outer VLAN tag). If the

Table continues on the next page...

Table 476. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			received frame has no VLAN tag (determined from VLAN classification output; i.e. if the outer VLAN tag is not valid or that VLAN tag data is taken from port outer VLAN configuration register), this additional lookup is not performed.
3-1	3	--	Reserved.
7-4	4	IPV	Internal Priority Value New Internal Priority Value (IPV) to be assigned to the frame, if OIPV is set to 1. The least significant 3 bits are used, most significant bit is reserved.
8	1	OIPV	Override Internal Priority Value (IPV) 0b = Don't override frame IPV. 1b = Override frame IPV with value configured in IPV field of this entry.
10-9	2	DR	Drop Resilience New Drop Resilience (DR) to be assigned to the frame, if ODR is 1.
11	1	ODR	Override Drop Resilience (DR) 0b = Do not override frame DR. 1b = Override frame DR with value configured in DR field of this entry.
12	1	IMIRE	Ingress Mirroring Enable 0b = No ingress mirroring action specified in this entry. 1b = Frame is mirrored to the mirror destination specified by IMDCR0 register. Note: Not applicable to ENETC function
13	1	TIMECAPE	Timestamp Capture Enable 0b = No change to the timestamp. 1b = Timestamp is to be captured. Note: Not applicable to ENETC function
14	1	--	Reserved.
15	1	SPPD	Source Port Pruning Disable. This field is used to enable/disable source port pruning. Source port pruning ensures that frames do not get forwarded back to the port they originated from. Once the final destination ports have been determined, source pruning when enabled, will set the port from which the frame originated from, to 0 in the destination port bitmap. This field is valid only if the FA field of this entry is set 010b or 100b (stream forwarding) and will overwrite any previous configuration/setting for source pruning. 0b = Source port pruning enabled. 1b = Source port pruning disabled. Note: Not applicable to ENETC function.

Table continues on the next page...

Table 476. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
17-16	2	ISQA	Ingress Sequence Action This field specifies whether the Frame Replication and Elimination for Reliability (FRER) sequence generation function (or action) is to be performed. Sequence generation consists of tagging a frame with a sequence number, following which the frame can be replicated and transmitted along multiple paths for redundancy. Implemented as per the IEEE 802.1CB. 00b = FRER sequence generation function is not performed. 01b = FRER sequence generation function is performed. 10b, 11b = Reserved. Note: Sequence Generation action is not applicable for ENETC function.
18	1	ORP	Override Rate Policer (instance) ID. 0b = Rate policer instance ID not overridden. 1b = Rate policer instance ID overridden. If ORP = 1, RP_EID specified in this entry overrides the default RP_EID (which is NULL) or PR_EID specified in Port filter.
19	1	OSGI	Override Stream Gate Instance Entry ID 0b = Stream Gate Instance Entry ID not overridden. 1b = Stream Gate Instance Entry ID overridden if OSGI = 1, specified SGI_EID in this entry overrides the default NULL SGI_EID.
23-20	4	HR	Host Reason This field specifies the Host Reason metadata when frame is redirected to the switch management port or copied to the switch management port (i.e. FA = 01b, 100b or 101b). Value specified has to be a software defined Host Reason (8-15). See Switch Management Port for details. Note: Not applicable to ENETC function.
26-24	3	FA	Forwarding Actions Switch function 000b = Discard. 001b = Re-direct frame to switch management port without any frame modification, along with Host Reason metadata set to the value configured in the HR field of this entry. Policing function will be applied to the frame if enabled. Stream and 802.1Q bridge forwarding functions are by-passed. 010b = Stream forwarding. It by-passes the 802.1Q bridge forwarding function. Frame is forwarded to the port(s) specified in the EGRESS_PORT_BITMAP field of this entry. 011b = 802.1Q bridge forwarding (VLAN processing and L2 forwarding). 100b = Copy to switch management port with specified HR and stream forwarding (stream forwarding action is as specified in FA = 010b). 101b = Copy to switch management port with specified HR and Bridge forwarding (Bridge forwarding action is as specified FA = 011b). All other values reserved.

Table continues on the next page...

Table 476. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			ENETC function 00b = Discard; 01b = Allow without setting the pre L2 filtering SI bitmap. 10b = Allow with setting the pre L2 filtering SI bitmap to the value configured in the SI_MAP field of this entry. If the pre L2 filtering SI bitmap was already set up by the ingress port filtering function, the pre L2 filtering SI bitmap is overwritten with the value configured in the SI_MAP field of this entry, The pre L2 filtering SI bitmap is used by the L2 filtering function to determine the final SI bitmap. All other values reserved.
28-27	2	SDU_TYPE	Service Data Unit type to use for MSDU (Maximum Service Data Unit) field. Overhead values are specified by register PRXSDUOR. 00b = PPDU (Physical Layer PDU). 01b = MPDU (MAC PDU). 10b = MSDU (MAC SDU). All other values reserved. Note: This field has to be valid if MSDU field is not 0.
31-29	3	--	Reserved.
47-32	16	MSDU	Maximum Service Data Unit This field specifies the maximum service data unit size allowed for packets received on this stream. This is part of IEEE 802.1Qci (Per-Stream Filtering and Policing) specification. If the received frame length exceeds this value, the frame is discarded. A value of zero in this field, disables the maximum service data unit check. Note: This check is not performed for cut-through frames.
54-48	7	IFME_LEN_CHANGE	Ingress Frame Modification Entry Frame Length Change This field represents the effective frame length change of the ingress frame modification actions which are configured in the ingress frame modification entry (IFM_EID) referred by this entry. This frame length change value is used to adjust the frame length metadata, which in turn is passed along the frame processing pipeline, and used by the egress scheduling and management function to determine the actual length of the frame on the transmission link. Note that the egress queue scheduling and management function is executed prior to applying the frame modification actions to the frame, and thus the need to know of the frame length change resulting from the frame modification actions since it requires knowledge of the exact length of a frame transmitted on a link. Specifying explicitly the frame length change avoids having the hardware to read the ingress frame modifications entry and compute the frame length change itself. Fame length change is specified in unit of bytes using a 2's complement notation. Supported range is -4 bytes (removing one VLAN tag) to +4 bytes (adding one VLAN tag). This field is ignored by hardware if the IFM_EID field is set to NULL.

Table continues on the next page...

Table 476. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Note: Not applicable to ENETC function.
59-55	5	EPORT	Egress Port This field specifies the identifier of the port requiring egress packet processing when the OETEID field of this entry is set to 01b (single-port Egress Treatment table access option). The port is expressed as an integer value. Switch Range: {0..SCAPR0[NUM_PORT]-1} Note: Not applicable to ENETC function.
61-60	2	OETEID	Override ET_EID This field is used to convey the egress packet processing actions to be applied to packets belonging to the stream associated with this entry. The egress packet processing actions are specified in the Egress Treatment table, where each table entry contains the egress packet processing actions to be applied to a scope of packets (e.g. stream) exiting on a particular egress port of the switch. The means by which one specifies the Egress Treatment table entries to be accessed, is through the Egress Treatment group. Within the hardware, an Egress Treatment group is determined by a base index (first Egress Treatment table entry of the group) and an applicability port bitmap (a bitmap corresponding to all ports of the switch) which indicates (with a 1) which ports have Egress Treatment table entries present. For more details, see Egress Treatment table access. Specifically, this field specifies how to derive the applicability port bitmap of the primary Egress Treatment group being assigned to packets belonging to the stream associated with this entry. The base index is specified in the ET_EID field of this entry. 00b = No egress packet processing actions specified. 01b = Single-port Egress Treatment table access. Only one port requires egress packet processing. That port identifier is specified in the EPORT field of this entry. The applicability port bitmap is set with a 1 for the port that is been identified in the EPORT field of this entry. For other ports, the applicability port bitmap is set to 0. The group assigned to packets belonging to the stream associated with this entry will have a single Egress Treatment table entry, for the port identified in the EPORT field of this entry. 10b = Multi-port packed Egress Treatment table access. Applicability port bitmap is set to the port bitmap specified in EGRESS_PORT_BITMAP field of this entry. The group assigned to packets belonging to the stream associated with this entry will have an Egress Treatment table entry for each port set to 1 in the applicability port bitmap. 11b = Multi-port absolute Egress Treatment table access. Applicability port bitmap is set with 1 for all ports. The group assigned to packets belonging to the stream associated with this entry will have Egress Treatment table entries for all ports on the switch.

Table continues on the next page...

Table 476. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Valid if FA = 010b..101b and ET_EID is not NULL. Note: Not applicable to ENETC function.
63-62	2	CTD	Cut-Through Disable 00b = Do not override cut-through state. 01b = Disable cut-through for the outgoing port specified in the EPORT field. Cut-through is not disabled for the other ports specified in destination port bitmap (i.e. EGRESS_PORT_BITMAP field of this entry). 10b = Disable cut-through for all ports specified in the destination port bitmap (i.e. EGRESS_PORT_BITMAP of this entry). 11b = Reserved. Valid if FA = 010b .. 101b. All other values reserved. Note: Not applicable to ENETC function.
95-64	32	ISQ_EID	Ingress Sequence Generation Entry ID This field specifies the Ingress Sequence Generation Entry ID to be used as an index into the Ingress Sequence Generation table. Valid only when ISQA is set to 1. 0x0..ISQGITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); ingress sequence generation function is by-passed Other values are reserved Note: Not applicable to ENETC functions.
127-96	32	RP_EID	Rate Policer Entry ID This field specifies the Rate Policer Entry ID to be used as an index into the Rate Policer table. Valid if ORP is set to 1. Override Rate Policer Entry ID with this value when ORP =1. 0x0..RPITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); rate policing function is by-passed Other values are reserved
159-128	32	SGI_EID	Stream Gate Instance Entry ID This field specifies the Stream Gate Instance Entry ID to be used as an index into the Stream Gate Instance table. 0x0..SGIITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); stream gating function is by-passed Other values are reserved
191-160	32	IFM_EID	Ingress Frame Modification Entry ID This field specifies the Ingress Frame Modification Entry ID which is encoded either as an index into the Frame Modification table or encoded as a set of frame modification actions. For the latter,

Table continues on the next page...

Table 476. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			there is no need to access the Frame Modification table as the actions are encoded in the Ingress Frame Modification Entry ID itself. If IFM_EID is set to 0xFFFF_FFFF (NULL), no ingress frame modification is performed. If IFM_EID is not set to NULL, bits 15-0 are used for IFM_EID, upper bits are ignored. Refer to Table Frame Modification Entry Definition for IFM_EID format. Note: Not applicable to ENETC function
223-192	32	ET_EID	Egress Treatment Entry ID This field specifies the index (or base index) to be used when accessing the Egress Treatment table. This field is valid if FA field is set to 010b .. 101b, and the OETEID field is set to value other than 00b. 0x0..ETTCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); egress treatment processing is by-passed Other values are reserved Note: Not applicable to ENETC function.
255-224	32	ISC_EID	Ingress Stream counter Index Index to a set of counters in common memory that will be updated for this Stream. 0x0..ICITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); no Ingress Stream Count table counters are updated Other values are reserved
279-256	24	EGRESS_PORT_BITMAP	Egress Port bitmap For FA = 010b, 100b (stream forwarding), this field identifies the destination port(s) to which the frame is to be forwarded for this stream. For FA = 011b, 101 (802.1Q bridge forwarding), this field identifies destination ports to which egress packet processing is required. The destination port(s) are represented as a bitmap, where each bit of the bitmap corresponds to a port on the switch. Least significant bit of the bitmap corresponds to the smallest port number; i.e. bit offset 0 of the bitmap corresponds to port numbered 0, bit offset 1 to port numbered 1, and so on. The port(s) to which the frame are to be forwarded (stream forwarding) and/or egress packet processing is required (VLAN/FDB), are identified by having their corresponding bit set to 1 in the bitmap. Valid bits in the bitmap is from 0..CAPR0[NUM_LINKS]-1 Bits beyond the valid range will be ignored. Note: Not applicable to ENETC function.
287-280	8	--	Reserved.
303-288	16	SI_MAP	Station Interface Map This field specifies a pre L2 filtering SI bitmap. Each bit of the bitmap corresponds to

Table continues on the next page...

Table 476. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			an SI on a given port. Least significant bit of the bitmap corresponds to the smallest SI number. The SIs to which the frame is to be delivered, are identified by having their corresponding bit set to 1 in the bitmap. Bit wise setting of Station Interface map. LSB is the lowest order of SI which is the primary SI, rest are VSIs. Number of VSIs are specified in CAPR0[NUM_VSI]. Bits beyond the valid range will be ignored. Note: This field is only valid for ENETC function when FA field is set to 10b.

53.4.2.3.2.2.6 Ingress Stream Filter Table

Table 477. Table ID 32 - Ingress Stream Filter Table Common Attributes

TABLE_ID	32		
TABLE_VERSION	0		
Table Type	Hash table		
Table Description	The Stream Filter table contains PSFP (IEEE 802.1Qci) stream filter information for stream filters selected based on the 3-bit PCP value (priority) of the received frame and the stream handle (i.e. Ingress Stream Entry ID). Table management command operations Add, Delete, Update and Query are supported.		
Number of Entries	The memory allotment to store the hash entries is shared across all of the (ENETC instance/ switch) function hash tables. The maximum amount of memory words available to store all of the function hash tables entries, is indicated in the register HTMCAPR[NUM_WORDS]). If the number of memory words in-use to store hash table entries for the function has reached this limit, the Add operation will fail. The number of memory words in-use by the function for hash table entries is indicated in the register HTMOR[AMOUNT].		
Default Reset Behavior	No entries in the table by default.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query.
	Exact Match Key Element Match (KEY_ELEMENT)	Yes	Following commands are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x8 = Add. 0xC = Add, followed by a Query.

Table continues on the next page...

Table 477. Table ID 32 - Ingress Stream Filter Table Common Attributes (continued)

	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	Yes	Only supported when command operation = 0x4 (Query). In the event of a search command requiring multiple iterations (see RESUME_ENTRY_ID), the user must not delete any entries in the table between iterations of that search.	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (ISF_EID)	4	4
	ACCESS_KEY	Access Key	8	4
	KEYE_DATA	Key Element	8	4
	CFGE_DATA	Configuration Element	16	4
Entry Alignment [bytes]	4			
Entry ID Management	Hardware assigns entry IDs.			
Response STATUS Applicable	Yes			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x280 = 0x280 = IS_EID in KEYE_DATA is invalid. 0x281 = Any of the following in CFGE_DATA is invalid - One or more of following Entry IDs are not in valid range or Entry ID specified is not Null. Checks are performed for following Entry IDs CFGE DATA – RP_EID, SGI_EID, ISC_EID. - SDU_TYPE is invalid			

Table 478. Table ID 32 - Ingress Stream Filter Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGEU	
ACCESS_KEY																																0x4
ACCESS_KEY																																0x8
CFGE_DATA																																0xC

Table continues on the next page...

Table 478. Table ID 32 - Ingress Stream Filter Table Request Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
...																																...
CFGE_DATA																																0x18

Table 479. Table ID 32 - Ingress Stream Filter Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. 0x1 = Query ENTRY_ID only. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 32 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
95-32	64	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. Bits 63-32: Reserved. If ACCESS_METHOD = 0x1 (Key Element Match): Bits 63-0: KEYE_DATA. KEYE_DATA Format. If ACCESS_METHOD = 0x2 (Search): Bits 31-0: SEARCH_CRITERIA. SEARCH_CRITERIA Format. Bits 63-32: Reserved. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.

Table continues on the next page...

Table 479. Table ID 32 - Ingress Stream Filter Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
223-96	128	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 480. Table ID 32 - Ingress Stream Filter Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS																																0x0
ENTRY_ID																																0x4
KEYE_DATA																																0x8
KEYE_DATA																																0xC
CFGE_DATA																																0x10
...																																...
CFGE_DATA																																0x1C

Table 481. Table ID 32 - Ingress Stream Filter Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	STATUS	STATUS Bits 31-0: RESUME_ENTRY_ID - An Entry ID which can be used as the RESUME_ENTRY_ID value within the ACCESS_KEY of a subsequent search to continue the search. When RESUME_ENTRY_ID is returned as the NULL Entry ID, this indicates that the search was completed successfully and no further iterations are required. This field is valid only in responses for operations which use the Search as the Access Method. For all other Access Methods, this field is reserved. Note: A NULL Entry ID is 0xFFFF_FFFF For generic details on the STATUS field, see Status. For generic details on the Search Access Method, see Search Access Method.
63-32	32	ENTRY_ID [1]	Entry ID When an unassigned entry ID is used to access an entry, hardware needs to process the command as if the entry were not found. For generic details on the Entry ID field, see Entry ID Management.
127-64	64	KEYE_DATA[1]	Key Element Data See KEYE_DATA_Format.

Table continues on the next page...

Table 481. Table ID 32 - Ingress Stream Filter Table Response Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
255-128	128	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 482. Table ID 32 - Ingress Stream Filter Table KEYE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
IS_EID																																0x0
--																													PCP			0x4

Table 483. Table ID 32 - Ingress Stream Filter Table KEYE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	IS_EID	Ingress Stream Entry ID Range: 0 to ISITCAPR[NUM_ENTRIES]-1 0x0..ISITCAPR[NUM_ENTRIES]-1 Other values are reserved
34-32	3	PCP	Priority Code Point Outer VLAN TAG PCP of the received frame
63-35	29	--	Reserved.

Table 484. Table ID 32 - Ingress Stream Filter Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
MSDU																--	SDU_TYPE	ORP	CTD	OSGI	TIMECAPE	IMIRE	ODR	DR	OIPV	IPV						0x0
RP_EID																																0x4
SGI_EID																																0x8
ISC_EID																																0xC

Table 485. Table ID 32 - Ingress Stream Filter Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
3-0	4	IPV	Internal Priority Value New Internal Priority Value (IPV) to be assigned to the frame, if OIPV is set to 1. The least significant 3 bits are used, most significant bit is reserved.
4	1	OIPV	Override Internal Priority Value (IPV) 0b = Don't override frame IPV. 1b = Override frame IPV with value configured in IPV field of this entry.
6-5	2	DR	Drop Resilience New Drop Resilience (DR) to be assigned to the frame, if ODR is set 1.
7	1	ODR	Override Drop Resilience (DR) 0b = Don't override frame DR. 1b = Override frame DR with value configured in DR field of this entry.
8	1	IMIRE	Ingress Mirroring Enable 0b = No ingress mirroring action specified in this entry. 1b = The frame is mirrored to the mirror destination specified in the IMDCR0 register. Note: Not applicable to ENETC function.
9	1	TIMECAPE	Timestamp Capture Enable 0b = No change to the timestamp. 1b = Timestamp is to be captured. Note: Not applicable to ENETC function.
10	1	OSGI	Override Stream Gate Instance Entry ID 0b = Stream Gate Instance Entry ID not overridden 1b = Stream Gate Instance Entry ID overridden; specified SGI_EID in this entry overrides the default null SGI_EID or SGI_EID specified in Ingress Stream table.
11	1	CTD	Cut-Through Disable 00b = Do not override cut-through state. 01b = Disable cut-through for all destined ports. Note: Not applicable to ENETC function.
12	1	ORP	Override Rate Policer (instance) ID 0b = Rate policer instance ID not overridden 1b = Rate policer instance ID overridden If the rate policer instance was already set by a match in the Ingress Port Filter table or Ingress Stream table and if ORP is set to 1, the rate policer instance value is overwritten with the value configured in the RP_EID field of this entry.
14:13	2	SDU_TYPE	Service Data Unit type to use for MSDU (Maximum Service Data Unit) field. Overhead values are specified by register PRXSDUOR. 00b = PPDU (Physical Layer PDU). 01b = MPDU (MAC PDU).

Table continues on the next page...

Table 485. Table ID 32 - Ingress Stream Filter Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			10b = MSDU (MAC SDU). All other values reserved. Note: This field has to be valid if MSDU field is not 0.
15	1	--	Reserved.
31-16	16	MSDU	Maximum Service Data Unit This field specifies the maximum service data unit size defined for the stream. This is part of IEEE 802.1Qci (Per-Stream Filtering and Policing) specification. If the received frame length exceeds this value, the frame is discarded. A value of zero in this field, disables the maximum SDU check. This field overrides the maximum SDU check field specified in the previous matched Ingress Stream table entry. Note: This check is not performed for cut-through frames.
63-32	32	RP_EID	Rate Policer Entry ID This field specifies the Rate Policer Entry ID to be used as an index into the Rate Policer table. Valid if ORP is set to 1. Override Rate Policer Entry ID with this value when ORP = 1. 0x0..RPITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); rate policing function is by-passed Other values are reserved
95-64	32	SGI_EID	Stream Gate Instance Entry ID This field specifies the Stream Gate Instance Entry ID to be used as an index into the Stream Gate Instance table. Stream Gating action will be performed if SGI_EID is not null. Valid if OSGI is set to 1. 0x0..SGIITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); stream gating function is by-passed Other values are reserved
127-96	32	ISC_EID	Ingress Stream Count Entry ID Index to a set of counters in common memory that will be updated for this stream. This field overrides the Ingress Stream Counter ID field specified in the previous matched Ingress Stream table entry. 0x0..ICITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); no Ingress Stream Count table counters are updated Other values are reserved

Table 486. Table ID 32 - Ingress Stream Filter Table SEARCH_CRITERIA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RESUME_ENTRY_ID																																0x0

Table 487. Table ID 32 - Ingress Stream Filter Table SEARCH_CRITERIA Description

Bits	Width [bits]	Name	Description
31-0	32	RESUME_ENTRY_ID	Resume Entry ID When starting a search, pass the NULL Entry ID. To continue a search, pass the RESUME_ENTRY_ID value returned in the CBD Response of the previous search command. Note: A NULL Entry ID is 0xFFFF_FFFF

53.4.2.3.2.2.7 Stream Gate Instance Table

Table 488. Table ID 36 - Stream Gate Instance Table Common Attributes

TABLE_ID	36
TABLE_VERSION	0
Table Type	Dynamic bounded index table
Table Description	The Stream Gate Instance (SGI) table contains the stream gate configuration and operational information for a NETC function. There is one SGI table for each ENETC instance and one SGI table for the entire switch. Each entry contains a set of parameters that defines a single stream gate instance. The stream gate instance parameters includes an administrative stream gate control list and an operational stream gate control list, along with related configuration and status parameters. A stream gate control list (SGCL) is used to specify time gates or time slots during which incoming frames from one or more streams will be accepted. This table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. The entries are stored in contiguous manner in the common memory. For this table, each table entry occupies one word (one to one corresponding between words and entries). Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) registers; S0SGIITMAR[NUM_WORDS] for switch and EaSGIITMAR[NUM_WORDS] for ENETC. These allocations can also be changed when the functions are initialized through their respective base function registers SGIITMAR[NUM_WORDS]. The index to access this table, is obtained earlier on, via lookups in the Ingress Stream or in the Ingress Stream Filter table. An SGI can hold two SGCLs, one as operational and the other as administrative. An administrative SGCL becomes operational at Config Change Time as defined in the IEEE 802.1Qci standard and remains operational till another SGCL is installed. An administrative SGCL is added to a SGI using the Add or Update operations. The actual SGCLs are stored in a separate table called the Stream Gate Control List table, with references to these SGCLs (that is, entry IDs) specified in the SGI table. It is recommended that the entry ID for each of these two SGCLs be predetermined and known by software, and persist till the SGI is no longer in use. Take the following into consideration when accessing the SGI table[1]: • **Initialization** - The 1588 timer must be initialized and configured before adding any SGCL to an SGI. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer is in normal operation, any change to the 1588 timer configuration (for example TMROFF_H/L), except for TMR_ADD updates, requires that all SGIs with an operational SGCL be deleted, and for SGIs that have no operational SGCLs, that the administrative SGCL, if present, be removed from the SGI.

Table continues on the next page...

Table 488. Table ID 36 - Stream Gate Instance Table Common Attributes (continued)

	• **Stream association -** An SGI can be associated with only one switch port. Streams from different switch ports cannot use the same SGI. • **Config Change Time** - Note that when Config Change Time has been reached, the actual installation of a new SGCL, will occur at the next event that will cause the operational SGCL to be traversed; that is reception of a frame or expiry of the refresh timestamp timer. For any SGI command processed by hardware prior to this event, the administrative SGCL is considered to be still pending although the Config Change Time may have been reached. • **Add operation** - When an ADMIN_SGCL_EID is specified in the Add operation, the SGI will operate with the initial SGI configuration till the ADMIN_SGCL_EID is installed at Config Change Time. An SGI can operate without an SGCL and with just the initial SGI configuration. To operate in this mode, specify a NULL ADMIN_SGCL_EID in the Add operation. • **Update operation** - This operation is used to add a new administrative SGCL to an existing SGI or to remove an administrative SGCL from an SGI. For the details on how to add a new administrative SGCL, refer to the sequence described below in Adding an administrative SGCL to an SGI. For the details on how to remove an administrative SGCL, refer to the sequence described below in Removing an administrative SGCL from an SGI. Note that if the operational SGCL is present, it will continue to operate as normal. • **Delete operation** - Perform the following steps to delete an SGI table entry: 1. Remove, if present in the SGI, the administrative SGCL by means of an SGI command containing an Update operation to the SGI's Admin Configuration Element with ADMIN_SGCL_EID set to NULL. 2. Issue a Delete operation against the SGI table entry that is to be deleted. 3. If needed, delete the SGCLs from the Stream Gate Control List table. Since there are two possible SGI_EIDs, which the deleted SGI may have used, software can then safely issue SGCL commands containing a Delete operation to those two SGCLs. A Delete operation to an SGCL has not been added before or has been already deleted, will execute with no error; that is, the Delete operation is not performed, and the response message indicates that no entry has been found (NUM_MATCHED in Response Message Header is set to 0). ***Adding an administrative SGCL to a SGI*** Follow these steps to add an administrative SGCL to an SGI: 1. Since software has only two possible SGCLs (and therefore, SGCL_EIDs) to use, software must first verify that at least one of them is unused by the hardware, and therefore, "free" to be configured and associated to the SGI. This procedure is started by issuing a query on the SGI to record the presence of both the administrative and operational SGCLs (found in SGISE_DATA[STATE]), and the SGCL_EIDs (found in ACFGE_DATA[ADMIN_SGCL_EID] and SGISE_DATA[OPER_SGCL_EID]), respectively, if valid, based on the value of SGISE_DATA[STATE]. • **Case 0** - Neither SGCL is present. An administrative SGCL may be configured and associated to the SGI. Proceed to Step 3. • **Case 1** - An administrative SGCL only is present. Execute the sequence described below in Removing an administrative SGCL from an SGI, but starting at step 2 (as step 1 has already performed in this sequence), to remove the administrative SGCL and then retry this sequence.

Table continues on the next page...

Table 488. Table ID 36 - Stream Gate Instance Table Common Attributes (continued)

	• **Case 2** - An administrative SGCL and an operational SGCL is present. Both SGCLs are currently in use; a new one may not be configured. Execute the sequence described below in Removing an administrative SGCL from an SGI, but starting at step 2 (as step 1 has already performed in this sequence), to remove the administrative SGCL and then retry this sequence. • **Case 3** - An operational SGCL only is present. An administrative SGCL can be configured and associated to the SGI. Proceed to Step 2. 2. Ensure that any stale SGCL is deleted from the SGCL table; a stale SGCL is an operational SGCL that ceased to be operational as it was replaced by a new SGCL. a. Since there are two possible SGI_EIDs which this SGI may use, any of the two lists which are not present, may be deleted. The SGI_EID(s) present in the SGI is indicated based on the combination of SGISE_DATA[STATE] and ACFGE_DATA[ADMIN_SGCL_EID]/SGISE_DATA[OPER_SGCL_EID]. b. Software can then safely issue SGCL commands containing a Delete operation to those SGCLs. A Delete operation to an SGCL has not been added before or has been already deleted, will execute with no error; i.e. the Delete operation is not performed, and the response message indicates that no entry has been found (NUM_MATCHED in Response Message Header is set to 0). 3. Software may chose an SGCL from its pool of two SGCLs for this SGI which is currently unused by the hardware (or not present) and issue a command containing an Add operation to SGCL table, to add the new administrative SGCL. 4. The new added SGCL may now be associated with the SGI by means of an SGI command containing an Update operation to the SGI's Admin Configuration Element with the new SGCL (e.g. setting ADMIN_SGCL_EID to new added SGCL_EID). ***Removing an administrative SGCL from an SGI*** Follow these steps to remove an administrative SGCL from an SGI: 1. This procedure is started by issuing a query on the SGI. Software records the presence of both the administrative and operational SGCLs (found in SGISE_DATA[STATE]), and the SGCL_EIDs (found in ACFGE_DATA[ADMIN_SGCL_EID] and SGISE_DATA[OPER_SGCL_EID]), respectively, if valid, based on the value of SGISE_DATA[STATE]. 2. Remove the administrative SGCL by means of an SGI command containing an Update operation to the SGI's Admin Configuration Element with ADMIN_SGCL_EID set to NULL. 3. Query the SGI again. a. If the query response indicates that no administrative SGCL is present, and if the “operational SGCL” encoding from SGISE_DATA[STATE] and SGISE_DATA[OPER_SGCL_EID] (if operational SGCL is present) are matching the queried values from Step 1, this indicates that the administrative SGCL was successfully removed from the SGI. Proceed to Step 4. b. If software finds that either an operational SGCL is now present when it did not in the Step 1 query, or if the SGCL_EID of the operational SGCL is different from the one in the Step 1 query, this indicates that the update command in Step 2 was processed by the hardware only after the administrative SGCL (which software was attempting

Table continues on the next page...

Table 488. Table ID 36 - Stream Gate Instance Table Common Attributes (continued)

	to uninstall) had become operational. In this case, software must not proceed to delete the SGCL. Do not proceed to step 4. 4. The administrative SGCL was successfully removed from the SGI, it is now safe to issue a SGCL command containing a Delete operation to the administrative SGCL.			
Number of Entries	Maximum number of entries is indicated in SGIITCAPR[NUM_ENTRIES]. Number of entries in-use is indicated in SGIITOR[NUM_ENTRIES].			
Default Reset Behavior	No entries in the table by default.			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x8 = Add.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (SGI_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	ACFGE_DATA	Admin Configuration Element	16	4
	ICFGE_DATA	Initial Configuration Element	1	1
	CFGE_DATA	Configuration Element	1	1
	SGISE_DATA	Stream Gate Instance State Element	13	4
Entry Alignment [bytes]	4			

Table continues on the next page...

Table 488. Table ID 36 - Stream Gate Instance Table Common Attributes (continued)

Entry ID Management	Software assigns entry IDs
Response STATUS Applicable	No
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x2C0 = SGCL_EID specified in out of range for Add or Update operation. Valid SGCL_EID values are {0..SGCLITCAPR[NUM_WORDS]-1}, NULL (0xFFFF_FFFF). 0x2C1 = SDU_TYPE is specified is invalid for Add or Update operation. 0x2C2 = Either the SGCL_EID specified as admin gate control list in Add or Update operation has not been allocated or SGCL_EID is not the first entry in gate control list, or the reference count in SGCL entry is not 0. 0x2C3 = SGCL_EID specified for Update operation is in invalid. When update is performed for an existing Admin control list, SGCL_EID specified in Update operation has to match that in the Stream Gate Instance. An Admin list exist in the Stream gate Instance if STATE in SGISE_DATA element = 2 or 3. 0x2C4 = ADMIN_BASE_TIME specified for Add or Update operation is more than 2^30ns in the past. (that is, current time - ADMIN_BASE_TIME is <2 ^30ns) 0x2C5 = Cumulated time value of CYCLE_TIME in Stream gate Control list plus CYCLE_TIME_EXT specified in Add or Update operation is >=2^30ns or CYCLE_TIME specified is 0.

1. The command sequences described in these considerations, assume that the entry ID for each of the two SGCLs be predetermined and known by software

Table 489. Table ID 36 - Stream Gate Instance Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--													SGISEU	CFGEU	ACFGEU	
ACCESS_KEY																																0x4
ACFGE_DATA																																0x8
...																																...
ACFGE_DATA																																0x14
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																ICFGE_DATA																0x18
																CFGE_DATA																

Table 490. Table ID 36 - Stream Gate Instance Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: ACFGEU - Admin Configuration Element Update. 0b = No update performed to the Admin Configuration Element. 1b = The data specified in the ACFGE_DATA field is written to the Admin Configuration Element of the table entry. Bit 1: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bit 2: SGISEU - Stream Gate Instance State Element Update. 0b = No update performed to the Stream Gate State Element. 1b = Reset the IRX, OEX flags to 0 Bits 15-3: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 36 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
191-64	128	ACFGE_DATA [1]	Admin Configuration Element Data See ACFGE_DATA Format.
199-192	8	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA Format.
207-200	8	ICFGE_DATA[1]	Initial Configuration Element Data See ICFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 491. Table ID 36 - Stream Gate Instance Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
SGISE_DATA																																0x4
...																																...
--								ICFGE_DATA								CFGE_DATA						SGISE_DATA										0x1C
ACFGE_DATA																																0x20
...																																...
ACFGE_DATA																																0x2C

Table 492. Table ID 36 - Stream Gate Instance Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID For generic details on the Entry ID field, see Entry ID Management.
231-32	200	SGISE_DATA[1]	Stream Gate Instance State Element Data See SGISE_DATA_Format.
239-232	8	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.
247-240	8	ICFGE_DATA[1]	Initial Configuration Element Data See ICFGE_DATA_Format.
255-248	8	--	Reserved. Alignment padding.
383-256	128	AFGE_DATA[1]	Admin Configuration Element Data See ACFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 493. Table ID 36 - Stream Gate Instance Table ACFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ADMIN_SGCL_EID																																0x0
ADMIN_BASE_TIME																																0x4
ADMIN_BASE_TIME																																0x8
ADMIN_CYCLE_TIME_EXT																																0xC

Table 494. Table ID 36 - Stream Gate Instance Table ACFGE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	ADMIN_SGCL_EID	Administrative Stream Gate Control List Entry ID This field specifies the Entry ID in the Stream Gate Control List table (expressed in units of words) of the administrative stream gate control list to be used for this stream gate instance. 0x0..SGCLITCAPR[NUM_WORDS]-1 0xFFFF_FFFF (NULL); implies no Gate Control List specified. Other values are reserved
95-32	64	ADMIN_BASE_TIME	Admin Base Time This field sets the base time, that is, the start time, of the administrative gate control sequence. The time is expressed in units of nanoseconds. The field implements the PSFPAdminBaseTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 PSFPAdminBaseTime parameter, which is the PTP time format, consisting of a 48-bit integer number of seconds and a 32-bit integer number of nanoseconds. Instead of using the PTP time format, the field expects the time to be specified as a 64-bit integer number of nanoseconds. Therefore, if this field is to be set to a value which was obtained from a managed object using the IEEE 802.1Q-2018 PTP time format, the value must be converted by software from the IEEE 802.1Q-2018 format to the 64-bit integer number of nanoseconds. This field is reserved when ADMIN_SGCL_EID is NULL.
127-96	32	ADMIN_CYCLE_TIME_EXT	Admin Cycle Time Extension The value supplied in this field is equivalent with the 802.1.Qci definition of "Cycle Time Extension". When the administrative stream gate control list becomes operational, this is the maximum amount of time by which the cycle time is permitted to be extended before a new administrative stream gate control list takes effect. The time is expressed as an integer number of nanoseconds. Only least significant 30 bits are valid. Upper two bits are ignored. It is required that CYCLE_TIME in Stream gate control list + ADMIN_CYCLE_TIME_EXT must be < 2^{30} -1. This field is reserved when ADMIN_SGCL_EID is NULL.

Table 495. Table ID 36 - Stream Gate Instance Table ICFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																								--	CTD	GST	OIPV	IPV				0x0

Table 496. Table ID 36 - Stream Gate Instance Table ICFGE_DATA Description

Bits	Width [bits]	Name	Description
3-0	4	IPV	Internal Priority Value (IPV) This field specifies the IPV that will be assigned to frames before the first administrative stream gate control list takes affect, otherwise the IPV value is determined by the stream gate control list entry. Valid if OIPV of this entry is set to 1. The least significant 3 bits are used, most significant bit is reserved.
4	5	OIPV	Override Internal Priority Value (IPV) 0b = Don't override frame IPV. 01 = Override frame IPV with the value configured in the IPV field of this entry if the first administrative stream gate control list hasn't taken effect, otherwise overwrite with the IPV value stored in the stream gate control list.
5	1	GST	Gate State This field specifies the gate state to use before the administrative stream gate control list takes affect. 0b = Closed; frames are not permitted to pass through. 1b = Open; frames are permitted to pass through.
6	1	CTD	Cut Through Disabled This field specifies whether to disable cut-through before the administrative stream gate control list takes affect. 0b = No action. 1b = Cut-through is disabled. Note: Not applicable to ENETC function.
7	1	--	Reserved.

Table 497. Table ID 36 - Stream Gate Instance Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																								--				SDU_TYPE		IRXEN	OEXEN	0x0

Table 498. Table ID 36 - Stream Gate Instance Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
0	1	OEXEN	Octets Exceeded Enable 0b = "Gate Closed Due To Octets Exceeded" function is disabled. 1b = "Gate Closed Due To Octets Exceeded" function is enabled. See OEX for status of this function.

Table continues on the next page...

Table 498. Table ID 36 - Stream Gate Instance Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
1	1	IRXEN	Invalid Receive Enable 0b = "Gate Closed Due To Invalid Rx" function is disabled. 1b = "Gate Closed Due To Invalid Rx" function is enabled. See IRX for status of this function.
3-2	2	SDU_TYPE	Specifies the type of PDU/SDU (Protocol/Service Data Unit) for Interval Octets Maximum check for Gate Entry. Refer to IOM_GE field in SGCL table) Overhead values are specified by register PRXSDUOR. 00b = PPDU (Physical Layer PDU). 01b = MPDU (MAC PDU). 10b = MSDU (MAC SDU). All other values reserved.
7-4	4	--	Reserved.

Table 499. Table ID 36 - Stream Gate Instance Table SGISE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
OPER_SGCL_EID																																0x0
CONFIG_CHANGE_TIME																																0x4
CONFIG_CHANGE_TIME																																0x8
OPER_BASE_TIME																																0xC
OPER_BASE_TIME																																0x10
OPER_CYCLE_TIME_EXT																																0x14
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						--			STATE					IRX	OEX	0x18

Table 500. Table ID 36 - Stream Gate Instance Table SGISE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	OPER_SGCL_EID	Operational Stream Gate Control List Entry ID This field indicates the Entry ID in the Stream Gate Control List table (expressed in units of words) of the stream gate control list that is currently operational. Set to NULL (0xFFFF_FFFF) if there is no Operational Stream Gate Control List.
95-32	64	CONFIG_CHANGE_TIME	Configuration Change Time If an administrative gate control list is pending, this is the time at which administrative gate control list is to become operational (implements the PSFPConfigChangeTime parameter as defined by IEEE 802.1Q-2018). If there is no administrative gate control

Table continues on the next page...

Table 500. Table ID 36 - Stream Gate Instance Table SGISE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			list and if an operational gate control list exists then this is the time the operational list became active. The actual value stored by hardware (and thus available to query) is expressed in units of nanoseconds. If this field is to be queried from an IEEE 802.1Q-2018 managed object, the value must be converted by software from the 64-bit integer number of nanoseconds to the IEEE 802.1Q-2018 PTP time format.
159:96	64	OPER_BASE_TIME	Operational Base Time The base time (start time) of the operational stream gate control sequence. The actual value stored by hardware (and thus available to query) is expressed in units of nanoseconds. The field implements the PSFPOperBaseTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 PSFPOperBaseTime parameter, which is the PTP time format, consisting of a 48-bit integer number of seconds and a 32-bit integer number of nanoseconds. Instead of using the PTP time format, the field expresses the time as a 64-bit integer number of nanoseconds. Therefore, if this field is to be queried from an IEEE 802.1Q-2018 managed object, the value must be converted by software from the 64-bit integer number of nanoseconds to the IEEE 802.1Q-2018 PTP time format.
191-160	32	OPER_CYCLE_TIME_EXT	Oper Cycle Time Extension The value supplied in this field is equivalent with the 802.1.Qci definition of "Cycle Time Extension". If Operational Gate Control List is active, this is the maximum amount of time by which the cycle time is permitted to be extended before a new Admin Gate Control List takes effect. Only least significant 30 bits are valid. Upper two bits are ignored.
192	1	OEX	Octets Exceeded Flag 0b = Normal operation. 1b = Gate closed permanently due to octets exceed. If OEXEN = 1, detection of octets received exceeding INTERVAL_OCTET_MAX in a gate interval results in the stream gate being permanently set to closed. OEX can be cleared by performing Stream Gate State Element Update (SGISEU) action.
193	1	IRX	Invalid Receive Flag 0b = Normal operation. 1b = Gate closed permanently due to invalid receive. If IRXEN=1, detection of incoming frames during time periods when the stream gate is in the closed state results in the stream gate being permanently set to closed state. IRX can be cleared by performing Stream Gate State Element Update (SGISEU) action.
195-194	3	STATE	Current Gate Instance State 000b = Reserved. 001b = Gate instance is operational but no stream gate control list specified. Default Gate Instance parameters are applied to received frame. 010b = New administrative stream gate control list pending. Default Gate Instance

Table continues on the next page...

Table 500. Table ID 36 - Stream Gate Instance Table SGISE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			parameters will be applied to frames till administrative stream gate control list takes effect. 011b = New administrative stream gate control list is pending while stream gate control list is operational. 100b = No administrative stream gate control list is pending. Operational stream gate control list is active. 101b .. 111b = Reserved.
199-196	3	--	Reserved.

53.4.2.3.2.2.8 Stream Gate Control List Table

Table 501. Table ID 37 - Stream Gate Control List Table Common Attributes

TABLE_ID	37
TABLE_VERSION	0
Table Type	Dynamic bounded index table
Table Description	This table stores the stream gate control lists (SGCLs). A SGCL is used to specify the time gates or time slots during which incoming frames from one more streams will be accepted. Each SGCL can have a variable number of gate control operation entries. For each Stream Gate Instance used, only two SGCLs may be allocated to the instance. It is recommended that the entry ID for each of these two SGCLs be predetermined and fixed by software, and persist till the Gate Stream Instance is no longer in use. This table is implemented as a linear array of words, accessed using an index (0, 1, 2, ..., n) that uniquely identifies a word within the array. The words are stored in a contiguous manner in the common memory. When accessing the table, the actual memory position of a word is calculated by hardware from the index based on the size of the word and the base memory address of the table. Each table entry is of variable size, and is used to hold one stream gate control list. The first word of the entry contains information specific to the entire stream gate control list. Subsequent consecutive words are contains gate control operation (or gate) specific configuration. Each word can hold 2 gate configurations. Number of words required for a stream gate control list is 1+N/2 where N is number of gates in the stream gate control list. If N is odd, only one gate configuration will be specified in the last word. The Stream Gate Control Entry ID (SGCL_EID) has to be specified in units of words such that it points to the first word in a stream gate control list. Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) registers; S0SGCLITMAR[NUM_WORDS] for switch and EaSGCLITMAR[NUM_WORDS] for ENETC. These allocations can also be changed when the functions are initialized through their respective base function registers SGCLITMAR[NUM_WORDS]. Table management command operations are performed using Add, Delete or Query operations. Note that the Update operation is not supported for this table. Memory fragmentation may occur if stream gate control lists of different sizes are repeatedly added/deleted. There is no hardware assist to manage such memory fragmentation, software must take care of this situation.

Table continues on the next page...

Table 501. Table ID 37 - Stream Gate Control List Table Common Attributes (continued)

	Following are the behavior of operations: • Add operation: This operation adds a stream gate control list to the Stream Gate Control List table. Hardware will perform the following checks to determine if the Add operation is valid before executing the Add operation. — Checks if the specified stream gate control list is within the allocated region to the Stream Gate Control List table in common memory. — Checks if the common memory locations for the stream gate control list are not already allocated. — Checks if the cumulative gate time in the stream gate control list is <2^30. If above conditions are not met the operation will be rejected. • Delete operation: This operation deletes a stream gate control list from the Stream Gate Control List table. The SGCL_EID must point to the first word of the stream gate control list that is to be deleted. Following checks are performed before executing the Delete operation. — Checks if SGCL_EID is in valid range of words for this table. — Checks if the stream gate control list has already been added by software. — Checks if stream gate control list is not in use (that is,. stream gate control list is not operational nor installed as an administrative list. Note that REF_COUNT is non zero if the stream gate control list is in use. — If above conditions are not met the operation will be rejected. • Query operation: This operation queries a stream gate control list. The following checks are performed before executing the Query operation. — Checks if SGCL_EID is in valid range of words for this table. — Checks to see if the stream gate control list has already been added. — If above conditions are not met the operation will be rejected. Note that due to dynamic size of stream gate control lists that get added/deleted can cause memory fragments.		
Number of Entries	Maximum number of entries is indicated in SGCLITCAPR[NUM_WORDS]. Number of entries in-use is indicated in SGCLTMOR[NUM_ENTRIES]. Number of words required for a stream gate control list is 1+N/2 where N is number of gate time slots in the stream gate control list.		
Default Reset Behavior	No entries in the table by default.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x4 = Query. 0x8 = Add.
	Exact Match Key Element Match (KEY_ELEMENT)	No	--

Table continues on the next page...

Table 501. Table ID 37 - Stream Gate Control List Table Common Attributes (continued)

	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (SGCL_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	Variable	4
	SGCLSE_DATA	Stream Gate Control List State Element	1	1
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. x2D0 = Number words required for the LIST_LENGTH specified for the Add operation exceeds the number of words allocated for SGCL table. (i.e. SGCL_EID + (LIST_LENGTH+1)/2 (rounding to the next higher integer) has to be less than or equal to SGCLITCAPR [NUM_WORDS]-1). x2D1 = TIME_INTERVAL_GE_N specified in Add operation is 0. Note that upper 2 bits of TIME_INTERVAL_GE_N are ignored, TIME_INTERVAL_GE_N[29:0] must not be 0. x2D2 = Cumulated time value of TIME_INTERVAL_GE_N[29:0] for the gate list specified in Add operation is >= 2^30ns. Where N the List Length specified in operation. Where N is the List Length of the Stream Gate Control List.			

Table 502. Table ID 37 - Stream Gate Control List Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--																
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																...
CFGE_DATA																																Variable

Table 503. Table ID 37 - Stream Gate Control List Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions No update actions supported. Bits 15-0: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 37 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
Variable-64	Variable	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 504. Table ID 37 - Stream Gate Control List Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
--																						SGCLSE_DATA										0x4
CFGE_DATA																																0x8
...																																...
CFGE_DATA																																Variable

Table 505. Table ID 37 - Stream Gate Control List Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Stream, Gate Control List is a variable sized entry. Following has to be true. Entry ID is the first word in SGCL and SGCL_EID + (LIST_LENGTH+1)/2 (rounding to the next higher integer) has to be less than or equal to SGCLITCAPR [NUM_WORDS]-1 For generic details on the Entry ID field, see Entry ID Management.
39-32	8	SGCLSE_DATA [1]	Stream, Gate Control List State Element Data See SGCLSE_DATA_Format.
63-40	24	--	Reserved. Alignment padding.
Variable-64	Variable	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 506. Table ID 37 - Stream Gate Control List Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CYCLE_TIME																																0x0
--									EXT_GTST	EXT_CTD	EXT_IPV				EXT_OIPV	--						LIST_LENGTH										0x4
TIME_INTERVAL_GE_0																																0x8
GTST_GE_0	IOMEN_GE_0	CTD_GE_0	OPIV_GE_0	IPV_GE_0							IOM_GE_0																						0xC
...																																...
TIME_INTERVAL_GE_N																																0x8+8N
GTST_GE_N	IOMEN_GE_N	CTD_GE_N	OPIV_GE_N	IPV_GE_N							IOM_GE_N																						0xC+8N

Table 507. Table ID 37 - Stream Gate Control List Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	CYCLE_TIME	Cycle Time This field specifies the cycle time of the stream gate control list. This is the period of time over which the sequence of operations in a stream gate control list repeats. The time is expressed in units of nanoseconds. Only least significant 30 bits are valid. Upper two bits are ignored. This value has to be a non-zero value else operation will be rejected. It is also required that ADMIN_CYCLE_TIME + ADMIN_CYCLE_TIME_EXT must be < 2^30 -1.
39-32	8	LIST_LENGTH	List Length This field indicates the number of entries in the stream gate control list. Software must set this value to the size of the list being configured minus one. Valid values are 0 (one entry) to 255 (256 entries). The operation will be rejected if list length is invalid.
47-40	8	--	Reserved.
48	1	EXT_OIPV	Extension Override Internal Priority Value This field specifies whether to override the frame IPV after the stream gate control list ends and before CYCLE_TIME restarts. 0b = Don't override frame IPV. 1b = Override frame IPV with the value configured in the EXT_IPV field of this entry.
52-49	4	EXT_IPV	List Extension Internal Priority Value This field specifies the IPV to be assigned to frames after the stream gate control list ends and before CYCLE_TIME restarts. Valid if the EXT_OIPV field of this entry is set to 1. The least significant 3 bits are used, most significant bit is reserved.
53	1	EXT_CTD	Extension Cut Through Disabled This field specifies whether to disable cut-through for frames received after the stream gate control list ends and before CYCLE_TIME restarts. 0b = No action. 1b = Cut-through disabled. Note: Not applicable to ENETC function.
54	1	EXT_GTST	Extension Gate State This field specifies the gate state that will be applied after the stream gate control list ends and before CYCLE_TIME restarts. List started. 0b = Closed. 1b = Open.
63-55	9	--	Reserved.
(127 + 64i)	32	TIME_INTERVAL_GE_i	Time Interval for Gate Entry i, where i = 0, ..., N-1 This field specifies the time interval for a gate entry. The time is expressed in units of nanoseconds. Only least significant 30 bits are valid. Upper two bits are ignored.

Table continues on the next page...

Table 507. Table ID 37 - Stream Gate Control List Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
- (64 + 64i)			This field has to be a non-zero value and the cumulative TIME_INTERVAL amount in a Gate List must not exceed 2^30-1. Cumulated TIME_INTERVAL amount in a Gate List is allowed to be greater than CYCLE_TIME, in such case the CYCLE TIME restarts before Gate List completes.
	24	IOM_GE_i	Interval Octets Maximum for Gate Entry i, where i = 0, ..., N-1 This field specifies the maximum bytes (octets) allowed to pass (open) gate entry i when IOMEN is enabled.
	4	IPV_GE_i	Internal Priority Value for Gate Entry i, where i = 0, ..., N-1 This field specifies the Internal Priority Value (IPV) that will be assigned to frames. Valid if OIPV_GE_i field of this entry is set to 1. The least significant 3 bits are used, most significant bit is reserved.
	1	OIPV_GE_i	Override Internal Priority Value for Gate Entry i, where i = 0, ..., N-1 0b = Don't override frame IPV. 1b = Override frame IPV with the value configured in IPV_GE_i of field this entry.
	1	CTD_GE_i	Cut Through Disable for Gate Entry i, where i = 0, ..., N-1 This field specifies whether to disable cut-through for frames received in this gate. 0b = No action. 1b = Cut-through is disabled. Note: Not applicable to ENETC function.
	1	IOMEN_GE_i	Interval Octet Maximum Enabled for Gate Entry i, where i = 0, ..., N-1 This field specifies whether to track the number of bytes (octets) received in the gate when the gate is permanently closed; if OEXEN=1 and octets received in this gate exceeds IOM_GE. 0b = Don't track count. 1b = Track count.
	1	GTST_GE_i	Gate State for Gate Entry i, where i = 0, ..., N-1 0b = Closed. 1b = Open. Frames received in this gate are dropped if gate state is closed.

Table 508. Table ID 37 - Stream Gate Control List Table SGCLSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																								REF_COUNT								0x0

Table 509. Table ID 37 - Stream Gate Control List Table SGCLSE_DATA Description

Bits	Width [bits]	Name	Description
0	8	REF_COUNT	Reference Count Indicates whether a stream gate control list (SGCL) is in-use in a Stream Gate Instance (SGI). An SGCL becomes in-use when it is added to an SGI (as the administrative SGCL), and remains in-use when it is installed as the operational SGCL. An administrative SGCL is no longer considered in-use when if it is removed from an SGI. An operational SGCL is no longer in-use when it is replaced with a new SGCL or when the SGI is deleted. 0x00 = Not in-use by an SGI. 0x01 = In-use by an SGI. All other values reserved. Note: An attempt to delete an SGCL with a Reference Count set to 1 will result in an error response; in this case, the SGCL is not deleted from the Stream Gate Control List table.

53.4.2.3.2.2.9 Ingress Stream Count Table

Table 510. Table ID 38 - Ingress Stream Count Table Common Attributes

TABLE_ID	38		
TABLE_VERSION	0		
Table Type	Dynamic bounded index table		
Table Description	Ingress stream related statistics are maintained in this table. The Ingress Stream Count table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. The entries are stored in contiguous manner in a common memory. Ingress Stream Counter Entry ID (ISC_EID) is used as an index to an entry in this table. The ISC_EID is specified in the Ingress Stream table. For this table, each table entry occupies 1 word. Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) registers; S0ISCITMAR[NUM_WORDS] for switch and EaISCITMAR[NUM_WORDS] for ENETC. These allocations can also be changed when the functions are initialized through their respective base function register ISCITMAR[NUM_WORDS]. Table management operations Add, Delete, Update and Query are supported.		
Number of Entries	Maximum number of entries is indicated in ISCITCAPR[NUM_ENTRIES]. Number of entries in-use is indicated in ISCITOR[NUM_ENTRIES].		
Default Reset Behavior	No entries in the table by default.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete.

Table continues on the next page...

Table 510. Table ID 38 - Ingress Stream Count Table Common Attributes (continued)

			0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update. 0x8 = Add.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (IGSCNT_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	STSE_DATA	Statistics Element	32	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description.			

Table 511. Table ID 38 - Ingress Stream Count Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															STSEU	
ACCESS_KEY																																0x4

Table 512. Table ID 38 - Ingress Stream Count Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: STSEU - Statistics Element Update. 0b = No update performed to the Statistics Element. 1b = All counters within the Statistics Element are reset. All other values reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 38 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.

Table 513. Table ID 38 - Ingress Stream Count Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
STSE_DATA																																0x4
...																																...
STSE_DATA																																0x20

Table 514. Table ID 38 - Ingress Stream Count Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Valid range of ISC_EID:{0.. ISCITCAPR[NUM_ENTRIES]-1}. For generic details on the Entry ID field, see Entry ID Management.

Table continues on the next page...

Table 514. Table ID 38 - Ingress Stream Count Table Response Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
287-32	256	STSE_DATA[1]	Statistics Element Data See STSE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 515. Table ID 38 - Ingress Stream Count Table STSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RX_COUNT																																0x0
--																																0x4
MSDU_DROP_COUNT																																0x8
--																																0xC
POLICER_DROP_COUNT																																0x10
																																0x14
SG_DROP_COUNT																																0x18
--																																0x1C

Table 516. Table ID 38 - Ingress Stream Count Table STSE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	RX_COUNT	Receive Count Count of received frames in the Ingress Stream table entry or Ingress Stream Filter table entry associated with this counter. \
95-64	32	MSDU_DROP_COUNT	MSDU Drop Count Count of received frames in the Ingress Stream table entry or Ingress Stream Filter table entry associated with this counter that did not pass the maximum SDU filter. Counter implemented in hardware is 32 bits.
159-128	32	POLICER_DROP_COUNT	Policer Drop Count Count of received frames in the Ingress Stream table entry or Ingress Stream Filter table entry associated with this counter that were dropped by the Policer.
223-192	32	SG_DROP_COUNT	Stream Gating Drop Count Count of received frames in the Ingress Stream table entry or Ingress Stream Filter table entry associated with this counter that did not pass the stream gate.

53.4.2.3.3 PSI-VSI Messaging

Hardware supports messages passing between the Virtual Station Interfaces and Physical Station Interface and vice-versa. Depending on the direction of the message, the format also differs since the usage cases vary.

VSIs may send a message to the PSI for general notification or to gain access to hardware resources which requires host inspection. These messages may vary in size and are handled as a partition copy between two memory regions owned by the respective participants. The PSI will respond with fail or success and a 16-bit message code. Both sides can be programmed to receive an interrupt during this event.

The PSI can send a register based message to one or more VSIs. This method is much simpler than the reverse and only provide a 16-bit message code. The intention could be to notify the VSI of a link state change. Only the VSI can be programmed to receive an interrupt during this event.

53.4.2.3.3.1 PSI-to-VSI Messaging

NETC supports sending and receiving 16-bit messages between the Physical Station Interface (PSI) and one or more Virtual Station Interfaces (VSIs).

To send a message from the PSI to one or more VSI the following takes place:

1. The VSI enables message receive interrupt by setting VSIIER[MRIE]=1.
2. The PSI writes the message send register PSIMSGSR with the 16-bit message code in the MC field and sets one or more MS bits to indicate the recipient of the message. Note that one message at a time is supported from the PSI to one or more VSIs.
3. The VSI will be interrupted when the message is received, alternatively the VSI can poll register bit VSIMSGRR[MR].
4. The VSI reads VSIMSGRR to get the message code from the MC field. The VSIMSGRR[MR] bit and the PSIMSGSR[MS] bit will clear automatically.
5. PSI detects PSIMSGSR[MS]=0 and knows the corresponding VSI has received the message.

53.4.2.3.3.2 VSI-to-PSI Messaging

NETC supports sending and receiving programmable sized messages between the Virtual Station Interface(s) and Physical Station Interface. The generic message format consists of a multiple of 32 bytes. The message format is software defined, NETC puts no restrictions on the content of a message.

To send a message from a virtual station interface (VSI) to the physical station interface (PSI), the following steps takes place:

1. VSI*n* guest software creates a message in memory.
2. VSI*n* guest software writes message send register VSIMSGSNDAR0 with lower address of message location in memory and the message size. This will set the message busy bit in the message status register VSIMSGSR[MB]=1. Note that this will also clear the previous message status and code in the message status register VSIMSGSR[MS/MC].
3. NETC will copy the message to the PSI's memory address space as indicated by the PSI-VSI message receive registers PSIV*a*MSGRCVAR0/1.
4. When the message has been copied, the message received bit is set in the PSI receive register, PSIMSGRR[MR*n*]. The PSI may set the interrupt enable bit SIIER[MR*a*IE] associated with a message received to receive an interrupt, or may opt to poll the message receive register PSIMSGRR.

   If a transfer error is encountered when copying the data, The PSI's message receive bit will never get set, only the VSI is notified by the error in VSIMSGSR[MS].
5. The host software will process the message and when ready to respond clear the message received bit by writing a 1 to PSIMSGRR[MR*n*]=1 (write to clear). At the same time the code field, PSIMSGRR[MC], is written to indicate a software defined response type back to the VSI. NETC will clear the senders busy bit VSIMSGSR[MB] and set the code field VSIMSGSR[MC] by copying from PSIMSGRR.

6. The sender can poll the message busy bit VSIMSGSR[MB] to see if it has been cleared or enable the interrupt enable bit SIIER[MSIE] associated with message send. When the message status register is read, fields VSIMSGSR[MS/MC] will indicate the result of the message request.

53.4.2.3.4 ENETC Statistics Counters

53.4.2.3.4.1 MAC Statistics Counters

For the MAC statistic counters, see section Ethernet MAC Statistics Counters for the Ethernet MAC statistics counters and section Pseudo MAC Statistics Counters for the pseudo MAC statistics counters.

53.4.2.3.4.2 Port Statistics Counters

The table below lists the available port level traffic statistics counters.

Table 517. Port Traffic Statistics Counters

Direction	Counter	Description
Rx	Ingress Stream Count table - RX_COUNT	Received frames counter on a per stream basis
Rx	Rate Policer table - statistics counters	Bytes count on a per Rate Policer instance. Also available are marked/remarked frames counts
Rx	Ingress Port Filter table - MATCH_COUNT	Ingress Port Filter entry matched count

The table below lists the available port level frame drops statistics counters.

Table 518. Port Frame Drop Counters

Direction	Counter	Description
Rx	PRXDCR	Frame drops count in the receive port datapath processing pipeline with the setting of the discard reason in the port's Rx discard count reason register 0/1 (PRXDCRR0/1)
Rx	PUFDMFR	Incremented for each dropped unicast frame by the L2 filtering function, due to MAC filtering yielding no SIs to receive the frame, unless the reason for yielding no SIs is MAC source address pruning, in which case the PFDMSAPR counter is incremented instead.
Rx	PMFDMFR	Incremented for each dropped multicast frame by the L2 filtering function, due to MAC filtering yielding no SIs to receive the frame, unless the reason for yielding no SIs is MAC source address pruning, in which case the PFDMSAPR counter is incremented instead.
Rx	PBFDSIR	Incremented for each dropped broadcast frame by the L2 filtering function, due to MAC filtering yielding no SIs to receive the frame due to all SI's having broadcast reject enabled.
Rx	PUFDVFR	Incremented for each dropped unicast frame by the L2 filtering function, due to VLAN filtering yielding no SIs to receive the frame.
Rx	PMFDVFR	Incremented for each dropped multicast frame by the L2 filtering function, due to VLAN filtering yielding no SIs to receive the frame.

Table continues on the next page...

Table 518. Port Frame Drop Counters (continued)

Direction	Counter	Description
Rx	PBFDVFR	Incremented for each dropped broadcast frame by the L2 filtering function, due to VLAN filtering yielding no SIs to receive the frame.
Rx	PFDMSAPR	Incremented for each dropped frame by the L2 filtering function, due to MAC filtering yielding no SIs to receive the frame due to MAC source address pruning.
Rx	Ingress Stream Count table - frame drops counters	Maximum SDU frame drops count, stream gate frame drops count and Rate Policer instance frame drops count on a per stream basis
Rx	Rate Policer table - DROP_FRAMES	Frame drops count on a per Rate Policer instance basis
Rx	PICDR*a*DCR	Number of frames discarded by Ingress Congestion Manager; a separate register is provided for each DR value. PICPDSR provides status flag for each DR and priority combination indicating if one or more frames have been discarded at this DR and priority

53.4.2.3.4.3 Station Interface (SI) Statistics Counters

The table below lists the Station Interface (SI) level traffic statistics counters.

Table 519. Station Interface (SI) Traffic Statistics Counters

Direction	Counter	Description
Rx	SIROCT0/1	Number of octets received (without frame errors) by the SI except preamble, SFD and CRC (if CRC has been removed from the frame).
Rx	SIRFRM0/1	Number of frames received by the SI without errors
Rx	SIRUCA0/1	Number of unicast frames received by the SI without errors
Rx	SIRMCA0/1	Number of multicast frames received by the SI without errors
Tx	SITOCT0/1	Number of octets transmitted (without error) by the SI except preamble, SFD and CRC (if CRC has been appended to the frame by the hardware)
Tx	SITFRM0/1	Number of frames transmitted by the SI without errors, with the exception of frames with error (BD status code point) 0x020 (frame dropped due to port reset) or 0x0A0 (multi-bit ECC error), which will be counted in this counter.
Tx	SITUCA0/1	Number of unicast frames transmitted by the SI without errors, with the exception of frames with error (BD status code point) 0x020 (frame dropped due to port reset) or 0x0A0 (multi-bit ECC error), which will be counted in this counter.
Tx	SITMCA0/1	Number of multicast frames transmitted by the SI without errors, with the exception of frames with error (BD status code point) 0x020 (frame dropped due to port reset) or 0x0A0 (multi-bit ECC error), which will be counted in this counter.

Table below lists the Station Interface (SI) level frame drops counters

Table 520. Station Interface (SI) Frame Drop Counters

Direction	Counter	Description
Rx	SIUPECTR	Frame drops counter due to non-existing BD ring or non-existing group, or due to SI disabled or BD ring disabled
Rx	RBaDCR	Receive BD ring frame drops counter due to lack of Rx BDs available
Rx/Tx	SIUNSBECTR	Frame drops counter due to system bus error

53.4.2.4 Switch

53.4.2.4.1 Switch Datapath

The switch datapath is implemented as a series of functional processing blocks pipelined together as shown in the figure below. This includes the Ethernet Rx I/F, the Ingress Packet Processing, the Replication and Egress Packet Processing, the Egress Queuing and Traffic Management and the Ethernet Tx I/F. A shared internal buffer memory used for internal frame buffering.

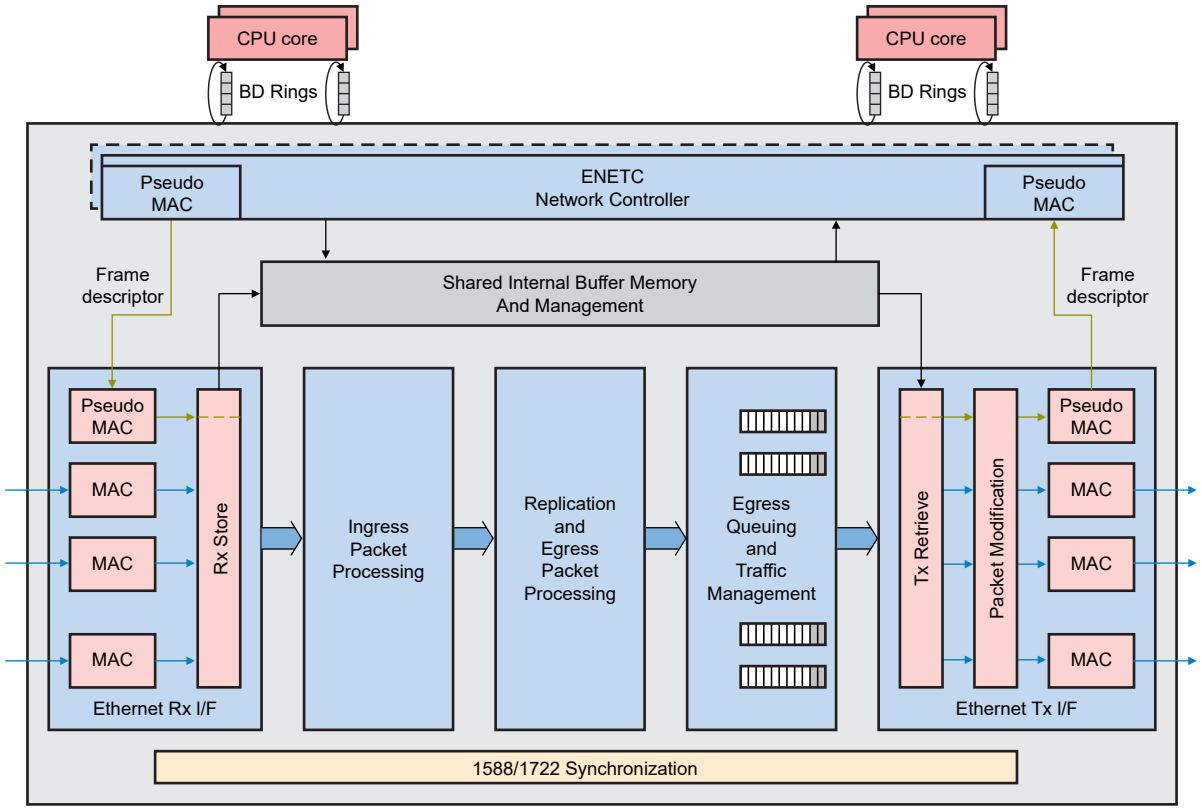


Figure 342. Switch datapath functional block diagram

The shared internal buffer memory is used to store and buffer data arriving from the Ethernet MACs; in the case of a pseudo MAC, frames have already been stored in the shared internal buffer memory by the ENETC port connected to the pseudo MAC. Frames remain stored in this memory as they move through the switch processing pipeline up to the point where frames have been transmitted to the outgoing port(s). In the case where the frame is being transmitted on a pseudo MAC, the frame remains stored in the shared internal buffer memory as only a frame descriptor is being transferred across that interface.

The Ingress Packet Processing block responsibilities are to parse the frame headers, to do the necessary table lookups (or classification) to determine the data that will drive the execution of the ingress packet processing functions such as filtering,

forwarding, policing and frame modifications. The actual execution of these ingress packet processing functions is also the responsibility of the Ingress Packet Processing, with the exception of the frame modifications. The frame modification actions are passed as frame metadata for actual modification into the frame when the frame is being transmitted on a MAC.

The Replication and Egress Packet Processing is responsible for implementing the frame replication (frame descriptor only) if needed, and packet processing functions in the context of the output port(s).

The Egress Queuing and Traffic Management block includes the support of 2 classes of traffic management mechanisms: queue scheduling and queue management. Queue scheduling refers to the procedures used to determine which frame to send next when bandwidth on the outgoing port is available. It is used primarily to manage the allocation of bandwidth among traffic classes. Queue management refers to the procedures that manage the queue sizes by primarily deciding when to start dropping frames.

In the case where the frame is being transmitted on a pseudo link (switch port connected internally to an ENETC port), there is no memory copy when transferring a frame across a pseudo link. Instead the frame descriptor along with the associated metadata, is passed to the other end of the link (i.e. in this case an ENETC port) when sending a frame across a pseudo link.

Both store-and-forward and cut-through modes of operation are supported. For both mode of operations, once the first 24 bytes of data is received and stored in internal memory, the Ingress Packet Processing (IPP) starts parsing the frame header (first 24 bytes). If the end of the parse graph has not been reached while parsing the first 24 bytes of the frame, the IPP waits for the next 24 bytes to be received so that it can resume parsing, and so on up to the maximum number of bytes that can be parsed which is 144 bytes.

In store-in-forward mode of operation, once parsing is completed and although the entire frame may not have been entirely received, the IPP carries on executing functions that do not require knowledge of the frame length (e.g. table lookups, ingress frame modifications). Once the entire frame is received, it will execute the processing functions that rely on the knowledge of the frame length, and then it will pass the frame to the next processing stage.

In cut-through mode of operation, once parsing is completed and although the entire frame may not have been entirely received, frame moves through the switch processing pipeline and starts getting transmitted on all the destination ports.

The cut-through mode of operation is not applicable (or not supported) for frames received from a pseudo MAC as a complete frame (or frame descriptor) is received one at a time (frame is already stored in the shared internal buffer memory).

53.4.2.4.1.1 Switch Shared Internal Buffer Memory Management

The shared internal buffer memory used for the switch internal frame buffering, is allocated from the common internal shared memory region. An allotment is configured at initialization time in the common internal shared memory region to track overall frame buffering usage of the switch.

Switch shared internal buffer memory is managed as set of memory buffer pools.

A buffer pool is a quantity of memory available for buffering a group of flows (e.g. frames having the same priority, frames received from the same port), while waiting to be transmitted on a port. A buffer pool tracks internal memory consumption with upper bound limits and optionally a non-shared portion when associated with a shared buffer pool. See figure below for an illustration of buffer pools and shared buffer pools.

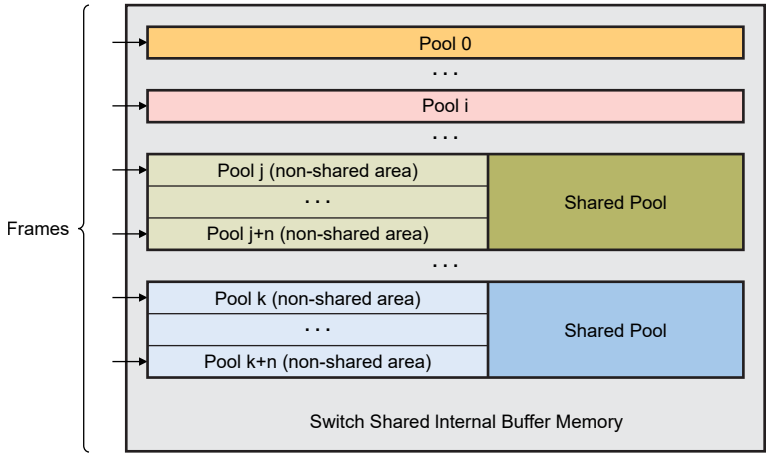


Figure 343. Buffer Pools and Shared Buffer Pools

A shared buffer pool has a set of pools as members and tracks only the amount that have exceeded the buffer pools' non-shared portion. If a buffer pool is below its maximum non-shared consumption, then there is no contribution to its associated shared pool.

A set of drop thresholds and flow control thresholds are defined to control internal memory consumption within a buffer pool and its associated shared buffer pool.

The sum of the maximum sizes of each buffer pool with no shared buffer pool association plus the sum of the non-shared allocations of each buffer pool associated with a shared buffer pool, and plus the sum of the maximum sizes of each shared buffer pool cannot exceed the total size of the switch shared internal buffer memory.

The maximum size of a buffer pool with no shared buffer pool association corresponds to the size configured in the Maximum Threshold field of its Buffer Pool table entry plus one maximum frame size. The non-shared allocation of a buffer pool associated with a shared buffer pool corresponds to the amount configured in the Shared Buffer Pool Threshold field of its Buffer Pool table entry. The maximum size of a shared buffer pool corresponds to the size configured in the Maximum Threshold field of its Shared Buffer Pool table entry plus one maximum frame size.

As frames are received from the MACs, their shared internal buffer memory usage is initially tracked as "unassigned". A received frame remains accounted as "unassigned" as it moves through the processing pipeline of the Ingress Packet Processing. The buffer pool assignment function is performed towards the end of the ingress packet processing pipeline, where the final QoS attributes (IPV and DR) of the frame have been determined. The buffer pool is selected by mapping the IPV of the frame to a buffer pool ID using the port buffer pool mapping configuration register 0/1 (PBPMCR0/1). This mapping allows each IPV (or set of IPV's) to be mapped to a specific buffer pool for QoS distinction. Once the Ingress Packet Processing buffer pool assignment logic has selected a buffer pool, the buffer pool (and its associated shared buffer pool) is checked to determine if there is available memory space to accommodate the new frame. The overall switch shared internal buffer memory usage is also checked to ensure that the switch shared internal buffer memory is not full. The memory availability checks for the buffer pool and overall switch shared internal buffer memory are performed without considering the length of the frame; the reason why a buffer pool or the switch buffering may exceeds its configured maximum size by one maximum frame size. The frame will be discarded if either the usage count of the selected buffer pool is already equal or above its configuration limit or the usage count of the overall switch shared internal buffer memory is already equal or above its configuration limit. If there is available memory space, then the buffer pool usage and the overall switch shared internal buffer memory usage is updated as follows:

- **Store-and-forward frame** - The buffer pool usage is updated by adding to the buffer pool usage, the length of the frame and the memory space required the store the frame descriptor (24 bytes) for each egress enqueue that will be attempted.
- **Cut-through frame** - The buffer pool usage is updated by adding to the buffer pool usage, the length of the first frame segment and the memory space required the store the frame descriptor for each egress enqueue that will be attempted. As subsequent segments of the frame are received the buffer pool usage is updated accordingly. No memory availability checks are performed for the subsequent segments of a cut-through frame.

Note that the shared internal buffer memory allocation for the frame descriptors is performed later, at the point where the egress enqueues are performed. If there is no memory available to store a frame descriptor, the hardware waits until memory becomes available; no egress frame drops are performed for this condition.

After frame transmission or when frame is dropped, buffer pool accounting and overall switch shared internal buffer memory accounting are updated accordingly; i.e. the accounting update occurs when the memory units used to store the frame are returned to the shared common memory free list.

The figure below shows an example of a buffer pool assignment configuration where the switch shared internal buffer memory is first partitioned by IPV, and then within a given a IPV partition, each source port is provided a guaranteed/dedicated amount of memory and a portion that is shared between all source ports.

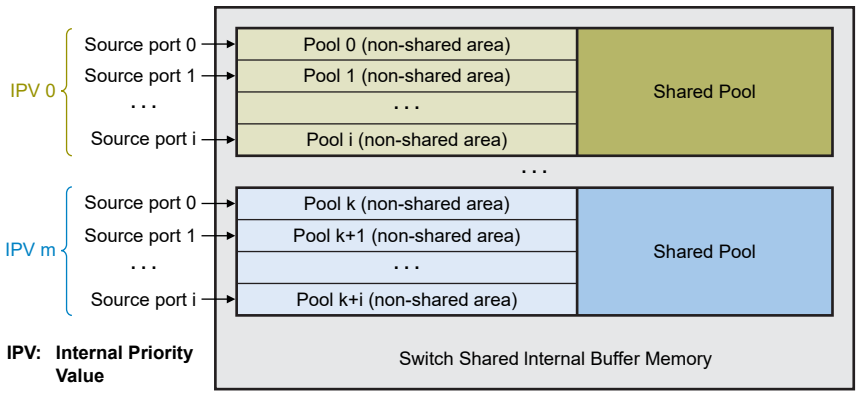


Figure 344. Buffer pools assignment example; by IPVs, followed by Source Ports

The figure below shows an example of a buffer pool assignment configuration where each source port or IPV pair is provided a guaranteed/dedicated amount of memory and the remaining switch shared internal buffer memory is shared.

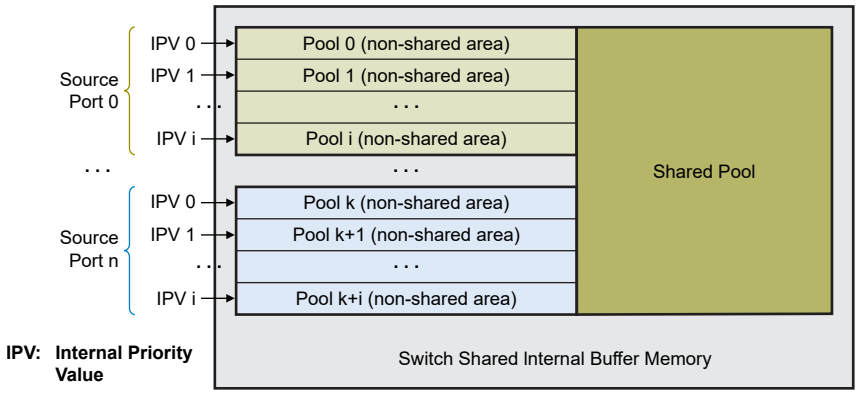


Figure 345. Buffer pools assignment example; non-shared per (source port, IPV) pair + shared portion

53.4.2.4.1.1.1 Buffer Pool Configuration

Buffer pool configuration parameters are as follows (see Buffer Pool Table):

- Buffer pool maximum threshold (12 bits) - Any assignment where the buffer pool usage count is already equal or above this threshold limit, will be denied.
- Shared buffer pool enable (1 bit) - If set to 1, this buffer pool is associated with a shared buffer pool.
- Shared buffer pool id (4 bits) - If shared buffer pool enable is set to 1, then this indicates the shared buffer pool ID to which this buffer pool belongs.

- Shared buffer pool threshold - If shared buffer pool enable is set to 1 and if buffer pool usage exceeds this threshold then the excess contribution is tracked in the shared buffer pool.
- Flow control configuration (2 bits)
  - 00 = Flow control disabled.
  - 01 = Flow control enabled using only buffer pool flow control state.
  - 10 = Flow control enabled using only shared buffer pool flow control state.
  - 11 = Flow control enabled using both buffer pool and shared buffer pool flow control state. The combined flow control states of both the buffer pool and the shared buffer pool must be 1 to assert flow control on. The flow control state of the buffer pool or the shared buffer pool must be 0 to assert flow control off.
- Flow control on threshold (12 bits) - Valid if flow control configuration = 01 or 11, if buffer pool usage crosses above this threshold the flow control state of the buffer pool is set to 1.
- Flow control off threshold (12 bits) - Valid if flow control configuration = 01 or 11, if buffer pool usage drops to this threshold or below, the flow control state of the buffer pool is set to 0.
- Priority vector (8 bits) - Valid if flow control is not equal to 0, when a flow control notification is sent (flow control on), this vector indicates which priorities are to be flow controlled for this buffer pool, one bit per priority level. If the MAC is configured in link pause mode, this vector is ignored. Note that NETC 3.0 and 3.1 doesn't support priority flow control, it only supports link pause mode.
- Flow control ports (1 bit per port) - Valid if flow control is not equal to 0, when a flow control notification is sent, this per-port bit mask indicates which ports are to be flow controlled for this buffer pool.
- Pool size (usage); also includes shared pool contribution.

53.4.2.4.1.1.2 Shared Buffer Pool Configuration

Shared buffer pool configuration parameters are as follows (see Shared Buffer Pool Table):

- Shared buffer pool maximum threshold (12 bits) - Any assignment where the shared buffer pool usage count is already equal or above this threshold limit, will be denied.
- Flow control on threshold (12 bits) - Valid if any associated buffer pool flow control configuration = 10 or 11. If shared buffer pool usage crosses above this threshold the flow control state of the shared buffer pool is set to 1.
- Flow control off threshold (12 bits) - Valid if any associated buffer pool flow control configuration = 10 or 11. If shared buffer pool usage drops to this threshold or below, the flow control state of the shared buffer pool is set to 0.

53.4.2.4.1.2 Ethernet Rx I/F

The Ethernet Rx I/F block integrates the Rx MAC (Ethernet and pseudo MACs) functionality and implements the interface functions specific from the MAC to the switch receive path.

53.4.2.4.1.2.1 Ethernet MAC

In the case where frames are received from an Ethernet MAC, the Ethernet Rx I/F stores them immediately in the shared internal buffer memory.

The Ethernet MAC detects the SFD or SMD, receives the frame, and checks for layer 2 data integrity (FCS errors) while forwarding data to MAC client. All frames >= 18B with valid SFD are forwarded to MAC client.

The last 4 bytes of the frame which represents the frame check sequence (FCS), are also stored in the shared internal buffer memory. The Ethernet MAC doesn't remove the FCS from the frame. The FCS carried in the received frame remains as the frame is processed by the switch. On transmission, if the data covered by the FCS is not modified, then the FCS value in the received frame is re-used as the FCS for the transmitted frame.

The first 144 bytes of the received frame in addition of being stored in the shared internal buffer memory, are also stored in a frame header memory which is accessed by the Ingress Packet Processing (IPP) for processing the received frame. Once the first 24 bytes of data is received and presented to the IPP, it starts parsing the frame header (first 24 bytes). If the end of the parse graph

has not been reached while parsing the first 24 bytes of the frame, the IPP waits for the next 24 bytes to be received so that it can resume parsing, and so on up to 144 bytes.

Both store-and-forward and cut-through modes of operation are supported.

In store-in-forward mode of operation, once parsing is completed and although the entire frame may not have been entirely received, the IPP proceeds with executing the required lookup-ups and resulting packet processing actions that aren't dependent on at the end of frame status (good/bad) and/or frame length. The IPP will then wait for the entire frame to be received before executing actions that are dependent on the end of frame status and/or frame length (e.g. policing, forwarding, byte counters updates).

In cut-through mode of operation, once parsing is completed and although the entire frame may not have been entirely received, frame moves through the switch processing pipeline and starts getting transmitted on all of the destination ports.

No drops are expected when delivering frames from the Ethernet Rx I/F to the IPP as the IPP is engineered to operate at wire-rate; no back-pressure is expected from the IPP. If an unforeseen overload condition of the IPP is detected, received frames from the MAC will be discarded. These frames discards are counted against the port's Rx discard count register (PRXDCR) along with the setting of the Pre-Classification Discard Reason (PCDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0).

The following metadata is passed along with the frame:

- Two timestamps (one sampling the synchronized clock whereby the other sampling the free running clock); for all receive frames, the timestamps are captured when MAC detects the one-octet start frame delimiter (SFD) of the frame. Currently, only the synchronized timestamp is used inside the switch. To report a frame receive timestamp from a switch port, the frame must be redirected or copied to the switch management port, with a Host Reason set to a codepoint other than a "regular forwarded frame" (for example, Ingress Port Filter (IPF) table entry match with the action to redirect or copy the frame to the switch management port, with Host Reason set to a non-zero value). PCR[TIMER_CS] determines which of the two timestamps (synchronized or free running) is to be reported.
- Indication that the frame was received from an Ethernet MAC and whether it was received from the Express MAC or the Preemptable MAC.
- Port number; acquired from the register LaBCR[SW_PORT_ENETC_INST], where a corresponds to the link number from which the frame was received.
- Frame length; calculated by the Ethernet RX I/F as it transfers the frame from the MAC Rx FIFO to internal memory.
- Indicator if MAC detected an error (for example, CRC error, invalid frame length). Errored frames are passed to the IPP where they will be discarded, or marked internally as bad, in the case where transmission of the frame has already started (cut-through mode of operation). On transmission, a bad CRC value is put in the frame FCS field to frames that have been internally marked as bad, to ensure that the next-hop device will correctly identify these frames as being bad.
- Indicator that the received frame was truncated due to the lack of free shared internal buffer memory available to store the entire frame. These frames are passed to the IPP where they will be discarded, or marked internally as bad, in the case where transmission of the frame has already started (cut-through mode of operation). These frames discards are counted against the port's Rx discard count register (PRXDCR) along with the setting of the Shared Memory Resource Exhaustion Discard Reason (SMREDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0).

For ports supporting IEEE 802.1Qbu frame preemption, there are two MACs used per port (express MAC and preemptable MAC). In this case, the Ethernet RX I/F implements the interface functions to both MACs.

53.4.2.4.1.2.2 Pseudo MAC

In the case where frames are received from a pseudo MAC, there is no need to store or copy these frames as they have already been stored in the shared internal buffer memory by the switch. The frame check sequence (FCS) is always present in the frame; the pseudo MAC doesn't remove the FCS from the frame. The FCS carried in the received frame remains as the frame is processed by the switch. On transmission, if the data covered by the FCS is not modified, then the FCS value in the received frame is re-used as the FCS for the transmitted frame.

When a frame is received, the first 144 bytes the frame is loaded in a frame header memory which is accessed by the Ingress Packet Processing for processing the received frame.

Cut-through mode is not applicable as frames are already stored in the shared internal buffer memory.

No drops are expected when delivering frames from the Ethernet Rx I/F to the Ingress Packet Processing (IPP) as the IPP is engineered to operate at wire-rate (assuming a minimum frame length of 64 bytes); no back-pressure is expected from the IPP. If an unforeseen overload condition of the IPP is detected, received frames from the pseudo MAC are no longer accepted; frames are not dropped, instead back-pressure is applied towards the ENETC port.

The following metadata is passed along with the frame:

- Two timestamps (one sampling the synchronized clock and the other sampling the free running clock) at the time the frame was transferred across the pseudo link. Currently, only the synchronized timestamp is used in the switch.
- Indication that the frame was received from a pseudo MAC.
- Port number; acquired from the register LaBCR[SW_PORT_ENETC_INST], where a corresponds to the link number from which the frame was received, unless the frame is received from a switch management port with an out of band indication that the frame should be processed as it was received from a specified switch port number that is different than the switch management port number. In this case, the specified switch port number is passed to the IPP, instead of the port number obtained via the aforementioned register.
- Frame length.

No errors are expected; FCS is not checked as there are no transmission of bits on pseudo link interface as only a frame descriptor is being transferred across that interface.

53.4.2.4.1.3 Ingress Packet Processing

The processing through the Ingress Packet Processing is shown in the figure below.

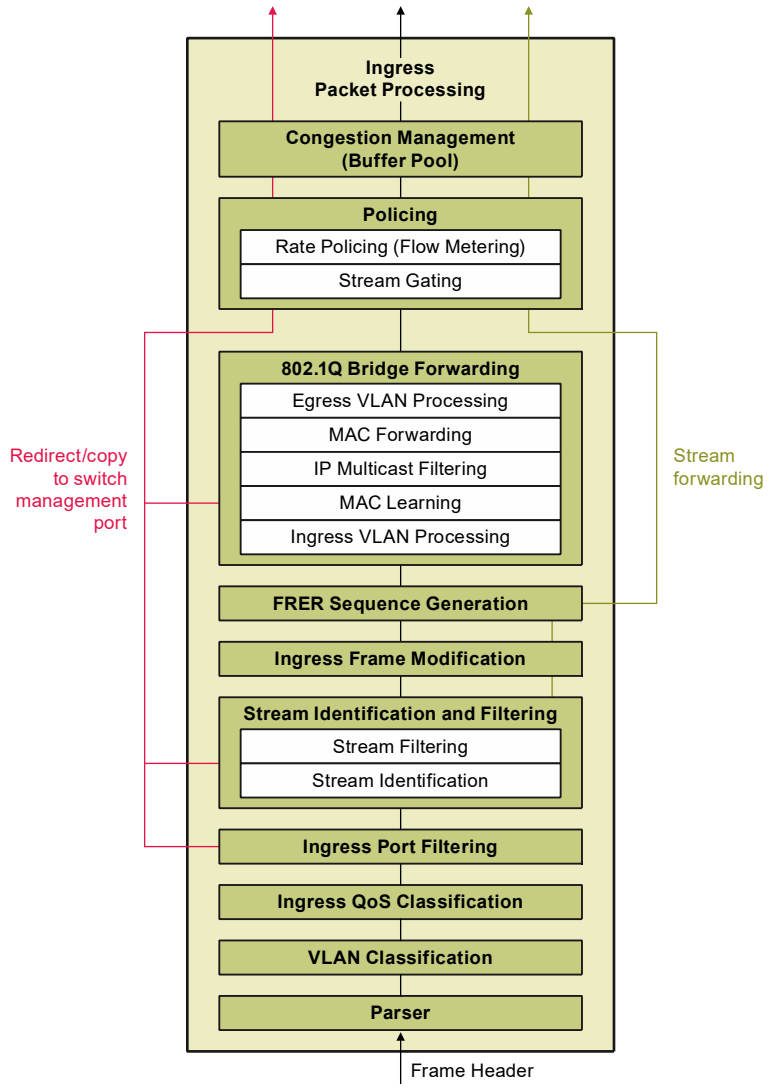


Figure 346. Ingress Packet Processing diagram

The Ingress Packet Processing block performs the following ingress packet processing functions in a pipeline manner:

- **Parser** - Parses the frames and extracts the required packet header fields in preparation for classification. Supports major protocols such as Ethernet, IPv4, IPv6, TCP and UDP.
- **VLAN classification** - Determines what VLAN tag data from the frame will be passed with the frame for later ingress processing.
- **Ingress QoS classification** - Provides initial internal identification of the QoS attributes that are used for subsequent QoS processing as frames go through the device. Internal QoS attributes indicate the given QoS (queueing priority and drop priority) as the frame moves through the device. At any queuing point, frames can be placed in the appropriate queue based on the internal queuing priority and can be dropped based on the internal drop priority.
- **Ingress port filtering** - Can deny or allow a frame. It may also override internal QoS attributes associated with the frame and set the stream ID or rate policer instance ID. It can also redirect/copy a frame to the switch management port and have the frame mirrored.
- **Stream identification and filtering** - Implements stream identification and filtering; compliant with IEEE 802.1CB and IEEE 802.1Qci (Per-Stream Filtering and Policing). It also provides the ability to specify, on per stream basis, the port(s) to which a frame should be output along with the egress processing actions that are to be applied, and with the ability

to enable/disable source port pruning. This capability is referred to as "stream forwarding" which by-passes the 802.1Q bridge forwarding function.

- **Ingress frame modification** - Provides ability to remove, add and modify a VLAN tag, along with replacing MAC addresses.
- **Frame Replication and Elimination for Reliability (FRER) sequence generation**- Sequence generation consists of tagging a frame with a sequence number, following which the frame can be replicated and transmitted along multiple paths for redundancy. Implemented as per the IEEE 802.1CB.
- **802.1Q bridge forwarding** - Performs the required forwarding lookups to determine the outgoing port(s), including flooding and broadcast/multicast handling. Supported bridge forwarding functions are
  - Ingress VLAN processing - Performs functions such as VLAN tagging checking, VLAN membership checking and Spanning Tree Protocol (STP) input port state checking. It also indicates the Filtering ID to be used by the forwarding function.
  - MAC learning - Entries are added to the FDB table when new unique MAC addresses are learned on the datapath. Entries found with port ID not matching incoming port ID are updated with incoming port ID. There is also the option to redirect/copy unknown frames to the switch management port for MAC learning purposes (software MAC learning).
  - IPv4 multicast filtering - Multicast frames are forwarded based on source IPv4 addresses and destination Iv4P addresses using IPv4 multicast filtering tables.
  - MAC forwarding - Frames are forwarded based on Layer 2 destination MAC addresses using the provided MAC Filtering or Forwarding Database (FDB) table.
  - Egress VLAN processing - Performs functions such as VLAN membership checking, Spanning Tree Protocol (STP), and output port state checking.
  - Storm control - Limits the amount of broadcast, multicast, and flooded unicast/multicast traffic.
- **Policing** - Implements traffic policing as per IEEE 802.1Qci (Per-Stream Filtering and Policing) to protect against time window and bandwidth violations.
- **Congestion Management** - Buffer pool assignment and checking that there is available memory space to accommodate the new frame.

53.4.2.4.1.3.1 Parser

The Parser parses up to L4 protocol headers in up to the first 144 bytes of the frame. Extracted field values and locations of relevant L2, L3 and L4 headers values are then used to perform the various table lookups and hash functions in subsequent processing stages of the switch.

During parsing, the parser examines and validates headers.

The parsing process can be expressed as a directed graph with nodes representing header types and edges (arrows) representing the sequence of headers. Parsing starts at the root node then walks from node to node using the next-header field values. Specifically, each node works within a protocol layer presuming a certain type of header, confirms the validity of the header, identifies boundaries of the header, extracts attributes and offsets, and determines the next header to examine beyond the current header.

Figure below shows the parse graph supported in the switch. The parse graph is structured by layers (L2, L3, and L4), with the details per each layer provided in the next sections.

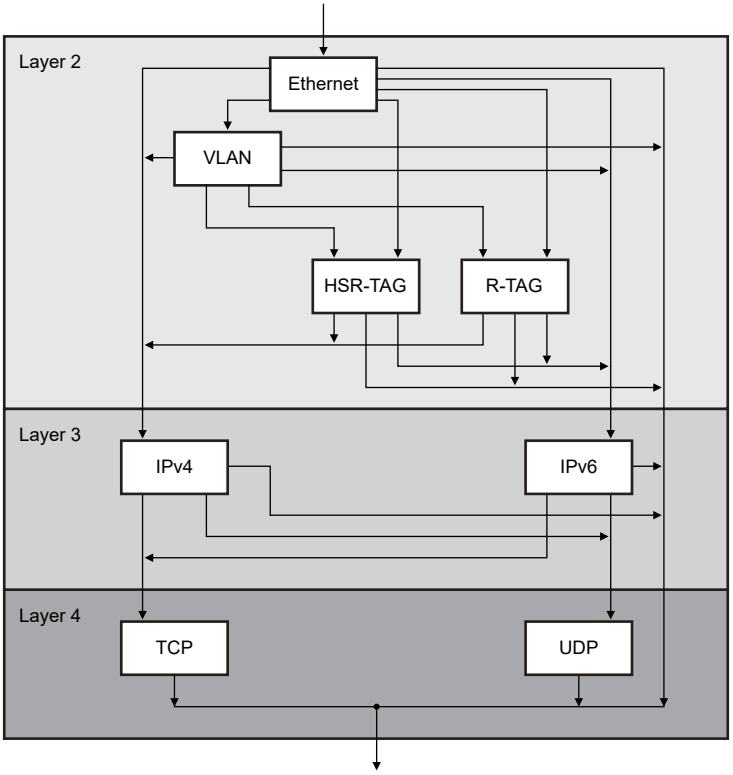


Figure 347. Switch Parse Graph

By default, the parser will parse and extract fields from each layer. There is the option to configure the parser to end parsing at L2 or L3 regardless of the content of the frame (for more details, see the port parser configuration register (PPCR)). The parser will also extract the payload (up to 24 bytes by default) of the last layer parsed (L2, L3 or L4). The maximum amount of payload to extract is configurable for each layer via the port parser configuration register (PPCR).

Parsing is terminated via one of the following:

- Upon reaching the natural end of a branch in the parse graph.
- Upon reaching a layer (L3 or L4) for which the parser has been configured not to parse.
- A parsing error is encountered; note that parse information from headers that have been successfully parsed is still used by table lookups and hash functions in subsequent processing stages of the switch.

53.4.2.4.1.3.1.1 Layer 2 Headers

As the first header in a frame, the parser supports Ethernet frames of the IEEE 802.3 (Ethernet II) format. IEEE 802.3 (Ethernet II) frames may be tagged with the IEEE 802.1Q VLAN tagging method. The 4-byte IEEE 802.1Q VLAN tag has two 2-byte components: Tag Protocol Identifier (TPID, residing within the type/length field) and the Tag Control Information (TCI). The following 4 TPID values are recognized as VLAN tags:

- The standard types 0x8100 and 0x88A8
- Two optional user programmable EtherType values known as CustTag1 and CustTag2; configured through CVLANR1 and CVLANR2 registers.

The TCI includes the following fields:

- PCP - Priority Code Point, the first three bits of the TCI define user priority.
- DEI - Drop Eligibility/ (single-bit field)
- VID - The 12-bit VLAN ID (VID) field identifies the VLAN the frame belongs to. The 12 bits allow for 4096 different VLANs. Out of the 4096 VLANs, the VID of zero is used to identify priority tagged frames and the value of 4095 is reserved.

Parser can parse frames with up to 2 VLAN tags. VLAN tags are named based on their position in the frame relative to other VLAN tags and the beginning of the frame i.e. the 802.3/Ethernet header consisting of Destination followed by Source MAC addresses.

The VLAN tag or header nearest the 802.3/Ethernet header is called the outer (or first) tag. The VLAN tag which is next closest to the 802.3 header is the inner (or second) tag. VLAN tags are expected to be contiguous, following one after the other, with no intervening non-VLAN tags.

Note that any VLAN tags which are within encapsulated frames and after any Upper Layer Protocol (ULP) header are not recognized. If only one VLAN tag is present in the frame, the inner VLAN tag is considered not present.

The outer and inner VLAN tag data (PCP, DEI and VID), if present, are extracted from the frame. The TPID from each VLAN tag is matched against the 4 known VLAN EtherTypes (2 standard, up to 2 custom) and a 2-bit TPID value is assigned identifying which known value matched (00: 0x8100, 01: 0x88A8, 10: CustTag1, 11: CustTag2). Matching is done against the list {0x8100, 0x88A8, CustTag1, CustTag2} in order and matching stops on the first match found. A 4-bit vector where each bit represents a TPID code point (bit 0 -> 00, bit 1 -> 01, bit 2 -> 10 and Bit 3 -> 11) is used by subsequent processing stages within the switch to perform non-contiguous masked matching operation against the TPID.

The information found in the next EtherType field (after VLAN parsing) is used to determine the next ULP header to parse. The EtherType field is located following the Source MAC address field if no VLAN tag present, or following the last VLAN tag. The ULP header is determined according to the “Ethernet - EtherType to Next ULP” mapping described in the table below.

EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends. If parsing reached end of frame, while more octets are expected to extract the 2-byte EtherType, the extracted EtherType is set to 0.

Table 521. Ethernet - EtherType to Next ULP Mapping

EtherType	Recognized Protocol	Next ULP to be parsed
x0800	IPv4	IPv4
x86DD	IPv6	IPv6
x88F7	PTP	Ends parsing
x892F	HSR Tag	HSR Tag
xF1C1	IEEE 802.1CB R-TAG	Redundancy tag
Pre-standard R-TAG EtherType register	IEEE 802.1CB draft 2.0 R-TAG	Redundancy tag

For High Availability Seamless Redundancy (HSR) tag, the parser extracts the 16-bit sequence number (last 2 bytes of the 6-byte HSR tag) and then locates the EtherType field following the 6-byte HSR tag. The next ULP header is determined according to the "HSR tag - EtherType to Next ULP" mapping described in the table below. If the EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 522. HSR tag - EtherType to Next ULP Mapping

EtherType	Recognized Protocol	Next ULP to be parsed
x0800	IPv4	IPv4
x86DD	IPv6	IPv6

For the 802.1CB R-TAG, the parser extracts the 16-bit sequence number (last 2 bytes of the 6-byte 802.1CB R-TAG) and then locates the EtherType field following the 6-byte R-TAG. The next ULP header is determined according to the “Redundancy tag - EtherType to Next ULP” mapping described in the table below. If the EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

The 4-byte pre-standard (802.1CB draft 2.0) R-TAG is also supported, The format of this tag is 2 bytes for the EtherType followed by 2 bytes for the sequence number. The recognized value for the Ethertype value of the pre-standard R-TAG is configured through the pre-standard R-TAG Ethertype register (PSRTAGETR). For the 4-byte pre-standard R-TAG , the parser extracts the 16-bit sequence number (last 2 bytes of the tag) and then locates the EtherType field following the 4-byte pre-standard R-TAG.

The next ULP header is determined according to the “Redundancy tag - EtherType to Next ULP” mapping described in the table below. If the EtherType field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 523. Redundancy tag - EtherType to Next ULP Mapping

EtherType	Recognized Protocol	Next ULP to be parsed
x0800	IPv4	IPv4
x86dd	IPv6	IPv6

Following information is captured by the parser for usage by subsequent processing stages within the switch:

- Source and Destination MAC addresses are extracted.
- Indication of whether the outer VLAN tag header is present and valid (no parsing error). If the header is present and valid, extracted VLAN tag data along with a 2-bit code identifying the TPID is also provided.
- Indication of whether the inner VLAN tag header is present and valid (no parsing error). If the header is present and valid, extracted VLAN tag data along with a 2-bit code identifying the TPID is also provided. If only one VLAN tag is present in the frame, the inner VLAN tag is considered not present.
- Whether destination MAC address is unicast, multicast or broadcast.
- A 3-bit redundancy tag type codepoint
  000b = No redundancy tag present
  001b = 802.1CB draft 2.0 R-TAG.
  010b = 802.1CB R-TAG.
  011b = HSR Tag
  100b ... 111b = Reserved.
- The sequence number is extracted if a redundancy tag is present (no parsing error).
- The last parsed EtherType is extracted; i.e. 2 bytes immediately after source MAC address or VLAN tag(s)/R-TAG/HSR (if any). If the EtherType wasn't parsed successfully (e.g. missing one byte due to frame too short), the EtherType is set to all 0s.
- Extracting up to 24 bytes located immediately after the payload EtherType. The payload EtherType is located immediately after the source MAC address or VLAN tag(s)/R-TAG/HSR (if any) if parsing is ending at this layer. The maximum amount of payload to extract is configurable via the port parser configuration register (PPCR).

Possible parsing errors for L2 headers are:

- Frame parsing reached end of frame while parsing a VLAN tag or R-TAG/HSR that expects more data. Frames shorter than 14 bytes excluding 4-byte FCS will never reach the parser as they will be discarded by the Ethernet Rx I/F.

There are no validation checks performed on the Source and Destination MAC addresses, and EtherType fields. Note that if an L2 header parsing error is detected, the frame is not discarded. Instead, only successfully parsed L2 header fields are extracted, but payload data is not extracted. The parse information is then passed along with the frame for subsequent processing within the IPP (e.g. table lookups and hash functions).

Note that any sequence tag information extracted by the parser for usage by subsequent processing stages, is overwritten if FRER sequence generation is enabled for the frame. NETC only supports ONE CB redundancy tag, one added as part of sequence generation if enabled, or a parsed redundancy tag if present.

53.4.2.4.1.3.1.2 Layer 3 Headers

The parser supports parsing of both IPv4 and IPv6 packets. Only the first IPv4 or IPv6 header encountered in a packet is parsed.

IPv4 options are not parsed. They are also not skipped meaning that the next ULP header will not be parsed if the packet has IPv4 options.

There is the option to configure the parser on per port basis, of not parsing the L3 header. This option is configured by setting PPCR[L3HFP] to 0. This option can be used to reduce the number of bytes required to be received before the frame can move through the switch processing pipeline and, in the case of cut-through, starts getting transmitted. This in turn can improve cut-through forwarding latency across the switch. This option should not be used if there are any table lookup entries that contain L3/L4 key fields that could be matched for that frame.

If a non-fragmented IPv4 packet (Fragment Offset is set to zero and the More Fragments flag is set to zero) or an initial fragment packet ((Fragment Offset is set to zero and the More Fragments flag is set to one) is parsed, then the parser uses the information stored in the Protocol field to determine the next ULP header to be parsed.

For IPv4 header, the next ULP header is determined according to the “IPv4 - Protocol to Next ULP” mapping described in the table below. If the Protocol field is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 524. IPv4 - Protocol to Next ULP Mapping

Protocol (decimal)	Recognized Protocol	Next ULP to be parsed
6	TCP	TCP
17	UDP	UDP

For IPv6 header, the IPv6 main header may be followed by an arbitrary number (0, 1 or multiple) of extension headers chained together through the Next Header field. IPv6 extension headers are not parsed. They are also not skipped meaning that the next ULP header will not be parsed if the packet has extension headers.

The parser uses the information stored in the Next Header field of the IPv6 main header to determine the next ULP header. The next ULP header is determined according to the “IPv6 – Next Header to Next ULP” mapping described in the table below. If the Next Header is set to a value other than the ones included in the table below, the next header is considered an unknown type and parsing ends.

Table 525. IPv6 - Next Header to Next ULP Mapping

Next Header (decimal)	Recognized Protocol	Next ULP to be parsed
6	TCP	TCP
17	UDP	UDP

Following information is captured by the parser for usage by subsequent processing stages.

- Indication of whether the outer IP header is present and valid (no parsing error). If the header is present and valid, indication of whether IP header is v6 or v4, is also provided.
- Indication of whether IPv4 options / IPv6 extensions are present. Note that value 59 (No Next Header) in the Next Header field of the IPv6 main header, is not considered an IPv6 extension header; as a result the IPv4 options / IPv6 extensions present indicator will be set to not present. See table below for the recognized IPv6 extension headers.
- IP source address , IP destination address, DSCP and Protocol fields are extracted.
- Extracting up to 24 bytes located immediately after the IP header if parsing is ending at this layer. The maximum amount of payload to extract is configurable via the port parser configuration register (PPCR). Note that if IPv4 options or IPv6 extensions are present, payload extraction starts at the beginning of the IPv4 Options/IPv6 Extensions header.

Possible parsing errors for L3 headers are:

- Frame parsing reached end of frame while parsing a header that expects more data.
- For each IPv4 header parsed, if any of the following conditions is detected:
  - Version number in IPv4 header is not equal to 4.
  - IPv4 Header Length field value is less than 20 bytes, or exceeds the received packet length (excludes L2 header), or exceeds the IPv4 header Total Packet Length field value

- For each IPv6 header parsed, if any of the following conditions is detected:
  - Version number in IPv6 header packet is not equal to 6.

Note that if an L3 header parsing error is detected, the frame is not discarded. Instead, the L3 header is considered not present, and parsing ends. The parse information from the L2 headers (including L2 payload if extracted) is passed along with the frame for subsequent processing within the IPP (e.g. table lookups).

Table 526. IPv6 extension headers

Protocol Number (decimal)	Description
0	IPv6 Hop-by-Hop Option
43	Routing Header for IPv6
44	Fragment Header for IPv6
50	Encapsulating Security Payload
51	Authentication Header
60	Destination Options for IPv6
135	Mobility Header
139	Host Identity Protocol
140	Shim6 Protocol
253	Use for experimentation and testing
254	Use for experimentation and testing

53.4.2.4.1.3.1.3 Layer 4 Headers

The parser supports parsing of both TCP and UDP packets. Only the first TCP or UDP header encountered in a packet is parsed.

The TCP header may have options but these aren't examined.

There is the option to configure the parser on per port basis, of not parsing the L4 header. This option is configured by setting PPCR[L4HFP] to 0. This option can be used to reduce the number of bytes required to be received before the frame can move through the switch processing pipeline and, in the case of cut-through, starts getting transmitted. This in turn can improve cut-through forwarding latency across the switch. This option should not be used if there are any table lookup entries that contain L4 key fields that could be matched for that frame.

Following information is captured by the parser for usage by subsequent processing stages within the switch:

- If the L4 header parsed is TCP, an indication that TCP is present (L4 codepoint set 01b).
- If the L4 header parsed is UDP, an indication that UDP is present (L4 codepoint set 10b).
- Source and destination port numbers are extracted.
- If no TCP nor UDP header is detected, an indication that other L4 is present (L4 codepoint set 00b).
- Extracting up to 24-bytes located immediately after the first 4 bytes of the TCP/UDP header. The maximum amount of payload to extract is configurable via the port parser configuration register (PPCR).

Possible parsing errors for L4 headers are:

- Frame parsing reached end of frame while extracting source and destination port numbers (L4 header shorter than 4 bytes).

Note that if an L4 header parsing error is detected, the frame is not discarded. Instead, the L4 header is considered other L4, and parsing ends. The parse information from the L3 headers (including L3 payload if extracted) is passed along with the frame for subsequent processing within the IPP (e.g. table lookups).

53.4.2.4.1.3.2 VLAN classification

VLAN classification processing stage determines what VLAN tag data will be passed with the frame for later ingress processing.

Following VLAN tag metadata provided by the parser is used by VLAN classification:

- Indication of whether the outer VLAN tag header is present and valid (no parsing error)
- Indication of whether the inner VLAN tag header is present and valid (no parsing error)
- Extracted outer VLAN tag data from frame
  - PCP, DEI, VID
  - TPID expressed as 2-bit code point
- Extracted inner VLAN tag data from frame
  - PCP, DEI, VID
  - TPID expressed as 2-bit code point

Output of the VLAN classification is:

- VLAN tag attribute indicators
  - Indication of whether the frame is untagged
  - Indication of whether outer VLAN tag is valid, if valid then the following attributes further qualifies it
    - VLAN tag data taken from port outer VLAN configuration; not asserted for the switch as VLAN assignment is performed later on.
    - VID is set 0 in the frame outer VLAN tag; i.e. priority tag
    - VLAN tag data taken from port inner VLAN configuration; not asserted for the switch as inner VLAN assignment is not supported.
  - Indication of whether inner VLAN tag is valid, if valid then the following attributes further qualifies it
    - VID is set to 0 in the frame inner VLAN tag; i.e. priority tag
- Outer VLAN tag data (PCP, DEI, VID) if outer VLAN tag is valid
- Inner VLAN tag data (PCP, DEI, VID) if inner VLAN tag is valid

The setting of the above VLAN classification output attribute indicators (along with appropriate VLAN tag data) is provided in the following steps:

1. All VLAN classification output indicators are cleared at the start of VLAN classification processing except for the outer/inner tag presence indicators which are taken from the parser. Outer/inner tag data and 2-bit TPID code point are also taken from the parser
2. If no tags are present then frame metadata is marked as untagged. No default VID is assigned at this point of the packet processing pipeline.
3. For each of the inner/outer tags which are (deemed) present the appropriate acceptance bitmap (PTAR[IVTPIDL/OVTPIDL]) is checked. If the tag's TPID's bit is set then the tag is acceptable otherwise it is deemed not present.
4. For each of the inner/outer tags if the tag is (deemed) present then frame metadata is marked as tag data being valid, and with the "VID is set to 0" qualifier set to 1 if the VID in the tag is 0. The "VID is set to 0" qualifier is a frame metadata attribute indicating whether this VLAN tag is a priority tag. When the "VID is set to 0" qualifier is set to 1, it indicates that this VLAN tag is a priority tag.

The resulting VLAN tag data and attribute indicators are passed along for upstream use.

53.4.2.4.1.3.3 QoS classification

QoS classification involves setting the internal QoS. The internal QoS indicates the given QoS (Internal Priority Value and Drop Resilience) as the frame moves through the device, and encounters resource contention and possibly be conditioned (e.g.

policing). Internally at any queuing point, frames can be placed in the appropriate queue based on the Internal Priority Value and can be dropped based on the Drop Resilience.

Internal Priority Value (IPV) is a 3-bit attribute assigned to a frame which is used for certain scheduling decisions in subsequent processing stages. A higher IPV value generally indicates higher priority , assuming strict priority scheduling is applied. The lowest priority is 0 and the highest is 7.

Drop Resilience (DR) is a 2-bit attribute assigned to a frame which is used to determine the likelihood of dropping the frame if there is congestion. A higher DR value generally indicates being more discardable. For example:

- Drop resilience 0

  Committed frames. With proper allocation of buffer and bandwidth resources within the network, it is possible to guarantee a lossless and timely delivery of a committed frames. In a "colored" mapping scheme such as IEEE 802.1Qci flow metering, this corresponds to Green.

- Drop resilience 1

  Normal frames. In a "colored" mapping scheme such as IEEE 802.1Q flow metering, this corresponds to Green.

- Drop resilience 2

  Non-committed frames. At sign of congestion, these frames can be dropped but should otherwise be delivered. In a "colored" mapping scheme such as IEEE 802.1Q flow metering, this corresponds to Yellow.

- Drop resilience 3

  Violating frames. These frames are delivered on a "best effort" basis. In a "colored" mapping scheme such as IEEE 802.1Q flow metering, this corresponds to Red.

Received frames are initially assigned an IPV and DR based on the setting for the port (PQOSMR[DIPV] and PQOSMR[DDR]).

The PCP and DEI from one of the sets of VLAN tag data may be used to override the default settings. This is achieved by setting PQOSMR[VE] to 1 to enable the use of VLAN tag data to determine IPV and DR and configuring PQOSMR[VS] to determine which VLAN tag (inner/outer) to be used.

If the selected VLAN tag is valid (VLAN classification output), the PCP and DEI from the selected VLAN tag data are used to determine IPV and DR based on the configured per port mapping VLANIPVMP*a*R0/1 and VLANDRMP*a*R, where *a* identifies a mapping profile. The profile to use (*a*) is specified in PQOSMR[VQMP].

IPV and DR values for the frame may also be changed by subsequent processing stages.

53.4.2.4.1.3.4 Ingress Port Filtering

This stage implements the ingress port filtering functions. There are two functions being performed; denial of service checks function and the ingress port filtering lookup function.

The denial of service (DOS) checks function is first executed to determine whether admit or drop incoming frames. The denial of service checks to be performed are specified in DoS L2 configuration register (DOSL2CR). Whether to perform the L2 DOS checks is specified in register PCR[L2DOSE].

If the frame hasn't been discarded by the previous function, the ingress port filtering lookup function is then executed by examining the frame header information to determine on whether to deny or allow a frame. It may also be used to override internal QoS attributes associated with the frame and set parameters (e.g. Rate Policer Entry ID, stream ID, SI bitmap) for the subsequent frame processing functions.

The ingress port filtering lookup function must be enabled to be executed (i.e. PIPFCR[EN] set to 1). If the ingress port filtering lookup function is disabled, the ingress port filtering lookup function is by-passed, and frames are passed to the next frame processing function.

The ingress port filtering lookup function involves performing a lookup against the ingress port filter table. The ingress port filter table is defined as a ternary match table. In a ternary match table, all entry key matching fields are composed of pairs of data and mask values, to support a per-bit ternary match capability. For each {data, mask} bit pair, the following combinations are supported:

- 00 = always match (X)
- 01 = match on 0
- 11 = match on 1
- 10 = always match (X); note that the table management query operation will always be returning 0 for any data bits which have their corresponding mask bit set to 0, regardless of whether they are set to 0 or 1 by the table management add or update operations.

In other words, to look for a match on a particular bit, set its mask bit to 1, and its corresponding data bit to the desired matching value, 0 or 1. A mask value of all 1's for a particular field means that field is required. To make a bit don't care, set its mask bit to 0.

In a ternary match table, a single lookup key can be matched by multiple table entries, and thus every entry must have a priority assigned by the software. When multiple entries match, the entry with the highest priority, is returned as the matching entry. Numerically higher values indicate higher priority when matching packets.

If no entry matches the frame, the frame is allowed in, and passed to the next frame processing function.

The ingress port filter table is implemented using a Ternary Content Addressable Memory (TCAM).

For programming of the table entries, see Ingress Port Filter Table.

The ingress port filter table entry includes the following matching key fields (each field is bit-wise maskable):

- Source port ID
- Switch port masquerading; frame received on a switch management port with an out of band indication that the frame should be processed as it was received from a switch port that is different from the switch management port.
- Source and destination MAC address
- Outer VLAN tag PCP, DEI and VID
- Inner VLAN tag PCP, DEI and VID
- Payload EtherType, located immediately after the source MAC address or VLAN tag(s)/R-TAG/HSR (if any).
- IPv4 source address and IPv4 destination address
- IPv6 source address and IPv6 destination address
- Protocol field present; for IPv6 packets the Protocol key (called the "Next Header" in IPv6) is not extracted if the packet has IPv6 extension headers.
- Protocol
- IP DSCP
- TCP/UDP header source and destination port
- Payload (up to 24-bytes)
  - When no IP header is present, the payload is matched against the frame bytes located immediately after the payload EtherType. The payload EtherType is located immediately after the source MAC address or VLAN tag(s)/R-TAG/HSR (if any).
  - When IP header is present and no TCP or UDP header is present, the payload is matched against the frame bytes located immediately after the IP header except when IPv4 options or IPv6 extensions are present. For the latter, payload matching starts at the beginning of the IPv4 Options/IPv6 Extensions header.
  - When TCP or UDP header is present, the payload is matched against the frame bytes located immediately after the first 4 bytes of the TCP/UDP header.
- Frame attributes bit fields
  - o Outer VLAN tag present
  - o Inner VLAN tag present
  - o IP header present

- Indication of whether IP header is v6 or v4
- TCP header present
- UDP header present
- Other L4 header present (not TCP or UDP)
- IPv4 option / IPv6 extension present
- Sequence Tag present

Note that when hardware constructs a table lookup key for a frame header field for which the frame header is not present in the frame, it sets the key field to zero. If zero corresponds to a valid value in a frame header field, this could lead to a false positive match. To avoid these false positive matches, the present frame header (or field) 1-bit key matching fields are defined in the table entry to explicitly specify whether the presence of a frame header (or field) must be present or not present in a frame header, for an entry to match.

Following are the possible actions when an ingress port filter entry is matched:

- Filtering action, one of the following:
  - Discard
  - Permit
  - Redirect/copy to switch management port
- Modifying internal QoS attributes (IPV and DR)
- Ingress Mirroring
- Capturing timestamp
- Disabling cut-through
- Setting internal parameter, one of the following
  - Enabling policing with a specified Rate Policer Entry ID
  - Specifying a Ingress Stream Entry ID along with lookup precedence

53.4.2.4.1.3.5 Stream Identification and Filtering

This stage implements the stream identification and filtering functions. Steam identification mainly involves classifying incoming traffic into a locally-significant identifier (global to the device), that identifies a stream within the device. This identifier is referred to as the Ingress Stream Entry ID (IS_EID). The stream identification function is implemented as per IEEE 802.1CB (Frame Replication and Elimination for Reliability (FREF)). Note that IEEE 802.1Qci (Per-Stream Filtering and Policing) exploits the stream identification functions specified in IEEE 802.1CB. The stream filtering function identifies the actions/functions to be applied to frames belonging to the identified stream. Namely, the main actions/functions are:

- Whether a frame is accepted or discarded.
- Maximum service data unit size defined for the stream as per IEEE 802.1Qci (Per-Stream Filtering and Policing).
- Stream gate instance and rate policer instance to be used to police the stream as per IEEE 802.1Qci (Per-Stream Filtering and Policing). The stream gating and rate policer functions are applied later in the ingress processing pipeline after it has been determined where to forward the frame.
- FREF (IEEE 802.1CB) sequence generation function. This function applies only to the switch.
- Stream forwarding function; the port(s) to which the frame should be output are provided in the stream context along with the egress packet processing actions to be applied to packets for each of the outgoing port(s); when the stream forwarding function is selected, the 802.1Q bridge forwarding function is by-passed. This function applies only to the switch.
- Ingress frame modifications such as ingress VLAN translation and IEEE 802.1CB active stream identification (replacing DMAC and/or VLAN). This function applies only to the switch.

- The Ingress Stream Counter ID that indicates the Ingress Stream Count table entry to be used to count matched frames and count discarded frames.

53.4.2.4.1.3.5.1 Stream Identification

Stream identification function mainly involves performing a series of table lookups to determine the Ingress Stream Entry ID (IS_EID), which in turn is used to access the Ingress Stream table to retrieve the action(s) to be applied to the frame. A default IS_EID, can be specified on a per port basis in PISIDCR register. Following are the lookup tables that can yield the IS_EID.

- **Ingress Port Filter table** - IS_EID can be specified in an Ingress Port Filter entry along with its relative precedent resolution value. Specifying IS_EID in ingress port filter entry is optional. If specified, its relative precedent resolution value determines if further lookups are required for stream identification.
- **Ingress Stream Identification table** - The Ingress Port Filter table lookup (prior stage) is followed up by a series of up to 2 hash based exact match table lookups into the Ingress Stream Identification table. The key used for stream identification exact match lookup is specified through a "key construction rule". A key construction rule specifies how the lookup key should be constructed. Specifically, it specifies metadata, specific frame header fields, and payload data that should be concatenated to form a key for the table lookup. A key construction rule is specified through a set of registers. For the switch, there are 4 key construction rules available for the entire switch, and are identified by the decimal number (0-3). For ENETC, there are 2 key construction rules per ENETC instance, and are identified by the decimal number (0-1). Note that the key construction register names imbeds the ID of the key construction rule that they are associated with. For example, ISIDKCaCR0, "a" represents the rule ID (decimal) that the register is associated with. Assignment of key construct rules to Ingress Stream Identification table lookups is achieved through key construction rule profiles. A key construction rule profile contains 2 construction key rules (one for the first exact match lookup and one for the second exact match lookup). Key construction rule profiles are derived through a static hard-wired assignment; profile 0 includes rules ID 0 and 1, profile 1 includes rules ID 2 and 3, and so on if it needs to be expanded in the future. For the switch, which of the two key construction rule profiles to use, is configurable on per switch port basis (register PISIDCR[KCPAIR]). For ENETC, key construction rule profile 0 is always used.

The following details the sequence of table lookups for determining the IS_EID (also illustrated in Figure 348).

1. If the Ingress Port Filtering function has yield an IS_EID and its relative precedent resolution value indicates that it has the highest precedence level (set to 0), then this IS_EID is selected, else go to step #2.
2. The first of the two hash based exact match lookups is performed (if enabled (PISIDCR[KC0EN]=1)) based on the key construction rule specified for this lookup. If a matching entry is found, the IS_EID specified in the entry is then selected. If this lookup is not enabled or a matching entry is not found and if the Ingress Port Filtering function has yield an IS_EID with a relative precedent resolution value falling between first and second exact match lookup (i.e. value set to 1), then this IS_EID is selected. Otherwise go to step #3.
3. The second of the two hash based exact match lookups is performed (if enabled (PISIDCR[KC1EN]=1)) based on the key construction rule specified for this lookup. If a matching entry is found then the IS_EID specified in the entry is selected. If this lookup is not enabled or a matching entry is not found and the Ingress Port Filtering function had yield an IS_EID then this IS_EID is selected. Else the default IS_EID is selected.

Note that if the selected IS_EID is NULL, the ingress stream frame processing will not be performed and the frame is passed on to the next stage of processing.

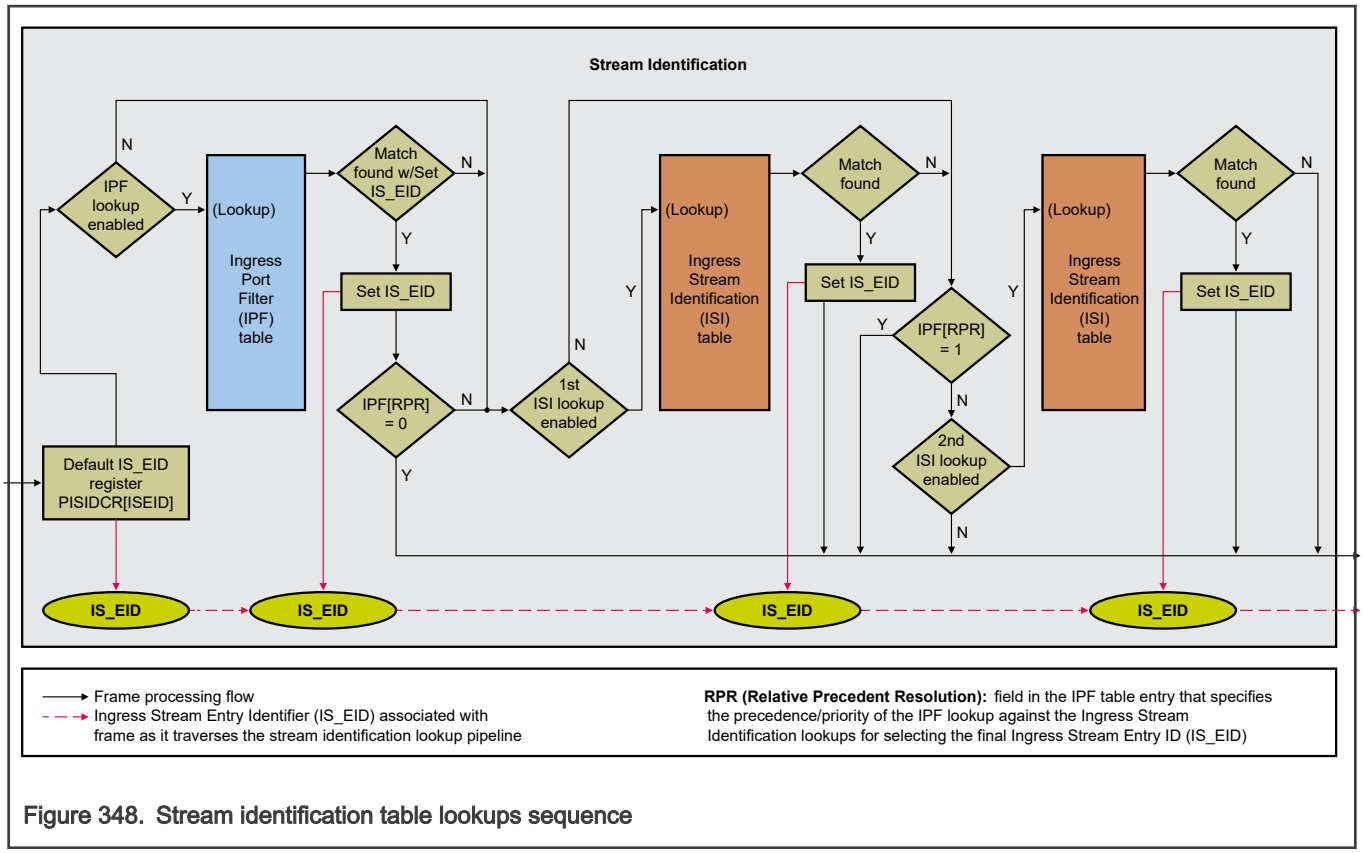


Figure 348. Stream identification table lookups sequence

Following registers are associated with lookups in the Ingress Stream Identification table.

- Ingress Stream Identification Capability Register (ISIDCAPR); indicates number of key construction rules, maximum key size, and so on.
- Ingress Stream Identification Hash Table Capability Register (ISIDHTCAPR); indicates which configuration access methods are supported.
- Switch registers for ingress stream identification key construction rules are noted below.
  - Ingress Stream Identification Key Construction a Operational register (ISIDKC*a*OR); indicates the number of entries in the Ingress Stream Identification table for particular key construction rule, where *a* =0..3 represents the key construction rule ID.
  - Ingress Stream Identification Key Construction Configuration Register 0 (ISIDKC*a*CR0) , where *a*=(0..3) represents the key construction rule ID.
  - Ingress Stream Identification Key Construction a Payload Field 0 Configuration Register (ISIDKC*a*PF0CR), where *a*=(0..3) represents the key construction rule ID.
  - Ingress Stream Identification Key Construction a Payload Field 1 Configuration Register (ISIDKC*a*PF1CR), where *a*=(0..3) represents the key construction rule ID.
  - Ingress Stream Identification Key Construction a Payload Field 2 Configuration Register (ISIDKC*a*PF2CR), where *a*=(0..3) represents the key construction rule ID.
  - Ingress Stream Identification Key Construction a Payload Field 3 Configuration Register (ISIDKC*a*PF3CR), where *a*=(0..3) represents the key construction rule ID.
  - Port Ingress Stream Identification Configuration Register (PISIDCR); used to enable/disable exact match ingress stream identification lookups and to specify which key construction rule profile to use.
- ENETC registers for ingress stream identification key construction are noted below.

- Ingress Stream Identification Key Construction a Operational register (ISIDKC*a*OR); indicates the number of entries in the Ingress Stream Identification table for particular key construction rule, where *a* =0..1 represents the key construction rule ID.
- Ingress Stream Identification Key Construction a Configuration Register 0 (ISIDKC*a*CR0), where *a*=(0..1) represents the key construction rule ID.
- Ingress Stream Identification Key Construction a Payload Field 0 Configuration Register (ISIDKC*a*PF0CR), where *a*=(0..1) represents the key construction rule ID.
- Ingress Stream Identification Key Construction a Payload Field 1 Configuration Register (ISIDKC*a*PF1CR), where *a*=(0..1) represents the key construction rule ID.
- Ingress Stream Identification Key Construction a Payload Field 2 Configuration Register (ISIDKC*a*PF2CR), where *a*=(0..1) represents the key construction rule ID.
- Ingress Stream Identification Key Construction a Payload Field 3 Configuration Register (ISIDKC*a*PF3CR), where *a*=(0..1) represents the key construction rule ID.
- Port Ingress Stream Identification Configuration Register (PISIDCR); used to enable/disable exact match ingress stream identification lookups.

The Ingress Stream Identification table is managed using command messages. See Ingress Stream Identification Table for the switch and Ingress Stream Identification Table for ENETC.The Ingress Stream Identification table entry includes the following matching key fields:

- Key construction rule ID (referred to as KEY_TYPE in the Ingress Stream Identification table)
- Source port ID; applicable to the switch only
- Source Port Masquerading (SPM); applicable to the switch only
- Frame portion of the key (up to 16 bytes) frame header key construction is specified in ISIDKC*a*CR0 and frame payload key construction is specified in ISIDKC*a*PF*b*CR , where *a* represents the rule ID and *b* represents the payload field.

The frame portion of the key format (frame key) is specified by the aforementioned key construction rule registers. The frame key entered in the Ingress Stream Identification table entry has to be in packed form in the order they appear in the frame with each field formatted in big endian format (order they are received from the link). If the constructed key is less than 16 bytes then the frame key entered in the Ingress Stream Identification table entry, must be padded with zeros for the rest of the bytes.

Protocol header data or payload data has to be added to the key in the Ingress Stream Identification entry if indicated as 'present' in key construction register. Format of protocol header data or payload data in 16 byte key (referred to as Key) is specified below.

- Source port and SPM are included in the key if respective 'present' bit is set in the key construction register. These values are part of the reserved area in the hash key and do not add bytes to the frame portion of the key.
- Following are the options for the Ethernet header:
  - If SMAC or DMAC is present in key construction register then each of these adds 6 bytes of data to the key. If both are present then DMAC has to follow SMAC.
  - If VID or PCP are indicated as present in key construction register then the following is added to the key: VID only, PCP only, VID and PCP, or 'no VLAN tag present in the frame'. VID and PCP can be specified for both inner and outer VLANs.
  - A one-byte frame attribute code point is included in the key indicating whether there is no 802.1CB R-TAG/HSR tag, 802.1CB draft 2.0 R-TAG present, 802.1CB R-TAG present or HSR tag present.
  - If EtherType is present in key construction register then an EtherType is added to the key. The EtherType extracted is located immediately after source MAC address or VLAN tag(s)/R-TAG/HSR (if any).
  - ISIDKCaPFbCR registers specify the key format for the payload data (where 'a' indicates the key construction rule used for the exact match lookups and 'b' indicates the payload field). The key format is big endian (network byte order).

53.4.2.4.1.3.5.2 Per-Stream Filtering

Based on the selected IS_EID, the Ingress Stream table entry is read from Ingress Stream table to obtain the actions to be performed for the received frame:

- Forwarding options for switch function - Discard frame, redirect/copy frame to switch management port, use 802.1Q bridge forwarding function or use stream forwarding which by-passes the 802.1Q bridge forwarding function.
- Forwarding options for ENETC function - Discard frame, forward with SI bitmap or forward without SI bitmap set.
- Specify whether to perform the maximum Service Data Unit (SDU) check. See Maximum SDU Check for more details.
- Override IPV, DR
- The frame is to be mirrored to the mirror destination specified in the IMDCR0 register (applicable to the switch only).
- Enable timestamp capture if not already enabled (applicable to the switch only).
- Enable/disable Ingress Sequence Generation (applicable to the switch only).
- Enable/disable Ingress Stream Gating operation. If enable, the stream gate instance identifier is provided to indicate the stream gate instance to be used.
- Option to override Rate Policer Entry ID.
- Option to disable cut through frame transfer on a specific destination port or on all destination ports (applicable to the switch only).
- Specify egress packet processing actions (applicable to the switch only)
- Specify the Ingress Stream Counter ID to count matched frames and count discarded frames. Discard counts are maintained for maximum SDU drops, Stream Gating frame drops and Rate Policing frame drops. If Ingress Stream Counter ID is set to NULL, no Ingress Stream Counter table entry is updated. If the entry referred to by the Ingress Stream Count ID is not found in the Ingress Stream Count table (entry has not been added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0).
- Option to specify performing an Ingress Stream Filter table lookup; to support a PSFP (IEEE 802.1Qci) stream filtering function that uses the 3-bit PCP value (priority) of the received frame and the stream handle (i.e. Ingress Stream Entry ID) to select a stream filter. Note that the Ingress Stream table is used to support stream filters that require an exact match only on the stream Id (PCP is wild-carded). Note that to support stream filters that requires exact match only on the PCP (stream Id is wild-carded), multiple Ingress Stream table entries must be created for a given stream, one for each of the PCP value that requires matching. For the latter, the PCP must be included as one of the parameter/key field to determine the Ingress Stream Entry ID (IS_EID).
- Option to specify Ingress Frame Modification Entry ID (IFM_EID) to perform ingress frame modification operation (applicable to the switch only).

If a frame is discarded, in addition to updating the relevant discard counts in the Ingress Stream Count table, discard count and discard reasons are updated in PRXDCR, PRXDCRRR and in PRXDCRR0/1 registers.

The Ingress Stream table is an index table. There is a single instance of this table for the switch and an instance per ENETC function. The IS_EID is used as the index to this table. For more details, see Ingress Stream Table for the switch and Ingress Stream Table for ENETC. Ingress stream table capabilities and configuration are specified in following registers:

- Ingress Stream Index Table Capability Register (ISITCAPR)
- Ingress Stream Index Table Memory Allocation Register (ISITMAR)
- Ingress Stream Index Table Operational Register (ISITOR)

If the entry referred to by the selected IS_EID (non-NULL value) is not found in the Ingress Stream table (entry has not been added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0). If the selected IS_EID is the default IS_EID specified in PISIDCR[ISEID], and is set to a value which indexes outside of the available Ingress Stream table entries, the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along

with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0). Note that IS_EIDs configured in tables (Ingress Port Filter table, Ingress Stream Identification table), are checked at configuration time to ensure that the ID falls within the range of valid indexes. If the IS_EID is out of range and is not set to NULL, the table configuration operation is not performed. If the IS_EID is set to NULL, the stream filtering function is by-passed.

53.4.2.4.1.3.5.3 Per-Stream, Per-Priority Filtering

To support a PSFP (IEEE 802.1Qci) stream filtering function that uses the 3-bit PCP value (priority) of the received frame and the stream handle (i.e. Ingress Stream Entry ID) to select a stream filter, an additional lookup (i.e. against the Ingress Stream Filter table) must be performed to obtain the PSFP (IEEE 802.1Qci) filtering and policing actions. If a lookup is performed against the Ingress Stream Filter table, the PSFP (IEEE 802.1Qci) filtering and policing actions from the Ingress Stream table entry are overridden by the actions from the matched Ingress Stream Filter table entry. If no match is found in the Ingress Stream Filter table, processing moves on with the actions from the matched Ingress Stream table entry.

As stated earlier on, in the case where the stream handle (i.e. Ingress Stream Entry ID) is used to select a stream filter, the Ingress Stream table entry contains the PSFP (IEEE 802.1Qci) filtering and policing actions that are to be applied to frames received on this stream. Furthermore to support stream filters that requires exact match only on the PCP (stream Id is wild-carded), multiple Ingress Stream table entries must be created for a given stream, one for each of the PCP that requires matching. Additionally, the PCP must be included as one of the parameter/key field to determine the Ingress Stream Entry ID (IS_EID).

The Ingress Stream Filter table is a hash table accessed using the IS_EID specified in the Ingress Stream table entry and PCP in the VLAN tag of the received frame. For more details, see Ingress Stream Filter Table for the switch and Ingress Stream Filter Table for ENETC.

If a match is found, the following actions can be specified:

- Options to override IPV, DR.
- Option to mirror frame to the mirror destination specified in the IMDCR0 register (applicable to the switch only).
- Option to disable cut through forwarding of frame if not already disabled (applicable to the switch only).
- Enable timestamp capture if not already enabled (applicable to the switch only).
- Option to enable/disable Ingress Stream Gating. If enable, the stream gate instance identifier is provided to indicate the stream gate instance to be used.
- Option to override Rate Policer Entry ID.
- Specify the Ingress Stream Counter ID to count matched frames and count discarded frames. Discard counts are maintained for maximum SDU drops, Stream Gating frame drops and Rate Policing frame drops. If Ingress Stream Counter ID is set to NULL, no Ingress Stream Counter table entry is updated. This field overrides the Ingress Stream Counter ID field specified in the previous matched Ingress Stream table entry. If the entry referred to by the Ingress Stream Count ID is not found in the Ingress Stream Count table (entry has not been added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0).
- Specify the maximum SDU check. See the following section for more details.

If a frame is discarded, in addition to updating the relevant discard counts in the Ingress Stream Count table, discard count and discard reasons are updated in PRXDCR, PRXDCRRR and in PRXDCRR0/1 registers.

53.4.2.4.1.3.5.4 Maximum SDU Check

Maximum SDU check is performed prior to policing where frame will be dropped if frame size exceeds maximum SDU. Note that the maximum SDU check is not performed, if the frame is being forwarded in the cut-through mode of operation, to at least one egress port. This means that the maximum SDU check is not performed for a frame that is replicated, where some copies are forwarded as cut-through frames while others are forwarded as store-and-forward frames. A value of zero disables the maximum SDU check.

53.4.2.4.1.3.6 Ingress Frame Modification

This function handles the ingress frame modifications, with the exception of the R-TAG/HSR header modification which is handled by the FRER sequence generation function. It provides ability to modify the MAC address and to remove, insert, swap VLAN tag.

The ingress frame modifications are specified in a Frame Modification table entry which in turn is referred to in an Ingress Stream table entry. For more details, see Frame Modification Table. If the index refers to an entry that is not found in the Frame Modification table (entry has not been added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0). There is also the option of encoding a subset of the frame modification actions in the actual frame modification entry ID. For the latter there is no need to access the frame modification table as the actions are encoded in the ID itself. For more details on the definition of the Frame Modification table entry ID, see Frame Modification Entry Definition.

The supported frame modification actions/operations are:

- Replacing destination MAC address (DMAC) with specified MAC address.
- Replacing VLAN tag (TPID, VID, PCP, and DEI).
- Adding one VLAN tag.
- Deleting one VLAN tag.

The above frame modifications take effect for constructing keys in subsequent 802.1Q bridge forwarding lookups.

As for other frame modifications, the effective frame length change of the ingress frame modification actions will be reflected in any of the egress processing functions. However, frame length changes resulting from the ingress frame modification actions will not be reflected in any ingress packet processing functions (e.g. policing function). Note that the ingress frame modifications effective length change must be explicitly configured by software.

The ingress frame modification actions along with other frame modification actions (i.e. FRER sequence generation modification and egress frame modifications) are passed along with the frame as metadata up to the Ethernet Tx I/F (or Tx MAC client) for actual frame modification (except for switch management port traffic with Host Reason set to non-zero where the actual frame modification will not take place, as the frame is to be delivered as received from the link). Ingress frame modification operations on the VLAN headers are based on the VLAN classification validated outer and inner VLAN header (present and valid).

In the case of replication, all ingress frame modification actions must also be applied to each replicated frame.

When adding or replacing a VLAN tag, TPID, PCP, DEI and VID are set as follows:

- The VID can be explicitly specified or copied from outer VLAN tag header.
- The PCP and DEI can be explicitly specified, copied from outer VLAN tag header of the frame. PCP can also be derived from a specified PCP to PCP mapping profile.
- The TPID can be explicitly specified or copied from outer VLAN tag header of the frame.

In general, if an ingress frame modification action cannot be performed (e.g. a delete outer VLAN header action on a frame that has no VLAN header), this will result in a misconfiguration event. This misconfiguration event is handled by discarding the frame and incrementing the port's Rx discard count register (PRXDCR) along with the setting of the Frame Modification Misconfiguration Error Discard Reason (FMMEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0).

53.4.2.4.1.3.7 Frame Replication and Elimination for Reliability (FRER) sequence generation

Sequence generation (ingress) consists of tagging a frame with a sequence number, following which the frame can be replicated and transmitted along multiple paths for redundancy. Implemented as per the IEEE 802.1CB. Egress sequence generation is not supported.

Frames entering the switch can be identified by a stream ID, which indexes the Ingress Stream table. The contents of the ingress stream table can enable sequence generation, and if so then the ingress stream table also provides an index into a sequence generation table (See Ingress Sequence Generation Table for more detail). If the index refers to an entry that is not found in the Ingress Sequence Generation table (entry has not been added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the

port's Rx discard count reason register 0 (PRXDCRR0). If the index is set to NULL, the ingress sequence generation function is by-passed.

The sequence generation table provides the tag type to be inserted (currently 802.1CB R-TAG, 802.1CB draft 2.0 R-TAG and HSR tag are supported), and the next sequence number value (SQG_NUM) to be used in that tag. SQG_NUM resets to 0 when the sequence generation function is initialized, increments by one after each packet that passes through the sequence generation function, and wraps around from max value (all 1's) to 0. The sequence number space is 16 bits, therefore sequence numbers will roll over and repeat every 65536 packets.

Logically, after the sequence tag (containing the generated sequence number) is applied to the frame, the frame is then replicated to be carried to its destination via multiple paths, with each copy of the frame carrying the same sequence number. Implementation-wise, the insertion of the sequence tag is done as each copy of the frame is read from internal memory and transmitted.

When inserted, the sequence tag can be located either immediately after the source MAC address of the frame, or with 1 or 2 VLAN tags between the source MAC and the sequence tag. If there was already a sequence tag in the received frame, the new sequence tag will be inserted before the existing sequence tag. Sequence number from the new sequence tag will be used in the sequence FRER recovery function, if the function is enabled.

53.4.2.4.1.3.8 802.1Q Bridge Forwarding

The 802.1Q bridge forwarding function is used to perform the required lookups to determine the outgoing ports, including flooding and broadcast/multicast handling.

Supported 802.1Q bridge forwarding functions are

- **Ingress VLAN processing** - Performs VLAN tagging checking, VLAN membership checking and Spanning Tree Protocol (STP) input port state checking. It also indicates the Filtering ID to be used by the forwarding function.
- **MAC learning (hardware)** - Entries are added to the FDB table when new unique MAC addresses are learned on the datapath.
- **IP multicast filtering** - Multicast frames are forwarded based on source IP addresses and destination IP addresses using L2 IP multicast filtering tables.
- **MAC forwarding** - Frames are forwarded based on Layer 2 destination MAC addresses using a MAC Filtering or Forwarding Database (FDB) table.
- **Egress VLAN processing** - Performs VLAN membership checking and Spanning Tree Protocol (STP) output port state checking.
- **Storm control** - Limits the amount of broadcast, multicast, and flooded unicast/multicast traffic.

53.4.2.4.1.3.8.1 Ingress VLAN Processing

The following checks are performed to determine to whether admit or drop incoming frames. See the per port registers BPDVR[RXTAGA] for more details.

- Accept Untagged frames
- Accept Priority tagged frames
- Accept Single tagged frames
- Accept Double tagged frames

For the above checks, the outer VLAN tag in a received frame is considered present if VLAN classification has deemed the outer VLAN tag valid. Similarly, The inner VLAN tag in a received frame is considered present if VLAN classification has deemed the inner VLAN tag valid. These checks do not take into consideration, any VLAN header modification specified by an earlier frame modification function.

If the frame is admitted, the next step consists of assigning a VID to the frame. All frames to be forwarded using the 802.1Q bridge forwarding function must be a member of a VLAN, either assigned from the frame VLAN header or through the port default configuration. The frame VLAN header input into this function takes into account any VLAN header modification operations specified in previous functions. For example, if the incoming frame was untagged and a VLAN header "add" action was specified

in the ingress frame modification function, the resulting VLAN header from that action is used as input to the VLAN assignment operation (i.e. treated as the VLAN header of the frame).

Assigned VID for a frame is dependent on whether the port is configured as VLAN unaware bridge or a VLAN aware bridge (see register BPDVR[RXVAM]).

- **VLAN-unaware** - If the source port of the incoming frame is attached to a VLAN-unaware bridge, a VID from the port default (BPDVR[VID]) is assigned to the frame regardless of whether the frame is tagged or untagged, and this VID is used for subsequent VLAN operations.
- **VLAN-aware** - If the source port of the incoming frame is attached to a VLAN-aware bridge, received frames are treated as described below.
  - If frame is untagged, or VLAN classification has deemed the outer VLAN tag valid with VID=0 (priority tag) or VLAN classification has deemed the outer VLAN tag not valid, and no ingress frame modifications have been specified, an internal VLAN header is built using the VID, DEI, PCP and TPID from the per port register BPDVR. This internal VLAN header is carried along with the frame as metadata, and the VID from this header is used for the subsequent VLAN operations.
  - If VLAN classification has deemed the outer VLAN tag valid with a non-zero VID (not a priority tag) and no ingress frame modifications have been specified, the assigned VID is taken from the frame's outer VLAN header.
  - If there are any ingress frame modification actions that have been specified, and the resulting ingress frame modification actions result in the frame containing an outer VLAN header with a non-zero VID (not a priority tag), the VID from this outer VLAN header is used for the subsequent VLAN operations. If the resulting ingress frame modification actions result in the frame containing no VLAN headers, or with an outer priority tag (VID=0), the frame is discarded and counted against the bridge port discard count register (BPDCR) along with the setting of the Untagged Frame Modification Misconfiguration Discard Reason (UFMMDR) flag to 1 in the Bridge port discard count reason register 0 (BPDCRR0).

Once a VID has been assigned to the frame, a lookup against the VLAN Filter table (See VLAN Filter Table for more details) using the VID assigned to the frame, is performed to determine the following:

- **VLAN membership** - The VLAN membership defines which ports belong to the VLAN. A check is performed to determine if the source port of the incoming frame matches one of the port that is a member of the VLAN. If no match is found, the frame is discarded.
- **Spanning tree group** - Spanning tree group to which the VLAN belongs is provided. The state of the source port in the spanning tree group is checked to determine action to be performed.
- **Filtering ID** - Filtering ID to be used when performing lookups against the FDB table and the L2 IPV4 Multicast Filter table.
- **VLAN handling options** - For example:
  - MAC learning options.
  - MAC forwarding options (e.g. whether to perform an FDB lookup).
  - Whether to perform an L2 IP Multicast Filtering lookups.

The related egress VLAN information retrieved from the VLAN Filter entry is carried along with the frame for usage during egress VLAN processing. If no match is found in the VLAN Filter table, information specified in the default VLAN configuration registers VFHTDECR0-VFHTDECR2 is used to determine the VLAN handling parameters and actions.

53.4.2.4.1.3.8.2 MAC Learning and Forwarding

The following tables are used to support the 802.1Q bridge forwarding operations:

L2 MAC Filtering or Forwarding Databases (FDB) table

Exact match table (augmented with a CAM for guaranteed entries) containing forwarding and/or filtering information about MAC addresses. Each table entry includes a FID and MAC address that may be unicast or multicast and a forwarding destination field containing a port bitmap identifying the associated port(s) with the MAC. The entry also specifies the egress packet processing to be performed for each of the destination ports. FDB table entries can be static or dynamic (can be aged out by software). Static

entries are added by the host whereby dynamic entries are added either by the host or by the hardware as MAC addresses are learned in the datapath. See FDB Table for more details.

The FDB is implemented as a hash table optionally augmented with a CAM for guaranteed entries. Entries are added as a hash entry first (hash space), and if an entry is unable to be added due to collision chain length having reached its maximum length, defined in the register HBTCAPR[MAX_COL], it will be added to the CAM and its associated indexed table. An entry will not be added to the hash space or CAM/indexed table if an entry limit has been hit. If there is at least one entry added to the CAM/indexed table (tracked internally), then the CAM is searched in parallel with searching of the hash table. If there is a match in the CAM, then the indexed table is read to find the full entry. If there is no match in the CAM, the hash search is checked for a match entry. This does not add any processing time since the read/search operations are done in parallel.

L2 IPv4 Multicast Filter table

Exact match table containing forwarding and/or filtering information about IP multicast addresses. The L2 IPv4 Multicast Filter table is implemented as hash table. L2 IPv4 Multicast Filter table entries can be static or dynamic (can be aged out by software). Contrary to the FDB table, entries whether static or dynamic can only be added by the host software. The following two L2 IPv4 Multicast Filter table entry types are supported:

- **IPV4 Source Specific Multicast (SSM) Multicast Filter table entry** - Each table entry includes the key type identifying the entry as a SSM table entry, a FID, source IPv4 address and destination multicast IPv4 address, and a forwarding destination field containing a port bitmap identifying the ports to which the frame should be forwarded for frames matching this entry. The entry also specifies the egress packet processing to be performed for each of the destination ports. For more details, see L2 IPV4 Multicast Filter Table.
- **IPV4 Any Source Multicast (ASM) Multicast Filter table entry** - Each table entry includes the key type identifying the entry as a ASM table entry, a FID and destination multicast IPv4 address, and a forwarding destination field containing a port bitmap identifying the port(s) within to which the frame should be forwarded for frames matching this entry. The entry also specifies the egress packet processing to be performed for each of the destination ports. For more details, see L2 IPV4 Multicast Filter Table.

An activity element is also present for each entry in the FDB or L2 IPv4 Multicast Filter table to allow the Host to age the entry if necessary. The activity element is represented as an 8-bit value where bit 0 (least significant bit) corresponds to the activity flag and bits 7-1 correspond to the activity counter.

When an FDB lookup (for the purpose of frame forwarding) or L2 IPv4 Multicast Filtering lookup finds a match, and the activity flag is set to 0, hardware will set the activity flag to 1, to indicate the entry has been accessed. In the case of impending congestion due to the hardware being heavily loaded simultaneously from both the control-plane and data-plane, hardware will not perform the above activity bit update operation to ensure adequate data-plane control-plane and command processing capacity. The activity element is updated by software when issuing the appropriate table management command as follows:

- If the activity flag is set to 1, then the entire 8-bit activity element is set to 0.
- If the activity flag is set to 0, then the 7-bit activity element counter is incremented by 1, however it does not roll over if the counter reaches the value of 127.

The MAC learning and forwarding function implements a 3-stage table lookup pipeline; MAC learning lookup followed by IP Multicast filtering and then MAC forwarding (details of this is given below).

MAC Learning

MAC learning is the process where one dynamically updates the FDB using the MAC source address of the frames received and the port in which the frames have entered the switch.

The MAC learning options (MLO) code point set by the ingress VLAN processing function determines what MAC learning processing occurs on the frame:

- 0x0 = Reserved.
- 0x1 = Disable MAC learning. SMAC FDB lookup is by-passed.
- 0x2 = Hardware MAC learning is enabled. See the Hardware MAC Learning sub-section below, for more details.
- 0x3 = Software MAC learning secure. FDB lookup based on FID and SMAC is performed and if an entry is not found, the frame is redirected to the switch management port. If an entry is found and the ingress port of the incoming frame

is not set in the FDB entry Destination Port Bitmap and BPCR[STAMVD] is set to 0 (station move allowed), the frame is redirected to the switch management port regardless of whether it is a dynamic or static FDB entry. If BPCR[STAMVD] is set to 1, the frame is discarded.

- 0x4 = Software MAC learning unsecure. FDB lookup based on FID and SMAC is performed and if an entry is not found, the frame is copied to the switch management port. If an entry is found and the ingress port of the incoming frame is not set in the FDB entry Destination Port Bitmap and BPCR[STAMVD] is set to 0 (station move allowed), the frame is copied to the switch management port regardless of whether it is a dynamic or static FDB entry. If BPCR[STAMVD] is set to 1, the frame is discarded.
- 0x5 = Disable MAC learning with SMAC validation. FDB lookup based on FID and SMAC is performed and if an entry is not found, the frame is discarded. If an entry (dynamic or static) is found and the ingress port of the incoming frame is not set in the FDB entry Destination Port Bitmap, the frame is discarded. The frame discard is counted against the bridge port discard count register (BPDCR) with discard reason BPDCRR0[MACLNFDR] set to 1.
- All other values reserved

The figure below shows the MAC learning flowchart.

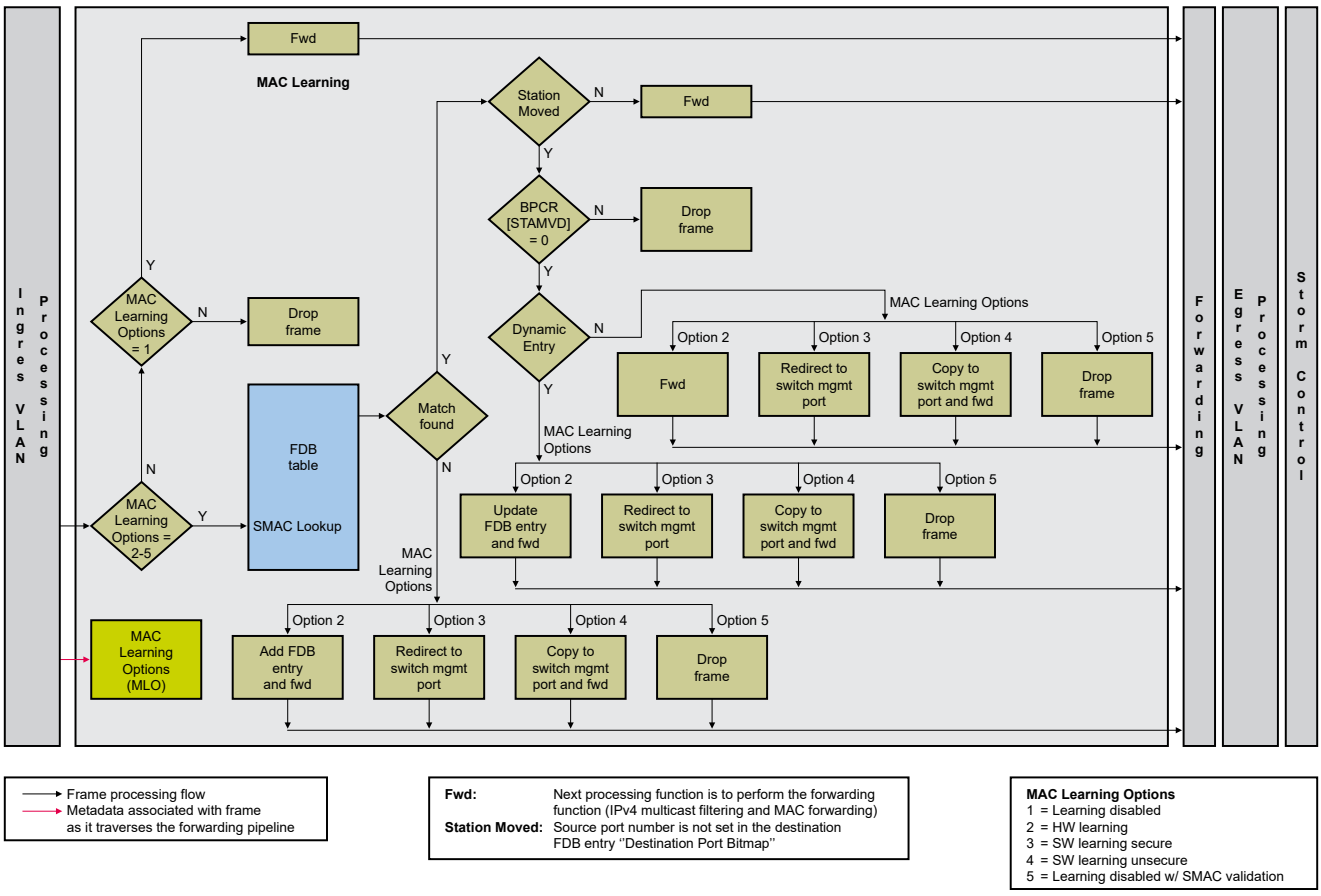


Figure 349. MAC Learning Flowchart

Hardware MAC Learning

When the hardware MAC learning function is selected, a lookup is made into FDB table using the FID and source MAC address. If there is no match then a new entry (dynamic) is added in the FDB table for that source MAC address unless:

- Maximum number of dynamic FDB entries from a port has been reached (see BPCR[DYN_LIMIT]); entries created by the MAC learning function are dynamic entries, note that software can also create dynamic entries via table management commands.

- Maximum number of dynamic FDB entries for the entire switch has been reached (see FDBHTMCR[DYN_LIMIT]); entries created by the MAC learning function are dynamic entries, note that software can also create dynamic entries via table management commands.
- Hash collision chain limit has been reached (limit is specified in HBTCAPR[MAX_COL]) and the CAM (guaranteed FDB entries) if present is full.
- Memory allocated to store the switch hash tables is full (the maximum amount of memory words available to store all hash tables entries, is indicated in the register HTMCAPR[NUM_WORDS]) or,
- Impeding congestion has been detected due to the hardware being heavily loaded simultaneously from both the control-plane and data-plane. In this situation, updating the FDB table with the learned MAC address is not performed to ensure adequate datapath and control-plane command processing capacity.

If the entry cannot be added, the frame is still forwarded normally and the Bridge Port MAC Learning Failure Status Register (BPMLFSR) is updated accordingly. The BPMLFSR register includes a bit field (w1c) for each of the possible MAC learning failure reasons.

- The per port dynamic FDB table entries limit has been reached.
- Memory allocated to store the hash tables is full (the maximum amount of memory words available to store all hash tables entries, is indicated in the register HTMCAPR[NUM_WORDS]) or FDB table dynamic entries limit has been reached.
- Hash collision chain limit has been reached (limit is specified in HBTCAPR[MAX_COL]).

If the source MAC address is found but the ingress port of the incoming frame is not set in the FDB entry Destination Port Bitmap and BPCR[STAMVD] is set to 0 (station move allowed), then the FDB entry is updated unless it is a static entry. The FDB table entry is updated by setting the ingress port of the incoming frame to 1, in the Destination Port Bitmap and setting all other ports in the Destination Port Bitmap to 0. In addition to updating the entry, the BPOR[NUM_DYN] counter for the port that has its bit changing from 0 to 1, will be incremented and the BPOR[NUM_DYN] counter for each of the ports that has its bit changing from 1 to 0, will be decremented. If it is a static entry, no update takes place and the frame moves to the next function (forwarding). If BPCR[STAMVD] is set to 1, the frame is discarded. Note that the port which has the bit changing from 0 to 1 may result in exceeding its maximum number of dynamic entries specified in BPCR[DYN_LIMIT]. If impeding congestion has been detected due to the hardware being heavily loaded simultaneously from both the control-plane and data-plane, the update to the FDB is not performed to ensure adequate datapath and control-plane command processing capacity.

No messages are sent to the host when hardware adds or updates an entry in the FDB table.

MAC Forwarding

The figure below shows the forwarding flowchart.

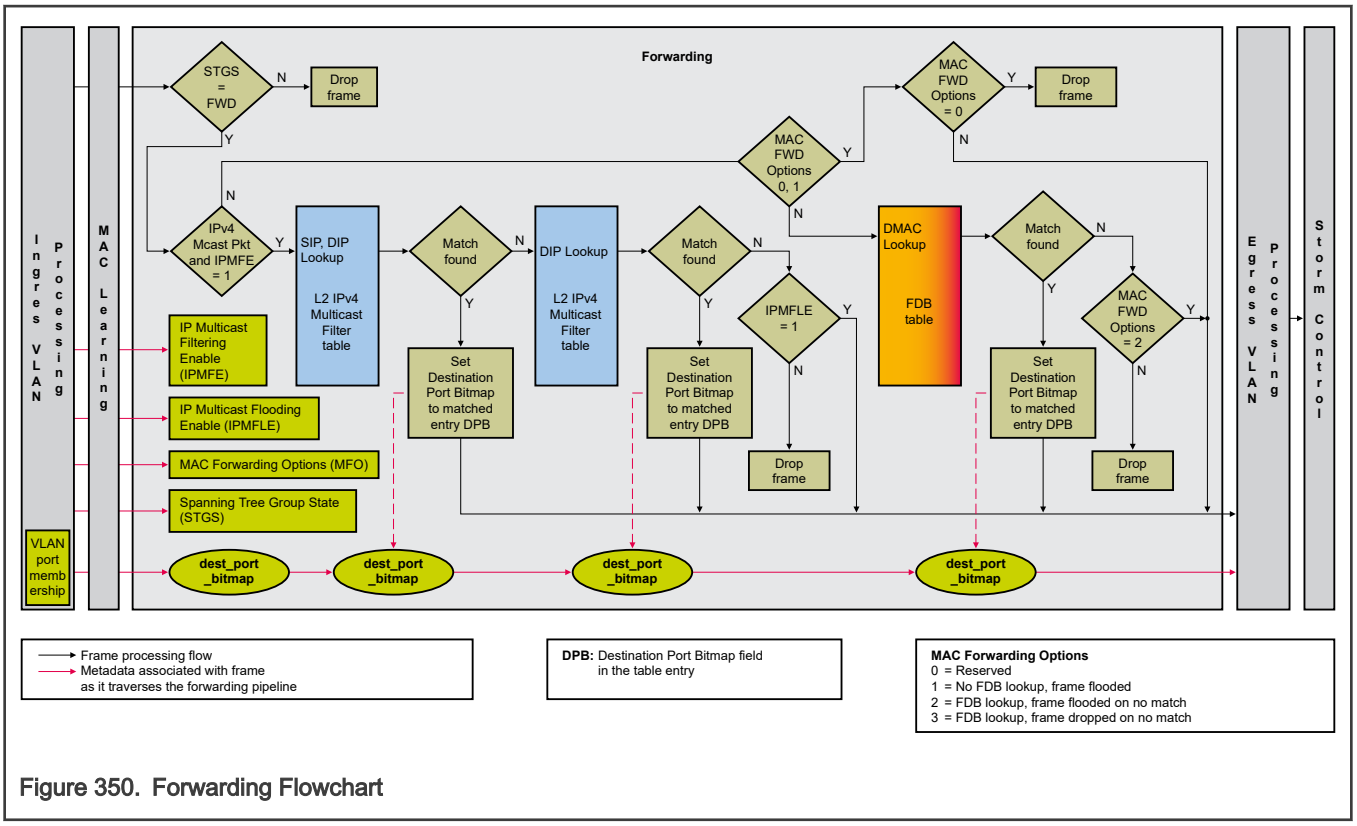


Figure 350. Forwarding Flowchart

IP Multicast Filtering

If the indicator of whether IP multicast filtering is to be performed (IP Multicast Filtering Enable (IPMFE) flag from ingress VLAN processing), is set to 1, and the frame is identified as a multicast IP packet (destination IPv4 address is 224.0.1.0-238.255.255.255), then IP multicast filtering is performed.

IP multicast filtering consists of performing a Source Specific Multicast (SSM) exact match lookup (if at least one entry added) utilizing both source IP address and destination IP address, and if there is no match, followed by Any Source Multicast (ASM) exact match lookup (if at least one entry added) utilizing destination IP address only.

If IP multicast filtering finds a match, frame is processed according to the matched entry; i.e. forward according to the port bitmap stored in the entry, followed by skipping MAC forwarding. If no match is found, frame is either admit or drop according to the setting of the IP Multicast Flooding Enable (IPMFLE) flag (from ingress VLAN processing). If the frame is discarded, the frame discard is counted against the bridge port discard count register (BPDCR) along with the setting of the IP Multicast Filter Discard Reason (IPMFDR) flag to 1 in the Bridge port discard count reason register 0 (BPDCRR0). If the frame is not identified as an IP multicast packet, processing moves to MAC forwarding.

If the indicator of whether IP multicast filtering is to be performed (IP Multicast Filtering Enable (IPMFE) flag from ingress VLAN processing), is set to 0, then processing moves to MAC forwarding.

MAC Forwarding

The MAC forwarding options (MFO) code point set by the ingress VLAN processing function determines what MAC forwarding processing occurs on the frame:

- If MFO=1, no FDB lookup is performed, the frame is flooded.
- If MFO=2, a lookup into the FDB using the destination MAC address and FID, is performed. If a match is found, the frame is forwarded according to the port bitmap stored in the entry. If the FDB lookup does not find a match, the frame is flooded.
- If MFO=3, a lookup into the FDB using the destination MAC address and FID, is performed. If a match is found, the frame is forwarded according to the port bitmap stored in the entry. If the FDB lookup does not find a match, the frame is discarded.

53.4.2.4.1.3.8.3 Egress VLAN Processing

Information obtained earlier from the VLAN Filter table entry is used to determine:

- **VLAN membership** - The 802.1Q bridge function can only forward a frame to ports that are members of the VLAN. Ports outside the VLAN membership are filtered out; i.e. their corresponding bit in the destination port bitmap is set to 0.
- **Spanning tree group** - Spanning tree group to which the VLAN belongs is provided. The state of the output port in the spanning tree group is checked to determine action to be performed. If the port state is not set to 2 (forwarding), the frame is not forwarded towards that port. No transmit discard counter is incremented unless the resulting destination port bit map is null, then the bridge port discard count register (BPDCR) is incremented with the NODESTDR reason flag set to 1 in the bridge port discard count register 0 (BPDCRR0).
- **Source pruning** - If source pruning is enabled (BPCR[SRCPRND]=0), a received frame is not allowed to be transmitted on the same port it was received. If source pruning is disabled (BPCR[SRCPRND]=1), a received frame is allowed to be transmitted on the same port it was received.
- **Egress processing actions** - For each port member of the VLAN, optionally specifies additional egress processing actions such as whether to delete and/or add a VLAN tag. These actions are specified using the Egress Treatment table. If there is no Egress Treatment entry configured or if the frame modification entry ID configured in the Egress Treatment table entry is set to null, port level egress VLAN header modifications specified through Bridge port default VLAN register (BPDVR) register are applied to the frame if the port is configured as a VLAN aware bridge.

53.4.2.4.1.3.8.4 Storm Control

A storm control policer can be specified via BPSCR0/R1 registers. A rate policer instance can be specified for each of the following types of traffic

- Multicast traffic
- Broadcast traffic
- Unknown (or flooded) unicast traffic
- Unknown (or flooded) multicast

If a frame belongs to a traffic type that has a storm control rate policer configured in BPSCR0/R1 registers, the Rate Policer Entry ID (RP_EID) specified in the register is used to access the Rate Policer Instance table. For more details on managing this table , See Rate Policer Table. If the entry referred to by RP_EID (non-NULL value) is not found in the Rate Policer table (entry has not been added to the table), or if the RP_EID is set to a value which indexes outside of the available Rate Policer table entries, the frame is discarded and counted against the bridge port discard count register (BPDCR) along with the setting of the Misconfiguration Discard Reason (MISCDR) flag to 1 in the port's bridge port discard count reason register 0 (BPDCRR0).

Rate policer instances used for storm control are distinct from the ones used for the rate policing. Rate policer instances cannot be shared between these two distinct functions. Note that a given frame can be subjected to both the storm control policing function and the rate policing function, but different rate policer instances must be used.

If the frame is being forwarded in the cut-through mode of operation, to at least one port, the initial segment size of the cut-through frame is used by the rate policer for checking if it accepts the frame. If the frame is accepted, the rate policer byte credits get updated as data is received (cut-through segment transfers).

Dropped frames by a rate policer instance are counted in against the Bridge port discard count register (BPDCR), the port's Rx discard count register (PRXDCR), the Rate Policer table entry drop counter and the Ingress Stream Count table entry counter.

53.4.2.4.1.3.9 Policing

This function implements traffic policing as per IEEE 802.1Qci (Per-Stream Filtering and Policing) to protect against time window (stream gating) and bandwidth violations (rate policing).

53.4.2.4.1.3.9.1 Stream Gating

Whether to perform stream gating is determined earlier by the stream identification and filtering function. If stream gating is to be performed, a Stream Gate Instance Entry ID (SGI_EID) is provided to access the Stream Gate Instance table.

The Stream Gate Instance table contains the stream gate configuration and operational information for the switch or an ENETC instance. Each entry contains a set of parameters that defines a single stream gate instance. A stream gate instance includes a gate control list, which specifies the time gates or time slots during which incoming frames from one or more streams will be accepted.

The gate control lists are typically generated offline, and then timely configured on each network device. To support no-downtime gate control list configuration changes, the standard requires that a network device supports two sets of gate control lists:

- **Operational gate control list** – Represents the currently active gate control list.
- **Administrative gate control list** - Provides a means of configuring a new gate control list prior to its installation in a running system. An administrative gate control list becomes operational at Config Change Time as defined in the IEEE 802.1 Qci specification and remains operational till another administrative stream gate control list is installed.

The hardware executes the gate control list cyclically based on a global time. The hardware starts executing the gate control list from the head, and then goes through the gate control entries as time goes on, and after the cycle time is passed, the hardware jumps back to the head of the gate control list.

A stream gate in a gate control list entry can be in one of the two states:

- **Open**: Frames pass through the gate
- **Closed**: Frames do not pass through the gate; frames are discarded.

There are two "Open Gate Check" modes (PSGCR[OGC]) available for determining if a frame can pass through an open gate:

- Only the start of frame needs to be within the open gate interval.
- Both the start and end of the frame need to be within the open gate interval.

For more details on managing this table, see Stream Gate Instance Table for ENETC and switch. If the entry referred to by SGI_EID is not found in the Stream Gate Instance table (entry has not been added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0). If the SGI_EID is set to NULL, the stream gating function is by-passed.

Note that a Stream Gate Instance has to be associated with streams received from the same port. Streams received from different ports cannot use the same Stream Gate Instance. This is not enforced by hardware; considered a software configuration error.

The actual stream gate control lists are stored in separate table called the Stream Gate Control List table, with their entry ID being specified in the Stream Gate Instance table. For a given Stream Gate Instance, only two stream gate control lists may be allocated to the instance. It is recommended that the entry ID for each of these two stream gate control lists be predetermined and fixed by software, and persist till the Gate Stream Instance is no longer in use.

Each Stream Gate Control List entry is of variable size, and is used to hold one stream gate control list. One or up to 256 gates can be specified for a gate control list. A list consists of one configuration 24-byte word plus a variable number of words to configure stream gates. Up to two stream gates can be specified in a word. The Stream Gate Control Entry ID (SGCL_EID) has to be specified in units of words such that it points to the first word in a stream gate control list.

The configuration word and the gate configurations for a gate control list has to be in consecutive words in Stream Gate Control List table. For example, if a gate control list starts at address offset 0x100 in SGCL table and has 9 gates specified, then, the gate list SGCL_EID is 0x100 and the stream gate control list configuration will be at address offset 0x100, configuration for gates 0 and 1 will be in 0x101, configuration for gates 2 and 3 are in 0x102, and so on, and configuration for gate 8 is in 0x105. Hence the gate control list starts at address offset 0x100 and ends at offset 0x105. As such the next gate control list can start at address offset 0x106.

For more details on managing the SGCL table, see Stream Gate Control List for ENETC and the switch.

For stream gating, frame reception timestamp can be aligned to an earlier reference timing point (e.g. at the remote port's transmit timing point). This is achieved by having the frame reception timestamp(s) adjusted by subtracting the delay specified in PSGCR[PDELAY], from the Start of Frame receive timestamp and from the End of Frame receive timestamp (in the case where PSGCR[OGC] is set to 1).

A stream gate control list contains an ordered list of gate operations. Each gate operation specifies the following:

- Gate state (open/close) that determines whether a frame is allowed to pass through the gate or not.
- Internal priority value (IPV).
- Time interval in nanoseconds.
- Interval maximum number of PDU/SDU (Protocol/Service Data Unit) bytes. An optional feature to limit the number of frame bytes that can pass an open gate during its time interval.
- Option to disable cut-through forwarding of frame if not already disabled (applicable to the switch only).

If the frame is being forwarded in the cut-through mode of operation, to at least one port, the initial segment size of the frame is used for checking interval maximum number of bytes. Frame is accepted if the accumulated interval byte count in a gate plus the number of bytes received in the first segment of the frame is less than or equal to the interval maximum number of PDU/SDU bytes, else the frame is discarded. If frame is accepted, accumulated interval byte count gets updated later when the entire frame is received. Note that in cut-through operation, it is possible that the interval maximum number of PDU/SDU bytes can be exceeded (at most by an PDU/SDU size minus the initial segment size).

If the frame is being forwarded in the store-and-forward mode of operation, to all ports identified as the destination for the frame, the frame is accepted if accumulated interval byte count in a gate plus the frame size is less than or equal to the interval maximum number of PDU/SDU bytes. Accumulated interval byte count in a gate is not updated if frame is discarded.

Statistics for stream gating related frame drops are kept in the Ingress Stream Count table.

Gate control list installation is carried on, as defined by IEEE 802.1Qci (Per-Stream Filtering and Policing)

NOTE

The 1588 timer must be initialized and configured before adding any administrative gate control list to a Stream Gate Instance. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer is in normal operation, any change to the 1588 timer configuration (for example, TMROFF_H/L), except for TMR_ADD updates, requires that all Stream Gate Instances with an operation gate control list be deleted, and for Stream Gate Instances that have no operation gate control list, that the administrative gate control list if present, be removed from the Stream Gate instance.

IERB registers:

- Stream Gating Lag Time For Timestamp Refresh Register (SGLTTR). Common for both ENETC and switch. Lag time is in units of nanoseconds. The refresh timestamp timer is used to trigger the hardware to traverse the stream gate control list up to the entry corresponding to the current time minus a lag time (value in SGLTTR) when there is no traffic being received on the port.
- Switch Stream Gate Instance Index Table Memory Allocation Register (S0SGIITMAR)
- Switch Stream Gate Control List Index Table Memory Allocation Register (S0SGCLITMAR)
- ENETC a Stream Gate Instance Index Table Memory Allocation Register (EaSGIITMAR)
- ENETC a Stream Gate Control List Index Table Memory Allocation Register (EaSGCLITMAR)

Switch / ENETC function registers:

- Stream Gate Capability Register (SGCAPR)
- Stream Gate Instance Index Table Capability Register (SGIITCAPR)
- Stream Gate Instance Index Table Memory Allocation Register (SGIITMAR)
- Stream Gate Instance Index Table Operational Register (SGIITOR)
- Stream Gate Control List Index Table Capability Register (SGCLITCAPR)
- Stream Gate Control List Index Table Memory Allocation Register (SGCLITMAR)
- Stream Gate Control List Table Memory Operational Register (SGCLTMOR) Port Rx Discard Count related Registers (PRXDCR, PRXDCRR0/1). When a discard is due to stream gate function the discard count will be updated in this register and following reasons will be specified.

— [SGCDR] - Stream Gate Closed discard reason.

— [MSDUEDR] Maximum SDU Exceed discard reason

53.4.2.4.1.3.9.2 Rate Policing

Whether to perform rate policing is determined in prior ingress processing functions. A rate policer instance (or more specifically a Rate Policer Entry ID (RP_EID)) can be specified in the Ingress Port Filter table, Ingress Stream table, and/or Ingress Stream Filter table. When more than one source specifies a rate policer instance, only one instance can be applied to a frame with the following precedent (with 1 having higher precedence):

1. Ingress Stream Filter table
2. Ingress Stream table
3. Ingress Port Filter table

By default (prior to any lookups), the RP_EID is set to NULL, meaning that no rate policing is to be applied to the frame.

Rate policing (or flow metering) is compliant to IEEE 802.1Qci specification.

If rate policing is to be performed, the Rate Policer Entry ID (RP_EID) specified by one of the above table lookups, is used to access the Rate Policer Instance table. For more details on managing this table , see Rate Policer Table for ENETC and Rate Policer Table for the switch. If the entry referred to by RP_EID (non-NULL value) is not found in the Rate Policer table (entry has not been added to the table), the frame is discarded and counted against the port's Rx discard count register (PRXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0). If the RP_EID is set to NULL, the rate policing function is by-passed.

The Rate Policer table contains the rate policer configuration and operational information for the entire function (switch or an ENETC function). Each entry contains a set of parameters that defines a single rate policer instance. A rate policer instance controls the amount of traffic (or bytes) that is allowed to go through for a particular received flow of frames.

The MEF (Metro Ethernet Forum) 10.3 technical specification "bandwidth profile algorithm for one flow per envelope processing without token sharing", is used.

When the rate policer instance is configured to operate in the color-aware mode, the incoming frame "color" is indicated through the Drop Resilience (DR) associated with the frame. When the rate policer instance is configured to operate in the color-blind mode, the incoming frame is assumed to always be green for the rate policer. The figure below shows the default decision making process. Some of the default decisions can be modified by configuration, for example setting MR, DOY or NDOR in a rate policer instance will change the default packet marking and drop decisions. Dropped frames by a rate policer instance are counted in against the port's Rx discard count register (PRXDCR), the Rate Policer table entry drop counter and the Ingress Stream Count table entry counter.

If the frame is being forwarded in the cut-through mode of operation (switch only), to at least one port, the initial segment size of the cut-through frame is used by the rate policer for making the decision to declare an arriving frame as Green, Yellow, or Red and then based on the color assigned to the frame, whether the frame is dropped or marked (overwriting the Drop Resilience (DR) assigned to the frame). If the frame is accepted, the rate policer byte credits get updated as data is received (cut-through segment transfers). If the total frame length turns out to be greater than the number of tokens in the bucket that was used to declare the color of the frame, then the bucket token count will go negative (effectively borrowing future tokens). This has a side effect of increasing the effective committed or excess burst size by up to a maximum frame size minus the size of the initial cut-through frame segment.

NOTE

The timer function must be configured and enabled before any rate policer instance is enabled (see Rate Policer table entry Functional Enable Element Data). Either the default nanosecond timer or the 1588 timer can be used to generate the required nanosecond resolution time. The default nanosecond timer is configured and enabled by default out of reset. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer is in normal operation, any change to the 1588 timer configuration (for example, TMROFF_H/L), except for TMR_ADD updates, requires that all rate policer instances be disabled (see Rate Policer table entry Functional Enable Element Data).

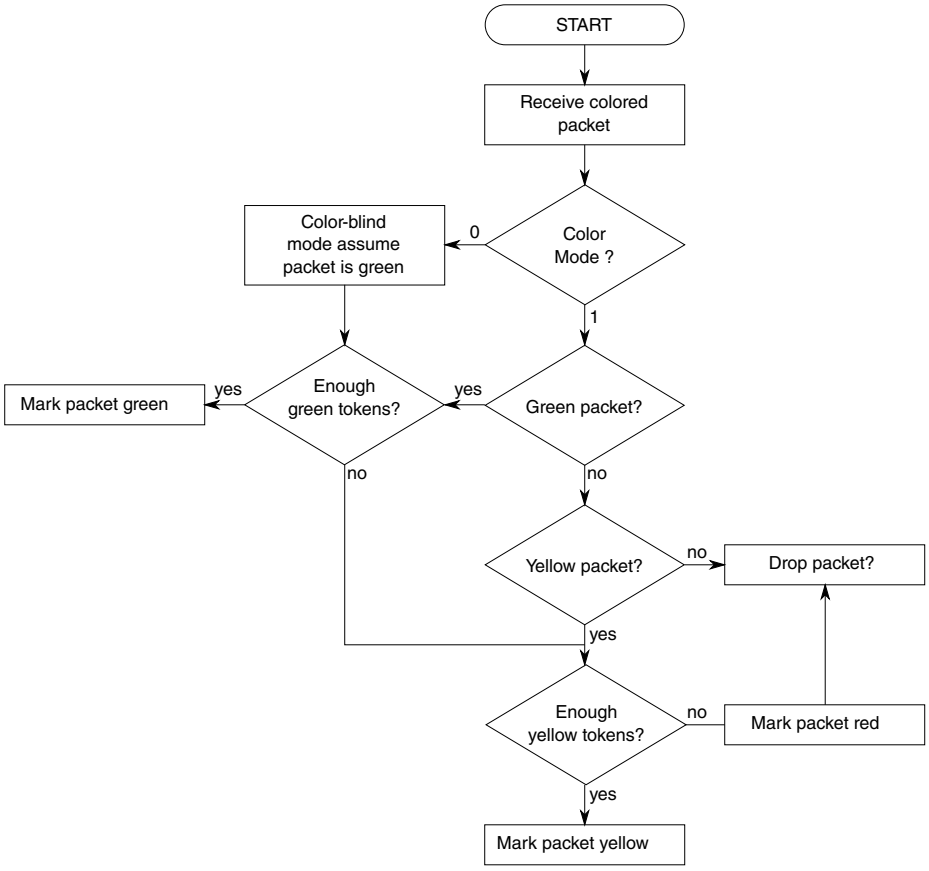


Figure 351. Bandwidth Profile Coloring Flow Diagram

53.4.2.4.1.3.10 Congestion Management

Buffer pool assignment and checking function is performed to ensure that there is available memory space to accommodate the new frame. If there is available memory space, then buffer pool accounting is updated accordingly, otherwise the frame is dropped.

The buffer pool is selected by mapping the IPV of the frame to a buffer pool ID using the port buffer pool mapping configuration register 0/1 (PBPMCR0/1). This mapping allows each IPV (or set of IPV's) to be mapped to a specific buffer pool for QoS distinction.

53.4.2.4.1.4 Replication and Egress Packet Processing

The Replication and Egress Packet Processing block is responsible for implementing the frame replication if needed, and packet processing functions in the context of the output port(s).

The switch destination port or set of destination ports to which the frame should be forwarded are determined by the Ingress Packet Processing block and are expressed as a destination bitmap, one bit per destination port.

For each of the ports that has its corresponding bit set in the destination bitmap, there may be the need to access the Egress Treatment table to retrieve the egress packet processing actions to be performed on the frame such as:

- **Egress frame modification** - Provides ability to remove, add and modify a single VLAN tag. It also provides the ability to modify Ethernet payload and Frame Replication and Elimination for Reliability (FRER) related egress header modifications. The actual frame modifications are not performed at this point, instead metadata is passed along with the

frame so that the actual frame modifications on the frame can be performed when the frame is being transferred to the MAC.

- **Frame Replication and Elimination for Reliability (FRER) sequence recovery** - Sequence recovery processing at this point involves checking that only one frame for each sequence number is passed, all duplicates are discarded. This way, multiple copies of the same frame that have been sent along different paths for redundancy will result in the first arriving frame being passed, and all other copies being removed. Implemented as per the IEEE 802.1CB.

In the case where the frame is destined to more than one port, only the frame descriptor is actually replicated, the frame itself is not. The frame is then passed to the Egress Queuing and Traffic Management block where it will be queued internally. In the case where the frame is destined to more than one port, multiple frame descriptors are passed, one for each egress port. If there is no memory available to store a frame descriptor, the hardware waits until memory becomes available; no egress frame drops are performed for this condition

53.4.2.4.1.4.1 Egress Treatment table access

Each entry in the Egress Treatment table contains the egress packet processing actions to be applied to a grouping or scope of packets exiting on a particular egress port of the switch. A scope of packets, for example could be the packets exiting a particular VLAN, matching a particular 802.1Q bridge forwarding entry or belonging to a stream identified at ingress. See Egress Treatment Table, for more details on the Egress Treatment table.

The means by which previous functions convey the Egress Treatment table entries to be accessed, is through the Egress Treatment group. An Egress Treatment group represents a group of Egress Treatment table entries.

Table entries belonging to the same group must have sequentially numbered indices and each table entry specifies the treatment of a scope of packets when they exit through a particular port of the switch. To conserve Egress Treatment table entries, an Egress Treatment group can omit table entries for ports for which there is no special treatment for that scope of packets. Therefore within the hardware, an Egress Treatment group is determined by a base index (first Egress Treatment table entry of the group) and an applicability port bitmap (a bitmap corresponding to all ports of the switch) which indicates (with a 1) which ports have Egress Treatment table entries. If the entry is not found in the Egress Treatment table (entry has not been added to the table), the frame towards that port associated with that entry is discarded. The frame discard is counted against the port's Tx discard count register (PTXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Tx discard count reason register 0 (PTXDCRR0).

Hardware allows a packet to belong to two scopes (or groupings) for the purpose of egress treatment, a primary (narrower) scope that is nested within a broader secondary scope. This is achieved by providing the ability to assign a primary and secondary Egress Treatment group to each packet. As a packet egresses on a particular port, hardware will only apply the egress processing actions defined by the secondary Egress Treatment group if there is no egress processing actions defined for that port by the primary Egress Treatment group or no primary Egress Treatment group has been assigned to the packet.

Ingress Stream table entries and FDB/L2 IPv4 Multicast filter table entries contain fields that allow matching frames to be (optionally) assigned to a primary Egress Treatment group. They contain a base index of the group and fields to convey the applicability port bitmap of the group. A primary Egress Treatment group assignment from the FDB/L2 IPv4 Multicast filter table has precedence over any assignment made via the Ingress Stream table.

The secondary Egress Treatment group may only be assigned through a lookup against the VLAN Filter table.

An example of a use case that makes use of both treatment groups simultaneously would be an ingress stream that forwards its packets through the 801.Q bridge forwarding function with the requirement to apply a specific egress treatment for that stream and not the egress treatment configured on the VLAN to which the stream belongs. In this example, other packets exiting the same VLAN but not identified to belong to that stream have the requirement of using the egress treatment configured on the VLAN. This use case can be supported with using both the primary and secondary treatment groups; i.e. packets belonging to the stream would be assigned 2 treatment groups (primary (stream) and secondary (VLAN)) whereby packets not belonging to the stream would be assigned only one treatment (secondary (VLAN)).

For each port that has its bit set in destination port bitmap, hardware determines the Egress Treatment table entry ID (or index) as follows:

1. Checks if the port has its bit set in the applicability port bitmap of the primary Egress Treatment group. If so then the index to access the Egress Treatment table is computed by adding an offset to the base index of the primary Egress Treatment group. That offset is derived from the applicability port bitmap as follows: starting from the lowest significant

bit of the bitmap, the first encountered bit set to 1, corresponds to offset 0, and so on. This continues till the port location in the bitmap is reached. If the port hasn't had its bit set in the applicability port bitmap of the primary Egress Treatment group, then step 2, below, is executed.

2. Checks if the port has its bit set in the applicability port bitmap of the secondary Egress Treatment group. If so then the index to access the Egress Treatment table is computed by adding an offset to the base index of the secondary Egress Treatment group. That offset is derived from the applicability port bitmap as follows: starting from the lowest significant bit of the bitmap, the first encountered bit set to 1, corresponds to offset 0, and so on. This continues till the port location in the bitmap is reached.

53.4.2.4.1.4.2 Egress frame modification

Egress frame modifications can be specified through the Egress Treatment table. Each Egress Treatment table entry stores a Frame Modification table entry ID. Bits 15-13 of the 32-bit Frame Modification table entry ID qualify whether the Frame Modification table entry ID is an index into the Frame Modification table or the frame modification actions are encoded in the Frame Modification table entry ID itself, and thus allowing the configuration of basic frame modification actions without needing to configure (or add) a Frame Modification table entry. For more details, see Frame Modification Table. When bits 15-13 of the Frame Modification table entry ID are all set to 000b, the modification table entry ID represents an index into the Frame Modification table. If the index refers to an entry that is not found in the Frame Modification table (entry has not been added to the table), the frame is discarded and counted against the port's Tx discard count register (PTXDCR) along with the setting of the Invalid Table Entry Discard Reason (ITEDR) flag to 1 in the port's Tx discard count reason register 0 (PTXDCRR0). When bits 15-13 of the Frame Modification table entry ID are set to a value other than 000b, the modification actions are encoded in the Entry ID itself. Note that bits 31-16 are reserved in the entry ID. For more details on the definition of the Frame Modification table entry ID, see Frame Modification Entry Definition.

Following are the egress packet modification actions/operations that can be specified through an entry in the frame modification table:

- Delete sequence generation tag (R-TAG, draft 2.0 R-TAG or HSR).
- Replacing destination MAC address (DMAC) and/or source MAC address (SMAC) with specified MAC address or copied from frame L2 header.
- Replacing VLAN tag (TPID, VID, PCP, and DEI).
- Deleting one VLAN tag.
- Adding one VLAN tag.
- Replacing up to 128 bytes of Ethernet payload.*
- Removing entire Ethernet payload and replacing with up to 2047 bytes of new payload.*
- Removing entire frame and replacing with a new frame of up to 2047 bytes.*

__ *Not valid for frames destined for pseudo link, or for any frames on an ASIL-B enabled device.

When adding or replacing a VLAN tag, TPID, PCP, DEI and VID are set as follows:

- The VID can be explicitly specified or copied from outer VLAN tag header of the frame.
- The PCP can be explicitly specified, copied from outer VLAN tag header or derived from a QoS to PCP mapping profile. PCP can also be derived from a specified PCP to PCP mapping profile.
- The DEI can be explicitly specified, copied from outer VLAN tag header or derived from a DR to DEI mapping.
- The TPID can be explicitly specified or copied from outer VLAN tag header of the frame.

For a particular port, if there is no Egress Treatment entry configured or if the frame modification entry ID configured in the Egress Treatment table entry is set to null, and the 802.1Q bridge forwarding function was executed, port level egress VLAN header modifications specified through Bridge port default VLAN register (BPDVR), are applied to the frame if the port is configured as a VLAN aware bridge (BPDVR[RXVAM=0]).

The egress frame modification actions along with other frame modification actions (i.e. FRER sequence generation modification and ingress frame modification) are passed along with the frame as metadata up to the Ethernet Tx I/F (or Tx MAC client) for actual frame modification (except for switch management port traffic with trap reason set to non-zero).

As for other frame modifications, the effective frame length change of the egress frame modification actions are reflected in the frame length metadata. Note that the egress frame modifications effective length change must be explicitly configured by software.

The updated frame length metadata is used by the egress queue scheduling and management function, which is executed prior to applying the frame modification actions to the frame, to determine the exact length of a frame transmitted on a link.

In general, if an egress frame modification action cannot be performed (e.g. a delete outer VLAN header action on a frame that has no VLAN header), this will result in a misconfiguration event. This misconfiguration event is handled according to the setting of the port's PFMCR register.

53.4.2.4.1.4.3 Frame Replication and Elimination for Reliability (FRER) sequence recovery

FRER recovery consists of recognizing packets with a sequence tag (R-TAG, draft 2.0 R-TAG or HSR tag), extracting the sequence number from that tag, and passing only one packet for each sequence number, rejecting all duplicates. This way, multiple copies of the same packet that have been sent along different paths for redundancy will result in the first arriving packet being passed, and all other copies removed.

Implementation-wise, the FRER recovery function in NETC is performed right before egress queuing in the switch, such that accepted packets are queued and rejected packets are discarded instead of being queued.

An Egress Treatment table entry (see Egress Treatment Table for more details) can specify the execution of FRER recovery function, and if so then the Egress Treatment table entry also provides an index into a Egress Sequence Recovery table (see Egress Sequence Recovery Table for more details).

Also note that cut-through forwarding must be disabled on any stream in which FRER recovery is enabled, because only frames that have passed FCS check must be passed to the recovery function. This is to prevent the possibility of a frame with bad FCS from being passed on by the recovery function.

The Egress Sequence Recovery table contains one entry per recovery function, and each entry contains a number of configuration elements, whose value is set when the table entries are configured, and operational elements, which are managed by the hardware and can be queried by software.

Summary of recovery function configuration element:

- **SQ_TAG (3b):** Indicates the sequence tag type (no tag, R-TAG, draft 2.0 R-TAG or HSR tag) expected by the recovery function. If a packet passing through the recovery function does not have a sequence tag, or the sequence tag type does not match SQ_TAG, then the packet is treated as a tag-less packet.
- **SQR_TNSQ (1b):** Take packets with no Sequence. Specifies the action taken (accept or reject) on packets passing through the recovery function (using the vector recovery algorithm) that are tagless, i.e. with no sequence tag or with a sequence tag type that does not match SQ_TAG. Used only if SQR_ALG = Vector recovery algorithm. If SQR_ALG = Match recovery, then tagless packets are always accepted.
- **SQR_TP (12b):** Recovery Timeout Period. Specifies the recovery timeout period in increments of 1.048576 milliseconds. A value of 0 disables recovery timeout. Each recovery function stores a timestamp of the last packet accepted (for sequence recovery) or the last packet seen (for individual recovery). Each time a packet passes through the recovery function, the time that has expired since the last packet timestamp (current time minus last packet time) is calculated, and if the expired time exceeds the configured recovery timeout period, then the recovery function is reset. The recovery timeout mechanism means that if a recovery function somehow gets out of step with its corresponding sequence generation function, then after the timeout period the next received packet will cause the recovery function to be reset, and data will again be passed.
- **SQR_ALG (1b): Recovery function algorithm:** Vector or Match. After reset, the vector recovery algorithm starts by always accepting the first received packet as valid. After the first packet has been accepted, all subsequent packets whose sequence number is in the window (last sequence number accepted +/- (recovery history length-1)) are checked against the recovery history to see if a previous packet with the same sequence number has already been accepted, and if so the current packet is rejected, otherwise it is accepted and the recovery history is updated to indicate the newly accepted sequence number. Packets whose sequence number falls outside of the recovery history window are rejected

and counted as rogue packets. The match recovery algorithm starts by always accepting the first received packet as valid, same as vector recovery. After the first packet has been accepted, all subsequent packets either match the last packet number accepted, and are therefore discarded, or do not, in which case they are accepted. The recovery history is not used by the match recovery algorithm.

- **SQR_TYPE: Recovery Function type:** Sequence or Individual. The recovery function type affects the way recovery timeout works. For a sequence recovery function, the stored packed timestamp is updated only for accepted packets, it is not updated for rejected packets. This means that a sequence recovery function that is out of step and is continually rejecting packets will be reset after the timeout period is reached, allowing it to resume sending packets. For an individual recovery function, the stored packed timestamp is updated for all packets. This means that an individual recovery function will not timeout even if all received packets are rejected, and can only timeout if packets stop arriving for a long time (timeout period or greater), and then start arriving again. The individual recovery function is intended to protect against a stuck transmitter that is sending the same packet over and over, by removing the repeating sequence numbered packets without timing out. Note that the implementation doesn't support performing individual recovery followed by sequence recovery.
- **SQR_HL (7b):** Recovery History Length. The history length used by the vector recovery algorithm is equal to SQR_HL + 1. This field is used only if SQR_ALG = Vector Recovery. The maximum supported history length is SoC specific and is limited by the width of the internal memory resource used to store the sequence recovery history. The maximum supported value of SQR_HL is indicated by FWDSQRCAPR[SQR_MAX_HL]. Any attempt to set a history length larger than the maximum supported will result in the history length being set to its maximum. The minimum valid history length for vector recovery is 2, and attempting to set a smaller history length of 1 (SQR_HL = 0) will result in a history length of 2.

Summary of recovery function operational data:

- SQR_NUM (16b): Holds the highest sequence number value received so far by the recovery function, or 0xFFFF after reset of the recovery function. The sequence number is an unsigned integer that rolls over from max to 0, therefore at the rollover boundary sequence number 0 is treated as larger than sequence number 0xFFFF.
- SQR_HISTORY (up to 128b): Recovery History. This is a bit vector that maintains a history of the sequence numbers of recently received packets, with each bit corresponding to sequence numbers from SQR_NUM down to SQR_NUM ? SQR_HL + 1. A 1 in a history bit indicates that a packet with that sequence number has been previously received, a 0 indicates that no packet with that sequence number has yet been received.
- TAKE_ANY (1b): Accept the next packet, regardless of its sequence number. This is set to 1 when the recovery function is reset, and cleared after the first packet is received. The recovery function always accepts the first packet after reset, which updates SQR_NUM and SQR_HISTORY. All subsequent packets are then accepted or rejected based on their sequence number and the recovery algorithm used.
- SQR_TS (12b): Recovery timestamp. This is the timestamp of the last packet accepted (for sequence recovery) or the last packet seen (for individual recovery).

In addition to the above listed configuration and operational elements, each recovery function maintains a number of statistics counters, which are described in Table 668.

Latent error detection provides a mechanism for detecting the failure of a path in a system using FRER to provide multiple redundant paths for a stream of packets. The receiver cannot detect such a failure since FRER is still providing valid packets from another path that has not failed, therefore detection of a path failure must be done at the point where the replicated packets are recovered and accepted or rejected.

The 802.1CB standard specifies that every instance of the sequence recovery function must also support a latent error detection function. But unlike the sequence recovery function, the latent error detection function is a periodic function driven by timers, not by packet arrivals.

Latent error detection operates on the assumption that, in a properly functioning Compound Stream employing n paths into the current system, there will be n - 1 packets discarded for every packet passed through the sequence recovery function (latent error detection is not used for individual recovery functions).

The latent error detection function is triggered periodically, and calculates the difference between the actual and expected number of rejected packets (expected = PassedPackets * (num_paths - 1)), and triggers an error if this difference exceeds a configured threshold.

In NETC the latent error detection function is performed in software. The Egress Sequence Recovery table query command is used to periodically read the PassedPackets and DiscardedPackets counters for each sequence recovery function that is in use, and determine if a latent error needs to be signaled.

53.4.2.4.1.5 Egress Queuing and Traffic Management

Frames received from the Replication and Egress Packet Processing block are queued internally onto Egress Traffic Management (ETM) class queues. ETM class queues are queues of frame descriptors and provide the main buffering for frames waiting to be scheduled for transmission to the port. For each outgoing switch port, a total of 8 ETM class queues is provided. Within a port, class queues are numbered from zero to N-1, where N is the number of class queues (currently at 8). There is a one-to-one mapping between class queues and traffic classes, class queue 0 maps to traffic class 0, class queue 1 to traffic class 1, and so on. Frames are placed in the appropriate queue based on the internal queuing priority (IPV) and can be dropped based on the internal drop priority (DR). An IPV to class queue number mapping is specified in the PIPV2QMR0 register. Class queue state, that is whether a class queue is empty or contains one or more frames, is indicated in PQOR register.

Egress traffic management processing involves the following mechanisms (details of this are given in the following sections).

- **ETM class queue scheduling** - Determines which frame to send next when an outgoing link is free. Used primarily to manage the allocation of bandwidth among ETM class queues.
- **ETM class queue congestion management** - Manages the queue sizes by primarily deciding when to start dropping frames. Multi priority (color-aware) tail drop based on instantaneous queue depth is supported.

A switch port can be configured to support IEEE 802.1Qbu preemption (if capability is provided). Preemption is enabled on a per port basis by setting MAC_MERGE_MMCSR[ME] to 1 or 2. The ability to designate one or more traffic classes (or class queues) as express (and the remaining traffic classes as preemptable) means that, when an express frame is ready for transmission and a preemptable frame is already in the process of being transmitted, the delay before the express frame transmission starts is at worst the sum of the minimum final (64B) and non final (MAC_MERGE_MMCSR[RAFS]) fragment sizes minus four octet times and will often be 64 octet times or less. For more details, see Preemption.

53.4.2.4.1.5.1 ETM Class Queue Scheduling

Following scheduling algorithms are supported:

- **IEEE 802.1Qav** - Credit-based shaping (CBS), also referred to as Forwarding and Queueing Enhancements for Time Sensitive Streams (FQTSS). A maximum of 7 traffic classes is supported with credit-based shaping (CBS) enabled. CBS should be enabled on numerically the highest traffic class(es). CBS requires that IEEE 802.1Qbv be enabled, with or without a gate control list configured.
- **IEEE 802.1Qbv** - Enhancements for Scheduled Traffic (EST), which provides time "slots" for specific classes of traffic in a manner similar to TDM. EST is also referred in the document as to 'time gate scheduling'.
- **Strict priority and fair queuing** - Configurable combinations of strict priority and fair queuing are supported. A proprietary algorithm, Weighted Bandwidth Fair Scheduling (WBFS) is used to implement the weighted fair queuing function. WBFS is not supported if EST is enabled or if preemption is enabled, that is only strict priority can be configured if EST and/or preemption are enabled.

A physical port supporting IEEE 802.1Qbu frame preemption contains two MACs; that is, MAC 0 and MAC 1. MAC 0 is called the express MAC and is used to handle the express traffic, whereby MAC 1 is called the preemptable MAC and is used to handle the preemptive traffic. When preemption is enabled, the express traffic classes tier and the preemptable traffic classes tier are scheduled independently based on the next transmission opportunity from their respective link tier. Furthermore, in-flight data is tracked and limited independently for both MACs using the same comfort threshold configured in the IERB register LaTXBCCTR.

When the port is attached to a pseudo MAC, the internal flow control signals from the pseudo link are connected directly to the scheduler. When asserted, the scheduler enters a pause state where it stops scheduling frames for all traffic classes. When de-asserted, the scheduler resumes to normal transmission of frames.

In order to support 802.1Qbv, the scheduling structure supports the concept of ‘transmission gates’. A transmission gate is a construct that connects or disconnects the scheduler (or the transmission selection function) from a traffic class (or queue), allowing or preventing the scheduler from selecting frames from that traffic class. A transmission gate has two states: open and closed. Only frames in a traffic class with an open transmission gate are eligible for scheduling.

When 802.1Qbv is disabled (or enabled but no gate control list is operational), the states of the transmission gates are set according to the state values configured in PDGSR[DGS_TC*n*], where '*n*' is the traffic class number. The power-on-reset settings for PDGSR[DGS_TC*n*], are gate open. The PDGSR[DGS_TC*n*] settings can be written at any time and take effect immediately if no gate control list is operational or if 802.1Qbv is disabled. When a traffic class transmission gate is set to the 'closed' state, frames are still queued onto its ETM class queue unless congested, but frames remain in the class queue as long as the transmission gate is closed.

The figure below shows ETM queuing structure, and with 2 traffic classes configured with CBS.

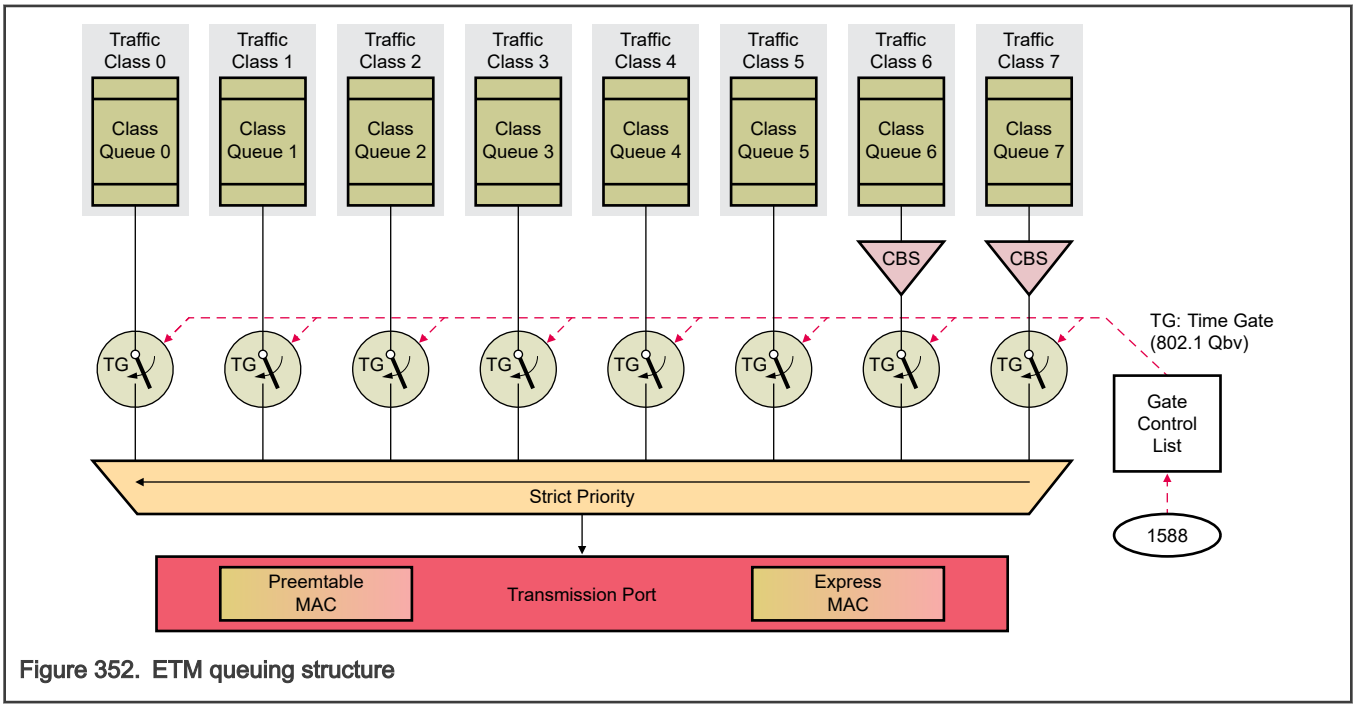


Figure 352. ETM queuing structure

53.4.2.4.1.5.1.1 Credit-Based Shaper - IEEE 802.1Qav

A credit-based shaper is implemented, as per IEEE 802.1Qav (Forwarding and Queueing Enhancements for Time Sensitive Streams (FQTSS)), which is needed for applications requiring bandwidth allocation. The credit-based shaper acts independently, per traffic class, to control the bandwidth distribution between normal traffic and time-sensitive traffic with respect to the total link bandwidth available. In order for the credit-based shaper algorithm to operate as intended, it is necessary for all traffic classes that support the algorithm to be numerically higher than any traffic classes that support the strict priority algorithm.

The Audio Video Bridging (AVB) profile specification (IEEE 802.1BA) states that the sum of AVB Class A and Class B traffic may not exceed 75% of the port transmit rate. The following example uses 70% for AVB frames and 30% for non-AVB frames. AVB class A and B uses one credit-based shaper each, while non AVB classes do not use any credit-based shapers.

Table 527. Bandwidth allocation example for AVB

Class	Percent Bandwidth (total 100%)
AVB Class A	50%
AVB Class B	20%
Non-AVB classes	30%

The following bandwidth availability parameters exist for each port, and for each traffic class that supports the credit-based shaper algorithm.

- *idleSlope*

  The actual bandwidth that is currently reserved for use by the queue(s) associated with a traffic class. Once the frame transmission is complete, *credit* increases back to zero at the rate of *idleSlope*, at which point, a further frame can be selected for transmission. Credits also increase at the rate of idleSlope up to *hiCredit* when there are frames available to send waiting for conflicting traffic to complete. If there is an operational gate control list that is active, then credits are accumulated only while the gate for the credit-based shaper traffic class, is open.

- *sendSlope*

  Credit consumption (per byte) while traffic class is transmitting is calculated as: sendSlope = idleSlope - portTxRate. If a class had 100% bandwidth, sendSlope would be 0, allowing for continuous transfers.

- *hiCredit*

  hiCredit is the maximum number of credits that can be accumulated by the credit-based shaper. The hiCredit is dependent on the maximum interference time for the traffic class.

  For the AVB Class A (highest priority), hiCredit is calculated as follows:

  hiCredit = maxSizedFrame * (idleSlope/portTxRate)

  For the AVB Class B (second highest priority), the maximum interference size is equal to the maximum burst size for the AVB Class A plus the maximum interference size for SR class A (which is equal to the maximum frame size for the port). Maximum burst size for the AVB Class A is calculated as follows:

  maxAVBClassABurst = (RA * M0)/(R0 - RA) + MA (equation (L-16) in IEEE Std 802.1Q-2018), where

  — RA: idleSlope for AVB Class A

  — R0: port transmit rate (specified in PCR[PSPEED])

  — M0: maximum sized frame for the port

  — MA: maximum sized frame for AVB Class A

  hiCredit = (maxAVBClassABurst + maxSizedFrame) * (idleSlope/portTxRate)

  The maximum sized frame for a traffic class is determines by PTCaTMSDUR[MAXSDU] (where a is the traffic class number)

- *loCredit*

  loCredit is the minimum number of credits that can be accumulated by the credit-based shaper. The loCredit is computed by hardware as follows:

  loCredit = (idleSlope - portTxRate) * maxSizedFrame/portTxRate

The figure below shows the different stages of credit changes for a traffic class as it transmits, waits or is idle.

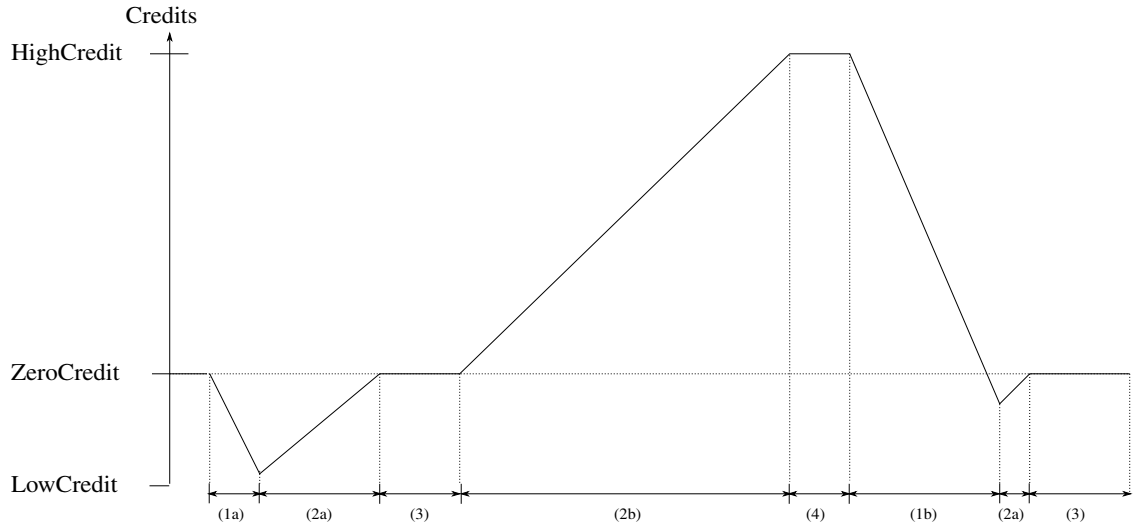


Figure 353. Stages of credit changes for shaper

To program the credit-based shaper for a class, follow these steps:

1. Enable time gate scheduling for the port by setting PTGSCR[TGE] to 1. An administrative gate control list does not need to be configured, for the credit-based shaper to operate. This also requires the timer function to be configured and enabled. Either the default nanosecond timer or the 1588 timer can be used to generate the required nanosecond resolution time.
2. Enable credit-based shaper for the port traffic class *a* by setting PTC*a*CBSR0[CBSE]=1.
3. Determine *hiCredit* from the formula described above and set PTC*a*CBSR1[HI_CREDIT].
4. Set the allocated bandwidth for the traffic class credit-based shaper which ranges from 1-100 (%), in register PTC*a*CBSR0[BW]. The total bandwidth for all enabled shapers must not exceed 100% or the allotted bandwidth.
5. Set PTXSDUOR register with the appropriate per frame overhead; that is number of bytes to be added to the actual length of each frame, to determine the Physical Layer PDU (PPDU) frame length on the link.

53.4.2.4.1.5.1.2 Time Gate Scheduling - IEEE 802.1Qbv

IEEE 802.1Qbv, also referred in this document as time gate scheduling, introduces a time based transmission mechanism operating on a global (offline) configured periodic schedule called the gate control list. IEEE 802.1Qbv amendment has been incorporated in the IEEE standard 802.1Q 2018; the mechanism itself is referred in the standard as "enhancements for scheduled traffic."

The mechanism is shown in the diagram below.

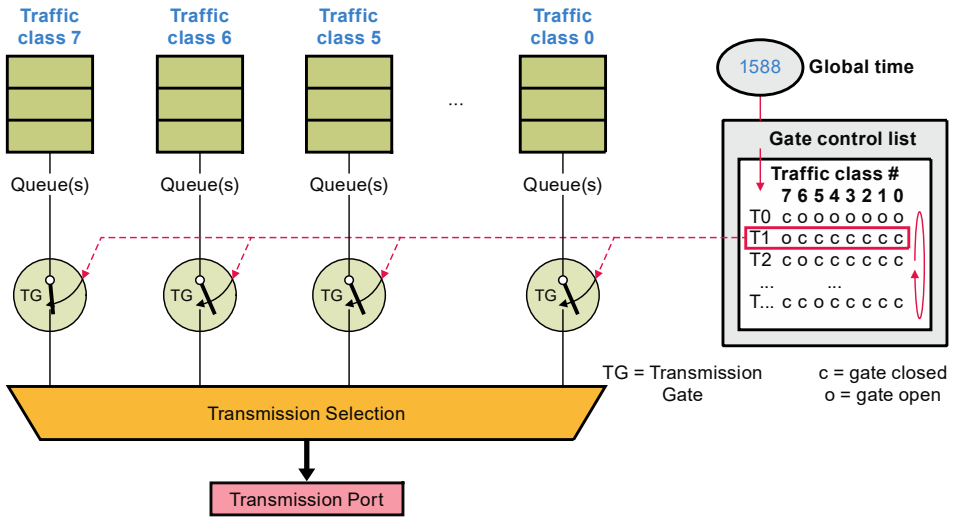


Figure 354. Time Gate Scheduling - IEEE 802.1Qbv

Each traffic class has a gate, which controls if the traffic class is open or closed. Only frames in traffic classes with open gate are eligible for transmission. For the switch, each traffic class has a single queue, and for ENETC, it is possible that a traffic class may have more than one queue (or Tx BD ring). In this case, all queues (or Tx BD rings) in a traffic class have/shared the same gate state.

The gate state in each traffic class is controlled by the gate control list. A gate control list consists of an ordered list of gate control entries, where each entry specifies an 8 bit gate state vector where each bit (one per gate), indicates whether the gate is open or closed. The gate control entry also specifies the time duration for the states.

The gate control lists are typically generated offline, and then timely configured on each network device. To support no-downtime gate control list configuration changes, the standard requires that a network device supports two sets of gate control lists:

- **Operational gate control list** – Represents the currently active gate control list.
- **Administrative gate control list** - Provides a means of configuring a new gate control list prior to its installation in a running system.

The hardware executes the (operational) gate control list cyclically based on a global time. The hardware starts executing the gate control list from the head, and then goes through the gate control entries as time goes on, and after the cycle time is passed, the hardware jumps back to the head of the gate control list.

The transmission selection process (or the scheduler) which executes at the port level, may only select frames from traffic classes that have their gates in the open state. The transmission selection uses a strict priority algorithm to select the next frame for transmission by querying/checking the open traffic classes in descending order from the highest priority traffic class to the lowest priority traffic class. If the open traffic class being queried has one or more frames waiting to be transmitted, then the scheduler determines if the frame at the head of the queue can be transmitted entirely before its traffic class gate closes. If it is determined that there is insufficient time available to transmit the entirety of that frame (transmission of the frame does not fit in the current open window), that frame will remain queued, and an attempt to schedule it will be made in the next open window associated with the frame traffic class. The transmission selection process then moves to the next lowest priority traffic class that has its gate open.

To set up time gate scheduling, follow these steps:

1. The 1588 timer must be initialized and configured before enabling time gate scheduling. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer is in normal operation, any change to the 1588 timer configuration (for example, in TMROFF_H/L), except for TMR_ADD updates, requires that time gate scheduling be disabled ((PTGSCR[TGE]=0). The state of each gate while time gate scheduling is disabled, can be controlled via register PDGSR[DGS_TCn], where n is the traffic class number.
2. Enable time gate scheduling for the port by setting PTGSCR[TGE] to 1.

3. The initial state of each gate, when no gate control list is operational, is determined by the setting in PDGSR[DGS_TCn], where n is the traffic class number. The power-on-reset setting for PDGSR[DGS_TCn] is gate open. The initial state of each gate can be changed by configuring PDGSR[DGS_TCn]. Note that the PDGSR[DGS_TCn] settings can be written at any time and take effect immediately if no gate control list is operational or if time gate scheduling is disabled. Thus, it may be required to re-configure PDGSR[DGS_TCn], before disabling time gate scheduling, if the state of the gates when time gate scheduling is disabled, differ from their initial state when time gate scheduling is enabled.
4. Set the maximum frame size for each traffic class in PTC*a*TMSDUR[MAXSDU]. Frames that exceed the limit are discarded.
5. Configure the (administrative) gate control list through the use of the Time Gate Scheduling table; for more details, see Time Gate Scheduling Table for the ENETC and Time Gate Scheduling Table for the switch. The Time Gate Scheduling table contains time gate scheduling configuration and operational information for a given NETC function. There is one Time Gate Scheduling table for each ENETC instance and one Time Gate Scheduling table for the entire switch. For the switch, there is one entry for each switch port, whereby for an ENETC instance, there is a single entry for the entire ENETC instance. Each entry includes both an administrative gate control list and an operational gate control list, along with the related configuration and status parameters. The gate control list includes the following parameters:
   - Size of the gate control list.
   - Base time (start time) of the schedule. Base time is expressed in units of nanoseconds.
   - Cycle time of gate control list. Cycle time is expressed in units of nanoseconds. The cycle time cannot be shorter than the sum of all gate control list entry durations.
   - Cycle extension time of gate control list. Cycle extension time is expressed in units of nanoseconds.
   - Gate control list.
6. When hardware processes a Time Gate Scheduling table command to configure a new (administrative) gate control list, it calculates the time at which the administrative gate control list is to become active (config change time), based upon the values of the base time and cycle time of the administrative gate control list. When the config change time of the administrative gate control list is reached, the hardware installs the gate control list (the gate control list becomes operational) and starts the execution of the gate control list. This function is performed in advance by having its time reference set to the 1588 time + E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET], to adjust for latency encountered in the transmit processing pipeline. Register PTGSATOR[ADV_TIME_OFFSET] is used to specify the latency across the MAC plus if needed, delays outside of NETC (for example, PHY delay). Register E*a*TGSLR[MIN_LOOKAHEAD] is used only for ENETC, to specify the latency encountered for scheduling, dequeuing, and loading frames from the host memory to NETC internal memory. Register S0TGSLR[MIN_LOOKAHEAD] is used only for the switch, to specify the latency encountered for scheduling and dequeuing frames. Note that E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] are IERB registers, and thus can only be setup at pre-boot initialization time.
7. Similarly, time gate scheduling of all frames is also done in the future to adjust for latency encountered in the transmit processing pipeline. As in the previous step, this is achieved by advancing the 1588 time reference by the amount of time (*LookAheadTime*) configured in E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET].
8. Set PTXSDUOR register with the appropriate per frame overhead; that is number of bytes to be added to the actual length of each frame, to determine the PPDU (Physical Layer PDU) frame length on the link.

Time Gate Scheduling Operation

The setting of the transmission gate state for the gate associated with each of the port's traffic classes and the execution of transmission selection function are performed in advance to mitigate against any long latency (e.g. DMA latency in the case of ENETC) and latency variation encountered in the transmit processing pipeline. This in turn leads to better timing accuracy and reduction in jitter. The time used to perform these function, is set to the 1588 time + E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]. Since the transmission selection function is performed in advance, it no longer relies on the MAC/link being idle for the indication that there is a transmission opportunity to transmit a frame. Instead the link is modeled by the transmission selection function to determine when frames are selected for transmission. Once the transmit transmission function selects a frame for transmission, next transmission opportunity provided by the link model, will occur after the time interval corresponding to the transmission time of the current selected frame at the rate configured in PCR[SPEED], with physical frame

overhead (IFG, preamble, SFD) accounted. The frame overhead is specified in PTXSDUOR. Time remaining for the transmission of a frame is tracked on every clock cycle. Note that the modeling of link also models the preemption function if enabled (such as stopping transmission of preemptible frame with added overhead of non-final fragments (mCRC, IFG, preamble, SFD) being accounted for, in the link model).

Each frame selected for transmission is tagged with a timestamp corresponding to the advance time that the time gate scheduler is operating on (1588 time + E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]). The timestamp of a frame determines its departure time, that is the time at which the frame should be transmitted to the link. Then at the point where frames are transferred to the MAC, a timestamp-based shaper is implemented where the frames are released only when their transmission time (timestamp) is reached. The timestamp-based shaper function can also be performed in advance by having the time used to perform this function advanced by the amount of time specified in PTGSATOR[ADV_TIME_OFFSET], to adjust for the latency encountered through the Ethernet interface logic (MAC + physical layer latency).

Time gate scheduling proceeds as follows when there is transmission opportunity on the link to transmit a frame. It first selects the highest priority traffic class for which there is a frame waiting to be transmitted at the head of the traffic class. It then determines if that frame can be transmitted. A frame for a given traffic class can only be transmitted if the transmission gate associated with that traffic class is open and there is sufficient time for the entire frame (length of the frame is known) to be transmitted before the next gate-closed operation associated with that gate is executed. If both conditions are not met, then the next highest priority traffic class for which there is a frame waiting to be transmitted at the head of the traffic class, is selected to determine if that frame can be transmitted based on the criteria described above.

To determine if a frame can be transmitted, the scheduler first checks if there is sufficient time left in the current gate control entry interval, to transmit the entirety of the frame. If there is insufficient time, then the pre-processed gate control list data is accessed to extract how many bytes (up to the maximum SDU length) that can be transmitted in the next gate operations for this traffic class. This gate control list pre-processing, which is performed when the gate control list is configured, consists of performing the following computation for each entry in the gate control list.

- For each traffic class that has its gate open, the next gate operations in the gate control list (next gate control entries) are inspected to accumulate the number of bytes that can be transmitted in each gate control entry, until the accumulated number of bytes reaches the configured maximum SDU frame size, or a gate-closed operation is found.
- If the end of the gate control list is reached and the gate control cycle time is longer than the total execution time of the sequence of gate operations, the last gate control entry is extended to the end of the gate control list cycle. Once the end of the gate control list cycle is reached (no gate-closed operation found), and the accumulated the number of bytes that can be transmitted has not reached the size of a maximum SDU frame, inspection continues from the beginning of the same gate control list.
- This accumulated number of bytes that can be transmitted, is then stored in the gate control list entry, to be used later by the scheduler to determine for a given traffic class, how many bytes that can be transmitted in the next gate operations. This pre-processed data avoids having the scheduler at run-time, examining the next entries in the gate control list.

If it is determined that there is insufficient time available to transmit the entirety of that frame (transmission of the frame does not fit in the current open window), that frame will remain queued, and an attempt to schedule it will be made in the next open window associated with the frame traffic class.

In the case where the existing gate control list cycle is extended so that it ends at the base time of the new gate control list to be installed, any inspection from previous gate control entries will not continue into this extension time interval. Frame scheduling during the extension time internal is permitted with the gate states set to the values of the last gate control entry of the existing gate control list.

If there is a new gate control list scheduled to be installed in the next gate control list cycle, the scheduler cannot take into consideration the new configuration as the scheduler is using pre-processed gate control list data to determine how many bytes that can be transmitted in the next gate operations. Furthermore, any of the next gate operations accumulated byte fields in the current operational gate control list, that resulted from inspecting back to the beginning of the gate control list, are no longer accurate. As a result, all of the next gate operations accumulated byte fields in the last cycle are overwritten to 0.

Note that once the time gate scheduler has cycled through the entire operational gate control list and has detected a condition where a frame at the head of a traffic class will never be able to be transmitted, that frame will be discarded in the case of switch. These frame discards are counted against the port's Tx discard count register (PTXDCR) along with the setting of the Time Gate Scheduling Frame Too Large Discard Reason (TGSFTLDR) flag to 1 in the port's Tx discard count reason register 0 (PTXDCRR0).

In the case of ENETC, the frame is not transmitted and the STATUS field of the transmit buffer descriptor is updated with the error code 0x080 (frame too large for time gating window). The conditions where a frame at the head of a traffic class will never be able to be transmitted are:

- A frame at the head of a traffic class, although shorter than the value configured in PTC*a*TMSDUR[MAXSDU] (where *a* is the traffic class number), cannot be transmitted because there is no time interval between any gate-open event and a subsequent gate-close event, sufficient, long enough to transmit the entirety of the frame.
- There are no gate-open events for that traffic class in the entire operational gate control list; its gate is closed in every entry of the operational gate control list.

In the absence of any gate control list configuration, the time gate scheduler if enabled, still executes the transmission selection function ahead of time based on the setting of E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD], E*a*TGSLR[ZERO_LOOKAHEAD] and PTGSATOR[ADV_TIME_OFFSET].

Gate Control List Installation

The time when a new gate control list takes affect is calculated based on the rules described below (where *LookAheadTime* = E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET])

If the operational gate control list is not active, PTGAGLSR[TG]=0, the time (i.e. *ConfigChangeTime*) at which the next gate control list configuration change is scheduled to occur is determined as follows (where *CurrentTime* is at the 1588 time reference):

- if *AdminBaseTime* >= *CurrentTime* + *LookAheadTime*, then *ConfigChangeTime* = *AdminBaseTime*
- if *AdminBaseTime* < *CurrentTime* + *LookAheadTime*, then *ConfigChangeTime* = (*AdminBaseTime* + N**AdminCycleTime*) where N is the smallest integer for which the relation (*ConfigChangeTime* >= *CurrentTime* + *LookAheadTime*) would be TRUE.

If the operational gate control list is active, PTGAGLSR[TG]=1, the time (i.e. *ConfigChangeTime*) at which the next gate control list configuration change is scheduled to occur is determined as follows (where the end of the current operational cycle (*EndCurrentCycle*), is the absolute end time of the currently operational cycle without the cycle extension time, at the time reference of 1588 time + EaTGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]):

- if *AdminBaseTime* < *EndCurrentCycle* + *OperCycleTime*, then *ConfigChangeTime* = *AdminBaseTime* + N**AdminCycleTime*) where N is the smallest integer for which the relation (*ConfigChangeTime* >= *EndCurrentCycle* + *OperCycleTime*) would be TRUE.
- if *AdminBaseTime* >= *EndCurrentCycle* + *OperCycleTime*, then *ConfigChangeTime* = *AdminBaseTime*

The *ConfigChangeTime* can be retrieved by issuing a query command against the Time Gate Scheduling table entry associated with the port.

When the administrative gate control is about to be installed, and operational gate control list is not active, PTGAGLSR[TG]=0, then *CycleStartTime* is set to *ConfigChangeTime*. If the operational gate control list is active, PTGAGLSR[TG]=1, *CycleStartTime*, is determined according to the following rules:

- If *ConfigChangeTime* > *EndCurrentCycle* + *OperCycleTimeExtension*, then *CycleStartTime* = *EndCurrentCycle*
- If *ConfigChangeTime* <= *EndCurrentCycle* + *OperCycleTimeExtension*, then *CycleStartTime* = *ConfigChangeTime*

If a new gate control list configuration is installed and starts to execute in the current cycle, the current cycle is truncated in order to start the first new cycle and any inspection in the old cycle would not continue into the new cycle in order to start the first new cycle. This could result for the time interval of the last entry of the old gate control list to be very short. This is turn could cause overrunning the next gate-close event, or cause not being able to transmit a frame because the time interval was shortened, and that frame could then get discarded if it had been waiting to be transmitted for a full cycle interval.

Time Gate Scheduling Example

Time gating may be better understood with an example gate control list showing how scheduling will be performed. The example below, Table 528, shows a non-overlapping open gate case, where two traffic classes are never allowed to transmit at the same time. The interval normally defines a multiple of time ticks, in this example assume that the interval and number of bytes allowed to be transferred are the same. The number of bytes per interval would normally be calculated based on the interval length, transmit rate, and time tick granularity.

The total length of the gate control list is 3000B when adding up each individual time interval. The (admin) cycle time provided must be 3000B or greater. If the cycle time is 3000B, then when entry 15 completes, entry 0 would start immediately. Additional time added to the cycle time will extend the entry 15 interval. Assume here that the cycle time is set to 3100.

Table 528. Example: Non-Overlapping Gate Control List

Entry	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Interval (bytes)	100	200	100	300	100	200	100	300	500	100	200	200	300	100	100	100
TC 0	0	0	**1**	**1**	0	0	0	0	**1**	**1**	0	0	0	0	0	0
TC 3	0	0	0	0	0	**1**	0	0	0	0	0	**1**	0	0	0	0
TC 7	**1**	**1**	0	0	0	0	**1**	**1**	0	0	0	0	**1**	**1**	**1**	**1**

In this example, traffic class 7 gate has an open window (gate in the open state) of 1000 bytes starting at the beginning of the 4th last entry and ending at the end of the 2nd entry of the next cycle (if no new gate control list is to be installed).

Lets look at a first scheduling example where two frames are about to be scheduled. Both frames are available at least *LookAheadTime* before they are required to be transmitted in the open time gate window (Interval 1). This time is needed for the device to fetch the frame from memory and prepare it for transmission.

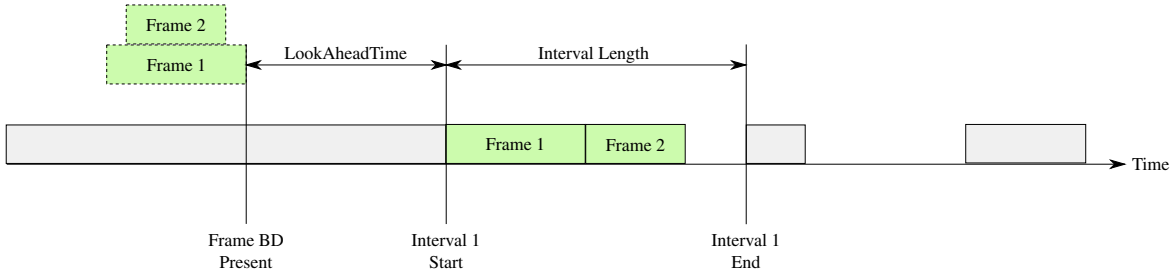


Figure 355. Scheduling of Frames

In this next example, consider the case where a best-effort first frame arrives "late" for scheduling within the next open gate window. Since the scheduler is always looking at LookAheadTime in the future it will calculate the amount of time left in a window at LookAheadTime from the current time to the end of the window. If the frame can still fit it will be scheduled for that window.

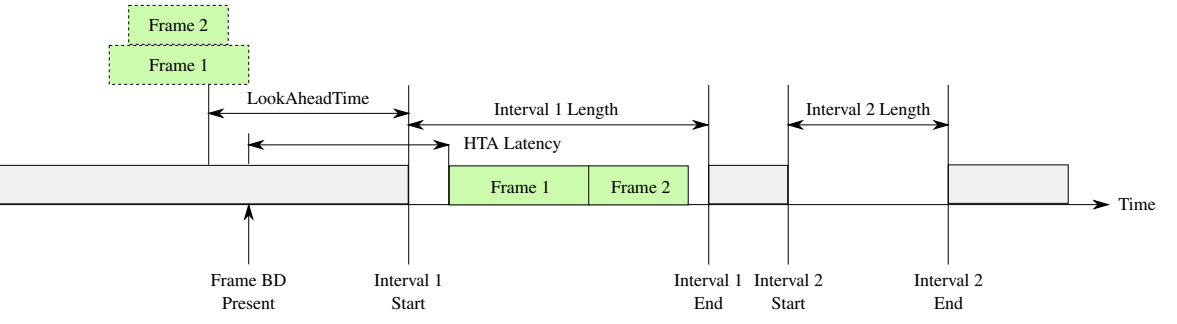


Figure 356. Scheduling of Frames - Late Arrival

Time Gate Scheduling in Combination with Preemption

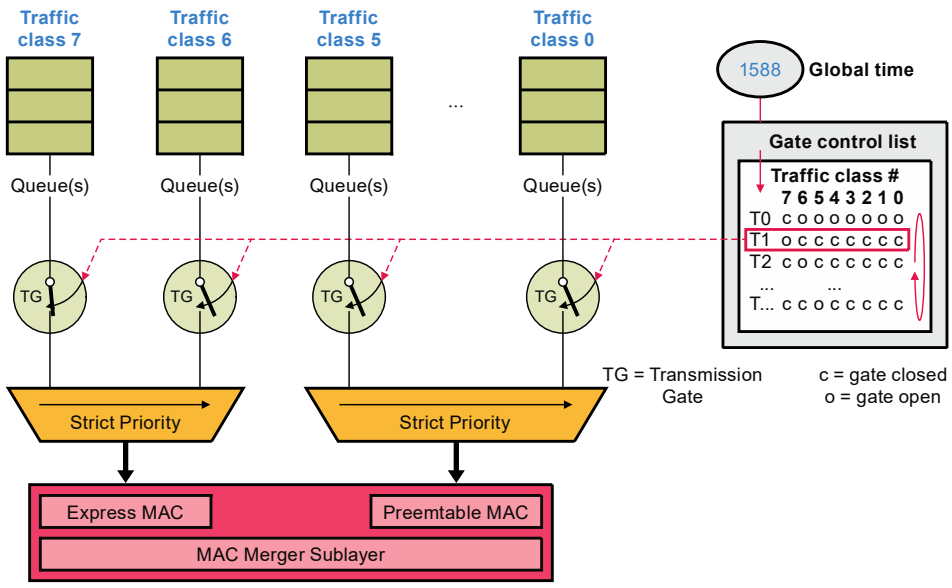


Figure 357. Time gate scheduling in combination with frame preemption

When time gate scheduling is used in combination with frame preemption, the "Hold and Release" mechanism can be used to protect the express traffic class window from interference from preemptable traffic, while at the same time minimizing the impact of the protected window on the amount of bandwidth that is available to preemptable traffic. The "Set-And-Hold-MAC" and "Set-And-Release-MAC" gate operations are used to specify the start and the end of a protected window.

When the time gate scheduler is processing a "Set-And-Hold-MAC" gate operation, asserting the Hold signal to the MAC merge sublayer is not always performed. Specifically, at the point of time where an Hold signal needs to be asserted (start of the hold advance interval), the time gate scheduler will not assert an Hold signal to the MAC merge sublayer under the following conditions:

- The preemptible frame transmission will complete its transmission before the Set-And-Hold-MAC time slot begins.
- There is no preemptible frame transmission.

The time gate scheduler will not schedule a preemptible frame during the Set-And-Hold-MAC time slot, regardless of whether or not an Hold signal was asserted. Furthermore, the time gate scheduler will not schedule a preemptible frame during the hold advance interval, if the preemptible frame transmission would continue into the Set-And-Hold-MAC time slot.

Delaying the scheduling of preemptable traffic classes to the start of the "Set-And-Release-MAC" time slot, has the benefit of selecting a newly arrived higher-priority preemptible frame, instead of the lower priority preemptible frames that were hold off by the scheduler prior and during the "Set-And-Hold-MAC" time slot. Note that once the time gate scheduling decides to not schedule a preemptible frame, any subsequent preemptible frames received will not be scheduled till the "Set-And-Release-MAC" time slot begins even if they there were of higher-priority.

In all other cases, the time gate scheduler will assert the Hold signal, in advance to the Set-And-Hold-MAC time slot by the time interval corresponding to the transmission of the worst case (longest) fragment length that cannot be preempted plus associated physical layer overheads. IEEE Std 802.3-2018 Clause 99 defines a minimum final fragment size of 64 bytes and allows the minimum non-final fragment size to be 64, 128, 192, or 256 bytes. For example, with a non-final fragment size of 64 bytes, a fragment that is 123 or fewer bytes in length cannot be preempted, because the non-final fragment would be less than 64 bytes (60 bytes + 4 bytes for mCRC). The minimum non-final fragment size is configured in MAC_MERGE_MMCSR[RAFS].

Note that time gate scheduler operates at the time reference of 1588 time + E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET] and assumes that there is no delay from asserting the Hold/Release signal to the MAC merge

sublayer and the MAC merge sublayer ceasing/resuming to transmit any preemptible frame. Since it operates in advance, instead of immediately asserting the Hold/Release signal, an Hold/Release message is generated, and tagged with a timestamp set to the advance time (1588 time + E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]), at which the Hold/Release signal must be asserted assuming no internal pipeline delay from the point where frames are transferred to the MAC to the point where frame preemption occurs. Then at the point where frames are transferred to the MAC, a timestamp-based shaper is implemented where the Hold/Release messages are released, which trigger immediate assertion of the Hold/Release signal to the MAC merge sublayer when their time (timestamp) is reached. This timestamp-based shaper function is performed in advance by the amount of time specified in PTGSHCR [HOLD_SKEW] (that is, 1588 time + PTGSHCR [HOLD_SKEW]), to adjust for the Hold/Release latency encountered through the Ethernet interface logic (Hold Release delay to the MAC merge sublayer + any external delay (for example, PHY delay)).

Take for example, where MAC_MERGE_MMCSR[RAFS] is set to 64 bytes, meaning that the minimum fragment size that can be preempted is 124 bytes. If the time gate schedule determines that the Hold must be asserted, the time for asserting the hold would be set to 163 bytes transmission time prior to the start of the Set-And-Hold-MAC time slot; 20 bytes (IFG, preamble, SFD) + 123 bytes + 20 bytes (IFG, preamble, SFD).

An increased IFG length and any other overhead increases in frame transmission time are accounted for by setting the PTXSDUOR register appropriately.

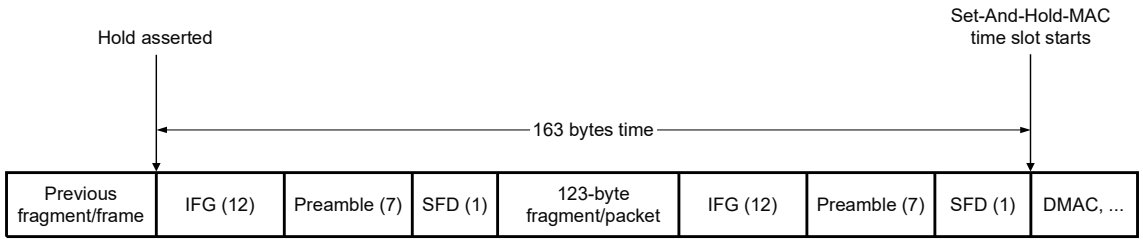


Figure 358. Hold Assertion Timing

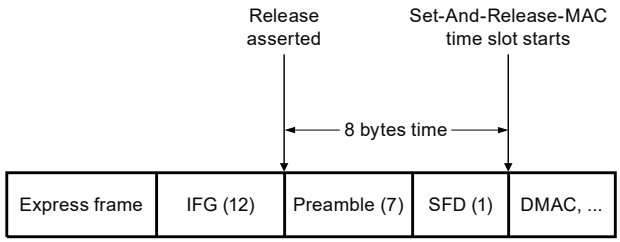


Figure 359. Release Assertion Timing

In the case where the "Hold and Release" mechanism is not used to protect the express traffic class window, and a preemptible frame has been scheduled and its transmission cannot be stopped by the time the express traffic gate opens, the scheduling of the express traffic class is delayed up to the point where the transmission of the preemptible frame has ceased.

If the schedule that applies to the preemptible traffic classes is constructed to disallow preemptible traffic to be transmitted at certain time (that is, there are "gate close" events for the preemptible traffic classes), then there is no way of telling in advance how many times a given frame will be preempted before its transmission is complete, and thus this configuration may result in overrunning the next gate-close event for the preemptible traffic classes concerned.

Note that the when gate control list preprocessing is inspecting the next gate operations in the gate control list (next gate control entries) for a preemptible traffic class, and encounters a gate control list entry that contains an express traffic class with its gate open, the accumulated number of bytes that can be transmitted is set to the configured maximum SDU frame size for the preemptible traffic class being inspected.

Time Gate Scheduling in Combination with Half Duplex Transmission

When time gate scheduling (802.1Qbv) is used in combination with half duplex transmission, the time gate scheduler compensates for deferred transmissions. Accurate transmission scheduling (that is, transmitting only within open gate time intervals) will be recovered for future frames in the event that one or more scheduled frame transmissions are deferred. These deferred frame(s) may violate their classes' open gate times as a result of the deferral, but it will not create a frame scheduling backlog that results in persistent gate violations once the deferred transmission has completed.

The time gate scheduler does not compensate for transmission back off delays due to collisions. A back off delay that follows a collision event may result in gate violations by the re-transmitted frames. In addition, collision back off delays will result in persistently late frame transmissions (potential gate violations) so long as there are transmit frames available for selection by the time gate scheduler; that is as long as transmit traffic is continuously flowing.

The time gate scheduler will recover accurate transmission scheduling during any idle time periods when there are no transmit frames available for selection. Once the sum of the idle time periods equals the total collision delays encountered, accurate transmission scheduling is reestablished.

53.4.2.4.1.5.1.3 Transmit Scheduling Timing

The figure below shows the moments in time related to current time a frame has been scheduled for IEEE 802.1 Qbv. This is to emphasize the role of S0TGSLR[MIN_LOOKAHEAD], PTGSATOR[ADV_TIME_OFFSET] and PTGSHCR[HOLD_SKEW] values.

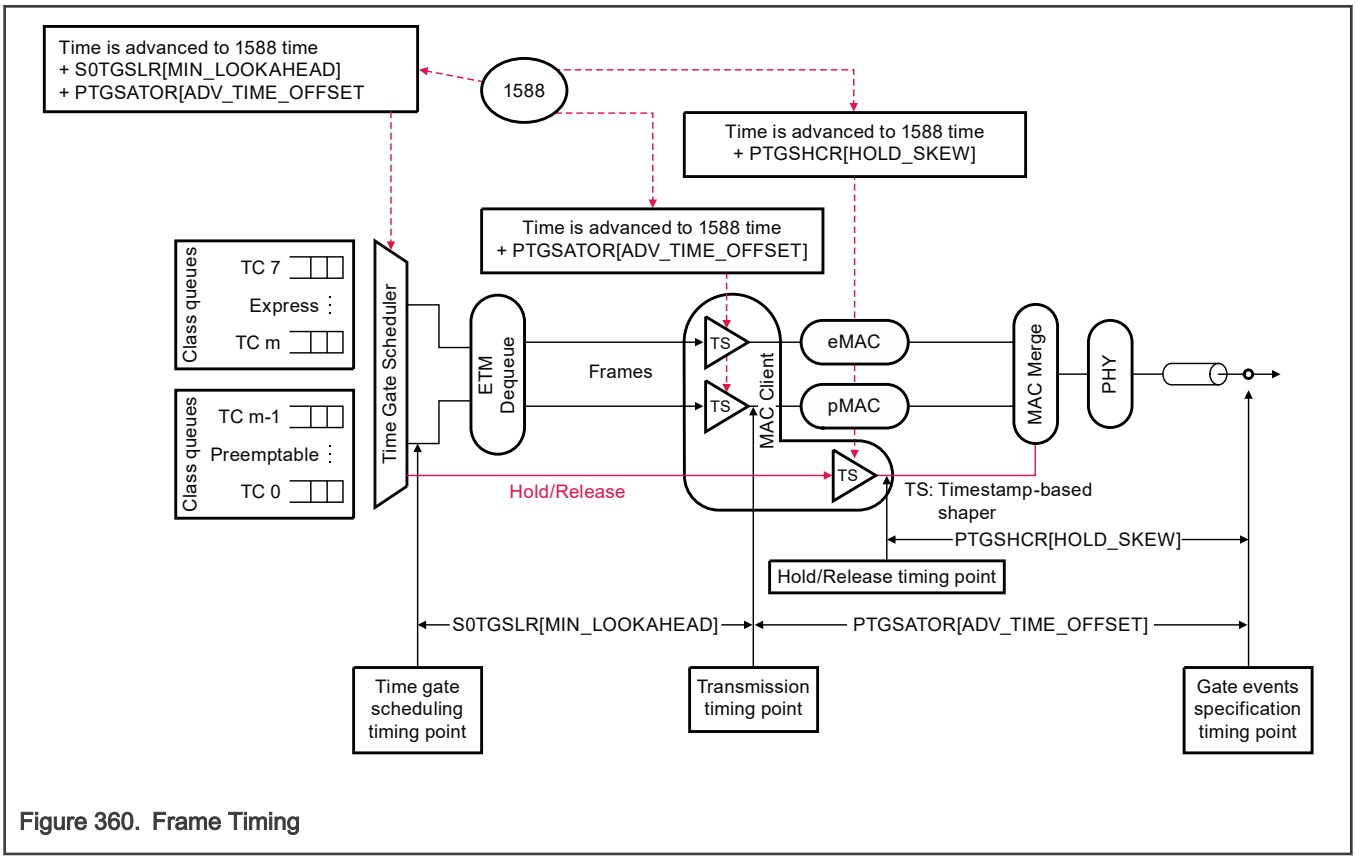


Figure 360. Frame Timing

For time gate scheduling, a frame that appears at the time gate schedule timing point (reference point at which the time of the gate events are specified (gate control list), would have been scheduled at 1588 time—S0TGSLR[MIN_LOOKAHEAD]—PTGSATOR[ADV_TIME_OFFSET] in the Egress Queuing and Traffic Management block. After which the frame would then have been dequeued, assigned a departure time corresponding to the 1588 time, and sent to the MAC client. The MAC client will then have forwarded the frame to the MAC at 1588 time—PTGSATOR[ADV_TIME_OFFSET]. The added latency through the MAC will have caused the frame to be sent out of the switch at 1588 time. For example, with a S0TGSLR[MIN_LOOKAHEAD] time of 100 ns, PTGSATOR[ADV_TIME_OFFSET] of 200 ns, and a gate opening of 50000 ns, the time gate scheduler will schedule the first frame of the gate at 49700 ns. The frame will then be forwarded to the MAC from the MAC client at 49800 ns and will appear on the output of the switch (time gate schedule timing point) at the gate opening of 50000 ns.

53.4.2.4.1.5.1.4 Strict priority and fair queuing

A total of 8 ETM class queues with configurable combinations of strict priority and fair scheduling is supported. A proprietary algorithm, Weighted Bandwidth Fair Scheduling (WBFS) is used to implement the weighted fair scheduling function. The WBFS algorithm is described in Weighted Bandwidth Fair Scheduling (WBFS) Algorithm.

The class queue scheduler of a switch port has 16 inputs numbered from 0 to 15. Input 0 to 7 are weighted fair whereby input 8 to 15 are strict priority. Strict priority input with a larger numeric value will be served before another with a smaller numeric value, Class queue 7 can be assigned to any of the scheduler inputs numbered from 8 to 15. Preceding queues are then assigned automatically in a contiguous downwards manner till the maximum number of class queues is reached (i.e. 8). For configuring the class queue schedulers, see ETM Class Scheduler Table.

For example, when class queue 7 is assigned to input 15 (the default value), all class queues (0 to 7) within the channel are treated as strict priority. When class queue 7 is assigned to input 7, all class queues (0 to 7) are treated as weighed fair queueing. Any assignment of the class queue 7 in between inputs 8 and 15 will result in some queues as strict priority and some with weighed fair queueing. If the assignment is to an input numerically lower than 7, then the behavior is that of the input 15 setting.

53.4.2.4.1.5.2 ETM class queue congestion management

A multi priority (color-aware) "tail drop" mechanism based on instantaneous queue depth is provided for managing the sizes of the egress class queues. Tail drop refers to a mechanism where arriving frames to a queue are dropped when the queue size reaches some maximum length threshold.

The multi priority (color-aware) "tail drop" mechanism is implemented through the use of congestion groups. A congestion group represents an aggregate of class queues on a given port, for the purpose of implementing queue related depth congestion control mechanisms. There are 8 congestion groups per port, allowing each class queue to have its own congestion group, or allowing several (or all) class queues in the same port to share the same congestion group, whichever is required. A congestion group tracks the number of instantaneous bytes that are queued up on all class queues that are associated with the congestion group. For each congestion group, a set of drop thresholds can be configured, one threshold for each drop resilience level, in order to decide when to start dropping frames arriving to a class queue associate with the congestion group. Note that from congestion group accounting perspective, each replicated frame is considered as a separate frame, meaning that the amount of bytes consumed by each frame reference is equivalent to the length of the entire frame.

Tail drop occurs if an enqueue is attempted with a frame marked as discard resiliency (DR) "x", and this enqueue would cause the number of bytes in the congestion group to exceed the threshold configured for DR "x". Note that hardware does a single check against the threshold corresponding to the DR value of the frame, it does not check other thresholds. Thus, it is important to configure the thresholds for all possible DR values.

Class queue association to a congestion group is configured through an update operation to the ETM Class Queue table, see ETM Class Queue Table for more details. By default, each class queue is associated with a dedicated congestion group. The tail drop thresholds are configured through an update operation to the ETM Congestion Group table, see ETM Congestion Group Table for more details. A query operation to the ETM Congestion Group table for a particular congestion group provides the number of bytes currently in use in all class queues that are members of that congestion group.

53.4.2.4.1.6 Ethernet Tx I/F

The Ethernet Tx I/F block integrates the Tx MAC functionality and implements the interface functions specific to the MAC from the transmit path.

53.4.2.4.1.6.1 Ethernet MAC

In the case where frames are transmitted to an Ethernet MAC, the Ethernet Tx I/F retrieves the frame from the shared internal buffer memory and transfers it to the Ethernet MAC.

Frame modification actions passed along with the frame as metadata, are performed as the frame is being transferred to the MAC.

For a given frame (store-and-forward or cut-through), FCS carried in the received frame is re-used as the FCS for the transmitted frame (the Ethernet MAC will not calculate and append the FCS), unless the following scenarios are encountered:

- There are frame modifications performed on the frame. In this scenario, the FCS carried in the received frame is removed from the frame, and the Ethernet MAC is instructed to calculate and append a new 4 bytes FCS to the frame. If the

Ethernet MAC is configured to pad frames to a minimum size of 64 bytes (PMa_COMMAND_CONFIG[TXP] set to 1), the Ethernet MAC will pad the frame with 0s as needed to extend the frame to 60 bytes and append a new 4 bytes FCS to the frame.

- There are no frame modifications performed on the frame and the frame length (including FCS) is shorter than 64 bytes and the outgoing port is configured to pad a frame to the minimum length of 64 bytes. (PMa_COMMAND_CONFIG[TXP] set to 1). In this scenario, the FCS carried in the received frame is not removed from the frame, and if the frame to be transmitted is shorter than 60 bytes, the MAC will pad the frame to 60 bytes with pad bytes of zeros, and then it will append a new calculated 4 bytes FCS to the frame. If the frame to be transmitted is shorter than 64 bytes but longer than 59 bytes, the MAC will append a new calculated FCS to the frame after the original FCS. If the frame to be transmitted is 64 bytes or longer, the MAC will not append an FCS to the frame.

The Ethernet MAC transmit function manages inter-packet gap and generates preamble and SFD or SMD.

The MAC supports configurable minimum frame size from 18 to 64B, and configurable maximum frame size up to 2000B. See Port MAC a Minimum Frame Length Register (PM0_MINFRM - PM1_MINFRM) and Port MAC a Maximum Frame Length Register (PM0_MAXFRM - PM1_MAXFRM) registers for more details.

NOTE

This device does not support transmitting carrier extension. If operating in half duplex mode with padding disabled, and transmitting frames shorter than 64 bytes, the system must ensure proper function since such frame transmissions are not at least one slotTime in duration.

After a frame has been sent to the Ethernet MAC, the Ethernet Tx I/F performs the following actions:

- For any given transmit frame, the Ethernet MAC returns the timestamp at the moment where the one-octet start frame delimiter (SFD) of the frame was transmitted. PMa_COMMAND_CONFIG[TS_MODE] determines which clock source (synchronized clock or free running clock) is used for sampling the timestamp. This timestamp is used as a response to a host request to capture the timestamp when the frame's SFD is transmitted by the switch's Ethernet MAC.
- Internal buffers holding the frame are released.

For any uncorrectable errors detected (such as cut-through end of frame errors, multi-bit ECC errors), the frame is discarded if the frame transfer to the MAC has not started. These frame discards are counted against the port's Tx discard count register (PTXDCR) along with the setting of the appropriate discard reason flag to 1 in the port's Tx discard count reason register 0 (PTXDCRR0). If the frame transmission has started (possible for both cut-through and store-and-forward mode of operations), the frame can no longer be discarded at this point. Instead the MAC is instructed to set the frame FCS field to a bad CRC value to ensure that the next hop identifies the frame as being bad. The error (no longer a frame discard) is counted against the port's MAC FCS counters (PMa_TERRn and PMa_TFCSn).

53.4.2.4.1.6.2 Pseudo MAC

For the pseudo MAC, if no frame modification actions have been specified, the original frame is always preserved in its entirety, including the FCS, and only the frame descriptor reference is passed across the pseudo link; thus there is no need to retrieve or copy these frames.

If frame modification actions have been specified, a new head (or start) of the frame called annotated frame header is created and stored in the shared internal memory buffer along with an FCS covering the annotated frame header. This FCS is distinct from the FCS covering the entire original frame. A reference to this annotated frame header is added to the frame descriptor before transferring the frame descriptor across the pseudo link. This annotated frame header represents the first n bytes of the frame, along with an indication of how many bytes from the start of the original frame are no longer valid. Frame descriptors transferred across a pseudo link include references to both the original frame and annotated header if present.

If there is no shared internal memory buffer available for storing the annotated frame header, the frame is discarded These frame discards are counted against the port's Tx discard count register (PTXDCR) along with the setting of the Shared Memory Resource Exhaustion Discard Reason (SMREDR) flag to 1 in the port's Tx discard count reason register 0 (PTXDCRR0).

53.4.2.4.2 Switch Management Port

A switch management port is a designated switch port to which frames can be mirrored, or redirected/copied. Redirected/copied frames include MAC control frames such as MAC learning, frames that match a classification table entry (e.g. Ingress Port Filter table) with an action to redirect or copy to the switch management port.

The switch allows one switch port that is connected internally to an ENETC port, to be designated as a switch management port (indicated in MPCR[PORT]). Switch ports connected to an Ethernet MAC cannot be designated as a switch management port.

The switch management port can also be a valid destination for the regular/normal output actions of the various lookup tables (such as FDB).

On a switch management port, each frame passed along includes out of band metadata with the following info:

- **From the switch to ENETC** - An indication of the source port ID from which the frame was received and the reason (referred to as the Host Reason) on why this frame was forwarded to the switch management port (e.g. MAC learning, ingress port filter table entry matched (i.e. software defined Host Reason), regular forwarded traffic). The Host Reason consists of a list of 4-bit codepoints; see table below for the definition of the Host Reason codepoints.
- **From ENETC to the switch** - An indication on how to forward the frame; (e.g.to direct the frame out a specific egress queue or forward as it was received from the switch management port itself or forward as it was received from another switch port (i.e. switch port masquerading).

Table 529. Host Reason Codepoints

Host Reason Codepoints (4 bits)	Description
0x0	Regular forwarded frame
0x1	Ingress mirror
0x2	MAC learning
0x3	Timestamp response. Note that a timestamp response is not associated with a received frame. It is used to identify a response that returns the transmission time of a frame on a switch port. See Tx two-step timestamp, for more details.
0x4 - 0x7	Reserved
0x8 - 0xF	Software defined Host Reason

Following are the trigger points in the ingress packet processing path where a frame can be re-directed or copied to the switch management port with a Host Reason set to a codepoint other than 0. As noted earlier, the switch management port can also be a valid destination from the regular/normal output actions of the various lookup tables (e.g. FDB).

- **Ingress port mirroring** - Any ingress mirroring trigger points can copy a frame explicitly to the switch management port (IMDCR0[MIRDEST]= 1). The mirrored frame copy metadata field Host Reason is set to "ingress mirror". The mirrored frame copy is not subjected to any further ingress processing (for example, policing, frame modifications). (See Ingress Mirroring, for more details).
- **Ingress Port Filtering** - Ingress Port Filter (IPF) table entry match with the filter forwarding action to redirect or copy the frame to the switch management port. Redirected frames to the switch management port afterwards are only subjected to the rate policing function if the IPF filter rate action is specified. Other subsequent ingress packet processing functions such as stream identification and filtering, ingress frame modification, FRER sequence generation, 802.1Q bridge forwarding, and stream gating are all by-passed. In the case of a “copy to the switch port management" action, ingress packet processing moves to the next stage of processing (that is, stream identification and filtering) but the frame copy to the switch management port is not subjected to any frame modifications (frame is delivered as received from the link). A “copy to the switch port management" action can be overridden by a “copy/redirect to the switch port management" action or a “discard” action specified in an Ingress Stream table entry, or by a “copy/redirect to the switch

port management" action resulting from MAC learning. Host Reason in the IPF table entry has to be set to a software defined Host Reason codepoint (8-15).

- **Ingress Stream** - Ingress Stream table entry match with the forwarding action to redirect or copy the frame to the switch management port. Redirected frames to the switch management port afterwards can be subjected to rate policing, stream gating, maximum SDU check, and ingress stream statistics update. Other subsequent ingress packet processing functions such as ingress frame modification, FRER sequence generation and 802.1Q bridge forwarding are all by-passed. In the case of a “copy to the switch port management" action, ingress packet processing moves to the next stage of processing but the frame copy to the switch management port is not subjected to any frame modifications (frame is delivered as received from the link). A “copy to the switch port management" action can be overridden by a “copy/redirect to the switch management port” action resulting from MAC learning. Host Reason in the Ingress Stream table entry has to be set to a software defined Host Reason codepoint (8-15).
- **Software MAC Learning** - MAC learning frames redirected or copied to the switch management port. The MAC learning frame metadata field Host Reason is set to "MAC learning". Copied/redirected MAC learning frames are subjected to storm control and policing. Note that no egress VLAN processing is performed on MAC learning frames, and any frame modification specified earlier will not be applied to the MAC learning frames.

As described above, multiple "redirect/copy to the switch port management" actions can be specified along the ingress processing path for a given frame, however, only a single "redirect/copy to the switch port management" action can be applied to a frame. For example:

- A "copy to switch management port" action set from a match in the Ingress Port Filter table lookup followed by a software MAC learning (unsecure) lookup. If the software MAC learning lookup performed into the FDB table, resulted in a match, a copy of the frame would be sent to the switch management port with the software defined Host Reason configured in the matched Ingress Port Filter entry. If the software MAC learning lookup performed into the FDB table, resulted in no match, a copy of the frame would be sent to the switch management port with the MAC learning Host Reason.
- A "redirect to switch management port" action set from a match in the Ingress Port Filter table with a software MAC learning function (secure/unsecure) enabled. The frame would be redirected to the switch management port with the software defined Host Reason configured in the matched Ingress Port Filter entry, as the software MAC learning function would not be executed.
- A "copy to switch management port" action set from a match in the Ingress Port Filter table lookup followed by "copy to switch management port" action set from a match in the Ingress Stream table lookup. A copy of the frame would be sent to the switch management port with the software defined Host Reason configured in the matched Ingress Stream entry.

Frames with Host Reason set to 0 (regular forwarded frame), are sent towards the switch management port via the Egress Queuing and Traffic Management function. Frames with a Host Reason set to a codepoint other than 0 (not regular forwarded frame), are sent directly to to the ENETC Ingress Congestion Manager (ICM) function, where they will be queued based on the internal QoS set by the switch, by-passing the switch Egress Packet Processing function, the switch Egress Queuing and Traffic Management function, and the ENETC Ingress Port Processing function. Furthermore, the pseudo MAC statistics are not updated for these frames.

Frames redirected or copied to the switch management port, as a result of matching a classification table entry (Ingress Port Filter table or Ingress Stream table), or as a result of MAC learning, may have multiple copies as a result of other functions being applied to the frame (ingress mirroring, regular forwarded traffic). Thus, a received frame may result in up to three frames being delivered to the switch management port:

- Frame with a software defined Host Reason or MAC learning Host Reason.
- Frame with an ingress mirror Host Reason.
- Frame with a regular forwarded frame Host Reason.

Frames (or management frames) redirected or copied to the switch management port, as a result of matching a classification table entry (Ingress Port Filter table or Ingress Stream table), or as a result of MAC learning, are not subjected to the switch congestion management. In the case where other copies exist, any discard to those copies does not impact the management frame copies. For example, if these copies are discarded due the switch congestion management (insufficient memory space to accommodate frame in buffer pool, or overall switch shared internal buffer memory usage is full), the management frame copy is not discarded,

and will be delivered directly to ENETC without consuming any switch buffering. Note that the management frames are subjected to ENETC congestion management (that is, ENETC Ingress Congestion Management (ICM)).

53.4.2.4.3 Cut-through Forwarding

In cut-through mode of operation, once parsing of the frame is completed and although the entire frame may not have been entirely received, the switch proceeds with ingress filtering and forwarding decisions and then queues the frame for transmission. This allows the switch to start transmission of a given frame to one or more egress ports while still receiving the remaining of frame on the ingress port.

53.4.2.4.3.1 Cut-through Forwarding Criteria

Cut-through forwarding is supported for both unicast and multicast frames. The decision on whether to proceed with cut-through forwarding is done on a per-egress port basis. Thus for a given multicast frame received on an ingress port with cut-through forwarding enabled, cut-through forwarding will only occur on the egress ports that meet the criteria listed below for cut-through forwarding. For egress ports not meeting the cut-through criteria, frame will be handled in the store-and forward mode of operation.

For cut-through forwarding to occur on a frame, all of the following criteria must be met:

- Both the frame's ingress port and egress port(s) must be configured to have cut-through forwarding enabled, using the port cut-through forwarding configuration register (PCTFCR).
- Cut-through forwarding of a frame will only occur when the link speed of the egress port is equal to or less than the link speed of the ingress port. This check is performed dynamically by the switch before deciding whether to proceed with cut-through forwarding or to revert back to the store-and forward mode of operation.
- Frame was received from the Ethernet Express MAC; cut-through forwarding is disabled for frames received from the Ethernet preemptable MAC.
- Cut-through forwarding has not been disabled by a switch packet processing action; see Disabling Cut-through Forwarding and Special Considerations for more details.

Cut-through mode of operation on a pseudo link is not applicable, and thus cut-through forwarding cannot be enabled on an ingress port or egress port attached to a pseudo link.

53.4.2.4.3.2 Cut-through Forwarding Timing

Once the first 24 bytes of data is received and presented to the Ingress Packet Processing (IPP), it starts parsing the frame header (first 24 bytes). If the end of the parse graph (up L4) has not been reached while parsing the first 24 bytes of the frame, the IPP waits for the next 24 bytes to be received so that it can resume parsing, and so on up to 144 bytes. By default, the parser will parse and extract fields from each protocol layer (L2, L3 and L4). There is the option to configure the parser to end parsing at L2 or L3 (via the port parser configuration register (PPCR)), regardless of the content of the frame. This option can be used to trigger cut-through forwarding earlier by reducing the number of bytes to be parsed.

Once parsing is completed and although the entire frame may not have been entirely received, frame moves through the switch processing pipeline and starts getting transmitted on all destination ports.

Functions such as MAC learning, counters and state updates relaying of the knowledge of the frame length, are delayed to the time when the end of frame is received.

53.4.2.4.3.3 Disabling Cut-through Forwarding

On a ingress port that has been enabled for cut-through forwarding, cut-through forwarding can be disabled as a frame moves through the switch processing pipeline, in the following ways:

- Ingress Port Filter table entry match with a cut-through forwarding disable action.
- Ingress Stream table entry match with a cut-through forwarding disable action; can be disabled on a specific destination port or on all destination ports.
- Ingress Stream Filter table entry match with a cut-through forwarding disable action.
- Stream gating

- Software can specify to disable cut-through forwarding before the administrative stream gate control list takes affect.
- Software can specify to disable cut-through forwarding for frames received after the stream gate control list ends and before CYCLE_TIME restarts.
- Software can specify to disable cut-through forwarding during a gate entry interval.

- FDB/L2 IPv4 Multicast filter table entry match with a disable cut-through action; can be disabled on a specific destination port or on all destination ports.

53.4.2.4.3.4 Special Considerations

Following are the specific frame handling actions taking when a frame is being forwarded in the cut-through mode of operation, to at least one port

- MAC learning
  - MAC learning is delayed to the end of frame. MAC learning is not performed if the received frame FCS is bad.
- Stream gating
  - The initial segment size of the cut-through frame is used for checking interval maximum number of bytes.
  - If a cut-through frame is accepted, accumulated interval byte count gets updated later when the entire frame is received.
- Rate policing
  - The initial segment size of the cut-through frame is used by the rate policer for checking if it accepts the frame.
  - If a cut-through frame is accepted, the rate policer byte credits get updated as data is received (on cut-through segment transfers).
- Storm control
  - The initial segment size of the cut-through frame is used by storm control policer to check if it accepts the frame.
  - If a cut-through frame is accepted, the storm control policer byte credit gets updated later when the entire frame is received.
- Congestion Management
  - The initial segment size of the cut-through frame is used by to determine if there is available memory space to accommodate the new frame in the selected buffer pool.
  - If a cut-through frame is accepted, buffer pool usage counts get updated as data is received (cut-through segment transfers).
- Frame Replication and Elimination for Reliability (FRER) sequence recovery
  - Cut-through forwarding must be disabled on any stream in which FRER recovery is enabled.
- Egress queuing and traffic management
  - Queue congestion management - The initial segment size of the cut-through frame is used by to determine if there is available space in the congestion group to accommodate the new frame. If a cut-through frame is accepted, congestion group usage counts get updated as data is received (cut-through segment transfers).
  - WBFS - Maximum frame size from the configured Transmit MSDU (PTCaTMSDUR, where a is the traffic class number) is assumed for a cut-through frame at the head a of the queue. Once the cut-through frame has been transmitted, the frame length is adjusted using the actual discovered length of the frame.
  - 802.1Qbv - Maximum frame size from the configured Transmit MSDU (PTCaTMSDUR, where a is the traffic class number) is assumed for a cut-through frame at the head of a queue. PTCaTMSDUR[SDU_TYPE] must be set to 1, if cut-through frames are expected.
- IEEE 802.1Qbu frame preemption

- When cut-through forwarding and IEEE 802.1Qbu frame preemption are enabled, great care should be taken to ensure that cut-through is disabled for traffic destined to the preemptable MAC. Initial IPV of a frame should be set in such way that the IPV doesn't map on egress to any traffic class that has been assigned to the preemptable MAC. Explicitly entries should be added in the Ingress Port Filter table, Ingress Stream table, or Ingress Stream Filter table to filter preemptable traffic with the action to disable cut-through forwarding and the action to set the IPV to a value that maps to a traffic class that has been assigned to the preemptable MAC. Note that sending a cut-through frame on a preemptable MAC can lead to a preempted fragment of size smaller than 64 bytes.

- HSR tag insertion
  - Cut-through mode of operation is not supported if insertion of HSR tag (or replacement R-TAG with HSR tag) has been configured, as it requires knowledge of the received frame length in order to set the HSR LSDU size field. If cut-through forwarding is occurring and insertion of HSR tag is configured, hardware will set the HSR LSDU size field based on the frame length configured In PTCaTMSDUR.
- Egress frame modification
  - Cut-through mode of operation is not supported if the frame modification action "removing entire Ethernet payload and replacing it with new Ethernet payload", has been configured or if the action "removing entire frame and replacing it with a new frame", has been configured.
- Ingress mirroring
  - Cut-through forwarding is disabled for the frames (copy) being sent to the mirrored destination.
- Per-stream filtering - maximum SDU check
  - The maximum SDU check is not applicable to cut-through frames, see Maximum SDU Check for more details.

53.4.2.4.3.5 Errored frame handling

When operating in cut-through mode, and if the frame transmission has been started already by the switch and the frame has an end of frame error, the switch puts a bad CRC value in the frame FCS field to ensure that the next hop identifies the frame as being bad.

53.4.2.4.4 Ingress Mirroring

Ingress mirroring provides capability to selectively copy frames entering the switch, to an egress switch port for the purpose of traffic monitoring. Ingress mirroring is enabled on the switch by setting register IMDCR0[MIREN] to 1. When ingress mirroring is enabled, the following methods are provided to select frames to be mirrored:

- All errored free frames entering a specific switch port. This method is selected by setting the port register flag PMCR[IMIRE] to 1.
- Ingress Port Filter table entry match with the action to mirror the frame.
- Ingress Stream table entry match with the action to mirror the frame.
- Ingress Stream Filter table entry match with the action to mirror the frame.
- FDB/L2 IPv4 Multicast filter table entry match with the action to mirror the frame.

The above methods provide the means to configure mirroring trigger points along the ingress processing path. All mirroring trigger points can be configured/active at the same time. The first mirroring trigger point hit by a frame along the ingress processing pipeline, will cause the generation of a mirrored frame copy. If the regular frame hits another mirroring trigger point afterwards as it continues to be processed by the ingress processing pipeline, that mirroring trigger point is ignored; no additional mirrored frame copies will be generated for that frame, there is only one mirrored frame copy generated for a given frame. Note that cut-through forwarding is disabled for the mirrored frame copy. The mirrored frame copy is forwarded to the mirrored destination, as a store-and-forward frame.

Once a trigger point generates a mirrored frame copy, the mirrored frame copy is forwarded immediately to the switch port designated as the destination mirror port. The mirrored frame copy is not subjected to any further ingress processing (e.g. policing, ingress frame modifications) with the following exception:

- In the case where there is insufficient space in the switch shared internal buffer memory to accommodate the received frame, the received frame is dropped and as a result none of the references to that frame can be forwarded including the mirrored frame copy.

Otherwise, any drop on the original frame afterwards doesn't impact the mirrored frame copy.

The destination mirror port is statically specified by configuring the ingress mirror destination configuration register 0 (IMDCR0); the options are:

- To forward explicitly to the switch management port (IMDCR0[MIRDEST] = 1); the exiting switch port is not explicitly specified, instead it is obtained from the register that specifies the switch port that has been designated as a switch management port. For this option, the mirrored frame copy metadata field Host Reason is set to "ingress mirror" (i.e. not as a regular forwarded frame).
- To forward explicitly to any switch port including the one designated as a switch management port (IMDCR0[MIRDEST] = 0 and mirror destination port specified in IMDCR0[PORT]). When a mirrored frame copy is forwarded explicitly to a switch port that has been designated as switch management port, the mirrored frame copy metadata Host Reason is set to "regular forwarded frame".

The DR and IPV assigned to the mirrored frame copy are obtained from the DR and IPV values configured in the ingress mirror destination configuration register 0 (IMDCR0). The mirrored frame copy DR and IPV are only used in the egress processing path (e.g. selecting egress class queue, performing congestion group check). Note that the IPV of the original frame is used to access the buffer pool mapping function to select the buffer pool.

Each frame mirrored copy is treated independently from the regular traffic (e.g. mirrored frame copy is in addition to any multicast and/or switch management port copies of the regular frame). If for example, if both regular frame and its mirrored copy exits the same port, the egress port transmits two independent copies of the frame.

From a congestion perspective, mirrored frames are treated in the same way as multicast frames, only one copy of the frame gets accounted for buffering purposes but on egress, as for multicast frames, each copy is subjected to ETM class queue congestion management; congestion group is checked to determine if there is available space to accommodate the mirrored frame copy, if so then the congestion group accounting is updated accordingly, otherwise the mirrored frame copy is dropped.

Ingress mirroring constraints are:

- Cut-through forwarding is disabled for the mirrored frame copy.
- No ingress frame modifications are supported for the mirrored frame copy.
- Egress frame modifications can be specified for the mirrored frame copy; the egress frame modifications are specified in the mirror destination configuration register 1 (IMDCR1). Egress frame modifications are supported only for the option IMDCR0[MIRDEST] = 0.

53.4.2.4.5 Time Capture

Time capture function provides the ability to measure latency of a given frame through the switch.

When measuring latency, the entry point is a physical Rx port and the exit point is a physical TX port.

The time capture function should not be used when replicating to multiple Tx Ports or when the Tx Port is attached to a pseudo link.

Latency is measured based on the captured synchronized timer timestamp at the Rx port and Tx port, and is captured relative to the SFD..

Frames whose latency is to be measured are identified through one of the following classification stages where the time capture feature may be enabled for that frame:

- Ingress Port Filter table.
- Ingress Stream table.
- Ingress Stream Filter table.
- FDB/L2 IPv4 Multicast filter table.

The time capture function operates in single arm mode, where the time capture function will be triggered for a single frame and then will disarm itself. If a trigger frame is not found within a timeout period, the time capture is disarmed. A maskable interrupt is generated when a capture trigger occurs or when timeout period elapsed. The timeout is specified in Time Capture Configuration Register (TCCR).

This time capture circuity can only operate correctly if all the Tx latency captures are done before a new Rx timestamp capture occurs. Arrival interval of trigger frames must be longer than the maximum egress queueing latency. The circuit will only arm when there is no trigger frame inflight in the switch.

53.4.2.4.5.1 Single arm mode

When the time capture function is armed, which is achieved by setting TCCR[ARM] register to 1, the first received frame which triggers a timestamp capture event, its Rx timestamp is captured in the TCRPTSR register. The latency is captured at the transmit port where the trigger frame is being forwarded to. The measured latency is captured in the transmit port registers PTCMINLR and PTCMAXLR.

A maskable interrupt is generated using the registers TCIER/TCRPIDR/TCMSIVR when either of the following occur:

- The last frame has been transmitted on the Ethernet MAC.
- The timeout value has expired.

For the first Rx trigger frame, in addition to capture its receive timestamp in TCRPTSR register, the TCRPSR[RX_CNT] register is incremented by 1 and the TCRPSR[RX_PORT] register is updated with the Rx port number of the trigger frame. Only the first frame (TCRPSR[RX_CNT] = 0) captures the Rx timestamp.

For subsequent Rx trigger frames (TCRPSR[RX_CNT] > 0), only the TCRPSR[RX_CNT] register is incremented up to a maximum value of 3. This is an ingress indication that trigger frames were received faster than expected, faster than the rate at which the circuity can be re-armed.

On the transmission of the first trigger frame on a port (PTCMINLR[COUNT]=0), the TCRPIDR[TX_PORTn] register bit is set and its port registers PTCMINLR and PTCMAXLR are loaded with the latency (latency = current 1588 time - TCRPTSR).

For subsequent transmission of trigger frames on a port, its port register PTCMINLR[COUNT] is incremented up to a maximum value of 3. This is an egress indication that trigger frames were received faster than expected, faster than the rate at which the circuity can be re-armed.

When all the copies of the first trigger frame are transmitted, if enabled, the time capture interrupt will be generated. When the circuity is re-armed by software, the TCIDR, TCRPSR, PTCMINLR and PTCMAXLR registers are all cleared.

In the single arm mode, the PTCMINLR and PTCMAXLR registers will hold the same value.

53.4.2.4.6 Switch Statistics Counters

53.4.2.4.6.1 MAC Statistics Counters

For the MAC statistic counters, see section Ethernet MAC Statistics Counters for the Ethernet MAC statistics counters and section Pseudo MAC Statistics Counters for the pseudo MAC statistics counters.

53.4.2.4.6.2 Ingress Statistics Counters

The table below lists the available ingress traffic statistics counters.

Table 530. Ingress Traffic Statistics Counters

Counter	Description
Ingress Stream Count table - RX_COUNT	Received octets counter (ifInOctets) on a per stream basis
Rate Policer table - statistics counters	Bytes count on a per rate policer instance. Also available are marked/ remarked frame counts

Table continues on the next page...

Table 530. Ingress Traffic Statistics Counters (continued)

Counter	Description
Ingress Port Filter table - MATCH_COUNT	Ingress Port Filter entry matched count
Cut-through forwarding count - CTFCR	Cut-through frames counter of frames forwarded to at least one egress port

The table below lists the available ingress frame drops statistics counters.

Table 531. Ingress Frame Drops Statistics Counters

Counter	Description
PRXDCR	Frame drops count in the ingress datapath processing pipeline on a per port basis with the setting of the discard reason in the port's Rx discard count reason register 0/1 (PRXDCRR0/1)
Ingress Stream Count table - frame drops counters	Maximum SDU frame drops count, stream gate frame drops count and rate policer instance frame drops count on a per stream basis
Rate Policer table - DROP_FRAMES	Frame drops count on a per rate policer instance basis
BPDCR	Frame drops count in the bridge forwarding processing function on a per source port basis with the setting of the discard reason in the port's Rx discard count reason register 0/1 (BPDCRR0/1)

53.4.2.4.6.3 Egress Statistics Counters

The table below lists the available egress traffic statistics counters.

Table 532. Egress Traffic Statistics Counters

Counter	Description
Egress Count table - ENQ_FRM_CNT	Enqueued frames count onto ETM class queue(s)
ETM Class Queue table - DEQ_BYTE_CNT	Dequeued bytes count on a per ETM class queue basis
ETM Class Queue table - DEQ_FRM_CNT	Dequeued frames count on a per ETM class queue basis
Egress Sequence Recovery table - traffic statistic counters	In order packets counts, out of order packets counts, lost packets counts and tag-less packets counts

The table below lists the available egress frame drops statistics counters.

Table 533. Egress Frame Drops Statistics Counters

Counter	Description
PTXDCR	Frame drops count in the egress datapath processing pipeline on a per port basis with the setting of the discard reason in the port's Tx discard count reason register 0/1 (PTXDCRR0/1)
Egress Count table - REJ_FRM_CNT	Rejected frames count; rejected from attempting to enqueue onto ETM class queue
ETM Class Queue table - REJ_BYTE_CNT	Rejected bytes count on a per ETM class queue basis
ETM Class Queue table - REJ_FRM_CNT	Rejected frames count on a per ETM class queue basis
ETM Class Queue table - frame drops counts	Frame drops counts due to frame descriptor pointer error

Table continues on the next page...

Table 533. Egress Frame Drops Statistics Counters (continued)

Counter	Description
Egress Sequence Recovery table - ROGUE_PACKETS	Packet drops count by the vector recovery algorithm whose sequence number have fallen outside of the recovery history window
Egress Sequence Recovery table - DUPLICATE_PACKETS	Packet drops count due to a duplicate sequence number

53.4.2.4.7 Table Management Commands

Table management commands use the NTMP protocol. Familiarity with the basic NTMP concepts, described in the NETC Table Management Protocol (NTMP) section of this document, is needed to understand the material documented in this section.

Some functionality is configured and controlled using control messages sent to the switch using a BD ring interface . This BD ring interface is referred to as the command BD ring.

Control messages are primarily used when the underlying functionality being configured may be too large to configure using direct registers (e.g. large tables).

There are two switch command BD rings, with the format of the descriptor detailed in NTMP Request Message Header Format and in NTMP Response Message Header Format. Command BD ring selection amongst all NETC functions' command BD rings (i.e. the pair of switch rings plus the per-SI ENETC rings) is performed in a round robin manner. Each command BD ring is serviced (one command per round) once in each round of the robin. Each round considers all command BD rings defined amongst all NETC functions.

With respect to the pair of switch command BD rings, control messages may be issued to both rings in parallel. There is no ordering of command execution between the two rings. This mechanism is advantageous when a mix of short and long duration commands are issued. Short duration commands issued in one of the rings will interleave with and pass longer duration commands (e.g. table dump) executing in the other ring. NETC hardware provides periodic interleaving opportunities during execution of long duration commands such that a short command targeting a hardware resource that is also servicing a long command will not be excessively delayed. This interleaving mechanism is non-configurable.

The NETC serializes commands within a single ring. It is safe to pipeline commands A and B within a ring where command A must complete before command B begins. Any ordered processing must be done either using a single ring, or waiting until command A has completed before issuing command B.

Each command ring is configured as follows:

1. Configure the command ring base address registers CBDRaBAR0 and CBDRaBAR0, where a is the ring number. The address must be 128 byte aligned.
2. Configure the command ring producer index register CBDRaPIR. Software should initialize the index value at this address to 0 to reflect an empty ring. Optionally if the ring is pre-initialized with BDs the index values should reflect the number of BDs added.
3. Configure the command ring consumer index register CBDRaCIR. This index value indicates the next BD to be processed by the hardware. When the ring is re-initialized this register should be set to 0. The consumer index is later updated by the hardware during processing of the ring.
4. Configure the command ring size CBDRaLENR to indicate how many 32B entries, are allowed in the ring. There is no wrap bit to indicate full/empty; one entry is used to differentiate between full and empty. When producer index + 1 equals consumer index, the ring is considered full. When the producer/consumer indices match, the ring is always considered empty.
5. Enable the command ring CBDRaMR [EN]=1.

To add one or more buffer descriptors (or commands) to an enabled transmit ring, follow the procedure below. To add one or more buffer descriptors (or commands) to an enabled transmit ring, follow the procedure below.

1. Software checks to make sure the ring is not full by examining the consumer index (CI) register, CBDRaCIR. Note that maximum number of BDs allowed on a ring is one less than ring size.

2. Software adds one or more BDs to the ring. A memory barrier must be used to guarantee update to memory. If software requires an interrupt when a BD has been transmitted, set BD[CCI] =1 and enable the interrupt through CBDRaIER[CBDCIE].
3. When CBDRaCIR [BDR_INDEX] match CBDRaPIR [BDR_INDEX], the ring is empty. After one or more BDs have been processed, the consumer index will be updated by hardware.

Commands supported by the switch are described in the table below.

Table 534. Supported Switch Commands

Table ID	Table Name	Table Type
5	Time Gate Scheduling Table	Indexed table
10	Rate Policer Table	Indexed table
13	Ingress Port Filter Table	Ternary match table
15	FDB Table	Hash table
16	L2 IPV4 Multicast Filter Table	Hash table
18	VLAN Filter Table	Hash table
22	ETM Class Queue Table	Indexed table
23	ETM Class Scheduler Table	Indexed table
30	Ingress Stream Identification Table	Hash table
31	Ingress Stream Table	Indexed table
32	Ingress Stream Filter Table	Hash table
33	Egress Treatment Table	Indexed table
34	Ingress Sequence Generation Table	Indexed table
35	Egress Sequence Recovery Table	Indexed table
36	Stream Gate Instance Table	Indexed table
37	Stream Gate Control List Table	Indexed table
38	Ingress Stream Count Table	Indexed table
39	Egress Count Table	Indexed table
40	Frame Modification Table	Indexed table
41	Buffer Pool Table	Indexed table
42	Shared Buffer Pool Table	Indexed table
43	ETM Congestion Group Table	Indexed table
44	Frame Modification Data Table	Indexed table

53.4.2.4.7.1 Time Gate Scheduling Table

Table 535. Table ID 5 - Time Gate Scheduling Table Common Attributes

TABLE_ID	5

Table continues on the next page...

Table 535. Table ID 5 - Time Gate Scheduling Table Common Attributes (continued)

TABLE_VERSION	0
Table Type	Static bounded index table
Table Description	The Time Gate Scheduling table contains time gate scheduling configuration and operational information for a NETC function. There is one Time Gate Scheduling table for each ENETC instance and one Time Gate Scheduling table for the entire switch. For the switch, there is one entry for each switch port, whereby for an ENETC instance, there is a single entry for the entire ENETC instance. Each entry includes both an administrative gate control list and an operational gate control list, along with related configuration and status parameters. There is an internal memory that is used to store the administrative gate control list and the operational gate control list of each table entry. The number of words available for all entries in the Time Gate Scheduling table is specified in register TGSTCAPR[NUM_WORDS]. The memory is managed using a single free list of fixed equal size units (or words) of memory. Free memory units from the free list are available to be dynamically allocated to any gate control lists subject to the total memory available and maximum gate control list length (TGSTCAPR[MAX_GCL_LEN]. Each gate control list is comprised of a sequence of gate operation entries where each entry occupies one word of memory. Table management command operations Update and Query are supported. After reset, each entry is present with all fields cleared to zero; there is no administrative gate control list and operational gate control list, indicated by their gate control list length being set to 0. An administrative gate control list can only be added by an Update operation, on a entry that contains no administrative gate control list (length is set to 0). An existing administrative gate control list cannot be directly updated/modified. The administrative gate control list contained in the entry must be first removed by issuing an Update operation that sets the administrative gate control list length to zero, followed by another Update operation that adds the new administrative gate control list. An administrative gate control list cannot be removed if its config change time is 1 ms away. Therefore, before attempting to remove an administrative gate control list, a check must be performed to determine if the config change time minus the current time is less than 1 ms. If so, then the administrative gate control list can no longer be removed as it will become the operational gate control list. There are two ways to obtain the config change time: • config change time is returned in the STATUS field of a command response to a command configuring an administrative gate control list. • config change time can be retrieved by issuing a query operation against the Time Gate scheduling table entry. Once an operational gate control list is installed, it cannot be removed by a table management command. Disabling time gate scheduling on a port (PTGSCR[TGE]=0), will remove both the operational gate control list and the administrative gate control list from the corresponding table entry. When adding an administrative gate control list while there is a currently running gate control list (that is operational gate control list), the base time of the administrative gate control list (ADMIN_BASE_TIME value) must be set to a time in the future. The minimum future time is equal to: *CurrentTime* + *LookAheadTime* + *TableCmdProcessingTime* + (2 × *OperCycleTime*) Where: *CurrentTime* = 1588 (synchronized) time at which software is to issue the Time Gate Scheduling table update command *LookAheadTime* = E*a*TGSLR/S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET] *TableCmdProcessingTime* = Hardware processing time of the Time Gate Scheduling table update command 21,200 NETC clock cycles; for example, @240Mhz ~ 90µs *OperCycleTime* = Cycle time of the operational gate control list

Table continues on the next page...

Table 535. Table ID 5 - Time Gate Scheduling Table Common Attributes (continued)

	Thus, if there is a currently running gate control list and the base time of the administrative gate control list to be added is within the following range: *AdminBaseTime* < *CurrentTime* + *LookAheadTime* + *TableCmdProcessingTime* + (2 × *OperCycleTime*) Where: *AdminBaseTime* = Base time of the administrative gate control list Then software must adjust the base time of the administrative gate control list (ADMIN_BASE_TIME value) as follows: *Adjusted AdminBaseTime* = *AdminBaseTime* + (N × *AdminCycleTime*) where N is the smallest integer for which the relation *Adjusted AdminBaseTime* ≥ *CurrentTime* + *LookAheadTime* + *TableCmdProcessingTime* + (2 × *OperCycleTime*) would be TRUE. When the administrative gate control list is installed (by the hardware) as the operational gate control list, the gate control list itself is not copied, instead the gate control list is assigned as the operational gate control list. A query to the administrative gate control list will then return 0 for the length indicating that there is no administrative gate control list present in the table entry. However, note that ADMIN_ BASE_TIME, ADMIN_CYCLE_TIME and ADMIN_CYCLE_TIME_EXT values are preserved, until a new administrative gate control list is configured.			
Number of Entries	For the switch, the number of entries is equal to the number of switch ports; number of switch ports is indicated in register SCAPR0[NUM_PORT]. For ENETC, there is a one table entry per table.			
Default Reset Behavior	After reset, each entry is present with all fields cleared to zero; there is no administrative gate control list and operational gate control list, indicated by their gate control list length being set to 0			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x2 = Update. 0x4 = Query.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	Variable	4

Table continues on the next page...

Table 535. Table ID 5 - Time Gate Scheduling Table Common Attributes (continued)

	OLSE_DATA	Operational List State Element	Variable	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	Yes			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = Reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x0D0 = Command issued when time gating function is disabled for the port. 0x0D1 = Update action attempted on an existing admin gate control. An existing admin gate control list cannot be modified, Delete admin gate control list first before creating a new admin list. (Use update action with ADMIN_CONTROL_LIST_LENGTH =0 to perform delete). 0x0D2 = Update action attempted exceeds TGSTCAPR[MAX_GCL_LEN]. 0x0D3 = Update action attempted exceeds TGSTCAPR[NUM_WORDS]. 0x0D4 = Insufficient resources to perform the requested operation (not enough free time gate list entries) 0x0D5 = Update action attempted with ADMIN_CYCLE_TIME, ADMIN_TIME_INTERVAL_GE_i or truncated ADMIN_TIME_INTERVAL_GE_n due ADMIN_CYCLE_TIME specified is not sufficient to transmit 64 byte of frame data + header overhead, or if a gate time interval specified is zero. Header overhead is specified in the PTXSDUOR register. 0x0D6 = Update action attempted with ADMIN_BASE_TIME specified that is smaller than *CurrentTime* + *LookAheadTime* + *TableCmdProcessingTime* + (2 × *OperCycleTime*). 0x0D7 = Update action attempted with ADMIN_CYCLE_TIME + ADMIN_CYCLE_TIME_EXT is greater than 2^{32}-1. 0x0D8 = Query action issued when config change occurred. Retry query. 0x0D9 = Update action attempted with ADMIN_HR_CB_GE_i set to an invalid value.			

Table 536. Table ID 5 - Time Gate Scheduling Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		
																--															CFGEU	0x0
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																...
CFGE_DATA																																Variable

Table 537. Table ID 5 - Time Gate Scheduling Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 5 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
Variable-64	Variable	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 538. Table ID 5 - Time Gate Scheduling Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS[31:0]																																0x0
STATUS[63:32]																																0x4
STATUS[95:64]																																0x8
ENTRY_ID																																0xC

Table continues on the next page...

Table 538. Table ID 5 - Time Gate Scheduling Table Response Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CFGE_DATA																																0x10
...																																...
CFGE_DATA																																Variable
OLSE_DATA																																4-Byte Aligned
...																																...
OLSE_DATA																																Variable

Table 539. Table ID 5 - Time Gate Scheduling Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
95-0	96	STATUS	Status Bits [63:0] contain the value of the ConfigChangeTime variable as computed by the SetConfigChangeTime() procedure during execution of the update action. The value is expressed in nanoseconds. Bits [95:64] are reserved for future use. For generic details on the STATUS field, see Status.
127-96	32	ENTRY_ID [1]	Entry ID Note that the Entry ID value which is supplied corresponds to the Port ID with which this time gate control list is associated. For generic details on the Entry ID field, see Entry ID Management.
Variable-128	Variable	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.
Variable-Variable	Variable	OLSE_DATA[1]	Operational List State Element Data See OLSE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 540. Table ID 5 - Time Gate Scheduling Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ADMIN_BASE_TIME																																0x0
ADMIN_BASE_TIME																																0x4
ADMIN_CYCLE_TIME																																0x8
ADMIN_CYCLE_TIME_EXT																																0xC
--																						ADMIN_CONTROL_LIST_LENGTH										0x10

Table continues on the next page...

Table 540. Table ID 5 - Time Gate Scheduling Table CFGE_DATA Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ADMIN_TIME_INTERVAL_GE_0																																0x14
--												HR_CB_GE_0				--								ADMIN_TC_STATES_GE_0								0x18
...																																...
ADMIN_TIME_INTERVAL_GE_N																																0x14+8N
--												HR_CB_GE_N				--								ADMIN_TC_STATES_GE_N								0x18+8N

Table 541. Table ID 5 - Time Gate Scheduling Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	ADMIN_BASE_TIME	Administrative Base Time This field sets the base time, i.e., the start time, of the administrative gate control sequence. The time is expressed in units of nanoseconds. The field implements the AdminBaseTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 AdminBaseTime parameter, which is the PTP time format, consisting of a 48-bit integer number of seconds and a 32-bit integer number of nanoseconds. Instead of using the PTP time format, the field expects the time to be specified as a 64-bit integer number of nanoseconds. Therefore, if this field is to be set to a value which was obtained from a managed object using the IEEE 802.1Q-2018 PTP time format, the value must be converted by software from the IEEE 802.1Q-2018 format to the 64-bit integer number of nanoseconds.
95-64	32	ADMIN_CYCLE_TIME	Administrative Cycle Time Sets the cycle time of the administrative gate control list. This is the period of time over which the sequence of operations in a gate control list repeats. Command will be rejected if ADMIN_CYCLE_TIME=0. The time is expressed as an integer number of nanoseconds. The field implements the AdminCycleTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 AdminCycleTime parameter, which is a rational number in units of seconds (numerator and denominator). Instead of using the IEEE 802.1Q-2018 AdminCycleTime parameter format, the field expects the time to be specified as a 32-bit integer number of nanoseconds. Therefore, if this field is to be set to a value which was obtained from a IEEE 802.1Q-2018 managed object, the value must be converted by software from the IEEE 802.1Q-2018 format to the 32-bit integer number of nanoseconds. Note that if the cycle time is shorter than the sum of the time intervals (see ADMIN_TIME_INTERVAL_GE_i), the sequence of operations is truncated, which in turn could cause the undesirable effect of overrunning a next gate-close event.
127-96	32	ADMIN_CYCLE_TIME_EXT	Administrative Cycle Time Extension This is the maximum amount of time by which the ADMIN_CYCLE_TIME is permitted to be extended when a new gate control sequence is being installed. The cycle time extension is used to extend the last complete cycle so that it ends at the new base time,

Table continues on the next page...

Table 541. Table ID 5 - Time Gate Scheduling Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			or in other words, that the old and new cycles line up. Setting the cycle time extension to a value that extends the last complete cycle to complete before the new base time (too short), will result in undefined behavior of the time gate operation. The time is expressed as an integer number of nanoseconds.
143-128	16	ADMIN_CONTROL_LIST_LENGTH	Administrative Control List Length This field indicates the number of entries in the administrative gate control list. Valid values for N are 1 to TGSTCAPR[MAC_GCL_LEN] for creating a new administrative gate control list and 0 to remove the administrative gate control list. The command will be rejected if list length is invalid.
159-144	16	--	Reserved
(223 + 64i) - (160 + 64i)	32	ADMIN_TIME_INTERVAL_GE_i	Administrative Time Interval for Gate Entry i, where i = 0, ..., N-1 Specified in nanoseconds. A minimum of 2 gate entries must be specified. Restrictions (applies to any time interval entries): - Minimum gate time interval specified must not be less than (((TGP +2) * 4) + 20)/NETC clock frequency, where TGP is the number of NETC ports (ENETC and switch) where time gating is enabled.
	8	ADMIN_TC_STATES_GE_i	Administrative Traffic Class Gate States for Gate Entry i, where i = 0, ..., N-1. Bit 0: Gate state for traffic class 0. Bit 1: Gate state for traffic class 1. Bit 2: Gate state for traffic class 2. Bit 3: Gate state for traffic class 3. Bit 4: Gate state for traffic class 4. Bit 5: Gate state for traffic class 5. Bit 6: Gate state for traffic class 6. Bit 7: Gate state for traffic class 7. For each of the traffic classes above, the following values apply: 0b = Gate closed. 1b = Gate open.
	8	--	Reserved
	4	ADMIN_HR_CB_GE_i	Administrative gate operation type (as per IEEE 802.1Q-2018) field for gate control list entry i. 0x0: SetGateStates. HoldRequest is unchanged. 0x1: Set-And-Hold-MAC. HoldRequest is set to value hold (1). 0x2: Set-And-Release-MAC. HoldRequest is set to value release (0).
	12	--	Reserved

Table 542. Table ID 5 - Time Gate Scheduling Table OLSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CONFIG_CHANGE_TIME																																0x0
CONFIG_CHANGE_TIME																																0x4
CONFIG_CHANGE_ERROR																																0x8
CONFIG_CHANGE_ERROR																																0xC
OPER_BASE_TIME																																0x10
OPER_BASE_TIME																																0x14
OPER_CYCLE_TIME																																0x18
OPER_CYCLE_TIME_EXT																																0x1C
--																OPER_CONTROL_LIST_LENGTH																0x20
OPER_TIME_INTERVAL_GE_0																																0x24
--														HR_CB_GE_0		--																0x28
...																																...
OPER_TIME_INTERVAL_GE_M																																0x24+8M
--														HR_CB_GE_M		--																0x28+8M

Table 543. Table ID 5 - Time Gate Scheduling Table OLSE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	CONFIG_CHANGE_TIME	This field specifies the time at which the administrative gate control list is to become active. The time is expressed in units of nanoseconds. The field implements the ConfigChangeTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 ConfigChangeTime parameter, which is the PTP time format, consisting of a 48-bit integer number of seconds and a 32-bit integer number of nanoseconds. Instead of using the PTP time format, the field expresses the time as a 64-bit integer number of nanoseconds. Therefore, if this field is to be queried from an IEEE 802.1Q-2018 managed object, the value must be converted by software from the 64-bit integer number of nanoseconds to the IEEE 802.1Q-2018 PTP time format.
127-64	64	CONFIG_CHANGE_ERROR	Count of the number of times that a configuration change (new schedule) has been requested with the old schedule still running and the requested base time was in the past.
191-128	64	OPER_BASE_TIME	Operational Base Time Indicates the base time, i.e., the start time, of the operational gate control sequence.

Table continues on the next page...

Table 543. Table ID 5 - Time Gate Scheduling Table OLSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			The time is expressed in units of nanoseconds. The time indicated in the field is advanced by the amount of time specified in EaTGSLR/ S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]. The field implements the OperBaseTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 OperBaseTime parameter, which is the PTP time format, consisting of a 48-bit integer number of seconds and a 32-bit integer number of nanoseconds. Instead of using the PTP time format, the field expresses the time as a 64-bit integer number of nanoseconds. Therefore, if this field is to be queried from an IEEE 802.1Q-2018 managed object, the value must be converted by software from the 64-bit integer number of nanoseconds to the IEEE 802.1Q-2018 PTP time format.
223-192	32	OPER_CYCLE_TIME	Operational Cycle Time Indicates the cycle time of the operational gate control list. This is the period of time over which the sequence of operations in a gate control list repeats. Command will be rejected if ADMIN_CYCLE_TIME=0. The time is expressed as an integer number of nanoseconds.
255-224	32	OPER_CYCLE_TIME_EXT	Operational Cycle Time Extension This is the maximum amount of time by which the OPER_CYCLE_TIME is permitted to be extended when a new gate control sequence is being installed. The time is expressed as an integer number of nanoseconds.
271-256	16	OPER_CONTROL_LIST_LENGTH	Operational Control List Length This field indicates the number of entries in the operational gate control list.
287-272	16	--	Reserved
(351 + 64i) - (288 + 64i)	32	OPER_TIME_INTERVAL_GE_i	Operational Time Interval for Gate Entry i, where i = 0, ..., M-1 Specified in nanoseconds. Restrictions (applies to any time interval entries): - Minimum time interval time is the time required to transmit 128 bytes. - Time interval needs to be greater than the advance offset time configured in PTGATOR[ADV_TIME_OFFSET]. - The sum of any 15 consecutive time intervals must be greater than the advance offset time configured in EaTGSLR[MIN_LOOKAHEAD] / S0TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]. - An entries' time interval should be able to transmit the Traffic Classes' MAX_SDU when fusing.
	8	OPER_TC_STATES_GE_i	Operational Traffic Class Gate States for Gate Entry i, where i = 0, ..., M-1 Bit 0: Gate state for traffic class 0. Bit 1: Gate state for traffic class 1. Bit 2: Gate state for traffic class 2. Bit 3: Gate state for traffic class 3. Bit 4: Gate state for traffic class 4. Bit 5: Gate state for traffic class 5.

Table continues on the next page...

Table 543. Table ID 5 - Time Gate Scheduling Table OLSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Bit 6: Gate state for traffic class 6. Bit 7: Gate state for traffic class 7. For each of the traffic classes above, the following values apply: 0b = Gate closed. 1b = Gate open.
	8	--	Reserved
	4	OPER_HR_CB_GE_i	Operational gate operation type (as per 802.1Q) field for gate control list entry i. 0x0: SetGateStates. HoldRequest is unchanged. 0x1: Set-And-Hold-MAC. HoldRequest is set to value hold (1). 0x2: Set-And-Release-MAC. HoldRequest is set to value release (0).
	12	--	Reserved

53.4.2.4.7.2 Rate Policer Table

Table 544. Table ID 10 - Rate Policer Table Common Attributes

TABLE_ID	10
TABLE_VERSION	0
Table Type	Dynamic bounded index table
Table Description	The Rate Policer table contains the rate policer configuration and operational information for a NETC function. There is one Rate Policer table for each ENETC instance and one Rate Policer table for the entire switch. Each table entry contains a set of parameters that defines a single rate policer instance. A rate policer instance controls the amount of traffic that is allowed to go through for a particular received flow of frames. A rate policer instance can be pointed to in the Ingress Port Filter, Ingress Stream, Ingress Stream Filter tables and storm control registers. Table entries used for storm control and table entries used for rate policing are mutual exclusive; these entries cannot be shared between these two distinct functions. Rate policers can be used across switch ports, and streams. In the case of bandwidth violations, a rate policer instance can change the drop resilience parameter associated with each frame and can discard frames on various basis such as the drop resilience of the frame. Table management command operations Add, Delete, Update and Query are supported. This table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. The entries are stored in contiguous manner in a common memory. For this table, each table entry occupies four words. Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) registers; S0RPITMAR[NUM_WORDS] for switch and EaRPITMAR[NUM_WORDS] for ENETC. These allocations can also be changed when the functions are initialized through their respective base function register RPITMAR[NUM_WORDS].
Number of Entries	Maximum number of entries is indicated in RPITCAPR[NUM_ENTRIES]. Number of entries in-use is indicated in RPITOR[NUM_ENTRIES].

Table continues on the next page...

Table 544. Table ID 10 - Rate Policer Table Common Attributes (continued)

Default Reset Behavior	No entries in the table by default.			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update. 0x8 = Add.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	18	4
	FEE_DATA	Functional Enable Element	1	1
	PSE_DATA	Policer State Element	1	1
	STSE_DATA	Statistics Element	84	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x120 = SDU_TYPE specified in CFGE_DATA is out of range.			

Table 545. Table ID 10 - Rate Policer Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--												STSEU	PSEU	FEEU	CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																...
xxx NOT PART OF DATA BUFFER xxx								FEE_DATA								CFGE_DATA																0x18

Table 546. Table ID 10 - Rate Policer Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The FEN bit is cleared and the rate policer instance is disabled. Then, the data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. The user must re-enable the rate policer instance manually after a configuration update by means of a Functional Enable Element update. This Functional Enable update may be done at the same time as the Configuration Element update. Bit 1: FEEU - Functional Enable Element Update. 0b = No update performed to the Functional Enable Element. 1b = The data specified in the FEE_DATA field is written to the Functional Enable Element of the table entry. Bit 2: PSEU - Policer State Element Update. 0b = No update performed to the Policer State Element. 1b = The MR state is reset to 0. Bit 3: STSEU - Statistics Element Update. 0b = No update performed to the Statistics Element. 1b = All counters within the Statistics Element are reset. Bits 15-4: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query.

Table continues on the next page...

Table 546. Table ID 10 - Rate Policer Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 10 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
207-64	144	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.
215-208	8	FEE_DATA[1]	Functional Enable Element Data See FEE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 547. Table ID 10 - Rate Policer Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
STSE_DATA																																0x4
...																																...
STSE_DATA																																0x54
CFGE_DATA																																0x58
...																																...
PSE_DATA								FEE_DATA								CFGE_DATA																0x68

Table 548. Table ID 10 - Rate Policer Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID The Entry ID represents the rate policer instance index with a range of 0 to <RPITCAPR[NUM_ENTRIES]>-1. Values which index outside of the available table entries are reserved and will result in an error. For generic details on the Entry ID field, see Entry ID Management.
703-32	672	STSE_DATA[1]	Statistics Element Data See STSE_DATA_Format.
847-704	144	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.
855-848	8	FEE_DATA[1]	Functional Enable Element Data See FEE_DATA_Format.
863-856	8	PSE_DATA[1]	Policer State Element Data See PSE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 549. Table ID 10 - Rate Policer Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CIR																																0x0
CBS																																0x4
EIR																																0x8
EBS																																0xC
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																--									SDU_TYPE	NDOR	CF	CM	DOY	MREN		0x10

Table 550. Table ID 10 - Rate Policer Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	CIR	Committed Information Rate Expressed in units of 3.725 bits per second, this parameter limits the average rate of frames declared green by the rate policer instance.

Table continues on the next page...

Table 550. Table ID 10 - Rate Policer Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
63-32	32	CBS	Committed Burst Size Expressed in bytes, this is the maximum fill (number of bytes) allowed in the Committed (C) token bucket. This parameter limits the number of bytes allowed in a burst of frames that are declared green by the rate policer instance.
95-64	32	EIR	Excess Information Rate Expressed in units of 3.725 bits per second, this parameter limits the average rate of frames declared yellow by the rate policer instance.
127-96	32	EBS	Excess Burst Size Expressed in bytes, this is the maximum fill (number of bytes) allowed in the Excess (E) token bucket. This parameter limits the number of bytes allowed in a burst of frames that are declared yellow by the rate policer instance.
128	1	MREN	Mark All Frames Red Enable This field enables the rate policer blocking function called "mark all frames red". When the "mark all frames red" feature is enabled, any frames dropped because they were marked red or yellow (if drop on yellow is enabled) will cause all subsequent frames to be marked red and the MR bit to be set to 1 within the Policer State Element. Frames marked "red" are always dropped unless the NDOR field is set to 1, in which case the red frames are not dropped, but their DR is set to 3. The hardware will continue to mark red all frames arriving at this rate policer instance until the MR bit is cleared via an update command to the Policer State Element. 0b = The rate policer blocking function "mark all frames red" feature is disabled. 1b = The rate policer blocking function "mark all frames red" feature is enabled. Not valid when NDOR=1.
129	1	DOY	Drop on Yellow 0b = Frames marked as yellow by the rate policer instance are not dropped. 1b = Frames marked as yellow by the rate policer instance are dropped.
130	1	CM	Color mode 0b = Color blind; pre-color information (encoded in the Drop Resilience (DR)) is not taken into account by the rate policer instance. 1b = Color aware; pre-color information (encoded in the Drop Resilience (DR)) is taken into account by the rate policer instance.
131	1	CF	Coupling flag This field enables the feature of coupling the Committed (C) bucket and Excess (E) bucket. When the buckets are coupled, and the C bucket becomes full (C tokens = CBS), any overflow C tokens will be added to the E bucket, up to the max fill of the E bucket (EBS). 0b = C and E token buckets are not coupled. 1b = C and E token buckets are coupled.

Table continues on the next page...

Table 550. Table ID 10 - Rate Policer Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
132	1	NDOR	No Drop on Red. When this field is set to 1, the red frames from this rate policer instance, are not dropped, but their DR is set to 3. Frames marked red by the flow meter blocking feature "mark all frames red" (enable/disable via the MREN) will also not be dropped. When this field is set to 0, frames marked "red" are always dropped.
134-133	2	SDU_TYPE	Service Data Unit Type This field specifies the type of PDU/SDU (Protocol/Service Data Unit) for a frame received by this rate policer instance when performing flow meter calculations. Overhead values are specified by register PRXSDUOR. 00b = PPDU (Physical Layer PDU). 01b = MPDU (MAC PDU). 10b = MSDU (MAC SDU). All other values reserved.
143-135	9	--	Reserved

Table 551. Table ID 10 - Rate Policer Table FEE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						--									FEN	0x0

Table 552. Table ID 10 - Rate Policer Table FEE_DATA Description

Bits	Width [bits]	Name	Description
0	1	FEN	Functional Enable This field is used to enable/disable a rate policer instance. 0b = The rate policer instance is disabled, but the token buckets and statistics counters and other configuration information are left intact, to allow reading them after a policer instance has been disabled. Frames directed to a policer instance that is disabled are passed through (not discarded). 1b = The rate policer instance is enabled based on the configuration information within the Configuration Element. Enabling a rate policer instance consists of setting the committed (C) token bucket to its full value (which is CBS), setting the excess (E) token bucket to its full value (which is EBS), and clearing all of the statistics counters to 0. Note that the timer function must be configured and enabled before any rate policer instance is enabled. Either the default nanosecond timer or the 1588 timer can be used to generate the required nanosecond resolution time. The default nanosecond timer is configured and enabled by default out of reset. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer

Table continues on the next page...

Table 552. Table ID 10 - Rate Policer Table FEE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			is in normal operation any change to the 1588 timer configuration (for example, TMROFF_H/L), except for TMR_ADD updates, requires that all rate policer instances be disabled.
7-1	7	--	Reserved.

Table 553. Table ID 10 - Rate Policer Table PSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						--									MR	0x0

Table 554. Table ID 10 - Rate Policer Table PSE_DATA Description

Bits	Width [bits]	Name	Description
0	1	MR	Mark Red Flag 0b = Indicates that the rate policer blocking "mark all frames red" function has not been triggered. 1b = Indicates that all frames arriving at this rate policer are marked red by the rate policer blocking "mark all frames red" function.
7-1	7	--	Reserved.

Table 555. Table ID 10 - Rate Policer Table STSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
BYTE_COUNT																																0x0
BYTE_COUNT																																0x4
DROP_FRAMES																																0x8
--																																0xC
DR0_GRN_FRAMES																																0x10
--																																0x14
DR1_GRN_FRAMES																																0x18
--																																0x1C
DR2_YLW_FRAMES																																0x20
--																																0x24
REMARK_YLW_FRAMES																																0x28

Table continues on the next page...

Table 555. Table ID 10 - Rate Policer Table STSE_DATA Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--																																0x2C
DR3_RED_FRAMES																																0x30
--																																0x34
REMARK_RED_FRAMES																																0x38
--																																0x3C
LTS																																0x40
BCI																																0x44
BCS	BCF																															0x48
BEI																																0x4C
BES	BEF																															0x50

Table 556. Table ID 10 - Rate Policer Table STSE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	BYTE_COUNT	Number of bytes received by the rate policer instance. Refer to SDU_TYPE for the SDU type used for byte count updates..
95-64	32	DROP_FRAMES	Number of frames dropped by the rate policer instance.
159-128	32	DR0_GRN_FRAMES	Number of frames marked green with DR=0 by the rate policer instance.
223-192	32	DR1_GRN_FRAMES	Number of frames marked green with DR=1 by the rate policer instance.
287-256	32	DR2_YLW_FRAMES	Number of frames marked yellow with DR=2 by the rate policer instance.
351-320	32	REMARK_YLW_FRAMES	Number of frames re-marked from green to yellow by the rate policer instance. If CM=(color-blind mode), this will equal DR2_YLW_FRAMES.
415-384	32	DR3_RED_FRAMES	Number of frames marked red (DR=3) by the rate policer instance.
479-448	32	REMARK_RED_FRAMES	Number of frames re-marked from green or yellow to red by the rate policer instance. If CM=0 (color-blind mode), this will equal DR3_RED_FRAMES.

Table continues on the next page...

Table 556. Table ID 10 - Rate Policer Table STSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
543-512	32	LTS	Last timestamp. This is the timestamp of the last frame (or periodic update) received by this rate policer instance. Frame timestamps in the receive datapath are applied at the MAC, and use the 32 lsb of the 64 bit 1588 time base, see also the TIMESTAMP field in the receive buffer descriptor.
575-544	32	BCI	Bucket Committed Integer (part) Indicates the integer part (32 bits) of the committed token bucket fill level, expressed in units of bytes.
606-576	31	BCF	Bucket Committed Fractional (part) Indicates the fractional part (31 bits) of the committed token bucket fill level, expressed in units of bytes.
607	1	BCS	Bucket Committed Sign-bit. Indicates the sign-bit value of the committed token bucket fill level. The committed token bucket fill level, expressed in units of bytes, is represented as a 64-bit 2’s complement value (BCI) with a 32-bits integer part and a 31-bits fractional part (BCF). The 2’s complement notation is used as the amount of tokens in the bucket can go negative (effectively borrowing future tokens) when forwarding cut-through frames. In cut-through mode, the amount of tokens required (and consumed initially) in the bucket for considering a frame ‘conforming’, is the initial segment size of the cut-through frame. Tokens then continue to get consumed as the rest of the cut-through frame is received, and then possibly resulting in the bucket going negative. The fractional part is mainly due to the replenishment of the bucket, as the replenishment rate is in units of 3.725 bits per second. Note that a fraction of a token (or byte) means from an algorithm perspective that the token is not considered available in the bucket.
639-608	32	BEI	Bucket Excess Integer (part) Indicates the integer part (32 bits) of the excess token bucket fill level, expressed in units of bytes.
670-640	31	BEF	Bucket Excess Fractional (part) Indicates the fractional part (31 bits) of the excess token bucket fill level, expressed in units of bytes.
671	1	BES	Bucket Excess Sign-bit. Indicates the sign-bit value of the excess token bucket fill level. The excess token bucket fill level, expressed in units of bytes, is represented as a 64-bit 2’s complement value with a 32-bits integer part (BEI) and a 31-bits fractional part (BEF). The 2’s complement notation is used as the amount of tokens in the bucket can go negative (effectively borrowing future tokens) when forwarding cut-through frames. In cut-through mode, the amount of tokens required (and consumed initially) in the bucket for considering a frame ‘conforming’, is the initial segment size of the cut-through frame. Tokens then continue to get consumed as the rest of the cut-through frame is received, and then possibly resulting in the bucket going negative. The

Table continues on the next page...

Table 556. Table ID 10 - Rate Policer Table STSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			fractional part is mainly due to the replenishment of the bucket, as the replenishment rate is in units of 3.725 bits per second. Note that a fraction of a token (or byte) means from an algorithm perspective that the token is not considered available in the bucket.

53.4.2.4.7.3 Ingress Port Filter Table

Table 557. Table ID 13 - Ingress Port Filter Table Common Attributes

TABLE_ID	13
TABLE_VERSION	0
Table Type	Hardware Managed Ternary Match table
Table Description	The Ingress Port Filter table contains a set of filters each capable of classifying incoming traffic using a mix of L2, L3, and L4 parsed and arbitrary field data. As a result of a filter match, several actions can be specified such as on whether to deny or allow a frame, overriding internal QoS attributes associated with the frame and setting parameters for the subsequent frame processing functions (e.g. stream identification, policing, ingress mirroring). Each entry corresponds to a filter. The Ingress Port Filter table is enabled per-port via PIPFCR[EN], when there is at least one entry added for that port. If disabled, or if no match is found, the frame is allowed and passed to the next frame processing function. The Ingress Port Filter entries are added using a precedence value. If a frame matches multiple entries, the entry with the higher precedence is used. The Ingress Port Filter table is implemented using a Ternary Content Addressable Memory (TCAM). TCAM provides deterministic and high-speed lookups, however they consume significant amount of die size area compared to other types of memories, and thus the need to optimize storage is important. As a result, a narrow (48-bit wide), but deep TCAM organization is used, where a table entry can occupy multiple sequential lines of the TCAM. The maximum number of TCAM lines occupied by an Ingress Port Filter entry is 14 lines. With the exception of the first TCAM line, TCAM lines for which all fields are masked out (their corresponding mask field set to all 0s) will not actually exist in the TCAM. However, if only the first line is not completely masked out, TCAM line 4 is added, where the reserved portion of the TCAM line (bits 47:40) are not masked out. As a result, an Ingress Port Filter entry may occupy from 2 lines to 14 lines, depending on how many TCAM lines are entirely don't cares. See IPF Entry Format for the Ingress Port Filter entry implementation format (that is, Ingress Port Filter table entry fields contained in each TCAM line). A maximum of 24 consecutive bytes can be used as payload data for a Ingress Port Filter entry. The per-port parser register PPCR is used to limit the amount of payload data needed, if no entries are added attempting to match the full 24B of payload. If the end of frame is received before the desired amount of payload has been extracted, then the payload data will be padded with zeros up to the next 6B multiple (TCAM line). There is a small opportunity for false matches, where the zero-padded payload matches an entry with actual payload data of zeros. However, since the payload is only padded to the next 6B boundary, the likelihood of false matches is reduced, since it only can false match an entry with zeros as the last few payload bytes, and the payload data ends at that 6B boundary. For example, if 16B of payload is required, and the end of frame happens after extracting 12B of payload, there will be no zero padding, since the payload data extracted ended on a 6B boundary. If the end of frame occurs after extracting 14B, then there will be 4B of zero padding, to complete that TCAM line. In this case, only the first 2B of zero padding will have a valid mask, the last 2B of padding will have a mask of "don't care" since we only care about 16B of payload data.

Table continues on the next page...

Table 557. Table ID 13 - Ingress Port Filter Table Common Attributes (continued)

	Software adds entries using the "Add followed by Query" operation and the Ternary Match Key Element Match Access Method. The PRECEDENCE value is used to insert the entry. Hardware automatically moves entries with lower PRECEDENCE values when adding a new entry. The ENTRY_ID assigned to the entry is returned as part of the response to the Query operation. Software can issue a Search operation to dump entries that have been added along with their ENTRY_ID values. A query operation cannot be performed using the Key Element (KEYE_DATA). If the entry being added is too large for the available memory, an error will be returned. The IPFTMOR per-function register can be used to help determine the current usage. When deleting entries, Software provides the ENTRY_ID of the entry to be deleted. Hardware will find the entry specified, delete all lines associated to the entry, and shift entries up to close the gap (if any). When software performs a Query operation, and requests the Key Element of a ternary match (TCAM) table to be returned, the Key Element information in the Response Data Buffer may not precisely match the data which was used when an Add operation was issued. Ternary match tables perform a lossy compression when storing the key element data within the key element and thus, when querying a ternary match table's key element, software must be aware of the following behavior: for any data bits which had their associated mask bit set to 0 (don't care), the resulting data bit in the query response will be 0 regardless of how that bit was set in the key element data when adding or updating the key element. The precedence value within a Key Element will be returned precisely upon query. The Search Access Method can only be used to dump the Ingress Port Filter table entries for a given function (i.e. match all entries for a given ENETC, or the Switch as a whole). Table management operations Add, Delete, Update and Query are supported.		
Number of Entries	Maximum number of words is indicated in IPFTCAPR[NUM_WORDS]. Number of words in-use is indicated in IPFTMOR[NUM_WORDS]. Note: Port Filter entries can vary in size, from 2 to 14 words.		
Default Reset Behavior	No entries in the table by default.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update.
	Exact Match Key Element Match (KEY_ELEMENT)	No	--
	Ternary Match Key Element Match (KEY_ELEMENT)	Yes	Following table management command operation is supported: 0xC = Add followed by a Query. When adding multiple entries with the same Key Element,

Table continues on the next page...

Table 557. Table ID 13 - Ingress Port Filter Table Common Attributes (continued)

			each entry will be added to the table with a new and unique Entry ID.	
	Search (SEARCH_CRITERIA)	Yes	Only supported when command operation = 0x4 (Query).	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (IPF_EID)	4	4
	ACCESS_KEY	Access Key	212	4
	CFGE_DATA	Configuration Element	8	4
	STSE_DATA	Statistics Element	8	4
	KEYE_DATA	Key Element	212	4
Entry Alignment [bytes]	4			
Entry ID Management	Hardware assigns entry IDs.			
Response STATUS Applicable	Yes			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x150 = HR value not valid. Only checked if command issued from the Switch and FLTFA=0x2 or FLTFA=0x3 0x151 = Entry being added does not fit in table. 0x152 = CFGE_DATA update without STSE_DATA update. 0x154 = RPR set to a reserved value. Only checked if FLTA=0x2. 0x155 = FLTA_TGT is outside valid range and not NULL. Only checked if FLTA>0x0. 0x156 = FLTA=0x3 when command issued from the Switch. 0x157 = FLTFA>0x1 when command issued from an ENETC PF.			

Table 558. Table ID 13 - Ingress Port Filter Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--														STSEU	CFGEU	
ACCESS_KEY																																0x4
...																																...
ACCESS_KEY																																0xD4

Table continues on the next page...

Table 558. Table ID 13 - Ingress Port Filter Table Request Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CFGE_DATA																																0xD8
CFGE_DATA																																0xDC

Table 559. Table ID 13 - Ingress Port Filter Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Whenever the configuration element is updated, the statistic counter must be reset. Note: If CFGEU=1 and STSEU=0, the command will be returned with an error. Bit 1: STSEU - Statistics Element Update. 0b = No update performed to the Statistics Element. 1b = All counters within the Statistics Element are reset. Note: if STSEU=1, CFGEU can be 0 or 1. Bits 15-2: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. 0x1 = Query ENTRY_ID only. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 13 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
1727-32	1696	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. Bits 1695-31: Reserved. If ACCESS_METHOD = 0x2 (Search): Bits 31-0: SEARCH_CRITERIA. See SEARCH_CRITERIA Format. Bits 1695-31: Reserved. If ACCESS_METHOD = 0x3 (Ternary Match Key Element Match): See Request

Table continues on the next page...

Table 559. Table ID 13 - Ingress Port Filter Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			Header Description. Bits 1695-0: KEYE_DATA All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
1791-1728	64	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 560. Table ID 13 - Ingress Port Filter Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS																																0x0
ENTRY_ID																																0x4
KEYE_DATA																																0x8
...																																...
KEYE_DATA																																0xD8
STSE_DATA																																0xDC
STSE_DATA																																0xE0
CFGE_DATA																																0xE4
CFGE_DATA																																0xE8

Table 561. Table ID 13 - Ingress Port Filter Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	STATUS	Status Bits 31-0: RESUME_ENTRY_ID - An Entry ID which can be used as the RESUME_ENTRY_ID value within the ACCESS_KEY of a subsequent search to continue the search. When RESUME_ENTRY_ID is returned as the NULL Entry ID, this indicates that the search was completed successfully and no further iterations are required. Note: A NULL Entry ID is 0xFFFF_FFFF. Note: This field is valid only in responses for commands which use the Search as the Access Method. For all other Access Methods, this field is reserved.

Table continues on the next page...

Table 561. Table ID 13 - Ingress Port Filter Table Response Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			For generic details on the STATUS field, see Status. For generic details on the Search Access Method, see Search Access Method.
63-32	32	ENTRY_ID [1]	Entry ID Entry ID value generated by Hardware, used to access existing entries. Note: A NULL Entry ID is 0xFFFF_FFFF. For generic details on the Entry ID field, see Entry ID Management.
1759-64	1632	KEYE_DATA[1]	Key Element Data See KEYE_DATA Format.
1723-1760	64	STSE_DATA[1]	Statistics Element Data See STSE_DATA Format.
1887-1824	64	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 562. Table ID 13 - Ingress Port Filter Table KEYE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--																PRECEDENCE																0x0
--																																0x04
FRM_ATTR_FLAGS_MASK																FRM_ATTR_FLAGS																0x8
--								SRC_PORT_MASK						SRC_PORT		--				DSCP_MASK						DSCP					0xC
OUTER_VLAN_TCI_MASK																OUTER_VLAN_TCI																0x10
DMAC																																0x14
DMAC_MASK																DMAC																0x18
DMAC_MASK																																0x1C
SMAC																																0x20
SMAC_MASK																SMAC																0x24
SMAC_MASK																																0x28
INNER_VLAN_TCI_MASK																INNER_VLAN_TCI																0x2C
ETHERTYPE_MASK																ETHERTYPE																0x30
--																IP_PROTOCOL_MASK																0x34

Table continues on the next page...

Table 562. Table ID 13 - Ingress Port Filter Table KEYE_DATA Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--																																0x38
--																																0x3C
--																																0x40
IP_SRC_ADDR																																0x44
...																																...
IP_SRC_ADDR																																0x50
--																																0x54
--																																0x58
IP_SRC_ADDR_MASK																																0x5C
...																																...
IP_SRC_ADDR_MASK																																0x68
L4_SRC_PORT_MASK																L4_SRC_PORT																0x6C
--																																0x70
IP_DEST_ADDR																																0x74
...																																...
IP_DEST_ADDR																																0x80
--																																0x84
--																																0x88
IP_DEST_ADDR_MASK																																0x8C
...																																...
IP_DEST_ADDR_MASK																																0x98
L4_DEST_PORT_MASK																L4_DEST_PORT																0x9C
--																																0xA0
PLD_MASK_BYTE_1								PLD_DATA_BYTE_1								PLD_MASK_BYTE_0								PLD_DATA_BYTE_0							0xA4
...																																...
PLD_MASK_BYTE_23								PLD_DATA_BYTE_23								PLD_MASK_BYTE_22								PLD_DATA_BYTE_22							0xD0

Table 563. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description

Bits	Width [bits]	Name	Description
15-0	16	PRECEDENCE	Precedence

Table continues on the next page...

Table 563. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Precedence value of an entry. When adding an entry, the existing entries are walked until an entry is found with a lower or equal Relative Priority value. The new entry will be inserted at that point. In the case of an equal Relative Priority value, the new entry will have precedence over the existing entry.
63-16	48	--	Reserved.
79-64	16	FRM_ATTR_FLAGS	Frame Attribute Flags The frame attribute flags to match. Bit 0: Switch port masquerading (Switch function) 0b = Switch port is not masqueraded. 1b = Switch port is masqueraded. Applicable only if the incoming port is designated as a switch management port. There is the option on per frame basis, to indicate to the switch to process the frame as it was received from a specified switch port other than the switch management port. This option is referred to as the switch port masquerading option. Bit 0: Reserved (ENETC function) Bit 1: Reserved. Bit 2: Outer VLAN Present. The outer VLAN tag is considered present if VLAN classification has deemed the outer VLAN tag valid and the VLAN tag wasn't taken from the port VLAN configuration register. 0b = Outer VLAN is not present. 1b = Outer VLAN is present. Bit 3: Inner VLAN Present. The inner VLAN tag is considered present if VLAN classification has deemed the inner VLAN tag valid and the VLAN tag wasn't taken from the port VLAN configuration register. 0b = Inner VLAN is not present. 1b = Inner VLAN is present. Bits 6-4: Sequence Tag Code. 000b = R-TAG/HSR tag is not present. 001b = 802.1CB draft 2.0 R-TAG is present. 010b = 802.1CB R-TAG is present. 011b = HSR Tag is present. 100b ... 111b = Reserved. Bit 7: IP Header Present. 0b = IP header is not present. 1b = IP header is present.

Table continues on the next page...

Table 563. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Bit 8: IP Version. 0b = IPv4. 1b = IPv6. Bit 9: IPv4 option / IPv6 extension present. 0b = IPv4 option / IPv6 extension is not present. 1b = IPv4 option / IPv6 extension is present. Bits 11-10: L4 Code. 00b = Other L4. The L4 Header is considered as other L4 if it is not one of the following L4 Headers. 01b = TCP header is present. 10b = UDP header is present. 11b = SCTP header is present (not supported for NETC 3.0 and 3.1). Bit 12: Wake-on-LAN Magic Packet Present. 0b = WoL magic packet is not present. 1b = WoL magic packet is present. Note: Valid if IPFCAPR[WOL]=1. Reserved for Switch function. Bits 15-13: Reserved. Note: When adding an entry, all fields and masks must be correctly set, regardless of the value of this field. i.e. if Bit 8 indicates this entry is for IPv4, the IP address masks must be set for the IPv4 bits only.
95-80	16	FRM_ATTR_FLAGS_MASK	Frame Attribute Flags Mask This field is used to mask the FRM_ATR_FLAGS field when determining matches.
101-96	6	DSCP	Differentiated Services Code Point This field is matched against the differentiated services code point (DSCP) of the received frame.
107-102	6	DSCP_MASK	Differentiated Services Code Point Mask This field is used to mask the DSCP field when hardware is matching this entry.
111-108	4	--	Reserved.
116-112	5	SRC_PORT	Source Port ID This field is matched against the Source port ID. Note: this field is reserved for ENETC.
121-117	5	SRC_PORT_MASK	Source Port ID Mask This field is used to mask the SRC_PORT field when hardware is matching this entry. Note: this field is reserved for ENETC.

Table continues on the next page...

Table 563. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
127-122	6	--	Reserved.
143-128	16	OUTER_VLAN_TCI	Outer VLAN Tag Control Information This field is matched against the outer VLAN tag TCI of the received frame. TCI includes 3-bit Priority Code Point (PCP), 1-bit Drop Eligibility (DEI) and 12-bit VLAN ID (VID). This field is defined in network byte order (big-endian). Most significant byte of the TCI is stored at the lowest byte offset of this field.
159-144	16	OUTER_VLAN_TCI_MASK	Outer VLAN Tag Control Information Mask This field is used to mask the OUTER_VLAN_TCI field when hardware is matching this entry.
207-160	48	DMAC	Destination MAC Address This field is matched against the destination MAC address of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the destination MAC address is stored at the lowest byte offset of this field.
255-208	48	DMAC_MASK	Destination MAC Address Mask This field is used to mask the DMAC field when hardware is matching this entry.
303-256	48	SMAC	Source MAC Address This field is matched against the source MAC address of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the source MAC address is stored at the lowest byte offset of this field.
351-304	48	SMAC_MASK	Source MAC Address Mask This field is used to mask the SMAC field when hardware is matching this entry.
367-352	16	INNER_VLAN_TCI	Inner VLAN Tag Control Information This field is matched against the inner VLAN tag TCI of the received frame. TCI includes 3-bit Priority Code Point (PCP), 1-bit Drop Eligibility (DEI) and 12-bit VLAN ID (VID). This field is defined in network byte order (big-endian). Most significant byte of the TCI is stored at the lowest byte offset of this field.
383-368	16	INNER_VLAN_TCI_MASK	Inner VLAN Tag Control Information Mask This field is used to mask the INNER_VLAN_TCI field when hardware is matching this entry.
399-384	16	ETHERTYPE	EtherType This field is matched against the 2-byte EtherType after VLAN tag(s)/R-TAG/HSR tag (if any), of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the EtherType is stored at the lowest byte offset of this field.

Table continues on the next page...

Table 563. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
415-400	16	ETHERTYPE_MASK	EtherType Mask This field is used to mask the ETHERTYPE field when hardware is matching this entry.
423-416	8	IP_PROTOCOL	IP Protocol Indicates which L4 protocol is encapsulated. This field is matched against the IPv4 Protocol or the IPv6 Next Header (upper layer) field of the received frame.
431-424	8	IP_PROTOCOL_MASK	IP Protocol Mask This field is used to mask the IP_PROTOCOL field when hardware is matching this entry.
543-432	112	--	Reserved.
671-544	128	IP_SRC_ADDR	IP Source Address This field is matched against the IP source address of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the IP address is stored at the lowest byte offset of this field. If this IP address defines an IPv6 address, then: Bits 127-0: IPv6 source address. If this IP address defines an IPv4 address, then: Bits 95-0: Reserved. Bits 127-96: IPv4 source address .
735-672	64	--	Reserved.
863-736	128	IP_SRC_ADDR_MASK	IP Source Address Mask This field is used to mask the IP_SRC_ADDR field when hardware is matching this entry. Note: If the IP_SRC_ADDR is intended to match an IPv4 frame, then bits 95-0 of this field must be 0.
879-864	16	L4_SRC_PORT	L4 Source Port This field is matched against the L4 destination port of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the port number is stored at the lowest byte offset of this field.
895-880	16	L4_SRC_PORT_MASK	L4 Source Port Mask This field is used to mask the L4_SRC_PORT field when hardware is matching this entry.

Table continues on the next page...

Table 563. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
927-896	32	--	Reserved.
1055-928	128	IP_DEST_ADDR	IP Destination Address This field is matched against the IP destination address of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the IP address is stored at the lowest byte offset of this field. If this IP address defines an IPv6 address, then: Bits 127-0: IPv6 destination address. If this IP address defines an IPv4 address, then: Bits 95-0: Reserved. Bits 127-96: IPv4 destination address .
1119-1056	64	--	Reserved.
1247-1120	128	IP_DEST_ADDR_MASK	IP Destination Address Mask This field is used to mask the IP_DEST_ADDR field when hardware is matching this entry. Note: If the IP_DEST_ADDR is intended to match an IPv4 frame, then bits 95-0 of this field must be 0.
1263-1248	16	L4_DEST_PORT	L4 Destination Port This field is matched against the L4 destination port of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the port number is stored at the lowest byte offset of this field.
1279-1264	16	L4_DEST_PORT_MASK	L4 Destination Port Mask This field is used to mask the L4_DEST_PORT field when hardware is matching this entry.
1311-1280	32	--	Reserved.
1319-1312	8	PLD_DATA_BYTE_0	Payload Byte 0 This field is matched against the most significant byte (or first byte) of the payload data of the received frame. Payload data starting position in the received frame is dependent at which point of the frame the parser has ended. - When no IP header is present, the payload data starts immediately after the payload EtherType. The payload EtherType is located immediately after the source MAC address or VLAN tag(s)/R-TAG/HSR tag (if any).

Table continues on the next page...

Table 563. Table ID 13 - Ingress Port Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			- When IP header is present and no TCP or UDP header is present, payload data starts immediately after the IP header except when IPv4 options or IPv6 extensions are present. For the latter, payload data starts at the beginning of the IPv4 Options/IPv6 Extensions header. - When TCP or UDP header is present, the payload data starts immediately after the TCP/UDP Ports.
1327-1320	8	PLD_MASK_BYTE_0	Payload Mask for Byte 0 This field is used to mask the PLD_DATA_BYTE_0 field when hardware is matching this entry.
1335-1328	8	PLD_DATA_BYTE_1	Payload Byte 1 This field is matched against the second-most significant byte (or second byte) of the payload data of the received frame.
1343-1336	8	PLD_MASK_BYTE_1	Payload Mask for Byte 1 This field is used to mask the PLD_DATA_BYTE_1 field when hardware is matching this entry.
...	...	...	...
1671-1664	8	PLD_DATA_BYTE_22	Payload Byte 22 This field is matched against the second-least significant byte (or second last byte) of the payload data of the received frame.
1679-1672	8	PLD_MASK_BYTE_22	Payload Mask for Byte 22 This field is used to mask the PLD_DATA_BYTE_22 field when hardware is matching this entry.
1687-1680	8	PLD_DATA_BYTE_23	Payload Byte 23 This field is matched against the least significant byte (or last byte) of the payload data of the received frame.
1695-1688	8	PLD_MASK_BYTE_23	Payload Mask for Byte 23 This field is used to mask the PLD_DATA_BYTE_23 field when hardware is matching this entry.

Table 564. Table ID 13 - Ingress Port Filter Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--									TIMECAPE	HR				CTD	RPR		FLTA		WOLTE	IMIRE	--	FLTFA		ODR	DR		OIPV	IPV					0x0
FLTA_TGT																																0x4

Table 565. Table ID 13 - Ingress Port Filter Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
3-0	4	IPV	Internal Priority Value New IPV to be assigned to frame, if OIPV is set. The least significant 3 bits are used, most significant bit is reserved.
4	1	OIPV	Override Internal Priority Value (IPV) 0b = Don't override frame IPV. 1b = Override frame IPV with value in IPV field of this entry.
6-5	2	DR	Drop Resilience New DR to be assigned to frame, if ODR is set.
7	1	ODR	Override Drop Resilience (DR) 0b = Don't override frame DR. 1b = Override frame DR with value in DR field of this entry.
9-8	2	FLTFA	Filter Forwarding Action (Switch function) 00b = Discard. 01b = Permit. 10b = Redirect frame to switch management port without any frame modification, along with Host Reason metadata set to the value configured in the HR field of this entry. Rate policing function will be applied to the frame if enabled (i.e. FLTA = 01b). Ingress Stream and 802.1Q bridge forwarding functions are by-passed. 11b = Copy frame to switch management port without any frame modification, along with Host Reason metadata set to the value configured in the HR field of this entry. Frame forwarding action not impacted. Filter Forwarding Action (ENETC function) 00b = Discard. 01b = Permit. 10b, 11b = Reserved.
10	1	--	Reserved.
11	1	IMIRE	Ingress Mirroring Enable 0b = No ingress mirroring action specified in this entry. 1b = The frame is mirrored to the mirror destination specified in the IMDCR0 register. Note: Not applicable to ENETC function.
12	1	WOLTE	Wake-on-LAN Trigger Enable 0b = WoL trigger is disabled. 1b = WoL trigger is enabled. When this field is set to 1b (WoL trigger is enabled), FLTA should be set to 11b (setting a pre L2 filtering SI bitmap) and optionally Override Internal Priority Value (IPV) and Override Drop Resilience (DR). Other actions should not be specified. Note: Valid if IPFCAPR[WOL]=1. Reserved for the Switch function.

Table continues on the next page...

Table 565. Table ID 13 - Ingress Port Filter Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
14-13	2	FLTA	Filter Action (Switch function) 00b = No action. 01b = Rate action with the Rate Policer Entry ID (RP_EID) set to the value configured in the FLTA_TGT field of this entry. 10b = Ingress stream identification action where the Ingress Stream Entry ID (IS_EID) is set to the value configured in the FLTA_TGT field of this entry. 11b = Reserved. Filter Action (ENETC function) 00b = No action. 01b = Rate policing action with the Rate Policer Entry ID (RP_EID) set to the value configured in the FLTA_TGT field of this entry. 10b = Ingress stream identification action where the Ingress Stream Entry ID (IS_EID) is set to the value configured in the FLTA_TGT field of this entry. 11b = Setting a pre L2 filtering SI bitmap that will be used by the L2 filtering function to determine the final SI bitmap. The value of pre L2 filtering SI bitmap is set to the value configured in the FLTA_TGT field of this entry.
16-15	2	RPR	Relative Precedent Resolution This field is used to select the final Ingress Stream Entry ID (IS_EID) when more than one table lookup can yield an IS_EID. This field specifies the precedence/priority of this table lookup entry relative to other table lookups that can yield the IS_EID. (IS_EID) Precedence Value for this entry. 00b = Highest precedence. 01b = Precedence is between ISIDKC0CR0 & ISIDKC1CR0 Ingress Stream Identification exact match lookup. Note that ISIDKC0CR0 lookup match has higher precedence value than ISIDKC1CR0's lookup match. 10b = Lowest precedence. 11b = Reserved. Valid if FLTA = 10b.
17	1	CTD	Cut through disable. 0b = No action. 1b = Cut-through is disabled for a frame matching this entry with this bit set. Valid when FLTFA = 01b. Note: Not applicable to ENETC function.
21-18	4	HR	Host Reason. This field specifies the Host Reason metadata when frame is redirected/copied to the switch management port (i.e. FLTFA = 10b or 11b). Value specified has to be a software defined Host Reason (8-15). See Switch Management Port for details. Note: Not applicable to ENETC function.
22	1	TIMECAPE	Timestamp Capture Enable 0b = No change to the timestamp.

Table continues on the next page...

Table 565. Table ID 13 - Ingress Port Filter Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			1b = Timestamp is to be captured; valid if FLTFA = 01b. Note: Not applicable to ENETC function.
31-23	9	--	Reserved.
63-32	32	FLTA_TGT	Target For Selected Filter Action (Switch function) If FLTA = 01b, this is a RP_EID (Rate Policer Table Entry ID). 0x0..RPITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); rate policing function is by-passed Other values are reserved If FLTA = 10b, this is an IS_EID (Ingress Stream Table Entry ID). 0x0..ISITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); stream filtering function is by-passed Other values are reserved Target For Selected Filter Action (ENETC Function) If FLTA = 01b, this is a RP_EID (Rate Policer Table Entry ID). 0x0..RPITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); rate policing function is by-passed Other values are reserved If FLTA = 10b, this is an IS_EID (Ingress Stream Table Entry ID). 0x0..ISITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); stream filtering function is by-passed Other values are reserved If FLTA = 11b, this is a pre L2 filtering SI bitmap. Each bit of the bitmap corresponds to an SI on a given port. Least significant bit of the bitmap corresponds to the smallest SI number. The SIs to which the frame is to be delivered, are identified by having their corresponding bit set to 1 in the bitmap. Bits 0-15: Bit wise setting of bitmap. LSB is the lowest order of SI. Bits 16-31: Reserved.

Table 566. Table ID 13 - Ingress Port Filter Table STSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
MATCH_COUNT																																0x0
MATCH_COUNT																																0x4

Table 567. Table ID 13 - Ingress Port Filter Table STSE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	MATCH_COUNT	Match Count A count of how many times this entry has been matched. In the event where a frame matches multiple entries, only the count of the entry with the highest relative priority will be incremented.

Table 568. Table ID 13 - Ingress Port Filter Table SEARCH_CRITERIA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RESUME_ENTRY_ID																																0x0

Table 569. Table ID 13 - Ingress Port Filter Table SEARCH_CRITERIA Description

Bits	Width [bits]	Name	Description
31-0	32	RESUME_ENTRY_ID	Resume Entry ID When starting a search, pass the NULL Entry ID. To continue a search, pass the RESUME_ENTRY_ID value returned in the CBD Response of the previous search command. Note: A NULL Entry ID is 0xFFFF_FFFF

Table 570. Table ID 13 - Ingress Port Filter Table Entry TCAM Layout

TCAM Line	Ingress Port FIlter Table Entry Fields (6 bytes per line)
1	OUTER_VLAN_TCI, FRM_ATTR_FLAGS, DSCP, SRC_PORT
2	DMAC
3	SMAC
4	IP_PROTOCOL, ETHERTYPE, INNER_VLAN_TCI
5-6	IP_SRC_ADDR (IPv6 address most significant 12 bytes)
7	IP_SRC_ADDR (IPv6 address least significant 4 bytes or IPv4 address), L4_SRC_PORT
8-9	IP_DEST_ADDR (IPv6 address most significant 12 bytes
10	IP_DEST_ADDR (IPv6 address least significant 4 bytes or IPv4 address), L4_DEST_PORT
11	PLD_DATA_BYTE_0, ..., PLD_DATA_BYTE_5
12	PLD_DATA_BYTE_6, ..., PLD_DATA_BYTE_11
13	PLD_DATA_BYTE_12, ..., PLD_DATA_BYTE_17
14	PLD_DATA_BYTE_18, ..., PLD_DATA_BYTE_23

See Table 562 for the description of the fields in the above table.

53.4.2.4.7.4 FDB Table

Table 571. Table ID 15 - FDB Table Common Attributes

TABLE_ID	15
TABLE_VERSION	0
Table Type	Hash Table
Table Description	The FDB table contains forwarding and/or filtering information about MAC addresses. The FDB table is used for MAC learning lookups and MAC forwarding lookups. Each table entry includes information such as a FID and MAC address that may be unicast or multicast and a forwarding destination field containing a port bitmap identifying the associated port(s) with the MAC address. FDB table entries can be static or dynamic. Static entries are added from software whereby dynamic entries are added either by software or by the hardware as MAC addresses are learned in the datapath. The FDB is implemented as a hash table optionally augmented with a CAM for guaranteed FDB entries (see register FDBHTCAPR). Entries are added as a hash entry first (hash space), and if an entry (static or dynamic) is unable to be added due to collision chain length having reached its maximum length, defined in the register HBTCAPR[MAX_COL], it will be added to the CAM and its associated indexed table. An entry will not be added to the hash space or CAM/indexed table if an entry limit has been hit (for example, the number of hash entries in-use across all switch hash tables, has reached the limit specified in the register HTMCAPR[NUM_WORDS])). Table management command operations Add, Delete, Update and Query are supported.
Number of Entries	The memory allotment to store the hash entries is shared across all switch hash tables. The maximum amount of memory words available to store all switch hash tables entries, is indicated in register HTMCAPR[NUM_WORDS]). If the number of memory words in-use to store hash table entries for the switch has reached this limit, the Add operation will fail, regardless of the fill level of the CAM. The number of memory words in-use by the switch for hash table entries is indicated in register HTMOR[AMOUNT]. Note that CAM-based FDB table entries are not counted in register HTMOR[AMOUNT]. The total amount of words available to store the CAM-based FDB table entries, is indicated in register GHTEMCAPR[NUM_WORDS]. The number of FDB entries in-use in the CAM is indicated in register FDBHTOR0[NUM_GENTRIES]. The amount of dynamic entries is limited to the limit configured in the per-port register BPCR[DYN_LIMIT]. When an Add operation is issued, for every port that has its bit set in the Destination Port Bitmap, hardware will check if its maximum number of dynamic entries allowed has been reached (BPCR[DYN_LIMIT]). If any port has reached its limit, then the entry is not added. If the entry is added, the BPOR[NUM_DYN] value for each port that has its bit set, will be incremented. When an Update operation is issued, for every port that has its bit changing from 0 to 1 in the Destination Port Bitmap, hardware will check if its maximum number of dynamic entries allowed has been reached (BPCR[DYN_LIMIT]). If any port has reached its limit, then the entry is not updated. If the entry is updated, the BPOR[NUM_DYN] value for each port that has its bit changing from 0 to 1 will be incremented and for each port that has its bit changing from 1 to 0, will be decremented. The amount of dynamic entries for the entire switch is limited to the limit configured in the register FDBHTMCR[DYN_LIMIT].

Table continues on the next page...

Table 571. Table ID 15 - FDB Table Common Attributes (continued)

	Number of static FDB entries in-use (hash-based and CAM-based entries) is indicated in register FDBHTOR0[STATIC_ENTRIES]. Number of dynamic FDB entries in-use (hash-based and CAM-based entries) is indicated in register FDBHTOR1[DYN_ENTRIES].			
Default Reset Behavior	There are no entries in the table by default.			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update.	
	Exact Match Key Element Match (KEY_ELEMENT)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update. 0x8 = Add. 0xC = Add, followed by a Query.	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	None.	
	Search (SEARCH_CRITERIA)	Yes	Not supported for Add operations. In the event of a search command requiring multiple iterations (see RESUME_ENTRY_ID), the user must not delete any entries in the table between iterations of that search.	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (L2FDB_EID)	4	4
	ACCESS_KEY	Access Key	32	4
	ACTE_DATA	Activity Element	1	1
	KEYE_DATA	Key Element	12	4
	CFGE_DATA	Configuration Element	12	4
Entry Alignment [bytes]	4			
Entry ID Management	Hardware assigns entry IDs.			

Table continues on the next page...

Table 571. Table ID 15 - FDB Table Common Attributes (continued)

Response STATUS Applicable	Yes
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x171 = Failed to add or update and entry because the BPCR[DYN_LIMIT] has been reached. 0x172 = Failed to add entry because the FDBHTMCR[DYN_LIMIT] has been reached. 0x173 = EPORT value not valid. Only checked if (OETEID=0x1 OR CTD=0x1) 0x174 = ET_EID is out of range and not NULL. Only checked if OETEID>0x0 0x175 = Parity error encountered when adding guaranteed entry

Table 572. Table ID 15 - FDB Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--														ACTEU	CFGEU	
ACCESS_KEY																																0x4
...																																...
ACCESS_KEY																																0x20
CFGE_DATA																																0x24
...																																...
CFGE_DATA																																0x2C

Table 573. Table ID 15 - FDB Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bit 1: ACTEU - Activity Element Update. 0b = No update performed to the Activity Element. 1b = If ACT_FLAG is 0, increment ACT_CNT by one. If ACT_FLAG is 1, reset ACT_FLAG to 0 and reset ACT_CNT to 0. Bits 15-2: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved

Table continues on the next page...

Table 573. Table ID 15 - FDB Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. 0x1 = Query ENTRY_ID only. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 15 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
287-32	256	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. Bits 255-32: Reserved If ACCESS_METHOD = 0x1 (Key Element Match): Bits 95-0: KEYE_DATA. See KEYE_DATA Format. Bits 255-96: Reserved If ACCESS_METHOD = 0x2 (Search): Bits 255-0: SEARCH_CRITERIA. SEARCH_CRITERIA Format. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
384-288	96	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 574. Table ID 15 - FDB Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS																																0x0
ENTRY_ID																																0x4
KEYE_DATA																																0x8
...																																...
KEYE_DATA																																0x10
CFGE_DATA																																0x14

Table continues on the next page...

Table 574. Table ID 15 - FDB Table Response Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
...																																...
CFGE_DATA																																0x1C
--																						ACTE_DATA										0x20

Table 575. Table ID 15 - FDB Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	STATUS	Status Bits 31-0: RESUME_ENTRY_ID - An Entry ID which can be used as the RESUME_ENTRY_ID value within the ACCESS_KEY of a subsequent search to continue the search. When RESUME_ENTRY_ID is returned as the NULL Entry ID, this indicates that the search was completed successfully and no further iterations are required. Note: A NULL Entry ID is 0xFFFF_FFFF. Note: This field is valid only in responses for commands which use the Search as the Access Method. For all other Access Methods, this field is reserved. For generic details on the STATUS field, see Status. For generic details on the Search Access Method, see Search Access Method.
63-32	32	ENTRY_ID [1]	Entry ID Entry ID value generated by hardware, used to access existing entries. When an unassigned entry ID is used to access an entry, H/W needs to process the command as if the entry were not found. Note: A NULL Entry ID is 0xFFFF_FFFF. For generic details on the Entry ID field, see Entry ID Management.
159-64	96	KEYE_DATA[1]	Key Element Data See KEYE_DATA_Format.
255-160	96	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.
263-256	8	ACTE_DATA[1]	Activity Element Data See ACTE_DATA_Format.
287-264	24	--	Reserved.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 576. Table ID 15 - FDB Table KEYE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
MAC_ADDR																																0x0
--																MAC_ADDR																0x4
--																				FID												0x8

Table 577. Table ID 15 - FDB Table KEYE_DATA Description

Bits	Width [bits]	Name	Description
47-0	48	MAC_ADDR	MAC Address This field is matched against the destination MAC address of the frame for MAC forwarding lookups and the source MAC address of the frame for MAC learning lookups. This field is defined in network byte order(big-endian). Most significant byte of the MAC address is stored at the lowest byte offset of this field.
63-48	16	--	Reserved.
75-64	12	FID	Filtering ID This field is matched against an identifier assigned to a VLAN or a set of VLANs. This identifier is referred to as a "Filtering ID", and is obtained from an ingress lookup into the VLAN Filter table.
95-76	20	--	Reserved.

Table 578. Table ID 15 - FDB Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--								PORT_BITMAP																								0x0
--																			TIMECAPE	DYNAMIC	CTD	--	IMIRE	EPORT						OET EID		0x4
ET_EID																																0x8

Table 579. Table ID 15 - FDB Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
23-0	24	PORT_BITMAP	Destination Port Bitmap This field identifies the destination port(s) to which the frame is to be forwarded. The destination port(s) are represented as a bitmap, where each bit of the bitmap corresponds to a port on the switch. Least significant bit of the bitmap corresponds to

Table continues on the next page...

Table 579. Table ID 15 - FDB Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			the smallest port number; i.e. bit offset 0 of the bitmap corresponds to port numbered 0, bit offset 1 to port numbered 1, and so on. The port(s) to which the frame are to be forwarded, are identified by having their corresponding bit set to 1 in the bitmap. Valid bits in the bitmap are from 0..SCAPR0[NUM_PORT]-1 Bits beyond the valid range will be ignored.
31-24	8	--	Reserved.
33-32	2	OETEID	Override ET_EID Option This field is used to convey the egress packet processing actions to be applied to packets matching this entry. The egress packet processing actions are specified in the Egress Treatment table, where each table entry contains the egress packet processing actions to be applied to a scope of packets (e.g. packets matching a particular 802.1Q bridge forwarding entry) exiting on a particular egress port of the switch. The means by which one specifies the Egress Treatment table entries to be accessed, is through the Egress Treatment group. Within the hardware, an Egress Treatment group is determined by a base index (first Egress Treatment table entry of the group) and an applicability port bitmap (a bitmap corresponding to all ports of the switch) which indicates (with a 1) which ports have Egress Treatment table entries present. For more details, see Egress Treatment table access. Specifically, this field specifies how to derive the applicability port bitmap of the primary Egress Treatment group being assigned to packets matching this entry. The base index is specified in the ET_EID field of this entry. 00b = No egress packet processing actions specified. Do not override ET_EID (and associated applicability port bitmap). 01b = Single-port Egress Treatment table access. Only one port requires egress packet processing. That port identifier is specified in the EPORT field of this entry. The applicability port bitmap is set with a 1 for the port that is been identified in the EPORT field of this entry. For other ports, the applicability port bitmap is set to 0. The group assigned to packets matching this entry will have a single Egress Treatment table entry, for the port identified in the EPORT field of this entry. 10b = Multi-port packed Egress Treatment table access. Applicability port bitmap is set to the port bitmap specified in PORT_BITMAP field of this entry. The group assigned to packets matching this entry will have an Egress Treatment table entry for each port set to 1 in the applicability port bitmap. 11b = Multi-port absolute Egress Treatment table access. Applicability port bitmap is set with 1 for all ports. The group assigned to packets matching this entry will have Egress Treatment table entries for all ports on the switch.
38-34	5	EPORT	Egress Port This field specifies the identifier of the port requiring egress packet processing when

Table continues on the next page...

Table 579. Table ID 15 - FDB Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			the Override ET_EID Option field is set to 01b (single-port Egress Treatment table access option). The port is expressed as an integer value, and must correspond to a port that is set in the destination port bitmap. Range: {0..SCAPR0[NUM_PORT]-1} Valid if OETEID=1 or CTD=1.
39	1	IMIRE	Ingress Mirroring Enable 0b = No ingress mirroring action specified in this entry. 1b = The frame is mirrored to the mirror destination specified in the IMDCR0 register.
40	1	--	Reserved.
42-41	2	CTD	Cut-Through Disable 00b = Do not override cut-through state. 01b = Disable cut-through for the outgoing port specified in the EPORT. Cut-through is not disabled for the other ports specified in destination port bitmap. 10b = Disable cut-through for all ports specified in the destination port bitmap. 11b = Reserved. Note: Cut-through should be disabled for any outgoing port where its Egress Treatment table entry specifies an Egress Sequence Recovery action. i.e.: Configuration Element Data ESQA field in the Egress Treatment table entry is set to 2.
43	1	DYNAMIC	Static or Dynamic Entry This field determines whether the entry is static or dynamic. 0b = Static entry. 1b = Dynamic entry. This field can be used in the aging software process as part of the decision as to whether the entry can be aged or not. Software can set this field to static or dynamic. When an entry is created by hardware (via the MAC learning function), this field is set to 1 (dynamic entry).
44	1	TIMECAPE	Timestamp Capture Enable when set. 0b = No change to the timestamp. 1b = Timestamp is to be captured.
63-45	19	--	Reserved.
95-64	32	ET_EID	Egress Treatment Table Entry ID This field specifies the index (or base index) to be used when accessing the Egress Treatment table. This field is valid if the Override ET_EID Option field is set to value other than 00b. 0x0..ETTCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL) Other values are reserved

Table 580. Table ID 15 - FDB Table ACTE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																								ACT_FLAG	ACT_CNT							0x0

Table 581. Table ID 15 - FDB Table ACTE_DATA Description

Bits	Width [bits]	Name	Description
6-0	7	ACT_CNT	Activity Counter Counter incremented by software via table management commands when the matching entry does not have its Activity Flag set.
7	1	ACT_FLAG	Activity Flag When hardware matches on an entry, it will set the Activity Flag, indicating it has been used by hardware.

Table 582. Table ID 15 - FDB Table SEARCH_CRITERIA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RESUME_ENTRY_ID																																0x0
KEYE_DATA																																0x4
...																																...
KEYE_DATA																																0xC
CFGE_DATA																																0x10
...																																...
CFGE_DATA																																0x18
--							ACTE_MC	--					CFGE_MC	--							KEYE_MC	ACTE_DATA									0x1C

Table 583. Table ID 15 - FDB Table SEARCH_CRITERIA Description

Bits	Width [bits]	Name	Description
31-0	32	RESUME_ENTRY_ID	Resume Entry ID. When starting a search, pass the NULL Entry ID. To continue a search, pass the RESUME_ENTRY_ID value returned in the CBD Response of the previous search command.

Table continues on the next page...

Table 583. Table ID 15 - FDB Table SEARCH_CRITERIA Description (continued)

Bits	Width [bits]	Name	Description
			Note: A NULL Entry ID is 0xFFFF_FFFF
127-32	96	KEYE_DATA	Key Element Match Data. The Key Element data to be used to match against the Key Elements in the table entries. See KEYE_DATA Format.
223-128	96	CFGE_DATA	Configuration Element Match Data. The Configuration Element data to be used to match against the Configuration Elements in the table entries. See CFGE_DATA Format.
231-224	8	ACTE_DATA	Activity Element Match Data. The Activity Element data to be used to match against the Activity Elements in the table entries. See ACTE_DATA Format.
233-232	2	KEYE_MC	Key Element Match 0x0 = Match Any Criteria. 0x1 = Match KEYE_DATA[FID] provided. 0x2 = Match KEYE_DATA[MAC_ADDR][MULTICAST]. Where the MAC Multicast bit is least significant bit of the most significant byte of the destination MAC address. 0x3 = Match KEYE_DATA[FID] and KEYE_DATA[MAC_ADDR][MULTICAST]. Where the MAC Multicast bit is least significant bit of the most significant byte of the destination MAC address.
239-234	6	--	Reserved.
242-240	3	CFGE_MC	Configuration Element Match Criteria 0x0: Match Any Criteria 0x1: Match CFGE_DATA[DYNAMIC] field 0x2: Match CFGE_DATA[PORT_BITMAP] field 0x3: Match CFGE_DATA[DYNAMIC & PORT_BITMAP] field All other values reserved. Note: If CFGE_DATA[PORT_BITMAP] is a subset of the ENTRY's PORT_BITMAP field, then it is a match.
247-243	5	--	Reserved.
248	1	ACTE_MC	Activity Element Match Criteria 0x0 = Match Any Criteria. 0x1 = Exact match with ACTE_DATA.
255-249	7	--	Reserved.

53.4.2.4.7.5 L2 IPV4 Multicast Filter Table

Table 584. Table ID 16 - L2 IPv4 Multicast Filter Table Common Attributes

TABLE_ID	16		
TABLE_VERSION	0		
Table Type	Hash table		
Table Description	The L2 IPV4 Multicast Filter table contains forwarding and/or filtering information about IPv4 multicast addresses. The L2 IPv4 Multicast Filter table is used for Source Specific Multicast (SSM) and Any Source Multicast (ASM) lookups. Each table entry includes information such as a FID and IPv4 multicast address and a forwarding destination field containing a port bitmap identifying the associated port(s) with the IPv4 multicast address. The lookup is performed using the FID from VLAN Filtering, and the IP addresses of the frame (SA and DA for SSM, DA for ASM). Table management command operations Add, Delete, Update and Query are supported.		
Number of Entries	The memory allotment to store the hash entries is shared across all switch hash tables. The maximum amount of memory words available to store all switch hash tables entries, is indicated in register HTMCAPR[NUM_WORDS]). If the number of memory words in-use to store hash table entries for the switch has reached this limit, the Add operation will fail. The number of memory words in-use by the switch for hash table entries is indicated in register HTMOR[AMOUNT].		
Default Reset Behavior	No entries in the table by default.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update.
	Exact Match Key Element Match (KEY_ELEMENT)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update. 0x8 = Add. 0xC = Add, followed by a Query.
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--
	Search (SEARCH_CRITERIA)	Yes	Not supported for Add operations. In the event of a search command requiring multiple iterations (see RESUME_ENTRY_ID), the user

Table continues on the next page...

Table 584. Table ID 16 - L2 IPv4 Multicast Filter Table Common Attributes (continued)

			must not delete any entries in the table between iterations of that search.	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (L2IPV4_EID)	4	4
	ACCESS_KEY	Access Key	32	4
	ACTE_DATA	Activity Element	1	4
	KEYE_DATA	Key Element	12	4
	CFGE_DATA	Configuration Element	12	4
Entry Alignment [bytes]	4			
Entry ID Management	Hardware assigns entry IDs.			
Response STATUS Applicable	Yes			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x181 = EPORT value not valid. Only checked if (OETEID=0x1 OR CTD=0x1) 0x182 = ET_EID is not NULL or within the valid range. Only checked if OETEID>0x0 0x183 = KEY_TYPE value not valid			

See Table ID 15 - Request Data Buffer Format.

See Table ID 15 - Request Data Buffer Description.

See Table ID 15 - Response Data Buffer Format.

See Table ID 15 - Response Data Buffer Description.

Table 585. Table ID 16 - L2 IPv4 Multicast Filter Table KEYE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--				FID												--												KEY_TYPE				0x0
IPV4_DEST_ADDR																																0x4
IPV4_SRC_ADDR																																0x8

Table 586. Table ID 16 - L2 IPv4 Multicast Filter Table KEYE_DATA Description

Bits	Width [bits]	Name	Description
3-0	4	KEY_TYPE	Key Type 0x0 = IPv4 ASM key. This key type identifies an Any Source Multicast (ASM) table entry, where the key consists of a filtering ID (FID) and destination multicast

Table continues on the next page...

Table 586. Table ID 16 - L2 IPv4 Multicast Filter Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			IPv4 address. 0x1 = IPv4 SSM key. This key type identifies a Source Specific Multicast (SSM) table entry, where the key consists of a filtering ID (FID), IPv4 source address and multicast IPv4 destination address. All other values reserved.
15-4	12	--	Reserved.
27-16	12	FID	Filtering ID This field is matched against an identifier assigned to a VLAN or a set of VLANs. This identifier is referred as to a "Filtering ID", and is obtained from an ingress lookup into the VLAN Filter table.
31-28	4	--	Reserved.
63-32	32	IPV4_DEST_ADDR	IPv4 Destination Address This field is matched against the IP destination address of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the IP address is stored at the lowest byte offset of this field.
95-64	32	IPV4_SRC_ADDR	IPv4 Source Address This field is matched against the IP source address of the received frame. This field is defined in network byte order (big-endian). Most significant byte of the IP address is stored at the lowest byte offset of this field. Valid if KEY_TYPE = 0x1.

See Table ID 15 - CFGE_DATA Format.

See Table ID 15 - CFGE_DATA Description.

See Table ID 15 - ACTE_DATA Format.

See Table ID 15 - ACTE_DATA Description.

Table 587. Table ID 16 - L2 IPv4 Multicast Filter Table SEARCH_CRITERIA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RESUME_ENTRY_ID																																0x0
KEYE_DATA																																0x4
...																																...
KEYE_DATA																																0xC
CFGE_DATA																																0x10
...																																...
CFGE_DATA																																0x18

Table continues on the next page...

Table 587. Table ID 16 - L2 IPv4 Multicast Filter Table SEARCH_CRITERIA Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--							ACTE_MC	--					CFGE_MC			--							KEYE_MC	ACTE_DATA								0x1C

Table 588. Table ID 16 - L2 IPv4 Multicast Filter Table SEARCH_CRITERIA Description

Bits	Width [bits]	Name	Description
31-0	32	RESUME_ENTRY_ID	Resume Entry ID. When starting a search, pass the NULL Entry ID. To continue a search, pass the RESUME_ENTRY_ID value returned in the CBD Response of the previous search command. Note: A NULL Entry ID is 0xFFFF_FFFF
127-32	96	KEYE_DATA	Key Element Match Data. The Key Element data to be used to match against the Key Elements in the table entries. See KEYE_DATA Format.
223-128	96	CFGE_DATA	Configuration Element Match Data. The Configuration Element data to be used to match against the Configuration Elements in the table entries. See CFGE_DATA Format.
231-224	8	ACTE_DATA	Activity Element Match Data. The Activity Element data to be used to match against the Activity Elements in the table entries. See ACTE_DATA Format.
232	1	KEYE_MC	Key Element Match 0x0 = Match Any Criteria. 0x1 = Match KEYE_DATA[FID] provided.
239-233	7	--	Reserved.
242-240	3	CFGE_MC	Configuration Element Match Criteria 0x0: Match Any Criteria 0x1: Match CFGE_DATA[DYNAMIC] field 0x2: Match CFGE_DATA[PORT_BITMAP] field 0x3: Match CFGE_DATA[DYNAMIC & PORT_BITMAP] field All other values reserved. Note: If CFGE_DATA[PORT_BITMAP] is a subset of the ENTRY's PORT_BITMAP field, then it is a match.

Table continues on the next page...

Table 588. Table ID 16 - L2 IPv4 Multicast Filter Table SEARCH_CRITERIA Description (continued)

Bits	Width [bits]	Name	Description
247-243	5	--	Reserved.
248	1	ACTE_MC	Activity Element Match Criteria 0x0 = Match Any Criteria. 0x1 = Exact match with ACTE_DATA.
255-249	7	--	Reserved.

53.4.2.4.7.6 VLAN Filter Table

Table 589. Table ID 18 - VLAN Filter Table Common Attributes

TABLE_ID	18		
TABLE_VERSION	0		
Table Type	Hash table		
Table Description	This table contains configuration and control information for each VLAN configured on the switch. Each VLAN entry includes the VLAN port membership, which FID to use in the FDB lookup, which spanning tree group to use, the egress frame modification actions to apply to a frame exiting form this VLAN, and various configuration and control parameters for this VLAN. The Search Access Method can only be used to dump the VLAN Filter table (match all entries). Table management command operations Add, Delete, Update and Query are supported.		
Number of Entries	The memory allotment to store the hash entries is shared across all switch hash tables. The maximum amount of memory words available to store all switch hash tables entries, is indicated in register HTMCAPR[NUM_WORDS]). If the number of memory words in-use to store hash table entries for the switch has reached this limit, the Add operation will fail. The number of memory words in-use by the switch for hash table entries is indicated in register HTMOR[AMOUNT].		
Default Reset Behavior	No entries in the table by default. There is a default entry configured in registers: VFHTDECR<0..2>		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query.
	Exact Match Key Element Match (KEY_ELEMENT)	Yes	Following table management command operations are supported: 0x1 = Delete.

Table continues on the next page...

Table 589. Table ID 18 - VLAN Filter Table Common Attributes (continued)

			0x2 = Update. 0x4 = Query. 0x8 = Add. 0xC = Add, followed by a Query.	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	Yes	Only supported when command operation = 0x4 (Query). In the event of a search command requiring multiple iterations (see RESUME_ENTRY_ID), the user must not delete any entries in the table between iterations of that search.	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (L2VF_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	KEYE_DATA	Key Element	2	2
	CFGE_DATA	Configuration Element	16	4
Entry Alignment [bytes]	4			
Entry ID Management	Hardware assigns entry IDs.			
Response STATUS Applicable	Yes			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x1A0 = BASE_ET_EID is out of range or MLO is not valid.			

Table 590. Table ID 18 - VLAN Filter Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																...

Table continues on the next page...

Table 590. Table ID 18 - VLAN Filter Table Request Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CFGE_DATA																																0x14

Table 591. Table ID 18 - VLAN Filter Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved.
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. 0x1 = Query ENTRY_ID only. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 18 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. If ACCESS_METHOD = 0x1 (Key Element Match): Bits 15-0: KEYE_DATA. See KEYE_DATA Format. Bits 31-16: Reserved. If ACCESS_METHOD = 0x2 (Search): Bits 31-0: SEARCH_CRITERIA. SEARCH_CRITERIA Format. For generic details on the Access Key field, see Access Key.
191-64	128	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 592. Table ID 18 - VLAN Filter Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS																																0x0
ENTRY_ID																																0x4
--																KEYE_DATA																0x8
CFGE_DATA																																0xC
...																																...
CFGE_DATA																																0x18

Table 593. Table ID 18 - VLAN Filter Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	STATUS	Status Bits 31-0: RESUME_ENTRY_ID - An Entry ID which can be used as the RESUME_ENTRY_ID value within the ACCESS_KEY of a subsequent search to continue the search. When RESUME_ENTRY_ID is returned as the NULL Entry ID, this indicates that the search was completed successfully and no further iterations are required. Note: A NULL Entry ID is 0xFFFF_FFFF. Note: This field is valid only in responses for commands which use the Search as the Access Method. For all other Access Methods, this field is reserved. For generic details on the STATUS field, see Status. For generic details on the Search Access Method, see Search Access Method.
63-32	32	ENTRY_ID [1]	Entry ID Entry ID value generated by Hardware, used to access existing entries. When an unassigned entry ID is used to access an entry, hardware needs to process the command as if the entry were not found. Note: A NULL Entry ID is 0xFFFF_FFFF. For generic details on the Entry ID field, see Entry ID Management.
79:64	16	KEYE_DATA[1]	Key Element Data See KEYE_DATA_Format.
95:80	16	--	Reserved. Alignment padding.
223-96	128	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 594. Table ID 18 - VLAN Filter Table KEYE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																--				VID												0x0

Table 595. Table ID 18 - VLAN Filter Table KEYE_DATA Description

Bits	Width [bits]	Name	Description
11-0	12	VID	VLAN ID This field specifies the VLAN Identifier (VID) encoded as a 12-bit value. This field is matched against the outer VID derived by VLAN classification function or by a change in the outer VID from an stream ingress frame modification action. Note that a change of the outer VID resulting from a stream ingress modification action will overwrite the outer VID derived by the VLAN classification function.
15-12	4	--	Reserved.

Table 596. Table ID 18 - VLAN Filter Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--				STG_ID				PORT_MEMBERSHIP																										0x0
--								IPMFLE	IPMFE	--	MFO		MLO		--				FID														0x4
--								ETA_PORT_BITMAP																										0x8
BASE_ET_EID																																0xC

Table 597. Table ID 18 - VLAN Filter Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
23-0	24	PORT_MEMBERSHIP	Port Membership Bitmap This field specifies the ports that are member of this VLAN. The VLAN membership ports are represented as a bitmap, where each bit of the bitmap corresponds to a port on the switch. Least significant bit of the bitmap corresponds to the smallest port number; i.e. bit offset 0 of the bitmap corresponds to port numbered 0, bit offset 1 to port numbered 1, and so on. The port(s) that are member of this VLAN, are identified by having their corresponding bit set to 1 in the bitmap. Received frames from a port that is not a member of this VLAN will be dropped. A port is not member of this VLAN if its bit in the port membership bitmap field is set to 0. Note: Bits beyond the valid range of switch ports will be ignored.
27-24	4	STG_ID	Spanning Tree Group Member ID This field specifies the spanning tree protocol group to which this VLAN belongs. The

Table continues on the next page...

Table 597. Table ID 18 - VLAN Filter Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			value in this field is expressed as an integer and is used to index the spanning tree protocol per-port state register (BPSTGSR) to obtain the port spanning tree protocol state. The BPSTGSR register maintains spanning tree protocol port states for every spanning tree protocol group. The information retrieved from this register is used to perform spanning tree protocol (STP) port state checking during ingress VLAN filtering processing and egress VLAN filtering processing. In the case of ingress VLAN filtering processing, the source port ID and spanning tree protocol group ID are used to access register (BPSTGSR) to obtain the spanning tree protocol port state. Disabled - Frames will be discarded. Learning - Learn SMAC, but do not forward the frame. Forwarding - Both MAC learning and forwarding functions are executed. In the case of egress VLAN filtering processing, for each port that has its bit set in the destination port bitmap, the destination port ID and spanning tree protocol group ID are used to access register (BPSTGSR) to obtain the spanning tree protocol port state Disabled - Frames will be discarded. Learning - Frames will be discarded. Forwarding - Frames will be forwarded.
31-28	4	--	Reserved.
43-32	12	FID	Filtering ID The Filtering ID (FID) is a locally significant identifier (global to the switch), that is used as a key value when hardware performs a lookup into the FDB table and the L2 IPV4 Multicast Filter table. The FID is used to identify a set of VIDs, which in turn allows sharing of the same address (MAC, IP) between multiple VIDs during lookups into the FDB table and L2 IPV4 Multicast Filter table. This allows the support of the independent and shared learning modes.
47-44	4	--	Reserved.
50-48	3	MLO	MAC Learning Options 000b = Reserved. 001b = Disable MAC learning. MAC learning function is not performed during the forwarding processing. 010b = Hardware MAC learning is performed. 011b = Software MAC learning secure. A MAC learning lookup is performed into the FDB table. If there is no match, no attempt is made to add a new entry, and the frame is redirect to the switch management port. If there is match, and the entry's port number does not match frame ingress port number, the frame is redirected to the switch management port if station move is allowed, otherwise the frame is discarded. 100b = Software MAC learning unsecure. A MAC learning lookup is performed into the FDB table. If there is no match, no attempt is made to add a new entry, and a copy of the frame is sent to the switch management port. If there is match, and the entry's port number does not match frame ingress port number, the frame is copied to the switch

Table continues on the next page...

Table 597. Table ID 18 - VLAN Filter Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			management port if station move is allowed, otherwise the frame is discarded. 101b = Disable MAC learning with SMAC validation. A MAC learning lookup is performed into the FDB table. If there is no match or there is a match but the ingress port is not a member of the FDB entry, the frame is discarded and counted against the bridge port discard count register (BPDCR) with discard reason BPDCRR0[MACLNFDR] set to 1. All other values reserved.
52-51	2	MFO	MAC Forwarding Options 00b = Reserved. 01b = No FDB lookup is performed, the frame is flooded. 10b = FDB lookup is performed, and if there is no match, the frame is flooded. 11b = FDB lookup is performed, and if there is no match, the frame is discarded.
53	1	--	Reserved.
54	1	IPMFE	IP Multicast Filtering Enable This field specifies whether IP multicast filtering is to be performed. IP multicast filtering is accomplished by executing one or two exact match lookups: Source Specific Multicast (SSM) exact match lookup (if at least one entry added) utilizing both source IP address and destination IP address, and if there is no match, followed by Any Source Multicast (ASM) exact match lookup (if at least one entry added) utilizing destination IP address only. If an entry is found, the frame is forwarded according to the matched table entry. If no entry is found, then the frame will be forwarded according to IPMFLE. 0b = No IP multicast filtering is performed. 1b = If the frame is identified as a multicast IP packet, then IP multicast filtering is performed. If the frame is not identified as an IP multicast packet, the IP multicast filtering is not performed. A frame is identified as a multicast IP packet if the destination IP address falls under the following address range: IPv4: 224.0.1.0-238.255.255.255
55	1	IPMFLE	IP Multicast Flooding Enable If IP multicast filtering is performed (IPMFE = 1b, and the frame is identified as a multicast IP packet), and there was no match found, then the frame is forwarded according to this field. 0b = IP Multicast Flooding disabled, frame is discarded. 1b = IP Multicast Flooding enabled, frame is flooded. If IP multicast filtering is disabled (IPMFE = 0b), this field is ignored by hardware.
63-56	8	--	Reserved.

Table continues on the next page...

Table 597. Table ID 18 - VLAN Filter Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
87-64	24	ETA_PORT_BITMAP	Egress Treatment Applicability Port Bitmap This field specifies the Egress Treatment applicability port bitmap. See BASE_ET_EID field for more details. All ports that have their bit set to 1 in the Egress Treatment applicability port bitmap must member of this VLAN. Valid if BASE_ET_EID is not NULL. Note: Bits beyond the valid range of switch ports will be ignored.
95-88	8	--	Reserved.
127-96	32	BASE_ET_EID	Base Egress Treatment Entry ID This field is used to convey the egress packet processing actions to be applied to packets matching this entry. The egress packet processing actions are specified in the Egress Treatment table, where each table entry contains the egress packet processing actions to be applied to a scope of packets (e.g. packets from a particular VLAN) exiting on a particular egress port of the switch. The means by which one specifies the Egress Treatment table entries to be accessed, is through the Egress Treatment group. Within the hardware, an Egress Treatment group is determined by a base index (first Egress Treatment table entry of the group) and an applicability port bitmap (a bitmap corresponding to all ports of the switch) which indicates (with a 1) which ports have Egress Treatment table entries present. For more details, see Egress Treatment table access. Specifically, this field specifies the base index of the secondary Egress Treatment group being assigned to packets matching this entry. The applicability port bitmap is set to the port bitmap specified in ETA_PORT_BITMAP field of this entry. The group assigned to packets matching this entry will have an Egress Treatment table entry for each port set to 1 in the applicability port bitmap. 0x0..ETTCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); egress treatment processing is by-passed Other values are reserved

Table 598. Table ID 18 - VLAN Filter Table SEARCH_CRITERIA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RESUME_ENTRY_ID																																0x0

Table 599. Table ID 18 - VLAN Filter Table SEARCH_CRITERIA Description

Bits	Width [bits]	Name	Description
31-0	32	RESUME_ENTRY_ID	Resume Entry ID When starting a search, pass the NULL Entry ID. To continue a search, pass

Table continues on the next page...

Table 599. Table ID 18 - VLAN Filter Table SEARCH_CRITERIA Description

Bits	Width [bits]	Name	Description
			the RESUME_ENTRY_ID value returned in the CBD Response of the previous search command. Note: A NULL Entry ID is 0xFFFF_FFFF.

53.4.2.4.7.7 ETM Class Queue Table

Table 600. Table ID 22 - ETM Class Queue Table Common Attributes

TABLE_ID	22		
TABLE_VERSION	0		
Table Type	Static bounded index table		
Table Description	The ETM Class Queue table contains class queue configuration and operational information. Each entry corresponds to a class queue on given switch port (multiple class queues per port). After reset all class queues are initialized and enabled by hardware, and thereafter remain operational. Class queues are always present and enabled.		
Number of Entries	Number of entries is equal to the total number of class queues on the switch. Total number of class queues is equal to the number of switch ports (SCAPR0[NUM_PORT]) multiplied by the number of class queues per port (8).		
Default Reset Behavior	After reset all class queues (or entries) are initialized and enabled by hardware, with the CQ2CG_MAP field set equal to the CQ number within the port, so that each class queue is associated with a dedicated congestion group; consecutive class queue numbers contain consecutive congestion group numbers.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update.
	Exact Match Key Element Match (KEY_ELEMENT)	No	--
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--
	Search (SEARCH_CRITERIA)	No	--

Table continues on the next page...

Table 600. Table ID 22 - ETM Class Queue Table Common Attributes (continued)

Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	1	1
	STSE_DATA	Statistics Element	52	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x1E0 = CQ2CG_MAP value out-of-range in update command Note for this table the Access Method field is ignored and always treated as Entry ID Match. No error is generated if an unsupported Access Method is requested. Note for this table the Request Data Buffer must always be 9 bytes, as specified below. An invalid request data buffer length will not return an error, but may cause a corrupted table entry.			

Table 601. Table ID 22 - ETM Class Queue Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--														STSEU	CFGEU	
ACCESS_KEY																																0x4
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						CFGE_DATA										0x8

Table 602. Table ID 22 - ETM Class Queue Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bit 1: STSEU - Statistics Element Update.

Table continues on the next page...

Table 602. Table ID 22 - ETM Class Queue Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			0b = No update performed to the Statistics Element. 1b = All counters (except FRM_CNT) within the Statistics Element are reset. Bits 15-2: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions .
31-28	4	TABLE_VERSION	TID 22 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
71-64	8	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 603. Table ID 22 - ETM Class Queue Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
STSE_DATA																																0x4
...																																...
STSE_DATA																																0x34
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						CFGE_DATA										0x38

Table 604. Table ID 22 - ETM Class Queue Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Note that the Entry ID value represents the Class Queue (CQ) ID to be accessed. ENTRY_ID is encoded as follows: Bits 3-0: CQ number within the port; range is 0..7 Bits 8-4: Switch port ID; range is 0..SCAPR0[NUM_PORT]-1 Bits 31-9: Reserved. Index outside of the above valid range will result in command error. For generic details on the Entry ID field, see Entry ID Management. 1 The ENTRY_ID field is to be present only for commands which preform a query.
447-32	416	STSE_DATA[1]	Statistics Element Data See STSE_DATA_Format.
455-448	8	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 605. Table ID 22 - ETM Class Queue Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						--						CQ2CG_MAP				0x0

Table 606. Table ID 22 - ETM Class Queue Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
3-0	4	CQ2CG_MAP	Class Queue to Congestion Group Mapping This field specifies the congestion group associated with the class queue identified in the Entry ID. This field is expressed as a 4-bit integer with most significant bit reserved, identifying a congestion group within the port to which this class queue is associated with. After reset, this field is set equal to the CQ number within the port, so that each class queue is associated with a dedicated congestion group; consecutive class queue numbers contain consecutive congestion group numbers.
7-4	4	--	Reserved.

Table 607. Table ID 22 - ETM Class Queue Table STSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
REJ_BYTE_CNT																																0x0
REJ_BYTE_CNT																																0x4
REJ_FRM_CNT																																0x8
REJ_FRM_CNT																																0xC
DEQ_BYTE_CNT																																0x10
DEQ_BYTE_CNT																																0x14
DEQ_FRM_CNT																																0x18
DEQ_FRM_CNT																																0x1C
DROP_ML_FRM_CNT																																0x20
DROP_ML_FRM_CNT																																0x24
DROP_MR_FRM_CNT																																0x28
DROP_MR_FRM_CNT																																0x2C
FRM_CNT																																0x30

Table 608. Table ID 22 - ETM Class Queue Table STSE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	REJ_BYTE_CNT	Reject Byte Count This is a cumulative count of all bytes in frames rejected in this class queue, due to tail drop. This count does not include the OAL specified in the associated ETM class scheduler, but does include the effect of any frame modification (e.g. VLAN tag header insertion and/or removal), and CRC append, that may be applied by the Ethernet Tx I/F.
127-64	64	REJ_FRM_CNT	Reject Frame Count This is a cumulative count of all frames rejected in this class queue, due to tail drop.
191-128	64	DEQ_BYTE_CNT	Dequeue Byte Count This is a cumulative count of the number of bytes dequeued from this class queue. This count does not include the OAL specified in the associated ETM class scheduler, but does include the effect of any frame modification (e.g. VLAN tag header insertion and/or removal), and CRC append, that may be applied by the Ethernet Tx I/F.
255-192	64	DEQ_FRM_CNT	Dequeue Frame Count This is a cumulative count of the number of frames dequeued from this class queue.
319-256	64	DROP_ML_FRM_CNT	Dropped Frames, Memory Lost Count of frames dropped from class queue with internal memory loss. Drop due to

Table continues on the next page...

Table 608. Table ID 22 - ETM Class Queue Table STSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			frame descriptor pointer error, the queue is still intact, but the internal memory units storing this frame are now lost.
383-320	64	DROP_MR_FRM_CNT	Dropped Frames, Memory Recovered Count of frames dropped from class queue with memory recovered. Drop due to frame descriptor error with pointer intact, the class queue is still intact, and the internal memory units storing this frame will be recovered.
415-384	32	FRM_CNT	Frame Count Number of frames currently queued on this queue.

53.4.2.4.7.8 ETM Class Scheduler Table

Table 609. Table ID 23 - ETM Class Scheduler Table Common Attributes

TABLE_ID	23		
TABLE_VERSION	0		
Table Type	Static bounded index table		
Table Description	The ETM Class Scheduler table contains class queue scheduling configuration information. Each entry corresponds to a class queue scheduler on given switch port (there is one scheduler per port). After reset all class queue schedulers are initialized and enabled by hardware, and thereafter remain operational. Class schedulers are always present and enabled.		
Number of Entries	Number of entries is equal to the total number of class queue schedulers which is equal to the number of switch ports (SCAPR0[NUM_PORT], as there is one class queue scheduler per switch port.		
Default Reset Behavior	After reset all class queue schedulers (or entries) are initialized and enabled by hardware, with the default that all class queues within a port are treated as strict priority (CQ_ASSG field set to zero).		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x2 = Update. 0x4 = Query.
	Exact Match Key Element Match (KEY_ELEMENT)	No	--
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--

Table continues on the next page...

Table 609. Table ID 23 - ETM Class Scheduler Table Common Attributes (continued)

	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (ETMCLS_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	12	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. Note for this table the Access Method field is ignored and always treated as Entry ID Match. No error is generated if an unsupported Access Method is requested. Note for this table the Request Data Buffer must always be 20 bytes, as specified below. An invalid request data buffer length will not return an error, but may cause a corrupted table entry.			

Table 610. Table ID 23 - ETM Class Scheduler Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																0xC
CFGE_DATA																																0x10

Table 611. Table ID 23 - ETM Class Scheduler Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update.

Table continues on the next page...

Table 611. Table ID 23 - ETM Class Scheduler Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 23 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
159-64	96	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 612. Table ID 23 - ETM Class Scheduler Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
CFGE_DATA																																0x4
...																																0x8
CFGE_DATA																																0xC

Table 613. Table ID 23 - ETM Class Scheduler Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Note that the Entry ID value represents the ETM Class Scheduler instance and is encoded as follows: Bits 4-0: Switch Port ID ; range is 0..SCAPR0[NUM_PORT]-1. Bits 31-5: Reserved. Values outside of the above range will result in an error. For generic details on the Entry ID field, see Entry ID Management.
127-32	96	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 614. Table ID 23 - ETM Class Scheduler Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--					OAL											--												CQ_ASSG				0x0
WBFS_WEIGHT_3								WBFS_WEIGHT_2								WBFS_WEIGHT_1								WBFS_WEIGHT_0								0x4
WBFS_WEIGHT_7								WBFS_WEIGHT_6								WBFS_WEIGHT_5								WBFS_WEIGHT_4								0x8

Table 615. Table ID 23 - ETM Class Scheduler Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
3-0	4	CQ_ASSG	Class Queue Assignment Class queue assignment to scheduler inputs. The class queue scheduler as 16 inputs numbered from 0 to 15. Input 0 to 7 are weighted fair whereby input 8 to 15 are strict priority. Strict priority input with a larger numeric value will be served before another with a smaller numeric value, Class queue 7 can be assigned to any of the scheduler inputs numbered from 8 to 15. Preceding queues are then assigned automatically in a contiguous downwards manner till the maximum number of class queues is reached (i.e. 8). For example, when class queue 7 is assigned to input 15 (the default value), all class queues (0 to 7) within the channel are treated as strict priority. When class queue 7 is assigned to input 7, all class queues (0 to 7) are treated as weighed fair queueing. Any assignment of the class queue 7 in between inputs 8 and 15 will result in some queues as strict priority and some with weighed fair queueing. If the assignment is to an input numerically lower than 7, then the behavior is that of the input 15 setting. Assignment of CQ 7 (CQ_ASSG) is coded as follows: 0: Scheduler input 15 1: Scheduler input 14 2: Scheduler input 13

Table continues on the next page...

Table 615. Table ID 23 - ETM Class Scheduler Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			3: Scheduler input 12 4: Scheduler input 11 5: Scheduler input 10 6: Scheduler input 9 7: Scheduler input 8 8: Scheduler input 7 After reset, this field is initialized to all 0.
15-4	12	--	Reserved
26-16	11	OAL	Overhead Accounting Length This value is an 11-bit unsigned number, and is added to the actual length of each frame when performing class scheduler WBFS calculations. After reset this field is set to 0. The overall effect of the frame modification actions that will be applied to each frame is automatically added to the actual frame length, and thus does not need to be accounted for in the OAL value.
31-27	5	--	Reserved
39-32	8	WBFS_WEIGHT_0	Weight for Scheduler Input 0 Weight for scheduler input 0 in the channel being configured. These weights are used by the weighted scheduling (WBFS) algorithm in the ETM class schedulers. Each 8-bit weight code is formatted as Bits 7-3: weight code y value Bits 2-0: weight code x value And the formula for converting weight code to effective weight is: $(2^x)/(1-(y/64))$. After reset, this field is initialized to all 0.
47-40	8	WBFS_WEIGHT_1	Weight for scheduler input 1 Weight for scheduler input 1 in the channel being configured. These weights are used by the weighted scheduling (WBFS) algorithm in the ETM class schedulers. Each 8-bit weight code is formatted as described in the WBFS_WEIGHT_0 field. After reset, this field is initialized to all 0.
55-48	8	WBFS_WEIGHT_2	Weight for scheduler input 2 Weight for scheduler input 2 in the channel being configured. These weights are used by the weighted scheduling (WBFS) algorithm in the ETM class schedulers. Each 8-bit weight code is formatted as described in the WBFS_WEIGHT_0 field. After reset, this field is initialized to all 0.
63-56	8	WBFS_WEIGHT_3	Weight for scheduler input 3 Weight for scheduler input 3 in the channel being configured. These weights are used by the weighted scheduling (WBFS) algorithm in the ETM class schedulers. Each 8-bit weight code is formatted as described in the WBFS_WEIGHT_0 field. After reset, this field is initialized to all 0.

Table continues on the next page...

Table 615. Table ID 23 - ETM Class Scheduler Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
71-64	8	WBFS_WEIGHT_4	Weight for scheduler input 4 Weight for scheduler input 4 in the channel being configured. These weights are used by the weighted scheduling (WBFS) algorithm in the ETM class schedulers. Each 8-bit weight code is formatted as described in the WBFS_WEIGHT_0 field. After reset, this field is initialized to all 0.
79-72	8	WBFS_WEIGHT_5	Weight for scheduler input 5 Weight for scheduler input 5 in the channel being configured. These weights are used by the weighted scheduling (WBFS) algorithm in the ETM class schedulers. Each 8-bit weight code is formatted as described in the WBFS_WEIGHT_0 field. After reset, this field is initialized to all 0.
87-80	8	WBFS_WEIGHT_6	Weight for scheduler input 6 Weight for scheduler input 6 in the channel being configured. These weights are used by the weighted scheduling (WBFS) algorithm in the ETM class schedulers. Each 8-bit weight code is formatted as described in the WBFS_WEIGHT_0 field. After reset, this field is initialized to all 0.
95-88	8	WBFS_WEIGHT_7	Weight for scheduler input 7 Weight for scheduler input 7 in the channel being configured. These weights are used by the weighted scheduling (WBFS) algorithm in the ETM class schedulers. Each 8-bit weight code is formatted as described in the WBFS_WEIGHT_0 field. After reset, this field is initialized to all 0.

53.4.2.4.7.9 Ingress Stream Identification Table

Table 616. Table ID 30 - Ingress Stream Identification Table Common Attributes

TABLE_ID	30
TABLE_VERSION	0
Table Type	Hash table
Table Description	The Ingress Stream Identifier table contains a set of entries each capable of classifying incoming traffic using a mix of L2 parsed and arbitrary data fields, into streams. As a result of an entry match, an Ingress Stream Entry ID is provided, which is a locally significant identifier (global to the switch), that identifies a stream within an ENETC instance or the switch. The Ingress Stream Identification table can be used to support the IEEE 802.CB stream identification functions. The Ingress Stream Identifier table is an exact match table where four logical tables can be defined using four different L2 parsed and arbitrary data fields key (key construction types). The key construction type (defined in registers) is included in the key element of an entry, to identify to which of the four logical tables the entry belongs to. The datapath can perform two exact match lookups, each against different Ingres Stream Identifier logical table. The two logical tables (or key construction type) to be used is configurable on per port basis. Table management command operations Add, Delete, Update and Query are supported.

Table continues on the next page...

Table 616. Table ID 30 - Ingress Stream Identification Table Common Attributes (continued)

Number of Entries	The memory allotment to store the hash entries is shared across all of the (ENETC instance/ switch) function hash tables. The maximum amount of memory words available to store all of the function hash tables entries, is indicated in the register HTMCAPR[NUM_WORDS]). If the number of memory words in-use to store hash table entries for the function has reached this limit, the Add operation will fail. The number of memory words in-use by the function for hash table entries is indicated in the register HTMOR[AMOUNT].			
Default Reset Behavior	No entries in the table by default.			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query.	
	Exact Match Key Element Match (KEY_ELEMENT)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x8 = Add. 0xC = Add, followed by a Query.	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	Yes	Only supported when command operation = 0x4 (Query). In the event of a search command requiring multiple iterations (see RESUME_ENTRY_ID), the user must not delete any entries in the table between iterations of that search.	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (ISI_EID)	4	4
	ACCESS_KEY	Access Key	20	4
	CFGE_DATA	Configuration Element	4	4
	KEYE_DATA	Key Element	20	4
Entry Alignment [bytes]	4			
Entry ID Management	Hardware assigns entry IDs.			

Table continues on the next page...

Table 616. Table ID 30 - Ingress Stream Identification Table Common Attributes (continued)

Response STATUS Applicable	Yes
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x260 = Port ID or specified in KEYE_DATA is out of range. Valid range of values for Port ID are {0..CAPR0[NUM_LINKS]-1}. KEY_TYPE for ENETC is not valid. 0x261 = IS_EID in invalid. Valid range of values for IS_EID are {0.. ISITCAPR[NUM_ENTRIES]-1} and not NULL IS_EID.

Table 617. Table ID 30 - Ingress Stream Identification Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGEU	
KEYE_DATA																																0x4
...																																...
KEYE_DATA																																0x14
CFGE_DATA																																0x18

Table 618. Table ID 30 - Ingress Stream Identification Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. 0x1 = Query ENTRY_ID only. All other values reserved. For generic details on the Query Actions code point, see Query Actions.

Table continues on the next page...

Table 618. Table ID 30 - Ingress Stream Identification Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
31-28	4	TABLE_VERSION	TID 30 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
191-32	160	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. Bits 159-32: Reserved If ACCESS_METHOD = 0x1 (Key Element Match): Bits 159-0: KEYE_DATA. See KEYE_DATA Format. If ACCESS_METHOD = 0x2 (Search): Bits 31-0: SEARCH_CRITERIA. SEARCH_CRITERIA Format. Bits 159-32: Reserved All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
223-192	32	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 619. Table ID 30 - Ingress Stream Identification Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS																																0x0
ENTRY_ID																																0x4
KEYE_DATA																																0x8
...																																...
KEYE_DATA																																0x18
CFGE_DATA																																0x1C

Table 620. Table ID 30 - Ingress Stream Identification Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	STATUS	STATUS Bits 31-0: RESUME_ENTRY_ID - An Entry ID which can be used as the RESUME_ENTRY_ID value within the ACCESS_KEY of a subsequent search to

Table continues on the next page...

Table 620. Table ID 30 - Ingress Stream Identification Table Response Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			continue the search. When RESUME_ENTRY_ID is returned as the NULL Entry ID, this indicates that the search was completed successfully and no further iterations are required. This field is valid only in responses for operations which use the Search as the Access Method. For all other Access Methods, this field is reserved. Note: A NULL Entry ID is 0xFFFF_FFFF For generic details on the STATUS field, see Status. For generic details on the Search Access Method, see Search Access Method.
63-32	32	ENTRY_ID [1]	Entry ID When an unassigned entry ID is used to access an entry, hardware needs to process the command as if the entry were not found. For generic details on the Entry ID field, see Entry ID Management.
223-64	160	KEYE_DATA[1]	Key Element Data See KEYE_DATA_Format.
255-224	32	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 621. Table ID 30 - Ingress Stream Identification Table KEYE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--																								SPM	SRC_PORT_ID						KEY_TYPE		0x0
FRAME_KEY																																0x4
...																																...
FRAME_KEY																																0x10

Table 622. Table ID 30 - Ingress Stream Identification Table KEYE_DATA Description

Bits	Width [bits]	Name	Description
1-0	2	KEY_TYPE	Key Type Switch Function 00b = Key construction is specified in ISIDKC0CR0 (header) and ISIDKC0PFbCR (payload), where 'b' represents the payload field. 01b = Key construction is specified in ISIDKC1CR0 (header) and ISIDKC1PFbCR (payload), where 'b' represents the payload field. 10b = Key construction is specified in ISIDKC2CR0 (header) and ISIDKC2PFbCR

Table continues on the next page...

Table 622. Table ID 30 - Ingress Stream Identification Table KEYE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			(payload), where 'b' represents the payload field. 11b = Key construction is specified in ISIDKC3CR0 (header) and ISIDKC3PFbCR (payload), where 'b' represents the payload field. ENETC Function 00b = Key construction is specified in ISIDKC0CR0 (header) and ISIDKC0PFbCR (payload), where 'b' represents the payload field. 01b = Key construction is specified in ISIDKC1CR0 (header) and ISIDKC1PFbCR (payload), where 'b' represents the payload field. All other values reserved.
6-2	5	SRC_PORT_ID	Source Port ID Switch Function Switch port ID expressed as an integer value to be used in key construction if PORTP is set to 1 in key construction register. ENETC Function Reserved. This field is not used for ENETC function.
7	1	SPM	Source Port Masquerading Switch port masquerading to be used in key construction if SPMP is set to 1 in key construction register. Switch Function 0b = Match frames from switch port(s). 1b = Match frame from switch management port(s) that has switch port masquerading. ENETC Function Reserved. This field is not used for ENETC function.
31-8	24	--	Reserved.
159-32	128	FRAME_KEY	Frame portion of the Key Key format is specified by the key construction registers ISIDKCaCR0 (frame header) and ISIDKCaPFbCR (frame payload), where ‘a’ represents the key construction rule ID and ‘b’ represents the payload field. The key entered has to be in packed form in the order they appear in the frame with each field formatted in big endian format (order they are received from the link). Maximum key size for the frame portion is 16 bytes. If the constructed key is less than 16 bytes then rest of the bytes have to padded with zeros (i.e. zeros must be entered in this field for the remaining bytes). If the key constructed is larger than 16 bytes, then this will result in no match.

Table 623. Table ID 30 - Ingress Stream Identification Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
IS_EID																																0x0

Table 624. Table ID 30 - Ingress Stream Identification Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	IS_EID	Ingress Stream Entry ID (IS_SID) This field specifies the index to use for accessing the Ingress Stream table. 0x0..ISITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); stream filtering function is by-passed Other values are reserved

Table 625. Table ID 30 - Ingress Stream Identification Table SEARCH_CRITERIA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RESUME_ENTRY_ID																																0x0

Table 626. Table ID 30 - Ingress Stream Identification Table SEARCH_CRITERIA Description

Bits	Width [bits]	Name	Description
31-0	32	RESUME_ENTRY_ID	Resume Entry ID. When starting a search, pass the NULL Entry ID. To continue a search, pass the RESUME_ENTRY_ID value returned in the CBD Response of the previous search operation. Note: A NULL Entry ID is 0xFFFF_FFFF

53.4.2.4.7.10 Ingress Stream Table

Table 627. Table ID 31 - Ingress Stream Table Common Attributes

TABLE_ID	31
TABLE_VERSION	0
Table Type	Dynamic bounded index table
Table Description	The Ingress Stream table contains configuration and operational information about streams identified by the stream identification function. Each entry includes the actions (or functions) that are to be applied to frames belonging to a particular stream, such as: • Whether a frame is accepted or discarded • Specifies the PSFP (IEEE 802.1Qci) filtering and policing actions. To support a PSFP (IEEE 802.1Qci) stream filtering function that uses the 3-bit PCP value (priority) of the received frame and the stream handle (i.e. Ingress Stream Entry ID) to select a stream filter, an additional lookup (i.e. Ingress Stream Filter table lookup) can be specified in the Ingress Stream table entry. • FREF (IEEE 802.1CB) sequence generation function (switch only) • Stream forwarding function; the 802.1Q bridge forwarding function is by-passed, instead the Ingress Stream table provides the port(s) to which the frame should be output to (switch only) • Ingress frame modification in addition to adding a sequence tag (switch only)

Table continues on the next page...

Table 627. Table ID 31 - Ingress Stream Table Common Attributes (continued)

<table>
<tr><td></td><td colspan="4">• Egress packet processing actions to be applied to packets received on this specific stream; typically used along with the stream forwarding function. (switch only)

The Ingress Stream table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. The entries are stored in contiguous manner in the common memory. Ingress Stream Entry ID (IS_EID) is used as an index to an entry in this table. For this table, each table entry occupies one word. Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) registers; S0ISITMAR[NUM_WORDS] for switch and EaISITMAR[NUM_WORDS] for ENETC. These allocations can also be changed when the functions are initialized through their respective base function registers ISITMAR[NUM_WORDS]. Table management command operations Add, Delete, Update and Query are supported.</td></tr>
<tr><td>Number of Entries</td><td colspan="4">Maximum number of entries is indicated in ISITCAPR[NUM_ENTRIES]. Number of entries in-use is indicated in ISITOR[NUM_ENTRIES].</td></tr>
<tr><td>Default Reset Behavior</td><td colspan="4">No entries in the table by default.</td></tr>
<tr><td rowspan="5">Access Method (ACCESS_METHOD)</td><td>Access Method</td><td>Supported</td><td colspan="2">Restrictions</td></tr>
<tr><td>Entry ID Match (ENTRY_ID)</td><td>Yes</td><td colspan="2">Following table management command operations are supported.

0x1 = Delete.
0x2 = Update.
0x4 = Query.
0x8 = Add.</td></tr>
<tr><td>Exact Match Key Element Match (KEY_ELEMENT)</td><td>No</td><td colspan="2">--</td></tr>
<tr><td>Ternary Match Key Element Match (KEY_ELEMENT)</td><td>No</td><td colspan="2">--</td></tr>
<tr><td>Search (SEARCH_CRITERIA)</td><td>No</td><td colspan="2">--</td></tr>
<tr><td rowspan="4">Data Buffer Component Attributes</td><td>Name</td><td>Description</td><td>Size [bytes]</td><td>Alignment [bytes]</td></tr>
<tr><td>ENTRY_ID</td><td>Entry Id (IS_EID)</td><td>4</td><td>4</td></tr>
<tr><td>ACCESS_KEY</td><td>Access Key</td><td>4</td><td>4</td></tr>
<tr><td>CFGE_DATA</td><td>Configuration Element</td><td>38</td><td>4</td></tr>
<tr><td>Entry Alignment [bytes]</td><td colspan="4">4</td></tr>
<tr><td>Entry ID Management</td><td colspan="4">Software assigns entry IDs.</td></tr>
<tr><td>Response STATUS Applicable</td><td colspan="4">No</td></tr>
</table>

Table continues on the next page...

Table 627. Table ID 31 - Ingress Stream Table Common Attributes (continued)

ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x270 = Option specified in one or more of the following fields is not valid – FA, CTD or ISQA, SDU_TYPE. Note that CTD and ISQA are checked for Switch function only. 0x271 = One or more of following: - Entry IDs are not in valid range or Entry ID is not NULL. - Check valid ranges specified for these Entry IDs in Ingress Stream table entry – RP_EID, SGI_EID, ISQ_EID, ET_EID or EPORT. Note that ISQ_EID, ET_EID are checked for Switch function only. - ET_EID is checked if (FA =010b .. 101b) & (OETEID!=0), - EPORT is checked if (FA = 010b .. 101b) & (OETEID= 0x1 OR CTD= 0x1) - HR is checked if FA = 001b, 100b, or 101b. HR specified cannot be 0x0. 0x272 = FM_EID format or index is out of range. Check performed is as follows: – FM_EID format option type is invalid. - FM_EID format is option 1 and the Index is out of range and not NULL, or FM_EID format is option 2 and VUDA or SQTA is out of range.

Table 628. Table ID 31 - Ingress Stream Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																...
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																CFGE_DATA																0x2C

Table 629. Table ID 31 - Ingress Stream Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved.

Table continues on the next page...

Table 629. Table ID 31 - Ingress Stream Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 31 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Table ID 31 Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
367-64	304	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA_Format

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 630. Table ID 31 - Ingress Stream Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
CFGE_DATA																																0x4
...																																...
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																CFGE_DATA																0x28

Table 631. Table ID 31 - Ingress Stream Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Valid range of IS_EID:{0.. ISITCAPR[NUM_ENTRIES]-1}. For generic details on the Entry ID field, see Entry ID Management.
335-32	304	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 632. Table ID 31 - Ingress Stream Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--			SDU_TYPE		FA			HR				OSGI	ORP	ISQA		SPPD	--	TIMECAPE	IMIRE	ODR	DR		OIPV	IPV			--				SFE	0x0
CTD		OETEID			EPORT					IFME_LEN_CHANGE						MSDU																0x4
ISQ_EID																																0x8
RP_EID																																0xC
SGI_EID																																0x10
IFM_EID																																0x14
ET_EID																																0x18
ISC_EID																																0x1C
--																EGRESS_PORT_BITMAP																0x20
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																SI_MAP																0x24

Table 633. Table ID 31 - Ingress Stream Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
0	1	SFE	Stream Filtering Enable To support a PSFP (IEEE 802.1Qci) stream filtering function that uses the 3-bit PCP value (priority) of the received frame and the stream handle (i.e. Ingress Stream Entry ID) to select a stream filter, an additional lookup (i.e. Ingress Stream Filter table lookup) must be performed to obtain the PSFP stream filter information. 0b = The Ingress Stream Filter table lookup is by-passed (or not required). 1b = The Ingress Stream Filter table lookup is performed using as a key, the Ingress Stream Entry ID and PCP (extracted from the received frame outer VLAN tag). If the received frame has no VLAN tag (determined from VLAN classification output; i.e. if the outer VLAN tag is not valid or that VLAN tag data is taken from port outer VLAN configuration register), this additional lookup is not performed.
3-1	3	--	Reserved.
7-4	4	IPV	Internal Priority Value New Internal Priority Value (IPV) to be assigned to the frame, if OIPV is set to 1. The least significant 3 bits are used, most significant bit is reserved.
8	1	OIPV	Override Internal Priority Value (IPV) 0b = Don't override frame IPV. 1b = Override frame IPV with value configured in IPV field of this entry.

Table continues on the next page...

Table 633. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
10-9	2	DR	Drop Resilience New Drop Resilience (DR) to be assigned to the frame, if ODR is 1.
11	1	ODR	Override Drop Resilience (DR) 0b = Do not override frame DR. 1b = Override frame DR with value configured in DR field of this entry.
12	1	IMIRE	Ingress Mirroring Enable 0b = No ingress mirroring action specified in this entry. 1b = Frame is mirrored to the mirror destination specified by IMDCR0 register. Note: Not applicable to ENETC function
13	1	TIMECAPE	Timestamp Capture Enable 0b = No change to the timestamp. 1b = Timestamp is to be captured. Note: Not applicable to ENETC function
14	1	--	Reserved.
15	1	SPPD	Source Port Pruning Disable. This field is used to enable/disable source port pruning. Source port pruning ensures that frames do not get forwarded back to the port they originated from. Once the final destination ports have been determined, source pruning when enabled, will set the port from which the frame originated from, to 0 in the destination port bitmap. This field is valid only if the FA field of this entry is set 010b or 100b (stream forwarding) and will overwrite any previous configuration/setting for source pruning. 0b = Source port pruning enabled. 1b = Source port pruning disabled. Note: Not applicable to ENETC function.
17-16	2	ISQA	Ingress Sequence Action This field specifies whether the Frame Replication and Elimination for Reliability (FRER) sequence generation function (or action) is to be performed. Sequence generation consists of tagging a frame with a sequence number, following which the frame can be replicated and transmitted along multiple paths for redundancy. Implemented as per the IEEE 802.1CB. 00b = FRER sequence generation function is not performed. 01b = FRER sequence generation function is performed. 10b, 11b = Reserved. Note: Sequence Generation action is not applicable for ENETC function.
18	1	ORP	Override Rate Policer (instance) ID. 0b = Rate policer instance ID not overridden. 1b = Rate policer instance ID overridden.

Table continues on the next page...

Table 633. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			If ORP = 1, RP_EID specified in this entry overrides the default RP_EID (which is NULL) or PR_EID specified in Port filter.
19	1	OSGI	Override Stream Gate Instance Entry ID 0b = Stream Gate Instance Entry ID not overridden. 1b = Stream Gate Instance Entry ID overridden if OSGI = 1, specified SGI_EID in this entry overrides the default NULL SGI_EID.
23-20	4	HR	Host Reason This field specifies the Host Reason metadata when frame is redirected to the switch management port or copied to the switch management port (i.e. FA = 01b, 100b or 101b). Value specified has to be a software defined Host Reason (8-15). See Switch Management Port for details. Note: Not applicable to ENETC function.
26-24	3	FA	Forwarding Actions Switch function 000b = Discard. 001b = Re-direct frame to switch management port without any frame modification, along with Host Reason metadata set to the value configured in the HR field of this entry. Policing function will be applied to the frame if enabled. Stream and 802.1Q bridge forwarding functions are by-passed. 010b = Stream forwarding. It by-passes the 802.1Q bridge forwarding function. Frame is forwarded to the port(s) specified in the EGRESS_PORT_BITMAP field of this entry. 011b = 802.1Q bridge forwarding (VLAN processing and L2 forwarding). 100b = Copy to switch management port with specified HR and stream forwarding (stream forwarding action is as specified in FA = 010b). 101b = Copy to switch management port with specified HR and Bridge forwarding (Bridge forwarding action is as specified FA = 011b). All other values reserved. ENETC function 00b = Discard; 01b = Allow without setting the pre L2 filtering SI bitmap. 10b = Allow with setting the pre L2 filtering SI bitmap to the value configured in the SI_MAP field of this entry. If the pre L2 filtering SI bitmap was already set up by the ingress port filtering function, the pre L2 filtering SI bitmap is overwritten with the value configured in the SI_MAP field of this entry, The pre L2 filtering SI bitmap is used by the L2 filtering function to determine the final SI bitmap. All other values reserved.
28-27	2	SDU_TYPE	Service Data Unit type to use for MSDU (Maximum Service Data Unit) field. Overhead values are specified by register PRXSDUOR. 00b = PPDU (Physical Layer PDU). 01b = MPDU (MAC PDU).

Table continues on the next page...

Table 633. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			10b = MSDU (MAC SDU). All other values reserved. Note: This field has to be valid if MSDU field is not 0.
31-29	3	--	Reserved.
47-32	16	MSDU	Maximum Service Data Unit This field specifies the maximum service data unit size allowed for packets received on this stream. This is part of IEEE 802.1Qci (Per-Stream Filtering and Policing) specification. If the received frame length exceeds this value, the frame is discarded. A value of zero in this field, disables the maximum service data unit check. Note: This check is not performed for cut-through frames.
54-48	7	IFME_LEN_CHANGE	Ingress Frame Modification Entry Frame Length Change This field represents the effective frame length change of the ingress frame modification actions which are configured in the ingress frame modification entry (IFM_EID) referred by this entry. This frame length change value is used to adjust the frame length metadata, which in turn is passed along the frame processing pipeline, and used by the egress scheduling and management function to determine the actual length of the frame on the transmission link. Note that the egress queue scheduling and management function is executed prior to applying the frame modification actions to the frame, and thus the need to know of the frame length change resulting from the frame modification actions since it requires knowledge of the exact length of a frame transmitted on a link. Specifying explicitly the frame length change avoids having the hardware to read the ingress frame modifications entry and compute the frame length change itself. Fame length change is specified in unit of bytes using a 2's complement notation. Supported range is -4 bytes (removing one VLAN tag) to +4 bytes (adding one VLAN tag). This field is ignored by hardware if the IFM_EID field is set to NULL. Note: Not applicable to ENETC function.
59-55	5	EPORT	Egress Port This field specifies the identifier of the port requiring egress packet processing when the OETEID field of this entry is set to 01b (single-port Egress Treatment table access option). The port is expressed as an integer value. Switch Range: {0..SCAPR0[NUM_PORT]-1} Note: Not applicable to ENETC function.
61-60	2	OETEID	Override ET_EID This field is used to convey the egress packet processing actions to be applied to packets belonging to the stream associated with this entry. The egress packet

Table continues on the next page...

Table 633. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			processing actions are specified in the Egress Treatment table, where each table entry contains the egress packet processing actions to be applied to a scope of packets (e.g. stream) exiting on a particular egress port of the switch. The means by which one specifies the Egress Treatment table entries to be accessed, is through the Egress Treatment group. Within the hardware, an Egress Treatment group is determined by a base index (first Egress Treatment table entry of the group) and an applicability port bitmap (a bitmap corresponding to all ports of the switch) which indicates (with a 1) which ports have Egress Treatment table entries present. For more details, see Egress Treatment table access. Specifically, this field specifies how to derive the applicability port bitmap of the primary Egress Treatment group being assigned to packets belonging to the stream associated with this entry. The base index is specified in the ET_EID field of this entry. 00b = No egress packet processing actions specified. 01b = Single-port Egress Treatment table access. Only one port requires egress packet processing. That port identifier is specified in the EPORT field of this entry. The applicability port bitmap is set with a 1 for the port that is been identified in the EPORT field of this entry. For other ports, the applicability port bitmap is set to 0. The group assigned to packets belonging to the stream associated with this entry will have a single Egress Treatment table entry, for the port identified in the EPORT field of this entry. 10b = Multi-port packed Egress Treatment table access. Applicability port bitmap is set to the port bitmap specified in EGRESS_PORT_BITMAP field of this entry. The group assigned to packets belonging to the stream associated with this entry will have an Egress Treatment table entry for each port set to 1 in the applicability port bitmap. 11b = Multi-port absolute Egress Treatment table access. Applicability port bitmap is set with 1 for all ports. The group assigned to packets belonging to the stream associated with this entry will have Egress Treatment table entries for all ports on the switch. Valid if FA = 010b..101b and ET_EID is not NULL. Note: Not applicable to ENETC function.
63-62	2	CTD	Cut-Through Disable 00b = Do not override cut-through state. 01b = Disable cut-through for the outgoing port specified in the EPORT field. Cut-through is not disabled for the other ports specified in destination port bitmap (i.e. EGRESS_PORT_BITMAP field of this entry). 10b = Disable cut-through for all ports specified in the destination port bitmap (i.e. EGRESS_PORT_BITMAP of this entry). 11b = Reserved. Valid if FA = 010b .. 101b. All other values reserved.

Table continues on the next page...

Table 633. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Note: Not applicable to ENETC function.
95-64	32	ISQ_EID	Ingress Sequence Generation Entry ID This field specifies the Ingress Sequence Generation Entry ID to be used as an index into the Ingress Sequence Generation table. Valid only when ISQA is set to 1. 0x0..ISQGITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); ingress sequence generation function is by-passed Other values are reserved Note: Not applicable to ENETC functions.
127-96	32	RP_EID	Rate Policer Entry ID This field specifies the Rate Policer Entry ID to be used as an index into the Rate Policer table. Valid if ORP is set to 1. Override Rate Policer Entry ID with this value when ORP =1. 0x0..RPITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); rate policing function is by-passed Other values are reserved
159-128	32	SGI_EID	Stream Gate Instance Entry ID This field specifies the Stream Gate Instance Entry ID to be used as an index into the Stream Gate Instance table. 0x0..SGIITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); stream gating function is by-passed Other values are reserved
191-160	32	IFM_EID	Ingress Frame Modification Entry ID This field specifies the Ingress Frame Modification Entry ID which is encoded either as an index into the Frame Modification table or encoded as a set of frame modification actions. For the latter, there is no need to access the Frame Modification table as the actions are encoded in the Ingress Frame Modification Entry ID itself. If IFM_EID is set to 0xFFFF_FFFF (NULL), no ingress frame modification is performed. If IFM_EID is not set to NULL, bits 15-0 are used for IFM_EID, upper bits are ignored. Refer to Table Frame Modification Entry Definition for IFM_EID format. Note: Not applicable to ENETC function
223-192	32	ET_EID	Egress Treatment Entry ID This field specifies the index (or base index) to be used when accessing the Egress Treatment table. This field is valid if FA field is set to 010b .. 101b, and the OETEID field is set to value other than 00b. 0x0..ETTCAPR[NUM_ENTRIES]-1

Table continues on the next page...

Table 633. Table ID 31 - Ingress Stream Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			0xFFFF_FFFF (NULL); egress treatment processing is by-passed Other values are reserved Note: Not applicable to ENETC function.
255-224	32	ISC_EID	Ingress Stream counter Index Index to a set of counters in common memory that will be updated for this Stream. 0x0..ICITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); no Ingress Stream Count table counters are updated Other values are reserved
279-256	24	EGRESS_PORT_BITMAP	Egress Port bitmap For FA = 010b, 100b (stream forwarding), this field identifies the destination port(s) to which the frame is to be forwarded for this stream. For FA = 011b, 101 (802.1Q bridge forwarding), this field identifies destination ports to which egress packet processing is required. The destination port(s) are represented as a bitmap, where each bit of the bitmap corresponds to a port on the switch. Least significant bit of the bitmap corresponds to the smallest port number; i.e. bit offset 0 of the bitmap corresponds to port numbered 0, bit offset 1 to port numbered 1, and so on. The port(s) to which the frame are to be forwarded (stream forwarding) and/or egress packet processing is required (VLAN/FDB), are identified by having their corresponding bit set to 1 in the bitmap. Valid bits in the bitmap is from 0..CAPR0[NUM_LINKS]-1 Bits beyond the valid range will be ignored. Note: Not applicable to ENETC function.
287-280	8	--	Reserved.
303-288	16	SI_MAP	Station Interface Map This field specifies a pre L2 filtering SI bitmap. Each bit of the bitmap corresponds to an SI on a given port. Least significant bit of the bitmap corresponds to the smallest SI number. The SIs to which the frame is to be delivered, are identified by having their corresponding bit set to 1 in the bitmap. Bit wise setting of Station Interface map. LSB is the lowest order of SI which is the primary SI, rest are VSIs. Number of VSIs are specified in CAPR0[NUM_VSI]. Bits beyond the valid range will be ignored. Note: This field is only valid for ENETC function when FA field is set to 10b.

53.4.2.4.7.11 Ingress Stream Filter Table

Table 634. Table ID 32 - Ingress Stream Filter Table Common Attributes

TABLE_ID	32
TABLE_VERSION	0

Table continues on the next page...

Table 634. Table ID 32 - Ingress Stream Filter Table Common Attributes (continued)

Table Type	Hash table			
Table Description	The Stream Filter table contains PSFP (IEEE 802.1Qci) stream filter information for stream filters selected based on the 3-bit PCP value (priority) of the received frame and the stream handle (i.e. Ingress Stream Entry ID). Table management command operations Add, Delete, Update and Query are supported.			
Number of Entries	The memory allotment to store the hash entries is shared across all of the (ENETC instance/ switch) function hash tables. The maximum amount of memory words available to store all of the function hash tables entries, is indicated in the register HTMCAPR[NUM_WORDS]). If the number of memory words in-use to store hash table entries for the function has reached this limit, the Add operation will fail. The number of memory words in-use by the function for hash table entries is indicated in the register HTMOR[AMOUNT].			
Default Reset Behavior	No entries in the table by default.			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query.	
	Exact Match Key Element Match (KEY_ELEMENT)	Yes	Following commands are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x8 = Add. 0xC = Add, followed by a Query.	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	Yes	Only supported when command operation = 0x4 (Query). In the event of a search command requiring multiple iterations (see RESUME_ENTRY_ID), the user must not delete any entries in the table between iterations of that search.	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (ISF_EID)	4	4
	ACCESS_KEY	Access Key	8	4
	KEYE_DATA	Key Element	8	4

Table continues on the next page...

Table 634. Table ID 32 - Ingress Stream Filter Table Common Attributes (continued)

	CFGE_DATA	Configuration Element	16	4
Entry Alignment [bytes]	4			
Entry ID Management	Hardware assigns entry IDs.			
Response STATUS Applicable	Yes			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x280 = 0x280 = IS_EID in KEYE_DATA is invalid. 0x281 = Any of the following in CFGE_DATA is invalid - One or more of following Entry IDs are not in valid range or Entry ID specified is not Null. Checks are performed for following Entry IDs CFGE DATA – RP_EID, SGI_EID, ISC_EID. - SDU_TYPE is invalid			

Table 635. Table ID 32 - Ingress Stream Filter Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGEU	
ACCESS_KEY																																0x4
ACCESS_KEY																																0x8
CFGE_DATA																																0xC
...																																...
CFGE_DATA																																0x18

Table 636. Table ID 32 - Ingress Stream Filter Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved

Table continues on the next page...

Table 636. Table ID 32 - Ingress Stream Filter Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. 0x1 = Query ENTRY_ID only. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 32 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
95-32	64	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. Bits 63-32: Reserved. If ACCESS_METHOD = 0x1 (Key Element Match): Bits 63-0: KEYE_DATA. KEYE_DATA Format. If ACCESS_METHOD = 0x2 (Search): Bits 31-0: SEARCH_CRITERIA. SEARCH_CRITERIA Format. Bits 63-32: Reserved. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
223-96	128	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 637. Table ID 32 - Ingress Stream Filter Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
STATUS																																0x0
ENTRY_ID																																0x4
KEYE_DATA																																0x8
KEYE_DATA																																0xC
CFGE_DATA																																0x10
...																																...
CFGE_DATA																																0x1C

Table 638. Table ID 32 - Ingress Stream Filter Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	STATUS	STATUS Bits 31-0: RESUME_ENTRY_ID - An Entry ID which can be used as the RESUME_ENTRY_ID value within the ACCESS_KEY of a subsequent search to continue the search. When RESUME_ENTRY_ID is returned as the NULL Entry ID, this indicates that the search was completed successfully and no further iterations are required. This field is valid only in responses for operations which use the Search as the Access Method. For all other Access Methods, this field is reserved. Note: A NULL Entry ID is 0xFFFF_FFFF For generic details on the STATUS field, see Status. For generic details on the Search Access Method, see Search Access Method.
63-32	32	ENTRY_ID [1]	Entry ID When an unassigned entry ID is used to access an entry, hardware needs to process the command as if the entry were not found. For generic details on the Entry ID field, see Entry ID Management.
127-64	64	KEYE_DATA[1]	Key Element Data See KEYE_DATA_Format.
255-128	128	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 639. Table ID 32 - Ingress Stream Filter Table KEYE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
IS_EID																																0x0
--																													PCP			0x4

Table 640. Table ID 32 - Ingress Stream Filter Table KEYE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	IS_EID	Ingress Stream Entry ID Range: 0 to ISITCAPR[NUM_ENTRIES]-1 0x0..ISITCAPR[NUM_ENTRIES]-1 Other values are reserved
34-32	3	PCP	Priority Code Point Outer VLAN TAG PCP of the received frame
63-35	29	--	Reserved.

Table 641. Table ID 32 - Ingress Stream Filter Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
MSDU																--	SDU_TYPE		ORP	CTD	OSGI	TIMECAPE	IMIRE	ODR	DR		OIPV	IPV					0x0
RP_EID																																0x4
SGI_EID																																0x8
ISC_EID																																0xC

Table 642. Table ID 32 - Ingress Stream Filter Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
3-0	4	IPV	Internal Priority Value New Internal Priority Value (IPV) to be assigned to the frame, if OIPV is set to 1. The least significant 3 bits are used, most significant bit is reserved.
4	1	OIPV	Override Internal Priority Value (IPV) 0b = Don't override frame IPV. 1b = Override frame IPV with value configured in IPV field of this entry.
6-5	2	DR	Drop Resilience New Drop Resilience (DR) to be assigned to the frame, if ODR is set 1.
7	1	ODR	Override Drop Resilience (DR) 0b = Don't override frame DR. 1b = Override frame DR with value configured in DR field of this entry.
8	1	IMIRE	Ingress Mirroring Enable 0b = No ingress mirroring action specified in this entry. 1b = The frame is mirrored to the mirror destination specified in the IMDCR0 register. Note: Not applicable to ENETC function.
9	1	TIMECAPE	Timestamp Capture Enable 0b = No change to the timestamp. 1b = Timestamp is to be captured. Note: Not applicable to ENETC function.
10	1	OSGI	Override Stream Gate Instance Entry ID 0b = Stream Gate Instance Entry ID not overridden 1b = Stream Gate Instance Entry ID overridden; specified SGI_EID in this entry overrides the default null SGI_EID or SGI_EID specified in Ingress Stream table.
11	1	CTD	Cut-Through Disable 00b = Do not override cut-through state.

Table continues on the next page...

Table 642. Table ID 32 - Ingress Stream Filter Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			01b = Disable cut-through for all destined ports. Note: Not applicable to ENETC function.
12	1	ORP	Override Rate Policer (instance) ID 0b = Rate policer instance ID not overridden 1b = Rate policer instance ID overridden If the rate policer instance was already set by a match in the Ingress Port Filter table or Ingress Stream table and if ORP is set to 1, the rate policer instance value is overwritten with the value configured in the RP_EID field of this entry.
14:13	2	SDU_TYPE	Service Data Unit type to use for MSDU (Maximum Service Data Unit) field. Overhead values are specified by register PRXSDUOR. 00b = PPDU (Physical Layer PDU). 01b = MPDU (MAC PDU). 10b = MSDU (MAC SDU). All other values reserved. Note: This field has to be valid if MSDU field is not 0.
15	1	--	Reserved.
31-16	16	MSDU	Maximum Service Data Unit This field specifies the maximum service data unit size defined for the stream. This is part of IEEE 802.1Qci (Per-Stream Filtering and Policing) specification. If the received frame length exceeds this value, the frame is discarded. A value of zero in this field, disables the maximum SDU check. This field overrides the maximum SDU check field specified in the previous matched Ingress Stream table entry. Note: This check is not performed for cut-through frames.
63-32	32	RP_EID	Rate Policer Entry ID This field specifies the Rate Policer Entry ID to be used as an index into the Rate Policer table. Valid if ORP is set to 1. Override Rate Policer Entry ID with this value when ORP = 1. 0x0..RPITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); rate policing function is by-passed Other values are reserved
95-64	32	SGI_EID	Stream Gate Instance Entry ID This field specifies the Stream Gate Instance Entry ID to be used as an index into the Stream Gate Instance table. Stream Gating action will be performed if SGI_EID is not null. Valid if OSGI is set to 1. 0x0..SGIITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); stream gating function is by-passed Other values are reserved

Table continues on the next page...

Table 642. Table ID 32 - Ingress Stream Filter Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
127-96	32	ISC_EID	Ingress Stream Count Entry ID Index to a set of counters in common memory that will be updated for this stream. This field overrides the Ingress Stream Counter ID field specified in the previous matched Ingress Stream table entry. 0x0..ICITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); no Ingress Stream Count table counters are updated Other values are reserved

Table 643. Table ID 32 - Ingress Stream Filter Table SEARCH_CRITERIA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RESUME_ENTRY_ID																																0x0

Table 644. Table ID 32 - Ingress Stream Filter Table SEARCH_CRITERIA Description

Bits	Width [bits]	Name	Description
31-0	32	RESUME_ENTRY_ID	Resume Entry ID When starting a search, pass the NULL Entry ID. To continue a search, pass the RESUME_ENTRY_ID value returned in the CBD Response of the previous search command. Note: A NULL Entry ID is 0xFFFF_FFFF

53.4.2.4.7.12 Egress Treatment Table

Table 645. Table ID 33 - Egress Treatment Table Common Attributes

TABLE_ID	33
TABLE_VERSION	0
Table Type	Dynamic bounded index table
Table Description	Each entry in the Egress Treatment table contains the egress packet processing actions to be applied to a grouping or scope of packets exiting on a particular egress port of the switch. A scope of packets, for example could be the packets exiting a particular VLAN, matching a particular 802.1Q bridge forwarding entry or belonging to a stream identified at ingress. The Egress Treatment table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. Egress Treatment Entry ID (ET_EID) is used as an index to an entry in this table. For this table, each table entry occupies 8 bytes. Note that when software performs a Query operation, and requests the Configuration Element data to be returned, the Configuration Element data in the Response Data Buffer may not precisely match the data which was used when an Add operation was issued. If EFM_MODE=00b

Table continues on the next page...

Table 645. Table ID 33 - Egress Treatment Table Common Attributes (continued)

	then the value of EFM_DATA_LEN is always returned as 0x0. If EFM_MODE=01b or EFM_MODE=10b then the value of EFM_LEN_CHANGE is always returned as 0x0.			
Number of Entries	Maximum number of entries is indicated in ETTCAPR. Number of entries in-use is indicated in ETTOR[NUM_ENTRIES]. Entry size is 8 bytes.			
Default Reset Behavior	No entries in the table by default.			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x8 = Add.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (ET_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	16	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x290 = Command option specified is invalid or not supported. ESQA is not 00 or 10 (others are reserved), or ECA > 1 (reserved) 0x291 = FM_EID format or index is out of range. Check performed is as follows			

Table continues on the next page...

Table 645. Table ID 33 - Egress Treatment Table Common Attributes (continued)

	-EFM_EID format option type is invalid, or EFM_EID format is option 1 and the Index is out of range and not Null, or EFM_EID format is option 2 and VUDA or SQTA is out of range. -the Egress Counter Table index EC_EID is out of range -The Egress Sequence Actions Target Entry ID ESQA_TGT_EID is out of range

Table 646. Table ID 33 - Egress Treatment Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																...
CFGE_DATA																																0x14

Table 647. Table ID 33 - Egress Treatment Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 33 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match):

Table continues on the next page...

Table 647. Table ID 33 - Egress Treatment Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			Bits 31-0: ENTRY_ID. See Response Buffer Description All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
191-64	128	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 648. Table ID 33 - Egress Treatment Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
CFGE_DATA																																0x4
...																																...
CFGE_DATA																																0x10

Table 649. Table ID 33 - Egress Treatment Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Egress Treatment Entry ID Valid Range: {0..ETTCAPR[NUM_WORDS]-1} ET_EID value outside table range results in an error. For generic details on the Entry ID field, see Entry ID Management.
159-32	128	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 650. Table ID 33 - Egress Treatment Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--					EFM_DATA_LEN											EFM_LEN_CHANGE							ECA			ESQA		--		EFM_MODE		0x0
EFM_EID																																0x4
EC_EID																																0x8
ESQA_TGT_EID																																0xC

Table 651. Table ID 33 - Egress Treatment Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
1-0	2	EFM_MODE	Egress Frame Modification Mode This field is used to determine how the EFM_LEN_CHANGE or EFM_DATA_LEN field is used to get the effective length of the frame, after modifications. See the description of these fields for more details. 00b = Normal/Default mode. The EFM_LEN_CHANGE field is used to make adjustments to the frame size. 01b = When EFM_EID[L2_ACT]=01b. The EFM_DATA_LEN field represents the new size of the frame. 10b = When EFM_EID[PLD_ACT]=001b. The EFM_DATA_LEN field represents the size of the Ethernet payload plus any changes to the L2 Header. 11b = Reserved.
3-2	2	--	Reserved
5-4	2	ESQA	Egress Sequence Actions 00b = No Sequence Action required. 01b = Reserved. 10b = Sequence Recovery action. See ESQA_TGT_EID. 11b = Reserved.
8-6	3	ECA	Egress Counter Action 000b = Do not increment egress frame counter. 001b = Increment egress frame counter. See EC_EID. 010b ..111b = Reserved.
15-9	7	EFM_LEN_CHANGE	Egress Frame Modification Length Change This field specifies the frame length change that represents the effective frame length change of the egress frame modification actions configured in the EFM_EID field of this entry. This frame length change value is used to adjust the frame length meta data, which in turn is passed to the egress scheduling and management functions to represent the actual length of the frame on the transmission link. Specifying explicitly the frame length change avoids having the hardware to read the egress frame modifications entry and compute the frame length change itself.

Table continues on the next page...

Table 651. Table ID 33 - Egress Treatment Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Frame Length change is specified in units of bytes using a 2's complement notation. Supported range is -10 bytes (removing one VLAN tag and one R-TAG/HSR) to +6 bytes (adding one VLAN tag and replacing a 4-byte 802.1CB draft 2.0 R-TAG with 6-byte 802.1CB R-TAG or HSR tag). This field is ignored by hardware if the EFM_EID field is set to null, or if EFM_MODE != 00b. As a result, when performing a Query operation the value returned may not precisely match the value used when the Add operation was issued.
26-16	11	EFM_DATA_LEN	Egress Frame Modification Data Length The value specified in this field is used to adjust the frame length meta data, which in turn is passed to the egress scheduling and management functions to represent the actual length of the frame on the transmission link. When EFM_MODE=01b, this field must be set to mirror the value held in EFM_EID[FMD_BYTES]. When EFM_MODE=10b, this field must be set to the value held in EFM_EID[FMD_BYTES] plus or minus any change to the L2 header due to egress L2 header frame modifications. For example, if a VLAN is being added, this field would be set to EFM_EID[FMD_BYTES]+4. EFM_EID[FMD_BYTES] minus the L2 header size decrease due to the egress L2 header modification actions specified, must be higher than 0. For example, if a VLAN tag is being removed, the minimum number of bytes that can be specified in EFM_EID[FMD_BYTES] is 5 bytes. Hardware uses the value of this field, to compute what would be the length of the frame on the transmission link, by first adding the value specified in this field to the size of the L2 header in the received frame. The resulted frame length is then adjusted for any ingress L2 header modifications. The Data Length field is specified in units of bytes. Supported range is 1 to 2047 bytes. Support for payload/frame creation when EFM_MODE=01b or 10b, not payload replacement. This field is ignored by hardware if the EFM_EID field is set to null, or if EFM_MODE != 01b or 10b. As a result, when performing a Query operation the value returned may not precisely match the value used when the Add operation was issued.
31-27	5	--	Reserved.
63-32	32	EFM_EID	Egress Frame Modification Entry Id. This field specifies the Egress Frame Modification Entry ID which is encoded either as

Table continues on the next page...

Table 651. Table ID 33 - Egress Treatment Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			an index into the Frame Modification table or encoded as a set of frame modification actions. For the latter, there is no need to access the frame modification table as the actions are encoded in the Egress Frame Modification Entry ID itself. If EFM_EID is set to 0xFFFF_FFFF (NULL), no egress frame modification is performed. If EFM_EID is not set to NULL, bits 15-0 are used for EFM_EID, upper bits are ignored. Refer to Frame Modification Entry Definition for EFM_EID format.
95-64	32	EC_EID	Egress Count Table Entry ID Index to a set of egress counters in common memory that will be updated. Valid if ECA not set 0 Valid Range: {0..ECTCAPR[NUM_WORDS]-1} Note: If EC_EID is not Null, bits 15-0 are used for EC_EID, upper bits are ignored, and as a result only 2 bytes of memory are used to store the EC_EID.
127-96	32	ESQA_TGT_EID	Egress Sequence Actions Target Entry ID Valid if ESQA=2, it then represent the ESQR_EID Valid Range: {0..ESQACAPR[NUM_WORDS]-1} Note: If ESQA_TGT_EID is not Null, bits 15-0 are used for ESQA_TGT_EID, upper bits are ignored, and as a result only 2 bytes of memory are used to store the ESQA_TGT_EID.

53.4.2.4.7.13 Ingress Sequence Generation Table

Table 652. Table ID 34 - Sequence Generation Table Common Attributes

TABLE_ID	34
TABLE_VERSION	0
Table Type	Dynamic bounded index table
Table Description	The Ingress Sequence Generation table provides the tag type to be inserted and the next sequence number value to be used in that tag. This table is used to implement the IEEE 802.1CB sequence generation function which consists of tagging a frame with a sequence number, following which the frame can be replicated and transmitted along multiple paths for redundancy. The Ingres Sequence Generation table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. The entries are stored in contiguous manner in the common memory. Ingress Sequence Generation Entry ID (ISQ_EID) is used as an index to an entry in this table. The Ingress Sequence Generation Entry ID is specified in the Ingress Stream table entry. For this table, there are 8 entries per word. Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) register S0ISQGITMAR[NUM_WORDS]. This allocation can also be changed when the function is initialized through its respective base function register ISQGITMAR[NUM_WORDS]. Table management command operations Add, Delete, Update and Query are supported.

Table continues on the next page...

Table 652. Table ID 34 - Sequence Generation Table Common Attributes (continued)

Number of Entries	Maximum number of entries is indicated in ISQGITCAPR[NUM_ENTRIES]. Number of entries in-use is indicated in ISQGITOR[NUM_ENTRIES].			
Default Reset Behavior	No entries in the table by default.			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update. 0x8 = Add.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (SQG_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	1	1
	SGSE_DATA	Sequence Generation State Element	2	2
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x2A0 = SQ_TAG specified is not valid.			

Table 653. Table ID 34 - Sequence Generation Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS														SGSEU	CFGEU	0x0
																--																
ACCESS_KEY																																0x4
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						CFGE_DATA										0x8

Table 654. Table ID 34 - Sequence Generation Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bit 1: SGSEU - Sequence Generation Element Update. 0b = No update performed to the Sequence Generation Element. 1b = Sequence Generation reset (SQG_NUM is reset to zero) Bits 15-2: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 34 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
71-64	8	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 655. Table ID 34 - Sequence Generation Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
SGSE_DATA																--						CFGE_DATA										0x4

Table 656. Table ID 34 - Sequence Generation Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Ingress Sequence Generation Entry ID Range: 0..ISQGITCAPR[NUM_ENTRIES]-1} For generic details on the Entry ID field, see Entry ID Management.
39-32	8	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.
47-40	8	--	Reserved.
63-48	16	SGSE_DATA[1]	Sequence Generation State Element Data See SGSE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 657. Table ID 34 - Sequence Generation Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																								--					SQ_TAG			0x0

Table 658. Table ID 34 - Sequence Generation Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
2-0	3	SQ_TAG	Sequence Tag Type This field specifies the sequence tag type to insert for sequence generation. 000b = Reserved. 001b = 802.1CB draft 2.0 R-TAG. 010b = 802.1CB R-TAG. 011b = HSR Tag. The HSR PathId field is set to the value configured in PLANIDCR[LANID]. The HSR LSDU size field is set to the frame length minus the MAC header, VLAN tag(s), HSR Ethertype and FCS. No padding is added, it is assumed that received short frames have been padded. 100b ... 111b = Reserved.

Table continues on the next page...

Table 658. Table ID 34 - Sequence Generation Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
7-3	5	--	Reserved.

Table 659. Table ID 34 - Sequence Generation Table SGSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																SQG_NUM																0x0

Table 660. Table ID 34 - Sequence Generation Table SGSE_DATA Description

Bits	Width [bits]	Name	Description
15-0	16	SQG_NUM	Sequence Generation Number. This field specifies the next sequence number value to be used in the tag to be added to the frame. Initialized to 0 when entry is added or reset is performed. SQG_NUM is incremented by one after each packet that passes through the sequence generation function. This field rolls over to 0 when maximum value of 0xFFFF is reached.

53.4.2.4.7.14 Egress Sequence Recovery Table

Table 661. Table ID 35 - Egress Sequence Recovery Table Common Attributes

TABLE_ID	35		
TABLE_VERSION	0		
Table Type	Static bounded index table		
Table Description	The Egress Sequence Recovery table contains one entry per FRER recovery function (implemented as per 802.1CB), and each entry contains a number of configuration elements, whose value is set when the table entries are configured, and operational elements, which are managed by the hardware and can be queried by software. Whether or not a FRER recovery function is to be performed is specified in the Egress Treatment table, and if so then the Egress Treatment table also provides an index into a Egress Sequence Recovery table. The Egress Sequence Recovery table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. The entries are stored in contiguous manner in a dedicated memory. Egress Sequence Actions Target Entry ID (ESQA_TGT_EID) is used as an index to an entry in this table. The ESQA_TGT_EID is specified in the Egress Treatment table.		
Number of Entries	Number of entries is indicated in ESQRTCAPR[NUM_ENTRIES].		
Default Reset Behavior	After reset all table fields are cleared to 0.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported:

Table continues on the next page...

Table 661. Table ID 35 - Egress Sequence Recovery Table Common Attributes (continued)

			0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (SQR_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	STSE_DATA	Statistics Element	52	4
	SRSE_DATA	Sequence Recovery State Element	20	4
	CFGE_DATA	Configuration Element	8	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description under section NTMP Response Message Header Format Note for this table the Access Method field is ignored and always treated as Entry ID Match. No error is generated if an unsupported Access Method is requested. Note for this table the Request Data Buffer must always be 16 bytes, as specified below. An invalid request data buffer length will not return an error, but may cause a corrupted table entry.			

Table 662. Table ID 35 - Egress Sequence Recovery Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--													SRSEU	STSEU	CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
CFGE_DATA																																0xC

Table 663. Table ID 35 - Egress Sequence Recovery Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bit 1: STSEU - Statistics Element Update. 0b = No update performed to the Statistics Element. 1b = Reset statistic counters. Note: Required when Driver periodically perform statistics read and reset operation to prevent HW counter rollover. Driver can support 64bit counters to applications even though HW supports much smaller counters. Bit 2: SRSEU - Sequence Recovery State Element Update. 0b = No update performed to the Sequence Recovery State Element. 1b = Sequence Recovery State reset which initializes the following element fields: SQR_NUM = 2**16-1, TAKE_ANY = 1, LCE = 0, SQR_HISTORY[]=0 Bits 15-3: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved.
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 35 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match):

Table continues on the next page...

Table 663. Table ID 35 - Egress Sequence Recovery Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
127-64	64	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 664. Table ID 35 - Egress Sequence Recovery Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID	0x0
STSE_DATA	0x4
...	...
STSE_DATA	0x34
CFGE_DATA	0x38
CFGE_DATA	0x3C
SRSE_DATA	0x40
...	...
SRSE_DATA	0x50

Table 665. Table ID 35 - Egress Sequence Recovery Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Egress Sequence Recovery Entry ID Range: {0..ESQRTCAPR[NUM_ENTRIES]-1} For generic details on the Entry ID field, see Entry ID Management.
447-32	416	STSE_DATA[1]	Statistics Element Data See STSE_DATA_Format.
511-448	64	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.
671-512	160	SRSE_DATA[1]	Sequence Recovery State Element Data See SRSE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 666. Table ID 35 - Egress Sequence Recovery Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--				SQR_FWL												--	SQR_HL							--		SQR_TYPE	SQR_ALG	SQR_TNSQ	SQ_TAG			0x0
--																						SQR_TP										0x4

Table 667. Table ID 35 - Egress Sequence Recovery Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
2-0	3	SQ_TAG	Sequence Tag This field is optional, and can be used to specify the expected sequence tag type in the frame. The default setting of 0 causes sequence recovery to be performed on any tag type, with frames treated as tagless only if no sequence tag is present. If SQ_TAG is non-zero, then this fields specifies the expected sequence tag type in the frame, sequence recovery is performed if the sequence tag type in the frame matches SQ_TAG, otherwise the frame is treated as tagless. 000b = Accept any incoming tag type. 001b = 802.1CB draft 2.0 R-TAG. 010b = 802.1CB R-TAG. 011b = HSR Tag 100b ... 111b = Reserved.
3	1	SQR_TNSQ	Sequence Recovery Take No Sequence This field specifies the action taken (accept or discard) on frames passing through a FRER recovery function that are tag-less, A frame is considered tagless if it has no sequence tag, or if SQ_TAG is not 0 and the tag type in the frame does not match the tag type specified in SQ_TAG. 0b = Discard frame and count in both the TAGLESS_PACKETS counter and in the port's PTXDCR register along with the setting of the SQRTNSQDR flag to 1 in the port's PTXDCRR0 register. 1b = Accept frame and do not perform recovery function. Note: Driver adds TaglessPackets to frerCpsSeqRcvyDiscardedPackets if SQR_TNSQ=1.
4	1	SQR_ALG	Sequence Recovery Algorithm This field specifies the recovery function algorithm. 0b = Vector algorithm. The vector recovery algorithm starts by always accepting the first received frame as valid. After the first frame has been accepted, all subsequent frames whose sequence number is in the window (last sequence number accepted +/- (recovery history length-1)) are checked against the recovery history to see if a previous frame with the same sequence number has already been accepted, and if

Table continues on the next page...

Table 667. Table ID 35 - Egress Sequence Recovery Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			so the current frame is rejected, otherwise it is accepted and the recovery history is updated to indicate the newly accepted sequence number. Frames whose sequence number falls outside of the recovery history window are rejected and counted as rogue frames. 1b = Match algorithm. The match recovery algorithm starts by always accepting the first received frame as valid, same as vector recovery. After the first frame has been accepted, all subsequent frames either match the last frame number accepted, and are therefore discarded, or do not, in which case they are accepted. The recovery history (SQR_HL) and the future window length (SQR_FWL) are not used by the match recovery algorithm.
5	1	SQR_TYPE	Sequence Recovery Function type This field specifies the type of recovery function supported: 0b = Sequence recovery function. For a sequence recovery function, the stored frame timestamp is updated only for accepted frames, it is not updated for rejected frames. This means that a sequence recovery function that is out of step and is continually rejecting frames will be reset after the timeout period is reached, allowing it to resume sending frames. 1b = Individual recovery function. For an individual recovery function, the stored frame timestamp is updated for all frames. This means that an individual recovery function will not timeout even if all received frames are rejected, and can only timeout if frames stop arriving for a long time (timeout period or greater), and then start arriving again.
7-6	2	--	Reserved.
14-8	7	SQR_HL	Sequence Recovery History Length This field specifies how many bits are used in the SQR History operational variable. The history length is equal to SQR_HL + 1. Valid if SQR_ALG = 0 Range: 1..ESQRCAPR[SQR_MAX_HL] Any attempt to set a history length larger than the maximum supported will result in the history length being set to its maximum. Any frame whose sequence number is less than or equal to SQR_NUM - history length (SQR_HL + 1) is rejected and counted as a rogue packet.
15	1	--	Reserved.
27-16	12	SQR_FWL	Sequence Recovery Future Window Length This field specifies the length of the recovery window for frames whose sequence number is greater than the last seen highest sequence number SQR_NUM. Valid if SQR_ALG = 0 The future window length is set to SQR_FWL + 1, so the default value is 1. Any frame whose sequence number is greater than or equal to SQR_NUM + SQR_FWL + 1 is rejected and counted as a rogue packet. Note this field is not part of the 802.1CB standard, to achieve behavior compliant to the standard the user should set SQR_FWL = SQR_HL.

Table continues on the next page...

Table 667. Table ID 35 - Egress Sequence Recovery Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
31-28	4	--	Reserved.
43-32	12	SQR_TP	Sequence Timeout Period This field specifies the recovery timeout period in increments of 1.048576 milliseconds. A value of 0 disables recovery timeout. Each recovery function stores a timestamp of the last frame accepted (for sequence recovery) or the last frame seen (for individual recovery). Each time a frame passes through the recovery function, the time that has expired since the last frame timestamp (current time minus last frame time) is calculated, and if the expired time exceeds the configured recovery timeout period, then the recovery function is reset.
63-44	20	--	Reserved.

Table 668. Table ID 35 - Egress Sequence Recovery Table STSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
IN_ORDER_PACKETS																																0x0
IN_ORDER_PACKETS																																0x4
OUT_OF_ORDER_PACKETS																																0x8
OUT_OF_ORDER_PACKETS																																0xC
ROGUE_PACKETS																																0x10
ROGUE_PACKETS																																0x14
DUPLICATE_PACKETS																																0x18
DUPLICATE_PACKETS																																0x1C
LOST_PACKETS																																0x20
LOST_PACKETS																																0x24
TAGLESS_PACKETS																																0x28
TAGLESS_PACKETS																																0x2C
SREC_RESETS																																0x30

Table 669. Table ID 35 - Egress Sequence Recovery Table STSE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	IN_ORDER_PACKETS	In Order Packets Count of packets that have passed the sequence recovery algorithm in a sequence order. Standard: frerCpsSeqRcvyPassedPackets = InOrderdPackets + OutOfOrderPackets + TaglessPackets (if SQR_TNSQ=1)

Table continues on the next page...

Table 669. Table ID 35 - Egress Sequence Recovery Table STSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Note: tsnCpsSidInputPackets = frerCpsSeqRcvyPassedPackets + frerCpsSeqRcvyDiscardedPackets
127-64	64	OUT_OF_ORDER_PACKETS	Out of Order Packets Incremented once for each packet accepted out-of-order by the Egress Sequence Recovery Algorithm. Count of packets that are out-of-order meaning that their sequence number is not one more than the one of the previous received packet. Standard: frerCpsSeqRcvyOutOfOrderPackets
191-128	64	ROGUE_PACKETS	Rogue Packets Count of packets discarded by the vector recovery algorithm whose sequence number have fallen outside of the recovery history window. The port's PTXDCR register is also incremented along with the setting of the SQRRDR flag to 1 in the port's PTXDCRR0 register. Standard: frerCpsSeqRcvyRoguePackets
255-192	64	DUPLICATE_PACKETS	Duplicate Packets Count of packets discarded due to a duplicate sequence number. The port's PTXDCR register is also incremented along with the setting of the SQRDDR flag to 1 in the port's PTXDCRR0 register. Standard: frerCpsSeqRcvyDiscardedPackets=RoguePackets + DuplicatePackets + TaglessPackets (if SQR_TNSQ=0) + EnqueueDiscardPackets
319-256	64	LOST_PACKETS	Lost Packets Count of packets lost by the vector recovery algorithm. A packet is counted as lost if its sequence number is not received on any ingress port. Standard: frerCpsSeqRcvyLostPackets
383-320	64	TAGLESS_PACKETS	Tag-Less Packets Count of packets received by the recovery function that have no sequence_number, or whose sequence tag type does not match SQ_TAG if SQ_TAG is non-zero. If the frame is discarded (SQR_TNSQ = 0), then the port's PTXDCR register is also incremented along with the setting of the SQRTNSQDR flag to 1 in the port's PTXDCRR0 register. Standard: frerCpsSeqRcvyTaglessPackets and frerCpsSeqEncErroredPackets
415-384	32	SREC_RESETS	Sequence Recovery Resets Count is incremented each time the sequence recovery function is reset due to recovery timeout. Standard: frerCpsSeqRcvyResets

Table 670. Table ID 35 - Egress Sequence Recovery Table SRSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--		SQR_TS												LCE	TAKE_ANY	SQR_NUM																0x0
SQR_HISTORY																																0x4
...																																...
SQR_HISTORY																																0x10

Table 671. Table ID 35 - Egress Sequence Recovery Table SRSE_DATA Description

Bits	Width [bits]	Name	Description
15-0	16	SQR_NUM	Sequence Recovery Number This field holds the highest sequence number value received so far by the recovery function, or 0xFFFF after reset of the recovery function. The sequence number is an unsigned integer that rolls over from maximum to 0, therefore at the rollover boundary sequence number 0 is treated as larger than sequence number 0xFFFF.
16	1	TAKE_ANY	Take Any This field is set to 1 when the recovery function is reset, and cleared after the first frame is received. The recovery function always accepts the first frame after reset, which updates SQR_NUM and SQR_HISTORY. All subsequent frames are then accepted or rejected based on their sequence number and the recovery algorithm used. Operational Element field which when set causes recovery algorithm to accept the next packet, no matter the value of its sequence number.
17	1	LCE	Lost Count Enable This field is reset to 0 and disables lost packet counting when the recovery function is reset. This bit is set to 1 the first time that the history window is moved and a 1 is detected in the history bits that are shifted out, i.e. there has been at least one sequence number received and shifted out of the history. Then lost packet counting will begin at the next history window move.
29-18	12	SQR_TS	Sequence Recovery Timestamp This field stored the timestamp of the last frame accepted (for sequence recovery) or the last frame seen (for individual recovery.
31-30	2	--	Reserved.
159-32	128	SQR_HISTORY	Recovery History This is a bit vector that maintains a history of the sequence numbers of recently received packets, with each bit corresponding to sequence numbers from SQR_NUM down to

Table continues on the next page...

Table 671. Table ID 35 - Egress Sequence Recovery Table SRSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			SQR_NUM - SQR_HL + 1. A 1 in a history bit indicates that a packet with that sequence number has been previously received, a 0 indicates that no packet with that sequence number has yet been received.

53.4.2.4.7.15 Stream Gate Instance Table

Table 672. Table ID 36 - Stream Gate Instance Table Common Attributes

TABLE_ID	36
TABLE_VERSION	0
Table Type	Dynamic bounded index table
Table Description	The Stream Gate Instance (SGI) table contains the stream gate configuration and operational information for a NETC function. There is one SGI table for each ENETC instance and one SGI table for the entire switch. Each entry contains a set of parameters that defines a single stream gate instance. The stream gate instance parameters includes an administrative stream gate control list and an operational stream gate control list, along with related configuration and status parameters. A stream gate control list (SGCL) is used to specify time gates or time slots during which incoming frames from one or more streams will be accepted. This table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. The entries are stored in contiguous manner in the common memory. For this table, each table entry occupies one word (one to one corresponding between words and entries). Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) registers; S0SGIITMAR[NUM_WORDS] for switch and EaSGIITMAR[NUM_WORDS] for ENETC. These allocations can also be changed when the functions are initialized through their respective base function registers SGIITMAR[NUM_WORDS]. The index to access this table, is obtained earlier on, via lookups in the Ingress Stream or in the Ingress Stream Filter table. An SGI can hold two SGCLs, one as operational and the other as administrative. An administrative SGCL becomes operational at Config Change Time as defined in the IEEE 802.1Qci standard and remains operational till another SGCL is installed. An administrative SGCL is added to a SGI using the Add or Update operations. The actual SGCLs are stored in a separate table called the Stream Gate Control List table, with references to these SGCLs (that is, entry IDs) specified in the SGI table. It is recommended that the entry ID for each of these two SGCLs be predetermined and known by software, and persist till the SGI is no longer in use. Take the following into consideration when accessing the SGI table[1]: • **Initialization** - The 1588 timer must be initialized and configured before adding any SGCL to an SGI. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer is in normal operation, any change to the 1588 timer configuration (for example TMROFF_H/L), except for TMR_ADD updates, requires that all SGIs with an operational SGCL be deleted, and for SGIs that have no operational SGCLs, that the administrative SGCL, if present, be removed from the SGI. • **Stream association -** An SGI can be associated with only one switch port. Streams from different switch ports cannot use the same SGI.

Table continues on the next page...

Table 672. Table ID 36 - Stream Gate Instance Table Common Attributes (continued)

	• **Config Change Time** - Note that when Config Change Time has been reached, the actual installation of a new SGCL, will occur at the next event that will cause the operational SGCL to be traversed; that is reception of a frame or expiry of the refresh timestamp timer. For any SGI command processed by hardware prior to this event, the administrative SGCL is considered to be still pending although the Config Change Time may have been reached. • **Add operation** - When an ADMIN_SGCL_EID is specified in the Add operation, the SGI will operate with the initial SGI configuration till the ADMIN_SGCL_EID is installed at Config Change Time. An SGI can operate without an SGCL and with just the initial SGI configuration. To operate in this mode, specify a NULL ADMIN_SGCL_EID in the Add operation. • **Update operation** - This operation is used to add a new administrative SGCL to an existing SGI or to remove an administrative SGCL from an SGI. For the details on how to add a new administrative SGCL, refer to the sequence described below in Adding an administrative SGCL to an SGI. For the details on how to remove an administrative SGCL, refer to the sequence described below in Removing an administrative SGCL from an SGI. Note that if the operational SGCL is present, it will continue to operate as normal. • **Delete operation** - Perform the following steps to delete an SGI table entry: 1. Remove, if present in the SGI, the administrative SGCL by means of an SGI command containing an Update operation to the SGI's Admin Configuration Element with ADMIN_SGCL_EID set to NULL. 2. Issue a Delete operation against the SGI table entry that is to be deleted. 3. If needed, delete the SGCLs from the Stream Gate Control List table. Since there are two possible SGI_EIDs, which the deleted SGI may have used, software can then safely issue SGCL commands containing a Delete operation to those two SGCLs. A Delete operation to an SGCL has not been added before or has been already deleted, will execute with no error; that is, the Delete operation is not performed, and the response message indicates that no entry has been found (NUM_MATCHED in Response Message Header is set to 0). ***Adding an administrative SGCL to a SGI*** Follow these steps to add an administrative SGCL to an SGI: 1. Since software has only two possible SGCLs (and therefore, SGCL_EIDs) to use, software must first verify that at least one of them is unused by the hardware, and therefore, "free" to be configured and associated to the SGI. This procedure is started by issuing a query on the SGI to record the presence of both the administrative and operational SGCLs (found in SGISE_DATA[STATE]), and the SGCL_EIDs (found in ACFGE_DATA[ADMIN_SGCL_EID] and SGISE_DATA[OPER_SGCL_EID]), respectively, if valid, based on the value of SGISE_DATA[STATE]. • **Case 0** - Neither SGCL is present. An administrative SGCL may be configured and associated to the SGI. Proceed to Step 3. • **Case 1** - An administrative SGCL only is present. Execute the sequence described below in Removing an administrative SGCL from an SGI, but starting at step 2 (as step 1 has already performed in this sequence), to remove the administrative SGCL and then retry this sequence.

Table continues on the next page...

Table 672. Table ID 36 - Stream Gate Instance Table Common Attributes (continued)

	• **Case 2** - An administrative SGCL and an operational SGCL is present. Both SGCLs are currently in use; a new one may not be configured. Execute the sequence described below in Removing an administrative SGCL from an SGI, but starting at step 2 (as step 1 has already performed in this sequence), to remove the administrative SGCL and then retry this sequence. • **Case 3** - An operational SGCL only is present. An administrative SGCL can be configured and associated to the SGI. Proceed to Step 2. 2. Ensure that any stale SGCL is deleted from the SGCL table; a stale SGCL is an operational SGCL that ceased to be operational as it was replaced by a new SGCL. a. Since there are two possible SGI_EIDs which this SGI may use, any of the two lists which are not present, may be deleted. The SGI_EID(s) present in the SGI is indicated based on the combination of SGISE_DATA[STATE] and ACFGE_DATA[ADMIN_SGCL_EID]/SGISE_DATA[OPER_SGCL_EID]. b. Software can then safely issue SGCL commands containing a Delete operation to those SGCLs. A Delete operation to an SGCL has not been added before or has been already deleted, will execute with no error; i.e. the Delete operation is not performed, and the response message indicates that no entry has been found (NUM_MATCHED in Response Message Header is set to 0). 3. Software may chose an SGCL from its pool of two SGCLs for this SGI which is currently unused by the hardware (or not present) and issue a command containing an Add operation to SGCL table, to add the new administrative SGCL. 4. The new added SGCL may now be associated with the SGI by means of an SGI command containing an Update operation to the SGI's Admin Configuration Element with the new SGCL (e.g. setting ADMIN_SGCL_EID to new added SGCL_EID). ***Removing an administrative SGCL from an SGI*** Follow these steps to remove an administrative SGCL from an SGI: 1. This procedure is started by issuing a query on the SGI. Software records the presence of both the administrative and operational SGCLs (found in SGISE_DATA[STATE]), and the SGCL_EIDs (found in ACFGE_DATA[ADMIN_SGCL_EID] and SGISE_DATA[OPER_SGCL_EID]), respectively, if valid, based on the value of SGISE_DATA[STATE]. 2. Remove the administrative SGCL by means of an SGI command containing an Update operation to the SGI's Admin Configuration Element with ADMIN_SGCL_EID set to NULL. 3. Query the SGI again. a. If the query response indicates that no administrative SGCL is present, and if the “operational SGCL” encoding from SGISE_DATA[STATE] and SGISE_DATA[OPER_SGCL_EID] (if operational SGCL is present) are matching the queried values from Step 1, this indicates that the administrative SGCL was successfully removed from the SGI. Proceed to Step 4. b. If software finds that either an operational SGCL is now present when it did not in the Step 1 query, or if the SGCL_EID of the operational SGCL is different from the one in the Step 1 query, this indicates that the update command in Step 2 was processed by the hardware only after the administrative SGCL (which software was attempting

Table continues on the next page...

Table 672. Table ID 36 - Stream Gate Instance Table Common Attributes (continued)

	to uninstall) had become operational. In this case, software must not proceed to delete the SGCL. Do not proceed to step 4. 4. The administrative SGCL was successfully removed from the SGI, it is now safe to issue a SGCL command containing a Delete operation to the administrative SGCL.			
Number of Entries	Maximum number of entries is indicated in SGIITCAPR[NUM_ENTRIES]. Number of entries in-use is indicated in SGIITOR[NUM_ENTRIES].			
Default Reset Behavior	No entries in the table by default.			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x2 = Update. 0x4 = Query. 0x8 = Add.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (SGI_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	ACFGE_DATA	Admin Configuration Element	16	4
	ICFGE_DATA	Initial Configuration Element	1	1
	CFGE_DATA	Configuration Element	1	1
	SGISE_DATA	Stream Gate Instance State Element	13	4
Entry Alignment [bytes]	4			

Table continues on the next page...

Table 672. Table ID 36 - Stream Gate Instance Table Common Attributes (continued)

Entry ID Management	Software assigns entry IDs
Response STATUS Applicable	No
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x2C0 = SGCL_EID specified in out of range for Add or Update operation. Valid SGCL_EID values are {0..SGCLITCAPR[NUM_WORDS]-1}, NULL (0xFFFF_FFFF). 0x2C1 = SDU_TYPE is specified is invalid for Add or Update operation. 0x2C2 = Either the SGCL_EID specified as admin gate control list in Add or Update operation has not been allocated or SGCL_EID is not the first entry in gate control list, or the reference count in SGCL entry is not 0. 0x2C3 = SGCL_EID specified for Update operation is in invalid. When update is performed for an existing Admin control list, SGCL_EID specified in Update operation has to match that in the Stream Gate Instance. An Admin list exist in the Stream gate Instance if STATE in SGISE_DATA element = 2 or 3. 0x2C4 = ADMIN_BASE_TIME specified for Add or Update operation is more than 2^30ns in the past. (that is, current time - ADMIN_BASE_TIME is <2 ^30ns) 0x2C5 = Cumulated time value of CYCLE_TIME in Stream gate Control list plus CYCLE_TIME_EXT specified in Add or Update operation is >=2^30ns or CYCLE_TIME specified is 0.

1. The command sequences described in these considerations, assume that the entry ID for each of the two SGCLs be predetermined and known by software

Table 673. Table ID 36 - Stream Gate Instance Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--													SGISEU	CFGEU	ACFGEU	
ACCESS_KEY																																0x4
ACFGE_DATA																																0x8
...																																...
ACFGE_DATA																																0x14
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																ICFGE_DATA																0x18
																CFGE_DATA																

Table 674. Table ID 36 - Stream Gate Instance Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: ACFGEU - Admin Configuration Element Update. 0b = No update performed to the Admin Configuration Element. 1b = The data specified in the ACFGE_DATA field is written to the Admin Configuration Element of the table entry. Bit 1: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bit 2: SGISEU - Stream Gate Instance State Element Update. 0b = No update performed to the Stream Gate State Element. 1b = Reset the IRX, OEX flags to 0 Bits 15-3: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 36 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
191-64	128	ACFGE_DATA [1]	Admin Configuration Element Data See ACFGE_DATA Format.
199-192	8	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA Format.
207-200	8	ICFGE_DATA[1]	Initial Configuration Element Data See ICFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 675. Table ID 36 - Stream Gate Instance Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
SGISE_DATA																																0x4
...																																...
--								ICFGE_DATA								CFGE_DATA						SGISE_DATA										0x1C
ACFGE_DATA																																0x20
...																																...
ACFGE_DATA																																0x2C

Table 676. Table ID 36 - Stream Gate Instance Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID For generic details on the Entry ID field, see Entry ID Management.
231-32	200	SGISE_DATA[1]	Stream Gate Instance State Element Data See SGISE_DATA_Format.
239-232	8	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.
247-240	8	ICFGE_DATA[1]	Initial Configuration Element Data See ICFGE_DATA_Format.
255-248	8	--	Reserved. Alignment padding.
383-256	128	AFGE_DATA[1]	Admin Configuration Element Data See ACFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 677. Table ID 36 - Stream Gate Instance Table ACFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ADMIN_SGCL_EID																																0x0
ADMIN_BASE_TIME																																0x4
ADMIN_BASE_TIME																																0x8
ADMIN_CYCLE_TIME_EXT																																0xC

Table 678. Table ID 36 - Stream Gate Instance Table ACFGE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	ADMIN_SGCL_EID	Administrative Stream Gate Control List Entry ID This field specifies the Entry ID in the Stream Gate Control List table (expressed in units of words) of the administrative stream gate control list to be used for this stream gate instance. 0x0..SGCLITCAPR[NUM_WORDS]-1 0xFFFF_FFFF (NULL); implies no Gate Control List specified. Other values are reserved
95-32	64	ADMIN_BASE_TIME	Admin Base Time This field sets the base time, that is, the start time, of the administrative gate control sequence. The time is expressed in units of nanoseconds. The field implements the PSFPAdminBaseTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 PSFPAdminBaseTime parameter, which is the PTP time format, consisting of a 48-bit integer number of seconds and a 32-bit integer number of nanoseconds. Instead of using the PTP time format, the field expects the time to be specified as a 64-bit integer number of nanoseconds. Therefore, if this field is to be set to a value which was obtained from a managed object using the IEEE 802.1Q-2018 PTP time format, the value must be converted by software from the IEEE 802.1Q-2018 format to the 64-bit integer number of nanoseconds. This field is reserved when ADMIN_SGCL_EID is NULL.
127-96	32	ADMIN_CYCLE_TIME_EXT	Admin Cycle Time Extension The value supplied in this field is equivalent with the 802.1.Qci definition of "Cycle Time Extension". When the administrative stream gate control list becomes operational, this is the maximum amount of time by which the cycle time is permitted to be extended before a new administrative stream gate control list takes effect. The time is expressed as an integer number of nanoseconds. Only least significant 30 bits are valid. Upper two bits are ignored. It is required that CYCLE_TIME in Stream gate control list + ADMIN_CYCLE_TIME_EXT must be < 2^{30} -1. This field is reserved when ADMIN_SGCL_EID is NULL.

Table 679. Table ID 36 - Stream Gate Instance Table ICFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																								--	CTD	GST	OIPV	IPV				0x0

Table 680. Table ID 36 - Stream Gate Instance Table ICFGE_DATA Description

Bits	Width [bits]	Name	Description
3-0	4	IPV	Internal Priority Value (IPV) This field specifies the IPV that will be assigned to frames before the first administrative stream gate control list takes affect, otherwise the IPV value is determined by the stream gate control list entry. Valid if OIPV of this entry is set to 1. The least significant 3 bits are used, most significant bit is reserved.
4	5	OIPV	Override Internal Priority Value (IPV) 0b = Don't override frame IPV. 01 = Override frame IPV with the value configured in the IPV field of this entry if the first administrative stream gate control list hasn't taken effect, otherwise overwrite with the IPV value stored in the stream gate control list.
5	1	GST	Gate State This field specifies the gate state to use before the administrative stream gate control list takes affect. 0b = Closed; frames are not permitted to pass through. 1b = Open; frames are permitted to pass through.
6	1	CTD	Cut Through Disabled This field specifies whether to disable cut-through before the administrative stream gate control list takes affect. 0b = No action. 1b = Cut-through is disabled. Note: Not applicable to ENETC function.
7	1	--	Reserved.

Table 681. Table ID 36 - Stream Gate Instance Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																								--				SDU_TYPE		IRXEN	OEXEN	0x0

Table 682. Table ID 36 - Stream Gate Instance Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
0	1	OEXEN	Octets Exceeded Enable 0b = "Gate Closed Due To Octets Exceeded" function is disabled. 1b = "Gate Closed Due To Octets Exceeded" function is enabled. See OEX for status of this function.

Table continues on the next page...

Table 682. Table ID 36 - Stream Gate Instance Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
1	1	IRXEN	Invalid Receive Enable 0b = "Gate Closed Due To Invalid Rx" function is disabled. 1b = "Gate Closed Due To Invalid Rx" function is enabled. See IRX for status of this function.
3-2	2	SDU_TYPE	Specifies the type of PDU/SDU (Protocol/Service Data Unit) for Interval Octets Maximum check for Gate Entry. Refer to IOM_GE field in SGCL table) Overhead values are specified by register PRXSDUOR. 00b = PPDU (Physical Layer PDU). 01b = MPDU (MAC PDU). 10b = MSDU (MAC SDU). All other values reserved.
7-4	4	--	Reserved.

Table 683. Table ID 36 - Stream Gate Instance Table SGISE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
OPER_SGCL_EID																																0x0
CONFIG_CHANGE_TIME																																0x4
CONFIG_CHANGE_TIME																																0x8
OPER_BASE_TIME																																0xC
OPER_BASE_TIME																																0x10
OPER_CYCLE_TIME_EXT																																0x14
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						--					STATE			IRX	OEX	0x18

Table 684. Table ID 36 - Stream Gate Instance Table SGISE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	OPER_SGCL_EID	Operational Stream Gate Control List Entry ID This field indicates the Entry ID in the Stream Gate Control List table (expressed in units of words) of the stream gate control list that is currently operational. Set to NULL (0xFFFF_FFFF) if there is no Operational Stream Gate Control List.
95-32	64	CONFIG_CHANGE_TIME	Configuration Change Time If an administrative gate control list is pending, this is the time at which administrative gate control list is to become operational (implements the PSFPConfigChangeTime parameter as defined by IEEE 802.1Q-2018). If there is no administrative gate control

Table continues on the next page...

Table 684. Table ID 36 - Stream Gate Instance Table SGISE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			list and if an operational gate control list exists then this is the time the operational list became active. The actual value stored by hardware (and thus available to query) is expressed in units of nanoseconds. If this field is to be queried from an IEEE 802.1Q-2018 managed object, the value must be converted by software from the 64-bit integer number of nanoseconds to the IEEE 802.1Q-2018 PTP time format.
159:96	64	OPER_BASE_TIME	Operational Base Time The base time (start time) of the operational stream gate control sequence. The actual value stored by hardware (and thus available to query) is expressed in units of nanoseconds. The field implements the PSFPOperBaseTime parameter as defined by IEEE 802.1Q-2018, however, the field does not use the time format defined for the IEEE 802.1Q-2018 PSFPOperBaseTime parameter, which is the PTP time format, consisting of a 48-bit integer number of seconds and a 32-bit integer number of nanoseconds. Instead of using the PTP time format, the field expresses the time as a 64-bit integer number of nanoseconds. Therefore, if this field is to be queried from an IEEE 802.1Q-2018 managed object, the value must be converted by software from the 64-bit integer number of nanoseconds to the IEEE 802.1Q-2018 PTP time format.
191-160	32	OPER_CYCLE_TIME_EXT	Oper Cycle Time Extension The value supplied in this field is equivalent with the 802.1.Qci definition of "Cycle Time Extension". If Operational Gate Control List is active, this is the maximum amount of time by which the cycle time is permitted to be extended before a new Admin Gate Control List takes effect. Only least significant 30 bits are valid. Upper two bits are ignored.
192	1	OEX	Octets Exceeded Flag 0b = Normal operation. 1b = Gate closed permanently due to octets exceed. If OEXEN = 1, detection of octets received exceeding INTERVAL_OCTET_MAX in a gate interval results in the stream gate being permanently set to closed. OEX can be cleared by performing Stream Gate State Element Update (SGISEU) action.
193	1	IRX	Invalid Receive Flag 0b = Normal operation. 1b = Gate closed permanently due to invalid receive. If IRXEN=1, detection of incoming frames during time periods when the stream gate is in the closed state results in the stream gate being permanently set to closed state. IRX can be cleared by performing Stream Gate State Element Update (SGISEU) action.
195-194	3	STATE	Current Gate Instance State 000b = Reserved. 001b = Gate instance is operational but no stream gate control list specified. Default Gate Instance parameters are applied to received frame. 010b = New administrative stream gate control list pending. Default Gate Instance

Table continues on the next page...

Table 684. Table ID 36 - Stream Gate Instance Table SGISE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			parameters will be applied to frames till administrative stream gate control list takes effect. 011b = New administrative stream gate control list is pending while stream gate control list is operational. 100b = No administrative stream gate control list is pending. Operational stream gate control list is active. 101b .. 111b = Reserved.
199-196	3	--	Reserved.

53.4.2.4.7.16 Stream Gate Control List Table

Table 685. Table ID 37 - Stream Gate Control List Table Common Attributes

TABLE_ID	37
TABLE_VERSION	0
Table Type	Dynamic bounded index table
Table Description	This table stores the stream gate control lists (SGCLs). A SGCL is used to specify the time gates or time slots during which incoming frames from one more streams will be accepted. Each SGCL can have a variable number of gate control operation entries. For each Stream Gate Instance used, only two SGCLs may be allocated to the instance. It is recommended that the entry ID for each of these two SGCLs be predetermined and fixed by software, and persist till the Gate Stream Instance is no longer in use. This table is implemented as a linear array of words, accessed using an index (0, 1, 2, ..., n) that uniquely identifies a word within the array. The words are stored in a contiguous manner in the common memory. When accessing the table, the actual memory position of a word is calculated by hardware from the index based on the size of the word and the base memory address of the table. Each table entry is of variable size, and is used to hold one stream gate control list. The first word of the entry contains information specific to the entire stream gate control list. Subsequent consecutive words are contains gate control operation (or gate) specific configuration. Each word can hold 2 gate configurations. Number of words required for a stream gate control list is 1+N/2 where N is number of gates in the stream gate control list. If N is odd, only one gate configuration will be specified in the last word. The Stream Gate Control Entry ID (SGCL_EID) has to be specified in units of words such that it points to the first word in a stream gate control list. Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) registers; S0SGCLITMAR[NUM_WORDS] for switch and EaSGCLITMAR[NUM_WORDS] for ENETC. These allocations can also be changed when the functions are initialized through their respective base function registers SGCLITMAR[NUM_WORDS]. Table management command operations are performed using Add, Delete or Query operations. Note that the Update operation is not supported for this table. Memory fragmentation may occur if stream gate control lists of different sizes are repeatedly added/deleted. There is no hardware assist to manage such memory fragmentation, software must take care of this situation.

Table continues on the next page...

Table 685. Table ID 37 - Stream Gate Control List Table Common Attributes (continued)

	Following are the behavior of operations: • Add operation: This operation adds a stream gate control list to the Stream Gate Control List table. Hardware will perform the following checks to determine if the Add operation is valid before executing the Add operation. — Checks if the specified stream gate control list is within the allocated region to the Stream Gate Control List table in common memory. — Checks if the common memory locations for the stream gate control list are not already allocated. — Checks if the cumulative gate time in the stream gate control list is <2^30. If above conditions are not met the operation will be rejected. • Delete operation: This operation deletes a stream gate control list from the Stream Gate Control List table. The SGCL_EID must point to the first word of the stream gate control list that is to be deleted. Following checks are performed before executing the Delete operation. — Checks if SGCL_EID is in valid range of words for this table. — Checks if the stream gate control list has already been added by software. — Checks if stream gate control list is not in use (that is,. stream gate control list is not operational nor installed as an administrative list. Note that REF_COUNT is non zero if the stream gate control list is in use. — If above conditions are not met the operation will be rejected. • Query operation: This operation queries a stream gate control list. The following checks are performed before executing the Query operation. — Checks if SGCL_EID is in valid range of words for this table. — Checks to see if the stream gate control list has already been added. — If above conditions are not met the operation will be rejected. Note that due to dynamic size of stream gate control lists that get added/deleted can cause memory fragments.		
Number of Entries	Maximum number of entries is indicated in SGCLITCAPR[NUM_WORDS]. Number of entries in-use is indicated in SGCLTMOR[NUM_ENTRIES]. Number of words required for a stream gate control list is 1+N/2 where N is number of gate time slots in the stream gate control list.		
Default Reset Behavior	No entries in the table by default.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete. 0x4 = Query. 0x8 = Add.
	Exact Match Key Element Match (KEY_ELEMENT)	No	--

Table continues on the next page...

Table 685. Table ID 37 - Stream Gate Control List Table Common Attributes (continued)

	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (SGCL_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	Variable	4
	SGCLSE_DATA	Stream Gate Control List State Element	1	1
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. x2D0 = Number words required for the LIST_LENGTH specified for the Add operation exceeds the number of words allocated for SGCL table. (i.e. SGCL_EID + (LIST_LENGTH+1)/2 (rounding to the next higher integer) has to be less than or equal to SGCLITCAPR [NUM_WORDS]-1). x2D1 = TIME_INTERVAL_GE_N specified in Add operation is 0. Note that upper 2 bits of TIME_INTERVAL_GE_N are ignored, TIME_INTERVAL_GE_N[29:0] must not be 0. x2D2 = Cumulated time value of TIME_INTERVAL_GE_N[29:0] for the gate list specified in Add operation is >= 2^30ns. Where N the List Length specified in operation. Where N is the List Length of the Stream Gate Control List.			

Table 686. Table ID 37 - Stream Gate Control List Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--																
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																...
CFGE_DATA																																Variable

Table 687. Table ID 37 - Stream Gate Control List Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions No update actions supported. Bits 15-0: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 37 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
Variable-64	Variable	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 688. Table ID 37 - Stream Gate Control List Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
--																						SGCLSE_DATA										0x4
CFGE_DATA																																0x8
...																																...
CFGE_DATA																																Variable

Table 689. Table ID 37 - Stream Gate Control List Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Stream, Gate Control List is a variable sized entry. Following has to be true. Entry ID is the first word in SGCL and SGCL_EID + (LIST_LENGTH+1)/2 (rounding to the next higher integer) has to be less than or equal to SGCLITCAPR [NUM_WORDS]-1 For generic details on the Entry ID field, see Entry ID Management.
39-32	8	SGCLSE_DATA [1]	Stream, Gate Control List State Element Data See SGCLSE_DATA_Format.
63-40	24	--	Reserved. Alignment padding.
Variable-64	Variable	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 690. Table ID 37 - Stream Gate Control List Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CYCLE_TIME																																0x0
--									EXT_GTST	EXT_CTD	EXT_IPV				EXT_OIPV	--						LIST_LENGTH										0x4
TIME_INTERVAL_GE_0																																0x8
GTST_GE_0	IOMEN_GE_0	CTD_GE_0	OPIV_GE_0	IPV_GE_0							IOM_GE_0																						0xC
...																																...
TIME_INTERVAL_GE_N																																0x8+8N
GTST_GE_N	IOMEN_GE_N	CTD_GE_N	OPIV_GE_N	IPV_GE_N							IOM_GE_N																						0xC+8N

Table 691. Table ID 37 - Stream Gate Control List Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	CYCLE_TIME	Cycle Time This field specifies the cycle time of the stream gate control list. This is the period of time over which the sequence of operations in a stream gate control list repeats. The time is expressed in units of nanoseconds. Only least significant 30 bits are valid. Upper two bits are ignored. This value has to be a non-zero value else operation will be rejected. It is also required that ADMIN_CYCLE_TIME + ADMIN_CYCLE_TIME_EXT must be < 2^30 -1.
39-32	8	LIST_LENGTH	List Length This field indicates the number of entries in the stream gate control list. Software must set this value to the size of the list being configured minus one. Valid values are 0 (one entry) to 255 (256 entries). The operation will be rejected if list length is invalid.
47-40	8	--	Reserved.
48	1	EXT_OIPV	Extension Override Internal Priority Value This field specifies whether to override the frame IPV after the stream gate control list ends and before CYCLE_TIME restarts. 0b = Don't override frame IPV. 1b = Override frame IPV with the value configured in the EXT_IPV field of this entry.
52-49	4	EXT_IPV	List Extension Internal Priority Value This field specifies the IPV to be assigned to frames after the stream gate control list ends and before CYCLE_TIME restarts. Valid if the EXT_OIPV field of this entry is set to 1. The least significant 3 bits are used, most significant bit is reserved.
53	1	EXT_CTD	Extension Cut Through Disabled This field specifies whether to disable cut-through for frames received after the stream gate control list ends and before CYCLE_TIME restarts. 0b = No action. 1b = Cut-through disabled. Note: Not applicable to ENETC function.
54	1	EXT_GTST	Extension Gate State This field specifies the gate state that will be applied after the stream gate control list ends and before CYCLE_TIME restarts. List started. 0b = Closed. 1b = Open.
63-55	9	--	Reserved.
(127 + 64i)	32	TIME_INTERVAL_GE_i	Time Interval for Gate Entry i, where i = 0, ..., N-1 This field specifies the time interval for a gate entry. The time is expressed in units of nanoseconds. Only least significant 30 bits are valid. Upper two bits are ignored.

Table continues on the next page...

Table 691. Table ID 37 - Stream Gate Control List Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
- (64 + 64i)			This field has to be a non-zero value and the cumulative TIME_INTERVAL amount in a Gate List must not exceed 2^30-1. Cumulated TIME_INTERVAL amount in a Gate List is allowed to be greater than CYCLE_TIME, in such case the CYCLE TIME restarts before Gate List completes.
	24	IOM_GE_i	Interval Octets Maximum for Gate Entry i, where i = 0, ..., N-1 This field specifies the maximum bytes (octets) allowed to pass (open) gate entry i when IOMEN is enabled.
	4	IPV_GE_i	Internal Priority Value for Gate Entry i, where i = 0, ..., N-1 This field specifies the Internal Priority Value (IPV) that will be assigned to frames. Valid if OIPV_GE_i field of this entry is set to 1. The least significant 3 bits are used, most significant bit is reserved.
	1	OIPV_GE_i	Override Internal Priority Value for Gate Entry i, where i = 0, ..., N-1 0b = Don't override frame IPV. 1b = Override frame IPV with the value configured in IPV_GE_i of field this entry.
	1	CTD_GE_i	Cut Through Disable for Gate Entry i, where i = 0, ..., N-1 This field specifies whether to disable cut-through for frames received in this gate. 0b = No action. 1b = Cut-through is disabled. Note: Not applicable to ENETC function.
	1	IOMEN_GE_i	Interval Octet Maximum Enabled for Gate Entry i, where i = 0, ..., N-1 This field specifies whether to track the number of bytes (octets) received in the gate when the gate is permanently closed; if OEXEN=1 and octets received in this gate exceeds IOM_GE. 0b = Don't track count. 1b = Track count.
	1	GTST_GE_i	Gate State for Gate Entry i, where i = 0, ..., N-1 0b = Closed. 1b = Open. Frames received in this gate are dropped if gate state is closed.

Table 692. Table ID 37 - Stream Gate Control List Table SGCLSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																								REF_COUNT								0x0

Table 693. Table ID 37 - Stream Gate Control List Table SGCLSE_DATA Description

Bits	Width [bits]	Name	Description
0	8	REF_COUNT	Reference Count Indicates whether a stream gate control list (SGCL) is in-use in a Stream Gate Instance (SGI). An SGCL becomes in-use when it is added to an SGI (as the administrative SGCL), and remains in-use when it is installed as the operational SGCL. An administrative SGCL is no longer considered in-use when if it is removed from an SGI. An operational SGCL is no longer in-use when it is replaced with a new SGCL or when the SGI is deleted. 0x00 = Not in-use by an SGI. 0x01 = In-use by an SGI. All other values reserved. Note: An attempt to delete an SGCL with a Reference Count set to 1 will result in an error response; in this case, the SGCL is not deleted from the Stream Gate Control List table.

53.4.2.4.7.17 Ingress Stream Count Table

Table 694. Table ID 38 - Ingress Stream Count Table Common Attributes

TABLE_ID	38		
TABLE_VERSION	0		
Table Type	Dynamic bounded index table		
Table Description	Ingress stream related statistics are maintained in this table. The Ingress Stream Count table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. The entries are stored in contiguous manner in a common memory. Ingress Stream Counter Entry ID (ISC_EID) is used as an index to an entry in this table. The ISC_EID is specified in the Ingress Stream table. For this table, each table entry occupies 1 word. Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) registers; S0ISCITMAR[NUM_WORDS] for switch and EaISCITMAR[NUM_WORDS] for ENETC. These allocations can also be changed when the functions are initialized through their respective base function register ISCITMAR[NUM_WORDS]. Table management operations Add, Delete, Update and Query are supported.		
Number of Entries	Maximum number of entries is indicated in ISCITCAPR[NUM_ENTRIES]. Number of entries in-use is indicated in ISCITOR[NUM_ENTRIES].		
Default Reset Behavior	No entries in the table by default.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete.

Table continues on the next page...

Table 694. Table ID 38 - Ingress Stream Count Table Common Attributes (continued)

			0x2 = Update. 0x4 = Query. 0x6 = Query, followed by Update. 0x8 = Add.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (IGSCNT_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	STSE_DATA	Statistics Element	32	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description.			

Table 695. Table ID 38 - Ingress Stream Count Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															STSEU	
ACCESS_KEY																																0x4

Table 696. Table ID 38 - Ingress Stream Count Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: STSEU - Statistics Element Update. 0b = No update performed to the Statistics Element. 1b = All counters within the Statistics Element are reset. All other values reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 38 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.

Table 697. Table ID 38 - Ingress Stream Count Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
STSE_DATA																																0x4
...																																...
STSE_DATA																																0x20

Table 698. Table ID 38 - Ingress Stream Count Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Valid range of ISC_EID:{0.. ISCITCAPR[NUM_ENTRIES]-1}. For generic details on the Entry ID field, see Entry ID Management.

Table continues on the next page...

Table 698. Table ID 38 - Ingress Stream Count Table Response Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
287-32	256	STSE_DATA[1]	Statistics Element Data See STSE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 699. Table ID 38 - Ingress Stream Count Table STSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
RX_COUNT																																0x0
--																																0x4
MSDU_DROP_COUNT																																0x8
--																																0xC
POLICER_DROP_COUNT																																0x10
																																0x14
SG_DROP_COUNT																																0x18
--																																0x1C

Table 700. Table ID 38 - Ingress Stream Count Table STSE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	RX_COUNT	Receive Count Count of received frames in the Ingress Stream table entry or Ingress Stream Filter table entry associated with this counter. \
95-64	32	MSDU_DROP_COUNT	MSDU Drop Count Count of received frames in the Ingress Stream table entry or Ingress Stream Filter table entry associated with this counter that did not pass the maximum SDU filter. Counter implemented in hardware is 32 bits.
159-128	32	POLICER_DROP_COUNT	Policer Drop Count Count of received frames in the Ingress Stream table entry or Ingress Stream Filter table entry associated with this counter that were dropped by the Policer.
223-192	32	SG_DROP_COUNT	Stream Gating Drop Count Count of received frames in the Ingress Stream table entry or Ingress Stream Filter table entry associated with this counter that did not pass the stream gate.

53.4.2.4.7.18 Egress Count Table

Table 701. Table ID 39 - Egress Count Table Common Attributes

TABLE_ID	39			
TABLE_VERSION	0			
Table Type	Static bounded index table			
Table Description	Egress related statistics are maintained in this table. The Egress Count table is implemented as a linear array of entries accessed using an index (0, 1, 2, ..., n) that uniquely identifies an entry within the array. Egress Counter Entry ID (EC_EID) is used as an index to an entry in this table. The EC_EID is specified in the Egress Treatment table. Egress Count table entries are always present and enabled.			
Number of Entries	Number of entries is indicated in ECTCAPR.			
Default Reset Behavior	All counters are reset to 0.			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x2 = Update (to reset the counters of an entry). 0x4 = Query. 0x6 = Query, followed by Update.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (EC_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	STSE_DATA	Statistics Element	8	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			

Table continues on the next page...

Table 701. Table ID 39 - Egress Count Table Common Attributes (continued)

ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. Note for this table the Access Method field is ignored and always treated as Entry ID Match. No error is generated if an unsupported Access Method is requested. Note for this table the Request Data Buffer must always be 8 bytes, as specified below. An invalid request data buffer length will not return an error.

Table 702. Table ID 39 - Egress Count Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															STSEU	
ACCESS_KEY																																0x4

Table 703. Table ID 39 - Egress Count Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: STSEU - Statistics Element Update. 0b = No update performed to the Statistics Element. 1b = All counters within the Statistics Element are reset. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions .
31-28	4	TABLE_VERSION	TID 39 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description.

Table continues on the next page...

Table 703. Table ID 39 - Egress Count Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.

Table 704. Table ID 39 - Egress Count Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
STSE_DATA																																0x4
...																																...
STSE_DATA																																0x10

Table 705. Table ID 39 - Egress Count Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID For generic details on the Entry ID field, see Entry ID Management.
159-32	128	STSE_DATA[1]	Statistics Element Data See STSE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 706. Table ID 39 - Egress Count Table STSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENQ_FRM_CNT																																0x0
ENQ_FRM_CNT																																0x4
REJ_FRM_CNT																																0x8
REJ_FRM_CNT																																0xC

Table 707. Table ID 39 - Egress Count Table STSE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	ENQ_FRM_CNT	Enqueued Frame Count Count associated with an Egress Treatment table entry that counts the number of frames enqueued on egress class queues.
127-64	64	REJ_FRM_CNT	Rejected Frame Count Count associated with an Egress Treatment table entry that counts the number of frames rejected in egress class queues, due to tail drop.

Request Data Buffer Format - See Request Data Buffer Format

Request Data Buffer Description - See Request Data Buffer Description

Response Data Format - See Response Data Buffer Format

Response Data Description - See Response Data Buffer Description

STSE_DATA Format - See STSE_DATA Format

STSE_DATA Description - See STSE_DATA Description

53.4.2.4.7.19 Frame Modification Table

Table 708. Table ID 40 - Frame Modification Table Common Attributes

TABLE_ID	40		
TABLE_VERSION	0		
Table Type	Dynamic bounded index table		
Table Description	Ingress and egress frame modification actions are specified in this table if Frame Modification entry ID (FM_EID) is used as an index. This table is implemented as a linear array of words, accessed using an index (0, 1, 2, ..., n) that uniquely identifies a word within the array. The words are stored in a contiguous manner in the common memory. When accessing the table, the actual memory position of a word is calculated by hardware from the index based on the size of the word and the base memory address of the table. For this table, each table entry occupies one word (one to one corresponding between words and entries). Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) register S0FMITMAR[NUM_WORDS]. This allocation can also be changed when the function is initialized through its respective base function register FMITMAR [NUM_WORDS].		
Number of Entries	Maximum number of entries is indicated in FMITCAPR[NUM_ENTRIES]. Number of entries in-use is indicated in FMITOR[NUM_ENTRIES].		
Default Reset Behavior	No entries in the table by default.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x1 = Delete.

Table continues on the next page...

Table 708. Table ID 40 - Frame Modification Table Common Attributes (continued)

			0x2 = Update. 0x4 = Query. 0x8 = Add.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (FM_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	24	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
Key Overlay Applicable	NA			
TABLE_ERROR_ID Codepoints	0x000 = no errors 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x300 = FM_EID format is invalid. 0x301 = Following fields specified are out of range - MAC_HDR_ACT, VLAN_HDR_ACT, SQT_ACT, OUTER_PCP_DEI_ACT, PLD_ACT. 0x302 = FMD_EID,FMD_BYTES specified is out of range. When FMD_EID is not set to Null, valid range is FMD_EID[15:0]*24 + FMD_BYTES <= (FMDITCAPR[NUM_WORDS]*24). Note also that if FMD_EID is not set to null, the valid range of bits for FMD_EID are [15:0]. Bits [31:16] are not used and will be ignored.			

Table 709. Table ID 40 - Frame Modification Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGU	
ACCESS_KEY																																0x4

Table continues on the next page...

Table 709. Table ID 40 - Frame Modification Table Request Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CFGE_DATA																																0x8
--																																--
CFGE_DATA																																0x1C

Table 710. Table ID 40 - Frame Modification Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 40 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
255-64	192	CFGE_DATA	Configuration Element Data See CFGE_DATA_Format.

Table 711. Table ID 40 - Frame Modification Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
CFGE_DATA																																0x4
--																																--
CFGE_DATA																																0x18

Table 712. Table ID 40 - Frame Modification Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID	Entry ID The Entry ID for the Frame Modification table,where the Entry ID encoded as an index into the Modification table. Valid range is {0..FMITCAPR[NUM_WORDS]-1}. For generic details on the Entry ID field, see Entry ID Management.
223-32	192	CFGE_DATA	Configuration Element Data See CFGE_DATA_Format.

Table 713. Table ID 40 - Frame Modification Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
DEST_MAC_ADDR																SMAC_PORT					SQT_ACT			OUTER_VID_ACT		VLAN_HDR_ACT		MAC_HDR_ACT			L2_ACT	0x0
DEST_MAC_ADDR																																0x4
--																PLD_ACT		OUTER_DEI_ACT		OUTER_PCP_ACT		OUTER_TPID_ACT		OUTER_VLAN_D		OUTER_VLAN_PCP		OUTER_VLAN_VID					0x8

Table continues on the next page...

Table 713. Table ID 40 - Frame Modification Table CFGE_DATA Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
																EI																
--																						PLD_OFFSET										0xC
--																FMD_BYTES																0x10
FMD_EID																																0x14

Table 714. Table ID 40 - Frame Modification Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
0	1	L2_ACT	Layer 2 Actions 0b = L2 actions are specified in L2 header action fields of this entry; i.e. MAC_HDR_ACT, VLAN_HDR_ACT and SQT_HDR_ACT 1b = The entire L2 PDU (or frame) is replaced with new L2 PDU of length FMD_BYTES specified by FMD_EID (FCS not present in the new L2 PDU). No L2 header actions or payload action in this entry should be specified. Note: Code point L2_ACT = 1b is not applicable for ingress frame modifications; if set to 1b, it will result in a misconfiguration event handled according to the port's PFMCR register. Note: This field must be set to 0 for traffic destined to a pseudo link. Note: This field must be set to 0 for any device with ASIL-B safety requirements.
3-1	3	MAC_HDR_ACT	Layer 2 Header MAC Actions 000b = no action 001b = Reserved. 010b = Replace SMAC with the contents of the PMAR0/1 register where port=SMAC_PORT 011b = Replace SMAC with the contents of the PMAR0/1 register where port=SMAC_PORT, and DMAC by specified DEST_MAC_ADDR field value. 100b = Replace SMAC with the contents of the PMAR0/1 register where port=SMAC_PORT, and DMAC by frame's SMAC 101b = Replace DMAC with specified DEST_MAC_ADDR field value 110b = Swap DMAC and SMAC 111b = Reserved. Note: MAC_HDR_ACT = 000b, 101b are the only valid code points for ingress frame modifications, any other code points

Table continues on the next page...

Table 714. Table ID 40 - Frame Modification Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			specified will result in a misconfiguration event handled according to the port's PFMCR register.
5-4	2	VLAN_HDR_ACT	Layer 2 VLAN Actions 00b = No action 01b = Delete outer VLAN header 10b = Add outer VLAN header. A new VLAN header is to be inserted in the outer position immediately following the SMAC address. The following action fields of this entry specify how to select the VID, PCP, DEI and TPID values that are to be used to construct the VLAN header. - OUTER_VID_ACT for VID - OUTER_PCP_ACT for PCP - OUTER_DEI_ACT for DEI - OUTER_TPID_ACT for TPID 11b = Replace outer VLAN header. The VLAN header in the outer position immediately following the SMAC address is to be modified. The following action fields of this entry specify how to select the VID, PCP, DEI and TPID values that are to be used to construct the VLAN header. - OUTER_VID_ACT for VID - OUTER_PCP_ACT for PCP - OUTER_DEI_ACT for DEI - OUTER_TPID_ACT for TPID Egress frame modifications assume that any ingress frame modifications specified have been applied to the frame prior to applying egress frame modifications. For example, if a frame is received with no VLAN tag, and a VLAN tag has been added as a result of ingress frame modification action, any references to the outer VLAN header of the frame in a egress frame modification action will use the VLAN header that was added by the ingress frame modification action. Similarly, if a frame is received with both outer and inner VLAN tags as being present and valid, and the outer VLAN tag has been deleted as a result of an ingress frame modification action, any references to the outer VLAN header in a egress frame modification action will use the inner VLAN header of the received frame. Note: For a Delete or Replace action, if no outer VLAN header is present, then a misconfiguration event will be generated and handled according to the port's PFMCR register. The outer VLAN header in a received frame is considered present if VLAN classification has deemed the outer VLAN header valid.

Table continues on the next page...

Table 714. Table ID 40 - Frame Modification Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
7-6	2	OUTER_VID_ACT	Outer VID Actions This field specifies how to select the VID value that is to be used to construct or modify the VLAN header. 00b = Use the VID from the outer VLAN header of the frame; in the case of a replace VLAN header action, this has the net effect of preserving the VID since the VID is being replaced by the same value. If no outer VLAN header is present then an misconfiguration event will be generated and handled according to the port's PFMCR register. The outer VLAN tag in a received frame is considered present if VLAN classification has deemed the outer VLAN tag valid. 01b = Use the VID specified in the OUTER_VLAN_VID field of this entry. 10b,11b = Reserved This field must be set to a non reserved value when VLAN_HDR_ACT field of this entry is set to 10b or 11b.
10-8	3	SQT_ACT	Sequence Tag Action 000b = No action. 001b = Remove R-TAG/draft 2.0 R-TAG/HSR tag; If R-TAG/draft 2.0 R-TAG/HSR tag not present, then a misconfiguration event will be generated and handled according to the port's PFMCR register, packet processing moves to the next function. 010b-111b: Reserved Note: Not applicable for ingress frame modifications; any code point other than 000b will result in a misconfiguration event handled according to the port's PFMCR register.
15-11	5	SMAC_PORT	Source MAC Address Register Port Specifies the port from which the MAC Address Register PMAR0/1 is to be read to obtain the Source MAC address. Valid if MAC_HDR_ACT=010b,011b,100b
63-16	48	DMAC	Destination MAC Address This field is defined in network byte order (big-endian). Most significant byte of the destination MAC address is stored at the lowest byte offset of this field. Valid if MAC_HDR_ACT = 011b,101b.
75-64	12	OUTER_VLAN_VID	Outer VLAN VID This field specifies the VID to be used in the outer VLAN header to be added or replaced when OUTER_VID_ACT field of this entry is set 01b.

Table continues on the next page...

Table 714. Table ID 40 - Frame Modification Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
78-76	3	OUTER_VLAN_PCP	Outer VLAN PCP This field specifies the 3-bit PCP to be used in the outer VLAN header to be added or replaced when OUTER_PCP_ACT field of this entry is set 001b.
79-79	1	OUTER_VLAN_DEI	Outer VLAN DEI This field specifies the 1-bit DEI to be used in the outer VLAN header to be added or replaced when OUTER_DEI_ACT field of this entry is set 01b.
82-80	3	OUTER_TPID_ACT	Outer TPID action Specifies the TPID value to be used to construct or modify the VLAN header. 000b = Use TPID from outer VLAN header, if no outer VLAN header is present then a misconfiguration event will be generated and handled according to the port's PFMCR register. The outer VLAN header in a received frame is considered present if VLAN classification has deemed the outer VLAN header valid. 001b = Set TPID to 0x8100 010b = Set TPID to 0x88A8 011b = Set TPID to C-VLAN as defined by CVLANR1[ETYPE] 100b = Set TPID to Custom S-VLAN as defined by CVLANR2[ETYPE] 101b-111b = Reserved This field must be set to a non reserved value when VLAN_HDR_ACT field of this entry is set to 10b or 11b.
85-83	3	OUTER_PCP_ACT	Outer PCP action This field specifies how to select the PCP value that is to be used to construct or modify the VLAN header. 000b = Use PCP value from outer VLAN header of the frame. 001b = Set to the PCP value specified in the OUTER_VLAN_PCP field of this entry. 010b = Use the PCP from outer VLAN header of the frame to access the PCP to PCP mapping profile specified in the port register PPCPDEIMR[IPCPMP/EPCPMP], to set the new PCP value. The actual PCP to PCP mapping is specified in register PCP2PCPMPaR, where a identifies the profile. For ingress modifications, port register PPCPDEIMR[IPCPMP] is used to select the mapping profile whereby for egress frame modifications, port register PPCPDEIMR[EPCPMP] is used to select the mapping profile. 011b = Use internal QoS associated with frame (IPV, DR) to access the QoS to PCP mapping profile specified in the egress port to set the new PCP value. The actual QOS to PCP mapping is specified in register QOSVLANMPaR0/1/2/3 where a identifies

Table continues on the next page...

Table 714. Table ID 40 - Frame Modification Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			the profile. 100b-111b: Reserved This field must be set to a non reserved value when VLAN_HDR_ACT field of this entry is set to 10b or 11b. Note: Code point OUTER_PCP_ACT = 011b is not applicable for ingress frame modifications; if set to 011b, it will result in a misconfiguration event handled according to the port's PFMCR register. Note: If access to the outer VLAN header is required (OUTER_PCP_ACT = 000b or 010b) and no outer VLAN is present, a misconfiguration event will be generated and handled according to the port's PFMCR register. The outer VLAN header in a received frame is considered present if VLAN classification has deemed the outer VLAN header valid. Note: If the PCP to PCP mapping profile specified in the port register PPCPDEIMR[IPCPMP/EPCPMP] is not valid, it will result in a misconfiguration event handled according to the port's PFMCR register.
87-86	2	OUTER_DEI_ACT	Outer DEI action This field specifies how to select the DEI value that is to be used to construct or modify the VLAN header. 00b = Use DEI value from outer VLAN header of the frame, if no outer VLAN header is present then a misconfiguration event will be generated and handled according to the port's PFMCR register. The outer VLAN header in a received frame is considered present if VLAN classification has deemed the outer VLAN header valid. 01b = Set to the DEI value specified in the OUTER_VLAN_DEI field of this entry. 10b = To construct the VLAN header, use internal DR associated with frame to access the DR to DEI mapping specified in the egress port register PPCPDEIMR[DRnDEI], to set new DEI value. To replace the VLAN header, preserve DEI if PPCPDEIMR[DRME]=0 or use the internal DR associated with frame to access the DR to DEI mapping specified in the egress port register PPCPDEIMR[DRnDEI], to set new DEI value, if PPCPDEIMR[DRME]=1. 11b = Reserved This field must be set to a non reserved value when VLAN_HDR_ACT field of this entry is set to 10b or 11b. Note: Code points OUTER_DEI_ACT = 10b is not applicable for ingress frame modifications; will result in a misconfiguration error according to the port's PFMCR register.

Table continues on the next page...

Table 714. Table ID 40 - Frame Modification Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
90-88	3	PLD_ACT	Payload Actions 000b = None 001b = Remove entire Ethernet payload including the payload Ethertype and insert with FMD_BYTES of data specified in FMD_EID. The the Ethernet payload (including the payload Ethertype) begins in the received frame immediately after source MAC address or VLAN tag(s) (up to 2) or R-TAG/HSR tag (if any). 010b = Replace FMD_BYTES of raw data in the Ethernet payload starting at PLD_OFFSET. Data is specified in FMD_EID. The maximum number of bytes that can be replaced is 128 bytes. If the frame is too short to replace all of the bytes specified by this replacement action, the Ethernet payload of the frame is replaced up to its last byte, and then the new calculated FCS is appended to the frame. The transmission of the frame completes without error. 011b-111b: Reserved Note: L2 header modification actions can be specified in combination to Ethernet payload modification actions. Ingress and egress L2 header modifications have no effect on the Ethernet payload modifications. When specifying an Ethernet payload modification action, the VLAN Ethertypes (2 standards, up to 2 customs) used to recognize a VLAN tag by the Ingress Packet Processing Parser must be set as acceptable Ethertypes for both the outer and inner VLAN tags (PTAR register). Note: Not applicable for ingress frame modifications; any value other than 000b will result in a misconfiguration event handled according to the port's PFMCR register. Note: This field must be set to 0 for traffic destined to a pseudo link. Note: This field must be set to 0 for any device with ASIL-B safety requirements.
95-91	5	--	Reserved.
103-96	8	PLD_OFFSET	Payload Offset Refer to PLD_ACT for more details Specifies the offset from the beginning of the payload Ethertype where replacement of FMD_BYTES starts. Valid if PLD_ACT= 010b.
127-104	24	--	Reserved.
143-128	16	FMD_BYTES	Frame Modification Bytes

Table continues on the next page...

Table 714. Table ID 40 - Frame Modification Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Specifies the number of bytes contained in the Frame Modification Data table for this entry, starting at word specified in FMD_EID. Valid if PLD_ACT = 001b,010b or L2_ACT = 1b. If L2_ACT = 1b, then the minimum number of bytes that can be specified in FMD_BYTES is 14 bytes.
159-144	16	--	Reserved.
191-160	32	FMD_EID	Frame Modification Data Entry ID FMD_EID is a pointer to a word in Frame Modification Data table for data replacement/insertion bytes. Must be valid if PLD_ACT = 001b, 010b or L2_ACT = 1b FMD_EID[15:0]*24 + FMD_BYTES has to be <= (FMDITCAPR[NUM_WORDS]*24) L2_ACT and PLD_ACT fields are ignored if this field is set to a null pointer (ffff). Note that if FMD_EID is not set to null the valid range of bits for FMD_EID are [15:0]. Bits [31:16] are not used and will be ignored.

53.4.2.4.7.20 Frame Modification Data Table

Table 715. Table ID 44 - Frame Modification Data Table Common Attributes

TABLE_ID	44
TABLE_VERSION	0
Table Type	Static bounded index table
Table Description	The Frame Modification Data table is a contiguous memory space in the common memory composed of 24-byte words. The number of 24-byte words in the Frame Modification Data table is defined in the register FMDITCAPR. The Entry ID is used to address a word within this memory space. Thus, Entry IDs are sequential from 0, 1, 2,..., N-1, where N is the number of 24-byte words. Neither the Add or Delete operation is supported for this table. All words are initialized and zeroed out of reset, and thus they may only be updated or queried. When updating, it is possible to update any number of bytes, which can span multiple words. The ENTRY_ID specifies the "base" word and the REQUEST_LENGTH field in the Table ID Request Message Header is used to implicitly specify the number of bytes to update. Since REQUEST_LENGTH includes the entire length of the request data buffer, including the four bytes of ACCESS_KEY and the four bytes of actions fields, the number of bytes to update will be interpreted as REQUEST_LENGTH minus these eight bytes. An Update operation must specify a number of bytes greater than zero to update, otherwise an error is returned.

Table continues on the next page...

Table 715. Table ID 44 - Frame Modification Data Table Common Attributes (continued)

	When updating, only the number of data bytes indicated by the REQUEST_LENGTH will be updated. For example, if REQUEST_LENGTH=d'12, then this indicates four data bytes to update and therefore only the first 4 bytes of the word specified by ENTRY_ID will be updated. Updates at an arbitrary location within a word, are not supported; updates are performed from the first byte in the word for as many bytes as were specified by the command. As mentioned earlier, an update may span multiple words. For example, if a command requests 50 bytes to be updated (REQUEST_LENGTH=d'58), the word at ENTRY_ID will be updated with the first 24 bytes of the request configuration data (see the Configuration Element Data definition), the word at ENTRY_ID + 1 will be updated with next 24 bytes, and the first two bytes of the word at ENTRY_ID + 2 will be updated with the last 2 bytes of the request configuration data. The remaining 22 bytes of the word at ENTRY_ID + 2 are undefined. When querying, it is possible to query any number of bytes, The ENTRY_ID specifies the "base" word and the RESPONSE_LENGTH field in the Table ID Request Message Header is used to specify the number of bytes to query. Since the RESPONSE_LENGTH field indicates the length of the entire response data buffer, including four bytes of ENTRY_ID, the number of bytes to query is the RESPONSE_LENGTH minus these four bytes. For example, if RESPONSE_LENGTH=d'12, then this indicates a request for eight data bytes to be queried, and the hardware will return the first eight data bytes at the word specified by ENTRY_ID in the query request message. Queries at an arbitrary location within a word are not supported; queries are performed from the first byte in the word for as many bytes as were specified by the command. A query may span multiple words. For example, if a command requests 50 bytes to be queried (RESPONSE_LENGTH=d'54), the word at ENTRY_ID will be will be returned in the first 24 bytes of the response configuration data (see the Configuration Element Data definition), the word at ENTRY_ID + 1 will be returned in bytes 24-47,and the first two bytes of the word at ENTRY_ID + 2 will be returned in bytes 48-49 of the response configuration data. A Query operation must specify a number of bytes greater than zero to query, otherwise an error is returned. It is possible to perform the query and update operations in the same command, and in this case, there is no requirement that the number of bytes to query be equal to the number of bytes being updated. When updating and/or querying, an error will be returned by any command which attempts to access (either reading or writing) memory which is outside the space allocated for this table. That is to say, the following must always be satisfied: numBytes + ENTRY_ID <= 24*N where N is the number of words in the Frame Modification Data table, specified in register FMDITCAPR; and where numBytes is the number of bytes on which to operate (whether for an update or query). Allocation of this table is first performed at pre-boot initialization time based on the setting of the Integrated Endpoint Register Block (IERB) register S0FMDITMAR[NUM_WORDS]. This allocation can also be changed when the function is initialized through its respective base function register FMDITMAR[NUM_WORDS].
Number of Entries	Number of entries/words is indicated in FMDITCAPR[NUM_WORDS].

Table continues on the next page...

Table 715. Table ID 44 - Frame Modification Data Table Common Attributes (continued)

Default Reset Behavior	All words in Frame Modification table are initialized to 0.			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x2 = Update. 0x4 = Query.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (FM_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	Variable	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
Key Overlay Applicable	NA			
TABLE_ERROR_ID Codepoints	0x000 = no errors 0x080-0x0FF = Generic codepoints; see ERROR field description.			

Table 716. Table ID 44 - Frame Modification Data Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGU	
ACCESS_KEY																																0x4

Table continues on the next page...

Table 716. Table ID 44 - Frame Modification Data Table Request Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CFGE_DATA																																0x8
--																																--
CFGE_DATA																																Variable

Table 717. Table ID 44 - Frame Modification Data Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 44 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
Variable-64	Variable	CFGE_DATA	Configuration Element Data See CFGE_DATA_Format.

Table 718. Table ID 44 - Frame Modification Data Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
CFGE_DATA																																0x4
--																																--
CFGE_DATA																																Variable

Table 719. Table ID 44 - Frame Modification Data Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID	Entry ID The Entry ID for the Frame Modification table,where the Entry ID encoded as an index into the Modification table. Valid range is {0..FMDITCAPR[NUM_WORDS]-1}. For generic details on the Entry ID field, see Entry ID Management.
Variable-32	Variable	CFGE_DATA	Configuration Element Data See CFGE_DATA_Format.

Table 720. Table ID 44 - Frame Modification Data Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
DATA																																0x0
--																																--
DATA																																Variable

Table 721. Table ID 44 - Frame Modification Data Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
Variable-0	Variable	DATA	Data Frame data that is used as payload data when frame data modification s performed. Data has to be aligned to word boundary and has to be specified in network byte order (big-endian).

53.4.2.4.7.21 Buffer Pool Table

Table 722. Table ID 41 - Buffer Pool Table Common Attributes

TABLE_ID	41			
TABLE_VERSION	0			
Table Type	Static bounded index table			
Table Description	The Buffer Pool table contains buffer pool configuration and operational information. Each entry corresponds to a buffer pool. The Entry ID value represents the buffer pool ID to access. See Switch Shared Internal Buffer Memory Management for a description of buffer pools. Buffer pools are always present and enabled.			
Number of Entries	Each Entry ID corresponds to a buffer pool ID. See BPCAPR[NUM_BP] register for the number of buffer pools.			
Default Reset Behavior	After reset all table fields are cleared to 0			
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**	
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x2 = Update. 0x4 = Query.	
	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (BP_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	20	4
	BPSE_DATA	Buffer Pool State Element	9	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			

Table continues on the next page...

Table 722. Table ID 41 - Buffer Pool Table Common Attributes (continued)

ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. 0x310 = SBP_EN is 1 and SBP_EID value is out-of-range in update command Note for this table the Access Method field is ignored and always treated as Entry ID Match. No error is generated if an unsupported Access Method is requested. Note for this table the Request Data Buffer must always be 28 bytes, as specified below. An invalid request data buffer length will not return an error, but may cause a corrupted table entry.

Table 723. Table ID 41 - Buffer Pool Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--														BPSEU	CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
...																																...
CFGE_DATA																																0x18

Table 724. Table ID 41 - Buffer Pool Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bit 1: BPSEU - Buffer Pool State Element Update. 0b = No update performed to the Buffer Pool State Element. 1b = Set the AMOUNT_USED_HWM to be the same as the current value of AMOUNT_USED. Bits 15-2: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved.

Table continues on the next page...

Table 724. Table ID 41 - Buffer Pool Table Request Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
			For generic details on the Query Actions code point, see Query Actions .
31-28	4	TABLE_VERSION	TID 41 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. This is the buffer pool ID to be configured or queried. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
223-64	160	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 725. Table ID 41 - Buffer Pool Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
BPSE_DATA																																0x4
...																																...
--																						BPSE_DATA										0xC
CFGE_DATA																																0x10
...																																...
CFGE_DATA																																0x20

Table 726. Table ID 41 - Buffer Pool Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Entry ID value represents the buffer pool ID to be accessed; range is 0..BPCAPR[NUM_BP]-1. For generic details on the Entry ID field, see Entry ID Management.

Table continues on the next page...

Table 726. Table ID 41 - Buffer Pool Table Response Data Buffer Description (continued)

Bits	Width [bits]	Name	Description
103-32	72	BPSE_DATA[1]	Buffer Pool State Element Data See BPSE_DATA_Format.
127-104	24	--	Reserved. Alignment padding.
287-128	160	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 727. Table ID 41 - Buffer Pool Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--				MAX_THRESH								PFC_VECTOR								--									FC_CFG		SBP_EN	0x0
--				FC_OFF_THRESH								--				FC_ON_THRESH																0x4
--												--				SBP_THRESH																0x8
SBP_EID																																0xC
FC_PORTS																																0x10

Table 728. Table ID 41 - Buffer Pool Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
0	1	SBP_EN	Shared Buffer Pool Enable 0b = No shared buffer pool is associated with this buffer pool. 1b = A shared buffer pool is associated with this buffer pool. The ID of the shared buffer pool is specified in the SBP_EID field of this entry. This setting should only be changed when the BP is first initialized, changing it when BP usage is non-zero may result in corrupted SBP usage count.
2-1	2	FC_CFG	Flow Control (FC) Configuration 00b = FC disabled. 01b = FC enabled using only buffer pool FC state. 10b = FC enabled using only shared buffer pool FC state. 11b = FC enabled using both buffer pool (BP) and shared buffer pool (SBP) FC state, the combined FC state of both BP and SBP must be 1 to assert FC ON, if either FC state is 0, FC OFF is asserted. Enabling flow control using the SBP is only valid if SBP_EN is set to 1. Setting FC_CFG = 10b or 11b and SBP_EN = 0 will result in undefined flow control behavior.

Table continues on the next page...

Table 728. Table ID 41 - Buffer Pool Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
7-3	5	--	Reserved.
15-8	8	PFC_VECTOR	Priority Flow Control (PFC) Vector When a flow control (FC) notification is sent (FC ON), this vector indicates which priorities are to be flow controlled for this buffer pool, one bit per PFC priority level. If the MAC is configured in link pause mode, this field is ignored. Valid if FC_CFG is not set 0. Not supported in NETC 3.0 and 3.1.
27-16	12	MAX_THRESH	Maximum Threshold The default setting of 0 disables maximum threshold checking, in which case all accounting update requests for this buffer pool (BP) will be accepted. If the threshold is non-zero, and BP usage is greater than or equal to this threshold, accounting update requests for this BP will be denied. This in turn, will result in discarding the frames associated with these denied accounting update requests. The threshold is in number of internal memory words (average of 20 bytes each), and is expressed as MANT = bits 11:4, EXP = bits 3:0, and threshold = MANT*2^{EXP}. Note that the maximum size of a buffer pool may go one maximum frame size beyond the value configured in this field as the datapath memory availability check for the buffer pool is performed without considering the length of the frame.
31-28	4	--	Reserved.
43-32	12	FC_ON_THRESH	Flow Control On Threshold If the buffer pool usage crosses this threshold, and if FC_ON_THRESH is greater than FC_OFF_THRESH, the flow control state of the buffer pool is set to 1. Valid if FC_CFG = 01b or 11b. The threshold is in number of internal memory words (average of 20 bytes each), and is expressed as MANT = bits 11:4, EXP = bits 3:0, and threshold = MANT*2^{EXP}.
47-44	4	--	Reserved.
59-48	12	FC_OFF_THRESH	Flow Control Off Threshold If buffer pool usage drops to this threshold or below, the flow control state of the buffer pool is set to 0. Valid if FC_CFG = 01b or 11b. The threshold is in number of internal memory words (average of 20 bytes each), and is expressed as MANT = bits 11:4, EXP = bits 3:0, and threshold = MANT*2^{EXP}.
63-60	4	--	Reserved.
75-64	16	SBP_THRESH	Shared Buffer Pool Threshold If buffer pool usage exceeds this threshold, the excess contribution is tracked in the shared buffer pool. Valid only if SBP_EN = 1. The threshold is in number of internal memory words (average of 20 bytes each), and is expressed as MANT = bits 11:4, EXP = bits 3:0, and threshold = MANT*2^{EXP}.

Table continues on the next page...

Table 728. Table ID 41 - Buffer Pool Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			This setting should only be changed when the BP is first initialized, changing it when BP usage is non-zero may result in corrupted SBP usage count.
95-76	20	--	Reserved.
127-96	32	SBP_EID	Shared Buffer Pool Entry ID This field indicates the shared buffer pool ID to which this buffer pool is associated with. Valid if SBP_EN = 1. This setting should only be changed when the BP is first initialized, changing it when BP usage is non-zero may result in corrupted SBP usage count.
159-128	32	FC_PORTS	Flow Control Ports When a flow control notification is sent (FC ON), this is a per-port bitmap indicates which ports are to be flow controlled for this buffer pool. Each bit of the bitmap corresponds to a port on the switch. Least significant bit of the bitmap corresponds to the smallest port number. The port(s) to be flow controlled, are identified by having their corresponding bit set to 1 in the bitmap Valid if FC_CFG is not set to 0.

Table 729. Table ID 41 - Buffer Pool Table BPSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
AMOUNT_USED																																0x0
AMOUNT_USED_HWM																																0x4
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						--								BPD	FC_STATE	0x8

Table 730. Table ID 41 - Buffer Pool Table BPSE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	AMOUNT_USED	Amount Used Buffer pool usage count, number of internal memory words (average of 20 bytes each) currently in use in this buffer pool. Includes both the dedicated portion (amount <= SBP_THRESH) and the shared portion (amount > SBP_THRESH).
63-32	32	AMOUNT_USED_HWM	Amount Used High Watermark Buffer pool usage high watermark, value sticks at the highest AMOUNT_USED seen since the last watermark reset.

Table continues on the next page...

Table 730. Table ID 41 - Buffer Pool Table BPSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			When the watermark is reset (update command with BPSEU = 1), AMOUNT_USED_HWM is set equal to AMOUNT_USED.
64	1	FC_STATE	Flow Control (FC) State FC state will assert to ON if FC is enabled, and if the FC ON threshold is exceeded, and if the usage is below the FC OFF threshold, i.e. must have OFF threshold < ON threshold. Once asserted, the state will go to OFF when the usage drops below the FC OFF threshold.
65	1	BPD	Buffer Pool Disabled 0b = Buffer pool enabled 1b = This buffer pool has been disabled due to an uncorrectable ECC error. The buffer pool can only be re-enabled by a switch FLR.
71-66	7	--	Reserved.

53.4.2.4.7.22 Shared Buffer Pool Table

Table 731. Table ID 42 - Shared Buffer Pool Table Common Attributes

TABLE_ID	42		
TABLE_VERSION	0		
Table Type	Static bounded index table		
Table Description	The Shared Buffer Pool table contains shared buffer pool configuration and operational information. Each entry corresponds to a shared buffer pool. The Entry ID value represents the shared buffer pool ID to access. See Switch Shared Internal Buffer Memory Management for a description of shared buffer pools. Shared buffer pools are always present.		
Number of Entries	Each Entry ID corresponds to a shared buffer pool ID. See BPCAPR[NUM_SBP] register for the number of shared buffer pools.		
Default Reset Behavior	After reset all table fields are cleared to 0		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x2 = Update. 0x4 = Query.
	Exact Match Key Element Match (KEY_ELEMENT)	No	--

Table continues on the next page...

Table 731. Table ID 42 - Shared Buffer Pool Table Common Attributes (continued)

	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (SBP_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	8	4
	SBPSE_DATA	Shared Buffer Pool State Element	9	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. Note for this table the Access Method field is ignored and always treated as Entry ID Match. No error is generated if an unsupported Access Method is requested. Note for this table the Request Data Buffer must always be 16 bytes, as specified below. An invalid request data buffer length will not return an error, but may cause a corrupted table entry.			

Table 732. Table ID 42 - Shared Buffer Pool Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--														BPSEU	CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
CFGE_DATA																																0xC

Table 733. Table ID 42 - Shared Buffer Pool Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bit 1: BPSEU - Buffer Pool State Element Update. 0b = No update performed to the Buffer Pool State Element. 1b = Set the AMOUNT_USED_HWM to be the same as the current value of AMOUNT_USED. Bits 15-2: Reserved. For generic details on the Update Actions bitmap, see Update Actions.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 42 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. This is the shared buffer pool ID to be configured or queried. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
127-64	64	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 734. Table ID 42 - Shared Buffer Pool Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
SBPSE_DATA																																0x4
...																																...

Table continues on the next page...

Table 734. Table ID 42 - Shared Buffer Pool Table Response Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--																								SBPSE_DATA								0xC
CFGE_DATA																																0x10
CFGE_DATA																																0x14

Table 735. Table ID 42 - Shared Buffer Pool Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Entry ID value represents the shared buffer pool ID to be accessed; range is 0..BPCAPR[NUM_SBP]-1. For generic details on the Entry ID field, see Entry ID Management.
103-32	72	SBPSE_DATA[1]	Shared Buffer Pool State Element Data See BPSE_DATA_Format.
127-104	24	--	Reserved. Alignment padding.
191-128	64	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 736. Table ID 42 - Shared Buffer Pool Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--				MAX_THRESH												--																		0x0
--				FC_OFF_THRESH												--			FC_ON_THRESH															0x4

Table 737. Table ID 42 - Shared Buffer Pool Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
15-0	16	--	Reserved.
27-16	12	MAX_THRESH	Maximum Threshold If shared buffer pool usage is greater than or equal to this threshold, accounting update requests for any buffer pool configured to use this shared pool will be denied. This in turn, will result in discarding the frames associated with these denied accounting update requests. The threshold is in number of internal memory words (average of 20 bytes each), and is expressed as MANT = bits 11:4, EXP = bits 3:0, and threshold = MANT*2^{EXP}.

Table continues on the next page...

Table 737. Table ID 42 - Shared Buffer Pool Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			Note that the maximum size of a shared buffer pool may go one maximum frame size beyond the value configured in this field as the datapath memory availability check for the shared buffer pool is performed without considering the length of the frame.
31-28	4	--	Reserved.
43-32	12	FC_ON_THRESH	Flow Control On Threshold If shared buffer pool usage crosses above this threshold, and if FC_ON_THRESH is greater than FC_OFF_THRESH, the FC state of the shared buffer pool is set to 1. Valid if FC_CFG = 10b or 11b. The threshold is in number of internal memory words (average of 20 bytes each), and is expressed as MANT = bits 11:4, EXP = bits 3:0, and threshold = MANT*2^{EXP}.
47-44	4	--	Reserved.
59-48	12	FC_OFF_THRESH	Flow Control Off Threshold If shared buffer pool usage drops to this threshold or below, the FC state of the shared buffer pool is set to 0. Valid if FC_CFG = 10b or 11b. The threshold is in number of internal memory words (average of 20 bytes each), and is expressed as MANT = bits 11:4, EXP = bits 3:0, and threshold = MANT*2^{EXP}.
63-60	4	--	Reserved.

Table 738. Table ID 42 - Shared Buffer Pool Table SBPSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
AMOUNT_USED																																0x0
AMOUNT_USED_HWM																																0x4
xxxxxxxxxxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxxxxxxxxxx																						--									FC_STATE	0x8

Table 739. Table ID 42 - Shared Buffer Pool Table SBPSE_DATA Description

Bits	Width [bits]	Name	Description
31-0	32	AMOUNT_USED	Amount Used Shared buffer pool usage count, number of internal memory words (average of 20 bytes each) currently in use in this shared buffer pool. A shared pool has a set of pools as members and tracks only the amount from all member pools that have exceeded the pools dedicated portion (SBP_THRESH of

Table continues on the next page...

Table 739. Table ID 42 - Shared Buffer Pool Table SBPSE_DATA Description (continued)

Bits	Width [bits]	Name	Description
			each member pool). If a pool is below its dedicated consumption then there is no contribution to its associated shared pool.
63-32	32	AMOUNT_USED_HWM	Amount Used High Watermark Shared buffer pool usage high watermark, value sticks at the highest AMOUNT_USED seen since the last watermark reset. When the watermark is reset (update command with BPSEU = 1), AMOUNT_USED_HWM is set equal to AMOUNT_USED.
64	1	FC_STATE	Flow Control State FC state will assert to ON if FC is enabled, and if the FC ON threshold is exceeded, and if the usage is below the FC OFF threshold, i.e. must have OFF threshold < ON threshold. Once asserted, the state will go to OFF when the usage drops below the FC OFF threshold.
71-65	7	--	Reserved.

53.4.2.4.7.23 ETM Congestion Group Table

Table 740. Table ID 43 - ETM Congestion Group Table Common Attributes

TABLE_ID	43		
TABLE_VERSION	0		
Table Type	Static bounded index table		
Table Description	The ETM Congestion Group table contains congestion group configuration and operational information. Each entry corresponds to a congestion group on given switch port. Each class queue within a port is mapped to a single congestion group, and each congestion group can have 0 or more class queues as members (n to 1 mapping). By default each class queue is mapped to a separate congestion group, see the ETM Class Queue Table for details of the mapping. Congestion groups are always present and enabled.		
Number of Entries	The number of congestion groups is equal to the number of class queues in the switch, see "ETM Class Queue Table".		
Default Reset Behavior	After reset all table fields are cleared to 0, all tail drop thresholds disabled.		
Access Method (ACCESS_METHOD)	**Access Method**	**Supported**	**Restrictions**
	Entry ID Match (ENTRY_ID)	Yes	Following table management command operations are supported: 0x2 = Update. 0x4 = Query.

Table continues on the next page...

Table 740. Table ID 43 - ETM Congestion Group Table Common Attributes (continued)

	Exact Match Key Element Match (KEY_ELEMENT)	No	--	
	Ternary Match Key Element Match (KEY_ELEMENT)	No	--	
	Search (SEARCH_CRITERIA)	No	--	
Data Buffer Component Attributes	**Name**	**Description**	**Size [bytes]**	**Alignment [bytes]**
	ENTRY_ID	Entry Id (SBP_EID)	4	4
	ACCESS_KEY	Access Key	4	4
	CFGE_DATA	Configuration Element	10	2
	STSE_DATA	Statistics Element	12	4
Entry Alignment [bytes]	4			
Entry ID Management	Software assigns entry IDs.			
Response STATUS Applicable	No			
ERROR codepoints specific to this table	0x000 = no errors 0x001-0x07F = reserved 0x080-0x0FF = Generic codepoints; see ERROR field description. Note for this table the Access Method field is ignored and always treated as Entry ID Match. No error is generated if an unsupported Access Method is requested. Note for this table the Request Data Buffer must always be 18 bytes, as specified below. An invalid request data buffer length will not return an error, but may cause a corrupted table entry.			

Table 741. Table ID 43 - ETM Congestion Group Table Request Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
TABLE_VERSION				QUERY_ACTIONS				--								UPDATE_ACTIONS																		0x0
																--															CFGEU	
ACCESS_KEY																																0x4
CFGE_DATA																																0x8
CFGE_DATA																																0xC
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																CFGE_DATA																0x10

Table 742. Table ID 43 - ETM Congestion Group Table Request Data Buffer Description

Bits	Width [bits]	Name	Description
15-0	16	UPDATE_ACTIONS	Update Actions Bit 0: CFGEU - Configuration Element Update. 0b = No update performed to the Configuration Element. 1b = The data specified in the CFGE_DATA field is written to the Configuration Element of the table entry. Bits 15-1: Reserved. For generic details on the Update Actions bitmap, see Update Actions. There is no update action provided to the Statistic Element, because the field in the Statistics Element represents a usage count of the congestion group; no specific reset on this field is allowed.
23-16	8	--	Reserved
27-24	4	QUERY_ACTIONS	Query Actions 0x0 = Full query. All other values reserved. For generic details on the Query Actions code point, see Query Actions.
31-28	4	TABLE_VERSION	TID 43 Table Version This field specifies the format of the NTMP table request and response data buffers. 0x0 = the supported table version is 0; all other values are not supported.
63-32	32	ACCESS_KEY	Access Key If ACCESS_METHOD = 0x0 (Entry ID Match): Bits 31-0: ENTRY_ID. See Response Buffer Description. All other values for ACCESS_METHOD are not applicable to this Table. For generic details on the Access Key field, see Access Key.
143-64	80	CFGE_DATA [1]	Configuration Element Data See CFGE_DATA Format.

1. This component is present only for commands which perform an update or add. For more generic details on the element data blocks, see Element Data Blocks.

Table 743. Table ID 43 - ETM Congestion Group Table Response Data Buffer Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
ENTRY_ID																																0x0
STSE_DATA																																0x4
STSE_DATA																																0x8

Table continues on the next page...

Table 743. Table ID 43 - ETM Congestion Group Table Response Data Buffer Format (continued)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CFGE_DATA																																0xC
CFGE_DATA																																0x10
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																CFGE_DATA																0x14

Table 744. Table ID 43 - ETM Congestion Group Table Response Data Buffer Description

Bits	Width [bits]	Name	Description
31-0	32	ENTRY_ID [1]	Entry ID Note that the Entry ID value represents the ETM Congestion Group instance and is encoded as follows: Bits 3-0: Congestion Group ID (0..PCAPR[NUM_CG]). Bits 8-4: Switch Port ID (0..SCAPR0[NUM_PORT]-1). Bits 31-4: Reserved. Values outside of the above range will result in an error. For generic details on the Entry ID field, see Entry ID Management.
95-32	64	STSE_DATA[1]	Statistics Element Data See STSE_DATA_Format.
175-96	80	CFGE_DATA[1]	Configuration Element Data See CFGE_DATA_Format.

1. This component is present only for commands which perform a query. For more generic details on the element data blocks, see Element Data Blocks.

Table 745. Table ID 43 - ETM Congestion Group Table CFGE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
--			TD_DR0_THRESH													OAL												TD_DR3_EN	TD_DR2_EN	TD_DR1_EN	TD_DR0_EN	0x0
--			TD_DR2_THRESH													--			TD_DR1_THRESH														0x4
xxxxxxxxx NOT PART OF DATA BUFFER xxxxxxxxx																--			TD_DR3_THRESH														0x8

Table 746. Table ID 43 - ETM Congestion Group Table CFGE_DATA Description

Bits	Width [bits]	Name	Description
0	1	TD_DR0_EN	Tail Drop Enable for DR0 Frames 0b = Disable tail drop for frames with discard resiliency of 0. 1b = Enable tail drop for frames with discard resiliency of 0.
1	1	TD_DR1_EN	Tail Drop Enable for DR1 Frames 0b = Disable tail drop for frames with discard resiliency of 1. 1b = Enable tail drop for frames with discard resiliency of 1.
2	1	TD_DR2_EN	Tail Drop Enable for DR2 Frames 0b = Disable tail drop for frames with discard resiliency of 2. 1b = Enable tail drop for frames with discard resiliency of 2.
3	1	TD_DR3_EN	Tail Drop Enable for DR3 Frames 0b = Disable tail drop for frames with discard resiliency of 3. 1b = Enable tail drop for frames with discard resiliency of 3.
15-4	12	OAL	Overhead Accounting Length This is a 12-bit, 2's complement value (range -2048 to +2047) representing a fixed per-frame overhead to be added to the actual length of a frame when performing tail drop threshold comparisons. Frame length + OAL is capped at 14'h3FFF maximum, and 0 minimum.
28-16	13	TD_DR0_THRESH	Tail Drop Threshold for DR0 Frames Tail drop occurs if an enqueue is attempted with a frame marked as discard resiliency 0 (DR0), and this enqueue would cause the number of bytes in the congestion group to exceed the threshold configured in TD_DR0_THRES. Valid only if TD_DR0_EN = 1. Coding of the 13-bit threshold value is as follows: Bits 12-5: TA Bits 4-0: Tn Tail Drop Threshold = TA * 2^{Tn}
31-29	3	--	Reserved.
44-32	13	TD_DR1_THRESH	Tail Drop Threshold for DR1 Frames Same description as TD_DR0_THRESH
47-45	3	--	Reserved.
60-48	13	TD_DR2_THRESH	Tail Drop Threshold for DR2 Frames Same description as TD_DR0_THRESH
63-60	4	--	Reserved.
76-64	13	TD_DR3_THRESH	Tail Drop Threshold for DR3 Frames Same description as TD_DR0_THRESH

Table continues on the next page...

Table 746. Table ID 43 - ETM Congestion Group Table CFGE_DATA Description (continued)

Bits	Width [bits]	Name	Description
79-77	3	--	Reserved.

Table 747. Table ID 43 - ETM Congestion Group Table STSE_DATA Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
BYTE_COUNT																																0x0
BYTE_COUNT																																0x4

Table 748. Table ID 43 - ETM Congestion Group Table STSE_DATA Description

Bits	Width [bits]	Name	Description
63-0	64	BYTE_COUNT	Byte Count Number of bytes currently in use in all class queues that are members of this congestion group. This count includes the OAL.

53.4.2.4.7.24 Frame Modification Entry Definition

The Entry ID for the Frame Modification table can be one of three distinct overlaid encodings. The first option (option 1 below) has the Entry ID encoded as an index into the Frame Modification table. The other two encoding options have the modification actions encoded in the Entry ID itself, allowing the configuration of basic frame modification actions without needing to configure (or add) a Frame Modification table entry. Format and description of the Frame Modification Entry ID are detailed in the tables below.

Table 749. Frame Modification Entry ID Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
0	0	0	Index													0x0
0	0	1	0	-							SQTA			VUDA		0x0
0	1	VARA		VID												0x0

Table 750. Frame Modification Entry ID Description

Bits	Width (Bits)	NAME	Description
12-0	13	Index	Index to Frame Modification Table Entry. This field is valid when bits 15-13 are set to 000b; i.e. the Frame Modification Entry ID is encoded as an Index into the Modification table. 0x0..FMITCAPR[NUM_WORDS]-1
1-0	2	VUDA	VLAN Update/Delete Action 00b = No Action. 01b = Replace VLAN's PCP/DEI based on the port's PPCPDEIMR. The tag's original

Table continues on the next page...

Table 750. Frame Modification Entry ID Description (continued)

Bits	Width (Bits)	NAME	Description
			VID and TPID are preserved. Uses the PCP from outer VLAN header of the frame to access the PCP to PCP mapping profile specified in the port register PPCPDEIMR[IPCPMP/EPCPMP], to set the new PCP value. The actual PCP to PCP mapping is specified in the register PCP2PCPMPaR, where *a* identifies the profile. For ingress modifications, port register PPCPDEIMR[IPCPMP] is used to select the mapping profile whereby for egress frame modifications, port register PPCPDEIMR[EPCPMP] is used to select the mapping profile. Preserves DEI if PPCPDEIMR[DRME]=0 or uses the internal DR associated with frame to access the DR to DEI mapping specified in the egress port register PPCPDEIMR[DRnDEI], to set new DEI value, if PPCPDEIMR[DRME]=1. 10b = Delete Outer VLAN Tag. 11b = Reserved. This field is valid when bits 15-12 are set to 0010b; i.e. the Frame Modification Entry ID is encoded as basic VLAN/R-TAG/HSR tag modification actions. Note: Misconfiguration error if replace or delete action is specified and if VLAN tag is not present in frame.
4-2	3	SQTA	Sequence Tag Action 000b = No Action. 001b = Remove R-TAG/HSR tag; If R-TAG/HSR tag not present, misconfiguration error. 010b .. 111b = Reserved. This field is valid when bits 15-12 are set to 0010b; i.e. the Frame Modification Entry ID is encoded as basic VLAN/ R-TAG/HSR tag modification actions. Note: SQTA must be set to 000b for Ingress frame modification, otherwise misconfiguration error.
11-0	12	VID	VLAN ID This field is valid when bits 15-14 are set to 01b; i.e. the Frame Modification Entry ID is encoded as basic VLAN modification actions.
13-12	2	VARA	VLAN Add/Replace Action 00b = Add VLAN with VID and PCP/DEI updated as described below. TPID=0x8100 01b = Add VLAN with VID and PCP/DEI updated as described below. TPID=0x88A8 10b = Replace VLAN with VID. The tag's original PCP, DEI and TPID are preserved. 11b = Replace VLAN with VID and PCP/DEI updated by port's PPCPDEIMR. The tag's original TPID is preserved. Uses the PCP from outer VLAN header of the frame to access the PCP to PCP mapping profile specified in the port register PPCPDEIMR[IPCPMP/EPCPMP], to set the new PCP value. The actual PCP to PCP mapping is specified in the register PCP2PCPMP*a*R, where *a* identifies the profile. For ingress modifications, the port register PPCPDEIMR[IPCPMP] is used to select the mapping profile whereby for egress frame modifications, the port register PPCPDEIMR[EPCPMP] is used to select the mapping profile. Preserves DEI if PPCPDEIMR[DRME]=0 or uses the internal DR associated with frame to access the DR to DEI mapping specified in the egress port register PPCPDEIMR[DRnDEI], to set new DEI value, if PPCPDEIMR[DRME]=1. For ingress frame modification with 00b or 01b, use the ingress port to select PCP and DEI from the Bridge port default VLAN register (BPDVR). For egress frame modification with 00b or 01b, use the internal QoS associated with the frame (IPV, DR) to access the QoS to PCP mapping profile to set the new PCP value. The profile

Table continues on the next page...

Table 750. Frame Modification Entry ID Description (continued)

Bits	Width (Bits)	NAME	Description
			is determined in the PQOSMR[QVMP] register corresponding to the egress port being used. The actual QOS to PCP mapping is specified in register QOSVLANMPaR0/1/2/3 where 'a' corresponds to the profile selected. Use internal DR associated with frame to access the DR to DEI mapping specified in the register PPCPDEIMR[DRnDEI] corresponding to the egress port being used, to set new DEI value. This field is valid when bits 15-14 are set to 01b; i.e. the Frame Modification Entry ID is encoded as basic VLAN modification actions. Note: Misconfiguration error if replace action is specified and if VLAN tag is not present in frame. Note: For the option 11b, if the PCP to PCP mapping profile specified in the port register PPCPDEIMR[IPCPMP/EPCPMP] is invalid, it results in a misconfiguration event handled according to the port's PFMCR register.

53.4.2.5 IEEE 1588 timer module

53.4.2.5.1 Overview

The (IEEE 1588) Timer module provides the current time with nanosecond resolution, precise periodic pulse, Pulse on timeout (alarm), and time capture on external pulse support. This blocks capabilities support implementing time synchronization as required for IEEE 1588 and IEEE 802.1AS-2020.

53.4.2.5.1.1 Block diagram

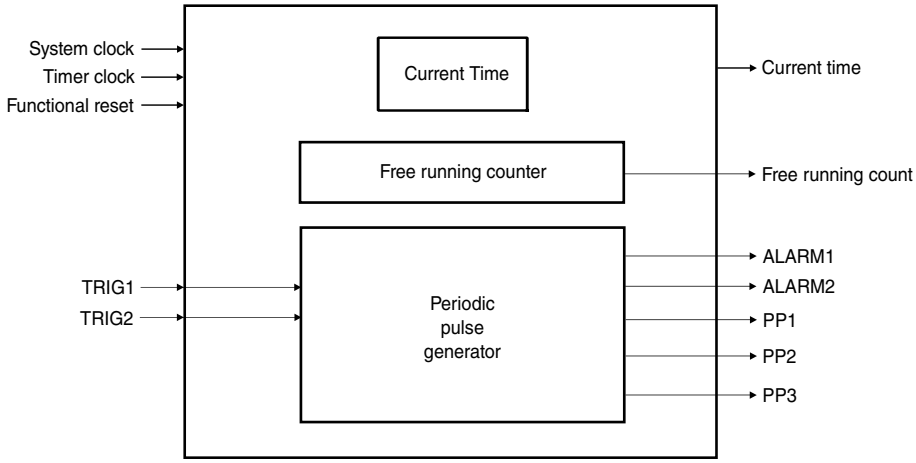


Figure 361. Block diagram

53.4.2.5.1.2 Features

The Timer module includes the following features:

- 64-bit running time with nanosecond resolution
- 64-bit free running time (NETC clock ticks)
- Software selectable clock
- 64-bit time offset to allow adjustment of local time to global time

- Software adjustable tick generation to allow adjustment for clock error and drift
- Time stamp capture from external GPIO input
  - Programmable input pulse polarity
  - Maskable interrupt on event
- Alarms generated at a future time
  - Programmable pulse to GPIO
  - Maskable interrupt
- Periodic pulse generation
  - Programmable pulse to GPIO
  - Maskable interrupt

53.4.2.5.2 Functional description

Note that the Timer has different hardware clock domains. Some registers are clocked by the standard system clock and are accessible whenever the Timer Function's PCIe Memory System Enable (MSE) bit is set. Other registers are clocked by the selected Timer clock and are only accessible when an active clock has been selected by setting TMR_CTRL[CK_SEL] and the module has been enabled by setting TMR_CTRL[TE]. See Table 751 for the list of registers which are in the timer clock domain. Table 752 lists some relevant registers in the normal clock domain.

Table 751. Timer Clock Domain Register List

Register Name
TMR_CNT_H - timer counter high register
TMR_CNT_L - timer counter low register
TMR_ADD - Timer drift compensation addend register
TMR_ACC - Timer accumulator register
TMR_PRSC - timer prescale
TMROFF_H - Timer offset high
TMROFF_L - Timer offset low
TMR_ALARM1_H - Timer alarm 1 high register
TMR_ALARM1_L - Timer alarm 1 low register
TMR_ALARM2_H - Timer alarm 2 high register
TMR_ALARM2_L - Timer alarm 2 low register
TMR_FIPER1 - Timer fixed period interval
TMR_ECTRL - Extended timer control register
TMR_CUR_TIME_L - Timer current time low register
TMR_CUR_TIME_H - Timer current time high register
TMR_TIME_CTRL - Timer time control register
TMR_FIPER2 - Timer fixed period interval
TMR_FIPER3 - Timer fixed period interval

Table continues on the next page...

Table 751. Timer Clock Domain Register List (continued)

Register Name
TMR_ETTS1_H - Time-stamp of general purpose external trigger
TMR_ETTS1_L - Time-stamp of general purpose external trigger
TMR_ETTS2_H - Time-stamp of general purpose external trigger
TMR_ETTS2_L - Time-stamp of general purpose external trigger

Table 752. Platform Clock Domain Register List

Register Name
TMR_ID - Module ID and version register
TMR_ID2 - Module ID and configuration register
TMR_CTRL - Timer control register
TMR_TEVENT - time-stamp event register
TMR_TEMASK - Timer event mask register
TMR_STAT - Timer Status
TMR_MSIVEC - Timer MSI-X vector register

53.4.2.5.2.1 Normal mode with drift and error adjustment

During normal operation, TMR_CNT increments by the value of TMR_CTRL[TCLK_PERIOD] and TMR_ACC increments by the value of TMR_ADD[ADDEND] every timer clock cycle. When TMR_ACC overflows, TMR_CNT increments by 1 to account for fractional part of the clock frequency. Software may update TMR_ADD to compensate for drift in timer clock.

The Timer module provides current time to other functions in the SoC such as time gate scheduling, time specific departure scheduling, stream gating and rate policing, and the Ethernet MAC for timestamp generation. Current time is equal to the 64-bit time in TMR_CNT_H/L plus the 64-bit offset in TMROFF_H/L, and also available in TMR_CUR_TIME_H/L.

53.4.2.5.2.2 Slave mode

In certain circumstances there may be other timers available in the system which provide 64-bit time values, perhaps multiple Timer modules. The Timer module can accept the time value from these other sources by putting it in slave mode by setting TMR_CTRL[SLV]. In this case the alarm, periodic pulse, and time stamp capture functionality of the Timer module is available, but the Timer module's 64-bit time and the timer clock adjust circuitry is not used.

53.4.2.5.2.3 Bypass mode

In some cases, an external phase, drift, and error adjusted clock is available. This clock may be selected by setting TMR_CTRL[CK_SEL] and used to directly update the 64-bit time by setting TMR_CTRL[BYP]. In this case the Timer module timer clock adjust accumulator is not used.

Note that TMR_CTRL[SLV] must be cleared ("master" mode selected) for bypass mode to work.

The time (in TMR_CNT) is updated by TMR_CTRL[TCLK_PERIOD] on each external clock.

53.4.2.5.2.4 Default counter mode

There is a Default nanosecond (ns) counter that can be used in a system that does not use 1588. This default ns counter is not dependent on configuring any of the 1588 functions.

The following three fields, inputs to the Timer block from the IERB, control the behaviour of this default counter:

1. ierb_load - which is the enable for the Default counter.

2. NETCCLKCR[FREQ], functionally equivalent to TMR_CTRL[TCLK_PERIOD], which is the clock frequency setting

3. NETCCLKFR[FRAC], functionally equivalent to TMR_ADD, that represents the fractional part of the clock period

There is a mux that chooses between Default ns counter and 1588 ns counter (TMR_CNT/TMR_CUR_TIME). When ierb_load is asserted and TMR_CTRL[TMR_TE] is 0, the Default counter starts counting and is the source for the ns output. If user configures 1588 and sets TMR_CTRL[TMR_EN]=1, then mux switches to choose 1588 ns counter and Default ns counter goes to 0 and stops counting

Please note that if 1588 function is required, user should configure all Timer registers before using any feature that depends on Default ns counter, else there could be a discontinuity in time of +/- 1 sec when the mux switches

Default counter value can be read from dedicated read only registers, TMR_DEF_CNT_L and TMR_DEF_CNT_H

When Default counter is enabled,and 1588 Timer is disabled (TMR_CTRL[TMR_TE]=0), TMR_CNT and TMR_CUR_TIME are not updated.

53.4.2.5.2.5 Clocking

This module works on two clocks; System clock and a separate Timer clock. Timer clock has two sources; the System clock and an externally connected clock. Timer clock selection is done by setting TMR_CTRL[CKSEL].

53.4.2.5.2.6 Reset

Table 753. Reset

Reset source	Description
Functional reset	Hardware reset for all registers.

53.4.2.5.2.7 Interrupts

Table 754 lists the various sources of 1588 timer related interrupts.

When an interrupt occurs TMR_TEVENT indicates the cause(s) of interrupts. Reading of TMR_TEVENT clears these indications.

Table 754. 1588 Module Interrupts

Interrupt	Description	Action taken by the 1588 module
ETS*n*	External trigger caused a time-stamp to be sampled.	None.
ALM*n*	Current time equaled the programmed alarm time.	The alarm output pin will be continuously driven asserted until TMR_ALARM*n* is disarmed.
PP*n*	Periodic pulse has been generated.	The FIPER*n* will continue pulsing until disabled.

53.4.2.5.2.8 Hardware assist for IEEE 1588/PTP compliant time keeping

IEEE 1588/802.1AS-2020 specify a mechanism for synchronizing multiple network connected nodes to the same clock based on external clock inputs from, for instance, GPS receivers. The Timer module provides functionality to support that and the following sections describe this functionality.

53.4.2.5.2.8.1 Timestamping of packets

The actual time stamping of packets is implemented by the Ethernet MAC using the time value provided by the Timer module.

The MAC is capable of recording the timestamp at the time of arrival of the frame at the MAC.

On transmit two types of timestamping are supported:

- "two-step" timestamping where the time of transmission of frame is recorded and provide to software
- "one-step" timestamping where the "Correction" field of a packet is updated based on the current time and the timestamp provided

Note that the one step form of Tx timestamping directly modifies the packet in a way which is compliant with the Precision Time Protocol (PTP). It does not simply insert the timestamp into the packet.

53.4.2.5.2.8.1.1 Rx timestamping

Each time a frame is received by the Ethernet MAC the current time is captured by the MAC at the start of the Start-of-Frame-Delimiter (SFD) for the frame. This timestamp is reported by the MAC with the packet and the lower 32b will be provided to software if the BD ring which the frame is directed to uses Extended RX Buffer Descriptor Format.

Rx timestamping of this form can be applied to any type of packet not just PTP packets.

53.4.2.5.2.8.1.2 Tx two-step timestamp

Two step Tx timestamping is the transmit equivalent of the Rx timestamping.

The main use case for Tx Timestamp is PTP (Precision Time Protocol) where it requires the Tx Timestamp when the frame was transmitted on the ethernet interface relative to the SFD.

There are multiple reasons why PTP protocol requires the Tx Timestamp information.

- For PTP two-step operation software utilizes the Tx Timestamp of the initial PTP frame (i.e.: sync, Pdelay_Resp) to update the correction field of the follow-up frame (i.e.: follow-up, Pdelay_Resp_Follow_Up).
- For Pdelay PTP frames, software utilizes the Tx Timestamp to calculate the peer delay.

This capability may also be used for non-PTP traffic if so desired.

When ENETC transmit a frame on an extended BD Ring with Extended Flags (E_FLAG) two-step timestamp offload set, the timestamp information is updated in the Transmit Descriptor buffer after successful transmit.

For ENETC connected attached to an Ethernet MAC, the timestamp is captured when the frame's SFD is transmitted by the Ethernet MAC.

For ENETC connected attached to a pseudo MAC, the timestamp is captured when the frame is dequeued and sent across the pseudo-link to the switch.

When ENETC is connected to the switch via pseudo MAC and software wants to capture the timestamp when the frame's SFD is transmitted by the switch's Ethernet MAC, the following procedure is followed:

1. ENETC Tx buffer descriptor has FL=00b, FLQ=10b, SMSO=1b (indicating direct switch enqueue), EGR_PORT set to the switch's port number and TSR=1b (Timestamp Reference Request).
2. After the frame is enqueued to the switch egress port, the Timestamp Identifier (TXTSID) is updated in the transmit descriptor.
3. After the switch dequeues and transmit the frame, it sends a transmit timestamp reference response message to ENETC's Rx BD Ring with Receive buffer descriptor containing the TXTSID, Timestamp value and a Host Reason of 0x3 for Timestamp response.
4. Since multiple timestamp reference requests may be sent prior to receiving a transmit timestamp reference response, the ENETC's Tx BD TXTSID is used to correlate with the receive transmit timestamp reference response to determine the correct response message.

NOTE

It is possible that a transmit timestamp reference response is never received due to an integrity fault within NETC.

53.4.2.5.2.8.1.3 Tx one-step timestamp

For one-step Tx timestamping the Ethernet MAC performs the following steps:

- Captures the current time as it sends the SFD for the frame

- Reads the 48b Correction field from the packet at the offset in the frame configured in PM*a*_SINGLE_STEP[OFFSET]
- Subtracts the timestamp provided by software with the packet from the SFD transmit time and adds that to the value read from The Correction field
- Writes the computed value back into the Correction field before transmission

One-step timestamping is used only with PTP frames.

Extended Tx BDs must be used, one-step timestamping must be enabled by setting the bit in the Extended Flags (E_FLAGS) field of the BD, and the packet specific timestamp must be supplied in the extended BD.

For use with the PTP protocol the timestamp provided is the lowest 30b of the time, read from the Timer modules time value, and placed in the PTP time field of the packet. Normally the Correction field of the packet is initialized to 0. This provides the receiver with the time at creation of the packet and a correction factor representing the time spent before transmission in a single message thus the name "one-step".

Note that the MAC is also capable of writing a correction word for the UDP checksum in the PTP packet, if the PTP in UDP is in use. This is enabled by setting PM*a*_SINGLE_STEP[CH].

Note that IEEE 1588 PTP one-step timestamping is not supported on the pseudo MAC.

53.4.2.5.2.8.2 Local synchronization of time

The Timer module supports synchronization of it's time with other, local (i.e. not network attached), time aware hardware. This includes the ability to take timestamps for external pulses (e.g. 1PPS inputs), and generate both periodic pulses (e.g. 1PPS outputs) and non-periodic pulses at specific future times.

53.4.2.5.2.8.2.1 Generating pulses at a specific future time (alarms)

The timer module is capable of generating a pulse on a GPIO and/or an interrupt at some future time. Two such alarms are supported and the 64-bit time is set in the appropriate TMR_ALARM*a*_H/L registers. Both alarms may not be assigned external GPIO pins in some SoCs.

The 64-bit alarm time is compared to the current time on each update and when the to match the alarm is triggered.
The comparison is for exact equivalence, so care must be taken that the alarm value is set to an integer multiple of the TMR_CTRL[TCLK_PERIOD] plus the offset and any initial value in the time otherwise the match will never occur.

The alarm can generate an interrupt to the host, if it is unmasked. It can also generate a pulse on an external GPIO. The polarity of the output signal is controlled with TMR_CTRL[ALM*a*P].

53.4.2.5.2.8.2.2 Generating periodic pulses

The Timer module can generate a periodic (Fixed Period - FIPER) pulse on a GPIO pin and/or an interrupt to the host. Three such periodic pulse sources are supported, and the period is set by programming the TMR_FIPER*a* registers.

The period is programmed as a number clock periods (TMR_CTRL[TCLK_PERIOD]). After that number of clock periods has passed, the pulse is generated, and the periodic timer starts over. In addition to generating a pulse an interrupt may also be generated if it is unmasked. Not all three periodic pulses may have GPIOs assigned on some SoCs.

It is possible to use ALARM1 to trigger the start of FIPER by setting TMR_CTRL[FS]. This allows having the periodic pulses occur at specific, controlled times.

53.4.2.5.2.8.2.3 Timestamping externals pulses

The Timer is capable of recording the timestamp on receipt of an external pulse on a GPIO pin. Up to two such external triggers may be supported, depending on the SoC.

The recorded value is saved in a 16 entry FIFO accessed through TMR_ETTS*a*_H/L. An interrupt can be generated when the trigger occurs, when the FIFO reaches a threshold, and if the FIFO overflows.

The polarity of the external trigger can be set using TMR_CTRL[ETEP*a*].

53.4.2.5.3 Initialization

Software must initialize the Timer module after cold boot. The following steps should be followed:

1. Set TMR_CTRL including CK_SEL. TE (Timer enable) should be enabled at this point and the clock selected must be active
2. Initialize TMR_TMASK
3. Initialize TMR_ADD
4. Set TMR_CTRL including TMR_CTRL[TCLK_PERIOD]

It is worth noting, again, that the clock selected must be active otherwise registers listed in Table 751 will not be accessible.

Setting of other registers, for instance alarm, periodic pulse (Fixed interval periodic pulse or FIPER), or external trigger registers, during this phase is permitted but optional.

53.4.2.5.3.1 Re-initialization

Re-initializing the Time module will disrupt the time produced by it which is used by various blocks throughout the system.

Users of the time include:

- Ethernet MACs for timestamping
- TSN related functionality such as Time gate scheduling (802.1Qbv) and Per-stream Filtering and Policing (PSFP)
- Media clock generation and recovery

Before re-initializing the Time module these users should be disabled or the time stamps they produce should be ignored.

Once use of the time is curtailed the following steps should be followed:

1. Disable Timer module by clearing TMR_CTRL[TE]
2. Perform a Function Level Reset (FLR) on the 1588 Function
3. Clear/disable any previously configured/enabled periodic pulse (FIPER), alarms, or time stamp capture settings
4. Clear outstanding events in TMR_TEVENT
5. Mask interrupts in TMR_TEMASK
6. Reconfigure the Timer modules as needed
7. Set TMR_CTRL including TMR_CTRL[TE]=1 to enable the Timer; both the synchronized timer and the free running timer are enabled

Users of the 64-bit time can then be re-enabled.

53.4.2.6 PHY Management Interface (MDIO)

The MDIO interface is a two-wire management interface. It provides a standardized method to access the management registers of PHY devices. NETC provides the following internal and external MDIO interfaces :

- One external master MDIO interface for managing external devices (PHYs). The external MDIO interface is controlled by EMDIO_* registers. EMDIO supports both Clause 22 and 45 protocols. The EMDIO provides a means for different software modules to share a single set of MDIO signals to access their PHYs. Each user can access its set of registers to initiate accesses on the MDIO and the EMDIO arbitrates between them, completing one access before proceeding with the next. It is required that each user of the EMDIO have exclusive access and control of its PHY.
- One internal MDIO interface per external link for managing Reverse Mode.
- One slave MDIO interface per RevMII-capable external link for managing Reverse Mode.

Depending on the PCI functions enabled in the system, the MDIO access points are determined as shown in the figure below.

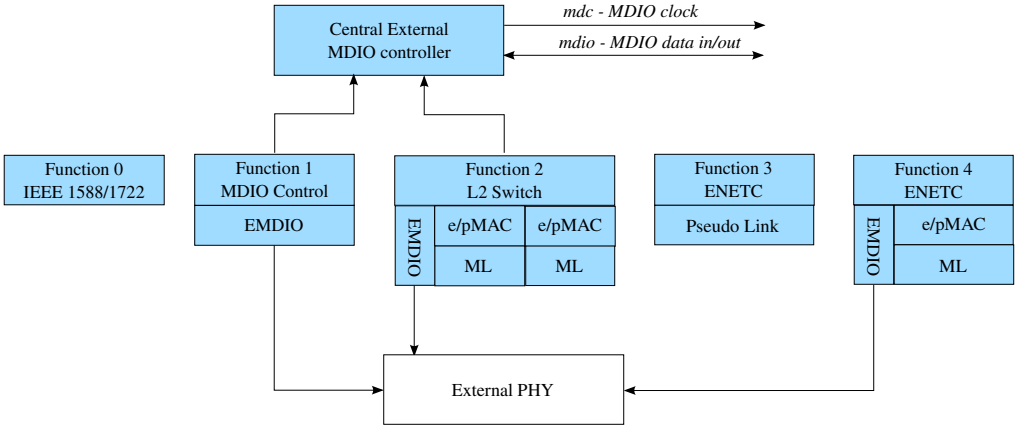


Figure 362. MDIO Access Points

53.4.2.6.1 Clause 22 External MDIO

A complete Clause 22 frame has a length of 64 bits, consisting of an optional 32-bit preamble, 14-bit command, 2-bit bus direction change and 16-bit data. Depending on the value of EMDIO_CFG[NEG], each bit is transferred on the rising or falling edge of MDIO clock (MDC signal). The MDIO data signal is tri-stated between frames.

Table 755. MDIO Clause 22 Frame Structure

ST	OP	PHYADDR	REGADDR	TA	DATA
2 bits	2 bits	5 bits	5 bits	2 bits	16 bits

Table 756. MDIO Clause 22 Frame Field Description

Field	Description
ST	Start indication. Value is 2'b01 for Clause 22 MDIO.
OP	Type of operation. 2'b01 for write, 2'b10 for read.
PHYADDR	PHY device address.
REGADDR	Register address.
TA	Turnaround time. Two bit-time is reserved for read operations to switch the data bus from write to read. PHY device will present its register contents in the data phase and drives the bus from the 2nd bit of the turnaround phase.
DATA	Data written to or read from PHY device.

EMDIO_CFG register should be configured before issuing any MDIO transactions.

- To generate a Clause 22 read transaction, follow these steps.
  1. Wait for EMDIO_CFG[BSY] = 0.
  2. Write EMDIO_CTL register with proper PHY_ADDR and REG_ADDR, with EMDIO_CTL[READ] set to 1 and EMDIO_CTL[POST_INC] cleared.
  3. Wait for EMDIO_CFG[BSY] = 0.
  4. Check EMDIO_CFG[MDIO_RD_ER]. If the addressed PHY does not respond, then EMDIO_CFG[MDIO_RD_ER] will be set.
  5. If no error, data is available in EMDIO_DATA register.

- To generate a Clause 22 write transaction, follow these steps.
  1. Wait for EMDIO_CFG[BSY] = 0.
  2. Write EMDIO_CTL register with proper PHY_ADDR and REG_ADDR, with EMDIO_CTL[READ] and [POST_INC] cleared.
  3. Write EMDIO_DATA register with proper data.

53.4.2.6.2 Clause 45 External MDIO

Clause 45 MDIO frame structure introduces indirect addressing. First, a write transaction to an address register is done, followed by a write or read transaction which will write the 16-bit data in EMDIO_DATA register to the device, or retrieve the register contents from the external device.

Table 757. MDIO Clause 45 Frame Structure

ST	OP	PORTADDR	DEVADDR	TA	DATA
2 bits	2 bits	5 bits	5 bits	2 bits	16 bits

Table 758. MDIO Clause 45 Frame Field Description

Field	Description
ST	Start indication. Value is 2'b00 for Clause 45 MDIO.
OP	Type of operation. 2'b00 for address write, 2'b01 for data write, 2'b10 for read, 2'b11 for read increment.
PORTADDR	MDIO port address.
DEVADDR	Device address (within a port).
TA	Turnaround time. Two bit-time is reserved for read operations to switch the data bus from write to read. PHY device will present its register contents in the data phase and drives the bus from the 2nd bit of the turnaround phase.
DATA	Data written to or read from PHY device.

EMDIO_CFG register should be configured before issuing any MDIO transactions.

- To generate a Clause 45 read transaction, follow these steps. Note that steps 2-4 can be skipped for subsequent accesses to the same PHY register.
  1. Wait for EMDIO_CFG[BSY] = 0.
  2. Write EMDIO_CTL register with proper PORT_ADDR and DEV_ADDR, with EMDIO_CTL[READ] and [POST_INC] cleared.
  3. Write EMDIO_ADDR register with proper device register address REG_ADDR.
  4. Wait for EMDIO_CFG[BSY] = 0.
  5. Write EMDIO_CTL register with proper PORT_ADDR and DEV_ADDR and read command set to 1. For normal read, set EMDIO_CTL[READ] to 1. For read and post-increment of PHY device register address, set EMDIO_CTL[POST_INC] to 1.
  6. Wait for EMDIO_CFG[BSY] = 0.
  7. Check EMDIO_CFG[MDIO_RD_ER]. If the addressed PHY does not respond, then EMDIO_CFG[MDIO_RD_ER] will be set.
  8. If no error, data is available in EMDIO_DATA register.

- To generate a Clause 45 write transaction, follow these steps. Note that steps 2-4 can be skipped for subsequent accesses to the same PHY register.
    1. Wait for EMDIO_CFG[BSY] = 0.
    2. Write EMDIO_CTL register with proper PORT_ADDR and DEV_ADDR, with EMDIO_CTL[READ] and [POST_INC] cleared.
    3. Write EMDIO_ADDR register with proper device register address REG_ADDR.
    4. Wait for EMDIO_CFG[BSY] = 0.
    5. Write EMDIO_DATA register with proper data.

53.4.2.6.3 Clause 22 PHY STATUS

EMDIO_CFG register should be configured before initiating PHY STATUS transactions.

- To initiate Clause 22 automatic read transactions, follow these steps.
    1. Wait for PHY_STATUS_CFG[BSY] = 0.
    2. Write PHY_STATUS_CTL register with proper PHY_ADDR and REG_ADDR.
    3. Write PHY_STATUS_MASK register with proper STATUS_MASK_LH and STATUS_MASK_HL.
    4. Write PHY_STATUS_CFG register with proper value of STATUS_INTERVAL.
    5. Wait for PHY_STATUS_CFG[BSY] = 0.
    6. Check PHY_STATUS_CFG[MDIO_RD_ER]. If the addressed PHY does not respond, then PHY_STATUS_CFG[MDIO_RD_ER] will be set.
    7. If no error, status information is available in PHY_STATUS_DATA register.
    8. Repeat steps 5,6,7 for successive reads of PHY STATUS.

53.4.2.6.4 Clause 45 PHY STATUS

EMDIO_CFG register should be configured before initiating PHY STATUS transactions.

- To initiate Clause 45 automatic read transactions, follow these steps.
    1. Wait for PHY_STATUS_CFG[BSY] = 0.
    2. Write PHY_STATUS_CTL register with proper PHY_ADDR and DEV_ADDR.
    3. Write PHY_STATUS_MASK register with proper STATUS_MASK_LH and STATUS_MASK_HL.
    4. Write PHY_STATUS_ADDR register with proper REGADDR.
    5. Wait for PHY_STATUS_CFG[BSY] = 0.
    6. Write PHY_STATUS_CFG register with proper value of STATUS_INTERVAL.
    7. Wait for PHY_STATUS_CFG[BSY] = 0.
    8. Check PHY_STATUS_CFG[MDIO_RD_ER]. If the addressed PHY does not respond, then PHY_STATUS_CFG[MDIO_RD_ER] will be set.
    9. If no error, status information is available in PHY_STATUS_DATA register.
    10. Repeat steps 7,8,9 for successive reads of PHY STATUS.

53.4.2.6.5 Clause 22 Internal MDIO

Clause 22 internal MDIO is used to access the RevMII controls. It is controlled via the eMAC MDIO registers for the respective port.

MDIO_CFG register should be configured before issuing any MDIO transactions.

- To generate a Clause 22 read transaction, follow these steps.
  1. Wait for MDIO_CFG[BSY] = 0.
  2. Write MDIO_CTL register with proper PHY_ADDR and REG_ADDR, with MDIO_CTL[READ] set to 1 and MDIO_CTL[POST_INC] cleared.
  3. Wait for MDIO_CFG[BSY] = 0.
  4. Check MDIO_CFG[MDIO_RD_ER]. If the addressed PHY does not respond, then MDIO_CFG[MDIO_RD_ER] will be set.
  5. If no error, data is available in EMDIO_DATA register.
- To generate a Clause 22 write transaction, follow these steps.
  1. Wait for MDIO_CFG[BSY] = 0.
  2. Write MDIO_CTL register with proper PHY_ADDR and REG_ADDR., with MDIO_CTL[READ] and [POST_INC] cleared.
  3. Write MDIO_DATA register with proper data.

53.4.2.7 Reverse Mode (RevMII)

In Reverse Mode, the link acts as a PHY rather than a MAC. The key differences between a MAC and a port acting as a PHY via Reverse Mode are:

- Instead of using an EMDIO port as a master, a PHY has an MDIO slave interface that supports Clause 22 operation.
- The PHY contains two sets of Clause 22 registers for PHY configuration, called the RevMII registers.
- In MII mode, the device drives MII_TXCLK and MII_RXCLK
- In RGMII mode, the PHY delays RGMII_TXC and RGMII_RXC relative to data as per RGMII specification for PHYs

The device configuration of the port in Reverse Mode is indicated by PIOCAPR[REVMII]=1. If PaIOCAPR[REVMII]=1, and LaMCAPR[MII_PROT]=0b000, then the speed of the interface is indicated by PaIOCAPR[REVMII_RATE]. Software must set the corresponding configuration bits in the MAC IF_MODE register. Reverse mode is supported in all interface modes.

The two sets of Clause 22 registers in the RevMII block contain the PHY configuration settings for PHY-side (representing internal device) and MAC-side (representing the link partner). The two sets of Clause 22 registers represent two virtual PHYs, with a virtual link between them, as shown in the following figure:

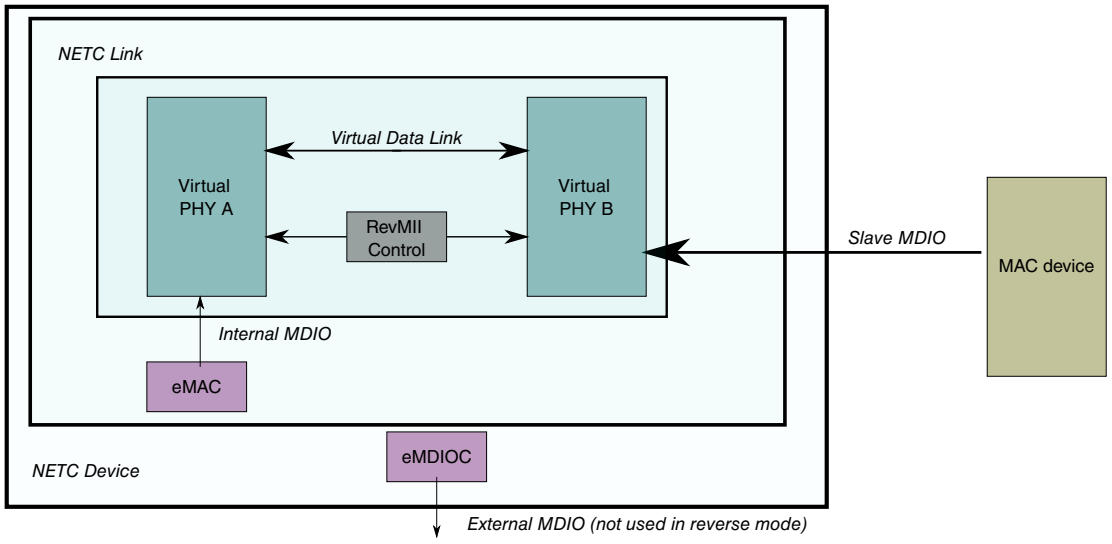


Figure 363. Reverse Mode

Accesses to PHY-side RevMII registers is over the internal MDIO interface from eMAC, using Clause 22 MDIO accesses with PHY Address 0h1F. Accesses to MAC-side RevMII MDIO space is via a per-link slave MDIO interface which supports Clause 22. When a link is configured for Reverse Mode, EMDIO is not supported for the link, and LaBCR[MDIO_PHYAD_PRTAD] determines the PHY address of the slave MDIO interface.

Application information:

- Half duplex is not supported for Reverse Mode.
- Reverse mode includes support for far-end loopback via MDIO CONTROL[14] on the MAC-side registers. If far-end loopback is enabled, received data is sent back out on the transmit interface without being sampled by the MAC reconciliation sublayer. Far-end loopback is supported for all modes, but for RGMII the link must be in synchronous operation (RGMII_TXC and RGMII_RXC from same clock source)
- Reverse mode includes support for link powerdown, via MDIO CONTROL[11], in which the interface transmit and receive functions are gated. Power down must be set to 0 for normal operation
- There is some asymmetry in the RevMII registers between MAC-side and PHY-side:
  - The MS_CTL[11] (MS_CFG_VALUE) is a RO bit. It will always read 0 for the MAC-side and 1 for the PHY-side. This also affects the state of MS_STA[14] (MS_CFG_RES) which is a read only copy of MS_CTL[11].
  - CONTROL[14] (LOOPBACK) is RW on the MAC-side and RO on the PHY-side.

53.4.2.7.1 Clause 22 Internal RevMII MDIO

The internal eMAC MDIO is used to access the REVMII registers.

For access to the RevMII register the PHY_ADDR must be set to 0h1f for both read and write transactions.

MDIO_CFG register should be configured before issuing any MDIO transactions.

- To generate a Clause 22 read transaction to the RevMII, follow these steps.
  1. Wait for MDIO_CFG[BSY] = 0.
  2. Write MDIO_CTL register with PHY_ADDR=0h1f and REG_ADDR, with MDIO_CTL[READ] set to 1 and MDIO_CTL[POST_INC] cleared.
  3. Wait for MDIO_CFG[BSY] = 0.

4. Check MDIO_CFG[MDIO_RD_ER]. If the addressed PHY does not respond, then MDIO_CFG[MDIO_RD_ER] will be set.
5. If no error, data is available in MDIO_DATA register.

- To generate a Clause 22 write transaction, follow these steps.
  1. Wait for MDIO_CFG[BSY] = 0.
  2. Write MDIO_CTL register with PHY_ADDR=0h1f and REG_ADDR, with MDIO_CTL[READ] and [POST_INC] cleared.
  3. Write MDIO_DATA register with proper data.

53.4.2.7.2 AN Theory of Operation

For AN enabled, initialization sequence is

- SW programs:
  1. AN advertisement register (offset 4)
  2. AN Next Page transmit register (offset 7)
  3. MASTER-SLAVE Control register (offset 9)
  4. CONTROL (offset 0), with AN enable = 1
- HW compares the values in offset 4,7,9 versus 5,8,10 to determine highest common speed mode
  - Speed is 1 Gbps if both 9.9 and 10.11 are set
  - else speed is 100 Mbps if both 4.9 and 5.9 are set
  - else speed is 10 Mbps if both 4.6 and 5.6 are set
  - If there is a common mode, then HW sets
    - STATUS[AN complete]=1
    - STATUS[Link Status]=1
  - If there is no common mode, HW does nothing (SW has to poll for AN complete and time out)
- PHY reset or setting CONTROL[Restart AN] re-trigger above speed calculation. Speed does not change if registers change without reset or restart AN.

For AN disable, initialization sequence is

- CONTROL (offset 0)[SPEED] with AN enable=0
  - Speed is 1 Gbps if CONTROL[SPEED]=10
  - else speed is 100 Mbps if CONTROL[SPEED]=01
  - else speed is 10 Mbps if CONTROL[SPEED]=00

53.4.2.7.2.1 AN Link Down Scenario: MAC initiated

MAC and PHY both go through regular initialization advertising support for 10/100/1000 speeds, and negotiate to 1 Gbps. MAC driver later decides to drop down to 100 Mbps, and changes advertised capability to remove 1000, then sets Restart AN=1

- MAC-side:
  - AN HW sets link down as a result of Restart AN
  - AN HW sees that highest common speed has changed to 100 Mbps, and changes its speed to 100 Mbps, setting AN done and link up
  - SW polls MDIO link state until it sees AN done and link up. Link up state is LL, so SW has to read it twice after AN done to see it set to 1

- SW changes MAC:IFMODE to match to match to new data link speed

- PHY-side:
  - PHY-side doesn't know about MAC-side's AN restart, but the HW sees that its speed choice is no longer matches what AN would choose, so it sets link down
  - SW polls MDIO link state and sees it's link down, and so sets AN restart
  - HW changes its speed to 100 Mbps and sets AN done and link up
  - SW polls MDIO link state until it sees AN done and link up. Link up state is LL, so SW has to read it twice after AN done to see it set to 1
  - PHY-side SW changes MAC:IFMODE to match to change actual data link speed

53.4.2.7.2.2 AN Link Down Scenario: PHY initiated

MAC and PHY both go through regular initialization advertising support for 10/100/1000 speeds, and negotiate to 1 Gbps. PHY driver later decides to drop down to 100 Mbps, and changes advertised capability to remove 1000, then sets Restart AN=1

- PHY-side:
  - AN HW sets link down as a result of Restart AN
  - AN HW sees that highest common speed has changed to 100 Mbps, and changes its speed to 100 Mbps, setting AN done and link up
  - SW polls MDIO link state until it sees AN done and link up. Link up state is LL, so SW has to read it twice after AN done to see it set to 1
- MAC-side:
  - MAC-side doesn't know about PHY-side's AN restart, but the HW sees that its speed choice is no longer matches what AN would choose, so it sets link down
  - SW polls MDIO link state and sees it's link down, and so sets AN restart
  - HW changes its speed to 100 Mbps and sets AN done and link up
  - SW polls MDIO link state until it sees AN done and link up. Link up state is LL, so SW has to read it twice after AN done to see it set to 1
  - SW changes MAC:IFMODE to match to match to new data link speed

53.4.2.8 Power Management

53.4.2.8.1 Power States

The PCIe Specification defines function power states (D-states) that enable the platform to establish and control power states for NETC ranging from fully on to fully off (drawing no power) and various in-between levels of power-saving states, annotated as D0-D3. Similarly, PCIe defines a series of link power states (L-states) that work specifically within the link layer. Since NETC is an integrated endpoint it does not have a PCIe link layer , and thus, there is no need to support link power states (L-states). In subsequent discussions mainly PCIe functions mastering transactions are considered, i.e. ENETC and switch. Other supporting NETC functions such as EMDIO and timer do not require any additional preparation to enter sleep state.

NETC only supports the required DO and D3 power states. These device states also indirectly reflect the states of the Ethernet link.

- $D0_{uninitialized}$ - a sub-state of D0
- $D0_{active}$ - a sub-state of D0
- $D3_{hot}$ - a sub-state of D3
- $D3_{cold}$ - a sub-state of D3

The table below describes the power states and transitions for a mastering NETC functions.

Table 759. Device Power State Description

Current State	Next States	Action	Power
$D0_{uninitialized}$ Uninitialized.			
$D0_{uninitialized}$	$D0_{uninitialized}$	During POR, all functions enter the $D0_{uninitialized}$ state. When a function completes a function level reset (FLR) it also enters the $D0_{uninitialized}$ state.	Main=On/Off
$D0_{uninitialized}$	$D0_{active}$	After configuration is complete, the function enters the $D0_{active}$ state, the fully operational state for a PCI Express function. A function enters the $D0_{active}$ state whenever the Bus Master Enable bit has been set.	Main=On
$D0_{active}$ Active. Normal functionality.			
$D0_{active}$	$D3_{hot}$	Prepare for sleep. **ENETC device driver**: Software must ensure that NETC's transmit is idle or quiesced. Either disable all ENETC station interfaces, disable all associated ENETC transmit rings or wait for ENETC transmit rings to drain.If connected to a switch, poll the switch port transmit busy bit, PSR[TX_BUSY] and the ENETC port transmit busy bit. Poll the port transmit busy bit, PSR[TX_BUSY], if connected to a MAC port. For ENETC functions not participating in wake-on event detection, external receive port should be disabled and software should wait for PSR[RX_BUSY]=0. ENETC functions connected directly to a switch port, should wait for switch function to quiesce first. Software creates Wake-on-LAN filters for ENETC in the Ingress Port Filter table to enable wake-on condition(s). Enabling wake-on mode by setting LPMR[WME]=1, activates the search for wake-on events. **Switch device driver**: Disable all external Ethernet receive ports and wait for PSR[RX_BUSY]=0. Receive ports connected to a remote ENETC function through a pseudo link should wait for remote ENETC function to quiesce the transmit path before disabling the port. Poll the transmit path busy bit for all ports, PSR[TX_BUSY], and wait for idle. **PCIe driver**: Saves PCIe configuration and requests the device to enter D3 power state by writing 11b to the Power State field in PCI-PM register PMCSR. Poll PCI-PM power state in PMCSR after a minimum of 10ms to determine if state transition was successful.	Main=On
$D3_{hot}$ Primary power is maintained. The system can keep NETC in this state for an arbitrary amount of time. This state can be used to achieve the following:			

Table continues on the next page...

Table 759. Device Power State Description (continued)

Current State	Next States	Action	Power
• prepare to transition the system into sleep or while the system sleeps • wake up the system if Wake-on-LAN events and PME is enabled • quiesce device before disabling LAN or removing clocks If enabled, NETC may generate PM_PME messages in this state to notify PM software that it needs to return to full active state. NETC is only capable to respond to PCIe configuration requests and will not be capable of mastering on the system bus or receiving device configurations. NETC will not forward any received frames to software; see L2 Filtering with Wake-on-LAN, for more details on how frames are filtered when operating in the Wake-on-LAN (WoL) mode.			
$D3_{hot}$	$D3_{cold}$	Transition to deep sleep. OS saves system state. Main power is off.	Main->Off
$D3_{hot}$	$D0_{uninitialized}$	Configuration software writes a value of 00b to the Power State field in the PCI PM registers while the No_Soft_Reset field in the PMCSR is clear. NETC PCI configuration space is not required to be maintained and as a result software is required to fully re-initialize the function.	Main=On
$D3_{hot}$	$D0_{active}$	Configuration software writes a value of 00b to the Power State field in the PCI PM registers while the No_Soft_Reset field in the PMCSR is set. NETC PCI configuration space is maintained.	Main=On
$D3_{hot}$	$D3_{hot}$	Wake event during sleep. Wake-on-LAN packet received by ENETC function. ENETC sets the PME status bit and sends a PM_PME message to Root Complex Event Collector (RCEC) which sets the PME status bit in the Root Status register. If software has set the PME interrupt enable bit in the Root Control register to 0b1, the Root Port then generates a level-sensitive interrupt to wake the system. **PCIe driver**: Reads configuration register Association Bitmap for Root Complex Integrated Endpoints in the event collector to determine device IDs for PME sources. Scans associated devices looking for any with a set PME status bit. For each device, it disables PME and informs the device driver that it is asserting wake up. The PCIe driver stops scanning for wake devices when it has made a complete pass through all PCIe devices in the list. Optionally clears the event collector PME interrupt enable bit in the Root Control register. Clears the event collector PME status bit in the root status register to deassert the PME interrupt signal. **Device driver**: Requests the device be put in D0 power state and sets any ENETC registers required to handle the Wake-on-LAN event. ENETC will return the frame associated with the Wake-on-LAN event, if such exist, when Wake-on-LAN mode is disabled.	Main=On
$D3_{cold}$ Primary power is not maintained. NETC does not operate on auxiliary power, hence it is not operational in this state which is equivalent to asserting HRESET.			

53.4.2.8.2 Energy Efficient Ethernet (EEE)

The IEEE 802.3 standard, Clause 78 and related, defines procedures to implement energy efficient Ethernet (EEE). It allows end stations to exchange low power idle (LPI) indications to indicate the link is not used and may be allowed to power down.

The MAC provides mechanisms to allow this signaling of low-power idle indications. It is then the responsibility of the application and management to use the information and implement power modes as necessary.

Transmitting LPI sequence to remote partner

To transmit LPI sequence to the remote partner, software needs to set PM0_COMMAND_CONFIG[TX_LOWP_ENA] to 1. MAC will start transmitting the sequence when current ongoing frame is completed. User is advised to inspect PM0_IEVENT[TX_EMPTY] before enabling LPI.

53.4.2.8.3 Wake-on-LAN Mode

Wake-on-LAN mode allows the NETC device and system to enter a low-power state and be woken when a specific event is detected on the Ethernet link. When operating in an active Wake-on-LAN mode, i.e. PCI-PM power state $D3_{hot}$, and a pre-programmed ENETC ingress port filter with a Wake-on-LAN trigger action (WOLTE), has been matched, the system can return to full operational mode when detecting such an event. In Wake-on-LAN mode, ENETC will not accept frames from upper layer software for transmission but assumes that device software has disabled transmit and made sure transmit pipeline is idle by checking transmit busy status bit. When Wake-on-LAN mode is enabled, ENETC will save the Wake-on-LAN frame to aid the system to respond more quickly and avoid dropping virtual connections. Additional frames may be saved based on the receive buffer capability and frame sizes received. It is recommended that traffic be differentiated appropriately based on Drop Resilience (DR) so that more discardable traffic is dropped first when internal memory congestion is detected.

The Wake-on-LAN events supported are:

- **Ingress Port Filter table entry match.**

  Wake-on-LAN trigger action as a result of Ingress Port Filter table entry match. An Ingress Port Filter table entry can match frames based on a mix of L2, L3, and L4 header fields including arbitrary fields (e.g. fields not identified by parsing).

  It can also match frames that have been identified by the MAC, as "magic packets". A magic packet consists of a specific pattern that can be found anywhere within the payload of an Ethernet frame. The pattern consists of 6 octets of 0xFF immediately followed by 16 times the MAC address of the station. The MAC address in the magic packet pattern, must match the MAC address configured in the PM*a*_MAC_ADDR_0/1 registers. Any data before or after this pattern is ignored. Therefore, any frame destination address, protocol-layer payload can be used to transport a magic packet pattern. magic packet detection is enabled by setting PM*a*_COMMAND_CONFIG[MG] to 1.

To enter Wake-on-LAN mode, the following steps should be considered.

1. Disable all other ports not participating in Wake-on-LAN.
   a. Disable the *Bus Master Enable* bit in the PCIe Command Register. This will disable the network controller and stop further transmissions.
   b. Disable MAC receive function.
   c. Transition the function to $D3_{hot}$ state. The reaction should be immediate for functions not supporting Wake-on-LAN. Note, all functions are required to support at a minimum D0 and D3 states.
2. Disable transmit of all ports participating in Wake-on-LAN event detection, by disabling all of their transmit BD rings, TB*a*MR[EN]=0.
3. Configure ENETC to report power management events to Event Collector by setting PCI_CFC_PCIPM_CTL_STAT[PME_EN]=1.
4. Configure the Wake-on-LAN trigger events in the ingress port filter table, including magic packet. Magic packet detection is enabled by setting PM*a*_COMMAND_CONFIG[MG] to 1.
5. Enable Wake-on-LAN for the port by setting LPMR[WME]=1. Setting of this bit (0b1) arms the search for a new event in the ingress port filter table.

a. Forces quiescing of ENETC

b. ENETC will drop received frames not matching the Wake-on-LAN frame. If a Wake-on-LAN frame is encountered before PCI-PM $D3_{hot}$ state is entered, the PWOSR[WOLA] signal is cleared (0b0) and subsequent frames are processed normally.

6. Software requests ENETC to enter PCI-PM $D3_{hot}$ state by writing 0b11 to the Power State field in PCI-PM register PCI_CFC_PCIPM_CTL_STAT.

   a. PCI-PM $D3_{hot}$ state will not be entered if PWOSR[WOLA]=0. Wake-on-LAN mode LPMR[WME] must be reenabled to again arm the ingress port filter function.

7. Hardware captures the request for PCI-PM power state change and waits for ENETC to quiesce before entering $D3_{hot}$ state.

   a. If a Wake-on-LAN event is detected, as indicated by PWOSR[WOLA]=0, the request for low-power mode is aborted.

8. When $D3_{hot}$ state is entered the Wake-on-LAN function continues the search for received frames matching any one of the Ingress Port Filter table entry with a Wake-on-LAN trigger action (WOLTE) . The received frames are dropped until the Wake-on-LAN event is detected after which frames (including the Wake-on-LAN frame event) are stored locally in ENETC.

   a. ICM stops forwarding frames to the host transfer agent.

9. Software waits a minimum of 10ms before polling the Power State field in PCI-PM register PCI_CFC_PCIPM_CTL_STAT. If the Power State is 0b11, then the transition was successful and the system may enter a low-power state where clocks are disabled or power removed, except for NETC. ENETC will not forward any traffic to BD rings in memory until software returns ENETC to PCIPM power state $D0_{active}$. If the Power State remains 0b00, one of the following has occurred:

   a. A quiesce state was never reached; software may poll the *Transaction Pending* bit in the PCIe Device Status Register.

   b. Wake-on-LAN was not enabled or an event not selected.

   c. Wake-on-LAN event was detected, as seen by PWOSR[WOLA]=0, before low-power mode was reached.

10. When a Wake-on-LAN event has occurred in low-power mode, ENETC sets the PME status bit in register PCI_CFC_PCIPM_CTL_STAT and sends a PME message to Root Complex Event Collector (RCEC) which sets the PME status bit in the Root Status register PCI_CFC_PCIE_ROOT_STAT. If software has set the PME interrupt enable bit in the Root Control register PCI_CFC_PCIE_ROOT_CTL to 0b1, the Event Collector then generates a level-sensitive interrupt to wake the system.

    a. A Wake-on-LAN frame may have its drop resilience modified, by the ingress port filter, to guarantee system delivery when internal memory congestion is detected.

11. Software wakes up from low-power mode and services the PME interrupt.

12. Software brings ENETC into PCI-PM D0 state by writing 0b00 to the Power State field in PCI-PM register PCI_CFC_PCIPM_CTL_STAT.

    a. At this time software may be required to re-initialize receive BD rings.

13. ENETC (ICM) will start forwarding frames to BD rings in main memory when LPMR[WME] is cleared (0b0). See L2 Filtering with Wake-on-LAN, for more details.

53.4.2.9 Clocking

This module has no clocking considerations.

53.4.2.10 Reset

NETC supports multiple types of reset, listed below in order of highest scope to smallest:

- Hard reset, also known as Power-on-Reset (POR)
- Soft reset
- Warm reset
- Function Level Reset (FLR)

Hard Reset

Hard reset restores all NETC state bits to a Power-on-State. This type of reset is issued at SoC level.

Soft Reset

Soft reset targets all functions as well as any shared or common infrastructure blocks, making this equivalent to a device reset. The use of soft reset would be in a scenario where an unrecoverable event has occurred which required a reset of NETC and a full chip level reset is not desirable.

Software should follow these steps prior to initiating a soft reset to avoid hanging any outstanding transactions in the system.

1. Software disables MAC receive function(s).
2. Software sets the soft reset bit NETCRR[SR]=1.
   - NETC stops further prefetching of BDs
   - NETC stops scheduling transmit frames
   - NETC completes any in-flight transmit frames processing.
     - If there is use of time gating or credit based shaping, worst case wait time is SaTGSLR/EaTGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET] + Transmit(MAX_SDU) + Writeback BD.
     - If there is no use of time gating or credit based shaping, worst case wait time is Transmit(MAX_SDU) + Writeback BD.
3. Software waits for either:
   - 100 ms (as per PCIe specification).
   - Shortest possible reset time as defined by IERB register NETCFLRCR.
   - NETCRR[SR] bit to clear by polling.
4. Software reconfigures NETC.

Warm Reset

Warm reset is similar to a soft reset but does not reset any of the IERB register values. The following steps generate a warm reset:

1. Software disables MAC receive function(s).
2. Software unlocks the IERB registers, currently in a locked state (i.e. NETCRR[LOCK]=1 and NETCSR[STATE]=0), by clearing (0b0) the NETCRR[LOCK] bit.
   - NETC stops further prefetching of BDs
   - NETC stops scheduling transmit frames
3. Software waits for NETC to complete any in-flight transmit frames
   - If there is use of time based scheduling, worst case wait time is SaTGSLR/EaTGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET] + Transmit(MAX_SDU) + Writeback BD.
   - If there is no use of time gating or credit based shaping, worst case wait time is Transmit(MAX_SDU) + Writeback BD.
4. Software either:
   - waits 100 ms (as per PCIe specification).
   - waits shortest possible reset time as defined by IERB register NETCFLRCR.
   - waits for NETCRR[LOCK] bit to clear by polling.

5. Software reconfigures NETC.

Function Level Reset

Function level reset is generated through the PCIe Function Level Reset (FLR) bit, or indirectly through warm or soft reset. There are two types of FLR reset; reset which occurs through the PCIe physical function (PF) and virtual interface reset, which is a reset of the PCIe virtual function (VF). FLR that targets a virtual function (VF), resets a single VF. Reset that targets a physical function (PF), resets a PF and all its associated VFs. An FLR that targets a VF does not affect the VFs existence or any address mapping assigned to it (VFs BAR*n* values and MSE). VF reset is a destructive process and must be avoided while the function is in use. It is expected it will be used as PCI functions change owners (and all existing configuration and state must be reverted to power-on defaults).

The Timer is a support function for other NETC critical functions such as time gating and Ethernet MAC timestamping. Due to these dependencies, the Timer cannot be reset by FLR alone, and requires a soft reset. Errors occurring for the Timer are reported as global.

Software should follow these steps when initiating a Function Level Reset:

1. Software initiates the FLR.
2. Software either:
   - waits 100 ms (as per PCIe specification).
   - waits PCIE_CFC_RTR_RTR2[FLR_TIME], if PCIE_CFC_RTR_RTR1[VALID]=1 in PCIe function.
   - polls PCI_CFC_PCIE_DEV_CTL[INIT_FLR] and waits for b0. If bit has not cleared within specified waiting time, there may be a potential HW failure and software is recommended to issue a soft reset for the device.
3. Software reconfigures the Function and enables it for normal operation.

NOTE

When a reset is triggered it will instruct the function to quiesce (drain pending frames and pipelines) for an extended period of time, followed by a reset of registers and finally clearing of table entries. The length of this entire sequence is controlled by IERB register NETCFLRCR.

53.4.2.10.1 Function Reset

Function reset applies to switch and ENETC instances which have physical or pseudo MAC port-links attached.

53.4.2.10.1.1 ENETC Function Reset

Reset of the ENETC function is equivalent to a PCIe Function Level Reset (FLR) of the Physical Function (PF), which includes the port and any station interface bound to the ENETC instance. The function reset is generated by writing to the PCI header Device Control Register.

Hardware will set the following register bits while trying to quiesce:

- PM*a*_COMMAND_CONFIG[RX_EN]=0b and PM*a*_COMMAND_CONFIG[TX_FLUSH]=1b if physical port present
- POR[RXDIS/TXDIS]=1b

The PCIe function number mapping for the ENETC instance can be found in IERB register E*a*BCR0[FN]. An FLR that targets a PCIe PF resets the PF's SR-IOV extended capabilities, the VFs (VSIs) are no longer allocated or enabled. The effects of the port reset are as follows:

- MSI configuration (stops any pending MSIs)
- In-flight frames in the Ethernet Rx I/F block are discarded.
- In the case of the Ethernet MAC, reception of the current frame from the Ethernet link will continue till the end of the frame, and then the frame will be discarded. Afterwards, no more frames from the Ethernet MAC are expected.
- In the case of the pseudo MAC, the other end of the pseudo link (switch) may continue sending frames, these frames received from the pseudo MAC will be discarded. Additionally, switch management frames with Host Reason set to non-zero which by-pass the pseudo MAC, will also be discarded.

- In-flight frames in the Ingress Port Processing block and the Host Transfer Agent Receive block will be drained and discarded.
- Both ICM queues are drained as fast as possible. Frames dequeued/drained from the ICM queues are immediately discarded.
- In-flight frames in the Host Transfer Agent Transmit block, Egress Port Processing block and Ethernet Tx I/F block are drained and discarded, with the error "frame dropped due to port reset" reported in their transmit buffer descriptor. Transmission of frames is stopped.
- Transmit BD rings are no longer serviced.
- Processing of in-flight table management commands will be completed. However no new table management commands will be accepted.
- Once there are no more frames in the ENETC instance and processing of in-flight table management commands is completed, all of the ENETC instance registers and tables are reset.
- Reset mailbox between VSI and associated PSI
- Flush port-link bound Ethernet MAC(s) (through the same mechanism as PM*a*_COMMAND_CONFIG[TX_FLUSH]) and reset of config and control registers. Note: FLR does not clear MAC statistics registers.

Since management registers for buffer descriptor rings will be reset, it could cause loss of unprocessed packets. To avoid loss of data, software should perform the following steps prior to reset:

1. Disable the network controller by setting bit SIMR[EN]=0, disable the receive link and wait for controller to be idle, PSR[TX_BUSY/RX_BUSY]=0
2. Save descriptor ring information (base address, size, consumer/producer indices)

NOTE

The IERB registers are unaffected by FLR. Setting the PCI_CFH_CMD[MEM_ACCESS]=1 for the function will copy register values from IERB after the FLR.

53.4.2.10.1.2 Switch Function Reset

Reset of the switch function is equivalent to a PCIe Function Level Reset (FLR), which includes all port-links bound to the switch instance (pseudo links as well). The function reset is generated by writing to the PCI header Device Control Register. The PCIe function number mapping for the switch instance can be found in IERB register S*a*BCR[FN].

Hardware will set the following register bits while trying to quiesce:

- PM*a*_COMMAND_CONFIG[RX_EN]=0b and PM*a*_COMMAND_CONFIG[TX_FLUSH]=1b if physical port present
- POR[RXDIS/TXDIS]=1b

The effects of the switch function reset is as follows:

- MSI configuration (stops any pending MSIs)
- In-flight frames in the Ethernet Rx I/F block are discarded.
- In the case of the Ethernet MAC, reception of the current frame from the Ethernet link will continue till the end of the frame, and then the frame will be discarded. Afterwards, no more frames from the Ethernet MAC are expected.
- In the case of the pseudo MAC, the other end of the pseudo link (ENETC) may continue sending frames, these frames received from the pseudo MAC will be discarded.
- In-flight frames in the Ingress Packet Processing block and the Replication and Egress Packet Processing block will be drained and discarded, with the exception of switch management frames with Host Reason set to non-zero, in which case, they will be delivered to ENETC (enqueued directly inside ENETC (ICM)); these frames are not discarded.
- All ETM class queues are drained as fast as possible. Frames dequeued/drained from the ETM class queues are immediately discarded.

- In-flight frames in the Ethernet Tx I/F datapath processing pipeline are discarded. Transmission of frames is stopped.
- Processing of in-flight table management commands will be completed. However no new table management commands will be accepted.
- Once there are no more frames in the switch and processing of in-flight table management commands is completed, all of the switch registers and tables are reset.
- Flush port-link bound Ethernet MAC(s) (through the same mechanism as PM*a*_COMMAND_CONFIG[TX_FLUSH]) and reset of config and control registers. Note: FLR does not clear MAC statistics registers.

NOTE

Note that the IERB registers are unaffected by FLR. Setting the PCI_CFH_CMD[MEM_ACCESS]=1 for the function will copy register values from IERB after an FLR.

53.4.2.10.2 VSI Function Reset

This is a reset of the Virtual Station Interfaces (VSIs). It is equivalent to PCIe Function Level Reset (FLR) of a VF, generated by writing to the PCI header Device Control register. The effects of the VF reset are as follows:

- MSI configuration (stops any pending MSIs)
- In-flight frames in the Host Transfer Agent Receive block for the VSI being reset will be drained and discarded.
- Transmit BD ring(s) for the VSI being reset, are no longer serviced.
- In-flight frames in the transmit datapath pipeline for the VSI being reset, will be drained and transmitted out the link. No frame discards are expected unless an error is detected.
- Processing of in-flight table management commands will be completed. However no new table management commands will be accepted for the VSI being reset.
- Once there are no more frames and processing of any in-flight table management commands is completed, for the VSI being reset, all of the VSI registers/fields and tables/entries are reset. The PSI's VSI registers (e.g. PSIaPMAR0/1) are also reset . These registers will need to be re-initialized after the VSI FLR operation completes, if these registers were updated prior to the completion of the VSI FLR.
- SIIDR[FLR*n*] is set to 1 in the PSI to indicate which interface is being reset.

Since management registers for buffer descriptor rings will be reset, it could cause loss of unprocessed packets. To avoid loss of data, software should perform the following steps prior to resetting the interface:

1. Disable the interface by setting bit SIMR[EN]=0 and wait for interface to be idle, SISR[TX_BUSY]=0
2. Save descriptor ring information (base address, size, consumer/producer indices)

53.4.2.10.3 EMDIO Controller Reset

Reset of the EMDIO controller can be achieved by issuing a Function Level Reset (FLR), generated by writing the PCI header Device Control register. Software should follow these steps:

1. Stop issuing MDIO transactions to external PHY.
2. Wait for any outstanding MDIO transactions to complete by reading the busy bit.
3. Issue the FLR.
4. Wait for the appropriate amount of time, as specified in the PCIe specification, before reinitializing the EMDIO controller.

53.4.2.10.4 Timer Reset

Function level reset is not applicable to timer as a supporting function and requires a soft reset.

53.4.2.11 Interrupts

53.4.2.11.1 Message Signaled Interrupt (MSI-X)

NETC supports generating MSI-X messages when an interrupt event occurs. These are functional interrupts that occur during the normal operation of NETC and usually signifies that there is data to process by the receiving entity, but it is also used for timer events.

An interrupt is generated whenever an interrupt event is detected by a NETC function and the associated interrupt enable bit is also set. An interrupt has a corresponding interrupt vector register which defines the index used during lookup of the address and data in the MSI-X Table. For interrupt grouping, two or more vector registers can share the same vector table index value. The address and data from the table is then used to generate a (32-bit) DWORD size posted memory write, which normally is routed towards the system interrupt controller.

System software initializes the message address and message data (from here on referred to as the “vector”) during device configuration, allocating a vector table to each MSI-X capable function. MSI-X supports the ability for software to control aliasing when fewer vectors are allocated than requested, plus the ability for each vector to use an independent address and data value, specified by a table that resides in Memory Space. Note that there is a mapping between NETC function and PCIe function defined by the binding configuration registers in the IERB space.

For an MSI-X to occur, the MSI-X generation must be enabled by setting PCI_CFC_MSIX_MSG_CTL[MSIX_EN]=1. MSI-X supports per-function masking and per-vector masking. Per-function masking, when set, masks all of the vectors associated with a function. See PCI_CFC_MSIX_MSG_CTL[FUNC_MASK], for more details. Per-vector masking is managed through a MSI-X Table entry. An MSI-X vector is masked when its associated MSI-X Table entry Mask bit or the MSI-X Function Mask bit is set. While a vector is masked, the function is prohibited from sending the associated message, and the function must set the associated Pending bit whenever the function would otherwise send the message. When software unmasks a vector whose associated Pending bit is set, the function must schedule sending the associated message, and clear the Pending bit as soon as the message has been sent.

The PCI MSI-X capability structure points to an MSI-X Table structure and a MSI-X Pending Bit Array (PBA) structure, each residing in Memory Space. Each structure is address mapped by an entry in the PCI Enhanced Allocation (EA) capability structure. The MSI-X BAR Indicator register field (BIR) indicates which EA entry used for the base address. A QWORD aligned MSI-X Offset indicates where the structure begins relative to the EA base address.

The MSI-X Table structure is illustrated in Table 760, typically contains multiple entries, each consisting of several fields: Message Address, Message Upper Address, Message Data, and Vector Control. Each entry is capable of specifying a unique vector. The Pending Bit Array (PBA) structure, illustrated in Table 765, contains the function’s Pending Bits, one per Table entry, organized as a packed array of bits within (64-bit) QWORDs. The last QWORD will not necessarily be fully populated.

NOTE

This implementation supports a reduced number of address bits. Total address bits available in each table entry for reading/writing are 32 bits.

Table 760. MSI-X Table Structure

DWORD 3	DWORD 2	DWORD 1	DWORD 0		
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 0	Base
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 1	Base + 1*16B
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 2	Base + 2*16B
...	...	...	...	...	
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry (N-1)	Base + (N-1)*16B

Table 761. Message Address for MSI-X Table Entries

Bits	Name	Description
31-2	Message Address	System-specified message lower address. For MSI-X messages, the contents of this field from an MSI-X Table entry specifies the lower portion of the (32-bit) DWORD-aligned address (ADDR[31:2]) for the memory write transaction. This field is read/write.
1-0	Message Address	For proper DWORD alignment, the state of these bits after reset must be 0. These bits are read only.

Table 762. Message Upper Address for MSI-X Table Entries

Bits	Name	Description
31-0	Message Upper Address	System-specified message upper address. For MSI-X messages, the contents of this field from an MSI-X Table entry specifies the upper portion of the (32-bit) DWORD-aligned address (ADDR[63:32]) for the memory write transaction. This field is read/write. **NOTE** This implementation supports a reduced number of address bits. Total address bits available in each table entry for reading/writing are 32 bits.

Table 763. Message Data for MSI-X Table Entries

Bits	Name	Description
31-0	Message Data	System-specified message data. For MSI-X messages, the contents of this field from an MSI-X Table entry specifies the data driven on DATA[31:0] during the memory write transaction's data phase. This field is read/write.

Table 764. Vector Control for MSI-X Table Entries

Bits	Name	Description
31-1	Reserved	After reset, the state of these bits must be 0. However, for potential future use, software must preserve the value of these reserved bits when modifying the value of other Vector Control bits. If software modifies the value of these reserved bits, the result is undefined.
0	Mask Bit	When this bit is set, the function is prohibited from sending a message using this MSI-X Table entry. However, any other MSI-X Table entries programmed with the same vector will still be capable of sending an equivalent message unless they are also masked. This bit's state after reset is 1 (entry is masked). This bit is read/write.

Table 765. MSI-X PBA Structure

Pending Bits 0 through 63	QWORD 0	Base
Pending Bits 64 through 127	QWORD 1	Base + 1*8B

Table continues on the next page...

Table 765. MSI-X PBA Structure (continued)

...	...	...
Pending Bits ((N-1) div 64)*64 through N-1	QWORD ((N-1) div 64)	Base + ((N-1) div 64)*8B

To request service using a given MSI-X Table entry, a function performs a DWORD memory write transaction using the contents of the Message Data field entry for data, the contents of the Message Upper Address field for the upper 32 bits of address, and the contents of the Message Address field entry for the lower 32 bits of address. A memory read transaction from the address targeted by the MSI-X message produces undefined results.

Table 766. Interrupt Classes

Description	Interrupt Handling	
Virtual Station Interface message transmitted to PSI.	**Device Detect**	VSIIDR[MS]
	Device Enable	VSIIER[MSIE]
	MSI-X Table Vector	SIMSIVR[VECTOR]
	PCIe Function Binding	Determined by E*a*VBCR and ENETC SR-IOV capability
	Comments VSI has completed message transfer to the PSI through a memory partition copy. VSIMSGSR[MS/MC] may provide additional information about the success of the transfer.	
Virtual Station Interface message received from PSI.	**Device Detect**	VSIIDR[MR]
	Device Enable	VSIIER[MRIE]
	MSI-X Table Vector	SIMSIVR[VECTOR]
	PCIe Function Binding	Determined by E*a*VBCR[NUM_VSI] and ENETC SR-IOV capability
	Comments VSI has received a 16-bit message from the PSI.	
Physical Station Interface received message from VSI *a*, where *a* is the VSI number	**Device Detect**	PSIIDR[MR*a*]
	Device Enable	PSIIER[MR*a*EN]
	MSI-X Table Vector	SIMSIVR[VECTOR]
	PCIe Function Binding	E*a*BCR0[FN]
	Comments PSI has received a message from one of its VSIs through a memory partition copy.	
Physical Station Interface detected FLR by VSI *a*, where a is the VSI number	**Device Detect**	PSIIDR[FLR*a*]
	Device Enable	PSIIER[FLR*a*EN]
	MSI-X Table Vector	SIMSIVR[VECTOR]
	PCIe Function Binding	E*a*BCR0[FN]
	Comments	

Table continues on the next page...

Table 766. Interrupt Classes (continued)

Description	Interrupt Handling	
	PSI has detected a PCI FLR initiated by one of its VSIs.	
Station Interface command BD ring frame completion	**Device Detect**	SICBDRIDR[CBDC]
	Device Enable/Mask	CBD[FI], SICBDRIER[CBDCIE]
	MSI-X Table Vector	SICMSIVR[VECTOR]
	PCIe Function Binding	E*a*BCR0[FN]
	Comments Completion of Station Interface command BD with CBD[FI] set.	
Station Interface transmit ring *a* threshold interrupt coalescing, where *a* is the transmit ring number	**Device Detect**	SITXIDR[TXT*a*], TB*a*IDR[TXT]
	Device Enable	TB*a*IER[TXTIE]
	MSI-X Table Vector	SIMSITR*a*VR[VECTOR]
	PCIe Function Binding	PSI: E*a*BCR0[FN], VSI: determined by E*a*VBCR[NUM_VSI] and ENETC SR-IOV capability
	Comments Station interface transmit ring has transmitted at least the number of packets specified by TB*a*ICR0[ICPT] or the threshold timer has expired since last transmitted packet as specified by TB*a*ICR1[ICTT].	
Station Interface transmit ring *a* frame completion, where *a* is the transmit ring number	**Device Detect**	SITXIDR[TXF*a*], TB*a*IDR[TXF]
	Device Enable	BD[FI], TB*a*IER[TXFIE]
	MSI-X Table Vector	SIMSITR*a*VR[VECTOR]
	PCIe Function Binding	PSI: E*a*BCR0[FN], VSI: determined by E*a*VBCR[NUM_VSI] and ENETC SR-IOV capability
	Comments Transmit of frame detected with buffer descriptor BD[FI]=1.	
Station Interface receive ring *a* threshold interrupt coalescing, where *a* is the receive ring number	**Device Detect**	SIRXIDR[RXT*a*], RB*a*IDR[RXT]
	Device Enable	RB*a*IER[RXTIE]
	MSI-X Table Vector	SIMSIRR*a*VR[VECTOR]
	PCIe Function Binding	PSI: E*a*BCR0[FN], VSI: determined by E*a*VBCR[NUM_VSI] and ENETC SR-IOV capability
	Comments Station interface receive ring holds at least the number of packets specified by RB*a*ICR0[ICPT] or the threshold timer has expired since last received packet as specified by RB*a*ICR1[ICTT].	
Ethernet port internal MDIO access completion	**Device Detect**	IMDIOIRR[PORT*n*], PM0_MDIO_CFG[CMP]
	Device Enable/Mask	PM0_MDIO_CFG[CIM]
	MSI-X Table Vector	IMDIOMSIVR[VECTOR]

Table continues on the next page...

Table 766. Interrupt Classes (continued)

Description	Interrupt Handling	
	PCIe Function Binding	S*a*BCR[FN], E*a*BCR0[FN]
	Comments Switch or ENETC port has received an internal MDIO request, which has completed, from one of the following: • RevMII MDIO Register IMDIOIRR is a summary of all pending MDIO port interrupts.	
Ethernet port external MDIO access completion	**Device Detect**	EMDIOIRR[PORT*n*], PEMDIOCR[CMP]
	Device Enable/Mask	PEMDIOCR[CIM]
	MSI-X Table Vector	EMDIOMSIVR[VECTOR]
	PCIe Function Binding	S*a*BCR[FN], E*a*BCR0[FN]
	Comments Switch or ENETC has issued an external MDIO request (to external off-chip PHY), by means of the EMDIO controller, which has completed. Register EMDIOIRR is a summary of all pending external MDIO port interrupts.	
Ethernet MAC magic packet detected	**Device Detect**	PM0_IEVENT[MGI]
	Device Enable/Mask	PM0_IMASK[MGI]
	MSI-X Table Vector	Entry 0
	PCIe Function Binding	S*a*BCR[FN], E*a*BCR[FN]
	Comments Switch or ENETC Ethernet MAC detected magic packet.	
External MDIO controller access completion	**Device Detect**	EMDIO_CFG[CMP]
	Device Enable	EMDIO_CFG[CIM]
	MSI-X Table Vector	Entry 0
	PCIe Function Binding	EMDIOBCR[FN]
	Comments Centralized external MDIO controller access (to external off-chip PHY) has completed.	
IEEE 1588 timer event detected	**Device Detect**	TMR_TEVENT[ETS2_OV] TMR_TEVENT[ETS1_OV] TMR_TEVENT[ETS2] TMR_TEVENT[ETS1] TMR_TEVENT[ETS2_THR] TMR_TEVENT[ETS1_THR] TMR_TEVENT[ALM2]

Table continues on the next page...

Table 766. Interrupt Classes (continued)

Description	Interrupt Handling	
		TMR_TEVENT[ALM1] TMR_TEVENT[PP3] TMR_TEVENT[PP2] TMR_TEVENT[PP1]
	Device Enable/Mask	TMR_TEMASK[ETS2_OV] TMR_TEMASK[ETS1_OV] TMR_TEMASK[ETS2] TMR_TEMASK[ETS1] TMR_TEMASK[ETS2_THR] TMR_TEMASK[ETS1_THR] TMR_TEMASK[ALM2] TMR_TEMASK[ALM1] TMR_TEMASK[PP3] TMR_TEMASK[PP2] TMR_TEMASK[PP1]
	MSI-X Table Vector	0
	PCIe Function Binding	T*a*BCR[FN]
	Comments The timer interrupt includes multiple events including (1) external trigger timestamp new/FIFO overflow/ FIFO threshold, (2) alarm and (3) periodic pulse.	
Switch command BD ring frame completion	**Device Detect**	CBDR*a*IDR[CBDC]
	Device Enable	CBDR*a*IER[CBDCIE]
	MSI-X Table Vector	CBDR*a*MSIVR[VECTOR]
	PCIe Function Binding	S*a*BCR[FN]
	Comments Completion of switch command BD with CBD[FI] set.	
Switch time capture	**Device Detect**	TCRPIDR[TIMEOUT], TCRPIDR[TRANSMIT]
	Device Enable	TCCR[ARM], TCIER[TIMEOUT], TCIER[TRANSMIT]
	MSI-X Table Vector	TCMSIVR[VECTOR]
	PCIe Function Binding	S*a*BCR[FN]
	Comments	

Table continues on the next page...

Table 766. Interrupt Classes (continued)

Description	Interrupt Handling
	A receive or transmit timestamp has been captured by the switch or a timeout has occurred. Port for receive timestamp is indicated in the register TCRPSR[RX_PORT]. Port of transmit timestamp is determined by TCIDR[TX_PORTS]. Latency values are captured in P*a*TCMINLR/P*a*TCMAXLR.

NOTE

Wake up interrupt is a power management event and is handled by the Root Complex Event Collector.

The internal/external MDIO controllers are considered embedded functions within NETC. This implies that legacy interrupts are converted to MSI-X events. Legacy interrupts are edge detected to set the port/global interrupt detect register.

53.4.2.11.2 Wired Interrupt

NETC requires the use of a single wired signal for error interrupts which is provided by the iEP-RC (INTA) by means of the Event Controller. All error interrupt events are reported to the Event Controller by internal messaging from NETC operating as an integrated endpoint. The Root Complex Event Collector will assert the PCI INTA pin when properly configured, see Error Reporting.

53.4.2.12 Weighted Bandwidth Fair Scheduling (WBFS) Algorithm

Weighted Bandwidth Fair Scheduling (WBFS) is a method of scheduling packets from queues to a link such that each of the queues get their fair share bandwidth that is proportionate to a weight value assigned to a queue. This algorithm also applies to non-leaf nodes in a hierarchical scheduler structure. The term "class" is used to refer to the input to the WBFS scheduler which can be the queue itself (e.g. ETM class queue) or an aggregation of queues (e.g. channel).

WBFS is used to schedule packets from classes within a group such that each class gets a "fair amount of bandwidth made available to that group. Bandwidth (transmit opportunities) that is made available to a group is fair shared among the classes of that group in proportion to a weight value configured for that class. For example, if class "X" has a weight of 5 and class "Y" has a weight of 1, class "X" has a fair share that is 5 times the fair share of class "Y". The weight values for each class can range from 1 to 248 (in 255 pseudo logarithmic steps), which can be configured differently for each group.

The premises for fairness, follow a max-min weighted fairness principle:

1. available bandwidth is divided and offered to their classes in proportion to their weight
2. offered bandwidth in excess of a class's demand is to be re-offered to classes with unsatisfied demands proportion to their weight

The algorithm classifies classes as underweight classes and overweight classes. Underweight classes are presumed to be classes where their bandwidth demands are satisfied. Overweight classes are presumed to be classes where their bandwidth demands are unsatisfied.

The algorithm:

- Maintains three lists of indices of classes; non-active underweight list, active underweight list and overweight list. The non-active/active underweight lists contain indices of classes that have been determined to be underweight and the overweight list contains indices of classes that have been determined to be overweight.
- The active underweight list is a FIFO ordered list of classes that became non-empty since the last scheduling opportunity. This list is sorted in the order in which the class became non-empty, from the first class becoming active to the last class becoming active.
- The non-active underweight list contains classes with no packets waiting to be transmitted.
- For each overweight class, a debt/deficit value in bytes is maintained. The overweight list is sorted based on the debt value, from the lowest debt value to the highest debt value.

- Confirms all classes as underweight upon initialization or re-initialization; i.e. all classes are put in the non-active underweight list.
- Once a class in the non-active underweight list becomes non-empty, it is moved to the end of the active underweight list.
- Select packets from the classes in the active underweight list before any packets from classes in the overweight list. Within a list (active underweight or overweight), the first element (or class) is selected.
- After selecting a packet from a class in the active underweight list, the class becomes overweight and as a result is moved to the overweight list with a debt value of P where P is equal to the number bytes in the packet transmitted divided by the weight assigned to the class.
- After selecting a packet from a class in the overweight list, the class remains overweight with a debt value of P where P is equal to the number bytes in the packet transmitted divided by the weight assigned to the class. To be fair, for all other classes in the overweight list, P is credited to the debt level of each of these classes down to 0 (debt value doesn't go negative). Once all classes have been updated with their new debt value, the overweight list is sorted based on the debt value, from the lowest debt value to the highest debt value.
- If the selected class from the overweight list is empty, the class is skipped and the next element (or class) in the overweight list is selected.
- A class in the overweight list where its debt becomes completely erased, and is without packets waiting to be transmitted, is moved to the non-active underweight list.
- If all classes are without packets at any point in time the algorithm is re-initialized such that all classes are considered underweight.

53.4.2.13 Hash Table Algorithm

The exact match hash table uses a novel approach when distributing entries to attempt to reduce the number of hash collisions. Any entry inserted into the hash table has 6 possible locations that it could be placed in. The algorithm involved in this attempts to minimize chain length overall, without impacting query time by providing many more possible starting locations for collisions.

When adding a new entry, the lookup key is hashed with a series of bit shifts, swaps and XORs. This hash result will map the lookup key to a specific hash root location in a 3D row x column x plane space. The row directly corresponds to a specific address in the exact match table space. The plane specifies independently addressable parallel portions of external memory[1]. Each plane holds 3 locations for a row. The column then identifies 1 of the 3 locations in that scope. Each location holds a 32b structure that contains the billeting [2] state, the chain length and the root pointer to the exact match table entry. If the location is empty (chain length 0), the new entry is added here. If the location is not empty (chain length >0), the alternate column locations are tested to find which column has the smallest chain length[3] which has not already hosted a billet[2]. If testing an alternate column location results in a lower chain length, it is chosen in place of the predetermined location and the entry is prepended to the chain stored there. The alternate location's billeting state is updated accordingly to capture the hash result indicating where the billet originated. If no lower chain length choice is found, the entry is appended at the end of the chain stored at the predetermined location. If there are no locations available because the chain length has reached the configured maximum (HBTCR[MAX_COL]) then the new entry will not be added.

When searching for an entry, the lookup key is hashed with the same series of bit shifts, swaps and XORs. Again, the hash result maps to a specific row x column x plane hash root location. It is not known if the entry had been placed in an alternate location, so at this time, the 5 alternate columns on the row[3] are first tested for the presence of a matching billeting state value. If found in any of the alternates, the entry will exist as the first entry in the chain. If not found in the alternate columns, the chain at the predetermined location (based on row,column,plane) is traversed to find a matching entry (or confirmation that it is not found).

This algorithm provides a much improved collision distribution compared to a simple single entry collision chain, resulting an a much higher hash space utilization, without sacrificing a maximum bounds to query time (when considering the maximum collision chain value HBTCR[MAX_COL]).

1. The memory holding the hash table entries is physically stored in 2 parallel memory instances, and this allows for even better entry distribution explained below.
2. Billeting here refers to the process by which an entry is placed in a location that is not it's native location.

3. Two alternate column locations exist on the same plane at the same row and 3 alternate column locations exist on the alternate plane at a transformed row. The transformation is based on the hash result, which further 'mixes' the possible locations for any added entry.

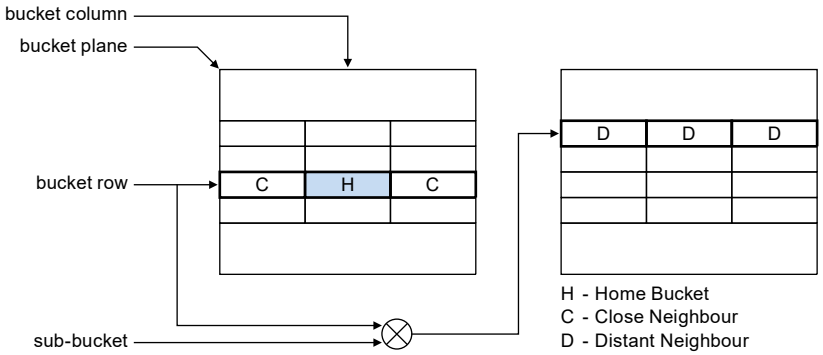


Figure 364. Hash root memory organization

53.4.3 External Signal Description

Table 767. External Signals

Signal Name	Description	I/O
Ethernet		
The MII, RMII and RGMII interfaces are supported on Ethernet (link) 0-4. The GMII interface is supported on Ethernet (link) 3 and 4. Each Ethernet (link) can be configured to connect either to a parallel interface (MII, RMII and RGMII) or a serial (SerDes) interface, using a configuration register on the device. The NETC Ethernet (link) signal mapping to the external parallel interface, is given in Table 768. The NETC Ethernet (link) signal mapping to the on chip SerDes module, is given in Table 769.		
MII_TXCLK	MII transmit clock supplied by the PHY.	I
MII_TXEN	MII Transmit Enable.	O
MII_TXD[3:0]	MII transmit data bus.	O
MII_TXER	Indicates to the PHY that the current transmission data is corrupted, or the transmit is in a low power state.	O
MII_RXCLK	MII receive clock supplied by the PHY.	I
MII_RXDV	Indicates the receive data bus is valid. Asserted beginning with the first preamble byte and lasting until the last CRC byte.	I
MII_RXD[3:0]	MII receive data bus.	I
MII_RXER	Error indication from the PHY, that the current frame contains erroneous data, or the line is in low power state.	I
MII_CRS	Carrier sense. Used for half-duplex mode only.	I
MII_COL	Collision detect. Used for half-duplex mode only.	I
GMII_TXCLK_ENA	Transmit clock enable	I
GMII_TXEN	GMII Transmit Enable.	O
GMII_TXD[7:0]	GMII transmit data bus.	O

Table continues on the next page...

Table 767. External Signals (continued)

Signal Name	Description	I/O
GMII_TXER	Indicates to the PHY that the current transmission data is corrupted, or the transmit is in a low power state.	O
GMII_RXCLK_ENA	Receive clock enable	I
GMII_RXDV	Indicates the receive data bus is valid. Asserted beginning with the first preamble byte and lasting until the last CRC byte.	I
GMII_RXD[7:0]	GMII receive data bus.	I
GMII_RXER	Error indication from the PHY, that the current frame contains erroneous data, or the line is in low power state.	I
RMII_REF_CLK	RMII 50MHz reference clock input	I
RMII_TXEN	RMII Transmit Enable.	O
RMII_TXD[1:0]	RMII transmit data bus.	O
RMII_RXD[1:0]	RMII receive data bus.	I
RMII_RXER	Error indication from the PHY, that the current frame contains erroneous data, or the line is in low power state.	I
RMII_CRS_DV	Carrier sense and receive data valid.	I
RGMII_TXD[3:0]	RGMII transmit data bus [3:0] on rising edge of RGMII_TXC, byte data [7:4] on falling edge.	O
RGMII_TXCTL	RGMII transmit data enable on rising edge of RGMII_TXC, transmit error on falling edge.	O
RGMII_TXC	RGMII transmit clock (output)	O
RGMII_RXD[3:0]	RGMII receive data bus [3:0] on rising edge of RGMII_RXC, byte data [7:4] on falling edge.	I
RGMII_RXCTL	RGMII receive data valid on rising edge of RGMII_RXC, receive error on falling edge.	I
RGMII_RXC	RGMII receive clock (input)	I
ETHn_MDIO		
The Ethernet slave MDIO interface is used for an external MAC to access and configure the PHY features of the external port, when the port is operating in the RevMII mode (IF_MODE[REVMII]=1). There is one slave MDIO interface per external port that supports RevMII. The RevMII mode is supported on Ethernet (link) 2, 3 and 4.		
ETHn_MDC	Input clock, where n represents Ethernet (link) 2, 3 and 4.	I
ETHn_MDIO	Bidirectional data, clocked by ETHn_MDC, where n represents Ethernet (link) 2, 3 and 4.	I/O
EMDIO		
The EMDIO interface is used for access to and configuration of external Ethernet PHYs. Use of this interface is via the global EMDIO function (PF1) or by virtual EMDIO per external port.		
NETC_EMDIOC	Output clock.	O
NETC_EMDIO	Bidirectional data, clocked by EMDIOC. The device supports adding delay on EMDIO data output (command and write data), controlled by EMDIO_CFG[EHOLD] and [MDIO_HOLD].	I/O
IEEE 1588 Timer		

Table continues on the next page...

Table 767. External Signals (continued)

Signal Name	Description	I/O
NETC_TMR_CLK	1588 clock in. External high precision timer reference clock input (chip external input pin). Max 200MHz.	I
NETC_TMR_TRIG1	1588 trigger in 1. External timer trigger input 1.This is an asynchronous general purpose input (chip external input pin).	I
NETC_TMR_TRIG2	1588 trigger in 2. External timer trigger input 2.This is an asynchronous general purpose input (chip external input pin).	I
NETC_TMR_GCLK	1588 clock out. Phase aligned timer clock divider output (chip external output pin).	O
NETC_TMR_PP1	1588 pulse out 1. Timer pulse per period 1. It is phase aligned with 1588 timer clock (chip external output pin).	O
NETC_TMR_PP2	1588 pulse out 2. Timer pulse per period 2. It is phase aligned with 1588 timer clock (chip external output pin).	O
NETC_TMR_PP3	1588 pulse out 3. Timer pulse per period 3. It is phase aligned with 1588 timer clock (chip external output pin).	O
NETC_TMR_ALARM1	Timer current time is equal to or greater than alarm time comparator register. User deactivate this output by writing to the TMR_ALARM1_L register. After deactivating it, the user can rearm the alarm by writing to the TMR_ALARM1_H, enabling the triggered alarm to assert the output (chip external output pin).	O
NETC_TMR_ALARM2	Timer current time is equal to or greater than alarm time comparator register. User deactivate this output by writing to the TMR_ALARM2_L register. After deactivating it, the user can rearm the alarm by writing to the TMR_ALARM2_H, enabling the triggered alarm to assert the output (chip external output pin).	O

In the table below, *n* represents the Ethernet link number (0-4).

Table 768. NETC Ethernet link mapping to parallel interface

External Signal	MII	RMII	RGMII
ETH*n*_RX_CLK	MII_RXCLK		RGMII_RXC
ETH*n*_RX_DV	MII_RXDV	RMII_CRS_DV	RGMII_RXCTL
ETH*n*_RX_ER	MII_RXER	RMII_RXER	
ETH*n*_RX_DATA0	MII_RXD0	RMII_RXD0	RGMII_RXD0
ETH*n*_RX_DATA1	MII_RXD1	RMII_RXD1	RGMII_RXD1
ETH*n*_RX_DATA2	MII_RXD2		RGMII_RXD2
ETH*n*_RX_DATA3	MII_RXD3		RGMII_RXD3
ETH*n*_TX_CLK	MII_TXCLK	RMII_REF_CLK	RGMII_TXC
ETH*n*_TX_EN	MII_TXEN	RMII_TXEN	RGMII_TXCTL
ETH*n*_TX_ER	MII_TXER		
ETH*n*_TX_DATA0	MII_TXD0	RMII_TXD0	RGMII_TXD0
ETH*n*_TX_DATA1	MII_TXD1	RMII_TXD1	RGMII_TXD1

Table continues on the next page...

Table 768. NETC Ethernet link mapping to parallel interface (continued)

External Signal	MII	RMII	RGMII
ETH*n*_TX_DATA2	MII_TXD2		RGMII_TXD2
ETH*n*_TX_DATA3	MII_TXD3		RGMII_TXD3
ETH*n*_CRS	MII_CRS		
ETH*n*_COL	MII_COL		

Table 769. NETC Ethernet link mapping to serial interface

SerDes Module Port	NETC Ethernet Link
SERDES0 PORT0	Ethernet Link 0 MII
SERDES1 PORT0	Ethernet Link 1 MII
SERDES2 PORT0	Ethernet Link 2 MII
SERDES1 PORT1	Ethernet Link 3 GMII
SERDES1 PORT2	Ethernet Link 4 GMII

53.4.4 Initialization

This section covers the basic steps to configure NETC for basic operation. It is not intended to provide a complete setup of the device as there are many features and parameters that are optional.

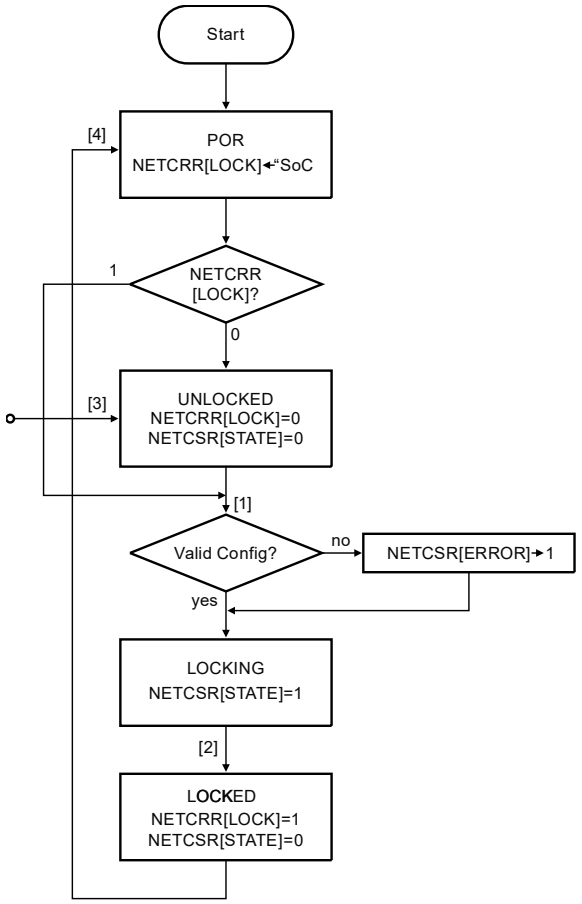
The NETC initialization consists of two initialization stages:

1. Global initialization
2. Functions initialization

53.4.4.1 Global Initialization

Global initialization starts from the device reset state, and involves setting the IERB registers and initializing components that are not associated to one specific function (for example, common memory, Ingress Port Filter TCAM, Time Gate Scheduling Memory, and so on).

The global initialization sequence (or state machine) is shown in the figure below.



POR State
NETCRR[LOCK] reset value is taken from SoC, once SoC releases internal hard reset signal to NETC

UNLOCKED State
In this state IERB registers have normal write access.

LOCKING State
In this state the following occurs:
- Global registers where their PoR values are coming from the IERB registers, are loaded with IERB values
- Free list initialization, etc
- NETCSR[STATE] = 1
  OSR[STATE] = 1 for the switch function and all of the ENETC physical functions.

Write access not allowed to IERB registers, except NETCRR.

LOCKED State
NETC is ready (operational state).
Write access not allowed to IERB registers, except NETCRR.

Transitions
[1] NETCRR[LOCK]←1 **or** PCI_CFH_CMD[MEM_ACCESS]←1(one or more functions).
If IERB is misconfigured while locking, the NETCSR[ERROR] bit is set. This bit is mainly for S/W debug as no trusted entity should initialize IERB incorrectly. Operating while in an error state can cause undefined results.

[2] Initialization completed..

[3] NETCRR[LOCK]←0: Warm Reset (Unlock)
Moving from a locked state to an unlocked state for reconfiguration of IERB registers, forces FLR to all functions, while IERB state remains. See Reset section for details of FLR behavior.

[4] NETCRR[SR]←1: Soft Reset
Return NETC to PoR state, see Reset section for more details.

Figure 365. NETC Global Initialization State Machine

The first part of the global initialization involves setting the IERB registers, which can be set to the PoR IERB values or the updated values from the boot-time software. The latter requires that the IERB be put in the “UNLOCKED” state, before the IERB registers can be updated by the boot-time software. When the IERB enters the LOCKING state, the initialization of global resources starts, which also includes copying of the IERB register values that are used to determine the reset values of the global NETC registers, into their respective global NETC registers.

The global initialization sequence completes when the IERB transitions to the “LOCKED” state, including setting NETCSR[STATE] to 0, which is used to indicate to the boot-time software that the global initialization is completed. Once the global initialization is completed, the initialization of each NETC function can then start.

Below are more details on how IERB registers are initialized during the global initialization sequence.

- **Hard reset** - SoC signals the PoR value of NETCRR[LOCK], before releasing NETC from a hard reset (or PoR).
  - If the PoR value of NETCRR[LOCK] is set 1b, then after NETC is released from the hard reset, the IERB registers are set to their PoR values, and the IERB is transitioned to the LOCKING state.
  - If the PoR value of NETCRR[LOCK] is set 0b, then after NETC is released from the hard reset, the IERB registers are set to their PoR values, and the IERB is transitioned to the UNLOCKED state. The boot-time software then can update the IERB registers. Once the update of the IERB registers is completed, the boot-time software should read NETCSR[ERROR] to ensure the configuration is valid. If valid, then it must set NETCRR[LOCK] to 1b, which then will cause the IERB to transition to the LOCKING state.
- **Soft reset –** A soft reset is initiated by setting NETCRR[SR] to 1. The PoR value of NETCRR[LOCK] comes from PoR value signaled by the SoC, at the time the NETC executes the soft reset sequence.

— If the PoR value of NETCRR[LOCK] is set 1b, then the IERB registers are set to their PoR values, and the IERB is transitioned to the LOCKING state.

— If the PoR value of NETCRR[LOCK] is set 0b, then the IERB registers are set to their PoR values, and the IERB is transitioned to the UNLOCKED state. The boot-time software can poll NETCRR[SR] until it returns 0b, which indicates that the soft reset sequence is completed. It then can proceed with updating the IERB registers. Once the update of the IERB registers is completed, boot-time software should read NETCSR[ERROR] to ensure that the configuration is valid. If valid, then it must set NETCRR[LOCK] to 1b, which will cause the IERB to transition to the LOCKING state.

- **Warm reset –** A warm reset is initiated by setting NETCRR[LOCK] to 0b. This triggers the IERB to transition to the UNLOCKED state, with the IERB register retaining their values. The boot-time software can poll NETCRR[LOCK] until it returns 0b, which indicates that the warm reset sequence is completed. It then can proceed with updating the IERB registers. Once the update of the IERB registers is completed, the boot-time software should read NETCSR[ERROR] to ensure that the configuration is valid. If valid, then it must set NETCRR[LOCK] to 1b, which will cause the IERB to transition to the LOCKING state.

53.4.4.2 Functions Initialization

The initialization of each function must be invoked after global initialization completes. A function initialization must also be invoked after its function level reset.

After the global initialization is completed, and prior to start the initialization of each NETC function, standard PCIe enumeration and discovery techniques can be used to find out the capabilities/functions supported by NETC. This includes timer, EMDIO, switch and ENETC functions. If SR-IOV capability is supported, number of VFs (virtual station interfaces) can be discovered. Base addresses for accessing device registers are determined by the PCIe Enhanced Allocation Capability structure. However, it is not necessary to discover these addresses through PCIe enumeration because these addresses are fixed and known for a given SoC address map. The BARs in ECAM are not used.

53.4.4.2.1 Common Function Initialization

The common function initialization procedures are illustrated in the figure below.

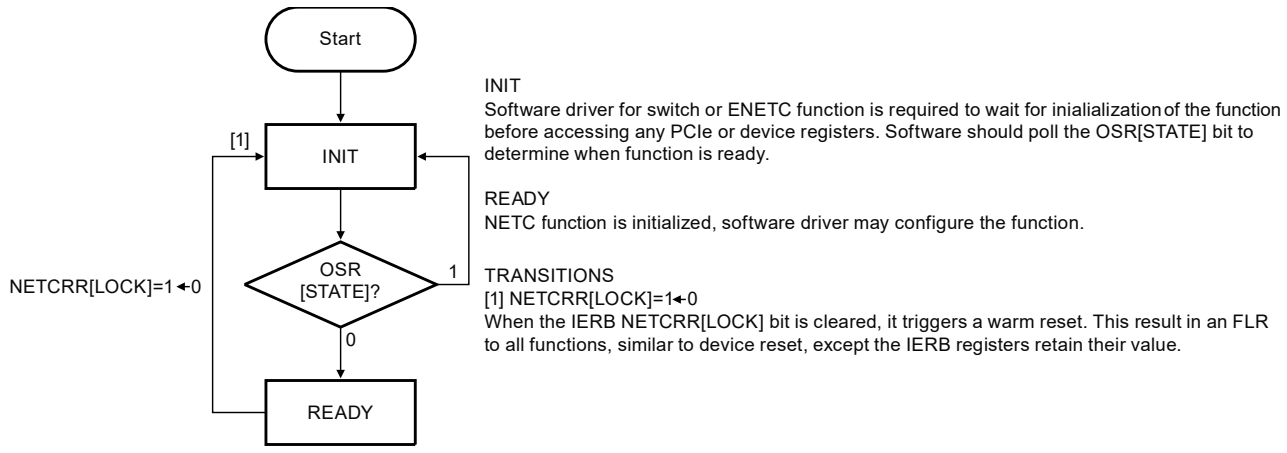


Figure 366. NETC function initialization

Perform the following initialization steps for each NETC function (non-virtual and virtual functions).

1. **Enable the function** – The initialization steps is dependent on whether the function is non-virtual or virtual. Enabling the function will load the IERB register values into the function registers, and then start the initialization of the function.

    a. **Non-virtual function** - Enable the function to master memory transaction requests by setting PCI_CFH_CMD[BUS_MASTER_EN] to 1b, and to respond to memory space accesses (register accesses), by setting PCI_CFH_CMD[MEM_ACCESS] to 1b.

b. **Virtual function** - VFs must be enabled in their corresponding PF's ECAM registers by setting VF_ENABLE and VF_MSE in the PF's PCIE_CFC_SRIOV_CTL register, to 1b. Each VF must then be enabled by setting their PCI_CFH_CMD[BUS_MASTER_EN] to 1b.

2. **Ensure the function is operationally ready** – For the switch or ENETC physical function where initialization may take some time, poll the OSR[STATE] bit to determine when the function is ready for configuration. When OSR[STATE] returns 0b, the function is operationally ready, and can now be configured. Before proceeding with configuration, OSR[ITM_STATE] should be checked to ensure that the sum of all index tables do not exceed the allotted memory size.
3. **Enable MSI-X interrupt** – If interrupt is going to be supported for a function, perform the following steps. See Message Signaled Interrupt (MSI-X), for more information.
    a. For each ENETC function, set/assign PSInCFGR2[NUM_MSIX] (where n is the SI number) based on the maximum MSI-X number indicated in ECAPR1[NUM_MSIX].
    b. Read PCI_CFC_EA_PE1_BASE (ENETC/timer/EMDIO/switch) or PCI_CFC_EA_PE3_BASE (virtual function) to determine the base address of the functions MSI-X table.
    c. Write the table vector entry, as selected by register specified by "MSI-X Table Vector" in Interrupt Classes, with address and data information (ensure that the per-vector mask bit is set to 0b).
    d. Enable MSI-X table by setting PCI_CFC_MSIX_MSG_CTL[MSIX_EN] to 1b and unmask interrupt generation by setting PCI_CFC_MSIX_MSG_CTL[FUNC_MASK] to 0b.
    e. Set the corresponding function vector registers and enable the selected interrupt using the register defined by "Device Enable" in Interrupt Classes.

53.4.4.2.2 ENETC Physical Function Basic Configuration

Perform the following configuration steps for each ENETC physical function:

1. **Discover ENETC physical function capabilities** – Software can read the ENETC capability registers to determine such things as number of Rx/Tx BD rings available, index table sizes, port features and Ethernet links bound to the ENETC instance, and so on. Then follow with the required port configuration (for example, MAC port speed, interface mode) and feature configuration.
2. **Index table memory configuration -** An ENETC instance may resize its index tables as needed within the index table memory size, specified by ITMCAPR[NUM_WORDS]. Afterwards, the OSR[ITM_STATE] must be read to ensure that the sum of all index tables do not exceed the allotted memory size. The function will not be able to configure any index table memory entries unless memory is properly configured.
3. **Tx/Rx BD ring assignment to SIs -** Distribute the Tx/Rx BD rings as necessary among SIs enabled for the ENETC instance. By default, all Tx/Rx BD rings available to the ENETC instance are assigned to the PSI.
4. **Configure the ENETC primary MAC addresses** – Set register PMAR0/1 for SI 0 and PSIaPMAR0/1 for SI 1, 2 .. a (optionally pre-configured in IERB).
5. **Allow SIs to be enabled** – Set PMR[SIaEN] to 1b for every SI present under the ENETC instance. By default, the PSI (SI 0) is set to 1b.
6. **Enable the PSI** - Enable the PSI to operate by setting SIMR[EN] to 1b. By default, the PSI (SI 0) is enabled.
7. **Configure Command BD ring** - Configure Command BD ring for the PSI. For more details, see Table Management Commands.
8. **Configure Tx/Rx BD rings** - Configure Tx/Rx BD rings for the PSI. For more details, see Transmit Buffer Descriptor Rings and Receive Buffer Descriptor Ring.
9. **Enable Tx MAC datapath**
    a. Set PM0_COMMAND_CONFIG[TX_EN] to 1b; this step is not applicable for a pseudo MAC.
    b. Set POR[TXDIS] to 0b.
10. **Enable Rx MAC datapath**

a. Set POR[RXDIS] to 0b.

b. Set PM0_COMMAND_CONFIG[RX_EN] to 1b; this is step is not applicable for a pseudo MAC.

53.4.4.2.3 ENETC Virtual Function Basic Configuration

Perform the following configuration steps for each ENETC virtual function:

1. **Discover ENETC virtual function capabilities** – Software can read the ENETC SI capability registers to determine such things as number of Rx/Tx BD rings available, offload features present, and so on. Then follow with the required feature configuration.
2. **Enable the VSI** - Enable the VSI to operate by setting SIMR[EN] to 1b.
3. **Configure Command BD ring** - Configure Command BD ring for the PSI. See Table Management Commands for more details.
4. **Configure Tx/Rx BD rings** - Configure Tx/Rx BD rings for the VSI. See Transmit Buffer Descriptor Rings and Receive Buffer Descriptor Ring, for more details.

53.4.4.2.4 Switch Basic Configuration

Perform the following configuration steps for each switch basic function:

1. **Discover switch function capabilities** – Software can read switch capability registers to determine such things as index table sizes, port features and Ethernet links bound to the switch, features present, and so on. Then follow with the required port configuration (for example, MAC port speed, interface mode) and feature configuration.
2. **Index table memory configuration -** A switch may resize its index tables as needed within the index table memory size, specified by ITMCAPR[NUM_WORDS]. Afterwards, the OSR[ITM_STATE] must be read to ensure that the sum of all index tables do not exceed the allotted memory size. The function will not be able to configure any index table memory entries unless memory is properly configured.
3. **Configure Command BD ring(s)**- Configure command BD ring(s) for the switch. See Table Management Commands, for more details.
4. **Enable switch for basic forwarding** - Currently, after power-on-reset, switch configuration is such that no packets are forwarded to the egress ports. So, the following basic configuration is needed to enable switch to forward packets:
    a. **Configure the VLAN filter hash table default entry configuration register 0 (VFHTDECR0)** – By default, to enable forwarding to all ports, set port bitmap in VFHTDECR0[PORTn] to 01b for every switch port.
    b. **Configure the VLAN filter hash table default entry configuration register 2 (VFHTDECR2**) - Out of reset, hardware MAC learning is enabled and FDB lookup (Table ID 15) is performed, and therefore, while transmitting a frame if there is no match, the frame is flooded. This configuration can be changed using the VLAN filter hash table default entry configuration registers 2 (VFHTDECR2).
    c. **Configure the Bridge port spanning tree group state register (BPSTGSR**) - BPSTGSR is configured for all supported switch ports. Set the BPSTGSR[STG_STATE] to 2 (forwarding) for all the STG groups.
5. **Enable Tx MAC datapath**-For every switch port, perform the following steps:
    a. Set PM0_COMMAND_CONFIG[TX_EN] to 1b; this is step is not applicable for a pseudo MAC.
    b. Set POR[TXDIS] to 0b.
6. **Enable Rx MAC datapath**-For every switch port, perform the following steps:
    a. Set POR[RXDIS] to 0b.
    b. Set PM0_COMMAND_CONFIG[RX_EN] to 1b; this step is not applicable for a pseudo MAC.

53.4.4.2.5 Timer Basic Configuration

For the timer initialization steps, refer to Initialization.

53.4.4.2.6 EMDIO Basic Configuration

For the EMDIO initialization steps, refer to PHY Management Interface (MDIO).

53.4.5 Application Information

This section describes the usage models for the following functional areas:

1. Forwarding Database (FDB) entry aging, and
2. Spanning Tree Protocol (STP) support

53.4.5.1 FDB Entry Aging Usage Model

53.4.5.1.1 NETC Aging Support

Aging of the FDB entries refers to the process of deleting the entries in the FDB table that have not been used for some time. An FDB entry is not used if no frames are forwarded to the MAC address stored in the FDB entry. If an FDB entry has not been used during a period of the aging time, the entry is eligible to be deleted. A typical aging time is in the order of 300 seconds or 5 minutes. However, the aging time is normally configurable, and the exact value used may depend on many factors.

It is worth noting that in most applications, it is not overly important for the aging procedure to age the entries at precisely the times they are expected to be aged. This is why the aging times are normally expressed in the units of seconds or minutes and not, for example, in milliseconds. In other words, it is often acceptable to delete the aged FDB entries with a lower time accuracy.

In NETC, FDB entries can be added to the FDB table (table ID 15) either by software or autonomously by hardware. If enabled, hardware can add or update FDB entries while processing a frame. While software can add either static or dynamic FDB entries, hardware can only add dynamic FDB entries. In normal circumstances the process of aging of the FDB entries applies only to the dynamic entries. The static FDB entries are normally never aged; they are intended to stay in the FDB table for longer periods of time, i.e., until they are explicitly removed by the application.

In NETC the process of aging of the FDB entries does not happen autonomously in hardware; software needs to be involved in order to implement the FDB entry aging function. NETC hardware does not delete the aged entries automatically nor it determines directly if an entry has aged or not. However, the following hardware mechanisms are available and can be used to assist software and make it relatively easy to implement the FDB entry aging function:

1. FDB entry activity flag - when hardware matches on an FDB entry (based on the DMAC) while processing a frame, it sets the activity flag in the entry to indicate that the entry has been used by hardware.
2. FDB entry activity counter - when hardware processes an activity update management command for an entry in the FDB table and the entry does not have its activity flag set, the activity counter is incremented. If, however, the activity flag is set, then both the activity flag and activity counter are reset. Software can issue the activity update management commands at predefined times and the value of the activity counter can then be used to estimate the period of how long an FDB entry has been inactive.
3. Management command search access method - while issuing an activity update management command for the FDB table, software can use the search access method to update the activity element of multiple FDB entries with a single management command. For example, a single activity update management command could be used to process all the dynamic entries in the FDB table. Similarly, a single delete management command could be used to delete all the FDB entries that match certain search criteria. Being able to use a single activity update or delete command instead of multiple commands can make the associated software logic much simpler.

An important factor in developing the aging strategy is the number of the FDB entries, in particular the number of the dynamic entries, that are supported in the system. In NETC the entries of the FDB table are stored in a dedicated internal memory together with the entries of all the other hash-based tables. The number of the memory entries allocated to all the hash-based tables in a switch is controlled through the IERB switch 0 hash table memory allotment register (S0HTMAR).

The number of the dynamic entries that can be created in the switch is controlled by the value of the DYN_LIMIT field of the FDB hash table memory configuration register (FDBHTMCR). Naturally, the switch entry limit should not be larger than the number of entries available in the hash table memory. The per switch port number of the dynamic entries that can be created is controlled by the value of the DYN_LIMIT field of the bridge port configuration register (BPCR). Again, any of the port entry limits should not be larger than the overall switch entry limit.

The above memory and entry limits have power on default values that are suitable for most of the typical applications. These power on default values can be modified if needed.

53.4.5.1.2 NETC Usage Model Overview

In NETC, software has flexibility in how it may implement the aging of the FDB table entries. In this section we provide an overview of one possible implementation and also point out some alternations and modifications to this implementation that could be exploited.

In general, the aging process involves two main procedures. One is to identify the FDB entries that have aged and the other one is to delete the aged entries. We consider these procedures one by one.

53.4.5.1.2.1 Aging of FDB Entries

To decide if an FDB entry has aged or not one must measure the time of how long the entry has not been used and compare it with the desired aging time. In NETC, the FDB table entries do not feature any entry creation or usage timestamps or autonomous counters, however, each entry features, the mentioned earlier, activity counter. Therefore, software can issue an FDB activity update management message which, if the entry continues to be unused, results in incrementing the FDB activity counter. Issuing such activity update management messages at specific times or with a specific period can provide a good estimate of how long an FDB entry has been inactive.

For example, if the activity update management messages were issued every minute, then an entry with an activity counter with a value of 5 would indicate that the entry has not been used in the last 5 to 6 minutes. If such activity update management commands used the search access method set to match the dynamic entries, then each such command would update the activity counter for all the dynamic FDB entries currently present in the FDB table.

In other words, a single activity update management command issued periodically with a relatively large period, e.g., one minute or more, could be used to update the activity counters in all the dynamic entries present in the FDB table providing inactivity time estimates for all these entries.

53.4.5.1.2.2 Deleting of Aged FDB Entries

With the inactivity time estimates available through the values stored in the activity counters, what is left to complete the FDB entry aging process is to delete the FDB entries that are considered to have aged. This could be done with a delete management command using the search access method set to match on the dynamic entries with a specific value of the activity counter. For example, such a delete management command could be issued periodically, e.g., every minute, and delete all the dynamic FDB entries with the activity counter having the value of 5.

53.4.5.1.2.3 Usage Model Summary

To sum up, in order to implement the FDB aging function, software can periodically issue an FDB table activity update and entry delete management commands. The aging time can be adjusted by modifying the period of how often the management commands are issued and also by choosing the activity count value, used as a search criterion in the delete command.

Consider the example depicted in the figure below. As shown, software periodically issues multi-entry search activity update and delete management commands. There are six such activity update and delete management commands shown. Each of the activity update commands processes all the dynamic FDB entries present in the table. Each of the delete commands attempts to delete all the dynamic FDB entries with the activity counter value of four (4). This means that the aging time in the example is equivalent to the time needed for the activity counter to reach the value of four.

Each of the triangle, square and circle symbols on the horizontal time axis in the middle of the figure indicates a time when a frame arrives at a switch port. Each of the three shapes indicates frames with a distinct destination MAC address. In other words, the triangle shape indicates one destination MAC address value, the square shape indicates another MAC address value, and the circle shape indicates a third MAC address value. There is one triangle frame, three circle frames and four square frames shown.

The FDB entries associated with the triangle, square and circle addresses are shown below the time axis. The states of these FDB entries are shown at times t_1, t_2, t_2, t_3, t_4, t_5, and t_6.

At time t_1 there are three dynamic entries present in the FDB table. The three entries correspond to the mentioned above triangle, square and circle MAC addresses. All three entries have their activity flag (F) reset (0) and the activity counter value (C) of zero

(0). Though it is not shown in the figure, it could be assumed that the triangle, square and circle entries were created by hardware when the frames with the triangle, square and circle source MAC addresses arrived at the switch prior to time t_1.

During the (t_1, t_2) time interval one triangle, one square and one circle frame arrive at the switch. The arrival of these frames results in setting the activity flags in all three entries. However, at time t_2 the three FDB entries look the same as they did at time t_1. This is because the activity update command issued before time t_2 resulted in resetting the activity flags and counters in all three entries.

During the (t_2, t_3) time interval one circle frame arrives at the switch before the activity update command is issued. Therefore, at time t_3 the FDB entry with the circle address has the activity flag and counter reset. The other two entries are not active (F=0), and their activity counter values are incremented to 1.

During the (t_3, t_4) time interval one square frame arrives before the activity update command is issued and one circle frame arrives after the activity command is issued. Therefore, at time t_4 the square FDB entry has the activity flag and counter reset. The circle FDB entry has the activity counter incremented to 1 by the activity update command and the activity flag set when the frame is received. The triangle entry continues to be inactive, so its activity counter value is incremented to 2.

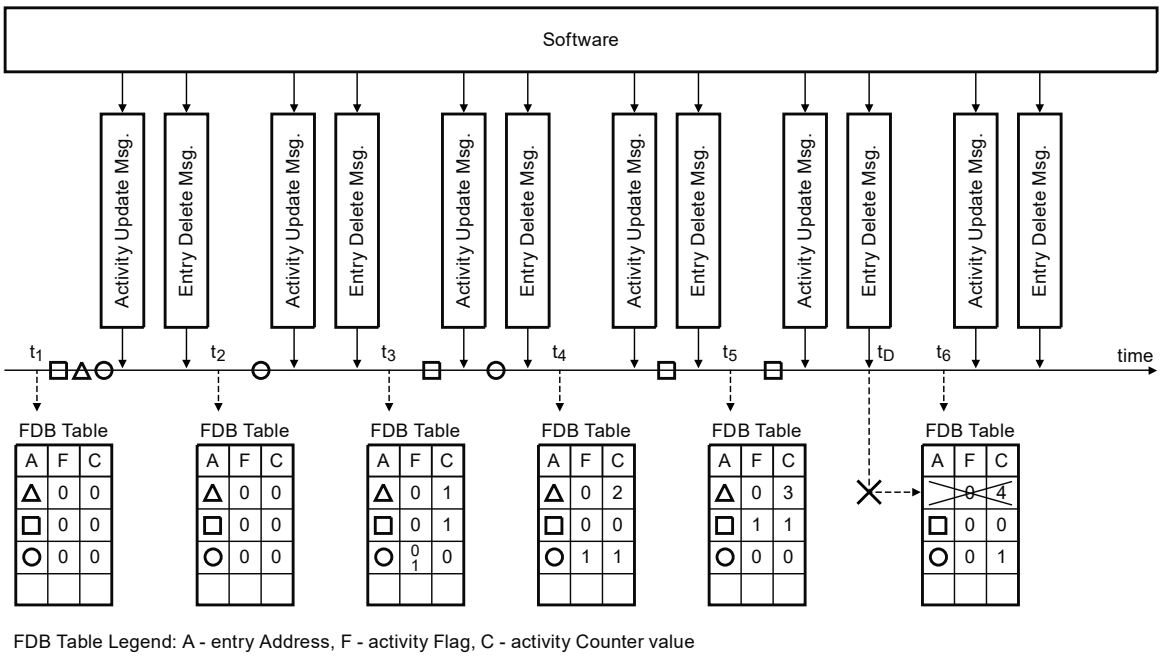


Figure 367. FDB Entry Aging Usage Model

During the (t_4, t_5) time interval one square frame arrives at the switch after the activity update command is issued. Therefore, at time t_5 the FDB entry with the square address has the activity counter value incremented to 1 by the activity update command and the activity flag set by the arrival of the frame. The previously active circle entry has its activity flag and counter reset by the activity update command. The triangle entry continues to be inactive and its activity counter value increments to 3.

During the (t_5, t_6) time interval one square frame arrives at the switch before the activity update command is issued. Therefore, at time t_6 the FDB entry for the square address has the activity flag and counter reset. The circle entry is inactive, so its activity counter value is incremented to 1. The triangle entry continues to be inactive and its activity counter value increments to 4 when processing the activity update message. The triangle entry has aged, and it became eligible for deletion. The delete command arriving at time tD results in deleting the triangle FDB entry so that at time t_6 the triangle entry is not present in the table anymore.

The procedure described in this section is an introduction into how software may implement the aging of the FDB entries in NETC.

53.4.5.1.3 Implementation Consideration

53.4.5.1.3.1 Exact Match of the FDB Activity Count Value

The value of the FDB entry activity counter can be used as the search criterion in search management commands such as multi-entry update, delete or query commands. In NETC, hardware matches on the exact activity counter value specified in the command. For example, if the specified activity counter value is 10, hardware will match on the value 10 and not on any other value, including values greater than 10. This in turn means that if an FDB entry is to be processed, e.g., queried or deleted, when its activity counter reaches the value of 10, the entry needs to be processed before the counter value increments. If the counter value is allowed to increment to 11 or a higher value, the activity counter value match criterion of 10 would result in no match for this entry.

As mentioned before, in the context of aging of the FDB entries, the activity counter match criterion can be used in the multi-entry delete management commands. In order to reliably delete the FDB entries with a certain value of the activity counter, software must ensure that the entries are deleted before their activity counter value could be incremented. One way to achieve this is to issue the aging activity update and delete management commands with the same frequency or the same period. In other words, each aging activity update command is always followed by an aging delete command which is followed by the next activity update command, etc. This ensures that the FDB activity counter values are not incremented beyond the value at which they are to be deleted.

53.4.5.2 Spanning Tree Protocol Usage Model

53.4.5.2.1 Usage Model Overview

The switch component of the NETC provides support for the Spanning Tree Protocol (STP) and the Multiple Spanning Tree Protocol (MSTP). STP variants such as the Rapid Spanning Tree Protocol (RSTP) are also supported. The STP support allows for basic interconnect of multiple Ethernet switches with redundant paths, without forwarding loops. Currently, NETC supports up to 16 concurrently active STP groups.

In NETC, the STP support takes place in the context of ingress and/or egress VLAN frame processing. During the VLAN processing of a frame, hardware first establishes which STP group the frame being processed is associated with. Once this is done, hardware retrieves the current STP port state for the established STP group and, depending on the STP port state value, the frame is either allowed to ingress or egress the system or it is dropped.

Note that the VLAN frame processing is not applied to frames being "stream forwarded". When the stream forwarding function is selected, the 802.1Q bridge forwarding function (includes VLAN frame processing) is by-passed.

The initial STP group ID to use is stored by software in the spanning tree group member ID (STG_ID) field of the switch VLAN filter table default entry configuration register (S0VFHTDECR0). This initial STP group ID is associated with all the frames ingressing into the switch. The STP group ID to use may then be modified depending on the result of the ingress VLAN filter table lookup. If the VLAN filter table lookup resulted in a match, then the STP group ID in the found entry is used. If the VLAN filter table lookup resulted in a miss, then the initially determined STP group ID is used. The STP group ID values stored in the VLAN filter table entries are, of course, controlled by software.

Once the STP group ID to use has been determined, hardware uses it to retrieve the current STP state of the port to decide how to process the frame. The port STP state values for all the supported STP groups are stored in the bridge port spanning tree group state registers (BPSTGSRs). The port STP state values for all the supported STP groups are, of course, controlled by software.

On ingress, after determining the STP state of the ingress port, the decision is made if the frame should trigger the learning action, be dropped, or processed further. On egress, the STP state of the egress port is used to decide if the frame should be dropped or transmitted. Note that at any given time, different egress ports may have different STP state values associated with them. This means that if a frame is replicated it could, depending on the port STP state value, successfully egress on some ports and be dropped on other egress ports.

In summary, the NETC STP support in a switch is controlled by software through the spanning tree group member ID (STG_ID) field of the switch VLAN filter table default entry configuration register, the bridge port spanning tree group state registers and the VLAN filter table entries. Software can modify the STP related fields in these registers and table entries and hardware uses these software supplied values while processing the frames.

The figure below illustrates this graphically. The figure shows the STP processing steps during the ingress frame processing. Initially, STG_ID* is assigned to the frame using the value found in the switch VLAN filter table default entry configuration register. Then a lookup is performed into the ingress VLAN filter table. If the lookup results in a match, the STG_ID found in the table entry is

used. The final STG_ID value is used to retrieve the STG group state for the port of interest and the frame is processed according to this STG group state value.

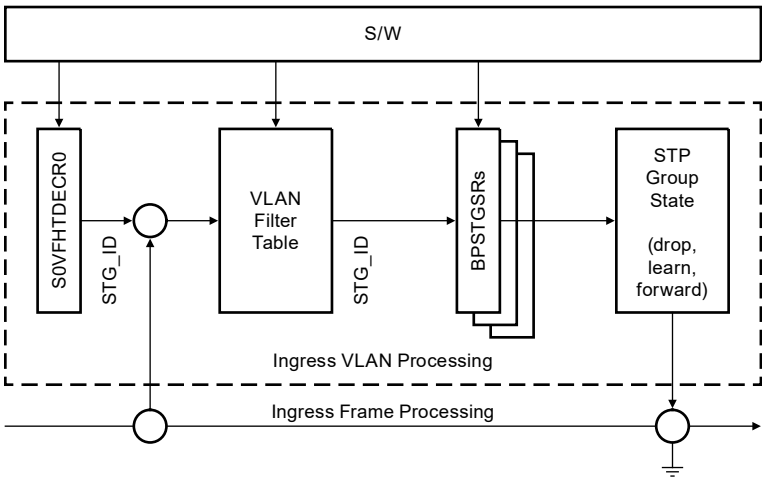


Figure 368. NETC Frame Ingress STP Processing

During the egress processing of the frame, the frame STG ID determined during the ingress processing is used once more to retrieve the current STG group state for the egress port of interest. This takes place for each egress port the frame egresses on.

53.4.5.2.2 Implementation Considerations

The STG_STATE fields in the bridge port spanning tree group state register (BPSTGSR) are 2 bits in size. Out of the 4 possible combinations available with the 2 bits, 3 are used to define STP states and the 4th one is reserved.

While RSTP protocol uses three states, the original STP protocol uses five states. While software can maintain all the five STP states, only three state values can be stored in the BPSTGSR register. In order to represent the five STP states with only the three available states in the STG_STATE field, the state mapping shown in the table below is recommended:

Table 770. Mapping of STP Protocol States to STG_STATE Field States

STP Protocol States	STG_STATE Field States
Disabled	Disabled
Blocking	Disabled
Listening	Disabled
Learning	Learning
Forwarding	Forwarding

The suggested state mapping results in the expected STP protocol behavior in which frames are dropped in the Disabled, Blocking, Listening and Learning STP states, learning takes place only in the Learning STP state and forwarding of frames takes place only in the Forwarding STP state.

53.4.6 Register Descriptions

NOTE

Systems may have software which runs at different privilege levels. In this document the term privileged or privileged software refers to the most privileged software layer, which could be Virtual Memory Manager (VMM) an operating system or software hypervisor.

This section provides a detailed description of all accessible NETC memory and registers. Virtualization and isolation is accomplished through the separation of registers using different PCI BARs. The overall mapping of these spaces, within the PCI Express memory allocated for NETC, can be seen in table Table 356. Features surrounding a multiple port implementation may have asymmetric capabilities, i.e. VSIs and transmit/receive rings.

PCIe register are byte accessible, while device registers are 32-bit only. Transactions larger than 32-bit to NETC memory mapped space are not supported, and will result in transaction error. 64-bit counters are represented as two 32-bit registers with a lower index 0 and upper index 1. For a consistent reading of a 64-bit counter, the lower index 0 must be read first, followed by index 1.

Reserved bits should be written with 0 and ignored on read. Unimplemented registers within a switch, ENETC and VSI function's address space reads as zero and returns a transaction error, while writes have no effect and may return error for some register implementations. A write to read-only register return an error only if specifically noted in the register description. For all other functions and mapped address spaces, including EMDIO, timer and PCIe, unimplemented registers return zero and will not generate an error, while writes have no effect. All other accesses to unmapped address space (not covered by a functions BARs) will result in a transaction error.

Dynamic Resource Allocation

Transmit and receive buffer descriptor rings is a generic resource which is bound to an ENETC instance through IERB registers at pre-boot time. The ENETC device driver is then allowed to allocate rings per station interface prior to enabling access to virtual station interfaces. Virtual Station Interfaces (VSIs) follow the same generic resource scheme as buffer descriptor rings, and are bound to an ENETC instance through IERB registers at pre-boot time. The ENETC device driver is then allowed to allocate MSI-X vectors to virtual station interfaces, prior to enabling them.

Switch Instance Memory Map

The switch memory map is 1MB. It includes the mapping of link specific port registers into the address space. The port registers are only present if there is a link bound to the switch port. The type of port depends on the link capability register L*a*CAPR and the link binding register L*a*BCR. The table below shows the register layout of the switch address space.

Table 771. Switch Instance Memory Mapping

Address Offset	Registers
0x0_0000 - 0x0_0FFF	Switch Specific Registers
0x0_1000 - 0x0_1FFF	Switch / ENETC Common Registers
0x0_2000 - 0x0_2FFF	Switch Service Registers
0x0_4000 - 0x0_7FFF (+ 0x4000*p)	Ethernet or Pseudo Link Port *p* (see Table 773)
0x1_0000 - 0x1_FFFF	NETC Global (Read-Only)

ENETC Instance Memory Map

The ENETC memory map is 256KB. It includes the mapping of port-link registers into the address space. The port registers are only present if there is a link bound to the ENETC instance. The type of port depends on the link capability register L*a*CAPR and the link binding register L*a*BCR. The table below shows the register layout of the ENETC address space.

Table 772. ENETC Instance Memory Mapping

Address Offset	Registers
0x0_0000 - 0x0_FFFF	Station Interface Registers (see Table 774)
0x1_0000 - 0x1_0FFF	ENETC Specific Registers
0x1_1000 - 0x1_1FFF	Switch / ENETC Common Registers
0x1_2000 - 0x1_2FFF	ENETC PSI Registers

Table continues on the next page...

Table 772. ENETC Instance Memory Mapping (continued)

Address Offset	Registers
0x1_4000 - 0x1_7FFF	Ethernet or Pseudo linkPort (see Table 773)
0x2_0000 - 0x2_FFFF	NETC Global (Read-Only)

The NETC architecture has a set of links that are bound to either an ENETC or a switch function. There are two types of links supported; Ethernet-link which attaches to either an ENETC or switch port, and a pseudo link which connects an ENETC and switch port together. These links and the corresponding ENETC/switch ports has physical registers associated with them. As these links are bound to an ENETC/switch function, these port-link registers will appear in the functions memory map. The table below shows the layout of registers within the 16KB port-link address space. See the ENETC and switch base memory map for the location of the port-link registers.

Table 773. Generic Port-Link Memory Mapping

Address Offset	Registers
0x0000-0x03FF	Common port registers
0x0400-0x0FFF	ENETC / Switch specific port registers
0x1000-0x13FF	Pseudo MAC registers [Pseudo link type only]
0x1000-0x13FF	Port Express MAC [Ethernet link type only]
0x1400-0x17FF	Port Preemptable MAC [Ethernet link type only]
0x1800-0x1BFF	Port MAC Merge Registers [Ethernet link type only w/ frame preemption support]
0x1C00-0x1CFF	EMDIO PHY Access Registers [Ethernet link type only]
0x1D00-0x3FFF	Reserved

The ENETC station interface memory map is 64KB and is part of the 256KB ENETC instance memory map. All transmit and receive BD rings are by default bound by IERB register to station interface 0. These can be assigned to other station interfaces, if available, for the ENETC instance during the configuration process.Note that the physical station interface (PSI/SI0) differs slightly from the virtual station interface(s) (VSI/SI1+) register definition.

Table 774. ENETC Station Interface Memory Mapping

Address Offset	Registers
0x0000-0x00FF	General config/status
0x0100-0x01FF	VLAN config
0x0200-0x02FF	Messaging
0x0300-0x03FF	Statistical counters
0x0800-0x08FF	Command BD ring

Table continues on the next page...

Table 774. ENETC Station Interface Memory Mapping (continued)

Address Offset	Registers
0x0900-0x09FF	Capability
0x0A00-0x0BFF	Interrupts
0x0C00-0x0DFF	Reserved
0x0E00-0x0EFF	Errors
0x0F00-0x0FFF	Debug
0x1000-0x10FF	Station Interface MAC address filtering capability
0x1100-0x11FF	Station Interface VLAN filtering capability
0x1200-0x12FF	Reserved
0x1500-0x15FF	Time specific departure
0x1600-0x16FF	Reserved
0x1700-0x1FFF	Reserved
0x8000-0x80FF	BD ring 0 Tx (when mapped)
0x8100-0x81FF	BD ring 0 Rx (when mapped)
0x8200-0x82FF	BD ring 1 Tx (when mapped)
0x8300-0x83FF	BD ring 1 Rx (when mapped)
...	..
0x8E00-0x8EFF	BD ring 7 Tx (when mapped)
0x8F00-0x8FFF	BD ring 7 Rx (when mapped)
0x9000-0x90FF	BD ring 8 Tx (when mapped)
0x9100-0x91FF	BD ring 8 Rx (when mapped)
...	...
0x9E00-0x9EFF	BD ring 15 Tx (when mapped)
0x9F00-0x9FFF	BD ring 15 Rx (when mapped)
0xA000-0xFFFF	Reserved

Timer Instance Memory Map

The table below shows the 128KB IEEE 1588 memory map.

Table 775. IEEE 1588 Memory Map

Address Offset	Registers
0x0_0000 - 0x0_007C	General control/status registers
0x0_0080 - 0x0_00FF	IEEE 1588 timer registers
0x1_0000 - 0x1_FFFF	NETC Global (Read-Only)

EMDIO Instance Memory Map

The table below shows the 128KB EMDIO memory map.

Table 776. EMDIO Instance Memory Map

Address Offset	Registers
0x0_0000 - 0x0_FFFF	General control/status registers
0x1_0000 - 0x1_FFFF	NETC Global (Read-Only)

IERB Memory Map

Within the 16MB address space allocated to the Integrated Endpoint Root Complex by the SoC, a 64KB Integrated Endpoint Register Block (IERB) exists for each endpoint. This is used for pre-boot initialization, debug and non-customer configuration. The table below shows how the IERB memory space is organized.

Table 777. Integrated Endpoint Register Block Memory Map

Address Offset	Registers
0x0000-0x000C	NETC capability
0x0020-0x007F	Memory Capability
0x0080-0x01FF	Memory and Table Configuration
0x0200 + 0x10*r	Root Complex Instance r Configuration
0x0300-0x03FF	EMDIO Instance Configuration
0x0400 + 0x40*t	Timer Instance t Configuration
0x0800 + 0x20*l	Link l Capability and Binding
0x1000 + 400*w	Switch Instance w Configuration
0x2000 + 0x100*e	ENETC Instance e Configuration
0x4000 + 0x100*v	VSI Instance v Configuration

Privileged Memory Map

Within the 16MB address space allocated to the Integrated Endpoint Root Complex by the SoC, a 64KB Reserved Register Block exists for each endpoint. This is used for reset control and error handling during functional operation. The table below shows how the privileged memory space is divided.

Table 778. Privileged Register Block Memory Map

Address Offset	Registers
0x0100-0x01FF	NETC reset control and configuration status
0x0E00-0x0EFF	Global error registers

53.4.6.1 NETC PCI Express ECAM PF config register descriptions

This section describes the PCI Express Type0 config header and capabilities as they relate to the physical functions of NETC.

It should be noted that a PCIe function that is not present will return a Vendor ID of 0xFFFF for that function.

53.4.6.1.1 PCI_Config_Header_Type0 memory map

NETC_F0_PCI_HDR_TYPE0 base address: 6000_0000h

NETC_F1_PCI_HDR_TYPE0 base address: 6000_1000h

NETC_F2_PCI_HDR_TYPE0 base address: 6000_2000h

NETC_F3_PCI_HDR_TYPE0 base address: 6000_3000h

NETC_F4_PCI_HDR_TYPE0 base address: 6000_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PCI device ID and vendor ID register (PCI_CFH_DID_VID)	32	R	See section
4h	PCI command register (PCI_CFH_CMD)	16	RW	0000h
6h	PCI status register (PCI_CFH_STAT)	16	R	0010h
8h	PCI revision ID and classcode register (PCI_CFH_REVID_CLASSCODE)	32	R	See section
Ch	PCI cache line size register (PCI_CFH_CL_SIZE)	8	RW	00h
Eh	PCI header type register (PCI_CFH_HDR_TYPE)	8	R	80h
10h	PCI base address register 0 (PCI_CFH_BAR0)	32	R	0000_0000h
14h	PCI base address register 1 (PCI_CFH_BAR1)	32	R	0000_0000h
18h	PCI base address register 2 (PCI_CFH_BAR2)	32	R	0000_0000h
1Ch	PCI base address register 3 (PCI_CFH_BAR3)	32	R	0000_0000h
20h	PCI base address register 4 (PCI_CFH_BAR4)	32	R	0000_0000h
24h	PCI base address register 5 (PCI_CFH_BAR5)	32	R	0000_0000h
2Ch	PCI subsystem vendor ID register (PCI_CFH_SUBSYS_VID)	16	R	1957h
2Eh	PCI subsystem ID register (PCI_CFH_SUBSYS_ID)	16	R	See section
34h	PCI capabilities pointer register (PCI_CFH_CAP_PTR)	8	R	40h
40h	PCI PCIe capabilities list register (PCI_CFC_PCIE_CAP_LIST)	16	R	8010h
42h	PCI PCIe capabilities register (PCI_CFC_PCIE_CAP)	16	R	0092h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
44h	PCI PCIe device capabilities register (PCI_CFC_PCIE_DEV_CAP)	32	R	1000_0000h
48h	PCI PCIe device control register (PCI_CFC_PCIE_DEV_CTL)	16	RW	0000h
4Ah	PCI PCIe device status register (PCI_CFC_PCIE_DEV_STAT)	16	R	0000h
64h	PCI PCIe device capabilities 2 register (PCI_CFC_PCIE_DEV_CAP2)	32	R	0000_0000h
68h	PCI PCIe device control 2 register (PCI_CFC_PCIE_DEV_CTL2)	16	R	0000h
80h	PCI MSI-X capabilities list register (PCI_CFC_MSIX_CAP_LIST)	16	R	9011h
82h	PCI MSI-X message control register (PCI_CFC_MSIX_MSG_CTL)	16	RW	See section
84h	PCI MSI-X table offset/BIR register (PCI_CFC_MSIX_TABLE_OFF_BIR)	32	R	0000_0001h
88h	PCI MSI-X PBA offset/BIR register (PCI_CFC_MSIX_PBA_OFF_BIR)	32	R	0000_0801h
90h	PCI PCI-PM capabilities list register (PCI_CFC_PCIPM_CAP_LIST)	16	R	9C01h
92h	PCI PCI-PM capabilities register (PCI_CFC_PCIPM_CAP)	16	R	4803h
94h	PCI PCI-PM control and status register (PCI_CFC_PCIPM_CTL_STAT)	16	RW	0008h
97h	PCI PCI-PM capabilities data register (PCI_CFC_PCIPM_DATA)	8	R	00h
9Ch	PCI EA capabilities list register (PCI_CFC_EA_CAP_LIST)	16	R	0014h
9Eh	PCI EA capabilities register (PCI_CFC_EA_CAP)	16	R	See section
A0h	PCI EA per-entry 0 format register (PCI_CFC_EA_PE0_FMT)	32	R	80FF_0003h
A4h	PCI EA per-entry 0 base register (PCI_CFC_EA_PE0_BASE)	32	R	See section
A8h	PCI EA per-entry 0 max offset register (PCI_CFC_EA_PE0_MAXOFF)	32	R	See section
ACh	PCI EA per-entry 0 extended base register (PCI_CFC_EA_PE0_EXT_BASE)	32	R	0000_0000h
B0h	PCI EA per-entry 1 format register (PCI_CFC_EA_PE1_FMT)	32	R	8000_0123h
B4h	PCI EA per-entry 1 base register (PCI_CFC_EA_PE1_BASE)	32	R	See section
B8h	PCI EA per-entry 1 max offset register (PCI_CFC_EA_PE1_MAXOFF)	32	R	0000_FFFCh
BCh	PCI EA per-entry 1 extended base register (PCI_CFC_EA_PE1_EXT_BASE)	32	R	0000_0000h
C0h	PCI EA per-entry 2 format register (PCI_CFC_EA_PE2_FMT)	32	R	80FF_0493h
C4h	PCI EA per-entry 2 base register (PCI_CFC_EA_PE2_BASE)	32	R	60C1_0002h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C8h	PCI EA per-entry 2 max offset register (PCI_CFC_EA_PE2_MAXOFF)	32	R	0000_FFFCh
CCh	PCI EA per-entry 2 extended base register (PCI_CFC_EA_PE2_EXT_BASE)	32	R	0000_0000h
D0h	PCI EA per-entry 3 format register (PCI_CFC_EA_PE3_FMT)	32	R	8003_03B3h
D4h	PCI EA per-entry 3 base register (PCI_CFC_EA_PE3_BASE)	32	R	60C2_0002h
D8h	PCI EA per-entry 3 max offset register (PCI_CFC_EA_PE3_MAXOFF)	32	R	0000_FFFCh
DCh	PCI EA per-entry 3 extended base register (PCI_CFC_EA_PE3_EXT_BASE)	32	R	0000_0000h
100h	PCIe AER extended capability header (PCIE_CFC_AER_EXT_CAP_HDR)	32	R	1301_0001h
104h	PCIe AER uncorrectable error status register (PCIE_CFC_AER_UCORR_ERR_STAT)	32	RW	0000_0000h
108h	PCIe AER uncorrectable error mask register (PCIE_CFC_AER_UCORR_ERR_MASK)	32	RW	0040_0000h
10Ch	PCIe AER uncorrectable error severity register (PCIE_CFC_AER_UCORR_ERR_SEV)	32	RW	0040_0000h
110h	PCIe AER correctable error status register (PCIE_CFC_AER_CORR_ERR_STAT)	32	RW	0000_0000h
114h	PCIe AER correctable error mask register (PCIE_CFC_AER_CORR_ERR_MASK)	32	RW	0000_4000h
118h	PCIe AER capabilities and control register (PCIE_CFC_AER_CAP_CTL)	32	R	0000_0000h
130h	PCIe ACS capability header (PCIE_CFC_ACS_CAP_HDR)	32	R	1401_000Dh
134h	PCIe ACS capability register (PCIE_CFC_ACS_CAP)	16	R	0000h
136h	PCIe ACS control register (PCIE_CFC_ACS_CTL)	16	R	0000h
140h	PCIe readiness time reporting capability header (PCIE_CFC_RTR_CAP_HDR)	32	R	See section
144h	PCIe RTR readiness time reporting 1 register (PCIE_CFC_RTR_RTR1)	32	R	8000_0000h
148h	PCIe RTR readiness time reporting 2 register (PCIE_CFC_RTR_RTR2)	32	R	0000_0634h
150h	PCIe SR-IOV capability header (PCIE_CFC_SRIOV_CAP_HDR)	32	R	0001_0010h
154h	PCIe SR-IOV capability register (PCIE_CFC_SRIOV_CAP)	32	R	0000_0000h
158h	PCIe SR-IOV control register (PCIE_CFC_SRIOV_CTL)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
15Ah	PCIe SR-IOV status register (PCIE_CFC_SRIOV_STAT)	16	R	0000h
15Ch	PCIe SR-IOV initial VFs register (PCIE_CFC_SRIOV_INIT_VFS)	16	R	0001h
15Eh	PCIe SR-IOV total VFs register (PCIE_CFC_SRIOV_TOTAL_VFS)	16	R	0001h
160h	PCIe SR-IOV num VFs register (PCIE_CFC_SRIOV_NUM_VFS)	16	RW	0001h
162h	PCIe SR-IOV function dependency list register (PCIE_CFC_SRIOV_FUNC_DEP_LIST)	16	R	0004h
164h	PCIe SR-IOV first VF offset register (PCIE_CFC_SRIOV_FIRST_VF_OFF)	16	R	00FCh
166h	PCIe SR-IOV VF stride register (PCIE_CFC_SRIOV_VF_STRIDE)	16	R	0010h
16Ah	PCIe SR-IOV VF device ID register (PCIE_CFC_SRIOV_VF_DEV_ID)	16	R	EF00h
16Ch	PCIe SR-IOV supported page sizes ID register (PCIE_CFC_SRIOV_SUP_PAGE_SIZES)	32	R	0000_0013h
170h	PCIe SR-IOV system page size ID register (PCIE_CFC_SRIOV_SYS_PAGE_SIZE)	32	R	0000_0010h
174h - 188h	PCIe SR-IOV VF base address register a (PCIE_CFC_VF_BAR0 - PCIE_CFC_VF_BAR5)	32	R	0000_0000h
18Ch	PCIe SR-IOV VF migration state array offset register (PCIE_CFC_SRIOV_VF_MIG_STATE_ARR_OFF)	32	R	0000_0000h

53.4.6.1.2 PCI device ID and vendor ID register (PCI_CFH_DID_VID)

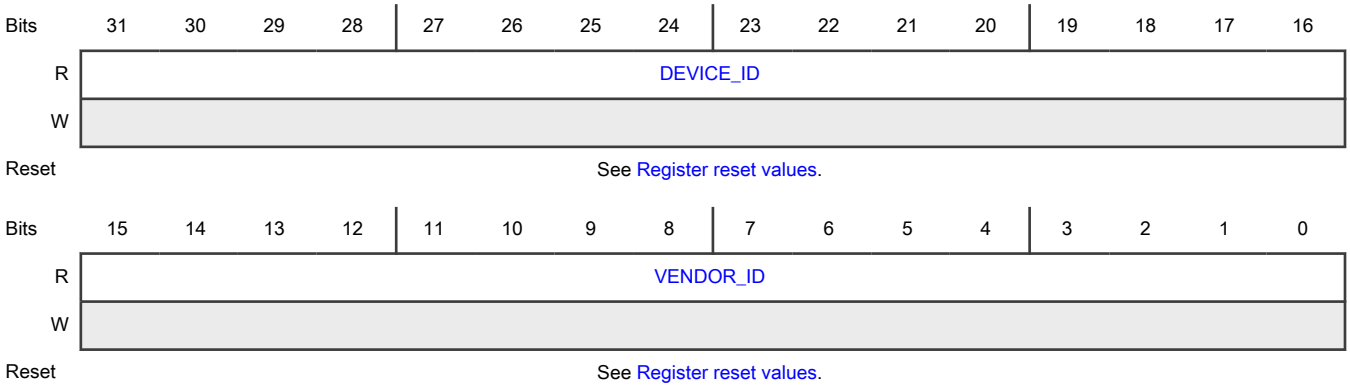
Offset

Register	Offset
PCI_CFH_DID_VID	0h

Function

This is the PCI config header device and vendor ID register. The value read is taken from IERB register <FUNC>__CFH_DIDVID.

Diagram



Register reset values

Register	Reset value
PCI_CFH_DID_VID	NETC_F0_PCI_HDR_TYPE0: EE02_1957h NETC_F1_PCI_HDR_TYPE0: EE00_1957h NETC_F2_PCI_HDR_TYPE0: EEF2_1957h NETC_F3_PCI_HDR_TYPE0: E101_1957h NETC_F4_PCI_HDR_TYPE0: E110_1957h

Fields

Field	Function
31-16 DEVICE_ID	Device ID This field identifies the device ID of the device. This field in all VFs returns FFFFh when read. For SR-IOV the VF device ID is determined from the VF Device ID field in the SR-IOV capability register.
15-0 VENDOR_ID	Vendor ID This field identifies the manufacturer of the device.

53.4.6.1.3 PCI command register (PCI_CFH_CMD)

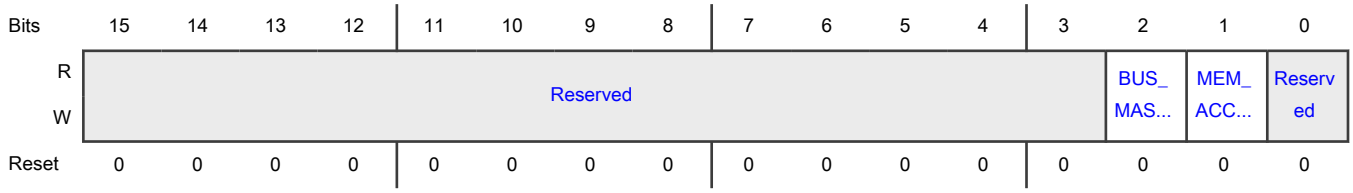
Offset

Register	Offset
PCI_CFH_CMD	4h

Function

This is the PCI config header command register.

Diagram



Fields

Field	Function
15-3 —	Reserved
2 BUS_MASTER_EN	Bus master enable Controls the ability of a PCI Express Endpoint to issue Memory and I/O Read/Write Requests. When this bit is Set, the PCI Express Function is allowed to issue Memory or I/O Requests. When this bit is Clear, the PCI Express Function is not allowed to issue any Memory or I/O Requests. Note that as MSI/MSI-X interrupt Messages are inband memory writes, setting the Bus Master Enable bit to 0b disables MSI/MSI-X interrupt Messages as well.
1 MEM_ACCESS	Controls a device's response to memory space accesses. 0 Disabled 1 Enabled
0 —	Reserved

53.4.6.1.4 PCI status register (PCI_CFH_STAT)

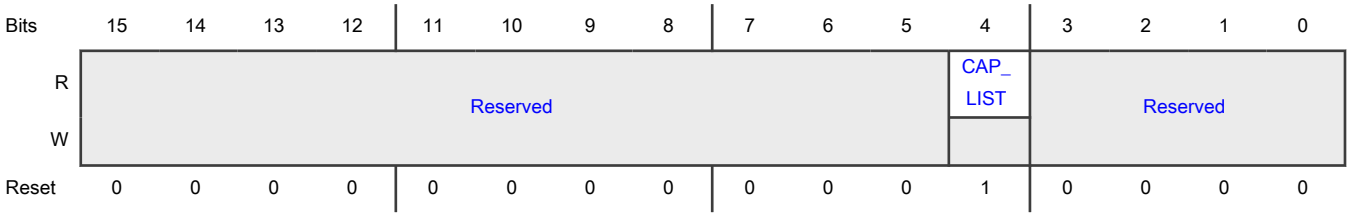
Offset

Register	Offset
PCI_CFH_STAT	6h

Function

This is the PCI config header status register.

Diagram



Fields

Field	Function
15-5 —	Reserved
4 CAP_LIST	Capabilities List Indicates the presence of an Extended Capability list item. Since all PCI Express device Functions are required to implement the PCI Express Capability structure, this bit must be hardwired to 1b.
3-0 —	Reserved

53.4.6.1.5 PCI revision ID and classcode register (PCI_CFH_REVID_CLASSCODE)

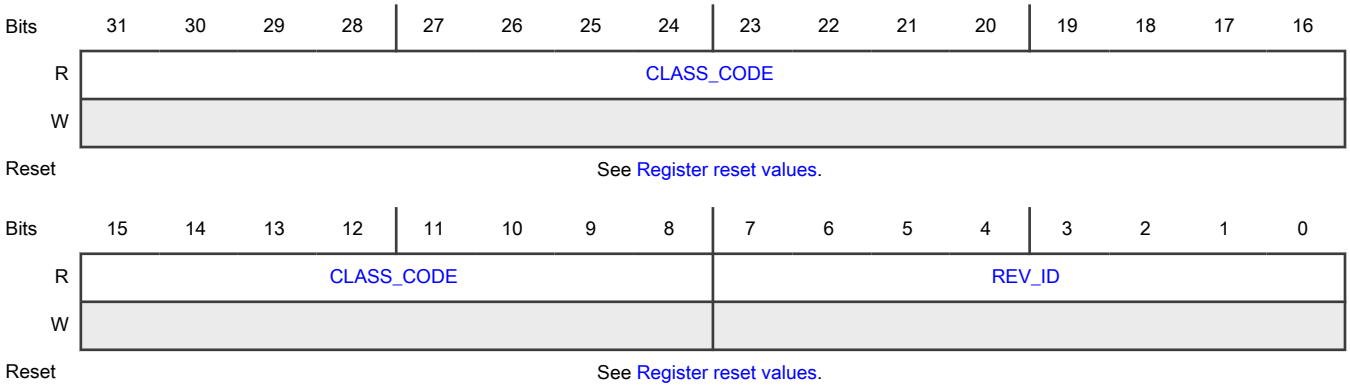
Offset

Register	Offset
PCI_CFH_REVID_CLASSCODE	8h

Function

This is the PCI config header revision ID and classcode register.

Diagram



Register reset values

Register	Reset value
PCI_CFH_REVID_CLASSCODE	NETC_F0_PCI_HDR_TYPE0,NETC_F1_PCI_HDR_TYPE0: 0880_0103h NETC_F2_PCI_HDR_TYPE0–NETC_F4_PCI_HDR_TYPE0: 0200_0103h

Fields

Field	Function
31-8 CLASS_CODE	Class code The Class Code register is read-only and is used to identify the generic function of the device and, in some cases, a specific register level programming interface. The register is broken into three byte size fields. The upper byte (at offset 0Bh) is a base class code which broadly classifies the type of function the device performs. The middle byte (at offset 0Ah) is a sub-class code which identifies more specifically the function of the device. The lower byte (at offset 09h) identifies a specific register-level programming interface (if any) so that device independent software can interact with the device. The field in the PF and associated VFs must return the same value when read. For Ethernet Controllers, the class code is 02h and the sub-class code is 00h. The register interface values must be 00h for Root Complex and 01h for Integrated Endpoint.
7-0 REV_ID	Revision ID This register specifies a device specific revision identifier and is a vendor defined extension to the Device ID. This field will match IPBRR0[MAJOR] setting.

53.4.6.1.6 PCI cache line size register (PCI_CFH_CL_SIZE)

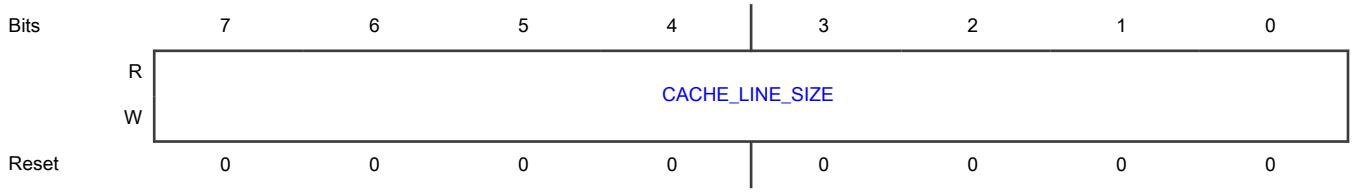
Offset

Register	Offset
PCI_CFH_CL_SIZE	Ch

Function

This is the PCI config header cache line size register.

Diagram



Fields

Field	Function
7-0 CACHE_LINE_SIZE	Cache line size The Cache Line Size register is set by the system firmware or the operating system to system cache line size. This field is implemented by PCI Express devices as a read/write field for legacy compatibility purposes but has no effect on any PCI Express device behavior.

53.4.6.1.7 PCI header type register (PCI_CFH_HDR_TYPE)

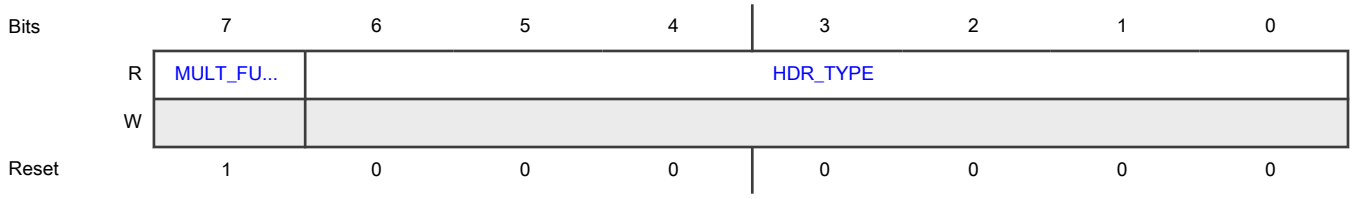
Offset

Register	Offset
PCI_CFH_HDR_TYPE	Eh

Function

This is the PCI config header header type register.

Diagram



Fields

Field	Function
7 MULT_FUNC_DEV	Multi-function device When set, indicates that the device may contain multiple functions, but not necessarily. Software is permitted to probe for functions other than function 0. When clear, software must not probe for functions other than function 0 unless explicitly indicated by another mechanism, such as an ARI or SR-IOV Capability structure. Except where stated otherwise, it is recommended that this bit be set if there are multiple functions, and clear if there is only one function.
6-0	Header type

Table continues on the next page...

Table continued from the previous page...

Field	Function
HDR_TYPE	This field identifies the layout of the second part of the predefined header (beginning at byte 10h in Configuration Space). This register is hardwired to 00h (Type 0 header for both PFs and VFs).

53.4.6.1.8 PCI base address register 0 (PCI_CFH_BAR0)

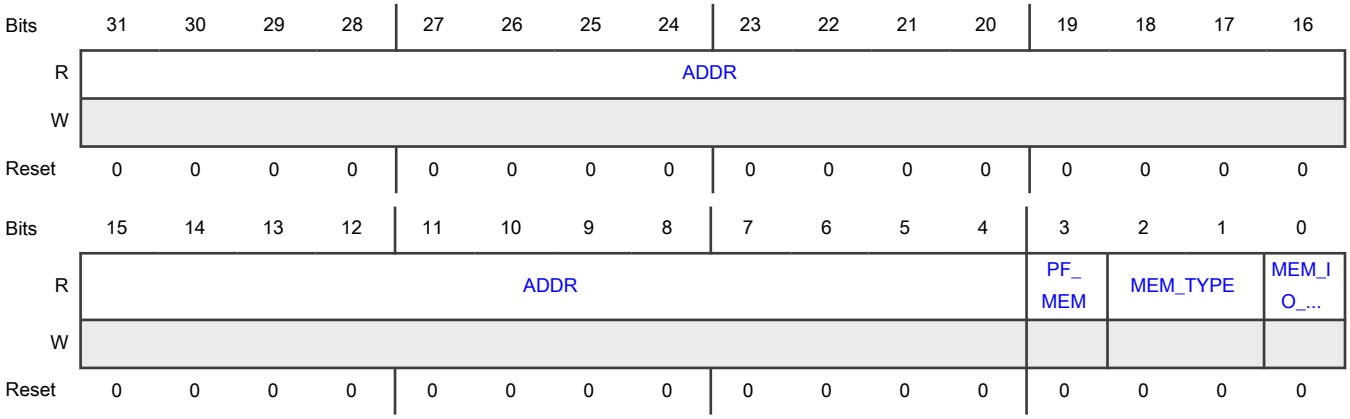
Offset

Register	Offset
PCI_CFH_BAR0	10h

Function

This is the PCI config header base address register 0.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.1.9 PCI base address register 1 (PCI_CFH_BAR1)

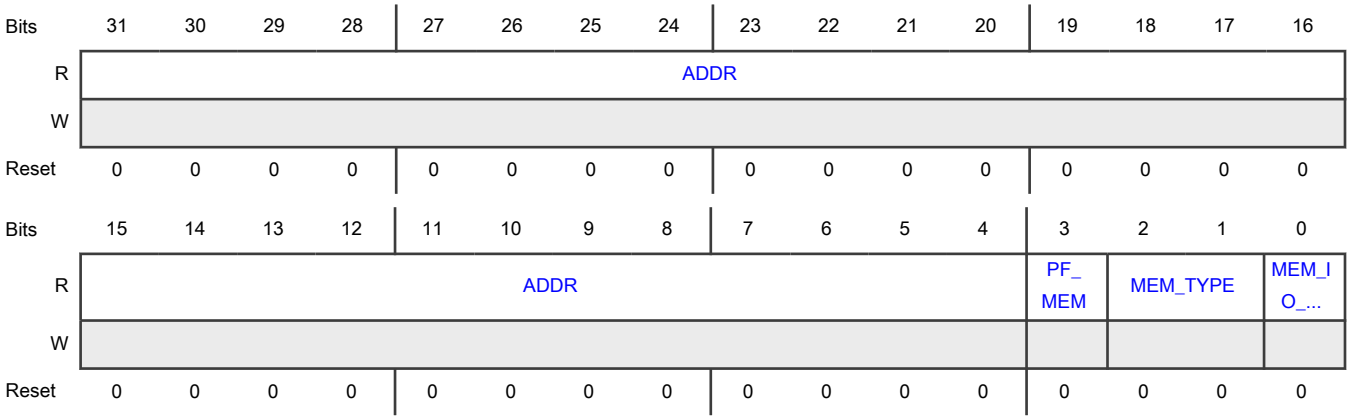
Offset

Register	Offset
PCI_CFH_BAR1	14h

Function

This is the PCI config header base address register 1.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.1.10 PCI base address register 2 (PCI_CFH_BAR2)

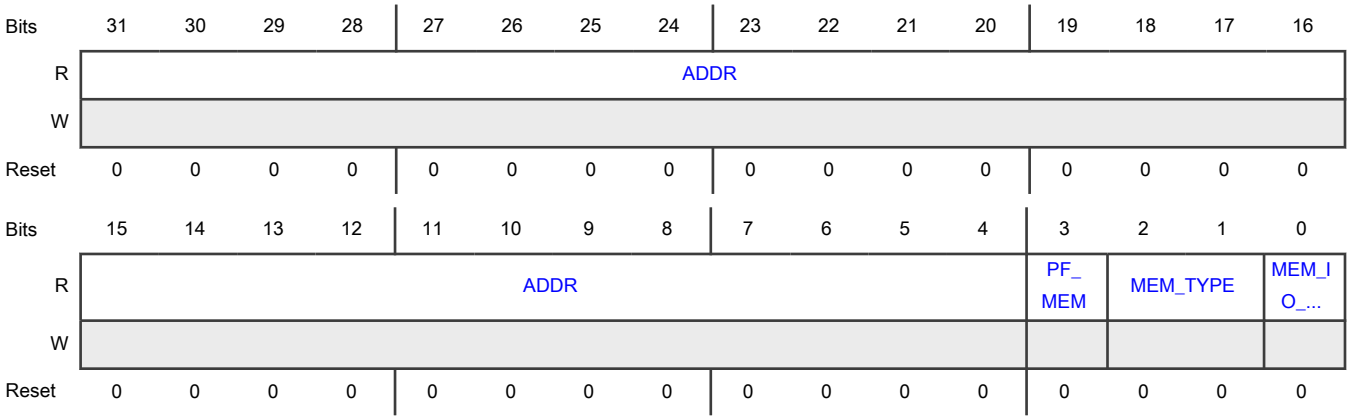
Offset

Register	Offset
PCI_CFH_BAR2	18h

Function

This is the PCI config header base address register 2.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.1.11 PCI base address register 3 (PCI_CFH_BAR3)

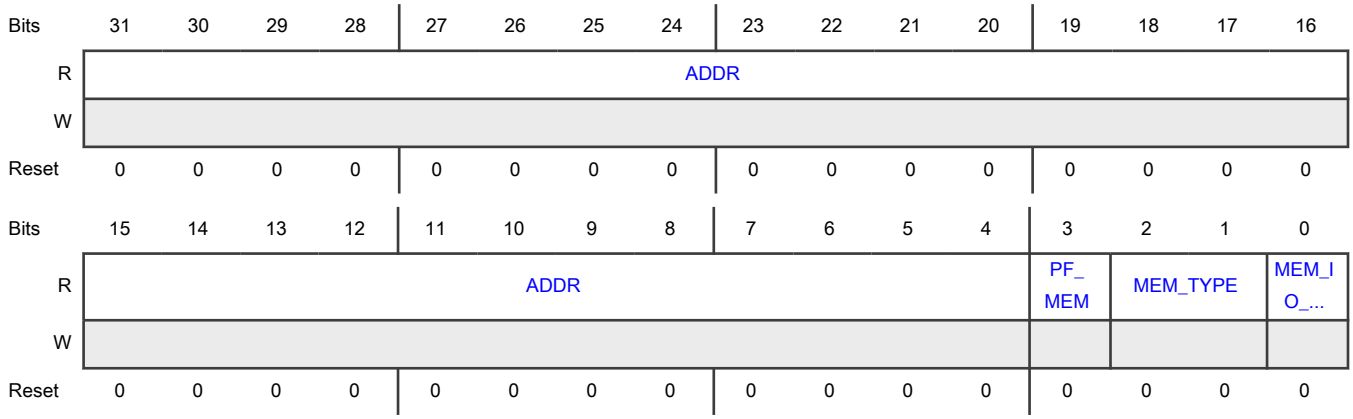
Offset

Register	Offset
PCI_CFH_BAR3	1Ch

Function

This is the PCI config header base address register 3.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.1.12 PCI base address register 4 (PCI_CFH_BAR4)

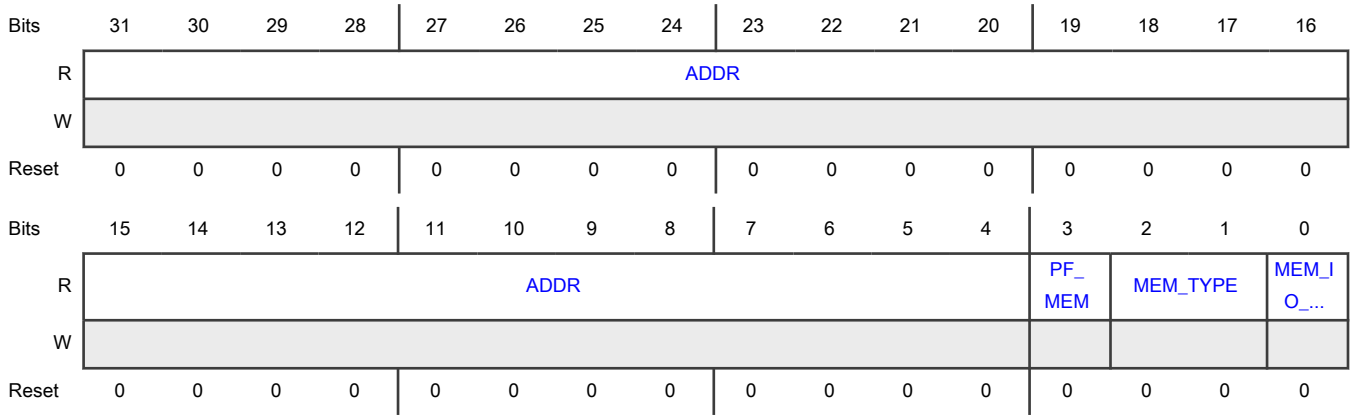
Offset

Register	Offset
PCI_CFH_BAR4	20h

Function

This is the PCI config header base address register 4.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.1.13 PCI base address register 5 (PCI_CFH_BAR5)

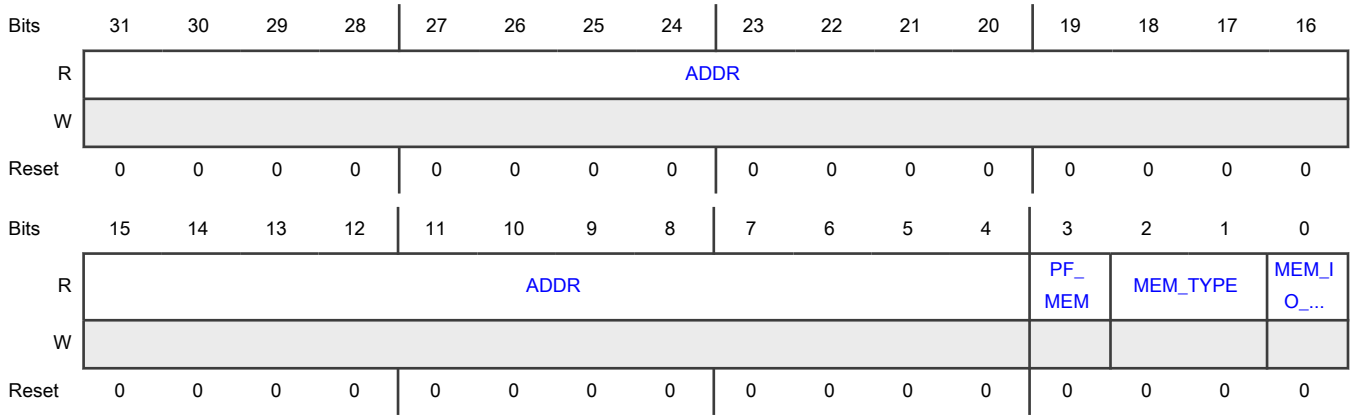
Offset

Register	Offset
PCI_CFH_BAR5	24h

Function

This is the PCI config header base address register 5.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.1.14 PCI subsystem vendor ID register (PCI_CFH_SUBSYS_VID)

Offset

Register	Offset
PCI_CFH_SUBSYS_VID	2Ch

Function

This is the PCI config header subsystem vendor ID register. The value read is taken from IERB register <FUNC>_CFH_SIDSVID.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SUBSYSTEM_VENDOR_ID															
W																
Reset	0	0	0	1	1	0	0	1	0	1	0	1	0	1	1	1

Fields

Field	Function
15-0 SUBSYSTEM_VENDOR_ID	This Read Only field identifies the manufacturer of the subsystem. The field in the PF and associated VFs must return the same value when read.

53.4.6.1.15 PCI subsystem ID register (PCI_CFH_SUBSYS_ID)

Offset

Register	Offset
PCI_CFH_SUBSYS_ID	2Eh

Function

This is the PCI config header subsystem ID register. The value read is taken from IERB register <FUNC>_IEP_CFH_SIDSVID.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SUBSYSTEM_ID															
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PCI_CFH_SUBSYS_ID	NETC_F0_PCI_HDR_TYPE0: EE02h NETC_F1_PCI_HDR_TYPE0: EE00h NETC_F2_PCI_HDR_TYPE0: EEF2h NETC_F3_PCI_HDR_TYPE0: E101h NETC_F4_PCI_HDR_TYPE0: E110h

Fields

Field	Function
15-0 SUBSYSTEM_ID	This Read Only field identifies the particular subsystem. The Device may have a different value in the PF and the VF.

53.4.6.1.16 PCI capabilities pointer register (PCI_CFH_CAP_PTR)

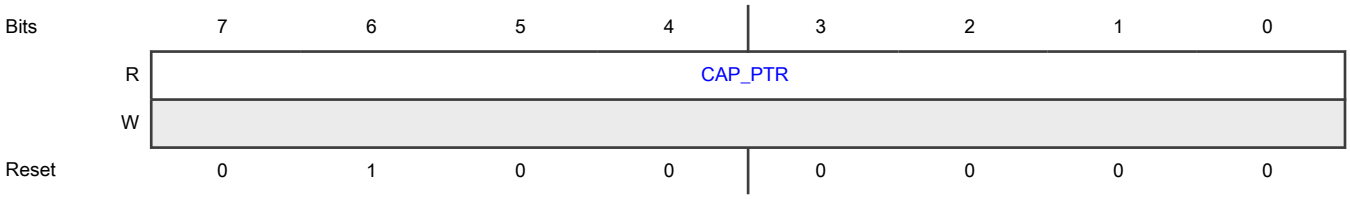
Offset

Register	Offset
PCI_CFH_CAP_PTR	34h

Function

This is the PCI config header capabilities pointer register.

Diagram



Fields

Field	Function
7-0 CAP_PTR	This register is used to point to a linked list of new capabilities implemented by this device. This register is only valid if the “Capabilities List” bit in the Status Register is set. If implemented, the bottom two bits are reserved and should be set to 00b. Software should mask these bits off before using this register as a pointer in Configuration Space to the first entry of a linked list of new capabilities. Hardwired to 40h.

53.4.6.1.17 PCI PCIe capabilities list register (PCI_CFC_PCIE_CAP_LIST)

Offset

Register	Offset
PCI_CFC_PCIE_CAP_LIST	40h

Function

This is the PCI config capability PCIe capabilities list register.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NEXT_CAP_PTR								CAP_ID							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
15-8 NEXT_CAP_PTR	This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of capabilities. Hardwired to 80h.
7-0 CAP_ID	Indicates the PCI Express Capability structure. Hardwired to 10h.

53.4.6.1.18 PCI PCIe capabilities register (PCI_CFC_PCIE_CAP)

Offset

Register	Offset
PCI_CFC_PCIE_CAP	42h

Function

This is the PCI config capability PCIe capabilities register.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		INT_MSG_NUM					Reserved	DEV_PORT_TYPE				CAP_VER			
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0

Fields

Field	Function
15-14 —	Reserved
13-9	Interrupt message number

Table continues on the next page...

Table continued from the previous page...

Field	Function
INT_MSG_NUM	This field indicates which MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of this capability structure. For MSI-X, the value in this field indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. If both MSI and MSI-X are implemented, they are permitted to use different vectors, though software is permitted to enable only one mechanism at a time. If MSI-X is enabled, the value in this field must indicate the vector for MSI-X. If MSI is enabled or neither is enabled, the value in this field must indicate the vector for MSI. If software enables both MSI and MSI-X at the same time, the value in this field is undefined. Errors for ENETC are handled through the Event Collector. Hardwired to 00h.
8 —	Reserved
7-4 DEV_PORT_TYPE	Device/Port type Indicates the specific type of this PCI Express Function. Hardwired to 9h to indicate Root Complex Integrated Endpoint.
3-0 CAP_VER	Capability Version Indicates PCI-SIG defined PCI Express Capability structure version number. Hardwired to 2h.

53.4.6.1.19 PCI PCIe device capabilities register (PCI_CFC_PCIE_DEV_CAP)

Offset

Register	Offset
PCI_CFC_PCIE_DEV_CAP	44h

Function

This is the PCI config capability PCIe device capabilities register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			FLR_CAP	Reserved											
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 FLR_CAP	Function level reset capability A value of 1b indicates the function supports the optional Function Level Reset (FLR) mechanism. Hardwired to 1b.
27-0 —	Reserved

53.4.6.1.20 PCI PCIe device control register (PCI_CFC_PCIE_DEV_CTL)

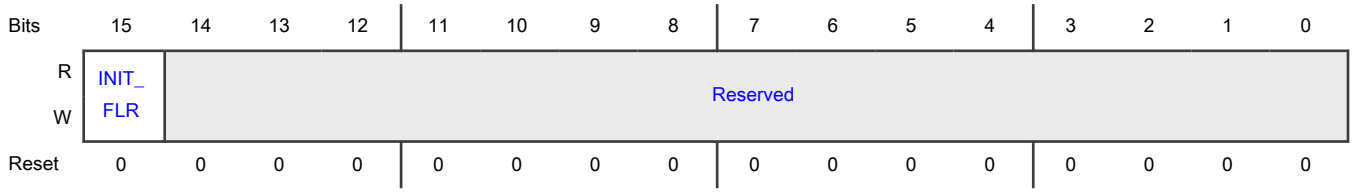
Offset

Register	Offset
PCI_CFC_PCIE_DEV_CTL	48h

Function

This is the PCI config capability PCIe device control register.

Diagram



Fields

Field	Function
15 INIT_FLR	Initiate function level reset A write of b1 initiates Function Level Reset (FLR). The bit will self-clear when the reset sequence is complete and function can be re-configured by software.
14-0 —	Reserved

53.4.6.1.21 PCI PCIe device status register (PCI_CFC_PCIE_DEV_STAT)

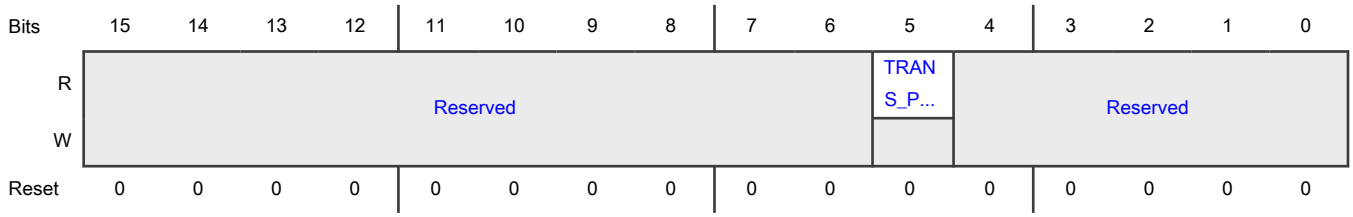
Offset

Register	Offset
PCI_CFC_PCIE_DEV_STAT	4Ah

Function

This is the PCI config capability PCIe device status register.

Diagram



Fields

Field	Function
15-6 —	Reserved
5	Transaction pending

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRANS_PEND	If set indicates that the Function has outstanding transactions on its external master interface.
4-0 —	Reserved

53.4.6.1.22 PCI PCIe device capabilities 2 register (PCI_CFC_PCIE_DEV_CAP2)

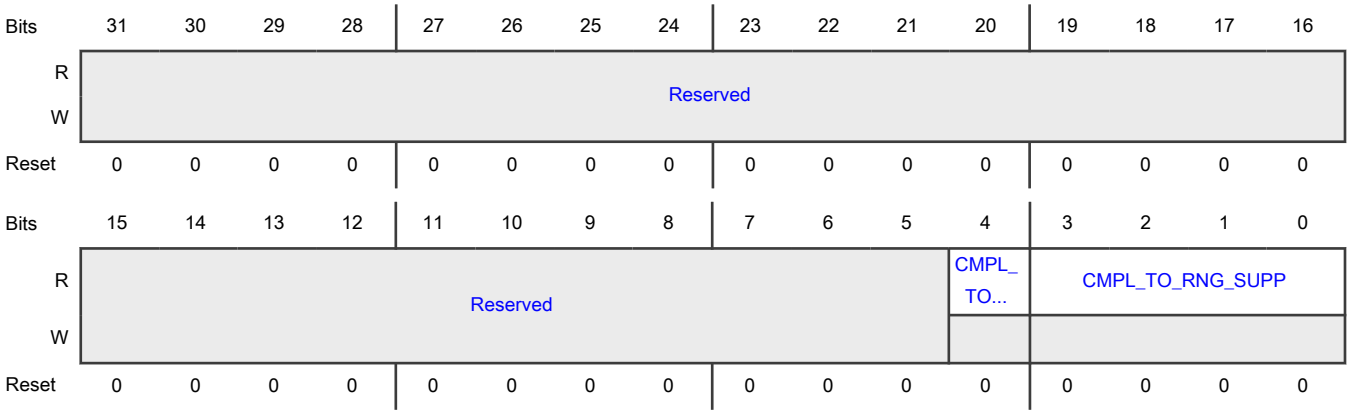
Offset

Register	Offset
PCI_CFC_PCIE_DEV_CAP2	64h

Function

This is the PCI config capability PCIe device capabilities 2 register.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 CMPL_TO_DIS_SUPP	Completion Timeout Disable Supported Not supported. Hardwired to b0.
3-0	Completion Timeout Ranges Supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
CMPL_TO_RNG_SUPP	Completion Timeout programming not supported, the Function assumes a timeout value in the range 50 us to 50 ms.

53.4.6.1.23 PCI PCIe device control 2 register (PCI_CFC_PCIE_DEV_CTL2)

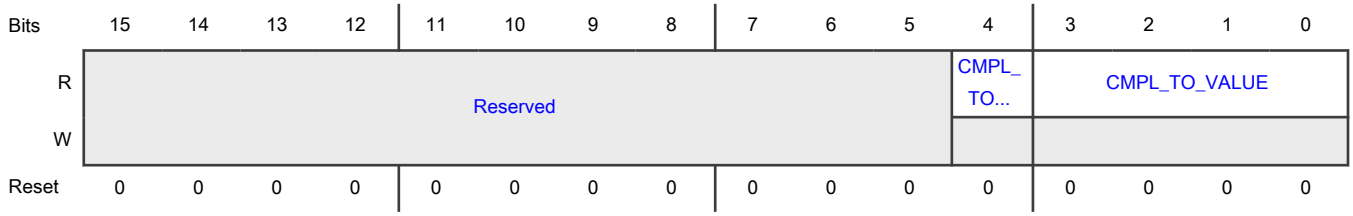
Offset

Register	Offset
PCI_CFC_PCIE_DEV_CTL2	68h

Function

This is the PCI config capability PCIe device control 2 register.

Diagram



Fields

Field	Function
15-5 —	Reserved
4 CMPL_TO_EN	Completion Timeout Enable Not supported. Hardwired to b0.
3-0 CMPL_TO_VALUE	Completion Timeout Value Completion Timeout programming not supported – the Function assumes a timeout value in the range 50 us to 50 ms. Hardwired to b0000.

53.4.6.1.24 PCI MSI-X capabilities list register (PCI_CFC_MSIX_CAP_LIST)

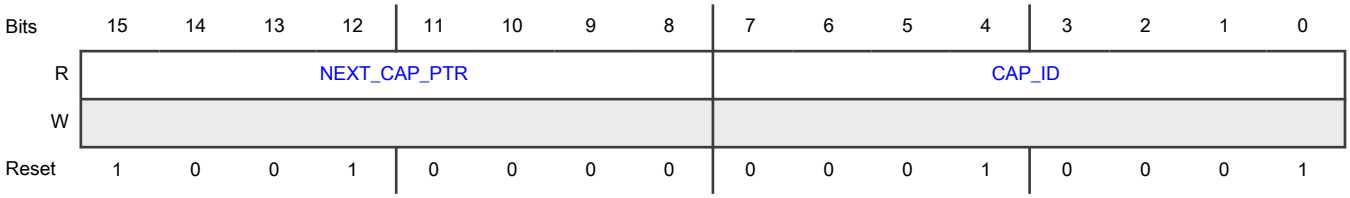
Offset

Register	Offset
PCI_CFC_MSIX_CAP_LIST	80h

Function

This is the PCI config capability MSI-X capabilities list register.

Diagram



Fields

Field	Function
15-8 NEXT_CAP_PTR	This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of capabilities. Hardwired to 90h.
7-0 CAP_ID	Indicates the MSI-X Capability structure. Hardwired to 11h.

53.4.6.1.25 PCI MSI-X message control register (PCI_CFC_MSIX_MSG_CTL)

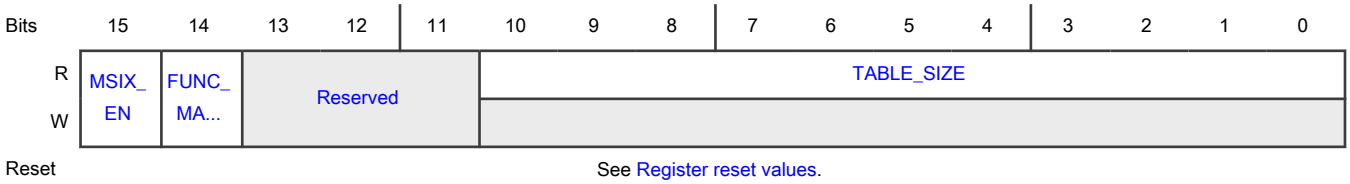
Offset

Register	Offset
PCI_CFC_MSIX_MSG_CTL	82h

Function

This is the PCI config capability MSI-X message control register.

Diagram



Register reset values

Register	Reset value
PCI_CFC_MSIX_MSG_CTL	NETC_F0_PCI_HDR_TYPE0,NETC_F1_PCI_HDR_TYPE0: 0000h NETC_F2_PCI_HDR_TYPE0: 0005h NETC_F3_PCI_HDR_TYPE0: 000Bh NETC_F4_PCI_HDR_TYPE0: 0017h

Fields

Field	Function
15 MSIX_EN	MSI-X enable If set, the function is permitted to use MSI-X to request service. System configuration software sets this bit to enable MSI-X. A device driver is prohibited from writing this bit to mask a function's service request. If 0, the function is prohibited from using MSI-X to request service. This bit's state after reset is 0 (MSI-X is disabled).
14 FUNC_MASK	Function mask If 1, all of the vectors associated with the function are masked, regardless of their per-vector Mask bit states. If 0, each vector's Mask bit determines whether the vector is masked or not. Setting or clearing the MSI-X Function Mask bit has no effect on the state of the per-vector Mask bits. This bit's state after reset is 0 (unmasked).
13-11 —	Reserved
10-0 TABLE_SIZE	Table size System software reads this field to determine the MSI-X Table Size N, which is encoded as N-1. For example, a returned value of "00000000011" indicates a table size of 4. **NOTE** ENETC function table size is defined by PSI0CFGR2[NUM_MSIX] and is immediately reflected here. All other functions use IERB register *MCR[NUM_MSIX].

53.4.6.1.26 PCI MSI-X table offset/BIR register (PCI_CFC_MSIX_TABLE_OFF_BIR)

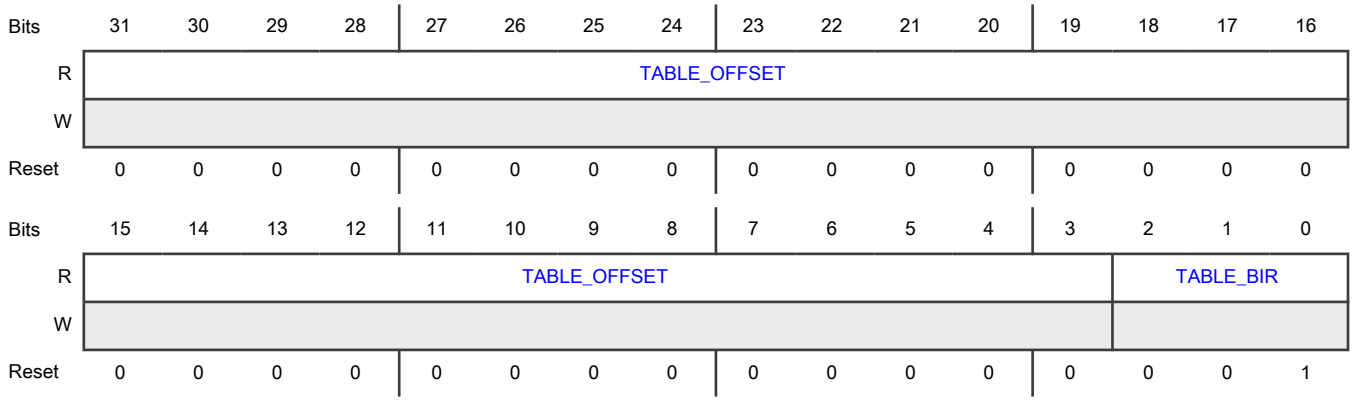
Offset

Register	Offset
PCI_CFC_MSIX_TABLE_OFF_BIR	84h

Function

This is the PCI config capability MSI-X table offset/BIR register.

Diagram



Fields

Field	Function
31-3 TABLE_OFFSET	Table Offset Used as an offset from the address contained by one of the function's Base Address registers to point to the base of the MSI-X Table. The lower 3 Table BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset.
2-0 TABLE_BIR	Table BIR Indicates which entry in the Enhanced Allocation capability with a matching BEI, is used to map the Function's MSI-X Table into Memory Space. For a 64-bit Base Address register, the Table BIR indicates the lower DWORD. Physical functions hardwires this field to 001b (PE entry 1) for MSI-X table.

53.4.6.1.27 PCI MSI-X PBA offset/BIR register (PCI_CFC_MSIX_PBA_OFF_BIR)

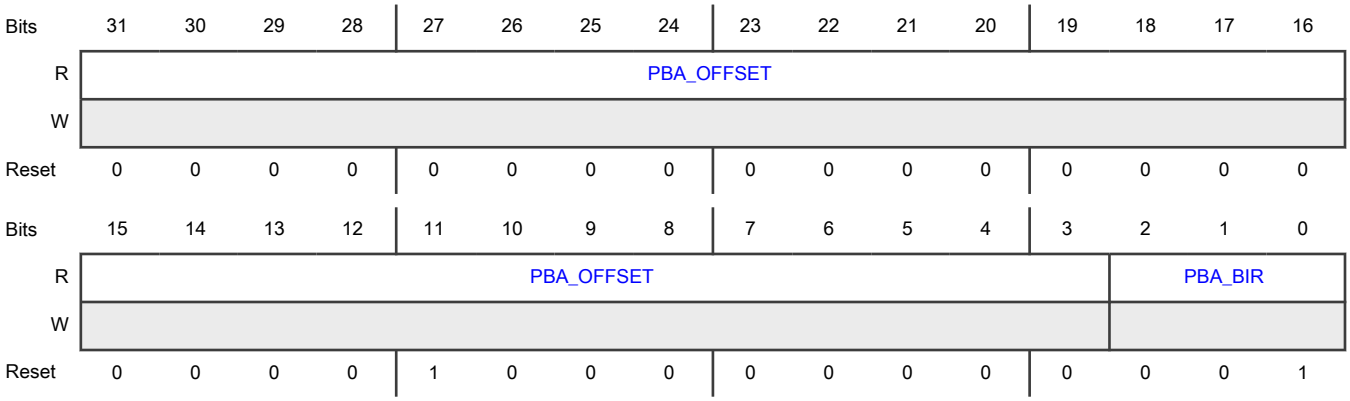
Offset

Register	Offset
PCI_CFC_MSIX_PBA_OFF_BIR	88h

Function

This is the PCI config capability MSI-X PBA offset/BIR register.

Diagram



Fields

Field	Function
31-3 PBA_OFFSET	PBA Offset Used as an offset from the address contained by one of the function's Base Address registers to point to the base of the MSI-X PBA. The lower 3 PBA BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset. The PBA structure directly follows the maximum possible MSI-X table size (2KB).
2-0 PBA_BIR	PBA BIR Indicates which entry in the Enhanced Allocation capability with a matching BEI, is used to map the Function's MSI-X PBA into Memory Space. For a 64-bit Base Address Register, the PBA BIR indicates the lower DWORD. Physical function hardwires this field to 001b (PE entry 1) for MSI-X PBA.

53.4.6.1.28 PCI PCI-PM capabilities list register (PCI_CFC_PCIPM_CAP_LIST)

Offset

Register	Offset
PCI_CFC_PCIPM_CAP_LIST	90h

Function

This is the PCI config capability PCI-PM capabilities list register.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NEXT_CAP_PTR								CAP_ID							
W																
Reset	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
15-8 NEXT_CAP_PTR	This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of capabilities. Hardwired to 9Ch.
7-0 CAP_ID	Indicates the PCI-PM Capability structure. Hardwired to 01h.

53.4.6.1.29 PCI PCI-PM capabilities register (PCI_CFC_PCIPM_CAP)

Offset

Register	Offset
PCI_CFC_PCIPM_CAP	92h

Function

This is the PCI config capability PCI-PM capabilities register.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PME_SUPPORT				Reserved									VERSION		
W																
Reset	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
15-11 PME_SUPPORT	PME support Indicates the PM states within which the function is capable of sending PME message. 0 in a bit indicates PME notification is not supported in the respective PM state. Bit State

Table continues on the next page...

Table continued from the previous page...

Field	Function
	15 $D3_{cold}$ 14 $D3_{hot}$ 13 D2 12 D1 11 D0 ENETC only supports generating PM_PME notifications from D0 and D3hot states.
10-3 —	Reserved
2-0 VERSION	Version ENETC complies with the PCI PM specification, rev 1.2.

53.4.6.1.30 PCI PCI-PM control and status register (PCI_CFC_PCIPM_CTL_STAT)

Offset

Register	Offset
PCI_CFC_PCIPM_CTL_STAT	94h

Function

This is the PCI config capability PCI-PM control and status register.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PME_STA...	Reserved						PME_EN	Reserved				NO_SOFT...	Reserved	PWR_STATE	
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
15 PME_STATUS	PME status This bit is set to 1b when the function detects a Wake-on-LAN event independent of the state of the PME_EN bit. Write 1b to clear.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>NETC_F0_PCI_HDR_TYPE0</td><td>—</td><td>PCI_CFC_PCIPM_CTL_STAT</td></tr>
<tr><td>NETC_F1_PCI_HDR_TYPE0</td><td>—</td><td>PCI_CFC_PCIPM_CTL_STAT</td></tr>
<tr><td>NETC_F2_PCI_HDR_TYPE0</td><td>—</td><td>PCI_CFC_PCIPM_CTL_STAT</td></tr>
<tr><td>NETC_F3_PCI_HDR_TYPE0</td><td>PCI_CFC_PCIPM_CTL_STAT</td><td>—</td></tr>
<tr><td>NETC_F4_PCI_HDR_TYPE0</td><td>—</td><td>PCI_CFC_PCIPM_CTL_STAT</td></tr>
</table>
</td></tr>
<tr><td>14-9
—</td><td>Reserved</td></tr>
<tr><td>8
PME_EN</td><td>PME enable
Enable function to generate PM_PME messages to Root Complex Event Collector.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>NETC_F0_PCI_HDR_TYPE0</td><td>—</td><td>PCI_CFC_PCIPM_CTL_STAT</td></tr>
<tr><td>NETC_F1_PCI_HDR_TYPE0</td><td>—</td><td>PCI_CFC_PCIPM_CTL_STAT</td></tr>
<tr><td>NETC_F2_PCI_HDR_TYPE0</td><td>—</td><td>PCI_CFC_PCIPM_CTL_STAT</td></tr>
<tr><td>NETC_F3_PCI_HDR_TYPE0</td><td>PCI_CFC_PCIPM_CTL_STAT</td><td>—</td></tr>
<tr><td>NETC_F4_PCI_HDR_TYPE0</td><td>—</td><td>PCI_CFC_PCIPM_CTL_STAT</td></tr>
</table>
</td></tr>
<tr><td>7-4
—</td><td>Reserved</td></tr>
<tr><td>3
NO_SOFT_RST</td><td>No soft reset</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When set (“1”), this bit indicates that when NETC transitions from $D3_{hot}$ to $D0_{active}$ because of modifying Power State bits in the PCI_CFC_PCIPM_CTL_STAT register, no internal reset is issued and Configuration Context is preserved. Upon transition from the $D3_{hot}$ to the $D0_{active}$ state, no additional operating system intervention is required to preserve PCIe Configuration Context beyond writing the Power State bits. When clear (“0”), NETC performs an internal reset upon transitioning from $D3_{hot}$ to $D0_{uninitialized}$ via software control of the Power State bits in the PCI_CFC_PCIPM_CTL_STAT register. Configuration Context is lost when performing the soft reset. Upon transition from the $D3_{hot}$ to the $D0_{uninitialized}$ state, full re-initialization sequence is needed to return the device to $D0_{active}$. Regardless of this bit, NETC preserves the PME context when PME is enabled.
2 —	Reserved
1-0 PWR_STATE	Power state This field is used to set and report the power state of a function as follows: 00b = D0 01b = D1 (not supported by NETC) 10b = D2 (not supported by NETC) 11b = D3 If, for any reason, the operating system software attempts to put a function into a power management state that the function does not support, the function should handle this gracefully and remain in whatever state it was in before the request. The PWR_STATE bits will reflect the current state of the function not the intended invalid state which was written.

53.4.6.1.31 PCI PCI-PM capabilities data register (PCI_CFC_PCIPM_DATA)

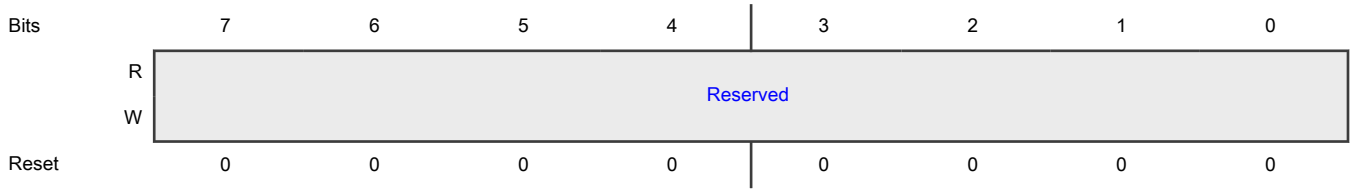
Offset

Register	Offset
PCI_CFC_PCIPM_DATA	97h

Function

This is the PCI config capability PCI-PM data register. Not supported by ENETC since AUX power not supported in $D3_{cold}$.

Diagram



Fields

Field	Function
7-0 —	Reserved

53.4.6.1.32 PCI EA capabilities list register (PCI_CFC_EA_CAP_LIST)

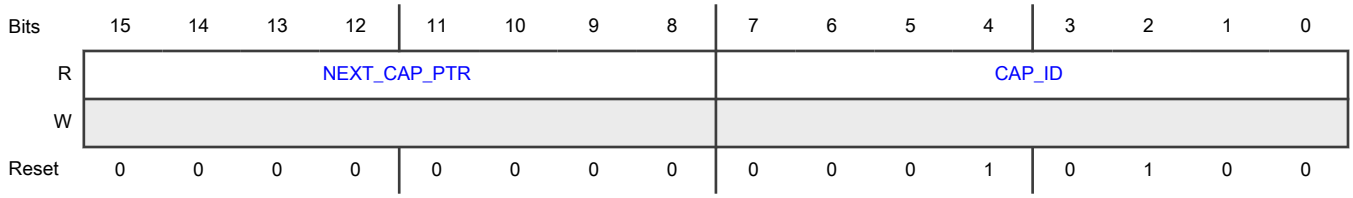
Offset

Register	Offset
PCI_CFC_EA_CAP_LIST	9Ch

Function

This is the PCI config capability enhanced allocation capabilities list register.

Diagram



Fields

Field	Function
15-8 NEXT_CAP_PTR	This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of capabilities. Hardwired to 00h.
7-0 CAP_ID	Indicates the Enhanced Allocation (EA) Capability structure. Hardwired to 14h.

53.4.6.1.33 PCI EA capabilities register (PCI_CFC_EA_CAP)

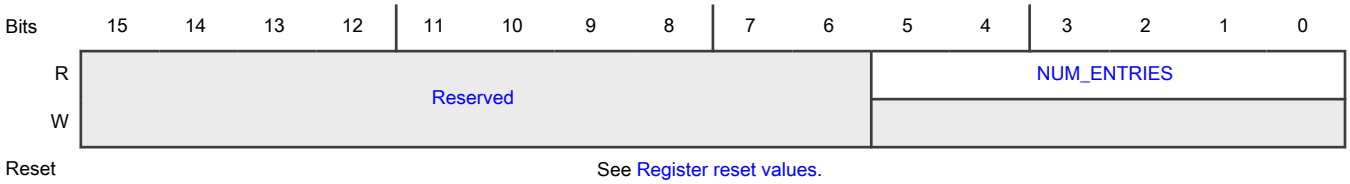
Offset

Register	Offset
PCI_CFC_EA_CAP	9Eh

Function

This is the PCI config capability enhanced allocation capabilities register.

Diagram



Register reset values

Register	Reset value
PCI_CFC_EA_CAP	NETC_F0_PCI_HDR_TYPE0–NETC_F3_PCI_HDR_TYPE0: 0002h NETC_F4_PCI_HDR_TYPE0: 0004h

Fields

Field	Function
15-6 —	Reserved
5-0 NUM_ENTRIES	Number of entries Number of entries following the first DW of the capability.

53.4.6.1.34 PCI EA per-entry 0 format register (PCI_CFC_EA_PE0_FMT)

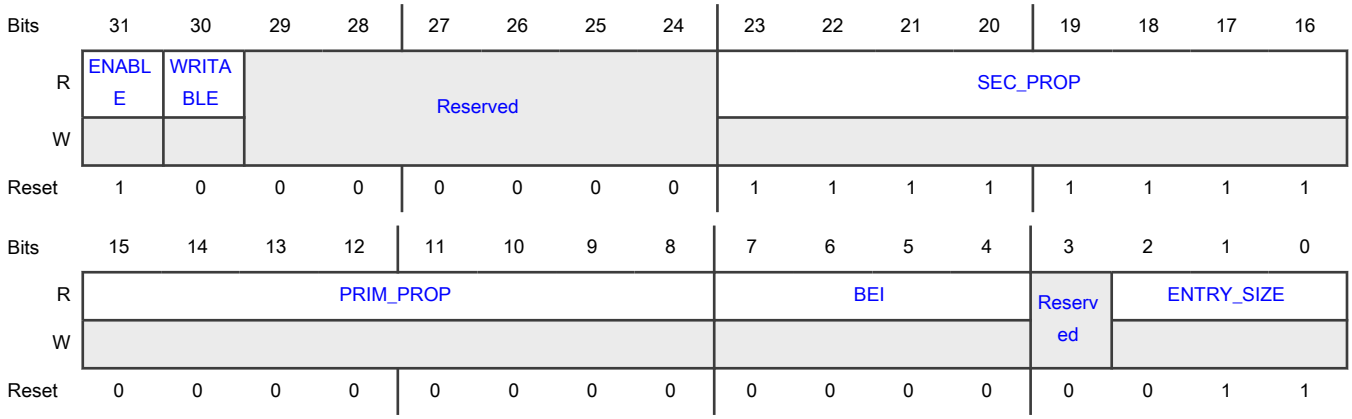
Offset

Register	Offset
PCI_CFC_EA_PE0_FMT	A0h

Function

This is the PCI config capability enhanced allocation per-entry format register.

Diagram



Fields

Field	Function
31 ENABLE	Enable 0 Disabled 1 Enabled
30 WRITABLE	Writable 1b indicates that the Base, MaxOffset and Field Size fields for this entry are RW, 0b indicates those fields are HWInit. Hardwired to 0b.
29-24 —	Reserved
23-16 SEC_PROP	Secondary properties Alternative property when system cannot use the primary property.
15-8 PRIM_PROP	Primary properties Value (h) Resource Definition 00 Memory Space, Non-Prefetchable. The corresponding Base and MaxOffset fields are HwInit. 01 Memory Space, Prefetchable. The corresponding Base and MaxOffset fields are HwInit. 03 For use only by Physical Functions to indicate resources for Virtual Function use, Memory Space, Prefetchable. The corresponding Base and MaxOffset fields are HwInit. 04 For use only by Physical Functions to indicate resources for Virtual Function use, Memory Space, Non-Prefetchable. The corresponding Base and MaxOffset fields are HwInit. FF Entry unavailable for use. For other settings see PCIe specification.
7-4 BEI	BAR equivalent indicator This field indicates the equivalent PCI BAR for this entry. For 64-bit BARs, the BEI value indicates the equivalent PCI BAR location for lower DWORD.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	A Type 0 function is permitted to use EA to allocate resources for itself, and such resources must indicate a BEI value of 0-5, 7 or 8. Physical functions are allowed to use EA to allocate resources for its associated Virtual Functions, and such resources must indicate a BEI value of 9-14.
3 —	Reserved
2-0 ENTRY_SIZE	Entry size Number of DWs following the initial DW in this entry. Hardwired to 011b for 64b base and 32b MaxOffset.

53.4.6.1.35 PCI EA per-entry 0 base register (PCI_CFC_EA_PE0_BASE)

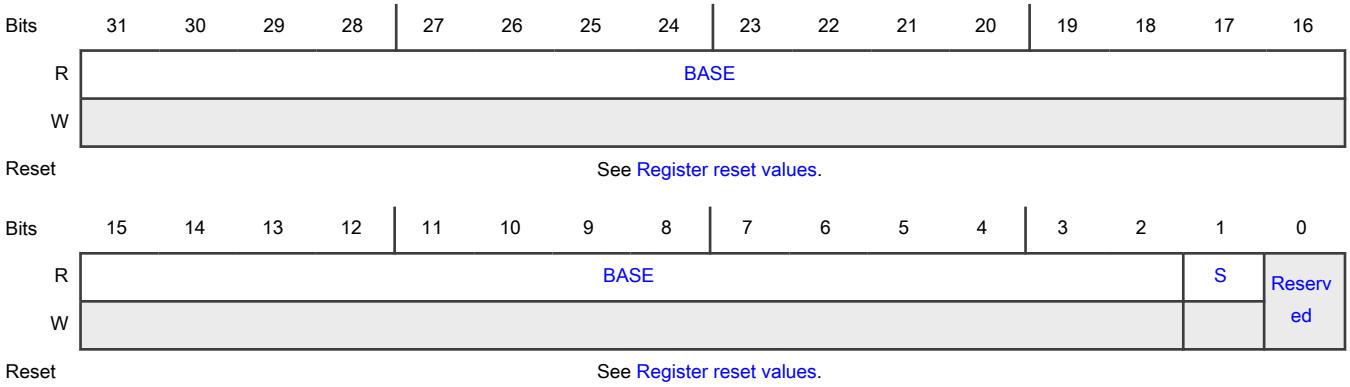
Offset

Register	Offset
PCI_CFC_EA_PE0_BASE	A4h

Function

This is the PCI config capability enhanced allocation per-entry base register.

Diagram



Register reset values

Register	Reset value
PCI_CFC_EA_PE0_BASE	NETC_F0_PCI_HDR_TYPE0: 60B8_0002h NETC_F1_PCI_HDR_TYPE0: 60BA_0002h NETC_F2_PCI_HDR_TYPE0: 60A0_0002h NETC_F3_PCI_HDR_TYPE0: 60B0_0002h NETC_F4_PCI_HDR_TYPE0: 60B4_0002h

Fields

Field	Function
31-2 BASE	Base DW address of the start of the resource range. The value in the Base field ([63:2]) plus the value in the MaxOffset field indicates the address of the last included DW of the resource range. All addresses are naturally aligned for NETC
1 S	Field size 0 32-bit 1 64-bit
0 —	Reserved

53.4.6.1.36 PCI EA per-entry 0 max offset register (PCI_CFC_EA_PE0_MAXOFF)

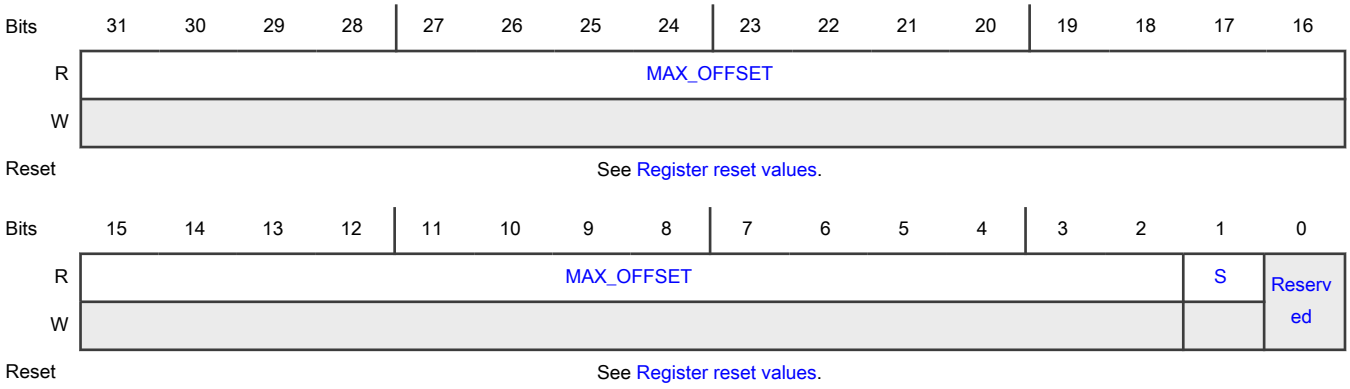
Offset

Register	Offset
PCI_CFC_EA_PE0_MAXOFF	A8h

Function

This is the PCI config capability enhanced allocation per-entry max offset register.

Diagram



Register reset values

Register	Reset value
PCI_CFC_EA_PE0_MAXOFF	NETC_F0_PCI_HDR_TYPE0,NETC_F1_PCI_HDR_TYPE0: 0001_FFFCh NETC_F2_PCI_HDR_TYPE0: 000F_FFFCh NETC_F3_PCI_HDR_TYPE0,NETC_F4_PCI_HDR_TYPE0: 0003_FFFCh

Fields

Field	Function
31-2 MAX_OFFSET	Max offset The value in the Base field ([63:2]) plus the value in the MaxOffset field indicates the address of the last included DW of the resource range. Bits [1:0] of the MaxOffset are not included in the MaxOffset field, and must always be interpreted as 11b. **NOTE** For SR-IOV, this value is the size and alignment for a single VF. Multiply the aperture size with the value set in NumVFs to determine the total amount of space the BAR or BAR pair will map after VF Enable and VF MSE are Set.
1 S	Field size 0 32-bit 1 64-bit
0 —	Reserved

53.4.6.1.37 PCI EA per-entry a extended base register (PCI_CFC_EA_PE0_EXT_BASE - PCI_CFC_EA_PE3_EXT_BASE)

Offset

Register	Offset
PCI_CFC_EA_PE0_EXT_BASE	ACh
PCI_CFC_EA_PE1_EXT_BASE	BCh
PCI_CFC_EA_PE2_EXT_BASE	CCh
PCI_CFC_EA_PE3_EXT_BASE	DCh

Function

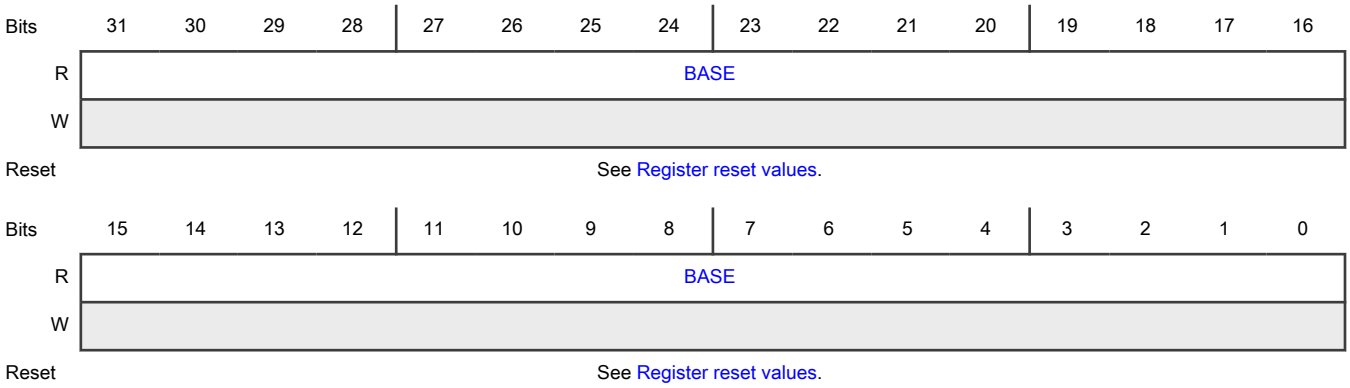
This is the PCI config capability enhanced allocation per-entry extended base register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	PCI_CFC_EA_PE0_EXT_BASE–PCI_CFC_EA_PE1_EXT_BASE	PCI_CFC_EA_PE2_EXT_BASE–PCI_CFC_EA_PE3_EXT_BASE
NETC_F1_PCI_HDR_TYPE0	PCI_CFC_EA_PE0_EXT_BASE–PCI_CFC_EA_PE1_EXT_BASE	PCI_CFC_EA_PE2_EXT_BASE–PCI_CFC_EA_PE3_EXT_BASE
NETC_F2_PCI_HDR_TYPE0	PCI_CFC_EA_PE0_EXT_BASE–PCI_CFC_EA_PE1_EXT_BASE	PCI_CFC_EA_PE2_EXT_BASE–PCI_CFC_EA_PE3_EXT_BASE
NETC_F3_PCI_HDR_TYPE0	PCI_CFC_EA_PE0_EXT_BASE–PCI_CFC_EA_PE1_EXT_BASE	PCI_CFC_EA_PE2_EXT_BASE–PCI_CFC_EA_PE3_EXT_BASE
NETC_F4_PCI_HDR_TYPE0	PCI_CFC_EA_PE0_EXT_BASE–PCI_CFC_EA_PE3_EXT_BASE	—

Diagram



Register reset values

Register	Reset value
PCI_CFC_EA_PE0_EXT_BASE–PCI_CFC_EA_PE1_EXT_BASE	NETC_F0_PCI_HDR_TYPE0–NETC_F4_PCI_HDR_TYPE0: 0000_0000h
PCI_CFC_EA_PE2_EXT_BASE–PCI_CFC_EA_PE3_EXT_BASE	0000_0000h

Fields

Field	Function
31-0 BASE	Base DW high address of the start of the resource range. The value in the Base field ([63:2]) plus the value in the MaxOffset field indicates the address of the last included DW of the resource range. All addresses are naturally aligned for NETC

53.4.6.1.38 PCI EA per-entry 1 format register (PCI_CFC_EA_PE1_FMT)

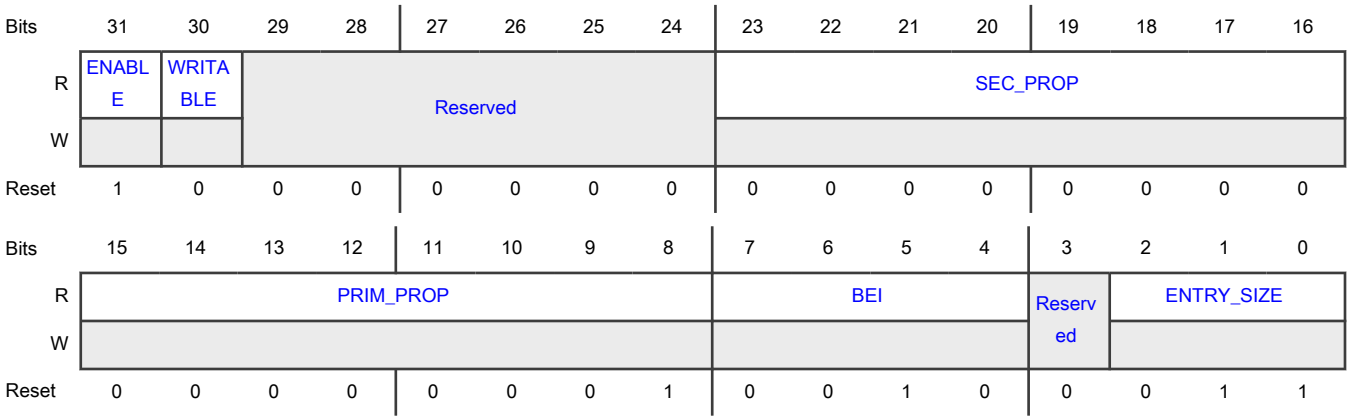
Offset

Register	Offset
PCI_CFC_EA_PE1_FMT	B0h

Function

This is the PCI config capability enhanced allocation per-entry format register.

Diagram



Fields

Field	Function
31 ENABLE	Enable 0 Disabled 1 Enabled
30 WRITABLE	Writable 1b indicates that the Base, MaxOffset and Field Size fields for this entry are RW, 0b indicates those fields are HWInit. Hardwired to 0b.

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-24 —	Reserved
23-16 SEC_PROP	Secondary properties Alternative property when system cannot use the primary property.
15-8 PRIM_PROP	Primary properties Value (h) Resource Definition 00 Memory Space, Non-Prefetchable. The corresponding Base and MaxOffset fields are HwInit. 01 Memory Space, Prefetchable. The corresponding Base and MaxOffset fields are HwInit. 03 For use only by Physical Functions to indicate resources for Virtual Function use, Memory Space, Prefetchable. The corresponding Base and MaxOffset fields are HwInit. 04 For use only by Physical Functions to indicate resources for Virtual Function use, Memory Space, Non-Prefetchable. The corresponding Base and MaxOffset fields are HwInit. FF Entry unavailable for use. For other settings see PCIe specification.
7-4 BEI	BAR equivalent indicator This field indicates the equivalent PCI BAR for this entry. For 64-bit BARs, the BEI value indicates the equivalent PCI BAR location for lower DWORD. A Type 0 function is permitted to use EA to allocate resources for itself, and such resources must indicate a BEI value of 0-5, 7 or 8. Physical functions are allowed to use EA to allocate resources for its associated Virtual Functions, and such resources must indicate a BEI value of 9-14.
3 —	Reserved
2-0 ENTRY_SIZE	Entry size Number of DWs following the initial DW in this entry. Hardwired to 011b for 64b base and 32b MaxOffset.

53.4.6.1.39 PCI EA per-entry 1 base register (PCI_CFC_EA_PE1_BASE)

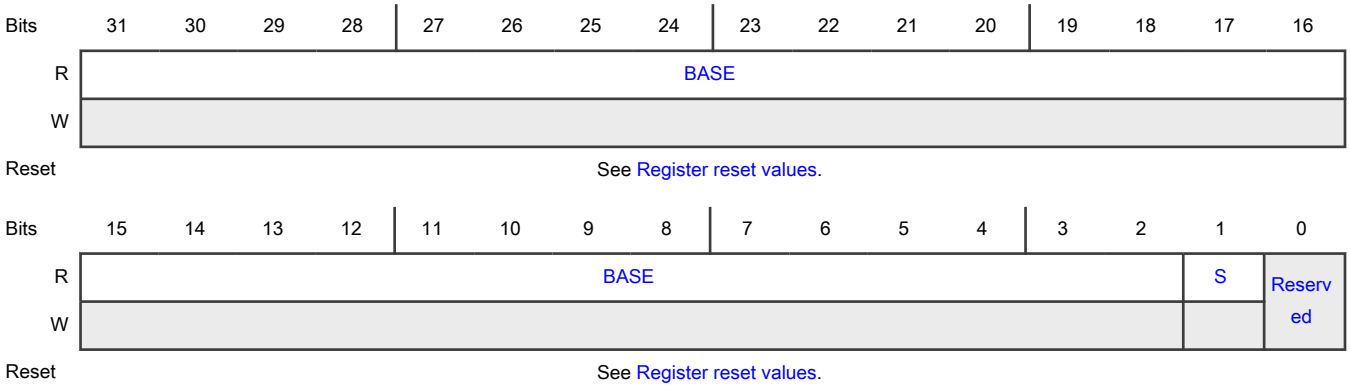
Offset

Register	Offset
PCI_CFC_EA_PE1_BASE	B4h

Function

This is the PCI config capability enhanced allocation per-entry base register.

Diagram



Register reset values

Register	Reset value
PCI_CFC_EA_PE1_BASE	NETC_F0_PCI_HDR_TYPE0: 60BC_0002h NETC_F1_PCI_HDR_TYPE0: 60BD_0002h NETC_F2_PCI_HDR_TYPE0: 60BE_0002h NETC_F3_PCI_HDR_TYPE0: 60BF_0002h NETC_F4_PCI_HDR_TYPE0: 60C0_0002h

Fields

Field	Function
31-2 BASE	Base DW address of the start of the resource range. The value in the Base field ([63:2]) plus the value in the MaxOffset field indicates the address of the last included DW of the resource range. All addresses are naturally aligned for NETC
1 S	Field size 0 32-bit 1 64-bit
0 —	Reserved

53.4.6.1.40 PCI EA per-entry a max offset register (PCI_CFC_EA_PE1_MAXOFF - PCI_CFC_EA_PE3_MAXOFF)

Offset

Register	Offset
PCI_CFC_EA_PE1_MAXOFF	B8h
PCI_CFC_EA_PE2_MAXOFF	C8h
PCI_CFC_EA_PE3_MAXOFF	D8h

Function

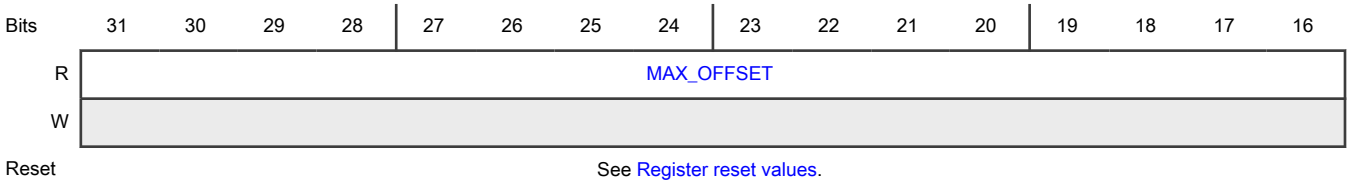
This is the PCI config capability enhanced allocation per-entry max offset register.

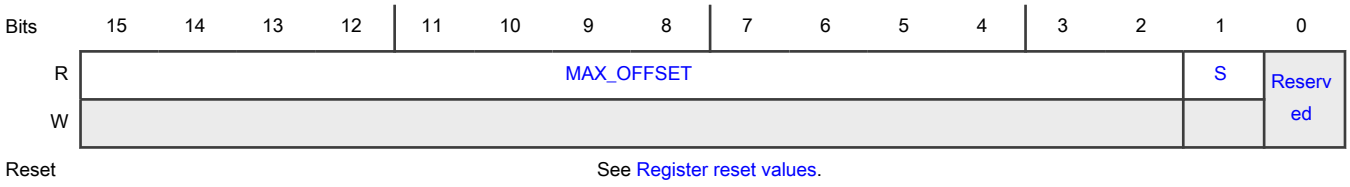
NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	PCI_CFC_EA_PE1_MAXOFF	PCI_CFC_EA_PE2_MAXOFF–PCI_CFC_EA_PE3_MAXOFF
NETC_F1_PCI_HDR_TYPE0	PCI_CFC_EA_PE1_MAXOFF	PCI_CFC_EA_PE2_MAXOFF–PCI_CFC_EA_PE3_MAXOFF
NETC_F2_PCI_HDR_TYPE0	PCI_CFC_EA_PE1_MAXOFF	PCI_CFC_EA_PE2_MAXOFF–PCI_CFC_EA_PE3_MAXOFF
NETC_F3_PCI_HDR_TYPE0	PCI_CFC_EA_PE1_MAXOFF	PCI_CFC_EA_PE2_MAXOFF–PCI_CFC_EA_PE3_MAXOFF
NETC_F4_PCI_HDR_TYPE0	PCI_CFC_EA_PE1_MAXOFF–PCI_CFC_EA_PE3_MAXOFF	—

Diagram





Register reset values

Register	Reset value
PCI_CFC_EA_PE1_MAXOFF	NETC_F0_PCI_HDR_TYPE0–NETC_F4_PCI_HDR_TYPE0: 0000_FFFCh
PCI_CFC_EA_PE2_MAXOFF–PCI_CFC_EA_PE3_MAXOFF	0000_FFFCh

Fields

Field	Function
31-2 MAX_OFFSET	Max offset The value in the Base field ([63:2]) plus the value in the MaxOffset field indicates the address of the last included DW of the resource range. Bits [1:0] of the MaxOffset are not included in the MaxOffset field, and must always be interpreted as 11b. **NOTE** For SR-IOV, this value is the size and alignment for a single VF. Multiply the aperture size with the value set in NumVFs to determine the total amount of space the BAR or BAR pair will map after VF Enable and VF MSE are Set.
1 S	Field size 0 32-bit 1 64-bit
0 —	Reserved

53.4.6.1.41 PCI EA per-entry 2 format register (PCI_CFC_EA_PE2_FMT)

Offset

Register	Offset
PCI_CFC_EA_PE2_FMT	C0h

Function

This is the PCI config capability enhanced allocation per-entry format register.

NOTE

Each module instance supports a different number of registers.

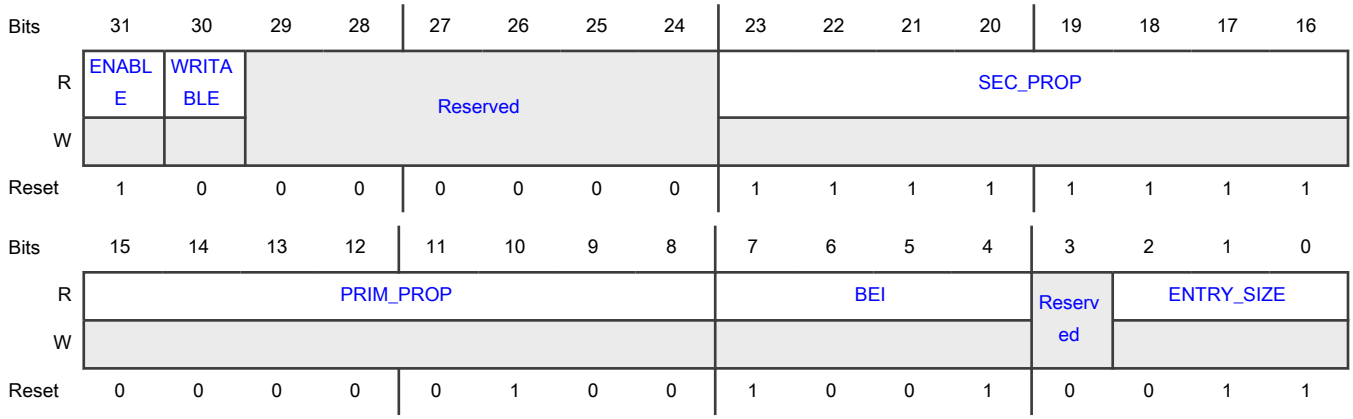
Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE2_FMT

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
NETC_F1_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE2_FMT
NETC_F2_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE2_FMT
NETC_F3_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE2_FMT
NETC_F4_PCI_HDR_TYPE0	PCI_CFC_EA_PE2_FMT	—

Diagram



Fields

Field	Function
31 ENABLE	Enable 0 Disabled 1 Enabled
30 WRITABLE	Writable 1b indicates that the Base, MaxOffset and Field Size fields for this entry are RW, 0b indicates those fields are HWInit. Hardwired to 0b.
29-24 —	Reserved
23-16 SEC_PROP	Secondary properties Alternative property when system cannot use the primary property.
15-8 PRIM_PROP	Primary properties Value (h) Resource Definition 00 Memory Space, Non-Prefetchable. The corresponding Base and MaxOffset fields are HwInit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01 Memory Space, Prefetchable. The corresponding Base and MaxOffset fields are HwInit. 03 For use only by Physical Functions to indicate resources for Virtual Function use, Memory Space, Prefetchable. The corresponding Base and MaxOffset fields are HwInit. 04 For use only by Physical Functions to indicate resources for Virtual Function use, Memory Space, Non-Prefetchable. The corresponding Base and MaxOffset fields are HwInit. FF Entry unavailable for use. For other settings see PCIe specification.
7-4 BEI	BAR equivalent indicator This field indicates the equivalent PCI BAR for this entry. For 64-bit BARs, the BEI value indicates the equivalent PCI BAR location for lower DWORD. A Type 0 function is permitted to use EA to allocate resources for itself, and such resources must indicate a BEI value of 0-5, 7 or 8. Physical functions are allowed to use EA to allocate resources for its associated Virtual Functions, and such resources must indicate a BEI value of 9-14.
3 —	Reserved
2-0 ENTRY_SIZE	Entry size Number of DWs following the initial DW in this entry. Hardwired to 011b for 64b base and 32b MaxOffset.

53.4.6.1.42 PCI EA per-entry 2 base register (PCI_CFC_EA_PE2_BASE)

Offset

Register	Offset
PCI_CFC_EA_PE2_BASE	C4h

Function

This is the PCI config capability enhanced allocation per-entry base register.

NOTE

Each module instance supports a different number of registers.

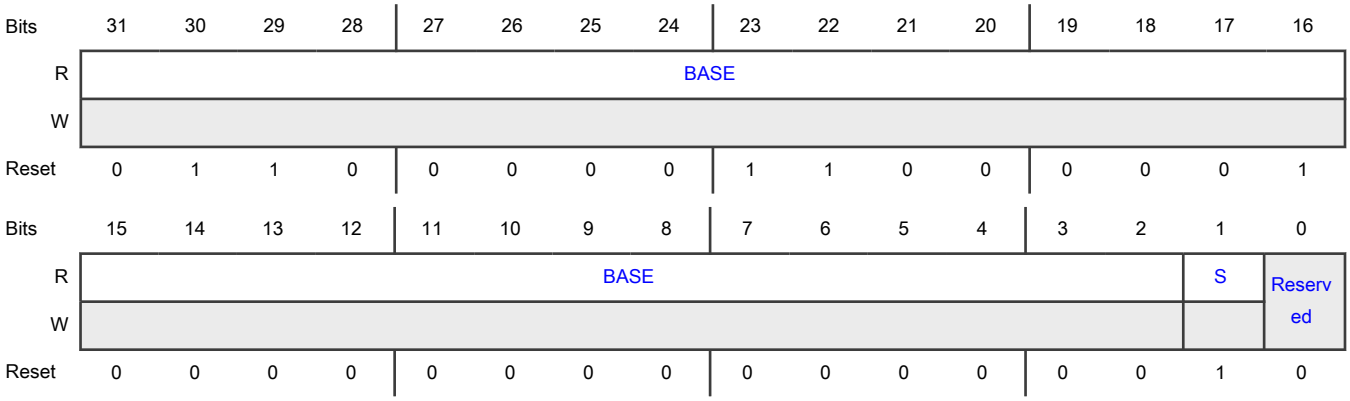
Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE2_BASE
NETC_F1_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE2_BASE

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
NETC_F2_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE2_BASE
NETC_F3_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE2_BASE
NETC_F4_PCI_HDR_TYPE0	PCI_CFC_EA_PE2_BASE	—

Diagram



Fields

Field	Function
31-2 BASE	Base DW address of the start of the resource range. The value in the Base field ([63:2]) plus the value in the MaxOffset field indicates the address of the last included DW of the resource range. All addresses are naturally aligned for NETC
1 S	Field size 0 32-bit 1 64-bit
0 —	Reserved

53.4.6.1.43 PCI EA per-entry 3 format register (PCI_CFC_EA_PE3_FMT)

Offset

Register	Offset
PCI_CFC_EA_PE3_FMT	D0h

Function

This is the PCI config capability enhanced allocation per-entry format register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE3_FMT
NETC_F1_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE3_FMT
NETC_F2_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE3_FMT
NETC_F3_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE3_FMT
NETC_F4_PCI_HDR_TYPE0	PCI_CFC_EA_PE3_FMT	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ENABLE	WRITABLE	Reserved						SEC_PROP							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRIM_PROP								BEI				Reserved	ENTRY_SIZE		
W																
Reset	0	0	0	0	0	0	1	1	1	0	1	1	0	0	1	1

Fields

Field	Function
31 ENABLE	Enable 0 Disabled 1 Enabled
30 WRITABLE	Writable 1b indicates that the Base, MaxOffset and Field Size fields for this entry are RW, 0b indicates those fields are HWInit. Hardwired to 0b.
29-24 —	Reserved
23-16	Secondary properties

Table continues on the next page...

Table continued from the previous page...

Field	Function
SEC_PROP	Alternative property when system cannot use the primary property.
15-8 PRIM_PROP	Primary properties Value (h) Resource Definition 00 Memory Space, Non-Prefetchable. The corresponding Base and MaxOffset fields are HwInit. 01 Memory Space, Prefetchable. The corresponding Base and MaxOffset fields are HwInit. 03 For use only by Physical Functions to indicate resources for Virtual Function use, Memory Space, Prefetchable. The corresponding Base and MaxOffset fields are HwInit. 04 For use only by Physical Functions to indicate resources for Virtual Function use, Memory Space, Non-Prefetchable. The corresponding Base and MaxOffset fields are HwInit. FF Entry unavailable for use. For other settings see PCIe specification.
7-4 BEI	BAR equivalent indicator This field indicates the equivalent PCI BAR for this entry. For 64-bit BARs, the BEI value indicates the equivalent PCI BAR location for lower DWORD. A Type 0 function is permitted to use EA to allocate resources for itself, and such resources must indicate a BEI value of 0-5, 7 or 8. Physical functions are allowed to use EA to allocate resources for its associated Virtual Functions, and such resources must indicate a BEI value of 9-14.
3 —	Reserved
2-0 ENTRY_SIZE	Entry size Number of DWs following the initial DW in this entry. Hardwired to 011b for 64b base and 32b MaxOffset.

53.4.6.1.44 PCI EA per-entry 3 base register (PCI_CFC_EA_PE3_BASE)

Offset

Register	Offset
PCI_CFC_EA_PE3_BASE	D4h

Function

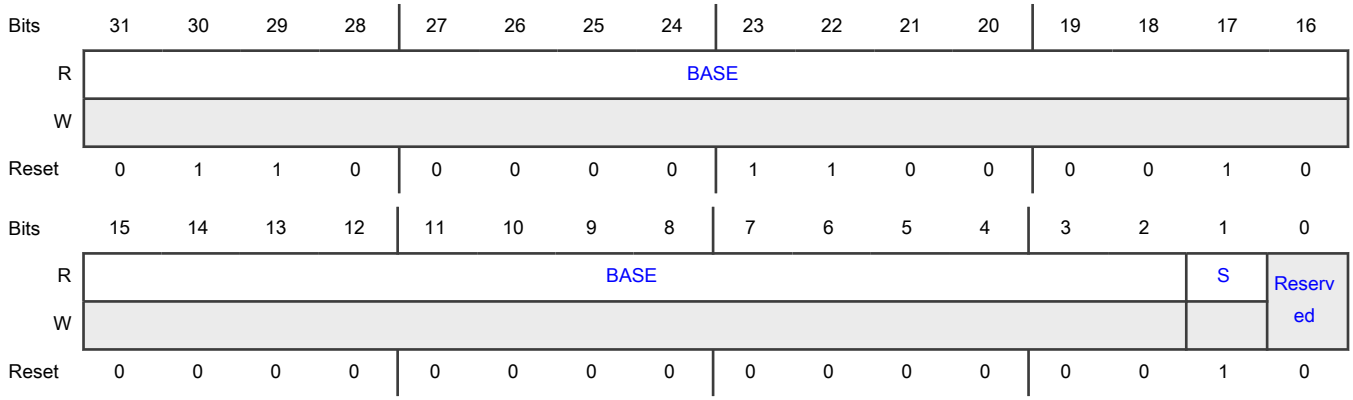
This is the PCI config capability enhanced allocation per-entry base register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE3_BASE
NETC_F1_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE3_BASE
NETC_F2_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE3_BASE
NETC_F3_PCI_HDR_TYPE0	—	PCI_CFC_EA_PE3_BASE
NETC_F4_PCI_HDR_TYPE0	PCI_CFC_EA_PE3_BASE	—

Diagram



Fields

Field	Function
31-2 BASE	Base DW address of the start of the resource range. The value in the Base field ([63:2]) plus the value in the MaxOffset field indicates the address of the last included DW of the resource range. All addresses are naturally aligned for NETC
1 S	Field size 0 32-bit 1 64-bit
0 —	Reserved

53.4.6.1.45 PCIe AER extended capability header (PCIE_CFC_AER_EXT_CAP_HDR)

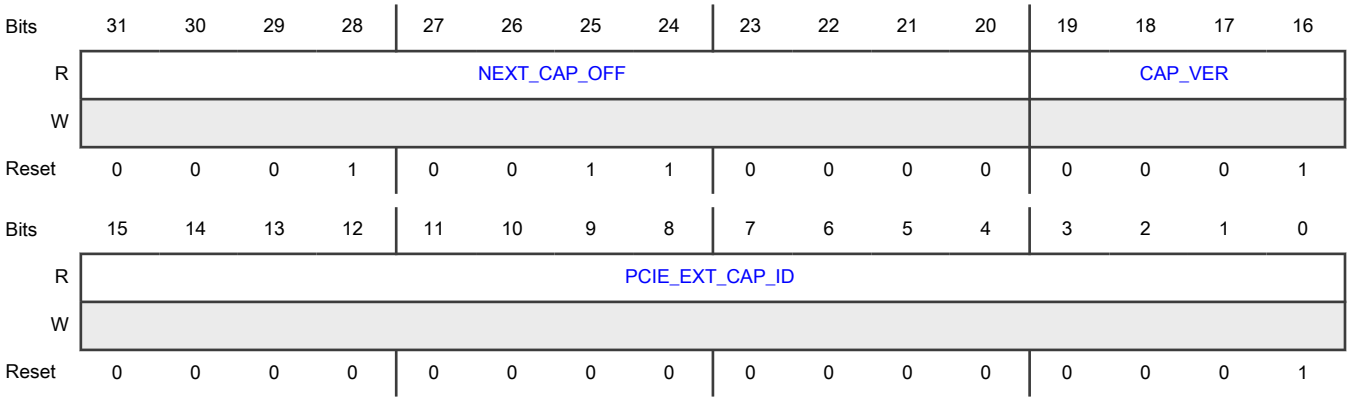
Offset

Register	Offset
PCIE_CFC_AER_EXT_CAP_HDR	100h

Function

This is the PCIe config capability Advanced Error Reporting extended capability header.

Diagram



Fields

Field	Function
31-20 NEXT_CAP_OFF	Next capability offset This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of capabilities.
19-16 CAP_VER	Capability version Must be 1h for this version of the specification.
15-0 PCIE_EXT_CAP_ID	The Extended Capability ID for the AER Extended Capability is 0001h.

53.4.6.1.46 PCIe AER uncorrectable error status register (PCIE_CFC_AER_UCORR_ERR_STAT)

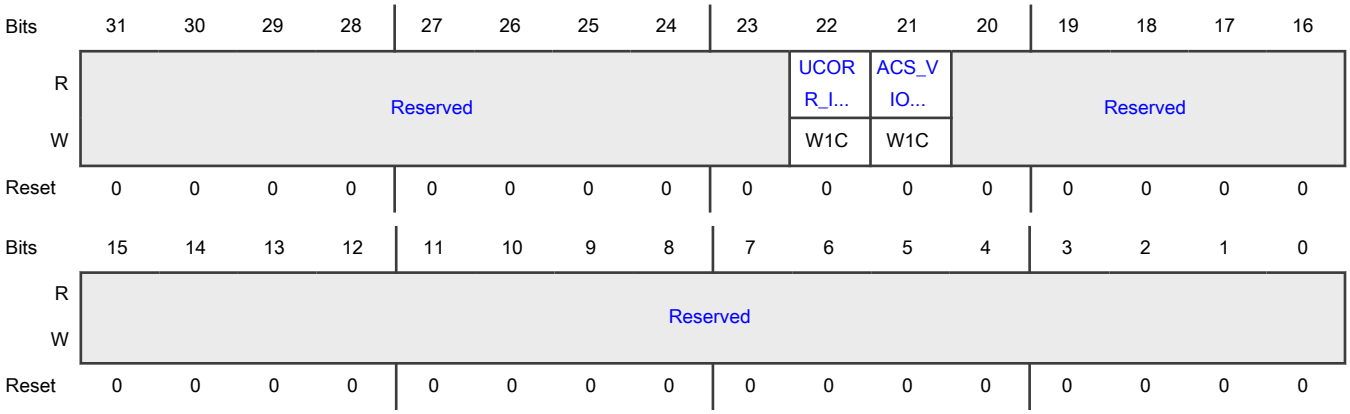
Offset

Register	Offset
PCIE_CFC_AER_UCORR_ERR_STAT	104h

Function

This is the PCIe config capability Advanced Error Reporting uncorrectable error status register. The Uncorrectable Error Status register indicates error detection status of individual errors on a PCI Express device Function. An individual error status bit that is set indicates that a particular error was detected; software may clear an error status by writing a 1b to the respective bit.

Diagram



Fields

Field	Function
31-23 —	Reserved
22 UCORR_INT_ERR	Uncorrectable internal error Set to indicate some uncorrectable internal error such as a multi-bit error in an internal memory has occurred. **NOTE** This bit is “sticky” and will be preserved across function level resets.
21 ACS_VIOLATION_STAT	ACS violation status Set to indicate an ACS violation such as an access from a disallowed source. **NOTE** This bit is “sticky” and will be preserved across function level resets.
20-0 —	Reserved

53.4.6.1.47 PCIe AER uncorrectable error mask register (PCIE_CFC_AER_UCORR_ERR_MASK)

Offset

Register	Offset
PCIE_CFC_AER_UCORR_ERR_MASK	108h

Function

This is the PCIe config capability Advanced Error Reporting uncorrectable error mask register. The Uncorrectable Error Mask register controls reporting of individual errors by the device Function to the PCI Express Root Complex Event Collector via a PCI Express error Message. A masked error (respective bit set in the mask register) is not recorded or reported in the Header Log or First Error Pointer, and is not reported to the PCI Express Root Complex Event Collector by this Function. There is a mask bit per error bit of the Uncorrectable Error Status register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved									UCORR_I...	Reserved					
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-23 —	Reserved
22 UCORR_INT_ERR_MASK	Uncorrectable internal error mask 0 Not masked 1 Masked **NOTE** This bit is “sticky” and will be preserved across function level resets.
21-0 —	Reserved

53.4.6.1.48 PCIe AER uncorrectable error severity register (PCIE_CFC_AER_UCORR_ERR_SEV)

Offset

Register	Offset
PCIE_CFC_AER_UCORR_ERR_SEV	10Ch

Function

This is the PCIe config capability Advanced Error Reporting uncorrectable error severity register. The Uncorrectable Error Severity register controls whether an individual error is reported as a Non-fatal or Fatal error. An error is reported as fatal when the corresponding error bit in the severity register is Set. If the bit is Clear, the corresponding error is considered non-fatal.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved									UCORR_I...	Reserved					
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-23 —	Reserved
22 UCORR_INT_SEV	Uncorrectable internal severity 0 Non-fatal 1 Fatal **NOTE** This bit is “sticky” and will be preserved across function level resets.
21-0 —	Reserved

53.4.6.1.49 PCIe AER correctable error status register (PCIE_CFC_AER_CORR_ERR_STAT)

Offset

Register	Offset
PCIE_CFC_AER_CORR_ERR_STAT	110h

Function

This is the PCIe config capability Advanced Error Reporting correctable error status register. The Correctable Error Status register reports error status of individual correctable error sources on a PCI Express device Function. When an individual error status bit is Set, it indicates that a particular error occurred; software may clear an error status by writing a 1b to the respective bit.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	CORR_IN...	Reserved													
W		W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14 CORR_INT_ERR	Correctable internal error Set to indicate some correctable internal error has occurred such as a single bit error in an internal memory. **NOTE** This bit is “sticky” and will be preserved across function level resets.
13-0 —	Reserved

53.4.6.1.50 PCIe AER correctable error mask register (PCIE_CFC_AER_CORR_ERR_MASK)

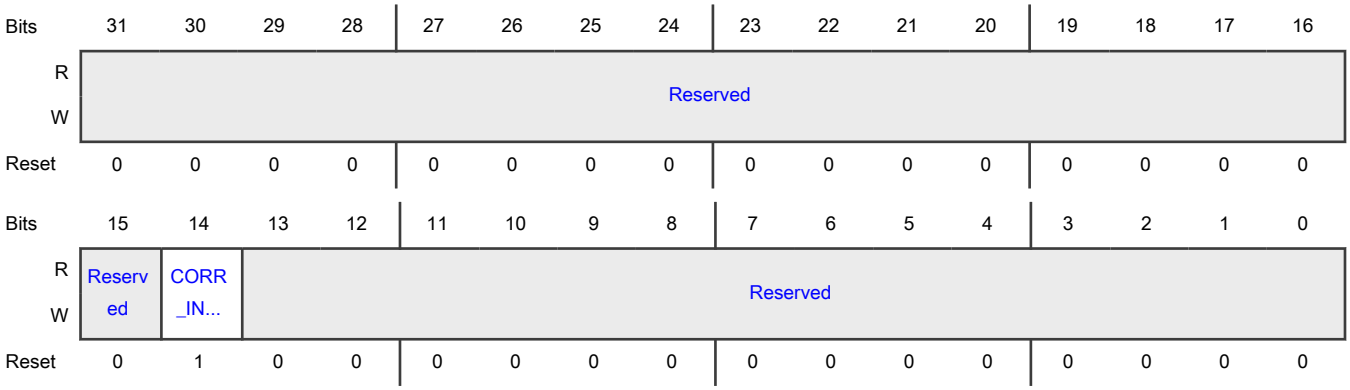
Offset

Register	Offset
PCIE_CFC_AER_CORR_ERR_MASK	114h

Function

This is the PCIe config capability Advanced Error Reporting correctable error mask register. The Correctable Error Mask register controls reporting of individual correctable errors by this Function to the PCI Express Root Complex via a PCI Express error Message. A masked error (respective bit set in the mask register) is not reported to the PCI Express Root Complex by this Function. There is a mask bit per error bit in the Correctable Error Status register.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 CORR_INT_MASK	Correctable internal error mask 0 Not masked 1 Masked **NOTE** This bit is “sticky” and will be preserved across function level resets. **WARNING** Function F0 (Timer) and F1 (EMDIO) must not clear mask bit, enabling reporting of correctable interrupts, to prevent loss of uncorrectable error reporting.
13-0 —	Reserved

53.4.6.1.51 PCIe AER capabilities and control register (PCIE_CFC_AER_CAP_CTL)

Offset

Register	Offset
PCIE_CFC_AER_CAP_CTL	118h

Function

This is the PCIe config capability Advanced Error Reporting capabilities and control register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												FIRST_ERR_PTR			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5 —	Reserved
4-0 FIRST_ERR_PTR	First error pointer The First Error Pointer is a field that identifies the bit position of the first error reported in the Uncorrectable Error Status register. **NOTE** This field is “sticky” and will be preserved across function level resets.

53.4.6.1.52 PCIe ACS capability header (PCIE_CFC_ACS_CAP_HDR)

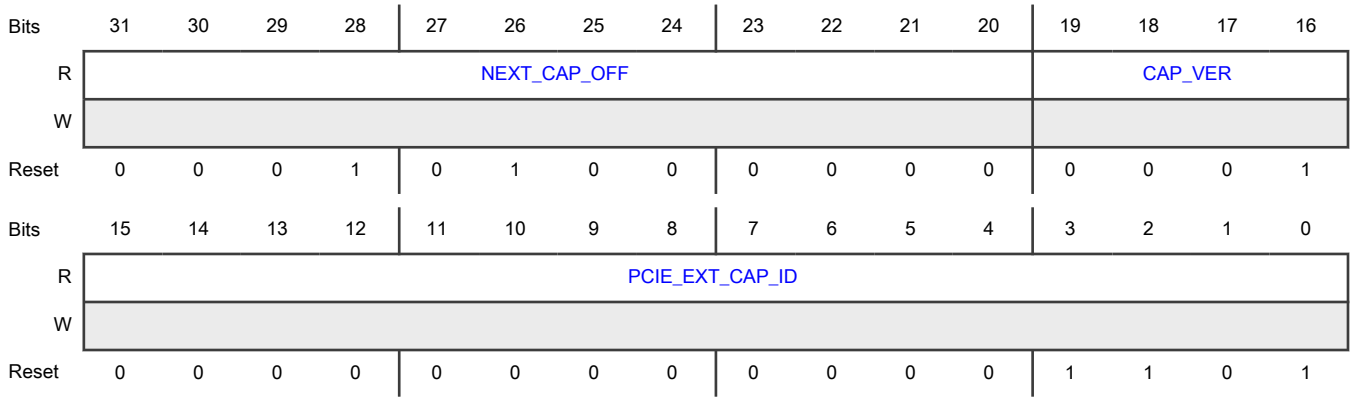
Offset

Register	Offset
PCIE_CFC_ACS_CAP_HDR	130h

Function

This is the PCIe config capability Access Control Services (ACS) capability header.

Diagram



Fields

Field	Function
31-20 NEXT_CAP_OFF	Next capability offset This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of capabilities.
19-16 CAP_VER	Capability version Must be 1h for this version of the specification.
15-0 PCIE_EXT_CAP_ID	The Extended Capability ID for the ACS Extended Capability is 000Dh.

53.4.6.1.53 PCIe ACS capability register (PCIE_CFC_ACS_CAP)

Offset

Register	Offset
PCIE_CFC_ACS_CAP	134h

Function

This is the PCIe config capability Access Control Services (ACS) capability register.

Diagram

Fields

Field	Function
15-7 —	Reserved
6 ACS_DIR_TRANS_P2P	ACS direct translated P2P (T) Set to indicate that the Function supports direct translated peer-to-peer.
5-3 —	Reserved
2 ACS_P2P_REQ_REDIR	ACS P2P request dedirect (R) Set to indicate that the Function supports peer-to-peer request redirection.
1 ACS_TRANS_BLOCK	ACS translation blocking (B) Set to indicate that the Function supports Translation Blocking.
0 —	Reserved

53.4.6.1.54 PCIe ACS control register (PCIE_CFC_ACS_CTL)

Offset

Register	Offset
PCIE_CFC_ACS_CTL	136h

Function

This is the PCIe config capability Access Control Services (ACS) control register.

Diagram

Fields

Field	Function
15-7 —	Reserved
6 ACS_DIR_TRANS_P2P_EN	ACS direct translated P2P enable (T) When set the Function determines how to direct peer-to-peer requests based on ATS Ignored if ACS Translation Blocking Enable is set. This bit is read-only if the functionality is not implemented.
5-3 —	Reserved
2 ACS_P2P_REQ_REDIR_EN	ACS P2P request dedirect enable (R) When set the Function directs peer-to-peer requests to the SoC interconnect. This bit is read-only if the functionality is not implemented.
1 ACS_TRANS_BLOCK_EN	ACS translation blocking enable (B) When set the Function directs peer-to-peer requests to the SoC interconnect. This bit is read-only if the functionality is not implemented.
0 —	Reserved

53.4.6.1.55 PCIe readiness time reporting capability header (PCIE_CFC_RTR_CAP_HDR)

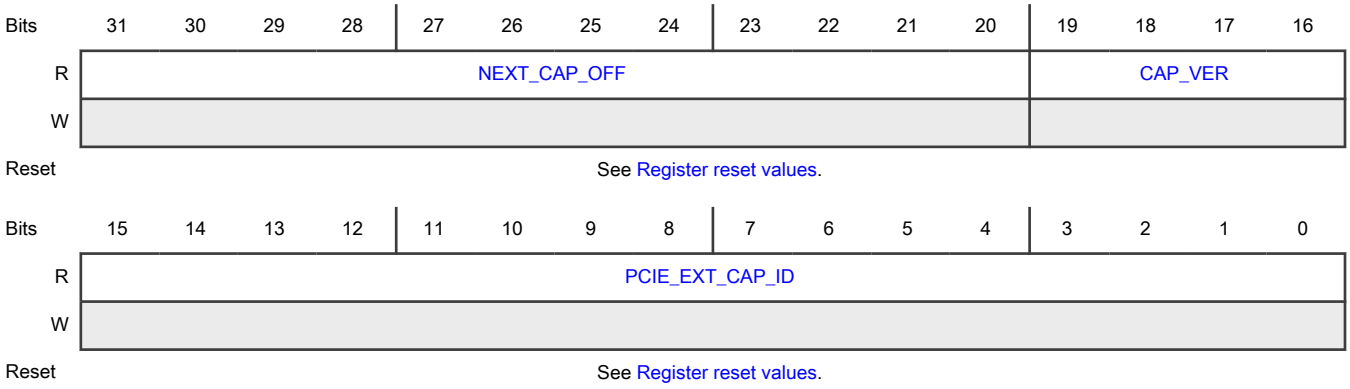
Offset

Register	Offset
PCIE_CFC_RTR_CAP_HDR	140h

Function

This is the PCIe config capability readiness time reporting capability header.

Diagram



Register reset values

Register	Reset value
PCIE_CFC_RTR_CAP_HDR	NETC_F0_PCI_HDR_TYPE0–NETC_F3_PCI_HDR_TYPE0: 0001_0022h NETC_F4_PCI_HDR_TYPE0: 1501_0022h

Fields

Field	Function
31-20 NEXT_CAP_OFF	Next capability offset This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of capabilities.
19-16 CAP_VER	Capability version Must be 1h for this version of the specification.
15-0 PCIE_EXT_CAP_ID	The Extended Capability ID for the Readiness Time Reporting Extended Capability is 0022h.

53.4.6.1.56 PCIe RTR readiness time reporting 1 register (PCIE_CFC_RTR_RTR1)

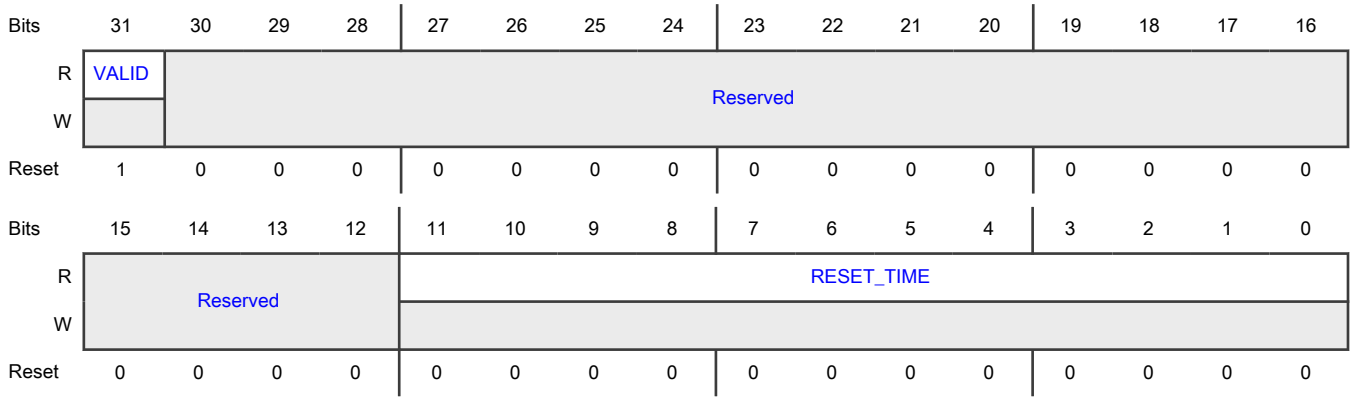
Offset

Register	Offset
PCIE_CFC_RTR_RTR1	144h

Function

This is the PCIe config capability Readiness Time Reporting 1 register.

Diagram



Fields

Field	Function
31 VALID	Valid If Set, indicates that all time values in this capability are valid. If Clear, indicates that the time values in this capability are not yet available. If this bit remains Clear and 1 minute has elapsed after all associated device driver(s) have started, software is permitted to assume that this bit will never be set.
30-12 —	Reserved
11-0 RESET_TIME	Reset Time The time the Function requires to become Configuration-Ready after the completion of Conventional Reset. This field is undefined when the Valid bit is Clear. This field must be less than or equal to the encoded value A1Eh.

53.4.6.1.57 PCIe RTR readiness time reporting 2 register (PCIE_CFC_RTR_RTR2)

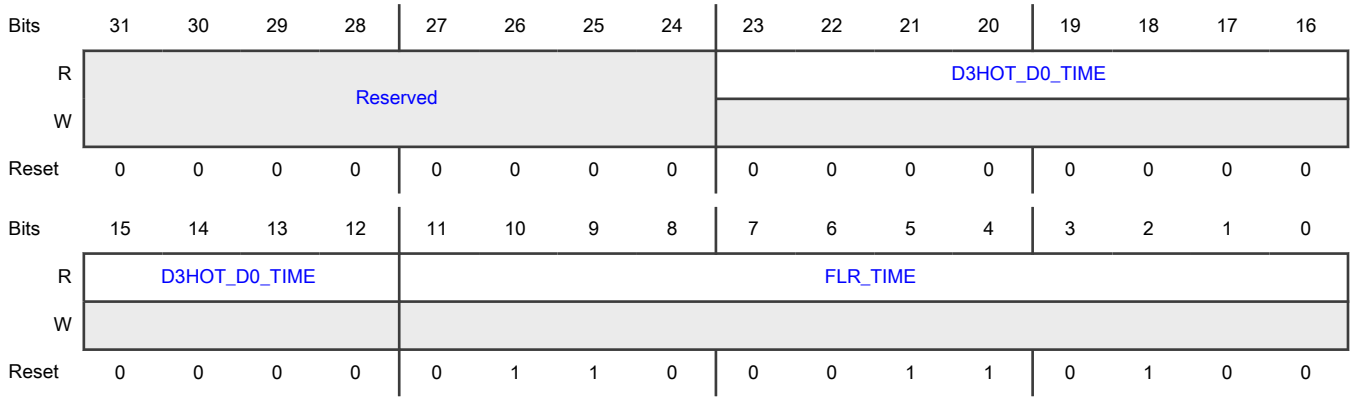
Offset

Register	Offset
PCIE_CFC_RTR_RTR2	148h

Function

This is the PCIe config capability Readiness Time Reporting 2 register.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-12 D3HOT_D0_TIME	D3 hot to D0 time $D3_{hot}$ to D0 Time is the time that the Function requires after it is directed from $D3_{hot}$ to D0 before it is Configuration-Ready and has returned to either $D0_{uninitialized}$ or $D0_{active}$ state (see the PCI Bus Power Management Interface Specification). This field is undefined when the PCIE_CFC_RTR_RTR1[VALID] bit is Clear. This field must be less than or equal to the encoded value 80Ah.
11-0 FLR_TIME	FLR Time The time that the Function requires to become Configuration-Ready after it was issued an FLR. This field is undefined when the PCIE_CFC_RTR_RTR1[VALID] bit is Clear. This field must be less than or equal to the encoded value A1Eh. **NOTE** The value is taken directly from IERB register NETCFLRCR. The value in this field is encoded, where bits 8-0 is the value, and bits 11-9 is the scale. The scale is hardwired as 32.768us.

53.4.6.1.58 PCIe SR-IOV capability header (PCIE_CFC_SRIOV_CAP_HDR)

Offset

Register	Offset
PCIE_CFC_SRIOV_CAP_HDR	150h

Function

This is the PCIe config capability SR-IOV capability header.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CAP_HDR
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CAP_HDR
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CAP_HDR
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CAP_HDR
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_CAP_HDR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NEXT_CAP_OFF												CAP_VER			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCIE_EXT_CAP_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-20 NEXT_CAP_OFF	Next capability offset This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of capabilities.
19-16 CAP_VER	Capability version Must be 1h for this version of the specification.
15-0 PCIE_EXT_CAP_ID	The Extended Capability ID for the ACS Extended Capability is 0010h.

53.4.6.1.59 PCIe SR-IOV capability register (PCIE_CFC_SRIOV_CAP)

Offset

Register	Offset
PCIE_CFC_SRIOV_CAP	154h

Function

This is the PCIe config capability SR-IOV capability register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CAP
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CAP
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CAP
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CAP
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_CAP	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VF_MIGRATION_ING_MSG_NUM												Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														ARI_CAP...	VF_MIGR...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-21	VF migration interrupt message number Migration is supported only with MR-IOV.

Table continues on the next page...

Table continued from the previous page...

Field	Function
VF_MIGRATION_ING_MSG_NUM	
20-2 —	Reserved
1 ARI_CAP_HIER_PRSV	ARI capable hierarchy preserved ARI is not supported by Integrated End Points
0 VF_MIGRATION_CAP	VF migration capable Migration is supported only with MR-IOV.

53.4.6.1.60 PCIe SR-IOV control register (PCIE_CFC_SRIOV_CTL)

Offset

Register	Offset
PCIE_CFC_SRIOV_CTL	158h

Function

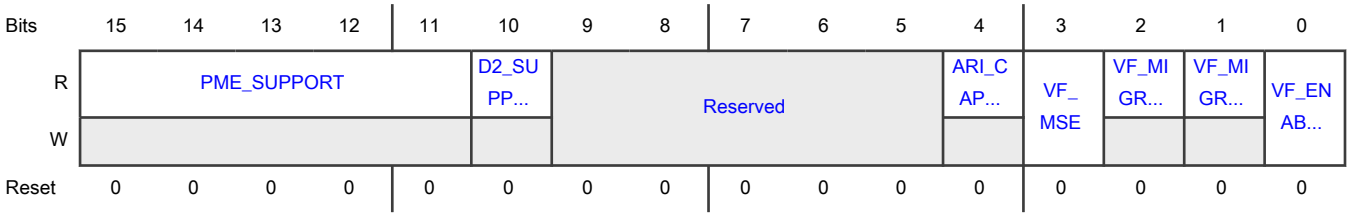
This is the PCIe config capability SR-IOV control register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CTL
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CTL
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CTL
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_CTL
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_CTL	—

Diagram



Fields

Field	Function
15-11 PME_SUPPORT	PME support Wake-on-LAN events not supported by ENETC VFs. Hardwired to 00000b.
10 D2_SUPPORT	PCI-PM D2 support Not supported by ENETC. Hardwired to 0b.
9-5 —	Reserved
4 ARI_CAP_HIERARCHY	ARI capable hierarchy ARI is not supported by Integrated End Points
3 VF_MSE	Memory Space Enable for Virtual Functions When VF Enable is Set, VF memory space will respond only when VF MSE is Set.
2 VF_MIGRATION_INT_ENABLE	VF migration interrupt enable Migration is supported only with MR-IOV. Hardwired to 0b.
1 VF_MIGRATION_ENABLE	VF migration enable Migration is supported only with MR-IOV. Hardwired to 0b.
0 VF_ENABLE	VF enable 0 Disabled 1 Enabled

53.4.6.1.61 PCIe SR-IOV status register (PCIE_CFC_SRIOV_STAT)

Offset

Register	Offset
PCIE_CFC_SRIOV_STAT	15Ah

Function

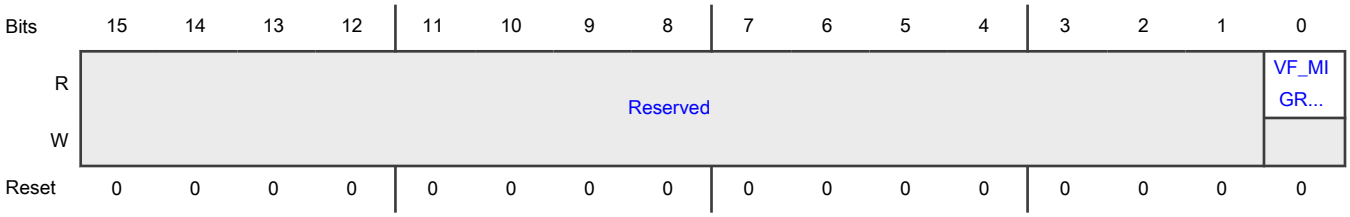
This is the PCIe config capability SR-IOV status register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_STAT
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_STAT
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_STAT
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_STAT
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_STAT	—

Diagram



Fields

Field	Function
15-1 —	Reserved
0 VF_MIGRATION_STATUS	VF migration status Migration is supported only with MR-IOV. Hardwired to 0b.

53.4.6.1.62 PCIe SR-IOV initial VFs register (PCIE_CFC_SRIOV_INIT_VFS)

Offset

Register	Offset
PCIE_CFC_SRIOV_INIT_VFS	15Ch

Function

This is the PCIe config capability SR-IOV initial VFs register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_INIT_VFS
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_INIT_VFS
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_INIT_VFS
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_INIT_VFS
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_INIT_VFS	—

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INITIAL_VFS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
15-0 INITIAL_VFS	Initial VFs Number of VFs that are initially associated with the PF.

53.4.6.1.63 PCIe SR-IOV total VFs register (PCIE_CFC_SRIOV_TOTAL_VFS)

Offset

Register	Offset
PCIE_CFC_SRIOV_TOTAL_VFS	15Eh

Function

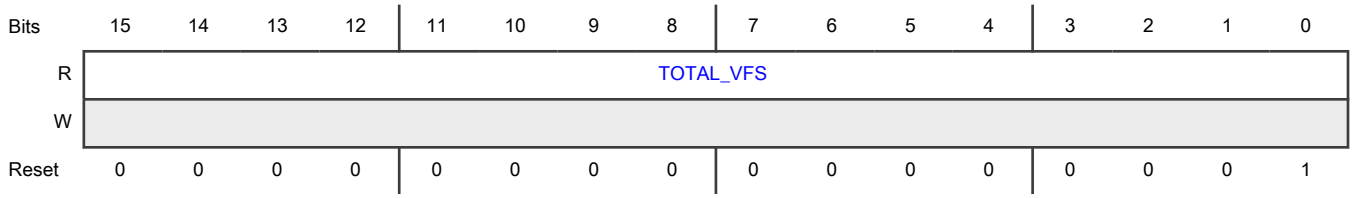
This is the PCIe config capability SR-IOV total VFs register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_TOTAL_VFS
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_TOTAL_VFS
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_TOTAL_VFS
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_TOTAL_VFS
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_TOTAL_VFS	—

Diagram



Fields

Field	Function
15-0 TOTAL_VFS	Total VFs Maximum number of VFs that could be associated with the PF.

53.4.6.1.64 PCIe SR-IOV num VFs register (PCIE_CFC_SRIOV_NUM_VFS)

Offset

Register	Offset
PCIE_CFC_SRIOV_NUM_VFS	160h

Function

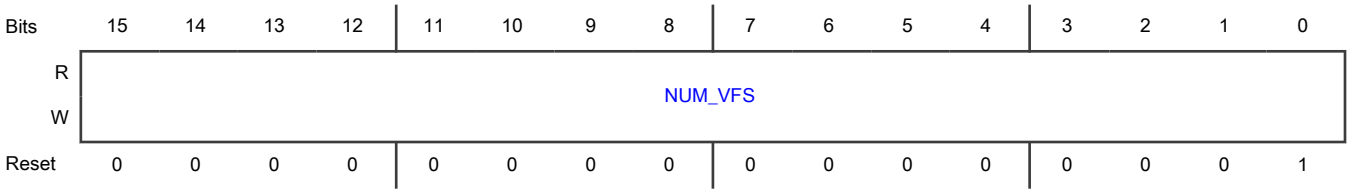
This is the PCIe config capability SR-IOV num VFs register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_NUM_VFS
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_NUM_VFS
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_NUM_VFS
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_NUM_VFS
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_NUM_VFS	—

Diagram



Fields

Field	Function
15-0 NUM_VFS	Number of VFs Controls the number of VFs that are visible. A VF exists if the VF number is less than or equal to TOTAL_VFS. NUM_VFS may only be written while VF_ENABLE is clear. If NUM_VFS is written when VF_ENABLE is set, the results are undefined. By default all VFs are visible.

53.4.6.1.65 PCIe SR-IOV function dependency list register (PCIE_CFC_SRIOV_FUNC_DEP_LIST)

Offset

Register	Offset
PCIE_CFC_SRIOV_FUNC_DEP_LIST	162h

Function

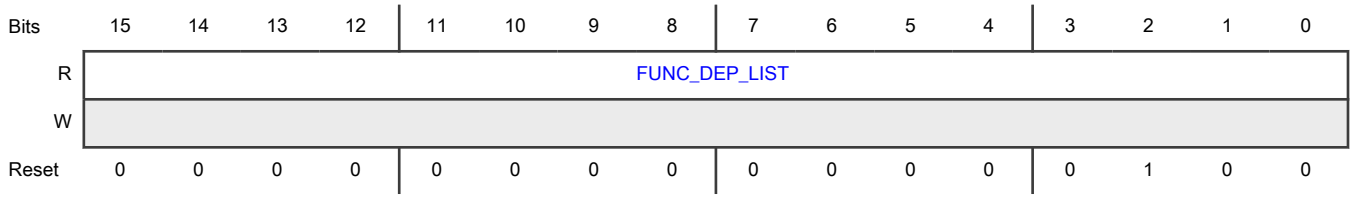
This is the PCIe config capability SR-IOV function dependency list register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_FUNC_DEP_LIST
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_FUNC_DEP_LIST
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_FUNC_DEP_LIST
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_FUNC_DEP_LIST
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_FUNC_DEP_LIST	—

Diagram



Fields

Field	Function
15-0 FUNC_DEP_LIST	Function dependency list The Function Dependency Link contains the RID of the PF implementing this Capability structure.

53.4.6.1.66 PCIe SR-IOV first VF offset register (PCIE_CFC_SRIOV_FIRST_VF_OFF)

Offset

Register	Offset
PCIE_CFC_SRIOV_FIRST_VF_OFF	164h

Function

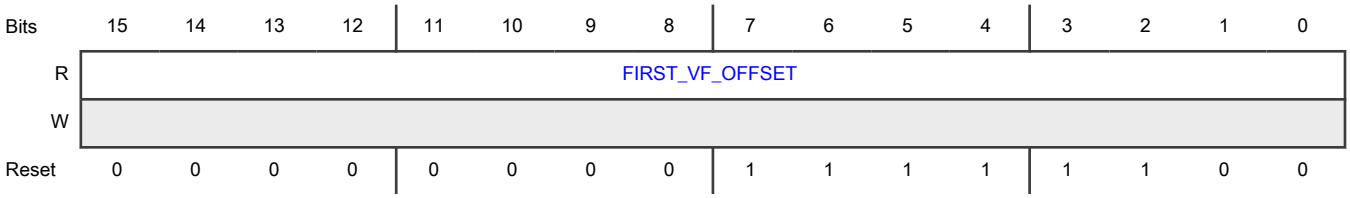
This is the PCIe config capability SR-IOV first VF offset register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_FIRST_VF_OFF
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_FIRST_VF_OFF
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_FIRST_VF_OFF
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_FIRST_VF_OFF
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_FIRST_VF_OFF	—

Diagram



Fields

Field	Function
15-0 FIRST_VF_OFFSET	First VF offset First VF Offset is a constant and defines the Routing ID offset of the first VF that is associated with the PF that contains this Capability structure. The first VF's 16-bit Routing ID is calculated by adding the contents of this field to the Routing ID of the PF containing this field ignoring any carry, 20 using unsigned, 16-bit arithmetic.

53.4.6.1.67 PCIe SR-IOV VF stride register (PCIE_CFC_SRIOV_VF_STRIDE)

Offset

Register	Offset
PCIE_CFC_SRIOV_VF_STRIDE	166h

Function

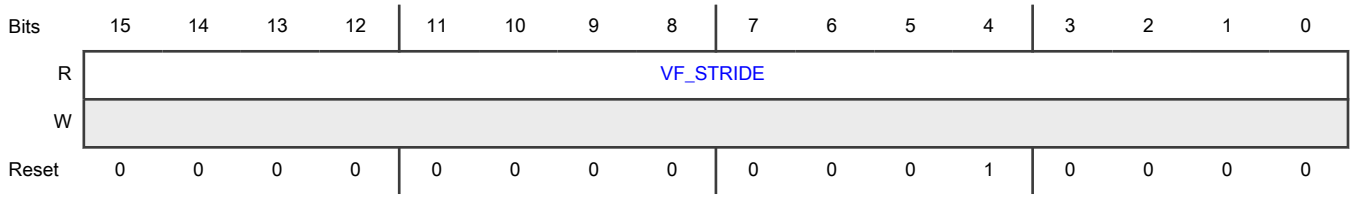
This is the PCIe config capability SR-IOV VF stride register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_STRIDE
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_STRIDE
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_STRIDE
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_STRIDE
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_VF_STRIDE	—

Diagram



Fields

Field	Function
15-0 VF_STRIDE	VF stride VF Stride defines the Routing ID offset from one VF to the next one for all VFs associated with the PF that contains this Capability structure.

53.4.6.1.68 PCIe SR-IOV VF device ID register (PCIE_CFC_SRIOV_VF_DEV_ID)

Offset

Register	Offset
PCIE_CFC_SRIOV_VF_DEV_ID	16Ah

Function

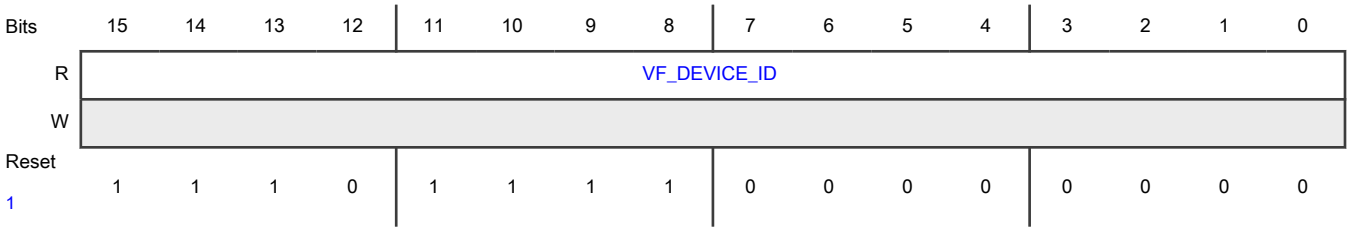
This is the PCIe config capability SR-IOV VF device ID register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_DEV_ID
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_DEV_ID
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_DEV_ID
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_DEV_ID
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_VF_DEV_ID	—

Diagram



1. The value read is taken from IERB register PF*a*_IEP_CFC_VFDID.

Fields

Field	Function
15-0 VF_DEVICE_ID	VF device ID This field contains the Device ID that should be presented for every VF to the SI.

53.4.6.1.69 PCIe SR-IOV supported page sizes ID register (PCIE_CFC_SRIOV_SUP_PAGE_SIZES)

Offset

Register	Offset
PCIE_CFC_SRIOV_SUP_PAGE_SIZES	16Ch

Function

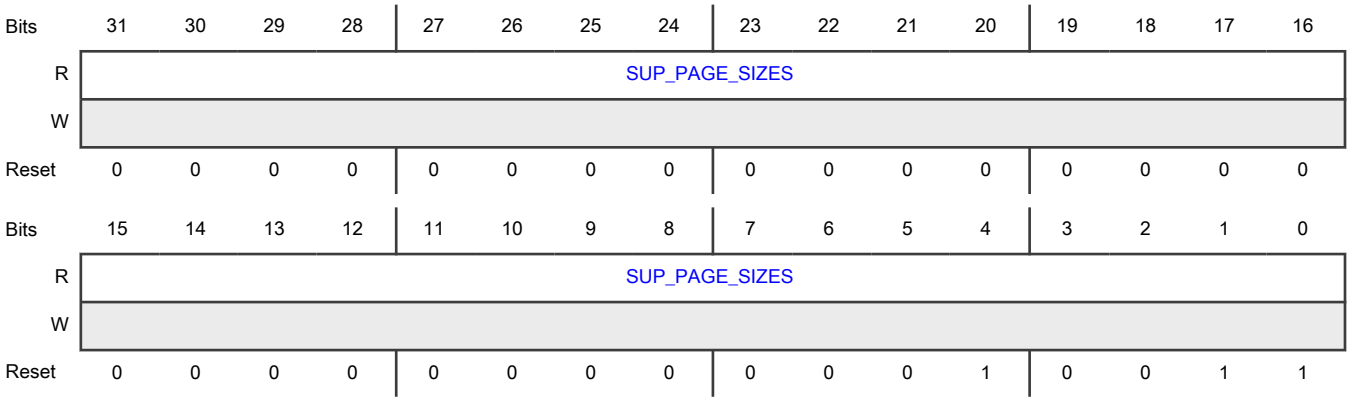
This is the PCIe config capability SR-IOV supported page sizes register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_SUP_PAGE_SIZES
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_SUP_PAGE_SIZES
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_SUP_PAGE_SIZES
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_SUP_PAGE_SIZES
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_SUP_PAGE_SIZES	—

Diagram



Fields

Field	Function
31-0 SUP_PAGE_SIZES	Supported page sizes The field indicates all the page sizes supported by the PF and, as required by the SR-IOV specification, ENETC supports 4 KB, 8 KB and 64 KB page sizes. Based on the Supported Page Size field, the system

Table continues on the next page...

Field	Function
	software sets the System Page Size field which is used to map the VF BAR memory addresses. Each VF BAR address is aligned to the system page boundary.

53.4.6.1.70 PCIe SR-IOV system page size ID register (PCIE_CFC_SRIOV_SYS_PAGE_SIZE)

Offset

Register	Offset
PCIE_CFC_SRIOV_SYS_PAGE_SIZE	170h

Function

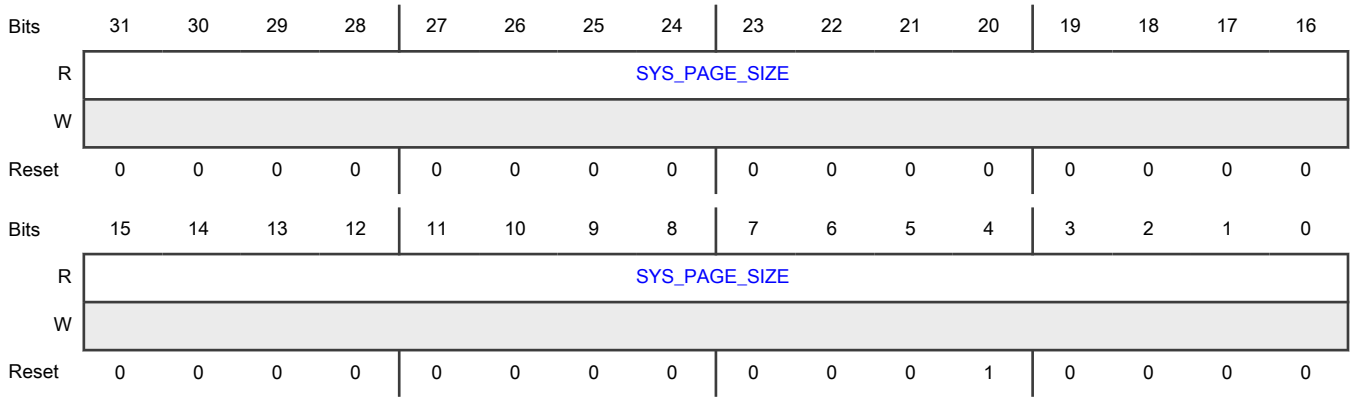
This is the PCIe config capability SR-IOV system page size register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_SYS_PAGE_SIZE
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_SYS_PAGE_SIZE
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_SYS_PAGE_SIZE
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_SYS_PAGE_SIZE
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_SYS_PAGE_SIZE	—

Diagram



Fields

Field	Function
31-0 SYS_PAGE_SIZE	System page size This field defines the page size the system will use to map the VFs' memory addresses. Software must set the value of the System Page Size to one of the page sizes set in the Supported Page Sizes. As with Supported Page Sizes, if bit *n* is set in System Page Size, the VFs associated with this PF are required to support a page size of 2^{n+12}. For example, if bit 1 is Set, the system is using an 8-KB page size. The results are undefined if System Page Size is zero. The results are undefined if more than one bit is set in System Page Size. The results are undefined if a bit is Set in System Page Size that is not Set in Supported Page Sizes. VF Enable must be zero when System Page Size is written. The results are undefined if System Page Size is written when VF Enable is Set. Default value is 10h (i.e., 64 KB).

53.4.6.1.71 PCIe SR-IOV VF base address register a (PCIE_CFC_VF_BAR0 - PCIE_CFC_VF_BAR5)

Offset

Register	Offset
PCIE_CFC_VF_BAR0	174h
PCIE_CFC_VF_BAR1	178h
PCIE_CFC_VF_BAR2	17Ch
PCIE_CFC_VF_BAR3	180h
PCIE_CFC_VF_BAR4	184h
PCIE_CFC_VF_BAR5	188h

Function

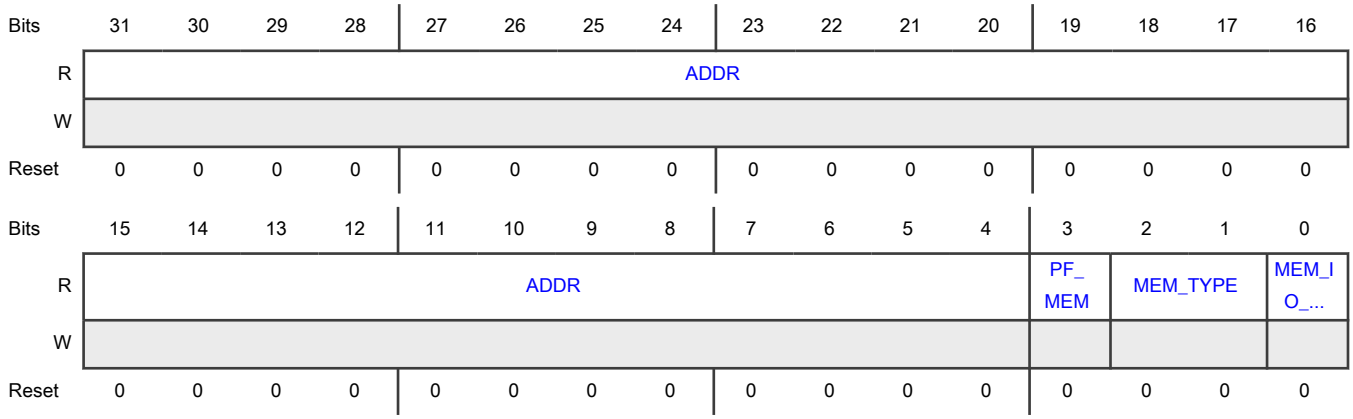
This is the PCIe config capability SR-IOV VF base address register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_VF_BAR0–PCIE_CFC_VF_BAR5
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_VF_BAR0–PCIE_CFC_VF_BAR5
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_VF_BAR0–PCIE_CFC_VF_BAR5
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_VF_BAR0–PCIE_CFC_VF_BAR5
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_VF_BAR0–PCIE_CFC_VF_BAR5	—

Diagram



Fields

Field	Function
31-4 ADDR	Address space (low register for 64-bit BARs). Not supported, hardwired to 000000h.
3 PF_MEM	BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	BARs used. This register is hardwired to 0h.

53.4.6.1.72 PCIe SR-IOV VF migration state array offset register (PCIE_CFC_SRIOV_VF_MIG_STATE_ARR_OFF)

Offset

Register	Offset
PCIE_CFC_SRIOV_VF_MIG_STATE_ARR_OFF	18Ch

Function

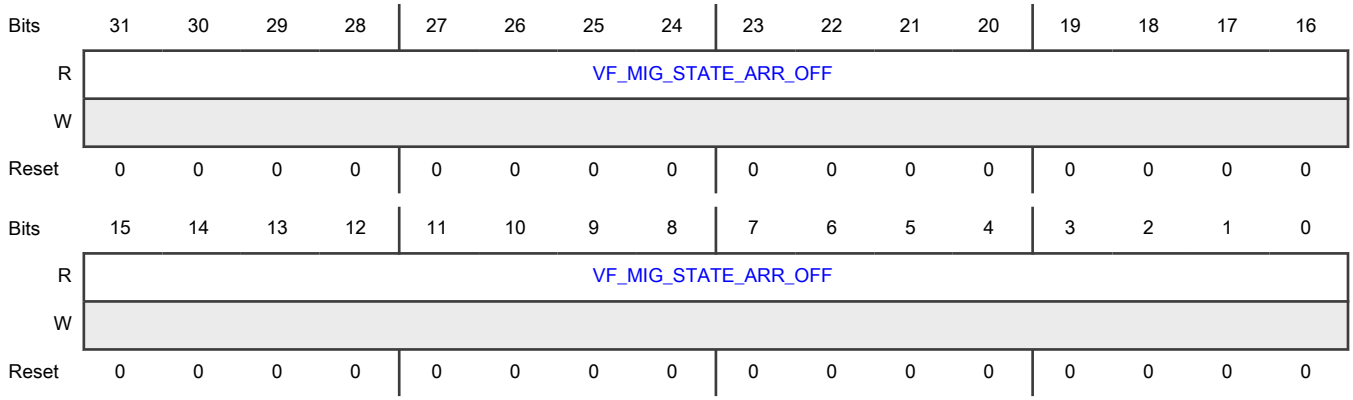
This is the PCIe config capability SR-IOV VF migration state array offset register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
NETC_F0_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_MIG_STATE_ARR_OFF
NETC_F1_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_MIG_STATE_ARR_OFF
NETC_F2_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_MIG_STATE_ARR_OFF
NETC_F3_PCI_HDR_TYPE0	—	PCIE_CFC_SRIOV_VF_MIG_STATE_ARR_OFF
NETC_F4_PCI_HDR_TYPE0	PCIE_CFC_SRIOV_VF_MIG_STATE_ARR_OFF	—

Diagram



Fields

Field	Function
31-0 VF_MIG_STATE_ARR_OFF	VF migration state array offset VF migration not supported. Hardwired to all 0b's.

53.4.6.2 NETC PCI Express ECAM VF config register descriptions

This section describes the PCI Express Type0 config header and capabilities as they relate to the virtual functions of NETC.

53.4.6.2.1 VF_PCI_Config_Header_Type0 memory map

NETC_VF1_PCI_HDR_TYPE0 base address: 6010_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PCI device ID and vendor ID register (PCI_CFH_DID_VID)	32	R	FFFF_FFFFh
4h	PCI command register (PCI_CFH_CMD)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
6h	PCI status register (PCI_CFH_STAT)	16	R	0010h
8h	PCI revision ID and classcode register (PCI_CFH_REVID_CLASSCODE)	32	R	0200_0103h
Ch	PCI cache line size register (PCI_CFH_CL_SIZE)	8	R	00h
Dh	PCI latency timer register (PCI_CFH_LAT_TIMER)	8	R	00h
Eh	PCI header type register (PCI_CFH_HDR_TYPE)	8	R	00h
Fh	PCI BIST register (PCI_CFH_BIST)	8	R	00h
10h	PCI base address register 0 (PCI_CFH_BAR0)	32	R	0000_0000h
14h	PCI base address register 1 (PCI_CFH_BAR1)	32	R	0000_0000h
18h	PCI base address register 2 (PCI_CFH_BAR2)	32	R	0000_0000h
1Ch	PCI base address register 3 (PCI_CFH_BAR3)	32	R	0000_0000h
20h	PCI base address register 4 (PCI_CFH_BAR4)	32	R	0000_0000h
24h	PCI base address register 5 (PCI_CFH_BAR5)	32	R	0000_0000h
28h	PCI cardbus CIS register (PCI_CFH_CARDBUS_CIS)	32	R	0000_0000h
2Ch	PCI subsystem vendor ID register (PCI_CFH_SUBSYS_VID)	16	R	1957h
2Eh	PCI subsystem ID register (PCI_CFH_SUBSYS_ID)	16	R	EF00h
30h	PCI expansion ROM base address register (PCI_CFH_EXP_ROM_BA)	32	R	0000_0000h
34h	PCI capabilities pointer register (PCI_CFH_CAP_PTR)	8	R	40h
40h	PCI PCIe capabilities list register (PCI_CFC_PCIE_CAP_LIST)	16	R	8010h
42h	PCI PCIe capabilities register (PCI_CFC_PCIE_CAP)	16	R	0092h
44h	PCI PCIe device capabilities register (PCI_CFC_PCIE_DEV_CAP)	32	R	1000_0000h
48h	PCI PCIe device control register (PCI_CFC_PCIE_DEV_CTL)	16	RW	0000h
4Ah	PCI PCIe device status register (PCI_CFC_PCIE_DEV_STAT)	16	R	0000h
64h	PCI PCIe device capabilities 2 register (PCI_CFC_PCIE_DEV_CAP2)	32	R	0000_0000h
68h	PCI PCIe device control 2 register (PCI_CFC_PCIE_DEV_CTL2)	16	R	0000h
80h	PCI MSI-X capabilities list register (PCI_CFC_MSIX_CAP_LIST)	16	R	0011h
82h	PCI MSI-X message control register (PCI_CFC_MSIX_MSG_CTL)	16	RW	0000h
84h	PCI MSI-X table offset/BIR register (PCI_CFC_MSIX_TABLE_OFF_BIR)	32	R	0000_0003h
88h	PCI MSI-X PBA offset/BIR register (PCI_CFC_MSIX_PBA_OFF_BIR)	32	R	0000_0803h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
100h	PCIe AER extended capability header (PCIE_CFC_AER_EXT_CAP_HDR)	32	R	1301_0001h
104h	PCIe AER uncorrectable error status register (PCIE_CFC_AER_UCORR_ERR_STAT)	32	RW	0000_0000h
108h	PCIe AER uncorrectable error mask register (PCIE_CFC_AER_UCORR_ERR_MASK)	32	RW	0040_0000h
10Ch	PCIe AER uncorrectable error severity register (PCIE_CFC_AER_UCORR_ERR_SEV)	32	RW	0040_0000h
110h	PCIe AER correctable error status register (PCIE_CFC_AER_CORR_ERR_STAT)	32	RW	0000_0000h
114h	PCIe AER correctable error mask register (PCIE_CFC_AER_CORR_ERR_MASK)	32	RW	0000_4000h
118h	PCIe AER capabilities and control register (PCIE_CFC_AER_CAP_CTL)	32	R	0000_0000h
130h	PCIe ACS capability header (PCIE_CFC_ACS_CAP_HDR)	32	R	1401_000Dh
134h	PCIe ACS capability register (PCIE_CFC_ACS_CAP)	16	R	0000h
136h	PCIe ACS control register (PCIE_CFC_ACS_CTL)	16	R	0000h
140h	PCIe readiness time reporting capability header (PCIE_CFC_RTR_CAP_HDR)	32	R	0001_0022h
144h	PCIe RTR readiness time reporting 1 register (PCIE_CFC_RTR_RTR1)	32	R	8000_0000h
148h	PCIe RTR readiness time reporting 2 register (PCIE_CFC_RTR_RTR2)	32	R	0000_0634h

53.4.6.2.2 PCI device ID and vendor ID register (PCI_CFH_DID_VID)

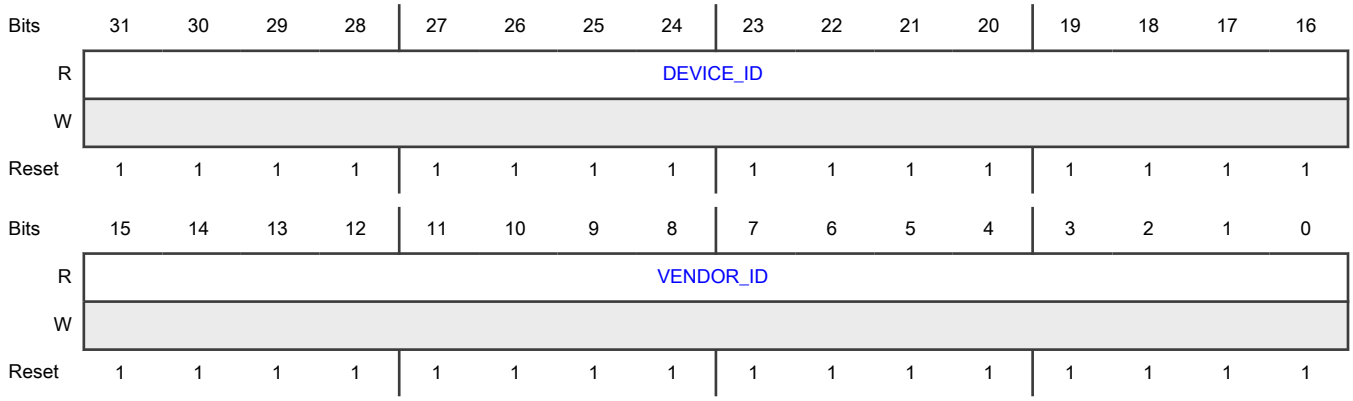
Offset

Register	Offset
PCI_CFH_DID_VID	0h

Function

This is the PCI config header device and vendor ID register.

Diagram



Fields

Field	Function
31-16 DEVICE_ID	Device ID This field in all VFs returns FFFFh when read. For SR-IOV the VF device ID is determined from the VF Device ID field in the SR-IOV capability register.
15-0 VENDOR_ID	Vendor ID Returns 0xFFFF for all VFs.

53.4.6.2.3 PCI command register (PCI_CFH_CMD)

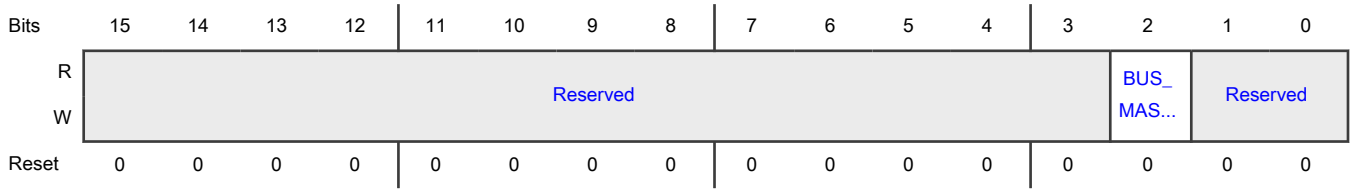
Offset

Register	Offset
PCI_CFH_CMD	4h

Function

This is the PCI config header command register.

Diagram



Fields

Field	Function
15-3 —	Reserved
2 BUS_MASTER_EN	Bus master enable Controls the ability of a PCI Express Endpoint to issue Memory and I/O Read/Write Requests. When this bit is Set, the PCI Express Function is allowed to issue Memory or I/O Requests. When this bit is Clear, the PCI Express Function is not allowed to issue any Memory or I/O Requests. Note that as MSI/MSI-X interrupt Messages are inband memory writes, setting the Bus Master Enable bit to 0b disables MSI/MSI-X interrupt Messages as well. Transactions for a VF that has its Bus Master Enable set must not be blocked by transactions for VFs that have their Bus Master Enable cleared.
1-0 —	Reserved

53.4.6.2.4 PCI status register (PCI_CFH_STAT)

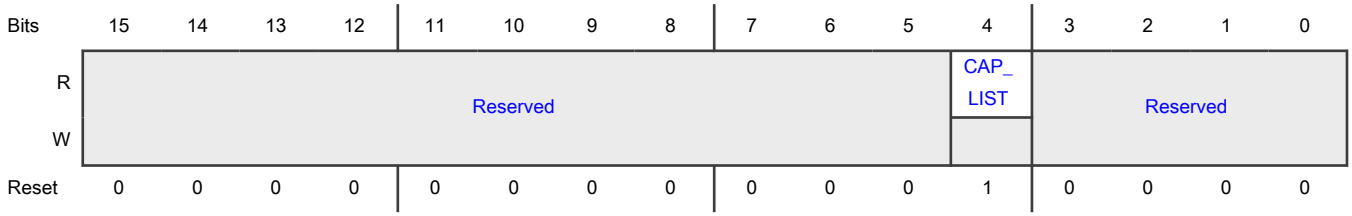
Offset

Register	Offset
PCI_CFH_STAT	6h

Function

This is the PCI config header status register.

Diagram



Fields

Field	Function
15-5 —	Reserved
4 CAP_LIST	Capabilities List Indicates the presence of an Extended Capability list item. Since all PCI Express device Functions are required to implement the PCI Express Capability structure, this bit must be hardwired to 1b.
3-0 —	Reserved

53.4.6.2.5 PCI revision ID and classcode register (PCI_CFH_REVID_CLASSCODE)

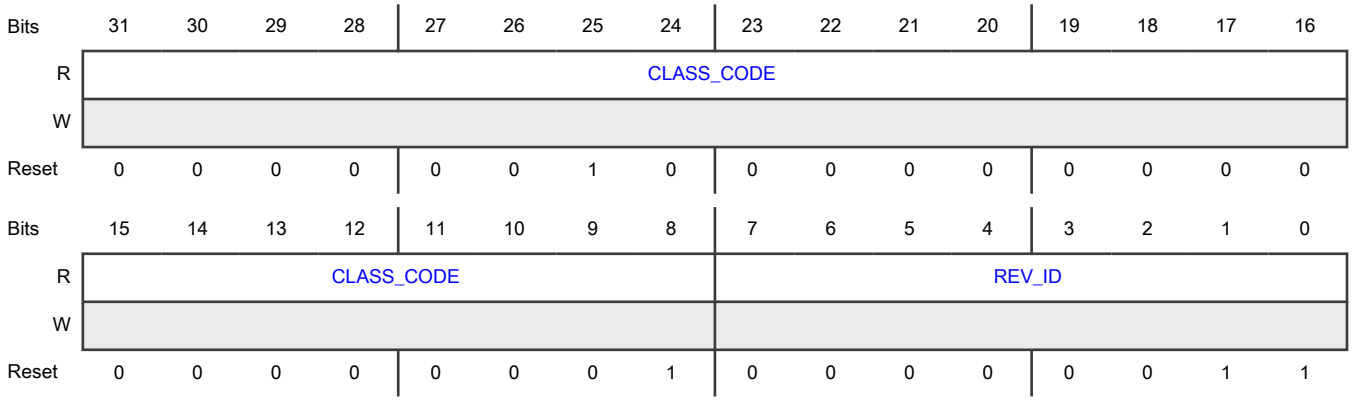
Offset

Register	Offset
PCI_CFH_REVID_CLASSCODE	8h

Function

This is the PCI config header revision ID and classcode register.

Diagram



Fields

Field	Function
31-8 CLASS_CODE	Class code Returns the same value as the PF.
7-0 REV_ID	Revision ID This register specifies a device specific revision identifier and is a vendor defined extension to the Device ID. This field will match IPBRR0[MAJOR] setting.

53.4.6.2.6 PCI cache line size register (PCI_CFH_CL_SIZE)

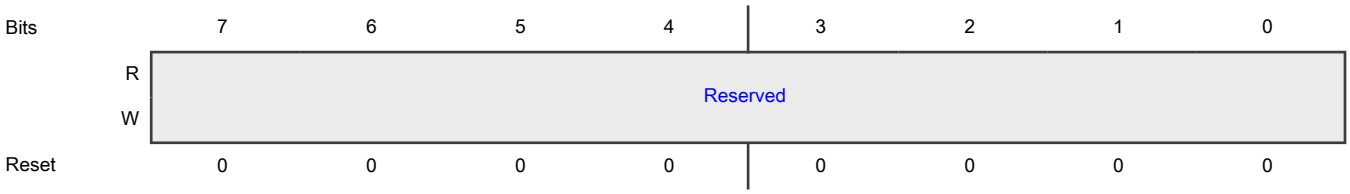
Offset

Register	Offset
PCI_CFH_CL_SIZE	Ch

Function

This is the PCI config header cache line size register.

Diagram



Fields

Field	Function
7-0 —	Cache line size For Virtual Functions, this field is reserved and hardwired to 0b.

53.4.6.2.7 PCI latency timer register (PCI_CFH_LAT_TIMER)

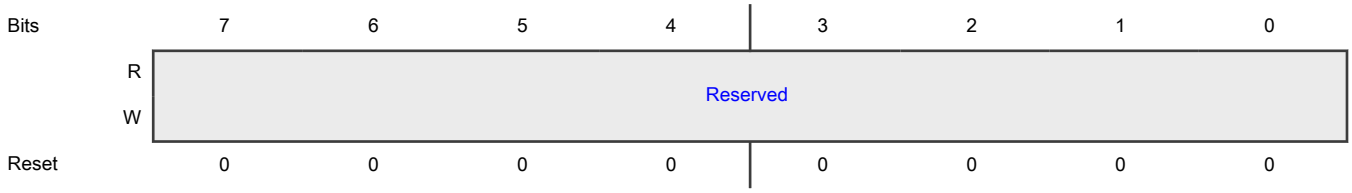
Offset

Register	Offset
PCI_CFH_LAT_TIMER	Dh

Function

This is the PCI config header latency timer register.

Diagram



Fields

Field	Function
7-0 —	Latency timer Does not apply to PCI Express. Hardwired to 00h.

53.4.6.2.8 PCI header type register (PCI_CFH_HDR_TYPE)

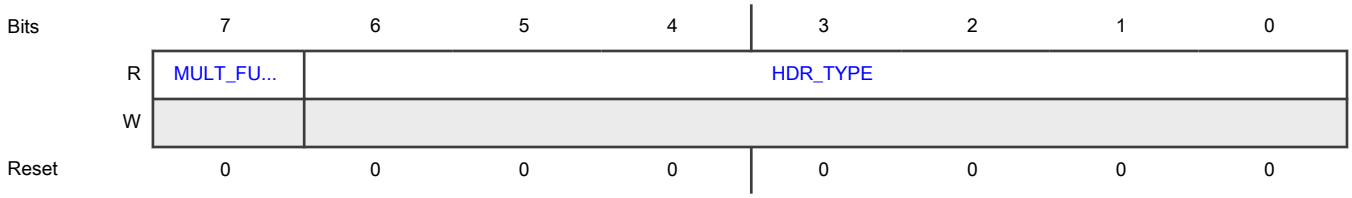
Offset

Register	Offset
PCI_CFH_HDR_TYPE	Eh

Function

This is the PCI config header type register.

Diagram



Fields

Field	Function
7 MULT_FUNC_DEV	Multi-function device For VFs this field is not applicable and is hardwired to 0b.
6-0 HDR_TYPE	Header type This field identifies the layout of the second part of the predefined header (beginning at byte 10h in Configuration Space). This register is hardwired to 00h (Type 0 header for both PFs and VFs).

53.4.6.2.9 PCI BIST register (PCI_CFH_BIST)

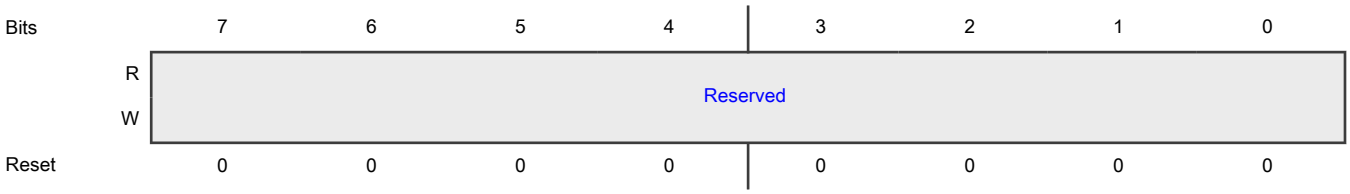
Offset

Register	Offset
PCI_CFH_BIST	Fh

Function

This is the PCI config header BIST register.

Diagram



Fields

Field	Function
7-0 —	Not supported. This register is hardwired to 00h.

53.4.6.2.10 PCI base address register 0 (PCI_CFH_BAR0)

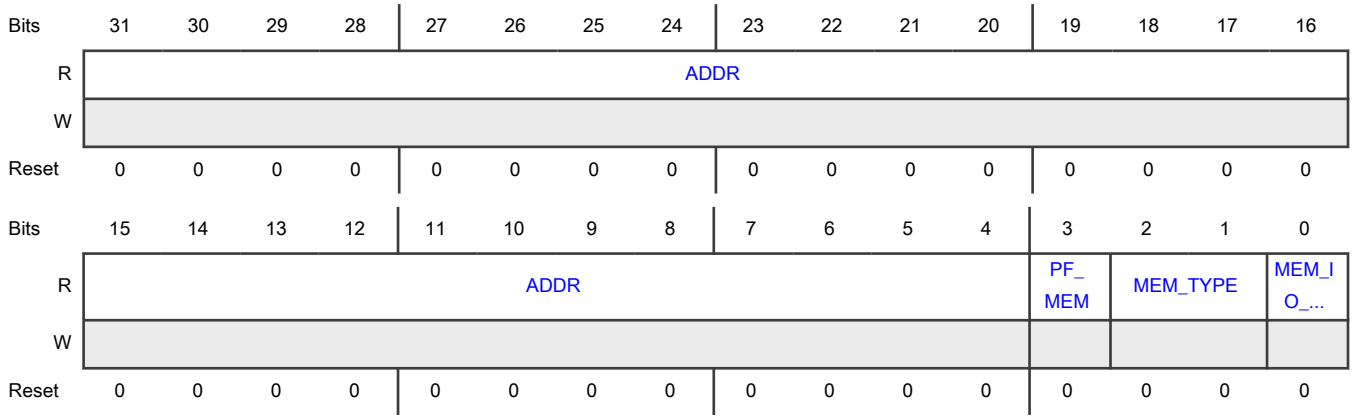
Offset

Register	Offset
PCI_CFH_BAR0	10h

Function

This is the PCI config header base address register 0.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.2.11 PCI base address register 1 (PCI_CFH_BAR1)

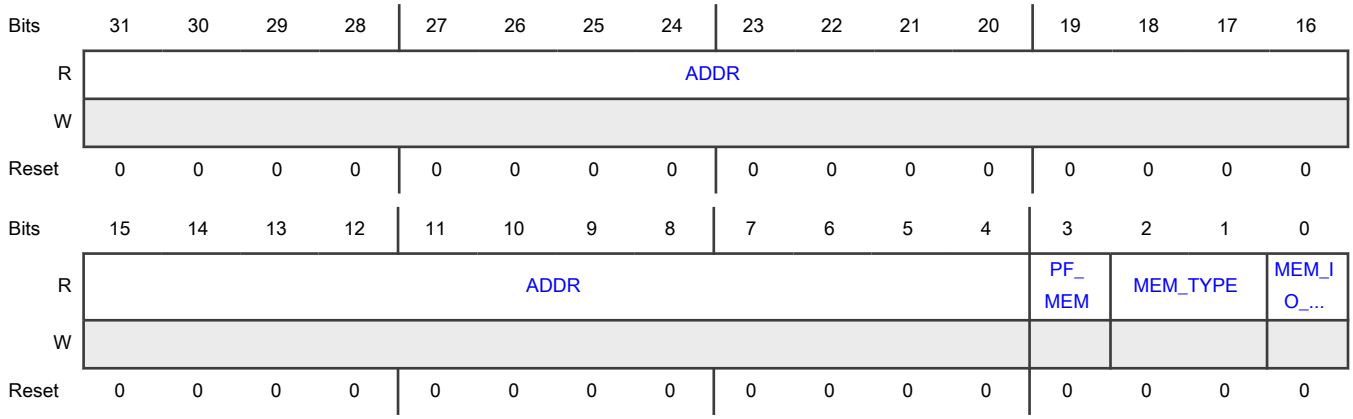
Offset

Register	Offset
PCI_CFH_BAR1	14h

Function

This is the PCI config header base address register 1.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.2.12 PCI base address register 2 (PCI_CFH_BAR2)

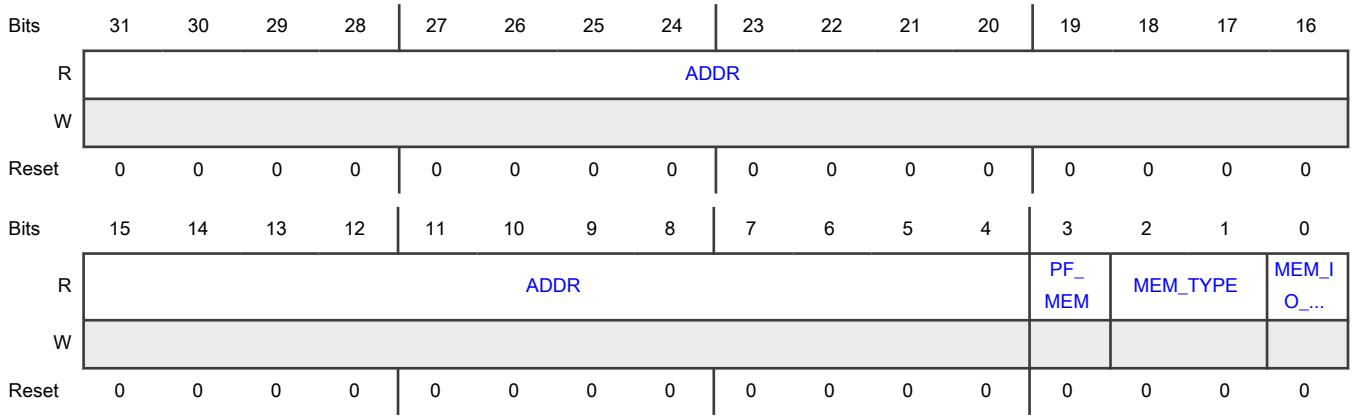
Offset

Register	Offset
PCI_CFH_BAR2	18h

Function

This is the PCI config header base address register 2.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.2.13 PCI base address register 3 (PCI_CFH_BAR3)

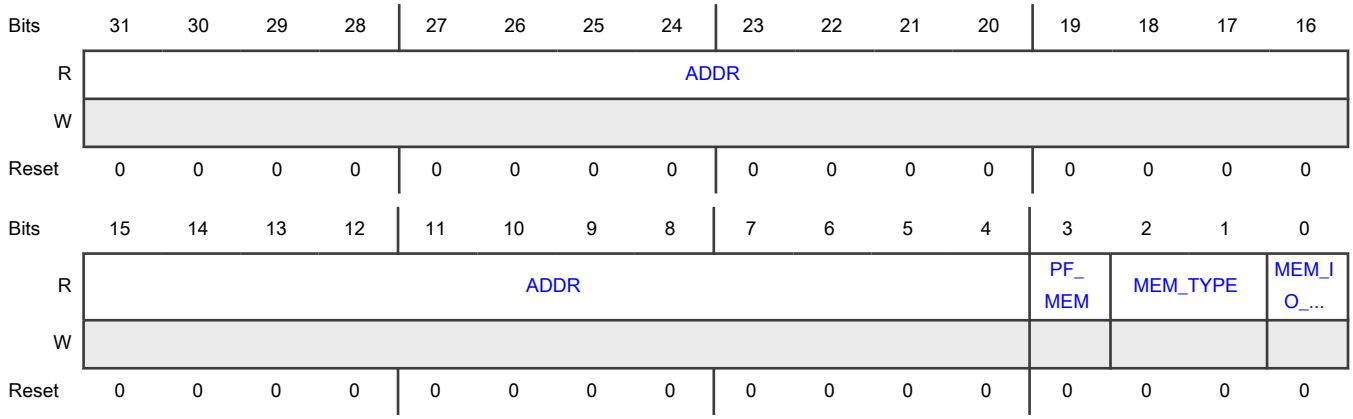
Offset

Register	Offset
PCI_CFH_BAR3	1Ch

Function

This is the PCI config header base address register 3.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.2.14 PCI base address register 4 (PCI_CFH_BAR4)

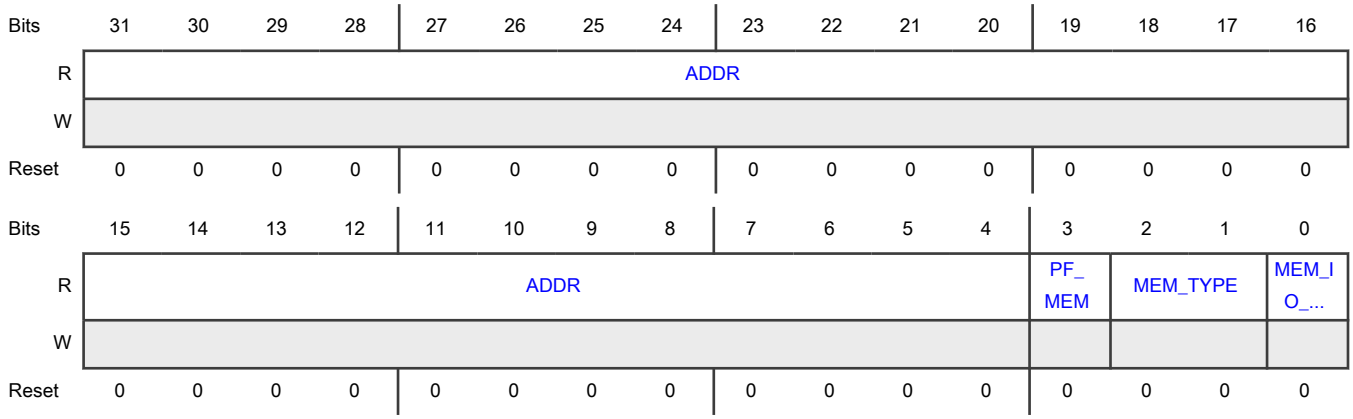
Offset

Register	Offset
PCI_CFH_BAR4	20h

Function

This is the PCI config header base address register 4.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.2.15 PCI base address register 5 (PCI_CFH_BAR5)

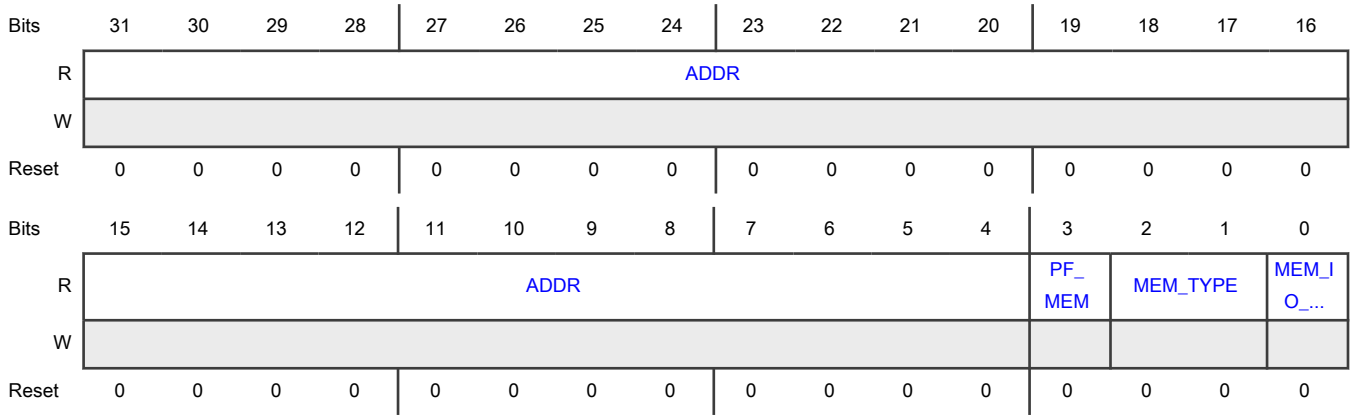
Offset

Register	Offset
PCI_CFH_BAR5	24h

Function

This is the PCI config header base address register 5.

Diagram



Fields

Field	Function
31-4 ADDR	EA BARs used. This register is hardwired to 000000h.
3 PF_MEM	EA BARs used. This register is hardwired to 0h.
2-1 MEM_TYPE	EA BARs used. This register is hardwired to 0h.
0 MEM_IO_IND	EA BARs used. This register is hardwired to 0h.

53.4.6.2.16 PCI cardbus CIS register (PCI_CFH_CARDBUS_CIS)

Offset

Register	Offset
PCI_CFH_CARDBUS_CIS	28h

Function

This is the PCI config header cardbus CSI register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CARDBUS_CIS_PTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CARDBUS_CIS_PTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CARDBUS_CIS_PTR	Not applicable. This register is hardwired to 0000_0000h.

53.4.6.2.17 PCI subsystem vendor ID register (PCI_CFH_SUBSYS_VID)

Offset

Register	Offset
PCI_CFH_SUBSYS_VID	2Ch

Function

This is the PCI config header subsystem vendor ID register. The value read is taken from IERB register E*a*_CFH_SIDSVID.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SYBSYSTEM_VENDOR_ID															
W																
Reset	0	0	0	1	1	0	0	1	0	1	0	1	0	1	1	1

Fields

Field	Function
15-0 SYBSYSTEM_VENDOR_ID	This Read Only field identifies the manufacturer of the subsystem. The field in the PF and associated VFs must return the same value when read.

53.4.6.2.18 PCI subsystem ID register (PCI_CFH_SUBSYS_ID)

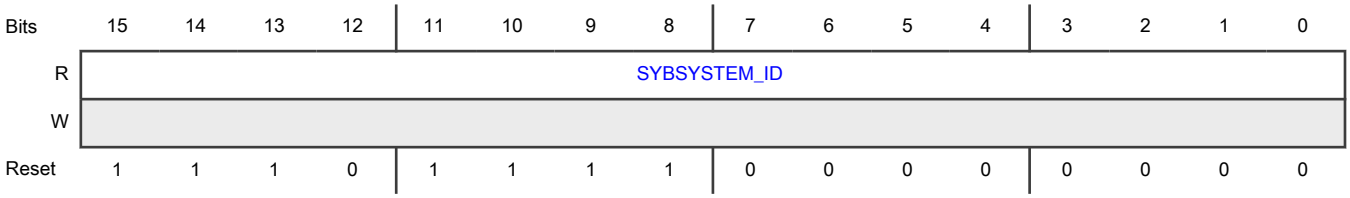
Offset

Register	Offset
PCI_CFH_SUBSYS_ID	2Eh

Function

This is the PCI config header subsystem ID register. The value read is taken from IERB register E*a*_CFC_VFDID.

Diagram



Fields

Field	Function
15-0 SYBSYSTEM_ID	This Read Only field identifies the particular subsystem. The Device may have a different value in the PF and the VF.

53.4.6.2.19 PCI expansion ROM base address register (PCI_CFH_EXP_ROM_BA)

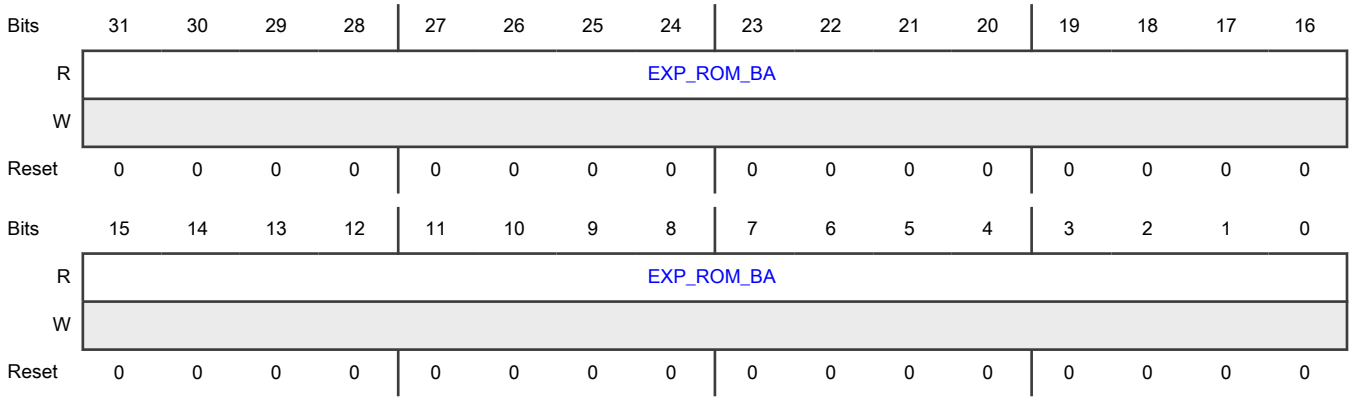
Offset

Register	Offset
PCI_CFH_EXP_ROM_BA	30h

Function

This is the PCI config header expansion ROM base address register.

Diagram



Fields

Field	Function
31-0 EXP_ROM_BA	Not applicable. This register is hardwired to 0000_0000h.

53.4.6.2.20 PCI capabilities pointer register (PCI_CFH_CAP_PTR)

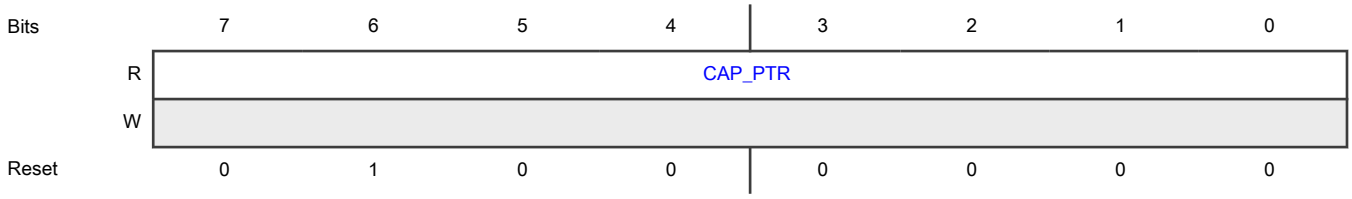
Offset

Register	Offset
PCI_CFH_CAP_PTR	34h

Function

This is the PCI config header capabilities pointer register.

Diagram



Fields

Field	Function
7-0 CAP_PTR	This register is used to point to a linked list of new capabilities implemented by this device. This register is only valid if the “Capabilities List” bit in the Status Register is set. If implemented, the bottom two bits are reserved and should be set to 00b. Software should mask these bits off before using this register as a pointer in Configuration Space to the first entry of a linked list of new capabilities. Hardwired to 40h.

53.4.6.2.21 PCI PCIe capabilities list register (PCI_CFC_PCIE_CAP_LIST)

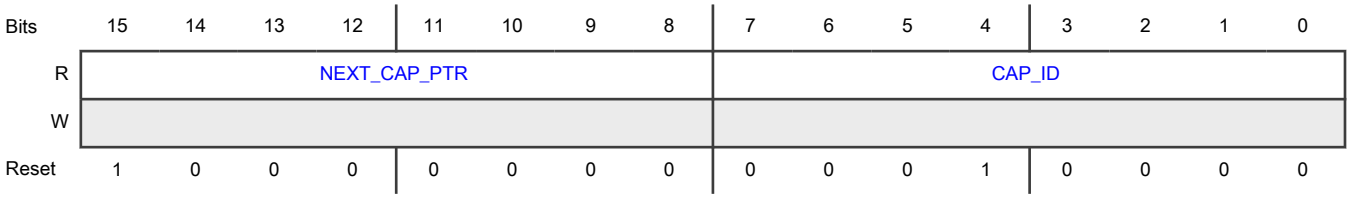
Offset

Register	Offset
PCI_CFC_PCIE_CAP_LIST	40h

Function

This is the PCI config capability PCIe capabilities list register.

Diagram



Fields

Field	Function
15-8 NEXT_CAP_PTR	This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of capabilities. Hardwired to 80h.
7-0 CAP_ID	Indicates the PCI Express Capability structure. Hardwired to 10h.

53.4.6.2.22 PCI PCIe capabilities register (PCI_CFC_PCIE_CAP)

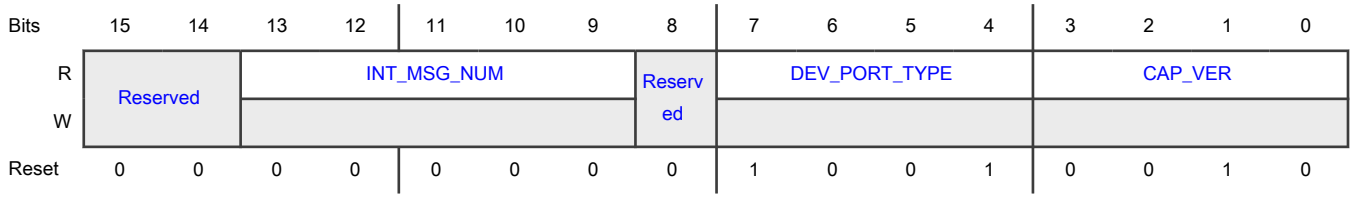
Offset

Register	Offset
PCI_CFC_PCIE_CAP	42h

Function

This is the PCI config capability PCIe capabilities register.

Diagram



Fields

Field	Function
15-14 —	Reserved
13-9 INT_MSG_NUM	Interrupt message number This field indicates which MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of this capability structure. For MSI-X, the value in this field indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. If both MSI and MSI-X are implemented, they are permitted to use different vectors, though software is permitted to enable only one mechanism at a time. If MSI-X is enabled, the value in this field must indicate the vector for MSI-X. If MSI is enabled or neither is enabled, the value in this field must indicate the vector for MSI. If software enables both MSI and MSI-X at the same time, the value in this field is undefined. Errors for ENETC are handled through the Event Collector. Hardwired to 00h.
8 —	Reserved
7-4 DEV_PORT_TYPE	Device/Port type Indicates the specific type of this PCI Express Function. Hardwired to 9h to indicate Root Complex Integrated Endpoint.
3-0 CAP_VER	Capability Version Indicates PCI-SIG defined PCI Express Capability structure version number. Hardwired to 2h.

53.4.6.2.23 PCI PCIe device capabilities register (PCI_CFC_PCIE_DEV_CAP)

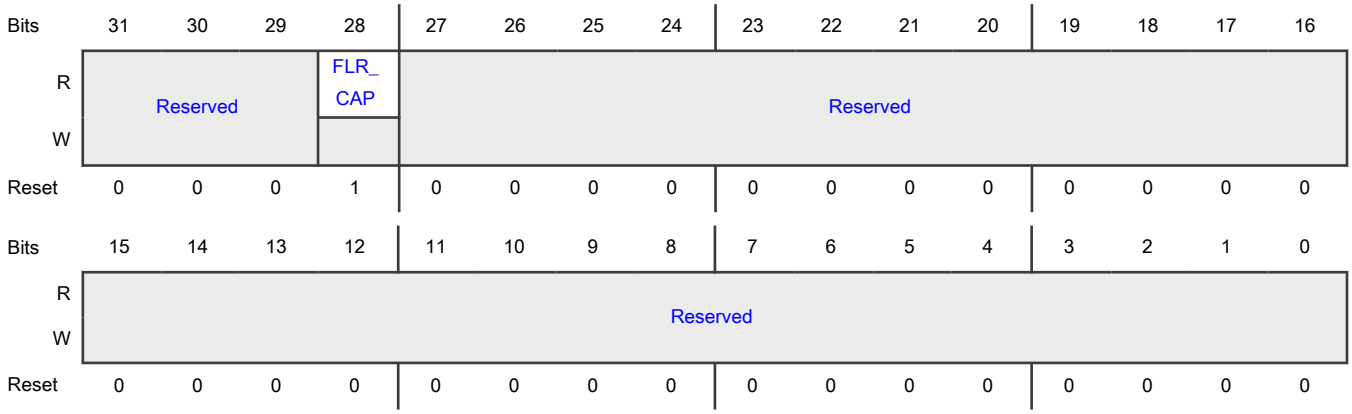
Offset

Register	Offset
PCI_CFC_PCIE_DEV_CAP	44h

Function

This is the PCI config capability PCIe device capabilities register.

Diagram



Fields

Field	Function
31-29 —	Reserved
28 FLR_CAP	Function level reset capability A value of 1b indicates the function supports the optional Function Level Reset (FLR) mechanism. Hardwired to 1b.
27-0 —	Reserved

53.4.6.2.24 PCI PCIe device control register (PCI_CFC_PCIE_DEV_CTL)

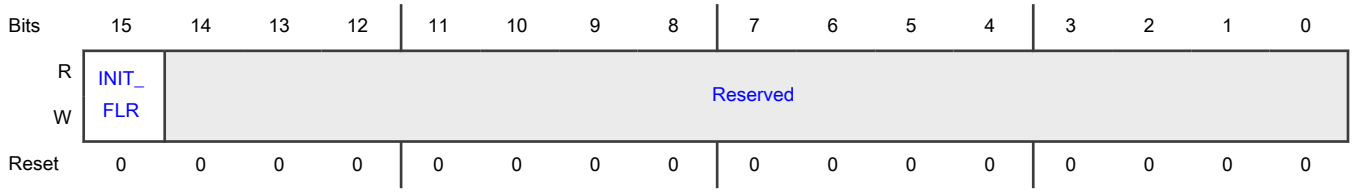
Offset

Register	Offset
PCI_CFC_PCIE_DEV_CTL	48h

Function

This is the PCI config capability PCIe device control register.

Diagram



Fields

Field	Function
15 INIT_FLR	Initiate function level reset A write of b1 initiates Function Level Reset (FLR). The bit will self-clear when the reset sequence is complete and function can be re-configured by software.
14-0 —	Reserved

53.4.6.2.25 PCI PCIe device status register (PCI_CFC_PCIE_DEV_STAT)

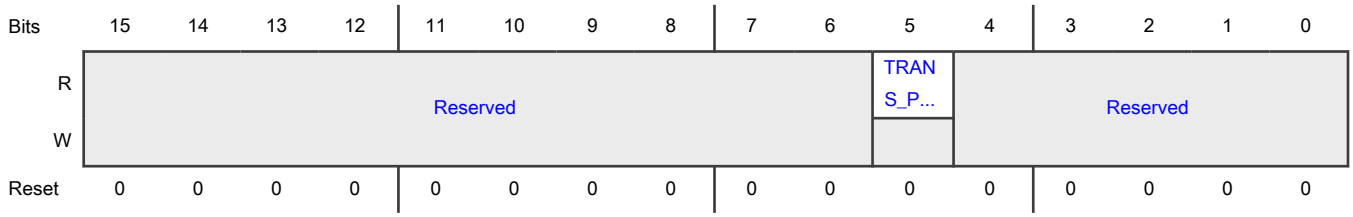
Offset

Register	Offset
PCI_CFC_PCIE_DEV_STAT	4Ah

Function

This is the PCI config capability PCIe device status register.

Diagram



Fields

Field	Function
15-6 —	Reserved
5	Transaction pending

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRANS_PEND	If set indicates that the Function has outstanding transactions on its external master interface.
4-0 —	Reserved

53.4.6.2.26 PCI PCIe device capabilities 2 register (PCI_CFC_PCIE_DEV_CAP2)

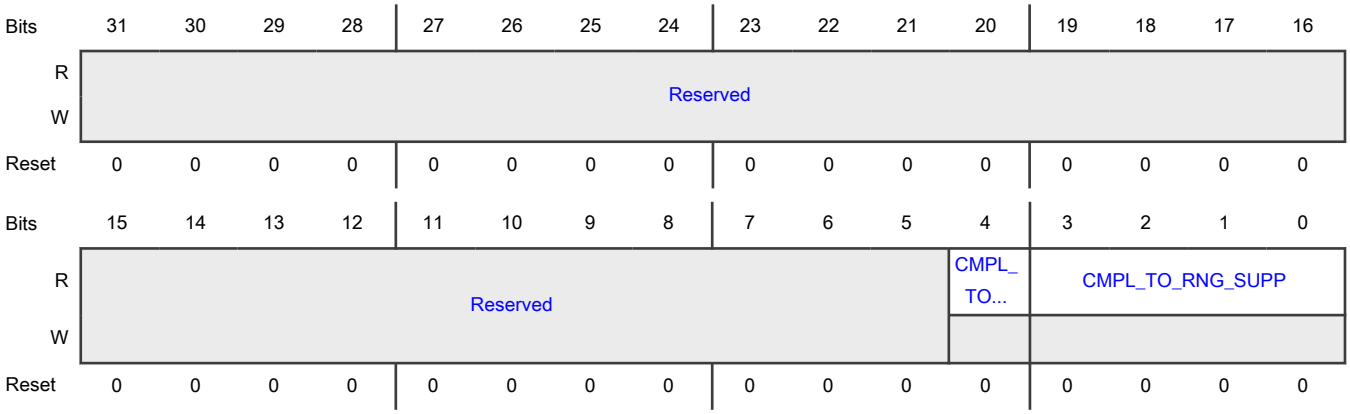
Offset

Register	Offset
PCI_CFC_PCIE_DEV_CAP2	64h

Function

This is the PCI config capability PCIe device capabilities 2 register.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 CMPL_TO_DIS_SUPP	Completion Timeout Disable Supported When set (b1) indicates support for the Completion Timeout Disable mechanism.
3-0	Completion Timeout Ranges Supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
CMPL_TO_RNG_SUPP	Completion Timeout programming not supported, the Function assumes a timeout value in the range 50 µs to 50 ms.

53.4.6.2.27 PCI PCIe device control 2 register (PCI_CFC_PCIE_DEV_CTL2)

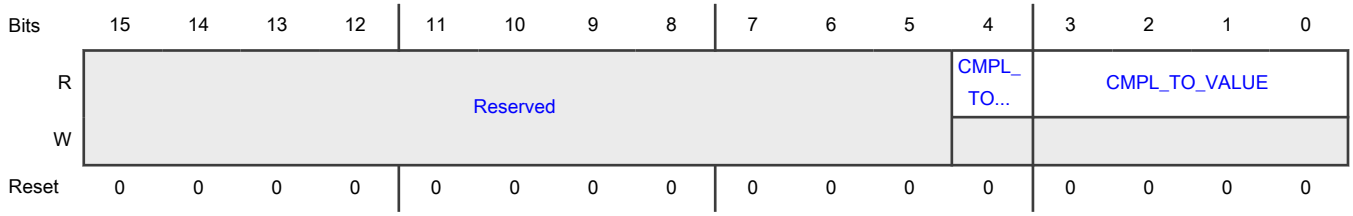
Offset

Register	Offset
PCI_CFC_PCIE_DEV_CTL2	68h

Function

This is the PCI config capability PCIe device control 2 register.

Diagram



Fields

Field	Function
15-5 —	Reserved
4 CMPL_TO_EN	Completion Timeout Enable Not supported. Hardwired to b0.
3-0 CMPL_TO_VALUE	Completion Timeout Value Completion Timeout programming not supported – the Function assumes a timeout value in the range 50 µs to 50 ms. Hardwired to b0000.

53.4.6.2.28 PCI MSI-X capabilities list register (PCI_CFC_MSIX_CAP_LIST)

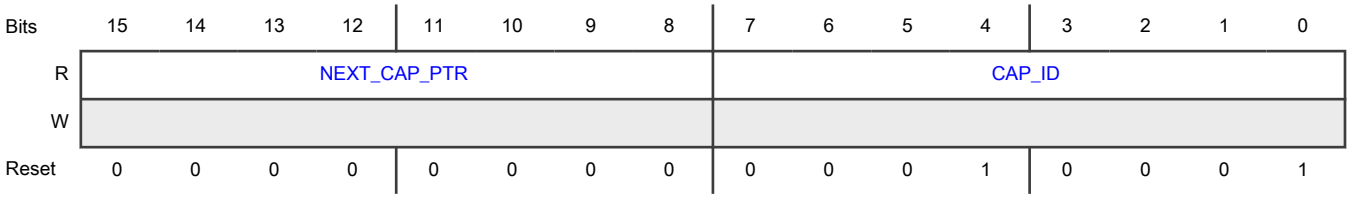
Offset

Register	Offset
PCI_CFC_MSIX_CAP_LIST	80h

Function

This is the PCI config capability MSI-X capabilities list register.

Diagram



Fields

Field	Function
15-8 NEXT_CAP_PTR	This field contains the offset to the next PCI Capability structure or 00h if no other items exist in the linked list of capabilities. Hardwired to 00h.
7-0 CAP_ID	Indicates the MSI-X Capability structure. Hardwired to 11h.

53.4.6.2.29 PCI MSI-X message control register (PCI_CFC_MSIX_MSG_CTL)

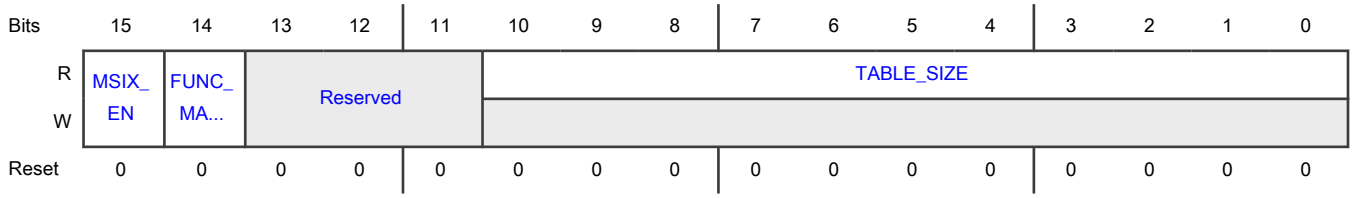
Offset

Register	Offset
PCI_CFC_MSIX_MSG_CTL	82h

Function

This is the PCI config capability MSI-X message control register.

Diagram



Fields

Field	Function
15 MSIX_EN	MSI-X enable If set, the function is permitted to use MSI-X to request service. System configuration software sets this bit to enable MSI-X. A device driver is prohibited from writing this bit to mask a function's service request. If 0, the function is prohibited from using MSI-X to request service. This bit's state after reset is 0 (MSI-X is disabled).
14 FUNC_MASK	Function mask If 1, all of the vectors associated with the function are masked, regardless of their per-vector Mask bit states. If 0, each vector's Mask bit determines whether the vector is masked or not. Setting or clearing the MSI-X Function Mask bit has no effect on the state of the per-vector Mask bits. This bit's state after reset is 0 (unmasked).
13-11 —	Reserved
10-0 TABLE_SIZE	Table size System software reads this field to determine the MSI-X Table Size N, which is encoded as N-1. For example, a returned value of "00000000011" indicates a table size of 4. **NOTE** Table size is defined by ENETC register PSI*a*CFGR2[NUM_MSIX], mapped as VF*a*, and is immediately reflected here.

53.4.6.2.30 PCI MSI-X table offset/BIR register (PCI_CFC_MSIX_TABLE_OFF_BIR)

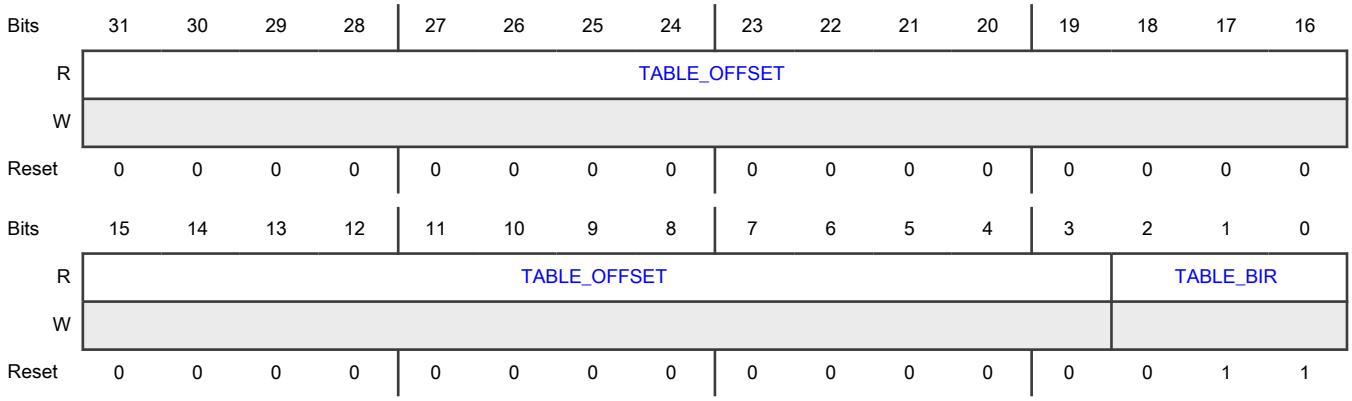
Offset

Register	Offset
PCI_CFC_MSIX_TABLE_OFF_BIR	84h

Function

This is the PCI config capability MSI-X table offset/BIR register.

Diagram



Fields

Field	Function
31-3 TABLE_OFFSET	Table Offset Used as an offset from the address contained by one of the function's Base Address registers to point to the base of the MSI-X Table. The lower 3 Table BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset.
2-0 TABLE_BIR	Table BIR Indicates which entry in the Enhanced Allocation capability with a matching BEI, is used to map the Function's MSI-X Table into Memory Space. For a 64-bit Base Address register, the Table BIR indicates the lower DWORD. Virtual functions hardwires this field to 011b (PE entry 3) for MSI-X table.

53.4.6.2.31 PCI MSI-X PBA offset/BIR register (PCI_CFC_MSIX_PBA_OFF_BIR)

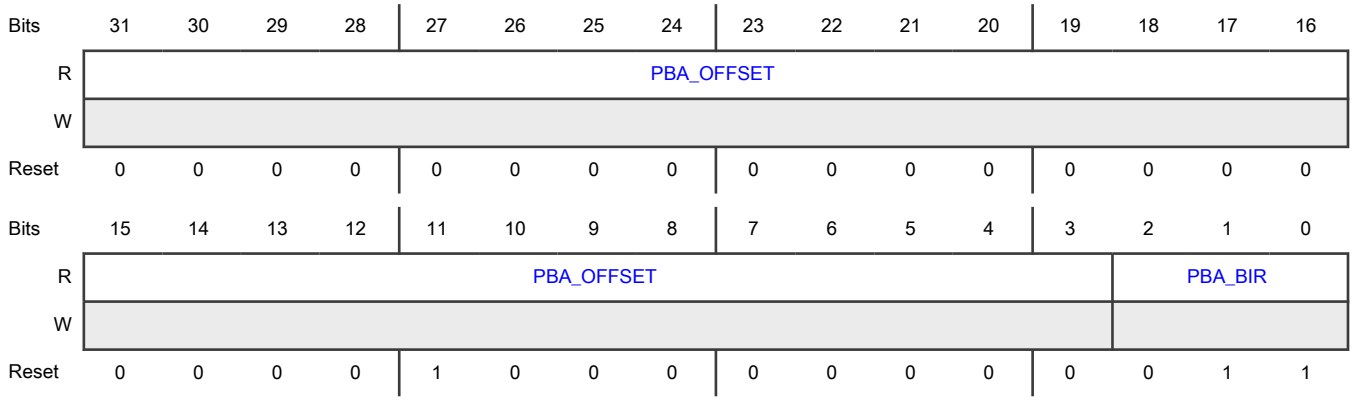
Offset

Register	Offset
PCI_CFC_MSIX_PBA_OFF_BIR	88h

Function

This is the PCI config capability MSI-X PBA offset/BIR register.

Diagram



Fields

Field	Function
31-3 PBA_OFFSET	PBA Offset Used as an offset from the address contained by one of the function's Base Address registers to point to the base of the MSI-X PBA. The lower 3 PBA BIR bits are masked off (set to zero) by software to form a 32-bit QWORD-aligned offset. The PBA structure directly follows the maximum possible MSI-X table size (2KB).
2-0 PBA_BIR	PBA BIR Indicates which entry in the Enhanced Allocation capability with a matching BEI, is used to map the Function's MSI-X PBA into Memory Space. For a 64-bit Base Address register, the PBA BIR indicates the lower DWORD. Virtual function hardwires this field to 011b (PE entry 3) for MSI-X PBA.

53.4.6.2.32 PCIe AER extended capability header (PCIE_CFC_AER_EXT_CAP_HDR)

Offset

Register	Offset
PCIE_CFC_AER_EXT_CAP_HDR	100h

Function

This is the PCIe config capability Advanced Error Reporting extended capability header.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NEXT_CAP_OFF												CAP_VER			
W																
Reset	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCIE_EXT_CAP_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-20 NEXT_CAP_OFF	Next capability offset This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of capabilities.
19-16 CAP_VER	Capability version Must be 1h for this version of the specification.
15-0 PCIE_EXT_CAP_ID	The Extended Capability ID for the AER Extended Capability is 0001h.

53.4.6.2.33 PCIe AER uncorrectable error status register (PCIE_CFC_AER_UCORR_ERR_STAT)

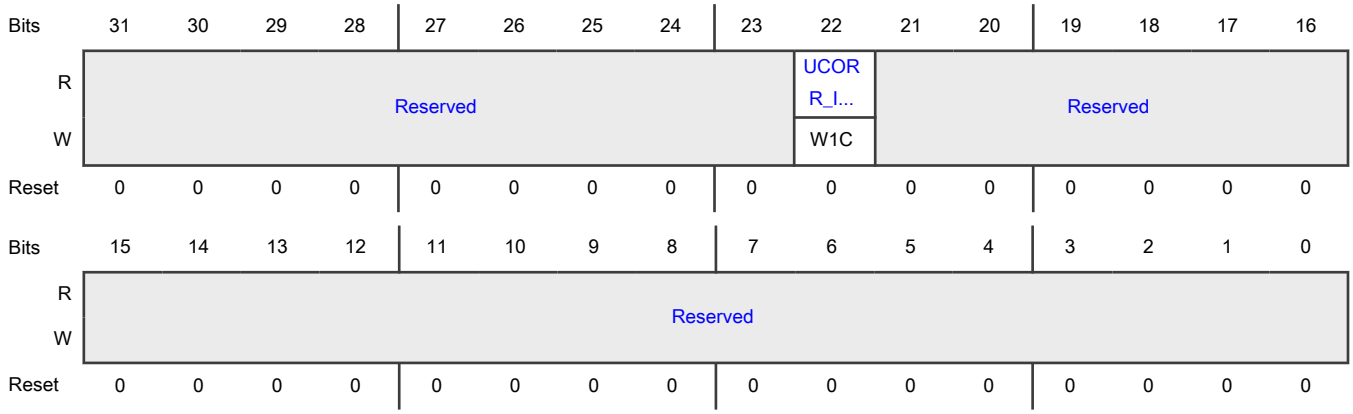
Offset

Register	Offset
PCIE_CFC_AER_UCORR_ERR_STAT	104h

Function

This is the PCIe config capability Advanced Error Reporting uncorrectable error status register. The Uncorrectable Error Status register indicates error detection status of individual errors on a PCI Express device Function. An individual error status bit that is set indicates that a particular error was detected; software may clear an error status by writing a 1b to the respective bit.

Diagram



Fields

Field	Function
31-23 —	Reserved
22 UCORR_INT_ERR	Uncorrectable internal error Set to indicate some uncorrectable internal error such as a multi-bit error in an internal memory has occurred. **NOTE** This bit is “sticky” and will be preserved across function level resets.
21-0 —	Reserved

53.4.6.2.34 PCIe AER uncorrectable error mask register (PCIE_CFC_AER_UCORR_ERR_MASK)

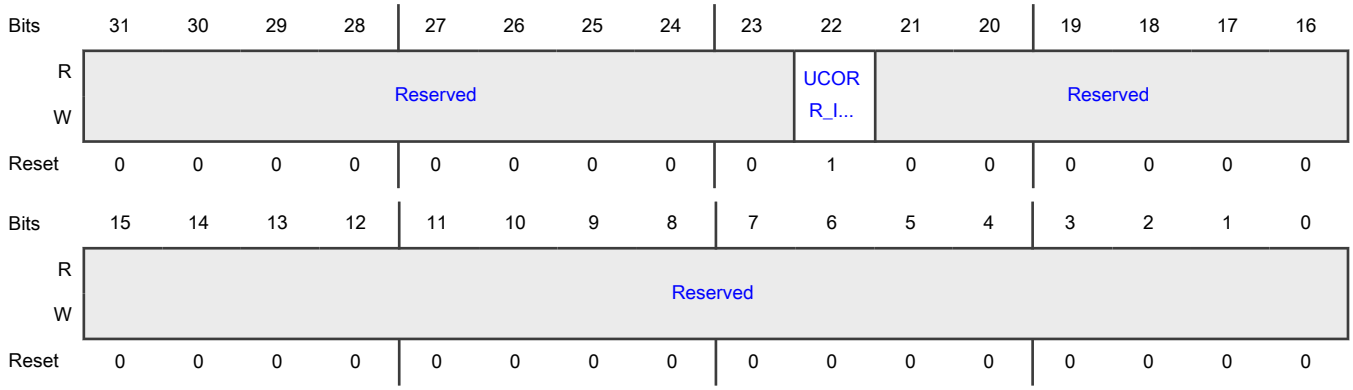
Offset

Register	Offset
PCIE_CFC_AER_UCORR_ERR_MASK	108h

Function

This is the PCIe config capability Advanced Error Reporting uncorrectable error mask register. The Uncorrectable Error Mask register controls reporting of individual errors by the device Function to the PCI Express Root Complex Event Collector via a PCI Express error Message. A masked error (respective bit set in the mask register) is not recorded or reported in the Header Log or First Error Pointer, and is not reported to the PCI Express Root Complex Event Collector by this Function. There is a mask bit per error bit of the Uncorrectable Error Status register.

Diagram



Fields

Field	Function
31-23 —	Reserved
22 UCORR_INT_ERR_MASK	Uncorrectable internal error mask 0 Not masked 1 Masked **NOTE** This bit is “sticky” and will be preserved across function level resets.
21-0 —	Reserved

53.4.6.2.35 PCIe AER uncorrectable error severity register (PCIE_CFC_AER_UCORR_ERR_SEV)

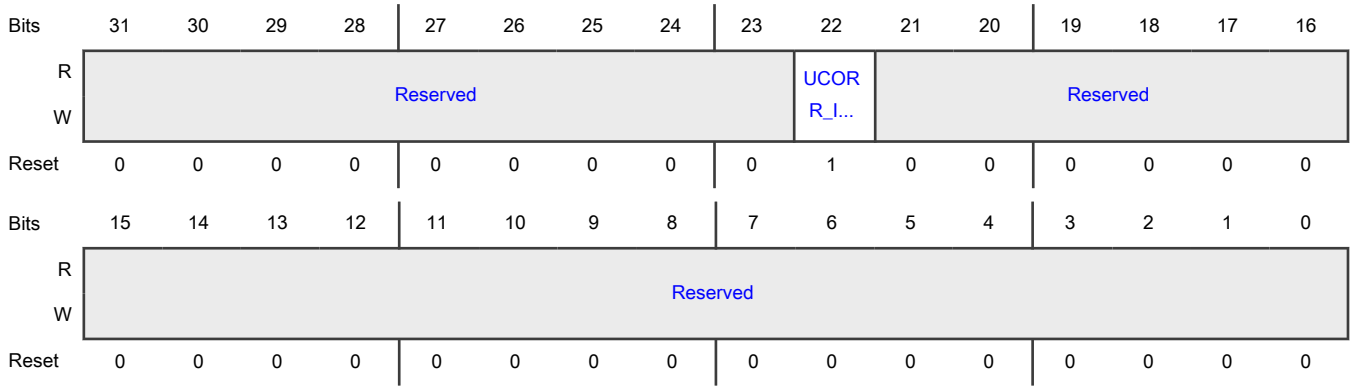
Offset

Register	Offset
PCIE_CFC_AER_UCORR_ERR_SEV	10Ch

Function

This is the PCIe config capability Advanced Error Reporting uncorrectable error severity register. The Uncorrectable Error Severity register controls whether an individual error is reported as a Non-fatal or Fatal error. An error is reported as fatal when the corresponding error bit in the severity register is Set. If the bit is Clear, the corresponding error is considered non-fatal.

Diagram



Fields

Field	Function
31-23 —	Reserved
22 UCORR_INT_SEV	Uncorrectable internal severity 0 Non-fatal 1 Fatal **NOTE** This bit is “sticky” and will be preserved across function level resets.
21-0 —	Reserved

53.4.6.2.36 PCIe AER correctable error status register (PCIE_CFC_AER_CORR_ERR_STAT)

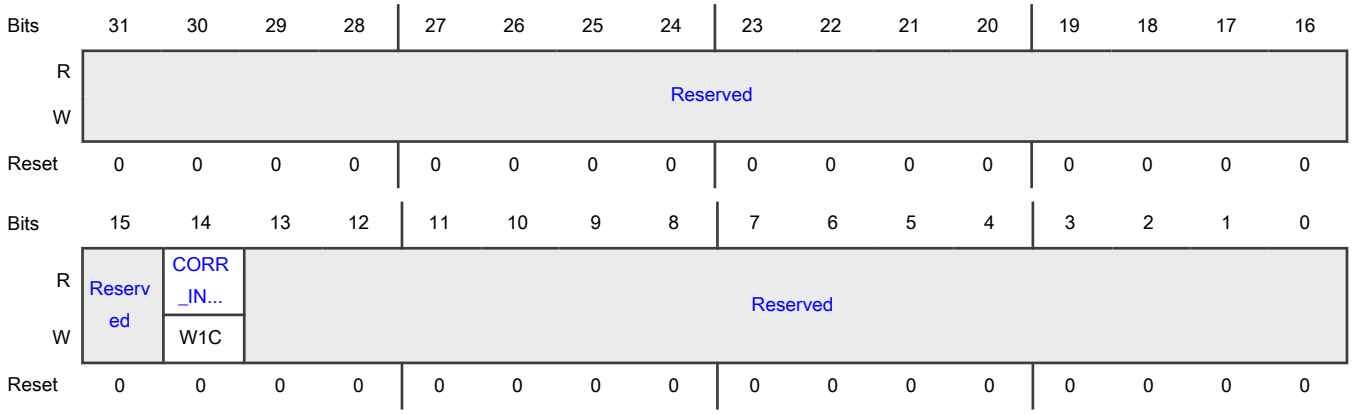
Offset

Register	Offset
PCIE_CFC_AER_CORR_ERR_STAT	110h

Function

This is the PCIe config capability Advanced Error Reporting correctable error status register. The Correctable Error Status register reports error status of individual correctable error sources on a PCI Express device Function. When an individual error status bit is Set, it indicates that a particular error occurred; software may clear an error status by writing a 1b to the respective bit.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 CORR_INT_ERR	Correctable internal error Set to indicate some correctable internal error has occurred such as a single bit error in an internal memory. **NOTE** This bit is “sticky” and will be preserved across function level resets.
13-0 —	Reserved

53.4.6.2.37 PCIe AER correctable error mask register (PCIE_CFC_AER_CORR_ERR_MASK)

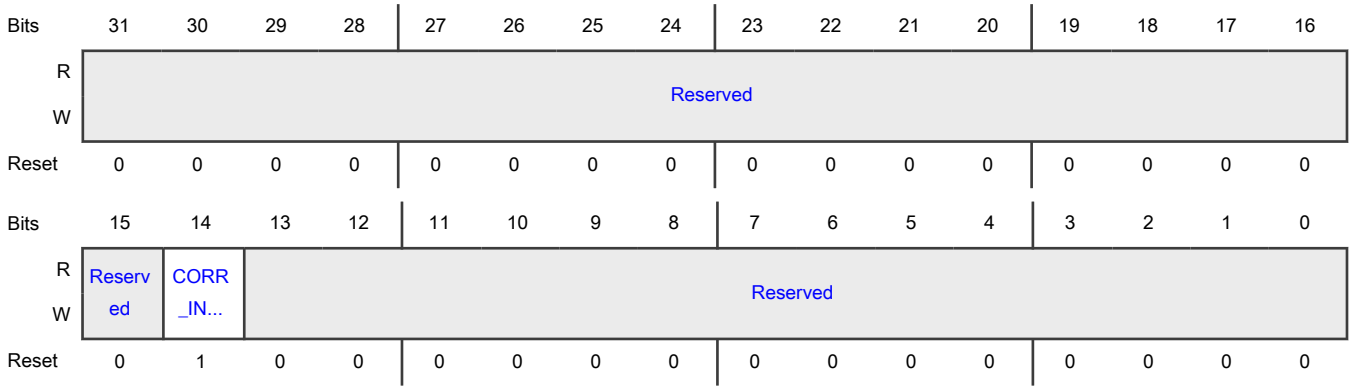
Offset

Register	Offset
PCIE_CFC_AER_CORR_ERR_MASK	114h

Function

This is the PCIe config capability Advanced Error Reporting correctable error mask register. The Correctable Error Mask register controls reporting of individual correctable errors by this Function to the PCI Express Root Complex via a PCI Express error Message. A masked error (respective bit set in the mask register) is not reported to the PCI Express Root Complex by this Function. There is a mask bit per error bit in the Correctable Error Status register.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 CORR_INT_MASK	Correctable internal error mask 0 Not masked 1 Masked **NOTE** This bit is “sticky” and will be preserved across function level resets.
13-0 —	Reserved

53.4.6.2.38 PCIe AER capabilities and control register (PCIE_CFC_AER_CAP_CTL)

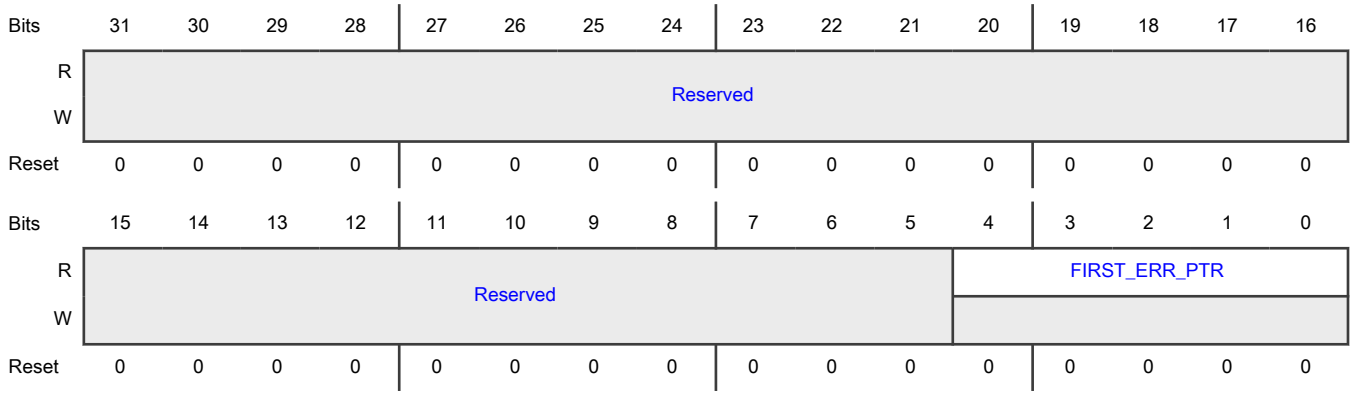
Offset

Register	Offset
PCIE_CFC_AER_CAP_CTL	118h

Function

This is the PCIe config capability Advanced Error Reporting capabilities and control register.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 FIRST_ERR_PTR	First error pointer The First Error Pointer is a field that identifies the bit position of the first error reported in the Uncorrectable Error Status register.

53.4.6.2.39 PCIe ACS capability header (PCIE_CFC_ACS_CAP_HDR)

Offset

Register	Offset
PCIE_CFC_ACS_CAP_HDR	130h

Function

This is the PCIe config capability Access Control Services (ACS) capability header.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NEXT_CAP_OFF												CAP_VER			
W																
Reset	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCIE_EXT_CAP_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1

Fields

Field	Function
31-20 NEXT_CAP_OFF	Next capability offset This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of capabilities.
19-16 CAP_VER	Capability version Must be 1h for this version of the specification.
15-0 PCIE_EXT_CAP_ID	The Extended Capability ID for the ACS Extended Capability is 000Dh.

53.4.6.2.40 PCIe ACS capability register (PCIE_CFC_ACS_CAP)

Offset

Register	Offset
PCIE_CFC_ACS_CAP	134h

Function

This is the PCIe config capability Access Control Services (ACS) capability register.

Diagram

Fields

Field	Function
15-7 —	Reserved
6 ACS_DIR_TRANS_P2P	ACS direct translated P2P (T) Set to indicate that the Function supports direct translated peer-to-peer.
5-3 —	Reserved
2 ACS_P2P_REQ_REDIR	ACS P2P request redirect (R) Set to indicate that the Function supports peer-to-peer request redirection.
1 ACS_TRANS_BLOCK	ACS translation blocking (B) Set to indicate that the Function supports Translation Blocking.
0 —	Reserved

53.4.6.2.41 PCIe ACS control register (PCIE_CFC_ACS_CTL)

Offset

Register	Offset
PCIE_CFC_ACS_CTL	136h

Function

This is the PCIe config capability Access Control Services (ACS) control register.

Diagram

Fields

Field	Function
15-7 —	Reserved
6 ACS_DIR_TRANS_P2P_EN	ACS direct translated P2P enable (T) When set the Function determines how to direct peer-to-peer requests based on ATS Ignored if ACS Translation Blocking Enable is set. This bit is read-only if the functionality is not implemented.
5-3 —	Reserved
2 ACS_P2P_REQ_REDIR_EN	ACS P2P request redirect enable (R) When set the Function directs peer-to-peer requests to the SoC interconnect. This bit is read-only if the functionality is not implemented.
1 ACS_TRANS_BLOCK_EN	ACS translation blocking enable (B) When set the Function directs peer-to-peer requests to the SoC interconnect. This bit is read-only if the functionality is not implemented.
0 —	Reserved

53.4.6.2.42 PCIe readiness time reporting capability header (PCIE_CFC_RTR_CAP_HDR)

Offset

Register	Offset
PCIE_CFC_RTR_CAP_HDR	140h

Function

This is the PCIe config capability readiness time reporting capability header.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NEXT_CAP_OFF												CAP_VER			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCIE_EXT_CAP_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0

Fields

Field	Function
31-20 NEXT_CAP_OFF	Next capability offset This field contains the offset to the next PCI Express Capability structure or 000h if no other items exist in the linked list of capabilities.
19-16 CAP_VER	Capability version Must be 1h for this version of the specification.
15-0 PCIE_EXT_CAP_ID	The Extended Capability ID for the Readiness Time Reporting Extended Capability is 0022h.

53.4.6.2.43 PCIe RTR readiness time reporting 1 register (PCIE_CFC_RTR_RTR1)

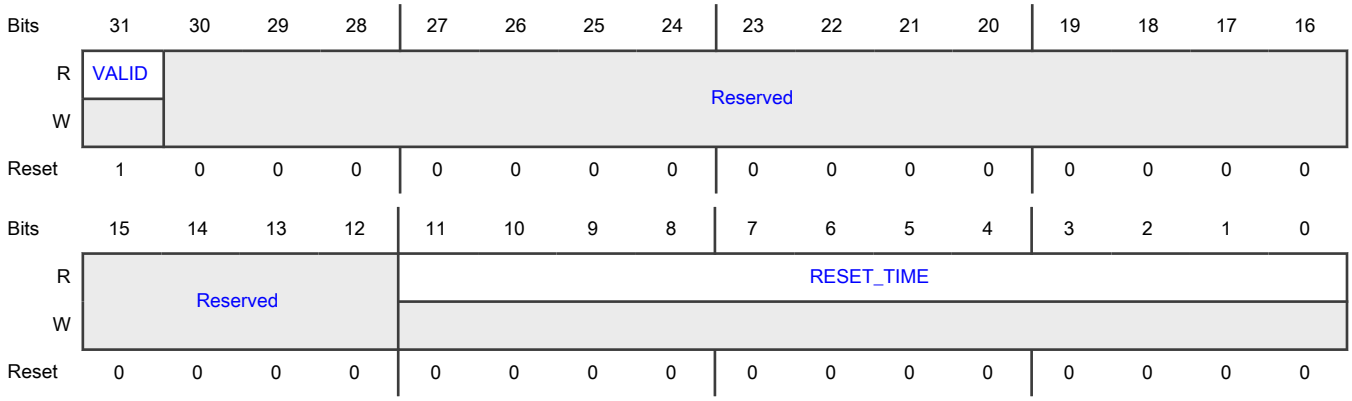
Offset

Register	Offset
PCIE_CFC_RTR_RTR1	144h

Function

This is the PCIe config capability Readiness Time Reporting 1 register.

Diagram



Fields

Field	Function
31 VALID	Valid If Set, indicates that all time values in this capability are valid. If Clear, indicates that the time values in this capability are not yet available. If this bit remains Clear and 1 minute has elapsed after all associated device driver(s) have started, software is permitted to assume that this bit will never be set.
30-12 —	Reserved
11-0 RESET_TIME	Reset Time The time the Function requires to become Configuration-Ready after the completion of Conventional Reset. This field is undefined when the Valid bit is Clear. This field must be less than or equal to the encoded value A1Eh.

53.4.6.2.44 PCIe RTR readiness time reporting 2 register (PCIE_CFC_RTR_RTR2)

Offset

Register	Offset
PCIE_CFC_RTR_RTR2	148h

Function

This is the PCIe config capability Readiness Time Reporting 2 register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								D3HOT_D0_TIME							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	D3HOT_D0_TIME				FLR_TIME											
W																
Reset	0	0	0	0	0	1	1	0	0	0	1	1	0	1	0	0

Fields

Field	Function
31-24 —	Reserved
23-12 D3HOT_D0_TIME	D3 hot to D0 time $D3_{hot}$ to D0 Time is the time that the Function requires after it is directed from $D3_{hot}$ to D0 before it is Configuration-Ready and has returned to either $D0_{uninitialized}$ or $D0_{active}$ state (see the PCI Bus Power Management Interface Specification). This field is undefined when the PCIE_CFC_RTR_RTR1[VALID] bit is Clear. This field must be less than or equal to the encoded value 80Ah.
11-0 FLR_TIME	FLR Time The time that the Function requires to become Configuration-Ready after it was issued an FLR. This field is undefined when the PCIE_CFC_RTR_RTR1[VALID] bit is Clear. This field must be less than or equal to the encoded value A1Eh. **NOTE** The value is taken directly from IERB register NETCFLRCR. The value in this field is encoded, where bits 8-0 is the value, and bits 11-9 is the scale. The scale is hardwired as 32.768us.

53.4.6.3 NETC Integrated Endpoint Register Block register descriptions

This section describes the Integrated Endpoint Register Block (IERB) registers. The IERB is a 64 KB size page containing registers that are used for pre-boot initialization, debug, and non-customer configuration.

The lower 32 KB of the IERB are reserved for architected registers required for IEPs. The upper 32 KB are used for NETC specific resources.

53.4.6.3.1 IERB memory map

NETC_IERB base address: 6080_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Capability register 0 (CAPR0)	32	R	0111_0651h
4h	Capability register 1 (CAPR1)	32	R	000E_000Eh
8h	Capability register 2 (CAPR2)	32	R	0000_0034h
Ch	Capability register 3 (CAPR3)	32	R	0008_0008h
20h	Common memory capability register (CMCAPR)	32	R	0000_2800h
30h	Ingress port filter ternary memory capability register (IPFTMCAPR)	32	R	0000_00C4h
44h	Time gate scheduling memory capability register (TGSMCAPR)	32	R	0000_0700h
80h	Shared memory depletion threshold register (SMDTR)	32	RW	0000_0040h
84h	ENETC receive shared memory buffer allotment register (ERSMBAR)	32	RW	0000_05D2h
C0h	HTA 0 HP configuration register (HTA0HPCR)	32	RW	0000_0C00h
C4h	HTA 0 LP configuration register (HTA0LPCR)	32	RW	0000_0C00h
100h	Hash bucket table memory allocation register (HBTMAR)	32	RW	0640_0200h
104h	Hash bucket table configuration register (HBTCR)	32	RW	0000_0032h
108h	Guaranteed hash table entry memory capability register (GHTEMCAPR)	32	R	0000_0000h
170h	NETC FLR configuration register (NETCFLRCR)	32	RW	0000_0634h
178h	NETC clock period fractional register (NETCCLKFR)	32	RW	2AAA_AAAAh
17Ch	NETC clock configuration register (NETCCLKCR)	32	RW	0004_00F0h
180h	System bus configuration register (SBCR)	32	RW	0000_000Ah
184h	System bus outstanding transaction control register (SBOTCR)	32	RW	0000_0000h
190h	Stream gating lag time for refresh register (SGLTTR)	32	RW	0000_0014h
200h	Root complex 0 binding configuration register (R0BCR)	32	R	0000_0000h
208h	Root complex 0 MSI-X cache attribute register (RC0MSICAR)	32	RW	0000_0002h
20Ch	Root complex 0 MSI access management qualifier register (RC0MSIAMQR)	32	RW	1000_0000h
300h	EMDIO binding configuration register (EMDIOBCR)	32	R	8000_0100h
314h	EMDIO MSI-X configuration register (EMDIOMCR)	32	R	0000_0000h
320h	EMDIO config header device ID and vendor ID register (EMDIO_CFH_DIDVID)	32	RW	EE00_1957h
324h	EMDIO config header subsystem ID and subsystem vendor ID register (EMDIO_CFH_SIDSVID)	32	RW	EE00_1957h
348h - 34Ch	EMDIO boot loader parameter register a (EMDIOBLPR0 - EMDIOBLPR1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
350h	EMDIO configuration register (EMDIO_CFG)	32	RW	0000_0010h
400h	Timer 0 binding configuration register (T0BCR)	32	R	See section
414h	Timer 0 MSI-X configuration register (T0MCR)	32	RW	0000_0000h
420h	Timer 0 config header device ID and vendor ID register (T0_CFH_DIDVID)	32	RW	EE02_1957h
424h	Timer 0 config header subsystem ID and subsystem vendor ID register (T0_CFH_SIDSVID)	32	RW	EE02_1957h
448h - 44Ch	Timer 0 boot loader parameter register b (T0BLPR0 - T0BLPR1)	32	RW	0000_0000h
1000h	Link 0 capability register (L0CAPR)	32	R	3707_7000h
1004h	Link 0 MAC capability register (L0MCAPR)	32	R	See section
1008h	Link 0 I/O capability register (L0IOCAPR)	32	R	See section
1010h	Link 0 binding configuration register (L0BCR)	32	RW	0000_0000h
1014h	Link 0 transmit byte credit comfort threshold register (L0TXBCCTR)	32	RW	0000_0200h
1020h	Link 0 end 0 MAC address register 0 (L0E0MAR0)	32	RW	0000_0000h
1024h	Link 0 end 0 MAC address register 1 (L0E0MAR1)	32	RW	0000_0000h
1040h	Link 1 capability register (L1CAPR)	32	R	3707_7000h
1044h	Link 1 MAC capability register (L1MCAPR)	32	R	See section
1048h	Link 1 I/O capability register (L1IOCAPR)	32	R	See section
1050h	Link 1 binding configuration register (L1BCR)	32	RW	0000_0001h
1054h	Link 1 transmit byte credit comfort threshold register (L1TXBCCTR)	32	RW	0000_0200h
1060h	Link 1 end 0 MAC address register 0 (L1E0MAR0)	32	RW	0000_0000h
1064h	Link 1 end 0 MAC address register 1 (L1E0MAR1)	32	RW	0000_0000h
1080h	Link 2 capability register (L2CAPR)	32	R	3707_7000h
1084h	Link 2 MAC capability register (L2MCAPR)	32	R	See section
1088h	Link 2 I/O capability register (L2IOCAPR)	32	R	See section
1090h	Link 2 binding configuration register (L2BCR)	32	RW	0000_0002h
1094h	Link 2 transmit byte credit comfort threshold register (L2TXBCCTR)	32	RW	0000_0200h
10A0h	Link 2 end 0 MAC address register 0 (L2E0MAR0)	32	RW	0000_0000h
10A4h	Link 2 end 0 MAC address register 1 (L2E0MAR1)	32	RW	0000_0000h
10C0h	Link 3 capability register (L3CAPR)	32	R	3707_7000h
10C4h	Link 3 MAC capability register (L3MCAPR)	32	R	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10C8h	Link 3 I/O capability register (L3IOCAPR)	32	R	See section
10D0h	Link 3 binding configuration register (L3BCR)	32	RW	0000_0003h
10D4h	Link 3 transmit byte credit comfort threshold register (L3TXBCCTR)	32	RW	0000_0200h
10E0h	Link 3 end 0 MAC address register 0 (L3E0MAR0)	32	RW	0000_0000h
10E4h	Link 3 end 0 MAC address register 1 (L3E0MAR1)	32	RW	0000_0000h
1100h	Link 4 capability register (L4CAPR)	32	R	3707_7000h
1104h	Link 4 MAC capability register (L4MCAPR)	32	R	See section
1108h	Link 4 I/O capability register (L4IOCAPR)	32	R	See section
1110h	Link 4 binding configuration register (L4BCR)	32	RW	0000_0040h
1114h	Link 4 transmit byte credit comfort threshold register (L4TXBCCTR)	32	RW	0000_0200h
1120h	Link 4 end 0 MAC address register 0 (L4E0MAR0)	32	RW	0000_0000h
1124h	Link 4 end 0 MAC address register 1 (L4E0MAR1)	32	RW	0000_0000h
1140h	Link 5 capability register (L5CAPR)	32	R	3707_7010h
1144h	Link 5 MAC capability register (L5MCAPR)	32	R	See section
1150h	Link 5 binding configuration register (L5BCR)	32	R	0004_0041h
1154h	Link 5 transmit byte credit comfort threshold register (L5TXBCCTR)	32	RW	0000_0200h
1160h	Link 5 end 0 MAC address register 0 (L5E0MAR0)	32	RW	0000_0000h
1164h	Link 5 end 0 MAC address register 1 (L5E0MAR1)	32	RW	0000_0000h
1168h	Link 5 end 1 MAC address register 0 (L5E1MAR0)	32	RW	0000_0000h
116Ch	Link 5 end 1 MAC address register 1 (L5E1MAR1)	32	RW	0000_0000h
2000h	Switch 0 binding configuration register (S0BCR)	32	R	8000_0200h
2014h	Switch 0 MSI-X configuration register (S0MCR)	32	RW	0000_0005h
2020h	Switch 0 config header device ID and vendor ID register (S0_CFH_DIDVID)	32	RW	EEF2_1957h
2024h	Switch 0 config header subsystem ID and subsystem vendor ID register (S0_CFH_SIDSVID)	32	RW	EEF2_1957h
2038h	Switch 0 command cache attribute register (S0CCAR)	32	RW	0202_0202h
2040h	Switch 0 access management qualifier register (S0AMQR)	32	RW	1200_0000h
2048h - 204Ch	Switch 0 boot loader parameter register b (S0BLPR0 - S0BLPR1)	32	RW	0000_0000h
2060h	Switch 0 shared memory buffer allotment register (S0SMBAR)	32	RW	0000_0C42h
2080h	Switch 0 hash table memory allotment register (S0HTMAR)	32	RW	0000_08C1h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2084h	Switch 0 index table memory allocation register (S0ITMAR)	32	RW	0000_05D0h
2088h	Switch 0 ingress port filter table memory allocation register (S0IPFTMAR)	32	RW	0000_008Ch
20A0h	Switch 0 rate policer index table memory allocation register (S0RPITMAR)	32	RW	0000_0080h
20A4h	Switch 0 ingress stream counter index table memory allocation register (S0ISCITMAR)	32	RW	0000_0180h
20A8h	Switch 0 ingress stream index table memory allocation register (S0ISITMAR)	32	RW	0000_0180h
20ACh	Switch 0 ingress sequence generation index table memory allocation register (S0ISQGITMAR)	32	RW	0000_0030h
20B4h	Switch 0 stream gate instance index table memory allocation register (S0SGIITMAR)	32	RW	0000_0020h
20B8h	Switch 0 stream gate control list index table memory allocation register (S0SGCLITMAR)	32	RW	0000_00C0h
20BCh	Switch 0 frame modification index table memory allocation register (S0FMITMAR)	32	RW	0000_0040h
20C0h	Switch 0 frame modification data index table memory allocation register (S0FMDITMAR)	32	RW	0000_0100h
20F0h	Switch 0 time gate scheduling table allocation register (S0TGSTAR)	32	RW	0000_0500h
20F4h	Switch 0 time gate scheduling lookahead register (S0TGSLR)	32	RW	0000_0064h
2204h	Switch 0 management port configuration register (S0MPCR)	32	R	0000_0004h
2210h	Switch 0 VLAN Filter (hash) table default entry configuration registers 0 (S0VFHTDECR0)	32	RW	0000_0000h
2214h	Switch 0 VLAN filter hash table default entry configuration registers 1 (S0VFHTDECR1)	32	RW	FFFF_0000h
2218h	Switch 0 VLAN filter hash table default entry configuration registers 2 (S0VFHTDECR2)	32	RW	1200_0000h
3000h	ENETC 0 binding configuration register 0 (E0BCR0)	32	R	8000_0300h
3004h	ENETC 0 binding configuration register 1 (E0BCR1)	32	R	0004_0004h
3008h	ENETC 0 binding configuration register 2 (E0BCR2)	32	R	0004_0004h
3010h	ENETC 0 VSI binding configuration register (E0VBCR)	32	R	0000_0000h
3014h	ENETC 0 MSI-X configuration register (E0MCR)	32	RW	0000_000Bh
3020h	ENETC 0 config header device ID and vendor ID register (E0_CFH_DIDVID)	32	RW	E101_1957h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3024h	ENETC 0 config header subsystem ID and subsystem vendor ID register (E0_CFH_SIDSVID)	32	RW	E101_1957h
3028h	ENETC 0 config capability VF device ID register (E0_CFC_VFDID)	32	RW	EF00_0000h
3030h	ENETC 0 buffer cache attribute register 0 (E0BCAR)	32	RW	0202_0202h
3034h	ENETC 0 message cache attribute register (E0MCAR)	32	RW	0002_0002h
3038h	ENETC 0 command cache attribute register (E0CAR)	32	RW	0202_0202h
3040h	ENETC 0 access management qualifier register (E0AMQR)	32	RW	1200_0000h
3048h	ENETC 0 boot loader parameter register 0 (E0BLPR0)	32	RW	0000_0000h
304Ch	ENETC 0 boot loader parameter register 1 (E0BLPR1)	32	RW	0000_0000h
3050h	ENETC 0 receive memory buffer entitlement register (E0RXMBER)	32	RW	0000_0000h
3054h	ENETC 0 receive memory buffer limit register (E0RXMBLR)	32	RW	0000_0000h
3070h	ENETC 0 transmit high priority tier byte credit register (E0TXHPTBCR)	32	RW	0000_05DCh
3074h	ENETC 0 transmit low priority tier byte credit register (E0TXLPTBCR)	32	RW	0000_05DCh
3080h	ENETC 0 hash table memory allotment register (E0HTMAR)	32	RW	0000_0008h
3084h	ENETC 0 index table memory allocation register (E0ITMAR)	32	RW	0000_0050h
3088h	ENETC 0 ingress port filter table memory allocation register (E0IPFTMAR)	32	RW	0000_001Ch
30A0h	ENETC 0 rate policer index table memory allocation register (E0RPITMAR)	32	RW	0000_0008h
30A4h	ENETC 0 ingress stream counter index table memory allocation register (E0ISCITMAR)	32	RW	0000_0008h
30A8h	ENETC 0 ingress stream index table memory allocation register (E0ISITMAR)	32	RW	0000_0008h
30B4h	ENETC 0 stream gate instance index table memory allocation register (E0SGIITMAR)	32	RW	0000_0008h
30B8h	ENETC 0 stream gate control list index table memory allocation register (E0SGCLITMAR)	32	RW	0000_0030h
30F0h	ENETC 0 time gate scheduling table allocation register (E0TGSTAR)	32	RW	0000_0100h
30F4h	ENETC 0 time gate scheduling lookahead register (E0TGSLR)	32	RW	0000_2710h
3100h	ENETC 1 binding configuration register 0 (E1BCR0)	32	R	8000_0400h
3104h	ENETC 1 binding configuration register 1 (E1BCR1)	32	R	000A_000Ah
3108h	ENETC 1 binding configuration register 2 (E1BCR2)	32	R	0004_0004h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3110h	ENETC 1 VSI binding configuration register (E1VBCR)	32	R	0000_0001h
3114h	ENETC 1 MSI-X configuration register (E1MCR)	32	RW	0000_0018h
3120h	ENETC 1 config header device ID and vendor ID register (E1_CFH_DIDVID)	32	RW	E110_1957h
3124h	ENETC 1 config header subsystem ID and subsystem vendor ID register (E1_CFH_SIDSVID)	32	RW	E110_1957h
3128h	ENETC 1 config capability VF device ID register (E1_CFC_VFDID)	32	RW	EF00_0000h
3130h	ENETC 1 buffer cache attribute register 0 (E1BCAR)	32	RW	0202_0202h
3134h	ENETC 1 message cache attribute register (E1MCAR)	32	RW	0002_0002h
3138h	ENETC 1 command cache attribute register (E1CAR)	32	RW	0202_0202h
3140h	ENETC 1 access management qualifier register (E1AMQR)	32	RW	1200_0000h
3148h	ENETC 1 boot loader parameter register 0 (E1BLPR0)	32	RW	0000_0000h
314Ch	ENETC 1 boot loader parameter register 1 (E1BLPR1)	32	RW	0000_0000h
3150h	ENETC 1 receive memory buffer entitlement register (E1RXMBER)	32	RW	0000_0000h
3154h	ENETC 1 receive memory buffer limit register (E1RXMBLR)	32	RW	0000_0000h
3170h	ENETC 1 transmit high priority tier byte credit register (E1TXHPTBCR)	32	RW	0000_05DCh
3174h	ENETC 1 transmit low priority tier byte credit register (E1TXLPTBCR)	32	RW	0000_05DCh
3180h	ENETC 1 hash table memory allotment register (E1HTMAR)	32	RW	0000_0010h
3184h	ENETC 1 index table memory allocation register (E1ITMAR)	32	RW	0000_0030h
3188h	ENETC 1 ingress port filter table memory allocation register (E1IPFTMAR)	32	RW	0000_001Ch
31A0h	ENETC 1 rate policer index table memory allocation register (E1RPITMAR)	32	RW	0000_0010h
31A4h	ENETC 1 ingress stream counter index table memory allocation register (E1ISCITMAR)	32	RW	0000_0010h
31A8h	ENETC 1 ingress stream index table memory allocation register (E1ISITMAR)	32	RW	0000_0010h
31B4h	ENETC 1 stream gate instance index table memory allocation register (E1SGIITMAR)	32	RW	0000_0000h
31B8h	ENETC 1 stream gate control list index table memory allocation register (E1SGCLITMAR)	32	RW	0000_0000h
31F0h	ENETC 1 time gate scheduling table allocation register (E1TGSTAR)	32	RW	0000_0100h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
31F4h	ENETC 1 time gate scheduling lookahead register (E1TGSLR)	32	RW	8000_2710h
4000h	VSI 0 access management qualifier register (V0AMQR)	32	RW	1200_0000h
4008h - 400Ch	VSI 0 boot loader parameter register b (V0BLPR0 - V0BLPR1)	32	RW	0000_0000h
4010h	VSI 0 primary MAC address register 0 (V0PMAR0)	32	RW	0000_0000h
4014h	VSI 0 primary MAC address register 1 (V0PMAR1)	32	RW	0000_0000h

53.4.6.3.2 Capability register 0 (CAPR0)

Offset

Register	Offset
CAPR0	0h

Function

This is the capability register 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	NUM_VSI							NUM_ENETC				Reserved	NUM_SW		
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				NUM_LINKS				NUM_TMR		Reserved	NUM_EMD...	NUM_RC			
W																
Reset	0	0	0	0	0	1	1	0	0	1	0	1	0	0	0	1

Fields

Field	Function
31 —	Reserved
30-24	Total number of ENETC VSI's instances supported.

Table continues on the next page...

Table continued from the previous page...

Field	Function
NUM_VSI	Range: 0..64
23-19 NUM_ENETC	Number of ENETC instances supported. Range: 0..23
18 —	Reserved
17-16 NUM_SW	Number of switch instances supported. Range: 0..2
15-13 —	Reserved
12-8 NUM_LINKS	Indicates the number of links supported (internal and external). Range: 0..31
7-6 NUM_TMR	Number of timer instances supported. Range: 0..2
5 —	Reserved
4 NUM_EMDIO	Number of EMDIO instances supported. Range: 0..1
3-0 NUM_RC	Number of Root Complexes supported. Range: 0..15

53.4.6.3.3 Capability register 1 (CAPR1)

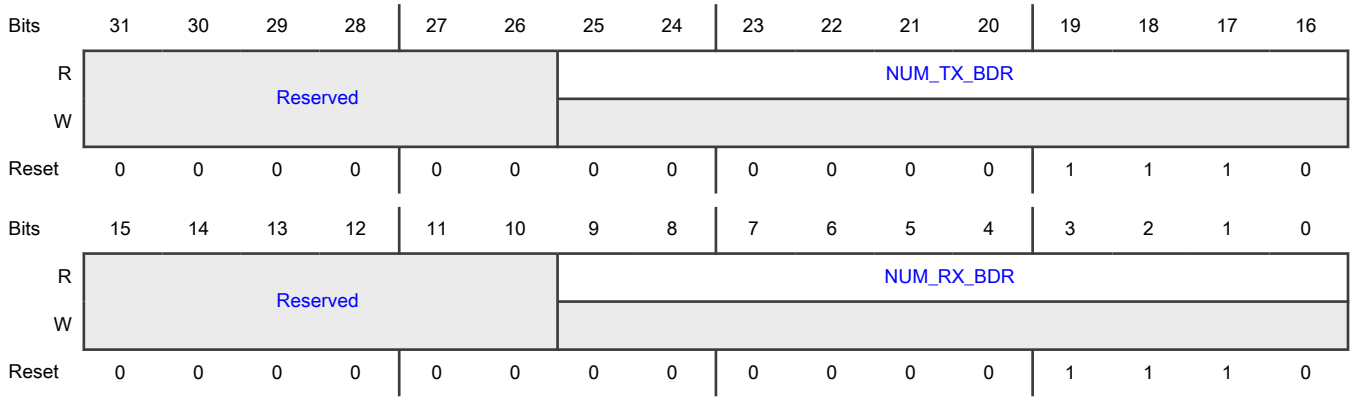
Offset

Register	Offset
CAPR1	4h

Function

This is the capability register 1.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 NUM_TX_BDR	Total number of transmit BD rings supported by NETC. Range: 0..1023
15-10 —	Reserved
9-0 NUM_RX_BDR	Total number of receive BD rings supported by NETC. Range: 0..1023

53.4.6.3.4 Capability register 2 (CAPR2)

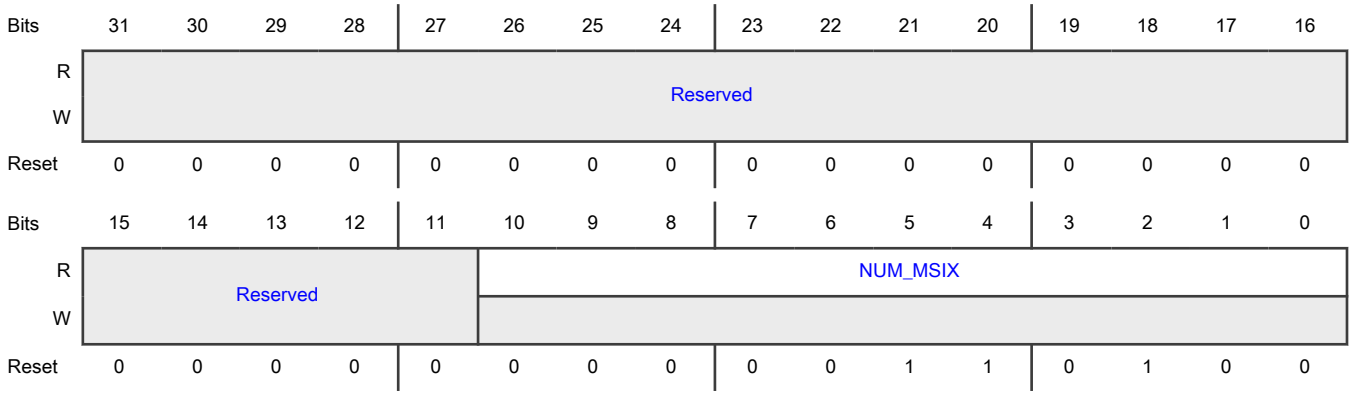
Offset

Register	Offset
CAPR2	8h

Function

This is the capability register 2.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-0 NUM_MSIX	Number of MSI-X table entries available for allocation by NETC functions. Formula: NUM_MSIX+1 Range: 1..1024 **NOTE** If allocated entries to NETC functions exceed this value, error is indicated by NETCSR[ERROR]. One entry is always allocated to the EMDIO controller.

53.4.6.3.5 Capability register 3 (CAPR3)

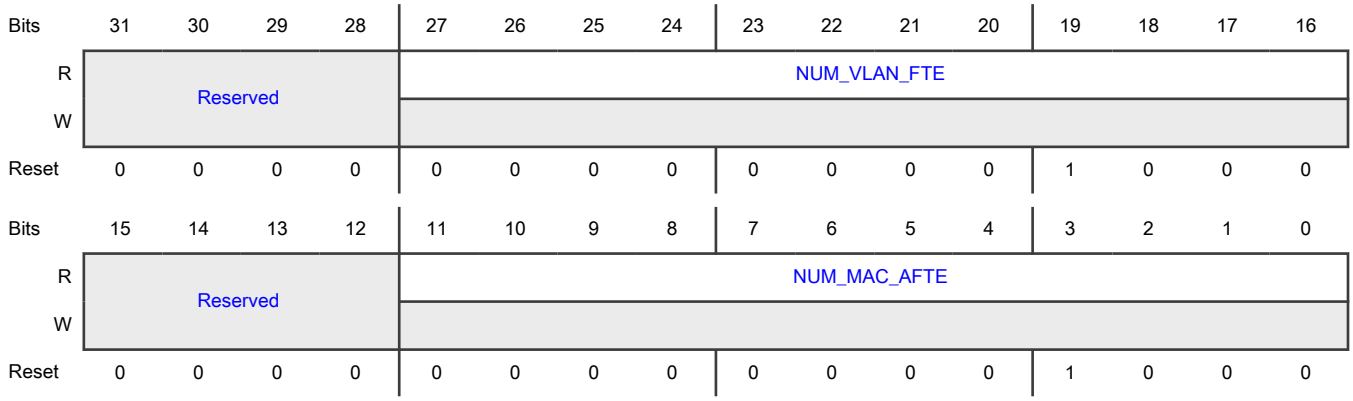
Offset

Register	Offset
CAPR3	Ch

Function

This is the capability register 3.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 NUM_VLAN_FTE	Total number of ENETC SI VLAN filter rules supported by NETC.
15-12 —	Reserved
11-0 NUM_MAC_AFTE	Total number of ENETC SI MAC address filter rules supported by NETC.

53.4.6.3.6 Common memory capability register (CMCAPR)

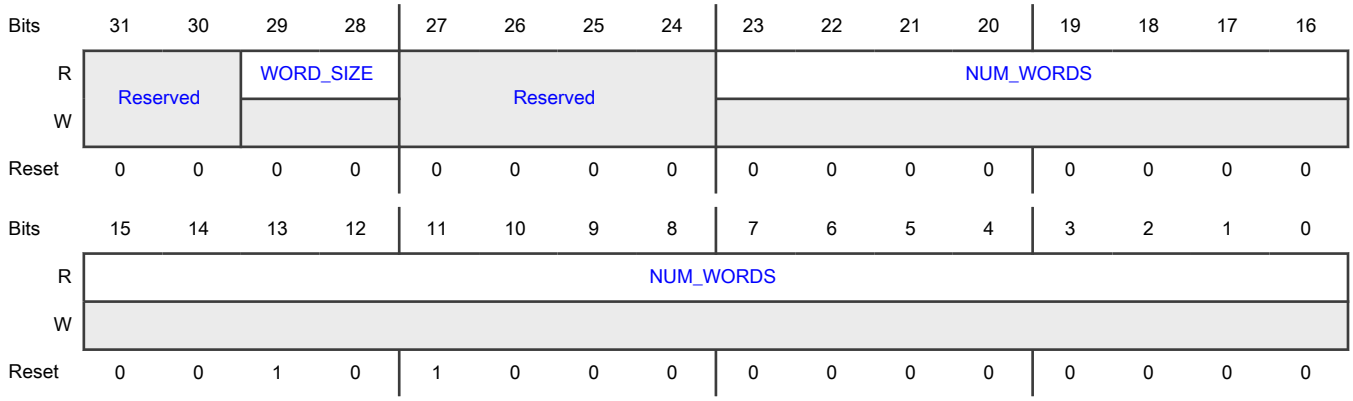
Offset

Register	Offset
CMCAPR	20h

Function

This is the common memory capability register.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-28 WORD_SIZE	Word size in bytes 0: 24B 1-3: reserved
27-24 —	Reserved
23-0 NUM_WORDS	Total amount of common memory in words available to NETC. This memory is partitioned between hash bucket table memory, each NETC function's index table memory with the remainder for buffering and hash tables.

53.4.6.3.7 Ingress port filter ternary memory capability register (IPFTMCAPR)

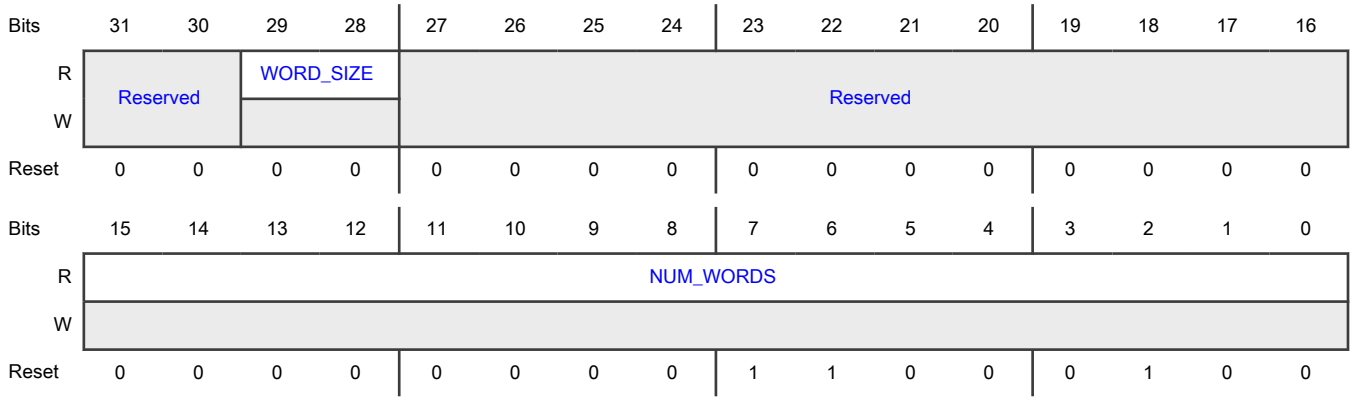
Offset

Register	Offset
IPFTMCAPR	30h

Function

This is the ingress port filter ternary memory capability register.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-28 WORD_SIZE	Word size in bits 0: 48 bits 1-3: reserved
27-16 —	Reserved
15-0 NUM_WORDS	Total amount of ternary memory in words available to NETC for ingress port filtering. This memory is partitioned between various NETC functions. **NOTE** If sum of allocated entries (S*a*IPFTMAR and E*a*IPFTMAR) exceed this value, error is indicated by NETCSR[ERROR].

53.4.6.3.8 Time gate scheduling memory capability register (TGSMCAPR)

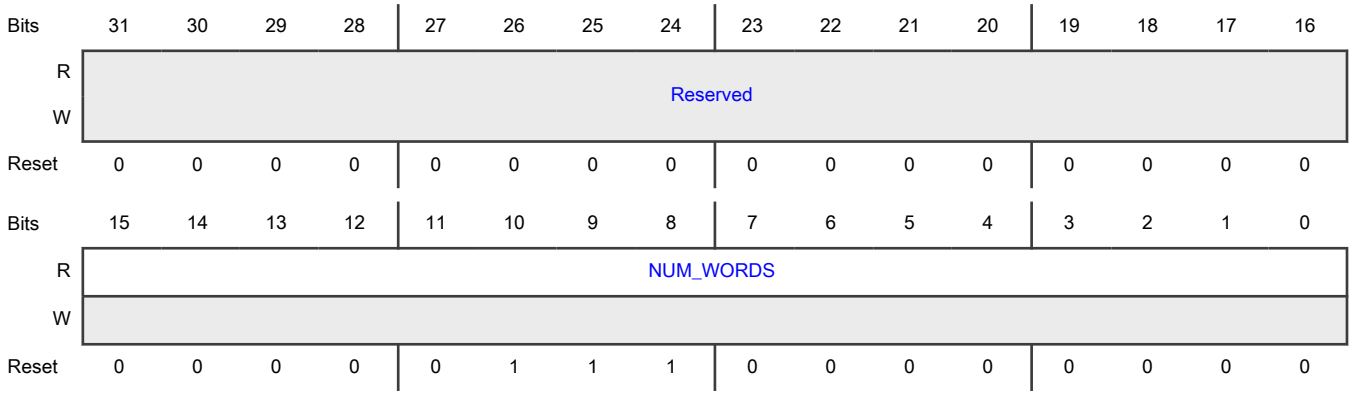
Offset

Register	Offset
TGSMCAPR	44h

Function

This is the time gate scheduling memory capability register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	Total amount of Time Gate Scheduling memory in words available to NETC. **NOTE** This memory may be partitioned between switch and ENETC Instances. If sum of allocated entries (S*a*TGSTAR and E*a*TGSTAR) exceed this value, error is indicated by NETCSR[ERROR].

53.4.6.3.9 Shared memory depletion threshold register (SMDTR)

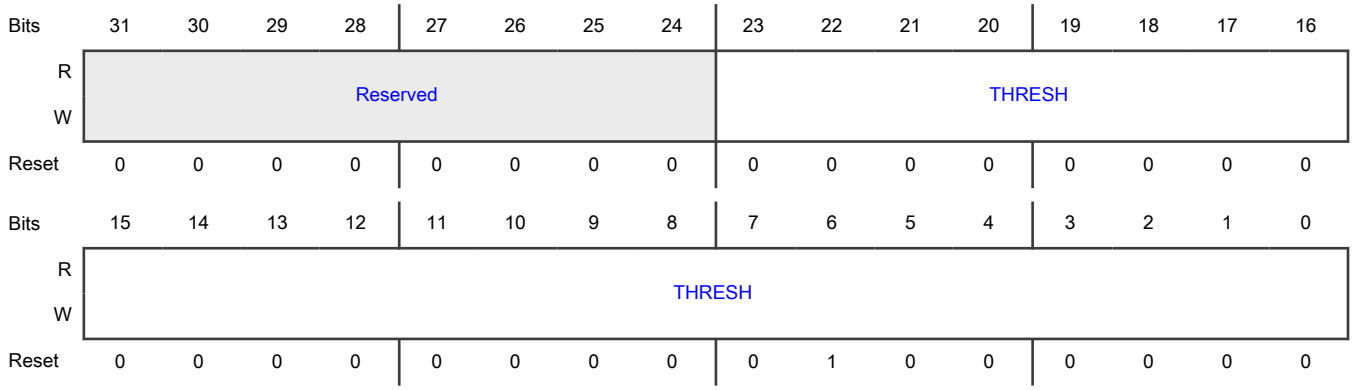
Offset

Register	Offset
SMDTR	80h

Function

This is the shared memory depletion threshold register.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 THRESH	Shared memory depletion threshold in Words. This sets the minimum amount of free memory that should be maintained in the datapath sub system, and when the amount of free memory falls below this threshold, a depletion indication is asserted, which may trigger "intelligent drop" of frames from the ingress queues in the ICM. Setting this field to 0 disables dropping of frames by Ingress Congestion Manager (ICM) due to internal shared memory depletion. It is recommended to set this field to 64 Words. **NOTE** This threshold is an added security in case of switch and ENETC common memory allocation results in amount of buffers used by all ENETCs being less than ERSMBAR[THRESH] value. There are error scenarios, such as Buffer loss or Free List corruption (see SMLCR for more info), where this condition may occur.

53.4.6.3.10 ENETC receive shared memory buffer allotment register (ERSMBAR)

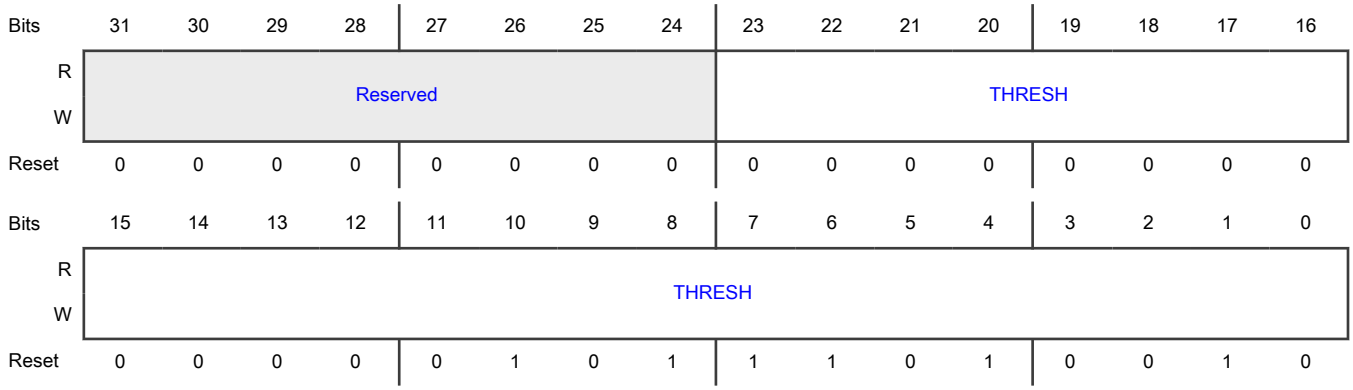
Offset

Register	Offset
ERSMBAR	84h

Function

This is the ENETC receive shared memory buffer allotment register.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 THRESH	Threshold in words for internal receive buffer memory used by all ENETC functions. If this threshold is exceeded, smart-drop from ICM queues may be triggered. The threshold should not be lower than the total number of ENETC instances multiplied by 2 times the maximum frame size.

53.4.6.3.11 HTA 0 HP configuration register (HTA0HPCR)

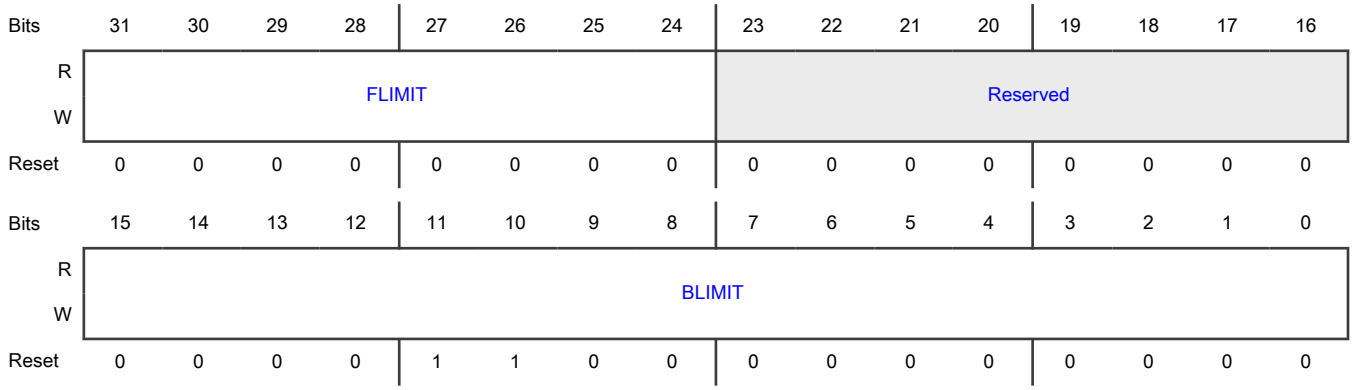
Offset

Register	Offset
HTA0HPCR	C0h

Function

This is the HTA high priority configuration register.

Diagram



Fields

Field	Function
31-24 FLIMIT	HTA global high priority frame limit. Sets the maximum amount of high priority frames that can be underway within HTA. 0x0 = no limit **NOTE** This limit is shared between ENETC instances using WBFS algorithm proportional to ENETC instance PSPEED.
23-16 —	Reserved
15-0 BLIMIT	HTA global high priority byte limit setting. Sets the maximum amount of high priority DMA work (in bytes) that can be underway within HTA. 0x0 = no limit **NOTE** This limit is shared between ENETC instances using WBFS algorithm proportional to ENETC instance PSPEED.

53.4.6.3.12 HTA 0 LP configuration register (HTA0LPCR)

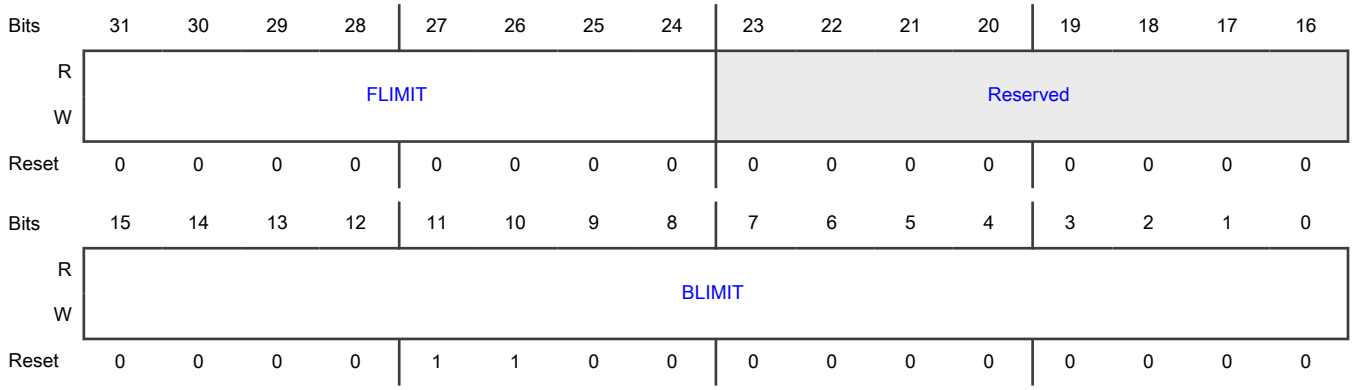
Offset

Register	Offset
HTA0LPCR	C4h

Function

This is the HTA low priority configuration register.

Diagram



Fields

Field	Function
31-24 FLIMIT	HTA global low priority Frame Limit. Sets the maximum amount of low priority frames that can be underway within HTA. 0x0 = no limit **NOTE** This is not per-port. WBFS is used to share this limit among ports.
23-16 —	Reserved
15-0 BLIMIT	HTA global low priority byte limit setting. Sets the maximum amount of low priority DMA work (in bytes) that can be underway within HTA. 0x0 = no limit **NOTE** This is not per-port. WBFS is used to share this limit among ports. Default: 2x1536B.

53.4.6.3.13 Hash bucket table memory allocation register (HBTMAR)

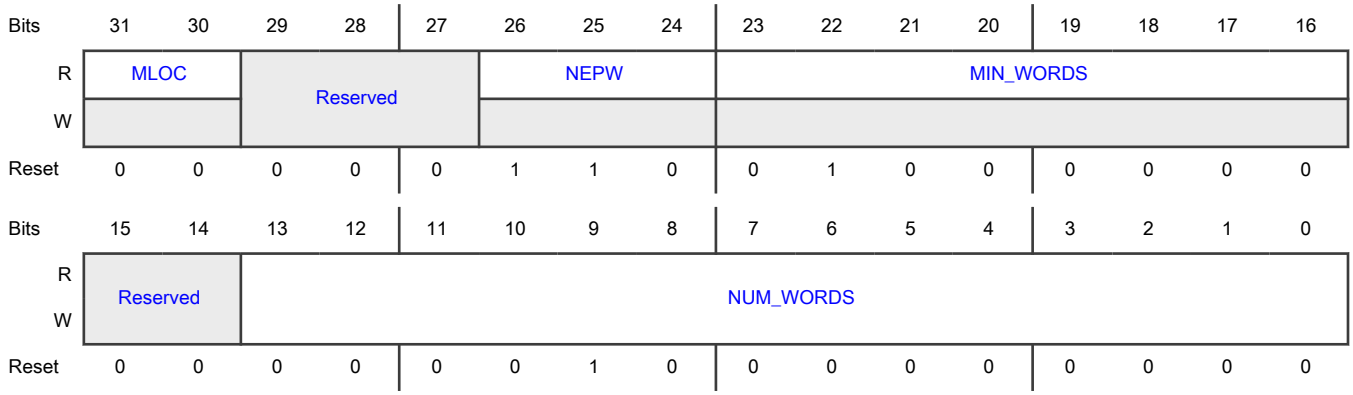
Offset

Register	Offset
HBTMAR	100h

Function

This is the hash bucket table memory allocation memory allocation register.

Diagram



Fields

Field	Function
31-30 MLOC	Indicates memory location of this table 0: Common memory 1-3: Reserved
29-27 —	Reserved
26-24 NEPW	Number of table entries per word.
23-16 MIN_WORDS	Minimum number of words required. NUM_WORDS should not be set to a value lower than MIN_WORDS.
15-14 —	Reserved
13-0 NUM_WORDS	Number of words allocated from Common Memory. **NOTE** This value is used to set the R/O global register HBTCAPR[NUM_ENTRIES] where NUM_ENTRIES = NUM_WORDS * NEPW. The allocated amount must be a power of 2 and greater than or equal to MIN_WORDS otherwise, a lower power of 2 is used and an error is indicated by NETCSR[ERROR].

53.4.6.3.14 Hash bucket table configuration register (HBTCR)

Offset

Register	Offset
HBTCR	104h

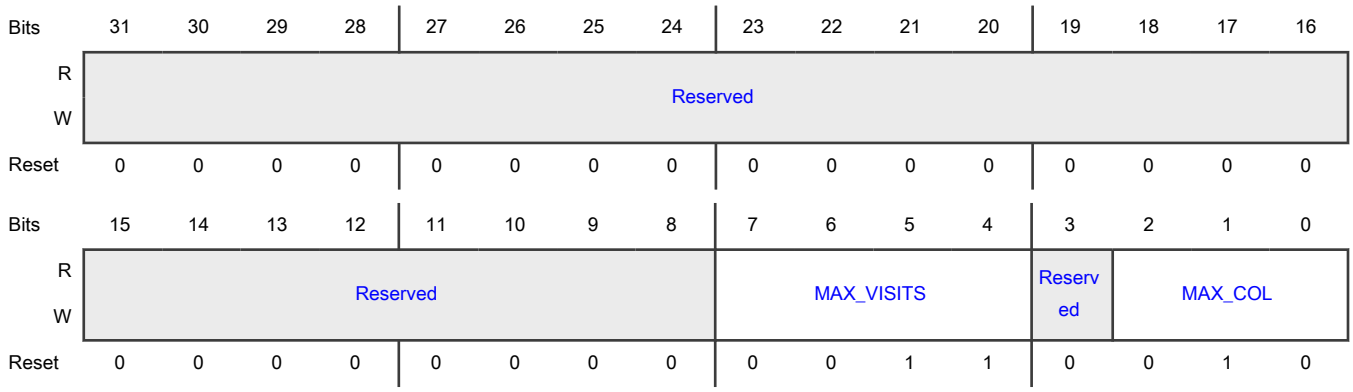
Function

This is the hash bucket table configuration register.

NOTE

The value of this register will be immediately reflected in the corresponding read-only global register HBTCAPR accessible to all switch and ENETC functions.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-4 MAX_VISITS	Specifies the maximum number of Hash Entries Visited during a Search Table Management Command. Range: 64..1K Formula: 64*(MAX_VISITS+1) MAX_VISITS=0..15 **NOTE** Value mirrored to Global HBTCAPR register which is in all Switch and ENETC Physical Functions.
3 —	Reserved
2-0 MAX_COL	Specifies the maximum EM/IM Hash collisions chain length allowed. Maximum is 3. Range: 1..3 Formula: MAX_COL+1 Don't allow entries to be added if it exceeds this value. **NOTE** Value mirrored to Global HBTCAPR register which is in all Switch and ENETC Physical Functions.

53.4.6.3.15 Guaranteed hash table entry memory capability register (GHTEMCAPR)

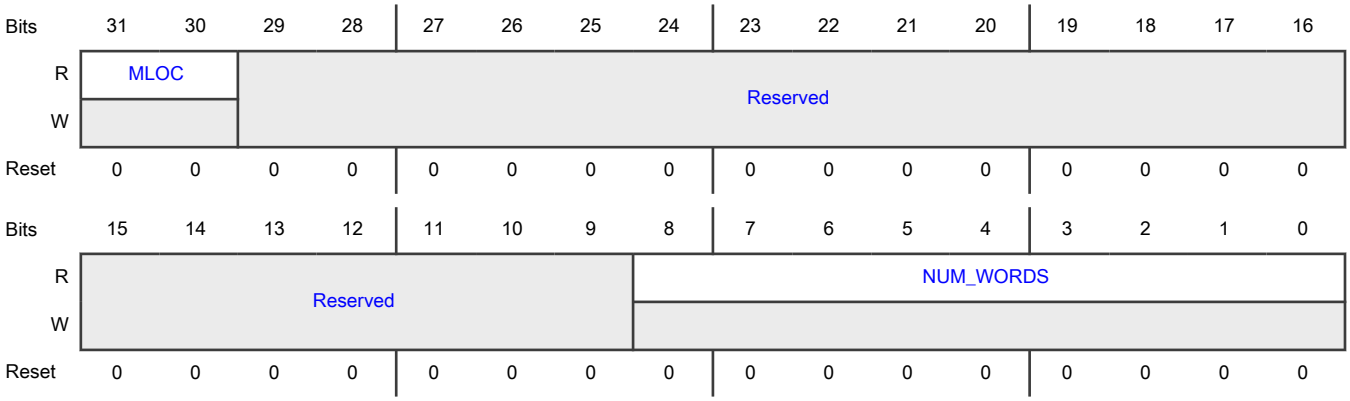
Offset

Register	Offset
GHTEMCAPR	108h

Function

This is the guaranteed hash table entry memory capability register.

Diagram



Fields

Field	Function
31-30 MLOC	Indicates memory location of this table 0: Common memory 1-3: Reserved
29-9 —	Reserved
8-0 NUM_WORDS	Total amount of words available to store the guaranteed hash table entries. When a hash table entry cannot be added because the hash bucket collision limit is exceeded, the entry can be added using a CAM (Context Addressable Memory) where the entry is stored in this table. Each guaranteed hash table entry occupies one word. **NOTE** This is only available for switch's FDB table and is reflected in value copied to switch's FDBHTCAPR[NUM_GMAC].

53.4.6.3.16 NETC FLR configuration register (NETCFLRCR)

Offset

Register	Offset
NETCFLRCR	170h

Function

Indicates the duration of the Function Level Reset (FLR) in steps of 32.768 microseconds.

The reset sequence includes the following three parts:

1. Quiesce (e.g.: processing of rings has stopped, pending frames flushed)
2. Reset of registers
3. Flush of table entries (e.g.: RFS/RSS). Hard coded to 4000 NETC cycles.

NOTE

The format of this register is identical to the PCIe Readiness Time Encoding format. Content is directly reflected in the PCIe Readiness Time Reporting Extended Capability. The maximum supported counter value is 2^{20} NETC cycles, and is calculated as NETCCLKCR[FREQ] x NETCFLRCR[VALUE] x 32.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				SCALE				VALUE							
W																
Reset	0	0	0	0	0	1	1	0	0	0	1	1	0	1	0	0

Fields

Field	Function
31-12 —	Reserved
11-9 SCALE	Scale Scale used for VALUE, i.e. multiplier. Formula: $2^{5*SCALE}$ns where SCALE={0..5}

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Hard coded to 32.768 microseconds.
8-0 VALUE	Time duration value expressed in SCALE units.

53.4.6.3.17 NETC clock period fractional register (NETCCLKFR)

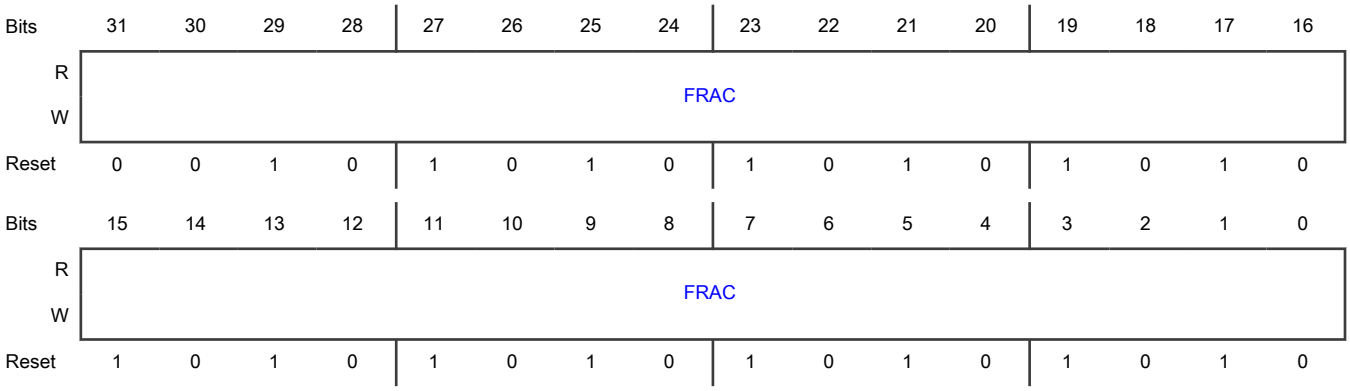
Offset

Register	Offset
NETCCLKFR	178h

Function

This is the NETC clock period fractional register.

Diagram



Fields

Field	Function
31-0 FRAC	NETC Clock Period's fractional nanosecond expressed as the number of fractional nanoseconds per NETC clock. Value is configured as the fractional portion of 1000/NETCCLKR[FREQ] * (2^{32}). Example: If Freq = 400 MHz, PERIOD=2 and FRAC=0x8000_0000. **NOTE** This value is writable in IERB (by pre-boot initialization), which is then copied to timer TMR_ADD during IERB lock.

53.4.6.3.18 NETC clock configuration register (NETCCLKCR)

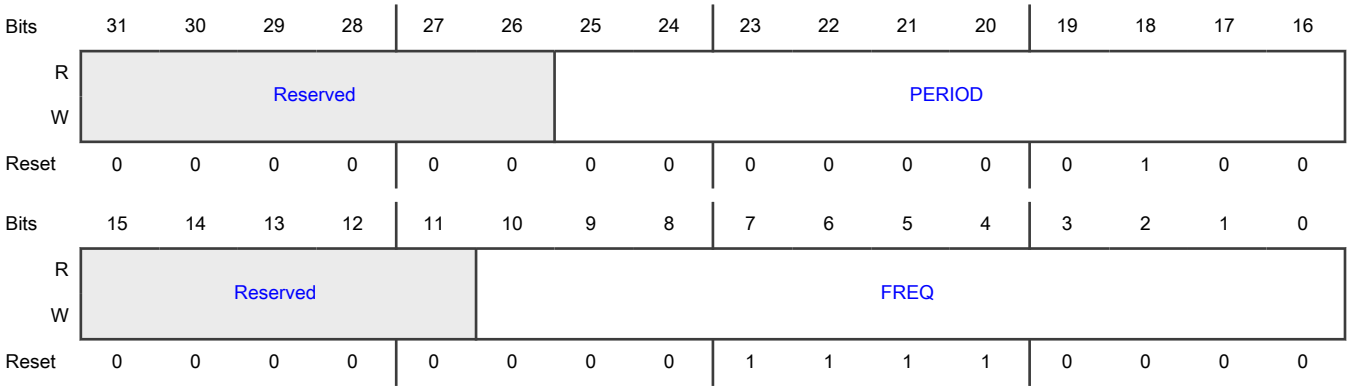
Offset

Register	Offset
NETCCLKCR	17Ch

Function

This is the NETC clock configuration register.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 PERIOD	Period NETC Clock Period expressed as the number of nanoseconds per NETC clock. Value is configured as the integer portion of 1000/FREQ. **NOTE** This value is writable in IERB (by pre-boot initialization), which is then copied to timer TMR_CTRL[PERIOD] during IERB lock.
15-11 —	Reserved
10-0 FREQ	Frequency This is the NETC clock frequency in MHz. **NOTE** Value is reflected in global register NETCCLKR.

53.4.6.3.19 System bus configuration register (SBCR)

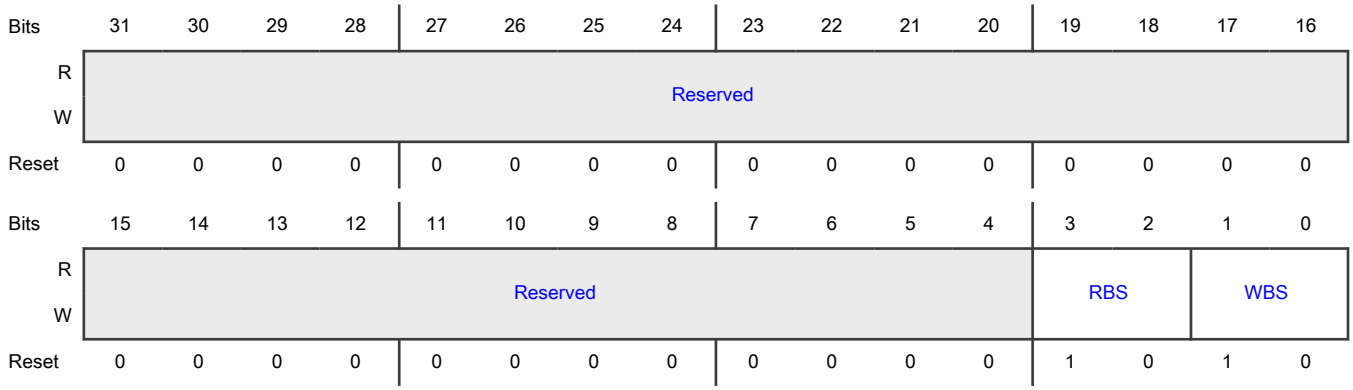
Offset

Register	Offset
SBCR	180h

Function

This is the system bus configuration register. It allows you to specify the address alignments at which system memory transactions issued by NETC are cracked into separate AXI transactions. For example, a setting of 64B will cause NETC to perform a large data transfer as a series of 64B AXI transactions, issued on natural 64B address alignments. If the large transfer does not start on a natural 64B aligned address, then the first 64B AXI transaction issued by NETC will not be 64B in size but will end at a 64B aligned addres such that the remaining 64B transactions in the large transfer are 64B address aligned.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-2 RBS	System Bus Maximum Read Burst Size. It indicates the maximum number of bytes to transfer in a read transaction. The size of the AXI bus is 64 bits. 0: 16B; ARLEN = 1 (2 data transfers) 1: 32B; ARLEN = 3 (4 data transfers) 2: 64B; ARLEN = 7 (8 data transfers) 3: 128B; ARLEN = 15 (16 data transfers)
1-0 WBS	System Bus Maximum Write Burst Size. It indicates the maximum number of bytes to transfer in a write transaction. The size of the AXI bus is 64 bits. 0: 16B; AWLEN = 1 (2 data transfers) 1: 32B; AWLEN = 3 (4 data transfers)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	2: 64B; AWLEN = 7 (8 data transfers) 3: 128B; AWLEN = 15 (16 data transfers) Note: The default setting of WBS=2 (64B) is recommended. WBS=3 may provide marginal performance benefit in configurations where system memory write latency is extremely high. Using WBS=3 (128B) setting requires that Receive Buffer Descriptor memory addresses be 128B address-aligned.

53.4.6.3.20 System bus outstanding transaction control register (SBOTCR)

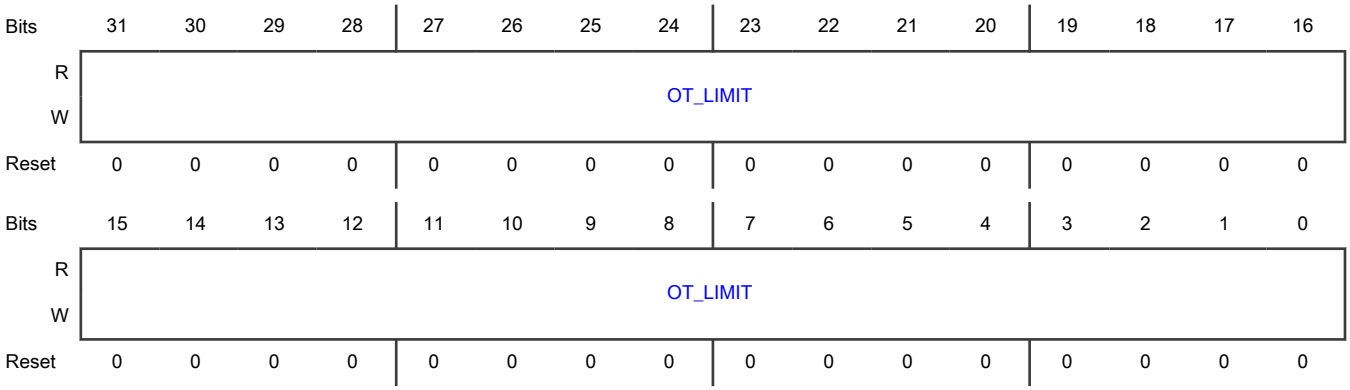
Offset

Register	Offset
SBOTCR	184h

Function

This is the system bus outstanding transaction control register.

Diagram



Fields

Field	Function
31-0 OT_LIMIT	This is a 32-bit read-write field, however only bits 23-16 and 7-0 have any effect on device behavior. Bits 31-24 and 15-8 are reserved for future use and should be written as 0x0. OT_LIMIT[7:0] encode a limit on outstanding non-critical NETC transactions, where "transaction" is defined as a read or write within a 32B address-aligned region. Setting this limit to a non-zero value N will cause NETC to initiate outstanding accesses to no more than N 32 byte address regions (one access per region). Valid range is 0-128, with a value of 0 meaning no limiting is applied. OT_LIMIT[23:16] encode a limit on all NETC transactions, relative to OT_LIMIT[7:0]. In other words, total outstanding NETC transactions (both critical and non-critical) are limited to OT_LIMIT[7:0] + OT_LIMIT[23:16] if OT_LIMIT[23:16] is non-zero.

Table continues on the next page...

Field	Function
	If OT_LIMIT[23:16] is 0 then no total transaction limit is applied and NETC will not throttle the issuing of critical transactions. Recommended setting for all applications is 0x0, however application characterization may determine that other client (CPU) access latency is improved by applying an upper limit on NETC DDR transactions.

53.4.6.3.21 Stream gating lag time for refresh register (SGLTTR)

Offset

Register	Offset
SGLTTR	190h

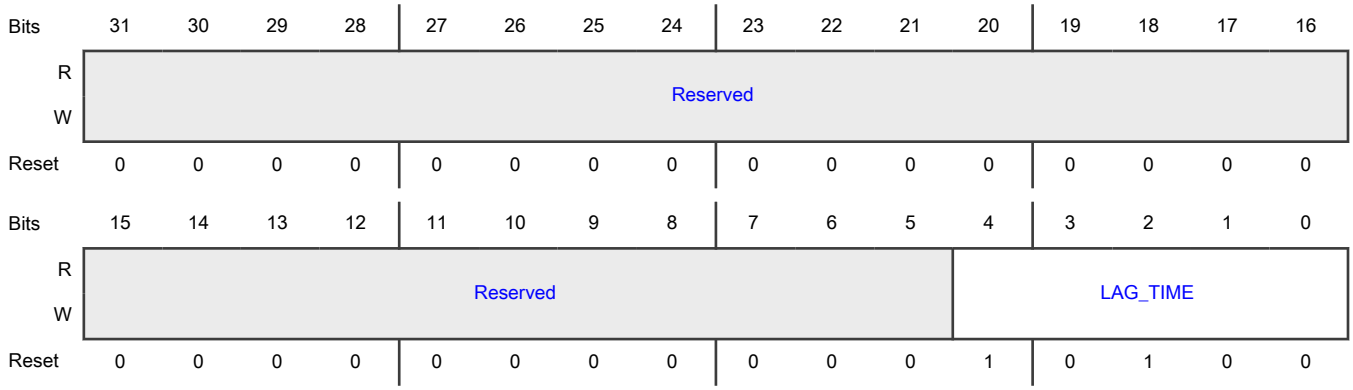
Function

SGLTTR register is used for stream gating to periodically update the last event time in each of the Stream Gating Instance table entries to ensure that the it doesn't go stale.

The last time value that is maintained in stream gating context is 30 bits (1 sec), hence it is valid for only one second in the past from current frame arrival time. Stream gating context is event triggered as such If there is no frame arrival for 1 second in the past the last time value will become stale. To ensure that the last event time value doesn't go stale, periodic updates are performed to advance the last event time to bring it closer to the current time but still keep in the past so that frame arrival time plus the propagation delay for the frame is still less than last event time.

SGLTTR value determines how far in the past the last time needs to be updated. Value programmed has to take into account the worst case propagation delay for the frame (i.e.. from the start of frame to when the end of frame is observed) as the last event time has to be less than the current frame arrival time. As such determining LAG_TIME setting in SGLTTR register depends on slowest line rate supported and worst case delay a frame can encounter. Default value is 65536ns. LAG_TIME specified must be less than or equal to 24 (i.e., Lag time is a nanosecond has to be less than or equal to 2^24)

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 LAG_TIME	Lag time is a nanosecond value = 2^{LAG_TIME} . Default LAG_TIME set to 0x14 means 2^{20} = 1.05 ms. To determine the lag time do the following. Calculate frame transfer delay for a maximum sized frame at the slowest line rate; Multiply by 2 to include worst case preemption delay; Then select LAG_TIME such that 2^{LAG_TIME} > determined total delay.

53.4.6.3.22 Root complex 0 binding configuration register (R0BCR)

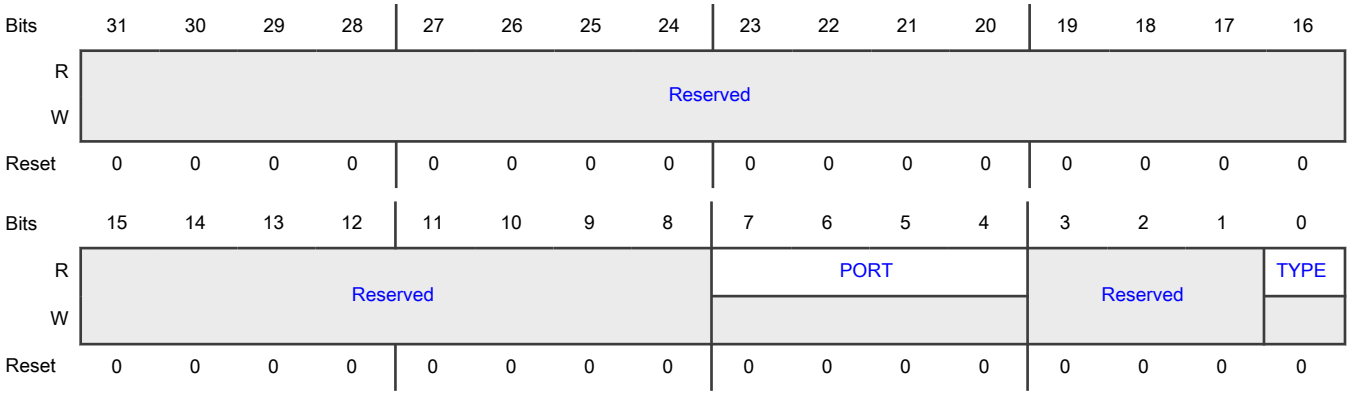
Offset

Register	Offset
R0BCR	200h

Function

This is the root complex binding configuration register.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-4 PORT	Port number Indicates how to address the PCIe target based on TYPE. 0: Not used

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1: Route-by-port number to direct accesses to proper PCIe controller
3-1 —	Reserved
0 TYPE	Indicates the type of root complex and routing 0: RCiEP 1: PCIe RC

53.4.6.3.23 Root complex 0 MSI-X cache attribute register (RC0MSICAR)

Offset

Register	Offset
RC0MSICAR	208h

Function

This is the root complex MSI-X cache attribute register. It determines the system interface attribute setting used for writes of all root complex MSI-X events.

NOTE

This register is continuously sampled, any changes will have immediate effect on MSI-X.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved									MSI_WRS...	MSI_WRDOMAIN		MSI_WRCACHE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6 MSI_WRSNP	MSI-X write snoop This is the snoop attribute setting used when NETC generates MSI-X events. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
5-4 MSI_WRDOMAIN	MSI-X write domain This is the domain attribute setting used when NETC generates MSI-X events. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
3-0 MSI_WRCACHE	MSI-X write cache type This is the cache attribute setting used when NETC generates MSI-X events. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.

53.4.6.3.24 Root complex 0 MSI access management qualifier register (RC0MSIAMQR)

Offset

Register	Offset
RC0MSIAMQR	20Ch

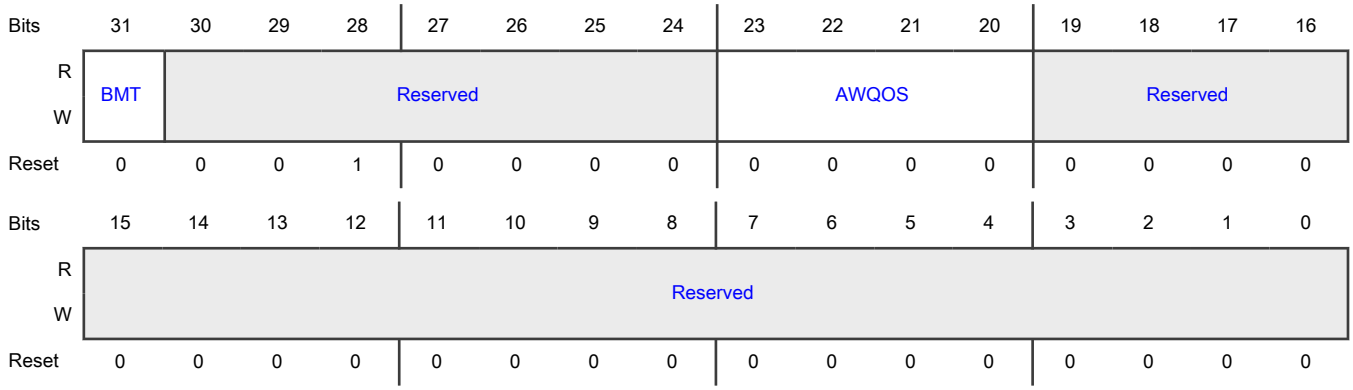
Function

This is the root complex MSI access management qualifier register. It provides attributes for MSI-X memory write accesses for all functions of NETC.

NOTE

Changing the value in this register has immediate effect unless otherwise noted. Care should be taken when making changes to the values for an enabled and operational timer function to ensure that consistent behavior is maintained. It is strongly encouraged, but not required, that the function be disabled before making changes.

Diagram



Fields

Field	Function
31 BMT	Bypass memory translation The bit is an indication to the SMMU to by-pass memory translation whenever the PF is performing memory transactions, effectively handling the memory address as a true physical address. 0 Memory translation 1 Bypass memory translation
30-24 —	Reserved
23-20 AWQOS	Address Write QoS.
19-0 —	Reserved

53.4.6.3.25 EMDIO binding configuration register (EMDIOBCR)

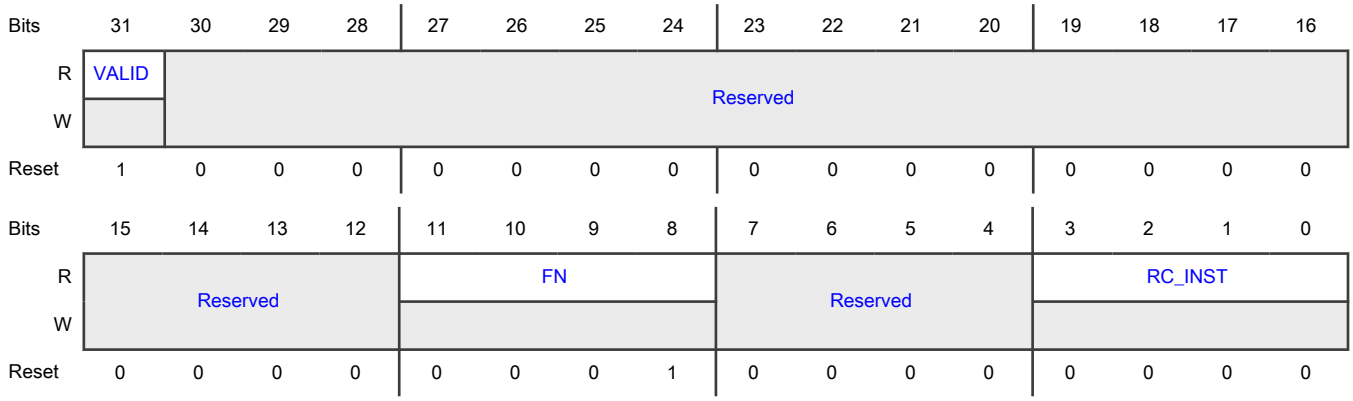
Offset

Register	Offset
EMDIOBCR	300h

Function

This is the EMDIO binding configuration register.

Diagram



Fields

Field	Function
31 VALID	If set, this EMDIO function is valid.
30-12 —	Reserved
11-8 FN	PCI device function number for global EMDIO controller. **NOTE** For assignment of function number, see PCIe Function Number Assignment.
7-4 —	Reserved
3-0 RC_INST	Root complex instance number.

53.4.6.3.26 EMDIO MSI-X configuration register (EMDIOMCR)

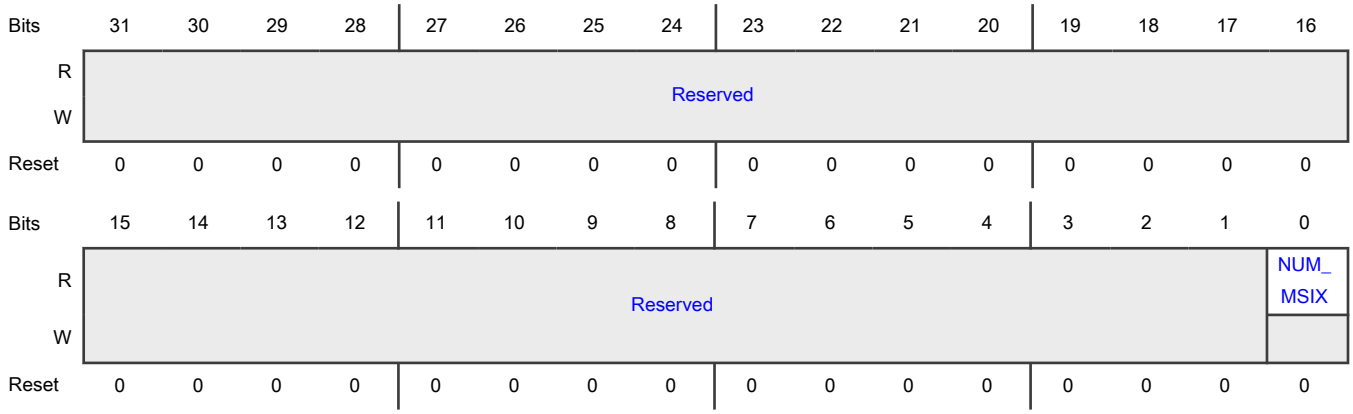
Offset

Register	Offset
EMDIOMCR	314h

Function

This is the EMDIO MSI-X configuration register.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 NUM_MSIX	Number of MSI-X vectors supported. Formula: NUM_MSIX+1 Range: 1

53.4.6.3.27 EMDIO config header device ID and vendor ID register (EMDIO_CFH_DIDVID)

Offset

Register	Offset
EMDIO_CFH_DIDVID	320h

Function

This is the EMDIO PCI Device and Vendor ID register.

NOTE

This register is continuously sampled, any changes will have immediate effect on reading the PCI function Type0 header.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DEVICE_ID															
Reset	1	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VENDOR_ID															
Reset	0	0	0	1	1	0	0	1	0	1	0	1	0	1	1	1

Fields

Field	Function
31-16 DEVICE_ID	Device ID This field identifies the device ID of the device shown in the PCIe Device ID Register (02h).
15-0 VENDOR_ID	Vendor ID This field identifies the manufacturer of the device as shown in the PCIe Vendor ID Register (00h). The default value will indicate Freescale (0x1957).

53.4.6.3.28 EMDIO config header subsystem ID and subsystem vendor ID register (EMDIO_CFH_SIDSVID)

Offset

Register	Offset
EMDIO_CFH_SIDSVID	324h

Function

This is the EMDIO PCI Subsystem ID and Subsystem Vendor ID register.

NOTE

This register is continuously sampled, any changes will have immediate effect on reading the PCI function Type0 header.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SUBSYSTEM_DEVICE_ID															
Reset	1	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SUBSYSTEM_VENDOR_ID															
Reset	0	0	0	1	1	0	0	1	0	1	0	1	0	1	1	1

Fields

Field	Function
31-16 SUBSYSTEM_DEVICE_ID	Subsystem Device ID This field identifies the particular subsystem as shown in the PCIe Subsystem ID Register (2Eh).
15-0 SUBSYSTEM_VENDOR_ID	Subsystem Vendor ID This field identifies the manufacturer of the subsystem as shown in the PCIe Subsystem Vendor ID Register (2Ch). The default value is the same as EMDIO_CFH_DIDVID[VENDOR_ID].

53.4.6.3.29 EMDIO boot loader parameter register a (EMDIOBLPR0 - EMDIOBLPR1)

Offset

Register	Offset
EMDIOBLPR0	348h
EMDIOBLPR1	34Ch

Function

This register provides a mean for boot S/W (Pre-Boot Loader, boot ROM) to communicate parameters with running (device driver) software. Runtime software uses read-only global register FBLPR*a*.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PARAM_VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PARAM_VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PARAM_VAL	Boot loader parameter value.

53.4.6.3.30 EMDIO configuration register (EMDIO_CFG)

Offset

Register	Offset
EMDIO_CFG	350h

Function

This is the EMDIO configuration register.

NOTE

This value is writable in IERB (by pre-boot initialization), and is then immediately reflected in EMDIO register MDIO_CFG.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										MDC_ MODE	MDIO_ MO...	Reserved			
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5 MDC_MODE	MDC pin mode 0 Push-pull 1 Open drain
4 MDIO_MODE	MDIO pin mode 0 Push-pull 1 Open drain
3-0 —	Reserved

53.4.6.3.31 Timer 0 binding configuration register (T0BCR)

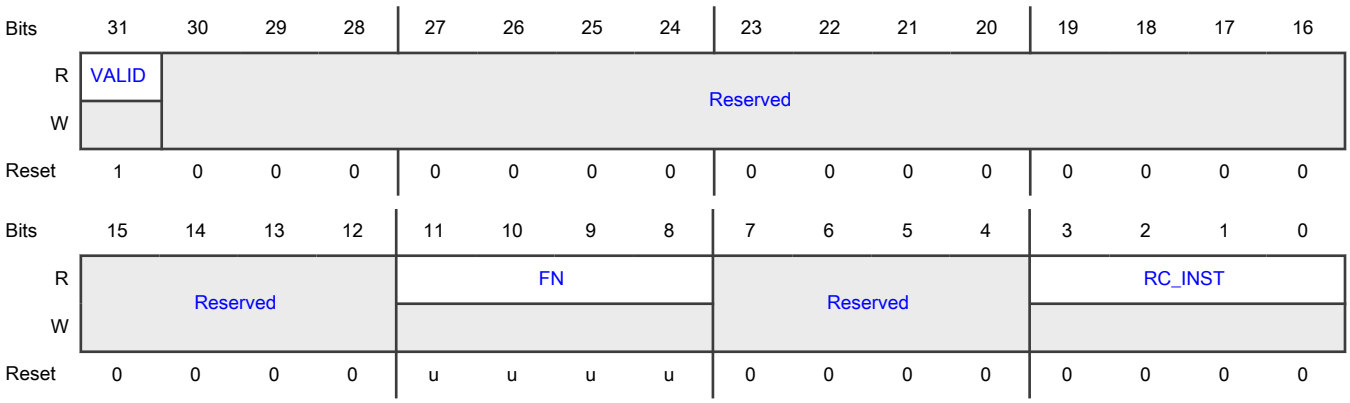
Offset

Register	Offset
T0BCR	400h

Function

This is the timer binding configuration register.

Diagram



Fields

Field	Function
31 VALID	If set, this timer function is valid.
30-12 —	Reserved
11-8 FN	PCI device function number. **NOTE** For assignment of function number, see PCIe Function Number Assignment.
7-4 —	Reserved
3-0 RC_INST	Root complex instance number.

53.4.6.3.32 Timer 0 MSI-X configuration register (T0MCR)

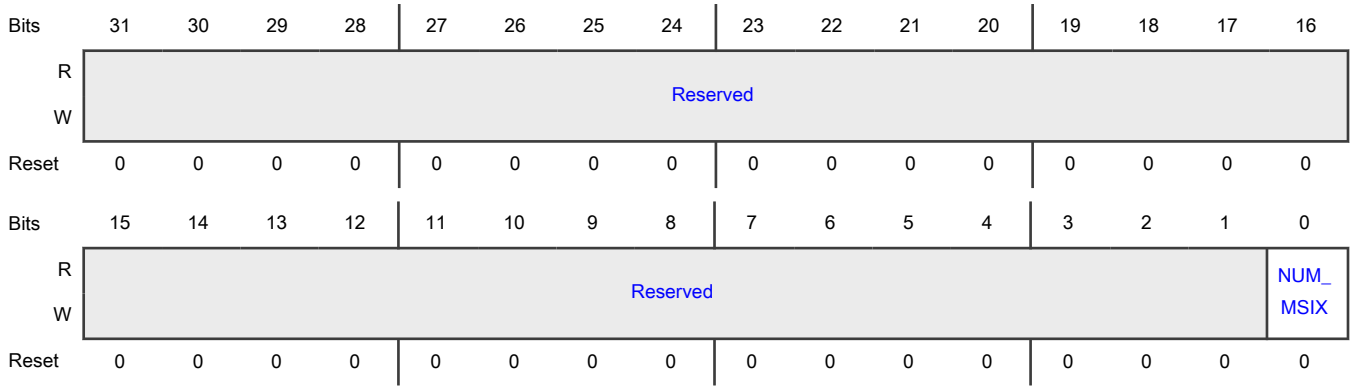
Offset

Register	Offset
T0MCR	414h

Function

This is the timer MSI-X configuration register.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 NUM_MSIX	Number of MSI-X vectors supported for timer function. Formula: NUM_MSIX+1 Range: 1..2 **NOTE** Reset value of timer register TMRCAPR[NUM_MSIX]. If total number of allocated MSI-X vectors exceed CAPR2[NUM_MSIX], error is indicated by NETCSR[ERROR].

53.4.6.3.33 Timer 0 config header device ID and vendor ID register (T0_CFH_DIDVID)

Offset

Register	Offset
T0_CFH_DIDVID	420h

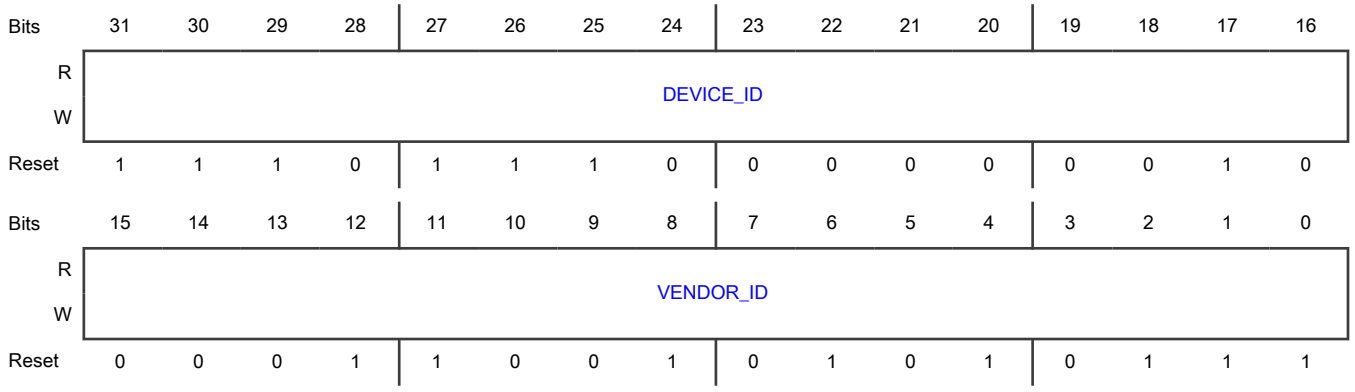
Function

This is the timer PCI Device and Vendor ID register.

NOTE

This register is continuously sampled, any changes will have immediate effect on reading the PCI function Type0 header.

Diagram



Fields

Field	Function
31-16 DEVICE_ID	Device ID This field identifies the device ID of the device shown in the PCIe Device ID Register (02h).
15-0 VENDOR_ID	Vendor ID This field identifies the manufacturer of the device as shown in the PCIe Vendor ID Register (00h). The default value will indicate Freescale (0x1957).

53.4.6.3.34 Timer 0 config header subsystem ID and subsystem vendor ID register (T0_CFH_SIDSVID)

Offset

Register	Offset
T0_CFH_SIDSVID	424h

Function

This is the timer PCI Subsystem ID and Subsystem Vendor ID register.

NOTE

This register is continuously sampled, any changes will have immediate effect on reading the PCI function Type0 header.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SUBSYSTEM_DEVICE_ID															
Reset	1	1	1	0	1	1	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SUBSYSTEM_VENDOR_ID															
Reset	0	0	0	1	1	0	0	1	0	1	0	1	0	1	1	1

Fields

Field	Function
31-16 SUBSYSTEM_ DEVICE_ID	Subsystem Device ID This field identifies the particular subsystem as shown in the PCIe Subsystem ID Register (2Eh).

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 SUBSYSTEM_VENDOR_ID	Subsystem Vendor ID This field identifies the manufacturer of the subsystem as shown in the PCIe Subsystem Vendor ID Register (2Ch). The default value is the same as T*a*_CFH_DIDVID[VENDOR_ID].

53.4.6.3.35 Timer 0 boot loader parameter register b (T0BLPR0 - T0BLPR1)

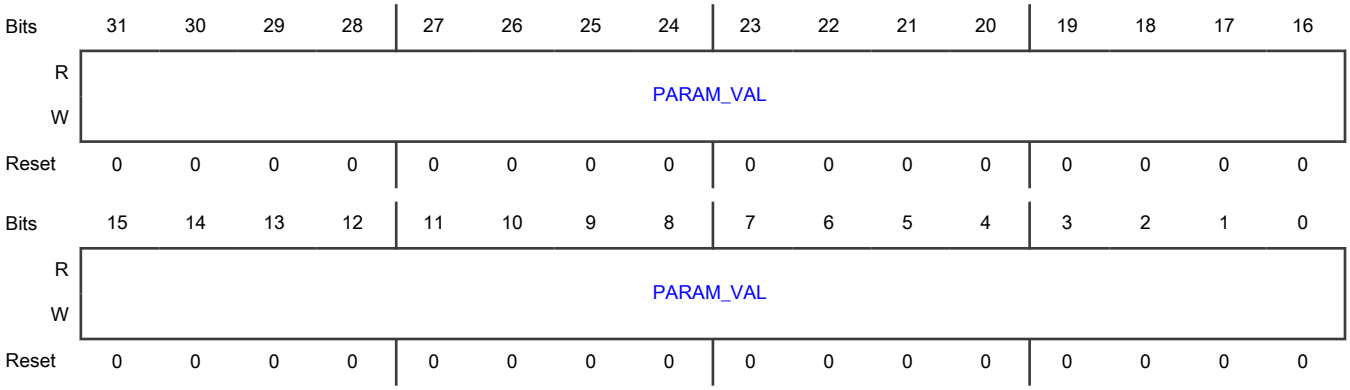
Offset

Register	Offset
T0BLPR0	448h
T0BLPR1	44Ch

Function

This register provides a mean for boot S/W (Pre-Boot Loader, boot ROM) to communicate parameters with running (device driver) software. Runtime software uses read-only global register TMRBLPR*a*.

Diagram



Fields

Field	Function
31-0 PARAM_VAL	Boot loader parameter value.

53.4.6.3.36 Link a capability register (L0CAPR - L4CAPR)

Offset

Register	Offset
L0CAPR	1000h
L1CAPR	1040h
L2CAPR	1080h
L3CAPR	10C0h
L4CAPR	1100h

Function

This is the link capability register. L*a*BCR defines the binding of this link to a function's port. The value of this register is immediately reflected in the function's port register PCAPR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		CBS	TGS	NUM_CG				Reserved				NUM_Q			
W																
Reset	0	0	1	1	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NUM_TC				Reserved							LINK_TY...	Reserved			
W																
Reset	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 CBS	Credit Based Shaping When set, Credit Based Shaping (CBS i.e: 802.1Qav) is supported.
28 TGS	Time Gate Scheduling When set, time gate scheduling is supported on this port. That is, Enhanced Traffic Selection from IEEE 802.1Qbv standard. See TGSTCAPR for more info.
27-24	Number of congestion groups supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
NUM_CG	Formula: NUM_CG+1 **NOTE** Valid if link end is bound to a switch port.
23-20 —	Reserved
19-16 NUM_Q	Number of Egress Traffic Management (ETM) class queues supported Formula: NUM_Q+1 **NOTE** Valid if link is bound to a switch.
15-12 NUM_TC	Number of Traffic Classes (TCs) supported Formula: NUM_TC+1
11-5 —	Reserved
4 LINK_TYPE	Indicates the link type 0 External Link 1 Pseudo Link
3-0 —	Reserved

53.4.6.3.37 Link a MAC capability register (L0MCAPR - L4MCAPR)

Offset

Register	Offset
L0MCAPR	1004h
L1MCAPR	1044h
L2MCAPR	1084h
L3MCAPR	10C4h
L4MCAPR	1104h

Function

This is the link MAC capability register. L*a*BCR defines the binding of this link to a function's port. The value of this register is reflected in the function's port register PMCAPR. If L*a*CAPR[LINK_TYPE]=1, both end of the link has the same value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				MII_PROT				Reserved							
W																
Reset	0	0	0	0	u[1]	u	u	u	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			MIN_MPDU	Reserved	FP		HD	Reserved	PIPG	EFPAD		Reserved	MAC_VAR		
W																
Reset	0	0	0	1	0	1	0	1	0	1	1	0	0	0	0	1

1. The MII protocol is an input to NETC for each Ethernet port which is specified by a configuration register on the SoC.

Fields

Field	Function
31-28 —	Reserved
27-24 MII_PROT	Indicates the MII protocol supported 0 MII 1 RMII 2 RGMII 3 GMII All other settings reserved. **NOTE** Valid if MAC_VAR=1. Error is flagged by NETCSR[ERROR] if reserved value is specified.
23-13 —	Reserved
12 MIN_MPDU	Minimum MAC Protocol Data Unit (PDU) size check 0 Minimum MAC PDU size check is not supported 1 Minimum MAC PDU size check is supported. See PM*a*_MINFRM register for more information.
11 —	Reserved
10-9 FP	Indicates if frame preemption is supported 0 Frame preemption is not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1 Standard frame preemption is supported 2 Enhanced frame preemption is supported. That is, the ability to preempt transmission of a frame at any byte boundary greater than 64B. That is, not limited to preempt only at multiple of 64B as dictated by the standard. 3 Reserved **NOTE** Valid if MAC_VAR=1
8 HD	Half Duplex capability Indicates if half duplex mode is supported on this link. 0 Half duplex mode is not supported 1 Half duplex mode is supported **NOTE** Valid if MAC_VAR=1
7 —	Reserved
6 PIPG	Configurable preamble/IPG capability Indicates if configurable Preamble and IPG is supported 0 Static: Inter Frame Gap (IPG) size is 12B and Preamble is 7B 1 Configurable IPG and Preamble size **NOTE** Valid if MAC_VAR=1
5-4 EFPAD	Egress frame padding capability Egress frame padding capability indicates if egress frames smaller than 64B are padded with zeroes. 0 Frame padding is not supported. 1 All frames less than 64B are padded. 2 Configurable frame padding per port. Note: If frame preemption is supported, both the express and pre-emptable MACs padding option are configurable independently. 3 Reserved. **NOTE** Not applicable for switch port when MAC_VAR=0.
3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-0 MAC_VAR	MAC Variant 0 Pseudo MAC 1 1G mEMAC, supporting 10/100 Mbps, 1 Gbps All other values reserved.

53.4.6.3.38 Link a I/O capability register (L0IOCAPR - L1IOCAPR)

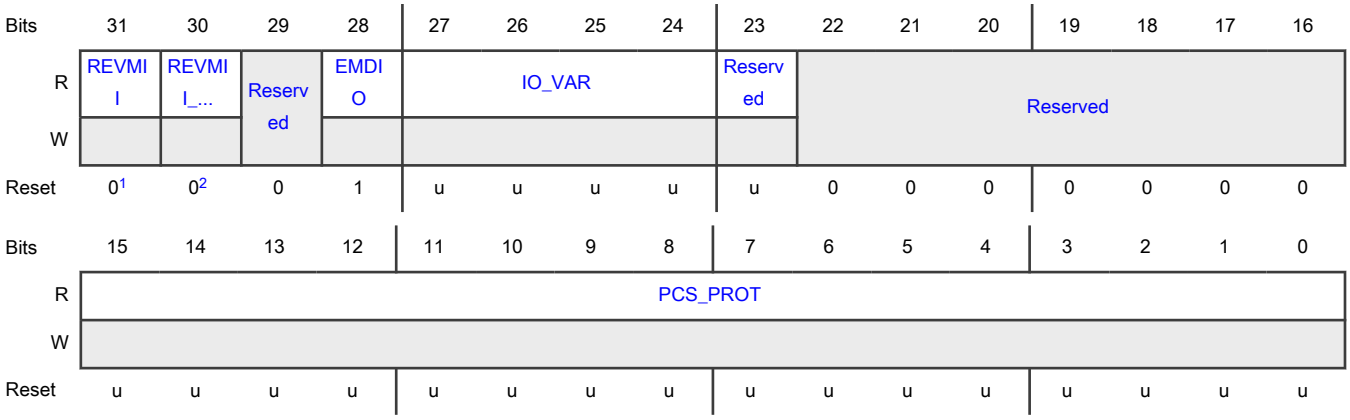
Offset

Register	Offset
L0IOCAPR	1008h
L1IOCAPR	1048h

Function

Link I/O capability register. L*a*BCR defines the binding of this link to a function's port. The value of this register is reflected in the function's port register PIOCAPR.

Diagram



1. RevMII support depends on SoC capability
2. RevMII rate support depends on SoC capability

Fields

Field	Function
31 REVMII	Reverse Mode Device Configuration If set, the device has configured link *a* for Reverse operation and IMDIO is supported to access RevMII registers. SW must set PM*a*_IF_MODE[REVMII] to match.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 REVMII_RATE	RevMII MII rate If REVMII=1 and MII_PROT=MII, then 0: MII interface is operating at 100 Mbps 1: MII interface is operating at 10 Mbps If RevMII=0 or MII_PROT!=MII, this field has no meaning.
29 —	Reserved
28 EMDIO	External MDIO supported.
27-24 IO_VAR	IO Variants supported 0: None 15: Custom All other values reserved. **NOTE** Error is flagged by NETCSR[ERROR] if reserved value is specified.
23 —	Reserved
22-16 —	Reserved
15-0 PCS_PROT	PCS protocols supported xxxx xxxx xxxx xxx1: 1G SGMII xxxx xxxx xxxx xx1x: OC-SGMII (i.e. 2.5G SGMII) xxxx xxxx xxxx x1xx: QGSMII xxxx xxxx xxxx 1xxx: XFI xxxx xxxx xxx1 xxxx: SFI xxxx xxxx xx1x xxxx: 10GBase-KR xxxx xxxx x1xx xxxx: 10G-SXGMII 0000 0000 0000 0000: No PCS protocols supported If PCS_PROT != 0, then IMDIO is also supported to access PCS registers

53.4.6.3.39 Link 0 binding configuration register (L0BCR)

Offset

Register	Offset
L0BCR	1010h

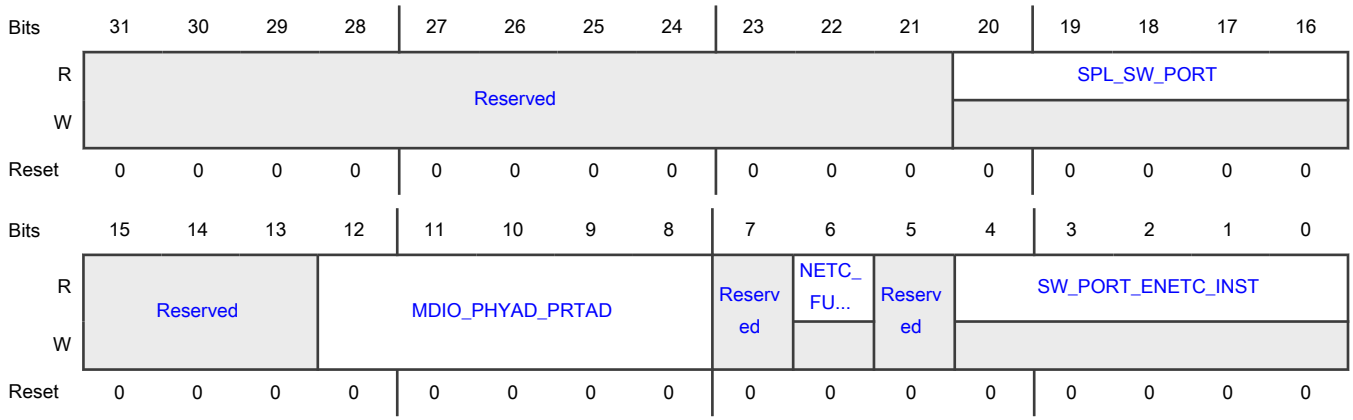
Function

Register configures the binding of link ends to NETC function. If this is a pseudo link, both link ends are bounded.

It is assumed that:

1. both ends of the pseudo link has the same LINK_TYPE
2. primary link end specifies its NETC function
3. if this is a pseudo link, then the other link end's NETC function is SWITCH
4. both ends of pseudo link's has the same ETM capabilities

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 SPL_SW_PORT	Secondary pseudo link end's switch port number. Valid if L*a*CAPR[LINK_TYPE]=Pseudo MAC. **NOTE** If more than one link refers to same SPL_SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are deemed unusable.
15-13 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
12-8 MDIO_PHYAD_PRTAD	This value indicate an MDIO PHY address. Depending on the link mode setting L*a*lOCAPR[REVMII], the value has the following meaning: 0 Indicates link external MDIO PHY's address for clause 22 and external MDIO port address for clause 45. 1 RevMII mode - Indicates link MDIO PHY's address when operating as a slave.
7 —	Reserved
6 NETC_FUNC	Primary link end's NETC Function Type. 0: Switch 1: ENETC **NOTE** NETCv3 does not currently support NETC_FUNC=Switch & LINK_TYPE=Pseudo MAC. If set, error is indicated by NETCSR[ERROR] and this link becomes unusable.
5 —	Reserved
4-0 SW_PORT_ENETC_INST	NETC_FUNC=ENETC: Primary link end's ENETC instance number mapped to this link end. NETC_FUNC=Switch: Primary link end's switch port number mapped to primary link end. **NOTE** If more than one link end refers to same ENETC_INST or SW_INST/SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are unusable.

53.4.6.3.40 Link a transmit byte credit comfort threshold register (L0TXBCCTR - L5TXBCCTR)

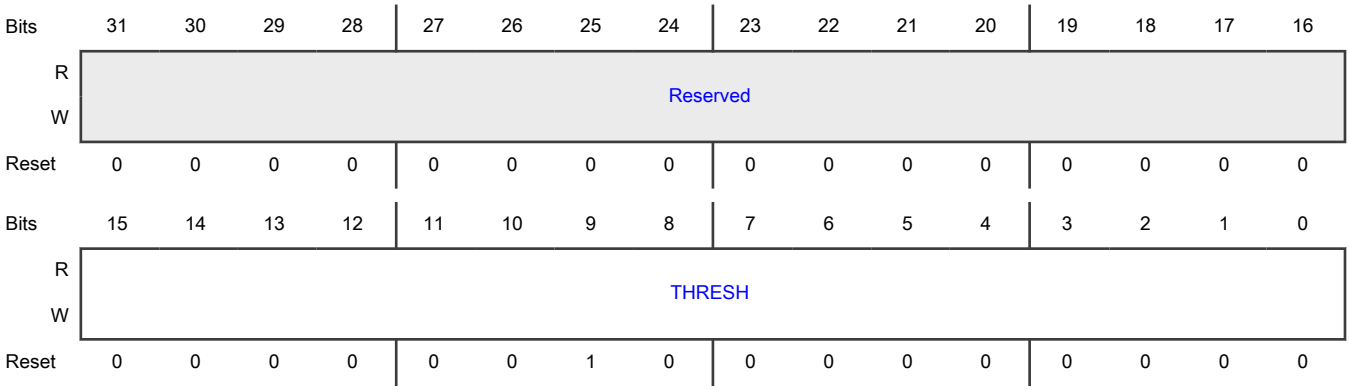
Offset

Register	Offset
L0TXBCCTR	1014h
L1TXBCCTR	1054h
L2TXBCCTR	1094h
L3TXBCCTR	10D4h
L4TXBCCTR	1114h
L5TXBCCTR	1154h

Function

This is the link transmit byte credit comfort threshold register. This register is used only if the link is bound to a switch port. This register is not used if the link is bound to an ENETC port.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 THRESH	Link's transmit byte credit comfort threshold from ETM towards port. Default value is 512B.

53.4.6.3.41 Link a end 0 MAC address register 0 (L0E0MAR0 - L5E0MAR0)

Offset

Register	Offset
L0E0MAR0	1020h
L1E0MAR0	1060h
L2E0MAR0	10A0h
L3E0MAR0	10E0h
L4E0MAR0	1120h
L5E0MAR0	1160h

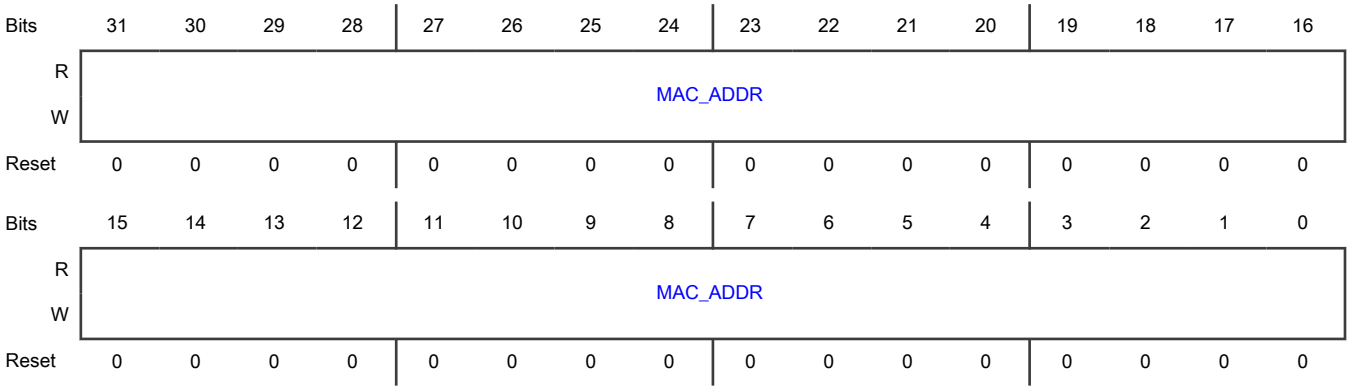
Function

This is the primary link end's MAC address register 0. It determines the initial value for port register PMAR0 in the ENETC or switch function as determined by the link binding.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to port equivalent register during IERB lock. All updates after this must be done through the corresponding port register.

Diagram



Fields

Field	Function
31-0 MAC_ADDR	MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset. This field contains the 4 most significant bytes of the MAC address. For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then PMAR0 equals 14131211h.

53.4.6.3.42 Link a end 0 MAC address register 1 (L0E0MAR1 - L5E0MAR1)

Offset

Register	Offset
L0E0MAR1	1024h
L1E0MAR1	1064h
L2E0MAR1	10A4h
L3E0MAR1	10E4h
L4E0MAR1	1124h
L5E0MAR1	1164h

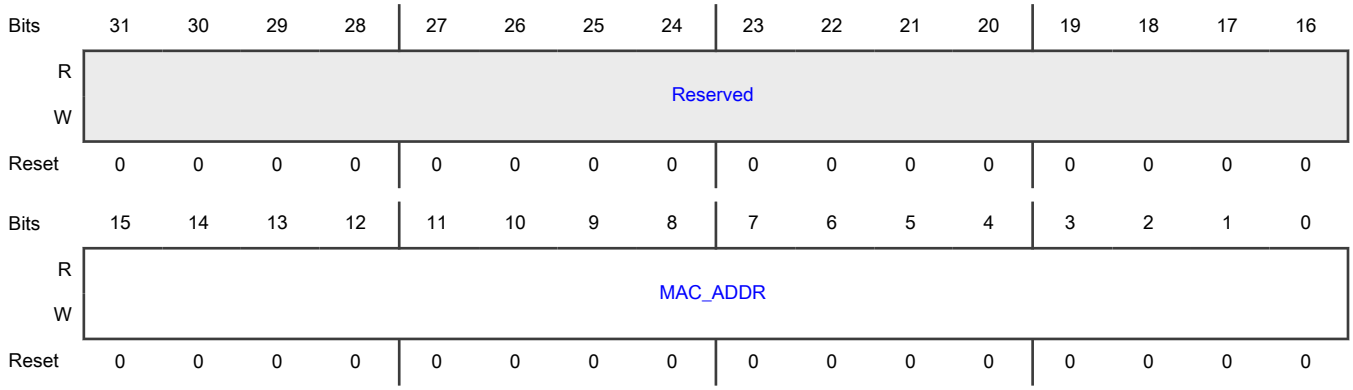
Function

This is the primary link end's MAC address register 1. It determines the initial value for port register PMAR1 in the ENETC or switch function as determined by the link binding.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to port equivalent register during IERB lock. All updates after this must be done through the corresponding port register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 MAC_ADDR	MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset. This field contains the 2 least significant bytes of the MAC address. For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then PMAR1 equals xxxx1615h (where x should be set to 0)

53.4.6.3.43 Link 1 binding configuration register (L1BCR)

Offset

Register	Offset
L1BCR	1050h

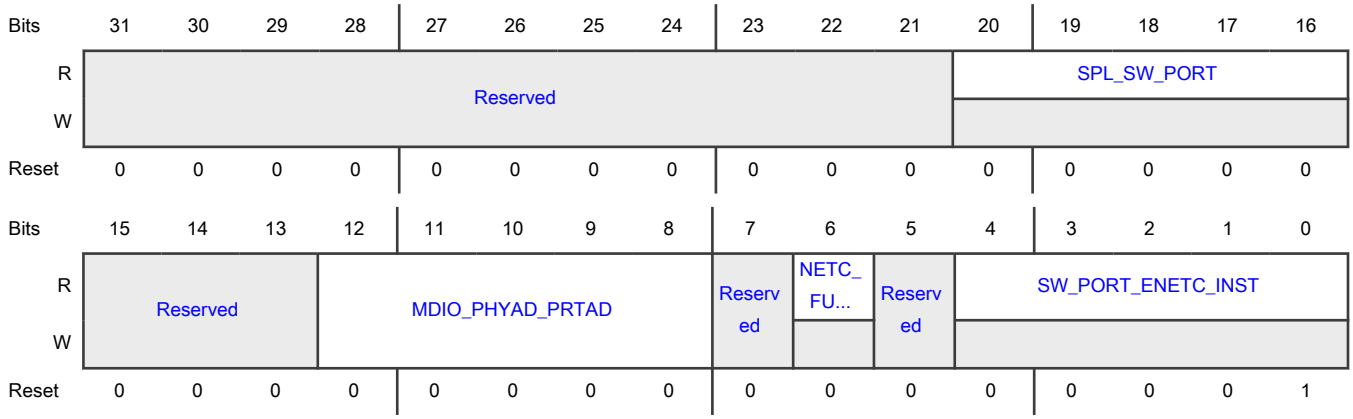
Function

Register configures the binding of link ends to NETC function. If this is a pseudo link, both link ends are bounded.

It is assumed that:

1. both ends of the pseudo link has the same LINK_TYPE
2. primary link end specifies its NETC function
3. if this is a pseudo link, then the other link end's NETC function is SWITCH
4. both ends of pseudo link's has the same ETM capabilities

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 SPL_SW_PORT	Secondary pseudo link end's switch port number. Valid if L*a*CAPR[LINK_TYPE]=Pseudo MAC. **NOTE** If more than one link refers to same SPL_SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are deemed unusable.
15-13 —	Reserved
12-8 MDIO_PHYAD_PRTAD	This value indicate an MDIO PHY address. Depending on the link mode setting L*a*lOCAPR[REVMII], the value has the following meaning: 0 Indicates link external MDIO PHY's address for clause 22 and external MDIO port address for clause 45. 1 RevMII mode - Indicates link MDIO PHY's address when operating as a slave.
7 —	Reserved
6 NETC_FUNC	Primary link end's NETC Function Type. 0: Switch 1: ENETC **NOTE** NETCv3 does not currently support NETC_FUNC=Switch & LINK_TYPE=Pseudo MAC. If set, error is indicated by NETCSR[ERROR] and this link becomes unusable.
5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4-0 SW_PORT_ENETC_INST	NETC_FUNC=ENETC: Primary link end's ENETC instance number mapped to this link end. NETC_FUNC=Switch: Primary link end's switch port number mapped to primary link end. **NOTE** If more than one link end refers to same ENETC_INST or SW_INST/SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are unusable.

53.4.6.3.44 Link a I/O capability register (L2IOCAPR - L4IOCAPR)

Offset

Register	Offset
L2IOCAPR	1088h
L3IOCAPR	10C8h
L4IOCAPR	1108h

Function

Link I/O capability register. L*a*BCR defines the binding of this link to a function's port. The value of this register is reflected in the function's port register PIOCAPR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REVMII	REVMII_...	Reserved	EMDIO	IO_VAR				Reserved	Reserved						
W																
Reset	u[1]	u[2]	0	1	u	u	u	u	u	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCS_PROT															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

1. RevMII support depends on SoC capability
2. RevMII rate support depends on SoC capability

Fields

Field	Function
31 REVMII	Reverse Mode Device Configuration If set, the device has configured link *a* for Reverse operation and IMDIO is supported to access RevMII registers. SW must set PM*a*_IF_MODE[REVMII] to match.
30 REVMII_RATE	RevMII MII rate If REVMII=1 and MII_PROT=MII, then 0: MII interface is operating at 100 Mbps 1: MII interface is operating at 10 Mbps If RevMII=0 or MII_PROT!=MII, this field has no meaning.
29 —	Reserved
28 EMDIO	External MDIO supported.
27-24 IO_VAR	IO Variants supported 0: None 15: Custom All other values reserved. **NOTE** Error is flagged by NETCSR[ERROR] if reserved value is specified.
23 —	Reserved
22-16 —	Reserved
15-0 PCS_PROT	PCS protocols supported xxxx xxxx xxxx xxx1: 1G SGMII xxxx xxxx xxxx xx1x: OC-SGMII (i.e. 2.5G SGMII) xxxx xxxx xxxx x1xx: QGSMII xxxx xxxx xxxx 1xxx: XFI xxxx xxxx xxx1 xxxx: SFI xxxx xxxx xx1x xxxx: 10GBase-KR xxxx xxxx x1xx xxxx: 10G-SXGMII 0000 0000 0000 0000: No PCS protocols supported If PCS_PROT != 0, then IMDIO is also supported to access PCS registers

53.4.6.3.45 Link 2 binding configuration register (L2BCR)

Offset

Register	Offset
L2BCR	1090h

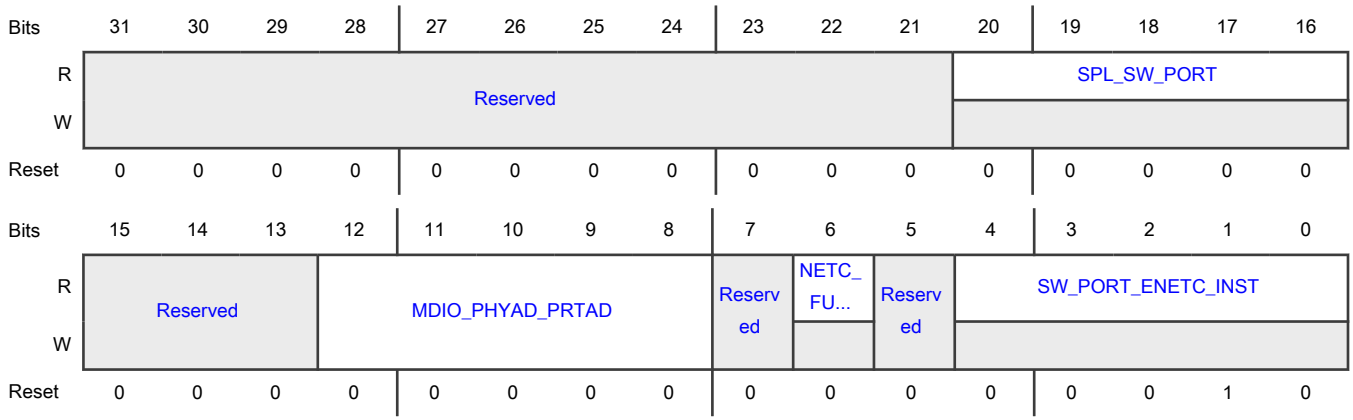
Function

Register configures the binding of link ends to NETC function. If this is a pseudo link, both link ends are bounded.

It is assumed that:

1. both ends of the pseudo link has the same LINK_TYPE
2. primary link end specifies its NETC function
3. if this is a pseudo link, then the other link end's NETC function is SWITCH
4. both ends of pseudo link's has the same ETM capabilities

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 SPL_SW_PORT	Secondary pseudo link end's switch port number. Valid if L*a*CAPR[LINK_TYPE]=Pseudo MAC. **NOTE** If more than one link refers to same SPL_SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are deemed unusable.
15-13 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
12-8 MDIO_PHYAD_PRTAD	This value indicate an MDIO PHY address. Depending on the link mode setting L*a*lOCAPR[REVMII], the value has the following meaning: 0 Indicates link external MDIO PHY's address for clause 22 and external MDIO port address for clause 45. 1 RevMII mode - Indicates link MDIO PHY's address when operating as a slave.
7 —	Reserved
6 NETC_FUNC	Primary link end's NETC Function Type. 0: Switch 1: ENETC **NOTE** NETCv3 does not currently support NETC_FUNC=Switch & LINK_TYPE=Pseudo MAC. If set, error is indicated by NETCSR[ERROR] and this link becomes unusable.
5 —	Reserved
4-0 SW_PORT_ENETC_INST	NETC_FUNC=ENETC: Primary link end's ENETC instance number mapped to this link end. NETC_FUNC=Switch: Primary link end's switch port number mapped to primary link end. **NOTE** If more than one link end refers to same ENETC_INST or SW_INST/SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are unusable.

53.4.6.3.46 Link 3 binding configuration register (L3BCR)

Offset

Register	Offset
L3BCR	10D0h

Function

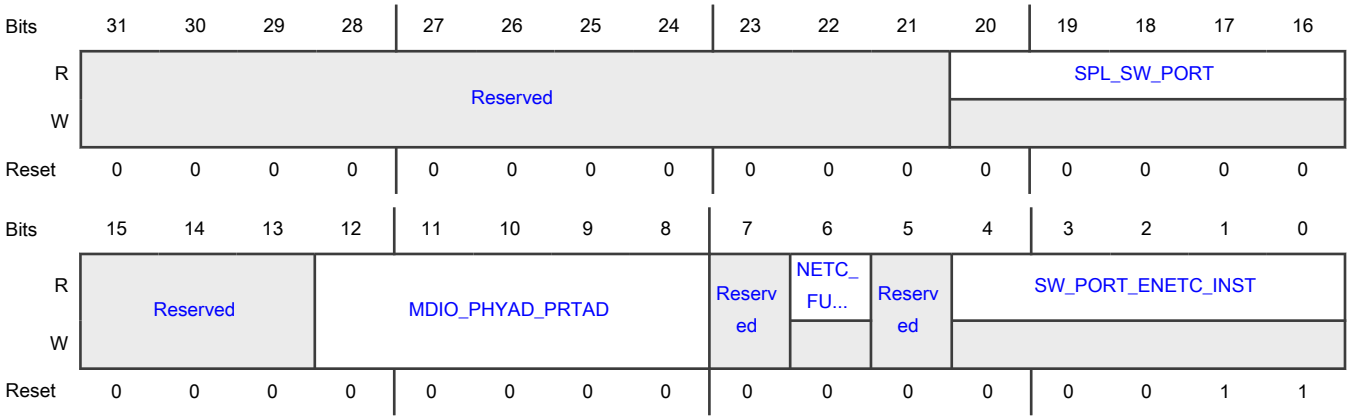
Register configures the binding of link ends to NETC function. If this is a pseudo link, both link ends are bounded.

It is assumed that:

1. both ends of the pseudo link has the same LINK_TYPE
2. primary link end specifies its NETC function

3. if this is a pseudo link, then the other link end's NETC function is SWITCH
4. both ends of pseudo link's has the same ETM capabilities

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 SPL_SW_PORT	Secondary pseudo link end's switch port number. Valid if LaCAPR[LINK_TYPE]=Pseudo MAC. **NOTE** If more than one link refers to same SPL_SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are deemed unusable.
15-13 —	Reserved
12-8 MDIO_PHYAD_PRTAD	This value indicate an MDIO PHY address. Depending on the link mode setting LaIOCAPR[REVMII], the value has the following meaning: 0 Indicates link external MDIO PHY's address for clause 22 and external MDIO port address for clause 45. 1 RevMII mode - Indicates link MDIO PHY's address when operating as a slave.
7 —	Reserved
6 NETC_FUNC	Primary link end's NETC Function Type. 0: Switch 1: ENETC

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE NETCv3 does not currently support NETC_FUNC=Switch & LINK_TYPE=Pseudo MAC. If set, error is indicated by NETCSR[ERROR] and this link becomes unusable.
5 —	Reserved
4-0 SW_PORT_ENETC_INST	NETC_FUNC=ENETC: Primary link end's ENETC instance number mapped to this link end. NETC_FUNC=Switch: Primary link end's switch port number mapped to primary link end. **NOTE** If more than one link end refers to same ENETC_INST or SW_INST/SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are unusable.

53.4.6.3.47 Link 4 binding configuration register (L4BCR)

Offset

Register	Offset
L4BCR	1110h

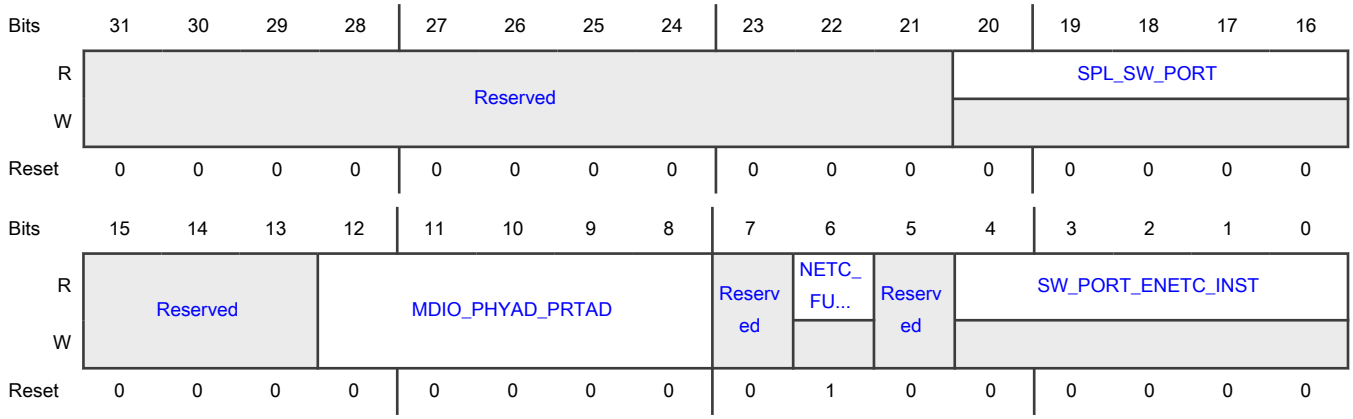
Function

Register configures the binding of link ends to NETC function. If this is a pseudo link, both link ends are bounded.

It is assumed that:

1. both ends of the pseudo link has the same LINK_TYPE
2. primary link end specifies its NETC function
3. if this is a pseudo link, then the other link end's NETC function is SWITCH
4. both ends of pseudo link's has the same ETM capabilities

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 SPL_SW_PORT	Secondary pseudo link end's switch port number. Valid if L*a*CAPR[LINK_TYPE]=Pseudo MAC. **NOTE** If more than one link refers to same SPL_SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are deemed unusable.
15-13 —	Reserved
12-8 MDIO_PHYAD_PRTAD	This value indicate an MDIO PHY address. Depending on the link mode setting L*a*lOCAPR[REVMII], the value has the following meaning: 0 Indicates link external MDIO PHY's address for clause 22 and external MDIO port address for clause 45. 1 RevMII mode - Indicates link MDIO PHY's address when operating as a slave.
7 —	Reserved
6 NETC_FUNC	Primary link end's NETC Function Type. 0: Switch 1: ENETC **NOTE** NETCv3 does not currently support NETC_FUNC=Switch & LINK_TYPE=Pseudo MAC. If set, error is indicated by NETCSR[ERROR] and this link becomes unusable.
5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4-0 SW_PORT_ENETC_INST	NETC_FUNC=ENETC: Primary link end's ENETC instance number mapped to this link end. NETC_FUNC=Switch: Primary link end's switch port number mapped to primary link end. **NOTE** If more than one link end refers to same ENETC_INST or SW_INST/SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are unusable.

53.4.6.3.48 Link 5 capability register (L5CAPR)

Offset

Register	Offset
L5CAPR	1140h

Function

This is the link capability register. L*a*BCR defines the binding of this link to a function's port. The value of this register is immediately reflected in the function's port register PCAPR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		CBS	TGS	NUM_CG				Reserved				NUM_Q			
W																
Reset	0	0	1	1	0	1	1	1	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NUM_TC				Reserved							LINK_TY...	Reserved			
W																
Reset	0	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-30	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
29 CBS	Credit Based Shaping When set, Credit Based Shaping (CBS i.e: 802.1Qav) is supported.
28 TGS	Time Gate Scheduling When set, time gate scheduling is supported on this port. That is, Enhanced Traffic Selection from IEEE 802.1Qbv standard. See TGSTCAPR for more info.
27-24 NUM_CG	Number of congestion groups supported Formula: NUM_CG+1 **NOTE** Valid if link end is bound to a switch port.
23-20 —	Reserved
19-16 NUM_Q	Number of Egress Traffic Management (ETM) class queues supported Formula: NUM_Q+1 **NOTE** Valid if link is bound to a switch.
15-12 NUM_TC	Number of Traffic Classes (TCs) supported Formula: NUM_TC+1
11-5 —	Reserved
4 LINK_TYPE	Indicates the link type 0 External Link 1 Pseudo Link
3-0 —	Reserved

53.4.6.3.49 Link 5 MAC capability register (L5MCAPR)

Offset

Register	Offset
L5MCAPR	1144h

Function

This is the link MAC capability register. L*a*BCR defines the binding of this link to a function's port. The value of this register is reflected in the function's port register PMCAPR. If L*a*CAPR[LINK_TYPE]=1, both end of the link has the same value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				MII_PROT				Reserved							
W																
Reset	0	0	0	0	u[1]	u	u	u	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			MIN_MPDU	Reserved	FP		HD	Reserved	PIPG	EFPAD		Reserved	MAC_VAR		
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

1. The MII protocol is an input to NETC for each Ethernet port which is specified by a configuration register on the SoC.

Fields

Field	Function
31-28 —	Reserved
27-24 MII_PROT	Indicates the MII protocol supported 0 MII 1 RMII 2 RGMII 3 GMII All other settings reserved. **NOTE** Valid if MAC_VAR=1. Error is flagged by NETCSR[ERROR] if reserved value is specified.
23-13 —	Reserved
12 MIN_MPDU	Minimum MAC Protocol Data Unit (PDU) size check 0 Minimum MAC PDU size check is not supported 1 Minimum MAC PDU size check is supported. See PM*a*_MINFRM register for more information.
11	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
10-9 FP	Indicates if frame preemption is supported 0 Frame preemption is not supported 1 Standard frame preemption is supported 2 Enhanced frame preemption is supported. That is, the ability to preempt transmission of a frame at any byte boundary greater than 64B. That is, not limited to preempt only at multiple of 64B as dictated by the standard. 3 Reserved **NOTE** Valid if MAC_VAR=1
8 HD	Half Duplex capability Indicates if half duplex mode is supported on this link. 0 Half duplex mode is not supported 1 Half duplex mode is supported **NOTE** Valid if MAC_VAR=1
7 —	Reserved
6 PIPG	Configurable preamble/IPG capability Indicates if configurable Preamble and IPG is supported 0 Static: Inter Frame Gap (IPG) size is 12B and Preamble is 7B 1 Configurable IPG and Preamble size **NOTE** Valid if MAC_VAR=1
5-4 EFPAD	Egress frame padding capability Egress frame padding capability indicates if egress frames smaller than 64B are padded with zeroes. 0 Frame padding is not supported. 1 All frames less than 64B are padded. 2 Configurable frame padding per port. Note: If frame preemption is supported, both the express and pre-emptable MACs padding option are configurable independently. 3 Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Not applicable for switch port when MAC_VAR=0.
3 —	Reserved
2-0 MAC_VAR	MAC Variant 0 Pseudo MAC 1 1G mEMAC, supporting 10/100 Mbps, 1 Gbps All other values reserved.

53.4.6.3.50 Link 5 binding configuration register (L5BCR)

Offset

Register	Offset
L5BCR	1150h

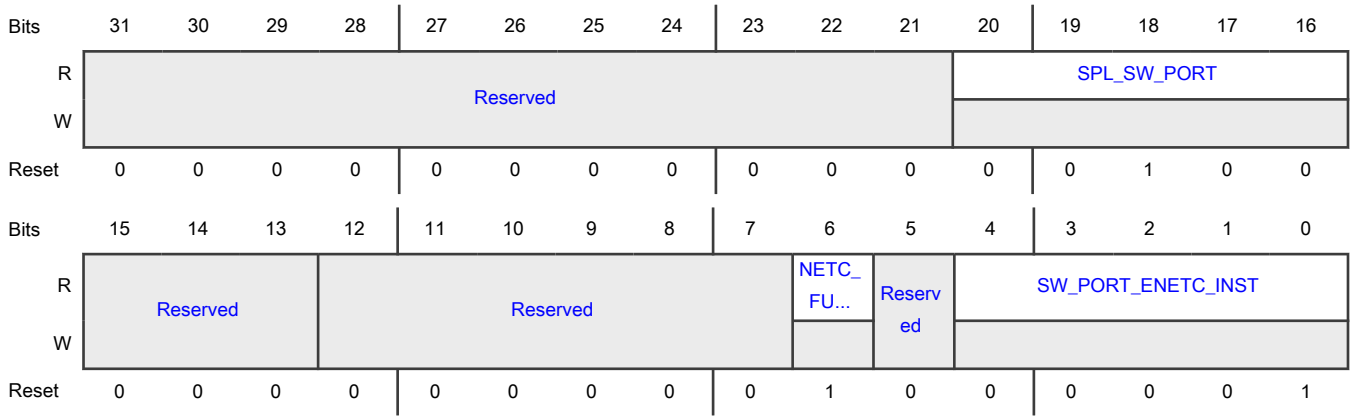
Function

Register configures the binding of link ends to NETC function. If this is a pseudo link, both link ends are bounded.

It is assumed that:

1. both ends of the pseudo link has the same LINK_TYPE
2. primary link end specifies its NETC function
3. if this is a pseudo link, then the other link end's NETC function is SWITCH
4. both ends of pseudo link's has the same ETM capabilities

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 SPL_SW_PORT	Secondary pseudo link end's switch port number. Valid if L*a*CAPR[LINK_TYPE]=Pseudo MAC. **NOTE** If more than one link refers to same SPL_SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are deemed unusable.
15-13 —	Reserved
12-7 —	Reserved
6 NETC_FUNC	Primary link end's NETC Function Type. 0: Switch 1: ENETC **NOTE** NETCv3 does not currently support NETC_FUNC=Switch & LINK_TYPE=Pseudo MAC. If set, error is indicated by NETCSR[ERROR] and this link becomes unusable.
5 —	Reserved
4-0 SW_PORT_ENETC_INST	NETC_FUNC=ENETC: Primary link end's ENETC instance number mapped to this link end. NETC_FUNC=Switch: Primary link end's switch port number mapped to primary link end. **NOTE** If more than one link end refers to same ENETC_INST or SW_INST/SW_PORT, a link mapping error is flagged by NETCSR[ERROR] and the links detecting the misconfiguration are unusable.

53.4.6.3.51 Link 5 end 1 MAC address register 0 (L5E1MAR0)

Offset

Register	Offset
L5E1MAR0	1168h

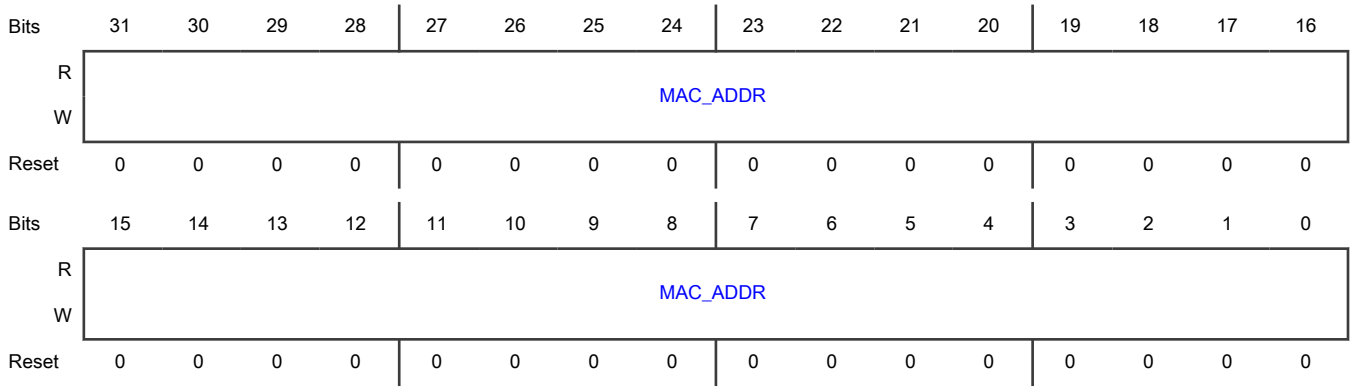
Function

This is the secondary link end's MAC address register 0. It determines the initial value for port register PMAR0 in a switch function as determined by the link binding.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to port equivalent register during IERB lock. All updates after this must be done through the corresponding port register.

Diagram



Fields

Field	Function
31-0 MAC_ADDR	MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset. This field contains the 4 most significant bytes of the MAC address. For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then PMAR0 equals 14131211h.

53.4.6.3.52 Link 5 end 1 MAC address register 1 (L5E1MAR1)

Offset

Register	Offset
L5E1MAR1	116Ch

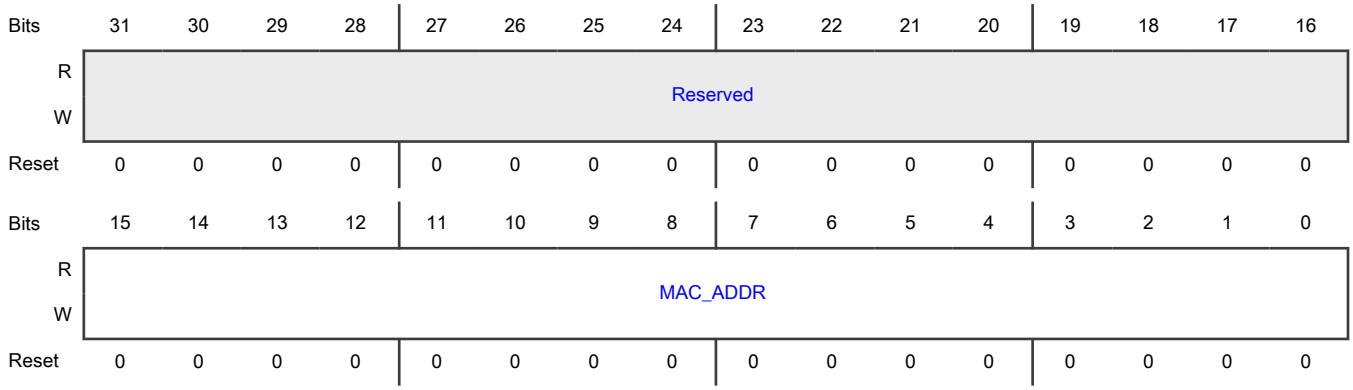
Function

This is the secondary link end's MAC address register 1. It determines the initial value for port register PMAR1 in a switch function as determined by the link binding.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to port equivalent register during IERB lock. All updates after this must be done through the corresponding port register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 MAC_ADDR	MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset. This field contains the 2 least significant bytes of the MAC address. For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then PMAR1 equals xxxx1615h (where x should be set to 0)

53.4.6.3.53 Switch 0 binding configuration register (S0BCR)

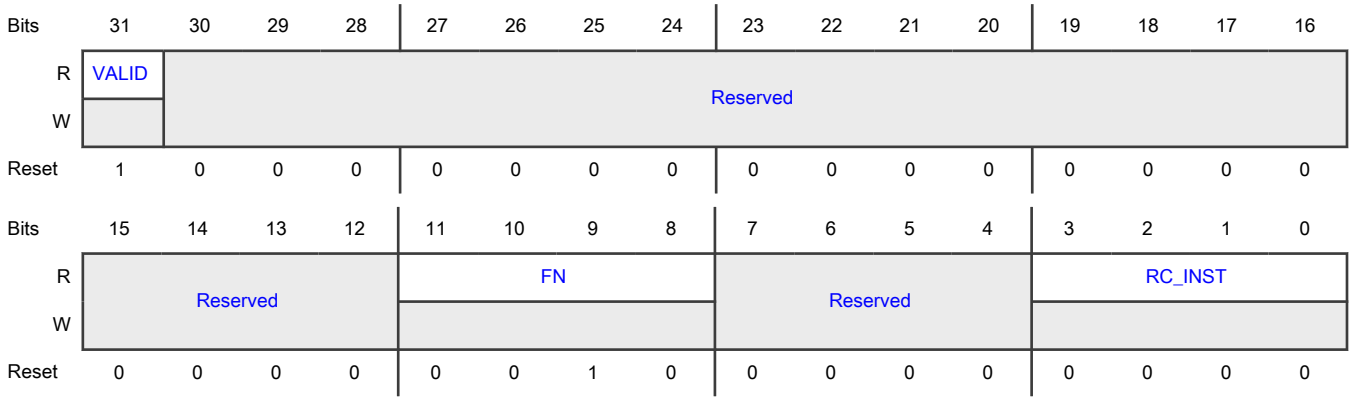
Offset

Register	Offset
S0BCR	2000h

Function

This is the switch binding configuration register.

Diagram



Fields

Field	Function
31 VALID	Set if switch instance is bounded to at least 1 link.
30-12 —	Reserved
11-8 FN	PCI device function number. **NOTE** For assignment of function number, see PCIe Function Number Assignment.
7-4 —	Reserved
3-0 RC_INST	Root complex instance number.

53.4.6.3.54 Switch 0 MSI-X configuration register (S0MCR)

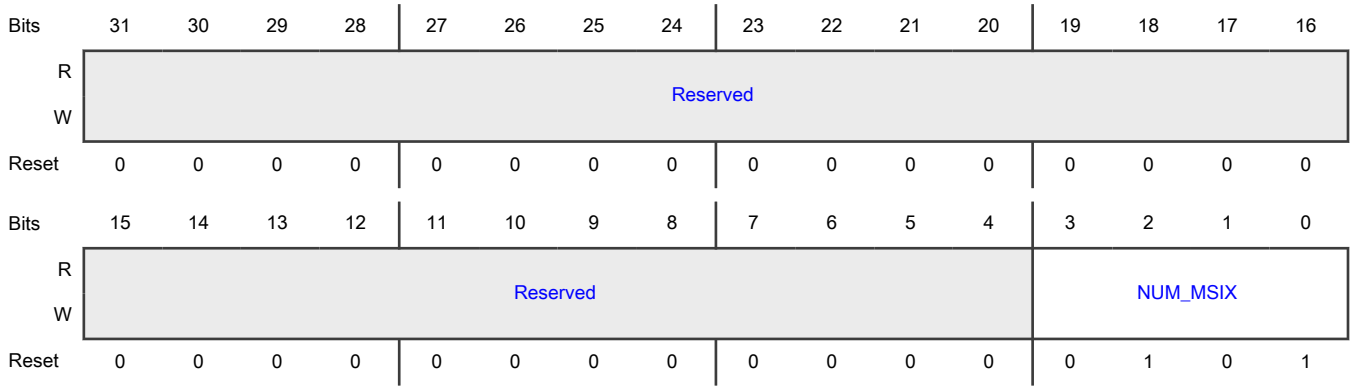
Offset

Register	Offset
S0MCR	2014h

Function

This is the switch MSI-X configuration register.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 NUM_MSIX	Number of MSI-X vectors supported for switch function. Formula: NUM_MSIX+1 Range: 1..16 **NOTE** Reset value of switch register SCAPR0[NUM_MSIX]. If total number of allocated MSI-X vectors exceed CAPR2[NUM_MSIX], error is indicated by NETCSR[ERROR].

53.4.6.3.55 Switch 0 config header device ID and vendor ID register (S0_CFH_DIDVID)

Offset

Register	Offset
S0_CFH_DIDVID	2020h

Function

This is the switch PCI Device and Vendor ID register.

NOTE

This register is continuously sampled, any changes will have immediate effect on reading the PCI function Type0 header.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DEVICE_ID															
Reset	1	1	1	0	1	1	1	0	1	1	1	1	0	0	1	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VENDOR_ID															
Reset	0	0	0	1	1	0	0	1	0	1	0	1	0	1	1	1

Fields

Field	Function
31-16 DEVICE_ID	Device ID This field identifies the device ID of the device shown in the PCIe Device ID Register (02h).
15-0 VENDOR_ID	Vendor ID This field identifies the manufacturer of the device as shown in the PCIe Vendor ID Register (00h). The default value will indicate Freescale (0x1957).

53.4.6.3.56 Switch 0 config header subsystem ID and subsystem vendor ID register (S0_CFH_SIDSVID)

Offset

Register	Offset
S0_CFH_SIDSVID	2024h

Function

This is the switch PCI Subsystem ID and Subsystem Vendor ID register.

NOTE

This register is continuously sampled, any changes will have immediate effect on reading the PCI function Type0 header.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SUBSYSTEM_DEVICE_ID															
Reset	1	1	1	0	1	1	1	0	1	1	1	1	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SUBSYSTEM_VENDOR_ID															
Reset	0	0	0	1	1	0	0	1	0	1	0	1	0	1	1	1

Fields

Field	Function
31-16 SUBSYSTEM_DEVICE_ID	Subsystem Device ID This field identifies the particular subsystem as shown in the PCIe Subsystem ID Register (2Eh).
15-0 SUBSYSTEM_VENDOR_ID	Subsystem Vendor ID This field identifies the manufacturer of the subsystem as shown in the PCIe Subsystem Vendor ID Register (2Ch). The default value is the same as S*a*_CFH_DIDVID[VENDOR_ID].

53.4.6.3.57 Switch 0 command cache attribute register (S0CCAR)

Offset

Register	Offset
S0CCAR	2038h

Function

This is the switch command cache attribute register. It is used to determine system interface attribute settings for reads and writes of command buffer descriptors and data.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to switch register CCAR during IERB lock. All updates after this must be done through the corresponding switch register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	CRDSNP	CRDDOMAIN		CRDCACHE				Reserved	CBD_RDS...	CBD_RDDOMAIN		CBD_RDCACHE			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	CWRSNP	CWRDOMAIN		CWRCACHE				Reserved	CBD_WRS...	CBD_WRDOMAIN		CBD_WRCACHE			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31 —	Reserved
30 CRDSNP	Read data snoop This is the snoop attribute setting used when switch reads command data from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
29-28 CRDDOMAIN	Read data domain This is the domain attribute setting used when switch reads command data from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
27-24 CRDCACHE	Read data cache type This is the cache attribute setting used when switch reads command data from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
23 —	Reserved
22 CBD_RDSNP	Command descriptor read snoop See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
21-20 CBD_RDDOMAIN	Command buffer descriptor read domain This is the domain attribute setting used when switch reads the command buffer descriptor from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-16 CBD_RDCACHE	Command buffer descriptor read cache type This is the cache attribute setting used when switch reads the command buffer descriptor from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
15 —	Reserved
14 CWRSNP	Write data snoop This is the snoop attribute setting used when switch writes command data to memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
13-12 CWRDOMAIN	Write data domain This is the domain attribute setting used when switch writes command data to memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
11-8 CWRCACHE	Write data cache type This is the cache attribute setting used when switch writes command data to memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
7 —	Reserved
6 CBD_WRSNP	Command buffer descriptor write snoop This is the snoop attribute setting used when switch writes the command buffer descriptor in memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
5-4 CBD_WRDOMAIN	Command buffer descriptor write domain This is the domain attribute setting used when switch writes the command buffer descriptor in memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
3-0 CBD_WRCACHE	Command buffer descriptor write cache type This is the cache attribute setting used when switch writes the command buffer descriptor in memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.

53.4.6.3.58 Switch 0 access management qualifier register (S0AMQR)

Offset

Register	Offset
S0AMQR	2040h

Function

This is the switch access management qualifier register. It provides attributes for each memory access transaction performed by the switch instance.

NOTE

Changing the value in this register has immediate effect unless otherwise noted. Care should be taken when making changes to the values for an enabled and operational switch function to ensure that consistent behavior is maintained. It is strongly encouraged, but not required, that the function be disabled before making changes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BMT	Reserved							AWQOS				ARQOS			
W																
Reset	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BMT	Bypass memory translation The bit is an indication to the SMMU to by-pass memory translation whenever the PF is performing memory transactions, effectively handling the memory address as a true physical address. 0 Memory translation 1 Bypass memory translation
30-24 —	Reserved
23-20 AWQOS	Address Write QoS. AWQOS[3:1] signals are controlled by this register field's bits [3:1]. AWQOS[0] is controlled by the HTA block and is 0 for low priority DMA transactions and 1 for high priority DMA transactions. Only HTA Command threads are used for the switch and these threads are always low priority.

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-16 ARQOS	Address Read QoS. ARQOS[3:1] signals are controlled by this register field's bits [3:1]. ARQOS[0] is controlled by the HTA block and is 0 for low priority DMA transactions and 1 for high priority DMA transactions. Only HTA Command threads are used for the switch and these threads are always low priority.
15-0 —	Reserved

53.4.6.3.59 Switch 0 boot loader parameter register b (S0BLPR0 - S0BLPR1)

Offset

Register	Offset
S0BLPR0	2048h
S0BLPR1	204Ch

Function

This register provides a mean for boot S/W (Pre-Boot Loader, boot ROM) to communicate parameters with running (device driver) software. Runtime software uses read-only global register FBLPR*a*.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PARAM_VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PARAM_VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PARAM_VAL	Boot loader parameter value.

53.4.6.3.60 Switch 0 shared memory buffer allotment register (S0SMBAR)

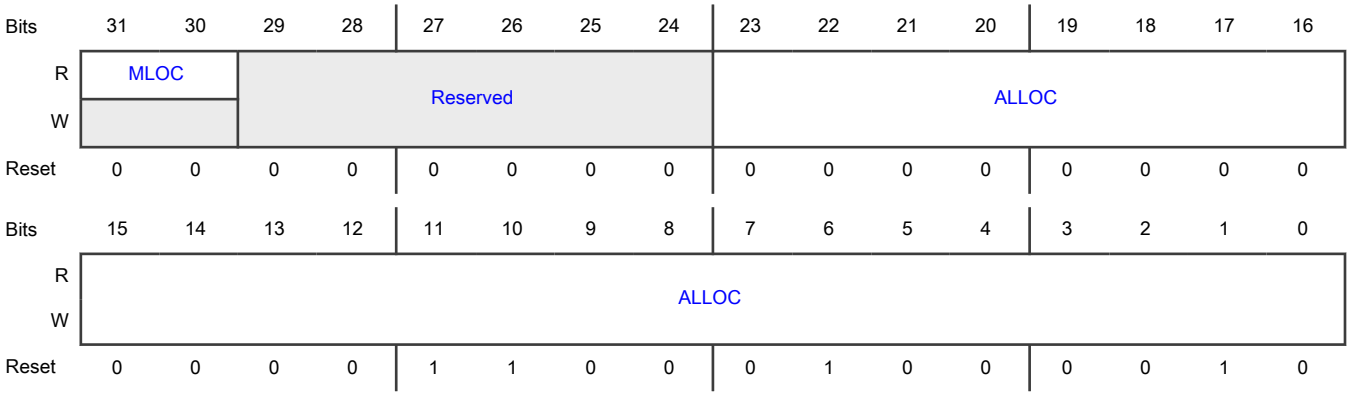
Offset

Register	Offset
S0SMBAR	2060h

Function

This is the switch shared memory buffer allotment register.

Diagram



Fields

Field	Function
31-30 MLOC	Indicates memory location 0: Common memory 1-3: Reserved
29-24 —	Reserved
23-0 ALLOC	Number of words allotted for the switch frame buffering memory. The amount of memory allotted should not be lower than the total number of switch ports multiplied by 2 times the maximum frame size that can be transmitted on the express MAC or the pseudo MAC plus 2 times the maximum frame size that can be transmitted on the preemptable MAC (the later applies only if preemption is enabled/used). **NOTE** Memory is managed via buffer pool mechanism.

53.4.6.3.61 Switch 0 hash table memory allotment register (S0HTMAR)

Offset

Register	Offset
S0HTMAR	2080h

Function

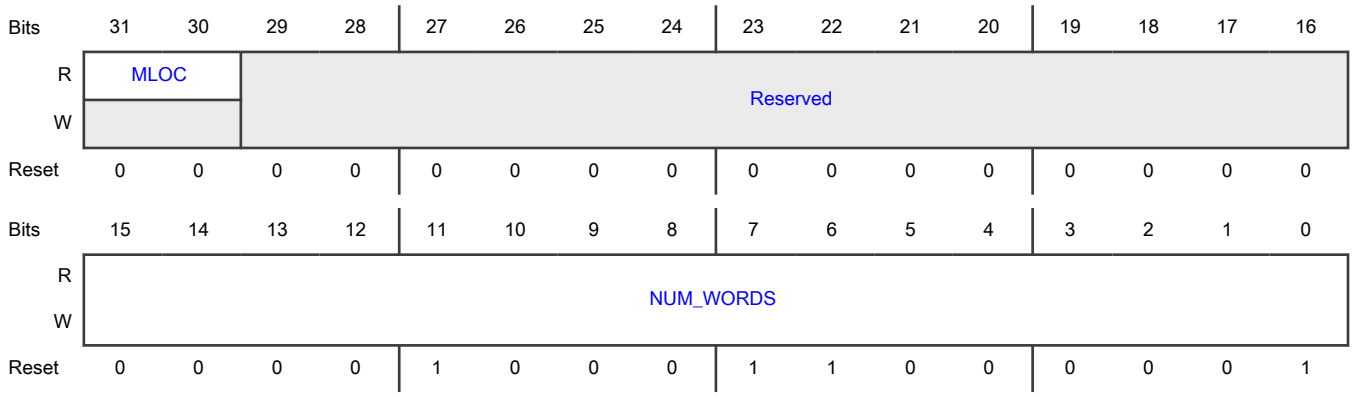
This is the switch hash table memory allotment register.

Maximum number of words allotted to the switch exact match hash tables from the common memory's shared region.

The hash tables are:

- Ingress Stream Identification table
- Ingress Stream Filter table
- VLAN Filter table
- FDB table
- L2 IPv4 Multicast Filter table

Diagram



Fields

Field	Function
31-30 MLOC	Indicates memory location of this table 0: Common memory 1-3: Reserved
29-16 —	Reserved
15-0 NUM_WORDS	Maximum number of words allotted to the switch exact match hash tables from the common memory's shared region.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Software driver is responsible to manage how the hash table memory is distributed amongst various hash tables. **NOTE** Reset value of switch HTMCAPR register. This value is writable in IERB (by pre-boot initialization), which is then copied to the switch function register during IERB lock.

53.4.6.3.62 Switch 0 index table memory allocation register (S0ITMAR)

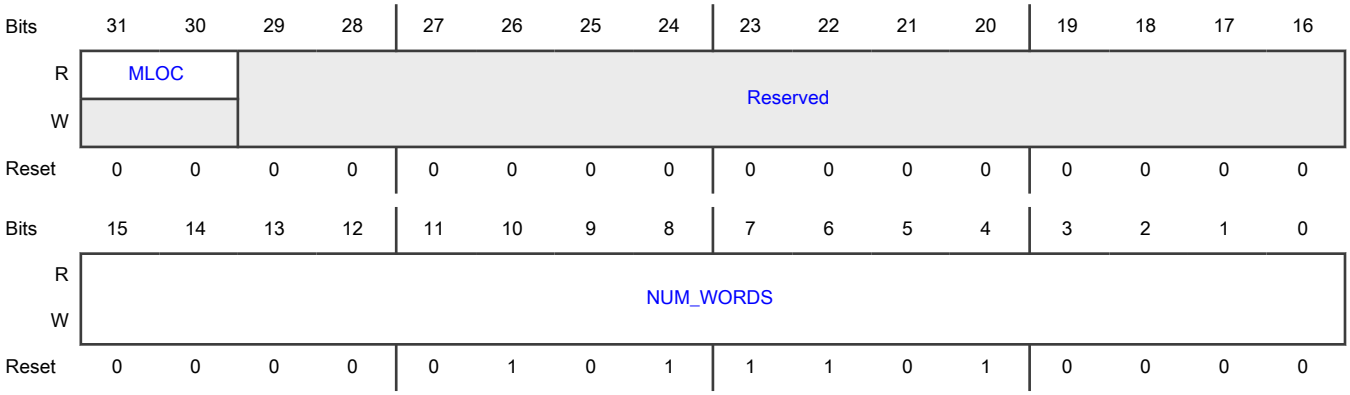
Offset

Register	Offset
S0ITMAR	2084h

Function

This is the switch index table memory allocation register.

Diagram



Fields

Field	Function
31-30 MLOC	Indicates memory location of this table 0: Common memory 1-3: Reserved
29-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 NUM_WORDS	Number of words allocated to the switch's index table memory. **NOTE** Reset value of switch instance ITMCAPR register. Distribution of the table memory to various tables is done via various Index Table Memory Allocation Registers (xITMAR).

53.4.6.3.63 Switch 0 ingress port filter table memory allocation register (S0IPFTMAR)

Offset

Register	Offset
S0IPFTMAR	2088h

Function

This is the switch ingress port filter table memory allocation register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ALLOC															
Reset	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 ALLOC	Number of words allocated to Ingress Port Filter table from ingress port filter ternary memory. **NOTE** Reset value of switch register IPFTCAPR.

53.4.6.3.64 Switch 0 rate policer index table memory allocation register (S0RPITMAR)

Offset

Register	Offset
S0RPITMAR	20A0h

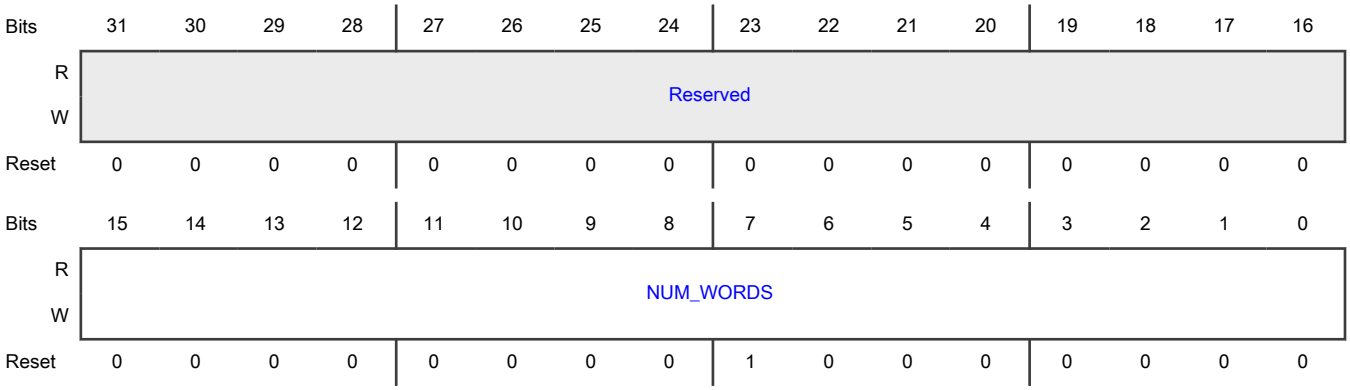
Function

This is the switch rate policer index table memory allocation register.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to switch register RPITMAR during IERB lock. All updates after this must be done through the corresponding switch register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. Number of entries assigned to this table is ROUNDDOWN(NUM_WORDS / 4). This value is writable in IERB (by pre-boot initialization), which is R/W in the switch version of this register. **NOTE** Each entry occupies 4 words.

53.4.6.3.65 Switch 0 ingress stream counter index table memory allocation register (S0ISCITMAR)

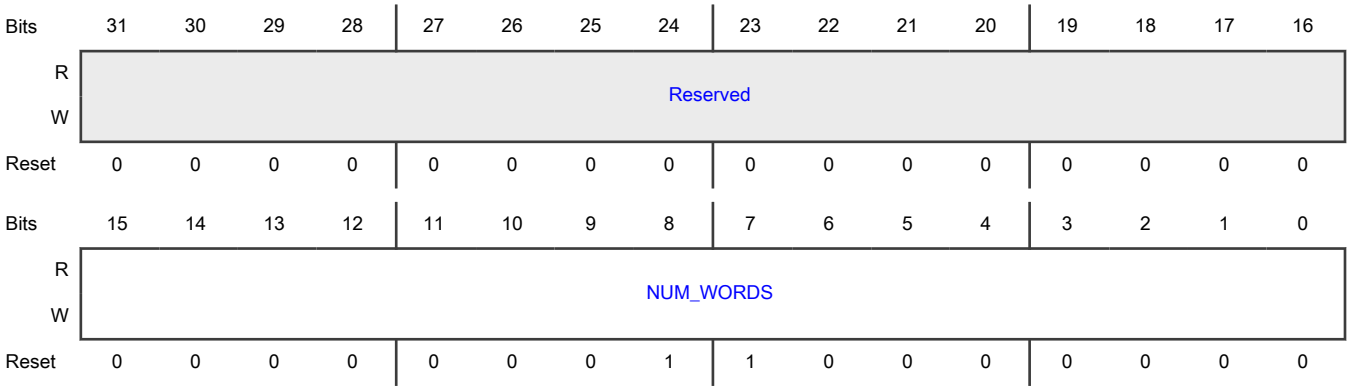
Offset

Register	Offset
S0ISCITMAR	20A4h

Function

This is the switch ingress stream counter index table memory allocation register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** Each entry occupies 1 word.

53.4.6.3.66 Switch 0 ingress stream index table memory allocation register (S0ISITMAR)

Offset

Register	Offset
S0ISITMAR	20A8h

Function

This is the switch ingress stream index table memory allocation register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NUM_WORDS															
Reset	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. This value is writable in IERB (by pre-boot initialization), which is R/W in the switch version of this register. **NOTE** Each entry occupies 1 word.

53.4.6.3.67 Switch 0 ingress sequence generation index table memory allocation register (S0ISQGITMAR)

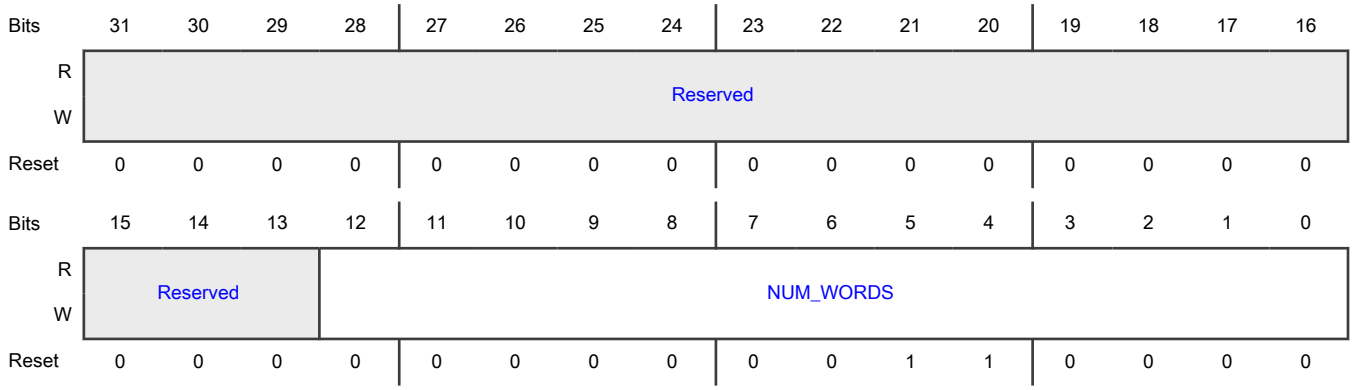
Offset

Register	Offset
S0ISQGITMAR	20ACh

Function

This is the switch ingress sequence generation index table memory allocation register.

Diagram



Fields

Field	Function
31-13 —	Reserved
12-0 NUM_WORDS	The number of words from index table memory assigned to this table. Number of entries assigned to the table is 8 * NUM_WORDS. This value is writable in IERB (by pre-boot initialization), which is R/W in the switch version of this register. **NOTE** There are 8 entries per word.

53.4.6.3.68 Switch 0 stream gate instance index table memory allocation register (S0SGIITMAR)

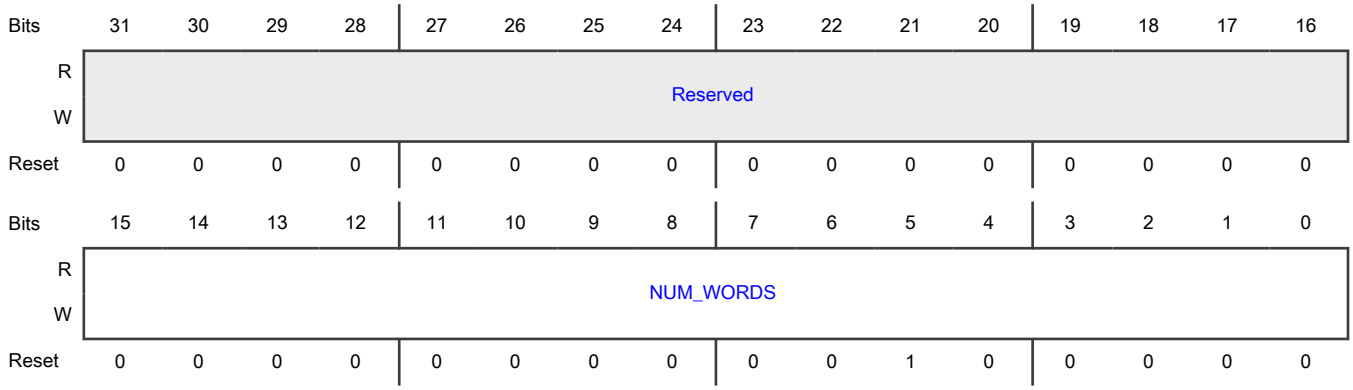
Offset

Register	Offset
S0SGIITMAR	20B4h

Function

This is the switch stream gate instance index table memory allocation register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. This value is writable in IERB (by pre-boot initialization), which is R/W in the switch version of this register. **NOTE** Each entry occupies 1 word.

53.4.6.3.69 Switch 0 stream gate control list index table memory allocation register (S0SGCLITMAR)

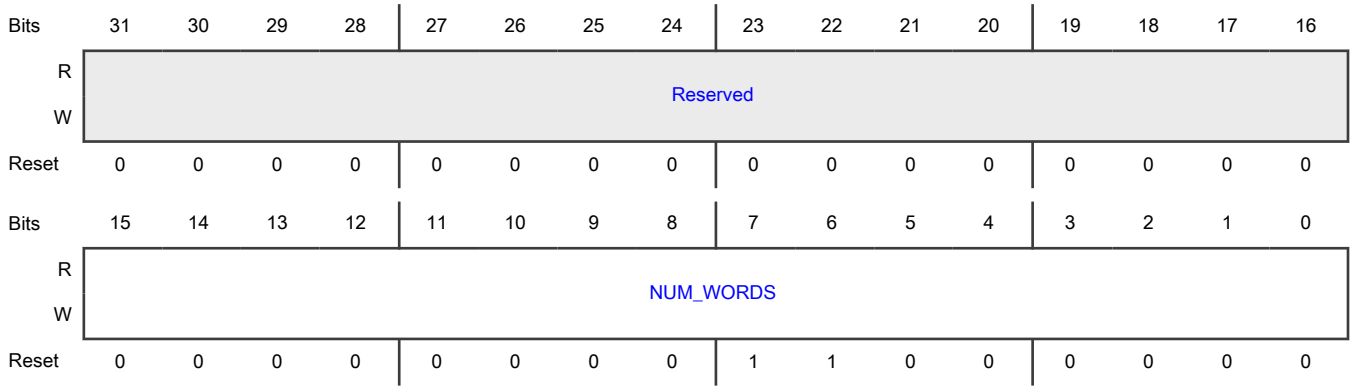
Offset

Register	Offset
S0SGCLITMAR	20B8h

Function

This is the switch stream gate control list index table memory allocation register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** Each entry utilizes a variable number of words depending on the number of gates specified. Entry size in words is calculated as: 1 + ROUNDUP(NUM_GATES * 0.5).

53.4.6.3.70 Switch 0 frame modification index table memory allocation register (S0FMITMAR)

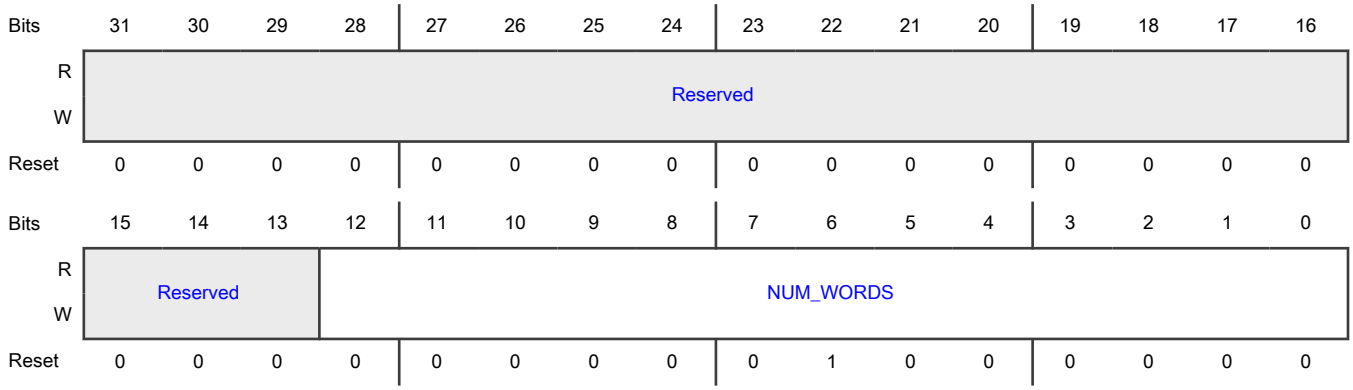
Offset

Register	Offset
S0FMITMAR	20BCh

Function

This is the switch frame modification index table memory allocation register.

Diagram



Fields

Field	Function
31-13 —	Reserved
12-0 NUM_WORDS	The number of words from index table memory assigned to this table. This value is writable in IERB (by pre-boot initialization), which is R/W in the switch version of this register. **NOTE** Each entry occupies 1 word.

53.4.6.3.71 Switch 0 frame modification data index table memory allocation register (S0FMDITMAR)

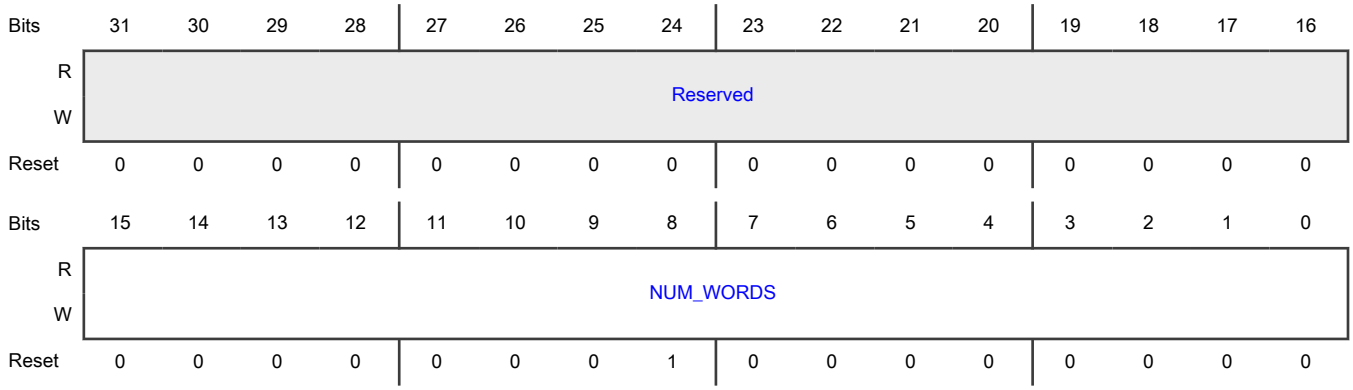
Offset

Register	Offset
S0FMDITMAR	20C0h

Function

This is the switch frame modification data index table memory allocation register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. This value is writable in IERB (by pre-boot initialization), which is R/W in the switch version of this register. **NOTE** Each entry utilizes of a variable number of words depending on the data size requested by frame modification entry.

53.4.6.3.72 Switch 0 time gate scheduling table allocation register (S0TGSTAR)

Offset

Register	Offset
S0TGSTAR	20F0h

Function

This is the switch egress gate list table allocation register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				NUM_WORDS											
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 NUM_WORDS	This field specifies the number of words in the Time Gate Scheduling internal memory (TGSMCAPR) allocated to support the switch Time Gate Scheduling table, which in turn contains the administrative gate control list and the operational gate control list of each switch port. The Time Gate Scheduling internal memory is shared between the switch and all the ENETC instances supporting time gate scheduling. The sum of all of the individual allocation (S0TGSTAR and E*a*TGSTARs) must not exceed the total size of the Time Gate Scheduling internal memory. The memory within this allocation is managed using a single free list of fixed equal size units (or words) of memory. Free memory units from the free list are available to be dynamically allocated to any gate control lists subject to the total memory available. Each gate control list is comprised of a sequence of gate operation entries where each entry occupies one word of memory. Note that a gate control list can contain a variable number of gate control operation entries. Hardware doesn't limit the number of gate operation entries (or memory) used by a specific switch port. **NOTE** Reset value of switch function's register TGSTCAPR.

53.4.6.3.73 Switch 0 time gate scheduling lookahead register (S0TGSLR)

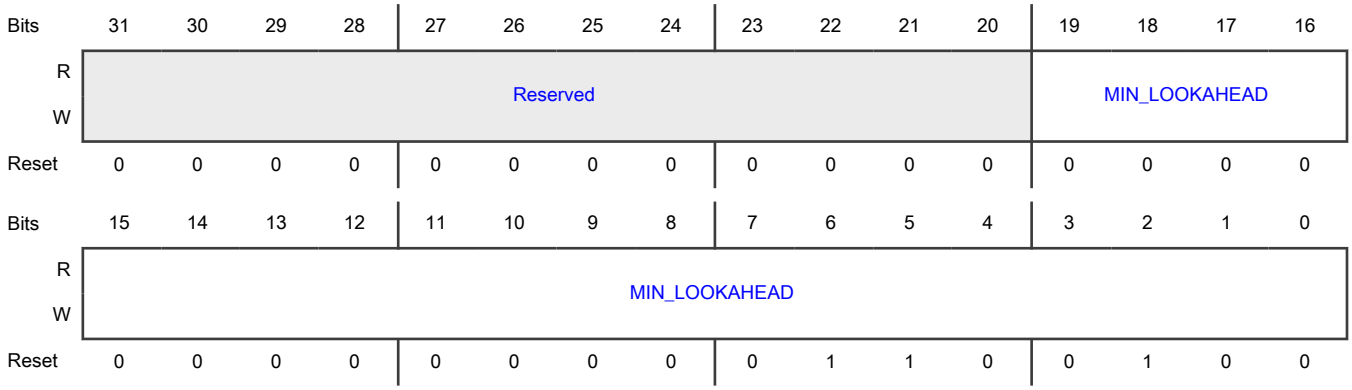
Offset

Register	Offset
S0TGSLR	20F4h

Function

This is the time gate scheduling lookahead register.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-0 MIN_LOOKAHEAD	Minimum lookahead This field specifies the amount of time to advance the IEEE 1588 time scale used by the time gate scheduler (at the frame scheduling timing point), to account for the time required to schedule and dequeue a frame. The time is specified in units of nanoseconds. The IEEE 1588 time scale used by the time gate scheduler, can also be advanced, on per port basis, by the time amount specified in PTGSATOR[ADV_TIME_OFFSET], to adjust for delay across the MAC plus if needed, delays outside of NETC (e.g. PHY delay). Both advanced times are cumulative, the IEEE 1588 time scale used by the time gate scheduler (at the frame scheduling timing point) on a given port, will be advanced by the amount of time resulting from adding the S0TGSLR[MIN_LOOKAHEAD] time to the PTGSATOR[ADV_TIME_OFFSET] time. Setting a too low value can cause a frame to miss its scheduled time-gated window or can cause a frame to overrun its scheduled time-gated window (not detected). Setting a too high value can lead to reduce link efficiency as opportunities to transmit a frame during a scheduled time-gated window can be lost. **NOTE** Value may need to be increased from the reset on value if NETC is running with a low clock frequency.

53.4.6.3.74 Switch 0 management port configuration register (S0MPCR)

Offset

Register	Offset
S0MPCR	2204h

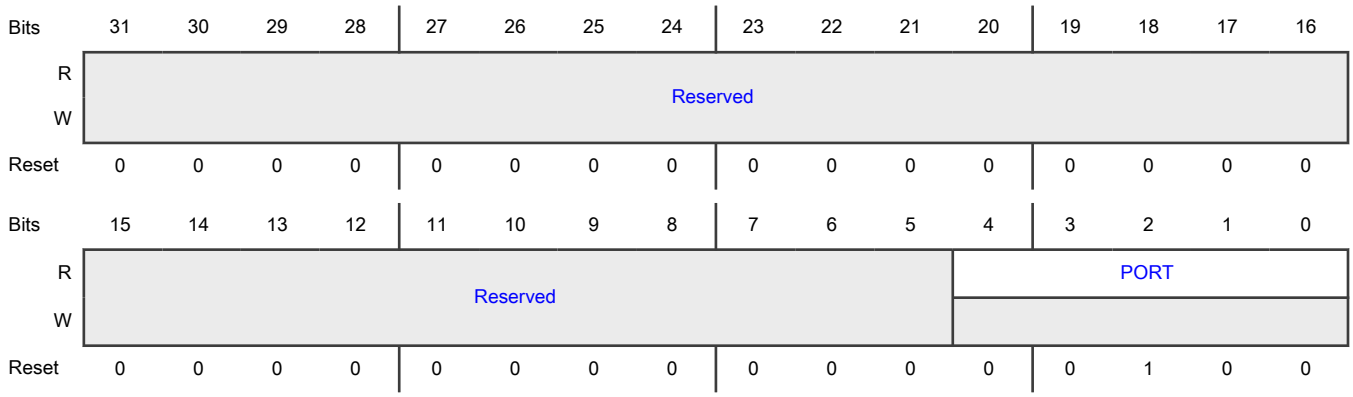
Function

This is the switch management port configuration register

NOTE

The value of this register will be immediately reflected in the corresponding read-only switch register of the same name (MPCR).

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 PORT	Specifies the destination port for frames identified as management. The value of this field will be immediately reflected in the corresponding read-only switch register MPCR.

53.4.6.3.75 Switch 0 VLAN Filter (hash) table default entry configuration registers 0 (S0VFHTDECR0)

Offset

Register	Offset
S0VFHTDECR0	2210h

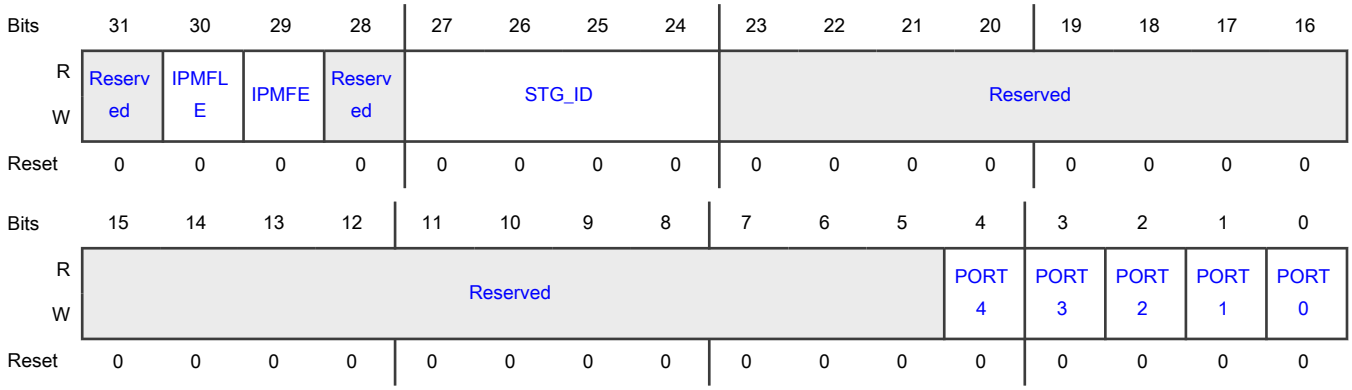
Function

This is the switch VLAN filter hash table default entry configuration registers 0.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to switch register VFHTDECR0 during IERB lock. All updates after this must be done through the corresponding switch register.

Diagram



Fields

Field	Function
31 —	Reserved
30 IPMFLE	IP Multicast Flooding Enable If IP multicast filtering is performed (IPMFE = 1b, and the frame is identified as a multicast IP packet), and there was no match found, then the frame is forwarded according to this field. If IP multicast filtering is disabled (IPMFE = 0b), this field is ignored by hardware. 0b - IP Multicast Flooding disabled, the frame is discarded. 1b - IP Multicast Flooding enabled, the frame is flooded.
29 IPMFE	IP Multicast Filtering Enable Refer to the VLAN Filter table entry IPMFE field description, for more details 0b - No IP multicast filtering is performed. 1b - If the frame is identified as a multicast IP packet, then IP multicast filtering is performed. If the frame is not identified as an IP multicast packet, the IP multicast filtering is not performed.
28 —	Reserved
27-24 STG_ID	Spanning Tree Group Member ID Refer to the VLAN Filter table entry STG_ID field description, for more details
23-5 —	Reserved
4-0 PORTn	Port n. 0b - Port n is not a member of this VLAN. 1b - Port n is a member of this VLAN.

53.4.6.3.76 Switch 0 VLAN filter hash table default entry configuration registers 1 (S0VFHTDECR1)

Offset

Register	Offset
S0VFHTDECR1	2214h

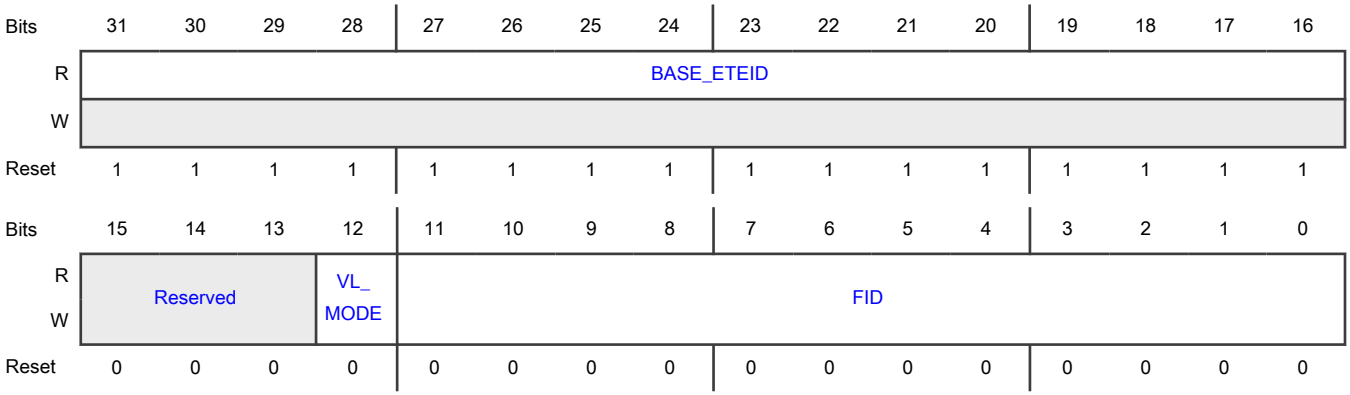
Function

This is the switch VLAN Filter (hash) table default entry configuration registers 1.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to switch register VFHTDECR1 during IERB lock. All updates after this must be done through the corresponding switch register.

Diagram



Fields

Field	Function
31-16 BASE_ETEID	Base Egress Treatment Entry ID Refer to the VLAN Filter table entry BASE_ET_EID field description, for more details.
15-13 —	Reserved
12 VL_MODE	VLAN Learning Mode 0: Independent VLAN learning: FID is set to the VID assigned to the frame 1: Shared VLAN learning: Use the FID specified in this register **NOTE** Used to determine the FID when doing a lookup in the FDB table.
11-0	Filtering ID. Valid if VL_MODE = 1 (Shared).

Table continues on the next page...

Table continued from the previous page...

Field	Function
FID	The Filtering ID (FID) is a locally significant identifier (global to the switch), that is used as a key value when hardware performs a lookup into the FDB table and the L2 IPV4 Multicast Filter table. The FID is used to identify a set of VIDs, which in turn allows sharing of the same address (MAC, IP) between multiple VIDs during lookups into the FDB table and L2 IPV4 Multicast Filter table.

53.4.6.3.77 Switch 0 VLAN filter hash table default entry configuration registers 2 (S0VFHTDECR2)

Offset

Register	Offset
S0VFHTDECR2	2218h

Function

This is the switch VLAN Filter (hash) table default entry configuration registers 2.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to switch register VFHTDECR2 during IERB lock. All updates after this must be done through the corresponding switch register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			MFO		MLO			Reserved							
Reset	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											ES_PORT4	ES_PORT3	ES_PORT2	ES_PORT1	ES_PORT0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-27 MFO	MAC forwarding options: 0: Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1: No FDB lookup is performed, the frame is flooded 2: FDB lookup is performed, and if there is no match, the frame is flooded. 3: FDB lookup is performed, and if there is no match, the frame is discarded.
26-24 MLO	MAC learning options: 0: Reserved 1: Disable MAC learning. MAC learning function is not performed during the forwarding processing. 2: Hardware MAC learning is performed. 3: Software MAC learning secure. A MAC learning lookup is performed into the FDB table. If there is no match, no attempt is made to add a new entry, and the frame is redirect to the switch management port. If there is match, and the entry's port number does not match frame ingress port number, the frame is redirected to the switch management port if station move is allowed, otherwise the frame is discarded. 4: Software MAC learning unsecure. A MAC learning lookup is performed into the FDB table. If there is no match, no attempt is made to add a new entry, and a copy of the frame is sent to the switch management port. If there is match, and the entry's port number does not match frame ingress port number, the frame is copied to the switch management port if station move is allowed, otherwise the frame is discarded. 5: Disable MAC learning with SMAC validation. A MAC learning lookup is performed into the FDB table. If there is no match or there is a match but the ingress port is not a member of the FDB entry, the frame is discarded and counted against the bridge port discard count register (BPDCR) with discard reason BPDCRR0[MACLNFDR] set to 1.
23-5 —	Reserved
4-0 ES_PORTn	Egress Treatment Applicability Port. Valid if BASE_ETEID is not null. This field is used to set the Egress Treatment applicability port bitmap. If this field (port n) is set to 1, port n will have its bit set to 1 in the Egress Treatment applicability port bitmap.

53.4.6.3.78 ENETC 0 binding configuration register 0 (E0BCR0)

Offset

Register	Offset
E0BCR0	3000h

Function

This is the ENETC binding configuration register 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VALID	Reserved														
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				FN				Reserved				RC_INST			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VALID	If set, this ENETC instance is associated to a link end.
30-12 —	Reserved
11-8 FN	PCI device function number. **NOTE** For assignment of function number, see PCIe Function Number Assignment.
7-4 —	Reserved
3-0 RC_INST	Root complex instance this function is bound to.

53.4.6.3.79 ENETC 0 binding configuration register 1 (E0BCR1)

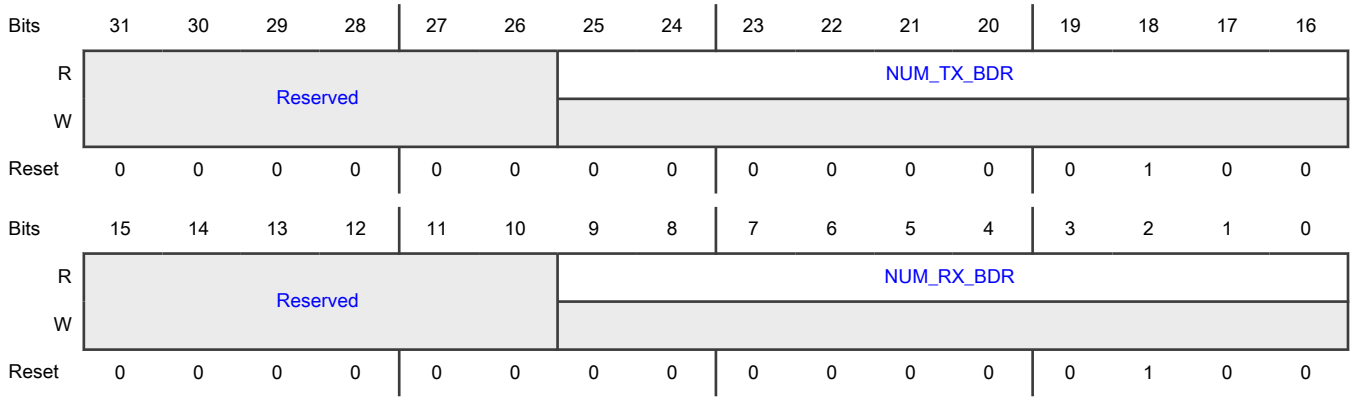
Offset

Register	Offset
E0BCR1	3004h

Function

This is the ENETC binding configuration register 1.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 NUM_TX_BDR	Number of Tx BD rings supported by ENETC. Range: 0..1023 Reset value of ENETC register field ECAPR0[NUM_TX_BDR]. **NOTE** If total number of allocated Rx BD rings exceed CAPR1[NUM_RX_BDR], error is indicated by NETCSR[ERROR].
15-10 —	Reserved
9-0 NUM_RX_BDR	Number of Rx BD rings supported by ENETC. Range: 0..1023 Reset value of ENETC register field ECAPR0[NUM_RX_BDR]. **NOTE** If total number of allocated Rx BD rings exceed CAPR1[NUM_RX_BDR], error is indicated by NETCSR[ERROR].

53.4.6.3.80 ENETC a binding configuration register 2 (E0BCR2 - E1BCR2)

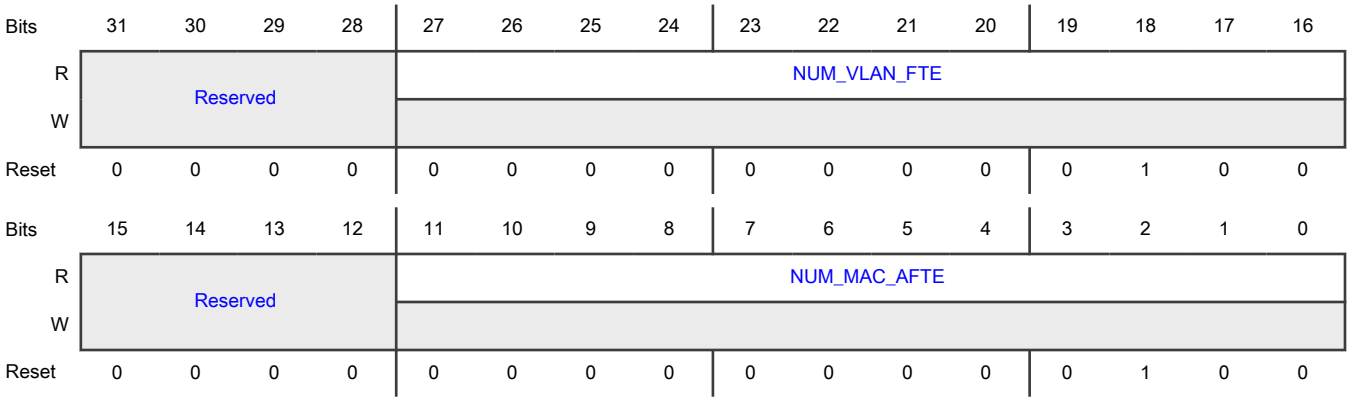
Offset

Register	Offset
E0BCR2	3008h
E1BCR2	3108h

Function

This is the ENETC binding configuration register 2.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 NUM_VLAN_FTE	Number of ENETC SI VLAN filter rules supported. **NOTE** Reset value of ENETC register field PSIVLANFCAPR[NUM_VLAN_FTE].
15-12 —	Reserved
11-0 NUM_MAC_AFTE	Number of ENETC SI MAC address filter rules supported. **NOTE** Reset value of ENETC register field PSIMACFCAPR[NUM_MAC_AFTE].

53.4.6.3.81 ENETC 0 VSI binding configuration register (E0VBCR)

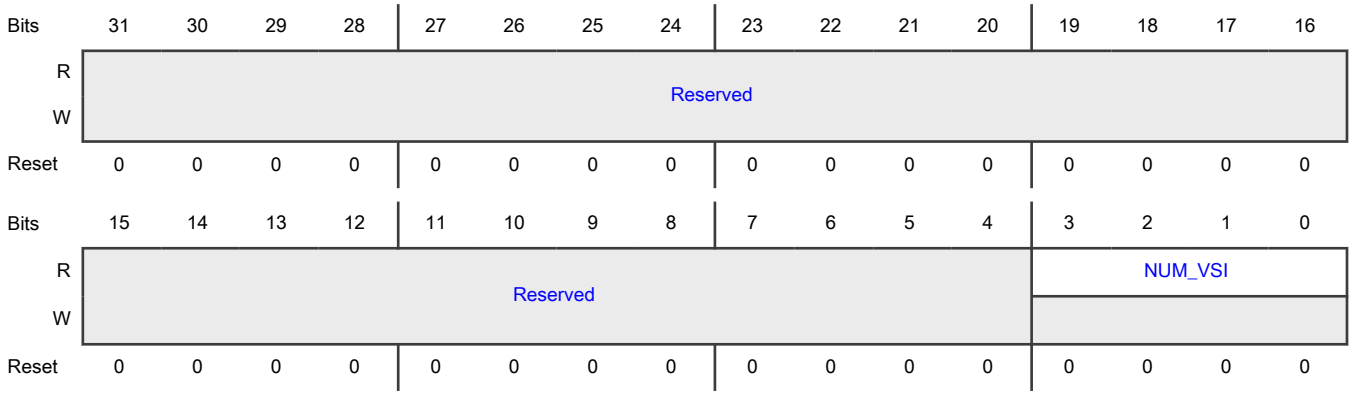
Offset

Register	Offset
E0VBCR	3010h

Function

This is the ENETC VSI binding configuration register.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 NUM_VSI	Indicates the number of VSIs supported for this ENETC instance. **NOTE** VSI PCIe function numbers will be sequentially assigned. The VSI 0 function number can be determined from the PCIe SR-IOV capability registers. If allocated VSI instances exceed CAPR0[NUM_VSI], error is indicated by NETCSR[ERROR].

53.4.6.3.82 ENETC 0 MSI-X configuration register (E0MCR)

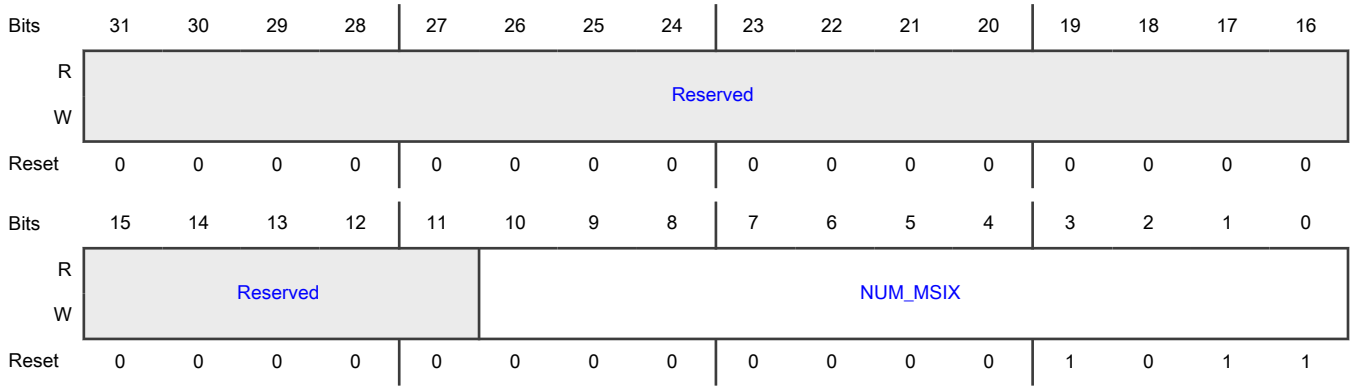
Offset

Register	Offset
E0MCR	3014h

Function

This is the ENETC MSI-X configuration register.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-0 NUM_MSIX	Number of MSI-X vectors supported for ENETC function. Formula: NUM_MSIX+1 Range: 1..1024 **NOTE** Reset value of ENETC register ECAPR1[NUM_MSIX]. If total number of allocated MSI-X vectors exceed CAPR2[NUM_MSIX], error is indicated by NETCSR[ERROR].

53.4.6.3.83 ENETC 0 config header device ID and vendor ID register (E0_CFH_DIDVID)

Offset

Register	Offset
E0_CFH_DIDVID	3020h

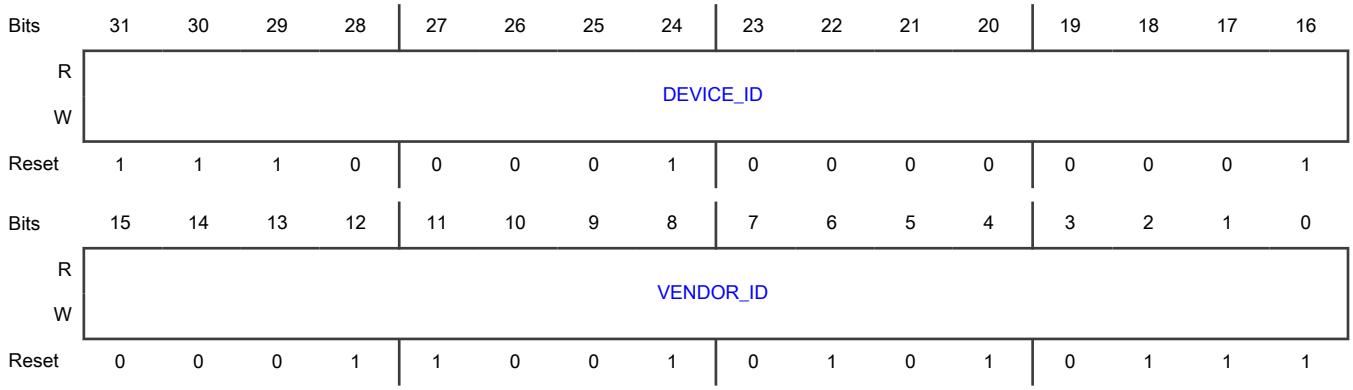
Function

This is the ENETC PCI Device and Vendor ID register.

NOTE

This register is continuously sampled, any changes will have immediate effect on reading the PCI function Type0 header.

Diagram



Fields

Field	Function
31-16 DEVICE_ID	Device ID This field identifies the device ID of the device shown in the PCIe Device ID Register (02h).
15-0 VENDOR_ID	Vendor ID This field identifies the manufacturer of the device as shown in the PCIe Vendor ID Register (00h). The default value will indicate Freescale (0x1957).

53.4.6.3.84 ENETC 0 config header subsystem ID and subsystem vendor ID register (E0_CFH_SIDSVID)

Offset

Register	Offset
E0_CFH_SIDSVID	3024h

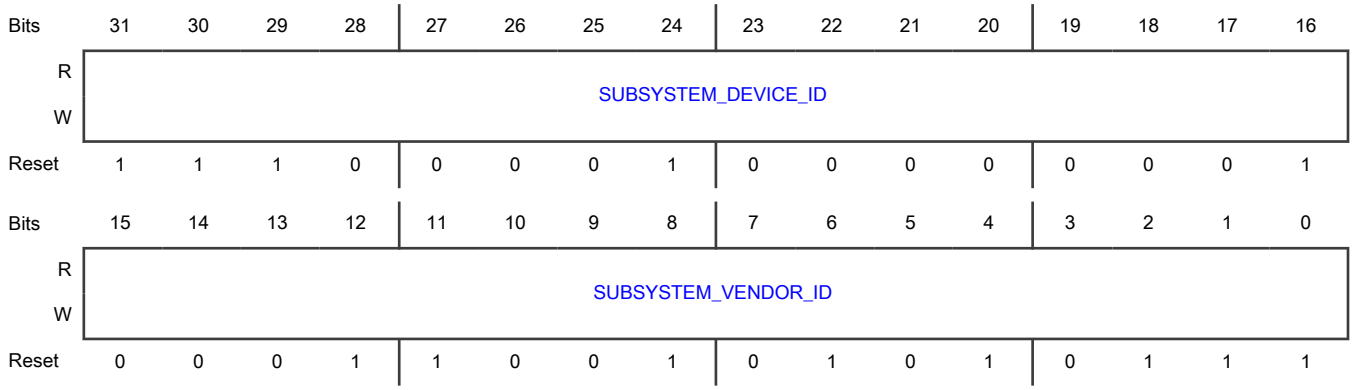
Function

This is the ENETC PCI Subsystem ID and Subsystem Vendor ID register.

NOTE

This register is continuously sampled, any changes will have immediate effect on reading the PCI function Type0 header.

Diagram



Fields

Field	Function
31-16 SUBSYSTEM_DEVICE_ID	Subsystem Device ID This field identifies the particular subsystem as shown in the PCIe Subsystem ID Register (2Eh).
15-0 SUBSYSTEM_VENDOR_ID	Subsystem Vendor ID This field identifies the manufacturer of the subsystem as shown in the PCIe Subsystem Vendor ID Register (2Ch). The default value is the same as E*a*_CFH_DIDVID[VENDOR_ID].

53.4.6.3.85 ENETC a config capability VF device ID register (E0_CFC_VFDID - E1_CFC_VFDID)

Offset

Register	Offset
E0_CFC_VFDID	3028h
E1_CFC_VFDID	3128h

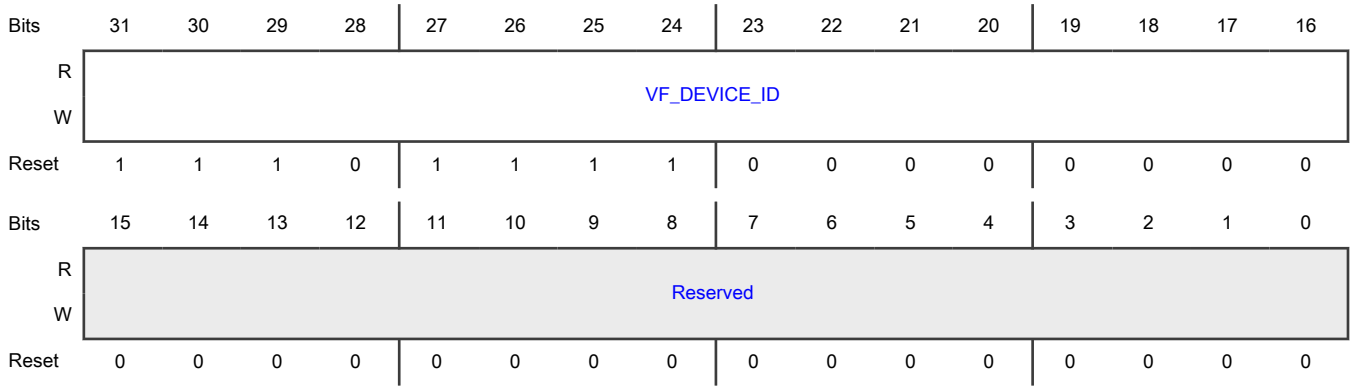
Function

This is the PCIe SR-IOV Capability Structure VF Device ID register.

NOTE

This register is continuously sampled, any changes will have immediate effect on reading the PCI function Type0 header.

Diagram



Fields

Field	Function
31-16 VF_DEVICE_ID	VF Device ID This field identifies the device ID for a virtual function as shown in the PCIe SR-IOV Capability Structure (20h).
15-0 —	Reserved

53.4.6.3.86 ENETC a buffer cache attribute register 0 (E0BCAR - E1BCAR)

Offset

Register	Offset
E0BCAR	3030h
E1BCAR	3130h

Function

This is the ENETC buffer cache attribute register. It is used to determine system interface attribute settings for reads and writes of Tx/Rx buffer descriptors and data.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to ENETC register SIBCAR during IERB lock. All updates after this must be done through the corresponding ENETC register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	RDSNP	RDDOMAIN		RDCACHE				Reserved	BD_RDSNP	BD_RDDOMAIN		BD_RDCACHE			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	WRSNP	WRDOMAIN		WRCACHE				Reserved	BD_WRSNP	BD_WRDOMAIN		BD_WRCACHE			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31 —	Reserved
30 RDSNP	Read data snoop This is the snoop attribute setting used when ENETC reads frame data from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
29-28 RDDOMAIN	Read data domain This is the domain attribute setting used when ENETC reads frame data from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
27-24 RDCACHE	Read data cache type This is the cache attribute setting used when ENETC reads frame data from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
23 —	Reserved
22 BD_RDSNP	Buffer descriptor read snoop See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
21-20 BD_RDDOMAIN	Buffer descriptor read domain This is the domain attribute setting used when ENETC reads the buffer descriptor from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-16 BD_RDCACHE	Buffer descriptor read cache type This is the cache attribute setting used when ENETC reads the buffer descriptor from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
15 —	Reserved
14 WRSNP	Write data snoop This is the snoop attribute setting used when ENETC writes frame data to memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
13-12 WRDOMAIN	Write data domain This is the domain attribute setting used when ENETC writes frame data to memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
11-8 WRCACHE	Write data cache type This is the cache attribute setting used when ENETC writes frame data to memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
7 —	Reserved
6 BD_WRSNP	Buffer descriptor write snoop This is the snoop attribute setting used when ENETC writes the buffer descriptor in memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
5-4 BD_WRDOMAIN	Buffer descriptor write domain This is the domain attribute setting used when ENETC writes the buffer descriptor in memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
3-0 BD_WRCACHE	Buffer descriptor write cache type This is the cache attribute setting used when ENETC writes the buffer descriptor in memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.

53.4.6.3.87 ENETC a message cache attribute register (E0MCAR - E1MCAR)

Offset

Register	Offset
E0MCAR	3034h
E1MCAR	3134h

Function

This is the ENETC message cache attribute register. It is used to determine system interface attribute settings for PSI-VSI messaging reads and writes.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to ENETC register SIMCAR during IERB lock. All updates after this must be done through the corresponding ENETC register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved									MSG_RDS...	MSG_RDDOMAIN		MSG_RDCACHE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved									MSG_WRS...	MSG_WRDOMAIN		MSG_WRCACHE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-23 —	Reserved
22 MSG_RDSNP	SI messaging read data snoop This is the snoop attribute setting used when ENETC reads PSI-VSI message from memory during partition copy. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
21-20 MSG_RDDOMAIN	SI messaging read data domain This is the domain attribute setting used when ENETC reads PSI-VSI message from memory during partition copy.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
19-16 MSG_RDCACHE	SI messaging read data cache type This is the cache attribute setting used when ENETC reads PSI-VSI message from memory during partition copy. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
15-7 —	Reserved
6 MSG_WRSNP	SI messaging write snoop This is the snoop attribute setting used when ENETC writes PSI-VSI message during partition copy. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
5-4 MSG_WRDOMAIN	SI messaging write domain This is the domain attribute setting used when ENETC writes PSI-VSI message during partition copy. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
3-0 MSG_WRCACHE	SI messaging write cache type This is the cache attribute setting used when ENETC writes PSI-VSI message during partition copy. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.

53.4.6.3.88 ENETC a command cache attribute register (E0CAR - E1CAR)

Offset

Register	Offset
E0CAR	3038h
E1CAR	3138h

Function

This is the ENETC command cache attribute register. It is used to determine system interface attribute settings for reads and writes of command buffer descriptor and data.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to ENETC register SICCAR during IERB lock. All updates after this must be done through the corresponding ENETC register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	CRDSNP	CRDDOMAIN		CRDCACHE				Reserved	CBD_RDS...	CBD_RDDOMAIN		CBD_RDCACHE			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	CWRSNP	CWRDOMAIN		CWRCACHE				Reserved	CBD_WRS...	CBD_WRDOMAIN		CBD_WRCACHE			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31 —	Reserved
30 CRDSNP	Read data snoop This is the snoop attribute setting used when ENETC reads command data from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
29-28 CRDDOMAIN	Read data domain This is the domain attribute setting used when ENETC reads command data from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
27-24 CRDCACHE	Read data cache type This is the cache attribute setting used when ENETC reads command data from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
23 —	Reserved
22 CBD_RDSNP	Command descriptor read snoop See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
21-20 CBD_RDDOMAIN	Command buffer descriptor read domain This is the domain attribute setting used when ENETC reads the command buffer descriptor from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-16 CBD_RDCACHE	Command buffer descriptor read cache type This is the cache attribute setting used when ENETC reads the command buffer descriptor from memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
15 —	Reserved
14 CWRSNP	Write data snoop This is the snoop attribute setting used when ENETC writes command data to memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
13-12 CWRDOMAIN	Write data domain This is the domain attribute setting used when ENETC writes command data to memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
11-8 CWRCACHE	Write data cache type This is the cache attribute setting used when ENETC writes command data to memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
7 —	Reserved
6 CBD_WRSNP	Command buffer descriptor write snoop This is the snoop attribute setting used when ENETC writes the command buffer descriptor in memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
5-4 CBD_WRDOMAIN	Command buffer descriptor write domain This is the domain attribute setting used when ENETC writes the command buffer descriptor in memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
3-0 CBD_WRCACHE	Command buffer descriptor write cache type This is the cache attribute setting used when ENETC writes the command buffer descriptor in memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.

53.4.6.3.89 ENETC a access management qualifier register (E0AMQR - E1AMQR)

Offset

Register	Offset
E0AMQR	3040h
E1AMQR	3140h

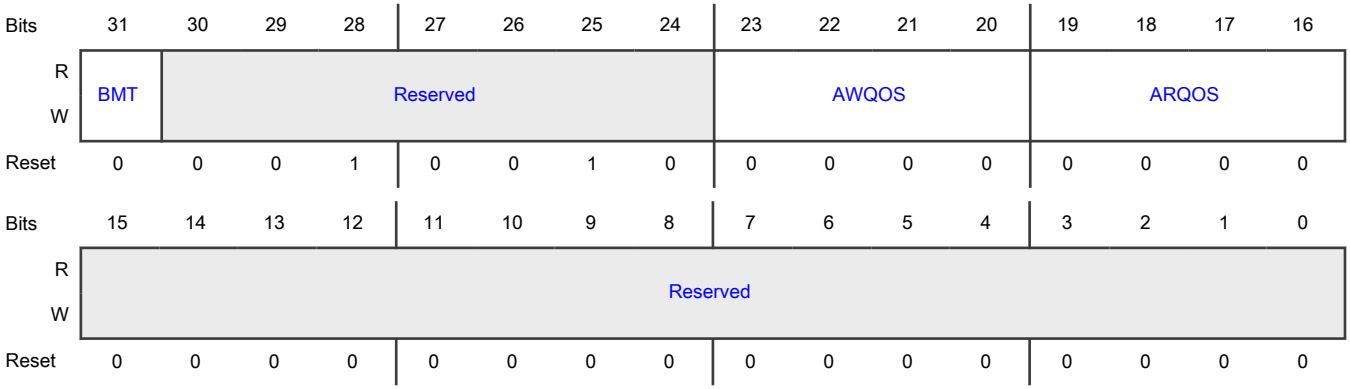
Function

This is the ENETC access management qualifier register. It provides attributes for each memory access transaction performed by the ENETC instance.

NOTE

Changing the value in this register has immediate effect unless otherwise noted. Care should be taken when making changes to the values for an enabled and operational ENETC function to ensure that consistent behavior is maintained. It is strongly encouraged, but not required, that the function be disabled before making changes.

Diagram



Fields

Field	Function
31 BMT	Bypass memory translation The bit is an indication to the SMMU to by-pass memory translation whenever the PF is performing memory transactions, effectively handling the memory address as a true physical address. 0 Memory translation 1 Bypass memory translation
30-24 —	Reserved
23-20 AWQOS	Address Write QoS. AWQOS[3:1] signals are controlled by this register field's bits [3:1]. AWQOS[0] is controlled by the HTA block and is 0 for low priority DMA transactions and 1 for high priority DMA transactions.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	HTA Tx threads are always high priority. HTA Command threads are always low priority. HTA Rx threads are either high or low priority. For each frame dequeued from ICM, one or more HTA Rx threads are dispatched, with their priority set to the priority of the ICM queue (2 tiers) from which the frame was dequeued from.
19-16 ARQOS	Address Read QoS. ARQOS[3:1] signals are controlled by this register field's bits [3:1]. ARQOS[0] is controlled by the HTA block and is 0 for low priority DMA transactions and 1 for high priority DMA transactions. HTA Tx threads are always high priority. HTA Command threads are always low priority. HTA Rx threads are either high or low priority. For each frame dequeued from ICM, one or more HTA Rx threads are dispatched, with their priority set to the priority of the ICM queue (2 tiers) from which the frame was dequeued from.
15-0 —	Reserved

53.4.6.3.90 ENETC a boot loader parameter register b (E0BLPR0 - E1BLPR1)

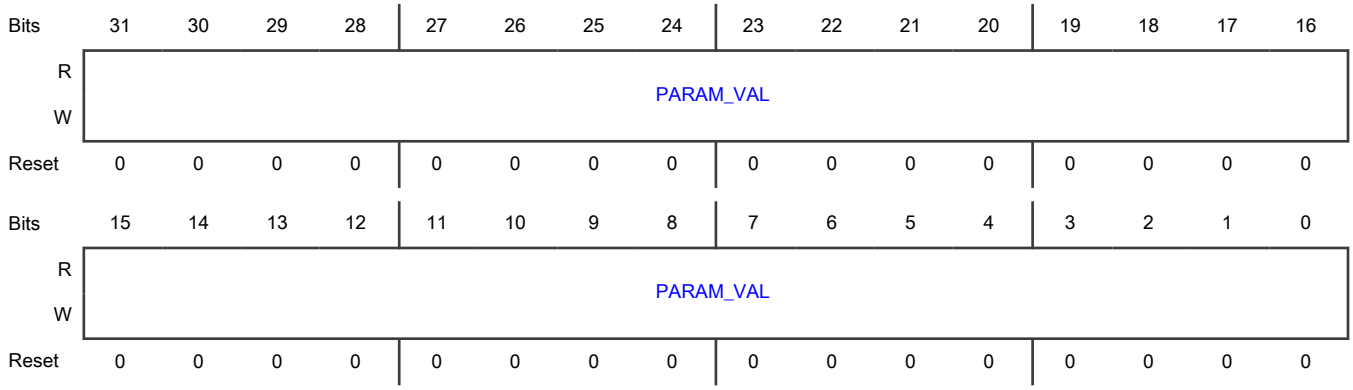
Offset

Register	Offset
E0BLPR0	3048h
E0BLPR1	304Ch
E1BLPR0	3148h
E1BLPR1	314Ch

Function

This register provides a mean for boot S/W (Pre-Boot Loader, boot ROM) to communicate parameters with running (device driver) software. Runtime software uses read-only global register FBLPR*a*.

Diagram



Fields

Field	Function
31-0 PARAM_VAL	Boot loader parameter value.

53.4.6.3.91 ENETC a receive memory buffer entitlement register (E0RXMBER - E1RXMBER)

Offset

Register	Offset
E0RXMBER	3050h
E1RXMBER	3150h

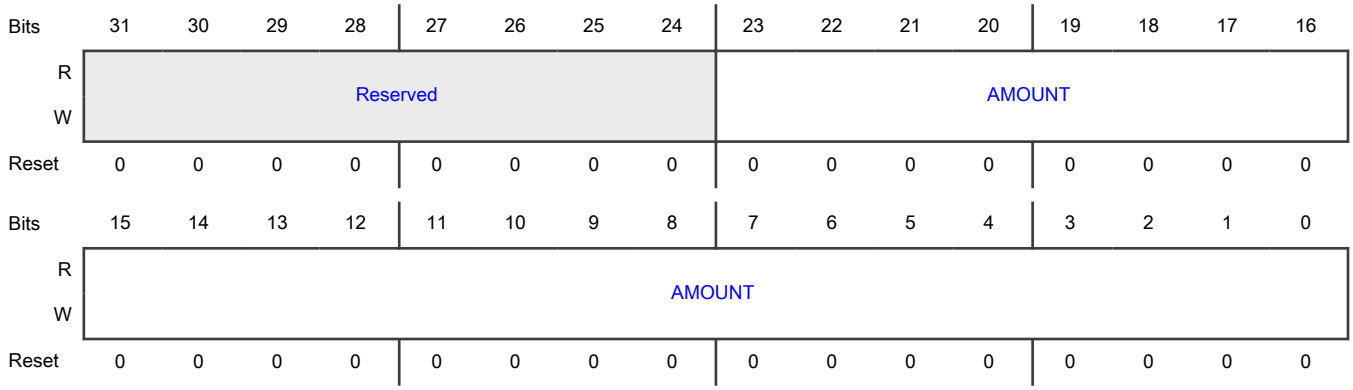
Function

This is the ENETC receive memory buffer entitlement register.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to ENETC register RXMBER during IERB lock. All updates after this must be done through the corresponding ENETC register.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 AMOUNT	Receive memory buffer entitlement in words

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This receive Memory Buffer Entitlement is used in determining smart drop for ingress congestion control. If a smart drop is to be performed, due receive memory buffer depletion, the receive queue that exceeds entitlement amount the most will be selected for discard. Default value of 0 for this register results in fair share of the receive buffer memory across all ENETC instances.

53.4.6.3.92 ENETC a receive memory buffer limit register (E0RXMBLR - E1RXMBLR)

Offset

Register	Offset
E0RXMBLR	3054h
E1RXMBLR	3154h

Function

This is the ENETC receive memory buffer limit register.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to ENETC register RXMBLR during IERB lock. All updates after this must be done through the corresponding ENETC register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								LIMIT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LIMIT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0	Receive buffer memory limit in words.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LIMIT	This limit is used by the per-ENETC receive queues implemented for ingress congestion control. Ingress frames will be discarded if queue length exceeds this limit. Value of 0 means the number of memory words used within a port's priority tier is limited to 16k-1.

53.4.6.3.93 ENETC a transmit high priority tier byte credit register (E0TXHPTBCR - E1TXHPTBCR)

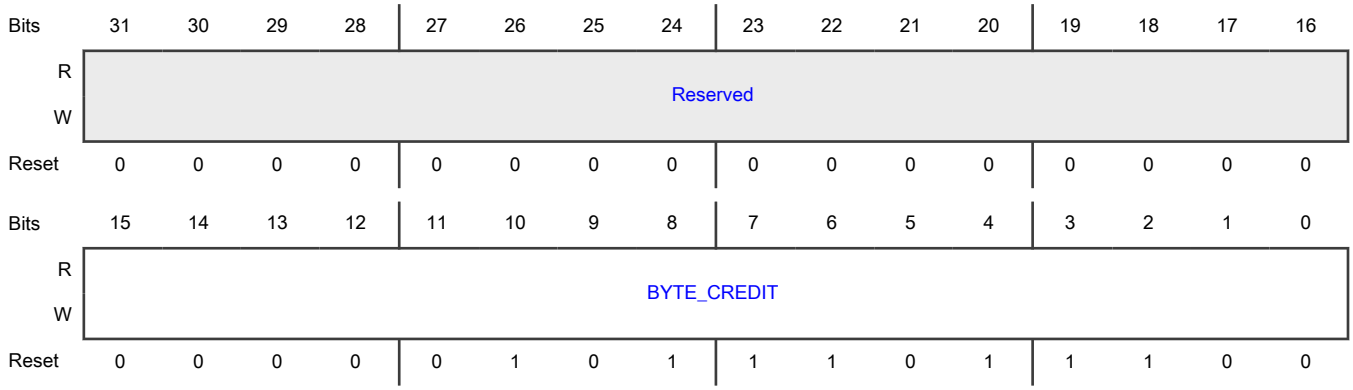
Offset

Register	Offset
E0TXHPTBCR	3070h
E1TXHPTBCR	3170h

Function

This is the ENETC transmit high priority tier byte credit register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 BYTE_CREDIT	This register field is used to configure the maximum number of high priority byte credits for the port-per-HTA Transmit priority (2 priorities) byte credit-based flow control mechanism. The port-per-HTA Transmit priority byte credit-based flow control mechanism is used to limit the amount of frame bytes in the ENETC transmit datapath that have been scheduled for transmission but not yet transferred to the MAC, for a given port. The byte credit count associated with the high priority tier is initialized to the value configured in this register field. The high priority tier is used for traffic directed to the express MAC or pseudo MAC. The byte credit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	count is decremented when a high priority tier frame is scheduled by HTA transmit and incremented in chunks of 72 bytes, as the frame is being sent to the MAC. HTA is allowed to schedule a frame as long as the byte credit count is higher than 0. This will limit the number of frame bytes in the ENETC high priority transmit path to "E0TXHPTBCR[BYTE_CREDIT] + maximum frame size". It is recommended to set the maximum number of high priority byte credits to the larger value of the following two formulas: • Maximum link speed that can be configured in PCR[PSPEED] divided by 8 (unit in bytes), and then multiplied by EaTGSLR[MIN_LOOKAHEAD]; amount of byte credits required when time gate scheduling operation is enabled. For example, with PCR[PSPEED] set to 2 Gbit/s and EaTGSLR[MIN_LOOKAHEAD] set to 10 µs, the formula yields 2500 byte credits. • 700 + (maximum frame size divided by 2); amount of byte credits required when time gate scheduling operation is disabled; assuming a maximum frame size of 1536 bytes, the formula yields 1500 byte credits (is round up to the nearest 100). A value of zero in this register field disables the port-per-HTA Transmit priority byte credit-based flow control mechanism, for the high-priority tier.

53.4.6.3.94 ENETC a transmit low priority tier byte credit register (E0TXLPTBCR - E1TXLPTBCR)

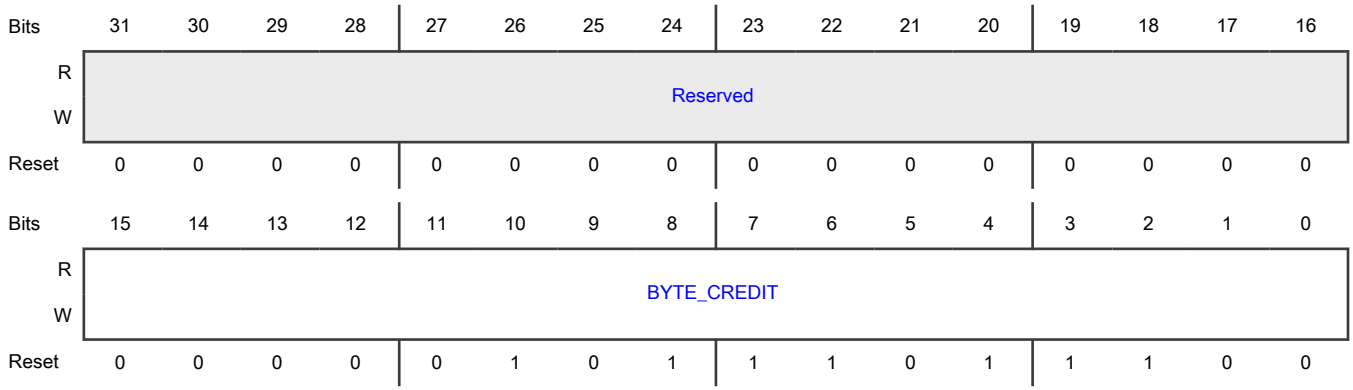
Offset

Register	Offset
E0TXLPTBCR	3074h
E1TXLPTBCR	3174h

Function

This is the ENETC transmit low priority tier byte credit register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 BYTE_CREDIT	This register field is used to configure the maximum number of low priority byte credits for the port-per-HTA Transmit priority (2 priorities) byte credit-based flow control mechanism. The port-per-HTA Transmit priority byte credit-based flow control mechanism is used to limit the amount of frame bytes in the ENETC transmit datapath that have been scheduled for transmission but not yet transferred to the MAC, for a given port. The byte credit count associated with the low priority tier is initialized to the value configured in this register field. The low priority tier only applies when IEEE 802.1Qbu frame preemption is enabled, and specifically used for traffic directed to the preemptable MAC. The byte credit count is decremented when a low priority tier frame is scheduled by HTA transmit and incremented in chunks of 72 bytes, as the frame is being sent to the MAC. HTA is allowed to schedule a frame as long as the byte credit count is higher than 0. This will limit the number of frame bytes in the ENETC low priority transmit path to "E0TXLPTBCR[BYTE_CREDIT] + maximum frame size". It is recommended to set the maximum number of low priority byte credits to the larger value of the following two formulas: • Maximum link speed that can be configured in PCR[PSPEED] divided by 8 (unit in bytes), and then multiplied by EaTGSLR[MIN_LOOKAHEAD]; amount of byte credits required when time gate scheduling operation is enabled. For example, with PCR[PSPEED] set to 2 Gbit/s and EaTGSLR[MIN_LOOKAHEAD] set to 10 µs, the formula yields 2500 byte credits. • 700 + (maximum frame size divided by 2); amount of byte credits required when time gate scheduling operation is disabled; assuming a maximum frame size of 1536 bytes, the formula yields 1500 byte credits (is round up to the nearest 100). A value of zero in this register field disables the port-per-HTA Transmit priority byte credit-based flow control mechanism, for the low-priority tier.

53.4.6.3.95 ENETC 0 hash table memory allotment register (E0HTMAR)

Offset

Register	Offset
E0HTMAR	3080h

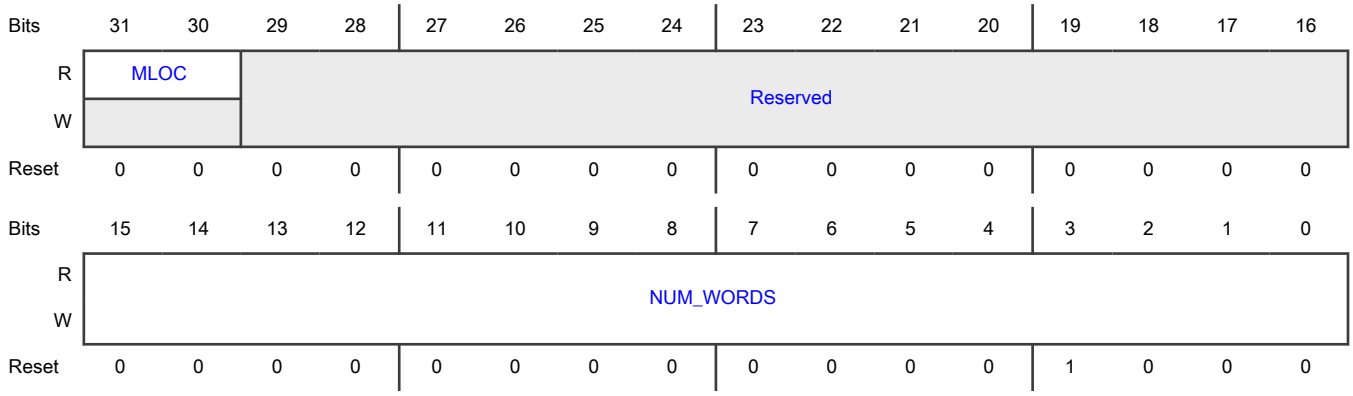
Function

This is the ENETC hash table memory allotment register.

Number of words allotted to exact match hash tables from the common memory's shared region. Software driver is responsible to manage how the hash table memory is distributed amongst various hash tables. The hash tables are:

- Ingress Stream Identification table
- Ingress Stream Filter table

Diagram



Fields

Field	Function
31-30 MLOC	Indicates memory location of this table 0: Common memory 1-3: Reserved
29-16 —	Reserved
15-0 NUM_WORDS	Number of words allotted to exact match hash table from the common memory's shared region. **NOTE** Reset value of ENETC's HTMCAPR register. This value is writable in IERB (by pre-boot initialization), which is then copied to the ENETC function register during IERB lock.

53.4.6.3.96 ENETC 0 index table memory allocation register (E0ITMAR)

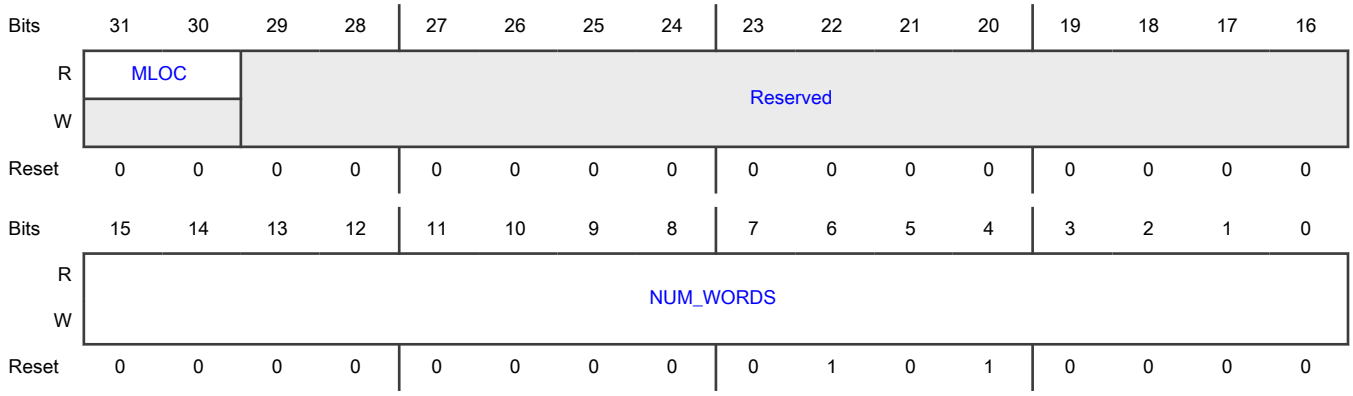
Offset

Register	Offset
E0ITMAR	3084h

Function

This is the ENETC index table memory allocation register.

Diagram



Fields

Field	Function
31-30 MLOC	Indicates memory location of this table 0: Common memory 1-3: Reserved
29-16 —	Reserved
15-0 NUM_WORDS	Number of WORDS allocated to ENETC's index table memory. **NOTE** Reset value of ENETC instance ITMCAPR register. Distribution of the table memory to various tables is done via various Index Table Memory Allocation Registers (xITMAR).

53.4.6.3.97 ENETC a ingress port filter table memory allocation register (E0IPFTMAR - E1IPFTMAR)

Offset

Register	Offset
E0IPFTMAR	3088h
E1IPFTMAR	3188h

Function

This is the ENETC ingress port filter table memory allocation register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ALLOC															
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 ALLOC	Number of words allocated to the ENETC Ingress Port Filter table from ingress port filter ternary memory. **NOTE** Reset value for ENETC register IPFTCAPR.

53.4.6.3.98 ENETC 0 rate policer index table memory allocation register (E0RPITMAR)

Offset

Register	Offset
E0RPITMAR	30A0h

Function

This is the ENETC rate policer index table memory allocation register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NUM_WORDS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. Number of entries assigned to the table is ROUNDDOWN(NUM_WORDS / 4). This value is writable in IERB (by pre-boot initialization), which is R/W in the ENETC version of this register. **NOTE** Each entry occupies 4 words.

53.4.6.3.99 ENETC 0 ingress stream counter index table memory allocation register (E0ISCITMAR)

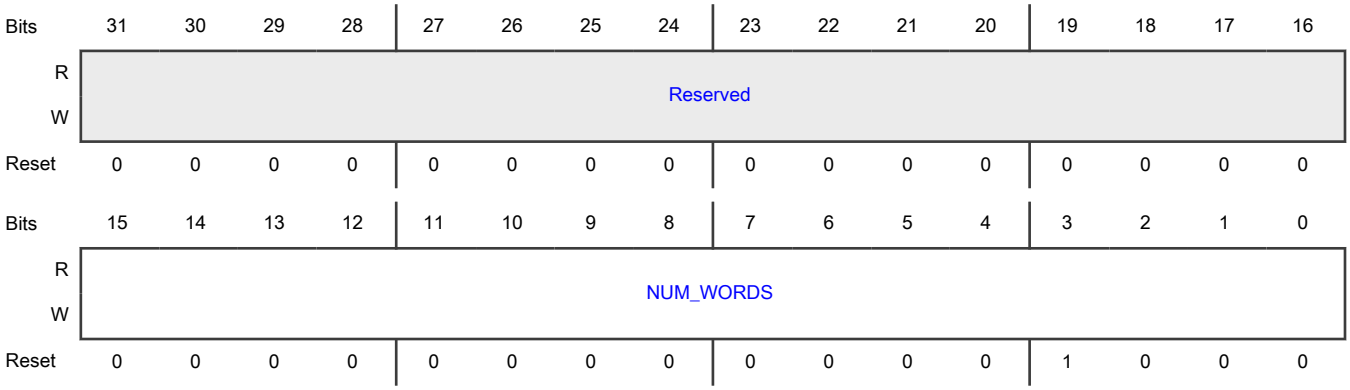
Offset

Register	Offset
E0ISCITMAR	30A4h

Function

This is the ENETC ingress stream counter index table memory allocation register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0	The number of words from index table memory assigned to this table.

Table continues on the next page...

Table continued from the previous page...

Field	Function
NUM_WORDS	**NOTE** Each entry occupies 1 word.

53.4.6.3.100 ENETC 0 ingress stream index table memory allocation register (E0ISITMAR)

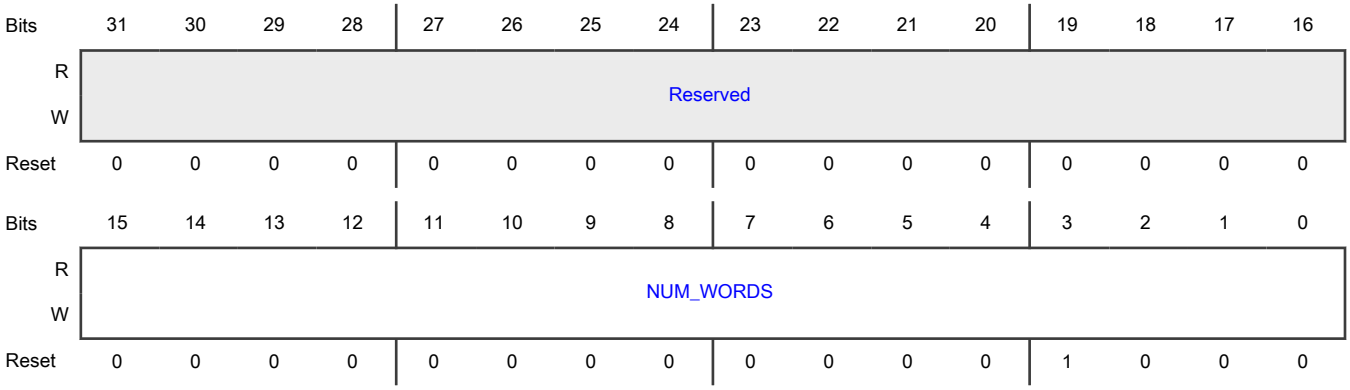
Offset

Register	Offset
E0ISITMAR	30A8h

Function

This is the ENETC ingress stream index table memory allocation register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. This value is writable in IERB (by pre-boot initialization), which is R/W in the ENETC version of this register. **NOTE** Each entry occupies 1 word.

53.4.6.3.101 ENETC 0 stream gate instance index table memory allocation register (E0SGIITMAR)

Offset

Register	Offset
E0SGIITMAR	30B4h

Function

This is the ENETC stream gate instance index table memory allocation register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NUM_WORDS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. This value is writable in IERB (by pre-boot initialization), which is R/W in the ENETC version of this register. **NOTE** Each entry occupies 1 word.

53.4.6.3.102 ENETC 0 stream gate control list index table memory allocation register (E0SGCLITMAR)

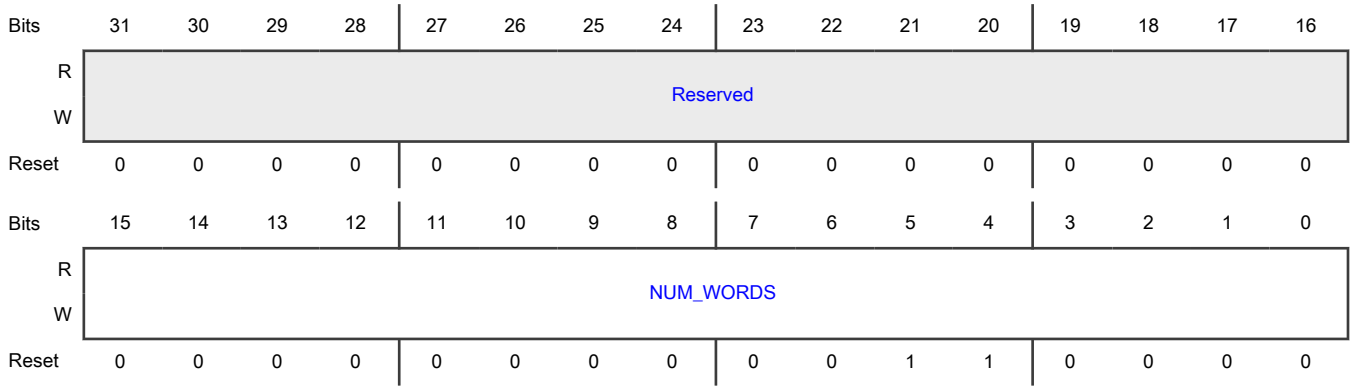
Offset

Register	Offset
E0SGCLITMAR	30B8h

Function

This is the ENETC stream gate control list index table memory allocation register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** Each entry utilizes a variable number of words depending on the number of gates specified. Entry size in words is calculated as: 1 + ROUNDUP(NUM_GATES * 0.5).

53.4.6.3.103 ENETC a time gate scheduling table allocation register (E0TGSTAR - E1TGSTAR)

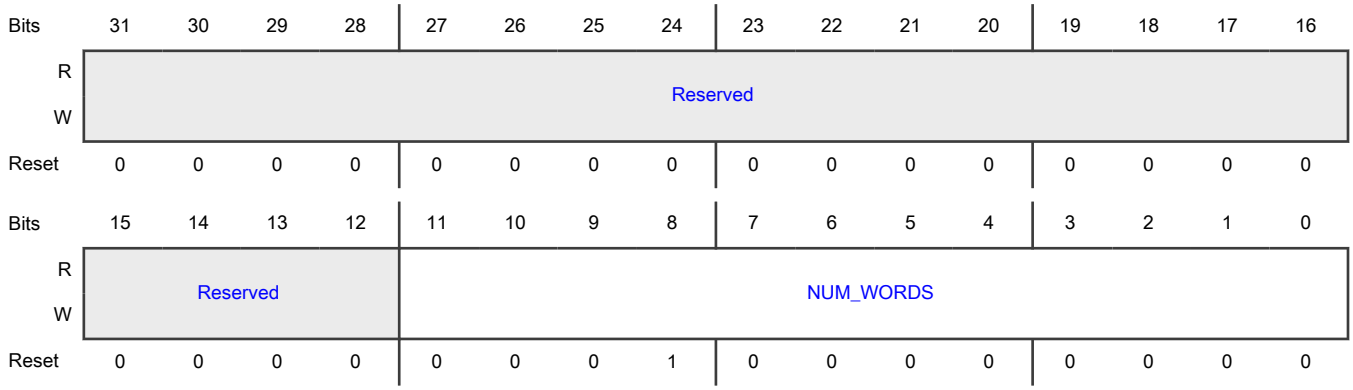
Offset

Register	Offset
E0TGSTAR	30F0h
E1TGSTAR	31F0h

Function

This is the ENETC time gate scheduling table allocation register.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 NUM_WORDS	This field specifies the number of words in the Time Gate Scheduling internal memory (TGSMCAPR) allocated to support the ENETC instance a Time Gate Scheduling table, which in turn contains the administrative gate control list and the operational gate control list for the ENETC instance. The Time Gate Scheduling internal memory is shared between the switch and all of the ENETC instances supporting time gate scheduling. The sum of all of the individual allocation (S0TGSTAR and E*a*TGSTARs) must not exceed the total size of the Time Gate Scheduling internal memory. The memory withing this allocation is managed using a single free list of fixed equal size units (or words) of memory. Free memory units from the free list are available to be dynamically allocated to any gate control lists subject to the total memory available. Each gate control list is comprised of a sequence of gate operation entries where each entry occupies one word of memory. Note that a gate control list can contain a variable number of gate control operation entries. **NOTE** Reset value of ENETC register TGSTCAPR.

53.4.6.3.104 ENETC 0 time gate scheduling lookahead register (E0TGSLR)

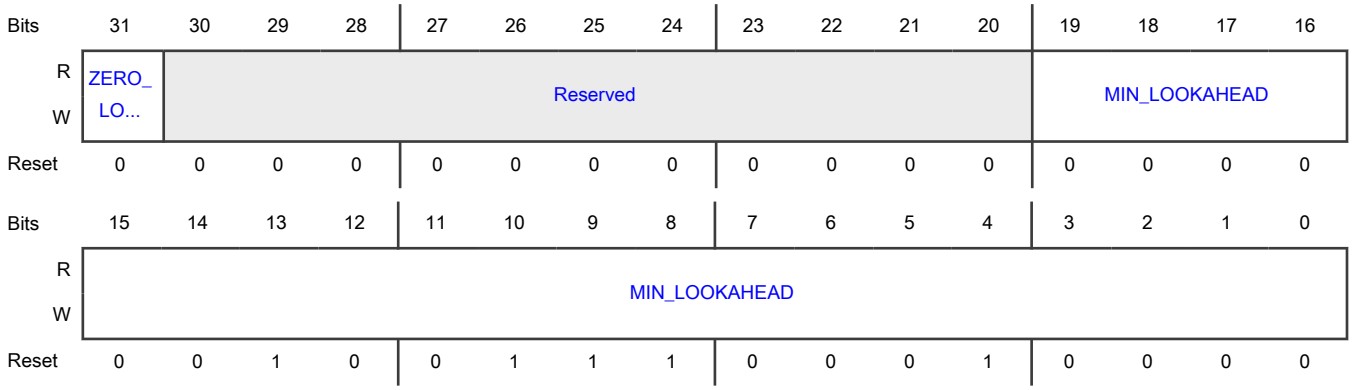
Offset

Register	Offset
E0TGSLR	30F4h

Function

This is the time gate scheduling lookahead register.

Diagram



Fields

Field	Function
31 ZERO_LOOKAHEAD	Zero Lookahead **NOTE** For ENETC bound to pseudo MAC, the default reset behaviour is to rate limit the port to the rate specified in PCR[PSPEED] without any DMA compensation delay. Only applicable if ENETC bound to pseudo MAC (L*a*CAPR[LINKTYPE=1). There is no equivalent field for the switch as there is no need to compensate for any DMA delay; power on reset value for the S0TGSLR[MIN_LOOKAHEAD] field is suitable to rate limit a switch port bound to pseudo MAC. 0b - Use MIN_LOOKAHEAD value 1b - If a gate control list is configured or when time specific departure is enabled on any traffic class (PTCaTSDR[TSDE] set to 1, where a corresponds to the traffic class number), use MIN_LOOKAHEAD value, otherwise use value of zero
30-20 —	Reserved
19-0 MIN_LOOKAHEAD	Minimum lookahead This field specifies the amount of time to advance the IEEE 1588 time scale used by the time-based scheduler (at the frame scheduling timing point), to account for the time required to schedule, dequeue and load (or DMA) frames from the host memory to NETC internal memory. The time is specified in units of nanoseconds. The IEEE 1588 time scale used by the time-based scheduler, can also be advanced, on per port basis, by the time amount specified in PTGSATOR[ADV_TIME_OFFSET], to adjust for delay across the MAC plus if needed, delays outside of NETC (e.g. PHY delay). Both advanced times are cumulative, the IEEE 1588 time scale used by the time-based scheduler (at the frame scheduling timing point) on a given port, will be advanced by the amount of time resulting from adding the EaTGSLR[MIN_LOOKAHEAD] time to the PTGSATOR[ADV_TIME_OFFSET] time. Setting a too low value in the case of time gate scheduling, can cause a frame to miss it's scheduled time-gated window (frame dropped with appropriate error reported in return BD) or can cause a frame to

Table continues on the next page...

Table continued from the previous page...

Field	Function
	overrun its scheduled time-gated window (not detected). In the case of time specific departure, it can cause the frame to be transmitted later than the specified transmit time. Setting a too high value in the case of time gate scheduling, can cause a port to consume excessive transmit internal memory. It is recommended to set the minimum lookahead to 10,000 nanoseconds.

53.4.6.3.105 ENETC 1 binding configuration register 0 (E1BCR0)

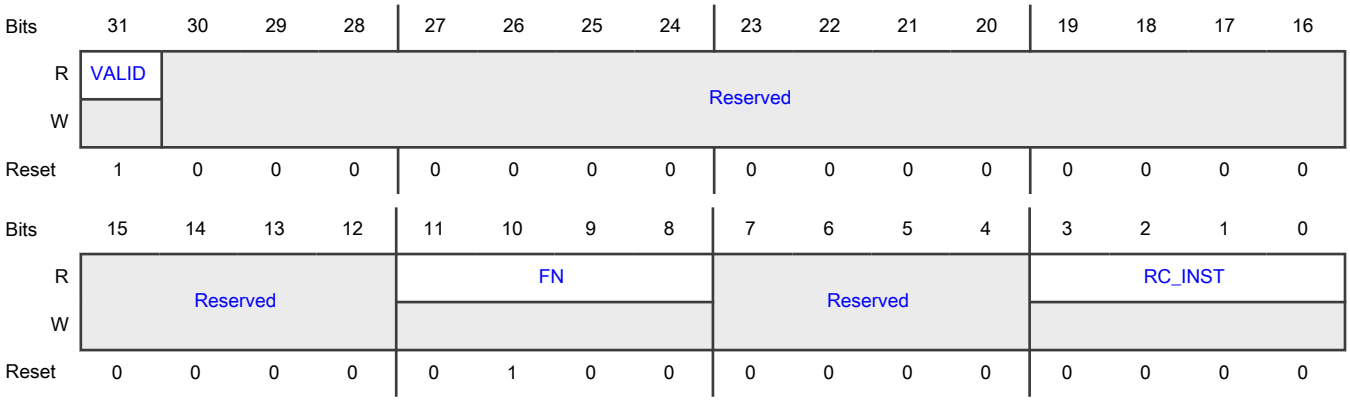
Offset

Register	Offset
E1BCR0	3100h

Function

This is the ENETC binding configuration register 0.

Diagram



Fields

Field	Function
31 VALID	If set, this ENETC instance is associated to a link end.
30-12 —	Reserved
11-8	PCI device function number.

Table continues on the next page...

Table continued from the previous page...

Field	Function
FN	**NOTE** For assignment of function number, see PCIe Function Number Assignment.
7-4 —	Reserved
3-0 RC_INST	Root complex instance this function is bound to.

53.4.6.3.106 ENETC 1 binding configuration register 1 (E1BCR1)

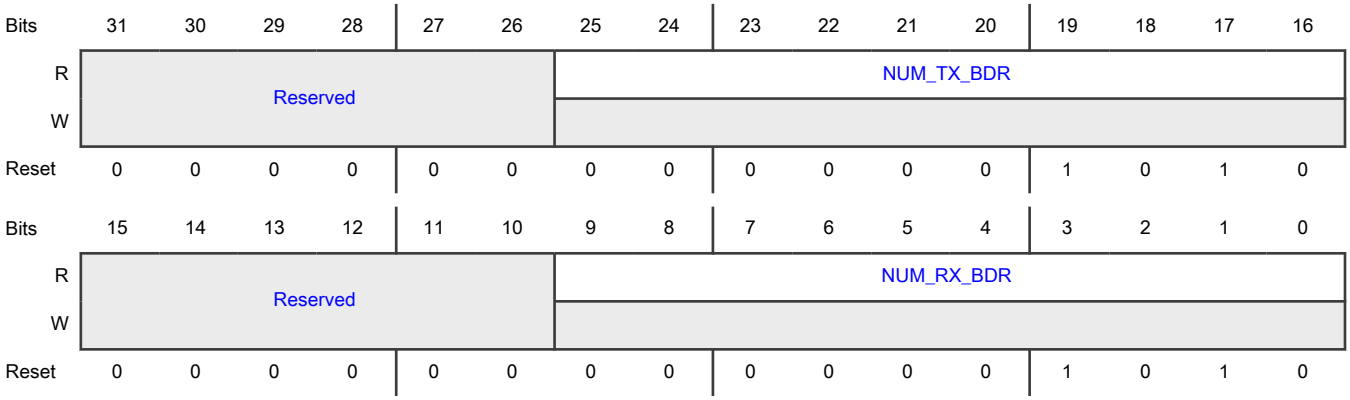
Offset

Register	Offset
E1BCR1	3104h

Function

This is the ENETC binding configuration register 1.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16	Number of Tx BD rings supported by ENETC.

Table continues on the next page...

Table continued from the previous page...

Field	Function
NUM_TX_BDR	Range: 0..1023 Reset value of ENETC register field ECAPR0[NUM_TX_BDR]. **NOTE** If total number of allocated Rx BD rings exceed CAPR1[NUM_RX_BDR], error is indicated by NETCSR[ERROR].
15-10 —	Reserved
9-0 NUM_RX_BDR	Number of Rx BD rings supported by ENETC. Range: 0..1023 Reset value of ENETC register field ECAPR0[NUM_RX_BDR]. **NOTE** If total number of allocated Rx BD rings exceed CAPR1[NUM_RX_BDR], error is indicated by NETCSR[ERROR].

53.4.6.3.107 ENETC 1 VSI binding configuration register (E1VBCR)

Offset

Register	Offset
E1VBCR	3110h

Function

This is the ENETC VSI binding configuration register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												NUM_VSI			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-4 —	Reserved
3-0 NUM_VSI	Indicates the number of VSIs supported for this ENETC instance. **NOTE** VSI PCIe function numbers will be sequentially assigned. The VSI 0 function number can be determined from the PCIe SR-IOV capability registers. If allocated VSI instances exceed CAPR0[NUM_VSI], error is indicated by NETCSR[ERROR].

53.4.6.3.108 ENETC 1 MSI-X configuration register (E1MCR)

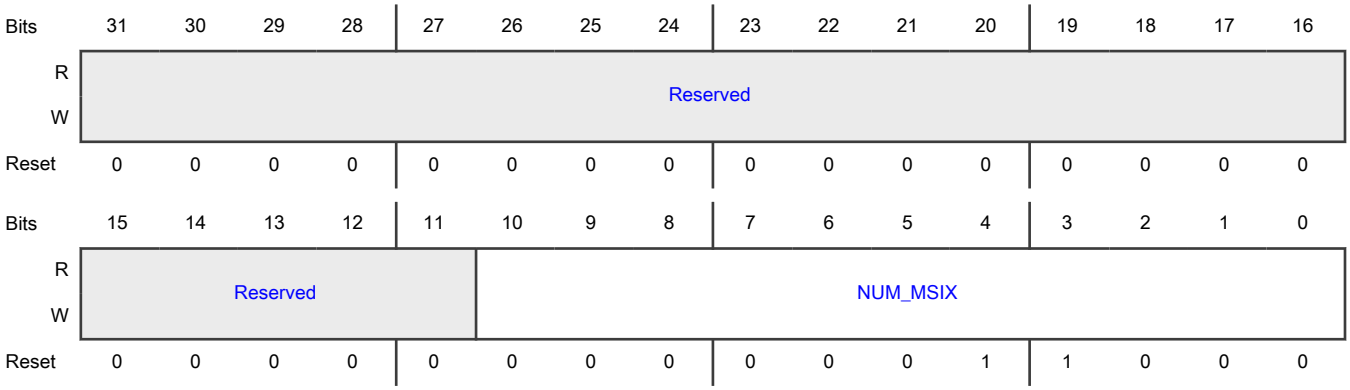
Offset

Register	Offset
E1MCR	3114h

Function

This is the ENETC MSI-X configuration register.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-0	Number of MSI-X vectors supported for ENETC function.

Table continues on the next page...

Table continued from the previous page...

Field	Function
NUM_MSIX	Formula: NUM_MSIX+1 Range: 1..1024 **NOTE** Reset value of ENETC register ECAPR1[NUM_MSIX]. If total number of allocated MSI-X vectors exceed CAPR2[NUM_MSIX], error is indicated by NETCSR[ERROR].

53.4.6.3.109 ENETC 1 config header device ID and vendor ID register (E1_CFH_DIDVID)

Offset

Register	Offset
E1_CFH_DIDVID	3120h

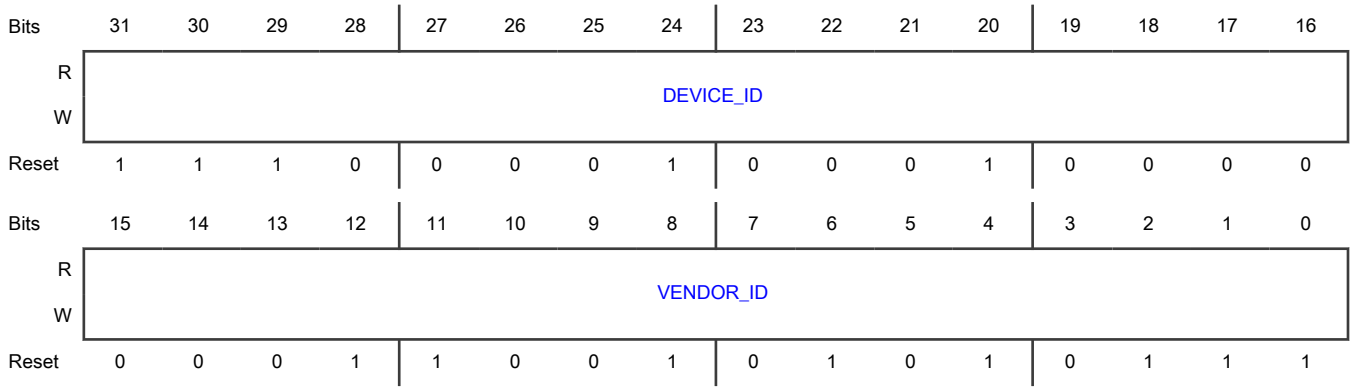
Function

This is the ENETC PCI Device and Vendor ID register.

NOTE

This register is continuously sampled, any changes will have immediate effect on reading the PCI function Type0 header.

Diagram



Fields

Field	Function
31-16 DEVICE_ID	Device ID This field identifies the device ID of the device shown in the PCIe Device ID Register (02h).
15-0	Vendor ID

Table continues on the next page...

Table continued from the previous page...

Field	Function
VENDOR_ID	This field identifies the manufacturer of the device as shown in the PCIe Vendor ID Register (00h). The default value will indicate Freescale (0x1957).

53.4.6.3.110 ENETC 1 config header subsystem ID and subsystem vendor ID register (E1_CFH_SIDSVID)

Offset

Register	Offset
E1_CFH_SIDSVID	3124h

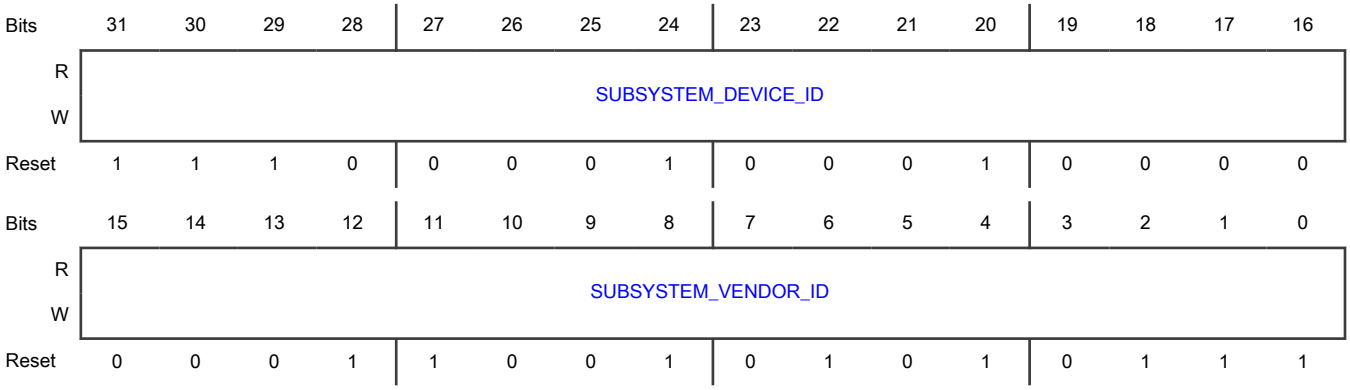
Function

This is the ENETC PCI Subsystem ID and Subsystem Vendor ID register.

NOTE

This register is continuously sampled, any changes will have immediate effect on reading the PCI function Type0 header.

Diagram



Fields

Field	Function
31-16 SUBSYSTEM_DEVICE_ID	Subsystem Device ID This field identifies the particular subsystem as shown in the PCIe Subsystem ID Register (2Eh).
15-0 SUBSYSTEM_VENDOR_ID	Subsystem Vendor ID This field identifies the manufacturer of the subsystem as shown in the PCIe Subsystem Vendor ID Register (2Ch). The default value is the same as E*a*_CFH_DIDVID[VENDOR_ID].

53.4.6.3.111 ENETC 1 hash table memory allotment register (E1HTMAR)

Offset

Register	Offset
E1HTMAR	3180h

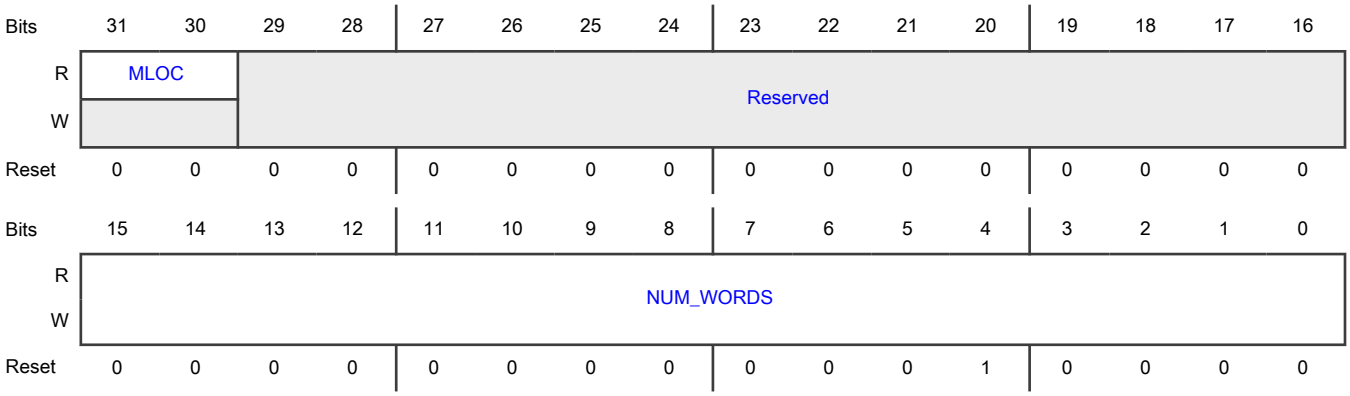
Function

This is the ENETC hash table memory allotment register.

Number of words allotted to exact match hash tables from the common memory's shared region. Software driver is responsible to manage how the hash table memory is distributed amongst various hash tables. The hash tables are:

- Ingress Stream Identification table
- Ingress Stream Filter table

Diagram



Fields

Field	Function
31-30 MLOC	Indicates memory location of this table 0: Common memory 1-3: Reserved
29-16 —	Reserved
15-0 NUM_WORDS	Number of words allotted to exact match hash table from the common memory's shared region. **NOTE** Reset value of ENETC's HTMCAPR register. This value is writable in IERB (by pre-boot initialization), which is then copied to the ENETC function register during IERB lock.

53.4.6.3.112 ENETC 1 index table memory allocation register (E1ITMAR)

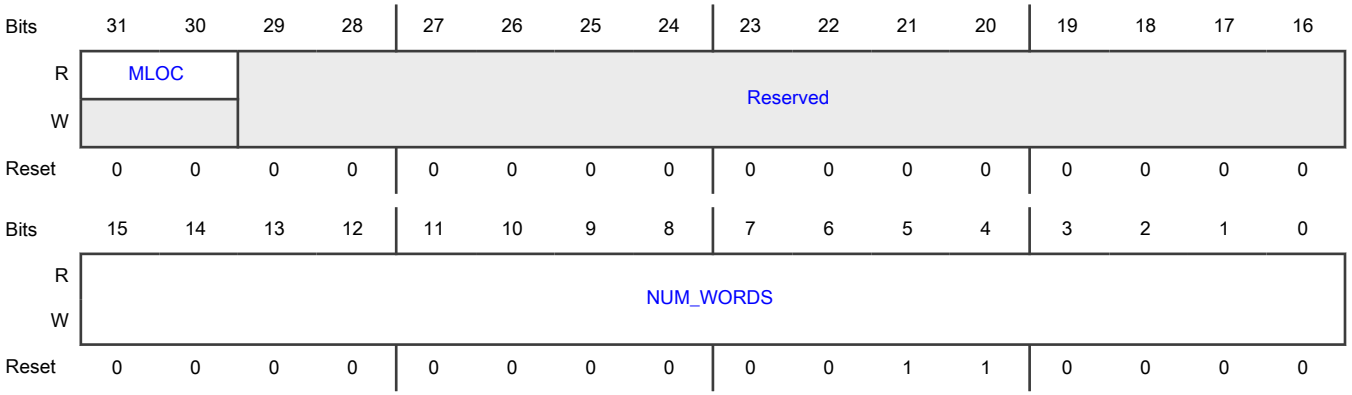
Offset

Register	Offset
E1ITMAR	3184h

Function

This is the ENETC index table memory allocation register.

Diagram



Fields

Field	Function
31-30 MLOC	Indicates memory location of this table 0: Common memory 1-3: Reserved
29-16 —	Reserved
15-0 NUM_WORDS	Number of WORDS allocated to ENETC's index table memory. **NOTE** Reset value of ENETC instance ITMCAPR register. Distribution of the table memory to various tables is done via various Index Table Memory Allocation Registers (xITMAR).

53.4.6.3.113 ENETC 1 rate policer index table memory allocation register (E1RPITMAR)

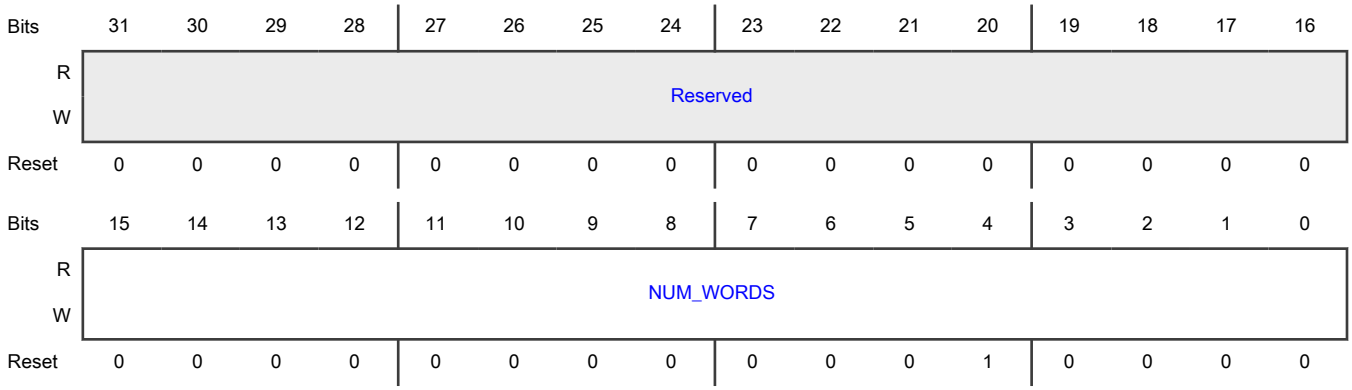
Offset

Register	Offset
E1RPITMAR	31A0h

Function

This is the ENETC rate policer index table memory allocation register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. Number of entries assigned to the table is ROUNDDOWN(NUM_WORDS / 4). This value is writable in IERB (by pre-boot initialization), which is R/W in the ENETC version of this register. **NOTE** Each entry occupies 4 words.

53.4.6.3.114 ENETC 1 ingress stream counter index table memory allocation register (E1ISCITMAR)

Offset

Register	Offset
E1ISCITMAR	31A4h

Function

This is the ENETC ingress stream counter index table memory allocation register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NUM_WORDS															
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** Each entry occupies 1 word.

53.4.6.3.115 ENETC 1 ingress stream index table memory allocation register (E1ISITMAR)

Offset

Register	Offset
E1ISITMAR	31A8h

Function

This is the ENETC ingress stream index table memory allocation register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NUM_WORDS															
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. This value is writable in IERB (by pre-boot initialization), which is R/W in the ENETC version of this register. **NOTE** Each entry occupies 1 word.

53.4.6.3.116 ENETC 1 stream gate instance index table memory allocation register (E1SGIITMAR)

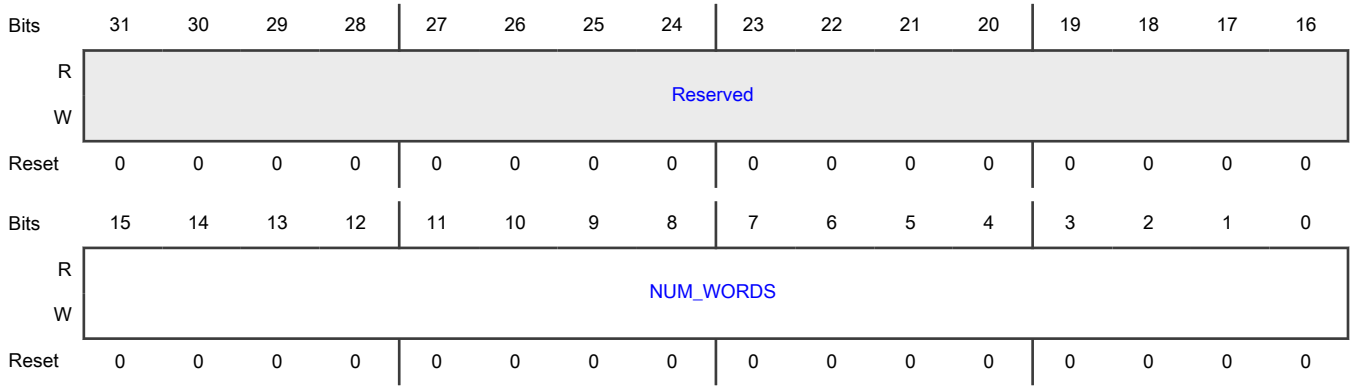
Offset

Register	Offset
E1SGIITMAR	31B4h

Function

This is the ENETC stream gate instance index table memory allocation register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. This value is writable in IERB (by pre-boot initialization), which is R/W in the ENETC version of this register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Each entry occupies 1 word.

53.4.6.3.117 ENETC 1 stream gate control list index table memory allocation register (E1SGCLITMAR)

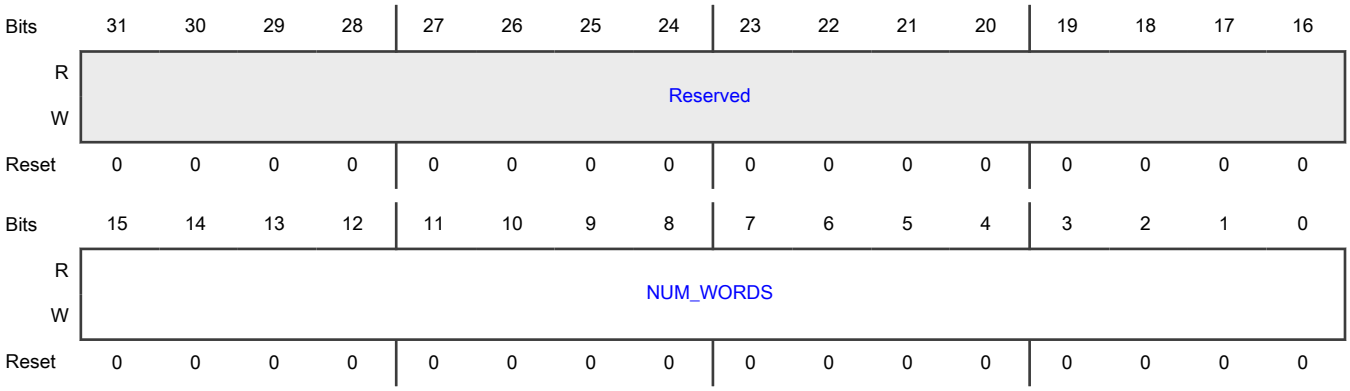
Offset

Register	Offset
E1SGCLITMAR	31B8h

Function

This is the ENETC stream gate control list index table memory allocation register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** Each entry utilizes a variable number of words depending on the number of gates specified. Entry size in words is calculated as: 1 + ROUNDUP(NUM_GATES * 0.5).

53.4.6.3.118 ENETC 1 time gate scheduling lookahead register (E1TGSLR)

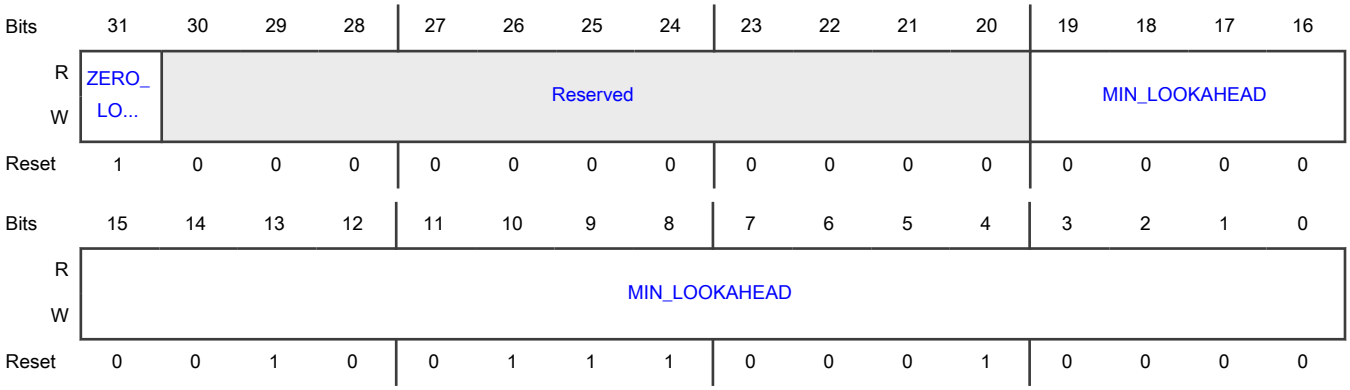
Offset

Register	Offset
E1TGSLR	31F4h

Function

This is the time gate scheduling lookahead register.

Diagram



Fields

Field	Function
31 ZERO_LOOKAHEAD	Zero Lookahead **NOTE** For ENETC bound to pseudo MAC, the default reset behaviour is to rate limit the port to the rate specified in PCR[PSPEED] without any DMA compensation delay. Only applicable if ENETC bound to pseudo MAC (L*a*CAPR[LINKTYPE=1). There is no equivalent field for the switch as there is no need to compensate for any DMA delay; power on reset value for the S0TGSLR[MIN_LOOKAHEAD] field is suitable to rate limit a switch port bound to pseudo MAC. 0b - Use MIN_LOOKAHEAD value 1b - If a gate control list is configured or when time specific departure is enabled on any traffic class (PTCaTSDR[TSDE] set to 1, where a corresponds to the traffic class number), use MIN_LOOKAHEAD value, otherwise use value of zero
30-20 —	Reserved
19-0	Minimum lookahead

Table continues on the next page...

Table continued from the previous page...

Field	Function
MIN_LOOKAHEAD	This field specifies the amount of time to advance the IEEE 1588 time scale used by the time-based scheduler (at the frame scheduling timing point), to account for the time required to schedule, dequeue and load (or DMA) frames from the host memory to NETC internal memory. The time is specified in units of nanoseconds. The IEEE 1588 time scale used by the time-based scheduler, can also be advanced, on per port basis, by the time amount specified in PTGSATOR[ADV_TIME_OFFSET], to adjust for delay across the MAC plus if needed, delays outside of NETC (e.g. PHY delay). Both advanced times are cumulative, the IEEE 1588 time scale used by the time-based scheduler (at the frame scheduling timing point) on a given port, will be advanced by the amount of time resulting from adding the EaTGSLR[MIN_LOOKAHEAD] time to the PTGSATOR[ADV_TIME_OFFSET] time. Setting a too low value in the case of time gate scheduling, can cause a frame to miss it's scheduled time-gated window (frame dropped with appropriate error reported in return BD) or can cause a frame to overrun its scheduled time-gated window (not detected). In the case of time specific departure, it can cause the frame to be transmitted later than the specified transmit time. Setting a too high value in the case of time gate scheduling, can cause a port to consume excessive transmit internal memory. It is recommended to set the minimum lookahead to 10,000 nanoseconds.

53.4.6.3.119 VSI 0 access management qualifier register (V0AMQR)

Offset

Register	Offset
V0AMQR	4000h

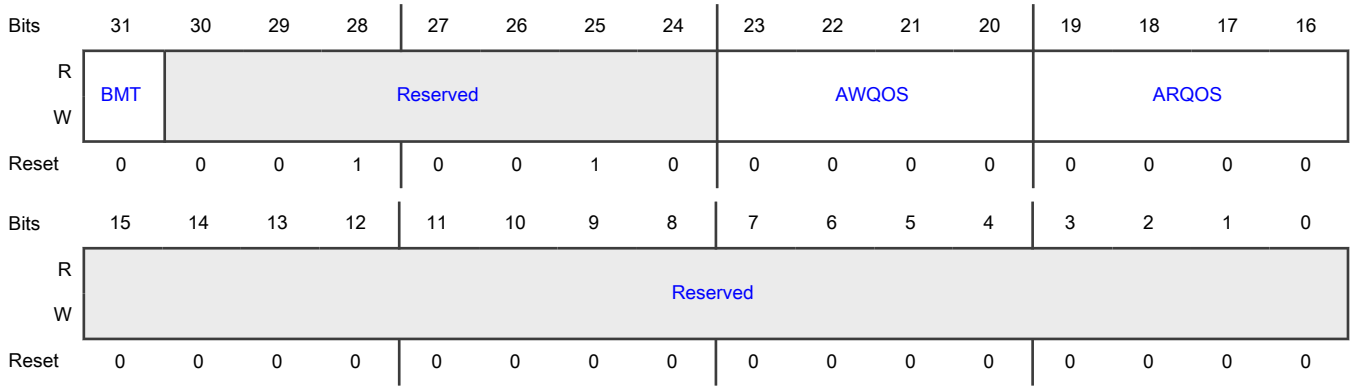
Function

This is the VSI access management qualifier register. It provides attributes for each memory access transaction performed by the VSI instance.

NOTE

Changing the value in this register has immediate effect unless otherwise noted. Care should be taken when making changes to the values for an enabled and operational virtual function to ensure that consistent behavior is maintained. It is strongly encouraged, but not required, that the virtual function be disabled before making changes.

Diagram



Fields

Field	Function
31 BMT	Bypass memory translation The bit is an indication to the SMMU to by-pass memory translation whenever the PF is performing memory transactions, effectively handling the memory address as a true physical address. 0 Memory translation 1 Bypass memory translation
30-24 —	Reserved
23-20 AWQOS	Address Write QoS. AWQOS[3:1] signals are controlled by this register field's bits [3:1]. AWQOS[0] is controlled by the HTA block and is 0 for low priority DMA transactions and 1 for high priority DMA transactions.
19-16 ARQOS	Address Read QoS. ARQOS[3:1] signals are controlled by this register field's bits [3:1]. ARQOS[0] is controlled by the HTA block and is 0 for low priority DMA transactions and 1 for high priority DMA transactions.
15-0 —	Reserved

53.4.6.3.120 VSI 0 boot loader parameter register b (V0BLPR0 - V0BLPR1)

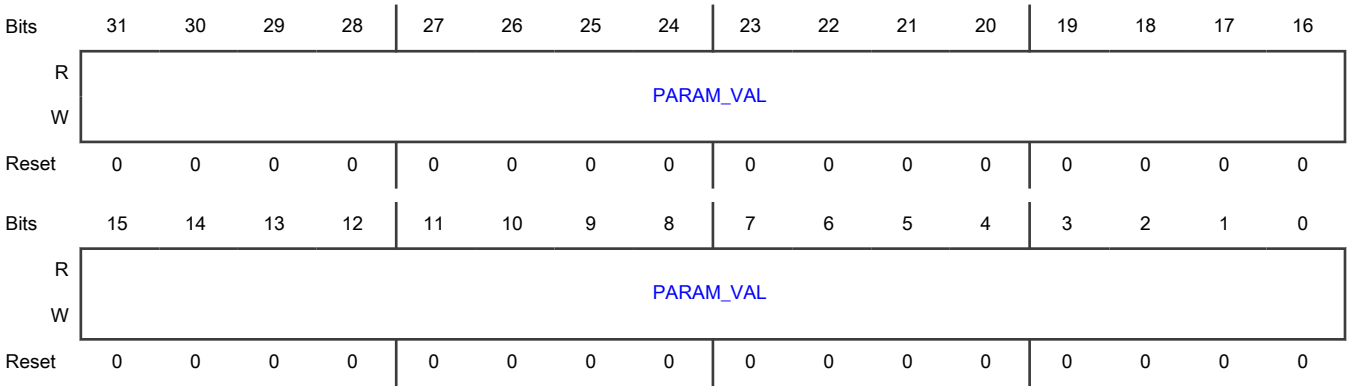
Offset

Register	Offset
V0BLPR0	4008h
V0BLPR1	400Ch

Function

This register provides a mean for boot S/W (Pre-Boot Loader, boot ROM) to communicate parameters with running (device driver) software. Runtime software uses read-only global register V*a*FBLPR*b*.

Diagram



Fields

Field	Function
31-0 PARAM_VAL	Boot loader parameter value.

53.4.6.3.121 VSI 0 primary MAC address register 0 (V0PMAR0)

Offset

Register	Offset
V0PMAR0	4010h

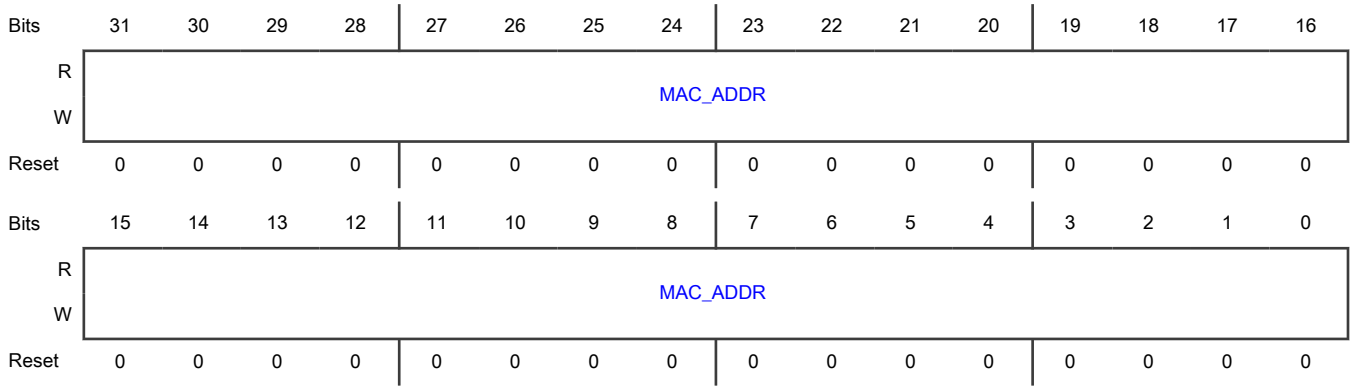
Function

This is the VSI instance primary MAC address register 0. It determines the initial value for ENETC register PSI*a*PMAR0, where a = 1, 2 .. n. .

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to ENETC register PSI*a*PMAR0 , where a = 1, 2 .. n, during IERB lock; V0PMAR0 is copied to PSI1PMAR0, V1PMAR0 is copied to PSI2PMAR0, and so on. All updates after this must be done through the corresponding register PSI*a*PMAR0.

Diagram



Fields

Field	Function
31-0 MAC_ADDR	MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset (register bit offset 0-7). This field contains the 4 most significant bytes of the MAC address. For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then V*a*PMAR0 equals 14131211h.

53.4.6.3.122 VSI 0 primary MAC address register 1 (V0PMAR1)

Offset

Register	Offset
V0PMAR1	4014h

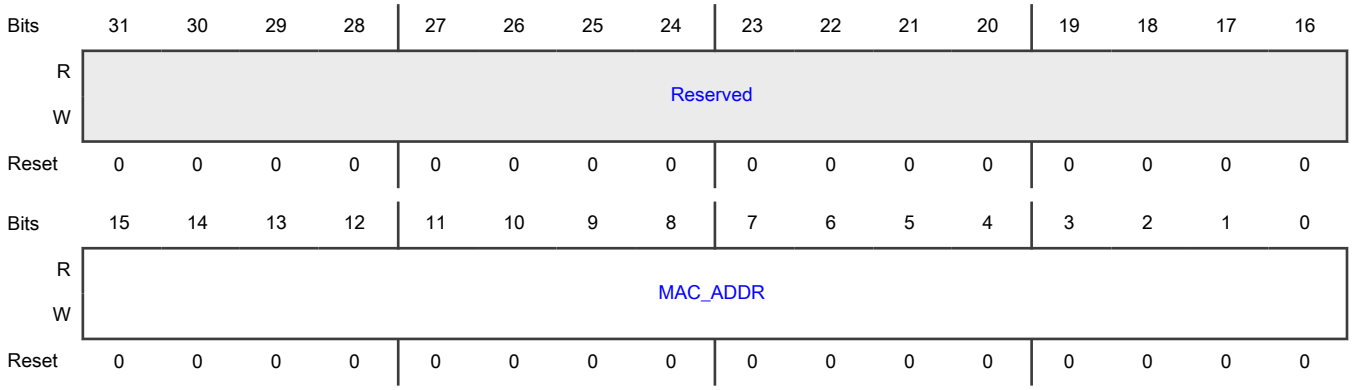
Function

This is the VSI instance primary MAC address register 1. It determines the initial value for register PSI*a*PMAR1, where a = 1, 2 .. n.

NOTE

This value is writable in IERB (by pre-boot initialization), which is then copied to ENETC register PSI*a*PMAR1 , where a = 1, 2 .. n, during IERB lock; V0PMAR1 is copied to PSI1PMAR1, V1PMAR1 is copied to PSI2PMAR1, and so on. All updates after this must be done through the corresponding register PSI*a*PMAR1.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 MAC_ADDR	MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset. This field contains the 2 least significant bytes of the MAC address (least significant byte is stored in register bit offset 8-15). For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then V*a*PMAR1 equals xxxx1615h (where x should be set to 0)

53.4.6.4 NETC privileged register descriptions

This section describes the NETC privileged register page which is 64KB in size. It controls global reset and global error handling for NETC. Errors that affect only a function such as ENETC or switch have a similar set of registers in their respective address mapped region.

53.4.6.4.1 Privileged memory map

NETC_PRIV pcie ea bei 0 offset: 6090_0000h

Offset	Register	Width (In bits)	Access	Reset value
100h	NETC reset register (NETCRR)	32	RW	See section
104h	NETC status register (NETCSR)	32	R	0000_0000h
208h	Memory Error Injection Config Register (MEICR)	32	RW	0000_0000h
E00h	Correctable memory error configuration register (CMECR)	32	RW	0000_0000h
E04h	Correctable memory error status register (CMESR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
E0Ch	Correctable memory error count register (CMECTR)	32	R	0000_0000h
E30h	Uncorrectable non-fatal memory error configuration register (UNMECR)	32	RW	0000_0000h
E34h	Uncorrectable non-fatal memory error status register 0 (UNMESR0)	32	RW	0000_0000h
E38h	Uncorrectable non-fatal memory error status register 1 (UNMESR1)	32	R	0000_0000h
E3Ch	Uncorrectable non-fatal memory error count register (UNMECTR)	32	R	0000_0000h
E40h	Uncorrectable fatal memory error configuration register (UFMECR)	32	RW	0000_0000h
E44h	Uncorrectable fatal memory error status register 0 (UFMESR0)	32	RW	0000_0000h
E48h	Uncorrectable fatal memory error status register 1 (UFMESR1)	32	R	0000_0000h

53.4.6.4.2 NETC reset register (NETCRR)

Offset

Register	Offset
NETCRR	100h

Function

This is the NETC reset register. Refer to Reset section for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														LOCK	SR
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	u[1]	0

1. Configured by SoC

Fields

Field	Function
31-2 —	Reserved
1 LOCK	Lock This bit can be read after reset to determine if the IERB registers are configurable. Registers are writable only when LOCK=0. If set to 1, while LOCK=0, all IERB configuration registers will be write inhibited. This also starts the initialization sequence of NETC, see NETCSR[STATE]. The lock is complete when NETCSR[STATE]=0. If written with 0, while LOCK=1, IERB configuration registers are unlocked and a warm reset is performed. The LOCK bit will return to 0 after warm reset sequence is done, hence software may poll this bit for completion. **NOTE** Warm reset is not to be used for interrupting NETC initialization sequence.
0 SR	Soft reset When set (0b1), resets all registers, and essential logic back to POR state. Register NETCFLRCR indicates the minimum amount of time for the reset sequence to complete. All functions are issued a request to quiesce during this time. This bit will self-clear and software may poll this bit to determine when soft reset sequence is complete.

53.4.6.4.3 NETC status register (NETCSR)

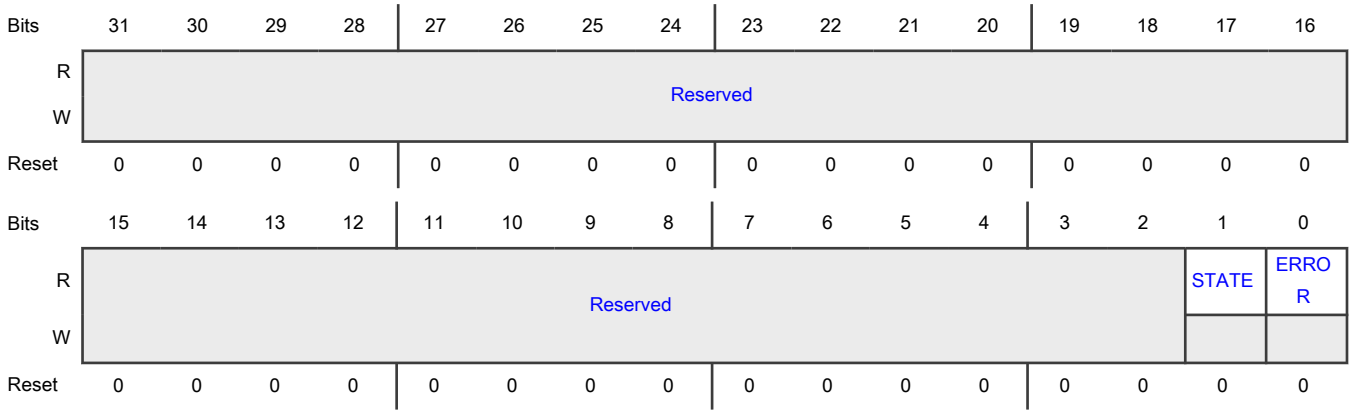
Offset

Register	Offset
NETCSR	104h

Function

This is the NETC status register.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 STATE	Indicates NETC's global operational state 0: NETC is ready for operation. 1: NETC has inhibited write access to IERB registers and is in the process of initializing NETC. The OSR register in NETC base functions include the state of this bit and may also include function specific initialization requirements. A function should wait to become operational until after the OSR[STATE] bit clears.
0 ERROR	Error 0: Configuration is valid. 1: The current IERB configuration setting is invalid due to error in binding or resource allocation. Operating NETC in current condition could lead to undefined behavior. The following IERB registers are checked for compliance: • IPFTMCAPR[NUM_WORDS] • TGSMCAPR[NUM_WORDS] • HBTMAR[NUM_WORDS] • L*a*BCR • E*a*BCR1[NUM_RX_BDR/NUM_TX_BDR] • E*a*BCR2[NUM_VSI] • T*a*MCR[NUM_MSIX], S*a*MCR[NUM_MSIX], E*a*MCR[NUM_MSIX] The following I/O configuration inputs are checked for compliance: • L*a*MCAPR[MII_PROT] • L*a*IOCAPR[IO_VAR]

53.4.6.4.4 Memory Error Injection Config Register (MEICR)

Offset

Register	Offset
MEICR	208h

Function

NETC is capable of injecting memory faults in all the locations where ECC is checked (on all memory read interfaces). One bit is flipped on the data bus for single bit errors, and two bits for multi-bit errors.

Software must arm the Safety Mechanism (SM), and the next transaction coming through this SM will trigger the fault, detection, and reporting. It is up to software to target a specific PCI function, or a specific engine (using engine disables) if a specific error detection and report is desired.

Only one fault injection can be configured at a time.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EN								ARM		Reserved					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												MEM_ID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 EN	Enable A write value of h86 will enable this function. Any other value will disable it.
23-22 ARM	Armed 0: Disabled 1: Single Bit ECC Error 2: Multi Bit ECC Error 3: Reserved (disabled) This field is self-clearing upon memory error injection completion. S/W may disarm error injection at any time but prior trigger may have completed.
21-5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4-0 MEM_ID	Memory ID See Table 392 for list of memory identifiers.

53.4.6.4.5 Correctable memory error configuration register (CMECR)

Offset

Register	Offset
CMECR	E00h

Function

The correctable memory error configuration register can disable the event reporting to the Root Complex Event Collector. While correctable errors do not cause any harm, detecting many of these events could indicate a more severe issue. The threshold can be set to trigger the event.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								THRESHOLD							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 THRESHOLD	Threshold Determines how many single bit ECC errors must be detected before error status bit is set and correctable memory error is reported to PCIe Event Collector. 0 Disabled (no reporting) >0 Threshold value (1-255).

53.4.6.4.6 Correctable memory error status register (CMESR)

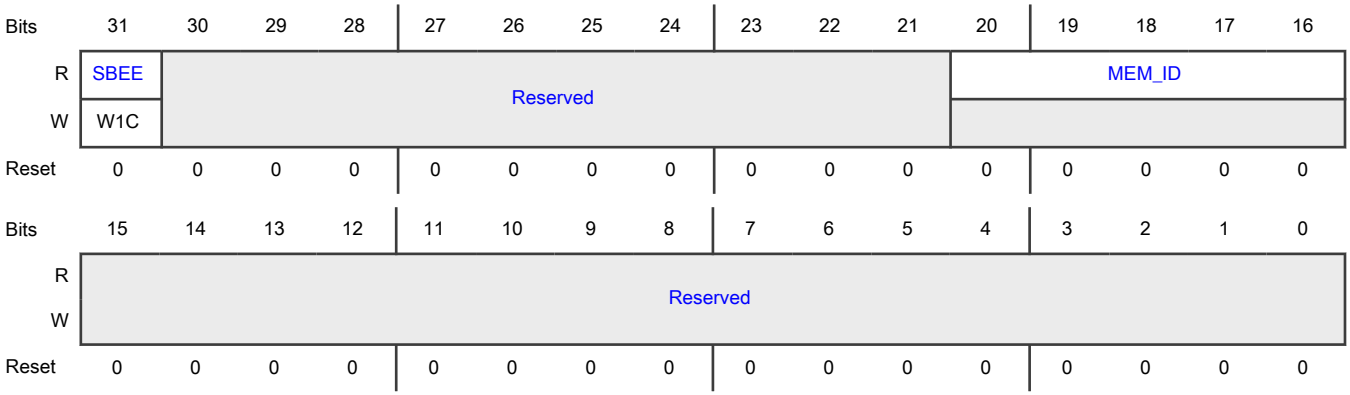
Offset

Register	Offset
CMESR	E04h

Function

This is the correctable memory error status register.

Diagram



Fields

Field	Function
31 SBEE	Single-bit ECC error When set, a threshold number of single-bit ECC error has occurred in the memory defined by MEM_ID. Write 1 to clear.
30-21 —	Reserved
20-16 MEM_ID	Memory ID See Table 392 for list of memory identifiers.
15-0 —	Reserved

53.4.6.4.7 Correctable memory error count register (CMECTR)

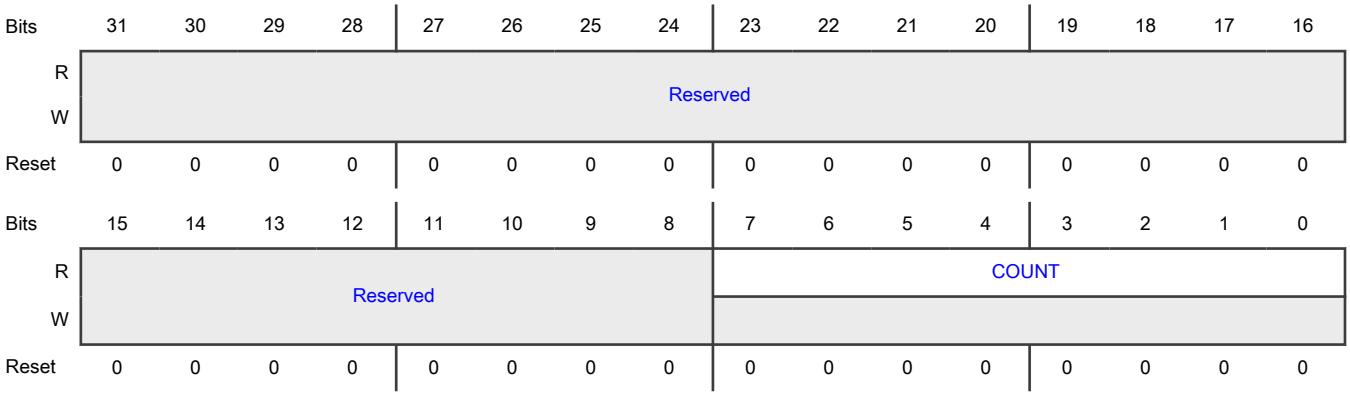
Offset

Register	Offset
CMECTR	E0Ch

Function

This is the correctable memory error count register which tracks how many events have been detected.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Count Indicates how many single-bit ECC error have been detected. After a read operation, the field's value clears to 0.

53.4.6.4.8 Uncorrectable non-fatal memory error configuration register (UNMECR)

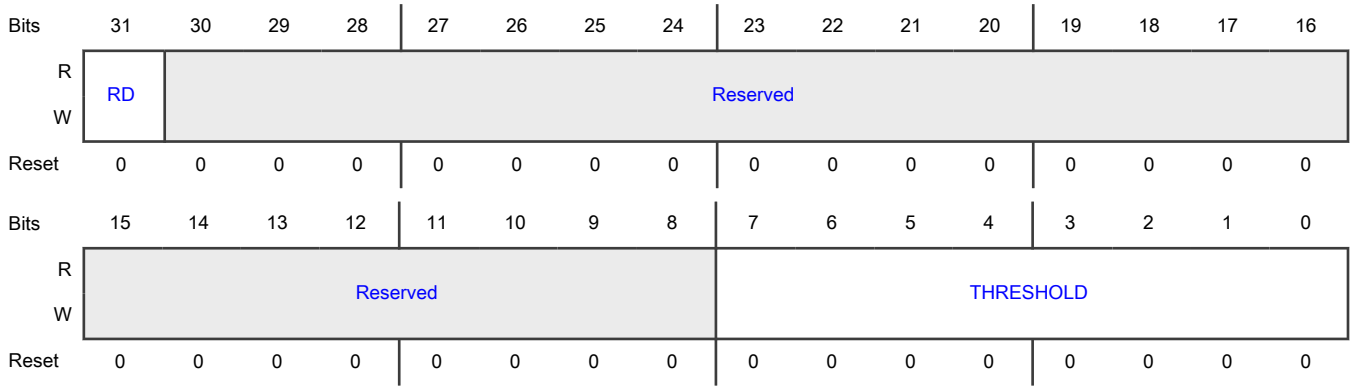
Offset

Register	Offset
UNMECR	E30h

Function

The uncorrectable non-fatal memory error configuration register can disable event reporting to all PCIe functions.

Diagram



Fields

Field	Function
31 RD	Report disable When reporting is enabled and UNMESR0[MBEE] is set, an uncorrectable error is reported to all PCIe functions' Advanced Error Reporting capability register, PCIE_CFC_AER_UCORR_ERR_STAT. 0 Enabled 1 Disabled
30-8 —	Reserved
7-0 THRESHOLD	Threshold Determines how many non-fatal memory errors must be detected before error status bit is set. 0 Disabled (no reporting) >0 Threshold value (1-255).

53.4.6.4.9 Uncorrectable non-fatal memory error status register 0 (UNMESR0)

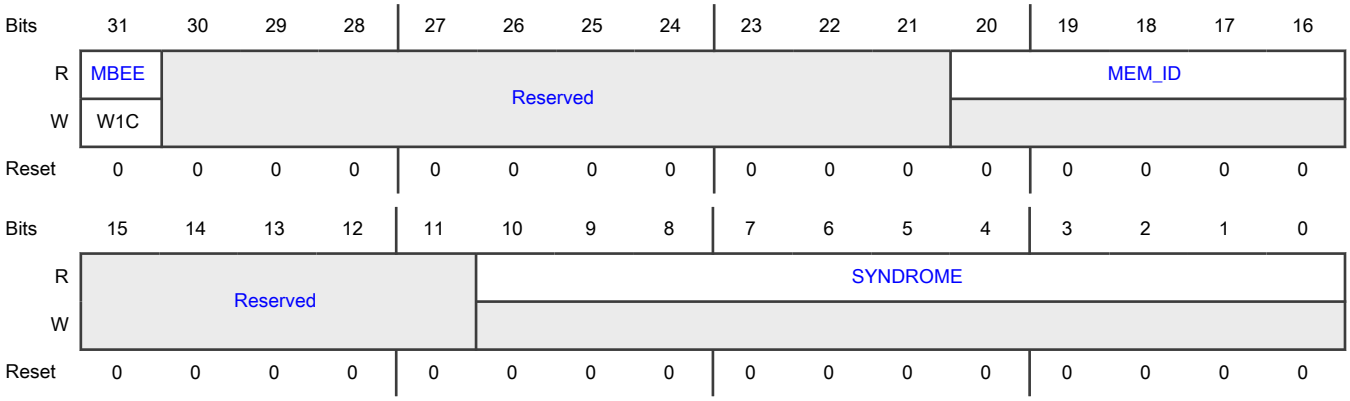
Offset

Register	Offset
UNMESR0	E34h

Function

This is the uncorrectable non-fatal memory error status register 0.

Diagram



Fields

Field	Function
31 MBEE	Multi-bit ECC error When set, a programmable threshold number of multi-bit ECC errors has occurred in internal memory as defined by MEM_ID. The reaction to the error may result in dropping the frame being processed or in stomping on the FCS of the frame if the error is detected while the transmission of the frame is in progress. Write 1 to clear.
30-21 —	Reserved
20-16 MEM_ID	Memory ID See Table 392 for list of memory identifiers.
15-11 —	Reserved
10-0 SYNDROME	Syndrome Identifies the first pertinent bit position (column) in memory that caused the error.

53.4.6.4.10 Uncorrectable non-fatal memory error status register 1 (UNMESR1)

Offset

Register	Offset
UNMESR1	E38h

Function

This is the uncorrectable non-fatal memory error status register 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ADDR	Address Address information (row) of last ECC event.

53.4.6.4.11 Uncorrectable non-fatal memory error count register (UNMECTR)

Offset

Register	Offset
UNMECTR	E3Ch

Function

This is the uncorrectable non-fatal memory error count register which tracks how many events have been detected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								COUNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Count Indicates how many frames have been dropped due to internal memory error. After a read operation, the field's value clears to 0.

53.4.6.4.12 Uncorrectable fatal memory error configuration register (UFMECR)

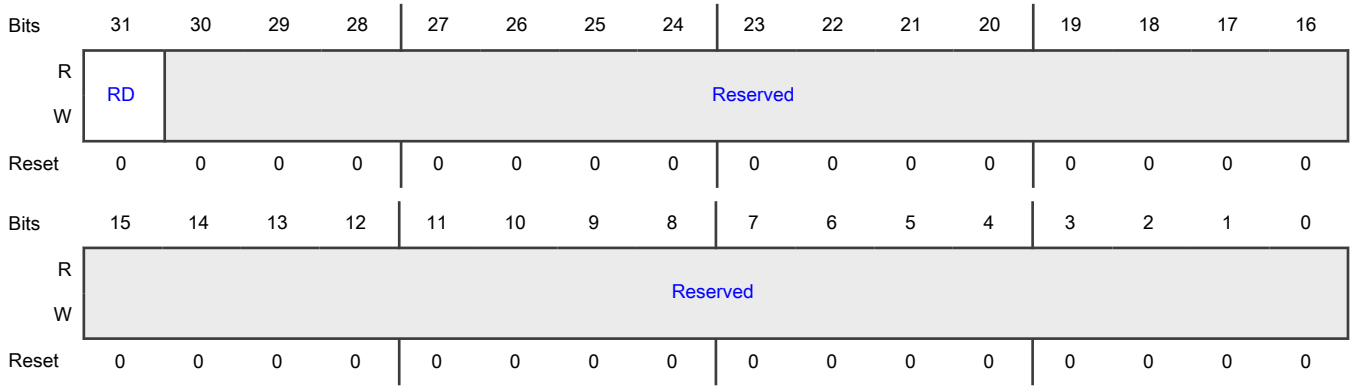
Offset

Register	Offset
UFMECR	E40h

Function

The uncorrectable fatal memory error configuration register can disable event reporting to all PCIe functions.

Diagram



Fields

Field	Function
31 RD	Report disable When reporting is enabled, and UFMESR0[MBEE] is set, an uncorrectable error is reported to all PCIe functions' Advanced Error Reporting capability register, PCIE_CFC_AER_UCORR_ERR_STAT. 0 Enabled 1 Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-0 —	Reserved

53.4.6.4.13 Uncorrectable fatal memory error status register 0 (UFMESR0)

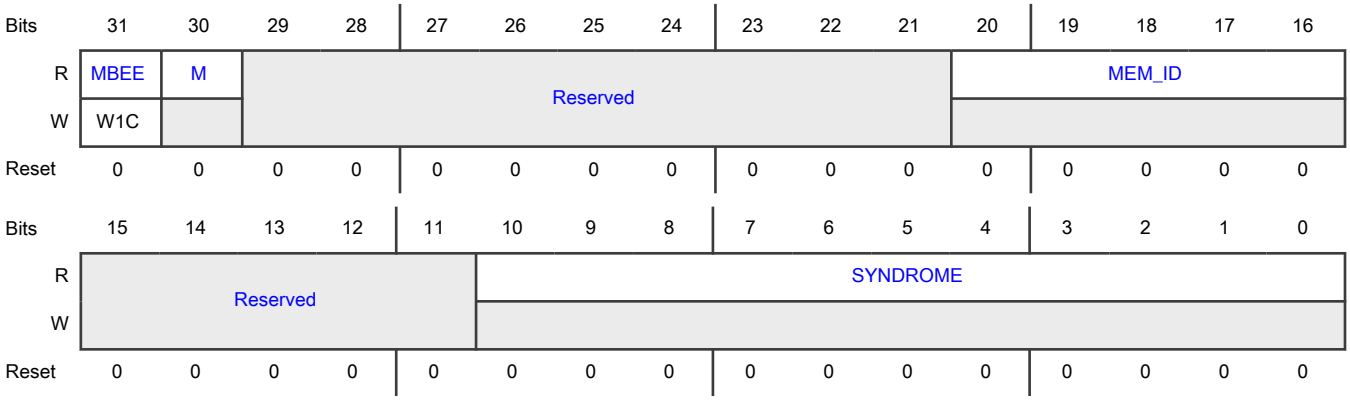
Offset

Register	Offset
UFMESR0	E44h

Function

This is the uncorrectable fatal memory error status register 0.

Diagram



Fields

Field	Function
31 MBEE	Multi-bit ECC error When set, a fatal multi-bit ECC error has occurred in the memory defined by MEM_ID. The function has entered a safe state: • PCIe bus master enable is cleared (0b0) for all functions • Station interface is disabled for all functions (SIMR[EN]=0b0) • Pseudo MAC ports are disabled (POR[TXDIS/RXDIS]=0b1) • Ethernet-MAC ports are disabled (PM*a*_COMMAND_CONFIG[RX_EN/TX_EN]=0b0) For recovery, S/W should reset NETC by issuing a Soft Reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear.
30 M	Multiple Multiple multi-bit ECC error events detected. The last event is captured.
29-21 —	Reserved
20-16 MEM_ID	Memory ID See Table 392 for list of memory identifiers.
15-11 —	Reserved
10-0 SYNDROME	Syndrome Identifies the first pertinent bit position (column) in memory that caused the error.

53.4.6.4.14 Uncorrectable fatal memory error status register 1 (UFMESR1)

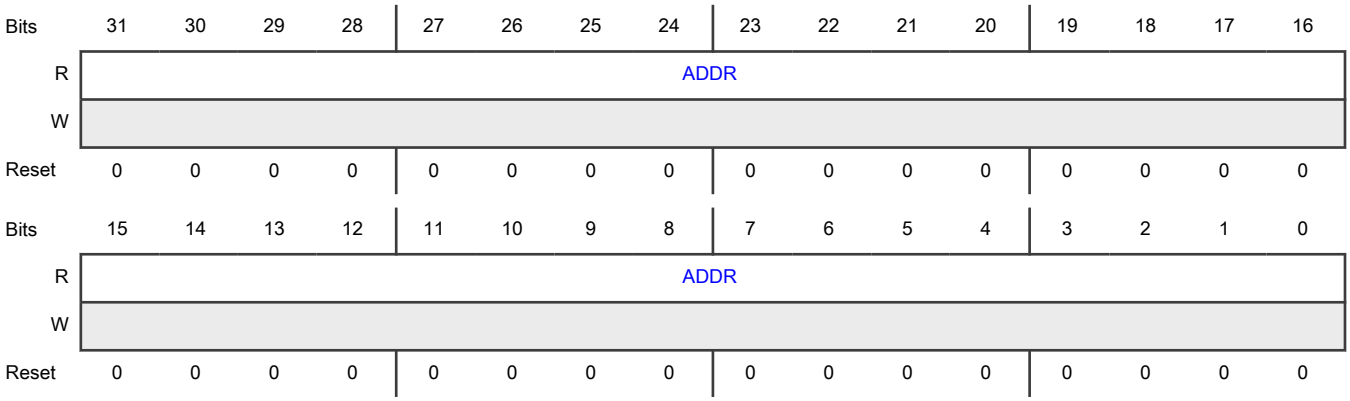
Offset

Register	Offset
UFMESR1	E48h

Function

This is the uncorrectable fatal memory error status register 1.

Diagram



Fields

Field	Function
31-0 ADDR	Address Address information (row) of last ECC event.

53.4.6.5 Switch base register descriptions

This section describes the switch specific base registers for one or more instances.

There may be differences in actual implemented registers or register fields as well as reset values between instances. The function's base address register value can be discovered from reading the PCIe Enhanced Allocation Base Address Register Equivalent Index 0 (EA BEI 0).

53.4.6.5.1 Switch memory map

SW0_BASE pcie ea bei 0 offset: 60A0_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Switch capability register 0 (SCAPR0)	32	R	0225_0005h
4h	Switch capability register 1 (SCAPR1)	32	R	0007_001Ch
8h	Buffer pool capability register (BPCAPR)	32	R	0002_0008h
18h	Forwarding capability register (FCAPR)	32	R	0000_0003h
40h	Shared memory buffer capability register (SMBCAPR)	32	R	0000_0C42h
44h	Shared memory buffer operational register 0 (SMBOR0)	32	R	0000_0000h
48h	Shared memory buffer operational register 1 (SMBOR1)	32	R	0000_0000h
80h	Command cache attribute register (CCAR)	32	RW	0202_0202h
400h	Management port configuration register (MPCR)	32	R	0000_0004h
420h	Ingress mirror destination configuration register 0 (IMDCR0)	32	RW	0000_00C0h
424h	Ingress mirror destination configuration register 1 (IMDCR1)	32	RW	0000_0000h
440h	Cut-through forwarding count register (CTFCR)	32	RW	0000_0000h
800h	Command BDR 0 mode register (CBDR0MR)	32	RW	0000_0000h
804h	Command BDR 0 status register (CBDR0SR)	32	R	0000_0000h
810h	Command BDR base address register 0 (CBDR0BAR0)	32	RW	0000_0000h
814h	Command BDR 0 base address register 1 (CBDR0BAR1)	32	RW	0000_0000h
818h	Command BDR 0 producer index register (CBDR0PIR)	32	RW	0000_0000h
81Ch	Command BDR 0 consumer index register (CBDR0CIR)	32	RW	0000_0000h
820h	Command BDR 0 length register (CBDR0LENR)	32	RW	0000_0000h
830h	Command BDR 1 mode register (CBDR1MR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
834h	Command BDR 1 status register (CBDR1SR)	32	R	0000_0000h
840h	Command BDR base address register 0 (CBDR1BAR0)	32	RW	0000_0000h
844h	Command BDR 1 base address register 1 (CBDR1BAR1)	32	RW	0000_0000h
848h	Command BDR 1 producer index register (CBDR1PIR)	32	RW	0000_0000h
84Ch	Command BDR 1 consumer index register (CBDR1CIR)	32	RW	0000_0000h
850h	Command BDR 1 length register (CBDR1LENR)	32	RW	0000_0000h
8A0h	Command BDR 0 interrupt enable register (CBDR0IER)	32	RW	0000_0000h
8A4h	Command BDR 0 interrupt detect register (CBDR0IDR)	32	RW	0000_0000h
8A8h	Command BDR 0 MSI-X vector register (CBDR0MSIVR)	32	RW	0000_0000h
8B0h	Command BDR 1 interrupt enable register (CBDR1IER)	32	RW	0000_0000h
8B4h	Command BDR 1 interrupt detect register (CBDR1IDR)	32	RW	0000_0000h
8B8h	Command BDR 1 MSI-X vector register (CBDR1MSIVR)	32	RW	0000_0000h
900h	QoS to VLAN mapping profile 0 register 0 (QOSVLANMP0R0)	32	RW	0000_0000h
904h	QoS to VLAN mapping profile 0 register 1 (QOSVLANMP0R1)	32	RW	0000_0000h
908h	QoS to VLAN mapping profile 0 register 2 (QOSVLANMP0R2)	32	RW	0000_0000h
90Ch	QoS to VLAN mapping profile 0 register 3 (QOSVLANMP0R3)	32	RW	0000_0000h
920h	QoS to VLAN mapping profile 1 register 0 (QOSVLANMP1R0)	32	RW	0000_0000h
924h	QoS to VLAN mapping profile 1 register 1 (QOSVLANMP1R1)	32	RW	0000_0000h
928h	QoS to VLAN mapping profile 1 register 2 (QOSVLANMP1R2)	32	RW	0000_0000h
92Ch	QoS to VLAN mapping profile 1 register 3 (QOSVLANMP1R3)	32	RW	0000_0000h
B00h - B04h	PCP to PCP mapping profile a register (PCP2PCPMP0R - PCP2PCPMP1R)	32	RW	0000_0000h
2000h	Bridge capability register (BRCAPR)	32	R	0000_103Dh
2008h	VLAN filter hash table capability register (VFHTCAPR)	32	R	00B0_0000h
200Ch	VLAN filter hash table operational register (VFHTOR)	32	R	0000_0000h
2010h	VLAN Filter (hash) table default entry configuration registers 0 (VFHTDECR0)	32	RW	0000_0000h
2014h	VLAN filter hash table default entry configuration registers 1 (VFHTDECR1)	32	RW	FFFF_0000h
2018h	VLAN filter hash table default entry configuration registers 2 (VFHTDECR2)	32	RW	1200_0000h
2020h	FDB hash table capability register (FDBHTCAPR)	32	R	00B0_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2024h	FDB hash table memory configuration register (FDBHTMCR)	32	RW	0000_0000h
2028h	FDB hash table operational register 0 (FDBHTOR0)	32	R	0000_0000h
202Ch	FDB hash table operational register 1 (FDBHTOR1)	32	R	0000_0000h
2040h	IP multicast filter hash table capability register (IPMFHTCAPR)	32	R	0070_0000h
2044h	IPv4 multicast filter hash table operation register (IPv4MFHTOR)	32	R	0000_0000h

53.4.6.5.2 Switch capability register 0 (SCAPR0)

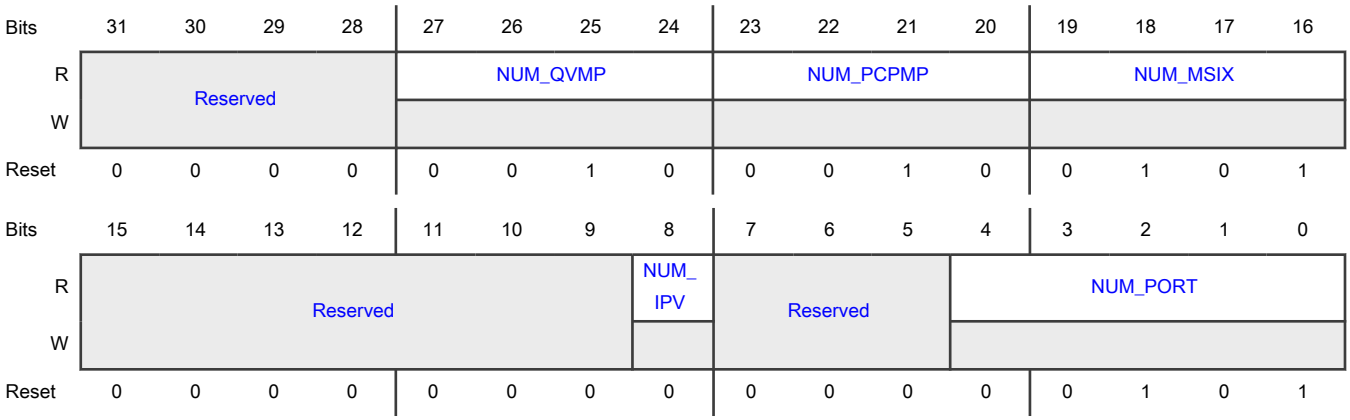
Offset

Register	Offset
SCAPR0	0h

Function

This is the switch capability register 0.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24	Number of QoS to VLAN PCP mapping profiles supported. Defines range of *a* in QOSVLANMPaR*a*R.

Table continues on the next page...

Table continued from the previous page...

Field	Function
NUM_QVMP	**NOTE** Value of 0 implies no Mapping Profiles supported.
23-20 NUM_PCPMP	Number of VLAN PCP to PCP mapping profiles supported. Define range of *a* in PCP2PCPMP*a*R; where *a* = {0..NUM_PCPMP-1}. **NOTE** Value of 0 implies no mapping profiles supported.
19-16 NUM_MSIX	Number of MSI-X vectors supported by switch function. Range: 1..16 Formula: NUM_MSIX+1 **NOTE** Value derived from IERB's S*a*MCR register.
15-9 —	Reserved
8 NUM_IPV	Number of IPVs supported 0: 8 IPV 1: Reserved
7-5 —	Reserved
4-0 NUM_PORT	Number of ports supported. **NOTE** Value derived from IERB's L*a*BCR registers. where a corresponds to the link number.

53.4.6.5.3 Switch capability register 1 (SCAPR1)

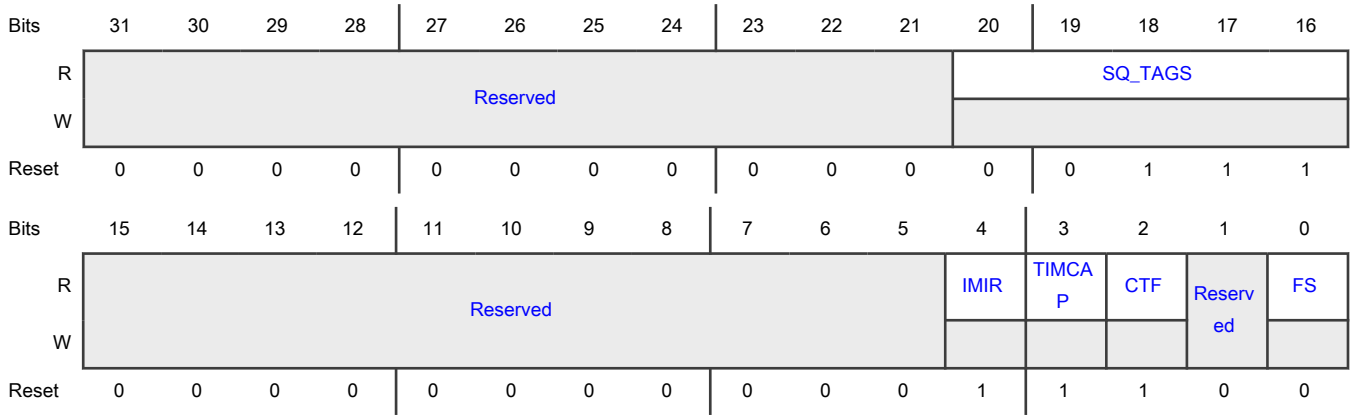
Offset

Register	Offset
SCAPR1	4h

Function

This is the switch capability register 1.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 SQ_TAGS	Indicates which FRER Sequence Tags are supported xxxx1: 802.1CB draft 2.0 R-TAG xxx1x: 802.1CB R-TAG xx1xx: HSR Tag All other values reserved.
15-5 —	Reserved
4 IMIR	Ingress mirroring functionality supported. 0: Not supported 1: Supported
3 TIMCAP	Time capture capability supported. 0: Not supported 1: Supported
2 CTF	Cut-through forwarding supported. 0: Not supported 1: Supported
1 —	Reserved
0	Functional safety capability supported.

Table continues on the next page...

Table continued from the previous page...

Field	Function
FS	If set, additional integrity error checks are performed; this includes, but not limited to, internal FCS and parity error checks. 0: Not supported 1: Supported

53.4.6.5.4 Buffer pool capability register (BPCAPR)

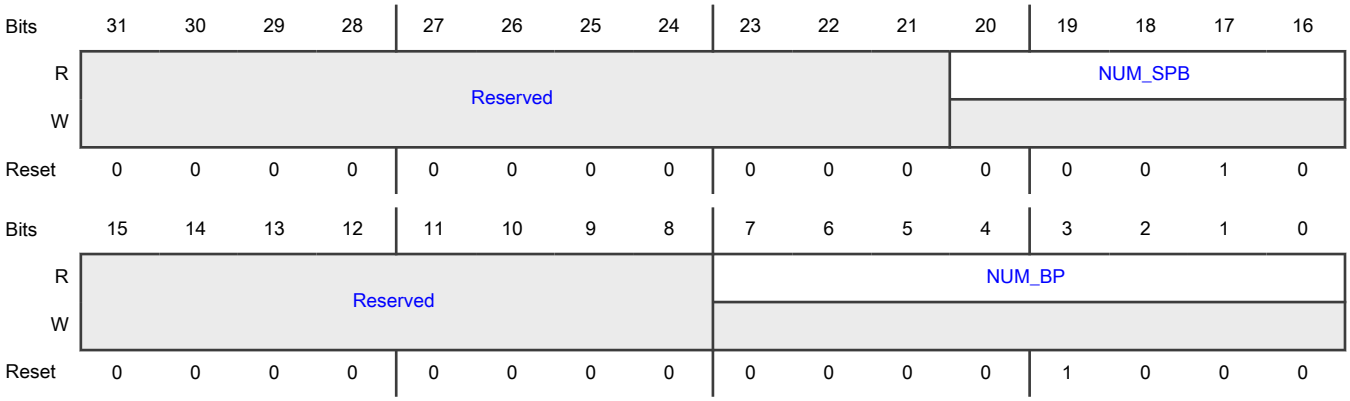
Offset

Register	Offset
BPCAPR	8h

Function

This is the buffer pool capability register

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 NUM_SPB	Number of shared buffer pools supported.
15-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 NUM_BP	Number of buffer pools supported.

53.4.6.5.5 Forwarding capability register (FCAPR)

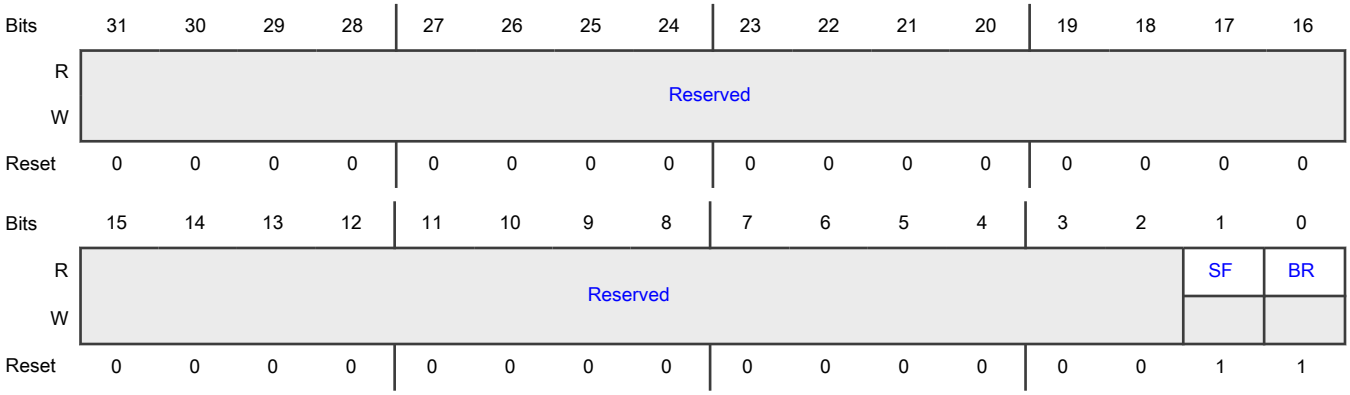
Offset

Register	Offset
FCAPR	18h

Function

This is the forwarding capability register.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 SF	Stream forwarding supported Stream forwarding provides the ability to identify a stream and forward it to a set of ports with ability to enable source port pruning. 0b - Not supported 1b - Supported
0	802.1Q bridge forwarding support.

Table continues on the next page...

Table continued from the previous page...

Field	Function
BR	See BRCAPR for more information. 0b - Not supported 1b - Supported

53.4.6.5.6 Shared memory buffer capability register (SMBCAPR)

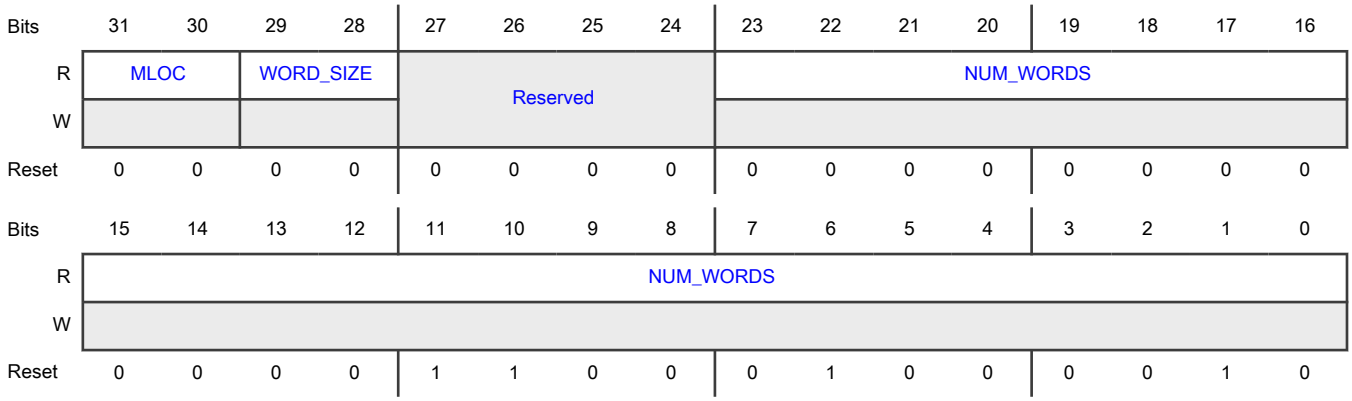
Offset

Register	Offset
SMBCAPR	40h

Function

This is the shared memory buffer capability register.

Diagram



Fields

Field	Function
31-30 MLOC	Indicates memory location 00b - Common memory 01b-11b - Reserved
29-28 WORD_SIZE	Word size in bytes. 00b - 24 bytes 01b-11b - Reserved
27-24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23-0 NUM_WORDS	Number of words available for the switch frame buffering. This value is writable in IERB register S*a*SMBAR (by pre-boot initialization), and is immediately reflected here. **NOTE** Memory is managed via buffer pool mechanism.

53.4.6.5.7 Shared memory buffer operational register 0 (SMBOR0)

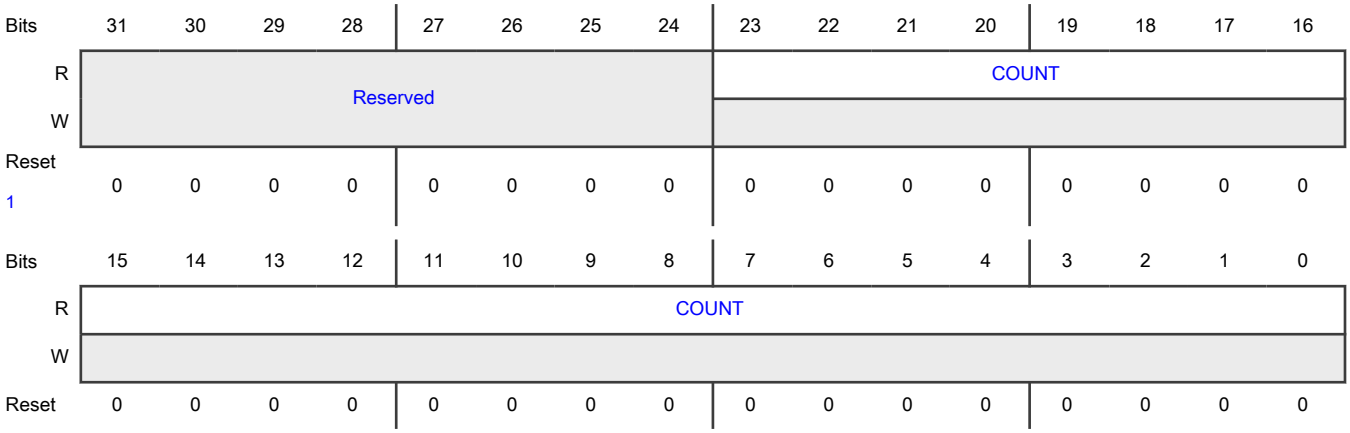
Offset

Register	Offset
SMBOR0	44h

Function

This is the shared memory buffer operational register 0.

Diagram



1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-0 COUNT	Number of words in use for frame buffering.

53.4.6.5.8 Shared memory buffer operational register 1 (SMBOR1)

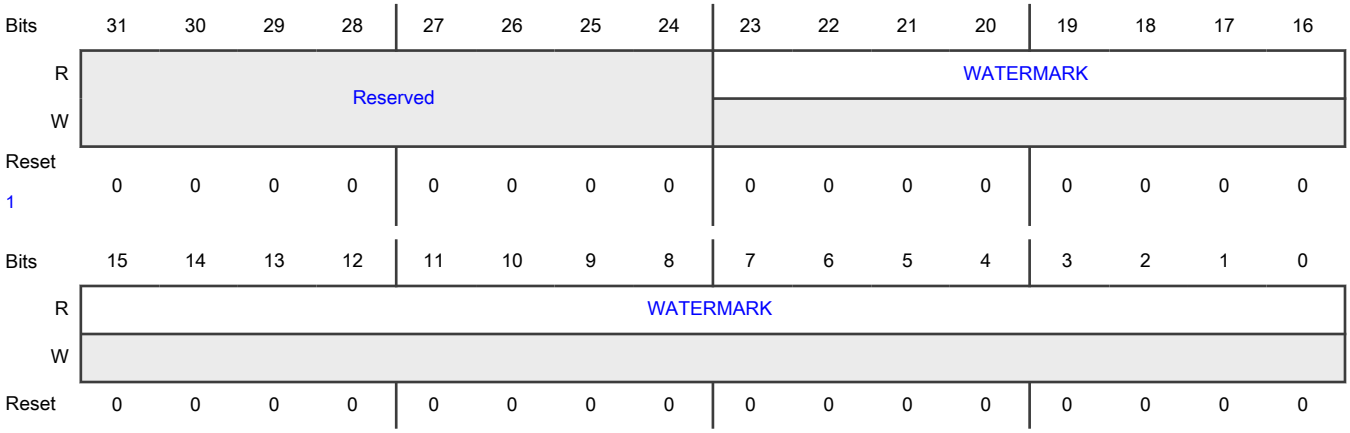
Offset

Register	Offset
SMBOR1	48h

Function

This is the shared memory buffer operational register 1.

Diagram



1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-24 —	Reserved
23-0 WATERMARK	High watermark of words in use for frame buffering since the last read. **NOTE** Value reset to SMBOR0[COUNT] after a read.

53.4.6.5.9 Command cache attribute register (CCAR)

Offset

Register	Offset
CCAR	80h

Function

This is the command cache attribute register. It is used to determine system attributes for reads and writes of command descriptor and data.

NOTE

The switch will sample the IERB value of this register when PCIe Memory Access bit is set (b1). All updates after this must be done through this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	CRDSNP	CRDDOMAIN		CRDCACHE				Reserved	CBD_RDS...	CBD_RDDOMAIN		CBD_RDCACHE			
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	CWRSNP	CWRDOMAIN		CWRCACHE				Reserved	CBD_WRS...	CBD_WRDOMAIN		CBD_WRCACHE			
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31 —	Reserved
30 CRDSNP	Read data snoop This is the snoop attribute setting used when switch reads command data from host memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
29-28 CRDDOMAIN	Read data domain This is the domain attribute setting used when switch reads command data from host memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
27-24	Read data cache type

Table continues on the next page...

Table continued from the previous page...

Field	Function
CRDCACHE	This is the cache attribute setting used when switch reads command data from host memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
23 —	Reserved
22 CBD_RDSNP	Command descriptor read snoop See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
21-20 CBD_RDDOMAIN	Command buffer descriptor read domain This is the domain attribute setting used when switch reads the command buffer descriptor from host memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
19-16 CBD_RDCACHE	Command buffer descriptor read cache type This is the cache attribute setting used when switch reads the command buffer descriptor from host memory. See System Interface Read Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
15 —	Reserved
14 CWRSNP	Write data snoop This is the snoop attribute setting used when switch writes command data to host memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
13-12 CWRDOMAIN	Write data domain This is the domain attribute setting used when switch writes command data to host memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
11-8 CWRCACHE	Write data cache type This is the cache attribute setting used when switch writes command data to host memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 CBD_WRSNP	Command buffer descriptor write snoop This is the snoop attribute setting used when switch writes the command buffer descriptor in host memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
5-4 CBD_WRDOMAIN	Command buffer descriptor write domain This is the domain attribute setting used when switch writes the command buffer descriptor in host memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.
3-0 CBD_WRCACHE	Command buffer descriptor write cache type This is the cache attribute setting used when switch writes the command buffer descriptor in host memory. See System Interface Write Transaction Attribute Definitions table in Station interface buffer cache attribute register (SIBCAR) for valid settings.

53.4.6.5.10 Management port configuration register (MPCR)

Offset

Register	Offset
MPCR	400h

Function

This is the management port configuration register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												PORT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-5 —	Reserved
4-0 PORT	Switch Management Port Specifies the switch port number designated as a switch management port. Reset value is determined by it's equivalent IERB Register (S0MPCR)

53.4.6.5.11 Ingress mirror destination configuration register 0 (IMDCR0)

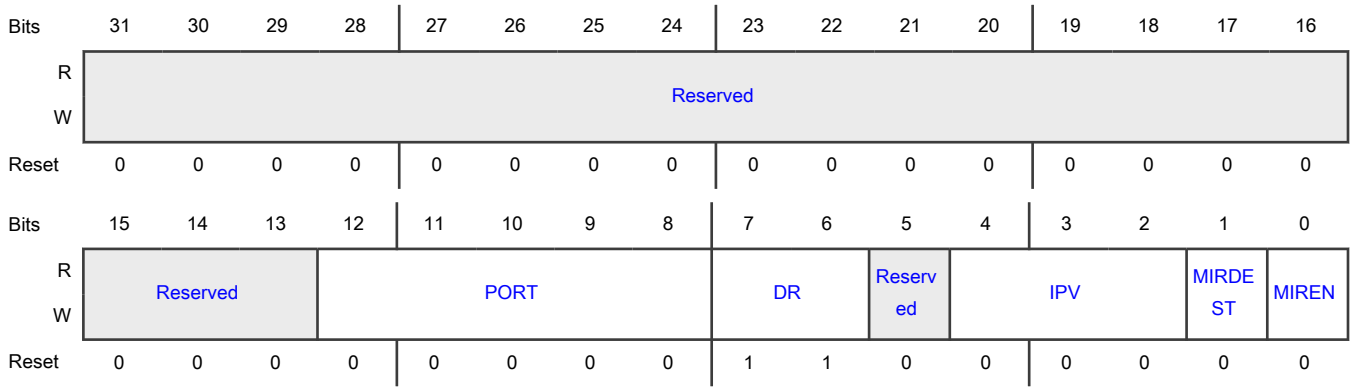
Offset

Register	Offset
IMDCR0	420h

Function

This is the ingress mirror destination configuration register 0.

Diagram



Fields

Field	Function
31-13 —	Reserved
12-8 PORT	Port where ingress mirrored frames are sent. Valid if MIRDEST=0.
7-6	Mirrored packet's DR (drop resilience).

Table continues on the next page...

Table continued from the previous page...

Field	Function
DR	
5 —	Reserved
4-2 IPV	Mirrored packet's IPV.
1 MIRDEST	Indicates the mirror destination 0b - Port as specified by PORT field 1b - Switch management port
0 MIREN	Mirror enable. 0b - Ingress mirroring disabled 1b - Ingress mirroring enabled

53.4.6.5.12 Ingress mirror destination configuration register 1 (IMDCR1)

Offset

Register	Offset
IMDCR1	424h

Function

This is the ingress mirror destination configuration register 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								EFM_LEN_CHANGE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EFMEID															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-23 —	Reserved
22-16 EFM_LEN_CHANGE	Egress Frame Modification Frame Length change in 2s complement notation. This field specifies the frame length change that represents the effective frame length change of the egress frame modification actions configured in the EFMEID field of this register. This frame length change value is used to adjust the frame length metadata, which in turn is passed to the egress scheduling and management functions to represent the actual length of the frame on the transmission link. Specifying explicitly the frame length change avoids having the hardware to read the egress frame modifications entry and compute the frame length change itself. Frame Length change is specified in unit of bytes using a 2's complement notation. Supported range: -10 bytes (removing one VLAN tag and one R-TAG/HSR) to +6 bytes (adding one VLAN tag and replacing a 4-byte 802.1CB draft 2.0 R-TAG with 6-byte 802.1CB R-TAG or HSR tag). Valid if EFMEID is not null (i.e: not 0xFFFF).
15-0 EFMEID	Egress Frame Modification Entry Id Only applicable if MIRDEST=0. Note that EFMEID[L2_ACT]=1b and EFMEID[PLD_ACT]=001b are not supported **NOTE** 0xFFF is a Null Frame Modification Entry.

53.4.6.5.13 Cut-through forwarding count register (CTFCR)

Offset

Register	Offset
CTFCR	440h

Function

This is the cut-through forwarding count register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COUNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COUNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 COUNT	Count the number of cut-through frames which are forwarded to at least 1 egress port.

53.4.6.5.14 Command BDR a mode register (CBDR0MR - CBDR1MR)

Offset

Register	Offset
CBDR0MR	800h
CBDR1MR	830h

Function

This is the mode register for command buffer descriptor ring.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EN	Reserved														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EN	Enable command buffer descriptor ring 0b - Disabled 1b - Enabled. When the ring is non-empty, command buffer descriptors will be processed
30-0 —	Reserved

53.4.6.5.15 Command BDR a status register (CBDR0SR - CBDR1SR)

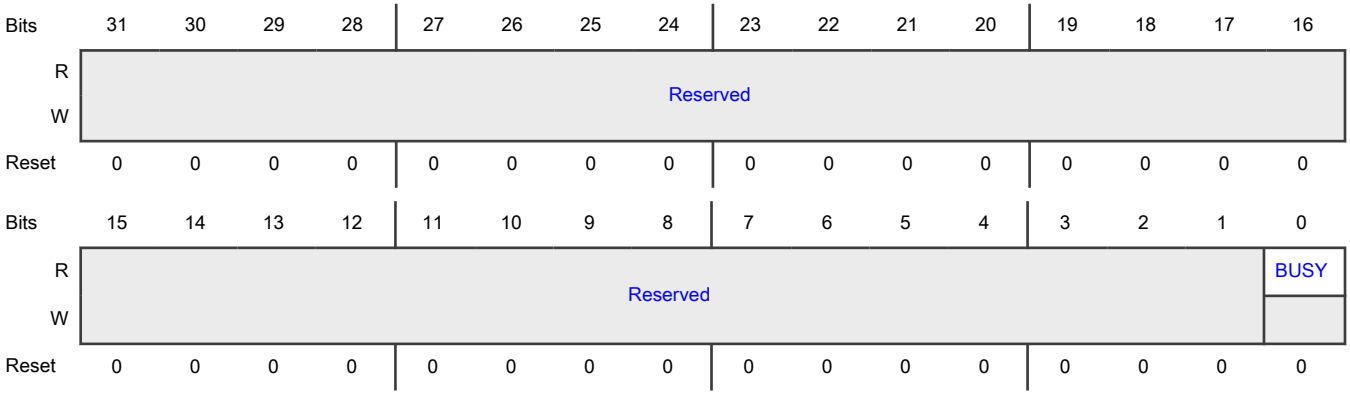
Offset

Register	Offset
CBDR0SR	804h
CBDR1SR	834h

Function

This is the command buffer descriptor ring status register.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 BUSY	Busy. This field indicates whether the hardware is busy processing commands from the command BD ring.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Idle 1b - Busy

53.4.6.5.16 Command BDR base address register 0 (CBDR0BAR0 - CBDR1BAR0)

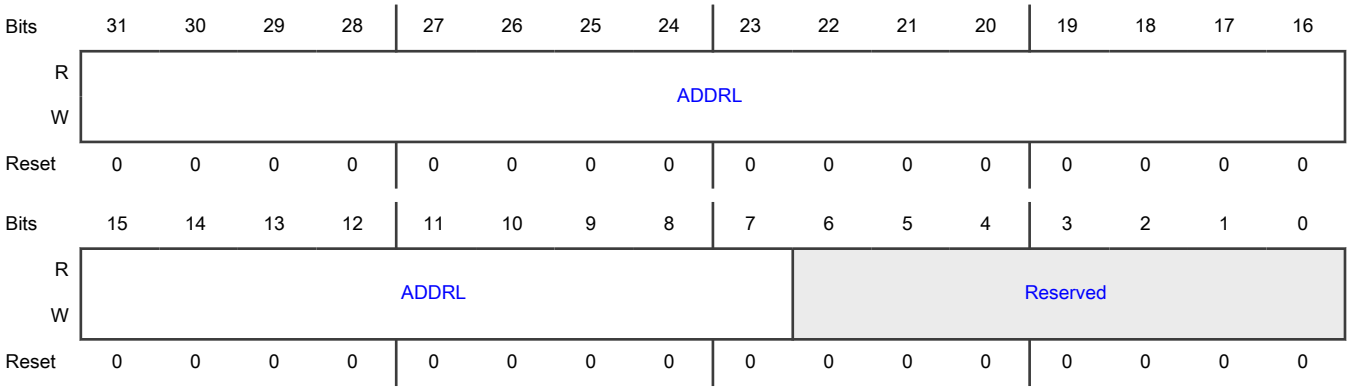
Offset

Register	Offset
CBDR0BAR0	810h
CBDR1BAR0	840h

Function

This is the lower address register for the base memory address location of the command BD ring. The address must be 128 byte aligned.

Diagram



Fields

Field	Function
31-7 ADDRL	Lower address bits 31-7.
6-0 —	Reserved

53.4.6.5.17 Command BDR a base address register 1 (CBDR0BAR1 - CBDR1BAR1)

Offset

Register	Offset
CBDR0BAR1	814h
CBDR1BAR1	844h

Function

This is the upper address register for the base memory address location of the command BD ring.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ADDRH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDRH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ADDRH	Upper address bits 63-32.

53.4.6.5.18 Command BDR a producer index register (CBDR0PIR - CBDR1PIR)

Offset

Register	Offset
CBDR0PIR	818h
CBDR1PIR	848h

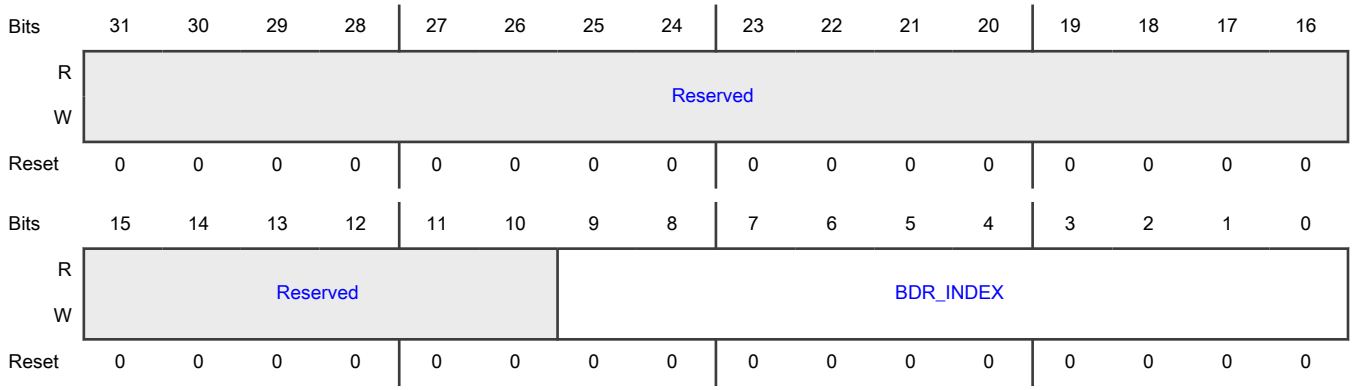
Function

This is the command BDR producer index register. When the BD ring is enabled the value of this register and the consumer index register, CBDR*a*CIR, determines switch's unprocessed BDs of the ring. This value also determines how many BD's are committed to the BDR by software. Software should initialize this register while the ring is disabled to reflect an empty state or a pre-loaded BDR state.

To add one or more buffer descriptors to an enabled command ring, software should first check to make sure the ring is not full by reading CBDR*a*CIR. Note that maximum number of BDs allowed on a ring is one less than the ring size. A memory barrier must be used to guarantee update of the BD(s) to memory before writing the producer index register.

There is no wrap bit to indicate full/empty. One entry is used to differentiate between full and empty. When producer index + 1 equals consumer index, the ring is considered full. When the producer/consumer indices match, the ring is always considered empty.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 BDR_INDEX	Command buffer descriptor ring producer index. Range of index depends on ring size CBDR*a*LENR[LENGTH].

53.4.6.5.19 Command BDR a consumer index register (CBDR0CIR - CBDR1CIR)

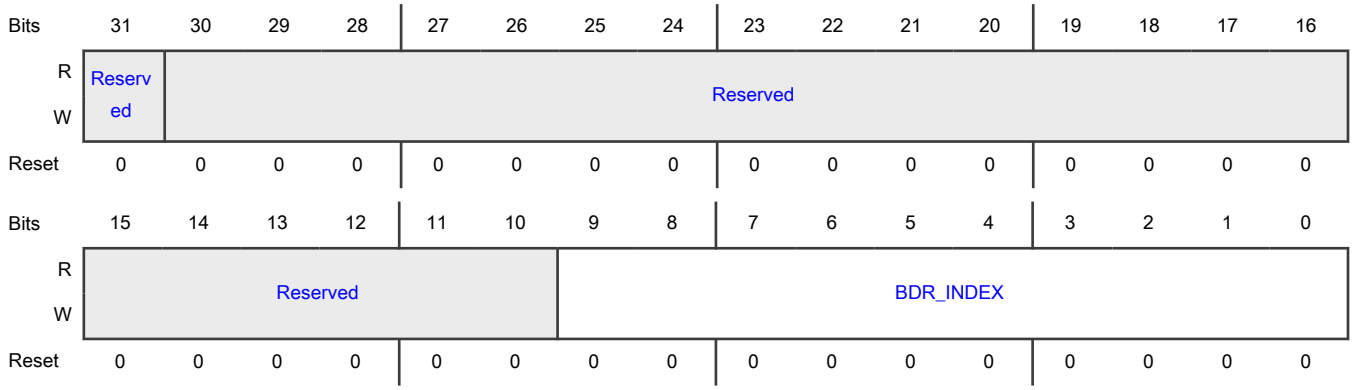
Offset

Register	Offset
CBDR0CIR	81Ch
CBDR1CIR	84Ch

Function

This is the command BDR consumer index register. It reflects the next BD to be processed by NETC and trails the CBDR*a*PIR index value. Software should only update/reset this value when the BD ring is re-initialized. Disabling the BD ring has no effect on this register. Software can calculate available entries in the ring by using the software producer index and this register value.

Diagram



Fields

Field	Function
31 —	Reserved
30-10 —	Reserved
9-0 BDR_INDEX	Command buffer descriptor ring consumer index. Range of index depends on ring size CBDR*s*LENR[LENGTH].

53.4.6.5.20 Command BDR a length register (CBDR0LENR - CBDR1LENR)

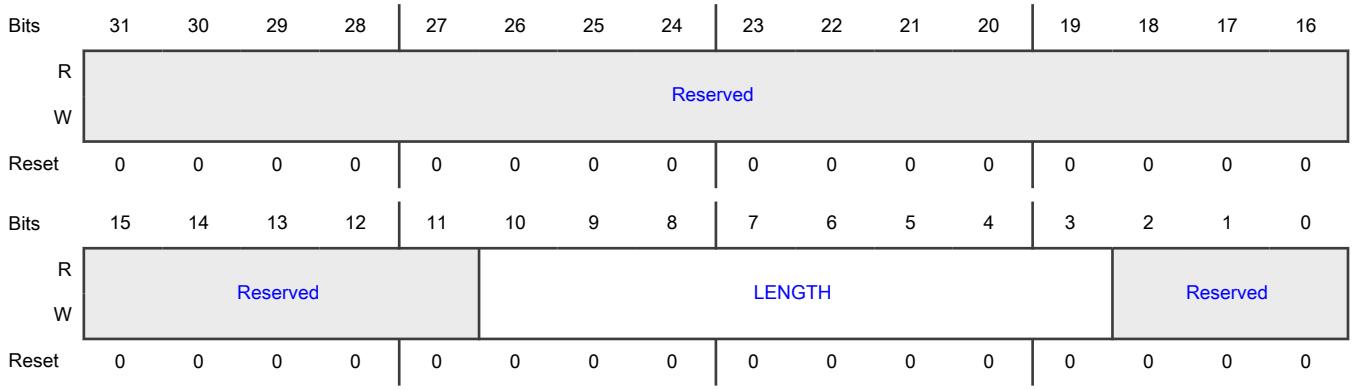
Offset

Register	Offset
CBDR0LENR	820h
CBDR1LENR	850h

Function

This is the command BDR length register. It determines the size of the transmit ring in sets of 8 BDs.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-3 LENGTH	BD ring length Size of ring in sets of 8 BDs. Maximum ring size is 1K.
2-0 —	Reserved

53.4.6.5.21 Command BDR a interrupt enable register (CBDR0IER - CBDR1IER)

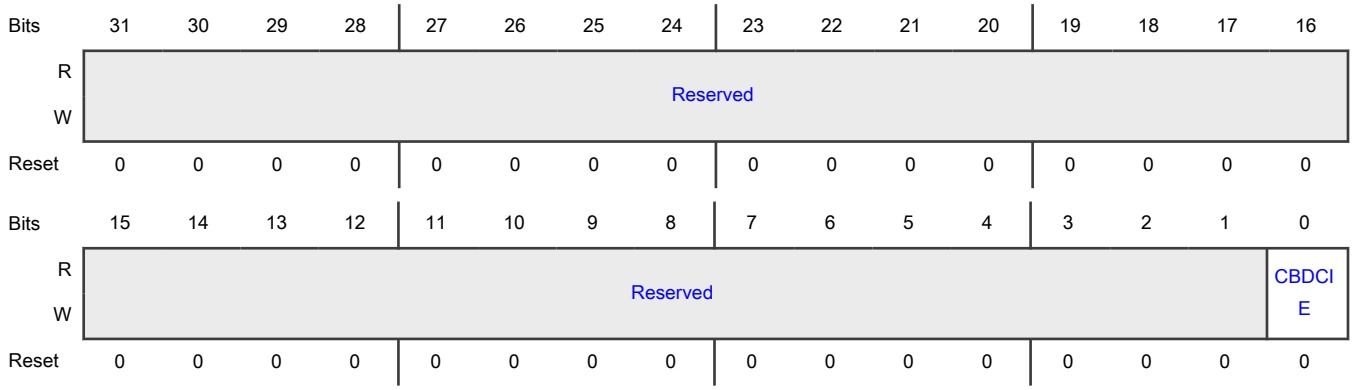
Offset

Register	Offset
CBDR0IER	8A0h
CBDR1IER	8B0h

Function

This is the interrupt enable register for the command BD ring. If the enable bit and the corresponding detect bit in CBDR*a*IDR is set, interrupt will be triggered for the ring.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CBDCIE	Command BD completion interrupt enable 0 Disabled 1 Enabled

53.4.6.5.22 Command BDR a interrupt detect register (CBDR0IDR - CBDR1IDR)

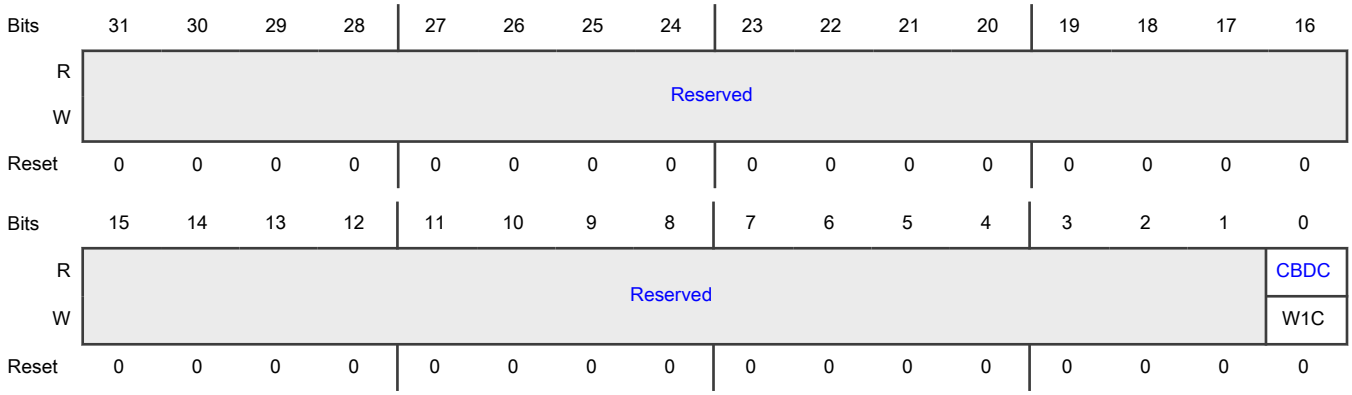
Offset

Register	Offset
CBDR0IDR	8A4h
CBDR1IDR	8B4h

Function

This is the interrupt detect register for the command buffer descriptor ring. Write 1 to clear.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CBDC	Command BD completed 0b - No BD with CI=1 completed 1b - Processed BD with CI=1

53.4.6.5.23 Command BDR a MSI-X vector register (CBDR0MSIVR - CBDR1MSIVR)

Offset

Register	Offset
CBDR0MSIVR	8A8h
CBDR1MSIVR	8B8h

Function

This is the command BDR MSI-X vector register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												VECTOR			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 VECTOR	Index into MSI-X address/data table. Range: 0..SCAPR0[NUM_MSIX]-1 **NOTE** Specifying a vector number outside of the range will not generate an MSI-X write.

53.4.6.5.24 QoS to VLAN mapping profile 0 register 0 (QOSVLANMP0R0)

Offset

Register	Offset
QOSVLANMP0R0	900h

Function

This is the port QoS to VLAN mapping profile register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IPV1_DR3				IPV1_DR2				IPV1_DR1				IPV1_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IPV0_DR3				IPV0_DR2				IPV0_DR1				IPV0_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 IPV1_DR3	The IPV1DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
27-24 IPV1_DR2	The IPV1DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
23-20 IPV1_DR1	The IPV1DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
19-16 IPV1_DR0	The IPV1DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
15-12 IPV0_DR3	The IPV0DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
11-8 IPV0_DR2	The IPV0DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
7-4 IPV0_DR1	The IPV0DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 IPV0_DR0	The IPV0DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

53.4.6.5.25 QoS to VLAN mapping profile 0 register 1 (QOSVLANMP0R1)

Offset

Register	Offset
QOSVLANMP0R1	904h

Function

This is the port QoS to VLAN mapping profile register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IPV3_DR3				IPV3_DR2				IPV3_DR1				IPV3_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IPV2_DR3				IPV2_DR2				IPV2_DR1				IPV2_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 IPV3_DR3	The IPV3DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
27-24 IPV3_DR2	The IPV3DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
23-20	The IPV3DR1 field is defined as a 4-bit field as follows:

Table continues on the next page...

Table continued from the previous page...

Field	Function
IPV3_DR1	Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
19-16 IPV3_DR0	The IPV3DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
15-12 IPV2_DR3	The IPV2DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
11-8 IPV2_DR2	The IPV2DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
7-4 IPV2_DR1	The IPV2DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
3-0 IPV2_DR0	The IPV2DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

53.4.6.5.26 QoS to VLAN mapping profile 0 register 2 (QOSVLANMP0R2)

Offset

Register	Offset
QOSVLANMP0R2	908h

Function

This is the port QoS to VLAN mapping profile register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IPV5_DR3				IPV5_DR2				IPV5_DR1				IPV5_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IPV4_DR3				IPV4_DR2				IPV4_DR1				IPV4_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 IPV5_DR3	The IPV5DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
27-24 IPV5_DR2	The IPV5DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
23-20 IPV5_DR1	The IPV5DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
19-16 IPV5_DR0	The IPV5DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
15-12 IPV4_DR3	The IPV4DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
11-8 IPV4_DR2	The IPV4DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
7-4 IPV4_DR1	The IPV4DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 IPV4_DR0	The IPV4DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

53.4.6.5.27 QoS to VLAN mapping profile 0 register 3 (QOSVLANMP0R3)

Offset

Register	Offset
QOSVLANMP0R3	90Ch

Function

This is the port QoS to VLAN mapping profile register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IPV7_DR3				IPV7_DR2				IPV7_DR1				IPV7_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IPV6_DR3				IPV6_DR2				IPV6_DR1				IPV6_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 IPV7_DR3	The IPV7DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
27-24 IPV7_DR2	The IPV7DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
23-20	The IPV7DR1 field is defined as a 4-bit field as follows:

Table continues on the next page...

Table continued from the previous page...

Field	Function
IPV7_DR1	Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
19-16 IPV7_DR0	The IPV7DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
15-12 IPV6_DR3	The IPV6DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
11-8 IPV6_DR2	The IPV6DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
7-4 IPV6_DR1	The IPV6DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
3-0 IPV6_DR0	The IPV6DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

53.4.6.5.28 QoS to VLAN mapping profile 1 register 0 (QOSVLANMP1R0)

Offset

Register	Offset
QOSVLANMP1R0	920h

Function

This is the port QoS to VLAN mapping profile register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IPV1_DR3				IPV1_DR2				IPV1_DR1				IPV1_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IPV0_DR3				IPV0_DR2				IPV0_DR1				IPV0_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 IPV1_DR3	The IPV1DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
27-24 IPV1_DR2	The IPV1DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
23-20 IPV1_DR1	The IPV1DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
19-16 IPV1_DR0	The IPV1DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
15-12 IPV0_DR3	The IPV0DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
11-8 IPV0_DR2	The IPV0DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
7-4 IPV0_DR1	The IPV0DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 IPV0_DR0	The IPV0DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

53.4.6.5.29 QoS to VLAN mapping profile 1 register 1 (QOSVLANMP1R1)

Offset

Register	Offset
QOSVLANMP1R1	924h

Function

This is the port QoS to VLAN mapping profile register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IPV3_DR3				IPV3_DR2				IPV3_DR1				IPV3_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IPV2_DR3				IPV2_DR2				IPV2_DR1				IPV2_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 IPV3_DR3	The IPV3DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
27-24 IPV3_DR2	The IPV3DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
23-20	The IPV3DR1 field is defined as a 4-bit field as follows:

Table continues on the next page...

Table continued from the previous page...

Field	Function
IPV3_DR1	Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
19-16 IPV3_DR0	The IPV3DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
15-12 IPV2_DR3	The IPV2DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
11-8 IPV2_DR2	The IPV2DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
7-4 IPV2_DR1	The IPV2DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
3-0 IPV2_DR0	The IPV2DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

53.4.6.5.30 QoS to VLAN mapping profile 1 register 2 (QOSVLANMP1R2)

Offset

Register	Offset
QOSVLANMP1R2	928h

Function

This is the port QoS to VLAN mapping profile register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IPV5_DR3				IPV5_DR2				IPV5_DR1				IPV5_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IPV4_DR3				IPV4_DR2				IPV4_DR1				IPV4_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 IPV5_DR3	The IPV5DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
27-24 IPV5_DR2	The IPV5DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
23-20 IPV5_DR1	The IPV5DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
19-16 IPV5_DR0	The IPV5DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
15-12 IPV4_DR3	The IPV4DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
11-8 IPV4_DR2	The IPV4DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
7-4 IPV4_DR1	The IPV4DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 IPV4_DR0	The IPV4DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

53.4.6.5.31 QoS to VLAN mapping profile 1 register 3 (QOSVLANMP1R3)

Offset

Register	Offset
QOSVLANMP1R3	92Ch

Function

This is the port QoS to VLAN mapping profile register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IPV7_DR3				IPV7_DR2				IPV7_DR1				IPV7_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IPV6_DR3				IPV6_DR2				IPV6_DR1				IPV6_DR0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 IPV7_DR3	The IPV7DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
27-24 IPV7_DR2	The IPV7DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
23-20	The IPV7DR1 field is defined as a 4-bit field as follows:

Table continues on the next page...

Table continued from the previous page...

Field	Function
IPV7_DR1	Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
19-16 IPV7_DR0	The IPV7DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
15-12 IPV6_DR3	The IPV6DR3 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
11-8 IPV6_DR2	The IPV6DR2 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
7-4 IPV6_DR1	The IPV6DR1 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)
3-0 IPV6_DR0	The IPV6DR0 field is defined as a 4-bit field as follows: Bit 3 - Reserved Bit 2:0 - Priority Code Point (PCP)

53.4.6.5.32 PCP to PCP mapping profile a register (PCP2PCPMP0R - PCP2PCPMP1R)

Offset

Register	Offset
PCP2PCPMP0R	B00h
PCP2PCPMP1R	B04h

Function

This is the PCP to PCP mapping profile register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	PCP7			Reserved	PCP6			Reserved	PCP5			Reserved	PCP4		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	PCP3			Reserved	PCP2			Reserved	PCP1			Reserved	PCP0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 PCP7	PCP7 priority code point mapped value.
27 —	Reserved
26-24 PCP6	PCP6 priority code point mapped value.
23 —	Reserved
22-20 PCP5	PCP5 priority code point mapped value.
19 —	Reserved
18-16 PCP4	PCP4 priority code point mapped value.
15 —	Reserved
14-12 PCP3	PCP3 priority code point mapped value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 —	Reserved
10-8 PCP2	PCP2 priority code point mapped value.
7 —	Reserved
6-4 PCP1	PCP1 priority code point mapped value.
3 —	Reserved
2-0 PCP0	PCP0 priority code point mapped value.

53.4.6.5.33 Bridge capability register (BRCAPR)

Offset

Register	Offset
BRCAPR	2000h

Function

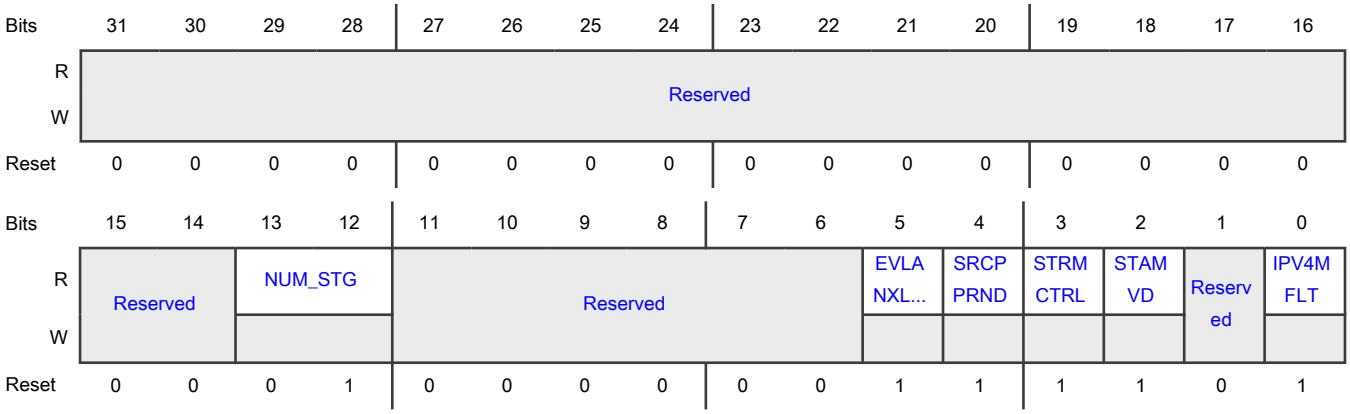
The following are basic bridge capabilities provided:

- **VLAN filtering:** Each VLAN belongs to a port membership set. Ingress VLAN filtering ensures that only frames belonging to the port member is allowed. Similarly, egress VLAN filtering ensures that we only forward frames to a port which is a member of the VLAN
- **MAC Learning** using FDB table; MAC learning options supported per VLAN are:
  1. Disable MAC learning
  2. Hardware MAC learning
  3. Software MAC learning secure; if SMAC is not found, or if an entry is found and the entry's port number does not match the frame ingress port number (and station move is allowed), redirect frame to switch management port.
  4. Software MAC learning unsecure; if an entry is not found, or if an entry is found and the entry's port number does not match the frame ingress port number (and station move is allowed), copy frame to switch management port.
  5. Disable MAC learning with SMAC validation. Ensure SMAC is in FDB otherwise discard frame.
- **MAC Forwarding** using FDB table. A MAC address may be a unicast, multicast or broadcast address. MAC forwarding options supported per VLAN if MAC DA is not found in FDB are:

1. Flood to all port members of the VLAN
2. Discard

- Other capabilities are specified in this CAPR.

Diagram



Fields

Field	Function
31-14 —	Reserved
13-12 NUM_STG	Number of Spanning Tree Groups supported. 0: Not supported 1: Support 16 Spanning Tree Groups 2-3: Reserved The Spanning Tree Groups are defined in BPSTGSR.
11-6 —	Reserved
5 EVLANXLATE	Egress VLAN translation supported **NOTE** Accomplished using VLAN Filter table entry which can optionally specify an Egress Treatment table entry for each port membership in the VLAN. Each Egress Treatment table entry may specify a Frame Modification table entry which contains the egress VLAN translation action. 0b - Egress VLAN translation not supported 1b - Egress VLAN translation supported
4 SRCPPRND	Source port pruning disable supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When supported, it provides the capability to configure on a per port basis to transmit a frame on the same port it was received. Default bridge capability is to prevent a frame from being transmitted on same port. 0b - Source port pruning disable not supported 1b - Source port pruning disable supported
3 STRMCTRL	Storm Control supported. Storm control provides the ability to rate limit broadcast, multicast, unknown multicast, and unknown unicast traffic on a given port. 0b - Not supported 1b - Supported
2 STAMVD	Station Move Disable supported Provides the option to discard a frame when the MAC learning function finds an entry with the ingress port of the incoming frame not set in the FDB entry Destination Port Bitmap (station has moved). 0b - Not supported 1b - Supported
1 —	Reserved
0 IPV4MFLT	L2 IPv4 multicast filtering supported. 0b - Not supported 1b - Supported

53.4.6.5.34 VLAN filter hash table capability register (VFHTCAPR)

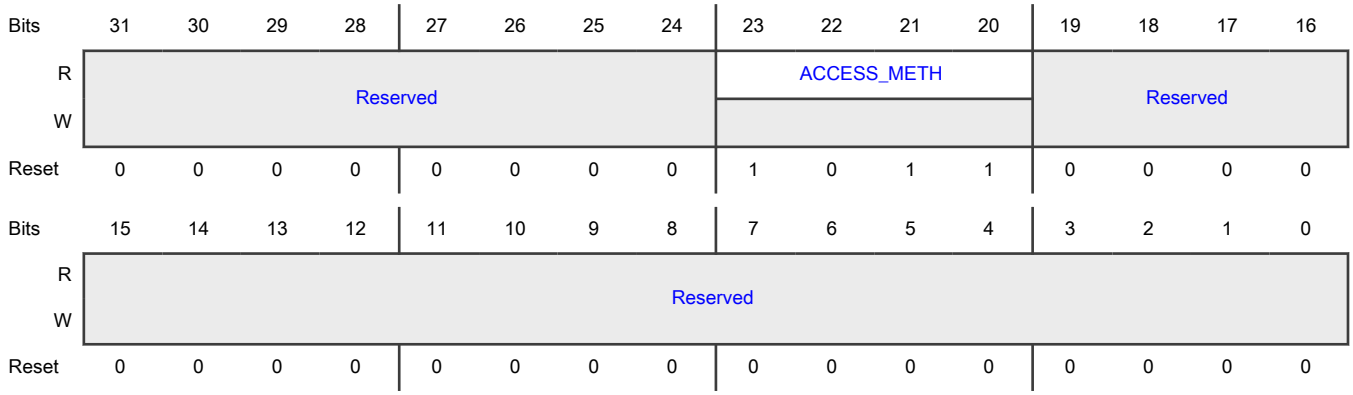
Offset

Register	Offset
VFHTCAPR	2008h

Function

This is the VLAN filter hash table capability register.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-0 —	Reserved

53.4.6.5.35 VLAN filter hash table operational register (VFHTOR)

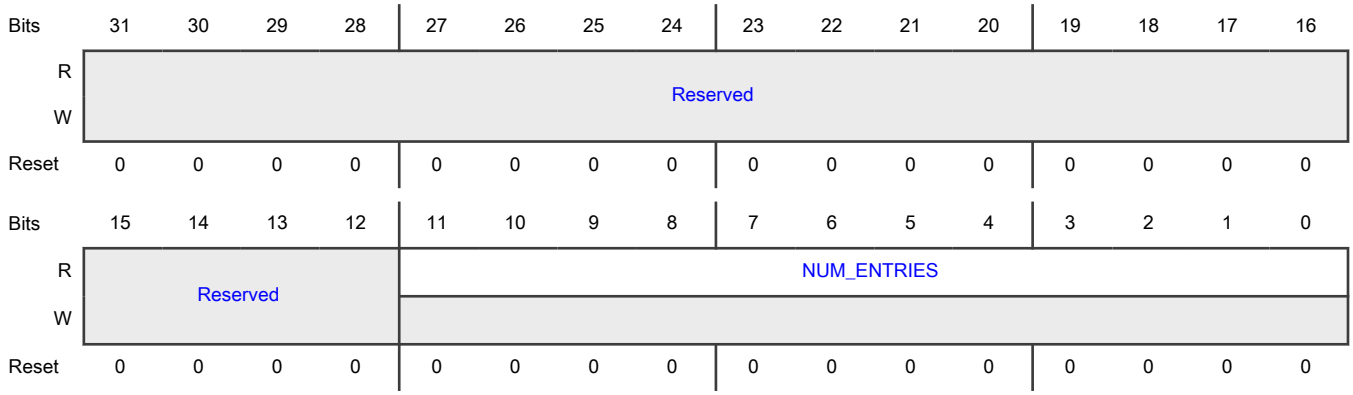
Offset

Register	Offset
VFHTOR	200Ch

Function

This is the VLAN Filter (hash) table operational register.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 NUM_ENTRIES	Number of entries in-use by the VLAN Filter table.

53.4.6.5.36 VLAN Filter (hash) table default entry configuration registers 0 (VFHTDECR0)

Offset

Register	Offset
VFHTDECR0	2010h

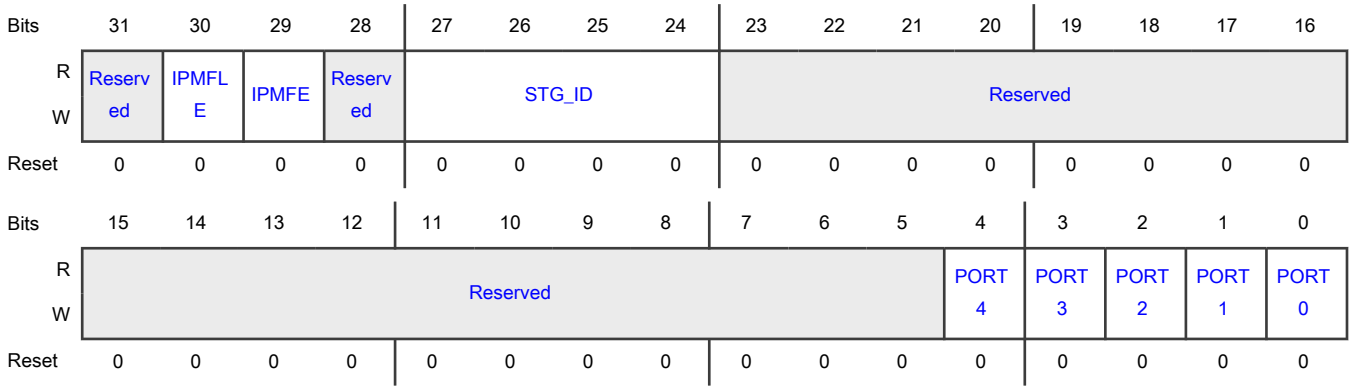
Function

This is the VLAN filter hash table default entry configuration registers 0, which determines the default entry when not found in VLAN filter lookup.

NOTE

The switch will sample the IERB value of this register (S0VFHTDECR0) when PCIe Memory Access bit is set (b1). All updates after this must be done through this register.

Diagram



Fields

Field	Function
31 —	Reserved
30 IPMFLE	IP Multicast Flooding Enable If IP multicast filtering is performed (IPMFE = 1b, and the frame is identified as a multicast IP packet), and there was no match found, then the frame is forwarded according to this field. If IP multicast filtering is disabled (IPMFE = 0b), this field is ignored by hardware. 0b - IP Multicast Flooding disabled, the frame is discarded. 1b - IP Multicast Flooding enabled, the frame is flooded.
29 IPMFE	IP Multicast Filtering Enable Refer to the VLAN Filter table entry IPMFE field description, for more details 0b - No IP multicast filtering is performed. 1b - If the frame is identified as a multicast IP packet, then IP multicast filtering is performed. If the frame is not identified as an IP multicast packet, the IP multicast filtering is not performed.
28 —	Reserved
27-24 STG_ID	Spanning Tree Group Member ID. Refer to the VLAN Filter table entry STG_ID field description, for more details
23-5 —	Reserved
4-0 PORTn	Port n This field specifies if port number n is a member of this VLAN. Port membership is used for source pruning in ingress VLAN filtering and destination pruning in egress VLAN filtering. This field is also used as a default

Table continues on the next page...

Table continued from the previous page...

Field	Function
	destination for a miss from an FDB table lookup (or for the case where the VLAN is configured to not perform an FDB lookup). 0b - Port is not a member of this VLAN. 1b - Port is a member of this VLAN.

53.4.6.5.37 VLAN filter hash table default entry configuration registers 1 (VFHTDECR1)

Offset

Register	Offset
VFHTDECR1	2014h

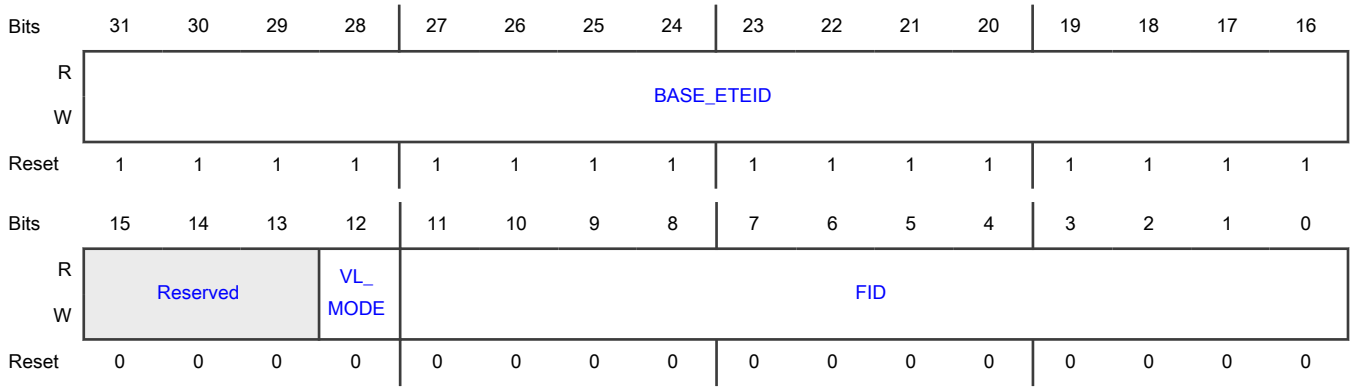
Function

This is the VLAN Filter (hash) table default entry configuration registers 1.

NOTE

The switch will sample the IERB value of this register (S0VFHTDECR1) when PCIe Memory Access bit is set (b1). All updates after this must be done through this register.

Diagram



Fields

Field	Function
31-16 BASE_ETEID	Base Egress Treatment Entry ID Refer to the VLAN Filter table entry BASE_ET_EID field description, for more details.
15-13 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 VL_MODE	VLAN Learning Mode 0: Independent VLAN learning: FID is set to the VID assigned to the frame 1: Shared VLAN learning: Use the FID specified in this register **NOTE** Used to determine the FID when doing a lookup in the FDB table.
11-0 FID	Filtering ID. Valid if VL_MODE = 1 (Shared). The Filtering ID (FID) is a locally significant identifier (global to the switch), that is used as a key value when hardware performs a lookup into the FDB table and the L2 IPV4 Multicast Filter table. The FID is used to identify a set of VIDs, which in turn allows sharing of the same address (MAC, IP) between multiple VIDs during lookups into the FDB table and L2 IPV4 Multicast Filter table.

53.4.6.5.38 VLAN filter hash table default entry configuration registers 2 (VFHTDECR2)

Offset

Register	Offset
VFHTDECR2	2018h

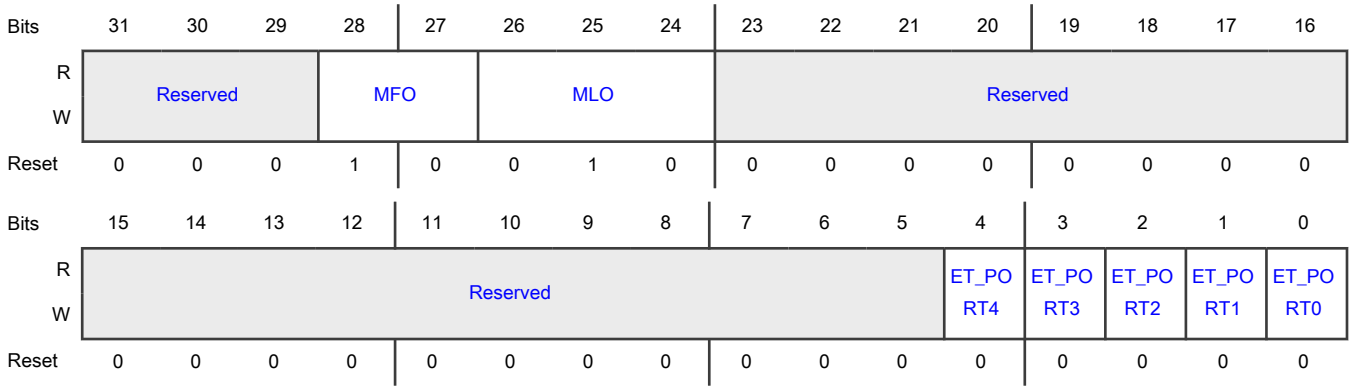
Function

This is the VLAN Filter (hash) table default entry configuration registers 2.

NOTE

The switch will sample the IERB value of this register (S0VFHTDECR2) when PCIe Memory Access bit is set (b1). All updates after this must be done through this register.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-27 MFO	MAC forwarding options: 0: Reserved 1: No FDB lookup is performed, the frame is flooded. 2: FDB lookup is performed, and if there is no match, the frame is flooded. 3: FDB lookup is performed, and if there is no match, the frame is discarded.
26-24 MLO	MAC learning options: 0: Reserved 1: Disable MAC learning. MAC learning function is not performed during the forwarding processing. 2: Hardware MAC learning is performed. 3: Software MAC learning secure. A MAC learning lookup is performed into the FDB table. If there is no match, no attempt is made to add a new entry, and the frame is redirect to the switch management port. If there is match, and the entry's port number does not match frame ingress port number, the frame is redirected to the switch management port if station move is allowed, otherwise the frame is discarded. 4: Software MAC learning unsecure. A MAC learning lookup is performed into the FDB table. If there is no match, no attempt is made to add a new entry, and a copy of the frame is sent to the switch management port. If there is match, and the entry's port number does not match frame ingress port number, the frame is copied to the switch management port if station move is allowed, otherwise the frame is discarded. 5: Disable MAC learning with SMAC validation. A MAC learning lookup is performed into the FDB table. If there is no match or there is a match but the ingress port is not a member of the FDB entry, the frame is discarded and counted against the bridge port discard count register (BPDCR) with discard reason BPDCRR0[MACLNFDR] set to 1. All other values reserved.
23-5 —	Reserved
4-0 ET_PORTn	Egress Treatment Applicability Port n. This field specifies if port number n has an entry in the Egress Treatment table for the Egress Treatment group (base index) specified in VFHTDECR1[BASE_ETEID]. This field is valid if VFHTDECR1[BASE_ETEID] is not null. 0b - Port has no entry in the Egress Treatment table 1b - Port has an entry in the Egress Treatment table

53.4.6.5.39 FDB hash table capability register (FDBHTCAPR)

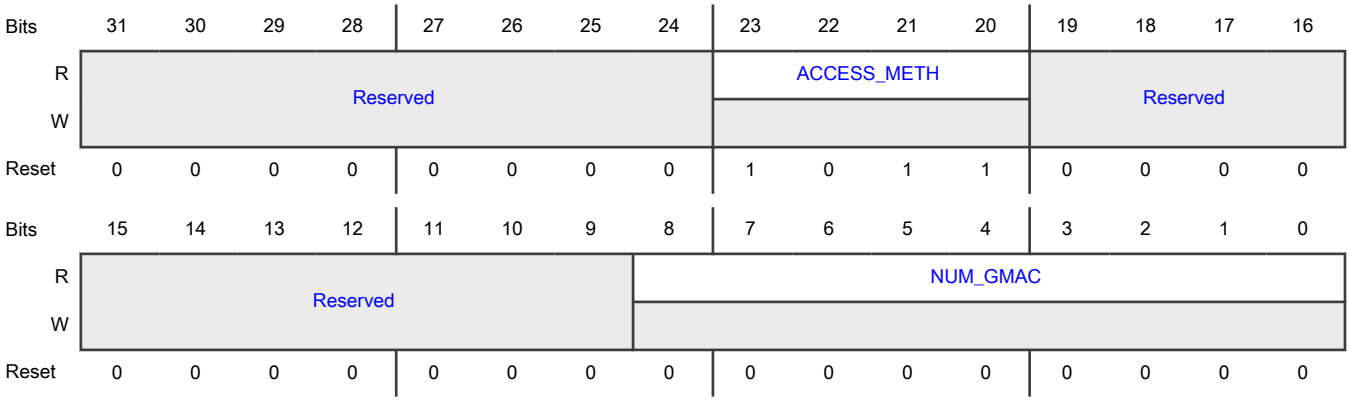
Offset

Register	Offset
FDBHTCAPR	2020h

Function

This is the FDB (hash) table capability register.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-9 —	Reserved
8-0 NUM_GMAC	Number of guaranteed MAC addresses which can be added to the FDB table if an entry cannot be added due to collision limit being exceeded.

53.4.6.5.40 FDB hash table memory configuration register (FDBHTMCR)

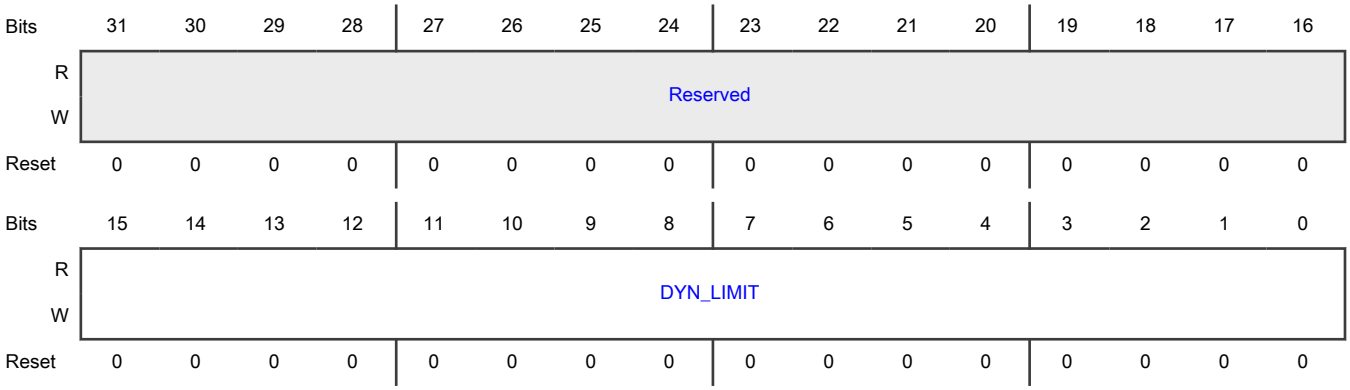
Offset

Register	Offset
FDBHTMCR	2024h

Function

This is the FDB (hash) table memory configuration register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 DYN_LIMIT	This field specifies the maximum number of dynamic entries allowed in the FDB table for the entire switch. **NOTE** A value of 0 implies no global switch limit imposed for dynamic entries. Dynamic entries can be added by the hardware MAC learning function or a FDB table management command. In the case where the hardware MAC learning function cannot add a dynamic FDB entry due to this limit having been reached, the frame continues to be forwarded and BPMLFSR[FFDBTRFR] failure reason is set.

53.4.6.5.41 FDB hash table operational register 0 (FDBHTOR0)

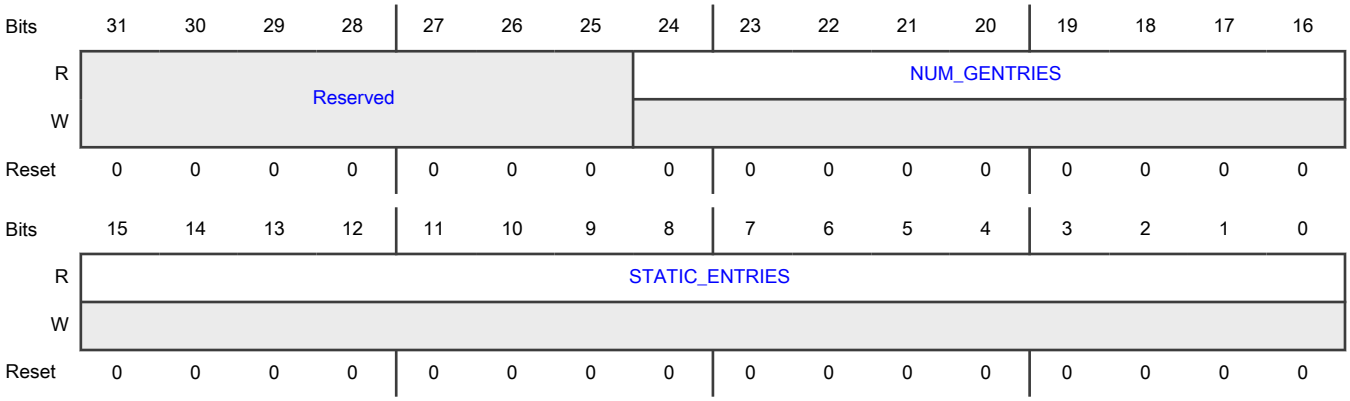
Offset

Register	Offset
FDBHTOR0	2028h

Function

This is the FDB (hash) table operational register 0.

Diagram



Fields

Field	Function
31-25 —	Reserved
24-16 NUM_GENTRIES	The number of guaranteed MAC entries in-use in the FDB table. Each guaranteed MAC entry is used to store the content of a switch FDB entry, when the FDB table entry is added as a Content-Addressable-Memory (CAM) entry, as opposed to an hash entry (hash space), due to collision chain length having reached its maximum length.
15-0 STATIC_ENTRIES	Number of static entries in-use in the FDB table.

53.4.6.5.42 FDB hash table operational register 1 (FDBHTOR1)

Offset

Register	Offset
FDBHTOR1	202Ch

Function

This is the FDB (hash) table operational register 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HWM_DYN_ENTRIES															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DYN_ENTRIES															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 HWM_DYN_ENTRIES	High water mark of dynamic entries in-use in the FDB table. **NOTE** Value reset to DYN_ENTRIES when read.
15-0 DYN_ENTRIES	Number of dynamic entries in-use in the FDB table.

53.4.6.5.43 IP multicast filter hash table capability register (IPMFHTCAPR)

Offset

Register	Offset
IPMFHTCAPR	2040h

Function

This is the L2 IPv4 Multicast Filter (hash) table capability register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								ACCESS_METH				Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: xxx1: Index xx1x: EntryId x1xx: Search 1xxx: Reserved
19-0 —	Reserved

53.4.6.5.44 IPv4 multicast filter hash table operation register (IPv4MFHTOR)

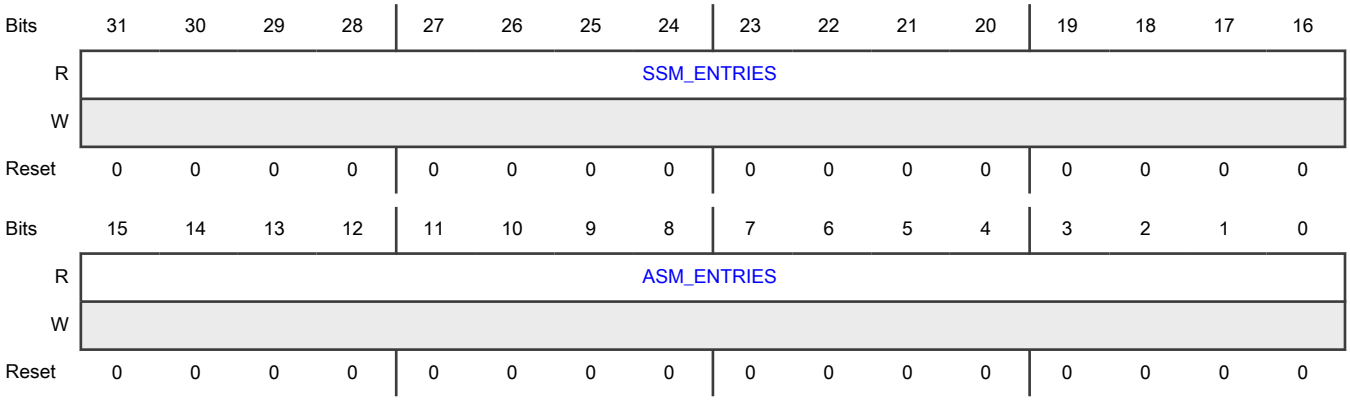
Offset

Register	Offset
IPv4MFHTOR	2044h

Function

This is the IPv4 multicast filter hash table operational register.

Diagram



Fields

Field	Function
31-16	Number of IPV4 Source Specific Multicast (SSM) Multicast Filter table entries in-use.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SSM_ENTRIES	
15-0 ASM_ENTRIES	Number of IPV4 Any Source Multicast (ASM) Multicast Filter table entries in-use.

53.4.6.6 ENETC base register descriptions

This section describes the ENETC specific base registers for one or more instances.

There may be differences in actual implemented registers or register fields as well as reset values between instances. The function's base address register value can be discovered from reading the PCIe Enhanced Allocation Base Address Register Equivalent Index 0 (EA BEI 0).

53.4.6.6.1 ENETC memory map

Instance	Address space	PCIe EA BEI 0 offset
ENETC0_BASE	PCI_F3_BAR_0	60B1_0000h
ENETC1_BASE	PCI_F4_BAR_0	60B5_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ENETC capability register 0 (ECAPR0)	32	R	See section
4h	ENETC capability register 1 (ECAPR1)	32	R	See section
8h	ENETC capability register 2 (ECAPR2)	32	R	See section
10h	Port mode register (PMR)	32	RW	0001_0000h
80h	Port outer native VLAN register (PONVLANR)	32	RW	0000_0000h
84h	Port inner native VLAN register (PINVLANR)	32	RW	0000_0000h
88h	Port VLAN classification control register (PVCLCTR)	32	RW	0000_0000h
9Ch	Parser checksum configuration register (PARCSCR)	32	RW	0000_0000h
A0h - ACh	Parser custom Ethertype a configuration register (PARCE0CR - PARCE3CR)	32	RW	0000_0000h
108h	Port pause ON threshold register (PPAUONTR)	32	RW	0000_0000h
10Ch	Port pause OFF threshold register (PPAUOFFTR)	32	RW	0000_0000h
120h	Port receive memory buffer entitlement register (PRXMBER)	32	R	0000_0000h
124h	Port receive memory buffer limit register (PRXMBLR)	32	R	0000_0000h
128h	Port receive buffer count register (PRXBCR)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
12Ch	Port receive buffer count high watermark register (PRXBCHWMR)	32	R	0000_0000h
140h	Port ingress congestion DR0 discard count register (PICDR0DCR)	32	R	0000_0000h
148h	Port ingress congestion DR0 discard count read-reset register (PICDR0DCRRR)	32	R	0000_0000h
150h	Port ingress congestion DR1 discard count register (PICDR1DCR)	32	R	0000_0000h
158h	Port ingress congestion DR1 discard count read-reset register (PICDR1DCRRR)	32	R	0000_0000h
160h	Port ingress congestion DR2 discard count register (PICDR2DCR)	32	R	0000_0000h
168h	Port ingress congestion DR2 discard count read-reset register (PICDR2DCRRR)	32	R	0000_0000h
170h	Port ingress congestion DR3 discard count register (PICDR3DCR)	32	R	0000_0000h
178h	Port ingress congestion DR3 discard count read-reset register (PICDR3DCRRR)	32	R	0000_0000h
180h	Port ingress congestion priority discard status register (PICPDSR)	32	RW	0000_0000h
200h	Port station interface promiscuous MAC mode register (PSIPMMR)	32	RW	See section
204h	Port station interface promiscuous VLAN mode register (PSIPVMR)	32	RW	See section
208h	Port broadcast frames dropped due to MAC filtering register (PBFDSIR)	32	R	0000_0000h
20Ch	Port frame drop MAC source address pruning register (PFDMSAPR)	32	R	0000_0000h
280h	Port station interface MAC address filtering capability register (PSIMAFCAPR)	32	R	0000_0004h
284h	Port unicast frames dropped due to MAC filtering register (PUFDMFR)	32	R	0000_0000h
288h	Port multicast frames dropped due to MAC filtering register (PMFDMFR)	32	R	0000_0000h
2C0h	Port station interface VLAN filtering capability register (PSIVLANFCAPR)	32	R	0000_0004h
2C4h	Port station interface VLAN filtering mode register (PSIVLANFMR)	32	RW	0000_0000h
2D0h	Port unicast frames dropped VLAN filtering register (PUFDVFR)	32	R	0000_0000h
2D4h	Port multicast frames dropped VLAN filtering register (PMFDVFR)	32	R	0000_0000h
2D8h	Port broadcast frames dropped VLAN filtering register (PBFDVFR)	32	R	0000_0000h
340h	Port low power mode register (PLPMR)	32	RW	0000_0000h
344h	Port wake-on status register (PWOSR)	32	R	0000_0000h
370h	Receive IPV to ICM priority mapping register 0 (IPV2ICMPMR0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
380h	Transmit priority to traffic class mapping register 0 (PRIO2TCMR0)	32	RW	7654_3210h
390h - 3ACh	Port traffic class a time specific departure register (PTC0TSDR - PTC7TSDR)	32	RW	0000_0000h
800h	Switch management capability register (SMCAPR)	32	R	See section
880h - 8B8h	Switch management host reason a receive BD ring mapping register (SMHR1BDRMR - SMHR15BDRMR)	32	RW	0000_0000h
2000h	Port station interface 0 primary MAC address register 0 (PSI0PMAR0)	32	R	0000_0000h
2004h	Port station interface 0 primary MAC address register 1 (PSI0PMAR1)	32	R	0000_0000h
2008h	Port station interface 0 VLAN register (PSI0VLANR)	32	RW	0000_0000h
2010h	Port station interface 0 configuration register 0 (PSI0CFGR0)	32	RW	See section
2018h	Port station interface 0 configuration register 2 (PSI0CFGR2)	32	RW	See section
2030h	Port station interface 0 VSI MAC address filtering configuration register (PSI0VMAFCFGR)	32	RW	0000_0004h
2034h	Port station interface 0 VLAN filtering configuration register (PSI0VLANFCFGR)	32	RW	0000_0004h
2050h	Port station interface 0 unicast MAC hash filter register 0 (PSI0UMHFR0)	32	RW	0000_0000h
2054h	Port station interface 0 unicast MAC hash filter register 1 (PSI0UMHFR1)	32	RW	0000_0000h
2058h	Port station interface 0 multicast MAC hash filter register 0 (PSI0MMHFR0)	32	RW	0000_0000h
205Ch	Port station interface 0 multicast MAC hash filter register 1 (PSI0MMHFR1)	32	RW	0000_0000h
2060h	Port station interface 0 VLAN hash filter register 0 (PSI0VHFR0)	32	RW	0000_0000h
2064h	Port station interface 0 VLAN hash filter register 1 (PSI0VHFR1)	32	RW	0000_0000h
2080h	Port station interface 1 primary MAC address register 0 (PSI1PMAR0)	32	RW	0000_0000h
2084h	Port station interface 1 primary MAC address register 1 (PSI1PMAR1)	32	RW	0000_0000h
2088h	Port station interface 1 VLAN register (PSI1VLANR)	32	RW	0000_0000h
2090h	Port station interface 1 configuration register 0 (PSI1CFGR0)	32	RW	0000_0000h
2094h	Port station interface 1 configuration register 1 (PSI1CFGR1)	32	RW	7654_3210h
2098h	Port station interface 1 configuration register 2 (PSI1CFGR2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20B0h	Port station interface 1 VSI MAC address filtering configuration register (PSI1VMAFCFGR)	32	RW	0000_0000h
20B4h	Port station interface 1 VLAN filtering configuration register (PSI1VLANFCFGR)	32	RW	0000_0000h
20D0h	Port station interface 1 unicast MAC hash filter register 0 (PSI1UMHFR0)	32	RW	0000_0000h
20D4h	Port station interface 1 unicast MAC hash filter register 1 (PSI1UMHFR1)	32	RW	0000_0000h
20D8h	Port station interface 1 multicast MAC hash filter register 0 (PSI1MMHFR0)	32	RW	0000_0000h
20DCh	Port station interface 1 multicast MAC hash filter register 1 (PSI1MMHFR1)	32	RW	0000_0000h
20E0h	Port station interface 1 VLAN hash filter register 0 (PSI1VHFR0)	32	RW	0000_0000h
20E4h	Port station interface 1 VLAN hash filter register 1 (PSI1VHFR1)	32	RW	0000_0000h

53.4.6.6.2 ENETC capability register 0 (ECAPR0)

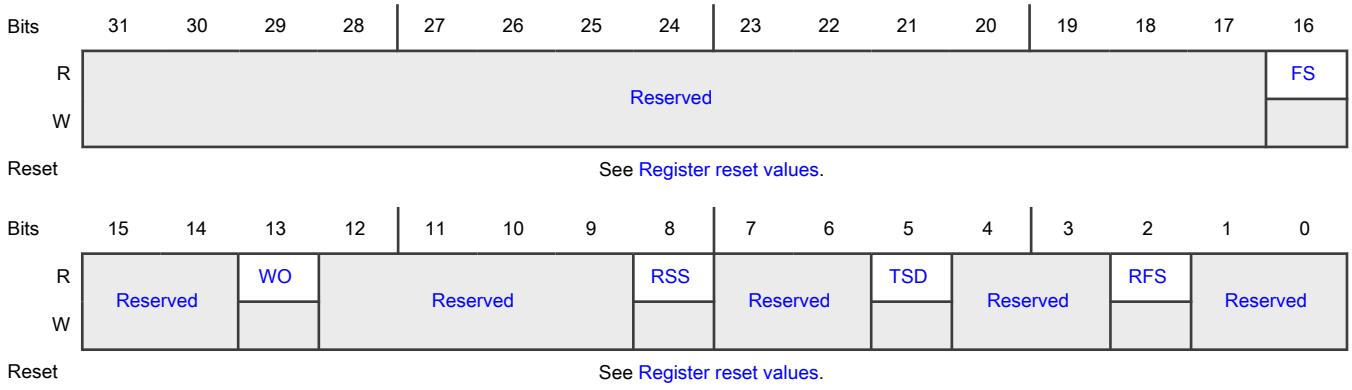
Offset

Register	Offset
ECAPR0	0h

Function

This is the ENETC capability register 0 which reflects the hardware capability regarding offload functions and transmit/receive rings.

Diagram



Register reset values

Register	Reset value
ECAPR0	ENETC0_BASE: 0000_2020h ENETC1_BASE: 0000_0020h

Fields

Field	Function
31-17 —	Reserved
16 FS	Functional safety capability supported. If set, additional integrity error checks are performed; this includes, but not limited to, internal FCS and parity error checks. 0: Not supported 1: Supported
15-14 —	Reserved
13 WO	Support for Wake-on-LAN in low-power mode 0: Not supported 1: Supported
12-9 —	Reserved
8 RSS	Support for RSS 0: Not supported 1: Supported
7-6 —	Reserved
5 TSD	Support for time specific departure. 0: Not supported 1: Supported
4-3 —	Reserved
2	Receive flow steering support

Table continues on the next page...

Table continued from the previous page...

Field	Function
RFS	0: Not supported 1: Supported
1-0 —	Reserved

53.4.6.6.3 ENETC capability register 1 (ECAPR1)

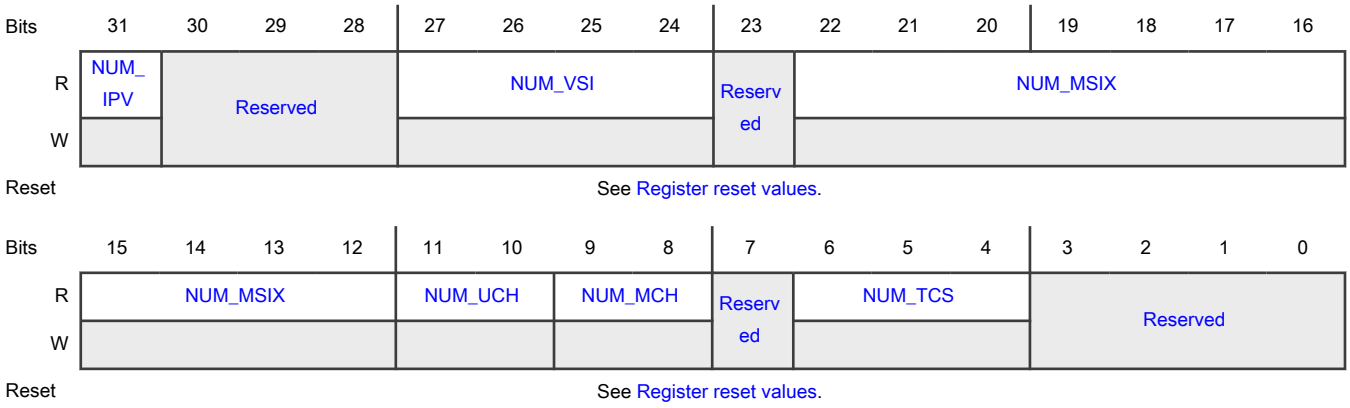
Offset

Register	Offset
ECAPR1	4h

Function

This is the ENETC capability register 1 which reflects the hardware capability.

Diagram



Register reset values

Register	Reset value
ECAPR1	ENETC0_BASE: 0000_B070h ENETC1_BASE: 0101_8070h

Fields

Field	Function
31 NUM_IPV	Indicates the number of IPVs supported. 0: 8 IPVs 1: 16 IPVs
30-28 —	Reserved
27-24 NUM_VSI	Indicates the number of VSI supported. This value is determined by IERB register E*a*VBCR, and is immediately reflected here.
23 —	Reserved
22-12 NUM_MSIX	Number of MSI-X Total number of MSI-X vectors provided for ENETC instance for allocation to PSI and any associated VSIs Range: 1..1024 Formula: NUM_MSIX+1
11-10 NUM_UCH	Number of uni-cast hash entry (1bit/entry) per SI for L2 MAC Filtering 00: 64 unicast bins 01: 128 unicast bins 10: 256 unicast bins 11: 512 unicast bins
9-8 NUM_MCH	Number of multi-cast hash entry (1bit/entry) per SI for L2 MAC Filtering 00: 64 multi-cast bins 01: 128 multi-cast bins 10: 256 multi-cast bins 11: 512 multi-cast bins
7 —	Reserved
6-4 NUM_TCS	Number of traffic classes 0 - 1 Traffic class (0) 1 - 2 Traffic classes (0-1) ... 7 - 8 Traffic classes (0-7)
3-0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

53.4.6.6.4 ENETC capability register 2 (ECAPR2)

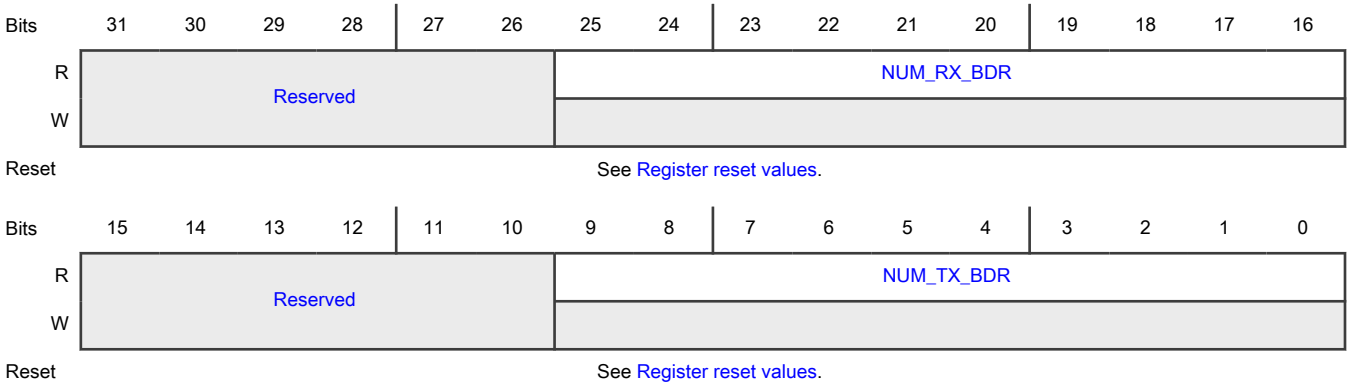
Offset

Register	Offset
ECAPR2	8h

Function

This is the ENETC capability register 2 which reflects the hardware capability.

Diagram



Register reset values

Register	Reset value
ECAPR2	ENETC0_BASE: 0004_0004h ENETC1_BASE: 000A_000Ah

Fields

Field	Function
31-26 —	Reserved
25-16	Number of total receive buffer descriptor rings assigned to ENETC

Table continues on the next page...

Table continued from the previous page...

Field	Function
NUM_RX_BDR	Range: 0..1023
15-10 —	Reserved
9-0 NUM_TX_BDR	Number of total transmit buffer descriptor rings assigned to ENETC Range: 0..1023

53.4.6.6.5 Port mode register (PMR)

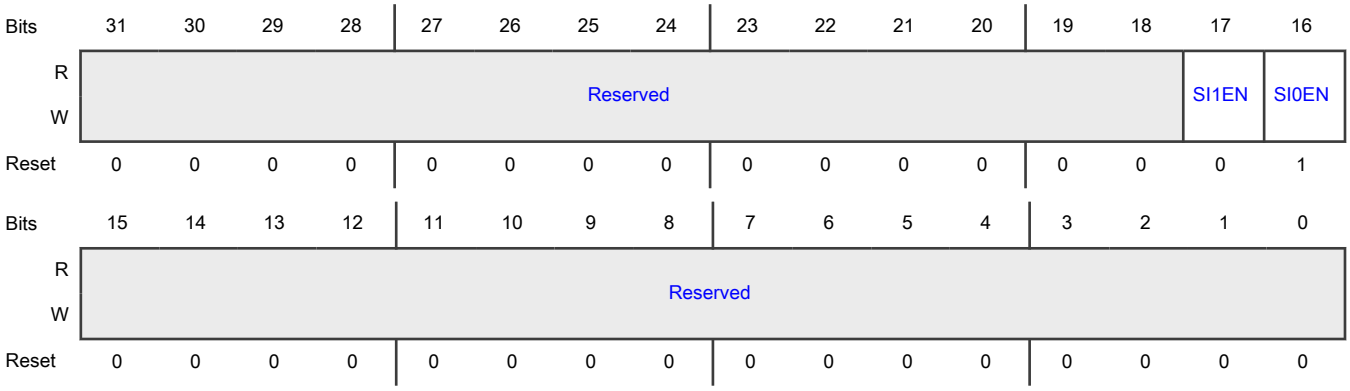
Offset

Register	Offset
PMR	10h

Function

This is the ENETC mode register for the port.

Diagram



Fields

Field	Function
31-18 —	Reserved
17 SI1EN	Enable station interface 1. This is the master enable for a station interface. It is ANDed with the SIMR[EN] bit setting for each station interface.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>See register bit EN for functional description.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>ENETC0_BASE</td><td>—</td><td>PMR</td></tr><tr><td>ENETC1_BASE</td><td>PMR</td><td>—</td></tr></table>
0b - Disabled
1b - Enabled</td></tr>
<tr><td>16
SI0EN</td><td>Enable station interface 0.
This is the master enable for a station interface. It is ANDed with the SIMR[EN] bit setting for each station interface.
See register bit EN for functional description.
0b - Disabled
1b - Enabled</td></tr>
<tr><td>15-0
—</td><td>Reserved</td></tr>
</table>

53.4.6.6.6 Port outer native VLAN register (PONVLANR)

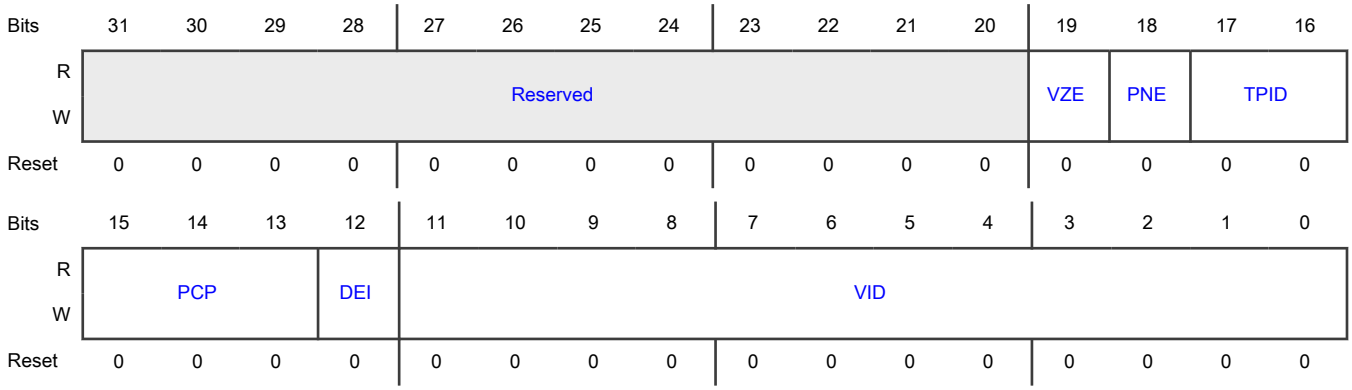
Offset

Register	Offset
PONVLANR	80h

Function

This is the outer native VLAN for the port. It is used for classification when untagged frames are received by the port.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VZE	VID 0 Enable Enables and disables usage of the port outer native VLAN VID when the VID in the frame's outer VLAN tag is zero (priority tagged frame). 0b - Disabled 1b - Enabled
18 PNE	Port Native VLAN Enable Enables and disables usage of the port outer native VLAN (PCP, DEI and VID) when the outer VLAN tag is not present in the frame. 0b - Disabled 1b - Enabled
17-16 TPID	Identifies which known VLAN Ethertype is used 00b - Standard C-VLAN 0x8100 01b - Standard S-VLAN 0x88A8 10b - Custom VLAN as defined by CVLANR1[ETYPE] 11b - Custom VLAN as defined by CVLANR2[ETYPE]
15-13 PCP	Priority code point A 3-bit field which refers to the IEEE 802.1p class of service and maps to the frame priority level.
12 DEI	Drop eligible indicator May be used separately or in conjunction with PCP to indicate frames eligible to be dropped in the presence of congestion.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-0 VID	VLAN identifier The VLAN identifier is a 12-bit field specifying the VLAN to which the frame belongs.

53.4.6.6.7 Port inner native VLAN register (PINVLANR)

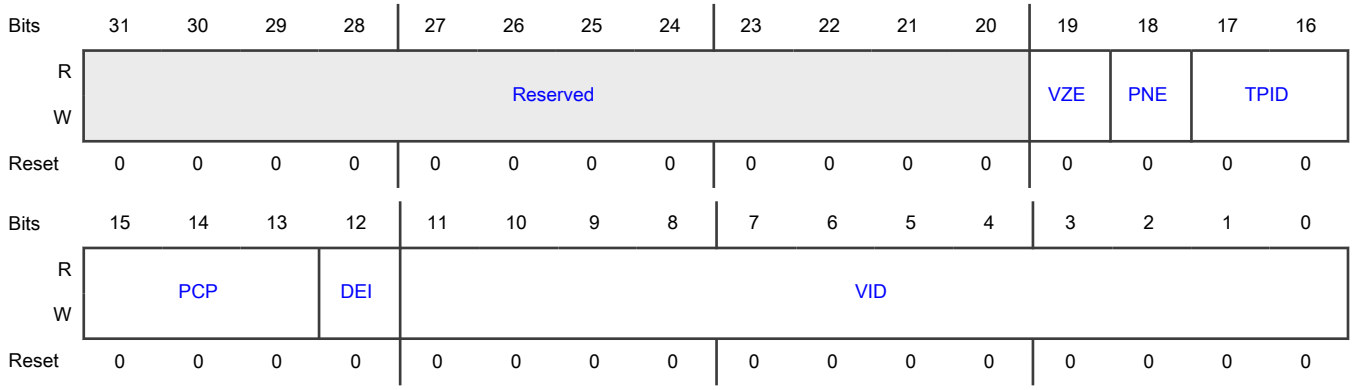
Offset

Register	Offset
PINVLANR	84h

Function

This is the inner native VLAN for the port. It is used for classification when untagged frames are received by the port.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VZE	VID 0 Enable Enables and disables usage of the port inner native VLAN VID when the VID in the frame's inner VLAN tag is zero (priority tagged frame). 0b - Disabled 1b - Enabled
18	Port Native VLAN Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
PNE	Enables and disables usage of the port inner native VLAN (PCP, DEI and VID) when the inner VLAN tag is not present in the frame. 0b - Disabled 1b - Enabled
17-16 TPID	Identifies which known VLAN Ethertype is used 00b - Standard C-VLAN 0x8100 01b - Standard S-VLAN 0x88A8 10b - Custom VLAN as defined by CVLANR1[ETYPE] 11b - Custom VLAN as defined by CVLANR2[ETYPE]
15-13 PCP	Priority code point A 3-bit field which refers to the IEEE 802.1p class of service and maps to the frame priority level.
12 DEI	Drop eligible indicator May be used separately or in conjunction with PCP to indicate frames eligible to be dropped in the presence of congestion.
11-0 VID	VLAN identifier The VLAN identifier is a 12-bit field specifying the VLAN to which the frame belongs.

53.4.6.6.8 Port VLAN classification control register (PVCLCTR)

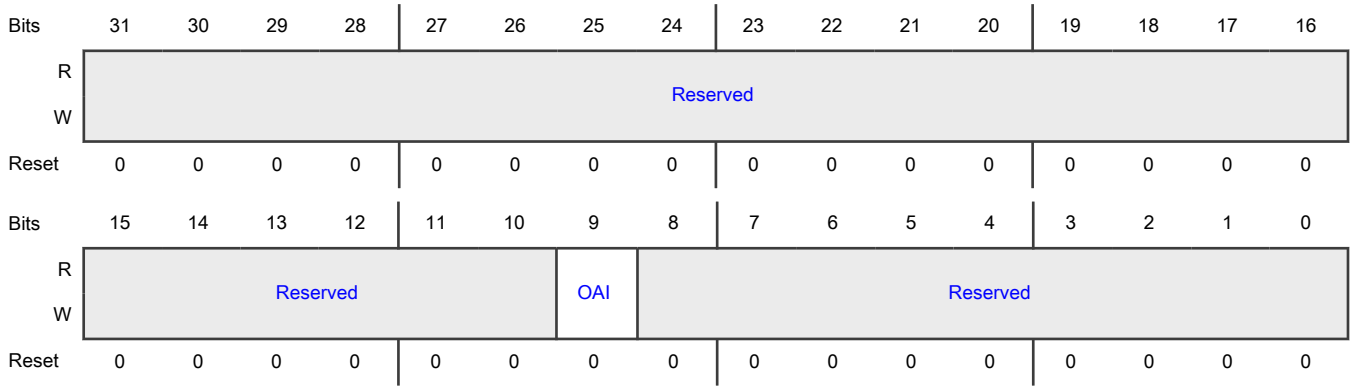
Offset

Register	Offset
PVCLCTR	88h

Function

This register controls the VLAN classification.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 OAI	Outer as Inner 0b - Indicates that the Inner is not valid if only one tag is found 1b - Indicates that the outer should be used as the Inner if only one tag is found
8-0 —	Reserved

53.4.6.6.9 Parser checksum configuration register (PARCSCR)

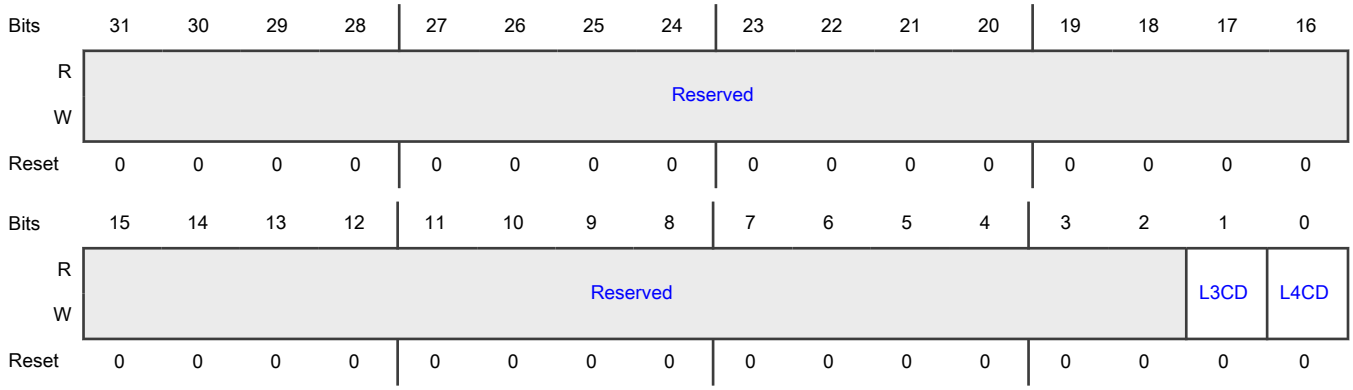
Offset

Register	Offset
PARCSCR	9Ch

Function

This is the parser checksum configuration register.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 L3CD	Layer 3 IPv4 Header checksum validation Disable. 0b - Enabled 1b - Disabled
0 L4CD	Layer 4 TCP and UDP checksum validation Disable. 0b - Enabled 1b - Disabled

53.4.6.6.10 Parser custom Ethertype a configuration register (PARCE0CR - PARCE3CR)

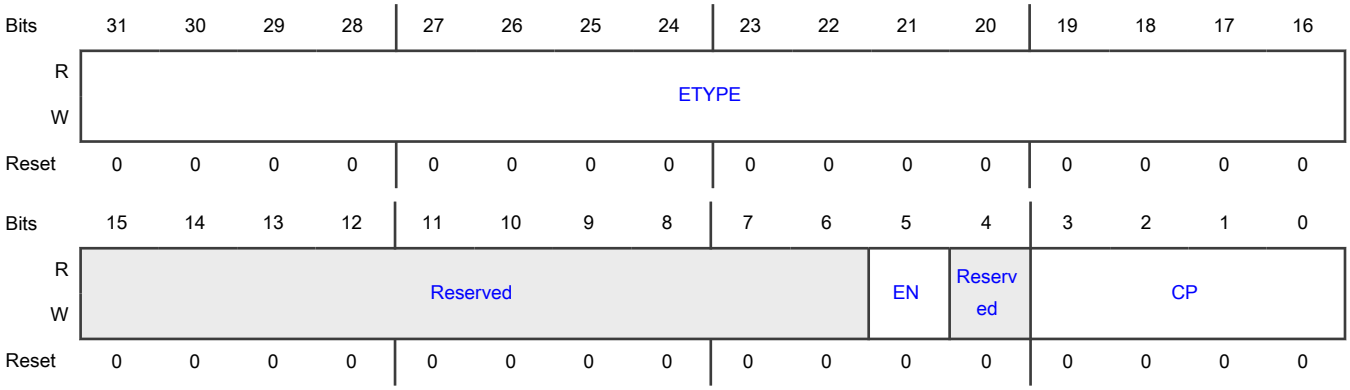
Offset

Register	Offset
PARCE0CR	A0h
PARCE1CR	A4h
PARCE2CR	A8h
PARCE3CR	ACh

Function

The HTA parser custom Ethertype register.

Diagram



Fields

Field	Function
31-16 ETYPE	ETYPE Upon detecting this ether type the associated code point will be mapped to the parse summary as a Non IP code point. All HTA parsing will stop
15-6 —	- Reserved
5 EN	Enable Enables the detection and mapping. 0b - Disabled 1b - Enabled
4 —	- Reserved
3-0 CP	Code Point This value is mapped to the parse summary as a Non IP code point with the parse summary "L2 Only" field set to the value stored in this field. Only a value of 0 (Unknown) is currently supported.

53.4.6.6.11 Port pause ON threshold register (PPAUONTR)

Offset

Register	Offset
PPAUONTR	108h

Function

This is the pause ON threshold register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								THRESH							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	THRESH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 THRESH	Monitors the amount of data accumulated in the receive buffer and if this amount exceeds the Pause ON threshold (expressed in words), it then enters the "Pause ON" state if Pause is enabled. Pause ON threshold has to be greater value than Pause OFF threshold. Setting this register to 0 or setting PPAUOFFTR register to 0 disables pause generation.

53.4.6.6.12 Port pause OFF threshold register (PPAUOFFTR)

Offset

Register	Offset
PPAUOFFTR	10Ch

Function

This is the pause OFF threshold register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								THRESH							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	THRESH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 THRESH	Monitors the amount of data accumulated in the receive buffer and if this amount goes below the Pause OFF threshold (expressed in words) it then enters the "Pause OFF" state if Pause is enabled. Pause ON threshold has to be greater value than Pause OFF threshold. Setting this register to 0 or setting PPAUONTR to 0 disables pause generation.

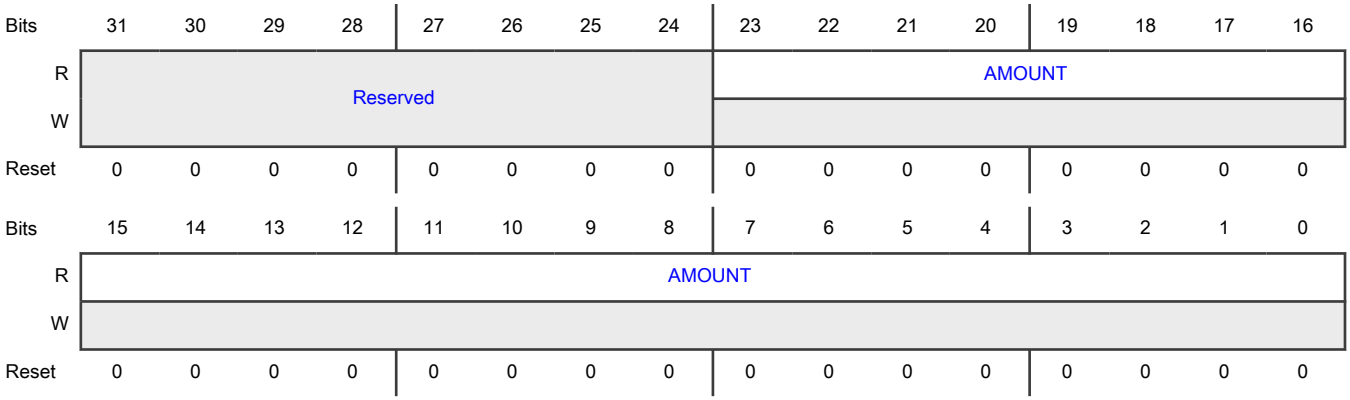
53.4.6.6.13 Port receive memory buffer entitlement register (PRXMBER)

Offset

Register	Offset
PRXMBER	120h

Function
This is the port receive memory buffer entitlement register.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 AMOUNT	Receive memory buffer entitlement in words This receive Memory Buffer Entitlement is used in determining smart drop for ingress congestion control. If a smart drop is to be performed, due receive memory buffer depletion, the receive queue that exceeds

Table continues on the next page...

Table continued from the previous page...

Field	Function
	entitlement amount the most will be selected for discard. Default value of 0 for this register results in fair share of the receive buffer memory across all ENETC instances. This value is writable in IERB (by pre-boot initialization), and is immediately reflected here. This value is R/W in the IERB space, and R/O by the ENETC function.

53.4.6.6.14 Port receive memory buffer limit register (PRXMBLR)

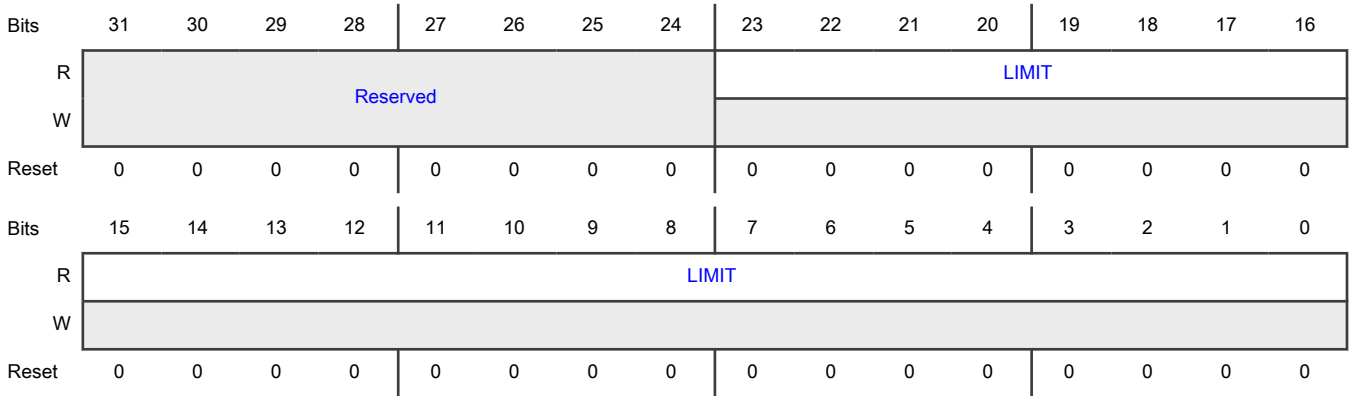
Offset

Register	Offset
PRXMBLR	124h

Function

This is the port receive memory buffer limit register.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 LIMIT	Receive buffer memory limit in words. This limit is used by the per-ENETC receive queues implemented for ingress congestion control. Ingress frames will be discarded if queue length exceeds this limit. Value of 0 means the number of memory words used within a port's priority tier is limited to 16k-1. This value is writable in IERB (by pre-boot initialization), and is immediately reflected here. This value is R/W in the IERB space, and R/O by the ENETC function.

53.4.6.6.15 Port receive buffer count register (PRXBCR)

Offset

Register	Offset
PRXBCR	128h

Function
This is the port receive buffer count register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								COUNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 COUNT	Current receive buffer usage count in words for this port.

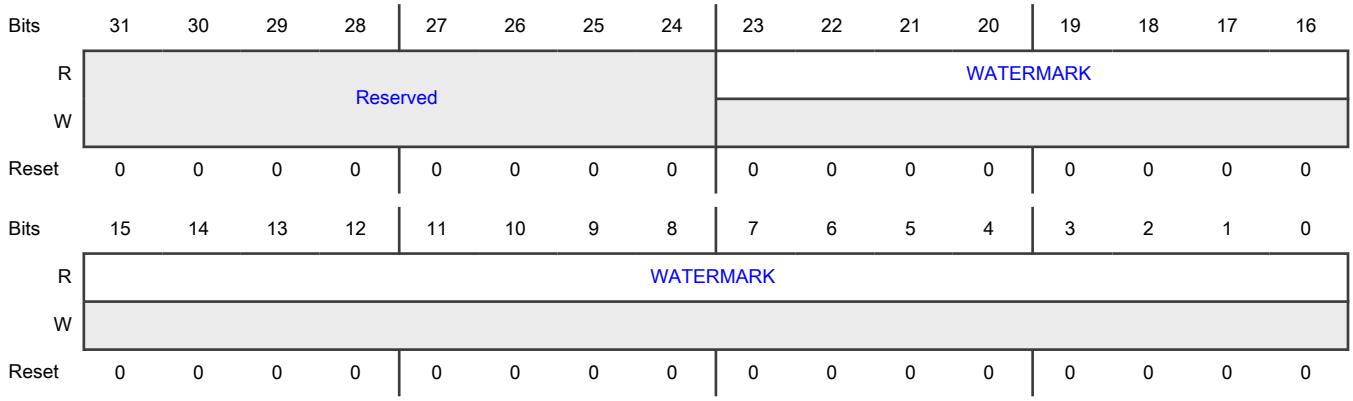
53.4.6.6.16 Port receive buffer count high watermark register (PRXBCHWMR)

Offset

Register	Offset
PRXBCHWMR	12Ch

Function
This is the port receive buffer count high watermark register.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 WATERMARK	High watermark in words for receive buffer usage since the last read of this register. After a read, this register is set equal to PRXBCR.

53.4.6.6.17 Port ingress congestion DR0 discard count register (PICDR0DCR)

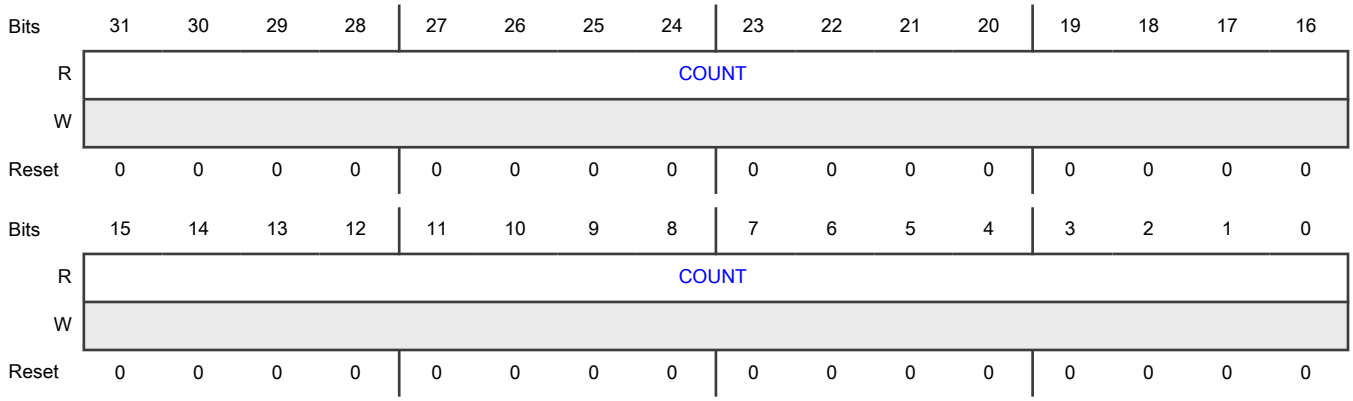
Offset

Register	Offset
PICDR0DCR	140h

Function

This is the port ingress congestion DR0 discard count register.

Diagram



Fields

Field	Function
31-0 COUNT	Number of frames discarded by Ingress Congestion Manager, for this port, and for each discard resilience (DR) value. A separate register is provided for each DR value. Reading this register provides the count of all frames discarded at this DR level, since the last reset of this count, without resetting the count. Resetting the count after the read can be done by reading the associated PICDR*a*DCRR register instead of this one. The discard count is incremented by the occurrence of the following events: • Port receive buffer limit exceeded • Internal memory congestion

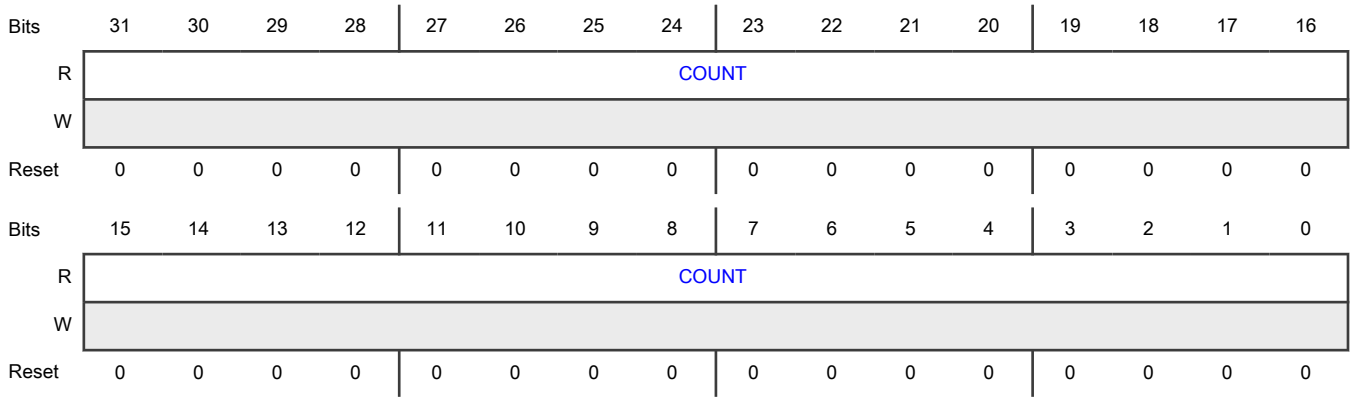
53.4.6.6.18 Port ingress congestion DRa discard count read-reset register (PICDR0DCRRR - PICDR3DCRRR)

Offset

Register	Offset
PICDR0DCRRR	148h
PICDR1DCRRR	158h
PICDR2DCRRR	168h
PICDR3DCRRR	178h

Function
This is the port ingress congestion DR discard count read-reset register.

Diagram



Fields

Field	Function
31-0 COUNT	Reading this register provides the count of all frames discarded at this DR level, since the last reset of this count, and also resets the count after the read. Other than the count reset, this register works the same as PICDR*a*DCR. After a read operation, the field's value clears to 0.

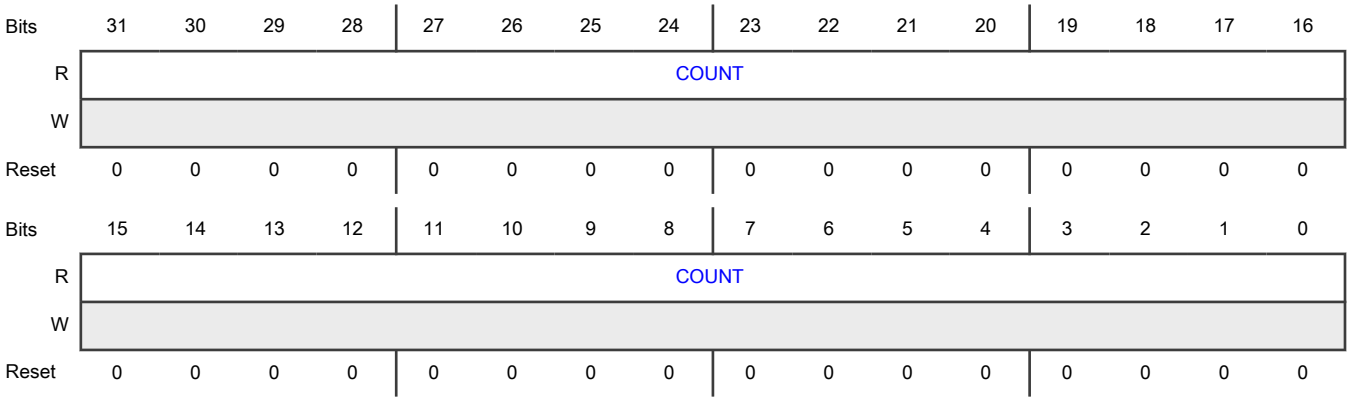
53.4.6.6.19 Port ingress congestion DR1 discard count register (PICDR1DCR)

Offset

Register	Offset
PICDR1DCR	150h

Function
This is the port ingress congestion DR1 discard count register.

Diagram



Fields

Field	Function
31-0 COUNT	Number of frames discarded by Ingress Congestion Manager, for this port, and for each discard resilience (DR) value. A separate register is provided for each DR value. Reading this register provides the count of all frames discarded at this DR level, since the last reset of this count, without resetting the count. Resetting the count after the read can be done by reading the associated PICDR*a*DCRR register instead of this one. The discard count is incremented by the occurrence of the following events: • Port receive buffer limit exceeded • Internal memory congestion

53.4.6.6.20 Port ingress congestion DR2 discard count register (PICDR2DCR)

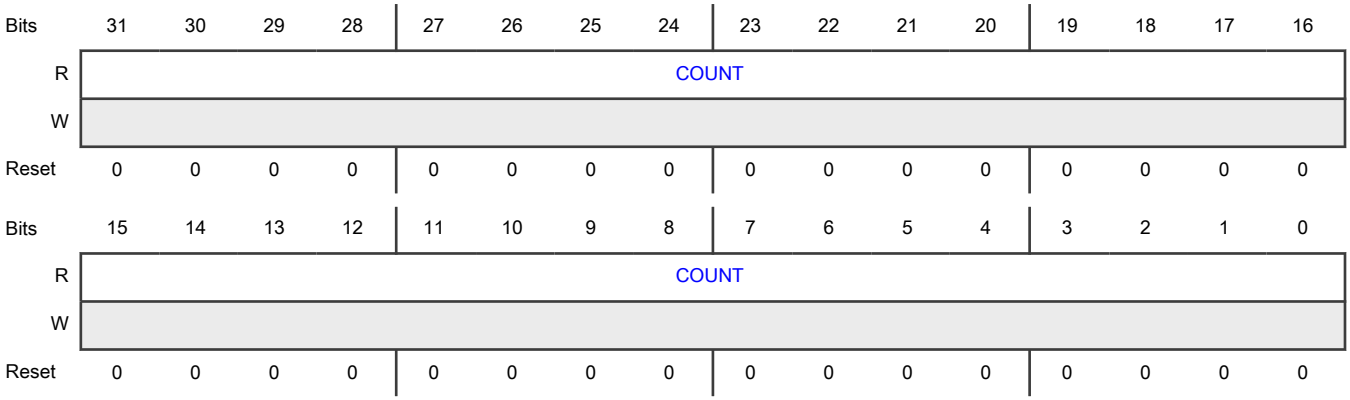
Offset

Register	Offset
PICDR2DCR	160h

Function

This is the port ingress congestion DR2 discard count register.

Diagram



Fields

Field	Function
31-0 COUNT	Number of frames discarded by Ingress Congestion Manager, for this port, and for each discard resilience (DR) value. A separate register is provided for each DR value. Reading this register provides the count of all frames discarded at this DR level, since the last reset of this count, without resetting the count. Resetting the count after the read can be done by reading the associated PICDR*a*DCRR register instead of this one. The discard count is incremented by the occurrence of the following events: • Port receive buffer limit exceeded • Internal memory congestion

53.4.6.6.21 Port ingress congestion DR3 discard count register (PICDR3DCR)

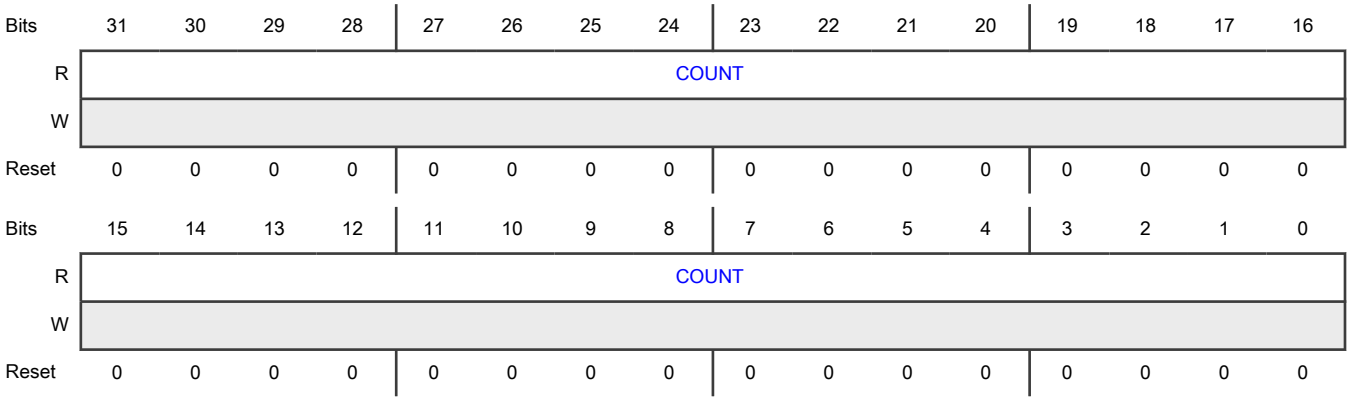
Offset

Register	Offset
PICDR3DCR	170h

Function

This is the port ingress congestion DR3 discard count register.

Diagram



Fields

Field	Function
31-0 COUNT	Number of frames discarded by Ingress Congestion Manager, for this port, and for each discard resilience (DR) value. A separate register is provided for each DR value. Reading this register provides the count of all frames discarded at this DR level, since the last reset of this count, without resetting the count. Resetting the count after the read can be done by reading the associated PICDR*a*DCRR register instead of this one. The discard count is incremented by the occurrence of the following events: • Port receive buffer limit exceeded • Internal memory congestion

53.4.6.6.22 Port ingress congestion priority discard status register (PICPDSR)

Offset

Register	Offset
PICPDSR	180h

Function

This is the port ingress congestion priority discard status register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			DR3_P1DS	Reserved			DR3_P0DS	Reserved			DR2_P1DS	Reserved			DR2_P0DS
W				W1C				W1C				W1C				W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			DR1_P1DS	Reserved			DR1_P0DS	Reserved			DR0_P1DS	Reserved			DR0_P0DS
W				W1C				W1C				W1C				W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DR3_P1DS	DR3 priority 1 discard status The bit will be set to 1 if one or more frames have been discarded at this DR and priority. The bit is cleared by a write to this register with a 1 in that bit position (write 1 to clear).
27-25 —	Reserved
24 DR3_P0DS	DR3 priority 0 discard status The bit will be set to 1 if one or more frames have been discarded at this DR and priority. The bit is cleared by a write to this register with a 1 in that bit position (write 1 to clear).
23-21 —	Reserved
20 DR2_P1DS	DR2 priority 1 discard status The bit will be set to 1 if one or more frames have been discarded at this DR and priority. The bit is cleared by a write to this register with a 1 in that bit position (write 1 to clear).
19-17 —	Reserved
16 DR2_P0DS	DR2 priority 0 discard status The bit will be set to 1 if one or more frames have been discarded at this DR and priority. The bit is cleared by a write to this register with a 1 in that bit position (write 1 to clear).
15-13	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
12 DR1_P1DS	DR1 priority 1 discard status The bit will be set to 1 if one or more frames have been discarded at this DR and priority. The bit is cleared by a write to this register with a 1 in that bit position (write 1 to clear).
11-9 —	Reserved
8 DR1_P0DS	DR1 priority 0 discard status The bit will be set to 1 if one or more frames have been discarded at this DR and priority. The bit is cleared by a write to this register with a 1 in that bit position (write 1 to clear).
7-5 —	Reserved
4 DR0_P1DS	DR0 priority 1 discard status The bit will be set to 1 if one or more frames have been discarded at this DR and priority. The bit is cleared by a write to this register with a 1 in that bit position (write 1 to clear).
3-1 —	Reserved
0 DR0_P0DS	DR0 priority 0 discard status The bit will be set to 1 if one or more frames have been discarded at this DR and priority. The bit is cleared by a write to this register with a 1 in that bit position (write 1 to clear).

53.4.6.6.23 Port station interface promiscuous MAC mode register (PSIPMMR)

Offset

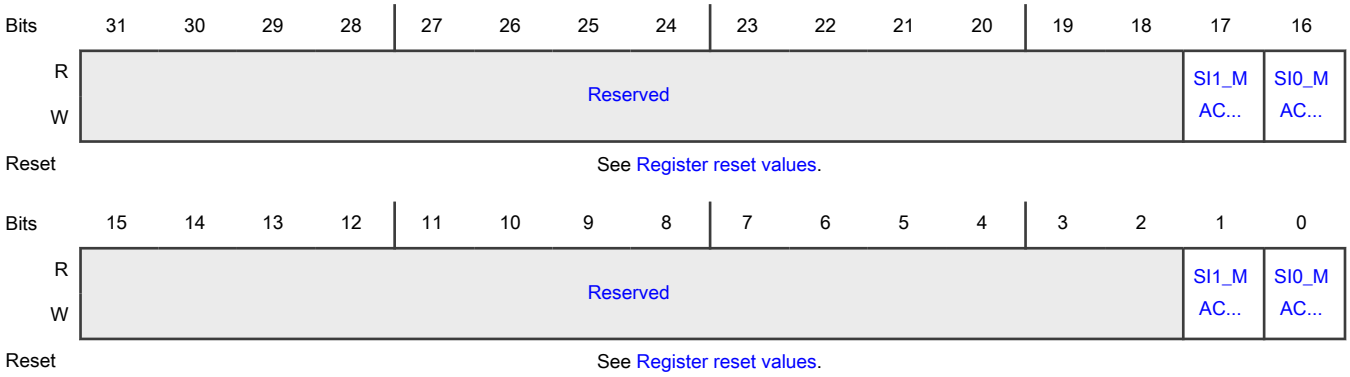
Register	Offset
PSIPMMR	200h

Function

This is the SI promiscuous MAC mode register which has settings for all the SIs associated with the port.

PF (PSI) FLR resets all bits to 0, except for SI0_MAC_UP and SI0_MAC_MP. VF (VSI) FLR resets the relevant VSI bits only.

Diagram



Register reset values

Register	Reset value
PSIPMMR	ENETC0_BASE: 0001_0001h ENETC1_BASE: 0003_0003h

Fields

Field	Function
31-18 —	Reserved
17 SI1_MAC_MP	SI 1 MAC multicast promiscuous. If MAC multicast promiscuous is enabled, SI 1 qualifies for reception of multicast frames regardless of their MAC address. **NOTE** This field is not supported in every instance. The following table includes only supported registers. 0b - Disabled 1b - Enabled
16 SI0_MAC_MP	SI 0 MAC multicast promiscuous. If MAC multicast promiscuous is enabled, SI 0 qualifies for reception of multicast frames regardless of their MAC address.

Instance	Field supported in	Field not supported in
ENETC0_BASE	—	PSIPMMR
ENETC1_BASE	PSIPMMR	—

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - Disabled
1b - Enabled</td></tr>
<tr><td>15-2
—</td><td>Reserved</td></tr>
<tr><td>1
SI1_MAC_UP</td><td>SI 1 MAC unicast promiscuous.
If MAC unicast promiscuous is enabled, SI 1 qualifies for reception of unicast frames regardless of their MAC address.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>ENETC0_BASE</td><td>—</td><td>PSIPMMR</td></tr><tr><td>ENETC1_BASE</td><td>PSIPMMR</td><td>—</td></tr></table>
0b - Disabled
1b - Enabled</td></tr>
<tr><td>0
SI0_MAC_UP</td><td>SI 0 MAC unicast promiscuous.
If MAC unicast promiscuous is enabled, SI 0 qualifies for reception of unicast frames regardless of their MAC address.
0b - Disabled
1b - Enabled</td></tr>
</table>

53.4.6.6.24 Port station interface promiscuous VLAN mode register (PSIPVMR)

Offset

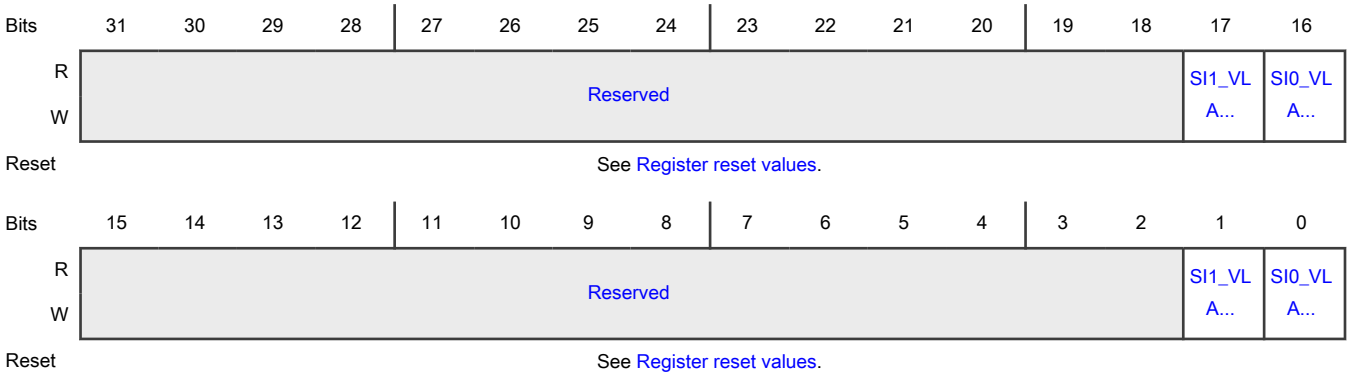
Register	Offset
PSIPVMR	204h

Function

This is the SI promiscuous VLAN mode register which have settings for all SIs associated with the port.

PF (PSI) FLR resets all bits to 0, except for SI0_VLAN_P and SI0_VLAN_UTA. VF (VSI) FLR resets the relevant VSI bits only.

Diagram



Register reset values

Register	Reset value
PSIPVMR	ENETC0_BASE: 0001_0001h ENETC1_BASE: 0003_0003h

Fields

Field	Function
31-18 —	Reserved
17 SI1_VLAN_UTA	SI 1 Untagged frames (no VLAN) accepted. This field specifies if SI 1 qualifies for reception of untagged frames. If no VLAN id is present for the purpose of VLAN filtering, this flag controls whether the frame may be received by a SI. Conditions under which no VLAN id is present are: • Frame is untagged and there is no port default VLAN configured for the selected VLAN tag. • Frame is tagged but the selected VLAN tag is deemed not present (e.g. TPID mismatched, selected tag(i.e. inner) not present in the frame) and there is no port default VLAN configured for the selected VLAN tag. If SI-based VLAN is used (along with being removed when frame is presented to software), untagged frames are not expected and thus it is recommended this flag be set to 0 (dropped untagged frames). **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
ENETC0_BASE	—	PSIPVMR
ENETC1_BASE	PSIPVMR	—

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - SI 1 does not qualify for reception of untagged frames
1b - SI 1 does qualify for reception of untagged frames</td></tr>
<tr><td>16
SI0_VLAN_UTA</td><td>SI 0 Untagged frames (no VLAN) accepted.
This field specifies if SI 0 qualifies for reception of untagged frames. If no VLAN id is present for the purpose of VLAN filtering, this flag controls whether the frame may be received by a SI. Conditions under which no VLAN id is present are:
• Frame is untagged and there is no port default VLAN configured for the selected VLAN tag.
• Frame is tagged but the selected VLAN tag is deemed not present (e.g. TPID mismatched, selected tag(i.e. inner) not present in the frame) and there is no port default VLAN configured for the selected VLAN tag.
If SI-based VLAN is used (along with being removed when frame is presented to software), untagged frames are not expected and thus it is recommended this flag be set to 0 (dropped untagged frames).
0b - SI 0 does not qualify for reception of untagged frames
1b - SI 0 does qualify for reception of untagged frames</td></tr>
<tr><td>15-2
—</td><td>Reserved</td></tr>
<tr><td>1
SI1_VLAN_P</td><td>SI 1 VLAN promiscuous.
This field specifies if SI 1 qualifies for the reception of all VLAN tags.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>ENETC0_BASE</td><td>—</td><td>PSIPVMR</td></tr>
<tr><td>ENETC1_BASE</td><td>PSIPVMR</td><td>—</td></tr>
</table>
0b - SI 1 does not qualify for the reception of all VLAN tags
1b - SI 1 does qualify for the reception of all VLAN tags</td></tr>
<tr><td>0
SI0_VLAN_P</td><td>SI 0 VLAN promiscuous.
This field specifies if SI 0 qualifies for the reception of all VLAN tags.
0b - SI 0 does not qualify for the reception of all VLAN tags
1b - SI 0 does qualify for the reception of all VLAN tags</td></tr>
</table>

53.4.6.6.25 Port broadcast frames dropped due to MAC filtering register (PBFDSIR)

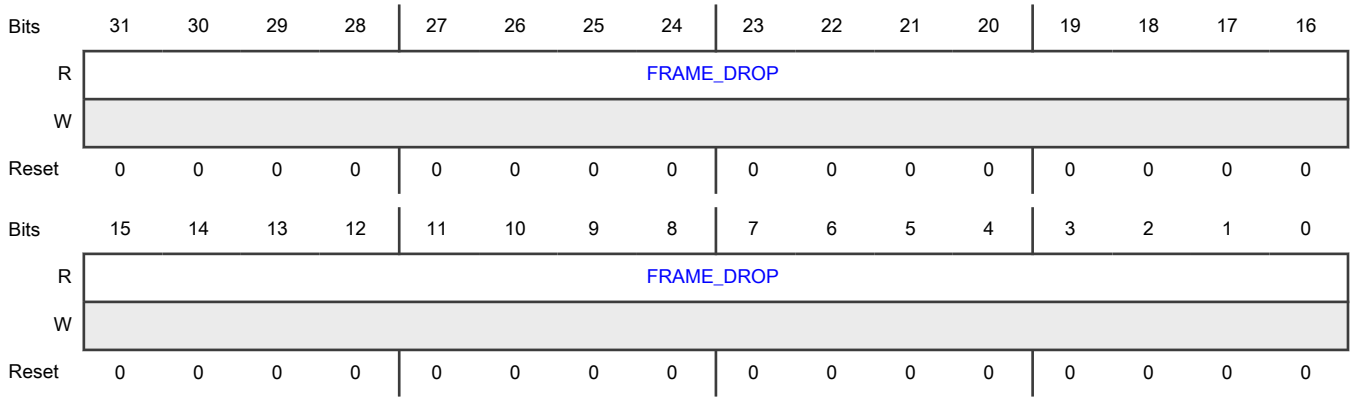
Offset

Register	Offset
PBFDSIR	208h

Function

This is the classification frame dropped due to MAC filtering counter. Counter is incremented when a broadcast frame is dropped due to MAC filtering.

Diagram



Fields

Field	Function
31-0 FRAME_DROP	MAC filter broadcast frame drop counter bits 31-0. Incremented for each dropped broadcast frame by the L2 filtering function, as no SI qualifies for reception of the frame due to all SI's having broadcast reject enabled, unless the reason for yielding no SIs is MAC source address pruning, in which case the PFDMSAPR counter is incremented instead.

53.4.6.6.26 Port frame drop MAC source address pruning register (PFDMSAPR)

Offset

Register	Offset
PFDMSAPR	20Ch

Function

This is the classification frame dropped due to MAC source address pruning counter. Counter is incremented when a frame is dropped due to MAC source address pruning.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FRAME_DROP															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FRAME_DROP															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 FRAME_DROP	MAC source address pruning frame drop counter bits 31-0. Incremented for each dropped frame by the L2 filtering function, as no SI qualifies for reception of the frame due to MAC source address pruning.

53.4.6.6.27 Port station interface MAC address filtering capability register (PSIMAFCAPR)

Offset

Register	Offset
PSIMAFCAPR	280h

Function

This is the station interface MAC address filtering capability register which reflects the number of MAC address filter table entries supported.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				NUM_MAC_AFTE											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 NUM_MAC_AFTE	Number of SI MAC address filter rules per port.

53.4.6.6.28 Port unicast frames dropped due to MAC filtering register (PUFDMFR)

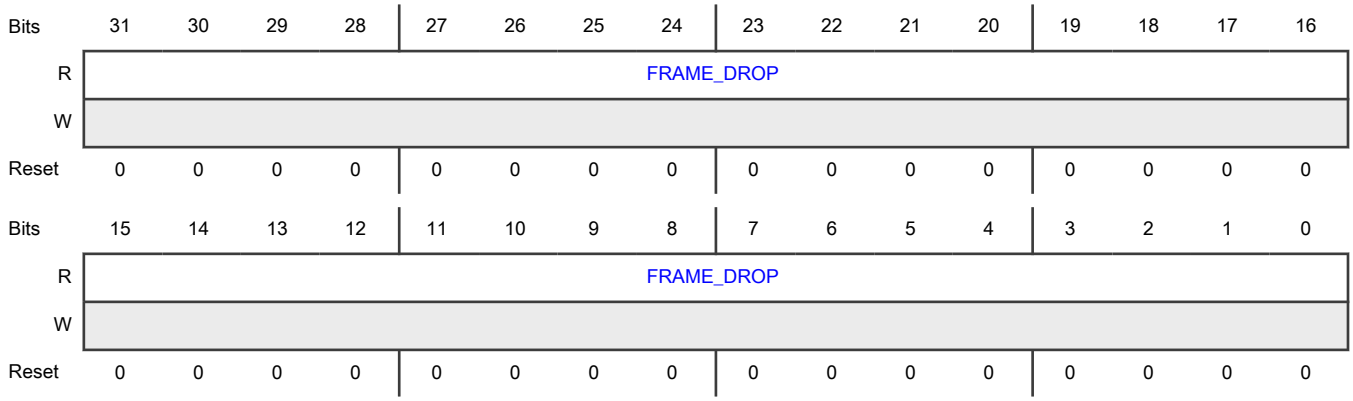
Offset

Register	Offset
PUFDMFR	284h

Function

This is the classification frame dropped due to MAC filtering counter. Counter is incremented when a unicast frame is dropped due to MAC filtering.

Diagram



Fields

Field	Function
31-0 FRAME_DROP	MAC filter unicast frame drop counter bits 31-0. Incremented for each dropped unicast frame by the L2 filtering function, as no SI qualifies for reception of the frame due to MAC filtering, unless the reason for yielding no SIs is MAC source address pruning, in which case the PFDMSAPR counter is incremented instead.

53.4.6.6.29 Port multicast frames dropped due to MAC filtering register (PMFDMFR)

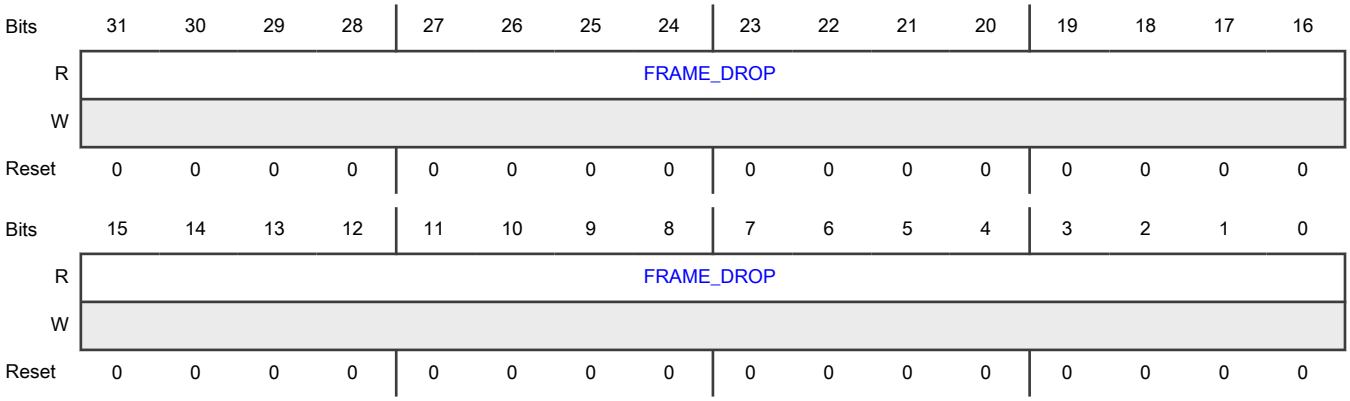
Offset

Register	Offset
PMFDMFR	288h

Function

This is the classification frame dropped due to MAC filtering counter. Counter is incremented when a multicast frame is dropped due to MAC filtering.

Diagram



Fields

Field	Function
31-0 FRAME_DROP	MAC filter multicast frame drop counter bits 31-0. Incremented for each dropped multicast frame by the L2 filtering function, as no SI qualifies for reception of the frame due to MAC filtering, unless the reason for yielding no SIs is MAC source address pruning, in which case the PFDMSAPR counter is incremented instead.

53.4.6.6.30 Port station interface VLAN filtering capability register (PSIVLANFCAPR)

Offset

Register	Offset
PSIVLANFCAPR	2C0h

Function

This is the station interface VLAN filtering capability register which reflects the number of VLAN filter table entries supported.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				NUM_VLAN_FTE											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-12 —	Reserved
11-0 NUM_VLAN_FTE	Number of VLAN filter table entries per port. Range: 0..4K-1

53.4.6.6.31 Port station interface VLAN filtering mode register (PSIVLANFMR)

Offset

Register	Offset
PSIVLANFMR	2C4h

Function

This is the station interface VLAN filtering mode register which controls the function.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															VS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 VS	VLAN tag select 0b - Inner VLAN tag will be used for VLAN filtering 1b - Outer VLAN tag will be used for VLAN filtering

53.4.6.6.32 Port unicast frames dropped VLAN filtering register (PUFDVFR)

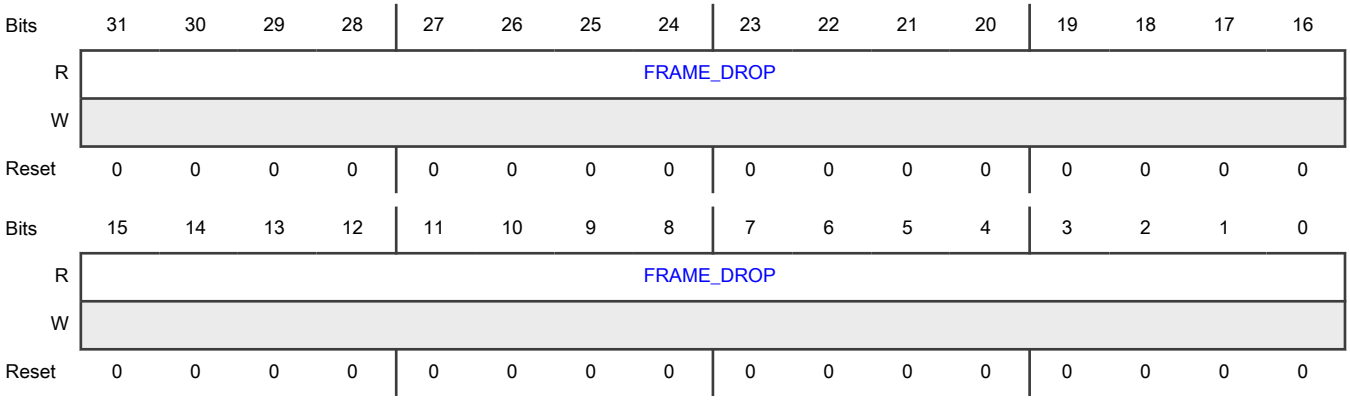
Offset

Register	Offset
PUFDVFR	2D0h

Function

This is the classification frame dropped due to VLAN filtering counter. Counter is incremented when a unicast frame is dropped due to VLAN filtering (i.e. frames that had a match for MAC filtering but didn't pass VLAN filtering).

Diagram



Fields

Field	Function
31-0 FRAME_DROP	VLAN filter unicast frame drop counter bits 31-0. Incremented for each dropped unicast frame by the L2 filtering function, as no SI qualifies for reception of the frame due to VLAN filtering.

53.4.6.6.33 Port multicast frames dropped VLAN filtering register (PMFDVFR)

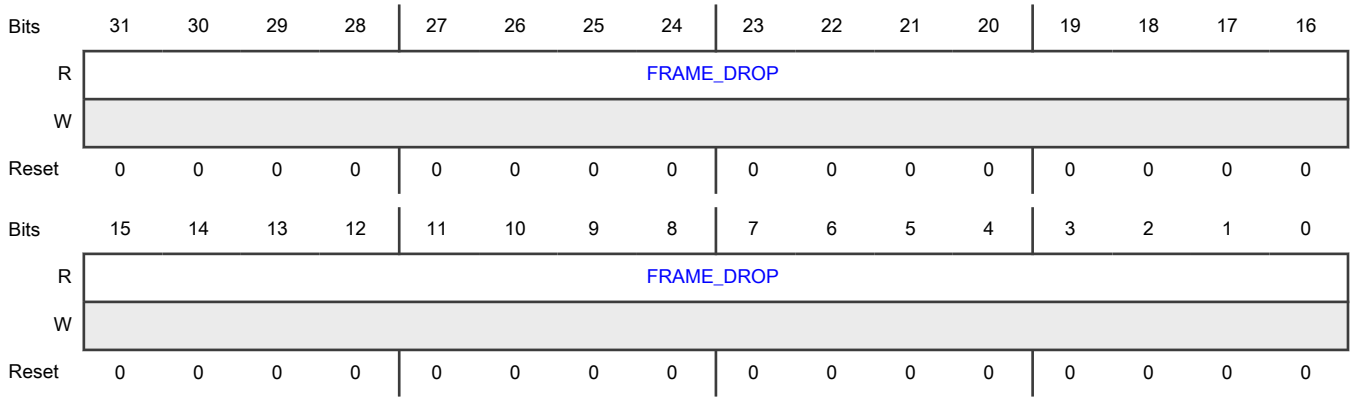
Offset

Register	Offset
PMFDVFR	2D4h

Function

This is the classification frame dropped due to VLAN filtering counter. Counter is incremented when a multicast frame is dropped due to VLAN filtering (i.e. frames that had a match for MAC filtering but didn't pass VLAN filtering).

Diagram



Fields

Field	Function
31-0 FRAME_DROP	VLAN filter multicast frame drop counter bits 31-0. Incremented for each dropped multicast frame by the L2 filtering function, as no SI qualifies for reception of the frame due to VLAN filtering.

53.4.6.6.34 Port broadcast frames dropped VLAN filtering register (PBFDVFR)

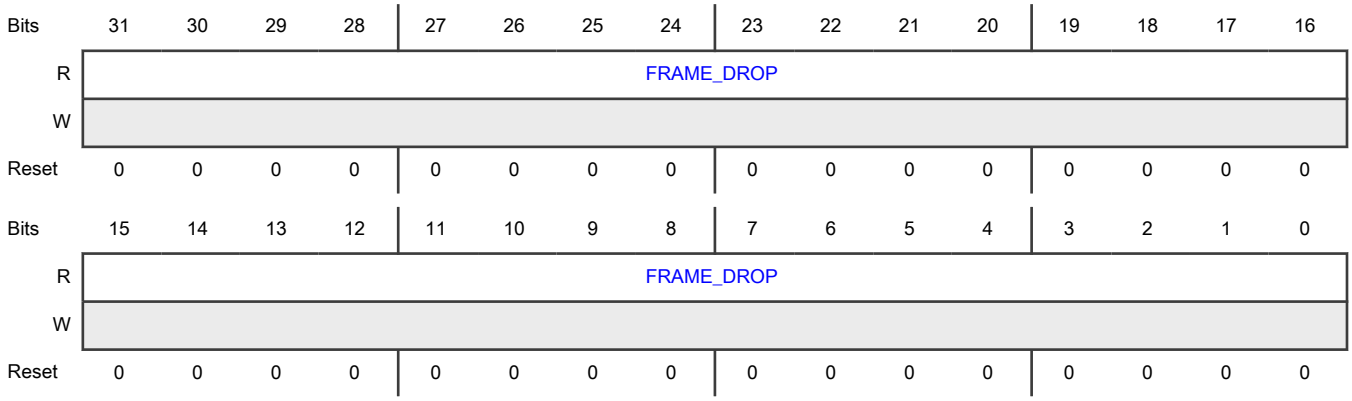
Offset

Register	Offset
PBFDVFR	2D8h

Function

This is the classification frame dropped due to VLAN filtering counter. Counter is incremented when a broadcast frame is dropped due to VLAN filtering (i.e. frames that had a match for MAC filtering but didn't pass VLAN filtering).

Diagram



Fields

Field	Function
31-0 FRAME_DROP	VLAN filter broadcast frame drop counter bits 31-0. Incremented for each dropped broadcast frame by the L2 filtering function, as no SI qualifies for reception of the frame due to VLAN filtering.

53.4.6.6.35 Port low power mode register (PLPMR)

Offset

Register	Offset
PLPMR	340h

Function

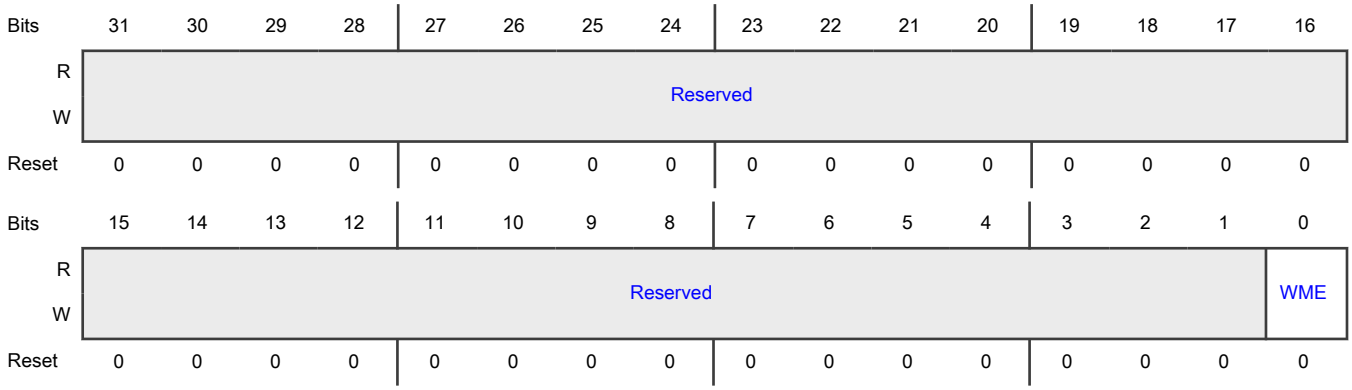
This is the low power mode register which provides the capability to enable the Wake-on-LAN feature. Enabling low power mode conveys to ENETC that low-power mode is about to be entered, and forces ENETC to progress to a quiesce state.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	PLPMR	—
ENETC1_BASE	—	PLPMR

Diagram



Fields

Field	Function
31-1 —	Reserved
0 WME	Wake-on-LAN mode enable When Wake-on-LAN mode is enabled, ENETC will detect Wake-on-LAN events. 0b - Disabled 1b - Enabled

53.4.6.6.36 Port wake-on status register (PWOSR)

Offset

Register	Offset
PWOSR	344h

Function

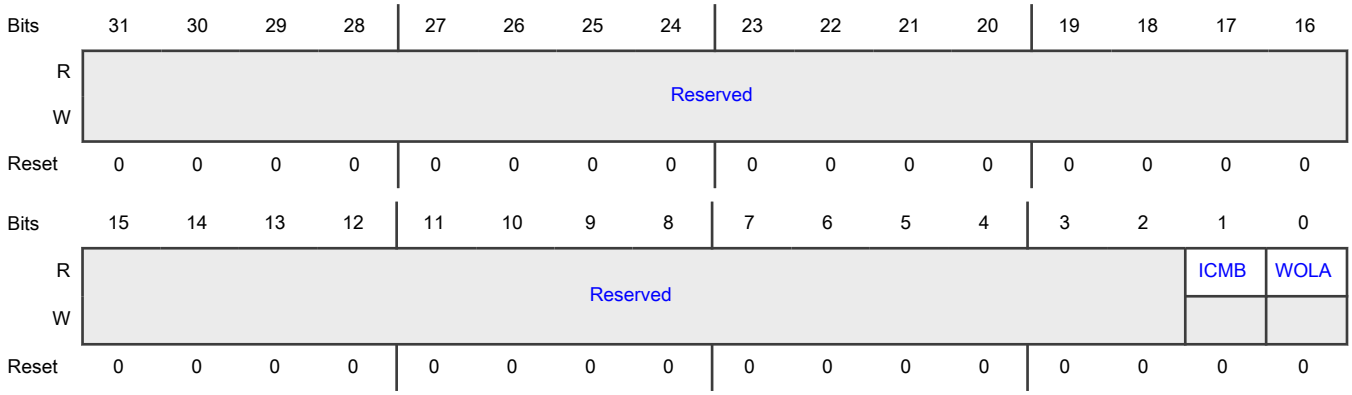
This is the Wake-on-LAN status register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	PWOSR	—
ENETC1_BASE	—	PWOSR

Diagram



Fields

Field	Function
31-2 —	Reserved
1 ICMB	ICM blocked 0b - Not blocked 1b - Blocked. ENETC has entered low-power mode and is blocking ICM from forwarding frames to host transfer agent until software exists low-power mode and PLPMR[WME]=0.
0 WOLA	Wake-On-LAN active 0b - Inactive 1b - Active. ENETC is actively searching for frames matching the Wake-on-LAN event criteria.

53.4.6.6.37 Receive IPV to ICM priority mapping register 0 (IPV2ICMPMR0)

Offset

Register	Offset
IPV2ICMPMR0	370h

Function

This is the receive IPV to ICM priority mapping register 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			IPV7ICM	Reserved			IPV6ICM	Reserved			IPV5ICM	Reserved			IPV4ICM
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			IPV3ICM	Reserved			IPV2ICM	Reserved			IPV1ICM	Reserved			IPV0ICM
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 IPV7ICM	Mapping of internal priority value (IPV) i={0..7} to ICM receive queue Priority. 0: Low Priority 1: High Priority
27-25 —	Reserved
24 IPV6ICM	Mapping of internal priority value (IPV) i={0..7} to ICM receive queue Priority. 0: Low Priority 1: High Priority
23-21 —	Reserved
20 IPV5ICM	Mapping of internal priority value (IPV) i={0..7} to ICM receive queue Priority. 0: Low Priority 1: High Priority
19-17 —	Reserved
16 IPV4ICM	Mapping of internal priority value (IPV) i={0..7} to ICM receive queue Priority. 0: Low Priority 1: High Priority
15-13	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
12 IPV3ICM	Mapping of internal priority value (IPV) i={0..7} to ICM receive queue Priority. 0: Low Priority 1: High Priority
11-9 —	Reserved
8 IPV2ICM	Mapping of internal priority value (IPV) i={0..7} to ICM receive queue Priority. 0: Low Priority 1: High Priority
7-5 —	Reserved
4 IPV1ICM	Mapping of internal priority value (IPV) i={0..7} to ICM receive queue Priority. 0: Low Priority 1: High Priority
3-1 —	Reserved
0 IPV0ICM	Mapping of internal priority value (IPV) i={0..7} to ICM receive queue Priority. 0: Low Priority 1: High Priority

53.4.6.6.38 Transmit priority to traffic class mapping register 0 (PRIO2TCMR0)

Offset

Register	Offset
PRIO2TCMR0	380h

Function

This register describes the transmit BD ring's priority to traffic class mapping for priority 0 to 7. Each priority should be mapped to a different TC (more than one priority shouldn't be mapped to the same TC). To support multiple transmit BD rings assigned to the same TC, the transmit BD rings should be assigned the same priority value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	PRIO7TC			Reserved	PRIO6TC			Reserved	PRIO5TC			Reserved	PRIO4TC		
Reset	0	1	1	1	0	1	1	0	0	1	0	1	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	PRIO3TC			Reserved	PRIO2TC			Reserved	PRIO1TC			Reserved	PRIO0TC		
Reset	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 PRIO7TC	Transmit BD ring priority to traffic class mapping
27 —	Reserved
26-24 PRIO6TC	Transmit BD ring priority to traffic class mapping
23 —	Reserved
22-20 PRIO5TC	Transmit BD ring priority to traffic class mapping
19 —	Reserved
18-16 PRIO4TC	Transmit BD ring priority to traffic class mapping
15 —	Reserved
14-12 PRIO3TC	Transmit BD ring priority to traffic class mapping

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 —	Reserved
10-8 PRIO2TC	Transmit BD ring priority to traffic class mapping
7 —	Reserved
6-4 PRIO1TC	Transmit BD ring priority to traffic class mapping
3 —	Reserved
2-0 PRIO0TC	Transmit BD ring priority to traffic class mapping

53.4.6.6.39 Port traffic class a time specific departure register (PTC0TSDR - PTC7TSDR)

Offset

For a = 0 to 7:

Register	Offset
PTCaTSDR	390h + (a × 4h)

Function

The traffic class time specific departure register enables the feature associated with the traffic class.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSDE	Reserved														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TSDE	Time specific Departure Enable **NOTE** The 1588 timer must be configured and enabled, and time gate scheduling for the port must be enabled (PTGSCR[TGE] set to 1), before enabling time specific departure on any traffic class. See IEEE 1588 timer module, for information on how to initialize and configure the 1588 timer. After the 1588 timer is in normal operation, any change to the 1588 timer configuration (for example in TMROFF_H/L), except for TMR_ADD updates, requires that time specific departure be disabled on all traffic classes. 0b - Disabled 1b - Enabled
30-0 —	Reserved

53.4.6.6.40 Switch management capability register (SMCAPR)

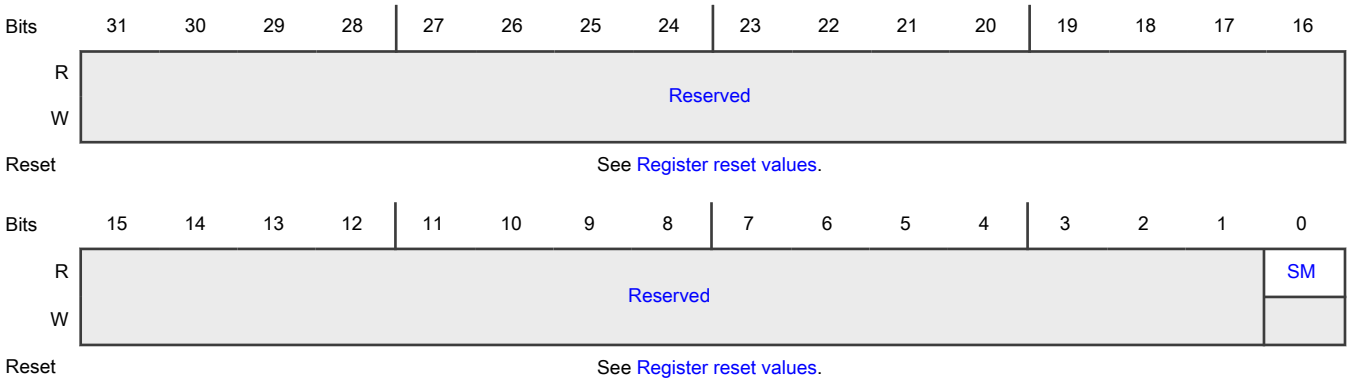
Offset

Register	Offset
SMCAPR	800h

Function

This is the switch management capability register.

Diagram



Register reset values

Register	Reset value
SMCAPR	ENETC0_BASE: 0000_0000h ENETC1_BASE: 0000_0001h

Fields

Field	Function
31-1 —	Reserved
0 SM	Switch Management **NOTE** Capability flag is set to 1 if the ENETC's port is connected to switch port via pseudo link and the switch has configured his management port to be this port. That is, if PCAPR[LINK_TYPE]=pseudo link and switch's Management Port Configuration Register MPCR[PORT] is the remote end of the ENETC's port. 0b - ENETC instance has no switch management capability 1b - ENETC instance has switch management capability

53.4.6.6.41 Switch management host reason a receive BD ring mapping register (SMHR1BDRMR - SMHR15BDRMR)

Offset

For a = 1 to 15:

Register	Offset
SMHRaBDRMR	87Ch + (a × 4h)

Function

This register applies to ENETC whose port is bound to a pseudo link (PCAPR[LINK_TYPE]=1) with the remote port being designated as a switch management port.

If the frame metadata field Host Reason is set to non-zero (i.e. not a regular forwarded frame), the BD ring is selected by mapping the Host Reason to a receive BD ring using this mapping register.

NOTE

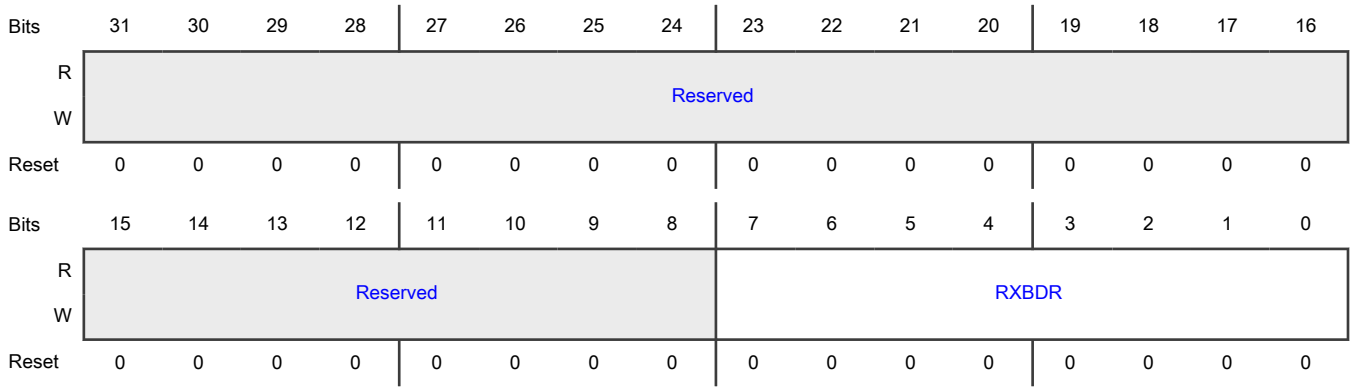
Valid if SMCAPR[SM]=1.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	—	SMHR1BDRMR–SMHR15BDRMR
ENETC1_BASE	SMHR1BDRMR–SMHR15BDRMR	—

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 RXBDR	Switch management (non-zero) host reason is directed to this receive BD ring. Note that frames with the metadata field Host Reason set to non-zero can only be received from SI 0.

53.4.6.6.42 Port station interface 0 primary MAC address register 0 (PSI0PMAR0)

Offset

Register	Offset
PSI0PMAR0	2000h

Function

This is the SI's primary MAC address register.

NOTE

This value is taken directly from the port PMAR0 register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRIM_MAC_ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRIM_MAC_ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PRIM_MAC_ADDR	Primary MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset (register bit offset 0-7). This field contains the 4 most significant bytes of the MAC address. For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then PSI*a*PMAR0 equals 14131211h.

53.4.6.6.43 Port station interface 0 primary MAC address register 1 (PSI0PMAR1)

Offset

Register	Offset
PSI0PMAR1	2004h

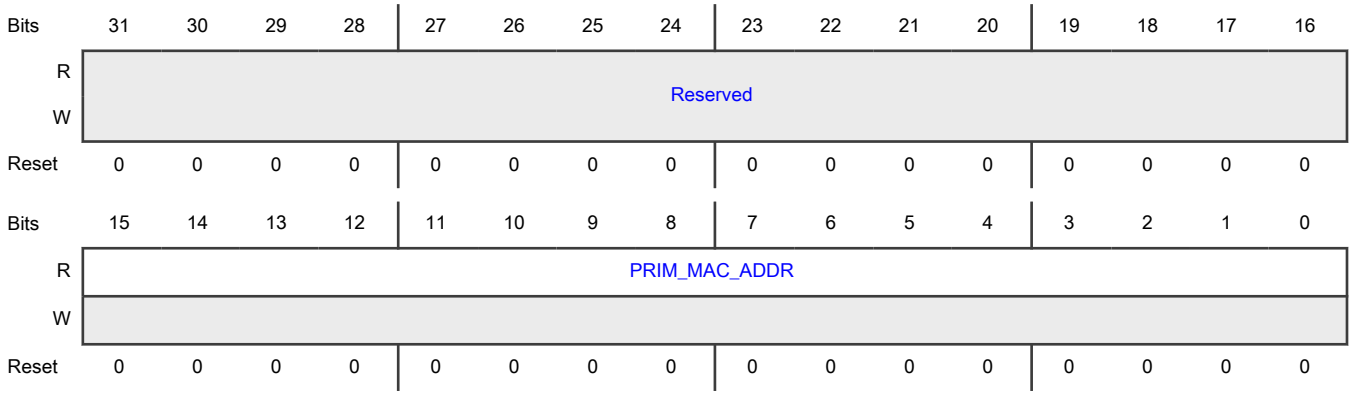
Function

This is the SI's primary MAC address register 1.

NOTE

This value is taken directly from the port PMAR1 register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 PRIM_MAC_ADDR	Primary MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset. This field contains the 2 least significant bytes of the MAC address (least significant byte is stored in register bit offset 8-15). For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then PSI*a*PMAR1 equals xxxx1615h (where x should be set to 0).

53.4.6.6.44 Port station interface a VLAN register (PSI0VLANR - PSI1VLANR)

Offset

Register	Offset
PSI0VLANR	2008h
PSI1VLANR	2088h

Function

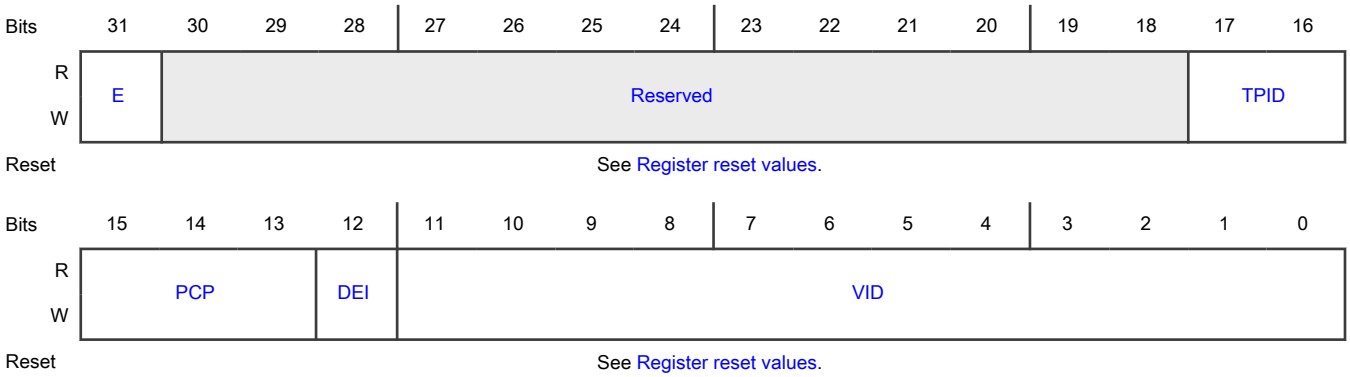
This is the SI-based VLAN. It is used for SI-based VLAN insertion (VLAN tag to be inserted) and SI-based VLAN removal (VLAN tag to be matched for removal)

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	PSI0VLANR	PSI1VLANR
ENETC1_BASE	PSI0VLANR–PSI1VLANR	—

Diagram



Register reset values

Register	Reset value
PSI0VLANR	ENETC0_BASE,ENETC1_BASE: 0000_0000h
PSI1VLANR	0000_0000h

Fields

Field	Function
31 E	Enable Enables and disables SI-based VLAN. 0b - Disabled 1b - Enabled; SI-based VLAN information is added on transmit and removed on receive.
30-18 —	Reserved
17-16 TPID	Tag protocol identifier 00b - Standard C-VLAN 0x8100 01b - Standard S-VLAN 0x88A8 10b - Custom VLAN as defined by CVLANR1[ETYPE]. Note that CVLANR1[V] is not checked for SI-based VLAN insertion; TPID value specified in CVLANR1[ETYPE] will be used to construct the VLAN header regardless of the value specified in CVLANR1[V].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Custom VLAN as defined by CVLANR2[ETYPE]. Note that CVLANR2[V] is not checked for SI-based VLAN insertion; TPID value specified in CVLANR2[ETYPE] will be used to construct the VLAN header regardless of the value specified in CVLANR2[V].
15-13 PCP	Priority code point A 3-bit field which refers to the IEEE 802.1p class of service and maps to the frame priority level.
12 DEI	Drop eligible indicator May be used separately or in conjunction with PCP to indicate frames eligible to be dropped in the presence of congestion.
11-0 VID	VLAN identifier The VLAN identifier is a 12-bit field specifying the VLAN to which the frame belongs.

53.4.6.6.45 Port station interface 0 configuration register 0 (PSI0CFGR0)

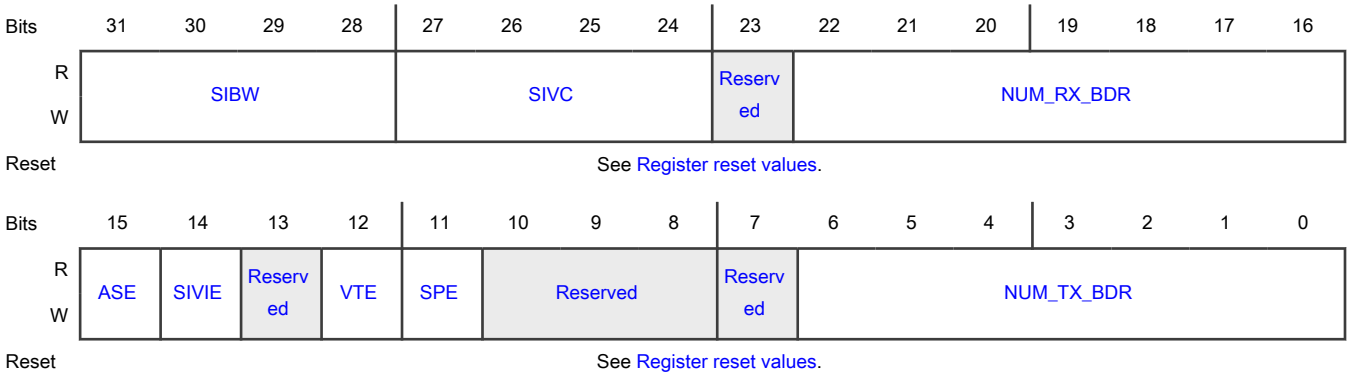
Offset

Register	Offset
PSI0CFGR0	2010h

Function

The station interface configuration register determines the features and capabilities of an SI.

Diagram



Register reset values

Register	Reset value
PSI0CFGR0	ENETC0_BASE: 0004_0004h ENETC1_BASE: 000A_000Ah

Fields

Field	Function
31-28 SIBW	Station interface traffic class bandwidth weight Frames are selected for transmission between station interfaces based on a per traffic class basis using the Weighted Fair Bandwidth Sharing algorithm. This value specifies a relative weight and is defined as credits per byte. A higher number indicates less bandwidth. For example, to balance the bandwidth for traffic class 7 between SI 0/1 to allow SI 0 twice the bandwidth, set the weight to 2 and 4 respectively. Note that a setting of 0x0 (reset default) will give an SI absolute priority over any SIs with a non-zero SIBW setting. If multiple SIs are configured with SIBW=0x0 then they each receive equal share of the traffic class bandwidth. 0: Highest bandwidth 1: Next highest bandwidth ... 14: Next lowest bandwidth 15: Lowest bandwidth
27-24 SIVC	Station interface VLAN control Determines which VLAN Ethertypes can be inserted by the SI driver (e.g. VF) either in the frame data or through the Tx BD. It also determines which VLAN Ethertypes are acceptable for VLAN removal (as seen by SI). Bit: 24 Standard C-VLAN 0x8100 25 Standard S-VLAN 0x88A8 26 Custom VLAN as defined by CVLANR1[ETYPE] 27 Custom VLAN as defined by CVLANR2[ETYPE]
23 —	Reserved
22-16 NUM_RX_BDR	Number of receive buffer descriptor rings assigned to the SI Range: 0..64 The following rules apply when modifying this field: • By default all rings are assigned to SI 0 (PSI) • Total number of assigned rings must not exceed the number of available rings defined by the port, ECAPR2[NUM_RX_BDR] • Assignment of rings to SIs should be done in increasing order, i.e. SI *n+1* cannot start ring management until SI *n* has been assigned rings • Reconfiguring number of assigned rings to SI *n* requires complete ring management reconfiguration for SI *n+1* and above
15	Anti-spoofing enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
ASE	When anti-spoofing is enabled, source MAC address used by SI is compared against the primary MAC address. If it is different, the frame is not transmitted and an error is reported in the status field of the BD. 0: Disabled 1: Enabled
14 SIVIE	SI-based VLAN Insertion Enable This field controls whether insertion of the SI-based VLAN into frames sent by this SI is enabled. 0b - SI-based VLAN insertion disabled 1b - SI-based VLAN insertion enabled
13 —	Reserved
12 VTE	VLAN Tag Extract This field controls whether SI-based VLAN tag is removed from the frame before delivery to the SI. This tag must be the first tag, immediately following the SMAC address to be recognized and removed. After extraction, the resulting frame is presented to the SI for processing. 0b - SI-based VLAN removal disabled 1b - SI-based VLAN removal enabled
11 SPE	Source Pruning Enable 0b - Disabled 1b - Enabled
10-8 —	Reserved
7 —	Reserved
6-0 NUM_TX_BDR	Number of transmit buffer descriptor rings assigned to the SI Range: 0..64 The following rules apply when modifying this field: • By default all rings are assigned to SI 0 (PSI) • Total number of assigned rings must not exceed the number of available rings defined by the port, ECAPR2[NUM_TX_BDR] • Assignment of rings to SIs should be done in increasing order, i.e. SI *n+1* cannot start ring management until SI *n* has been assigned rings • Reconfiguring number of assigned rings to SI *n* requires complete ring management reconfiguration for SI *n+1* and above

53.4.6.6.46 Port station interface 0 configuration register 2 (PSI0CFGR2)

Offset

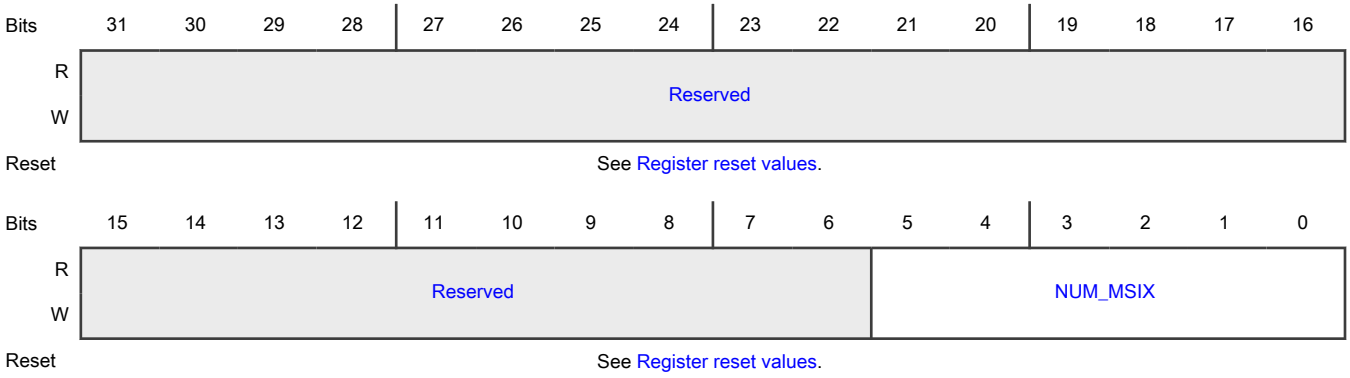
Register	Offset
PSI0CFGR2	2018h

Function

The station interface configuration register 2 determines the number of MSI-X table entries (vectors) allocated to a station interface.

The total number of vectors available to the ENETC instance is determined by ECAPR1[NUM_MSIX]. All vectors are initially assigned to the PSI, less 1 vector per VSI.

Diagram



Register reset values

Register	Reset value
PSI0CFGR2	ENETC0_BASE: 0000_000Bh ENETC1_BASE: 0000_0017h

Fields

Field	Function
31-6 —	Reserved
5-0 NUM_MSIX	Number of MSI-X Number of MSI-X vectors assigned to a Station Interface. Range: 1..64 Formula: NUM_MSIX+1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Reflected in SIPCAPR1[NUM_MSIX] read-only register.

53.4.6.6.47 Port station interface 0 VSI MAC address filtering configuration register (PSI0VMAFCFGR)

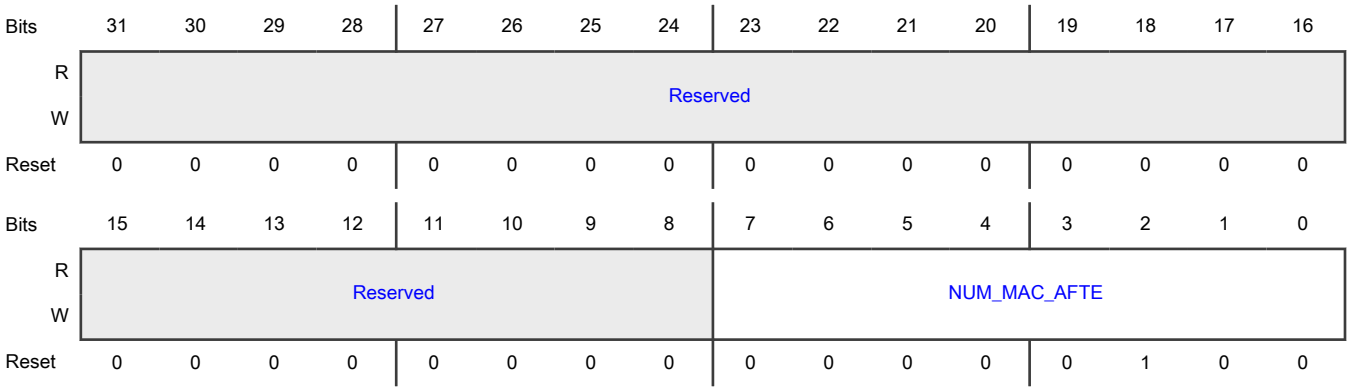
Offset

Register	Offset
PSI0VMAFCFGR	2030h

Function

The station interface MAC address filtering configuration register determines the capabilities of a SI.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 NUM_MAC_AFTE	Number of SI MAC address filter table entries assigned to the SI. Number of assigned entries to all SIs must not exceed the total number of available SI MAC address filter rules per port as defined by PSIMAFCAPR[NUM_MAC_AFTE]. This field should be initialized once at start-up, before SI capability discovery starts.

53.4.6.6.48 Port station interface 0 VLAN filtering configuration register (PSI0VLANFCFGR)

Offset

Register	Offset
PSI0VLANFCFGR	2034h

Function

The station interface VLAN filtering configuration register determines the capabilities of a SI.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								NUM_VLAN_FTE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 NUM_VLAN_FTE	Number of VLAN filter table entries assigned to the SI. Number of assigned entries to all SIs must not exceed the total number of available SI VLAN filter rules per port as defined by PSIVLANFCAPR[NUM_VLAN_FTE]. This field should be initialized once at start-up, before SI capability discovery starts. Modifying the setting during run-time may lead to undefined behavior.

53.4.6.6.49 Port station interface a unicast MAC hash filter register 0 (PSI0UMHFR0 - PSI1UMHFR0)

Offset

Register	Offset
PSI0UMHFR0	2050h
PSI1UMHFR0	20D0h

Function

This is the station interface unicast MAC hash filter register 0.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	PSI0UMHFR0	PSI1UMHFR0
ENETC1_BASE	PSI0UMHFR0–PSI1UMHFR0	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAC_HASH_FLT_LOW															
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAC_HASH_FLT_LOW															
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PSI0UMHFR0	ENETC0_BASE,ENETC1_BASE: 0000_0000h
PSI1UMHFR0	0000_0000h

Fields

Field	Function
31-0 MAC_HASH_FLT_LOW	Lower 32-bits (31-0) of unicast MAC hash filter table indexed by unicast MAC address hash.

53.4.6.6.50 Port station interface a unicast MAC hash filter register 1 (PSI0UMHFR1 - PSI1UMHFR1)

Offset

Register	Offset
PSI0UMHFR1	2054h
PSI1UMHFR1	20D4h

Function

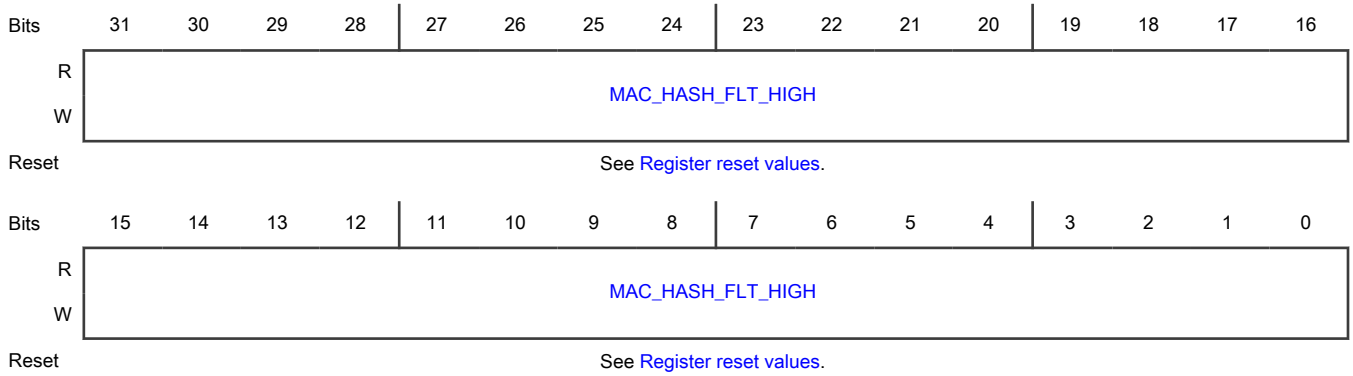
This is the station interface unicast MAC hash filter register 1.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	PSI0UMHFR1	PSI1UMHFR1
ENETC1_BASE	PSI0UMHFR1–PSI1UMHFR1	—

Diagram



Register reset values

Register	Reset value
PSI0UMHFR1	ENETC0_BASE,ENETC1_BASE: 0000_0000h
PSI1UMHFR1	0000_0000h

Fields

Field	Function
31-0	Upper 32-bits (63-32) of unicast MAC hash filter table indexed by unicast MAC address hash.

Table continues on the next page...

Field	Function
MAC_HASH_FLT_HIGH	

53.4.6.6.51 Port station interface a multicast MAC hash filter register 0 (PSI0MMHFR0 - PSI1MMHFR0)

Offset

Register	Offset
PSI0MMHFR0	2058h
PSI1MMHFR0	20D8h

Function

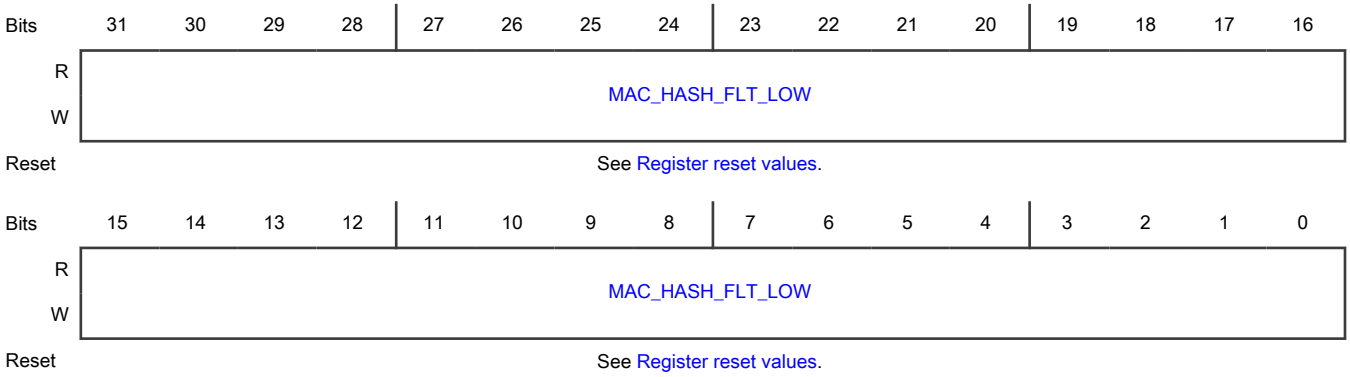
This is the station interface multicast MAC hash filter register 0.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	PSI0MMHFR0	PSI1MMHFR0
ENETC1_BASE	PSI0MMHFR0–PSI1MMHFR0	—

Diagram



Register reset values

Register	Reset value
PSI0MMHFR0	ENETC0_BASE,ENETC1_BASE: 0000_0000h
PSI1MMHFR0	0000_0000h

Fields

Field	Function
31-0 MAC_HASH_FLT_LOW	Lower 32-bits (31-0) of multicast MAC hash filter table indexed by unicast MAC address hash.

53.4.6.6.52 Port station interface a multicast MAC hash filter register 1 (PSI0MMHFR1 - PSI1MMHFR1)

Offset

Register	Offset
PSI0MMHFR1	205Ch
PSI1MMHFR1	20DCh

Function

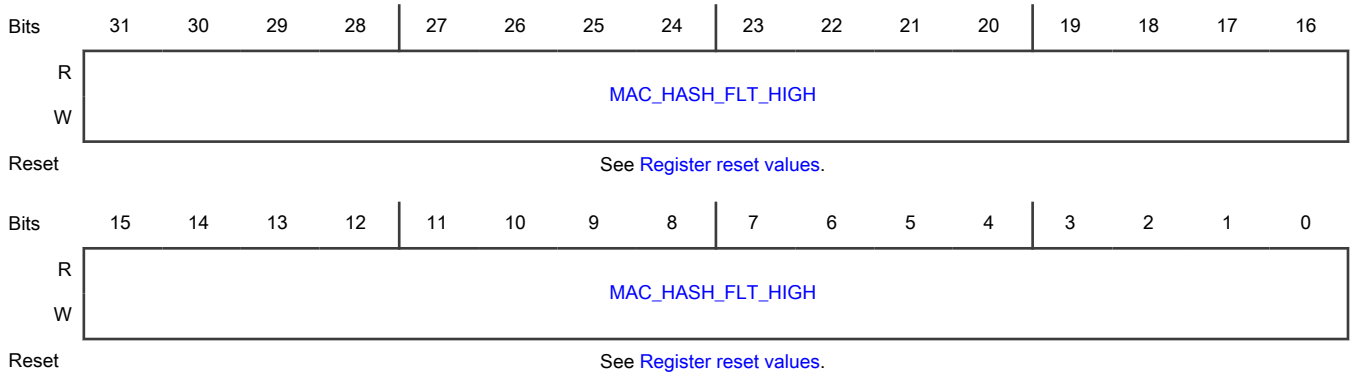
This is the station interface multicast MAC hash filter register 1.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	PSI0MMHFR1	PSI1MMHFR1
ENETC1_BASE	PSI0MMHFR1–PSI1MMHFR1	—

Diagram



Register reset values

Register	Reset value
PSI0MMHFR1	ENETC0_BASE,ENETC1_BASE: 0000_0000h
PSI1MMHFR1	0000_0000h

Fields

Field	Function
31-0 MAC_HASH_FLT_HIGH	Upper 32-bits (63-32) of multicast MAC hash filter table indexed by unicast MAC address hash.

53.4.6.6.53 Port station interface a VLAN hash filter register 0 (PSI0VHFR0 - PSI1VHFR0)

Offset

Register	Offset
PSI0VHFR0	2060h
PSI1VHFR0	20E0h

Function

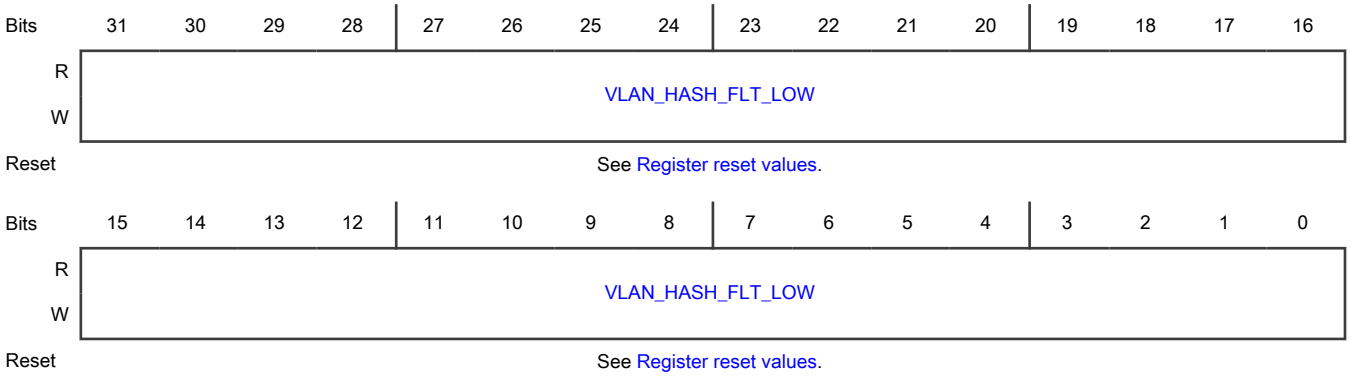
This is the station interface VLAN hash filter table register 0.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	PSI0VHFR0	PSI1VHFR0
ENETC1_BASE	PSI0VHFR0–PSI1VHFR0	—

Diagram



Register reset values

Register	Reset value
PSI0VHFR0	ENETC0_BASE,ENETC1_BASE: 0000_0000h
PSI1VHFR0	0000_0000h

Fields

Field	Function
31-0 VLAN_HASH_FLT_LOW	Lower 32-bits (31-0) of VLAN hash filter table indexed by VLAN hash.

53.4.6.6.54 Port station interface a VLAN hash filter register 1 (PSI0VHFR1 - PSI1VHFR1)

Offset

Register	Offset
PSI0VHFR1	2064h
PSI1VHFR1	20E4h

Function

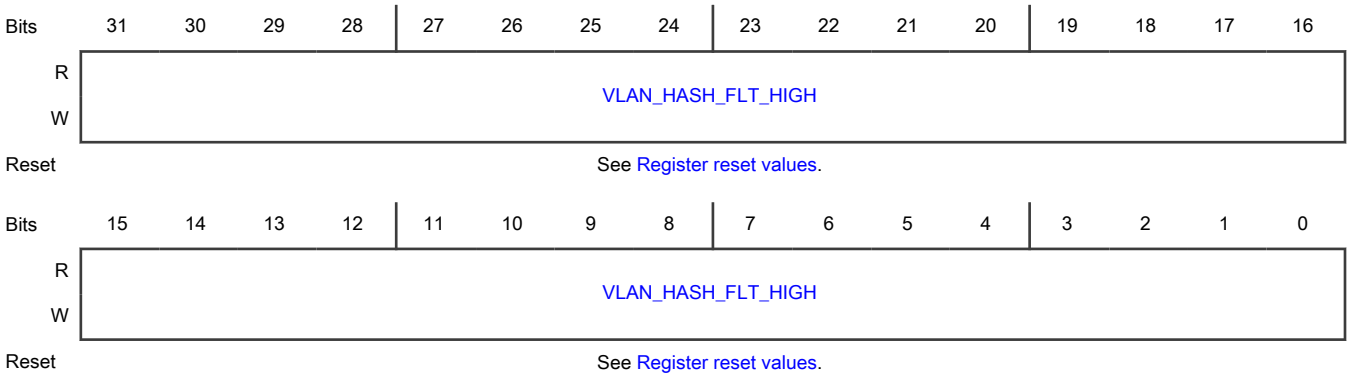
This is the station interface VLAN hash filter table register 1.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	PSI0VHFR1	PSI1VHFR1
ENETC1_BASE	PSI0VHFR1–PSI1VHFR1	—

Diagram



Register reset values

Register	Reset value
PSI0VHFR1	ENETC0_BASE,ENETC1_BASE: 0000_0000h
PSI1VHFR1	0000_0000h

Fields

Field	Function
31-0 VLAN_HASH_FLT_HIGH	Upper 32-bits (63-32) of VLAN hash filter table indexed by VLAN hash.

53.4.6.6.55 Port station interface 1 primary MAC address register 0 (PSI1PMAR0)

Offset

Register	Offset
PSI1PMAR0	2080h

Function

This is the SI's primary MAC address register.

NOTE

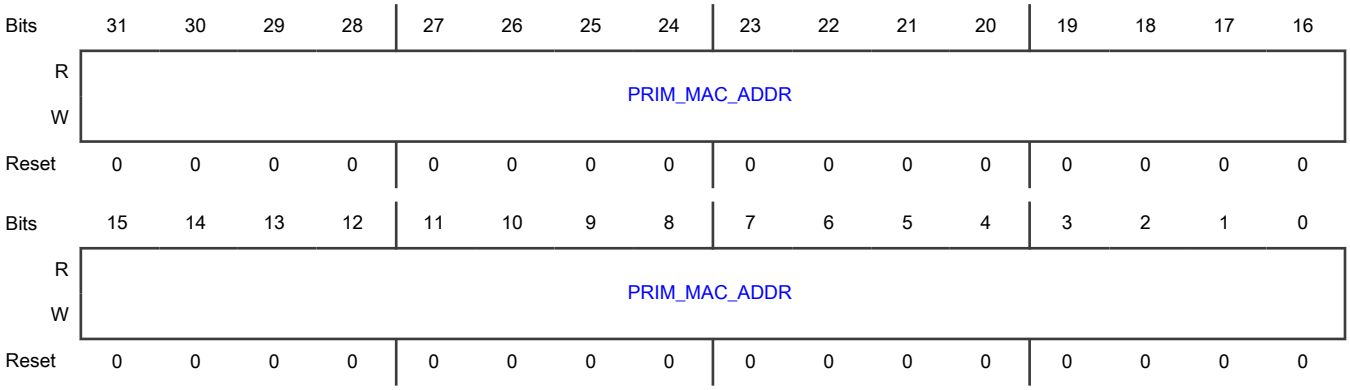
This value is taken directly from the port PMAR0 register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	—	PSI1PMAR0
ENETC1_BASE	PSI1PMAR0	—

Diagram



Fields

Field	Function
31-0 PRIM_MAC_ADDR	Primary MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset (register bit offset 0-7). This field contains the 4 most significant bytes of the MAC address. For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then PSI*a*PMAR0 equals 14131211h.

53.4.6.6.56 Port station interface 1 primary MAC address register 1 (PSI1PMAR1)

Offset

Register	Offset
PSI1PMAR1	2084h

Function

This is the SI's primary MAC address register 1.

NOTE

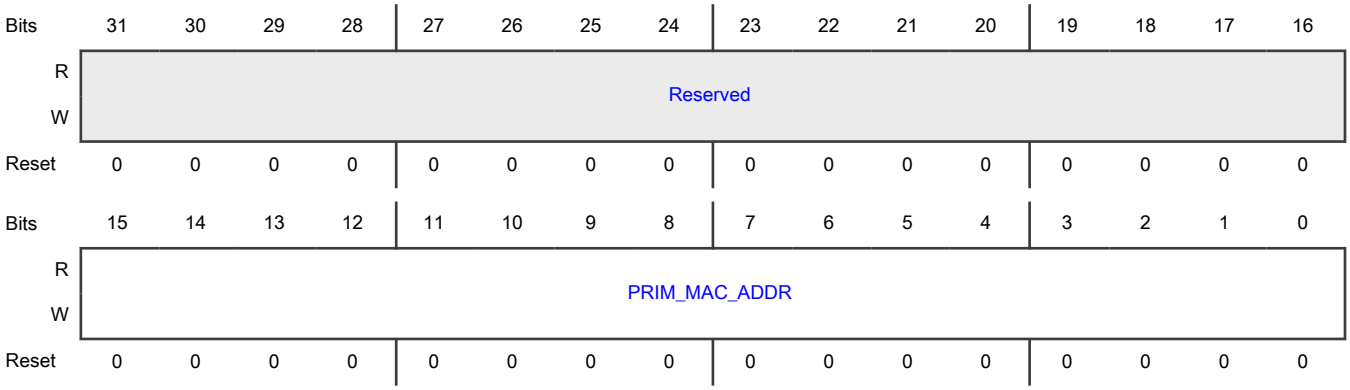
This value is taken directly from the port PMAR1 register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	—	PSI1PMAR1
ENETC1_BASE	PSI1PMAR1	—

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 PRIM_MAC_ADDR	Primary MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset. This field contains the 2 least significant bytes of the MAC address (least significant byte is stored in register bit offset 8-15). For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then PSI*a*PMAR1 equals xxxx1615h (where x should be set to 0).

53.4.6.6.57 Port station interface 1 configuration register 0 (PSI1CFGR0)

Offset

Register	Offset
PSI1CFGR0	2090h

Function

The station interface configuration register determines the features and capabilities of an SI.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	—	PSI1CFGR0
ENETC1_BASE	PSI1CFGR0	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIBW				SIVC				Reserved	NUM_RX_BDR						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ASE	SIVIE	Reserved	VTE	SPE	Reserved			Reserved	NUM_TX_BDR						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 SIBW	Station interface traffic class bandwidth weight Frames are selected for transmission between station interfaces based on a per traffic class basis using the Weighted Fair Bandwidth Sharing algorithm. This value specifies a relative weight and is defined as credits per byte. A higher number indicates less bandwidth. For example, to balance the bandwidth for traffic class 7 between SI 0/1 to allow SI 0 twice the bandwidth, set the weight to 2 and 4 respectively. Note that a setting of 0x0 (reset default) will give an SI absolute priority over any SIs with a non-zero SIBW setting. If multiple SIs are configured with SIBW=0x0 then they each receive equal share of the traffic class bandwidth. 0: Highest bandwidth 1: Next highest bandwidth ... 14: Next lowest bandwidth 15: Lowest bandwidth
27-24 SIVC	Station interface VLAN control

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Determines which VLAN Ethertypes can be inserted by the SI driver (e.g. VF) either in the frame data or through the Tx BD. It also determines which VLAN Ethertypes are acceptable for VLAN removal (as seen by SI). Bit: 24 Standard C-VLAN 0x8100 25 Standard S-VLAN 0x88A8 26 Custom VLAN as defined by CVLANR1[ETYPE] 27 Custom VLAN as defined by CVLANR2[ETYPE]
23 —	Reserved
22-16 NUM_RX_BDR	Number of receive buffer descriptor rings assigned to the SI Range: 0..64 The following rules apply when modifying this field: • By default all rings are assigned to SI 0 (PSI) • Total number of assigned rings must not exceed the number of available rings defined by the port, ECAPR2[NUM_RX_BDR] • Assignment of rings to SIs should be done in increasing order, i.e. SI *n+1* cannot start ring management until SI *n* has been assigned rings • Reconfiguring number of assigned rings to SI *n* requires complete ring management reconfiguration for SI *n+1* and above
15 ASE	Anti-spoofing enable When anti-spoofing is enabled, source MAC address used by SI is compared against the primary MAC address. If it is different, the frame is not transmitted and an error is reported in the status field of the BD. 0: Disabled 1: Enabled
14 SIVIE	SI-based VLAN Insertion Enable This field controls whether insertion of the SI-based VLAN into frames sent by this SI is enabled. 0b - SI-based VLAN insertion disabled 1b - SI-based VLAN insertion enabled
13 —	Reserved
12 VTE	VLAN Tag Extract

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field controls whether SI-based VLAN tag is removed from the frame before delivery to the SI. This tag must be the first tag, immediately following the SMAC address to be recognized and removed. After extraction, the resulting frame is presented to the SI for processing. 0b - SI-based VLAN removal disabled 1b - SI-based VLAN removal enabled
11 SPE	Source Pruning Enable 0b - Disabled 1b - Enabled
10-8 —	Reserved
7 —	Reserved
6-0 NUM_TX_BDR	Number of transmit buffer descriptor rings assigned to the SI Range: 0..64 The following rules apply when modifying this field: • By default all rings are assigned to SI 0 (PSI) • Total number of assigned rings must not exceed the number of available rings defined by the port, ECAPR2[NUM_TX_BDR] • Assignment of rings to SIs should be done in increasing order, i.e. SI *n+1* cannot start ring management until SI *n* has been assigned rings • Reconfiguring number of assigned rings to SI *n* requires complete ring management reconfiguration for SI *n+1* and above

53.4.6.6.58 Port station interface 1 configuration register 1 (PSI1CFGR1)

Offset

Register	Offset
PSI1CFGR1	2094h

Function

The station interface PRIO to TC mapping register maps the VSI traffic class a to set transmit traffic class.

The mapping is done from the PRIO2TCMR0 value to a traffic class, allowing the PSI to restrict (VSI) use.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	—	PSI1CFGR1
ENETC1_BASE	PSI1CFGR1	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	TC7_MAP			Reserved	TC6_MAP			Reserved	TC5_MAP			Reserved	TC4_MAP		
Reset	0	1	1	1	0	1	1	0	0	1	0	1	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	TC3_MAP			Reserved	TC2_MAP			Reserved	TC1_MAP			Reserved	TC0_MAP		
Reset	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 TC7_MAP	Traffic class mapping from PRIO2TCMR0 Maps the VSI traffic class 7 to set transmit traffic class.
27 —	Reserved
26-24 TC6_MAP	Traffic class mapping from PRIO2TCMR0 Maps the VSI traffic class 6 to set transmit traffic class.
23 —	Reserved
22-20 TC5_MAP	Traffic class mapping from PRIO2TCMR0 Maps the VSI traffic class 5 to set transmit traffic class.
19 —	Reserved
18-16 TC4_MAP	Traffic class mapping from PRIO2TCMR0 Maps the VSI traffic class 4 to set transmit traffic class.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 —	Reserved
14-12 TC3_MAP	Traffic class mapping from PRIO2TCMR0 Maps the VSI traffic class 3 to set transmit traffic class.
11 —	Reserved
10-8 TC2_MAP	Traffic class mapping from PRIO2TCMR0 Maps the VSI traffic class 2 to set transmit traffic class.
7 —	Reserved
6-4 TC1_MAP	Traffic class mapping from PRIO2TCMR0 Maps the VSI traffic class 1 to set transmit traffic class.
3 —	Reserved
2-0 TC0_MAP	Traffic class mapping from PRIO2TCMR0 Maps the VSI traffic class 0 to set transmit traffic class.

53.4.6.6.59 Port station interface 1 configuration register 2 (PSI1CFGR2)

Offset

Register	Offset
PSI1CFGR2	2098h

Function

The station interface configuration register 2 determines the number of MSI-X table entries (vectors) allocated to a station interface.

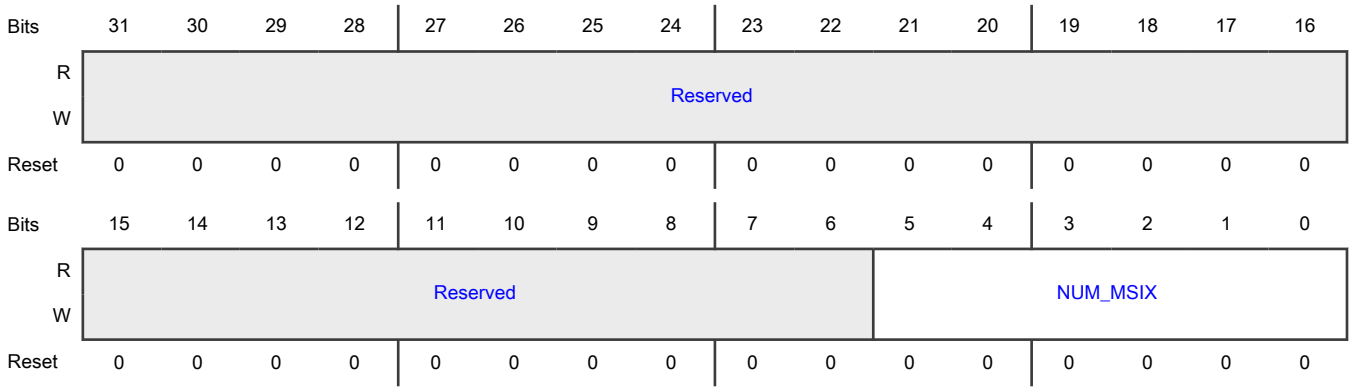
The total number of vectors available to the ENETC instance is determined by ECAPR1[NUM_MSIX]. All vectors are initially assigned to the PSI, less 1 vector per VSI.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	—	PSI1CFGR2
ENETC1_BASE	PSI1CFGR2	—

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 NUM_MSIX	Number of MSI-X Number of MSI-X vectors assigned to a Station Interface. Range: 1..64 Formula: NUM_MSIX+1 **NOTE** Reflected in SIPCAPR1[NUM_MSIX] read-only register.

53.4.6.6.60 Port station interface 1 VSI MAC address filtering configuration register (PSI1VMAFCFGR)

Offset

Register	Offset
PSI1VMAFCFGR	20B0h

Function

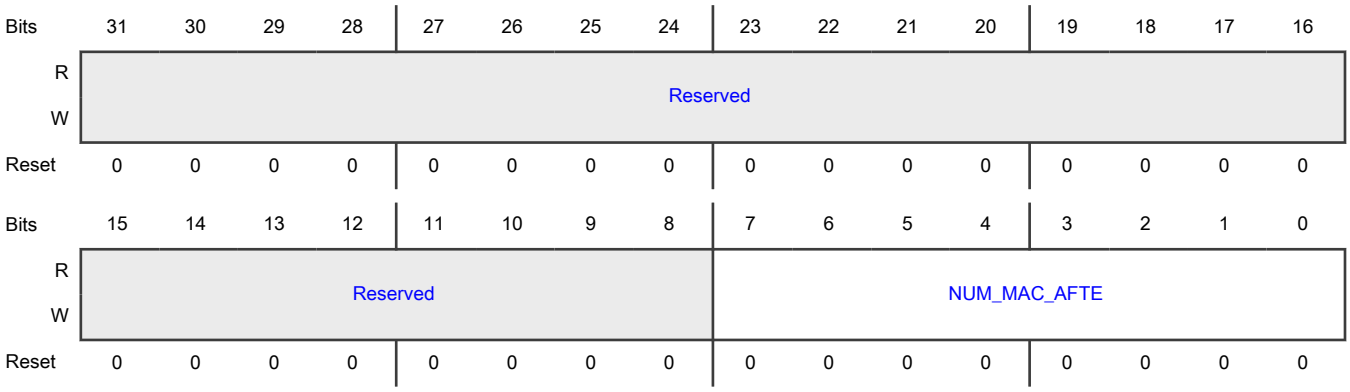
The station interface MAC address filtering configuration register determines the capabilities of a SI.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	—	PSI1VMAFCFGR
ENETC1_BASE	PSI1VMAFCFGR	—

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 NUM_MAC_AFTE	Number of SI MAC address filter table entries assigned to the SI. Number of assigned entries to all SIs must not exceed the total number of available SI MAC address filter rules per port as defined by PSIMAFCAPR[NUM_MAC_AFTE]. This field should be initialized once at start-up, before SI capability discovery starts.

53.4.6.6.61 Port station interface 1 VLAN filtering configuration register (PSI1VLANFCFGR)

Offset

Register	Offset
PSI1VLANFCFGR	20B4h

Function

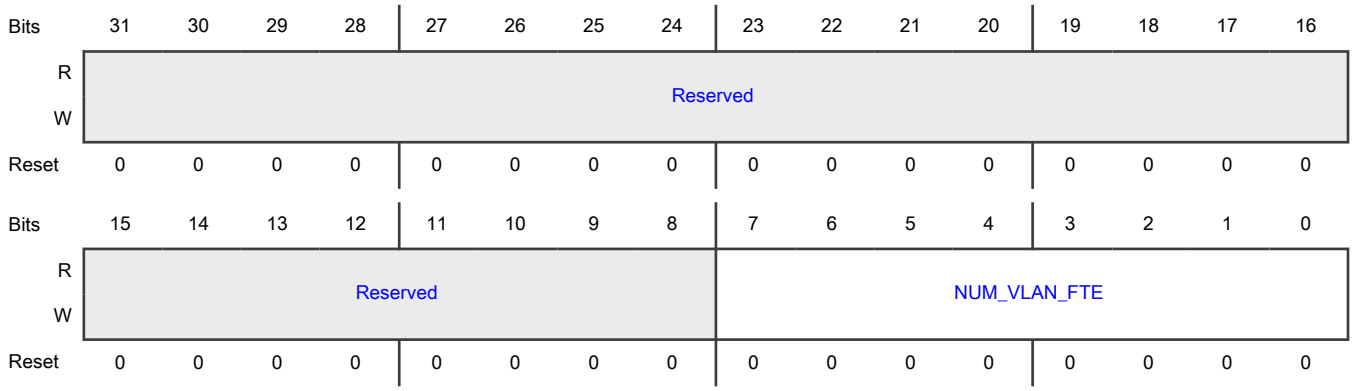
The station interface VLAN filtering configuration register determines the capabilities of a SI.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_BASE	—	PSI1VLANFCFGR
ENETC1_BASE	PSI1VLANFCFGR	—

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 NUM_VLAN_FTE	Number of VLAN filter table entries assigned to the SI. Number of assigned entries to all SIs must not exceed the total number of available SI VLAN filter rules per port as defined by PSIVLANFCAPR[NUM_VLAN_FTE]. This field should be initialized once at start-up, before SI capability discovery starts. Modifying the setting during run-time may lead to undefined behavior.

53.4.6.7 Switch and ENETC common base register descriptions

This section describes the switch and ENETC common registers for one or more instances.

There may be differences in actual implemented registers or register fields as well as reset values between instances. The function's base address register value can be discovered from reading the PCIe Enhanced Allocation Base Address Register Equivalent Index 0 (EA BEI 0).

53.4.6.7.1 Switch_ENETC_Common memory map

Instance	Address space	PCIe EA BEI 0 offset
ENETC0_COMMON	PCI_F3_BAR_0	60B1_0000h
ENETC1_COMMON	PCI_F4_BAR_0	60B5_0000h
SW0_COMMON	PCI_F2_BAR_0	60A0_0000h

Offset	Register	Width (In bits)	Access	Reset value
1000h	Ingress port capability register (IPCAPR)	32	R	See section
1004h	Egress port capability register (EPCAPR)	32	R	See section
1010h	Operational state register (OSR)	32	R	0000_0001h
1040h	Correctable memory error configuration register (CMECR)	32	RW	0000_0000h
1044h	Correctable memory error status register (CMESR)	32	RW	0000_0000h
104Ch	Correctable memory error count register (CMECTR)	32	R	0000_0000h
1060h	Uncorrectable non-fatal MAC error configuration register (UNMACECR)	32	RW	0000_0000h
1064h	Uncorrectable non-fatal MAC error status register (UNMACESR)	32	R	0000_0000h
1070h	Uncorrectable non-fatal system bus error configuration register (UNSBECR)	32	RW	0000_0000h
1074h	Uncorrectable non-fatal system bus error status register (UNSBESR)	32	RW	0000_0000h
107Ch	Uncorrectable non-fatal system bus error count register (UNSBECTR)	32	R	0000_0000h
1080h	Uncorrectable fatal system bus error configuration register (UFSBECR)	32	RW	0000_0000h
1084h	Uncorrectable fatal system bus error status register (UFSBESR)	32	RW	0000_0000h
1090h	Uncorrectable non-fatal memory error configuration register (UNMECR)	32	RW	0000_0000h
1094h	Uncorrectable non-fatal memory error status register 0 (UNMESR0)	32	RW	0000_0000h
1098h	Uncorrectable non-fatal memory error status register 1 (UNMESR1)	32	R	0000_0000h
109Ch	Uncorrectable non-fatal memory error count register (UNMECTR)	32	R	0000_0000h
10A0h	Uncorrectable fatal memory error configuration register (UFMECR)	32	RW	0000_0000h
10A4h	Uncorrectable fatal memory error status register 0 (UFMESR0)	32	RW	0000_0000h
10A8h	Uncorrectable fatal memory error status register 1 (UFMESR1)	32	R	0000_0000h
10E0h	Internal MDIO interrupt reason register (IMDIOIRR)	32	R	0000_0000h
10E4h	Internal MDIO MSI-X vector register (IMDIOMSIVR)	32	RW	0000_0000h
10E8h	External MDIO interrupt reason register (EMDIOIRR)	32	R	0000_0000h
10ECh	External MDIO MSI-X vector register (EMDIOMSIVR)	32	RW	0000_0000h
1100h	Time capture configuration register (TCCR)	32	RW	0000_0000h
1104h	Time capture interrupt enable register (TCIER)	32	RW	0000_0000h
1108h	Time capture receive port interrupt detect register (TCRPIDR)	32	RW	0000_0000h
110Ch	Time capture receive port status register (TCRPSR)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1114h	Time capture receive port timestamp register (TCRPTSR)	32	R	0000_0000h
1118h	Time capture MSI-X vector register (TCMSIVR)	32	RW	0000_0000h
1200h	Custom VLAN Ethertype register 1 (CVLANR1)	32	RW	0000_0000h
1204h	Custom VLAN Ethertype register 2 (CVLANR2)	32	RW	0000_0000h
1208h	Pre-Standard RTAG Ethertype register (PSRTAGETR)	32	RW	0000_0000h
1220h	DoS L2 configuration register (DOSL2CR)	32	RW	0000_0000h
1300h	VLAN to IPV mapping profile 0 register 0 (VLANIPVMP0R0)	32	RW	3322_1100h
1304h	VLAN to IPV mapping profile 0 register 1 (VLANIPVMP0R1)	32	RW	7766_5544h
1308h	VLAN to DR mapping profile 0 register (VLANDRMP0R)	32	RW	8888_8888h
1310h	VLAN to IPV mapping profile 1 register 0 (VLANIPVMP1R0)	32	RW	3322_1100h
1314h	VLAN to IPV mapping profile 1 register 1 (VLANIPVMP1R1)	32	RW	7766_5544h
1318h	VLAN to DR mapping profile 1 register (VLANDRMP1R)	32	RW	8888_8888h
1640h	Ingress port filter capability register (IPFCAPR)	32	R	See section
1644h	Ingress port filter table capability register (IPFTCAPR)	32	R	See section
1648h	Ingress port filter table memory operational register (IPFTMOR)	32	R	0000_0000h
1800h	Index table memory capability register (ITMCAPR)	32	R	See section
1810h	Rate policer capability register (RPCAPR)	32	R	0000_0003h
1814h	Rate policer index table capability register (RPITCAPR)	32	R	See section
1818h	Rate policer index table memory allocation register (RPITMAR)	32	RW	See section
181Ch	Rate policer index table operational register (RPITOR)	32	R	0000_0000h
1824h	Ingress stream counter index table capability register (ISCITCAPR)	32	R	See section
1828h	Ingress stream counter index table memory allocation register (ISCITMAR)	32	RW	See section
182Ch	Ingress stream counter index table operational register (ISCITOR)	32	R	0000_0000h
1830h	Ingress stream capability register (ISCAPR)	32	R	See section
1834h	Ingress stream index table capability register (ISITCAPR)	32	R	See section
1838h	Ingress stream index table memory allocation register (ISITMAR)	32	RW	See section
183Ch	Ingress stream index table operational register (ISITOR)	32	R	0000_0000h
1844h	Ingress sequence generation index table capability register (ISQGITCAPR)	32	R	0010_0180h
1848h	Ingress sequence generation index table memory allocation register (ISQGITMAR)	32	RW	0000_0030h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
184Ch	Ingress sequence generation index table operational register (ISQGITOR)	32	R	0000_0000h
1860h	Stream gate capability register (SGCAPR)	32	R	See section
1864h	Stream gate instance index table capability register (SGIITCAPR)	32	R	See section
1868h	Stream gate instance index table memory allocation register (SGIITMAR)	32	RW	See section
186Ch	Stream gate instance index table operational register (SGIITOR)	32	R	0000_0000h
1874h	Stream gate control list index table capability register (SGCLITCAPR)	32	R	See section
1878h	Stream gate control list index table memory allocation register (SGCLITMAR)	32	RW	See section
187Ch	Stream gate control list table memory operational register (SGCLTMOR)	32	R	0000_0000h
1880h	Frame modification ingress capability register (FMICAPR)	32	R	0000_000Ah
1884h	Frame modification egress capability register (FMECAPR)	32	R	0003_002Fh
1888h	Frame modification index table capability register (FMITCAPR)	32	R	0010_0040h
188Ch	Frame modification index table memory allocation register (FMITMAR)	32	RW	0000_0040h
1890h	Frame modification index table operational register (FMITOR)	32	R	0000_0000h
1894h	Frame modification data index table capability register (FMDITCAPR)	32	R	0010_0100h
1898h	Frame modification data index table memory allocation register (FMDITMAR)	32	RW	0000_0100h
18C0h	Egress treatment capability register (ETCAPR)	32	R	0000_0001h
18C4h	Egress treatment table capability register (ETTCAPR)	32	R	0010_0180h
18CCh	Egress treatment table operational register (ETTOR)	32	R	0000_0000h
18D4h	Time gate scheduling table capability register (TGSTCAPR)	32	R	See section
18DCh	Time gate scheduling table memory operation register (TGSTMOR)	32	R	0000_0000h
18E0h	Egress sequence recovery capability register (ESQRCAPR)	32	R	0000_010Fh
18E4h	Egress sequence recovery table capability register (ESQRTCAPR)	32	R	0010_0180h
18ECh	Egress counter table capability register (ECTCAPR)	32	R	0010_0180h
1900h	Hash table memory capability register (HTMCAPR)	32	R	See section
1904h	Hash table memory operational register (HTMOR)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1910h	Ingress stream identification capability register (ISIDCAPR)	32	R	See section
1914h	Ingress stream identification hash table capability register (ISIDHTCAPR)	32	R	00B0_0000h
1920h	Ingress stream identification key construction 0 operational register (ISIDKC0OR)	32	R	0000_0000h
1924h	Ingress stream identification key construction 0 configuration register 0 (ISIDKC0CR0)	32	RW	0000_0000h
1930h	Ingress stream identification key construction 0 payload field 0 configuration register (ISIDKC0PF0CR)	32	RW	0000_0000h
1934h	Ingress stream identification key construction 0 payload field 1 configuration register (ISIDKC0PF1CR)	32	RW	0000_0000h
1938h	Ingress stream identification key construction 0 payload field 2 configuration register (ISIDKC0PF2CR)	32	RW	0000_0000h
193Ch	Ingress stream identification key construction 0 payload field 3 configuration register (ISIDKC0PF3CR)	32	RW	0000_0000h
1940h	Ingress stream identification key construction 1 operational register (ISIDKC1OR)	32	R	0000_0000h
1944h	Ingress stream identification key construction 1 configuration register 0 (ISIDKC1CR0)	32	RW	0000_0000h
1950h	Ingress stream identification key construction 1 payload field 0 configuration register (ISIDKC1PF0CR)	32	RW	0000_0000h
1954h	Ingress stream identification key construction 1 payload field 1 configuration register (ISIDKC1PF1CR)	32	RW	0000_0000h
1958h	Ingress stream identification key construction 1 payload field 2 configuration register (ISIDKC1PF2CR)	32	RW	0000_0000h
195Ch	Ingress stream identification key construction 1 payload field 3 configuration register (ISIDKC1PF3CR)	32	RW	0000_0000h
1960h	Ingress stream identification key construction 2 operational register (ISIDKC2OR)	32	R	0000_0000h
1964h	Ingress stream identification key construction 2 configuration register 0 (ISIDKC2CR0)	32	RW	0000_0000h
1970h	Ingress stream identification key construction 2 payload field 0 configuration register (ISIDKC2PF0CR)	32	RW	0000_0000h
1974h	Ingress stream identification key construction 2 payload field 1 configuration register (ISIDKC2PF1CR)	32	RW	0000_0000h
1978h	Ingress stream identification key construction 2 payload field 2 configuration register (ISIDKC2PF2CR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
197Ch	Ingress stream identification key construction 2 payload field 3 configuration register (ISIDKC2PF3CR)	32	RW	0000_0000h
1980h	Ingress stream identification key construction 3 operational register (ISIDKC3OR)	32	R	0000_0000h
1984h	Ingress stream identification key construction 3 configuration register 0 (ISIDKC3CR0)	32	RW	0000_0000h
1990h	Ingress stream identification key construction 3 payload field 0 configuration register (ISIDKC3PF0CR)	32	RW	0000_0000h
1994h	Ingress stream identification key construction 3 payload field 1 configuration register (ISIDKC3PF1CR)	32	RW	0000_0000h
1998h	Ingress stream identification key construction 3 payload field 2 configuration register (ISIDKC3PF2CR)	32	RW	0000_0000h
199Ch	Ingress stream identification key construction 3 payload field 3 configuration register (ISIDKC3PF3CR)	32	RW	0000_0000h
1A00h	Ingress stream filter hash table capability register (ISFHTCAPR)	32	R	0070_0000h
1A04h	Ingress stream filter hash table operational register (ISFHTOR)	32	R	0000_0000h

53.4.6.7.2 Ingress port capability register (IPCAPR)

Offset

Register	Offset
IPCAPR	1000h

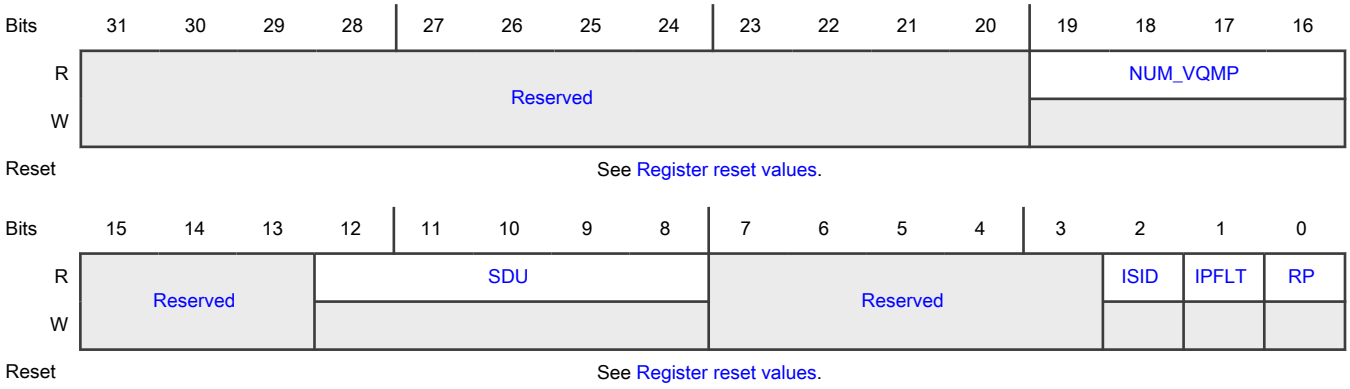
Function

This is the ingress port capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
IPCAPR	ENETC0_COMMON,ENETC1_COMMON: 0001_0707h SW0_COMMON: 0002_0707h

Fields

Field	Function
31-20 —	Reserved
19-16 NUM_VQMP	Specifies the number *a* of receive VLAN PCP/DE to QoS mapping profiles supported; see registers VLANIPVMP*a*R0/1 and VLANDRMP*a*R, where a={0..NUM_VQMP-1}.
15-13 —	Reserved
12-8 SDU	Indicates support for various PDU/SDUs (Protocol/Service Data Unit) definitions. Specifies the type of PDU/SDU (Protocol/Service Data Unit) whose length is being validated as seen on the line. xxxx1: PPDU (Physical Layer PDU): Preamble, IFG, SFD along with MPDU xxx1x: MPDU: (MAC PDU): MAC Header, MSDU and FCS xx1xx: MSDU (MAC SDU); MPDU minus 12B MAC Header and 4B CRC. All other settings reserved. **NOTE** This information is used in ingress by Rate Policer and Stream Gate to determine the SDU size.
7-3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 ISID	Ingress Stream Identification functionality supported. 0: Not supported 1: Supported See ISIDCAPR for more information.
1 IPFLT	Ingress Port Filtering supported (that is,: Ingress Port Filter table lookup). See IPFTCAPR for more information. 0: Not supported 1: Supported
0 RP	Rate Policer function supported. 0: Not supported 1: Supported See RPCAPR for more information.

53.4.6.7.3 Egress port capability register (EPCAPR)

Offset

Register	Offset
EPCAPR	1004h

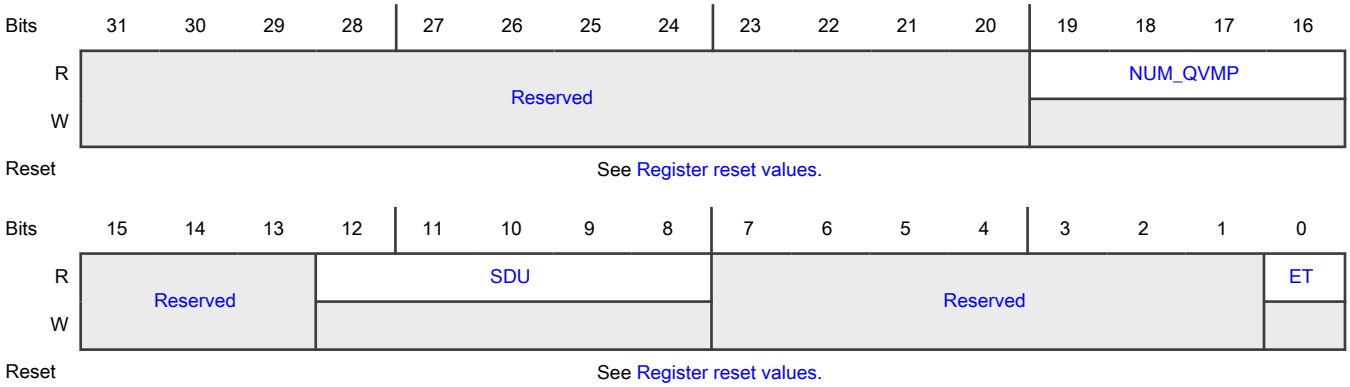
Function

This is the egress port capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
EPCAPR	ENETC0_COMMON,ENETC1_COMMON: 0000_0700h SW0_COMMON: 0002_0701h

Fields

Field	Function
31-20 —	Reserved
19-16 NUM_QVMP	Specifies the number of transmit QoS to VLAN PCP mapping profiles supported; see register QOSVLANMP*a*R0/1/2/3 where a={0..NUM_QVMP-1}. **NOTE** Valid for switch only.
15-13 —	Reserved
12-8 SDU	Indicates support for various PDU/SDUs (Protocol/Service Data Unit) definitions. Specifies the type of PDU/SDU (Protocol/Service Data Unit) whose length is being validated as seen on the line. xxxx1: PPDU (Physical Layer PDU): Preamble, IFG, SFD along with MPDU xxx1x: MPDU: (MAC PDU): MAC Header, MSDU and FCS xx1xx: MSDU (MAC SDU); MPDU minus 12B MAC Header and 4B CRC. All other settings reserved. **NOTE** This information is used in egress for Traffic Class maximum SDU length check.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-1 —	Reserved
0 ET	Egress Treatment function supported. 0: Not supported 1: Supported See ETCAPR for more information. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
ENETC0_COMMON	—	EPCAPR
ENETC1_COMMON	—	EPCAPR
SW0_COMMON	EPCAPR	—

53.4.6.7.4 Operational state register (OSR)

Offset

Register	Offset
OSR	1010h

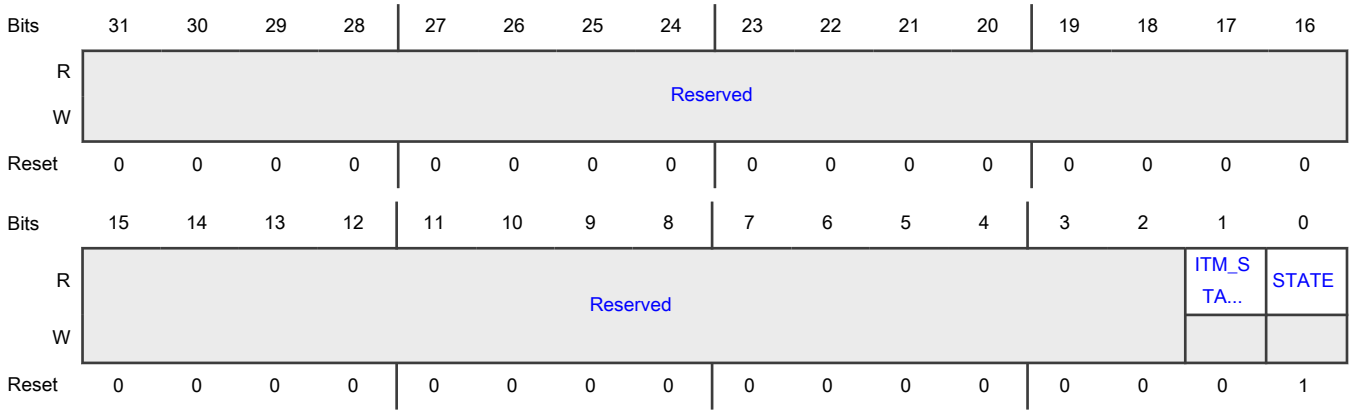
Function

This is the operational state register.

NOTE

Register is not affected by reset. A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 ITM_STATE	Indicates the index table memory (common memory) operational state. 0: Index table memory properly configured 1: Index table memory is overly-provisioned. That is, the sum of all the index tables' *ITMAR[NUM_WORDS] is greater than ITMCAPR[NUM_WORDS]. **CAUTION** If ITM_STATE is set, then all index table commands are rejected until this state is cleared. A write to a *ITMAR register will re-evaluate this state.
0 STATE	Indicates the function's operational state 0: Function is operationally ready. 1: Function is in the process of initializing memory. **CAUTION** Software must not write any device registers until state indicates it is operational.

53.4.6.7.5 Correctable memory error configuration register (CMECR)

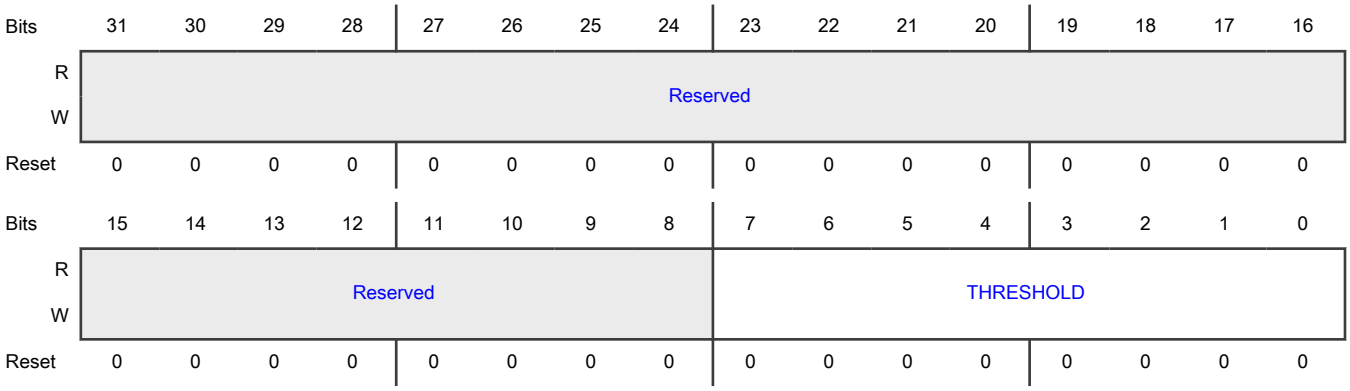
Offset

Register	Offset
CMECR	1040h

Function

The correctable memory error configuration register can disable the event reporting to the Root Complex Event Collector. While correctable errors do not cause any harm, detecting many of these events could indicate a more severe issue. The threshold can be set to trigger the event.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 THRESHOLD	Threshold Determines how many single bit ECC errors must be detected before error status bit is set and correctable memory error is reported to PCIe Event Collector. 0: Disabled (no reporting) >0: Threshold value (1-255)

53.4.6.7.6 Correctable memory error status register (CMESR)

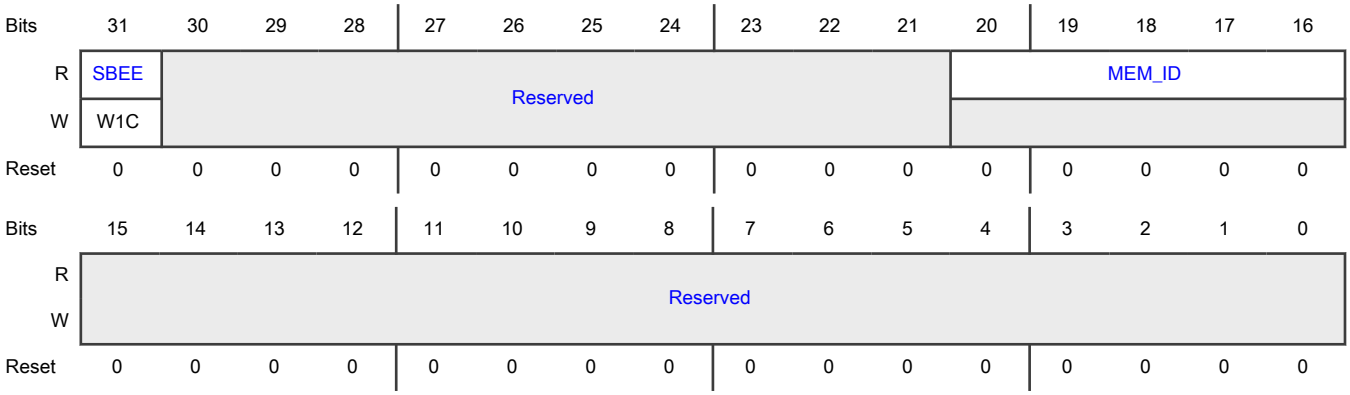
Offset

Register	Offset
CMESR	1044h

Function

This is the correctable memory error status register.

Diagram



Fields

Field	Function
31 SBEE	Single-bit ECC error When set, a threshold number of single-bit ECC error has occurred in the memory defined by MEM_ID. Write 1 to clear.
30-21 —	Reserved
20-16 MEM_ID	Memory ID See Table 392 for list of memory identifiers.
15-0 —	Reserved

53.4.6.7.7 Correctable memory error count register (CMECTR)

Offset

Register	Offset
CMECTR	104Ch

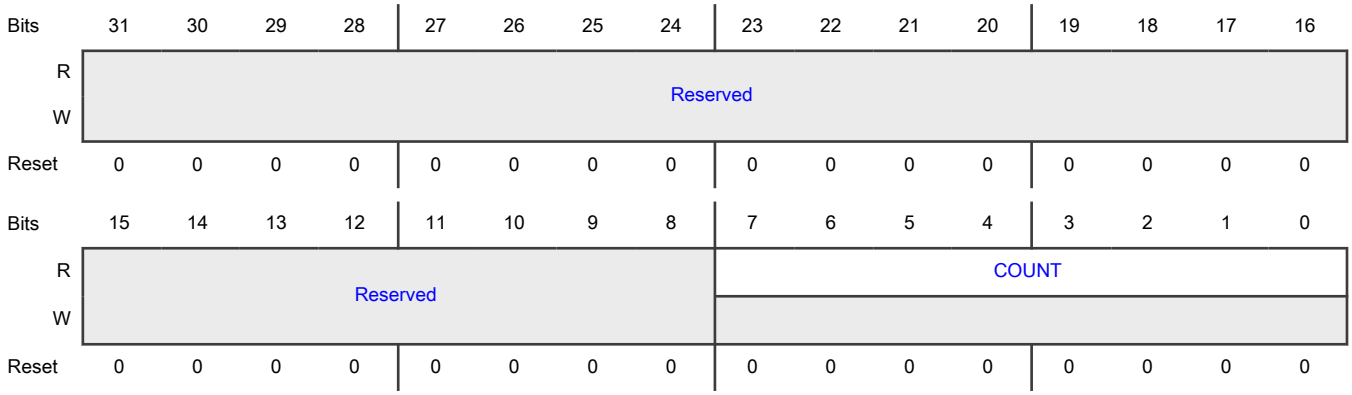
Function

This is the correctable memory error count register which tracks how many events have been detected.

NOTE

A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Count Indicates how many single-bit ECC error have been detected. After a read operation, the field's value clears to 0.

53.4.6.7.8 Uncorrectable non-fatal MAC error configuration register (UNMACECR)

Offset

Register	Offset
UNMACECR	1060h

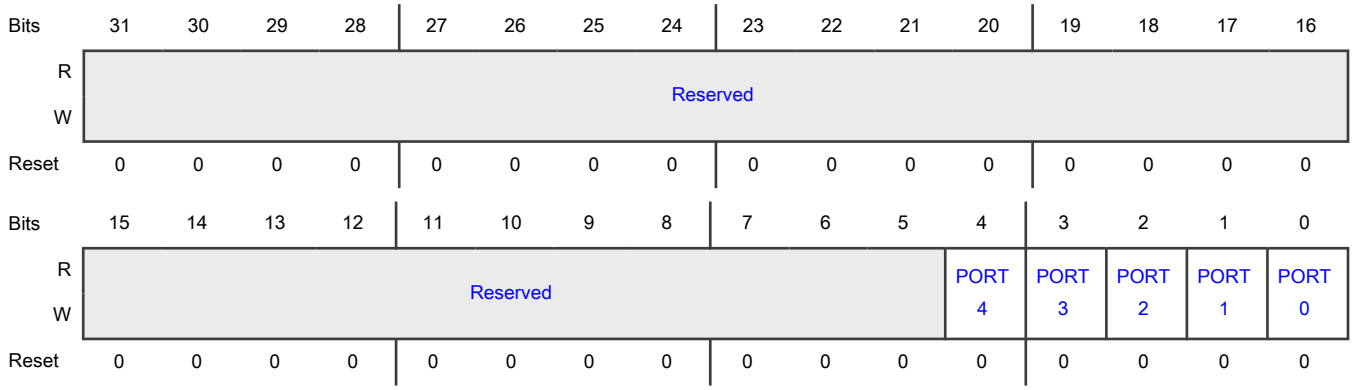
Function

The uncorrectable non-fatal MAC error configuration register can disable the event reporting to the Root Complex Event Collector.

NOTE

A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 PORT4	Report disable port When reporting is enabled, and UNMACESR[PORT] is set, an uncorrectable non-fatal error message is generated to the Root Complex Event Collector. 0: Enabled 1: Disabled **NOTE** If port is attached to pseudo MAC, this bit does not apply. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
ENETC0_COMMON	—	UNMACECR
ENETC1_COMMON	—	UNMACECR
SW0_COMMON	UNMACECR	—

3 PORT3	Report disable port When reporting is enabled, and UNMACESR[PORT] is set, an uncorrectable non-fatal error message is generated to the Root Complex Event Collector. 0: Enabled 1: Disabled

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
If port is attached to pseudo MAC, this bit does not apply.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>ENETC0_COMMON</td><td>—</td><td>UNMACECR</td></tr>
<tr><td>ENETC1_COMMON</td><td>—</td><td>UNMACECR</td></tr>
<tr><td>SW0_COMMON</td><td>UNMACECR</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>2
PORT2</td><td>
Report disable port
When reporting is enabled, and UNMACESR[PORT] is set, an uncorrectable non-fatal error message is generated to the Root Complex Event Collector.
0: Enabled
1: Disabled

NOTE
If port is attached to pseudo MAC, this bit does not apply.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>ENETC0_COMMON</td><td>—</td><td>UNMACECR</td></tr>
<tr><td>ENETC1_COMMON</td><td>—</td><td>UNMACECR</td></tr>
<tr><td>SW0_COMMON</td><td>UNMACECR</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>1
PORT1</td><td>
Report disable port
When reporting is enabled, and UNMACESR[PORT] is set, an uncorrectable non-fatal error message is generated to the Root Complex Event Collector.
0: Enabled
1: Disabled
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
If port is attached to pseudo MAC, this bit does not apply.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>ENETC0_COMMON</td><td>—</td><td>UNMACECR</td></tr>
<tr><td>ENETC1_COMMON</td><td>—</td><td>UNMACECR</td></tr>
<tr><td>SW0_COMMON</td><td>UNMACECR</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>0
PORT0</td><td>Report disable port
When reporting is enabled, and UNMACESR[PORT] is set, an uncorrectable non-fatal error message generated to the Root Complex Event Collector.
0: Enabled
1: Disabled

NOTE
If port is attached to pseudo MAC, this bit does not apply.</td></tr>
</table>

53.4.6.7.9 Uncorrectable non-fatal MAC error status register (UNMACESR)

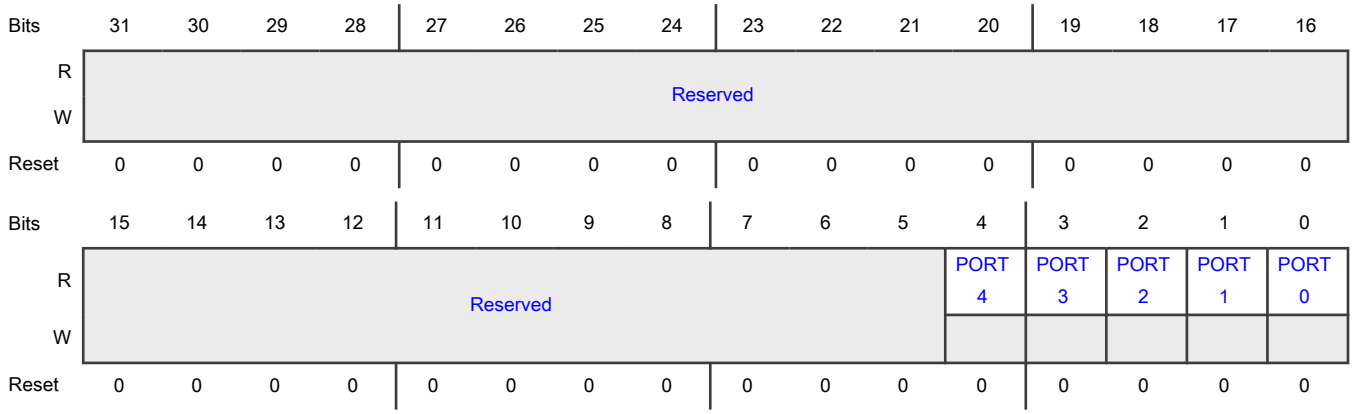
Offset

Register	Offset
UNMACESR	1064h

Function

This is the uncorrectable non-fatal MAC error status register.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 PORT4	Port 4 MAC error 0: No error 1: A non-fatal error has occurred in the Ethernet MAC bound to this port. See PM*a*_IEVENT for details and how to clear the event. **NOTE** If port is attached to pseudo MAC, this bit does not apply. **NOTE** This field is not supported in every instance. The following table includes only supported registers.
3 PORT3	Port 3 MAC error 0: No error 1: A non-fatal error has occurred in the Ethernet MAC bound to this port. See PM*a*_IEVENT for details and how to clear the event. **NOTE** If port is attached to pseudo MAC, this bit does not apply.

Instance	Field supported in	Field not supported in
ENETC0_COMMON	—	UNMACESR
ENETC1_COMMON	—	UNMACESR
SW0_COMMON	UNMACESR	—

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>ENETC0_COMMON</td><td>—</td><td>UNMACESR</td></tr>
<tr><td>ENETC1_COMMON</td><td>—</td><td>UNMACESR</td></tr>
<tr><td>SW0_COMMON</td><td>UNMACESR</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>2
PORT2</td><td>
Port 2 MAC error
0: No error
1: A non-fatal error has occurred in the Ethernet MAC bound to this port. See PMa_IEVENT for details and how to clear the event.
NOTE
If port is attached to pseudo MAC, this bit does not apply.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>ENETC0_COMMON</td><td>—</td><td>UNMACESR</td></tr>
<tr><td>ENETC1_COMMON</td><td>—</td><td>UNMACESR</td></tr>
<tr><td>SW0_COMMON</td><td>UNMACESR</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>1
PORT1</td><td>
Port 1 MAC error
0: No error
1: A non-fatal error has occurred in the Ethernet MAC bound to this port. See PMa_IEVENT for details and how to clear the event.
NOTE
If port is attached to pseudo MAC, this bit does not apply.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>ENETC0_COMMON</td><td>—</td><td>UNMACESR</td></tr><tr><td>ENETC1_COMMON</td><td>—</td><td>UNMACESR</td></tr><tr><td>SW0_COMMON</td><td>UNMACESR</td><td>—</td></tr></table>
0 PORT0	Port MAC error 0: No error 1: A non-fatal error has occurred in the Ethernet MAC bound to this port. See PM*a*_IEVENT for details and how to clear the event. **NOTE** If port is attached to pseudo MAC, this bit does not apply.

53.4.6.7.10 Uncorrectable non-fatal system bus error configuration register (UNSBECR)

Offset

Register	Offset
UNSBECR	1070h

Function

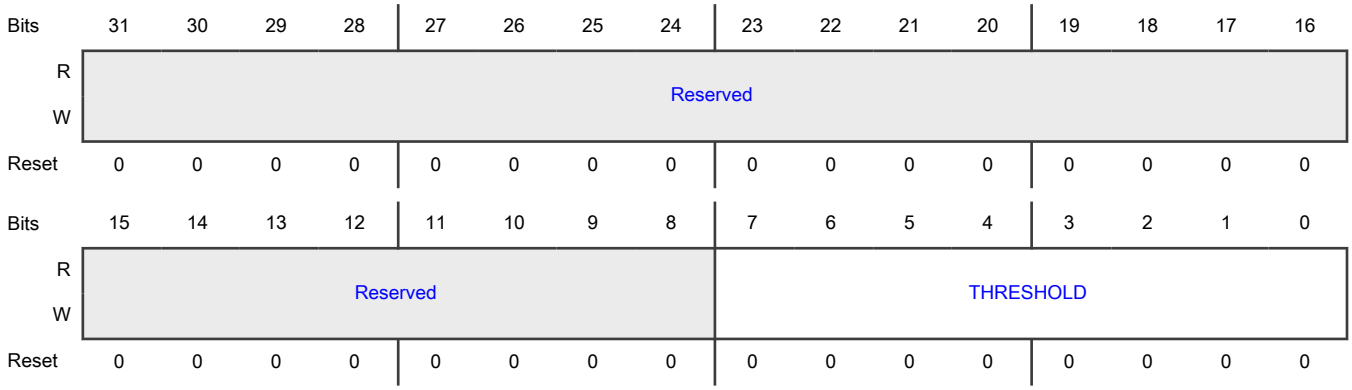
The uncorrectable non-fatal system bus error configuration register can disable the event reporting to the Root Complex Event Collector.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	UNSBECR
ENETC1_COMMON	—	UNSBECR
SW0_COMMON	UNSBECR	—

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 THRESHOLD	Threshold Determines how many non-fatal system bus errors must be detected before error status bit is set and an uncorrectable non-fatal error message is generated to the Root Complex Event Collector. 0: Disabled (no reporting) >0: Threshold value (1-255).

53.4.6.7.11 Uncorrectable non-fatal system bus error status register (UNSBESR)

Offset

Register	Offset
UNSBESR	1074h

Function

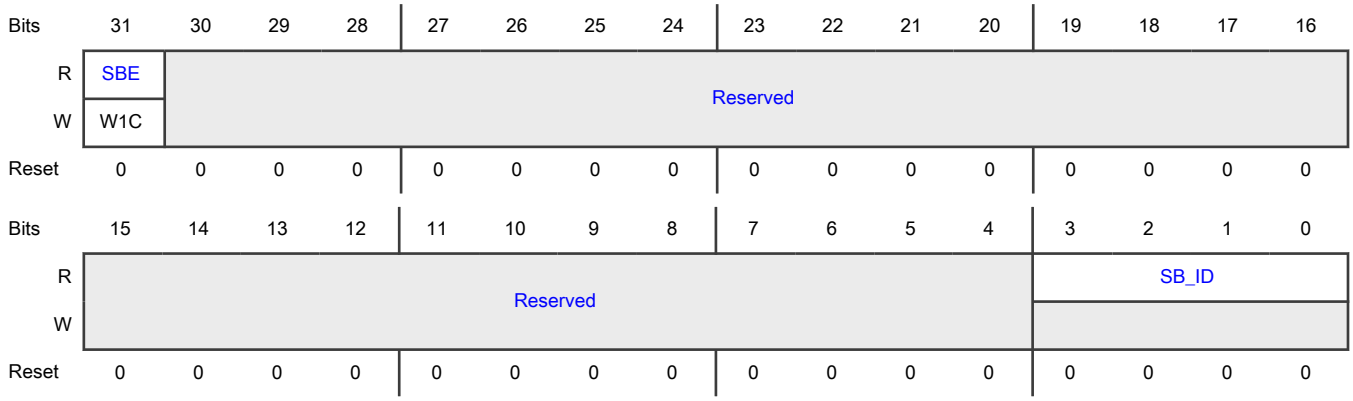
This is the uncorrectable non-fatal system bus error status register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	UNSBESR
ENETC1_COMMON	—	UNSBESR
SW0_COMMON	UNSBESR	—

Diagram



Fields

Field	Function
31 SBE	System bus error 0: No non-fatal system bus error has occurred, or the number of non-fatal system bus errors is lower than UNSBECR[THRESHOLD]. 1: A threshold number of non-fatal system bus error occurred due to NETC source attempting to read/write a memory target. The reaction to the error may include stalled descriptor ring processing, see Table 394 for details. Write 1 to clear.
30-4 —	Reserved
3-0 SB_ID	System Bus ID See Table 394 for list of system bus source identifiers.

53.4.6.7.12 Uncorrectable non-fatal system bus error count register (UNSBECTR)

Offset

Register	Offset
UNSBECTR	107Ch

Function

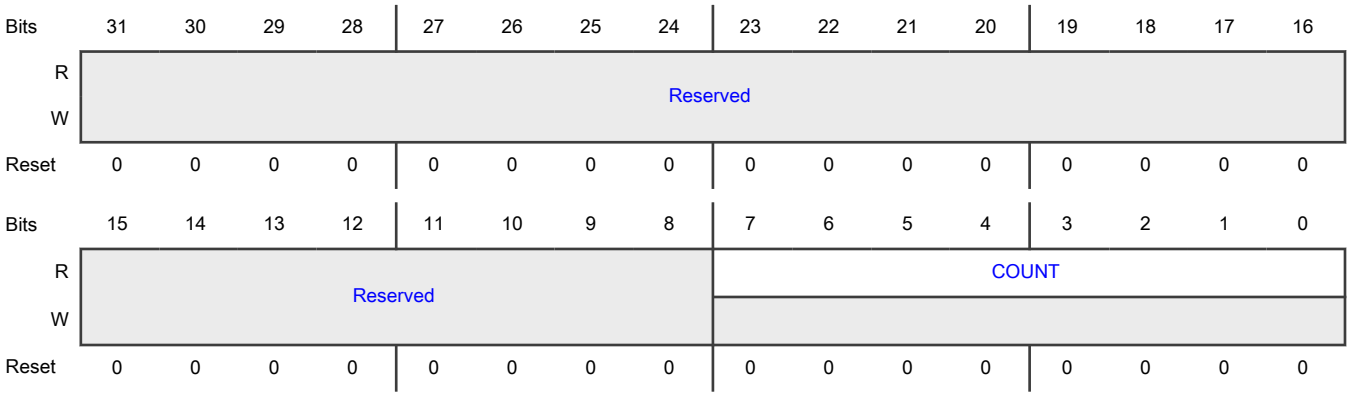
This is the uncorrectable non-fatal system bus error count register which tracks how many events have been detected.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	UNSBECTR
ENETC1_COMMON	—	UNSBECTR
SW0_COMMON	UNSBECTR	—

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Count Indicates how many system bus error events have been encountered. UNSBESR[SB_ID] indicates the last source. After a read operation, the field's value clears to 0.

53.4.6.7.13 Uncorrectable fatal system bus error configuration register (UFSBECR)

Offset

Register	Offset
UFSBECR	1080h

Function

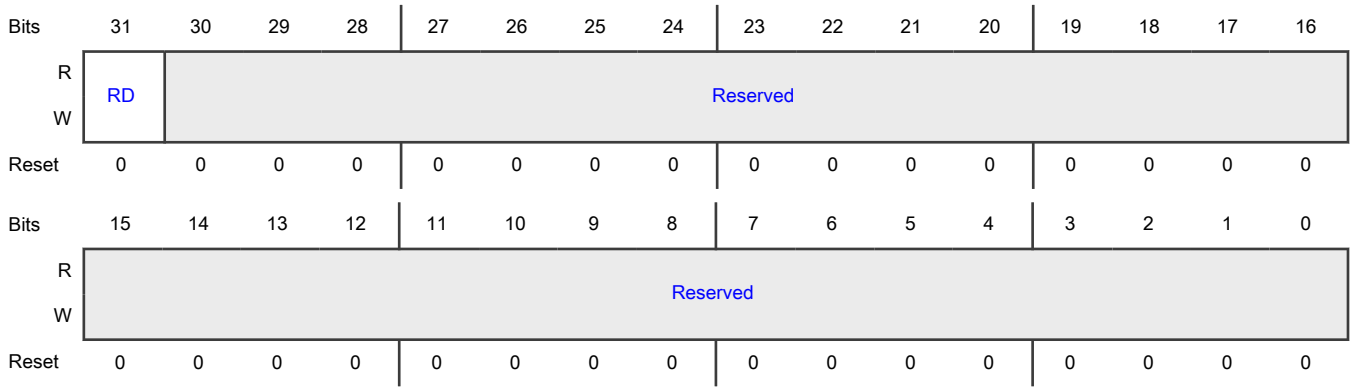
The uncorrectable fatal system bus error configuration register can disable event reporting to the PCIe function.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	UFSBECR
ENETC1_COMMON	—	UFSBECR
SW0_COMMON	UFSBECR	—

Diagram



Fields

Field	Function
31 RD	Report disable When reporting is enabled, and UFSBESR[SBE] is set, an uncorrectable error is reported to the PCIe function's Advanced Error Reporting capability register, PCIE_CFC_AER_UCORR_ERR_STAT. 0b - Enabled 1b - Disabled
30-0 —	Reserved

53.4.6.7.14 Uncorrectable fatal system bus error status register (UFSBESR)

Offset

Register	Offset
UFSBESR	1084h

Function

This is the uncorrectable fatal system bus error status register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	UFSBESR
ENETC1_COMMON	—	UFSBESR
SW0_COMMON	UFSBESR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBE	M	Reserved													
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SB_ID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBE	System bus error 0: No fatal system bus error has occurred. 1: A fatal system bus error occurred when a NETC source attempted to read/write a memory target. The function has entered a safe state: • PCIe bus master enable is cleared (0b0) • Pseudo MAC ports bound to instance are disabled (POR[TXDIS/RXDIS]=0b1) • Ethernet-MAC ports bound to instance are disabled (PM*a*_COMMAND_CONFIG[RX_EN/TX_EN]=0b0) For recovery, S/W should reset the function by issuing a Function Level Reset (FLR). Write 1 to clear.
30 M	Multiple 0: One or none fatal system bus error has occurred. 1: Multiple fatal system bus error events have occurred. The last event is captured.
29-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3-0 SB_ID	System Bus ID See Table 394 for list of system bus identifiers.

53.4.6.7.15 Uncorrectable non-fatal memory error configuration register (UNMECR)

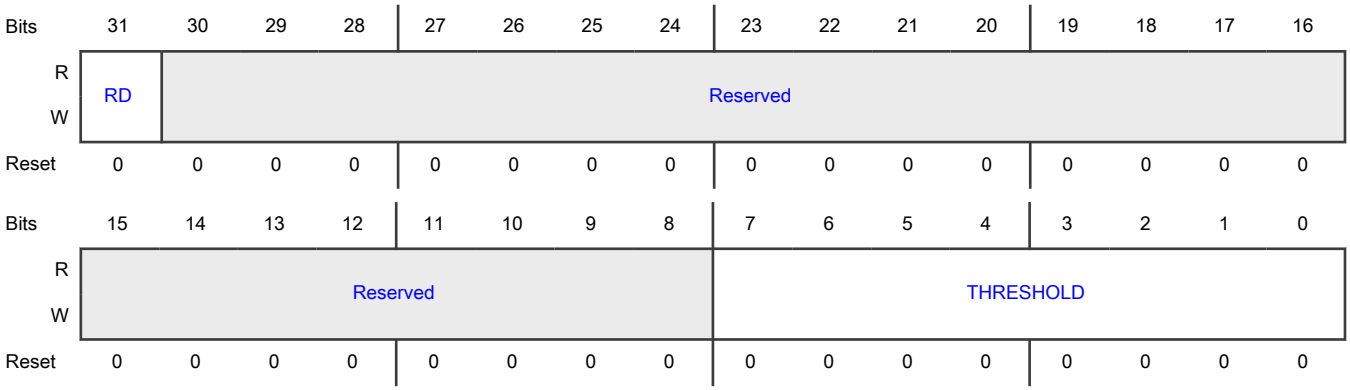
Offset

Register	Offset
UNMECR	1090h

Function

The uncorrectable non-fatal memory error configuration register can disable event reporting to the PCIe function.

Diagram



Fields

Field	Function
31 RD	Report disable When reporting is enabled and UNMESR0[MBEE] is set, an uncorrectable error is reported to the PCIe function's Advanced Error Reporting capability register, PCIE_CFC_AER_UCORR_ERR_STAT. 0: Enabled 1: Disabled
30-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 THRESHOLD	Threshold Determines how many non-fatal memory errors must be detected before error status bit is set. 0: Disabled (no reporting) >0: Threshold value (1-255).

53.4.6.7.16 Uncorrectable non-fatal memory error status register 0 (UNMESR0)

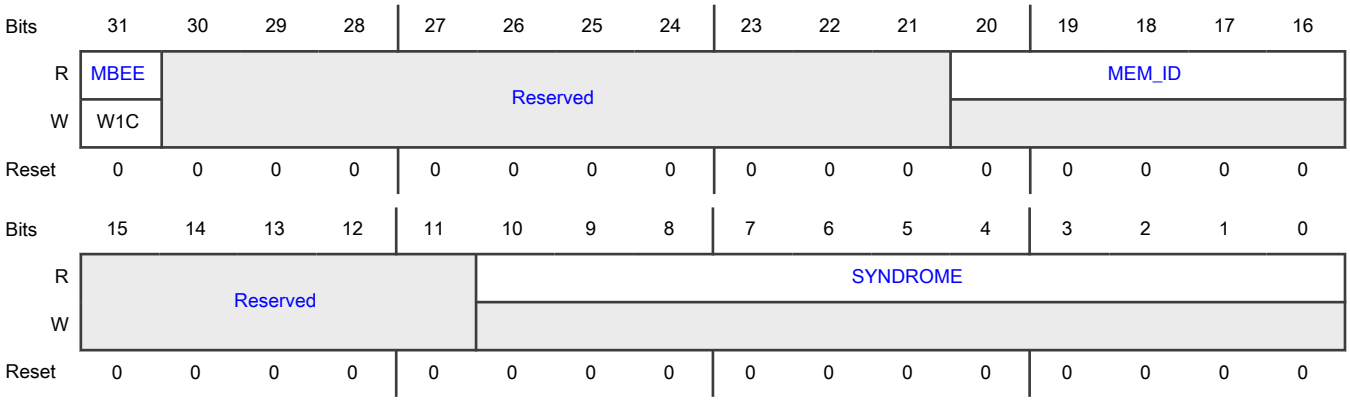
Offset

Register	Offset
UNMESR0	1094h

Function

This is the uncorrectable non-fatal memory error status register 0.

Diagram



Fields

Field	Function
31 MBEE	Multi-bit ECC error 0: No non-fatal multi-bit ECC error has occurred. 1: A threshold number of non-fatal multi-bit ECC errors has occurred in internal memory as defined by MEM_ID. The reaction to the error may result in dropping the frame being processed or in stomping on the FCS of the frame if the error is detected while the transmission of the frame is in progress. Write 1 to clear.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-21 —	Reserved
20-16 MEM_ID	Memory ID See Table 392 for list of memory identifiers.
15-11 —	Reserved
10-0 SYNDROME	Syndrome Identifies the first pertinent bit position (column) in memory that caused the error.

53.4.6.7.17 Uncorrectable non-fatal memory error status register 1 (UNMESR1)

Offset

Register	Offset
UNMESR1	1098h

Function

This is the uncorrectable non-fatal memory error status register 1.

NOTE

A write to this read-only register will return an error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ADDR	Address Address information (row) of last ECC event.

53.4.6.7.18 Uncorrectable non-fatal memory error count register (UNMECTR)

Offset

Register	Offset
UNMECTR	109Ch

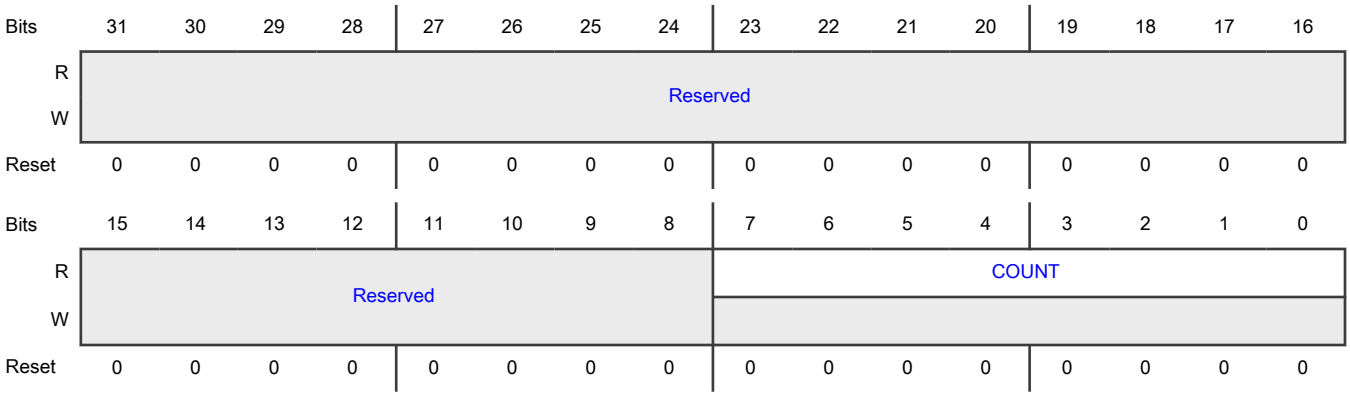
Function

This is the uncorrectable non-fatal memory error count register which tracks how many events have been detected.

NOTE

A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Count Indicates how many frames have been dropped due to internal memory error. After a read operation, the field's value clears to 0.

53.4.6.7.19 Uncorrectable fatal memory error configuration register (UFMECR)

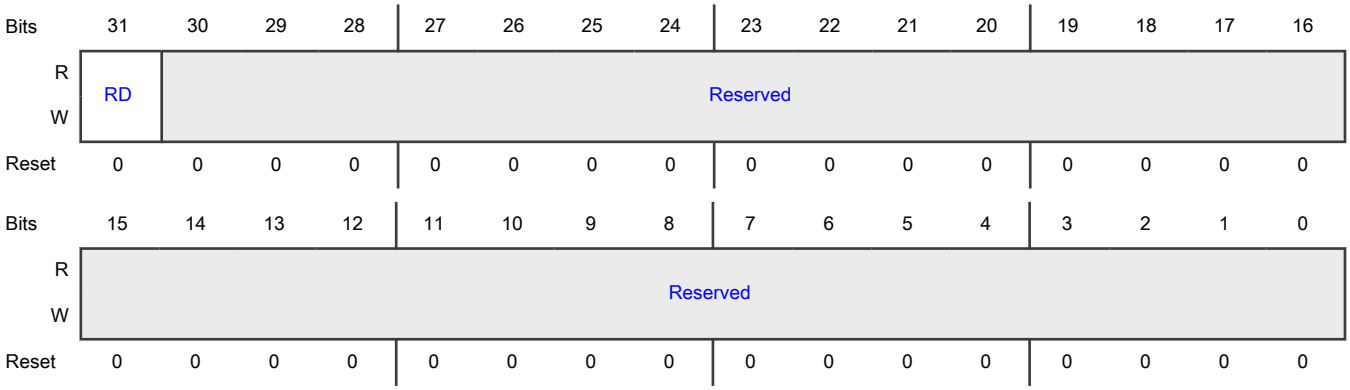
Offset

Register	Offset
UFMECR	10A0h

Function

The uncorrectable fatal memory error configuration register can disable event reporting to the PCIe function.

Diagram



Fields

Field	Function
31 RD	Report disable When reporting is enabled, and UFMESR0[MBEE] is set, an uncorrectable error is reported to the PCIe function's Advanced Error Reporting capability register, PCIE_CFC_AER_UCORR_ERR_STAT. 0: Enabled 1: Disabled
30-0 —	Reserved

53.4.6.7.20 Uncorrectable fatal memory error status register 0 (UFMESR0)

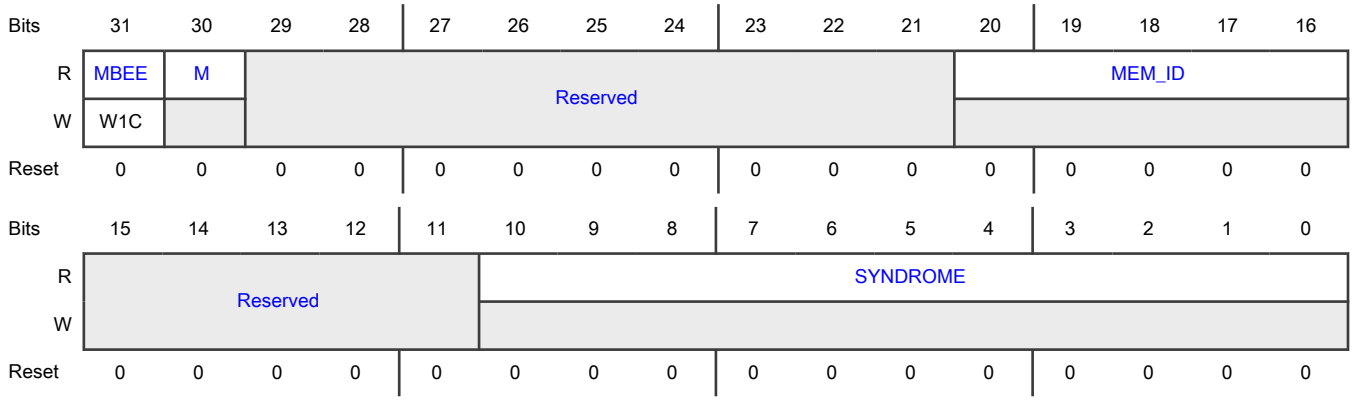
Offset

Register	Offset
UFMESR0	10A4h

Function

This is the uncorrectable fatal memory error status register 0.

Diagram



Fields

Field	Function
31 MBEE	Multi-bit ECC error 0: No fatal multi-bit ECC error has occurred. 1: A fatal multi-bit ECC error has occurred in internal memory as defined by MEM_ID. The function has entered a safe state: • PCIe bus master enable is cleared (0b0) • Station interface(s) disabled (SIMR[EN]=0b0) • Pseudo MAC ports bound to instance are disabled (POR[TXDIS/RXDIS]=0b1) • Ethernet-MAC ports bound to instance are disabled (PM*a*_COMMAND_CONFIG[RX_EN/TX_EN]=0b0) For recovery, S/W should reset the function by issuing a Function Level Reset (FLR). Write 1 to clear.
30 M	Multiple 0: One or none fatal multi-bit ECC error has occurred 1: Multiple fatal multi-bit ECC error events detected. The last event is captured.
29-21 —	Reserved
20-16 MEM_ID	Memory ID See Table 392 for list of memory identifiers.
15-11 —	Reserved
10-0 SYNDROME	Syndrome Identifies the first pertinent bit position (column) in memory that caused the error.

53.4.6.7.21 Uncorrectable fatal memory error status register 1 (UFMESR1)

Offset

Register	Offset
UFMESR1	10A8h

Function

This is the uncorrectable fatal memory error status register 1.

NOTE

A write to this read-only register will return an error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ADDR	Address Address information (row) of last ECC event.

53.4.6.7.22 Internal MDIO interrupt reason register (IMDIOIRR)

Offset

Register	Offset
IMDIOIRR	10E0h

Function

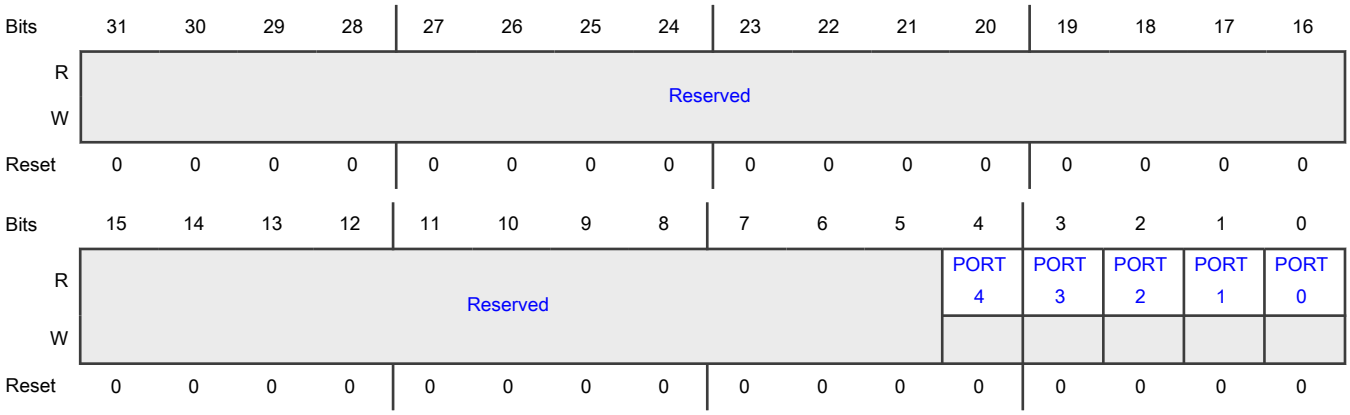
This is the internal MDIO interrupt reason register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	IMDIOIRR	—
ENETC1_COMMON	—	IMDIOIRR
SW0_COMMON	IMDIOIRR	—

Diagram



Fields

Field	Function
31-5 —	Reserved
4 PORT4	0: No port interrupt pending by Ethernet MAC MDIO 1: A port interrupt is pending by Ethernet MAC MDIO due to completion of an MDIO access. See PM0_MDIO_CFG. **NOTE** This field is not supported in every instance. The following table includes only supported registers.
3 PORT3	0: No port interrupt pending by Ethernet MAC MDIO 1: A port interrupt is pending by Ethernet MAC MDIO due to completion of an MDIO access. See PM0_MDIO_CFG.

Instance	Field supported in	Field not supported in
ENETC0_COMMON	—	IMDIOIRR
SW0_COMMON	IMDIOIRR	—

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>ENETC0_COMMON</td><td>—</td><td>IMDIOIRR</td></tr>
<tr><td>SW0_COMMON</td><td>IMDIOIRR</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>2
PORT2</td><td>0: No port interrupt pending by Ethernet MAC MDIO
1: A port interrupt is pending by Ethernet MAC MDIO due to completion of an MDIO access. See PM0_MDIO_CFG.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>ENETC0_COMMON</td><td>—</td><td>IMDIOIRR</td></tr>
<tr><td>SW0_COMMON</td><td>IMDIOIRR</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>1
PORT1</td><td>0: No port interrupt pending by Ethernet MAC MDIO
1: A port interrupt is pending by Ethernet MAC MDIO due to completion of an MDIO access. See PM0_MDIO_CFG.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>ENETC0_COMMON</td><td>—</td><td>IMDIOIRR</td></tr>
<tr><td>SW0_COMMON</td><td>IMDIOIRR</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>0
PORT0</td><td>0: No port interrupt pending by Ethernet MAC MDIO.
1: A port interrupt is pending by Ethernet MAC MDIO due to completion of an MDIO access. See PM0_MDIO_CFG.</td></tr>
</table>

53.4.6.7.23 Internal MDIO MSI-X vector register (IMDIOMSIVR)

Offset

Register	Offset
IMDIOMSIVR	10E4h

Function

This is the internal Ethernet MAC MDIO MSI-X vector control register. Number of vectors supported for the switch is defined in SCAPR0[NUM_MSIX]. Number of vectors supported for ENETC is defined in ECAPR1[NUM_MSIX].

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	IMDIOMSIVR	—
ENETC1_COMMON	—	IMDIOMSIVR
SW0_COMMON	IMDIOMSIVR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved										VECTOR					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5-0 VECTOR	Index into MSI-X address/data table. Vector used to handles all Ethernet MAC MDIO port interrupt. Range: 0..PSI0CFGR2[NUM_MSIX]-1

Table continued from the previous page...

Field	Function
	NOTE Specifying a vector number outside of the range will not generate an MSI-X write. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
ENETC0_COMMON	IMDIOMSIVR	—
SW0_COMMON	IMDIOMSIVR[3–0]	IMDIOMSIVR[5–4]

53.4.6.7.24 External MDIO interrupt reason register (EMDIOIRR)

Offset

Register	Offset
EMDIOIRR	10E8h

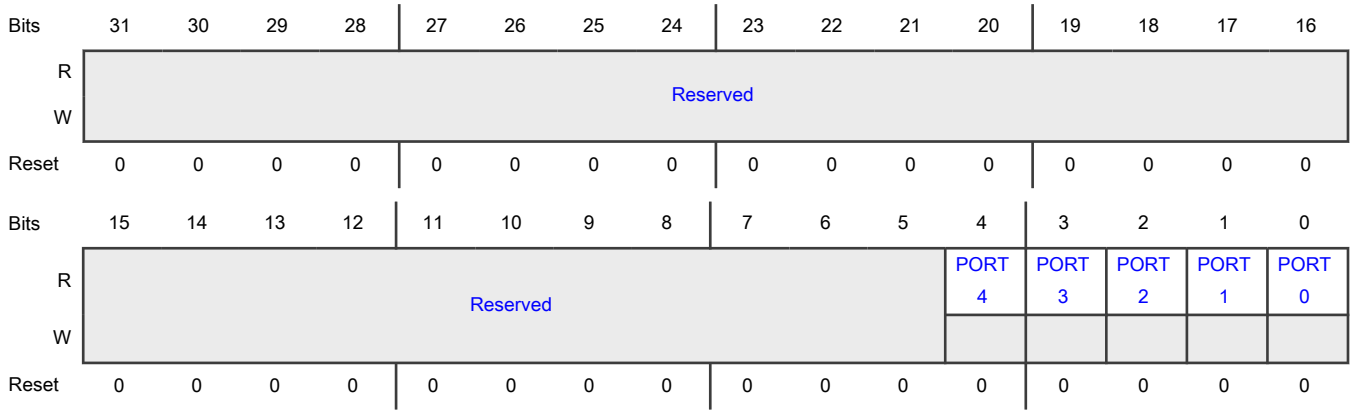
Function

This is the external MDIO interrupt reason register.

NOTE
Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	EMDIOIRR	—
ENETC1_COMMON	—	EMDIOIRR
SW0_COMMON	EMDIOIRR	—

Diagram



Fields

Field	Function
31-5 —	Reserved
4 PORT4	0: No port interrupt pending by EMDIO controller 1: A port interrupt is pending by EMDIO controller due to completion of a PHY MDIO access. **NOTE** This field is not supported in every instance. The following table includes only supported registers.
3 PORT3	0: No port interrupt pending by EMDIO controller 1: A port interrupt is pending by EMDIO controller due to completion of a PHY MDIO access. **NOTE** This field is not supported in every instance. The following table includes only supported registers.
2	0: No port interrupt pending by EMDIO controller

Instance	Field supported in	Field not supported in
ENETC0_COMMON	—	EMDIOIRR
SW0_COMMON	EMDIOIRR	—

Instance	Field supported in	Field not supported in
ENETC0_COMMON	—	EMDIOIRR
SW0_COMMON	EMDIOIRR	—

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>PORT2</td><td>1: A port interrupt is pending by EMDIO controller due to completion of a PHY MDIO access.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>ENETC0_COMMON</td><td>—</td><td>EMDIOIRR</td></tr><tr><td>SW0_COMMON</td><td>EMDIOIRR</td><td>—</td></tr></table></td></tr>
<tr><td>1
PORT1</td><td>0: No port interrupt pending by EMDIO controller
1: A port interrupt is pending by EMDIO controller due to completion of a PHY MDIO access.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>ENETC0_COMMON</td><td>—</td><td>EMDIOIRR</td></tr><tr><td>SW0_COMMON</td><td>EMDIOIRR</td><td>—</td></tr></table></td></tr>
<tr><td>0
PORT0</td><td>0: No port interrupt pending by EMDIO controller
1: A port interrupt is pending by EMDIO controller due to completion of a PHY MDIO access.</td></tr>
</table>

53.4.6.7.25 External MDIO MSI-X vector register (EMDIOMSIVR)

Offset

Register	Offset
EMDIOMSIVR	10ECh

Function

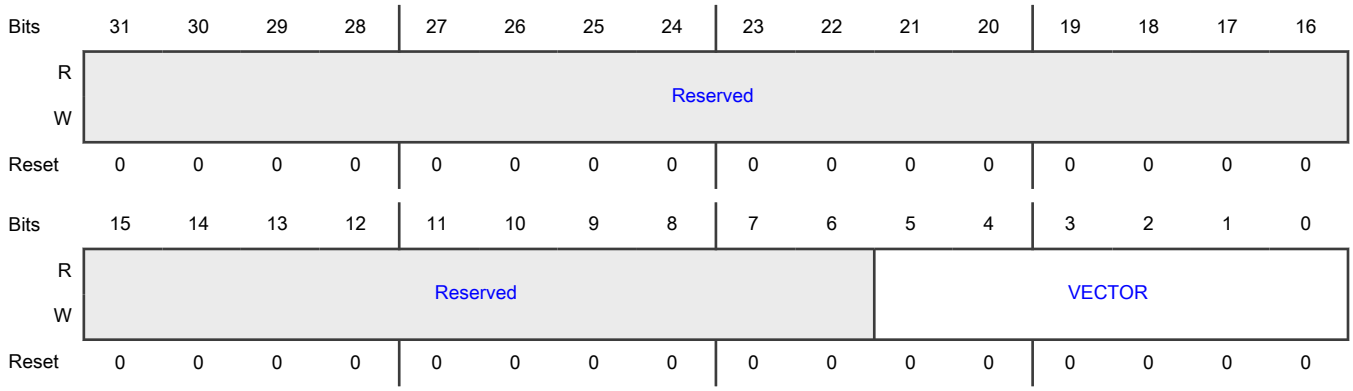
This is the external MDIO MSI-X vector control register. Number of vectors supported for the switch is defined in SCAPR0[NUM_MSIX]. Number of vectors supported for ENETC is defined in ECAPR1[NUM_MSIX].

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	EMDIOMSIVR	—
ENETC1_COMMON	—	EMDIOMSIVR
SW0_COMMON	EMDIOMSIVR	—

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 VECTOR	Index into MSI-X address/data table. Range: 0..PSI0CFGR2[NUM_MSIX]-1 **NOTE** Specifying a vector number outside of the range will not generate an MSI-X write. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
ENETC0_COMMON	EMDIOMSIVR	—
SW0_COMMON	EMDIOMSIVR[3–0]	EMDIOMSIVR[5–4]

53.4.6.7.26 Time capture configuration register (TCCR)

Offset

Register	Offset
TCCR	1100h

Function

This is the time capture configuration register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	TCCR
ENETC1_COMMON	—	TCCR
SW0_COMMON	TCCR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ARM		Reserved						TIMEOUT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TIMEOUT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 ARM	Indicates if Timestamp Capture function is armed. 0: unARMed 1: ARMed once. Timestamp capture is disarmed once the first time capture packet is successfully transmitted or once the timeout has expired. 2-3: Reserved
29-24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23-0 TIMEOUT	Indicates the duration time in nanoseconds the timestamp capture function is armed. If timestamp was not captured before timeout duration, an interrupt is generated. A timeout value of 0 implies no duration limit.

53.4.6.7.27 Time capture interrupt enable register (TCIER)

Offset

Register	Offset
TCIER	1104h

Function

This is the time capture interrupt enable register. When an interrupt enable bit is set and the corresponding detect bit in TCIDR, an interrupt is generated.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	TCIER
ENETC1_COMMON	—	TCIER
SW0_COMMON	TCIER	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TIMEOUT	TRANSMIT	Reserved													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TIMEOUT	Timeout interrupt This field is used to specify whether an interrupt is generated when a timeout event occurs. 0: No interrupt is generated 1: An interrupt is generated
30 TRANSMIT	Transmit interrupt This field is used to specify whether an interrupt is generated when a triggered frame is fully transmitted on all destined ports. 0: No interrupt is generated 1: An interrupt is generated
29-0 —	Reserved

53.4.6.7.28 Time capture receive port interrupt detect register (TCRPIDR)

Offset

Register	Offset
TCRPIDR	1108h

Function

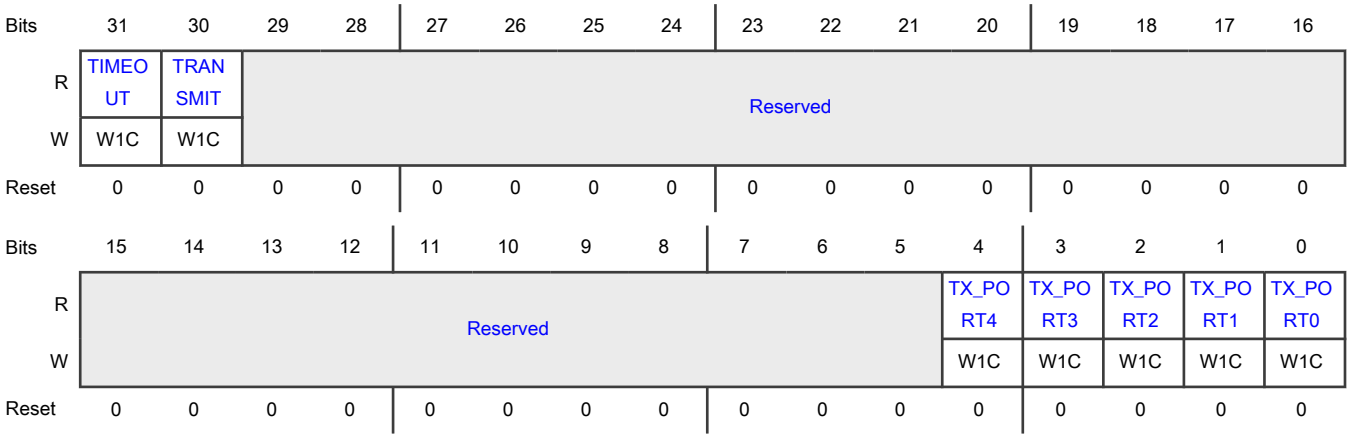
This is the time capture receive port interrupt detect register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	TCRPIDR
ENETC1_COMMON	—	TCRPIDR
SW0_COMMON	TCRPIDR	—

Diagram



Fields

Field	Function
31 TIMEOUT	Timeout interrupt This field indicates if a timeout interrupt has been generated. 0: No interrupt has been generated 1: An interrupt has been generated
30 TRANSMIT	Transmit interrupt This field indicates if an interrupt has been generated when a triggered frame is fully transmitted on all destined ports. 0: No interrupt has been generated 1: An interrupt has been generated
29-5 —	Reserved
4-0 TX_PORTn	Bit vector Indicating the transmit ports who captured the latency. Latency value is stored in the port's PTCMINLR and PTCMAXLR registers. Interrupt generated when the triggered frame is fully transmitted on all destined ports.

53.4.6.7.29 Time capture receive port status register (TCRPSR)

Offset

Register	Offset
TCRPSR	110Ch

Function

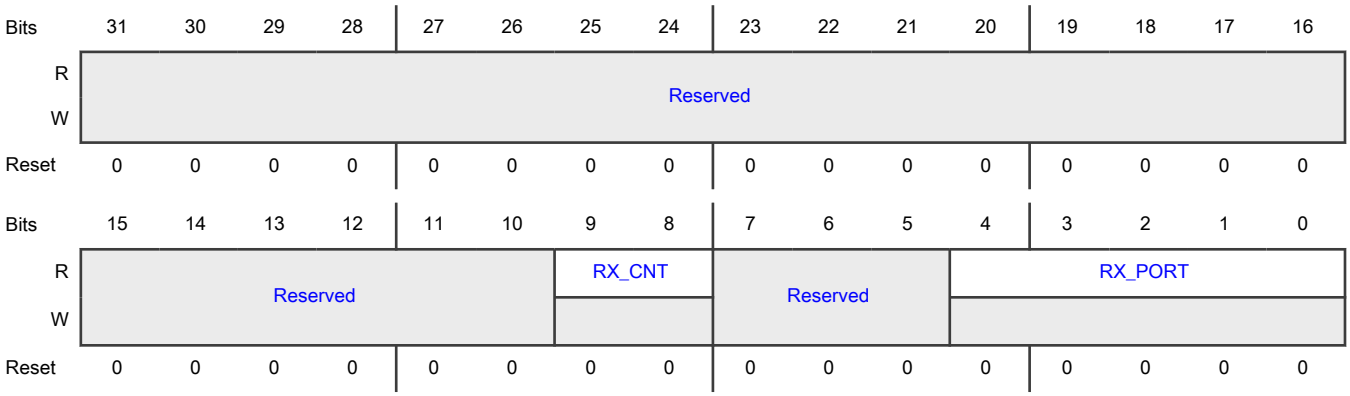
This is the time capture receive port status register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	TCRPSR
ENETC1_COMMON	—	TCRPSR
SW0_COMMON	TCRPSR	—

Diagram



Fields

Field	Function
31-10 —	Reserved
9-8 RX_CNT	Count the number of times the time capture function was triggered since it was ARMed. The counter is cleared when time capture is ARMed. The count does not roll over.
7-5 —	Reserved
4-0 RX_PORT	Receive port which captured the receive timestamp stored in TCRPTSR. The port is cleared when time capture is ARMed.

53.4.6.7.30 Time capture receive port timestamp register (TCRPTSR)

Offset

Register	Offset
TCRPTSR	1114h

Function

This is the time capture receive port timestamp register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	TCRPTSR
ENETC1_COMMON	—	TCRPTSR
SW0_COMMON	TCRPTSR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TIMESTAMP															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TIMESTAMP															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TIMESTAMP	Timestamp value in ns relative to SFD of the received frame. If received from pseudo MAC, timestamp value is relative to when the frame descriptor is transferred across the pseudo link. When the timestamp is captured, the TCRPSR[RX_CNT] register is incremented by 1 and the TCRPSR[RX_PORT] register is updated with the Rx port number of the trigger frame. Only the first frame (TCRPSR[RX_CNT] = 0) captures the Rx timestamp. Content is valid if TCRPSR[RX_CNT] is non-zero.

53.4.6.7.31 Time capture MSI-X vector register (TCMSIVR)

Offset

Register	Offset
TCMSIVR	1118h

Function

This is the time capture MSI-X vector register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	TCMSIVR
ENETC1_COMMON	—	TCMSIVR
SW0_COMMON	TCMSIVR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												VECTOR			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 VECTOR	Index into MSI-X address/data table. Vector used to handles timestamp capture interrupt sources. Range: 0..SCAPR0[NUM_MSIX]-1 **NOTE** Specifying a vector number outside of the range will not generate an MSI-X write.

53.4.6.7.32 Custom VLAN Ethertype register 1 (CVLANR1)

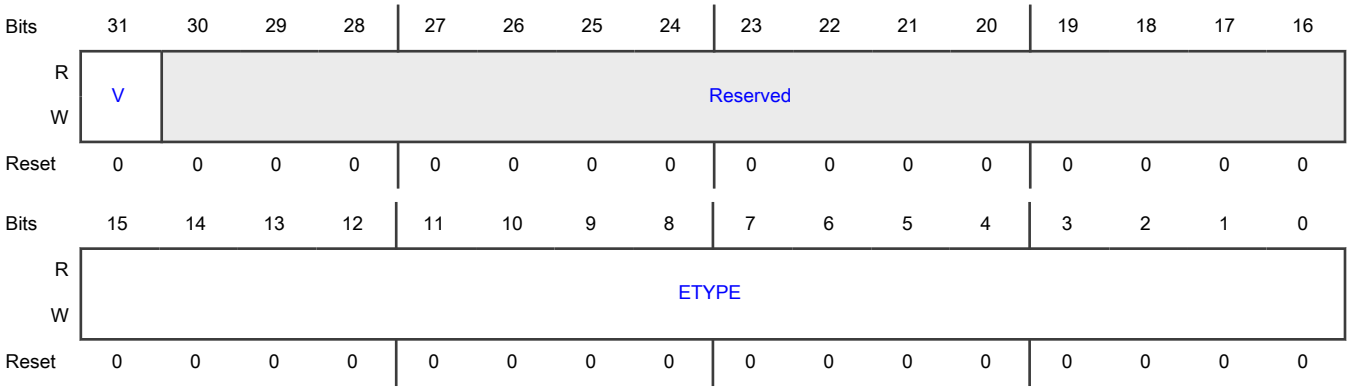
Offset

Register	Offset
CVLANR1	1200h

Function

This is the custom VLAN Ethertype register 1 that can be recognized as a VLAN tag by the parser.

Diagram



Fields

Field	Function
31 V	Valid 0: Not valid 1: Valid
30-16 —	Reserved
15-0 ETYPE	Ethertype

53.4.6.7.33 Custom VLAN Ethertype register 2 (CVLANR2)

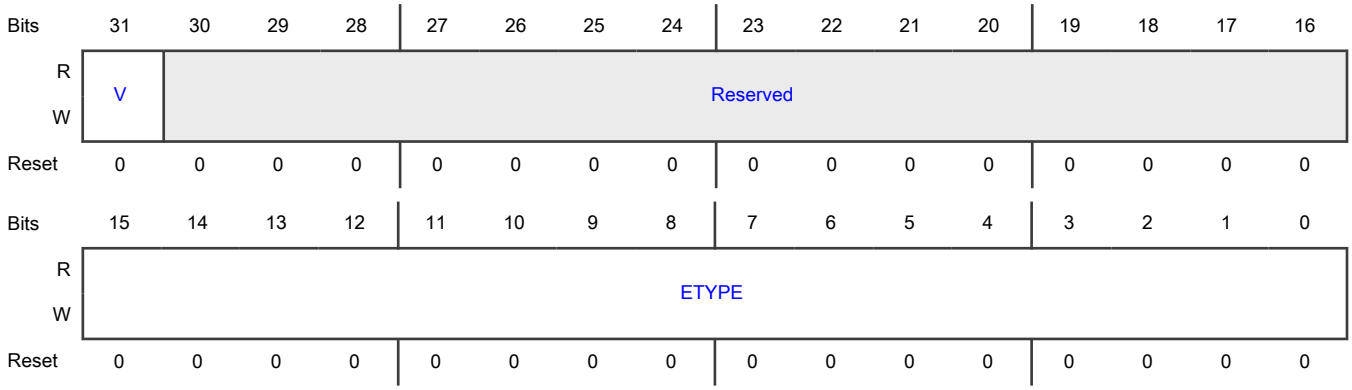
Offset

Register	Offset
CVLANR2	1204h

Function

This is the custom VLAN Ethertype register 2 that can be recognized as a VLAN tag by the parser.

Diagram



Fields

Field	Function
31 V	Valid 0: Not valid 1: Valid
30-16 —	Reserved
15-0 ETYPE	Ethertype

53.4.6.7.34 Pre-Standard RTAG Ethertype register (PSRTAGETR)

Offset

Register	Offset
PSRTAGETR	1208h

Function

This is the pre-Standard 802.1CB draft 2.0 R-TAG Ethertype register.

NOTE

Each module instance supports a different number of registers.

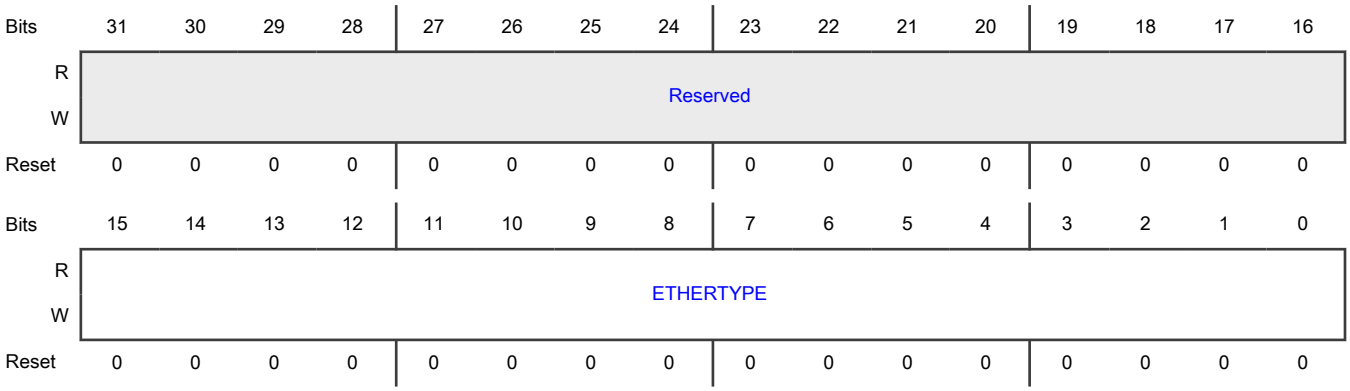
Instance	Register supported	Register not supported
ENETC0_COMMON	—	PSRTAGETR
ENETC1_COMMON	—	PSRTAGETR

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_COMMON	PSRTAGETR	—

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 ETHERTYPE	802.1CB draft 2.0 R-TAG Ethertype value. If this field is set to the standard 802.1CB R-TAG or HSR Ethertype value, hardware will treat the header as a HSR/R-TAG 6-byte header.

53.4.6.7.35 DoS L2 configuration register (DOSL2CR)

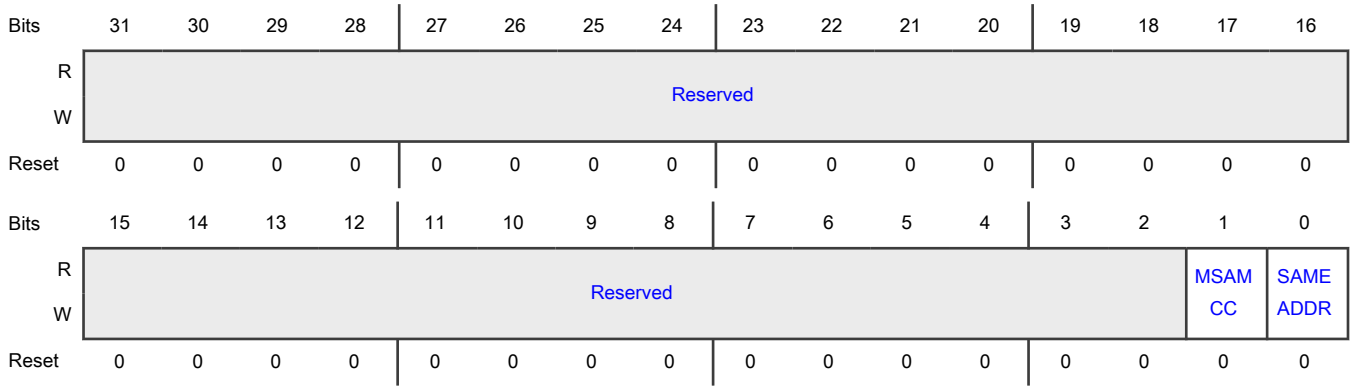
Offset

Register	Offset
DOSL2CR	1220h

Function

This is the DoS L2 configuration register.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 MSAMCC	This field specifies whether received frames with Multicast SMAC address are discarded. 0: Accepted 1: Discarded
0 SAMEADDR	This field specifies whether received frames with SMAC = DMAC are discarded. 0: Accepted 1: Discarded

53.4.6.7.36 VLAN to IPV mapping profile a register 0 (VLANIPVMP0R0 - VLANIPVMP1R0)

Offset

Register	Offset
VLANIPVMP0R0	1300h
VLANIPVMP1R0	1310h

Function

This is the VLAN PCP/DEI to Internal Priority Value (IPV) mapping profile register. When VLAN is present in the frame and the port is operating in VLAN enable mode (PQOSMR[VE]=1), the PCP+DEI field (PQOSMR[VS] determines which VLAN tag (inner/outer) to be used) is mapped to the IPV values as defined.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	VLANIPVMP0R0	VLANIPVMP1R0
ENETC1_COMMON	VLANIPVMP0R0	VLANIPVMP1R0
SW0_COMMON	VLANIPVMP0R0–VLANIPVMP1R0	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	PCP_DEI_7			Reserved	PCP_DEI_6			Reserved	PCP_DEI_5			Reserved	PCP_DEI_4		
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	PCP_DEI_3			Reserved	PCP_DEI_2			Reserved	PCP_DEI_1			Reserved	PCP_DEI_0		
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
VLANIPVMP0R0	ENETC0_COMMON–SW0_COMMON: 3322_1100h
VLANIPVMP1R0	3322_1100h

Fields

Field	Function
31 —	Reserved
30-28 PCP_DEI_7	IPV value used for receive data path.
27 —	Reserved
26-24 PCP_DEI_6	IPV value used for receive data path.
23 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
22-20 PCP_DEI_5	IPV value used for receive data path.
19 —	Reserved
18-16 PCP_DEI_4	IPV value used for receive data path.
15 —	Reserved
14-12 PCP_DEI_3	IPV value used for receive data path.
11 —	Reserved
10-8 PCP_DEI_2	IPV value used for receive data path.
7 —	Reserved
6-4 PCP_DEI_1	IPV value used for receive data path.
3 —	Reserved
2-0 PCP_DEI_0	IPV value used for receive data path.

53.4.6.7.37 VLAN to IPV mapping profile a register 1 (VLANIPVMP0R1 - VLANIPVMP1R1)

Offset

Register	Offset
VLANIPVMP0R1	1304h
VLANIPVMP1R1	1314h

Function

This is the VLAN PCP/DEI to Internal Priority Value (IPV) mapping register. When VLAN is present in the frame and the port is operating in VLAN enable mode (PQOSMR[VE]=1), the PCP+DEI field (PQOSMR[VS] determines which VLAN tag (inner/outer) to be used) is mapped to the IPV values as defined.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	VLANIPVMP0R1	VLANIPVMP1R1
ENETC1_COMMON	VLANIPVMP0R1	VLANIPVMP1R1
SW0_COMMON	VLANIPVMP0R1–VLANIPVMP1R1	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	PCP_DEI_15			Reserved	PCP_DEI_14			Reserved	PCP_DEI_13			Reserved	PCP_DEI_12		
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	PCP_DEI_11			Reserved	PCP_DEI_10			Reserved	PCP_DEI_9			Reserved	PCP_DEI_8		
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
VLANIPVMP0R1	ENETC0_COMMON–SW0_COMMON: 7766_5544h
VLANIPVMP1R1	7766_5544h

Fields

Field	Function
31 —	Reserved
30-28 PCP_DEI_15	IPV value used for receive data path.
27	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
26-24 PCP_DEI_14	IPV value used for receive data path.
23 —	Reserved
22-20 PCP_DEI_13	IPV value used for receive data path.
19 —	Reserved
18-16 PCP_DEI_12	IPV value used for receive data path.
15 —	Reserved
14-12 PCP_DEI_11	IPV value used for receive data path.
11 —	Reserved
10-8 PCP_DEI_10	IPV value used for receive data path.
7 —	Reserved
6-4 PCP_DEI_9	IPV value used for receive data path.
3 —	Reserved
2-0 PCP_DEI_8	IPV value used for receive data path.

53.4.6.7.38 VLAN to DR mapping profile a register (VLANDRMP0R - VLANDRMP1R)

Offset

Register	Offset
VLANDRMP0R	1308h
VLANDRMP1R	1318h

Function

This is the VLAN PCP/DEI to Drop Resilience (DR) mapping register. When VLAN is present in the frame and the port is operating in VLAN enable mode (PQOSMR[VE]=1), the PCP+DEI field (PQOSMR[VS] determines which VLAN tag (inner/outer) to be used) is mapped to the DR values as defined.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	VLANDRMP0R	VLANDRMP1R
ENETC1_COMMON	VLANDRMP0R	VLANDRMP1R
SW0_COMMON	VLANDRMP0R–VLANDRMP1R	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PCP_DEI_15		PCP_DEI_14		PCP_DEI_13		PCP_DEI_12		PCP_DEI_11		PCP_DEI_10		PCP_DEI_9		PCP_DEI_8	
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PCP_DEI_7		PCP_DEI_6		PCP_DEI_5		PCP_DEI_4		PCP_DEI_3		PCP_DEI_2		PCP_DEI_1		PCP_DEI_0	
Reset	See Register reset values.															

Register reset values

Register	Reset value
VLANDRMP0R	ENETC0_COMMON–SW0_COMMON: 8888_8888h
VLANDRMP1R	8888_8888h

Fields

Field	Function
31-30: PCP_DEI_15 29-28: PCP_DEI_14 27-26: PCP_DEI_13 25-24: PCP_DEI_12 23-22: PCP_DEI_11 21-20: PCP_DEI_10 19-18: PCP_DEI_9 17-16: PCP_DEI_8 15-14: PCP_DEI_7 13-12: PCP_DEI_6 11-10: PCP_DEI_5 9-8: PCP_DEI_4 7-6: PCP_DEI_3 5-4: PCP_DEI_2 3-2: PCP_DEI_1 1-0: PCP_DEI_0	DR value used for receive data path.

53.4.6.7.39 Ingress port filter capability register (IPFCAPR)

Offset

Register	Offset
IPFCAPR	1640h

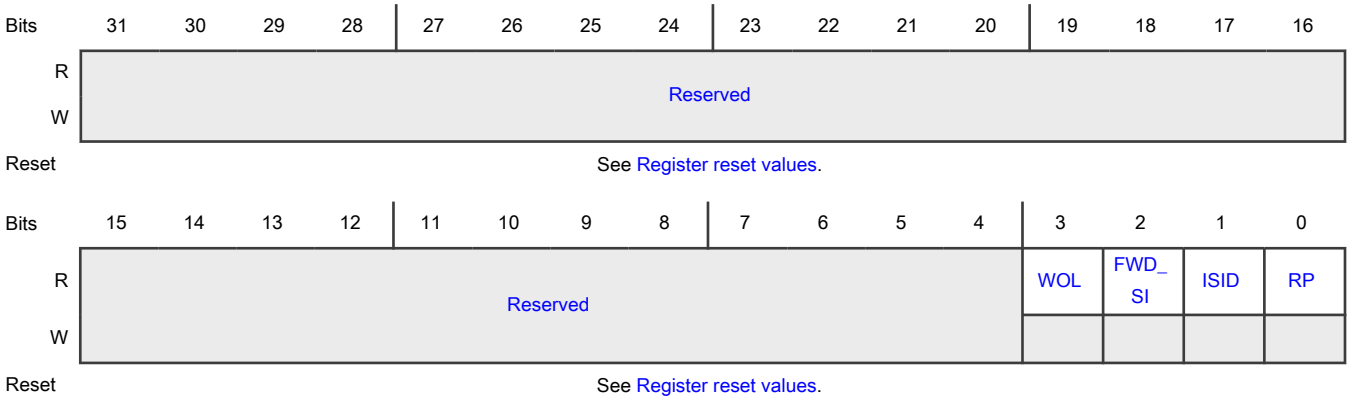
Function

This is the ingress port filter capability register. Specifies the set of additional actions the Port filter can perform. Basic actions supported are accept/reject the frame, override internal QoS and increment count when entry is hit.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
IPFCAPR	ENETC0_COMMON: 0000_000Fh ENETC1_COMMON: 0000_0007h SW0_COMMON: 0000_0003h

Fields

Field	Function
31-4 —	Reserved
3 WOL	Wake on LAN supported. 0: Not supported 1: Supported **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
ENETC0_COMMON	IPFCAPR	—
ENETC1_COMMON	IPFCAPR	—
SW0_COMMON	—	IPFCAPR

Table continued from the previous page...

Field	Function
2 FWD_SI	Forwarding to a set of Station Interfaces (SIs) supported. 0: Not supported 1: Supported **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
ENETC0_COMMON	IPFCAPR	—
ENETC1_COMMON	IPFCAPR	—
SW0_COMMON	—	IPFCAPR

1 ISID	Ingress Stream Identification supported. 0: Not supported 1: Supported See ISIDCAPR for more information.
0 RP	Rate Policer function supported 0: Not supported 1: Supported See RPCAPR for more information.

53.4.6.7.40 Ingress port filter table capability register (IPFTCAPR)

Offset

Register	Offset
IPFTCAPR	1644h

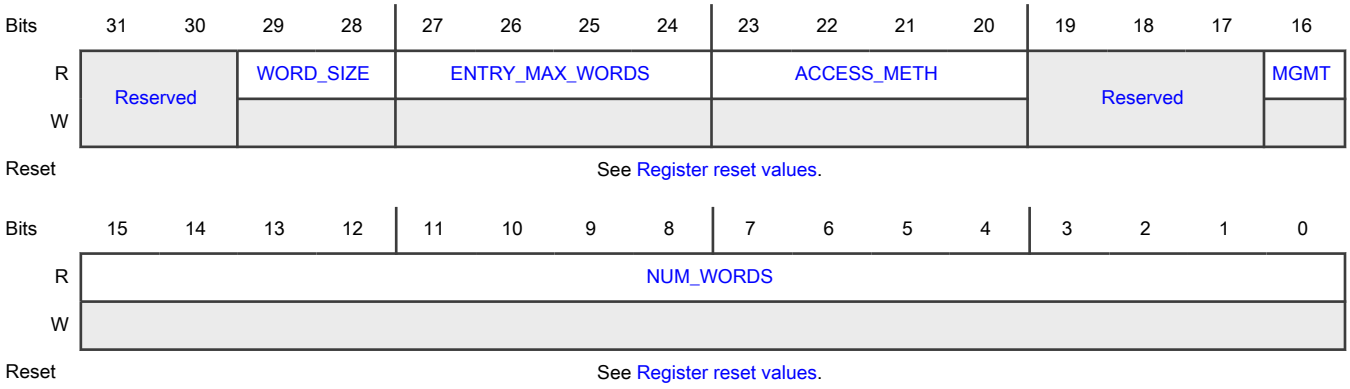
Function

This is the ingress port filter table capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
IPFTCAPR	ENETC0_COMMON,ENETC1_COMMON: 0ED1_001Ch SW0_COMMON: 0ED1_008Ch

Fields

Field	Function
31-30 —	Reserved
29-28 WORD_SIZE	Word size in bits of the ternary memory. 0: 48 bits 1-3: Reserved
27-24 ENTRY_MAX_WORDS	Maximum number of consecutive words which can form a TM Entry. **NOTE** Maximum Entry Key size is: ENTRY_MAX_WORDS * WORD_SIZE
23-20 ACCESS_METH	Indicates which Configuration Access Methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-17 —	Reserved
16	Indicates if table entries are managed by software driver or by hardware.

Table continues on the next page...

Table continued from the previous page...

Field	Function
MGMT	0: Managed by software 1: Managed by hardware
15-0 NUM_WORDS	Number of ternary memory words supported. **NOTE** Reset value from IERB register S*a*IPFTMAR/E*a*IPFTMAR when PCIe Memory Access bit is set (b1). All updates after this must be done through this register.

53.4.6.7.41 Ingress port filter table memory operational register (IPFTMOR)

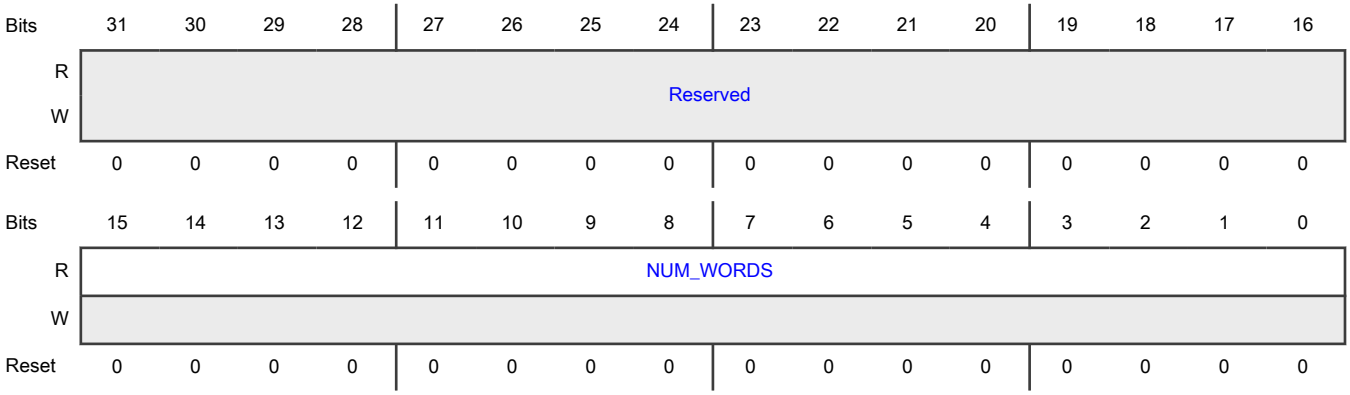
Offset

Register	Offset
IPFTMOR	1648h

Function

This is the ingress port filter table memory operational register

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	Number of words in-use.

53.4.6.7.42 Index table memory capability register (ITMCAPR)

Offset

Register	Offset
ITMCAPR	1800h

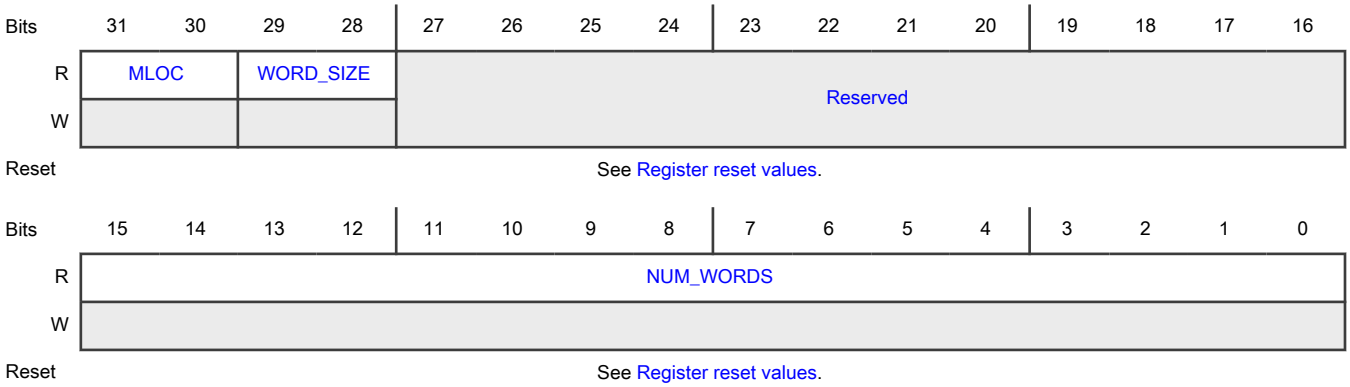
Function

This is the index table memory capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
ITMCAPR	ENETC0_COMMON: 0000_0050h ENETC1_COMMON: 0000_0030h SW0_COMMON: 0000_05D0h

Fields

Field	Function
31-30 MLOC	Indicates memory location of this table 0: Common memory 1: Reserved
29-28 WORD_SIZE	Word size in bytes. 0: 24 bytes

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1-3: Reserved
27-16 —	Reserved
15-0 NUM_WORDS	Number of Words in the Index table memory. This value is writable in IERB register S*al*TMAR/E*al*TMAR (by pre-boot initialization), and is immediately reflected here. **NOTE** Distribution of the Table memory to various tables is done via various Index Table Memory Allocation Registers (xITMAR).

53.4.6.7.43 Rate policer capability register (RPCAPR)

Offset

Register	Offset
RPCAPR	1810h

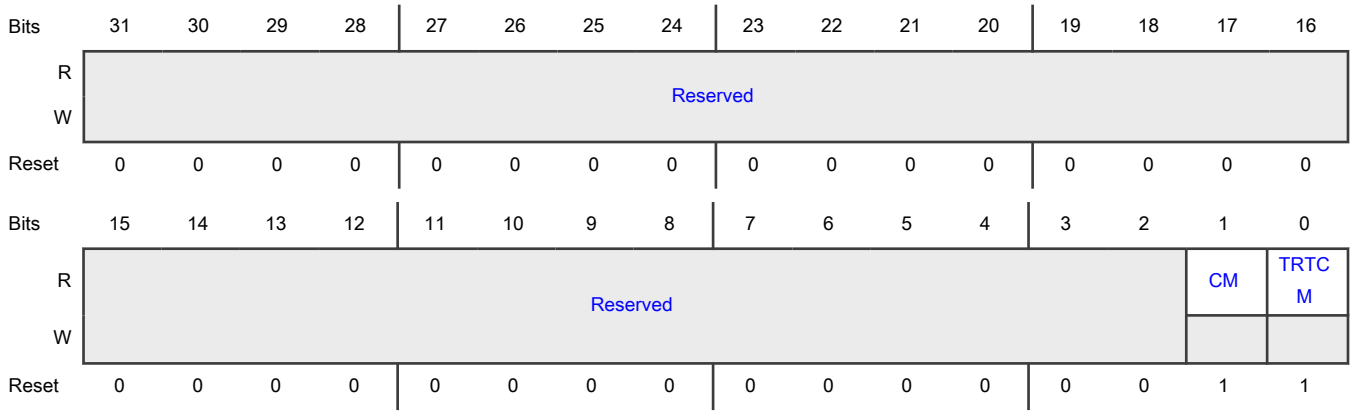
Function

This is the rate policer capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 CM	Color Mode capability 0: Support Color Blind mode only 1: Support Color Blind and Color Aware modes.
0 TRTCM	Two-Rate Three-Color Marker supported per MEF 10.3 standard. 0: Not supported 1: Supported

53.4.6.7.44 Rate policer index table capability register (RPITCAPR)

Offset

Register	Offset
RPITCAPR	1814h

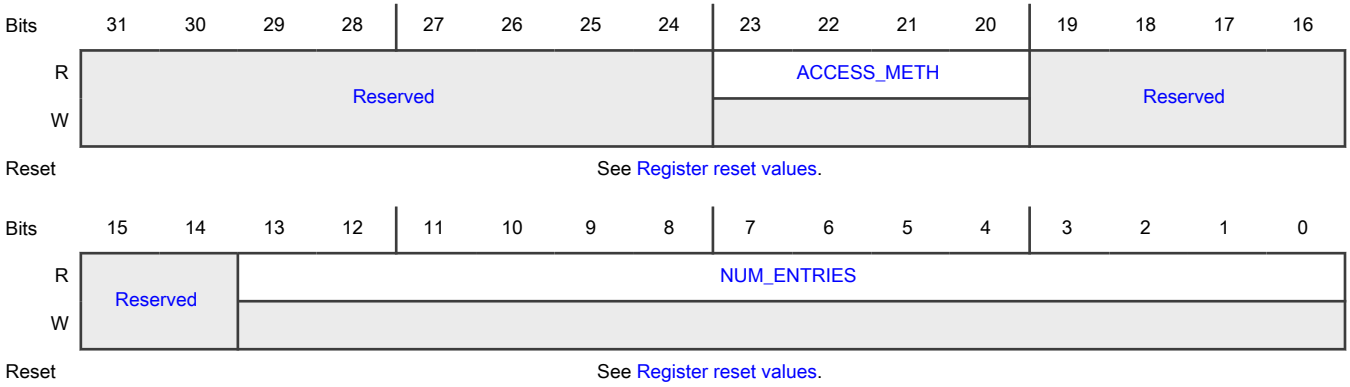
Function

This is the rate policer index table capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
RPITCAPR	ENETC0_COMMON: 0010_0002h ENETC1_COMMON: 0010_0004h SW0_COMMON: 0010_0020h

Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-14 —	Reserved
13-0 NUM_ENTRIES	The number of entries assigned to this table. Reset value is specified by ROUNDDOWN(RPITMAR/4).

53.4.6.7.45 Rate policer index table memory allocation register (RPITMAR)

Offset

Register	Offset
RPITMAR	1818h

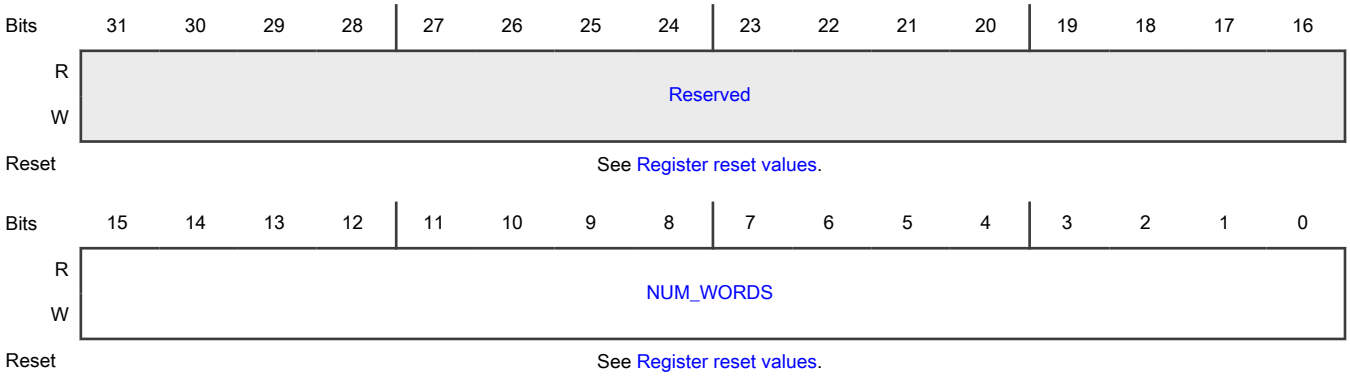
Function

This is the ENETC rate policer index table memory allocation register.

NOTE

ENETC will sample the value from IERB register E*a*RPITMAR when PCIe Memory Access bit is set (b1). All updates after this must be done through ENETC register.

Diagram



Register reset values

Register	Reset value
RPITMAR	ENETC0_COMMON: 0000_0008h ENETC1_COMMON: 0000_0010h SW0_COMMON: 0000_0080h

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. Number of entries indicated in RPITCAPR[NUM_ENTRIES] which is ROUNDDOWN(NUM_WORDS / 4). **NOTE** Each entry occupies 4 words.

53.4.6.7.46 Rate policer index table operational register (RPITOR)

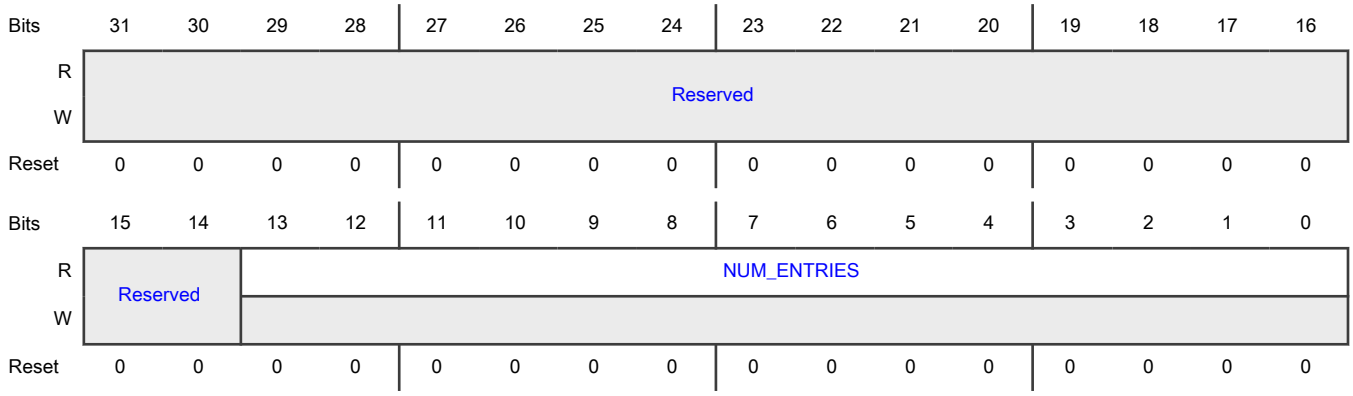
Offset

Register	Offset
RPITOR	181Ch

Function

This is the ENETC rate policer index table operational register.

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 NUM_ENTRIES	The number of entries in-use by this table.

53.4.6.7.47 Ingress stream counter index table capability register (ISCITCAPR)

Offset

Register	Offset
ISCITCAPR	1824h

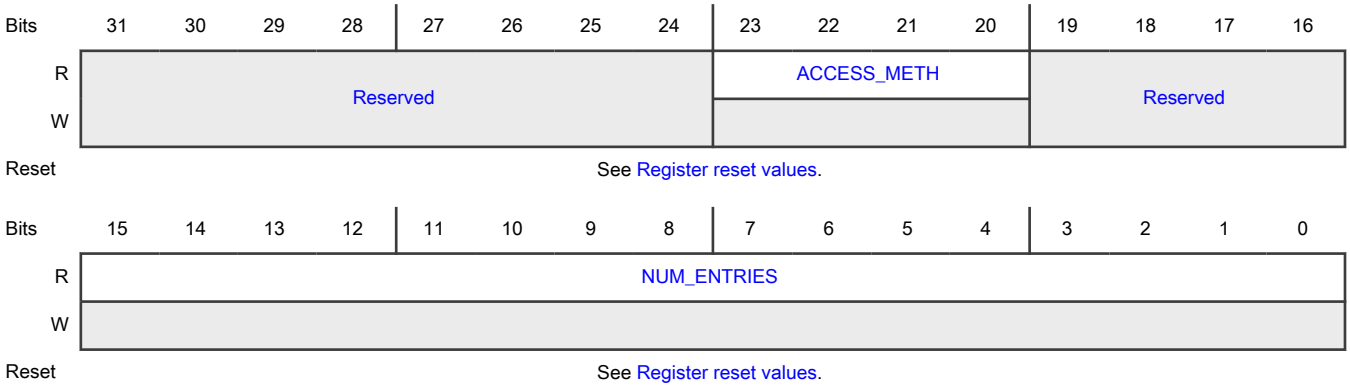
Function

This is the ingress stream counter index table capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
ISCITCAPR	ENETC0_COMMON: 0010_0008h ENETC1_COMMON: 0010_0010h SW0_COMMON: 0010_0180h

Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-16 —	Reserved
15-0 NUM_ENTRIES	The number of entries assigned to this table. **NOTE** The reset value of this field comes from ISCITMAR.

53.4.6.7.48 Ingress stream counter index table memory allocation register (ISCITMAR)

Offset

Register	Offset
ISCITMAR	1828h

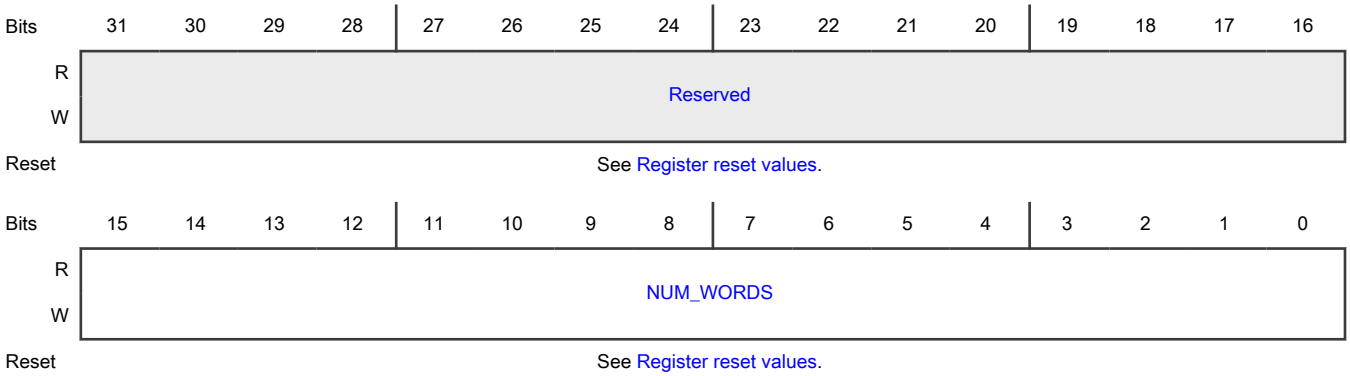
Function

This is the ingress stream counter index table memory allocation register.

NOTE

The switch will sample the value from IERB register S*a*ISCITMAR when PCIe Memory Access bit is set (b1). All updates after this must be done through switch register.

Diagram



Register reset values

Register	Reset value
ISCITMAR	ENETC0_COMMON: 0000_0008h ENETC1_COMMON: 0000_0010h SW0_COMMON: 0000_0180h

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** Each Entry consist of 1 word.

53.4.6.7.49 Ingress stream counter index table operational register (ISCITOR)

Offset

Register	Offset
ISCITOR	182Ch

Function

This is the ingress stream counter index table operational register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NUM_ENTRIES															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_ENTRIES	The number of entries allocated / in-use by this table.

53.4.6.7.50 Ingress stream capability register (ISCAPR)

Offset

Register	Offset
ISCAPR	1830h

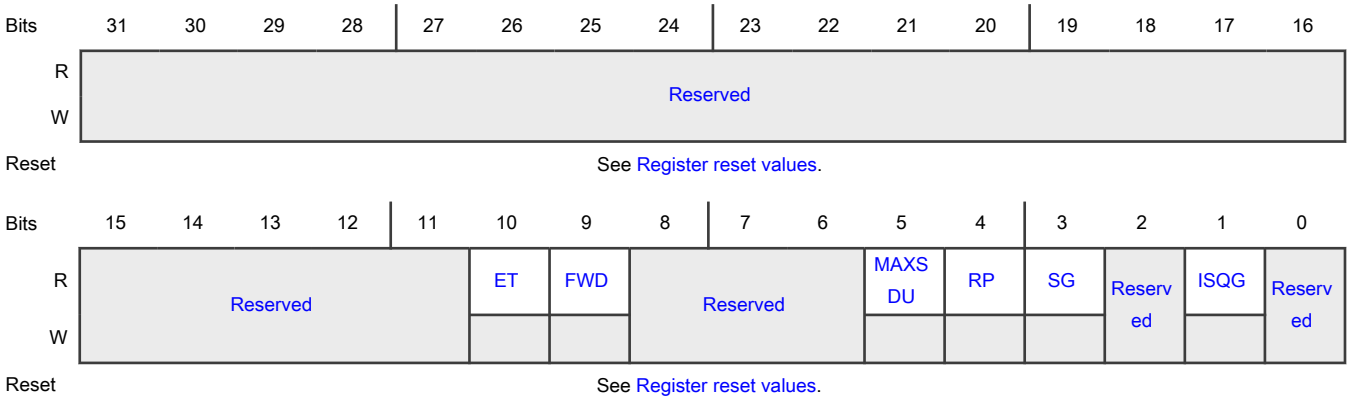
Function

This is the ingress stream table entry capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
ISCAPR	ENETC0_COMMON,ENETC1_COMMON: 0000_0238h SW0_COMMON: 0000_063Ah

Fields

Field	Function
31-11 —	Reserved
10 ET	Egress Treatment table entries specification supported. 0: Not supported 1: Supported **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
ENETC0_COMMON	—	ISCAPR
ENETC1_COMMON	—	ISCAPR
SW0_COMMON	ISCAPR	—

9 FWD	When set, can specify a set of destination to forward the frame.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8-6 —	Reserved
5 MAXSDU	Maximum SDU check supported. 0: Not supported 1: Supported Various SDUs supported are specified in IPCAPR.
4 RP	Rate Policer function specification supported. 0: Not supported 1: Supported See RPCAPR for more information.
3 SG	Stream Gating specification is supported. 0: Not supported 1: Supported See SGCAPR for more information.
2 —	Reserved
1 ISQG	Ingress Sequence Generation specification supported. 0: Not supported 1: Supported **NOTE** Refer to SCAPR1[SQ_TAGS] to determine which sequence tags are supported. **NOTE** This field is not supported in every instance. The following table includes only supported registers.
0	Reserved

Instance	Field supported in	Field not supported in
ENETC0_COMMON	—	ISCAPR
ENETC1_COMMON	—	ISCAPR
SW0_COMMON	ISCAPR	—

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

53.4.6.7.51 Ingress stream index table capability register (ISITCAPR)

Offset

Register	Offset
ISITCAPR	1834h

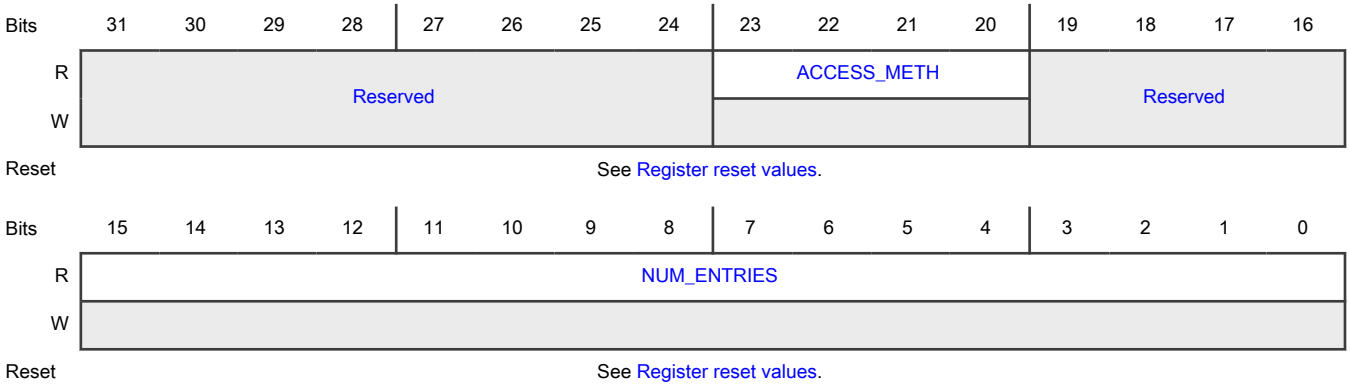
Function

This is the ingress stream index table capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
ISITCAPR	ENETC0_COMMON: 0010_0008h ENETC1_COMMON: 0010_0010h SW0_COMMON: 0010_0180h

Fields

Field	Function
31-24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-16 —	Reserved
15-0 NUM_ENTRIES	The number of entries assigned to this table. **NOTE** The reset value of this field comes from ISITMAR.

53.4.6.7.52 Ingress stream index table memory allocation register (ISITMAR)

Offset

Register	Offset
ISITMAR	1838h

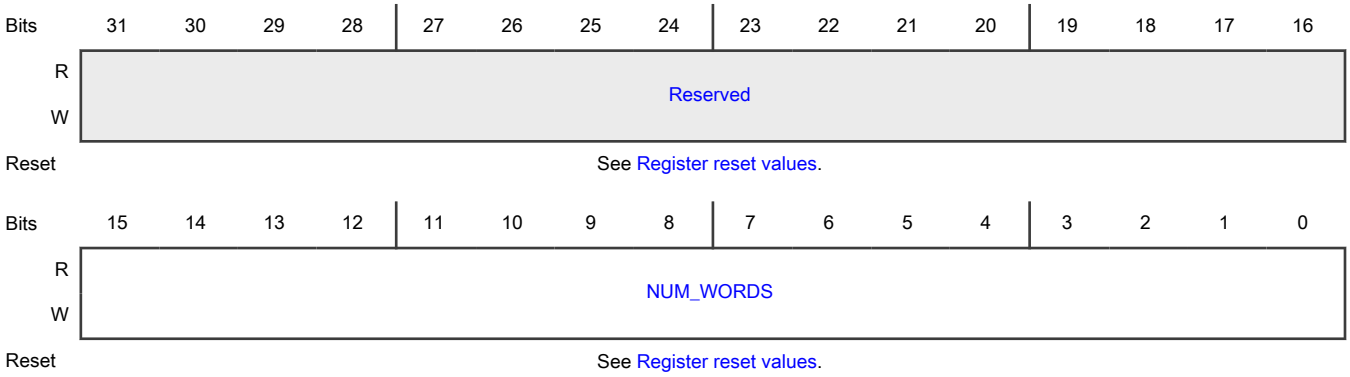
Function

This is the ingress stream index table memory allocation register.

NOTE

The function will sample the value from IERB register S*a*ISITMAR/E*a*ISITMAR when PCIe Memory Access bit is set (b1). All updates after this must be done through this register.

Diagram



Register reset values

Register	Reset value
ISITMAR	ENETC0_COMMON: 0000_0008h ENETC1_COMMON: 0000_0010h SW0_COMMON: 0000_0180h

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** Each Entry consist of 1 word.

53.4.6.7.53 Ingress stream index table operational register (ISITOR)

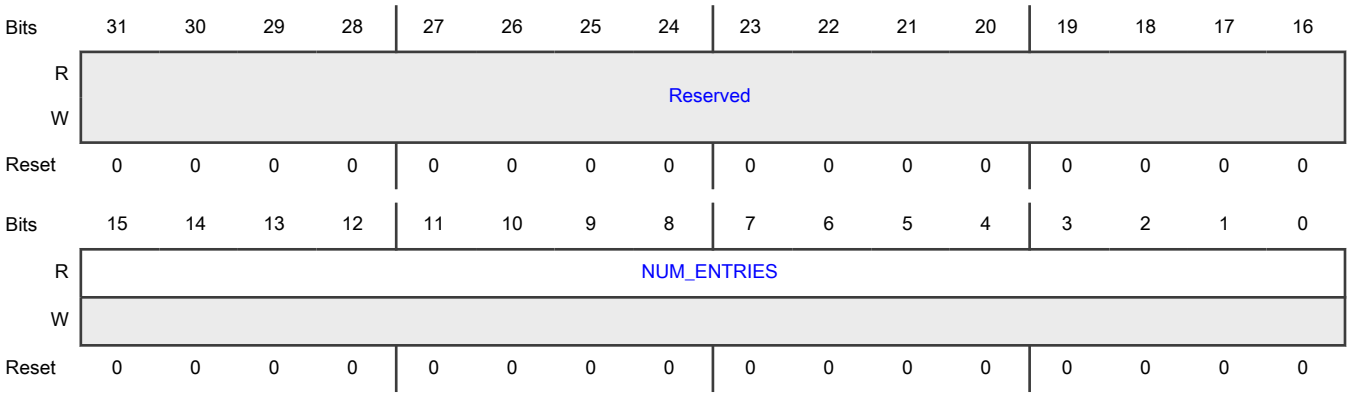
Offset

Register	Offset
ISITOR	183Ch

Function

This is the ingress stream index table operational register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_ENTRIES	The number of entries in-use by this table.

53.4.6.7.54 Ingress sequence generation index table capability register (ISQGITCAPR)

Offset

Register	Offset
ISQGITCAPR	1844h

Function

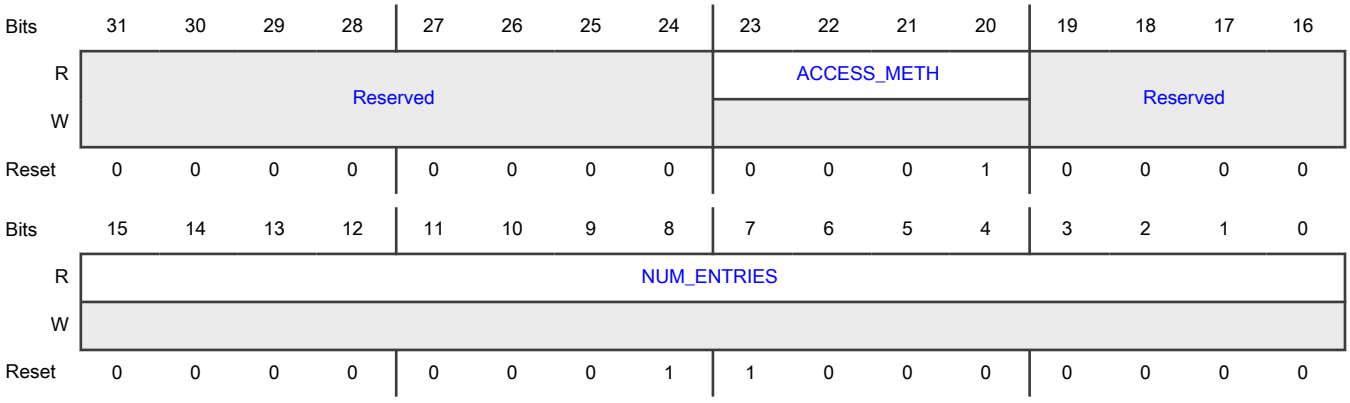
This is the ingress sequence generation index table capability register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	ISQGITCAPR
ENETC1_COMMON	—	ISQGITCAPR
SW0_COMMON	ISQGITCAPR	—

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-16 —	Reserved
15-0 NUM_ENTRIES	The number of entries assigned to this table. **NOTE** Reset value is determined by IERB register SISQGITMAR[NUM_WORDS] * 8.

53.4.6.7.55 Ingress sequence generation index table memory allocation register (ISQGITMAR)

Offset

Register	Offset
ISQGITMAR	1848h

Function

This is the ingress sequence generation index table memory allocation register.

NOTE

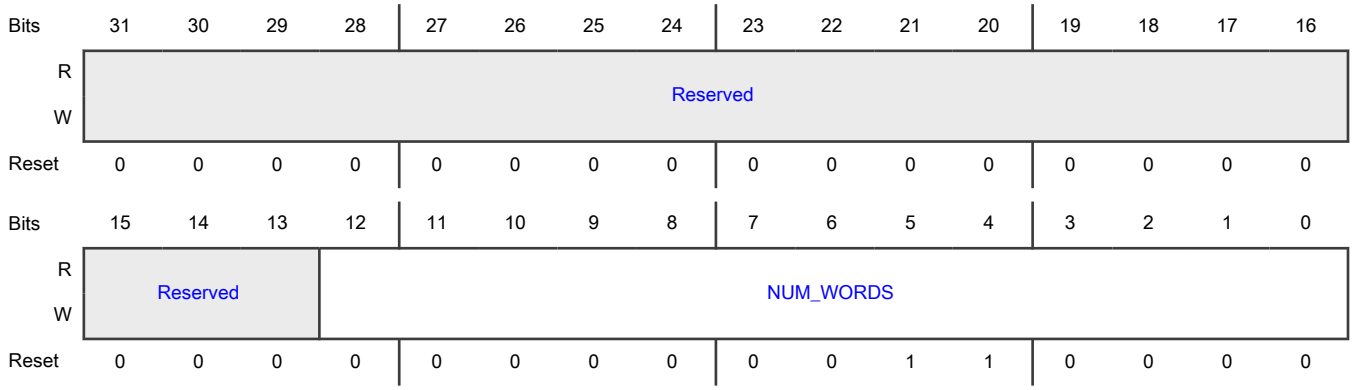
The function will sample the value from IERB register S*a*ISQGITMAR/E*a*ISQGITMAR when PCIe Memory Access bit is set (b1). All updates after this must be done through this register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	ISQGITMAR
ENETC1_COMMON	—	ISQGITMAR
SW0_COMMON	ISQGITMAR	—

Diagram



Fields

Field	Function
31-13 —	Reserved
12-0 NUM_WORDS	The number of words from index table memory assigned to this table. Number of entries assigned to the table is 8 * NUM_WORDS. **NOTE** There are 8 entries per word.

53.4.6.7.56 Ingress sequence generation index table operational register (ISQGITOR)

Offset

Register	Offset
ISQGITOR	184Ch

Function

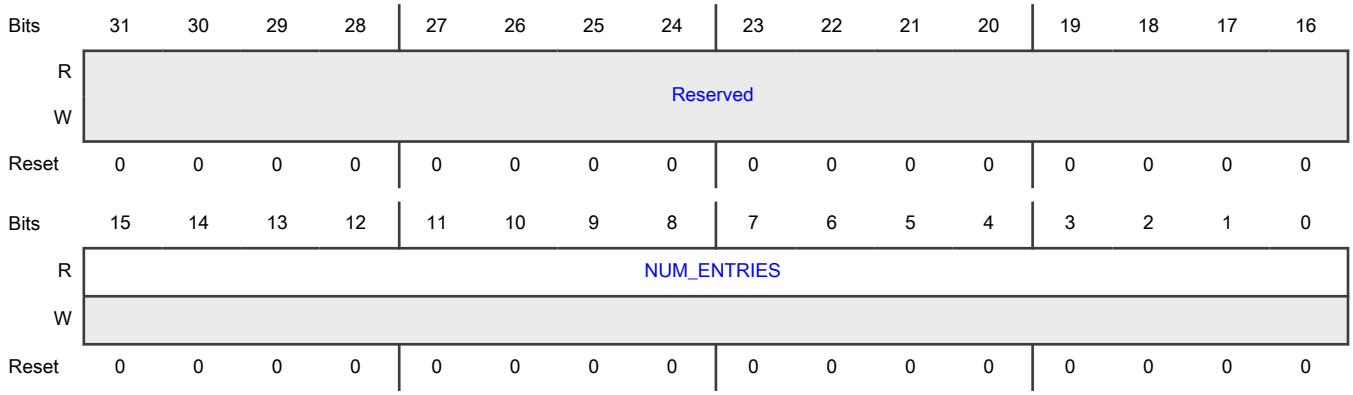
This is the ingress sequence generation index table operational register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	ISQGITOR
ENETC1_COMMON	—	ISQGITOR
SW0_COMMON	ISQGITOR	—

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_ENTRIES	The number of entries allocated / in-use by this table.

53.4.6.7.57 Stream gate capability register (SGCAPR)

Offset

Register	Offset
SGCAPR	1860h

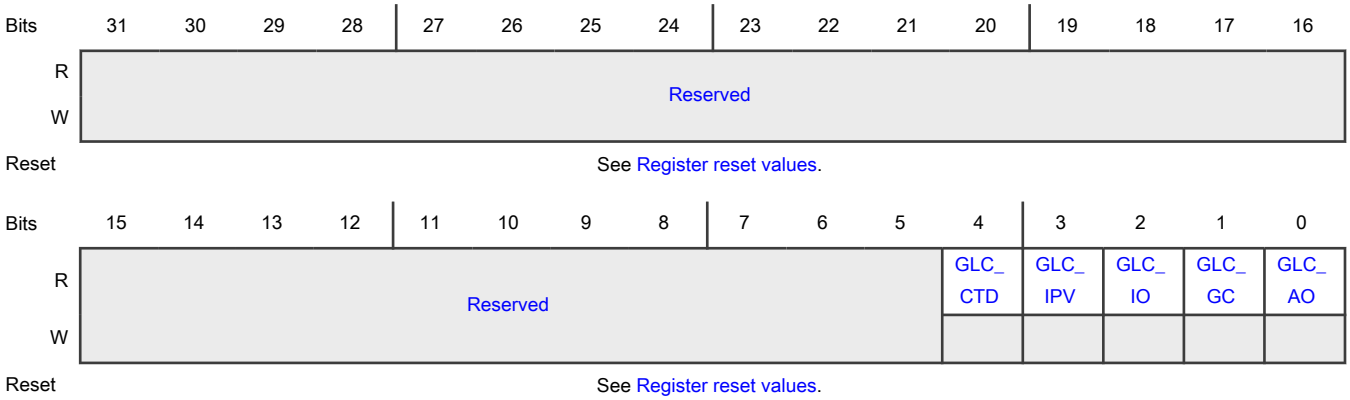
Function

This is the stream gate capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
SGCAPR	ENETC0_COMMON,ENETC1_COMMON: 0000_0007h SW0_COMMON: 0000_001Fh

Fields

Field	Function
31-5 —	Reserved
4 GLC_CTD	Each Gate Control List Entry supports configurable CTD (Cut-Through Disable state). 0: Not supported 1: Supported
3 GLC_IPV	Each Gate Control List Entry supports configurable IPV. 0: Not supported 1: Supported
2 GLC_IO	Each Gate Control List Entry supports Interval Max Octet check. 0: Not supported 1: Supported
1 GLC_GC	Support configurable option indicating if GCL's Gate Check is from SFD only or SFD until EOF. 0: SFD (Start of Frame Delimiter) only 1: configure SFD only or from SFD until EOF (End of Frame).
0	Support Administrative and Operational Gate Control List. 0: Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
GLC_AO	1: Supported

53.4.6.7.58 Stream gate instance index table capability register (SGIITCAPR)

Offset

Register	Offset
SGIITCAPR	1864h

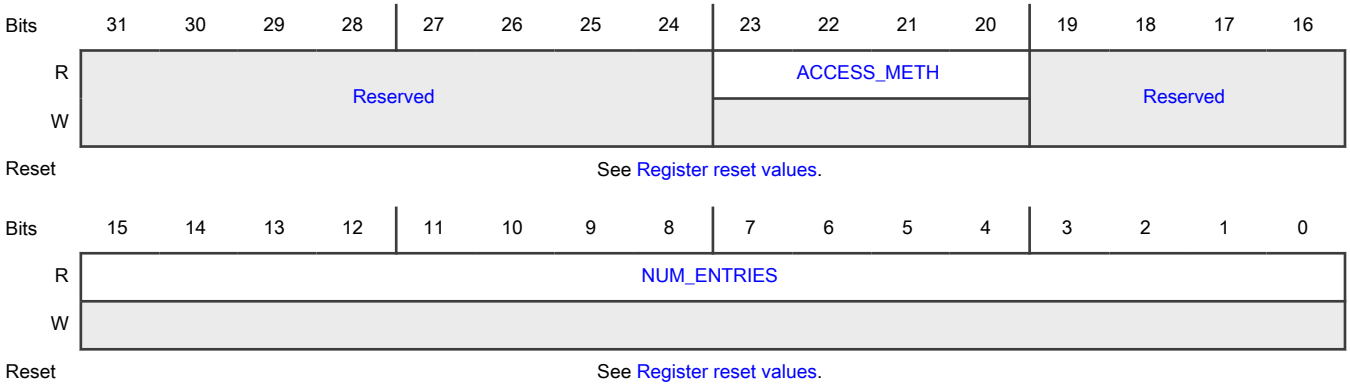
Function

This is the stream gate instance index table capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
SGIITCAPR	ENETC0_COMMON: 0010_0008h ENETC1_COMMON: 0010_0000h SW0_COMMON: 0010_0020h

Fields

Field	Function
31-24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-16 —	Reserved
15-0 NUM_ENTRIES	The number of entries assigned to this table.

53.4.6.7.59 Stream gate instance index table memory allocation register (SGIITMAR)

Offset

Register	Offset
SGIITMAR	1868h

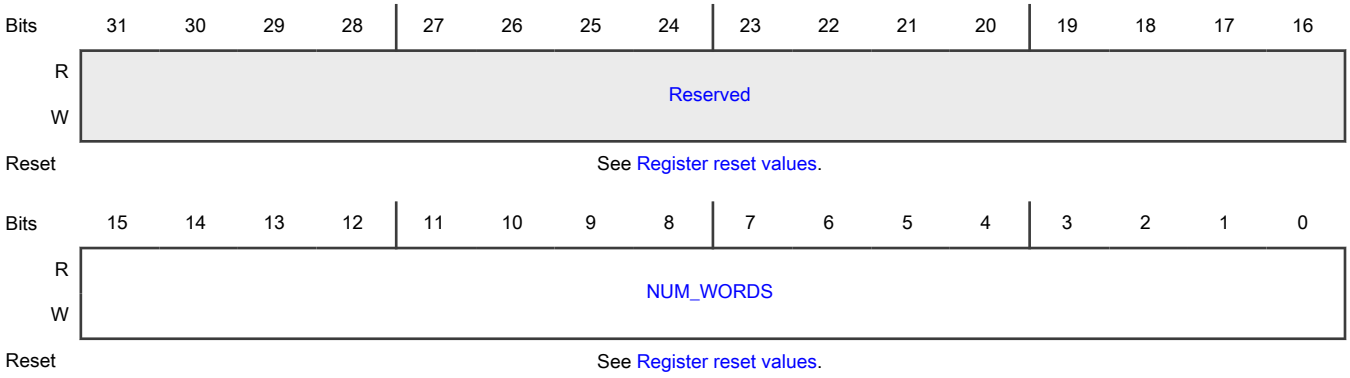
Function

This is the stream gate instance index table memory allocation register.

NOTE

The function will sample the value from IERB register S*a*SGIITMAR/E*a*SGIITMAR when PCIe Memory Access bit is set (b1). All updates after this must be done through this register.

Diagram



Register reset values

Register	Reset value
SGIITMAR	ENETC0_COMMON: 0000_0008h ENETC1_COMMON: 0000_0000h SW0_COMMON: 0000_0020h

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** Each entry occupies 1 word.

53.4.6.7.60 Stream gate instance index table operational register (SGIITOR)

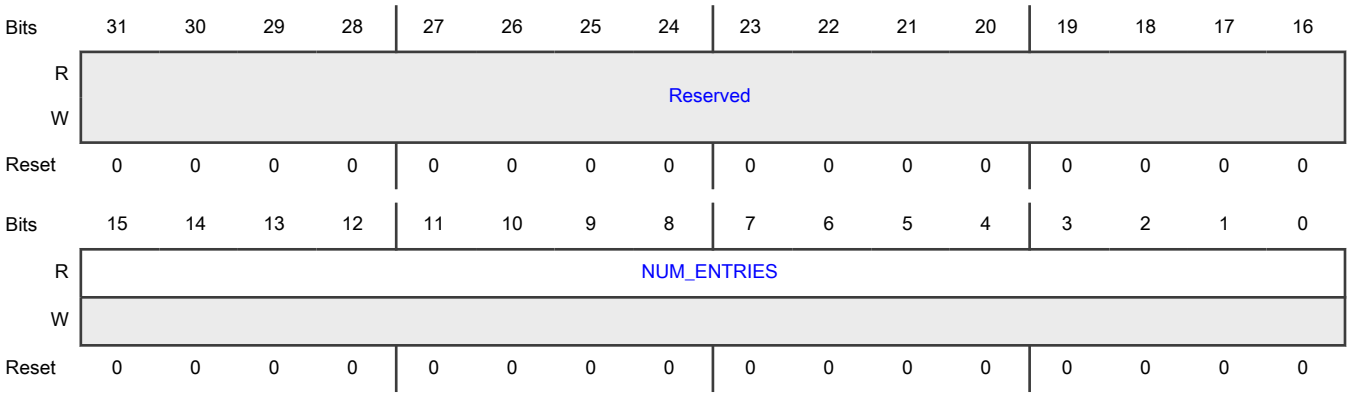
Offset

Register	Offset
SGIITOR	186Ch

Function

This is the stream gate instance index table operational register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_ENTRIES	The number of entries allocated / in-use by this table.

53.4.6.7.61 Stream gate control list index table capability register (SGCLITCAPR)

Offset

Register	Offset
SGCLITCAPR	1874h

Function

This is the stream gate control list index table capability register.

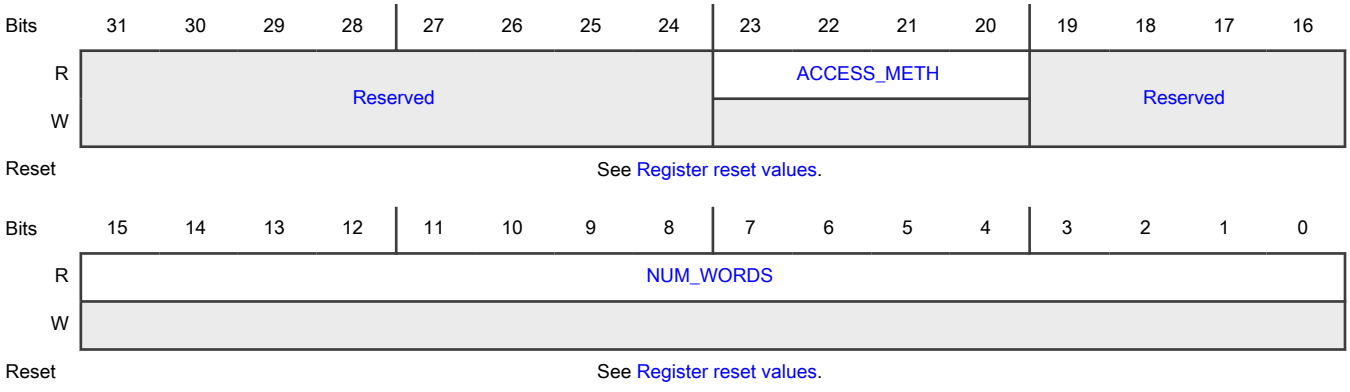
Basic capabilities are:

- each Gate Control List Entry support a configurable number of gates not exceeding 256.
- The SGL Index Table is managed by Software Driver
- The size of each gate is 1/2 a word.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
SGCLITCAPR	ENETC0_COMMON: 0010_0030h ENETC1_COMMON: 0010_0000h SW0_COMMON: 0010_00C0h

Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. Number of entries is variable depending on the number of gates in the list (NUM_GATES) specified. Entry size in words is calculated as: 1 + ROUNDUP(NUM_GATES * 0.5).

53.4.6.7.62 Stream gate control list index table memory allocation register (SGCLITMAR)

Offset

Register	Offset
SGCLITMAR	1878h

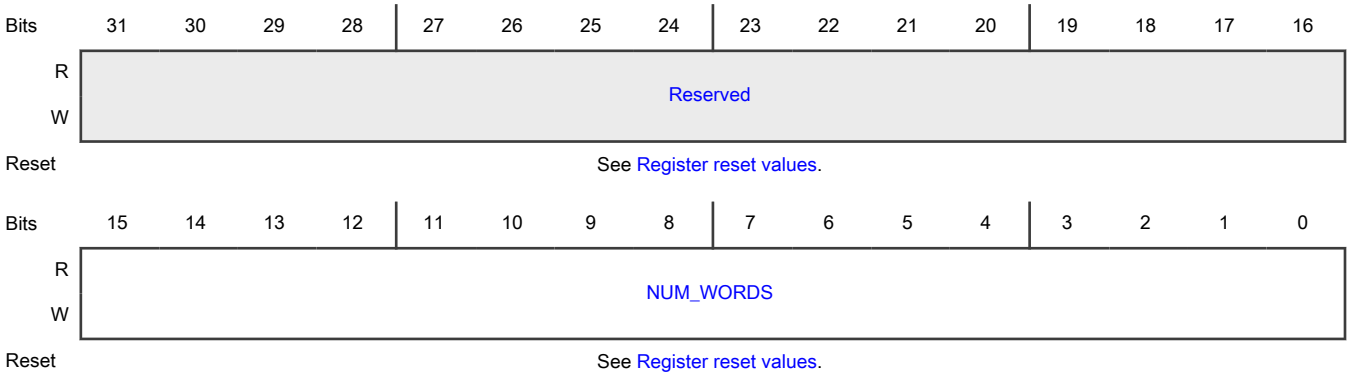
Function

This is the stream gate control list index table memory allocation register.

NOTE

The function will sample the value from IERB register S*a*SGCLITMAR/E*a*SGCLITMAR when PCIe Memory Access bit is set (b1). All updates after this must be done through this register.

Diagram



Register reset values

Register	Reset value
SGCLITMAR	ENETC0_COMMON: 0000_0030h ENETC1_COMMON: 0000_0000h SW0_COMMON: 0000_00C0h

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** Each entry utilizes a variable number of words depending on the number of gates specified. Entry size in words is calculated as: 1 + ROUNDUP(NUM_GATES * 0.5).

53.4.6.7.63 Stream gate control list table memory operational register (SGCLTMOR)

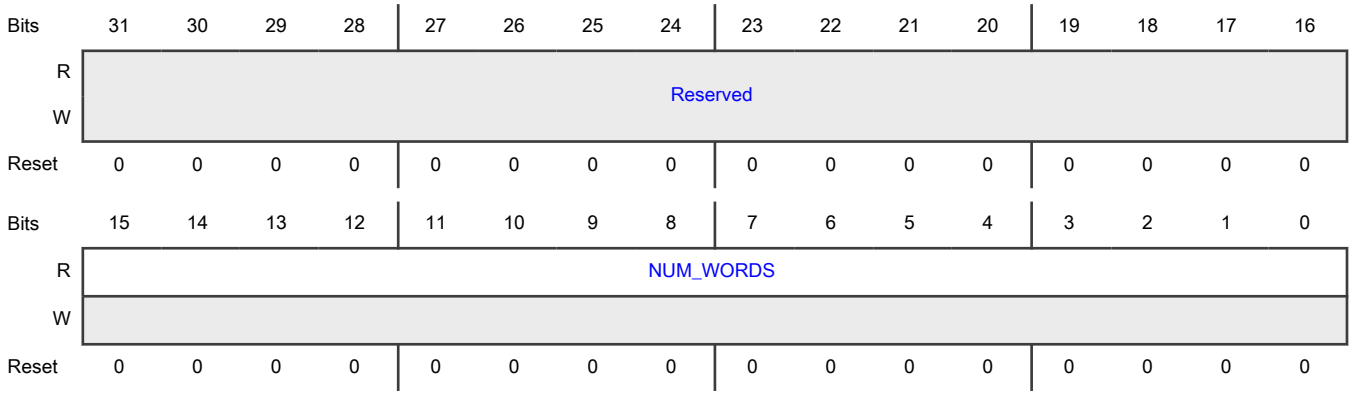
Offset

Register	Offset
SGCLTMOR	187Ch

Function

This is the stream gate control list table memory operational register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	Number of words in-use.

53.4.6.7.64 Frame modification ingress capability register (FMICAPR)

Offset

Register	Offset
FMICAPR	1880h

Function

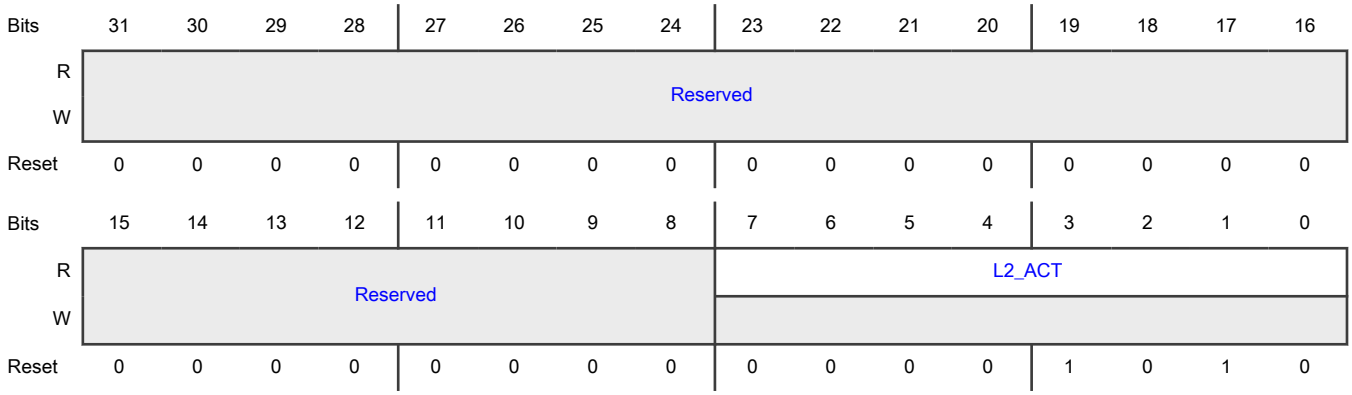
This is the frame modification ingress capability register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	FMICAPR
ENETC1_COMMON	—	FMICAPR
SW0_COMMON	FMICAPR	—

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 L2_ACT	Layer 2 frame modification actions supported xxxxxx1x: Replace MAC Destination Address with configurable value or from frame's MAC SA. xxxx1xxx: Delete, add or replace 1 VLAN tag All other settings reserved.

53.4.6.7.65 Frame modification egress capability register (FMECAPR)

Offset

Register	Offset
FMECAPR	1884h

Function

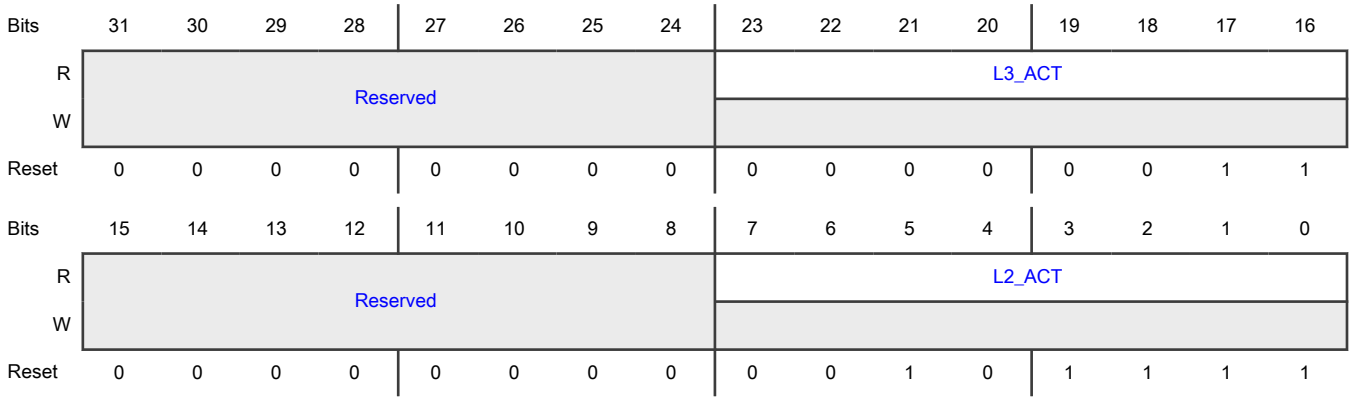
This is the frame modification egress capability register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	FMECAPR
ENETC1_COMMON	—	FMECAPR
SW0_COMMON	FMECAPR	—

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 L3_ACT	Layer 3 frame modification actions Actions supported are: xxxxxxx1: Replace Ethernet payload at a configured offset up to 128 bytes xxxxxx1x: Replace entire Ethernet payload All other settings reserved.
15-8 —	Reserved
7-0 L2_ACT	Layer 2 frame modification actions Actions supported are: xxxxxxx1: Replace entire frame with a new frame (CRC not present in replaced frame) xxxxxx1x: Replace MAC Destination Address with configurable value or from frame's MAC SA. xxxxx1xx: Replace MAC Source Address with frame's MAC DA or from port's MAC Register (PpMAR0/1). xxxx1xxx: Delete, add or replace 1 VLAN tag xx1xxxxx: Replace, insert or delete R-TAG/HSR tag. Note: Replacement of R-TAG/HSR tag allows Gateway function between different Redundancy protocols. Sequence tags supported are specified in CAPR1[SQ_TAGS]. All other settings reserved.

53.4.6.7.66 Frame modification index table capability register (FMITCAPR)

Offset

Register	Offset
FMITCAPR	1888h

Function

This is the frame modification index table capability register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	FMITCAPR
ENETC1_COMMON	—	FMITCAPR
SW0_COMMON	FMITCAPR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								ACCESS_METH				Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				NUM_ENTRIES											
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• x1xx: Ternary Match Key Element Match • 1xxx: Search
19-13 —	Reserved
12-0 NUM_ENTRIES	The number of entries assigned to this table. **NOTE** The reset value is taken from FMITMAR[NUM_WORDS].

53.4.6.7.67 Frame modification index table memory allocation register (FMITMAR)

Offset

Register	Offset
FMITMAR	188Ch

Function

This is the frame modification index table memory allocation register.

NOTE

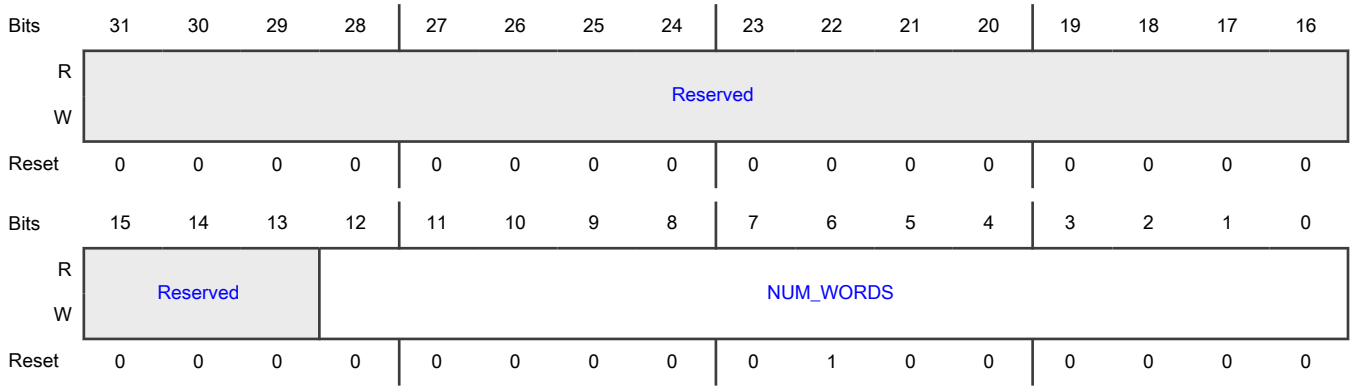
The switch will sample the value from IERB register S*a*FMITMAR when PCIe Memory Access bit is set (b1). All updates after this must be done through switch register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	FMITMAR
ENETC1_COMMON	—	FMITMAR
SW0_COMMON	FMITMAR	—

Diagram



Fields

Field	Function
31-13 —	Reserved
12-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** Each entry occupies 1 word.

53.4.6.7.68 Frame modification index table operational register (FMITOR)

Offset

Register	Offset
FMITOR	1890h

Function

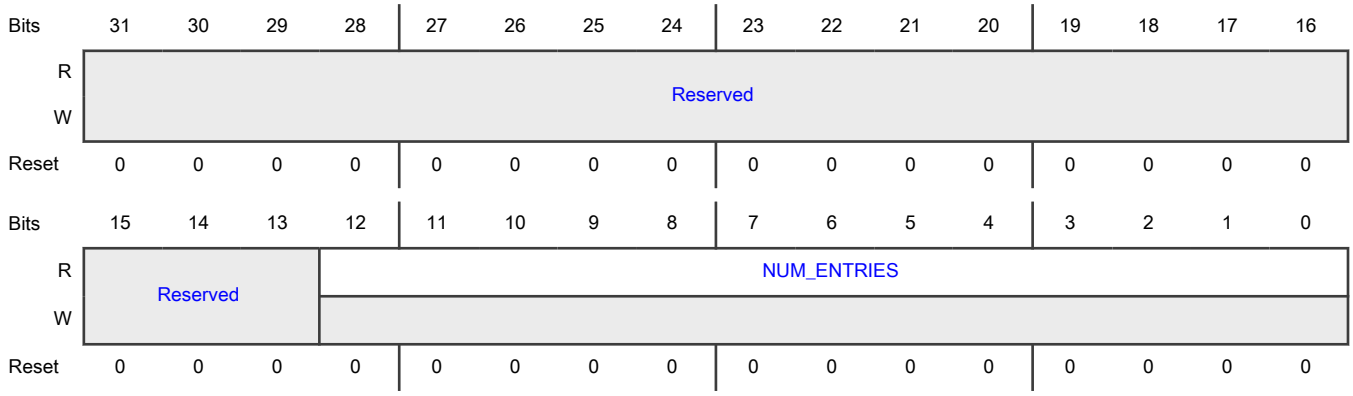
This is the frame modification index table memory operational register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	FMITOR
ENETC1_COMMON	—	FMITOR
SW0_COMMON	FMITOR	—

Diagram



Fields

Field	Function
31-13 —	Reserved
12-0 NUM_ENTRIES	The number of entries in-use by this table.

53.4.6.7.69 Frame modification data index table capability register (FMDITCAPR)

Offset

Register	Offset
FMDITCAPR	1894h

Function

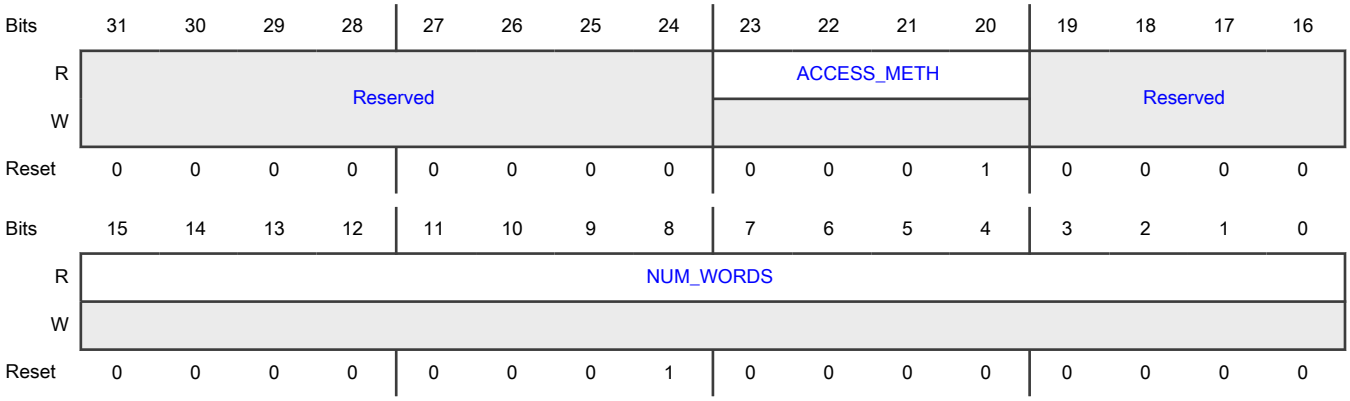
This is the frame modification data index table capability register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	FMDITCAPR
ENETC1_COMMON	—	FMDITCAPR
SW0_COMMON	FMDITCAPR	—

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** The reset value is taken from FMDITMAR[NUM_WORDS]. Each entry utilizes of a variable number words depending on the data size requested.

53.4.6.7.70 Frame modification data index table memory allocation register (FMDITMAR)

Offset

Register	Offset
FMDITMAR	1898h

Function

This is the frame modification data index table memory allocation register.

NOTE

The switch will sample the value from IERB register S*a*FMITMAR when PCIe Memory Access bit is set (b1). All updates after this must be done through switch register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	FMDITMAR
ENETC1_COMMON	—	FMDITMAR
SW0_COMMON	FMDITMAR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NUM_WORDS															
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words from index table memory assigned to this table. **NOTE** Each entry utilizes of a variable number words depending on the data size requested by frame modification entry.

53.4.6.7.71 Egress treatment capability register (ETCAPR)

Offset

Register	Offset
ETCAPR	18C0h

Function

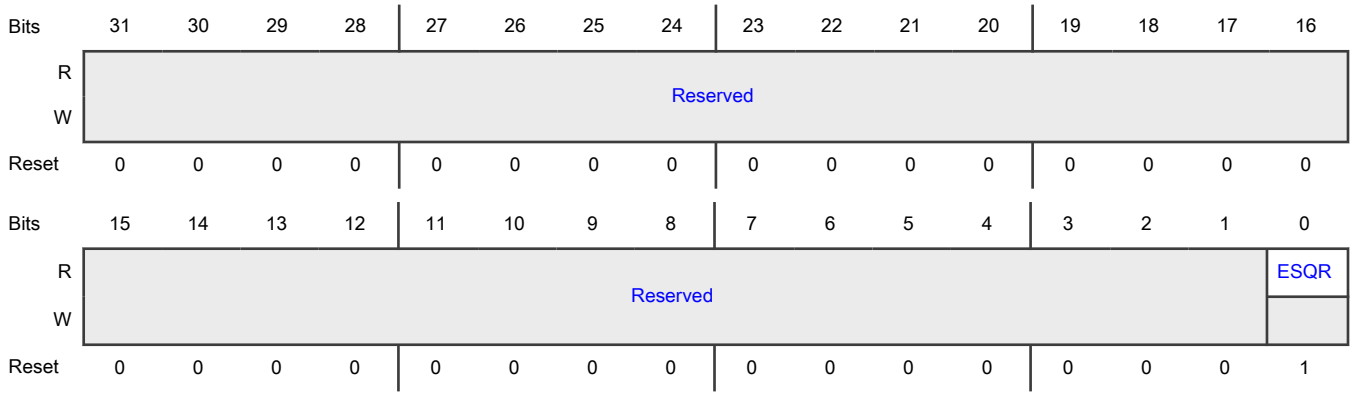
This is the egress treatment capability register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	ETCAPR
ENETC1_COMMON	—	ETCAPR
SW0_COMMON	ETCAPR	—

Diagram



Fields

Field	Function
31-1 —	Reserved
0 ESQR	Egress Sequence Recovery supported 0: Not supported 1: Supported **NOTE** Refer to SCAPR1[SQ_TAGS] to determine which sequence tags are supported.

53.4.6.7.72 Egress treatment table capability register (ETTCAPR)

Offset

Register	Offset
ETTCAPR	18C4h

Function

This is the Egress Treatment table capability register.

NOTE

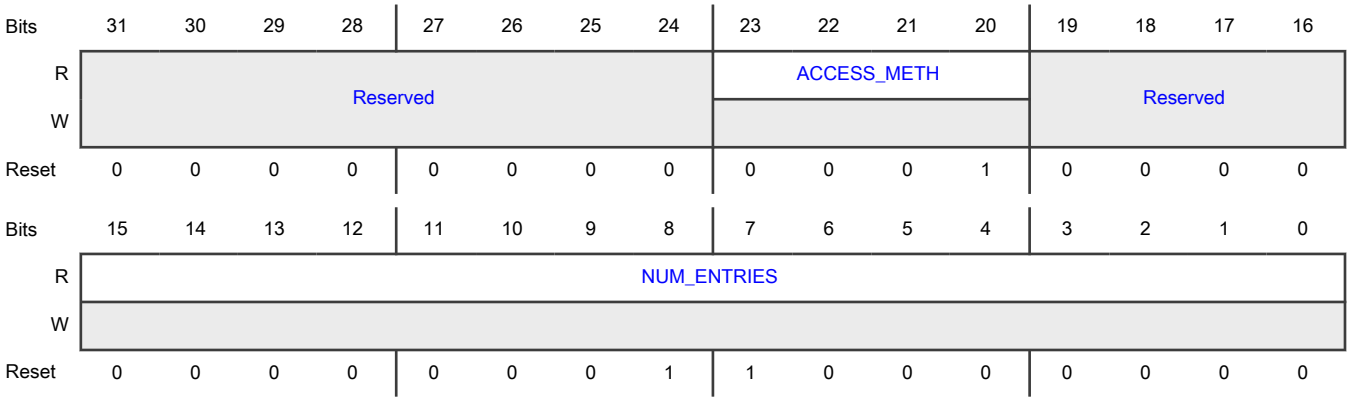
The egress treatment table is implemented using a dedicated memory as opposed to common memory for performance reasons.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	ETTCAPR
ENETC1_COMMON	—	ETTCAPR
SW0_COMMON	ETTCAPR	—

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20	Indicates which configuration access methods are supported:

Table continues on the next page...

Table continued from the previous page...

Field	Function
ACCESS_METH	• xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-16 —	Reserved
15-0 NUM_ENTRIES	The number of entries assigned to this table.

53.4.6.7.73 Egress treatment table operational register (ETTOR)

Offset

Register	Offset
ETTOR	18CCh

Function

This is the egress treatment table operational register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	ETTOR
ENETC1_COMMON	—	ETTOR
SW0_COMMON	ETTOR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NUM_ENTRIES															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 NUM_ENTRIES	Number of entries allocated / in-use by this table.

53.4.6.7.74 Time gate scheduling table capability register (TGSTCAPR)

Offset

Register	Offset
TGSTCAPR	18D4h

Function

This is the time gate scheduling table capability register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved						MAX_GCL_LEN		ACCESS_METH				Reserved			
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NUM_WORDS															
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
TGSTCAPR	ENETC0_COMMON,ENETC1_COMMON: 0220_0100h SW0_COMMON: 0220_0500h

Fields

Field	Function
31-26 —	Reserved
25-24 MAX_GCL_LEN	Maximum Gate Control List Length Indicates the maximum number of entries that can be specified in a gate control list even thought the total number of words available for storing the entries in the Time Gate Scheduling table (TGSTCAPR[NUM_WORDS]) wouldn't be exceeded. 00b - 64 01b - 128 10b - 256 11b - Reserved
23-20 ACCESS_METH	Access Method Indicates which configuration access methods are supported 1xxxb - Reserved x1xxb - Search xx1xb - EntryId xxx1b - Index
19-16 —	Reserved
15-0 NUM_WORDS	Number of Words Indicates the number of words available to this Time Gate Scheduling table. This corresponds to the total number of words available in the Time Gate Scheduling table, for storing the administrative gate control list and the operational gate control list of each table entry. This value is writable in IERB register S*a*TGSTAR/E*a*TGSTAR (by pre-boot initialization), and is immediately reflected here. **NOTE** Entry size is variable depending on the number of gates allocated for a given entry. Entry size in words is: NUM_GATES * 1.

53.4.6.7.75 Time gate scheduling table memory operation register (TGSTMOR)

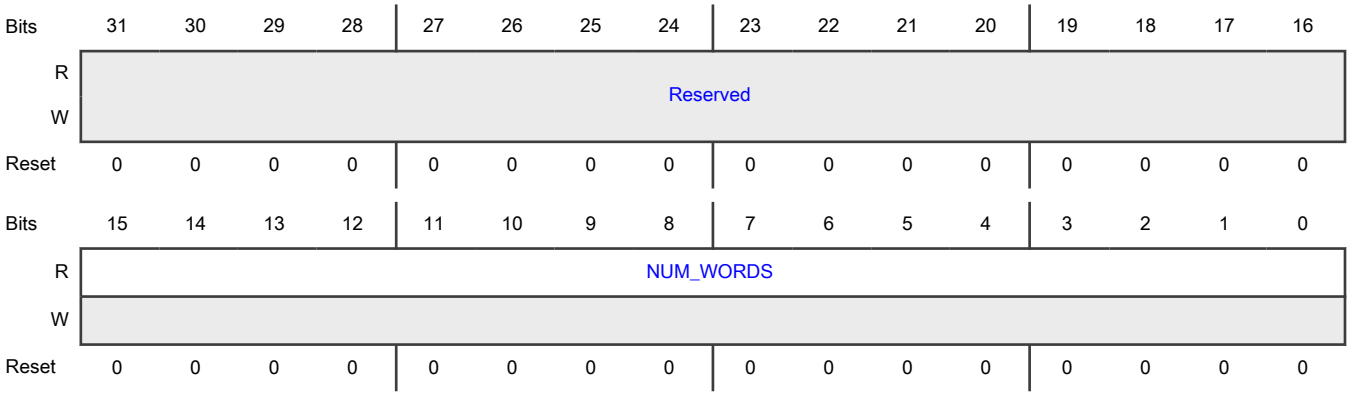
Offset

Register	Offset
TGSTMOR	18DCh

Function

This is the time gate scheduling table memory operation register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_WORDS	The number of words in-use.

53.4.6.7.76 Egress sequence recovery capability register (ESQRCAPR)

Offset

Register	Offset
ESQRCAPR	18E0h

Function

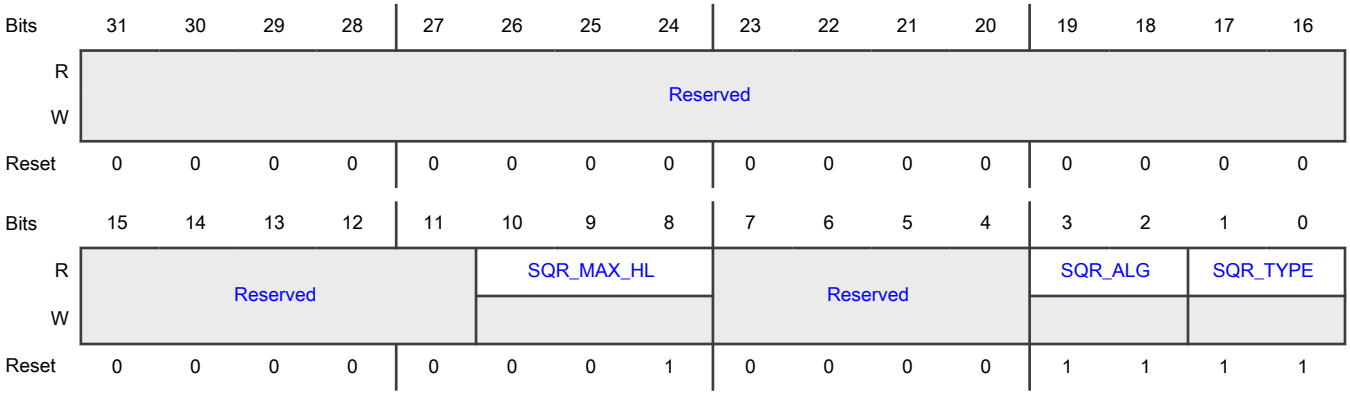
This is the egress sequence recovery capability register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	ESQRCAPR
ENETC1_COMMON	—	ESQRCAPR
SW0_COMMON	ESQRCAPR	—

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 SQR_MAX_HL	Maximum history length capability used by sequence recovery function. 0: 64 1: 128 2-7: Reserved
7-4 —	Reserved
3-2 SQR_ALG	Sequence Recovery Supported Algorithm. x1: Match Sequence Algorithm 1x: Vector Sequence Algorithm
1-0 SQR_TYPE	Support Individual Recovery function and/or Sequence Recovery function x1: Individual Recovery function supported 1x: Sequence Recovery function supported

53.4.6.7.77 Egress sequence recovery table capability register (ESQRTCAPR)

Offset

Register	Offset
ESQRTCAPR	18E4h

Function

This is the Egress Sequence Recovery table capability register.

NOTE

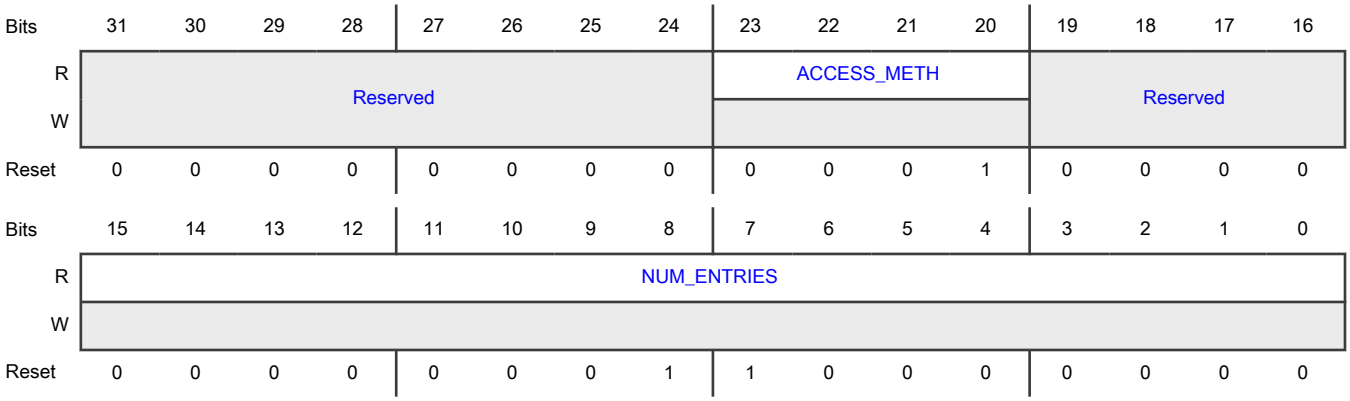
The table is implemented using a dedicated memory as opposed to common memory for performance reasons.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	ESQRTCAPR
ENETC1_COMMON	—	ESQRTCAPR
SW0_COMMON	ESQRTCAPR	—

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20	Indicates which configuration access methods are supported:

Table continues on the next page...

Table continued from the previous page...

Field	Function
ACCESS_METH	• xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-16 —	Reserved
15-0 NUM_ENTRIES	The number of entries assigned to this table

53.4.6.7.78 Egress counter table capability register (ECTCAPR)

Offset

Register	Offset
ECTCAPR	18ECh

Function

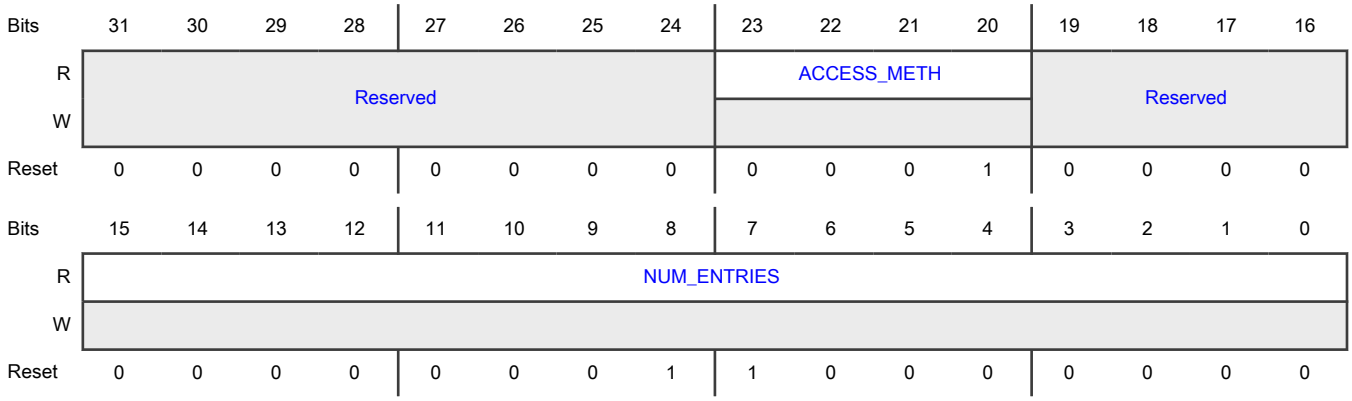
This is the Egress Counter table capability register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	—	ECTCAPR
ENETC1_COMMON	—	ECTCAPR
SW0_COMMON	ECTCAPR	—

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-16 —	Reserved
15-0 NUM_ENTRIES	The number of entries assigned to this table.

53.4.6.7.79 Hash table memory capability register (HTMCAPR)

Offset

Register	Offset
HTMCAPR	1900h

Function

This is the hash table memory capability register. Maximum number of words allotted to exact match hash tables from the common memory's shared region.

Software driver is responsible to manage how the hash table memory is distributed amongst various hash tables. Following hash table's entries are 1 word in size:

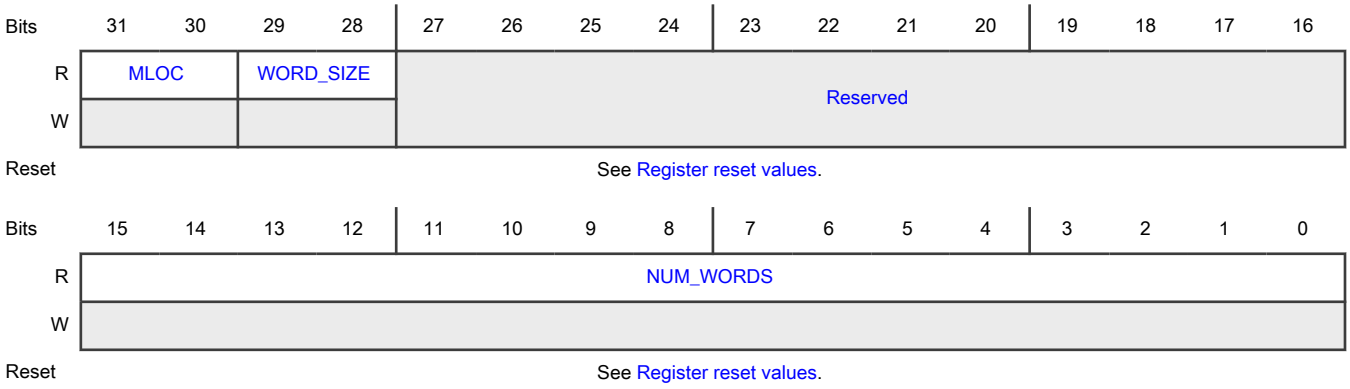
- Ingress Stream Identification table

- Ingress Stream Filter table
- VLAN Filter table
- FDB table
- L2 IPv4 Multicast Filter table

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
HTMCAPR	ENETC0_COMMON: 0000_0008h ENETC1_COMMON: 0000_0010h SW0_COMMON: 0000_08C1h

Fields

Field	Function
31-30 MLOC	Indicates memory location of this table 0: Common memory 1: Reserved
29-28 WORD_SIZE	Word size in bytes. 0: 24 bytes 1-3: Reserved
27-16 —	Reserved
15-0	Maximum number of words allotted to exact match hash table from the common memory's shared region.

Table continues on the next page...

Table continued from the previous page...

Field	Function
NUM_WORDS	**NOTE** Reset value from IERB's S0HTMAR/E*a*HTMAR.

53.4.6.7.80 Hash table memory operational register (HTMOR)

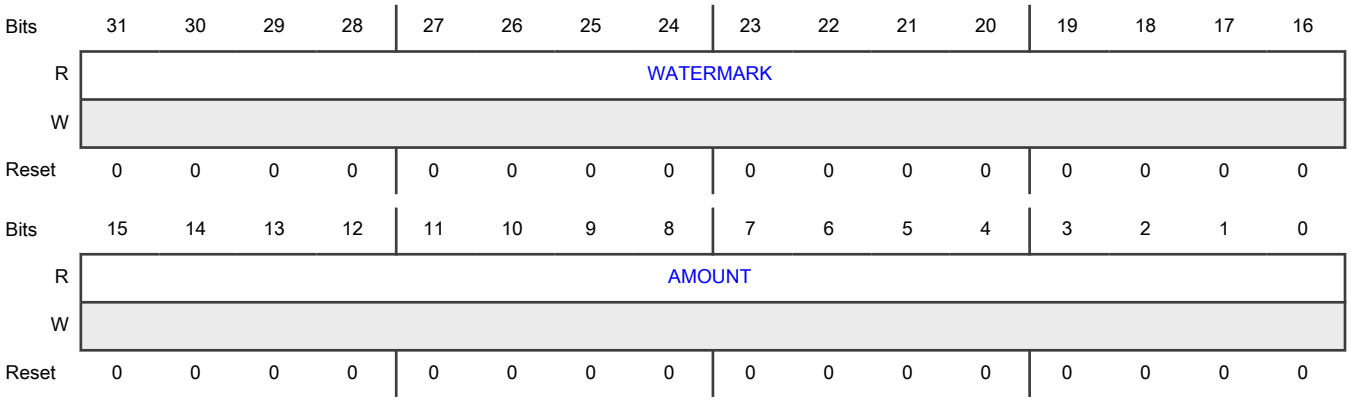
Offset

Register	Offset
HTMOR	1904h

Function

This is the hash table memory operational register.

Diagram



Fields

Field	Function
31-16 WATERMARK	High WaterMark of Words allocated. **NOTE** Value reset to AMOUNT when read.
15-0 AMOUNT	Number of Words in use by this function which has been allocated by the various hash tables.

53.4.6.7.81 Ingress stream identification capability register (ISIDCAPR)

Offset

Register	Offset
ISIDCAPR	1910h

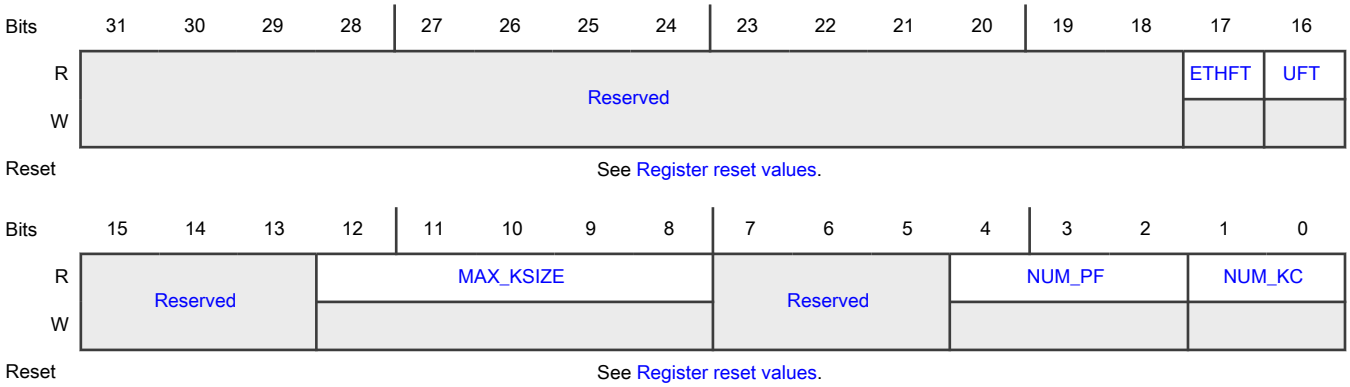
Function

This is the ingress stream identification capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Register reset values

Register	Reset value
ISIDCAPR	ENETC0_COMMON,ENETC1_COMMON: 0003_1011h SW0_COMMON: 0003_1013h

Fields

Field	Function
31-18 —	Reserved
17 ETHFT	Ethernet Frame Type (frame begins with standard 802.1 Ethernet header) supported. 0: Not supported 1: Supported That is, frame begins with a standard 802.1 Ethernet Header.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The set of optional fields and its size in the key are: • MAC DA (6B) • MAC SA (6B) • Outer VLAN (2B) • Inner VLAN (2B) • R-TAG/HSR (1B) **NOTE** For Outer and Inner VLAN, can optionally select the VID and/or PCP field; DE field is not part of the KEY.
16 UFT	Unknown Frame Type (no header field parsing of the frame is necessary to construct the key) supported. 0: Not supported 1: Supported That is, no field parsing of the frame is necessary to construct the key. The number of Payload fields supported is specified by NUM_PF.
15-13 —	Reserved
12-8 MAX_KSIZE	Maximum Key Size in bytes which can be constructed using the frame's fields.
7-5 —	Reserved
4-2 NUM_PF	Number of configurable Payload Fields supported. **NOTE** Determines the range of b={0..NUM_PF} in ISIDKC*a*CR0.
1-0 NUM_KC	Number of Exact Match Key Construction Instances supported for Ingress Stream Identification. Each Exact Match Key Construction instance can extract fields within the first 144B of the frame. Formula: NUM_KC+1 **NOTE** This determines *a* in ISIDKC*a*CR0. Valid if IPCAPR[ISID]=1.

53.4.6.7.82 Ingress stream identification hash table capability register (ISIDHTCAPR)

Offset

Register	Offset
ISIDHTCAPR	1914h

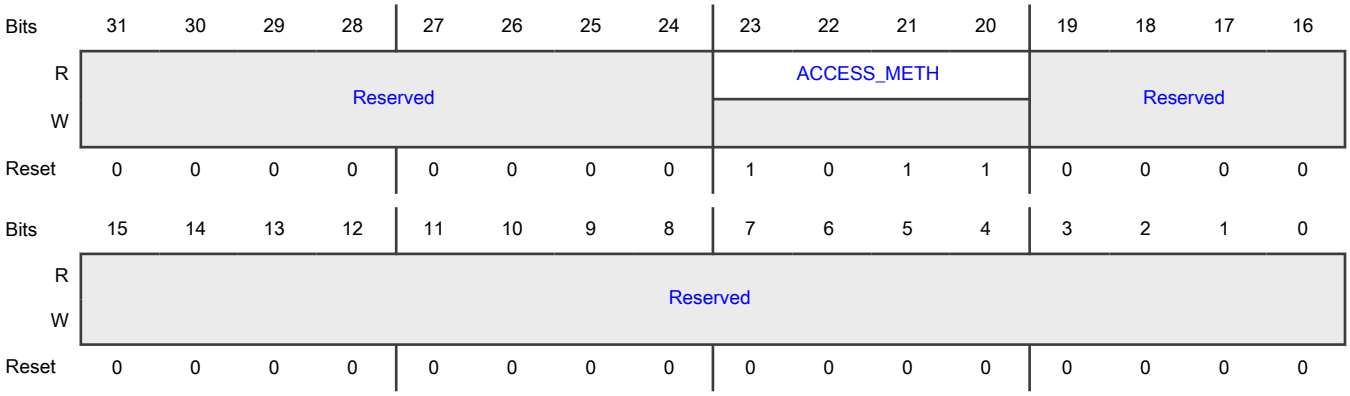
Function

This is the ingress stream identification hash table capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: • xxx1: EntryId Match • xx1x: Exact Match Key Element Match • x1xx: Ternary Match Key Element Match • 1xxx: Search
19-0 —	Reserved

53.4.6.7.83 Ingress stream identification key construction a operational register (ISIDKC0OR - ISIDKC3OR)

Offset

Register	Offset
ISIDKC0OR	1920h
ISIDKC1OR	1940h
ISIDKC2OR	1960h
ISIDKC3OR	1980h

Function

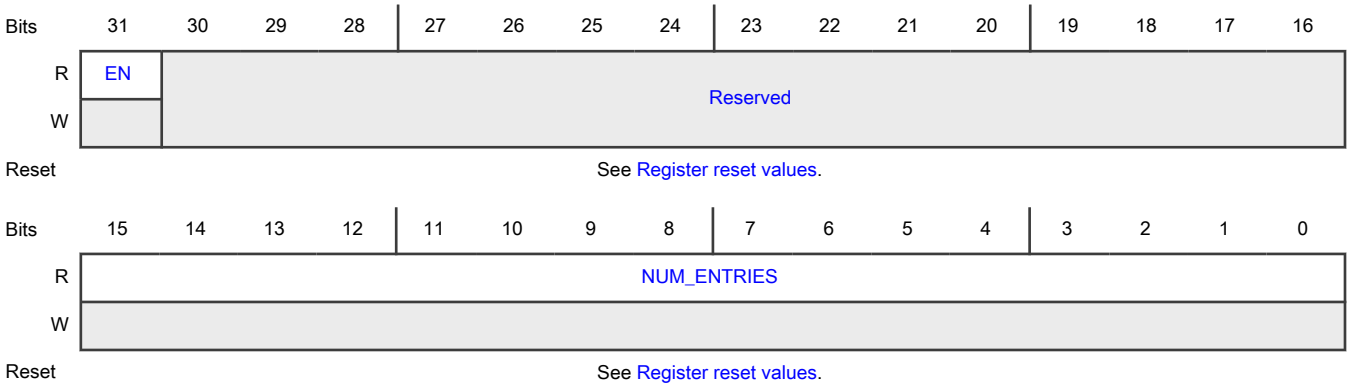
This is the ingress stream identification key construction operational register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	ISIDKC0OR–ISIDKC1OR	ISIDKC2OR–ISIDKC3OR
ENETC1_COMMON	ISIDKC0OR–ISIDKC1OR	ISIDKC2OR–ISIDKC3OR
SW0_COMMON	ISIDKC0OR–ISIDKC3OR	—

Diagram



Register reset values

Register	Reset value
ISIDKC0OR–ISIDKC1OR	ENETC0_COMMON–SW0_COMMON: 0000_0000h
ISIDKC2OR–ISIDKC3OR	0000_0000h

Fields

Field	Function
31 EN	Operational state of this key construction register: 0: Disabled: Exact Match Lookup must not utilize this key construction. 1: Enabled: Exact Match Lookup can utilize this key construction. **NOTE** Hardware sets this to 1 if Key Construction requested/specified in ISIDKCaCR0 and ISIDKCaPFbCR does not exceed ISIDCAPR[MAX_KSIZE]; otherwise it is set to 0. Hardware uses ISIDKCaCR0 as a trigger register to evaluate EN. As such any payload extraction for key construction must be configured first using ISIDKCaPFbCR before configuring ISIDKCaCR0.
30-16 —	Reserved
15-0 NUM_ENTRIES	Indicates the number of Ingress Stream Identification, i.e., table 30 entries added using this key. **NOTE** Software driver must ensure that this value is 0 in order to modify ISIDKC*a*CR0.

53.4.6.7.84 Ingress stream identification key construction a configuration register 0 (ISIDKC0CR0 - ISIDKC3CR0)

Offset

Register	Offset
ISIDKC0CR0	1924h
ISIDKC1CR0	1944h
ISIDKC2CR0	1964h
ISIDKC3CR0	1984h

Function

The function supports (ISIDCAPR[NUM_KC]+1)/2 pairs of key construction registers. For a switch function, PISIDCR[KCPAIR] specifies the key construction pair used when constructing ISID key.

NOTE

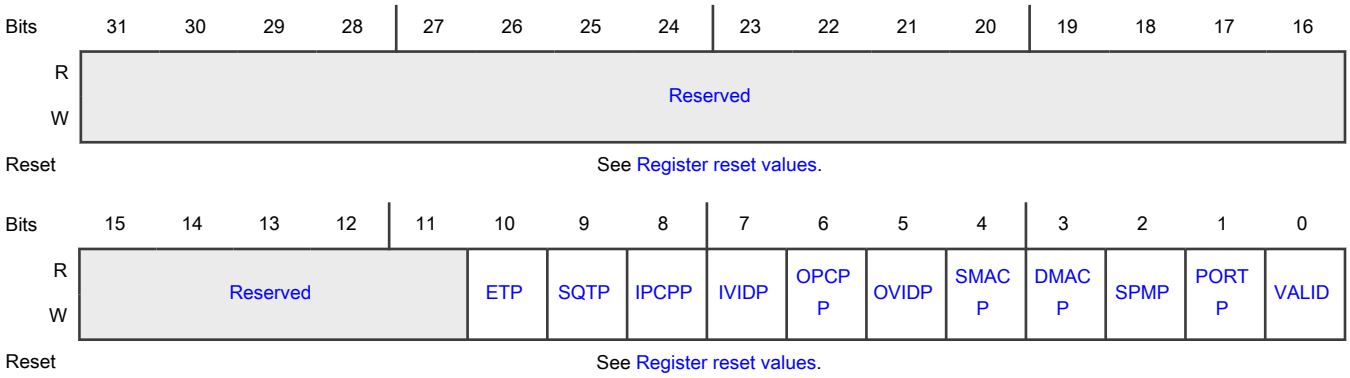
This register format is applicable when PCR[HDR_FMT]=0. Hardware uses ISIDKCaCR0 as a trigger register to evaluate ISIDKCaOR[EN]. As such any payload extraction for key construction has to be configured first using ISIDKCaPFbCR before configuring ISIDKCaCR0.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_COMMON	ISIDKC0CR0–ISIDKC1CR0	ISIDKC2CR0–ISIDKC3CR0
ENETC1_COMMON	ISIDKC0CR0–ISIDKC1CR0	ISIDKC2CR0–ISIDKC3CR0
SW0_COMMON	ISIDKC0CR0–ISIDKC3CR0	—

Diagram



Register reset values

Register	Reset value
ISIDKC0CR0–ISIDKC1CR0	ENETC0_COMMON–SW0_COMMON: 0000_0000h
ISIDKC2CR0–ISIDKC3CR0	0000_0000h

Fields

Field	Function
31-11 —	Reserved
10 ETP	EtherType Present. Specifies whether the EtherType after any VLAN or redundancy sequence (R-TAG, HSR tag) headers is present in the key. 0b - Not present 1b - Present
9 SQTP	Sequence Tag (code point) Present. Specifies whether a sequence tag code point is present in the key. The 1-byte sequence tag code points are defined as follows: 0: No R-TAG/HSR tag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1: 802.1CB draft 2.0 R-TAG 2: R-TAG 3: HSR Tag 4-255: Reserved 0b - Not present 1b - Present
8 IPCPP	Inner PCP Present. Specifies whether the inner PCP is present in the key. If IPCPP is set 1 or IVIDP is set 1, or both are set 1, a 2-byte value must be entered in the frame key field of the Ingress Stream Identification table entry with the following definition: • Bit 15 **V** - VLAN tag present in the frame (0 = not present, 1 = present). • Bit 14:12 **PCP** – PCP value to be matched if IPCPP is set to 1 (must be set to 0 if IPCPP=0). • Bit 11:0 **VID** – VID value to be matched if IVIDP is set to 1 (must be set to 0 if IVIDP=0). Most significant byte of the 2-byte 'V[15], PCP[14:12] and VID [11:0]' would be stored at the lowest byte offset. Least significant byte (that is VID[7:0]) would be stored at the next byte offset. Required settings for the following matches: • **Outer PCP only** - IPCPP set to 1 and IVIDP set to 0, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, set to '1, 0, VID value to match'. • **Outer VLAN ID and PCP** - IPCPP and IVIDP set to 1, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, set to '1, PCP value to match, VID value to match'. • **Outer VLAN not present** - IPCPP or IVIDP (or both) set to 1, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, all set to 0. The inner VLAN tag is considered present in the frame if VLAN classification has deemed the inner VLAN tag valid and the VLAN tag was not taken from the port VLAN configuration register. 0b - Not present 1b - Present
7 IVIDP	Inner VID Present. Specifies whether the inner VLAN ID is present in the key. If IVIDP is set 1 or IPCPP is set 1, or both are set 1, a 2-byte value must be entered in the frame key field of the Ingress Stream Identification table entry with the following definition: • Bit 15 **V** - VLAN tag present in the frame (0 = not present, 1 = present) • Bit 14:12 **PCP** – PCP value to be matched if IPCPP is set to 1 (must be set to 0 if IPCPP=0) • Bit 11:0 **VID** – VID value to be matched if IVIDP is set to 1 (must be set to 0 if IVIDP=0) Most significant byte of the 2-byte 'V[15], PCP[14:12] and VID [11:0]' would be stored at the lowest byte offset. Least significant byte (that is VID[7:0]) would be stored at the next byte offset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Required settings for the following matches: • **Inner VLAN ID only** - IVIDP set to 1 and IPCPP set to 0, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, set to '1, 0, VID value to match'. • **Inner VLAN ID and PCP** - IVIDP and IPCPP set to 1, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, set to '1, PCP value to match, VID value to match'. • **Inner VLAN not present**- IVIDP or IPCPP (or both) set to 1, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, all set to 0. The inner VLAN tag is considered present in the frame if VLAN classification has deemed the inner VLAN tag valid and the VLAN tag was not taken from the port VLAN configuration register. 0b - Not present 1b - Present
6 OPCPP	Outer PCP Present Specifies whether the outer PCP is present in the key. If OPCPP is set 1 or OVIDP is set 1, or both are set 1, a 2-byte value must be entered in the frame key field of the Ingress Stream Identification table entry with the following definition: • Bit 15 **V** - VLAN tag present in the frame (0 = not present, 1 = present) • Bit 14:12 **PCP** – PCP value to be matched if OPCPP is set to 1 (must be set to 0 if OPCPP=0) • Bit 11:0 **VID** – VID value to be matched if OVIDP is set to 1 (must be set to 0 if OVIDP=0) Most significant byte of the 2-byte 'V[15], PCP[14:12] and VID [11:0]' would be stored at the lowest byte offset. Least significant byte (i.e. VID[7:0]) would be stored at the next byte offset. Required settings for the following matches: • Outer PCP only - OPCPP set to 1 and OVIDP set to 0, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, set to '1, 0, VID value to match'. • Outer VLAN ID and PCP - OPCPP and OVIDP set to 1, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, set to '1, PCP value to match, VID value to match'. • Outer VLAN not present - OPCPP or OVIDP (or both) set to 1, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, all set to 0. The outer VLAN tag is considered present in the frame if VLAN classification has deemed the outer VLAN tag valid and the VLAN tag was not taken from the port VLAN configuration register. 0b - Outer VLAN header's PCP field not present in the key 1b - Outer VLAN header's PCP field present in the key
5 OVIDP	Outer VID Present Specifies whether the outer VLAN ID is present in the key. If OVIDP is set 1 or OPCPP is set 1, or both are set 1, a 2-byte value must be entered in the frame key field of the Ingress Stream Identification table entry with the following definition:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Bit 15 **V** - VLAN tag present in the frame (0 = not present, 1 = present) • Bit 14:12 **PCP** – PCP value to be matched if OPCPP is set to 1 (must be set to 0 if OPCPP=0) • Bit 11:0 **VID** – VID value to be matched if OVIDP is set to 1 (must be set to 0 if OVIDP=0) Most significant byte of the 2-byte 'V[15], PCP[14:12] and VID [11:0]' would be stored at the lowest byte offset. Least significant byte (i.e. VID[7:0]) would be stored at the next byte offset. Required settings for the following matches: • **Outer VLAN ID only** - OVIDP set to 1 and OPCPP set to 0, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, set to '1, 0, VID value to match'. • **Outer VLAN ID and PCP** - OVIDP and OPCPP set to 1, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, set to '1, PCP value to match, VID value to match'. • **Outer VLAN not present** - OVIDP or OPCPP (or both) set to 1, and 'V, PCP and VID' fields of the frame key entered in the Ingress Stream Identification entry, all set to 0. The outer VLAN tag is considered present in the frame if VLAN classification has deemed the outer VLAN tag valid and the VLAN tag was not taken from the port VLAN configuration register. 0b - Not present 1b - Present
4 SMACP	Source MAC (address) Present. Specifies whether the source MAC address is present in the key. 0b - Not present 1b - Present
3 DMACP	Destination MAC (address) Present Specifies whether the destination MAC address is present in the key. 0b - Not present 1b - Present
2 SPMP	Switch Port Masquerading (flag) Present Specifies whether the SPM flag is present in the key. SPM flag is defined as follows: 0: Frame originated from a switch port 1: Frame originated from management port (host) masquerading as switch port 0b - Not present 1b - Present
1 PORTP	Source Port Present Specifies whether the source port is present in the key. 0b - Not present 1b - Present

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 VALID	Valid Specifies whether, the key construction is valid. **NOTE** The precedence value of these entries are greater than ingress port filter table's entry IPF_CE[PR] value of 2. 0b - The entire key construction rule is not valid including any configuration payload key fields defined. 1b - The key construction rule is valid.

53.4.6.7.85 Ingress stream identification key construction a payload field b configuration register (ISIDKC0PF0CR - ISIDKC3PF3CR)

Offset

For a = 0 to 3; b = 0 to 3:

Register	Offset
ISIDKCaPFbCR	1930h + (a × 20h) + (b × 4h)

Function

Defines one of the 4 Ingress Stream Identification payload fields. Register is valid if its PFP field is set to 1.

If one payload field is to be specified, it has to be specified in ISIDKCaPF0CR with PFP set to 1 with payload size and offset specified and remaining payload registers have their PFP field set to 0.

If two payload fields are to be specified then the PFP of ISIDKCaPF0CR, ISIDKCaPF1CR must be set to 1 with payload size and offset specified and remaining payload field registers have their PFP field set to 0. Similar steps must be followed for specifying additional payload fields. (Note that if ISIDKCaPF0CR[PFP] =1,ISIDKCaPF1CR[PFP] =1,ISIDKCaPF2CR[PFP] =0 and if ISIDKCaPF3CR[PFP] =1, then payload size and offset specified in ISIDKCaPF0CR and ISIDKCaPF1CR will be used for key construction and ISIDKCaPF2CR, ISIDKCaPF3CR will not be considered for key construction).

NOTE

Hardware uses ISIDKCaCR0 as a trigger register to evaluate ISIDKCaOR[EN]. As such any payload extraction for key construction has to be configured first using ISIDKCaPFbCR before configuring ISIDKCaCR0.

NOTE

Each module instance supports a different number of registers.

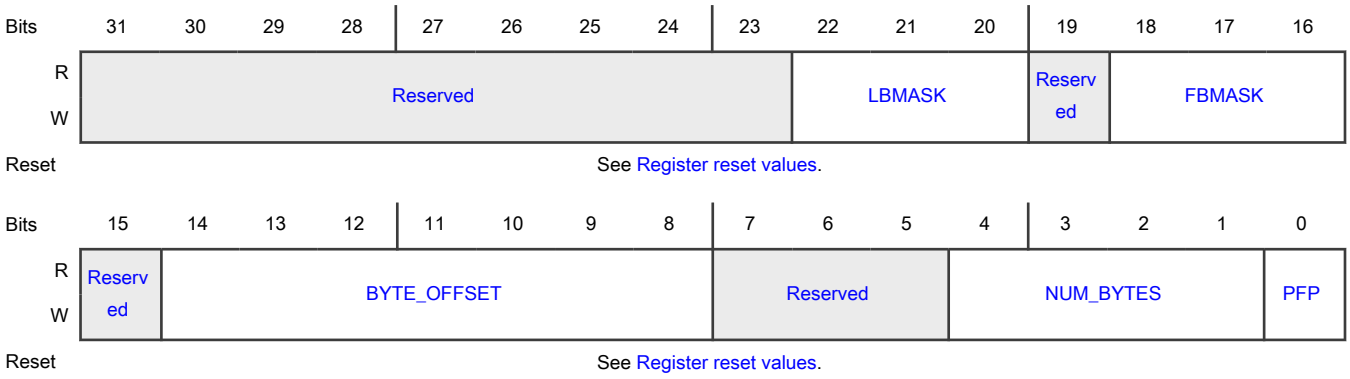
Instance	Register supported	Register not supported
ENETC0_COMMON	ISIDKC0PF0CR–ISIDKC1PF3CR	ISIDKC2PF0CR–ISIDKC3PF3CR
ENETC1_COMMON	ISIDKC0PF0CR–ISIDKC1PF3CR	ISIDKC2PF0CR–ISIDKC3PF3CR

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_COMMON	ISIDKC0PF0CR–ISIDKC3PF3CR	—

Diagram



Register reset values

Register	Reset value
ISIDKC0PF0CR–ISIDKC1PF3CR	ENETC0_COMMON–SW0_COMMON: 0000_0000h
ISIDKC2PF0CR–ISIDKC3PF3CR	0000_0000h

Fields

Field	Function
31-23 —	Reserved
22-20 LBMASK	Payload Last Byte Mask: Number of least significant bits from the last payload key byte to mask.
19 —	Reserved
18-16 FBMASK	Payload First Byte Mask: Number of most significant bits of first payload key byte to mask.
15 —	Reserved
14-8	Payload Byte Offset where field extraction begins.

Table continues on the next page...

Table continued from the previous page...

Field	Function
BYTE_OFFSET	Payload offset byte 0 for the first payload field (ISIDKCaPF0CR) starts immediately after the payload Ethertype. Payload Ethertype starts immediately after source MAC address or VLAN tag(s)/RTAG/HSR (if any). Payload byte offset for subsequent payload fields (ISIDKCaPF1CR .. ISIDKCaPFbCR) starts immediately after the last byte of the previous payload field. Hardware will not be construct the Ingress Stream Identification table lookup key if the payload key field is not within the frame or if it is not within 116 bytes of the payload; where payload starts after the payload Ethertype which in turn is after source MAC address or VLAN, tag(s)/R-TAG/HSR (if any). Since the key cannot be constructed, the Ingres Stream Identification table lookup using this key construction rule will be by-passed.
7-5 —	Reserved
4-1 NUM_BYTES	The NUM_BYTES field is used to specify the size of the payload key field, as follows: Size of the payload key field = NUM_BYTES + 1; range is 1 to 16 bytes
0 PFP	Payload field Present 0: This payload field register is not used for constructing a key. 1: NUM_BYTES+1 consecutive bytes starting at BYTE_OFFSET are included in the Key.

53.4.6.7.86 Ingress stream filter hash table capability register (ISFHTCAPR)

Offset

Register	Offset
ISFHTCAPR	1A00h

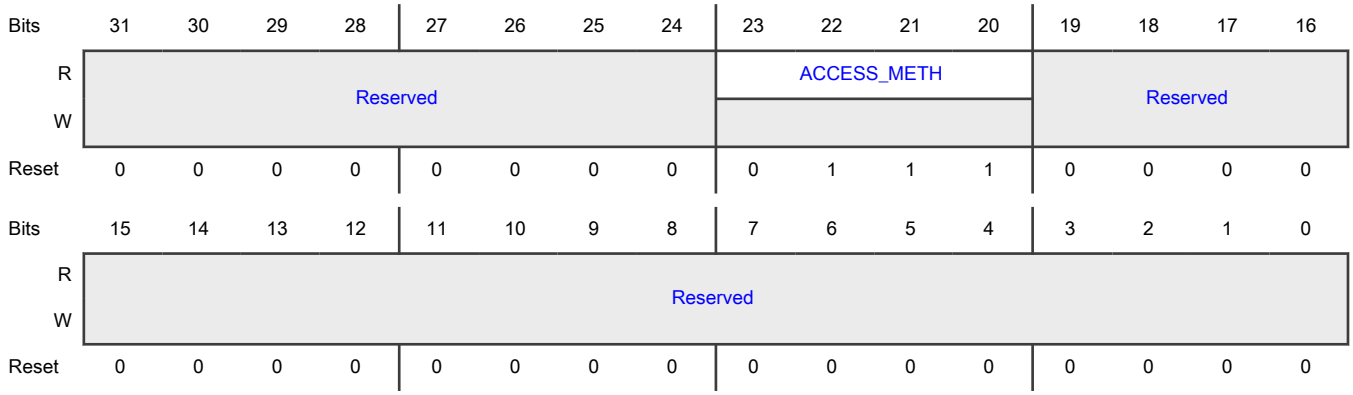
Function

This is the ingress stream filter hash table capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 ACCESS_METH	Indicates which configuration access methods are supported: xxx1: Index xx1x: EntryId x1xx: Search 1xxx: Reserved
19-0 —	Reserved

53.4.6.7.87 Ingress stream filter hash table operational register (ISFHTOR)

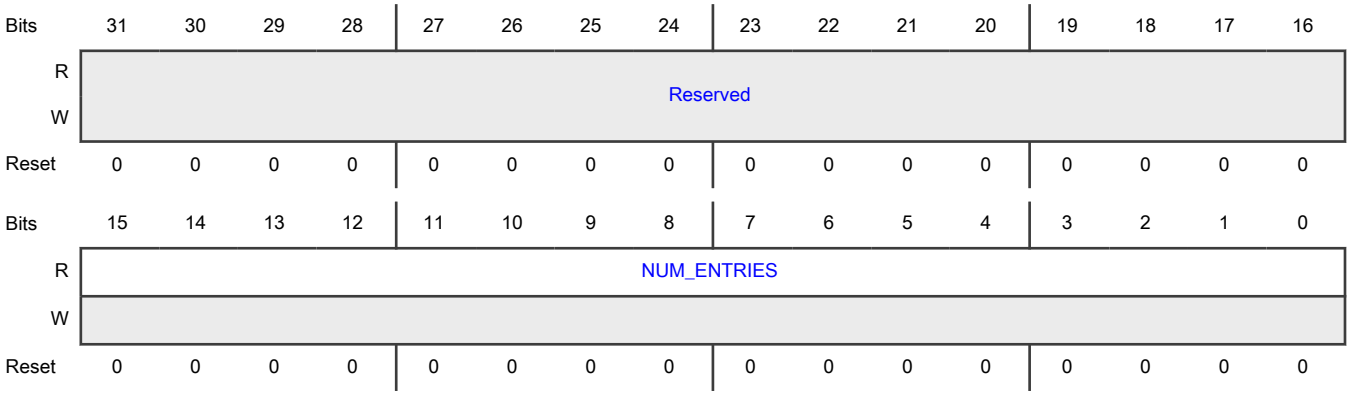
Offset

Register	Offset
ISFHTOR	1A04h

Function

This is the ingress stream filter hash table operational register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_ENTRIES	Number of entries in-use by this table.

53.4.6.8 Port register descriptions

This section describes the port registers. The register view differs depending on if the port applies to switch or ENETC instance.

53.4.6.8.1 Port memory map

Instance	Address space	PCIe EA BEI 0 offset
ENETC0_PORT	PCI_F3_BAR_0	60B1_4000h
ENETC1_PORT	PCI_F4_BAR_0	60B5_4000h
SW0_PORT0	PCI_F2_BAR_0	60A0_4000h
SW0_PORT1	PCI_F2_BAR_0	60A0_8000h
SW0_PORT2	PCI_F2_BAR_0	60A0_C000h
SW0_PORT3	PCI_F2_BAR_0	60A1_0000h
SW0_PORT4	PCI_F2_BAR_0	60A1_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Port capability register (PCAPR)	32	R	See section
4h	Port MAC capability register (PMCAPR)	32	R	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8h	Port I/O capability register (PIOCAPR)	32	R	See section
10h	Port configuration register (PCR)	32	RW	0063_0000h
20h	Port MAC address register 0 (PMAR0)	32	RW	0000_0000h
24h	Port MAC address register 1 (PMAR1)	32	RW	0000_0000h
50h	Port TPID acceptance register (PTAR)	32	RW	0000_0033h
54h	Port QoS mode register (PQOSMR)	32	RW	0000_0000h
60h	Port Queue Operational register (PQOR)	32	R	0000_0000h
80h	Port parser configuration register (PPCR)	32	RW	3131_3000h
84h	Port ingress port filter configuration register (PIPFCR)	32	RW	0000_0000h
A0h	Port stream gate configuration register (PSGCR)	32	RW	0000_0000h
100h	Port operational register (POR)	32	RW	0000_0000h
104h	Port status register (PSR)	32	R	0000_0000h
108h	Port receive SDU overhead register (PRXSDUOR)	32	RW	0000_0014h
10Ch	Port transmit SDU overhead register (PTXSDUOR)	32	RW	0000_0014h
110h	Port time gate scheduling control register (PTGSCR)	32	RW	See section
114h	Port time gate scheduling admin gate list status register (PTGAGLSR)	32	R	0000_0000h
118h	Port time gate scheduling admin gate list length register (PTGAGLLR)	32	R	0000_0000h
11Ch	Port time gating operational gate list length register (PTGOGLLR)	32	R	0000_0000h
120h	Port time gate scheduling advance time offset register (PTGSATOR)	32	RW	0000_0000h
124h	Port time gate scheduling hold advance register (PTGSHAR)	32	R	0000_0000h
128h	Port time gate scheduling release advance register (PTGSRAR)	32	R	0000_0000h
12Ch	Port time gate scheduling hold configuration register (PTGSHCR)	32	RW	0100_0000h
134h	Port frame preemption configuration register (PFPCR)	32	RW	0000_0000h
138h	Port default gate state register (PDGSR)	32	RW	0000_00FFh
1C0h	Port Rx discard count register (PRXDCR)	32	R	0000_0000h
1C8h	Port Rx discard count reason register 0 (PRXDCRR0)	32	RW	0000_0000h
1CCh	Port Rx discard count reason register 1 (PRXDCRR1)	32	RW	0000_0000h
1E0h	Port Tx discard count register (PTXDCR)	32	R	0000_0000h
1E8h	Port Tx discard count reason register 0 (PTXDCRR0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1ECh	Port Tx discard count reason register 1 (PTXDCRR1)	32	RW	0000_0000h
200h	Port time gate scheduling traffic class 0 status register (PTGSTC0SR)	32	R	0001_0000h
208h	Port traffic class 0 transmit maximum SDU register (PTC0TMSDUR)	32	RW	0001_0600h
210h	Port transmit traffic class 0 credit based shaper register 0 (PTC0CBSR0)	32	RW	0000_0000h
214h	Port traffic class 0 credit based shaper register 1 (PTC0CBSR1)	32	RW	0000_0000h
220h	Port time gate scheduling traffic class 1 status register (PTGSTC1SR)	32	R	0001_0000h
228h	Port traffic class 1 transmit maximum SDU register (PTC1TMSDUR)	32	RW	0001_0600h
230h	Port transmit traffic class 1 credit based shaper register 0 (PTC1CBSR0)	32	RW	0000_0000h
234h	Port traffic class 1 credit based shaper register 1 (PTC1CBSR1)	32	RW	0000_0000h
240h	Port time gate scheduling traffic class 2 status register (PTGSTC2SR)	32	R	0001_0000h
248h	Port traffic class 2 transmit maximum SDU register (PTC2TMSDUR)	32	RW	0001_0600h
250h	Port transmit traffic class 2 credit based shaper register 0 (PTC2CBSR0)	32	RW	0000_0000h
254h	Port traffic class 2 credit based shaper register 1 (PTC2CBSR1)	32	RW	0000_0000h
260h	Port time gate scheduling traffic class 3 status register (PTGSTC3SR)	32	R	0001_0000h
268h	Port traffic class 3 transmit maximum SDU register (PTC3TMSDUR)	32	RW	0001_0600h
270h	Port transmit traffic class 3 credit based shaper register 0 (PTC3CBSR0)	32	RW	0000_0000h
274h	Port traffic class 3 credit based shaper register 1 (PTC3CBSR1)	32	RW	0000_0000h
280h	Port time gate scheduling traffic class 4 status register (PTGSTC4SR)	32	R	0001_0000h
288h	Port traffic class 4 transmit maximum SDU register (PTC4TMSDUR)	32	RW	0001_0600h
290h	Port transmit traffic class 4 credit based shaper register 0 (PTC4CBSR0)	32	RW	0000_0000h
294h	Port traffic class 4 credit based shaper register 1 (PTC4CBSR1)	32	RW	0000_0000h
2A0h	Port time gate scheduling traffic class 5 status register (PTGSTC5SR)	32	R	0001_0000h
2A8h	Port traffic class 5 transmit maximum SDU register (PTC5TMSDUR)	32	RW	0001_0600h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2B0h	Port transmit traffic class 5 credit based shaper register 0 (PTC5CBSR0)	32	RW	0000_0000h
2B4h	Port traffic class 5 credit based shaper register 1 (PTC5CBSR1)	32	RW	0000_0000h
2C0h	Port time gate scheduling traffic class 6 status register (PTGSTC6SR)	32	R	0001_0000h
2C8h	Port traffic class 6 transmit maximum SDU register (PTC6TMSDUR)	32	RW	0001_0600h
2D0h	Port transmit traffic class 6 credit based shaper register 0 (PTC6CBSR0)	32	RW	0000_0000h
2D4h	Port traffic class 6 credit based shaper register 1 (PTC6CBSR1)	32	RW	0000_0000h
2E0h	Port time gate scheduling traffic class 7 status register (PTGSTC7SR)	32	R	0001_0000h
2E8h	Port traffic class 7 transmit maximum SDU register (PTC7TMSDUR)	32	RW	0001_0600h
2F0h	Port transmit traffic class 7 credit based shaper register 0 (PTC7CBSR0)	32	RW	0000_0000h
2F4h	Port traffic class 7 credit based shaper register 1 (PTC7CBSR1)	32	RW	0000_0000h
400h	Port buffer pool mapping configuration register 0 (PBPMCR0)	32	RW	0000_0000h
404h	Port buffer pool mapping configuration register 1 (PBPMCR1)	32	RW	0000_0000h
438h	Port PCP DEI mapping register (PPCPDEIMR)	32	RW	0000_0000h
440h	Port mirror configuration register (PMCR)	32	RW	0000_0000h
450h	Port cut through forwarding configuration register (PCTFCR)	32	RW	0000_0013h
458h	Port LANID configuration register (PLANIDCR)	32	RW	0000_0000h
460h	Port ingress stream identification configuration register (PISIDCR)	32	RW	FFFF_0000h
464h	Port frame modification configuration register (PFMCR)	32	RW	0000_0000h
470h	Port IPV to queue mapping register 0 (PIPV2QMR0)	32	RW	7654_3210h
4B0h	Port time capture minimum latency register (PTCMINLR)	32	R	0000_0000h
4B4h	Port time capture maximum latency register (PTCMAXLR)	32	R	0000_0000h
500h	Bridge port configuration register (BPCR)	32	RW	0000_0000h
510h	Bridge port default VLAN register (BPDVR)	32	RW	01F0_0000h
520h	Bridge port spanning tree group state register (BPSTGSR)	32	RW	0000_0002h
528h	Bridge port storm control register 0 (BPSCR0)	32	RW	0FFF_0FFFh
52Ch	Bridge port storm control register 1 (BPSCR1)	32	RW	0FFF_0FFFh
530h	Bridge port operational register (BPOR)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
580h	Bridge port discard count register (BPDCR)	32	R	0000_0000h
588h	Bridge port discard count reason register 0 (BPDCRR0)	32	RW	0000_0000h
58Ch	Bridge port discard count reason register 1 (BPDCRR1)	32	RW	0000_0000h
590h	Bridge port MAC learning failure status register (BPMLFSR)	32	RW	0000_0000h

53.4.6.8.2 Port capability register (PCAPR)

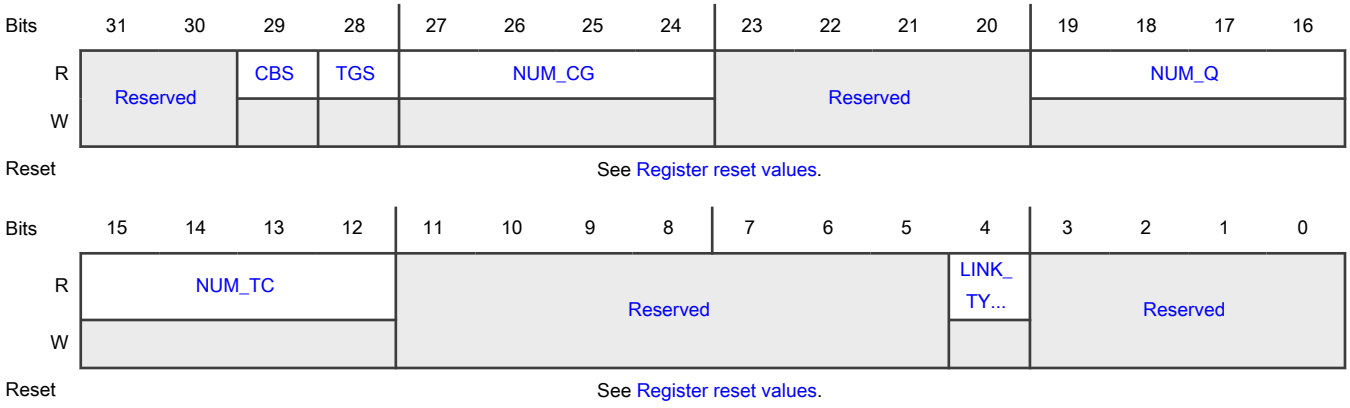
Offset

Register	Offset
PCAPR	0h

Function

This is the port capability register.

Diagram



Register reset values

Register	Reset value
PCAPR	ENETC0_PORT: 3707_7000h ENETC1_PORT: 3707_7010h SW0_PORT0–SW0_PORT3: 3707_7000h SW0_PORT4: 3707_7010h

Fields

Field	Function
31-30 —	Reserved
29 CBS	Credit Based Shaping When set, Credit Based Shaping (CBS i.e: 802.1Qav) is supported.
28 TGS	Time Gate Scheduling When set, time gate scheduling is supported on this port. That is, Enhanced Traffic Selection from IEEE 802.1Qbv standard. See TGSTCAPR for more info.
27-24 NUM_CG	Number of congestion groups supported Formula: NUM_CG+1 **NOTE** Valid if link end is bound to a switch port.
23-20 —	Reserved
19-16 NUM_Q	Number of Egress Traffic Management (ETM) class queues supported Formula: NUM_Q+1 **NOTE** Valid if link is bound to a switch.
15-12 NUM_TC	Number of Traffic Classes (TCs) supported Formula: NUM_TC+1
11-5 —	Reserved
4 LINK_TYPE	Indicates the link type 0 External Link 1 Pseudo Link
3-0 —	Reserved

53.4.6.8.3 Port MAC capability register (PMCAPR)

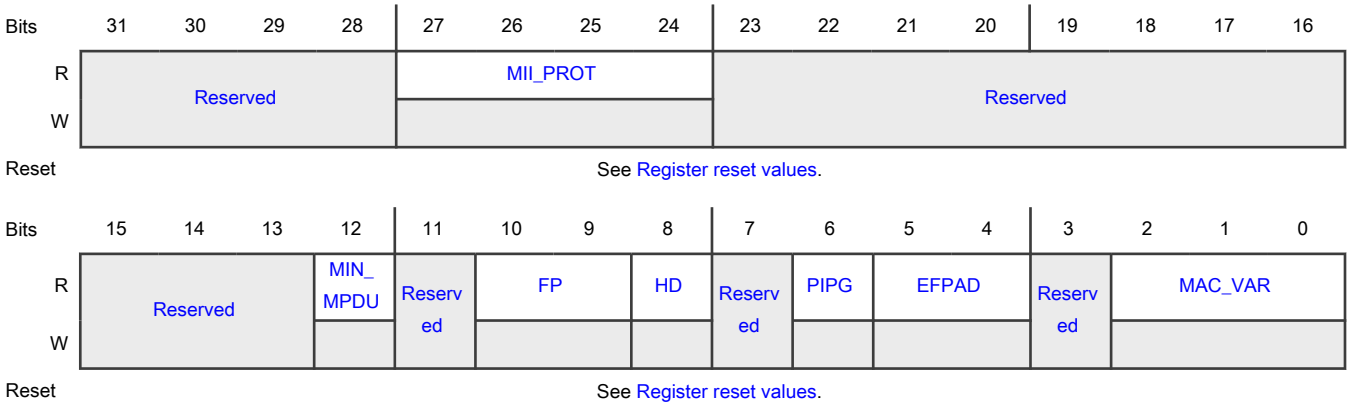
Offset

Register	Offset
PMCAPR	4h

Function

This is the port MAC capability register.

Diagram



Register reset values

Register	Reset value
PMCAPR	ENETC0_PORT: 0000_1561h ENETC1_PORT: 0000_0020h SW0_PORT0–SW0_PORT3: 0000_1561h SW0_PORT4: 0000_0000h

Fields

Field	Function
31-28 —	Reserved
27-24 MII_PROT	Indicates the MII protocol supported 0: MII 1: RMII 2: RGMII 3: GMII

Table continues on the next page...

Table continued from the previous page...

Field	Function
	All other settings reserved. **NOTE** Valid if MAC_VAR=1.
23-13 —	Reserved
12 MIN_MPDU	Minimum MAC Protocol Data Unit (PDU) size check 0: Minimum MAC PDU size check is not supported 1: Supports minimum MAC PDU check. See PM*a*_MINFRM register for more information.
11 —	Reserved
10-9 FP	Indicates if frame preemption is supported 0: Frame preemption is not supported 1: Reserved 2: Frame preemption is supported. The link can preempt transmission of a frame at any byte boundary as long as minimum fragment size requirements are met. 3: Reserved Valid if MAC_VAR=1.
8 HD	Indicates if Half Duplex Mode is supported on this link. 0: Half duplex mode is not supported 1: Half duplex mode is supported **NOTE** Valid if MAC_VAR=1
7 —	Reserved
6 PIPG	Indicates if configurable Preamble and IPG is supported 0: Static Inter Frame Gap (IPG) size is 12B and Preamble is 7B 1: Configurable IPG and Preamble size **NOTE** Valid if MAC_VAR=1
5-4	Egress frame padding capability indicates if egress frames smaller than 64B are padded with zeroes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
EFPAD	0: Frame padding is not supported. 1: All frames less than 64B are padded. 2: Configurable frame padding per port. Note: If frame preemption is supported, both the express and preemptable MACs padding option are configurable independently. 3: Reserved.
3 —	Reserved
2-0 MAC_VAR	MAC Variant 0: Pseudo MAC 1: 1G mEMAC, supporting 10/100 Mbps, 1 Gbps All other values reserved.

53.4.6.8.4 Port I/O capability register (PIOCAPR)

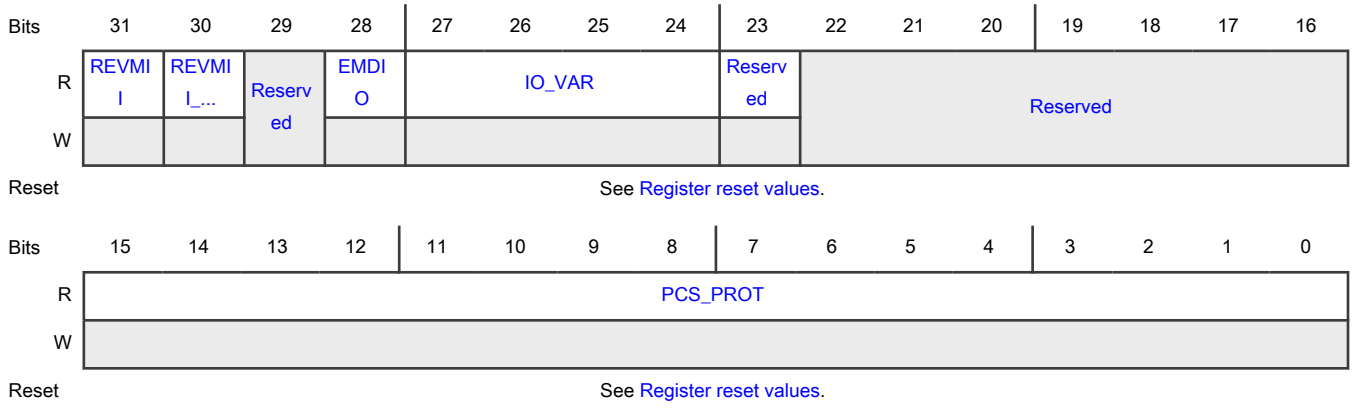
Offset

Register	Offset
PIOCAPR	8h

Function

This is the port I/O capability register.

Diagram



Register reset values

Register	Reset value
PIOCAPR	ENETC0_PORT: 1000_0000h ENETC1_PORT: 0000_0000h SW0_PORT0,SW0_PORT1: 1000_0000h SW0_PORT2,SW0_PORT3: 1000_0000h SW0_PORT4: 0000_0000h

Fields

Field	Function
31 REVMII	Reverse Mode Device Configuration If set, the device has configured link *i* for Reverse operation and IMDIO is supported to access RevMII registers. SW must set PMi_IF_MODE[REVMII] to match.
30 REVMII_RATE	RevMII MII rate If REVMII=1 and MII_PROT=MII, then 0: MII interface is operating at 100 Mbps 1: MII interface is operating at 10 Mbps If RevMII=0 or MII_PROT!=MII, this field has no meaning.
29 —	Reserved
28 EMDIO	External MDIO supported.
27-24 IO_VAR	IO Variants supported 0: None 15: Custom
23 —	Reserved
22-16 —	Reserved
15-0 PCS_PROT	PCS protocols supported xxxx xxxx xxxx xxx1: 1G SGMII xxxx xxxx xxxx xx1x: OC-SGMII (i.e. 2.5G SGMII) xxxx xxxx xxxx x1xx: QGSMII xxxx xxxx xxxx 1xxx: XFI

Table continues on the next page...

Table continued from the previous page...

Field	Function
	xxxx xxxx xxx1 xxxx: SFI xxxx xxxx xx1x xxxx: 10GBase-KR xxxx xxxx x1xx xxxx: 10G-SXGMII 0000 0000 0000 0000: No PCS protocols supported If PCS_PROT != 0, then IMDIO is also supported to access PCS registers

53.4.6.8.5 Port configuration register (PCR)

Offset

Register	Offset
PCR	10h

Function

This is the port configuration register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				PSPEED											
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			Reserved	Reserved			TIMER_CS	Reserved			L2DOSE	Reserved			HDR_FMT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-16 PSPEED	Transmit Port Speed The speed in 10Mbps increments at which the port is assumed to be running for scheduling purposes and to determine if cut-through forwarding can proceed. Formula: 10Mbps * (PSPEED+1)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Range: 10Mbps to 160Gbps
15-13 —	Reserved
12 —	Reserved
11-9 —	Reserved
8 TIMER_CS	Timer Clock Selection: On receive, this field determines which of the two Rx timestamps (synchronized or free running) is reported to the host. On transmit, this field determines which of the two clock sources (synchronized clock or free running clock) is used for sampling the Tx timestamp on the pseudo MAC. For the Ethernet MAC, the PMa_COMMAND_CONFIG[TS_MODE] register determines which of the two clock sources (synchronized clock or free running clock) is used for sampling the Tx timestamp. 0 Synchronized timestamp (receive) and synchronized source clock (transmit, pseudo MAC only) with unit of nanoseconds 1 Free running timestamp (receive) and free running source clock (transmit, pseudo MAC only) with unit of NETC clock ticks
7-5 —	Reserved
4 L2DOSE	L2 Ethernet DoS Protection Enable 0 L2 IP DoS protection is disabled. 1 L2 IP DoS protection is enabled. Refer to DOSL2CR, for more details. Valid if HDR_FMT=0.
3-1 —	Reserved
0 HDR_FMT	Indicates the frame format received/transmitted on the port 0 Ethernet frame format 1 Reserved

53.4.6.8.6 Port MAC address register 0 (PMAR0)

Offset

Register	Offset
PMAR0	20h

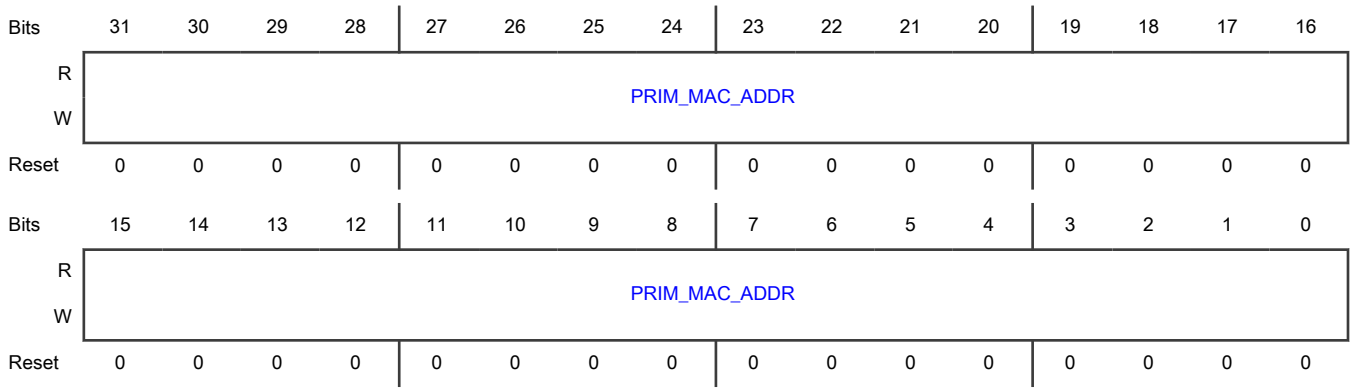
Function

This is the port MAC address register 0.

NOTE

The port will sample the IERB L*a*E*b*MAR0 value when PCIe Memory Access bit is set (0b1). All updates after this must be done through this register.

Diagram



Fields

Field	Function
31-0 PRIM_MAC_ADDR	Primary MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset (register bit offset 0-7). This field contains the 4 most significant bytes of the MAC address. For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then P*a*MAR0 equals 14131211h.

53.4.6.8.7 Port MAC address register 1 (PMAR1)

Offset

Register	Offset
PMAR1	24h

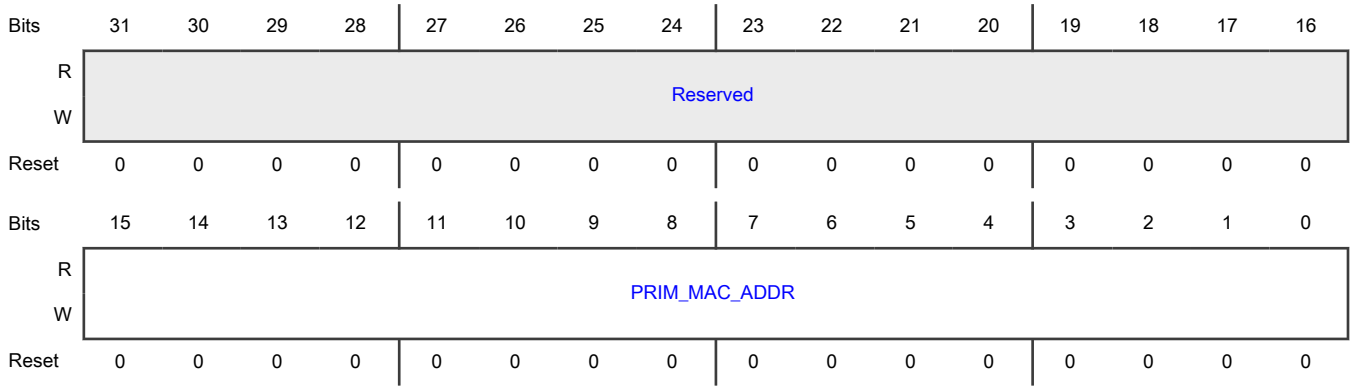
Function

This is the port MAC address register 1.

NOTE

The port will sample the IERB L*a*E*b*MAR1 value when PCIe Memory Access bit is set (0b1). All updates after this must be done through this register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 PRIM_MAC_ADDR	Primary MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset. This field contains the 2 least significant bytes of the MAC address (least significant byte is stored in register bit offset 8-15). For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then P*a*MAR1 equals xxxx1615h (where x should be set to 0).

53.4.6.8.8 Port TPID acceptance register (PTAR)

Offset

Register	Offset
PTAR	50h

Function

This register controls the VLAN classification.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								IVTPIDL				OVTPIDL			
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1

Fields

Field	Function
31-8 —	Reserved
7-4 IVTPIDL	Outer VLAN TPID List : bitmap identifying which TPIDs are acceptable as Inner VLAN tag xxx1 Standard C-VLAN 0x8100 xx1x Standard S-VLAN 0x88A8 x1xx Custom VLAN as defined by CVLANR1[ETYPE] 1xxx Custom VLAN as defined by CVLANR2[ETYPE]
3-0 OVTPIDL	Outer VLAN TPID List : bitmap identifying which TPIDs are acceptable as Outer VLAN tag xxx1 Standard C-VLAN 0x8100 xx1x Standard S-VLAN 0x88A8 x1xx Custom VLAN as defined by CVLANR1[ETYPE] 1xxx Custom VLAN as defined by CVLANR2[ETYPE]

53.4.6.8.9 Port QoS mode register (PQOSMR)

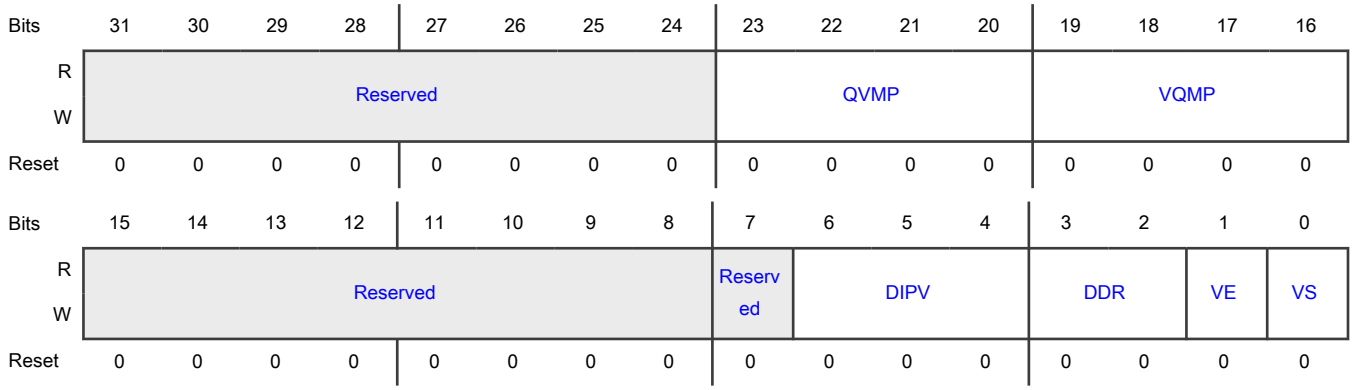
Offset

Register	Offset
PQOSMR	54h

Function

This is the QoS mode register for the port.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 QVMP	Mapping profile index Specifies the Transmit QoS to VLAN PCP Mapping Profile index *a* for accessing register QOSVLANMP*a*R. **NOTE** QVMP range is from {0..SCAPR0[NUM_QVMP]-1}. **NOTE** This field is not supported in every instance. The following table includes only supported registers.
19-16 VQMP	Mapping profile index Specifies the receive VLAN PCP/DE to QoS Mapping Profile index *a* for accessing registers VLANIPVMP*a*R0/1 and VLANDRMP*a*R. Valid if VE=1.

Instance	Field supported in	Field not supported in
ENETC0_PORT	—	PQOSMR
ENETC1_PORT	—	PQOSMR
SW0_PORT0	PQOSMR	—
SW0_PORT1	PQOSMR	—
SW0_PORT2	PQOSMR	—
SW0_PORT3	PQOSMR	—
SW0_PORT4	PQOSMR	—

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE VQMP range is from {0..IPCAPR[NUM_VQMP]-1}
15-8 —	Reserved
7 —	Reserved
6-4 DIPV	Default IPV Sets the default IPV for the port.
3-2 DDR	Default DR Sets the default DR for the port.
1 VE	VLAN enable 0 Defaults are used 1 Enables use of VLAN info to determine IPV and DR (based on VLANIPVMP*a*R0/1 and VLANDRMP*a*R)
0 VS	VLAN tag select: 0 Inner VLAN tag will be used if VE is set 1 Outer VLAN tag will be used if VE is set

53.4.6.8.10 Port Queue Operational register (PQOR)

Offset

Register	Offset
PQOR	60h

Function

This is the port queue operational register

NOTE
Each module instance supports a different number of registers.

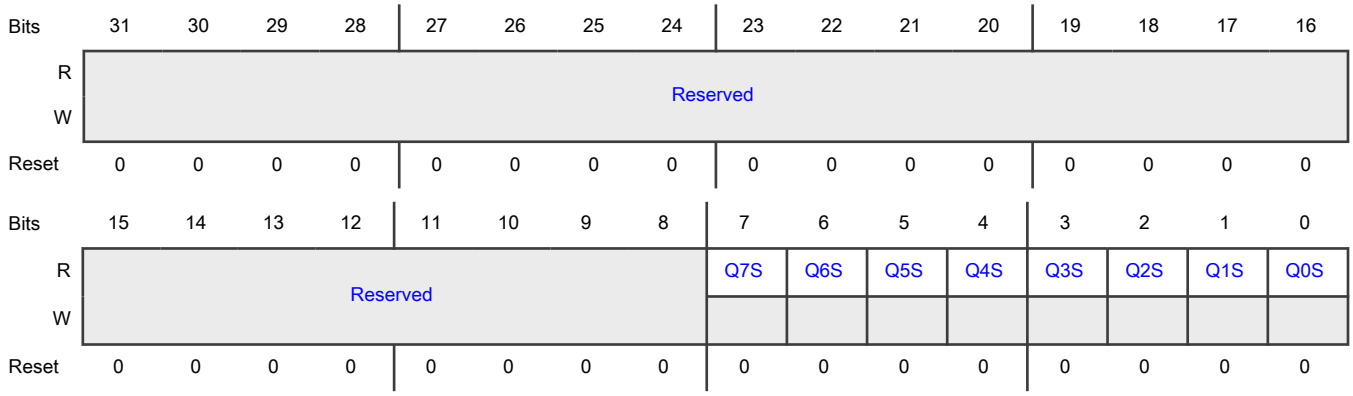
Instance	Register supported	Register not supported
ENETC0_PORT	—	PQOR

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
ENETC1_PORT	—	PQOR
SW0_PORT0	PQOR	—
SW0_PORT1	PQOR	—
SW0_PORT2	PQOR	—
SW0_PORT3	PQOR	—
SW0_PORT4	PQOR	—

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 QnS	Egress Traffic Management (ETM) class queue n's state where i={0..PCAPR[NUM_Q]} 0: Empty; no frames in the class queue 1: Non-empty; one or more frames in the class queue

53.4.6.8.11 Port parser configuration register (PPCR)

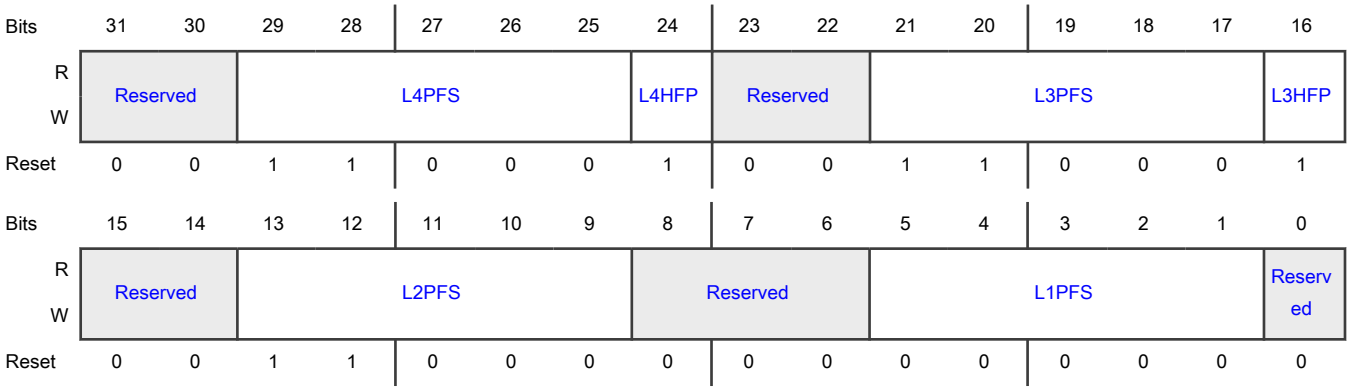
Offset

Register	Offset
PPCR	80h

Function

This register indicates how much the parser needs to look into the frame for the given port. Purpose is to minimize the number of bytes required to be received before keyed lookup operation. This improve packet processing latency for cut-through forwarding.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-25 L4PFS	L4 payload fields size in bytes This is the largest L4 Payload size used by any table lookup key by this port. Payload size range is from 0..24 Bytes, starting immediately after the Destination Port. **NOTE** This field is valid if L4HFP=1.
24 L4HFP	L4 Header fields present **NOTE** Valid if PCR[HDR_FMT]=0. 0b - No L4 header present. Indicates to the parser of not parsing the L4 header. Parsing in this case would go as far as the L3 header if configured to parse it (L3HFP=1) regardless of whether or not there is an L4 header in the frame. This option should not be used if there are any table lookup entries that contain L4 key fields that could be matched against a frame. 1b - Parse L4 header if present in the frame
23-22 —	Reserved
21-17 L3PFS	L3 payload fields size in bytes This is the largest L3 Payload size used by any table lookup key by this port.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Payload size range is from 0..24 Bytes, starting immediately after the Destination IP Address, when parsing is ending at this layer. **NOTE** Valid if L3HFP=1.
16 L3HFP	L3 header fields present **NOTE** Valid if PCR[HDR_FMT]=0. 0b - No L3 header present. Indicates to the parser of not parsing the L3 header. Parsing in this case would go as far as the L2 header regardless of whether or not there is an L3 header in the frame. This option should not be used if there are any table lookup entries that contain L3/L4 key fields that could be matched against a frame. 1b - Parse L3 header if present in the frame.
15-14 —	Reserved
13-9 L2PFS	L2 payload fields size in bytes This is the largest L2 payload size used by any table lookup key by this port. Payload size range is from 0..24 Bytes, starting after the payload EtherType. The payload EtherType starts immediately after the SMAC or VLANs/R-TAG/HSR tag if any, when parsing is ending at this layer. **NOTE** Valid if PCR[HDR_FMT]=0.
8-6 —	Reserved
5-1 L1PFS	Unused
0 —	Reserved

53.4.6.8.12 Port ingress port filter configuration register (PIPFCR)

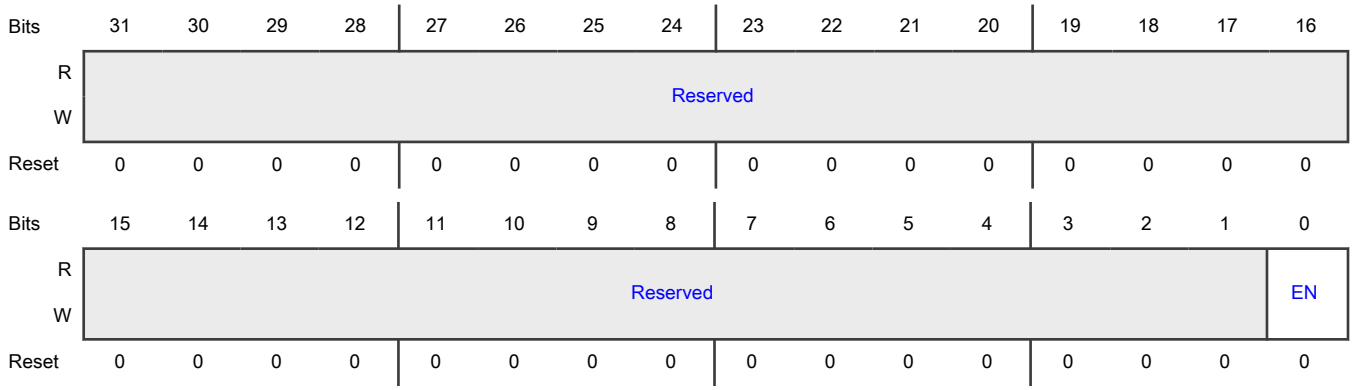
Offset

Register	Offset
PIPFCR	84h

Function

This is the port ingress port filter configuration register

Diagram



Fields

Field	Function
31-1 —	Reserved
0 EN	0 Ingress Port Filter table lookup is disabled for this port 1 Ingress Port Filter table lookup is enabled for this port

53.4.6.8.13 Port stream gate configuration register (PSGCR)

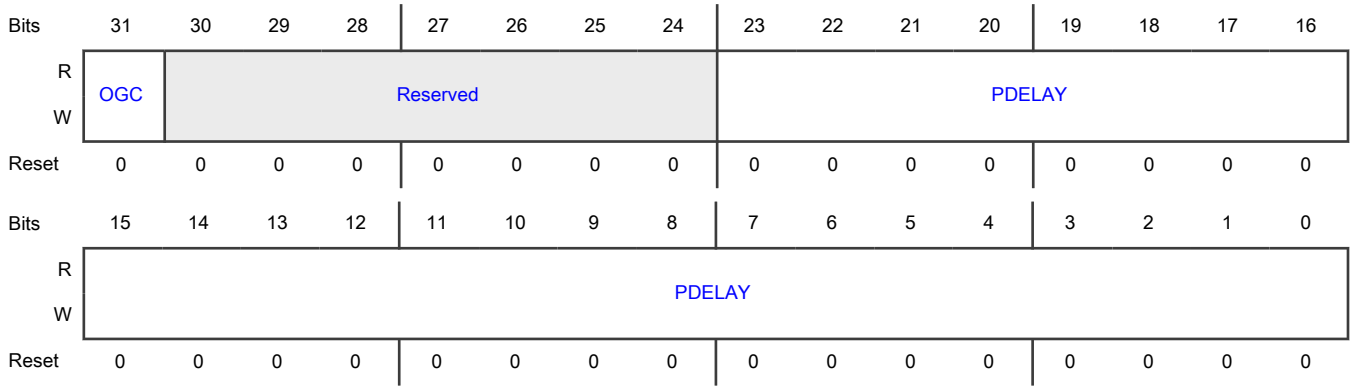
Offset

Register	Offset
PSGCR	A0h

Function

This is the port stream gate configuration register.

Diagram



Fields

Field	Function
31 OGC	Stream Gate Open Gate Check mode 0 Gate open check passes if the Start of Frame (SFD) is within the open gate interval 1 Gate open check passes if the entire frame is within the open gate interval. That is from SFD to End of Frame.
30-24 —	Reserved
23-0 PDELAY	Link propagation delay in nanoseconds. The frame reception timestamp(s) are adjusted by subtracting the link propagation delay from the Start of Frame receive timestamp and from the End of Frame receive timestamp (in the case where OGC is set to 1).

53.4.6.8.14 Port operational register (POR)

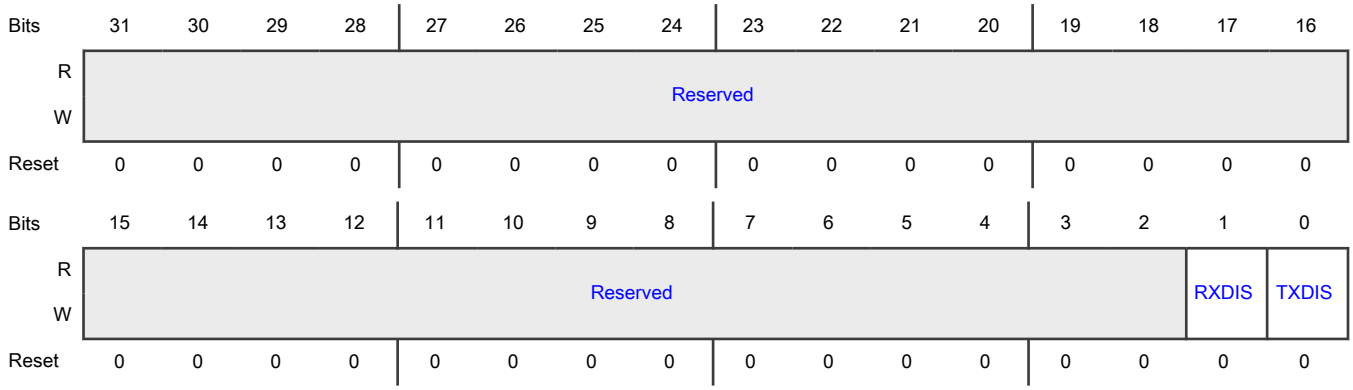
Offset

Register	Offset
POR	100h

Function

This is the port operational register

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RXDIS	Rx Disable. The frame discards are counted against the port's Rx discard count register (PRXDCR) along with the setting of the Receive Disable Discard Reason (RXDISDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0). 0b - Rx path is enabled. 1b - Rx path is disabled. Any partially Rx reassembled frame, is discarded. While the Rx path is disabled, any new frame received from the MAC is discarded.
0 TXDIS	Tx Disable. While disabled, the following behaviors take place: • Any frame to be enqueued internally onto any switch Egress Traffic Management (ETM) class queue, is discarded. The frame discard is counted against the port's Tx discard count register (PTXDCR) along with the setting of the "Transmit Disable prior to Enqueue to ETM Discard Reason" (TXDISEDR) flag to 1 in the port's Tx discard count reason register 0 (PTXDCRR0). • Buffer descriptors from the ENETC transmit BD rings are no longer loaded from the host memory to internal memory. • The transmit scheduling mechanisms (e.g. time gate scheduling) will continue to function but with no frames available to be scheduled. Once enabled, the transmit scheduling mechanisms will resume normal operation without requiring any additional intervention. 0b - Tx path is enabled 1b - Tx path is disabled. Any frame queued internally on the port for transmission is flushed. Direct enqueued host originated switch management frames, area also discarded. The frame discards are counted against the port's Tx discard count register (PTXDCR) along with the setting of the "Transmit Disable Discard Reason" (TXDISDR) to 1 in the port's Tx discard count reason register 0 (PTXDCRR0).

53.4.6.8.15 Port status register (PSR)

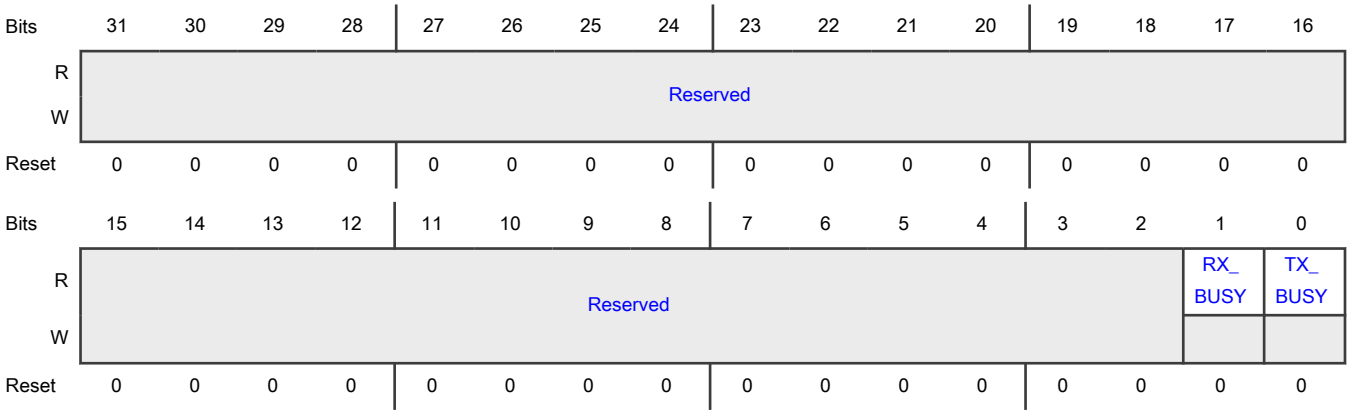
Offset

Register	Offset
PSR	104h

Function

This is the port status register.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RX_BUSY	Receive busy. Ingress (or receive) datapath processing pipeline is busy processing received frames from this port. 0b - Idle 1b - Busy
0 TX_BUSY	Transmit busy. Egress (or transmit) datapath processing pipeline is busy processing frames that are to be transmitted out of this port. 0b - Idle 1b - Busy

53.4.6.8.16 Port receive SDU overhead register (PRXSDUOR)

Offset

Register	Offset
PRXSDUOR	108h

Function

Overhead (number of bytes) to be added to the actual length of each frame based on the following selected SDU type:

0: PPDU (Physical Layer PDU) overhead: PPDU_BCO + MACSEC_BCO

1: MPDU (MAC PDU) overhead: MACSEC_BCO

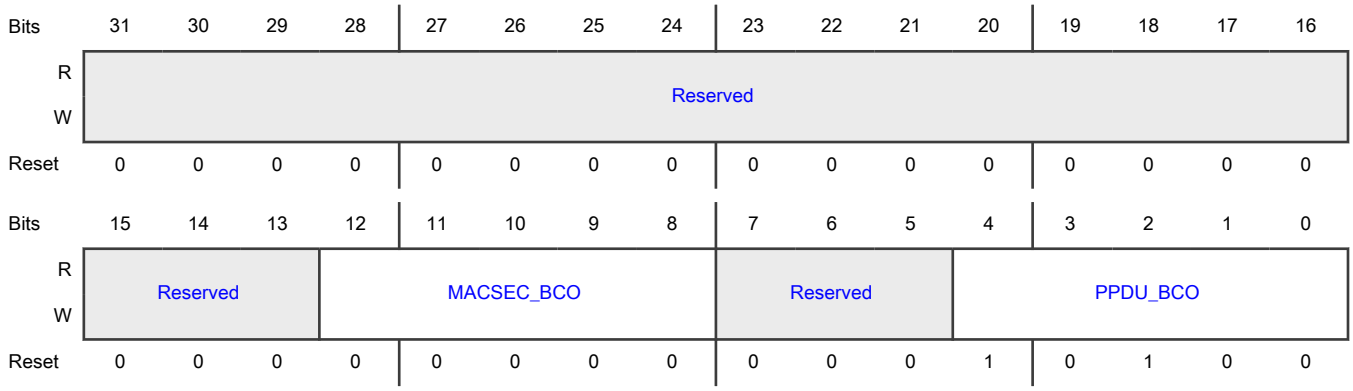
2: MSDU (MAC SDU) overhead: minus 16B (12B MAC Header + 4B FCS)

All other setting reserved.

This register is used by:

- Rate policing function
- Stream filtering function
- Stream gating function

Diagram



Fields

Field	Function
31-13 —	Reserved
12-8 MACSEC_BCO	MACSec byte count overhead Number of bytes of overhead due to MACSec encapsulation.
7-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 PPDU_BCO	PPDU Byte count overhead PPDU Byte count overhead which includes IPG, SFD and Preamble. **NOTE** Default is 20 which is IPG(12) + SFD(1) + Preamble(7).

53.4.6.8.17 Port transmit SDU overhead register (PTXSDUOR)

Offset

Register	Offset
PTXSDUOR	10Ch

Function

Overhead (number of bytes) to be added to the actual length of each frame based on the selected SDU type

0: PPDU (Physical Layer PDU): PPDU_BCO + MACSEC_BCO

1: MPDU (MAC PDU): MACSEC_BCO

2: MSDU (MAC SDU): minus 16B (12B MAC Header + 4B FCS)

This register is used by:

- Per traffic class maximum frame length check where the SDU type is specified by PTC*a*TMSDUR.
- Time Gate Scheduling (TGS) where PPDU length is used to ensure that entire frame can be transmitted on the wire while the gate is open.
- Credit Base Shaping (CBS) where PPDU length is used to calculate the amount of bandwidth usage on the link

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				MACSEC_BCO				Reserved				PPDU_BCO			
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

Fields

Field	Function
31-13 —	Reserved
12-8 MACSEC_BCO	MACSec byte count overhead Number of bytes of overhead due to MACSec encapsulation.
7-5 —	Reserved
4-0 PPDU_BCO	PPDU Byte count overhead PPDU Byte count overhead which includes IPG, SFD and Preamble. **NOTE** Default is 20 which is IPG(12) + SFD(1) + Preamble(7).

53.4.6.8.18 Port time gate scheduling control register (PTGSCR)

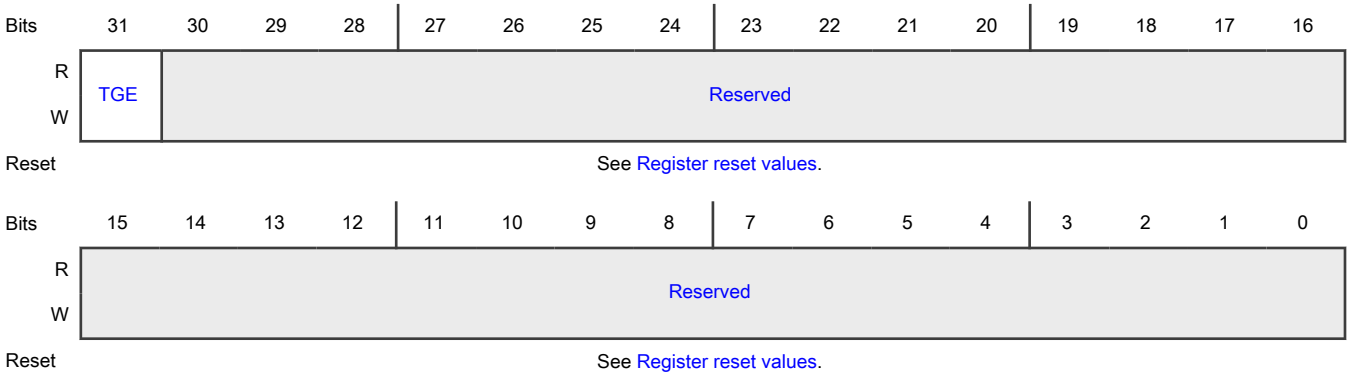
Offset

Register	Offset
PTGSCR	110h

Function

This register controls the port time gate scheduling operation.

Diagram



Register reset values

Register	Reset value
PTGSCR	ENETC0_PORT: 0000_0000h ENETC1_PORT: 8000_0000h SW0_PORT0–SW0_PORT3: 0000_0000h SW0_PORT4: 8000_0000h

Fields

Field	Function
31 TGE	Time Gating Enable Enables and disables the time gating function on the port. When disabling and then re-enabling, the disable time must be greater than the advance offset time configured in S*a*TGSLR/E*a*TGSLR[MIN_LOOKAHEAD] + PTGSATOR[ADV_TIME_OFFSET]. Software must also wait until PTGAGLSR[TG] is de asserted before re-enabling. 0b - Time gating disabled. The state of each gate is determined by the setting in PDGSR[DGS_TCn], where n is the traffic class number. If this field is set to 0 (disabling time gate scheduling), the operational gate control list and administrative gate control list (if present), will be removed. 1b - Time gating enabled. The initial state of each gate is determined by the setting in PDGSR[DGS_TCn], where n is the traffic class number. This field must be set to 1 (enabling time gate scheduling) before any administrative gate control list can be configured.
30-0 —	Reserved

53.4.6.8.19 Port time gate scheduling admin gate list status register (PTGAGLSR)

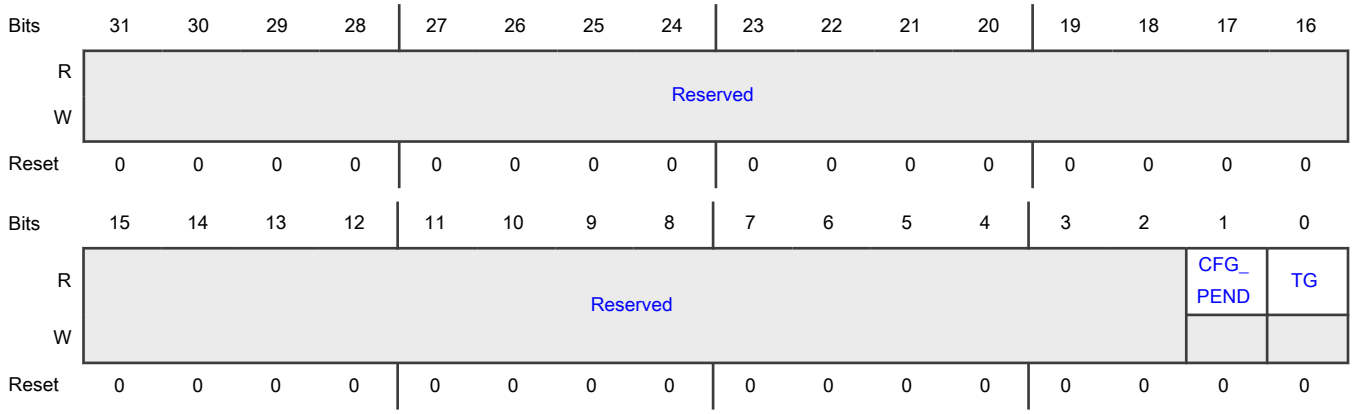
Offset

Register	Offset
PTGAGLSR	114h

Function

This is the port time gate scheduling admin gate list status register.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 CFG_PEND	Administrative gate control list pending 0 No administrative gate control list is configured 1 Administrative gate control list is pending (configured but not installed yet)
0 TG	Time gated (state of the operational gate control list) 0 No operational gate control list is active 1 Operational gate control list is active

53.4.6.8.20 Port time gate scheduling admin gate list length register (PTGAGLLR)

Offset

Register	Offset
PTGAGLLR	118h

Function

This is the port time gate scheduling administrative gate control list length register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADMIN_GATE_LIST_LENGTH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 ADMIN_GATE_LIST_LENGTH	Administrative gate control list length (number of entries). This field is read-only and reflects the current setting when an administrative gate control list is pending, PTGAGLSR[CFG_PEND] is set to 1.

53.4.6.8.21 Port time gating operational gate list length register (PTGOGLLR)

Offset

Register	Offset
PTGOGLLR	11Ch

Function

This is the port time gating operational gate control list length register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OPER_GATE_LIST_LENGTH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 OPER_GATE_LIST_LENGTH	Operational gate control list length (number of entries). This field is read-only and reflects the current setting when an operational gate control list is active, PTGAGLSR[TG] is set to 1.

53.4.6.8.22 Port time gate scheduling advance time offset register (PTGSATOR)

Offset

Register	Offset
PTGSATOR	120h

Function

This is the port time gate scheduling advance time offset register.

NOTE

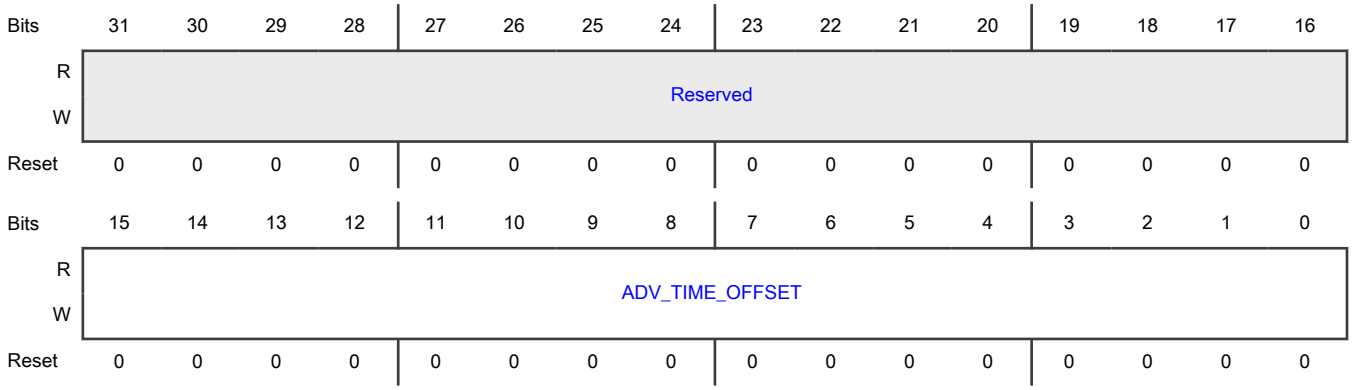
The function will sample the value in this register when PCIe Memory Access bit is set (0b1).

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	PTGSATOR	—
ENETC1_PORT	PTGSATOR	—
SW0_PORT0	—	PTGSATOR
SW0_PORT1	—	PTGSATOR
SW0_PORT2	—	PTGSATOR
SW0_PORT3	—	PTGSATOR
SW0_PORT4	—	PTGSATOR

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 ADV_TIME_OFFSET	Advance time offset This field specifies the amount of time to advance the IEEE 1588 time scale used by the time-based scheduler, to adjust for the latency encountered across the MAC plus if needed, delays outside of NETC (e.g. PHY delay). The time is specified in units of nanoseconds. The time value specified in this register advances both, the IEEE 1588 time scale used at the frame scheduling timing point (entry point of the time-based scheduler) and the IEEE 1588 time scale used at the point where frames are transferred to the MAC, which is implemented by a timestamp-based shaper (exit point of the time-based scheduler). Note that the IEEE 1588 time scale used by the time-based scheduler at the frame scheduling timing point, is also advanced, by the amount of time specified in S0TGSLR[MIN_LOOKAHEAD] for the switch and in E*a*TGSLR[MIN_LOOKAHEAD] for ENETC, to adjust for the time required to schedule, dequeue and for ENETC to load (or DMA) frames from the host memory to NETC internal memory. Advanced times specified respectively in S0TGSLR/E*a*TGSLR[MIN_LOOKAHEAD] and in PTGSATOR[ADV_TIME_OFFSET] are cumulative, at the frame scheduling timing point; i.e. the IEEE 1588 time scale used by the frame scheduling timing point (entry point of the time-based scheduler) on a given port. That is, the time reference at the frame scheduling point will be advanced by the amount of time resulting from adding the S0TGSLR/E*a*TGSLR[MIN_LOOKAHEAD] time to the PTGSATOR[ADV_TIME_OFFSET] time.

53.4.6.8.23 Port time gate scheduling hold advance register (PTGSHAR)

Offset

Register	Offset
PTGSHAR	124h

Function

This is the port time gate scheduling hold advance register

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	PTGSHAR	—
ENETC1_PORT	—	PTGSHAR
SW0_PORT0	PTGSHAR	—
SW0_PORT1	PTGSHAR	—
SW0_PORT2	PTGSHAR	—
SW0_PORT3	PTGSHAR	—
SW0_PORT4	—	PTGSHAR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HOLDADVANCE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 HOLDADVANCE	This field indicates the amount of time prior to the Set-And-Hold-MAC time slot for asserting the Hold. The time is expressed in units of nanoseconds. This time interval corresponds to the transmission of the worst case (longest) fragment length that cannot be preempted plus associated physical layer overheads. For preemption, IEEE defines a minimum final fragment size of 64 bytes and allows the minimum non-final fragment size to be 64, 128, 192, or 256 bytes. For example, with a non-final fragment size of 64 bytes, a fragment that is 123 or fewer bytes in length cannot be preempted, because the non-final fragment would

Table continues on the next page...

Table continued from the previous page...

Field	Function
	be less than 64 bytes (60 bytes + 4 bytes for mCRC). The minimum non-final fragment size is configured in MAC_MERGE_MMCSR[RAFS]. Thus in the case, where MAC_MERGE_MMCSR[RAFS] is set to 64 bytes, meaning that the worst case (longest) fragment length that cannot be preempted is 123 bytes, the time for asserting the hold, would be set to 163 bytes transmission time prior to start the Set-And-Hold-MAC time slot; 20 bytes (IFG, preamble, SFD) + 123 bytes (non-preemptable fragment) + 20 bytes (IFG, preamble, SFD). The value reported in this field is 163 + 64 * MAC_MERGE_MMCSR[RAFS] byte time at PCR[PSPEED]. Note that this value doesn't account for any latency encountered in the MAC and MAC merge sublayer. Software needs to read PTGSATOR[ADV_TIME_OFFSET] and PTGSHCR[HOLD_SKEW]) to determine the latency adjustment across the MAC and MAC Merge layer.

53.4.6.8.24 Port time gate scheduling release advance register (PTGSRAR)

Offset

Register	Offset
PTGSRAR	128h

Function

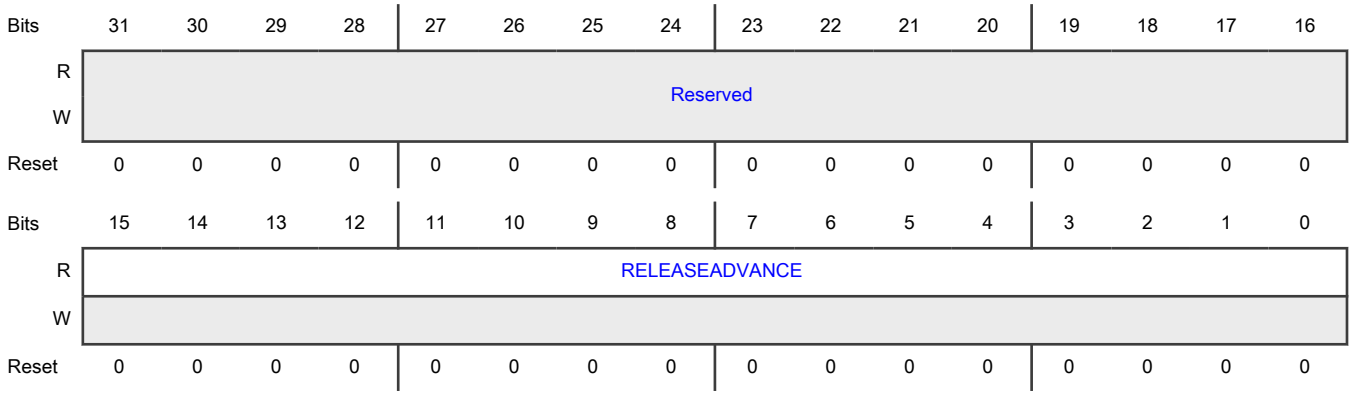
This is the port time gate scheduling release advance register

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	PTGSRAR	—
ENETC1_PORT	—	PTGSRAR
SW0_PORT0	PTGSRAR	—
SW0_PORT1	PTGSRAR	—
SW0_PORT2	PTGSRAR	—
SW0_PORT3	PTGSRAR	—
SW0_PORT4	—	PTGSRAR

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 RELEASEADVANCE	This field indicates the amount of time prior to the Set-And-Release-MAC time slot for asserting the Release. The time is expressed in units of nanoseconds. The value reported in this field is 8 byte time (preamble, SFD) at PCR[PSPEED]. Note that this value doesn't account for any latency encountered in the MAC and MAC merge sublayer. Software needs to read PTGSATOR[ADV_TIME_OFFSET] and PTGSHCR[HOLD_SKEW]) to determine the latency adjustment across the MAC and MAC Merge layer.

53.4.6.8.25 Port time gate scheduling hold configuration register (PTGSHCR)

Offset

Register	Offset
PTGSHCR	12Ch

Function

This register controls the port time gate scheduling hold configuration.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	PTGSHCR	—
ENETC1_PORT	—	PTGSHCR

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_PORT0	PTGSHCR	—
SW0_PORT1	PTGSHCR	—
SW0_PORT2	PTGSHCR	—
SW0_PORT3	PTGSHCR	—
SW0_PORT4	—	PTGSHCR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved												HOLD_SKEW			
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HOLD_SKEW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19-0 HOLD_SKEW	Hold-Skew in nanoseconds. This field specifies the amount of time to advance the IEEE 1588 time scale used in the transmit processing pipeline at the point where Hold/Release signal is asserted to the MAC merge sublayer. The time is specified in units of nanoseconds. This field is used to account for Hold/Release delay from the MAC client to the MAC merge sublayer plus if needed, frame delays outside of NETC (e.g. PHY delay). Note that if this field accounts for frame delays outside NETC, these delays must also be accounted in the same manner (same value) in PTGSATOR[ADV_TIME_OFFSET].

53.4.6.8.26 Port frame preemption configuration register (PFPCR)

Offset

Register	Offset
PFPCR	134h

Function

The port frame preemption configuration register enables the feature associated with the traffic class.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	PFPCR	—
ENETC1_PORT	—	PFPCR
SW0_PORT0	PFPCR	—
SW0_PORT1	PFPCR	—
SW0_PORT2	PFPCR	—
SW0_PORT3	PFPCR	—
SW0_PORT4	—	PFPCR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								FPE_TC7	FPE_TC6	FPE_TC5	FPE_TC4	FPE_TC3	FPE_TC2	FPE_TC1	FPE_TC0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7-0 FPE_TCn	Frame preemption enable for traffic class n 0 Disabled. Frames from traffic class n are transmitted on the express MAC. 1 Enabled. Frames from traffic class n are transmitted on the preemptable MAC.

53.4.6.8.27 Port default gate state register (PDGSR)

Offset

Register	Offset
PDGSR	138h

Function

Configures the default state of the port's traffic class gates.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								DGS_TC7	DGS_TC6	DGS_TC5	DGS_TC4	DGS_TC3	DGS_TC2	DGS_TC1	DGS_TC0
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Fields

Field	Function
31-8 —	Reserved
7-0 DGS_TCn	Default Gate State for Traffic Class n Configures the default state of the port's traffic class gates. This setting is used by hardware to set the state of the traffic class gates when no gate control list is operational, or when time gate scheduling is disabled (PTGSCR[TGE]=0b). It may be written at any time. It takes effect immediately if no gate control list is operational, or when time gate scheduling is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Closed 1b - Open

53.4.6.8.28 Port Rx discard count register (PRXDCR)

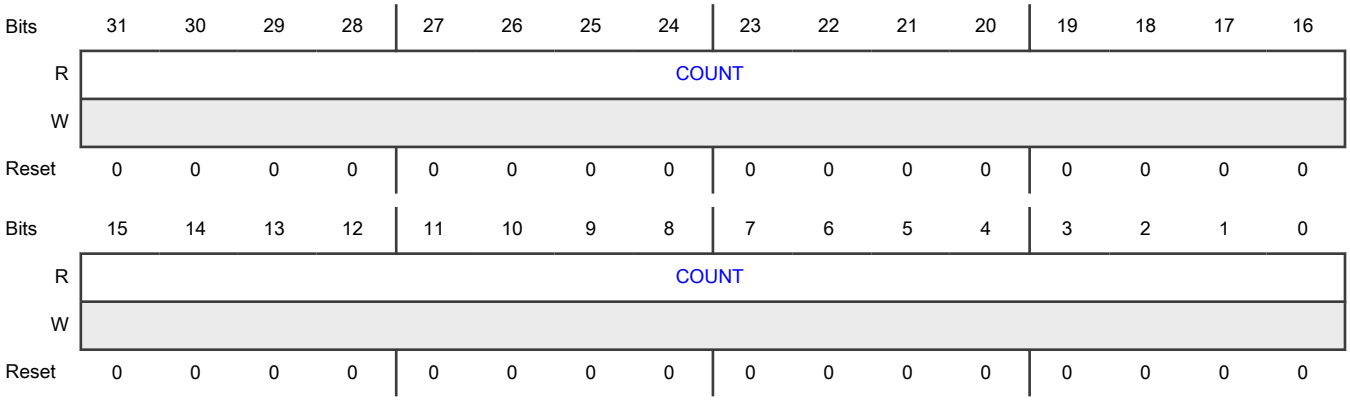
Offset

Register	Offset
PRXDCR	1C0h

Function

This is the port Rx discard count register.

Diagram



Fields

Field	Function
31-0 COUNT	Number of receive frames discarded

53.4.6.8.29 Port Rx discard count reason register 0 (PRXDCRR0)

Offset

Register	Offset
PRXDCRR0	1C8h

Function

This is the port Rx discard count reason register 0. Reasons are indicated progressively as they are detected with the exception of the reason flags PCDR, SMREDR, RXDISDR. See flags description below, for more details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														NODESTDR	Reserved
W															W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	L2DOSDR	SIFDR	ECCEDR	ITEDR	CMDR	FMMEDR	MSDUEDR	SGOEDR	SGCDR	ISFDR	RPDR	IPFDR	RXDISDR	SMREDR	PCDR
W		W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved
17 NODESTDR	No Destination Discard Reason Occurs if ingress stream's forwarding action is stream forwarding (FA=010b) to a null destination (EGRESS_PORT_BITMAP=0). Cleared when written to 1.
16 —	Reserved
15 —	Reserved
14 L2DOSDR	Layer 2 Denial of Service Discard Reason Occurs if a packet is discarded due to L2DOS protection. See PCR[L2DOSE] for more information. Cleared when written to 1.
13 SIFDR	Station Interface Filter Discard Reason Occurs if packet is discarded by the L2 Filtering function. For more granular L2 Filtering drop counters, see the following registers: PUFDMFR, PMFDMFR, PBFDSIR, PUFDVFR, PMFDVFR, PBFDVFR and PFDMSAPR. Cleared when written to 1.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>ENETC0_PORT</td><td>PRXDCRR0</td><td>—</td></tr>
<tr><td>ENETC1_PORT</td><td>PRXDCRR0</td><td>—</td></tr>
<tr><td>SW0_PORT0</td><td>—</td><td>PRXDCRR0</td></tr>
<tr><td>SW0_PORT1</td><td>—</td><td>PRXDCRR0</td></tr>
<tr><td>SW0_PORT2</td><td>—</td><td>PRXDCRR0</td></tr>
<tr><td>SW0_PORT3</td><td>—</td><td>PRXDCRR0</td></tr>
<tr><td>SW0_PORT4</td><td>—</td><td>PRXDCRR0</td></tr>
</table>
</td></tr>
<tr><td>12
ECCEDR</td><td>ECC Error Discard Reason.
• Unrecoverable ECC error detected while reading an entry from frame header memory, or from any of the lookup tables.
Cleared when written to 1.</td></tr>
<tr><td>11
ITEDR</td><td>Invalid Table Entry Discard Reason
• Table entry ID reference is not found in the table (entry has not been added to the table)
• Table entry ID is outside of its table allocation range
Cleared when written to 1.</td></tr>
<tr><td>10
CMDR</td><td>Congestion Management Discard Reason
• Frame discarded due to insufficient amount of memory space in buffer pool or in switch shared internal buffer memory.
Cleared when written to 1.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>ENETC0_PORT</td><td>—</td><td>PRXDCRR0</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

Field	Function

Instance	Field supported in	Field not supported in
ENETC1_PORT	—	PRXDCRR0
SW0_PORT0	PRXDCRR0	—
SW0_PORT1	PRXDCRR0	—
SW0_PORT2	PRXDCRR0	—
SW0_PORT3	PRXDCRR0	—
SW0_PORT4	PRXDCRR0	—

Field	Function
9 FMMEDR	Frame Modification Misconfiguration Error Discard Reason Cleared when written to 1. **NOTE** Valid only for Switch. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
ENETC0_PORT	—	PRXDCRR0
ENETC1_PORT	—	PRXDCRR0
SW0_PORT0	PRXDCRR0	—
SW0_PORT1	PRXDCRR0	—
SW0_PORT2	PRXDCRR0	—
SW0_PORT3	PRXDCRR0	—
SW0_PORT4	PRXDCRR0	—

Field	Function
8 MSDUEDR	Ingress Stream Maximum Service Data Unit Exceeded Discard Reason Occurs if packet received is greater than the Maximum SDU size configured for the stream. Cleared when written to 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If set,PRXDCRR1[TT] and PRXDCRR1[ENTRYID] are not updated.
7 SGOEDR	Stream Gate Octet Exceeded Discard Reason • Frame dropped due to octet count exceeded maximum specified for the open gate interval Cleared when written to 1.
6 SGCDR	Stream Gate Closed Discard Reason • Frame received on this port that is dropped because it didn't pass the stream gate. Cleared when written to 1.
5 ISFDR	Ingress Stream Forwarding Discard Reason Occurs if frame is associated with an ingress stream whose Forwarding Action (FA) is equal to discard. Cleared when written to 1.
4 RPDR	Rate Policer Discard Reason. Frame received on this port that is dropped by any of the rate policer instances. Cleared when written to 1.
3 IPFDR	Ingress Port Filter Discard Reason. Cleared when written to 1.
2 RXDISDR	Receive disable discard reason due to port being in receive disabled state (POR[RXDIS]=1). When this reason is reported by hardware (set to1), all other reason flags in registers PRXDCRR0/1 are cleared. Cleared when written to 1.
1 SMREDR	Shared Memory Resource Exhaustion Discard Reason Occurs if Ethernet Rx I/F cannot allocate shared internal buffer memory to store the entire received frame. When this reason is reported by hardware (set to1), all other reason flags in registers PRXDCRR0/1 are cleared. Cleared when written to 1.
0 PCDR	Pre-Classification Discard Reason Occurs if Parse Classifier Engine (PCE) block does not have any free processing thread to process the received frame. The PCE realizes the Ingress Packet Processing (Switch) and Ingress Port Processing (ENETC) functional blocks. When this reason is reported by hardware (set to1), all other reason flags in registers PRXDCRR0/1 are cleared. Cleared when written to 1.

53.4.6.8.30 Port Rx discard count reason register 1 (PRXDCRR1)

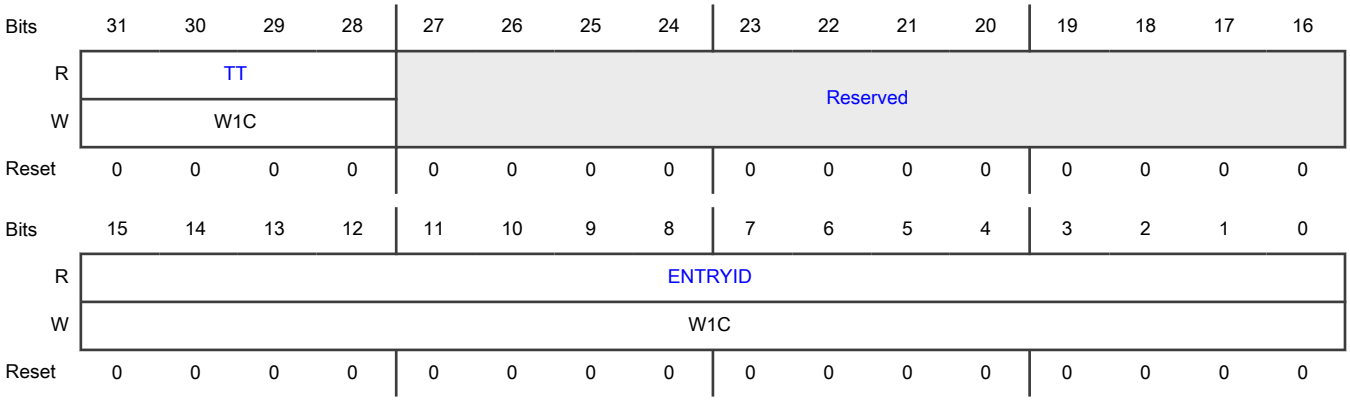
Offset

Register	Offset
PRXDCRR1	1CCh

Function

This is the port Rx discard count reason register 1. Reasons are indicated progressively as they are detected.

Diagram



Fields

Field	Function
31-28 TT	Table type which caused the discard. 0 None 1 Ingress Port Filter table 2 Rate Policer table 3 Ingress Stream Identifier table 4 Ingress Stream table 6 Ingress Sequence Generation table (switch only) 7 Stream Gating Instance table; A discard due to the Stream Gate Control list table is reflected here as a discard due to the Stream Gating Instance table. 8 Frame Modification table (switch only) 15 Congestion Management where ENTRYID represents Buffer Pool ID (switch only) All other settings reserved. Cleared when written to 1.
27-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-0 ENTRYID	Entry Id who last incremented discard count. Field represent Ingress Stream Identifier if ingress Stream has been identified; otherwise field is not updated. Cleared when written to 1.

53.4.6.8.31 Port Tx discard count register (PTXDCR)

Offset

Register	Offset
PTXDCR	1E0h

Function

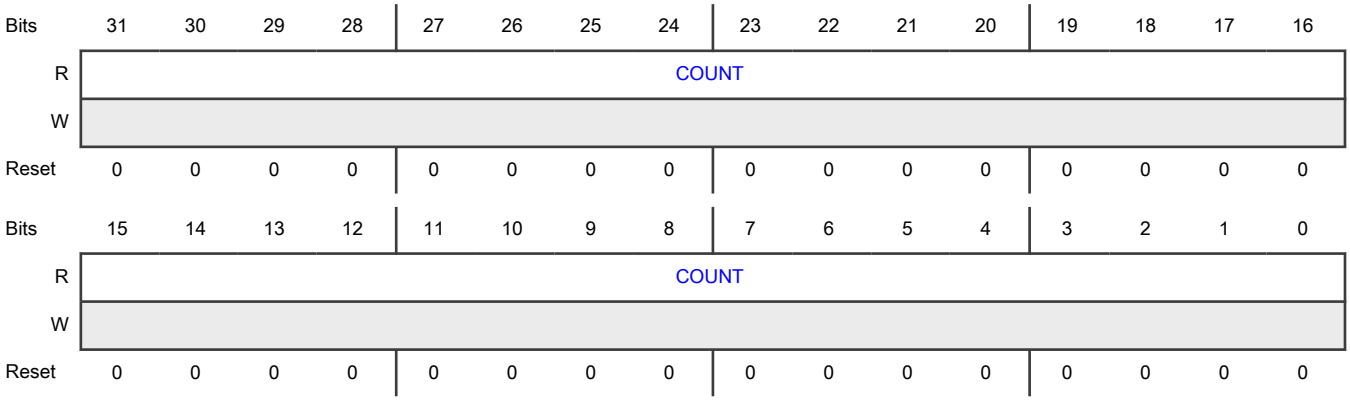
This is the port Tx discard count register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	PTXDCR
ENETC1_PORT	—	PTXDCR
SW0_PORT0	PTXDCR	—
SW0_PORT1	PTXDCR	—
SW0_PORT2	PTXDCR	—
SW0_PORT3	PTXDCR	—
SW0_PORT4	PTXDCR	—

Diagram



Fields

Field	Function
31-0 COUNT	Number of transmit frames discarded

53.4.6.8.32 Port Tx discard count reason register 0 (PTXDCRR0)

Offset

Register	Offset
PTXDCRR0	1E8h

Function

This is the port Tx discard count reason register 0.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	PTXDCRR0
ENETC1_PORT	—	PTXDCRR0
SW0_PORT0	PTXDCRR0	—
SW0_PORT1	PTXDCRR0	—
SW0_PORT2	PTXDCRR0	—
SW0_PORT3	PTXDCRR0	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_PORT4	PTXDCRR0	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SMREDR	SQRDDR	SQRRDR	Reserved	SQRTNSQ...	INVEQDR	ITEDR	QCONGDR	MSDUEDR	TXDISEDR	FMMDR	TGSFTLDR	Reserved	PEDR	ECCEDR	TXDISDR
W	W1C	W1C	W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C		W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 SMREDR	Shared Memory Resource Exhaustion Discard Reason. Occurs if Ethernet Tx I/F (or Tx MAC client) cannot allocate a memory resource when modifying the frame. Cleared when written to 1.
14 SQRDDR	Egress Stream Sequence Recovery Duplicate Discard Reason Occurs if packet's sequence_number is a duplicate applies for either Match or Vector Recovery Algorithm). Cleared when written to 1.
13 SQRRDR	Egress Stream Sequence Recovery Rogue Discard Reason Occurs if a packet is discarded by the vector recovery algorithm because its sequence number has fallen outside of the recovery history window. Cleared when written to 1.
12 —	Reserved
11 SQRTNSQDR	Egress Stream Sequence Recovery Take No Sequence Discard Reason Occurs if the sequence recovery function discards a frame because it is tagless and ESQA_TGT_EID[SQR_TNSQ]=0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	A frame is considered tagless if it has no sequence tag, or if ESQA_TGT_EID[SQ_TAG] is not 0 and the tag type in the frame does not match the tag type specified in ESQA_TGT_EID[SQ_TAG]. Cleared when written to 1.
10 INVEQDR	Invalid Enqueue Port or Queue Discard Reason. Occurs if an enqueue is attempted to an invalid switch port or ETM class queue. Cleared when written to 1.
9 ITEDR	Invalid Table Entry Discard Reason. Occurs if • Table entry reference is not valid (i.e: was not allocated) • Table entry ID is outside of its table allocation range Cleared when written to 1.
8 QCONGDR	Queue Congestion Discard Reason Occurs if a frame is dropped because it couldn't be enqueued to an Egress Traffic Management (ETM) class queue. Cleared when written to 1. **NOTE** Each ETM class *a* set of queue also has a discard related counts.
7 MSDUEDR	Egress Maximum Service Data Unit Exceeded Discard Reason Occurs if store-and-forward frame is greater than the port Traffic Class's configured Maximum SDU size. Refer to PTC*a*TMSDUR. Cleared when written to 1.
6 TXDISEDR	Transmit Disable prior to Enqueue to ETM Discard Reason due to Port being in a Transmit disabled state (POR[TXDIS]=1). Cleared when written to 1.
5 FMMDR	Frame Modification Misconfiguration Discard Reason Software Misconfiguration due to: • Invalid frame header modification (i.e: delete a VLAN which is not present, ...) • Action to delete more payload bytes than amount of payload available in the frame. • Frame size plus modifications is greater than 16383 bytes. Cleared when written to 1.
4 TGSFTLDR	Time Gate Scheduling Frame Too Large Discard Reason Time Gate scheduling is enabled, and the frame was discarded because one of the following conditions was encountered:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• The frame was too large to transmit during any gate-open interval. • There were no gate-open intervals for that traffic class in the entire operational gate control list; its gate was closed in every entry of the operational gate control list. Cleared when written to 1.
3 —	Reserved
2 PEDR	Parity Error Discard Reason Frame is discarded due to a parity error. Cleared when written to 1.
1 ECCEDR	ECC Error Discard Reason Frame is discarded due to unrecoverable ECC error detected while reading a table entry or frame from common memory. Cleared when written to 1.
0 TXDISDR	Transmit disable discard reason due to port being in transmit disabled state (POR[TXDIS]=1). This includes all frames flushed from Egress Traffic Management (ETM) class queues. Cleared when written to 1.

53.4.6.8.33 Port Tx discard count reason register 1 (PTXDCRR1)

Offset

Register	Offset
PTXDCRR1	1ECh

Function

This is the port Tx discard count reason register 1.

NOTE

Each module instance supports a different number of registers.

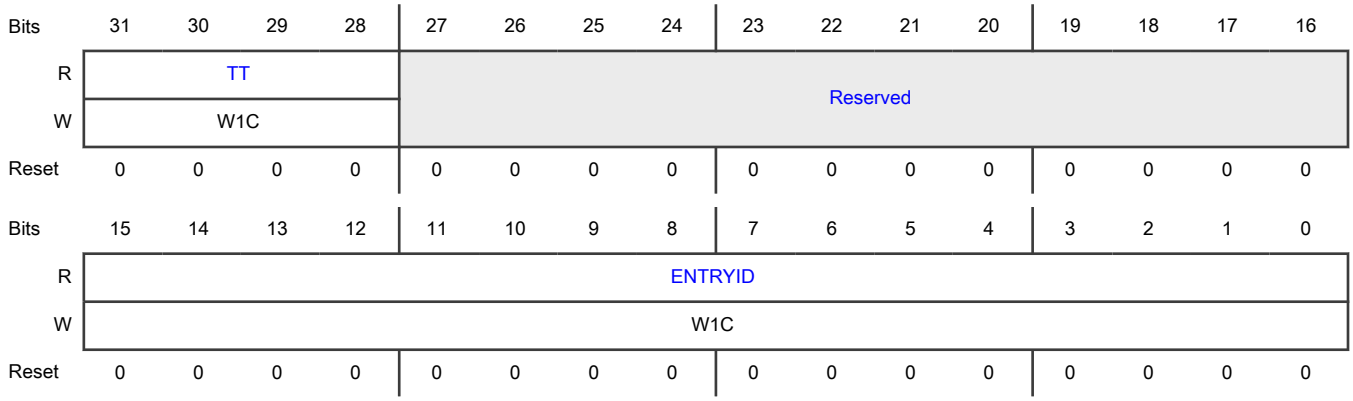
Instance	Register supported	Register not supported
ENETC0_PORT	—	PTXDCRR1
ENETC1_PORT	—	PTXDCRR1
SW0_PORT0	PTXDCRR1	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_PORT1	PTXDCRR1	—
SW0_PORT2	PTXDCRR1	—
SW0_PORT3	PTXDCRR1	—
SW0_PORT4	PTXDCRR1	—

Diagram



Fields

Field	Function
31-28 TT	Table type which was last accessed when frame was discarded. 0 None 1 Egress Treatment table 2 Sequence Recovery table 3 Egress Count table 4-15 Reserved Cleared when written to 1.
27-16 —	Reserved
15-0 ENTRYID	Entry Id who last incremented discard count. Field represents the Egress Treatment Entry ID if the TT field of this register is set to 1 or 2; otherwise field is not updated. Cleared when written to 1.

53.4.6.8.34 Port time gate scheduling traffic class 0 status register (PTGSTC0SR)

Offset

Register	Offset
PTGSTC0SR	200h

Function

The port time gate scheduling traffic class 0 status register controls traffic class specific parameters for the transmission gate.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															LH_STATE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 LH_STATE	Look-ahead gate state 0 Gate closed 1 Gate open **NOTE** IERB registers S*a*TGSLR / E*a*TGSLR[MIN_LOOKAHEAD] + the per port register PTGSATOR[ADV_TIME_OFFSET] specify the advance time of the gate state.
15-1 —	Reserved
0 —	Reserved

53.4.6.8.35 Port traffic class a transmit maximum SDU register (PTC0TMSDUR - PTC7TMSDUR)

Offset

For a = 0 to 7:

Register	Offset
PTCaTMSDUR	208h + (a × 20h)

Function

This the port traffic class transmit service data unit register. It sets the maximum Service Data Unit (SDU, aka MTU) which determines the largest frame which can be sent, in the associated traffic class, on this port. Frames that exceed the limit are discarded. This limit must not exceed 0x07D0 (2000 bytes). Default is 1536 bytes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved							SF_MAXS...	Reserved						SDU_TYPE	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MAXSDU															
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved
24 SF_MAXSDU_DIS	When cleared, the Tx Max SDU check is performed for Store and Forward frames. If the frame is greater than MAXSDU, it is discarded and counted in PTXDCR with discard reason of MSDUEDR. **NOTE** For Cut-Through frames, MAXSDU check is always performed.
23-18 —	Reserved
17-16 SDU_TYPE	Specifies the type of PDU/SDU (Protocol/Service Data Unit) whose length is being validated as seen on the link. 0: PPDU (Physical Layer PDU): Preamble, IFG, SFD along with MPDU. Not supported if cut-through frames are expected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1: MPDU: (MAC PDU): MAC Header, MSDU and FCS 2: MSDU (MAC SDU); MPDU minus 12B MAC Header and 4B FCS. Not supported if cut-through frames are expected. All other setting reserved. **NOTE** Overhead values are specified by register PTXSDUOR.
15-0 MAXSDU	Transmit Maximum SDU size in bytes (i.e: Max Frame size). When the frame is dequeued and the SDU size exceeds this value, the frame is discard and counted against Port Tx Discard Count Register (PTXDCR). **NOTE** For cut-through frames, this value is used as the presumptive maximum frame size.

53.4.6.8.36 Port transmit traffic class a credit based shaper register 0 (PTC0CBSR0 - PTC7CBSR0)

Offset

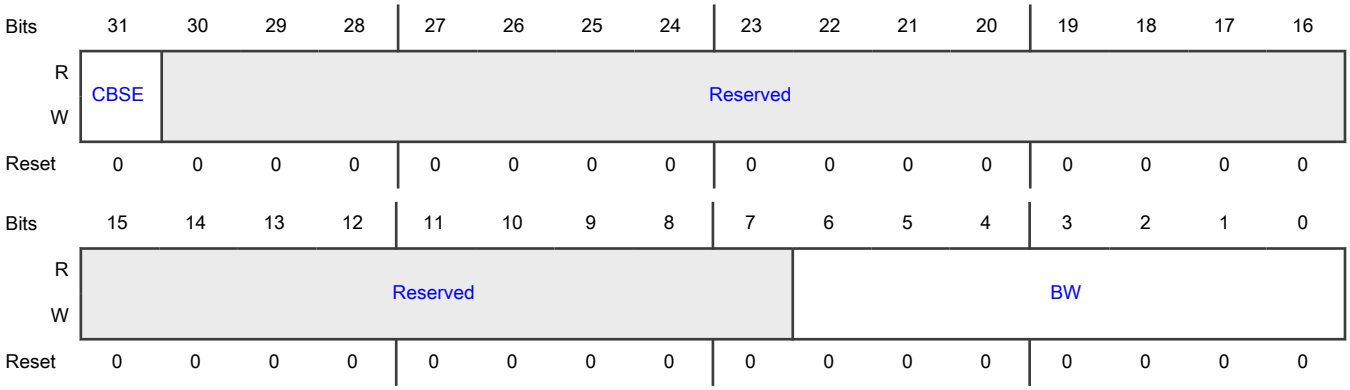
For a = 0 to 7:

Register	Offset
PTCaCBSR0	210h + (a × 20h)

Function

The port transmit traffic class credit based shaper register 0 is used to enable the credit-based shaper with the specification of the maximum bandwidth available to the traffic class.

Diagram



Fields

Field	Function
31 CBSE	Credit Based Shaper Enable Enables and disables the credit based shaper. 0b - Disabled 1b - Enabled
30-7 —	Reserved
6-0 BW	Bandwidth Bandwidth allocation is described in percentage units of the port transmit rate and ranging from 0-100 for the credit-based shaper. The sum of all traffic class credit-based shaper's bandwidth allocation cannot exceed 100. If there is an operational gate control list that is active, the bandwidth specified in this field only applies when the gate for the credit-based shaper traffic class, is open.

53.4.6.8.37 Port traffic class a credit based shaper register 1 (PTC0CBSR1 - PTC7CBSR1)

Offset

For a = 0 to 7:

Register	Offset
PTCaCBSR1	214h + (a × 20h)

Function

The port traffic class credit based shaper register 1 is used to specify the hiCredit parameter of the credit-based shaper.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HI_CREDIT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HI_CREDIT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 HI_CREDIT	hicredit (in credit units) The maximum allowed accumulation of credits when conflicting transfers occur. This value represents the bursting capability of a class and should be set based on the traffic class number. Number of credits per bit is calculated as follows (NETC ClockFrequency / portTransmitRate) * 100 For example: • One AVB class (credit-based shaper enabled); highest priority traffic class • Bandwidth allocated to AVB class (idleSlope) is 70% of the port transmit rate • Port transmit rate: 1 Gbit/s • Maximum sized Ethernet frame: 2000 bytes hiCredit (bits) = maxSizedFrame * (idleSlope/portTxRate) hiCredit (bits) = 16 000 * 0.7 hiCredit (bits) = 11 200 hiCredit (credits) = 11 200 * (400*10^6*100) / 1*10^9 hiCredit (credits) = 448 000

53.4.6.8.38 Port time gate scheduling traffic class 1 status register (PTGSTC1SR)

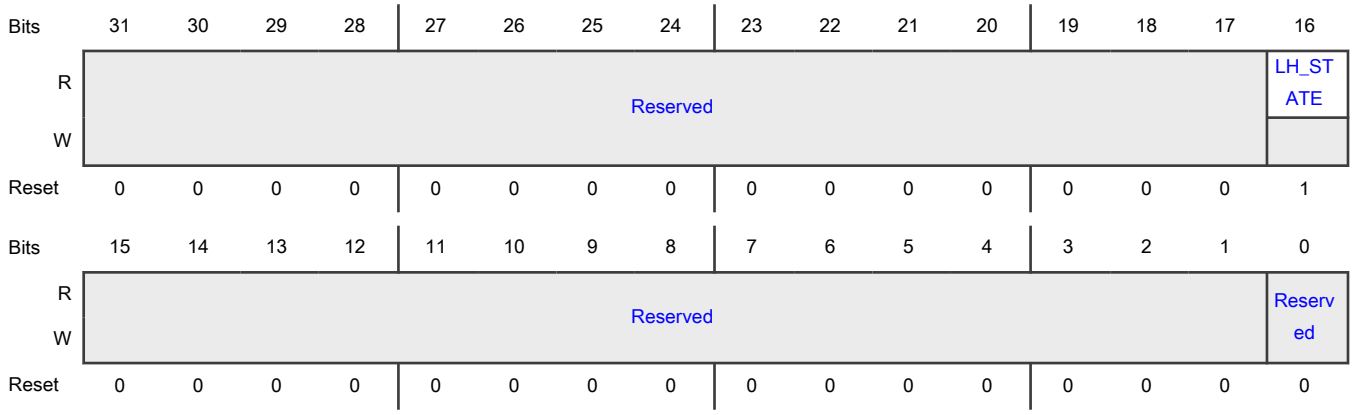
Offset

Register	Offset
PTGSTC1SR	220h

Function

The port time gate scheduling traffic class 1 status register controls traffic class specific parameters for the transmission gate.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 LH_STATE	Look-ahead gate state 0 Gate closed 1 Gate open **NOTE** IERB registers S*a*TGSLR / E*a*TGSLR[MIN_LOOKAHEAD] + the per port register PTGSATOR[ADV_TIME_OFFSET] specify the advance time of the gate state.
15-1 —	Reserved
0 —	Reserved

53.4.6.8.39 Port time gate scheduling traffic class 2 status register (PTGSTC2SR)

Offset

Register	Offset
PTGSTC2SR	240h

Function

The port time gate scheduling traffic class 2 status register controls traffic class specific parameters for the transmission gate.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															LH_STATE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 LH_STATE	Look-ahead gate state 0 Gate closed 1 Gate open **NOTE** IERB registers S*a*TGSLR / E*a*TGSLR[MIN_LOOKAHEAD] + the per port register PTGSATOR[ADV_TIME_OFFSET] specify the advance time of the gate state.
15-1 —	Reserved
0 —	Reserved

53.4.6.8.40 Port time gate scheduling traffic class 3 status register (PTGSTC3SR)

Offset

Register	Offset
PTGSTC3SR	260h

Function

The port time gate scheduling traffic class 3 status register controls traffic class specific parameters for the transmission gate.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															LH_STATE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 LH_STATE	Look-ahead gate state 0 Gate closed 1 Gate open **NOTE** IERB registers S*a*TGSLR / E*a*TGSLR[MIN_LOOKAHEAD] + the per port register PTGSATOR[ADV_TIME_OFFSET] specify the advance time of the gate state.
15-1 —	Reserved
0 —	Reserved

53.4.6.8.41 Port time gate scheduling traffic class 4 status register (PTGSTC4SR)

Offset

Register	Offset
PTGSTC4SR	280h

Function

The port time gate scheduling traffic class 4 status register controls traffic class specific parameters for the transmission gate.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															LH_STATE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 LH_STATE	Look-ahead gate state 0 Gate closed 1 Gate open **NOTE** IERB registers S*a*TGSLR / E*a*TGSLR[MIN_LOOKAHEAD] + the per port register PTGSATOR[ADV_TIME_OFFSET] specify the advance time of the gate state.
15-1 —	Reserved
0 —	Reserved

53.4.6.8.42 Port time gate scheduling traffic class 5 status register (PTGSTC5SR)

Offset

Register	Offset
PTGSTC5SR	2A0h

Function

The port time gate scheduling traffic class 5 status register controls traffic class specific parameters for the transmission gate.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															LH_STATE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 LH_STATE	Look-ahead gate state 0 Gate closed 1 Gate open **NOTE** IERB registers S*a*TGSLR / E*a*TGSLR[MIN_LOOKAHEAD] + the per port register PTGSATOR[ADV_TIME_OFFSET] specify the advance time of the gate state.
15-1 —	Reserved
0 —	Reserved

53.4.6.8.43 Port time gate scheduling traffic class 6 status register (PTGSTC6SR)

Offset

Register	Offset
PTGSTC6SR	2C0h

Function

The port time gate scheduling traffic class 6 status register controls traffic class specific parameters for the transmission gate.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															LH_STATE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 LH_STATE	Look-ahead gate state 0 Gate closed 1 Gate open **NOTE** IERB registers S*a*TGSLR / E*a*TGSLR[MIN_LOOKAHEAD] + the per port register PTGSATOR[ADV_TIME_OFFSET] specify the advance time of the gate state.
15-1 —	Reserved
0 —	Reserved

53.4.6.8.44 Port time gate scheduling traffic class 7 status register (PTGSTC7SR)

Offset

Register	Offset
PTGSTC7SR	2E0h

Function

The port time gate scheduling traffic class 7 status register controls traffic class specific parameters for the transmission gate.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															LH_STATE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 LH_STATE	Look-ahead gate state 0 Gate closed 1 Gate open **NOTE** IERB registers S*a*TGSLR / E*a*TGSLR[MIN_LOOKAHEAD] + the per port register PTGSATOR[ADV_TIME_OFFSET] specify the advance time of the gate state.
15-1 —	Reserved
0 —	Reserved

53.4.6.8.45 Port buffer pool mapping configuration register 0 (PBPMCR0)

Offset

Register	Offset
PBPMCR0	400h

Function

This is the (ingress) port buffer pool mapping configuration register 0.

NOTE
Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	PBPMCR0
ENETC1_PORT	—	PBPMCR0
SW0_PORT0	PBPMCR0	—
SW0_PORT1	PBPMCR0	—
SW0_PORT2	PBPMCR0	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_PORT3	PBPMCR0	—
SW0_PORT4	PBPMCR0	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IPV3_INDEX								IPV2_INDEX							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IPV1_INDEX								IPV0_INDEX							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24: IPV3_INDEX 23-16: IPV2_INDEX 15-8: IPV1_INDEX 7-0: IPV0_INDEX	Indicates its associated Buffer Pool table Entry ID. Range: 0..BPCAPR[NUM_BP]-1

53.4.6.8.46 Port buffer pool mapping configuration register 1 (PBPMCR1)

Offset

Register	Offset
PBPMCR1	404h

Function

This is the (ingress) port buffer pool mapping configuration register 1.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	PBPMCR1
ENETC1_PORT	—	PBPMCR1
SW0_PORT0	PBPMCR1	—
SW0_PORT1	PBPMCR1	—
SW0_PORT2	PBPMCR1	—
SW0_PORT3	PBPMCR1	—
SW0_PORT4	PBPMCR1	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IPV7_INDEX								IPV6_INDEX							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IPV5_INDEX								IPV4_INDEX							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24: IPV7_INDEX 23-16: IPV6_INDEX 15-8: IPV5_INDEX 7-0: IPV4_INDEX	Indicates its associated Buffer Pool table Entry ID. Range: 0..BPCAPR[NUM_BP]-1

53.4.6.8.47 Port PCP DEI mapping register (PPCPDEIMR)

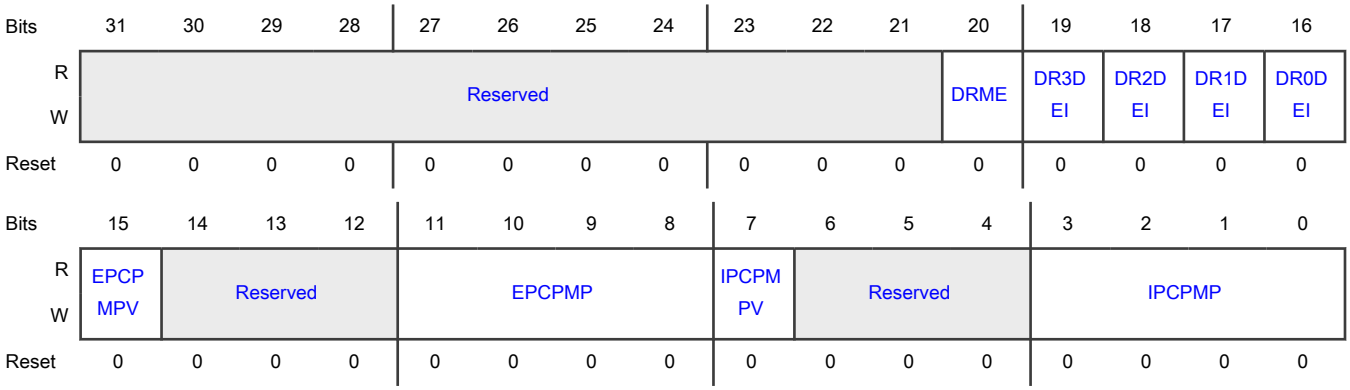
Offset

Register	Offset
PPCPDEIMR	438h

Function

This is the port PCP DEI mapping register

Diagram



Fields

Field	Function
31-21 —	Reserved
20 DRME	Indicates if mapping of internal QoS's DR value *d* to VLAN DEI is enabled. 0b - Preserve the DR value in the outer VLAN. 1b - Update DR value in the outer VLAN based on DEnDEI field.
19-16 DRnDEI	Mapping of internal QoS's DR value n to VLAN DEI.
15 EPCPMPV	Egress PCP to PCP Mapping Profile is valid. 0b - Egress PCP to PCP Mapping Profile is not valid. 1b - Egress frame modification of outer VLAN tag's PCP value is mapped to a new value based on EPCPMP instance.
14-12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-8 EPCPMP	Egress PCP to PCP Mapping Profile instance. This field represents *a* in PCP2PCPMPaR and is valid if EPCPMPV is equal to 1b.
7 IPCPMPV	Ingress PCP to PCP Mapping Profile is valid. **NOTE** Only applicable if outer VLAN tag is present. 0b - Ingress PCP to PCP Mapping Profile is not valid. 1b - Ingress frame modification of outer VLAN tag's PCP value is mapped to a new value based on IPCPMP instance
6-4 —	Reserved
3-0 IPCPMP	Ingress PCP to PCP Mapping Profile instance. This field represents *a* in PCP2PCPMP*a*R and is valid if IPCPMPV is equal to 1b.

53.4.6.8.48 Port mirror configuration register (PMCR)

Offset

Register	Offset
PMCR	440h

Function

This is the port mirror configuration register.

NOTE
Each module instance supports a different number of registers.

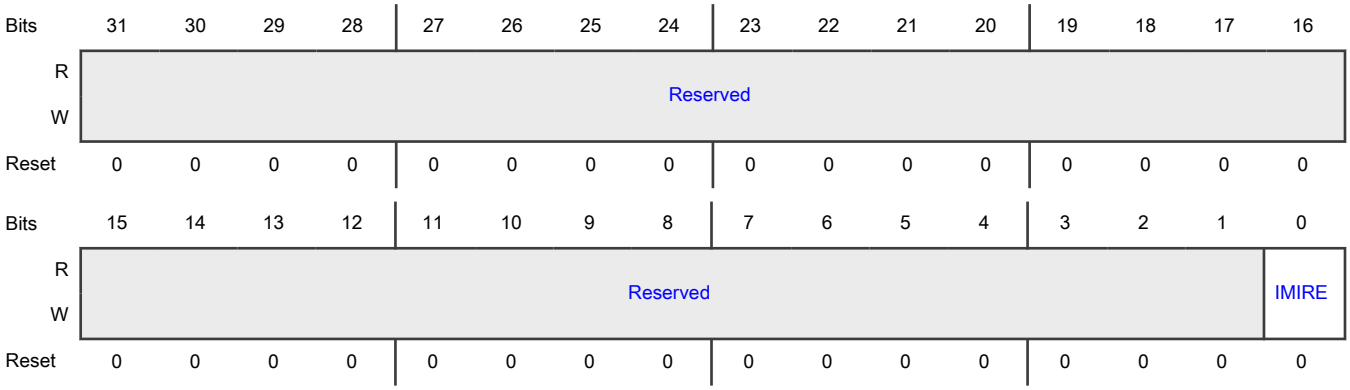
Instance	Register supported	Register not supported
ENETC0_PORT	—	PMCR
ENETC1_PORT	—	PMCR
SW0_PORT0	PMCR	—
SW0_PORT1	PMCR	—
SW0_PORT2	PMCR	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_PORT3	PMCR	—
SW0_PORT4	PMCR	—

Diagram



Fields

Field	Function
31-1 —	Reserved
0 IMIRE	Enable Ingress Mirroring on the port. When set, all ingress frames received on the port are replicated to the mirror destination specified in IMDCR0.

53.4.6.8.49 Port cut through forwarding configuration register (PCTFCR)

Offset

Register	Offset
PCTFCR	450h

Function

This is the port cut-through forwarding configuration register.

NOTE

If the port associated with a pseudo MAC (i.e: if PMCAPR[MAC_VAR]=0), cut-through is never permitted; a writes to this register is ignored.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	PCTFCR
ENETC1_PORT	—	PCTFCR
SW0_PORT0	PCTFCR	—
SW0_PORT1	PCTFCR	—
SW0_PORT2	PCTFCR	—
SW0_PORT3	PCTFCR	—
SW0_PORT4	—	PCTFCR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														ECTS	ICTS
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1

Fields

Field	Function
31-2 —	Reserved
1 ECTS	Egress Cut Through State 0 Cut-through frames are permitted on this port 1 Cut-through frames are not permitted on this port
0 ICTS	Ingress Cut Through State 0 Cut-through forwarding is allowed on this port 1 Cut-through forwarding is not permitted on this port

53.4.6.8.50 Port LANID configuration register (PLANIDCR)

Offset

Register	Offset
PLANIDCR	458h

Function

This is the port LANID configuration register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	PLANIDCR
ENETC1_PORT	—	PLANIDCR
SW0_PORT0	PLANIDCR	—
SW0_PORT1	PLANIDCR	—
SW0_PORT2	PLANIDCR	—
SW0_PORT3	PLANIDCR	—
SW0_PORT4	PLANIDCR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												LANID			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3-0 LANID	LAN Identifier This field represents the HSR's PathId this port belongs to. On transmit, if an HSR tag is added, this field is copied in the HSR tag's PathId field. **NOTE** Valid only on Ethernet ports.

53.4.6.8.51 Port ingress stream identification configuration register (PISIDCR)

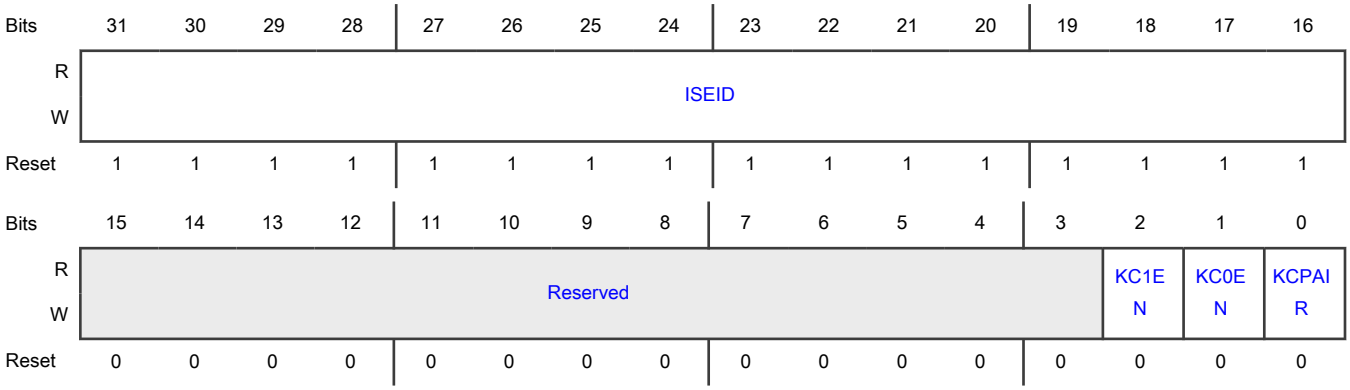
Offset

Register	Offset
PISIDCR	460h

Function

This is the port ingress stream identification configuration register.

Diagram



Fields

Field	Function
31-16 ISEID	Default Ingress Stream Entry ID. 0x0..ISITCAPR[NUM_ENTRIES]-1 0xFFFF_FFFF (NULL); stream filtering function is by-passed Other values are reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-3 —	Reserved
2 KC1EN	Key Construction 1 Enable 0 There is no exact match lookup performed for the second stream identification lookup. 1 An exact match lookup is performed for the second stream identification only if a lookup missed in the first stream identification stage. An exact match lookup is performed for the second stream identification regardless of whether or not there are entries in the table for the second stream identification. **NOTE** The precedence value of these entries are greater than an Ingress Port Filter table lookup match with a Relative Precedent Resolution value of 2.
1 KC0EN	Key Construction 0 Enable 0 There is no exact match lookup performed for the first stream identification lookup. 1 An exact match lookup is performed for the first stream identification regardless of whether or not there are entries in the table for the first stream identification. **NOTE** The precedence value of these entries are greater than an Ingress Port Filter table lookup match with a Relative Precedent Resolution value of 1 or 2.
0 KCPAIR	Indicates which Ingress Stream Identification Key Construction pair to use for this port. 0 Utilizes ISIDKC0CR0 and ISIDKC1CR0 1 Utilizes ISIDKC2CR0 and ISIDKC3CR0 (setting not applicable for ENETC)

53.4.6.8.52 Port frame modification configuration register (PFMCR)

Offset

Register	Offset
PFMCR	464h

Function

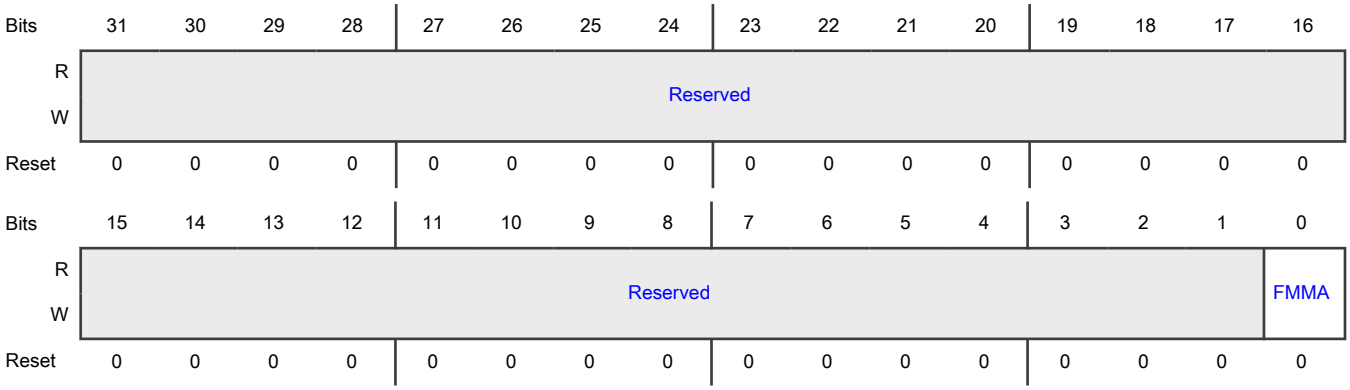
This is the port frame modification configuration register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	PFMCR
ENETC1_PORT	—	PFMCR
SW0_PORT0	PFMCR	—
SW0_PORT1	PFMCR	—
SW0_PORT2	PFMCR	—
SW0_PORT3	PFMCR	—
SW0_PORT4	PFMCR	—

Diagram



Fields

Field	Function
31-1 —	Reserved
0 FMMA	Frame Modification Misconfiguration Action For egress frame modifications, if a frame modification entry is configured such that the action requested cannot be performed, the following action is performed based on the value configured in this field: For ingress frame modifications, if a frame modification entry is configured such that the action requested cannot be performed, this will result in a misconfiguration event. This misconfiguration event is handled by discarding the frame and incrementing the port's Rx discard count register (PRXDCR) along with the setting of the Frame Modification Misconfiguration Error Discard Reason (FMMEDR) flag to 1 in the port's Rx discard count reason register 0 (PRXDCRR0). 0b - Discard the frame and counted against the port's Tx discard count register (PTXDCR) along with the setting of the Frame Modification Misconfiguration Discard Reason (FMMDR) flag to 1 in the port's Tx discard count reason register 0 (PTXDCRR0).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Transmit the frame without performing any of the ingress and egress frame modification actions specified.

53.4.6.8.53 Port IPV to queue mapping register 0 (PIPV2QMR0)

Offset

Register	Offset
PIPV2QMR0	470h

Function

This is the port IPV to queue mapping register 0.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	PIPV2QMR0
ENETC1_PORT	—	PIPV2QMR0
SW0_PORT0	PIPV2QMR0	—
SW0_PORT1	PIPV2QMR0	—
SW0_PORT2	PIPV2QMR0	—
SW0_PORT3	PIPV2QMR0	—
SW0_PORT4	PIPV2QMR0	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IPV7_Q				IPV6_Q				IPV5_Q				IPV4_Q			
Reset	0	1	1	1	0	1	1	0	0	1	0	1	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IPV3_Q				IPV2_Q				IPV1_Q				IPV0_Q			
Reset	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-28: IPV7_Q 27-24: IPV6_Q 23-20: IPV5_Q 19-16: IPV4_Q 15-12: IPV3_Q 11-8: IPV2_Q 7-4: IPV1_Q 3-0: IPV0_Q	Each IPV*i*_Q field is defined as follows (where i=IPV={0..7}): IPV - Internal Priority Value Q - Queue number

53.4.6.8.54 Port time capture minimum latency register (PTCMINLR)

Offset

Register	Offset
PTCMINLR	4B0h

Function

This is the port time capture minimum latency register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	PTCMINLR
ENETC1_PORT	—	PTCMINLR
SW0_PORT0	PTCMINLR	—
SW0_PORT1	PTCMINLR	—
SW0_PORT2	PTCMINLR	—
SW0_PORT3	PTCMINLR	—
SW0_PORT4	PTCMINLR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COUNT				LATENCY											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LATENCY															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 COUNT	Indicates the number of times a frame transmitted out of this port, has fulfilled the timestamp capture criteria. This field resets upon re-arming of the time capture feature. This field does not roll over.
29-0 LATENCY	Indicates the minimum latency between the Tx Timestamp and Rx Timestamp captured by the Ethernet MAC relative to SFD. The PTCMINLR and PTCMAXLR registers will hold the same latency value, as the time capture function can operate only in single arm mode (ARMed once).

53.4.6.8.55 Port time capture maximum latency register (PTCMAXLR)

Offset

Register	Offset
PTCMAXLR	4B4h

Function

This is the port time capture maximum latency register.

NOTE

Each module instance supports a different number of registers.

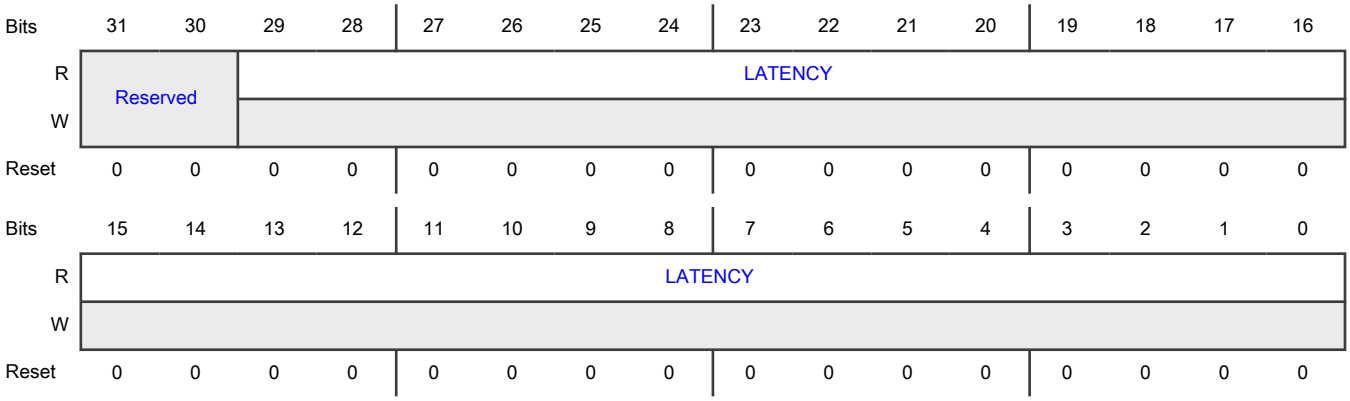
Instance	Register supported	Register not supported
ENETC0_PORT	—	PTCMAXLR
ENETC1_PORT	—	PTCMAXLR
SW0_PORT0	PTCMAXLR	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_PORT1	PTCMAXLR	—
SW0_PORT2	PTCMAXLR	—
SW0_PORT3	PTCMAXLR	—
SW0_PORT4	PTCMAXLR	—

Diagram



Fields

Field	Function
31-30 —	Reserved
29-0 LATENCY	Indicates the maximum latency between the Tx Timestamp and Rx Timestamp captured by the Ethernet MAC relative to SFD. The PTCMINLR and PTCMAXLR registers will hold the same latency value, as the time capture function can operate only in single arm mode (ARMed once).

53.4.6.8.56 Bridge port configuration register (BPCR)

Offset

Register	Offset
BPCR	500h

Function

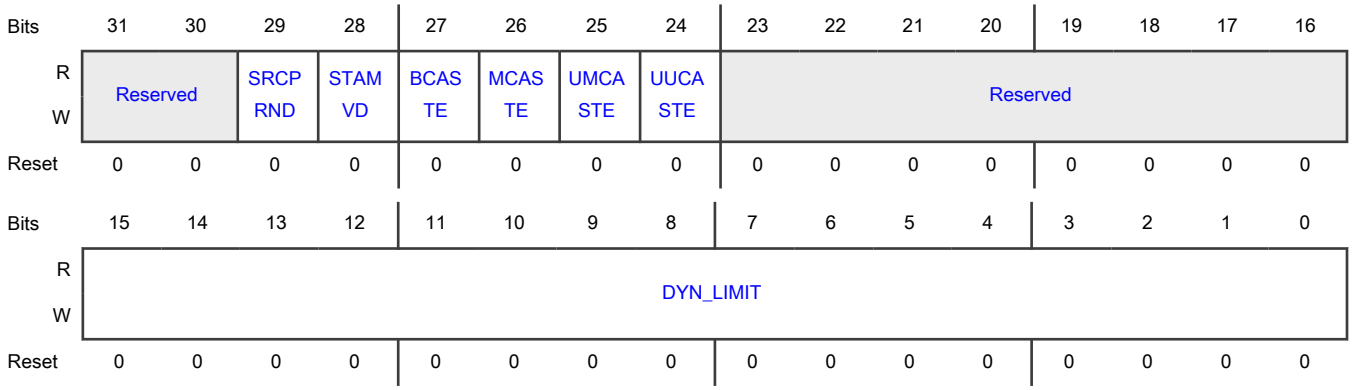
This is the bridge port configuration register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	BPCR
ENETC1_PORT	—	BPCR
SW0_PORT0	BPCR	—
SW0_PORT1	BPCR	—
SW0_PORT2	BPCR	—
SW0_PORT3	BPCR	—
SW0_PORT4	BPCR	—

Diagram



Fields

Field	Function
31-30 —	Reserved
29 SRCPRND	Source Port Pruning Disable Disables and enables source port pruning. Source port pruning prevents a frame from being transmitted on the same port it was received from. 0b - Enabled 1b - Disabled
28 STAMVD	Station Move Disallow

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Specifies whether a MAC station move is allowed. A MAC station move is an event detected by the MAC learning function where the source MAC address of a received frame is in the FDB table, but the port from which the frame was received is not set in its FDB entry Destination Port Bitmap. 0b - Allowed 1b - Disallowed. A received frame for which a MAC station move is detected, will be discarded.
27 BCASTE	Broadcast Storm Control Enable Enables and disables storm control for broadcast frames. The storm control rate policer used to limit broadcast frames is configured in BPSCR0. 0b - Disabled 1b - Enabled
26 MCASTE	Multicast Storm Control Enable Enables and disables storm control for multicast frames. The storm control rate policer used to limit multicast frames is configured in BPSCR1. 0b - Disabled 1b - Enabled
25 UMCASTE	Unknown Multicast Storm Control Enable Enables and disables storm control for unknown multicast frames. The storm control rate policer used to limit unknown multicast frames is configured in BPSCR1. 0b - Disabled 1b - Enabled
24 UUCASTE	Unknown Unicast Storm Control Enable Enables and disables storm control for unknown unicast frames. The storm control rate policer used to limit unknown unicast frames is configured in BPSCR0 0b - Disabled 1b - Enabled
23-16 —	Reserved
15-0 DYN_LIMIT	Dynamic Limit Specifies for a given port, the maximum number of dynamic entries in the FDB table. A value of 0 implies no limit except for maximum FDB dynamic entries allowed specified by FDBHTMCR[DYN_LIMIT]. Dynamic entries can be added by the hardware MAC learning function or by FDB table management command. In the case where the hardware MAC learning function cannot add a dynamic FDB entry due to this limit having been reached, the frame continues to be forwarded and BPMLFSR[BPMLLRFR] failure reason is set.

53.4.6.8.57 Bridge port default VLAN register (BPDVR)

Offset

Register	Offset
BPDVR	510h

Function

This is the bridge port VLAN register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	BPDVR
ENETC1_PORT	—	BPDVR
SW0_PORT0	BPDVR	—
SW0_PORT1	BPDVR	—
SW0_PORT2	BPDVR	—
SW0_PORT3	BPDVR	—
SW0_PORT4	BPDVR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved					TXTAGA		RXVAM	RXTAGA				Reserved			TPID
Reset	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PCP			DEI	VID											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-27	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
26-25 TXTAGA	Transmit Bridge Port VLAN Tag Action This field applies only for a VLAN aware bridge, where the frame outer VLAN tag's VID is equal to the port default VID (VID field in this register). Note that the frame outer VLAN tag, is after any ingress frame modification actions (e.g. adding a port default "internal" VLAN header) have been logically applied, it is not necessarily the VLAN tag from the received frame. 0 No egress VLAN modification performed 1 Delete outer VLAN tag 2 Replace outer VLAN tag's VID with 0; frame to be transmitted as a priority tag frame. 3 Reserved **NOTE** Only applies if there is no Egress Treatment entry configured or if the frame modification entry ID configured in the Egress Treatment table entry is set to null.
24 RXVAM	Receive VLAN Aware Mode All frames to be forwarded using the 802.1Q bridge forwarding function must be a member of a VLAN, either assigned from the frame VLAN header or through the port default configuration. The VID assignment is dependent on the value entering in this field: 0 VLAN Aware: If the frame is untagged, an internal VLAN header is added to the frame using the VID, DEI, PCP and TPID from this register. If the frame has a VLAN header with VID=0 (priority tag), that VLAN header is internally modified to replace its VID with the VID from this register. This internal VLAN header is carried along with the frame in the form of ingress frame modification metadata actions (to be used later on to modify the actual frame when egressing out a port). However the VID from this internal VLAN header is used immediately for the subsequent VLAN operations. If any ingress frame modification actions have already been specified, one must ensure that the resulting ingress frame modification operation outputs the frame with a VLAN header. The VID from this resulting VLAN header is used for the subsequent VLAN operations. If the resulting ingress frame modification operation outputs an untagged frame, the frame is discarded. If the frame is tagged and no ingress frame modification actions have been specified, the assigned VID is taken from the frame's outer VLAN header. 1 VLAN Unaware: The VID from this register is assigned to the frame regardless of whether the frame is tagged or untagged, and this VID is used for subsequent VLAN operations.
23-20 RXTAGA	Receive Bridge Port Tag Acceptance Criteria xxx0 Discard untagged xxx1 Accept untagged xx0x Discard priority tagged xx1x Accept priority tagged x0xx Discard single tagged x1xx Accept single tagged 0xxx Discard double tagged

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1xxx Accept double tagged Frames discarded are counted against the bridge port discard count register (BPDCR) along with the setting of the Discard due to Bridge Port Acceptance Criteria Discard Reason (BPACDR) flag to 1 in the Bridge port discard count reason register 0 (BPDCRR0).
19-17 —	Reserved
16 TPID	Specifies the TPID value to be used to construct the internal VLAN header. 0 Standard C-VLAN 0x8100 1 Standard S-VLAN 0x88A8
15-13 PCP	Priority code point This field specifies the 3-bit PCP to be used to construct the internal VLAN header.
12 DEI	Drop eligible indicator This field specifies the 1-bit DEI to be used to construct the internal VLAN header.
11-0 VID	VLAN identifier Specifies the VID value to be used to construct or modify the internal VLAN header.

53.4.6.8.58 Bridge port spanning tree group state register (BPSTGSR)

Offset

Register	Offset
BPSTGSR	520h

Function

This is the bridge port spanning tree group state register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	BPSTGSR
ENETC1_PORT	—	BPSTGSR
SW0_PORT0	BPSTGSR	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_PORT1	BPSTGSR	—
SW0_PORT2	BPSTGSR	—
SW0_PORT3	BPSTGSR	—
SW0_PORT4	BPSTGSR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STG_STATE15		STG_STATE14		STG_STATE13		STG_STATE12		STG_STATE11		STG_STATE10		STG_STATE9		STG_STATE8	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	STG_STATE7		STG_STATE6		STG_STATE5		STG_STATE4		STG_STATE3		STG_STATE2		STG_STATE1		STG_STATE0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-30: STG_STATE15 29-28: STG_STATE14 27-26: STG_STATE13 25-24: STG_STATE12 23-22: STG_STATE11 21-20: STG_STATE10 19-18: STG_STATE9 17-16: STG_STATE8	Each STG_STATEi field is defined as follows, where i = spanning tree protocol group = {0..15} Specifies the port spanning tree protocol state in spanning tree protocol group i {0..15}. The information in this register is used to perform spanning tree protocol (STP) port state checking during ingress VLAN filtering processing and egress VLAN filtering processing. In the case of ingress VLAN filtering processing, the source port ID and spanning tree protocol group ID are used to access register (BPSTGSR) to obtain the spanning tree protocol port state. 0 Disabled - Frames will be discarded. 1 Learning - Learn SMAC, but do not forward the frame. 2 Forwarding - Both MAC learning and forwarding functions are executed. 3 Reserved. In the case of egress VLAN filtering processing, for each port that has its bit set in the destination port bitmap, the corresponding destination port ID and spanning tree protocol group ID are used to access register (BPSTGSR) to obtain the spanning tree protocol port state. 0 Disabled - Frames will be discarded. 1 Learning - Frames will be discarded. 2 Forwarding - Frames will be forwarded.

Table continues on the next page...

Field	Function
15-14: STG_STATE7 13-12: STG_STATE6 11-10: STG_STATE5 9-8: STG_STATE4 7-6: STG_STATE3 5-4: STG_STATE2 3-2: STG_STATE1 1-0: STG_STATE0	3 Reserved.

53.4.6.8.59 Bridge port storm control register 0 (BPSCR0)

Offset

Register	Offset
BPSCR0	528h

Function

This is the bridge port storm control register 0.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	BPSCR0
ENETC1_PORT	—	BPSCR0
SW0_PORT0	BPSCR0	—
SW0_PORT1	BPSCR0	—
SW0_PORT2	BPSCR0	—
SW0_PORT3	BPSCR0	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_PORT4	BPSCR0	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				BCASTRPEID											
W																
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				UUCASTRPEID											
W																
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-28 —	Reserved
27-16 BCASTRPEID	Broadcast Rate Policer Entry ID. Valid if BPCR[BCASTE]=1. 0x0..RPITCAPR[NUM_ENTRIES]-1 Other values are reserved This field should be set to reference an entry that has been added to the Rate Policer table, before enabling the storm control policer (BPCR[BCASTE]=1).
15-12 —	Reserved
11-0 UUCASTRPEID	Unknown unicast Rate Policer Entry ID. Valid if BPCR[UUCASTE]=1. 0x0..RPITCAPR[NUM_ENTRIES]-1 Other values are reserved This field should be set to reference an entry that has been added to the Rate Policer table, before enabling the storm control policer (BPCR[UUCASTE]=1).

53.4.6.8.60 Bridge port storm control register 1 (BPSCR1)

Offset

Register	Offset
BPSCR1	52Ch

Function

This is the bridge port storm control register 1.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	BPSCR1
ENETC1_PORT	—	BPSCR1
SW0_PORT0	BPSCR1	—
SW0_PORT1	BPSCR1	—
SW0_PORT2	BPSCR1	—
SW0_PORT3	BPSCR1	—
SW0_PORT4	BPSCR1	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				UMCASTRPEID											
W																
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				MCASTRPEID											
W																
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-28	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
27-16 UMCASTRPEID	Unknown multicast Rate Policer Entry ID. Valid if BPCR[UMCASTE]=1. 0x0..RPITCAPR[NUM_ENTRIES]-1 Other values are reserved This field should be set to reference an entry that has been added to the Rate Policer table, before enabling the storm control policer (BPCR[UMCASTE]=1).
15-12 —	Reserved
11-0 MCASTRPEID	Known multicast Rate Policer Entry ID. Valid if BPCR[MCASTE]=1. 0x0..RPITCAPR[NUM_ENTRIES]-1 Other values are reserved This field should be set to reference an entry that has been added to the Rate Policer table, before enabling the storm control policer (BPCR[MCASTE]=1).

53.4.6.8.61 Bridge port operational register (BPOR)

Offset

Register	Offset
BPOR	530h

Function

This is the bridge port operational register.

NOTE

Each module instance supports a different number of registers.

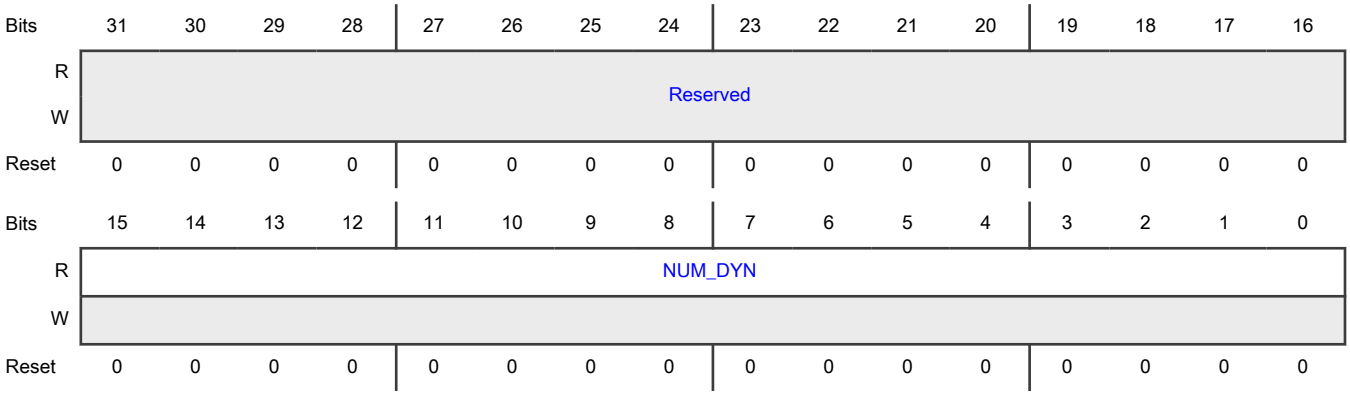
Instance	Register supported	Register not supported
ENETC0_PORT	—	BPOR
ENETC1_PORT	—	BPOR
SW0_PORT0	BPOR	—
SW0_PORT1	BPOR	—
SW0_PORT2	BPOR	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_PORT3	BPOR	—
SW0_PORT4	BPOR	—

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 NUM_DYN	Number of FDB entries dynamically learned against this port.

53.4.6.8.62 Bridge port discard count register (BPDCR)

Offset

Register	Offset
BPDCR	580h

Function

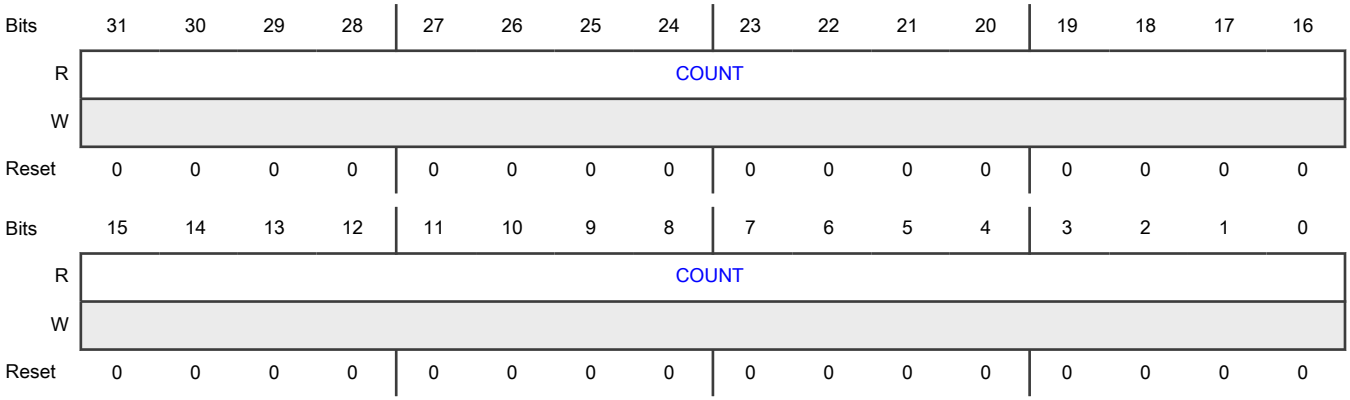
This is the bridge port discard count register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	BPDCR
ENETC1_PORT	—	BPDCR
SW0_PORT0	BPDCR	—
SW0_PORT1	BPDCR	—
SW0_PORT2	BPDCR	—
SW0_PORT3	BPDCR	—
SW0_PORT4	BPDCR	—

Diagram



Fields

Field	Function
31-0 COUNT	Frame discard count.

53.4.6.8.63 Bridge port discard count reason register 0 (BPDCRR0)

Offset

Register	Offset
BPDCRR0	588h

Function

This is the bridge port discard count reason register 0.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	BPDCRR0
ENETC1_PORT	—	BPDCRR0
SW0_PORT0	BPDCRR0	—
SW0_PORT1	BPDCRR0	—
SW0_PORT2	BPDCRR0	—
SW0_PORT3	BPDCRR0	—
SW0_PORT4	BPDCRR0	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		STRMCTR...	MISCDR	UFMMDR	IPMFDR	NODESTDR	MACFDDDR	STAMVDDR	Reserved			MACLNFDR	BPVFLTDR	ISTGSDR	BPACDR
W			W1C	W1C	W1C	W1C	W1C	W1C	W1C				W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved
13 STRMCTRLDR	Discard due to Storm Control Policer Discard Reason Rate policer discard/drop counters (per port (RXDCR), per stream (Ingress Stream Count table) and per rate policer instance (Rate Policer table)) are also incremented. Reason cleared when written to 1.
12 MISCDR	Misconfiguration Discard Reason. Frame may be discarded due to:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Storm control Rate Policer Entry ID specified is outside the Rate Policer table size. • Storm control Rate Policer Entry ID is not found in the Rate Policer table (entry has not been added to the table) Reason cleared when written to 1.
11 UFMMDR	Untagged Frame Modification Misconfiguration Discard Reason Frame was discarded due to bridge port configured as VLAN aware with a non-null Ingress Frame Modification Entry ID (IFM_EID) resulting in the frame containing no VLAN headers, or with an outer priority tag (VID=0). Reason cleared when written to 1.
10 IPMFDR	IP Multicast Filter Discard Reason Frame was discarded due to a miss in L2 IPV4 Multicast Filter table. Reason cleared when written to 1.
9 NODESTDR	No Destination Discard Reason L2 forwarding decision resulted in discarding the frame since it resulted in no destination. Following are the possible causes: • Forwarding entry (FDB or L2 IPV4 Multicast Filter) has a destination with null bitmap • Source port pruning where destination bitmap containing only the incoming port. • Spanning tree group output port state not set to 2 (forwarding), resulting in a null destination port bit map. Reason cleared when written to 1.
8 MACFDDDR	MAC Forwarding Default Discard Reason Occurs if MFO = 3 (FDB lookup is performed, and if there is no match, the frame is discarded), MFO is settable by BPCR and VLAN Filter Table Entry. Reason cleared when written to 1.
7 STAMVDDR	Station Move Disallowed Discard Reason Occurs if STAMVD=1 and Source MAC found in FDB with port not matching received port. STAMVD is settable by BPCR and Ingress Stream table entry. The Station Move Disallowed Discard reason has a highest precedence level than the Ingress STG State Discard reason. If both discard reasons are encountered for a given frame, the Station Move Disallowed Discard reason will be reported. Reason cleared when written to 1.
6-4 —	Reserved
3	MAC Learn Not Found Discard Reason

Table continues on the next page...

Table continued from the previous page...

Field	Function
MACLNFDR	Occurs if VFHTDECR2[MLO] or VLAN Filter Table Entry's MLO = 5 (Disable MAC learning with SMAC validation). Reason cleared when written to 1.
2 BPVFLTDR	VLAN Filter (Port VLAN Membership) Discard Reason Occurs when VID (after ingress frame modification if applicable) received is not a member of this port. Reason cleared when written to 1.
1 ISTGSDR	Ingress STG State Discard Reason Occurs if port's STG State is Disabled or Learning. Reason cleared when written to 1.
0 BPACDR	Discard due to Bridge Port Acceptance Criteria Discard Reason Frame was discarded because it failed the Bridge Port Tag Acceptance Criteria. See BPDVR[RXTAGA]. Reason cleared when written to 1.

53.4.6.8.64 Bridge port discard count reason register 1 (BPDCRR1)

Offset

Register	Offset
BPDCRR1	58Ch

Function

This is the bridge port discard count reason register 1.

NOTE

Each module instance supports a different number of registers.

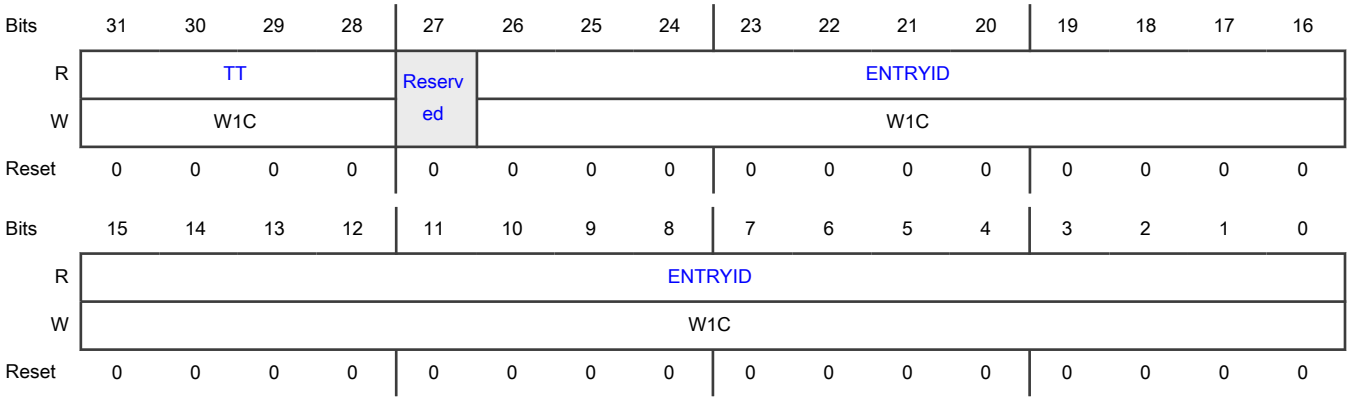
Instance	Register supported	Register not supported
ENETC0_PORT	—	BPDCRR1
ENETC1_PORT	—	BPDCRR1
SW0_PORT0	BPDCRR1	—
SW0_PORT1	BPDCRR1	—
SW0_PORT2	BPDCRR1	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_PORT3	BPDCRR1	—
SW0_PORT4	BPDCRR1	—

Diagram



Fields

Field	Function
31-28 TT	Table type which caused the discard. 0 None 1 VLAN Filter table 2 FDB table 3 L2 IPV4 Multicast Filter table - Any Source Multicast (ASM) 4 L2 IPV4 Multicast Filter table - Source Specific Multicast (SSM) 5 Rate Policer table due to storm control 6-15 Reserved Reason cleared when written to 1.
27 —	Reserved
26-0 ENTRYID	Entry ID who last incremented Discard Count. Field represents the entry ID when the frames is discarded for any of the following reasons: BPVFLTDR, STAMVDDR, STRMCTRLDR, MISCDR. Reason cleared when written to 1.

53.4.6.8.65 Bridge port MAC learning failure status register (BPMLFSR)

Offset

Register	Offset
BPMLFSR	590h

Function

This is the bridge port MAC learning failure status register. A MAC learning failure is not reported if the frame is discarded on ingress.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_PORT	—	BPMLFSR
ENETC1_PORT	—	BPMLFSR
SW0_PORT0	BPMLFSR	—
SW0_PORT1	BPMLFSR	—
SW0_PORT2	BPMLFSR	—
SW0_PORT3	BPMLFSR	—
SW0_PORT4	BPMLFSR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved													HCCLRFR	FFDBTRFR	BPMLLRFR
W														W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-3 —	Reserved
2 HCCLRFR	Hash Collision chain limit Reached Failure Reason An entry cannot be added because the hash, collision chain has reached the limit specified in HBTCAPR[MAX_COL]. Reason cleared when written to 1.
1 FFDBTRFR	Full FDB Table Reached Failure Reason A dynamically learned FDB entry cannot be added because either: • Hash table memory is full or • Number of dynamic FDB entries (FDBHTOR1[DYN_ENTRIES]) has reached its limit as specified by FDBHTMCR[DYN_LIMIT]. Reason cleared when written to 1.
0 BPMLLRFR	Bridge Port MAC Learn Limit Reached Failure Reason A dynamically learned FDB entry cannot be added because the number of dynamic FDB entries on this port (BPOR[NUM_DYN]) has reached its limit as specified by BPCR[DYN_LIMIT]. Reason cleared when written to 1.

53.4.6.9 Ethernet MAC port register descriptions

This section describes the Ethernet MAC port registers associated with a physical Ethernet link.

53.4.6.9.1 Ethernet_MAC_port memory map

Instance	Address space	Offset
ENETC0_ETH_MAC_PORT	PCI_F3_BAR_0	60B1_5000h
SW0_ETH_MAC_PORT0	PCI_F2_BAR_0	60A0_5000h
SW0_ETH_MAC_PORT1	PCI_F2_BAR_0	60A0_9000h
SW0_ETH_MAC_PORT2	PCI_F2_BAR_0	60A0_D000h
SW0_ETH_MAC_PORT3	PCI_F2_BAR_0	60A1_1000h

Offset	Register	Width (In bits)	Access	Reset value
8h	Port MAC 0 Command and Configuration Register (PM0_COMMAND_CONFIG)	32	RW	0000_8000h
Ch	Port MAC 0 MAC Address Register 0 (PM0_MAC_ADDR_0)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10h	Port MAC 0 MAC Address Register 1 (PM0_MAC_ADDR_1)	32	R	0000_0000h
14h	Port MAC 0 Maximum Frame Length Register (PM0_MAXFRM)	32	RW	0000_0600h
18h	Port MAC 0 Minimum Frame Length Register (PM0_MINFRM)	32	RW	0000_0040h
30h	Port MAC 0 Internal MDIO Configuration Register (PM0_MDIO_CFG)	32	RW	0000_1448h
34h	Port MAC 0 Internal MDIO Interface Control Register (PM0_MDIO_CTL)	32	RW	0000_0000h
38h	Port MAC 0 Internal MDIO Interface Data Register (PM0_MDIO_DATA)	32	RW	0000_0000h
40h	Port MAC 0 Interrupt Event Register (PM0_IEVENT)	32	RW	0000_0060h
44h	Port MAC 0 Transmit Inter-Packet Gap Length and Flexible Preamble length Register (PM0_TX_IPG_PREAMBLE)	32	RW	0000_070Ch
4Ch	Port MAC 0 Interrupt Mask Register(INT_MASK) (PM0_IMASK)	32	RW	0000_0000h
54h	Port MAC 0 Pause Quanta Register (PM0_PAUSE_QUANTA)	32	RW	0000_0000h
64h	Port MAC 0 Pause Quanta Threshold Register (PM0_PAUSE_THRESH)	32	RW	0000_0000h
74h	Port MAC 0 Receive Pause Status Register (PM0_RX_PAUSE_STATUS)	32	R	0000_0000h
B8h	Port MAC 0 EEE Low Power Wakeup Timer Register (PM0_LPWAKE_TIMER)	32	RW	0000_2000h
BCh	Port MAC 0 Transmit EEE Low Power Timer Register (PM0_SLEEP_TIMER)	32	RW	0000_0000h
C0h	Port MAC 0 IEEE1588 Single-Step Control Register (PM0_SINGLE_STEP)	32	RW	0000_0000h
D0h	Port MAC 0 half-duplex backoff entropy register (PM0_HD_BACKOFF_ENTROPY)	32	RW	0000_0000h
D4h	Port MAC 0 Half-Duplex Flow Control Register (PM0_HD_FLOW_CTRL)	32	RW	0BDC_0014h
E0h	Port MAC 0 Statistics Configuration Register (PM0_STATN_CONFIG)	32	RW	0000_0000h
100h	Port MAC 0 Receive Ethernet Octets Counter(etherStatsOctetsn) (PM0_REOCTn)	64	R	0000_0000_0000_0000h
108h	Port MAC 0 Receive Octets Counter(ifInOctetsn) (PM0_ROCTn)	64	R	0000_0000_0000_0000h
118h	Port MAC 0 Receive Valid Pause Frame Counter Register(aPAUSEMACCtrlFramesReceivedn) (PM0_RXPFn)	64	R	0000_0000_0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
120h	Port MAC 0 Receive Frame Counter Register(aFramesReceivedOKn) (PM0_RFRMn)	64	R	0000_0000_0000_0000h
128h	Port MAC 0 Receive Frame Check Sequence Error Counter Register() (PM0_RFCSn)	64	R	0000_0000_0000_0000h
130h	Port MAC 0 Receive VLAN Frame Counter Register(VLANReceivedOKn) (PM0_RVLANn)	64	R	0000_0000_0000_0000h
138h	Port MAC 0 Receive Frame Error Counter Register(ifInErrorsn) (PM0_RERRn)	64	R	0000_0000_0000_0000h
140h	Port MAC 0 Receive Unicast Frame Counter Register(ifInUcastPktsn) (PM0_RUCAn)	64	R	0000_0000_0000_0000h
148h	Port MAC 0 Receive Multicast Frame Counter Register(ifInMulticastPktsn) (PM0_RMCAn)	64	R	0000_0000_0000_0000h
150h	Port MAC 0 Receive Broadcast Frame Counter Register(ifInBroadcastPktsn) (PM0_RBCAn)	64	R	0000_0000_0000_0000h
158h	Port MAC 0 Receive Dropped Packets Counter Register(etherStatsDropEventsn) (PM0_RDRPn)	64	R	0000_0000_0000_0000h
160h	Port MAC 0 Receive Packets Counter Register(etherStatsPktsn) (PM0_RPKTn)	64	R	0000_0000_0000_0000h
168h	Port MAC 0 Receive Undersized Packet Counter Register(etherStatsUndersizePktsn) (PM0_RUNDn)	64	R	0000_0000_0000_0000h
170h	Port MAC 0 Receive 64-Octet Packet Counter Register(etherStatsPkts64OctetsN) (PM0_R64n)	64	R	0000_0000_0000_0000h
178h	Port MAC 0 Receive 65 to 127-Octet Packet Counter Register(etherStatsPkts65to127OctetsN) (PM0_R127n)	64	R	0000_0000_0000_0000h
180h	Port MAC 0 Receive 128 to 255-Octet Packet Counter Register(etherStatsPkts128to255OctetsN) (PM0_R255n)	64	R	0000_0000_0000_0000h
188h	Port MAC 0 Receive 256 to 511-Octet Packet Counter Register(etherStatsPkts256to511OctetsN) (PM0_R511n)	64	R	0000_0000_0000_0000h
190h	Port MAC 0 Receive 512 to 1023-Octet Packet Counter Register(etherStatsPkts512to1023OctetsN) (PM0_R1023n)	64	R	0000_0000_0000_0000h
198h	Port MAC 0 Receive 1024 to 1522-Octet Packet Counter Register(etherStatsPkts1024to1522OctetsN) (PM0_R1522n)	64	R	0000_0000_0000_0000h
1A0h	Port MAC 0 Receive 1523 to Max-Octet Packet Counter Register(etherStatsPkts1523toMaxOctetsN) (PM0_R1523Xn)	64	R	0000_0000_0000_0000h
1A8h	Port MAC 0 Receive Oversized Packet Counter Register(etherStatsOversizePktsn) (PM0_ROVRn)	64	R	0000_0000_0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1B0h	Port MAC 0 Receive Jabber Packet Counter Register(etherStatsJabbersn) (PM0_RJBRn)	64	R	0000_0000_0000_0000h
1B8h	Port MAC 0 Receive Fragment Packet Counter Register(etherStatsFragmentsn (PM0_RFRGn)	64	R	0000_0000_0000_0000h
1C0h	Port MAC 0 Receive Control Packet Counter Register (PM0_RCNPn)	64	R	0000_0000_0000_0000h
1C8h	Port MAC 0 Receive Dropped Not Truncated Packets Counter Register(etherStatsDropEventsn) (PM0_RDRNTPn)	64	R	0000_0000_0000_0000h
1D0h	Port MAC 0 Receive Valid Small Packet Counter Register (PM0_RMIN63n)	64	R	0000_0000_0000_0000h
200h	Port MAC 0 Transmit Ethernet Octets Counter(etherStatsOctetsn) (PM0_TEOCTn)	64	R	0000_0000_0000_0000h
208h	Port MAC 0 Transmit Octets Counter Register(ifOutOctetsn) (PM0_TOCTn)	64	R	0000_0000_0000_0000h
218h	Port MAC 0 Transmit Valid Pause Frame Counter Register(aPAUSEMACCtrlFramesReceivedn) (PM0_TXPFn)	64	R	0000_0000_0000_0000h
220h	Port MAC 0 Transmit Frame Counter Register(aFramesTransmittedOKn) (PM0_TFRMn)	64	R	0000_0000_0000_0000h
228h	Port MAC 0 Transmit Frame Check Sequence Error Counter Register() (PM0_TFCSn)	64	R	0000_0000_0000_0000h
230h	Port MAC 0 Transmit VLAN Frame Counter Register(VLANTransmittedOKn) (PM0_TVLANn)	64	R	0000_0000_0000_0000h
238h	Port MAC 0 Transmit Frame Error Counter Register(ifOutErrorsn) (PM0_TERRn)	64	R	0000_0000_0000_0000h
240h	Port MAC 0 Transmit Unicast Frame Counter Register(ifOutUcastPktsn) (PM0_TUCAn)	64	R	0000_0000_0000_0000h
248h	Port MAC 0 Transmit Multicast Frame Counter Register(ifOutMulticastPktsn) (PM0_TMCAn)	64	R	0000_0000_0000_0000h
250h	Port MAC 0 Transmit Broadcast Frame Counter Register(ifOutBroadcastPktsn) (PM0_TBCAn)	64	R	0000_0000_0000_0000h
260h	Port MAC 0 Transmit Packets Counter Register(etherStatsPktsn) (PM0_TPKTn)	64	R	0000_0000_0000_0000h
268h	Port MAC 0 Transmit Undersized Packet Counter Register(etherStatsUndersizePktsn) (PM0_TUNDn)	64	R	0000_0000_0000_0000h
270h	Port MAC 0 Transmit 64-Octet Packet Counter Register (etherStatsPkts64OctetsN) (PM0_T64n)	64	R	0000_0000_0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
278h	Port MAC 0 Transmit 65 to 127-Octet Packet Counter Register (etherStatsPkts65to127OctetsN) (PM0_T127n)	64	R	0000_0000_0000_0000h
280h	Port MAC 0 Transmit 128 to 255-Octet Packet Counter Register (etherStatsPkts128to255OctetsN) (PM0_T255n)	64	R	0000_0000_0000_0000h
288h	Port MAC 0 Transmit 256 to 511-Octet Packet Counter Register (etherStatsPkts256to511OctetsN) (PM0_T511n)	64	R	0000_0000_0000_0000h
290h	Port MAC 0 Transmit 512 to 1023-Octet Packet Counter Register (etherStatsPkts512to1023OctetsN) (PM0_T1023n)	64	R	0000_0000_0000_0000h
298h	Port MAC 0 Transmit 1024 to 1522-Octet Packet Counter Register (etherStatsPkts1024to1522OctetsN) (PM0_T1522n)	64	R	0000_0000_0000_0000h
2A0h	Port MAC 0 Transmit 1523 to TX_MTU-Octet Packet Counter Register (etherStatsPkts1523toMaxOctetsN) (PM0_T1523Xn)	64	R	0000_0000_0000_0000h
2C0h	Port MAC 0 Transmit Control Packet Counter Register (PM0_TCNPn)	64	R	0000_0000_0000_0000h
2D0h	Port MAC 0 Transmit Deferred Packet Counter Register(aFramesWithDeferredXmissions) (PM0_TDFRn)	64	R	0000_0000_0000_0000h
2D8h	Port MAC 0 Transmit Multiple Collisions Counter Register(aMultipleCollisionFrames) (PM0_TMCOLn)	64	R	0000_0000_0000_0000h
2E0h	Port MAC 0 Transmit Single Collision Counter(aSingleCollisionFrames) Register (PM0_TSCOLn)	64	R	0000_0000_0000_0000h
2E8h	Port MAC 0 Transmit Late Collision Counter(aLateCollisions) Register (PM0_TLCOLn)	64	R	0000_0000_0000_0000h
2F0h	Port MAC 0 Transmit Excessive Collisions Counter Register (PM0_TECOLn)	64	R	0000_0000_0000_0000h
300h	Port MAC 0 Interface Mode Control Register (PM0_IF_MODE)	32	RW	0000_4004h
408h	Port MAC 1 Command and Configuration Register (PM1_COMMAND_CONFIG)	32	RW	0000_8000h
40Ch	Port MAC 1 MAC Address Register 0 (PM1_MAC_ADDR_0)	32	R	0000_0000h
410h	Port MAC 1 MAC Address Register 1 (PM1_MAC_ADDR_1)	32	R	0000_0000h
414h	Port MAC 1 Maximum Frame Length Register (PM1_MAXFRM)	32	RW	0000_0600h
418h	Port MAC 1 Minimum Frame Length Register (PM1_MINFRM)	32	RW	0000_0040h
440h	Port MAC 1 Interrupt Event Register (PM1_IEVENT)	32	RW	0000_0060h
444h	Port MAC 1 Transmit Inter-Packet Gap Length and Flexible Preamble length Register (PM1_TX_IPG_PREAMBLE)	32	RW	0000_070Ch
44Ch	Port MAC 1 Interrupt Mask Register(INT_MASK) (PM1_IMASK)	32	RW	0000_0000h
454h	Port MAC 1 Pause Quanta Register (PM1_PAUSE_QUANTA)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
464h	Port MAC 1 Pause Quanta Threshold Register (PM1_PAUSE_THRESH)	32	RW	0000_0000h
474h	Port MAC 1 Receive Pause Status Register (PM1_RX_PAUSE_STATUS)	32	R	0000_0000h
4B8h	Port MAC 1 EEE Low Power Wakeup Timer Register (PM1_LPWAKE_TIMER)	32	RW	0000_2000h
4BCh	Port MAC 1 Transmit EEE Low Power Timer Register (PM1_SLEEP_TIMER)	32	RW	0000_0000h
4C0h	Port MAC 1 IEEE1588 Single-Step Control Register (PM1_SINGLE_STEP)	32	RW	0000_0000h
4D0h	Port MAC 1 half-duplex backoff entropy register (PM1_HD_BACKOFF_ENTROPY)	32	RW	0000_0000h
4D4h	Port MAC 1 Half-Duplex Flow Control Register (PM1_HD_FLOW_CTRL)	32	RW	0BDC_0014h
4E0h	Port MAC 1 Statistics Configuration Register (PM1_STATN_CONFIG)	32	RW	0000_0000h
500h	Port MAC 1 Receive Ethernet Octets Counter(etherStatsOctetsn) (PM1_REOCTn)	64	R	0000_0000_0000_0000h
508h	Port MAC 1 Receive Octets Counter(ifInOctetsn) (PM1_ROCTn)	64	R	0000_0000_0000_0000h
518h	Port MAC 1 Receive Valid Pause Frame Counter Register(aPAUSEMACCtrlFramesReceivedn) (PM1_RXPFn)	64	R	0000_0000_0000_0000h
520h	Port MAC 1 Receive Frame Counter Register(aFramesReceivedOKn) (PM1_RFRMn)	64	R	0000_0000_0000_0000h
528h	Port MAC 1 Receive Frame Check Sequence Error Counter Register() (PM1_RFCSn)	64	R	0000_0000_0000_0000h
530h	Port MAC 1 Receive VLAN Frame Counter Register(VLANReceivedOKn) (PM1_RVLANn)	64	R	0000_0000_0000_0000h
538h	Port MAC 1 Receive Frame Error Counter Register(ifInErrorsn) (PM1_RERRn)	64	R	0000_0000_0000_0000h
540h	Port MAC 1 Receive Unicast Frame Counter Register(ifInUcastPktsn) (PM1_RUCAn)	64	R	0000_0000_0000_0000h
548h	Port MAC 1 Receive Multicast Frame Counter Register(ifInMulticastPktsn) (PM1_RMCAn)	64	R	0000_0000_0000_0000h
550h	Port MAC 1 Receive Broadcast Frame Counter Register(ifInBroadcastPktsn) (PM1_RBCAn)	64	R	0000_0000_0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
558h	Port MAC 1 Receive Dropped Packets Counter Register(etherStatsDropEventsn) (PM1_RDRPn)	64	R	0000_0000_0000_0000h
560h	Port MAC 1 Receive Packets Counter Register(etherStatsPktsn) (PM1_RPKTn)	64	R	0000_0000_0000_0000h
568h	Port MAC 1 Receive Undersized Packet Counter Register(etherStatsUndersizePktsn) (PM1_RUNDn)	64	R	0000_0000_0000_0000h
570h	Port MAC 1 Receive 64-Octet Packet Counter Register(etherStatsPkts64OctetsN) (PM1_R64n)	64	R	0000_0000_0000_0000h
578h	Port MAC 1 Receive 65 to 127-Octet Packet Counter Register(etherStatsPkts65to127OctetsN) (PM1_R127n)	64	R	0000_0000_0000_0000h
580h	Port MAC 1 Receive 128 to 255-Octet Packet Counter Register(etherStatsPkts128to255OctetsN) (PM1_R255n)	64	R	0000_0000_0000_0000h
588h	Port MAC 1 Receive 256 to 511-Octet Packet Counter Register(etherStatsPkts256to511OctetsN) (PM1_R511n)	64	R	0000_0000_0000_0000h
590h	Port MAC 1 Receive 512 to 1023-Octet Packet Counter Register(etherStatsPkts512to1023OctetsN) (PM1_R1023n)	64	R	0000_0000_0000_0000h
598h	Port MAC 1 Receive 1024 to 1522-Octet Packet Counter Register(etherStatsPkts1024to1522OctetsN) (PM1_R1522n)	64	R	0000_0000_0000_0000h
5A0h	Port MAC 1 Receive 1523 to Max-Octet Packet Counter Register(etherStatsPkts1523toMaxOctetsN) (PM1_R1523Xn)	64	R	0000_0000_0000_0000h
5A8h	Port MAC 1 Receive Oversized Packet Counter Register(etherStatsOversizePktsn) (PM1_ROVRn)	64	R	0000_0000_0000_0000h
5B0h	Port MAC 1 Receive Jabber Packet Counter Register(etherStatsJabbersn) (PM1_RJBRn)	64	R	0000_0000_0000_0000h
5B8h	Port MAC 1 Receive Fragment Packet Counter Register(etherStatsFragmentsn (PM1_RFRGn)	64	R	0000_0000_0000_0000h
5C0h	Port MAC 1 Receive Control Packet Counter Register (PM1_RCNPn)	64	R	0000_0000_0000_0000h
5C8h	Port MAC 1 Receive Dropped Not Truncated Packets Counter Register(etherStatsDropEventsn) (PM1_RDRNTPn)	64	R	0000_0000_0000_0000h
5D0h	Port MAC 1 Receive Valid Small Packet Counter Register (PM1_RMIN63n)	64	R	0000_0000_0000_0000h
600h	Port MAC 1 Transmit Ethernet Octets Counter(etherStatsOctetsn) (PM1_TEOCTn)	64	R	0000_0000_0000_0000h
608h	Port MAC 1 Transmit Octets Counter Register(ifOutOctetsn) (PM1_TOCTn)	64	R	0000_0000_0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
618h	Port MAC 1 Transmit Valid Pause Frame Counter Register(aPAUSEMACCtrlFramesReceivedn) (PM1_TXPFn)	64	R	0000_0000_0000_0000h
620h	Port MAC 1 Transmit Frame Counter Register(aFramesTransmittedOKn) (PM1_TFRMn)	64	R	0000_0000_0000_0000h
628h	Port MAC 1 Transmit Frame Check Sequence Error Counter Register() (PM1_TFCSn)	64	R	0000_0000_0000_0000h
630h	Port MAC 1 Transmit VLAN Frame Counter Register(VLANTransmittedOKn) (PM1_TVLANn)	64	R	0000_0000_0000_0000h
638h	Port MAC 1 Transmit Frame Error Counter Register(ifOutErrorsn) (PM1_TERRn)	64	R	0000_0000_0000_0000h
640h	Port MAC 1 Transmit Unicast Frame Counter Register(ifOutUcastPktsn) (PM1_TUCAn)	64	R	0000_0000_0000_0000h
648h	Port MAC 1 Transmit Multicast Frame Counter Register(ifOutMulticastPktsn) (PM1_TMCAn)	64	R	0000_0000_0000_0000h
650h	Port MAC 1 Transmit Broadcast Frame Counter Register(ifOutBroadcastPktsn) (PM1_TBCAn)	64	R	0000_0000_0000_0000h
660h	Port MAC 1 Transmit Packets Counter Register(etherStatsPktsn) (PM1_TPKTn)	64	R	0000_0000_0000_0000h
668h	Port MAC 1 Transmit Undersized Packet Counter Register(etherStatsUndersizePktsn) (PM1_TUNDn)	64	R	0000_0000_0000_0000h
670h	Port MAC 1 Transmit 64-Octet Packet Counter Register (etherStatsPkts64OctetsN) (PM1_T64n)	64	R	0000_0000_0000_0000h
678h	Port MAC 1 Transmit 65 to 127-Octet Packet Counter Register (etherStatsPkts65to127OctetsN) (PM1_T127n)	64	R	0000_0000_0000_0000h
680h	Port MAC 1 Transmit 128 to 255-Octet Packet Counter Register (etherStatsPkts128to255OctetsN) (PM1_T255n)	64	R	0000_0000_0000_0000h
688h	Port MAC 1 Transmit 256 to 511-Octet Packet Counter Register (etherStatsPkts256to511OctetsN) (PM1_T511n)	64	R	0000_0000_0000_0000h
690h	Port MAC 1 Transmit 512 to 1023-Octet Packet Counter Register (etherStatsPkts512to1023OctetsN) (PM1_T1023n)	64	R	0000_0000_0000_0000h
698h	Port MAC 1 Transmit 1024 to 1522-Octet Packet Counter Register (etherStatsPkts1024to1522OctetsN) (PM1_T1522n)	64	R	0000_0000_0000_0000h
6A0h	Port MAC 1 Transmit 1523 to TX_MTU-Octet Packet Counter Register (etherStatsPkts1523toMaxOctetsN) (PM1_T1523Xn)	64	R	0000_0000_0000_0000h
6C0h	Port MAC 1 Transmit Control Packet Counter Register (PM1_TCNPn)	64	R	0000_0000_0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
6D0h	Port MAC 1 Transmit Deferred Packet Counter Register(aFramesWithDeferredXmissions) (PM1_TDFRn)	64	R	0000_0000_0000_0000h
6D8h	Port MAC 1 Transmit Multiple Collisions Counter Register(aMultipleCollisionFrames) (PM1_TMCOLn)	64	R	0000_0000_0000_0000h
6E0h	Port MAC 1 Transmit Single Collision Counter(aSingleCollisionFrames) Register (PM1_TSCOLn)	64	R	0000_0000_0000_0000h
6E8h	Port MAC 1 Transmit Late Collision Counter(aLateCollisions) Register (PM1_TLCOLn)	64	R	0000_0000_0000_0000h
6F0h	Port MAC 1 Transmit Excessive Collisions Counter Register (PM1_TECOLn)	64	R	0000_0000_0000_0000h
700h	Port MAC 1 Interface Mode Control Register (PM1_IF_MODE)	32	RW	0000_4004h
800h	Port MAC Merge Control and Status Register (MAC_MERGE_MMCSR)	32	RW	0502_0001h
808h	Port MAC Merge Frame Assembly Error Count Register (MAC_MERGE_MMFAECR)	32	RW	0000_0000h
80Ch	Port MAC Merge Frame SMD Error Count Register (MAC_MERGE_MMFSECR)	32	RW	0000_0000h
810h	Port MAC Merge Frame Assembly OK Count Register (MAC_MERGE_MMFAOCR)	32	RW	0000_0000h
814h	Port MAC Merge Fragment Count RX Register (MAC_MERGE_MMFCRXR)	32	RW	0000_0000h
818h	Port MAC Merge Fragment Count TX Register (MAC_MERGE_MMFCTXR)	32	RW	0000_0000h
81Ch	Port MAC Merge Hold Count Register (MAC_MERGE_MMHCR)	32	RW	0000_0000h
C00h	Port external MDIO configuration register (PEMDIOCR)	32	RW	See section
C04h	Port external MDIO interface control register (PEMDIOICR)	32	RW	0000_0000h
C08h	Port external MDIO interface data register (PEMDIOIDR)	32	RW	0000_0000h
C0Ch	Port external MDIO register address register (PEMDIORAR)	32	RW	0000_0000h
C10h	Port external MDIO status register (PEMDIOSR)	32	R	0000_8000h
C20h	PHY status configuration register (PPSCR)	32	RW	0000_0000h
C24h	Port PHY status control register (PPSCTRLR)	32	RW	0000_0000h
C28h	Port PHY status data register (PPSDR)	32	R	0000_0000h
C2Ch	Port PHY status register address register (PPSRAR)	32	RW	0000_0000h
C30h	Port PHY status event register (PPSER)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C34h	Port PHY status mask register (PPSMR)	32	RW	0000_0000h

53.4.6.9.2 Port MAC a Command and Configuration Register (PM0_COMMAND_CONFIG - PM1_COMMAND_CONFIG)

Offset

Register	Offset
PM0_COMMAND_CONFIG	8h
PM1_COMMAND_CONFIG	408h

Function
MAC Command and Configuration Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MG	TS_MODE	Reserved	RX_FLUSH	Reserved	SWR	Reserved	Reserved	TX_LOWP...	TX_FLUSH	Reserved		Reserved	HD_FCEN	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TXP	TS_PNT	CNT_FRM...	LPBK_MODE		LOOP_ENA	TX_ADDR...	PAUSE_I...	PAUSE_F...	Reserved					RX_EN	TX_EN
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MG	Magic Packet detection enable. 0 Normal operation or magic packet mode has exited with reception of a valid magic packet 1 All received frames are searched for the magic packet pattern.
30 TS_MODE	Transmit timestamp mode 0 Use synchronized ns time

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1 Use free running time
29 —	Reserved
28 RX_FLUSH	Ingress flush enable 0 Rx flush is not enabled 1 Rx flush is enabled. The MAC will immediately stop receiving line data and will flush the Rx pipeline out through the Rx FIFO with Rx error flag set. The MAC will stop receiving new packets until this bit is cleared. **NOTE** This operation does not require active Rx interface clocks.
27 —	Reserved
26 SWR	Software Reset. Self clearing bit. 1 Resets all statistic counters as well as the transmit and receive FIFOs. It should be issued after all traffic has been stopped as a result of clearing the rx/tx enable bits.
25 —	Reserved
24 —	Reserved
23 TX_LOWP_ENA	Transmit Low Power Idle Enable. Transmit low power idle is not supported in RMII mode or on MAC 1. When preemption is enabled, ensure that MAC 1 is idle (see PM1_IEVENT[TX_EMPTY]) before enabling the LPI on MAC 0. 0b - (default), the MAC operates in normal mode. 1b - The MAC completes the transmission of the current Frame and generates Low Power Idle Sequences to the line. It is advised to inspect IEVENT[TX_EMPTY] is set before enabling the LPI.
22 TX_FLUSH	Tx flush Egress Flush Enable. If set to 0b1, the MAC reads out the Tx FIFO and drops the data (data is not sent out on the line). If a frame is being transmitted when Tx flush is set to 1, the frame is aborted with an invalid FCS.
21-20 —	Reserved
19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 HD_FCEN	Half Duplex Flow Control Enable 0 Half duplex flow control is disabled 1 Half duplex flow control is enabled
17 —	Reserved
16 —	Reserved
15 TXP	Enable padding of frames in transmit direction (1, default). 0 - The MAC will not extend frames from the application to a minimum of 64 octets, allowing transmit of short frames (violating the Ethernet minimum size requirement). 1 - For any frames from the application which are less than 64 octets, the MAC will pad with 0s as needed to extend to 60 octets and append 4 octets of FCS. If the application is configured for user-generated or preserved CRC, the MAC will append the required padding bytes after the original FCS.
14 TS_PNT	Timestamp Point 0 Timestamp is captured based on SFD detect at boundary of MAC merge layer and pins/protocol gaskets. 1 Timestamp is captured based on PHY SFD detect pulse on Rx and Tx for 2-step timestamping. For 1-step timestamping, timestamp is captured at MAC merge layer pin/protocol gasket boundary regardless of this setting.
13 CNT_FRM_EN	Control frame reception enable 0 All control frames which are not PAUSE (opcode 0x0001) are discarded. 1 All control frames which are not PAUSE are forwarded to the user application.
12-11 LPBK_MODE	Loopback mode 0 MAC merge loopback using external Tx clock. The MAC Tx data post merge is looped back. If preemption is enabled on the port, both EMAC and PMAC should be in loopback mode. The external Tx clock for the selected interface must be running in this mode. 1: Reserved 2 MAC merge loopback using internal Tx clock. The MAC Tx data post merge is looped back. If preemption is enabled on the port, both EMAC and PMAC should be in loopback mode. The external Tx clock for the selected interface may be inactive in this mode. The target throughput per interface is as follows: XGMII - 10 Gbps SGMII - 2.5 Gbps GMII - 1 Gbps RGMII - 1 Gbps / 100 Mbps / 10 Mbps MII - 100 Mbps / 10 Mbps

Table continues on the next page...

Table continued from the previous page...

Field	Function
	RMII - 100 Mbps / 10 Mbps **NOTE** The target throughput rates will not be exactly matched for all device operating frequencies. For target rates of 1G/100M/10M, and given the operating frequency, f, the actual loopback throughput can be computed as follows: Actual = (Target * int($16*10^9$/f) * f) / $16*10^9$ For target rates of 2.5G/10G, and given the operating frequency, f, the actual loopback throughput can be computed as follows: Actual = (Target * int($16*10^9$/f * f * 3.2) / ($16*10^9$ * 3.1875) 3 Reserved
10 LOOP_ENA	Loopback enable 0 Configured for normal (non-loopback) operation. 1 Configured for internal loopback mode. Interface Tx clock must be running for internal loopback mode
9 TX_ADDR_INS	Transmit source MAC address insertion 0 MAC transmits the source MAC address unmodified from user application. 1 MAC overwrites the source MAC address with the MAC programmed address (PM*a*_MAC_ADDR_0/1).
8 PAUSE_IGN	Ignore PAUSE frame quanta 0 MAC stops transmit process for the duration specified in the PAUSE frame quanta of a received valid PAUSE frame. 1 MAC ignores received PAUSE frames.
7 PAUSE_FWD	Terminate/forward received PAUSE frames 0 MAC terminates received valid PAUSE frames. 1 MAC forwards received valid PAUSE frames to the user application.
6-2 —	Reserved
1 RX_EN	MAC receive path enable 0 MAC receive path is disabled 1 MAC receive path is enabled.
0	MAC transmit path enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TX_EN	0 MAC transmit path is disabled 1 MAC transmit path is enabled.

53.4.6.9.3 Port MAC a MAC Address Register 0 (PM0_MAC_ADDR_0 - PM1_MAC_ADDR_0)

Offset

Register	Offset
PM0_MAC_ADDR_0	Ch
PM1_MAC_ADDR_0	40Ch

Function

This is the port's MAC address register 0 which is used for transmit PAUSE frame generation and receive magic packet detection.

NOTE

This value is taken directly from the port PMAR0 register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAC_ADDR_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAC_ADDR_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MAC_ADDR_0	MAC address 0 This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset (register bit offset 0-7). This field contains the 4 most significant bytes of the MAC address. For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then PM*a*_MAC_ADDR_0 equals 14131211h.

53.4.6.9.4 Port MAC a MAC Address Register 1 (PM0_MAC_ADDR_1 - PM1_MAC_ADDR_1)

Offset

Register	Offset
PM0_MAC_ADDR_1	10h
PM1_MAC_ADDR_1	410h

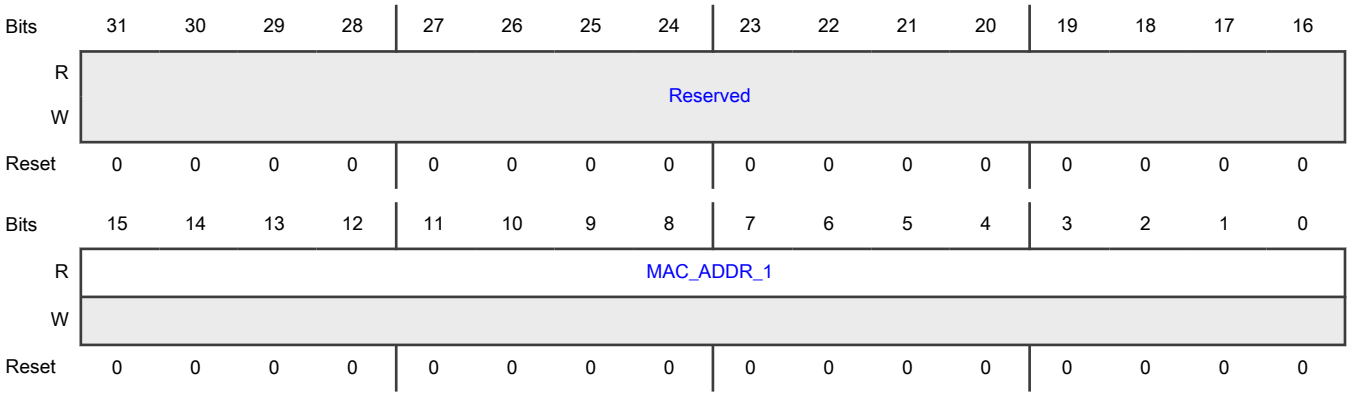
Function

This is the port's MAC address register 0 which is used for transmit PAUSE frame generation and receive magic packet detection.

NOTE

This value is taken directly from the port PMAR1 register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 MAC_ADDR_1	MAC address 1 This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset. This field contains the 2 least significant bytes of the MAC address (least significant byte is stored in register bit offset 8-15). For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then PM*a*_MAC_ADDR_1 equals xxxx1615h (where x should be set to 0).

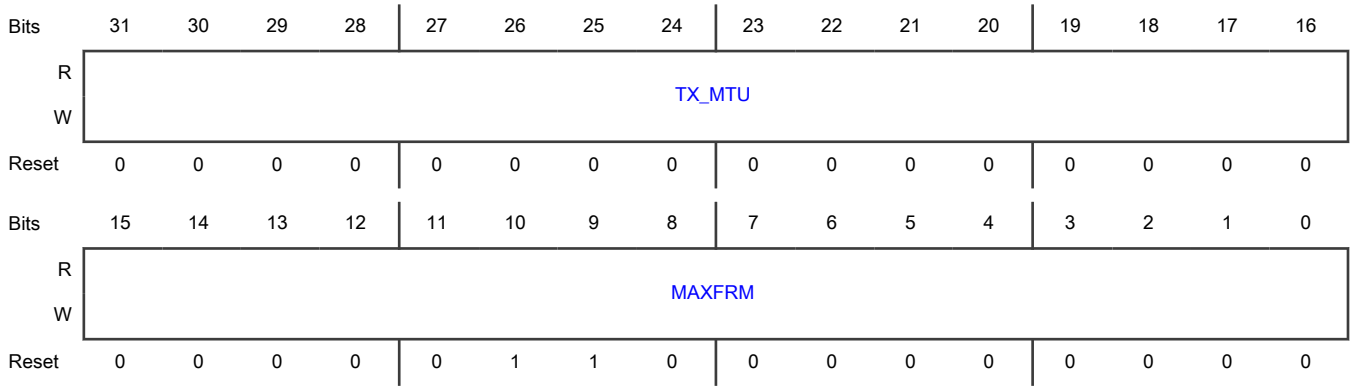
53.4.6.9.5 Port MAC a Maximum Frame Length Register (PM0_MAXFRM - PM1_MAXFRM)

Offset

Register	Offset
PM0_MAXFRM	14h
PM1_MAXFRM	414h

Function
MAC Maximum Frame Length Register

Diagram



Fields

Field	Function
31-16 TX_MTU	Maximum transmit frame length Specifies the different maximum length on transmit. It is used for statistical purposes, see T1523Xn counter (etherStatsPkts1523toXn), and to check for frame size error during transmit. Note that when a frame exceeds this stated maximum, the frame is not dropped. When set to 0 MAXFRM is used instead which is also the default setting.
15-0 MAXFRM	Maximum supported received frame length. The MAC supports reception of any frame size up to 2,000 bytes (0x07D0). Typical settings are 0x05EE (1,518 bytes) for standard untagged frames. Default setting is 0x0600 (1,536 bytes). Received frames that exceed this stated maximum are truncated.(A truncated frame which is anyway corrupted can be in the range of (MAXFRM+4) to (MAXFRM-7). When not using crc forwarding a truncated frame can be in the range of MAXFRM to (MAXFRM-7)).

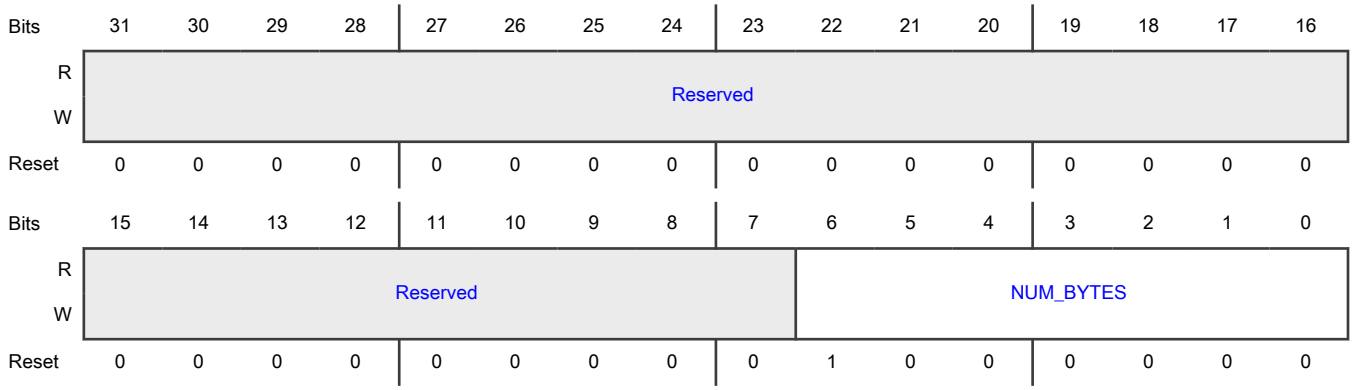
53.4.6.9.6 Port MAC a Minimum Frame Length Register (PM0_MINFRM - PM1_MINFRM)

Offset

Register	Offset
PM0_MINFRM	18h
PM1_MINFRM	418h

Function
MAC Minimum Frame Length Register

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 NUM_BYTES	Receive Minimum Frame Length size in bytes. Received frames greater than or equal to 18B and less than this value are discarded. Received frames shorter than 18B are discarded silently. The discarded frame count is reported in PM*a*_RUNDn stat register **NOTE** Frame includes MAC Header, payload and FCS. Setting the minimum frame length below 64B requires either full duplex or collision free half-duplex operation.

53.4.6.9.7 Port MAC 0 Internal MDIO Configuration Register (PM0_MDIO_CFG)

Offset

Register	Offset
PM0_MDIO_CFG	30h

Function

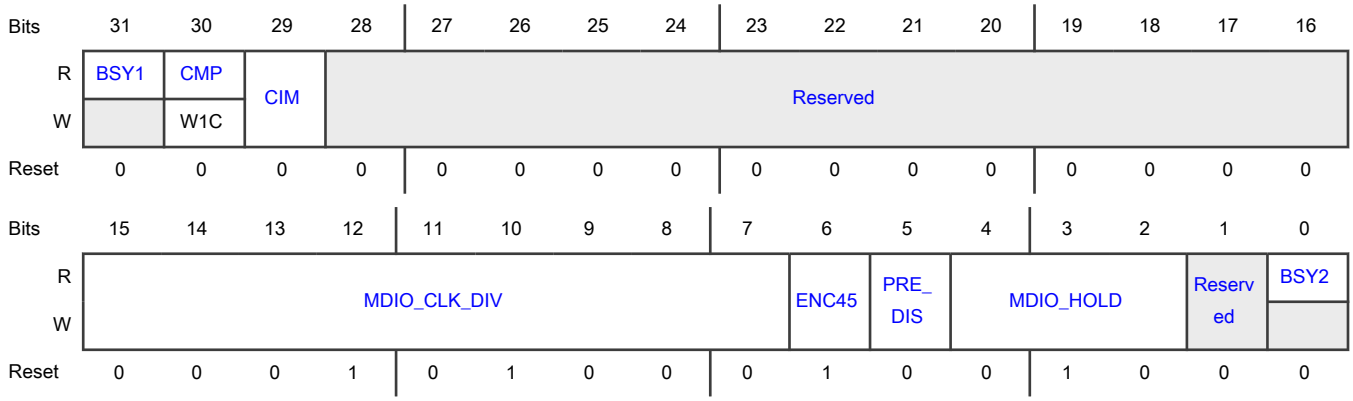
Internal MDIO is used to access RevMII registers.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_ETH_MAC_PORT	PM0_MDIO_CFG	—
SW0_ETH_MAC_PORT0	—	PM0_MDIO_CFG
SW0_ETH_MAC_PORT1	—	PM0_MDIO_CFG
SW0_ETH_MAC_PORT2	PM0_MDIO_CFG	—
SW0_ETH_MAC_PORT3	PM0_MDIO_CFG	—

Diagram



Fields

Field	Function
31 BSY1	MDIO busy 0 An MDIO transaction is not occurring; software may access other MDIO registers. 1 An MDIO transaction is occurring.
30 CMP	MDIO command completion event. Bit is cleared by writing `1'. 0 An MDIO command completion did not occur. 1 An MDIO command completion occurred.
29 CIM	MDIO command completion interrupt mask. 0 masked

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1 enabled
28-16 —	Reserved
15-7 MDIO_CLK_DIV	MDIO clock divisor. A value of 5-511 can be set to configure the MDIO clock frequency relative to NETC. Ratio is (2*MDIO_CLK_DIV)+1.Setting the divisor to 0 disables MDC.Note - The default is 40. Note - for external MDIO Ethernet management interface the default is 0. The minimum value which should be written to this bitfield is 2, which corresponds to a clock divide ratio of 5.
6 ENC45	Enable Clause 45 support. 0 - Clause 22 transactions are used. 1 - Clause 45 transactions are used.
5 PRE_DIS	MDIO preamble disable. 0 Generation of MDIO preamble is enabled (default operation) 1 Generation of MDIO preamble is disabled.
4-2 MDIO_HOLD	MDIO hold time 000 1 NETC cycle 001 3 NETC cycles 010 5 NETC cycles (default - recommended value) 011 7 NETC cycles 100 9 NETC cycles 101 11 NETC cycles 110 13 NETC cycles 111 15 NETC cycles
1 —	Reserved
0 BSY2	MDIO busy (same as bit 31) 0 An MDIO transaction is not occurring; software may access other MDIO registers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1 An MDIO transaction is occurring.

53.4.6.9.8 Port MAC 0 Internal MDIO Interface Control Register (PM0_MDIO_CTL)

Offset

Register	Offset
PM0_MDIO_CTL	34h

Function

MAC Internal MDIO Interface control register

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_ETH_MAC_PORT	PM0_MDIO_CTL	—
SW0_ETH_MAC_PORT0	—	PM0_MDIO_CTL
SW0_ETH_MAC_PORT1	—	PM0_MDIO_CTL
SW0_ETH_MAC_PORT2	PM0_MDIO_CTL	—
SW0_ETH_MAC_PORT3	PM0_MDIO_CTL	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BSY	Reserved														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	READ	Reserved														
W							PORT_ADDR						DEV_ADDR			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BSY	MDIO busy 0 An MDIO transaction is not occurring; software may access other MDIO registers. 1 An MDIO transaction is occurring.
30-16 —	Reserved
15 READ	MDIO read initiation. 1 A normal MDIO read transaction is initiated.
14-10 —	Reserved
9-5 PORT_ADDR	MDIO PHY address (Clause 22)
4-0 DEV_ADDR	MDIO register address (Clause 22)

53.4.6.9.9 Port MAC 0 Internal MDIO Interface Data Register (PM0_MDIO_DATA)

Offset

Register	Offset
PM0_MDIO_DATA	38h

Function
MAC Internal MDIO Interface Data Register

NOTE
Each module instance supports a different number of registers.

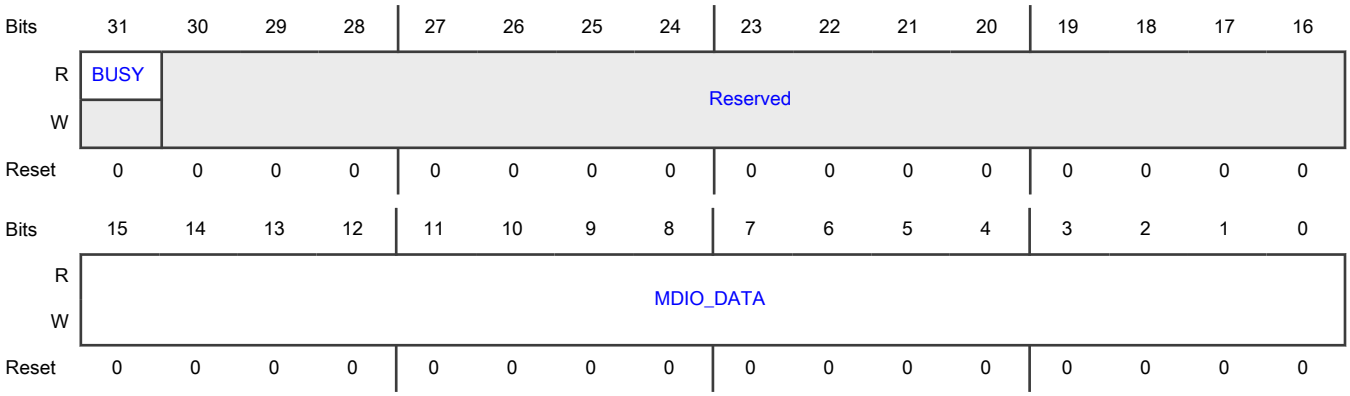
Instance	Register supported	Register not supported
ENETC0_ETH_MAC_PORT	PM0_MDIO_DATA	—
SW0_ETH_MAC_PORT0	—	PM0_MDIO_DATA
SW0_ETH_MAC_PORT1	—	PM0_MDIO_DATA
SW0_ETH_MAC_PORT2	PM0_MDIO_DATA	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
SW0_ETH_MAC_PORT3	PM0_MDIO_DATA	—

Diagram



Fields

Field	Function
31 BUSY	MDIO busy bit. The state of this bit is also reflected in MDIO_CFG[BSY]. 0 MDIO_DATA[MDIO_DATA] contains valid read data. 1 MDIO write transaction in progress; MDIO_DATA[MDIO_DATA] data is invalid.
30-16 —	Reserved
15-0 MDIO_DATA	16-bit MDIO data. Writing to this register initiates a write transaction to the PHY. For proper operation, the MDIO_CTL register must have been initialized first. The MDIO_CFG[BUSY] status bit is set immediately and cleared when the write transaction has completed. Reading this register returns data read from the PHY register specified in MDIO_CTL after a read transaction has completed. Read transactions are initiated by writing a 1 to MDIO_CTL[READ] or MDIO_CTL[POST_INC]. Read data is invalid if MDIO_CFG[BUSY] is set.

53.4.6.9.10 Port MAC a Interrupt Event Register (PM0_IEVENT - PM1_IEVENT)

Offset

Register	Offset
PM0_IEVENT	40h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
PM1_IEVENT	440h

Function

MAC Interrupt Event Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			MRG_AERR	MRG_SERR	Reserved			SPD_DUP	RX_CSD	TX_CSD	Reserved				
W				W1C	W1C				W1C	W1C	W1C					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	MGI	Reserved	RX_OVFL	TX_UNFL	TX_OVFL	Reserved			RX_EMPTY	TX_EMPTY	Reserved	Reserved			
W		W1C		W1C	W1C	W1C										
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 MRG_AERR	MAC merge frame assembly error event The count register for this error is the MAC Merge Frame Assembly Error Count Register (MAC_MERGE_MMFAECR).
27 MRG_SERR	MAC merge frame SMD error received event The count register for this error is the MAC Merge Frame SMD Error Count Register (MAC_MERGE_MMFSECR).
26-24 —	Reserved
23 SPD_DUP	Speed/Duplex Change 0 No speed/duplex change detected 1 Speed/Duplex change detected for RGMII or serial modes.
22 RX_CSD	Rx Clock Stop Detection 0b - Rx clock is running

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Rx clock has stopped. This flag is set if the Rx clock for the selected interface becomes inactive for 64 byte times. This flag is not set when Rx is disabled (PMa_COMMAND_CONFIG[RX_EN] = 0). **NOTE** For RGMII, this flag is not set when the Rx clock is stopped due to low power idle clock stop mode.
21 TX_CSD	Tx Clock Stop Detection 0b - Tx clock is running 1b - Tx clock has stopped. This flag is set if the Tx clock for the selected interface becomes inactive for 64 byte times. This flag is not set when Tx is disabled (PMa_COMMAND_CONFIG[TX_EN] = 0). **NOTE** For RGMII, this flag is not set when the Tx clock is stopped due to low power idle with clock stop mode enabled (PMa_IF_MODE[CLK_STOP] = 1).
20-15 —	Reserved
14 MGI	Magic packet detection indication event 0 Magic packet was not detected. 1 Magic packet was detected when PM*a*_COMMAND_CONFIG[MG] was set. PM*a*COMMAND_CONFIG[MG] is cleared upon magic packet detection.
13 —	Reserved
12 RX_OVFL	Receive FIFO overflow event. 0 The receive FIFO has not overflowed. 1 The receive FIFO overflowed. RxFIFO overflow is a non-fatal error.
11 TX_UNFL	Transmit FIFO underflow event. 0 The transmit FIFO has not underflowed. 1 The transmit FIFO underflowed. TxFIFO underflow is a non-fatal error.
10 TX_OVFL	Transmit FIFO overflow event. 0 The transmit FIFO has not overflowed. 1 The transmit FIFO overflowed. TxFIFO overflow is a fatal error and requires FLR to recover normal operation.
9-7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6 RX_EMPTY	Receive idle event 0 Rx is not idle: Rx fifo not empty or Rx packet receive in process 1 Rx is idle: Rx fifo is empty and no Rx packet receive in process.
5 TX_EMPTY	Transmit fifo empty event 0 TxFIFO is not empty 1 TxFIFO is empty. To ensure Tx is idle, wait 64 byte time after TX_EMPTY asserts.
4 —	Reserved
3-0 —	Reserved

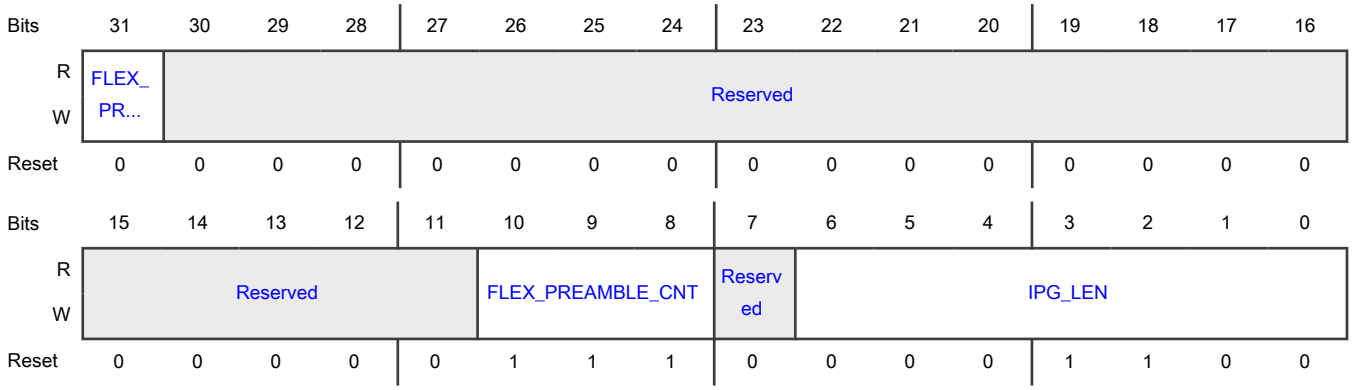
53.4.6.9.11 Port MAC a Transmit Inter-Packet Gap Length and Flexible Preamble length Register (PM0_TX_IPG_PREAMBLE - PM1_TX_IPG_PREAMBLE)

Offset

Register	Offset
PM0_TX_IPG_PREAMBLE	44h
PM1_TX_IPG_PREAMBLE	444h

Function
MAC Transmit Inter-Packet Gap Length and Preamble Length Control Register

Diagram



Fields

Field	Function
31 FLEX_PREAMBLE_EN	Enable Flexible Preamble Count Enables the insertion of less than 7bytes of 8'h55 in the Preamble
30-11 —	Reserved
10-8 FLEX_PREAMBLE_CNT	Flexible Preamble Count When enabled, this field determines the number of bytes of 8'h55 sent out as part of preamble. Valid values are 1 to 7, with 7 being default **NOTE** Generated pause frames always have 7B of preamble, regardless of the setting of FLEX_PREAMBLE_CNT.
7 —	Reserved
6-0 IPG_LEN	Transmit inter-packet gap value. IPG_LEN represents the minimum number of octets between packets. Valid values are 4..24. The default of 0xC (12 octets) is required to conform to IEEE 802.3.

53.4.6.9.12 Port MAC a Interrupt Mask Register(INT_MASK) (PM0_IMASK - PM1_IMASK)

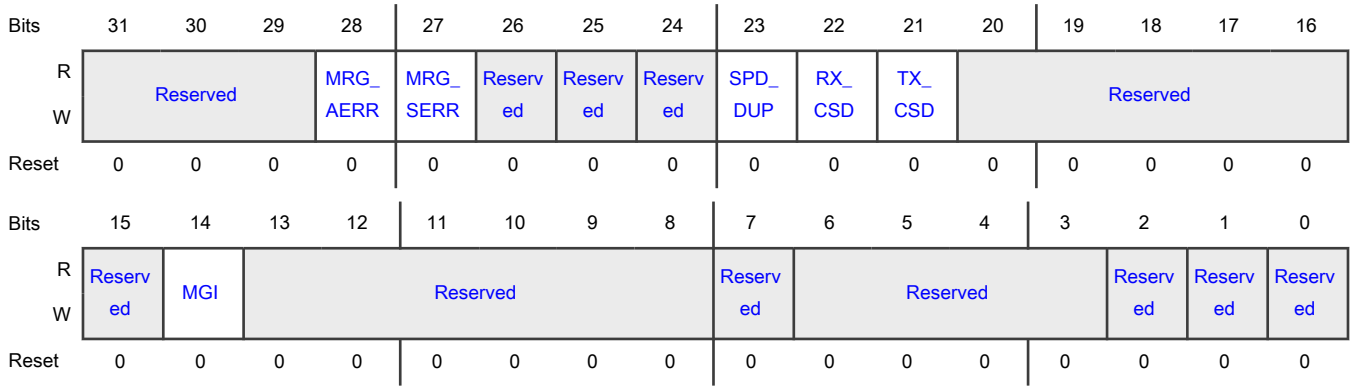
Offset

Register	Offset
PM0_IMASK	4Ch
PM1_IMASK	44Ch

Function

MAC Interrupt Mask Register(INT_MASK)

Diagram



Fields

Field	Function
31-29 —	Reserved
28 MRG_AERR	MAC merge frame assembly error event interrupt mask 0 masked 1 enabled
27 MRG_SERR	MAC merge frame SMD error received event interrupt mask 0 masked 1 enabled
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 SPD_DUP	Speed/Duplex change event mask. 0 masked 1 enabled
22	Rx Clock Stop Detection

Table continues on the next page...

Table continued from the previous page...

Field	Function
RX_CSD	0 masked 1 enabled
21 TX_CSD	Tx Clock Stop Detection 0 masked 1 enabled
20-15 —	Reserved
14 MGI	Magic packet detection indication event mask. 0 masked 1 enabled
13-8 —	Reserved
7 —	Reserved
6-3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

53.4.6.9.13 Port MAC a Pause Quanta Register (PM0_PAUSE_QUANTA - PM1_PAUSE_QUANTA)

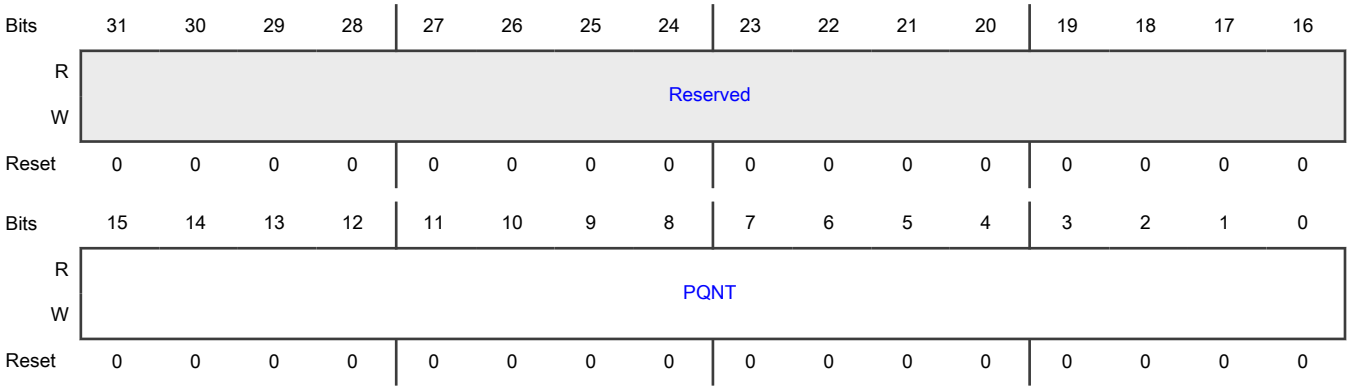
Offset

Register	Offset
PM0_PAUSE_QUANTA	54h
PM1_PAUSE_QUANTA	454h

Function

This is the port MAC pause quanta register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 PQNT	Value to be used for the quanta value when XOFF is triggered.

53.4.6.9.14 Port MAC a Pause Quanta Threshold Register (PM0_PAUSE_THRESH - PM1_PAUSE_THRESH)

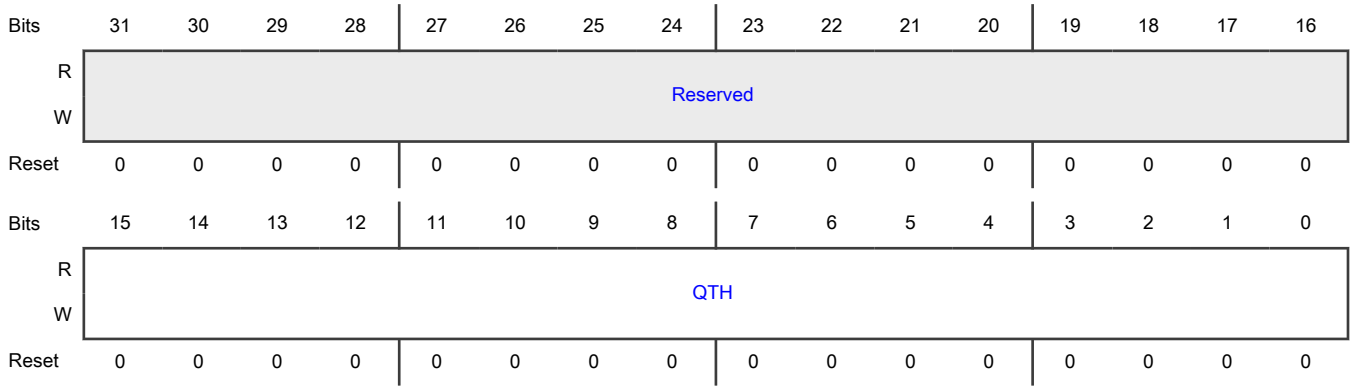
Offset

Register	Offset
PM0_PAUSE_THRESH	64h
PM1_PAUSE_THRESH	464h

Function

MAC Pause Quanta Threshold Register

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 QTH	Quanta threshold. When a quanta timer counts down and reaches this value, the MAC sends a refresh PAUSE frame with the programmed full quanta value if a pause condition still exists. Value must be greater or equal to 2 and less than PQNT. Since the refresh pause frame may be delayed by the transmission of an application frame always in progress, the recommended setting of QTH should take into account worst-case transmission delay to ensure the pause quanta does not expire before it can be refreshed.

53.4.6.9.15 Port MAC a Receive Pause Status Register (PM0_RX_PAUSE_STATUS - PM1_RX_PAUSE_STATUS)

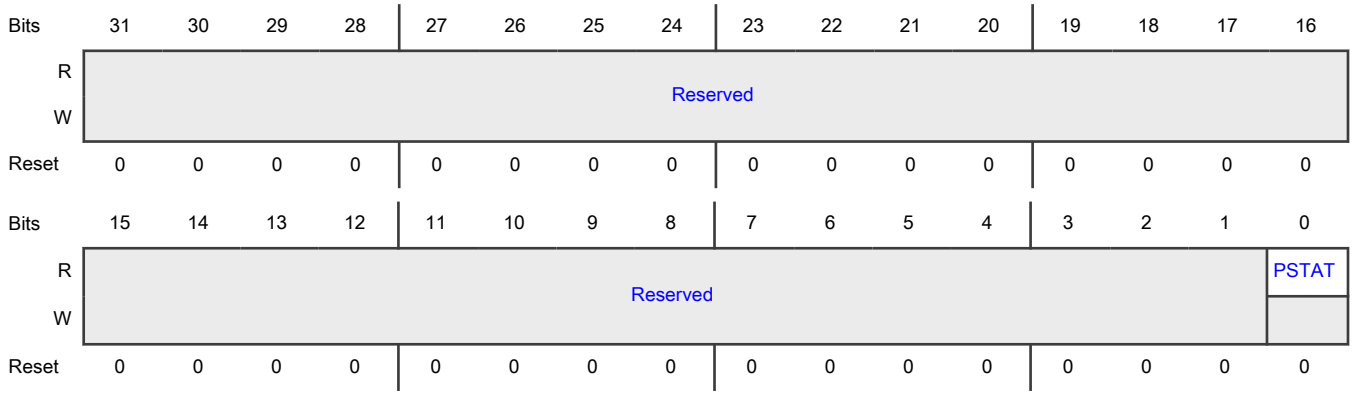
Offset

Register	Offset
PM0_RX_PAUSE_STATUS	74h
PM1_RX_PAUSE_STATUS	474h

Function

MAC Receive Pause Status Register

Diagram



Fields

Field	Function
31-1 —	Reserved
0 PSTAT	Pause status.

53.4.6.9.16 Port MAC a EEE Low Power Wakeup Timer Register (PM0_LPWAKE_TIMER - PM1_LPWAKE_TIMER)

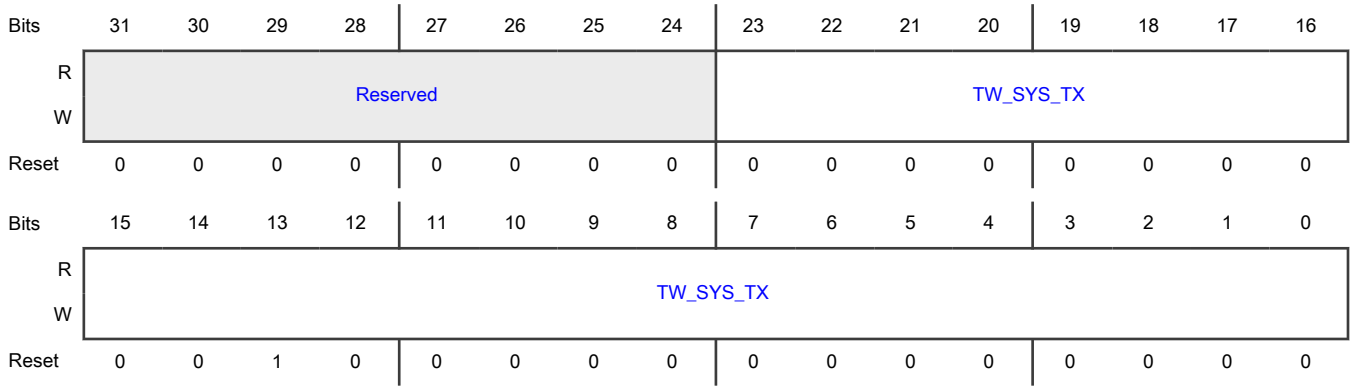
Offset

Register	Offset
PM0_LPWAKE_TIMER	B8h
PM1_LPWAKE_TIMER	4B8h

Function

MAC EEE Low Power Wakeup Timer Register

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 TW_SYS_TX	EEE System transmit wait time Defines the number of NETC cycles (which represents time) required by the PHY to wait before transmitting a new frame after the application has indicated it wants to end the low power state.

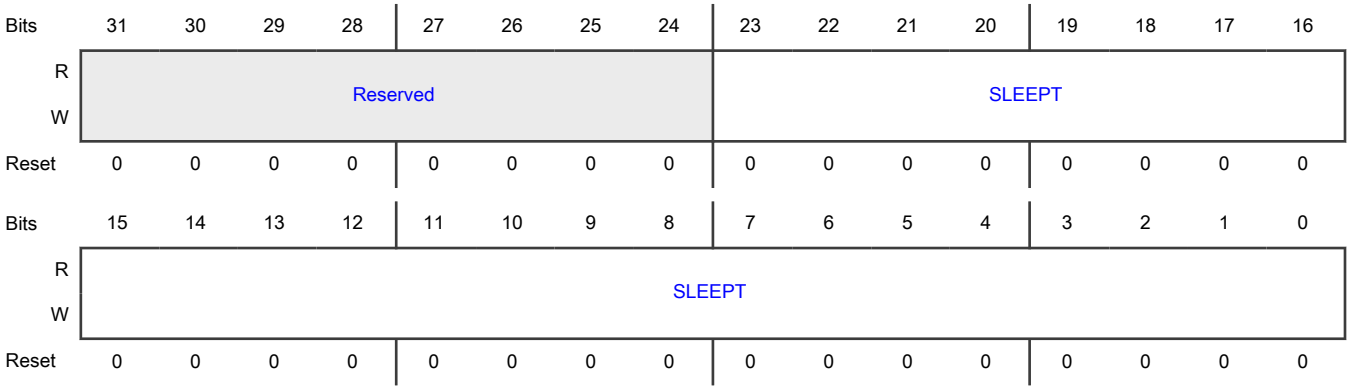
53.4.6.9.17 Port MAC a Transmit EEE Low Power Timer Register (PM0_SLEEP_TIMER - PM1_SLEEP_TIMER)

Offset

Register	Offset
PM0_SLEEP_TIMER	BCh
PM1_SLEEP_TIMER	4BCh

Function
MAC Transmit EEE Low Power Timer Register

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 SLEEPT	Defines the number of NETC cycles (which represents time) where Tx is idle before mac transmits low power EEE.A value of 0 does not activate low power EEE transmission.

53.4.6.9.18 Port MAC a IEEE1588 Single-Step Control Register (PM0_SINGLE_STEP - PM1_SINGLE_STEP)

Offset

Register	Offset
PM0_SINGLE_STEP	C0h
PM1_SINGLE_STEP	4C0h

Function
MAC IEEE1588 Single-Step Control Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EN	Reserved														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OFFSET									CH	Reserved					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EN	IEEE-1588 Single-Step enable.
30-16 —	Reserved
15-7 OFFSET	Start offset from the beginning of a frame where the field to update is found (index to MS byte).The offset must respect all MAC headers, VLAN tags and other protocol headers accordingly.
6 CH	Checksum update 1 - UDP checksum needs correction 0 - no UDP checksum update should be done
5-0 —	Reserved

53.4.6.9.19 Port MAC a half-duplex backoff entropy register (PM0_HD_BACKOFF_ENTROPY - PM1_HD_BACKOFF_ENTROPY)

Offset

Register	Offset
PM0_HD_BACKOFF_ENTROPY	D0h
PM1_HD_BACKOFF_ENTROPY	4D0h

Function

This is the Port MAC half-duplex backoff entropy register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SW_ENTR...	Reserved														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						HD_BACKOFF_ENTROPY									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SW_ENTROPY_VALID	SW programmable entropy valid 0 Not valid 1 Valid
30-10 —	Reserved
9-0 HD_BACKOFF_ENTROPY	Half duplex backoff entropy SW programmable entropy for random number generator for back-off in Half Duplex Mode.

53.4.6.9.20 Port MAC a Half-Duplex Flow Control Register (PM0_HD_FLOW_CTRL - PM1_HD_FLOW_CTRL)

Offset

Register	Offset
PM0_HD_FLOW_CTRL	D4h
PM1_HD_FLOW_CTRL	4D4h

Function
This is the half-duplex flow control register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				HD_BP_ON_MAX											
W																
Reset	0	0	0	0	1	0	1	1	1	1	0	1	1	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				HD_BP_OFF_MIN											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

Fields

Field	Function
31-28 —	Reserved
27-16 HD_BP_ON_MAX	Half-Duplex Back-Pressure On Maximum Maximum amount of time backpressure can stay asserted before stopping to prevent excess defer on link partner, in byte times. Per spec, maximum can be 3036 byte times (2*max basic frame size of 1518B).
15-12 —	Reserved
11-0 HD_BP_OFF_MIN	Half-Duplex Back-Pressure Off Minimum Minimum amount of time backpressure will stay off after reaching the ON max, before backpressure can reassert after checking if icm_pause_notification is still or again asserted, in byte times. Minimum is 20 byte times (normal IPG + normal preamble).

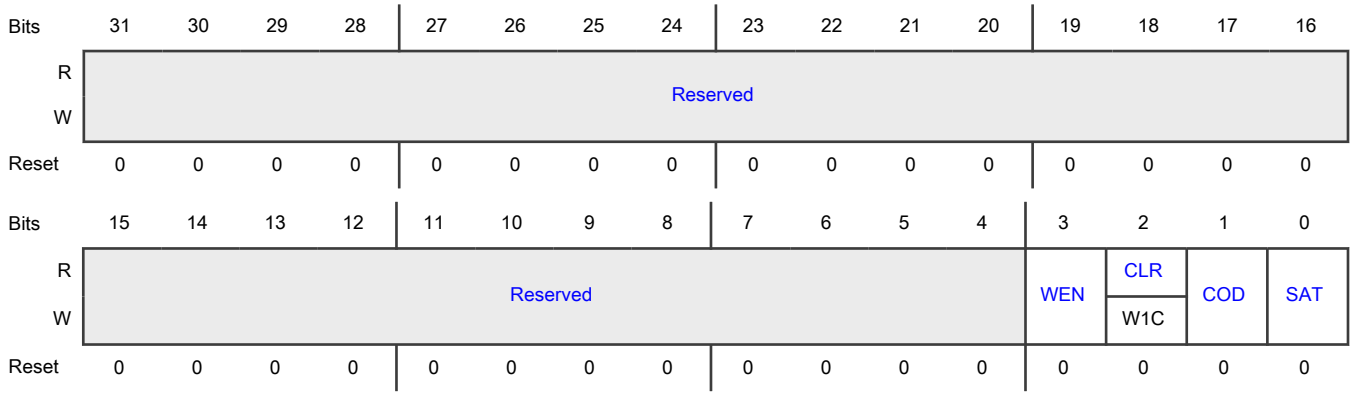
53.4.6.9.21 Port MAC a Statistics Configuration Register (PM0_STATN_CONFIG - PM1_STATN_CONFIG)

Offset

Register	Offset
PM0_STATN_CONFIG	E0h
PM1_STATN_CONFIG	4E0h

Function
MAC Statistics Configuration Register

Diagram



Fields

Field	Function
31-4 —	Reserved
3 WEN	Write enable for Tx/Rx stats registers 0 Attempted writes to Tx/Rx stats are ignored (default) 1 Writes to Tx/Rx stats are permitted
2 CLR	1 - all counters will be reset to 0. Note that this takes at least 32 NETC cycles. Self resetting bit, reads 0 always. The soft-reset from command config will also trigger this counter clear function.
1 COD	0 - (default) counters are not affected by read. 1 - a read to a counter resets it to 0.
0 SAT	0 - (default) counters are wrapping around 1- the counters saturate at the maximum value. Typically used in combination with CLR_ON_RD.

53.4.6.9.22 Port MAC a Receive Ethernet Octets Counter(etherStatsOctetsn) (PM0_REOCTn - PM1_REOCTn)

Offset

Register	Offset
PM0_REOCTn	100h
PM1_REOCTn	500h

Function
MAC Receive Ethernet Octets Counter(etherStatsOctetsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	REOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	REOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 REOCTn	Incremented for each octet received in both good and bad packets.

53.4.6.9.23 Port MAC a Receive Octets Counter(ifInOctetsn) (PM0_ROCTn - PM1_ROCTn)

Offset

Register	Offset
PM0_ROCTn	108h
PM1_ROCTn	508h

Function

MAC Receive Octets Counter(ifInOctetsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	ROCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	ROCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ROCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ROCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 ROCTn	Incremented for each octet received except preamble (that is, Header, Payload, Pad and FCS) for all valid frames and valid PAUSE frames received.

53.4.6.9.24 Port MAC a Receive Valid Pause Frame Counter Register(aPAUSEMACCtrlFramesReceivedn) (PM0_RXPFn - PM1_RXPFn)

Offset

Register	Offset
PM0_RXPFn	118h
PM1_RXPFn	518h

Function

MAC Receive Valid Pause Frame Counter Register(aPAUSEMACCtrlFramesReceivedn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RXPFn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RXPFn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXPFn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXPFn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RXPFn	Incremented for each valid PAUSE frame received .

53.4.6.9.25 Port MAC a Receive Frame Counter Register(aFramesReceivedOKn) (PM0_RFRMn - PM1_RFRMn)

Offset

Register	Offset
PM0_RFRMn	120h
PM1_RFRMn	520h

Function

MAC Receive Frame Counter Register(aFramesReceivedOKn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RFRMn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RFRMn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RFRMn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RFRMn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RFRMn	Incremented for each frame received without error, including PAUSE frames.

53.4.6.9.26 Port MAC a Receive Frame Check Sequence Error Counter Register() (PM0_RFCSn - PM1_RFCSn)

Offset

Register	Offset
PM0_RFCSn	128h
PM1_RFCSn	528h

Function

MAC Receive Frame Check Sequence Error Counter Register()

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RFCSn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RFCSn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RFCSn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RFCSn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RFCSn	Incremented for each frame received with a CRC-32 error but the frame is otherwise of correct length.

53.4.6.9.27 Port MAC a Receive VLAN Frame Counter Register(VLANReceivedOKn) (PM0_RVLANn - PM1_RVLANn)

Offset

Register	Offset
PM0_RVLANn	130h
PM1_RVLANn	530h

Function

MAC Receive VLAN Frame Counter Register(VLANReceivedOKn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RVLANn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RVLANn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RVLANn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RVLANn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RVLANn	Incremented for each valid VLAN tagged frame received with ethertype 0x8100

53.4.6.9.28 Port MAC a Receive Frame Error Counter Register(ifInErrorsn) (PM0_RERRn - PM1_RERRn)

Offset

Register	Offset
PM0_RERRn	138h
PM1_RERRn	538h

Function
MAC Receive Frame Error Counter Register(ifInErrorsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RERRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RERRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RERRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RERRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RERRn	Incremented for each frame received with an error (except for undersized/fragment frame): FIFO overflow error CRC error Payload length error Jabber and oversized error

53.4.6.9.29 Port MAC a Receive Unicast Frame Counter Register(ifInUcastPktsn) (PM0_RUCAn - PM1_RUCAn)

Offset

Register	Offset
PM0_RUCAn	140h
PM1_RUCAn	540h

Function
MAC Receive Unicast Frame Counter Register(ifInUcastPktsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RUCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RUCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RUCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RUCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RUCAn	Incremented for each valid frame received (on the receive FIFO interface) in which bit 0 of the destination address was 0 .

53.4.6.9.30 Port MAC a Receive Multicast Frame Counter Register(ifInMulticastPktsn) (PM0_RMCAn - PM1_RMCAn)

Offset

Register	Offset
PM0_RMCAn	148h
PM1_RMCAn	548h

Function

MAC Receive Multicast Frame Counter Register(ifInMulticastPktsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RMCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RMCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RMCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RMCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RMCAn	Incremented for each valid frame received (on the receive FIFO interface) in which bit 0 of the destination address was 1 but not the broadcast address (all bits set to 1). This count excludes PAUSE frames.

53.4.6.9.31 Port MAC a Receive Broadcast Frame Counter Register(ifInBroadcastPktsn) (PM0_RBCAn - PM1_RBCAn)

Offset

Register	Offset
PM0_RBCAn	150h
PM1_RBCAn	550h

Function

MAC Receive Broadcast Frame Counter Register(ifInBroadcastPktsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RBCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RBCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RBCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RBCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RBCAn	Incremented for each valid frame received (on the receive FIFO interface) in which all bits of the destination address were 1 .

53.4.6.9.32 Port MAC a Receive Dropped Packets Counter Register(etherStatsDropEventsn) (PM0_RDRPn - PM1_RDRPn)

Offset

Register	Offset
PM0_RDRPn	158h
PM1_RDRPn	558h

Function

MAC Receive Dropped Packets Counter Register(etherStatsDropEventsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RDRPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RDRPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RDRPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RDRPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RDRPn	Incremented for each dropped packet due to internal errors of the MAC client. Occurs when a receive FIFO overflows. Includes also packets truncated as a result of the receive FIFO overflow.

53.4.6.9.33 Port MAC a Receive Packets Counter Register(etherStatsPktsn) (PM0_RPKTn - PM1_RPKTn)

Offset

Register	Offset
PM0_RPKTn	160h
PM1_RPKTn	560h

Function

MAC Receive Packets Counter Register(etherStatsPktsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RPKTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RPKTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RPKTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RPKTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RPKTn	Incremented for each good or bad packet received.

53.4.6.9.34 Port MAC a Receive Undersized Packet Counter Register(etherStatsUndersizePktsn) (PM0_RUNDn - PM1_RUNDn)

Offset

Register	Offset
PM0_RUNDn	168h
PM1_RUNDn	568h

Function

MAC Receive Undersized Packet Counter Register(etherStatsUndersizePktsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RUNDn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RUNDn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RUNDn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RUNDn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RUNDn	Incremented for each packet received that was less than the length programmed in PM*a*_MINFRM register and greater than or equal to 18 octets.

53.4.6.9.35 Port MAC a Receive 64-Octet Packet Counter Register(etherStatsPkts64OctetsN) (PM0_R64n - PM1_R64n)

Offset

Register	Offset
PM0_R64n	170h
PM1_R64n	570h

Function

MAC Receive 64-Octet Packet Counter Register(etherStatsPkts64OctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	R64n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	R64n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	R64n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	R64n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 R64n	Incremented for each 64-octet frame received, good or bad.

53.4.6.9.36 Port MAC a Receive 65 to 127-Octet Packet Counter Register(etherStatsPkts65to127OctetsN) (PM0_R127n - PM1_R127n)

Offset

Register	Offset
PM0_R127n	178h
PM1_R127n	578h

Function

MAC Receive 65- to 127-Octet Packet Counter Register(etherStatsPkts65to127OctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	R127n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	R127n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	R127n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	R127n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 R127n	Incremented for each 65- to 127-octet frame received, good or bad.

53.4.6.9.37 Port MAC a Receive 128 to 255-Octet Packet Counter Register(etherStatsPkts128to255OctetsN) (PM0_R255n - PM1_R255n)

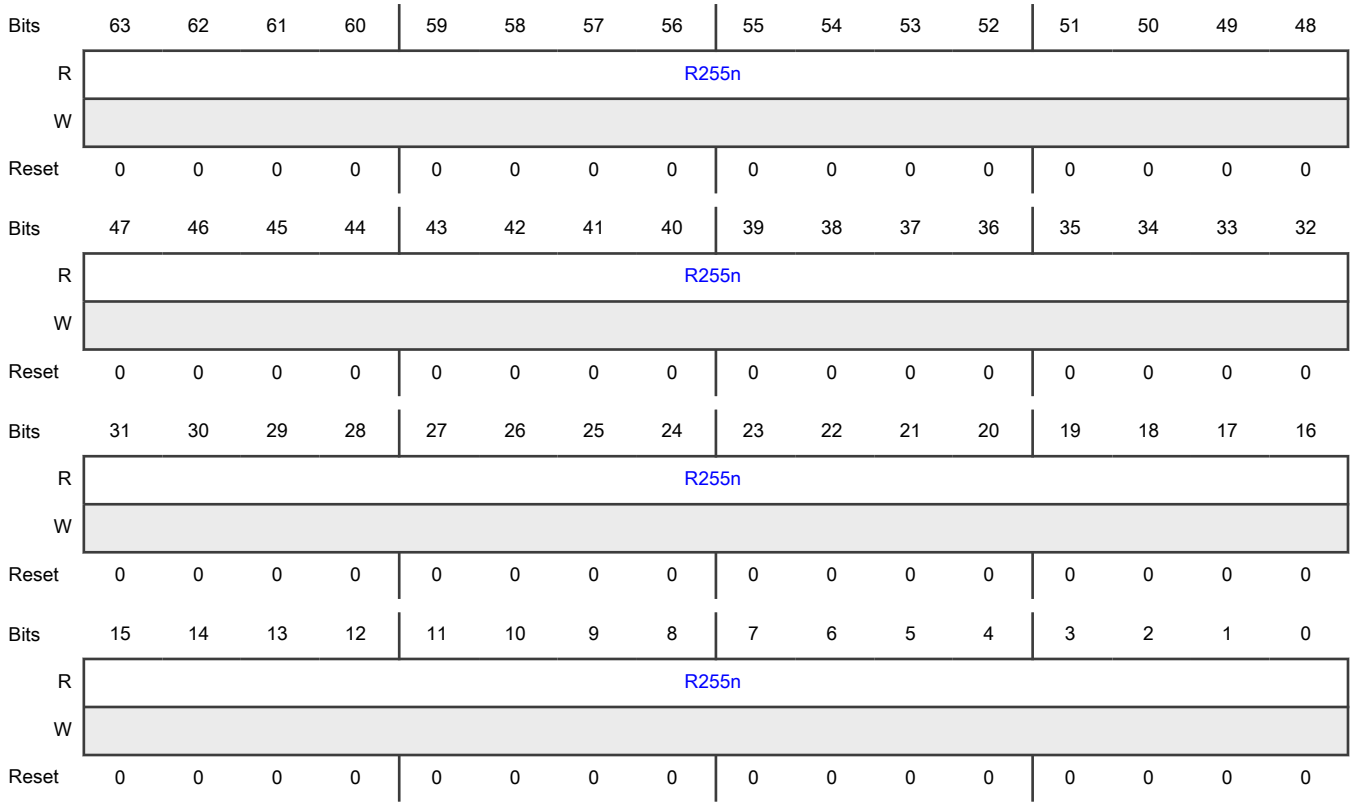
Offset

Register	Offset
PM0_R255n	180h
PM1_R255n	580h

Function

MAC Receive 128- to 255-Octet Packet Counter Register(etherStatsPkts128to255OctetsN)

Diagram



Fields

Field	Function
63-0 R255n	Incremented for each 128- to 255-octet frame received, good or bad.

53.4.6.9.38 Port MAC a Receive 256 to 511-Octet Packet Counter Register(etherStatsPkts256to511OctetsN) (PM0_R511n - PM1_R511n)

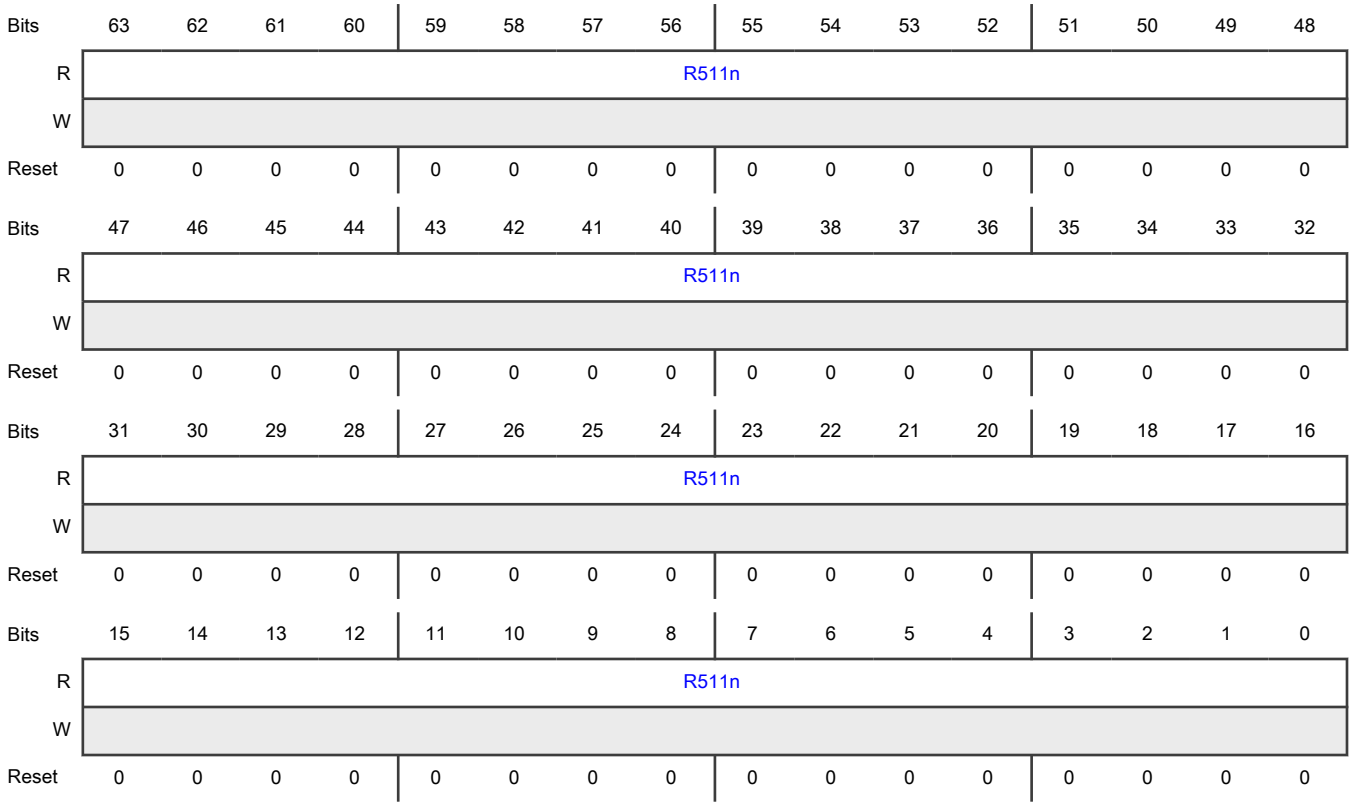
Offset

Register	Offset
PM0_R511n	188h
PM1_R511n	588h

Function

MAC Receive 256- to 511-Octet Packet Counter Register(etherStatsPkts256to511OctetsN)

Diagram



Fields

Field	Function
63-0 R511n	Incremented for each 256- to 511-octet frame received, good or bad.

53.4.6.9.39 Port MAC a Receive 512 to 1023-Octet Packet Counter Register(etherStatsPkts512to1023OctetsN) (PM0_R1023n - PM1_R1023n)

Offset

Register	Offset
PM0_R1023n	190h
PM1_R1023n	590h

Function

MAC Receive 512- to 1023-Octet Packet Counter Register(etherStatsPkts512to1023OctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	R1023n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	R1023n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	R1023n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	R1023n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 R1023n	Incremented for each 512- to 1023-octet frame received, good or bad.

53.4.6.9.40 Port MAC a Receive 1024 to 1522-Octet Packet Counter Register(etherStatsPkts1024to1522OctetsN) (PM0_R1522n - PM1_R1522n)

Offset

Register	Offset
PM0_R1522n	198h
PM1_R1522n	598h

Function

MAC Receive 1024- to 1522-Octet Packet Counter Register(etherStatsPkts1024to1522OctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	R1522n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	R1522n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	R1522n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	R1522n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 R1522n	Incremented for each 1024- to 1522-octet frame received, good or bad.

53.4.6.9.41 Port MAC a Receive 1523 to Max-Octet Packet Counter Register(etherStatsPkts1523toMaxOctetsN) (PM0_R1523Xn - PM1_R1523Xn)

Offset

Register	Offset
PM0_R1523Xn	1A0h
PM1_R1523Xn	5A0h

Function

MAC Receive 1523- to Max-Octet Packet Counter Register(etherStatsPkts1523toMaxOctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	R1523Xn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	R1523Xn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	R1523Xn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	R1523Xn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 R1523Xn	Incremented for each 1523-octet frame and larger (up to the maximum frame length specified in register PM*a*_MAXFRM[MAXFRM]) received, good or bad.

53.4.6.9.42 Port MAC a Receive Oversized Packet Counter Register(etherStatsOversizePktsn) (PM0_ROVRn - PM1_ROVRn)

Offset

Register	Offset
PM0_ROVRn	1A8h
PM1_ROVRn	5A8h

Function

MAC Receive Oversized Packet Counter Register(etherStatsOversizePktsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	ROVRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	ROVRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ROVRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ROVRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 ROVRn	Incremented for each packet which is larger than the maximum frame length specified in the MAXFRM(FRAME_LENGTH) register (excluding framing bits, but including FCS octets) received with a good frame check sequence.

Table continues on the next page...

Field	Function
	NOTE Packets received which are larger than the maximum length specified in PM0_MAXFRM are truncated at length=MAXFRM and forwarded immediately to the upper layers with FCS error. The Rx stats associated with this oversized packet are not updated until the actual end of packet is received by the MAC.

53.4.6.9.43 Port MAC a Receive Jabber Packet Counter Register(etherStatsJabbersn) (PM0_RJBRn - PM1_RJBRn)

Offset

Register	Offset
PM0_RJBRn	1B0h
PM1_RJBRn	5B0h

Function
MAC Receive Jabber Packet Counter Register(etherStatsJabbersn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RJBRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RJBRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RJBRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RJBRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RJBRn	Incremented for each packet which is larger than the maximum frame length specified in register PM*a*_MAXFRM[MAXFRM] (excluding framing bits, but including FCS octets) received with a bad frame check sequence. **NOTE** Packets received which are larger than the maximum length specified in PM0_MAXFRM are truncated at length=MAXFRM and forwarded immediately to the upper layers with FCS error. The Rx stats associated with this oversized packet are not updated until the actual end of packet is received by the MAC.

53.4.6.9.44 Port MAC a Receive Fragment Packet Counter Register(etherStatsFragmentsn (PM0_RFRGn - PM1_RFRGn)

Offset

Register	Offset
PM0_RFRGn	1B8h
PM1_RFRGn	5B8h

Function

MAC Receive Fragment Packet Counter Register(etherStatsFragmentsn

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RFRGn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RFRGn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RFRGn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RFRGn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RFRGn	Incremented for each packet which is shorter than the length programmed in PM*a*_MINFRM register and received with a wrong CRC. **NOTE** Fragments are dropped in MAC

53.4.6.9.45 Port MAC a Receive Control Packet Counter Register (PM0_RCNPn - PM1_RCNPn)

Offset

Register	Offset
PM0_RCNPn	1C0h
PM1_RCNPn	5C0h

Function
MAC Receive Control Packet Counter Register

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RCNPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RCNPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RCNPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCNPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RCNPn	Incremented for each valid control packet (type 0x8808) but not for PAUSE packets

53.4.6.9.46 Port MAC a Receive Dropped Not Truncated Packets Counter Register(etherStatsDropEventsn) (PM0_RDRNTPn - PM1_RDRNTPn)

Offset

Register	Offset
PM0_RDRNTPn	1C8h
PM1_RDRNTPn	5C8h

Function

MAC Receive Dropped Not Truncated Packets Counter Register(etherStatsDropEventsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RDRNTPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RDRNTPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RDRNTPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RDRNTPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RDRNTPn	Incremented for each fully dropped packet (not truncated) due to internal errors of the MAC client. Occurs when a receive FIFO overflows.

53.4.6.9.47 Port MAC a Receive Valid Small Packet Counter Register (PM0_RMIN63n - PM1_RMIN63n)

Offset

Register	Offset
PM0_RMIN63n	1D0h
PM1_RMIN63n	5D0h

Function

MAC Receive Packet Counter for Small Packets with valid CRC

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	RMIN63n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	RMIN63n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RMIN63n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RMIN63n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 RMIN63n	Incremented for each valid small packet less than 64B but greater or equal to the length programmed in PMa_MINFRM register

53.4.6.9.48 Port MAC a Transmit Ethernet Octets Counter(etherStatsOctetsn) (PM0_TEOCTn - PM1_TEOCTn)

Offset

Register	Offset
PM0_TEOCTn	200h
PM1_TEOCTn	600h

Function

MAC Transmit Ethernet Octets Counter(etherStatsOctetsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TEOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TEOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TEOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TEOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TEOCTn	Incremented for each octet transmitted in both good and bad packets.

53.4.6.9.49 Port MAC a Transmit Octets Counter Register(ifOutOctetsn) (PM0_TOCTn - PM1_TOCTn)

Offset

Register	Offset
PM0_TOCTn	208h
PM1_TOCTn	608h

Function

MAC Transmit Octets Counter Register(ifOutOctetsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TOCTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TOCTn	Incremented for each octet transmitted except preamble (that is, Header, Payload, Pad and FCS) for all valid frames and valid PAUSE frames transmitted.

53.4.6.9.50 Port MAC a Transmit Valid Pause Frame Counter Register(aPAUSEMACCtrlFramesReceivedn) (PM0_TXPFn - PM1_TXPFn)

Offset

Register	Offset
PM0_TXPFn	218h
PM1_TXPFn	618h

Function

MAC Transmit Valid Pause Frame Counter Register(aPAUSEMACCtrlFramesReceivedn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TXPFn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TXPFn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXPFn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXPFn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TXPFn	Incremented for each valid PAUSE frame transmitted . Note: Pause frames forwarded to the MAC from MAC Client are not counted by TXPFn.

53.4.6.9.51 Port MAC a Transmit Frame Counter Register(aFramesTransmittedOKn) (PM0_TFRMn - PM1_TFRMn)

Offset

Register	Offset
PM0_TFRMn	220h
PM1_TFRMn	620h

Function

MAC Transmit Frame Counter Register(aFramesTransmittedOKn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TFRMn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TFRMn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TFRMn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TFRMn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TFRMn	Incremented for each frame transmitted without error, including PAUSE frames.

53.4.6.9.52 Port MAC a Transmit Frame Check Sequence Error Counter Register() (PM0_TFCSn - PM1_TFCSn)

Offset

Register	Offset
PM0_TFCSn	228h
PM1_TFCSn	628h

Function

MAC Transmit Frame Check Sequence Error Counter Register()

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TFCSn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TFCSn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TFCSn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TFCSn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TFCSn	Incremented for each frame transmitted with a CRC-32 error except for underflows.

53.4.6.9.53 Port MAC a Transmit VLAN Frame Counter Register(VLANTransmittedOKn) (PM0_TVLANn - PM1_TVLANn)

Offset

Register	Offset
PM0_TVLANn	230h
PM1_TVLANn	630h

Function

MAC Transmit VLAN Frame Counter Register(VLANTransmittedOKn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TVLANn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TVLANn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TVLANn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TVLANn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TVLANn	Incremented for each valid VLAN tagged frame transmitted with ethertype 0x8100.

53.4.6.9.54 Port MAC a Transmit Frame Error Counter Register(ifOutErrorsn) (PM0_TERRn - PM1_TERRn)

Offset

Register	Offset
PM0_TERRn	238h
PM1_TERRn	638h

Function

MAC Transmit Frame Error Counter Register(ifOutErrorsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TERRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TERRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TERRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TERRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TERRn	Transmit frame error count Incremented for each frame transmitted with an error (i.e. frame is transmitted with a bad FCS): • FIFO overflow error • FIFO underflow error • memory double ECC errors • Parity errors • late-collision or excessive collisions (half-duplex only) • Request by the Ethernet I/F Tx function to set the frame FCS field to a bad CRC value. Possible causes: end_of_frame receive errors for cut-through frames, internal faults

53.4.6.9.55 Port MAC a Transmit Unicast Frame Counter Register(ifOutUcastPktsn) (PM0_TUCAn - PM1_TUCAn)

Offset

Register	Offset
PM0_TUCAn	240h
PM1_TUCAn	640h

Function

MAC Transmit Unicast Frame Counter Register(ifOutUcastPktsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TUCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TUCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TUCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TUCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TUCAn	Incremented for each valid frame transmitted (to the FIFO interface) in which bit 0 of the destination address was 0.

53.4.6.9.56 Port MAC a Transmit Multicast Frame Counter Register(ifOutMulticastPktsn) (PM0_TMCAn - PM1_TMCAn)

Offset

Register	Offset
PM0_TMCAn	248h
PM1_TMCAn	648h

Function

MAC Transmit Multicast Frame Counter Register(ifOutMulticastPktsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TMCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TMCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TMCAn	Incremented for each valid frame transmitted (to the FIFO interface) in which bit 0 of the destination address was 1 but not the broadcast address (all bits set to 1).

53.4.6.9.57 Port MAC a Transmit Broadcast Frame Counter Register(ifOutBroadcastPktsn) (PM0_TBCAn - PM1_TBCAn)

Offset

Register	Offset
PM0_TBCAn	250h
PM1_TBCAn	650h

Function

MAC Transmit Broadcast Frame Counter Register(ifOutBroadcastPktsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TBCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TBCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TBCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TBCAn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TBCAn	Incremented for each valid frame transmitted (to the FIFO interface) in which all bits of the destination address were 1 .

53.4.6.9.58 Port MAC a Transmit Packets Counter Register(etherStatsPktsn) (PM0_TPKTn - PM1_TPKTn)

Offset

Register	Offset
PM0_TPKTn	260h
PM1_TPKTn	660h

Function

MAC Transmit Packets Counter Register(etherStatsPktsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TPKTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TPKTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TPKTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TPKTn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TPKTn	Incremented for each good or bad packet transmitted.

53.4.6.9.59 Port MAC a Transmit Undersized Packet Counter Register(etherStatsUndersizePktsn) (PM0_TUNDn - PM1_TUNDn)

Offset

Register	Offset
PM0_TUNDn	268h
PM1_TUNDn	668h

Function

MAC Transmit Undersized Packet Counter Register(etherStatsUndersizePktsn)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TUNDn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TUNDn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TUNDn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TUNDn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TUNDn	Incremented for each packet transmitted that was less than 64 octets long with a good CRC.

53.4.6.9.60 Port MAC a Transmit 64-Octet Packet Counter Register (etherStatsPkts64OctetsN) (PM0_T64n - PM1_T64n)

Offset

Register	Offset
PM0_T64n	270h
PM1_T64n	670h

Function

MAC Transmit 64-Octet Packet Counter Register (etherStatsPkts64OctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	T64n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	T64n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	T64n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T64n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 T64n	Incremented for each 64-octet frame transmitted, good or bad.

53.4.6.9.61 Port MAC a Transmit 65 to 127-Octet Packet Counter Register (etherStatsPkts65to127OctetsN) (PM0_T127n - PM1_T127n)

Offset

Register	Offset
PM0_T127n	278h
PM1_T127n	678h

Function

MAC Transmit 65 to 127-Octet Packet Counter Register (etherStatsPkts65to127OctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	T127n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	T127n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	T127n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T127n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 T127n	Incremented for each 65 to 127-octet frame transmitted, good or bad.

53.4.6.9.62 Port MAC a Transmit 128 to 255-Octet Packet Counter Register (etherStatsPkts128to255OctetsN) (PM0_T255n - PM1_T255n)

Offset

Register	Offset
PM0_T255n	280h
PM1_T255n	680h

Function

MAC Transmit 128 to 255-Octet Packet Counter Register (etherStatsPkts128to255OctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	T255n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	T255n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	T255n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T255n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 T255n	Incremented for each 128 to 255-octet frame transmitted, good or bad.

53.4.6.9.63 Port MAC a Transmit 256 to 511-Octet Packet Counter Register (etherStatsPkts256to511OctetsN) (PM0_T511n - PM1_T511n)

Offset

Register	Offset
PM0_T511n	288h
PM1_T511n	688h

Function

MAC Transmit 256 to 511-Octet Packet Counter Register (etherStatsPkts256to511OctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	T511n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	T511n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	T511n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T511n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 T511n	Incremented for each 256 to 511-octet frame transmitted, good or bad.

53.4.6.9.64 Port MAC a Transmit 512 to 1023-Octet Packet Counter Register (etherStatsPkts512to1023OctetsN) (PM0_T1023n - PM1_T1023n)

Offset

Register	Offset
PM0_T1023n	290h
PM1_T1023n	690h

Function

MAC Transmit 512 to 1023-Octet Packet Counter Register (etherStatsPkts512to1023OctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	T1023n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	T1023n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	T1023n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T1023n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 T1023n	Incremented for each 512 to 1023-octet frame transmitted, good or bad.

53.4.6.9.65 Port MAC a Transmit 1024 to 1522-Octet Packet Counter Register (etherStatsPkts1024to1522OctetsN) (PM0_T1522n - PM1_T1522n)

Offset

Register	Offset
PM0_T1522n	298h
PM1_T1522n	698h

Function

MAC Transmit 1024 to 1522-Octet Packet Counter Register (etherStatsPkts1024to1522OctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	T1522n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	T1522n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	T1522n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T1522n															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 T1522n	Incremented for each 1024- to 1522-octet frame transmitted, good or bad.

53.4.6.9.66 Port MAC a Transmit 1523 to TX_MTU-Octet Packet Counter Register (etherStatsPkts1523toMaxOctetsN) (PM0_T1523Xn - PM1_T1523Xn)

Offset

Register	Offset
PM0_T1523Xn	2A0h
PM1_T1523Xn	6A0h

Function

MAC Transmit 1523- to TX_MTU-Octet Packet Counter Register (etherStatsPkts1523toMaxOctetsN)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	T1523Xn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	T1523Xn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	T1523Xn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T1523Xn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 T1523Xn	Incremented for each 1523-octet frame and larger (up to the maximum frame length specified in register PM*a*_MAXFRM[TX_MTU]) transmitted, good or bad.

53.4.6.9.67 Port MAC a Transmit Control Packet Counter Register (PM0_TCNPn - PM1_TCNPn)

Offset

Register	Offset
PM0_TCNPn	2C0h
PM1_TCNPn	6C0h

Function

MAC Transmit Control Packet Counter Register

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TCNPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TCNPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TCNPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TCNPn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TCNPn	Incremented for each valid control packet transmitted (type 0x8808) but not for PAUSE packets

53.4.6.9.68 Port MAC a Transmit Deferred Packet Counter Register(aFramesWithDeferredXmissions) (PM0_TDFRn - PM1_TDFRn)

Offset

Register	Offset
PM0_TDFRn	2D0h
PM1_TDFRn	6D0h

Function

MAC Transmit Deferred Packet Counter Register(aFramesWithDeferredXmissions)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TDFRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TDFRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TDFRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TDFRn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TDFRn	Increments for successful transmissions, without retransmits, that were deferred (half-duplex only).

53.4.6.9.69 Port MAC a Transmit Multiple Collisions Counter Register(aMultipleCollisionFrames) (PM0_TMCOLn - PM1_TMCOLn)

Offset

Register	Offset
PM0_TMCOLn	2D8h
PM1_TMCOLn	6D8h

Function

MAC Transmit Multiple Collisions Counter Register(aMultipleCollisionFrames)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TMCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TMCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TMCOLn	Increments for successful transmission after more than one retransmission (half-duplex only).

53.4.6.9.70 Port MAC a Transmit Single Collision Counter(aSingleCollisionFrames) Register (PM0_TSCOLn - PM1_TSCOLn)

Offset

Register	Offset
PM0_TSCOLn	2E0h
PM1_TSCOLn	6E0h

Function

MAC Transmit Single Collision Counter Register(aSingleCollisionFrames)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TSCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TSCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TSCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TSCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TSCOLn	Increments for successful transmission after one retransmission (half-duplex only).

53.4.6.9.71 Port MAC a Transmit Late Collision Counter(aLateCollisions) Register (PM0_TLCOLn - PM1_TLCOLn)

Offset

Register	Offset
PM0_TLCOLn	2E8h
PM1_TLCOLn	6E8h

Function

MAC Transmit Late Collision Counter Register(aLateCollisions)

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TLCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TLCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TLCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TLCOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TLCOLn	Late collision occurred. Frame corrupted / discarded (half-duplex only) For MII or RMII, late collisions may be detected if collision occurs after 60B after start of Ethernet packet (52B after SFD, for 7B preamble). For RGMII, late collisions may be detected if collision occurs after 49B after start of Ethernet packet.

53.4.6.9.72 Port MAC a Transmit Excessive Collisions Counter Register (PM0_TECOLn - PM1_TECOLn)

Offset

Register	Offset
PM0_TECOLn	2F0h
PM1_TECOLn	6F0h

Function

MAC Transmit Excessive Collisions Counter Register

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	TECOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	TECOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TECOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TECOLn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 TECOLn	Excessive collisions occurred. Frame was discarded (half-duplex only)

53.4.6.9.73 Port MAC a Interface Mode Control Register (PM0_IF_MODE - PM1_IF_MODE)

Offset

Register	Offset
PM0_IF_MODE	300h
PM1_IF_MODE	700h

Function

MAC Interface Mode Control Register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SSP		CLK_STOP	Reserved		Reserved		HD	Reserved	M10	REVMII	IFMODE			
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-16 —	Reserved
15 —	Reserved
14-13 SSP	Set Speed Set speed for RGMII mode **NOTE** RGMII receive clock must be active for speed changes to take effect. 00b - 100 Mbps 01b - 10 Mbps 10b - 1 Gbps 11b - reserved
12 CLK_STOP	Clock Stop RGMII Transmit Clock is Stoppable 0b - Not stoppable. RGMII transmit clock is not stoppable during low power idle 1b - Stoppable. RGMII transmit clock is stoppable during low power idle
11-10 —	Reserved
9-7 —	Reserved
6 HD	Half-duplex Half-duplex operation for MII/RMII 0b - full duplex 1b - half duplex. As per 802.3 annex 31B, PAUSE frames are only supported when the link is configured for full-duplex operation. When the link is set to operate in the half duplex mode (HD set to 1), the MAC must be configured to ignore received PAUSE frames by setting PMa_COMMAND_CONFIG[PAUSE_IGN] to 1. On transmit, upper layer processing functions must be configured in such a way that they won't generate requests to generate PAUSE frame.
5 —	Reserved
4 M10	0 - 100 Mbps RMII/MII 1 - 10 Mbps RMII/MII (MII speed select is valid only in case of RevMII)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - 100 Mbps 1b - 10 Mbps
3 REVMII	Reverse Mode 0b - Reverse mode disabled - port is in MAC mode 1b - Reverse mode enabled - port is in PHY mode
2-0 IFMODE	Interface mode 001 - MII mode 010 - GMII mode 011 - RMII mode 100 - RGMII mode All other values reserved.

53.4.6.9.74 Port MAC Merge Control and Status Register (MAC_MERGE_MMCSR)

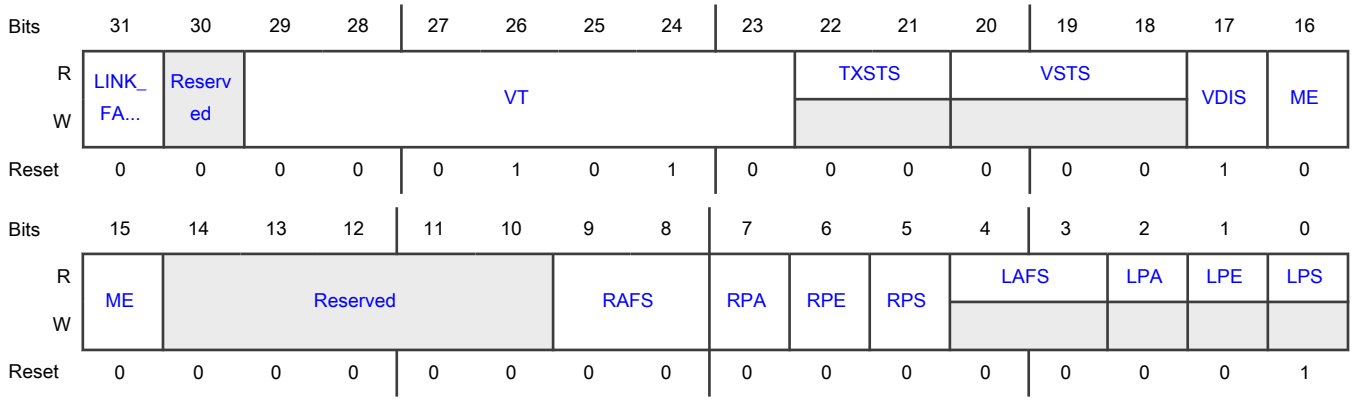
Offset

Register	Offset
MAC_MERGE_MMCSR	800h

Function

The individual fields in this register correspond to the capabilities necessary to claim conformance to the IEEE 802.3-2018 handling of interspersed express traffic.

Diagram



Fields

Field	Function
31 LINK_FAIL	Link Fail Software may set this bit to indicate to the mac merge layer that there has been a failure on the link. If set, the mac merge layer disables the preemption of frames. The mac merge layer will resume preempting frames if software subsequently sets this bit to 0.
30 —	Reserved
29-23 VT	Verify Time The value of this field defines the nominal wait time between verification attempts in milliseconds. Valid range is 1 to 127 inclusive. The default value is 10.
22-21 TXSTS	Merge status This read-only status field provides the state of the mac merge sublayer transmit status as defined in IEEE Std 802.3-2018 Clause 99. 00 Transmit preemption is inactive 01 Transmit preemption is active 10 Reserved 11 Reserved
20-18 VSTS	Verify status This read-only status field provides the state of the mac merge sublayer with respect to verification as defined in IEEE Std 802.3-2018 Clause 99. 000 Verification is disabled 001 Reserved 010 Verification is in progress 011 Verification was successful 100 Verification failed 101 Verification is in an undefined state 110 Reserved 111 Reserved
17 VDIS	Verify disabled Once enabled, the port must perform the verify process described in IEEE Std 802.3-2018 Clause 99 with the remote port. Verification should only be disabled for a closed system. In most systems, software should set this bit to a zero when setting MMCSR[ME]. Note: On start-up, software should wait for the verification process to complete (MMCSR[VSTS]=011) before initiating traffic.
16-15	Merge enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
ME	0 Frame preemption is not enabled 1: Frame preemption supported, transmit preempts frames on any byte boundary post the MinimumFragmentSize 2: Frame preemption supported, transmit preempts frames on 4B boundaries post the MinimumFragmentSize Preemption is disabled by default. Software must set this bit in order for preemption to occur. When 0, the mac merge layer will not preempt PMAC (MAC 1) frames. When set to 1, PMAC frames may be preempted by EMAC (MAC 0) frames.
14-10 —	Reserved
9-8 RAFS	Remote additional fragment size This writeable status field indicates the minimum size of non-final fragments supported by the remote port. The value is expressed in units of 64 octets of additional frame length. The minimum non-final fragment size is ((RAFS + 1) x 64 octets). This value indicates to the local preemption control block the smallest sized fragments that can be sent on TX. Can be used by software to store the value received by the Link Layer Discovery Protocol (LLDP) when receiving an Additional Ethernet Capabilities TLV.
7 RPA	Remote preemption active This writeable status bit indicates whether preemption is active for the remote port. Can be used by software to store the value received by the Link Layer Discovery Protocol (LLDP) when receiving an Additional Ethernet Capabilities TLV.
6 RPE	Remote preemption enabled This writeable status bit indicates whether preemption has been enabled for the remote port. Can be used by software to store the value received by the Link Layer Discovery Protocol (LLDP) when receiving an Additional Ethernet Capabilities TLV.
5 RPS	Remote preemption supported This writeable status bit indicates whether preemption capability is supported for the remote port. Can be used by software to store the value received by the Link Layer Discovery Protocol (LLDP) when receiving an Additional Ethernet Capabilities TLV.
4-3 LAFS	Local additional fragment size This read-only status field indicates the minimum size of non-final fragments supported by this port. The value is expressed in units of 64 octets of additional frame length. The minimum non-final fragment size is ((LAFS + 1) x 64 octets). This implementation is designed to be able to receive a minimum 64 octet fragment size. Write capability was added to this field to permit software to override the value sent to with the system. Can be used by the Link Layer Discovery Protocol (LLDP) when composing a Additional Ethernet Capabilities TLV.
2 LPA	Local preemption active

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This read-only status bit indicates whether preemption is active for this port. This bit will be set if preemption is both enabled and has completed the verification process. Can be used by the Link Layer Discovery Protocol (LLDP) when composing a Additional Ethernet Capabilities TLV. **NOTE** This bit may also read as 1 when preemption is enabled, but verification has failed. When composing a Additional Ethernet Capabilities TLV, this bit should qualified with successful verification status in VSTS. For example, LPA' = LPA && (VSTS == 3).
1 LPE	Local preemption enabled This read-only status bit indicates whether preemption has been enabled for this port. This bit is a reflection of the MMCSR[ME] control bit. Can be used by the Link Layer Discovery Protocol (LLDP) when composing a Additional Ethernet Capabilities TLV.
0 LPS	Local preemption supported This read-only status bit indicates whether preemption capability is supported for this port. This implementation supports preemption, so the value of the bit defaults to 1. Can be used by the Link Layer Discovery Protocol (LLDP) when composing a Additional Ethernet Capabilities TLV.

53.4.6.9.75 Port MAC Merge Frame Assembly Error Count Register (MAC_MERGE_MMFAECR)

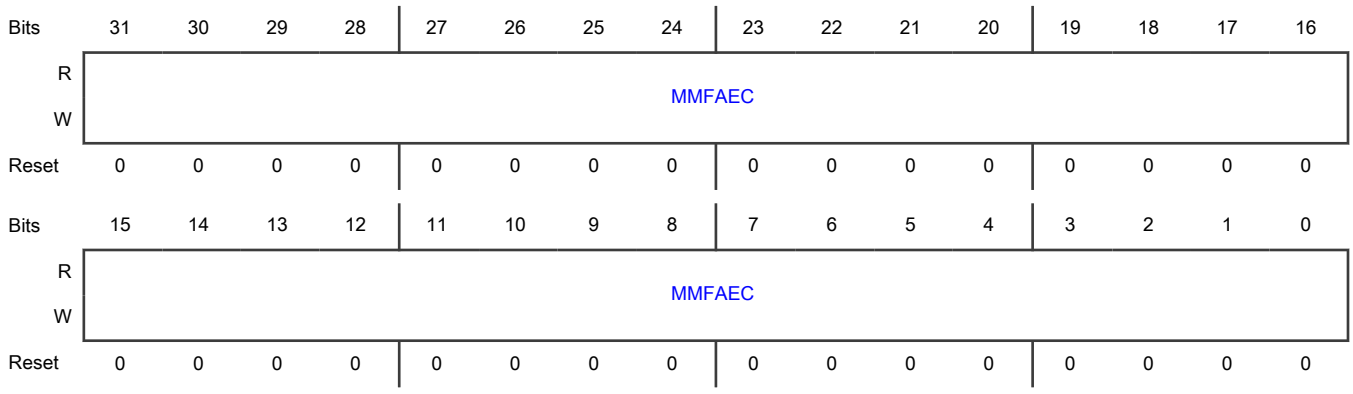
Offset

Register	Offset
MAC_MERGE_MMFAECR	808h

Function

A count of MAC frames with reassembly errors.

Diagram



Fields

Field	Function
31-0 MMFAEC	A count of MAC frames with reassembly errors.

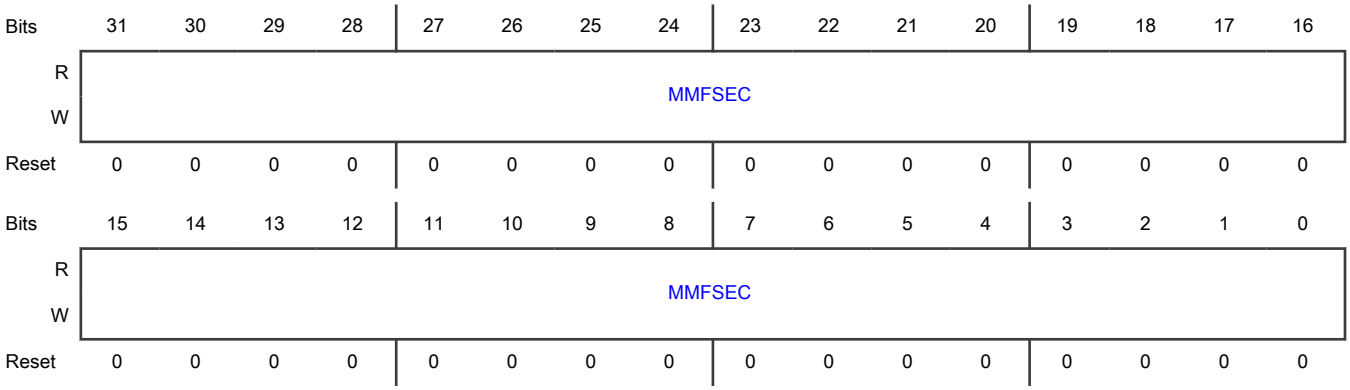
53.4.6.9.76 Port MAC Merge Frame SMD Error Count Register (MAC_MERGE_MMFSECR)

Offset

Register	Offset
MAC_MERGE_MMFSECR	80Ch

Function
A count of received MAC frames / MAC frame fragments rejected due to unknown SMD value or arriving with an SMD-C when no frame is in progress.

Diagram



Fields

Field	Function
31-0 MMFSEC	A count of received MAC frames / MAC frame fragments rejected due to unknown SMD value or arriving with an SMD-C when no frame is in progress.

53.4.6.9.77 Port MAC Merge Frame Assembly OK Count Register (MAC_MERGE_MMFAOCR)

Offset

Register	Offset
MAC_MERGE_MMFAOCR	810h

Function
A count of MAC frames that were successfully reassembled and delivered to the MAC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MMFAOC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MMFAOC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MMFAOC	A count of MAC frames that were successfully reassembled and delivered to the MAC.

53.4.6.9.78 Port MAC Merge Fragment Count RX Register (MAC_MERGE_MMFCRXR)

Offset

Register	Offset
MAC_MERGE_MMFCRXR	814h

Function
A count of the number of additional mPackets received due to preemption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MMFCRX															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MMFCRX															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MMFCRX	A count of the number of additional mPackets received due to preemption.

53.4.6.9.79 Port MAC Merge Fragment Count TX Register (MAC_MERGE_MMFCTXR)

Offset

Register	Offset
MAC_MERGE_MMFCTXR	818h

Function

A count of the number of additional mPackets transmitted due to preemption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MMFCTX															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MMFCTX															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MMFCTX	A count of the number of additional mPackets transmitted due to preemption.

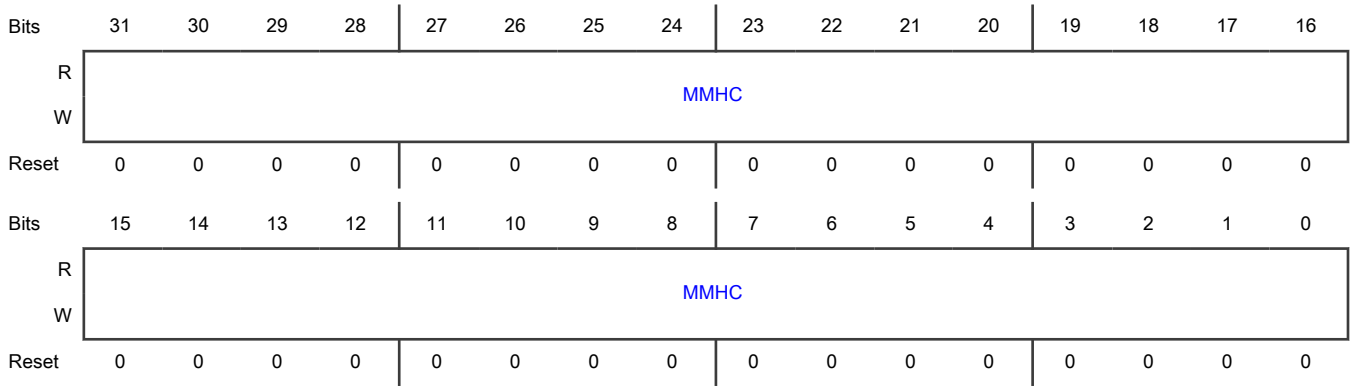
53.4.6.9.80 Port MAC Merge Hold Count Register (MAC_MERGE_MMHCR)

Offset

Register	Offset
MAC_MERGE_MMHCR	81Ch

Function
A count of the number of times the variable hold transitions from false to true.

Diagram



Fields

Field	Function
31-0 MMHC	A count of the number of times the variable hold transitions from false to true.

53.4.6.9.81 Port external MDIO configuration register (PEMDIOCR)

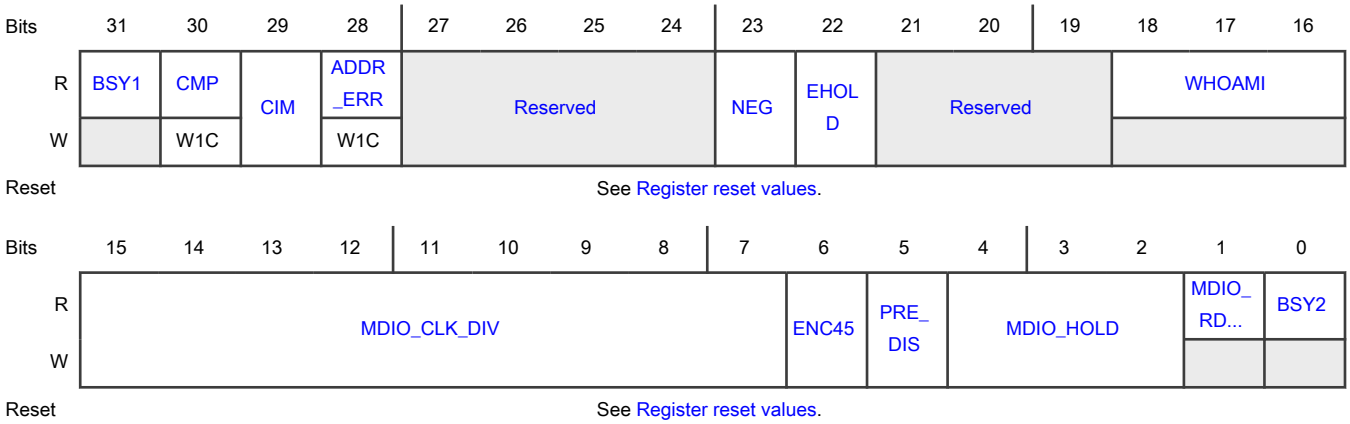
Offset

Register	Offset
PEMDIOCR	C00h

Function

Port external MDIO configuration register for access to the off-chip PHY.

Diagram



Register reset values

Register	Reset value
PEMDIOCR	ENETC0_ETH_MAC_PORT: 0004_1448h SW0_ETH_MAC_PORT0: 0000_1448h SW0_ETH_MAC_PORT1: 0001_1448h SW0_ETH_MAC_PORT2: 0002_1448h SW0_ETH_MAC_PORT3: 0003_1448h

Fields

Field	Function
31 BSY1	Busy 1 Indicates whether MDIO is busy. 0b - An MDIO transaction is not occurring; software may access other MDIO registers. 1b - An MDIO transaction is occurring.
30 CMP	MDIO Command Completion Bit is cleared by writing 1b. 0b - An MDIO command completion did not occur. 1b - An MDIO command completion occurred.
29 CIM	MDIO Command Completion Interrupt Mask 0b - Masked 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 ADDR_ERR	Address Error 0b - Normal 1b - Error. An access control violation has occurred. The request address used does not match the MDIO PHY's address (clause 22) or MDIO port address (clause 45) assigned.
27-24 —	Reserved
23 NEG	Negative Edge 0b - normal operation - positive edge 1b - MDIO is driven by master on MDC negative edge (default for external MDIOs)
22 EHOLD	Extended HOLD **NOTE** For Extended operation, MDIO hold time for MDIO_HOLD is specified as follows: 000b - 1 NETC cycle 001b - 9 NETC cycles 010b - 17 NETC cycles 011b - 25 NETC cycles 100b - 33 NETC cycles 101b - 41 NETC cycles 110b - 49 NETC cycles 111b - 57 NETC cycles 0b - Normal operation, MDIO hold time is as specified in PEMDIOCR[MDIO_HOLD]. 1b - Extended operation
21-19 —	Reserved
18-16 WHOAMI	Returns the link ID
15-7 MDIO_CLK_DIV	MDIO Clock Divisor A value of 5-511 can be set to configure the MDIO clock frequency relative to NETC clock. Ratio is (2*MDIO_CLK_DIV)+1. Setting the divisor to 0 disables MDC. The minimum value which should be written to this bit field is 2, which corresponds to a clock divide ratio of 5. **NOTE** The default is 40. For external MDIO Ethernet management interface, the default is 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 ENC45	Enable Clause 45 Support 0b - Clause 22 transactions are used. 1b - Clause 45 transactions are used.
5 PRE_DIS	MDIO Preamble Disable 0b - Generation of MDIO preamble is enabled (default operation). 1b - Generation of MDIO preamble is disabled.
4-2 MDIO_HOLD	MDIO Hold Time 000b - 1 NETC cycle 001b - 3 NETC cycles 010b - 5 NETC cycles (default - recommended value) 011b - 7 NETC cycles 100b - 9 NETC cycles 101b - 11 NETC cycles 110b - 13 NETC cycles 111b - 15 NETC cycles
1 MDIO_RD_ER	MDIO Read (and write) Error 0b - No error on last MDIO transaction (read or write). 1b - An error was detected on the last MDIO transaction (read or write). Errors on internal MDIO accesses can be triggered by an access to an invalid device, or by a write to a shared on-die PHY device that has not been locked.
0 BSY2	Busy 2 (same as bit 31) Indicates whether MDIO is busy. 0b - An MDIO transaction is not occurring; software may access other MDIO registers. 1b - An MDIO transaction is occurring.

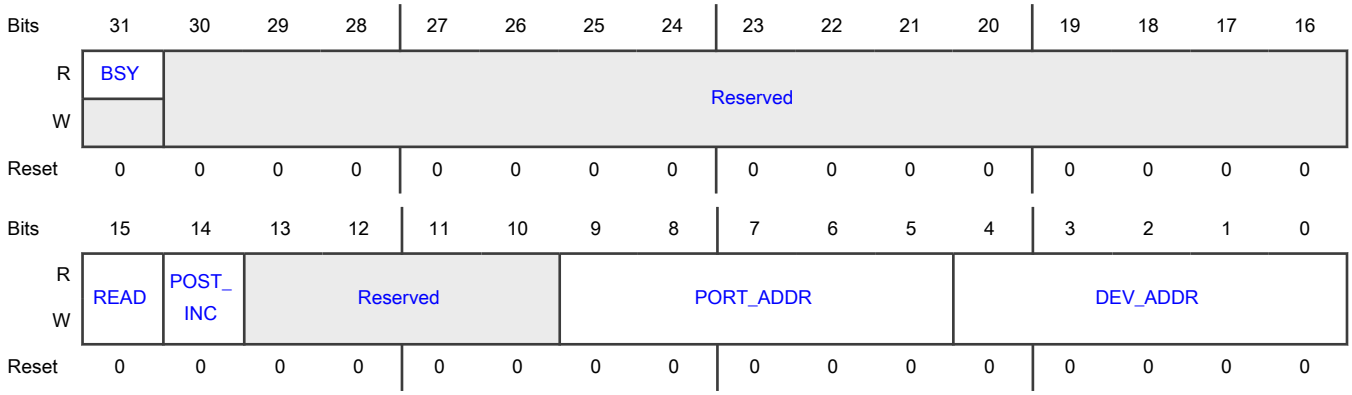
53.4.6.9.82 Port external MDIO interface control register (PEMDIOICR)

Offset

Register	Offset
PEMDIOICR	C04h

Function
This is the port external MDIO interface control register.

Diagram



Fields

Field	Function
31 BSY	MDIO busy 0 An MDIO transaction is not occurring; software may access other MDIO registers. 1 An MDIO transaction is occurring.
30-16 —	Reserved
15 READ	MDIO read initiation. 1 A normal MDIO read transaction is initiated. Self-clearing once transaction is complete.
14 POST_INC	MDIO read with address post-increment initiation. Self-clearing once transaction is complete. 0 (default operation) 1 An MDIO read with Clause 45 address post-increment transaction is initiated (Post-incrementation is performed in the PHY internal address register.)
13-10 —	Reserved
9-5 PORT_ADDR	5-bit MDIO port address (Clause 45) / PHY address (Clause 22)
4-0 DEV_ADDR	5-bit MDIO device address (Clause 45) / register address (Clause 22)

53.4.6.9.83 Port external MDIO interface data register (PEMDIOIDR)

Offset

Register	Offset
PEMDIOIDR	C08h

Function
This is the port external MDIO interface data register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MDIO_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 MDIO_DATA	16-bit MDIO data. Writing to this register initiates a write transaction to the PHY. For proper operation, the MDIO_CTL register must have been initialized first. The PEMDIOCR[BUSY] status bit is set immediately and cleared when the write transaction has completed. Reading this register returns data read from the PHY register specified in PEMDIOICR after a read transaction has completed. Read transactions are initiated by writing a 1 to PEMDIOICR[READ] or PEMDIOICR[POST_INC]. Read data is invalid if PEMDIOCR[BUSY] is set.

53.4.6.9.84 Port external MDIO register address register (PEMDIORAR)

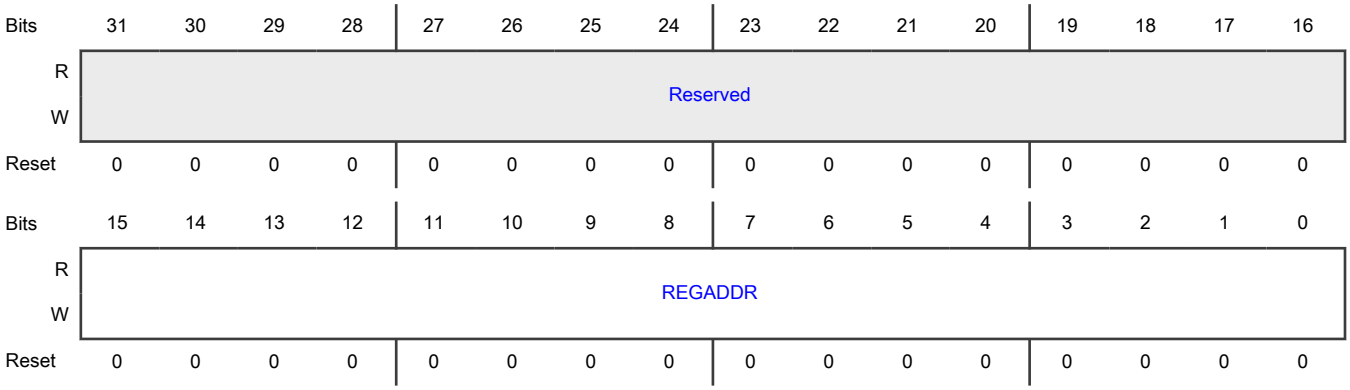
Offset

Register	Offset
PEMDIORAR	C0Ch

Function

This is the port external MDIO register address register, and is used to communicate with an attached Clause 45 PHY.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 REGADDR	MDIO PHY register address. MDIO PHY register address. Address of the register within the Clause 45 PHY device from which data is to be read or to which data is to be written. Writing to this register initiates an address-write transaction to set the PHY's internal address register to the value specified in PEMDIORAR[REGADDR]. For proper operation, the PEMDIOICR register must have been initialized prior to writing to this register.

53.4.6.9.85 Port external MDIO status register (PEMDIOSR)

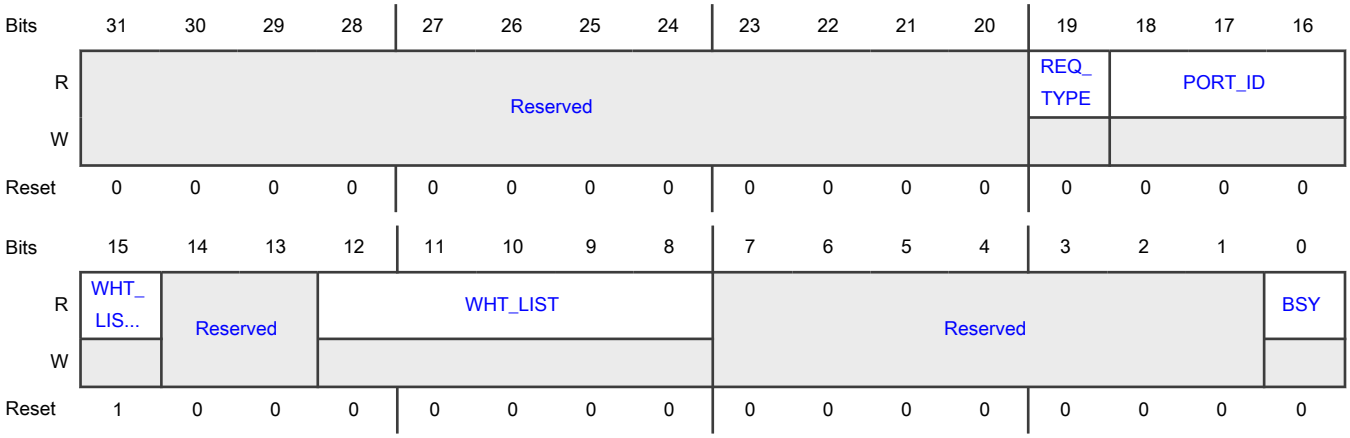
Offset

Register	Offset
PEMDIOSR	C10h

Function

This is the port external MDIO status register.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 REQ_TYPE	Port ID Returns the request type of the active virtual port. 0 MDIO request 1 PHY polling
18-16 PORT_ID	Port ID Returns the port ID of the active virtual port. Matches the value PEMDIOCR[WHOAMI] of active port.
15 WHT_LIST_ENA	PHY white list enable Returns the PHY white list enable of the active virtual port.
14-13 —	Reserved
12-8 WHT_LIST	PHY white list Returns the PHY white list address of the active virtual port.
7-1 —	Reserved
0 BSY	Global MDIO busy 0 An MDIO transaction is not occurring; software may access other MDIO registers. 1 An MDIO transaction is occurring.

53.4.6.9.86 PHY status configuration register (PPSCR)

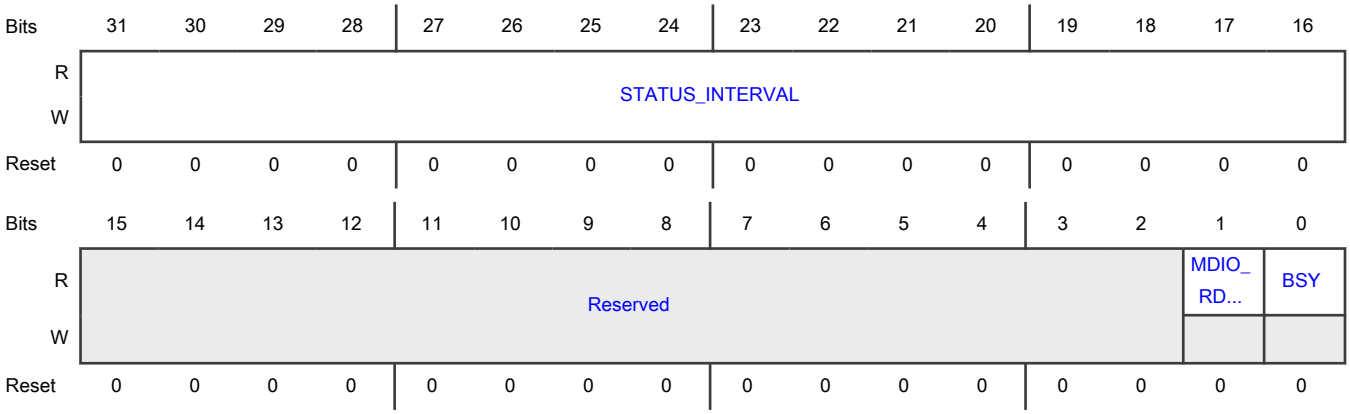
Offset

Register	Offset
PPSCR	C20h

Function

This is the PHY status configuration register.

Diagram



Fields

Field	Function
31-16 STATUS_INTERVAL	PHY status read interval Amount of time to wait between PHY status reads, in units of 1-2 ms. A value of 0 indicates auto PHY automatic status reads are disabled.
15-2 —	Reserved
1 MDIO_RD_ER	MDIO read error 1 The last read transaction received no response from a PHY; any data read should be considered invalid. (for example, The PHY address does not match any PHY available on the MDIO bus.)
0 BSY	MDIO busy 0 An MDIO transaction is not occurring; software may access other MDIO registers. 1 An MDIO transaction is occurring.

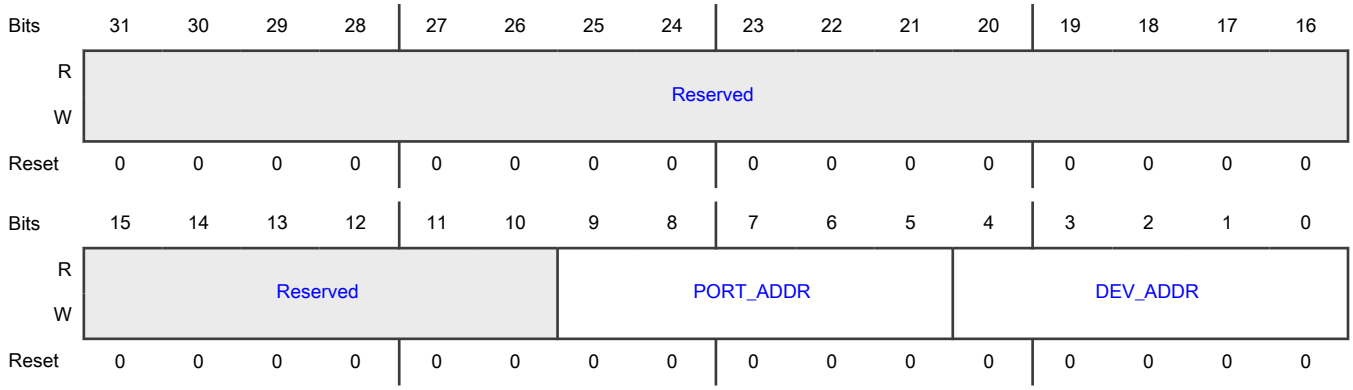
53.4.6.9.87 Port PHY status control register (PPSCTRLR)

Offset

Register	Offset
PPSCTRLR	C24h

Function
This is the port PHY status control register.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-5 PORT_ADDR	5-bit MDIO port address (Clause 45) / PHY address (Clause 22)
4-0 DEV_ADDR	5-bit MDIO device address (Clause 45) / register address (Clause 22)

53.4.6.9.88 Port PHY status data register (PPSDR)

Offset

Register	Offset
PPSDR	C28h

Function
This is the port PHY status data register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURR_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MDIO_DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 CURR_CNT	Current count Current value of status interval counter.
15-0 MDIO_DATA	16-bit MDIO data Reading this register returns data read from the PHY register specified in PHY_STATUS_CTL after a read transaction has completed. Read transactions are initiated automatically based on the ms counter setting. Read data is invalid if PHY_STATUS_CFG[BUSY] is set.

53.4.6.9.89 Port PHY status register address register (PPSRAR)

Offset

Register	Offset
PPSRAR	C2Ch

Function

This is the port PHY status register address register, and is used to communicate with an attached Clause 45 PHY.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REGADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 REGADDR	MDIO PHY register address. Address of the register within the Clause 45 PHY device from which data is to be read.

53.4.6.9.90 Port PHY status event register (PPSER)

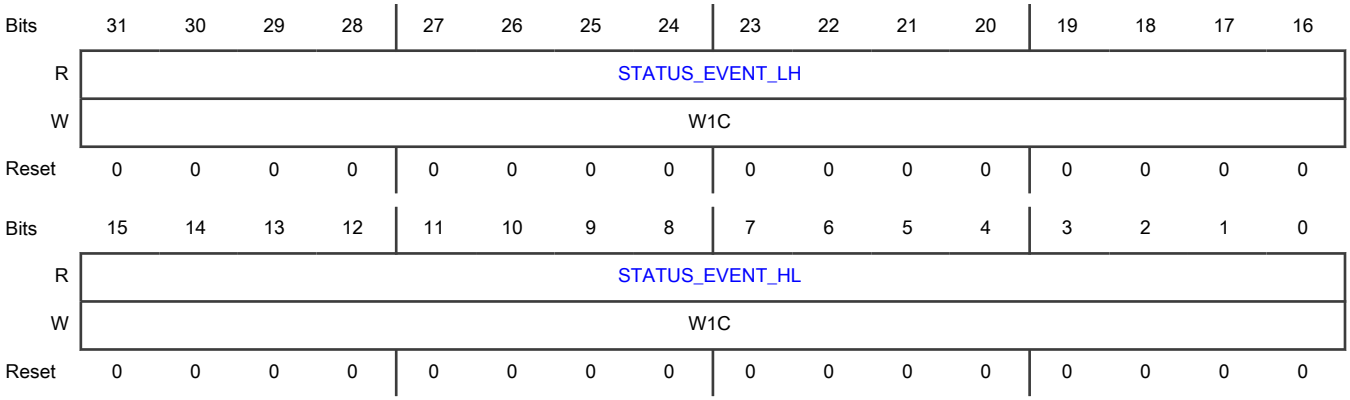
Offset

Register	Offset
PPSER	C30h

Function

This is the port PHY status event register, and is used to detect transitions in the data bits. Used in conjunction with the mask register to signal interrupts.

Diagram



Fields

Field	Function
31-16 STATUS_EVENT_LH	Status event low-to-high. Set to 1 if a 0->1 transition on a corresponding data bit has occurred. Write 1 to clear.
15-0 STATUS_EVENT_HL	Status event high-to-low. Set to 1 if a 1->0 transition on a corresponding data bit has occurred. Write 1 to clear.

53.4.6.9.91 Port PHY status mask register (PPSMR)

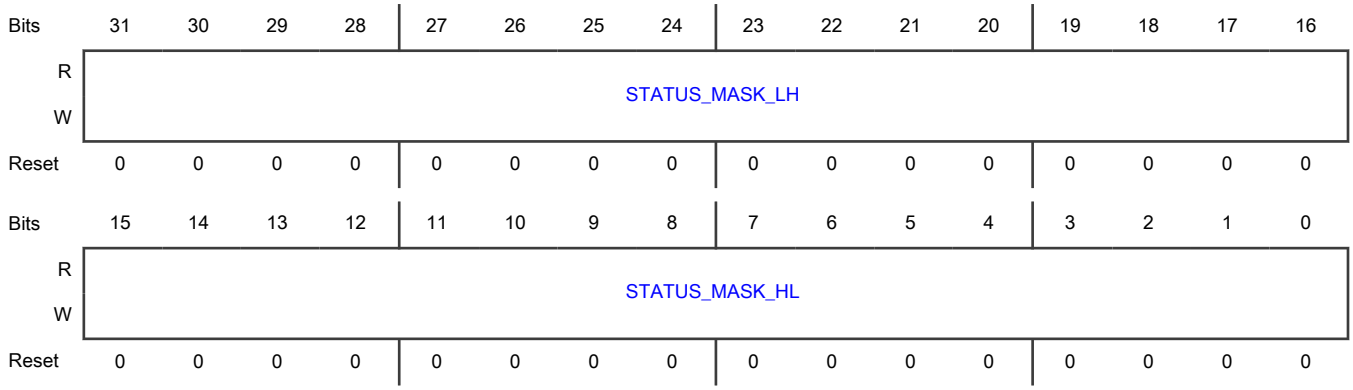
Offset

Register	Offset
PPSMR	C34h

Function

This is the port PHY status mask register, and is used in conjunction with the event register to signal interrupts.

Diagram



Fields

Field	Function
31-16 STATUS_MASK_LH	Status mask low-to-high. If set to 1, assert an interrupt if the corresponding event bit is set.
15-0 STATUS_MASK_HL	Status high-to-low mask. If set to 1, assert an interrupt if the corresponding event bit is set.

53.4.6.10 Pseudo MAC port register descriptions

This section describes the pseudo MAC port registers. The pseudo MAC registers exists whenever two ENETC and switch instances are connected together.

53.4.6.10.1 Pseudo_MAC_port memory map

Instance	Address space	Offset
ENETC1_PSEUDO_MAC_PORT	PCI_F4_BAR_0	60B5_5000h
SW0_PSEUDO_MAC_PORT4	PCI_F2_BAR_0	60A1_5000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Port pseudo MAC status register (PPMSR)	32	R	0000_0101h
10h	Port pseudo MAC configuration register (PPMCR)	32	RW	0001_0000h
80h - 84h	Port pseudo MAC receive octets counter (PPMROCR0 - PPMROCR1)	32	R	0000_0000h
88h - 8Ch	Port pseudo MAC receive unicast frame counter register (PPMRUFCR0 - PPMRUFCR1)	32	R	0000_0000h
90h - 94h	Port pseudo MAC receive multicast frame counter register (PPMRMFCR0 - PPMRMFCR1)	32	R	0000_0000h
98h - 9Ch	Port pseudo MAC receive broadcast frame counter register (PPMRBFCR0 - PPMRBFCR1)	32	R	0000_0000h
C0h - C4h	Port pseudo MAC transmit octets counter (PPMTOCR0 - PPMTOCR1)	32	R	0000_0000h
C8h - CCh	Port pseudo MAC transmit unicast frame counter register (PPMTUFCR0 - PPMTUFCR1)	32	R	0000_0000h
D0h - D4h	Port pseudo MAC transmit multicast frame counter register (PPMTMFCR0 - PPMTMFCR1)	32	R	0000_0000h
D8h - DCh	Port pseudo MAC transmit broadcast frame counter register (PPMTBFCR0 - PPMTBFCR1)	32	R	0000_0000h

53.4.6.10.2 Port pseudo MAC status register (PPMSR)

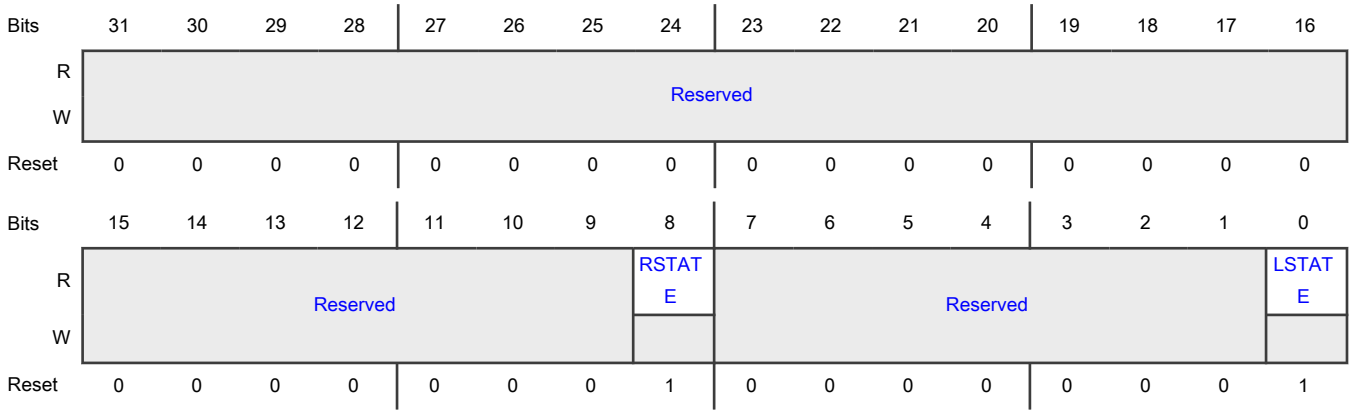
Offset

Register	Offset
PPMSR	0h

Function

This is the port pseudo MAC status register.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 RSTATE	Remote link end's state 0 - Link is down 1 - Link is up **NOTE** The operational state is always "Link is up" for the pseudo link. RSTATE will always return 1.
7-1 —	Reserved
0 LSTATE	Local link end's state 0 - Link is down 1 - Link is up **NOTE** The operational state is always "Link is up" for the pseudo link. LSTATE will always return 1.

53.4.6.10.3 Port pseudo MAC configuration register (PPMCR)

Offset

Register	Offset
PPMCR	10h

Function

This is the port pseudo MAC configuration register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC1_PSEUDO_MAC_PORT	PPMCR	—
SW0_PSEUDO_MAC_PORT4	—	PPMCR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															TXPAD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 TXPAD	Transmit Padding When set, HTA transmit will pad the frame to a minimum of 60 bytes with pad bytes of zeros after performing frame modifications, if any. It will then append the calculated FCS to the frame.
15-0 —	Reserved

53.4.6.10.4 Port pseudo MAC receive octets counter (PPMROCR0 - PPMROCR1)

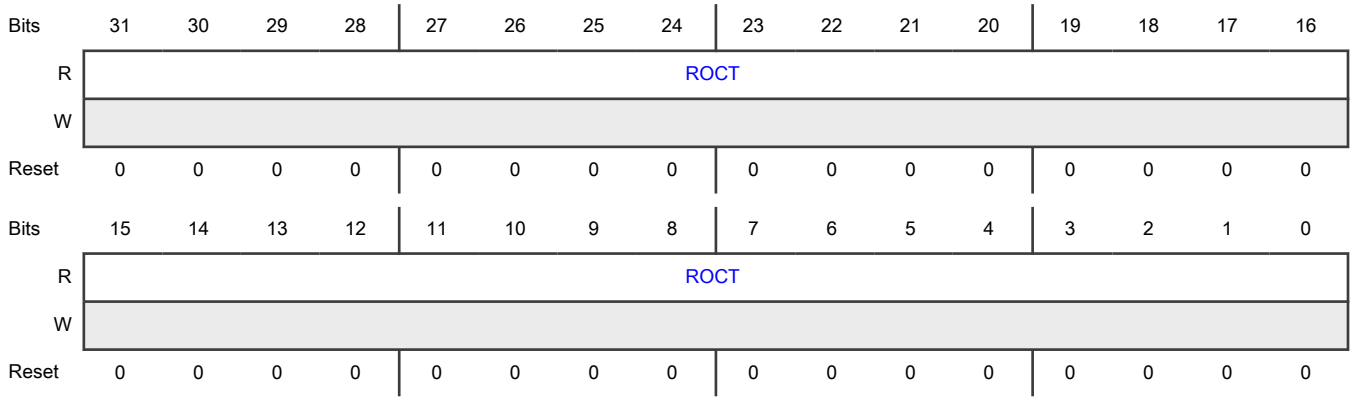
Offset

Register	Offset
PPMROCR0	80h
PPMROCR1	84h

Function

This is the port pseudo MAC receive octets counter. Register xCR0 corresponds to the lower 32b of the counter. xCR1 corresponds to the upper 32b. xCR0 must be read first, followed by a read to xCR1.

Diagram



Fields

Field	Function
31-0 ROCT	Incremented for each octet received (on the link) (that is, Ethernet header, payload, pad and FCS).

53.4.6.10.5 Port pseudo MAC receive unicast frame counter register (PPMRUFCR0 - PPMRUFCR1)

Offset

Register	Offset
PPMRUFCR0	88h
PPMRUFCR1	8Ch

Function

This is the port pseudo MAC receive unicast frame counter register. Register xCR0 corresponds to the lower 32b of the counter. xCR1 corresponds to the upper 32b. xCR0 must be read first, followed by a read to xCR1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RUCA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RUCA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RUCA	Incremented for each valid frame received (on the link) in which bit 0 of the destination address was 0.

53.4.6.10.6 Port pseudo MAC receive multicast frame counter register (PPMRMFCR0 - PPMRMFCR1)

Offset

Register	Offset
PPMRMFCR0	90h
PPMRMFCR1	94h

Function

This is the port pseudo MAC receive multicast frame counter register. Register xCR0 corresponds to the lower 32b of the counter. xCR1 corresponds to the upper 32b. xCR0 must be read first, followed by a read to xCR1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RMCA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RMCA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RMCA	Incremented for each frame received (on the link) in which bit 0 of the destination address was 1 but not the broadcast address (all bits set to 1).

53.4.6.10.7 Port pseudo MAC receive broadcast frame counter register (PPMRBFCR0 - PPMRBFCR1)

Offset

Register	Offset
PPMRBFCR0	98h
PPMRBFCR1	9Ch

Function

This is the port pseudo MAC receive broadcast frame counter register. Register xCR0 corresponds to the lower 32b of the counter. xCR1 corresponds to the upper 32b. xCR0 must be read first, followed by a read to xCR1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RBCA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RBCA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RBCA	Incremented for each frame received (on the link) in which all bits of the destination address were 1.

53.4.6.10.8 Port pseudo MAC transmit octets counter (PPMTOCR0 - PPMTOCR1)

Offset

Register	Offset
PPMTOCR0	C0h
PPMTOCR1	C4h

Function

This is the port pseudo MAC transmit octets counter. Register xCR0 corresponds to the lower 32b of the counter. xCR1 corresponds to the upper 32b. xCR0 must be read first, followed by a read to xCR1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TOCT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TOCT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TOCT	Incremented for each octet transmitted (on the link) (that is, Ethernet header, payload, pad and FCS).

53.4.6.10.9 Port pseudo MAC transmit unicast frame counter register (PPMTUFCR0 - PPMTUFCR1)

Offset

Register	Offset
PPMTUFCR0	C8h
PPMTUFCR1	CCh

Function

This is the port pseudo MAC transmit unicast frame counter register. Register xCR0 corresponds to the lower 32b of the counter. xCR1 corresponds to the upper 32b. xCR0 must be read first, followed by a read to xCR1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TUCA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TUCA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TUCA	Incremented for each frame transmitted (on the link) in which bit 0 of the destination address was 0.

53.4.6.10.10 Port pseudo MAC transmit multicast frame counter register (PPMTMFCR0 - PPMTMFCR1)

Offset

Register	Offset
PPMTMFCR0	D0h
PPMTMFCR1	D4h

Function

This is the port pseudo MAC transmit multicast frame counter register. Register xCR0 corresponds to the lower 32b of the counter. xCR1 corresponds to the upper 32b. xCR0 must be read first, followed by a read to xCR1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMCA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMCA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMCA	Incremented for each frame transmitted (on the link) in which bit 0 of the destination address was 1 but not the broadcast address (all bits set to 1).

53.4.6.10.11 Port pseudo MAC transmit broadcast frame counter register (PPMTBFCR0 - PPMTBFCR1)

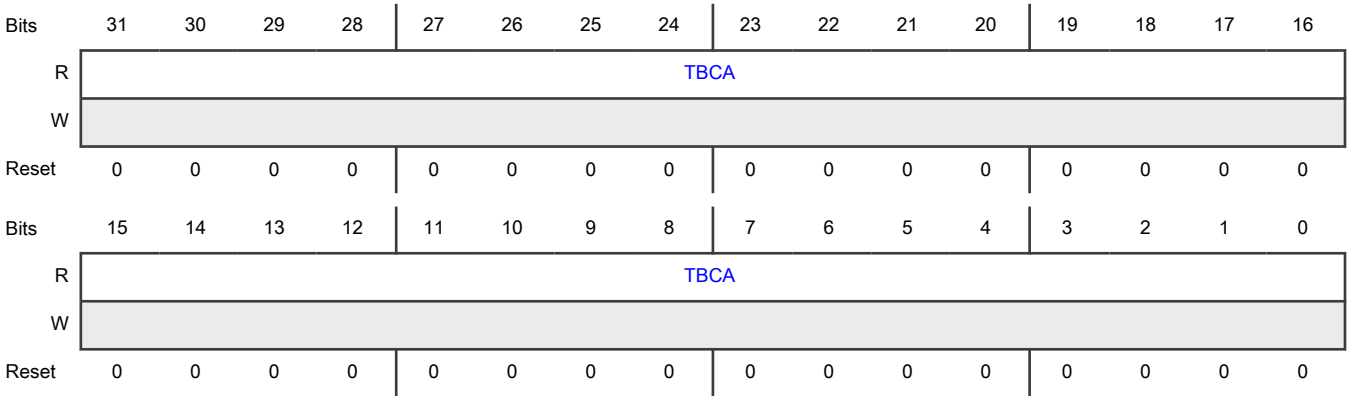
Offset

Register	Offset
PPMTBFCR0	D8h
PPMTBFCR1	DCh

Function

This is the port pseudo MAC transmit broadcast frame counter register. Register xCR0 corresponds to the lower 32b of the counter. xCR1 corresponds to the upper 32b. xCR0 must be read first, followed by a read to xCR1.

Diagram



Fields

Field	Function
31-0 TBCA	Incremented for each frame transmitted (on the link) in which all bits of the destination address were 1.

53.4.6.11 ENETC Station Interface register descriptions

This section describes ENETC station interface device registers for one or more base/physical functions.

There may be differences in actual implemented registers or register fields as well as reset values between functions that is, SI0 is the physical station interface while SI1+ are virtual station interfaces.

The physical station interface function's base address register value can be discovered from reading the PCIe Enhanced Allocation Base Address Register Equivalent Index 0 (EA BEI 0). Virtual station interfaces uses the PCIe Enhanced Allocation Base Address Register Equivalent Index 9 (EA BEI 9).

53.4.6.11.1 Station_Interface memory map

Instance	Address space	PCIe EA BEI offset
ENETC0_SI0	PCI_F3_BAR_0	60B0_0000h
ENETC1_SI0	PCI_F4_BAR_0	60B4_0000h
ENETC1_SI1	PCI_F4_BAR_9	60C1_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Station interface mode register (SIMR)	32	RW	See section
4h	Station interface status register (SISR)	32	R	0000_001Eh
18h	Station interface current time register 0 (SICTR0)	32	R	0000_0000h
1Ch	Station interface current time register 1 (SICTR1)	32	R	0000_0000h
20h	Station interface port capability register 0 (SIPCAPR0)	32	R	See section
24h	Station interface port capability register 1 (SIPCAPR1)	32	R	See section
30h	Station interface timer status register (SITSR)	32	R	0000_0000h
38h	Station interface receive BDR group control register (SIRBGCR)	32	RW	0000_0000h
40h	Station interface buffer cache attribute register (SIBCAR)	32	RW	0202_0202h
44h	Station interface message cache attribute register (SIMCAR)	32	RW	0002_0002h
48h	Station interface command cache attribute register (SICCAR)	32	RW	0202_0202h
80h	Station interface primary MAC address register 0 (SIPMAR0)	32	R	0000_0000h
84h	Station interface primary MAC address register 1 (SIPMAR1)	32	R	0000_0000h
90h	Station interface custom VLAN register 1 (SICVLANR1)	32	R	0000_0000h
94h	Station interface custom VLAN register 2 (SICVLANR2)	32	R	0000_0000h
100h	Station interface VLAN to IPV mapping register 0 (SIVLANIPVMR0)	32	RW	0000_0000h
104h	Station interface VLAN to IPV mapping register 1 (SIVLANIPVMR1)	32	RW	0000_0000h
150h	Station interface IPV to ring mapping register (SIIPVBDRMR0)	32	RW	0000_0000h
204h	Physical station interface message receive register (PSIMSGRR)	32	RW	0000_0000h
204h	Virtual station interface message send register (VSIMSGSR)	32	R	0000_0000h
208h	Physical station interface message send register (PSIMSGSR)	32	RW	0000_0000h
208h	Virtual station interface message receive register (VSIMSGRR)	32	R	0000_0000h
210h	PSI VSI 1 message receive address register 0 (PSIV1MSGRCVAR0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
210h	Virtual station interface message send register 0 (VSIMSGSNDAR0)	32	RW	0000_0000h
214h	PSI VSI 1 message receive address register 1 (PSIV1MSGRCVAR1)	32	RW	0000_0000h
214h	Virtual station interface message send address register 1 (VSIMSGSNDAR1)	32	RW	0000_0000h
300h	Station interface receive octets counter (ifInOctets) 0 (SIROCT0)	32	R	0000_0000h
304h	Station interface receive octets counter (ifInOctets) 1 (SIROCT1)	32	R	0000_0000h
308h	Station interface receive frame counter (aFrameReceivedOK) 0 (SIRFRM0)	32	R	0000_0000h
30Ch	Station interface receive frame counter (aFrameReceivedOK) 1 (SIRFRM1)	32	R	0000_0000h
310h	Station interface receive unicast frame counter (ifInUcastPkts) 0 (SIRUCA0)	32	R	0000_0000h
314h	Station interface receive unicast frame counter (ifInUcastPkts) 1 (SIRUCA1)	32	R	0000_0000h
318h	Station interface receive multicast frame counter (ifInMulticastPkts) 0 (SIRMCA0)	32	R	0000_0000h
31Ch	Station interface receive multicast frame counter (ifInMulticastPkts) 1 (SIRMCA1)	32	R	0000_0000h
320h	Station interface transmit octets counter (ifOutOctets) 0 (SITOCT0)	32	R	0000_0000h
324h	Station interface transmit octets counter (ifOutOctets) 1 (SITOCT1)	32	R	0000_0000h
328h	Station interface transmit frame counter (aFrameTransmittedOK) 0 (SITFRM0)	32	R	0000_0000h
32Ch	Station interface transmit frame counter (aFrameTransmittedOK) 1 (SITFRM1)	32	R	0000_0000h
330h	Station interface transmit unicast frame counter (ifOutUcastPkts) 0 (SITUCA0)	32	R	0000_0000h
334h	Station interface transmit unicast frame counter (ifOutUcastPkts) 1 (SITUCA1)	32	R	0000_0000h
338h	Station interface transmit multicast frame counter (ifOutMulticastPkts) 0 (SITMCA0)	32	R	0000_0000h
33Ch	Station interface transmit multicast frame counter (ifOutMulticastPkts) 1 (SITMCA1)	32	R	0000_0000h
3F0h	Station interface boot loader parameter register 0 (SIBLPR0)	32	R	0000_0000h
3F4h	Station interface boot loader parameter register 1 (SIBLPR1)	32	R	0000_0000h
800h	Station interface command BDR mode register (SICBDRMR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
804h	Station interface command BDR status register (SICBDRSR)	32	R	0000_0000h
810h	Station interface command BDR base address register 0 (SICBDRBAR0)	32	RW	0000_0000h
814h	Station interface command BDR base address register 1 (SICBDRBAR1)	32	RW	0000_0000h
818h	Station interface command BDR producer index register (SICBDRPIR)	32	RW	0000_0000h
81Ch	Station interface command BDR consumer index register (SICBDRCIR)	32	RW	0000_0000h
820h	Station interface command BDR length register (SICBDRLENR)	32	RW	0000_0000h
8A0h	Station interface command BDR interrupt enable register (SICBDRIER)	32	RW	0000_0000h
8A4h	Station interface command BDR interrupt detect register (SICBDRIDR)	32	RW	0000_0000h
900h	Station interface capability register 0 (SICAPR0)	32	R	See section
904h	Station interface capability register 1 (SICAPR1)	32	R	0002_0000h
908h	Station interface capability register 2 (SICAPR2)	32	R	0000_0000h
A00h	Physical station interface interrupt enable register (PSIIER)	32	RW	0000_0000h
A00h	Virtual station interface interrupt enable register (VSIIER)	32	RW	0000_0000h
A08h	Physical station interface interrupt detect register (PSIIDR)	32	RW	0000_0000h
A08h	Virtual station interface interrupt detect register (VSIIDR)	32	RW	0000_0000h
A18h	Station interface transmit interrupt detect register 0 (SITXIDR0)	32	RW	0000_0000h
A28h	Station interface receive interrupt detect register 0 (SIRXIDR0)	32	RW	0000_0000h
A30h	Station interface MSI-X vector register (SIMSIVR)	32	RW	0000_0000h
A34h	Station interface command MSI-X vector register (SICMSIVR)	32	RW	0000_0000h
A40h	Station interface timer interrupt enable register (SITMRIER)	32	RW	0000_0000h
A44h	Station interface timer interrupt detect register (SITMRIDR)	32	RW	0000_0000h
A4Ch	Station interface timer MSI-X vector register (SITMRMSIVR)	32	RW	0000_0000h
B00h - B24h	Station interface MSI-X transmit ring a vector register (SIMSITR0VR - SIMSITR9VR)	32	RW	See section
B80h - BA4h	Station interface MSI-X receive ring a vector register (SIMSIRR0VR - SIMSIRR9VR)	32	RW	See section
E00h	Station interface correctable memory error configuration register (SICMECR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
E04h	Station interface correctable memory error status register (SICMESR)	32	RW	0000_0000h
E0Ch	Station interface correctable memory error count register (SICMECTR)	32	R	0000_0000h
E10h	Station interface uncorrectable programming error configuration register (SIUPECR)	32	RW	0000_0000h
E14h	Station interface uncorrectable programming error status register (SIUPESR)	32	RW	0000_0000h
E1Ch	Station interface uncorrectable programming error count register (SIUPECTR)	32	R	0000_0000h
E20h	Station interface uncorrectable non-fatal system bus error configuration register (SIUNSBECR)	32	RW	0000_0000h
E24h	Station interface uncorrectable non-fatal system bus error status register (SIUNSBESR)	32	RW	0000_0000h
E2Ch	Station interface uncorrectable non-fatal system bus error count register (SIUNSBECTR)	32	R	0000_0000h
E30h	Station interface uncorrectable fatal system bus error configuration register (SIUFSBECR)	32	RW	0000_0000h
E34h	Station interface uncorrectable fatal system bus error status register (SIUFSBESR)	32	RW	0000_0000h
E40h	Station interface uncorrectable non-fatal memory error configuration register (SIUNMECR)	32	RW	0000_0000h
E44h	Station interface uncorrectable non-fatal memory error status register 0 (SIUNMESR0)	32	RW	0000_0000h
E48h	Station interface uncorrectable non-fatal memory error status register 1 (SIUNMESR1)	32	R	0000_0000h
E4Ch	Station interface uncorrectable non-fatal memory error count register (SIUNMECTR)	32	R	0000_0000h
E50h	Station interface uncorrectable fatal memory error configuration register (SIUFMECR)	32	RW	0000_0000h
E54h	Station interface uncorrectable fatal memory error status register 0 (SIUFMESR0)	32	RW	0000_0000h
E58h	Station interface uncorrectable fatal memory error status register 1 (SIUFMESR1)	32	R	0000_0000h
1000h	Station interface MAC address filter table capability register (SIMAFTCAPR)	32	R	See section
1100h	Station interface VLAN filter table capability register (SIVFTCAPR)	32	R	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8000h	Tx BDR 0 mode register (TB0MR)	32	RW	0000_0000h
8004h	Tx BDR 0 status register (TB0SR)	32	RW	0000_0000h
8010h	Tx BDR 0 base address register 0 (TB0BAR0)	32	RW	0000_0000h
8014h	Tx BDR 0 base address register 1 (TB0BAR1)	32	RW	0000_0000h
8018h	Tx BDR 0 producer index register (TB0PIR)	32	RW	0000_0000h
801Ch	Tx BDR 0 consumer index register (TB0CIR)	32	RW	0000_0000h
8020h	Tx BDR 0 length register (TB0LENR)	32	RW	0000_0000h
80A0h	Tx BDR 0 interrupt enable register (TB0IER)	32	RW	0000_0000h
80A4h	Tx BDR 0 interrupt detect register (TB0IDR)	32	R	0000_0000h
80A8h	Tx BDR 0 interrupt coalescing register 0 (TB0ICR0)	32	RW	0000_0000h
80ACh	Tx BDR 0 interrupt coalescing register 1 (TB0ICR1)	32	RW	0000_0000h
8100h	Rx BDR 0 mode register (RB0MR)	32	RW	0000_0000h
8104h	Rx BDR 0 status register (RB0SR)	32	RW	0000_0000h
8108h	Rx BDR 0 buffer size register (RB0BSR)	32	RW	0000_0000h
810Ch	Rx BDR 0 consumer index register (RB0CIR)	32	RW	0000_0000h
8110h	Rx BDR 0 base address register 0 (RB0BAR0)	32	RW	0000_0000h
8114h	Rx BDR 0 base address register 1 (RB0BAR1)	32	RW	0000_0000h
8118h	Rx BDR 0 producer index register (RB0PIR)	32	RW	0000_0000h
8120h	Rx BDR 0 length register (RB0LENR)	32	RW	0000_0000h
8180h	Rx BDR 0 drop count register (RB0DCR)	32	R	0000_0000h
81A0h	Rx BDR 0 interrupt enable register (RB0IER)	32	RW	0000_0000h
81A4h	Rx BDR 0 interrupt detect register (RB0IDR)	32	R	0000_0000h
81A8h	Rx BDR 0 interrupt coalescing register 0 (RB0ICR0)	32	RW	0000_0000h
81ACh	Rx BDR 0 interrupt coalescing register 1 (RB0ICR1)	32	RW	0000_0000h
8200h	Tx BDR 1 mode register (TB1MR)	32	RW	0000_0000h
8204h	Tx BDR 1 status register (TB1SR)	32	RW	0000_0000h
8210h	Tx BDR 1 base address register 0 (TB1BAR0)	32	RW	0000_0000h
8214h	Tx BDR 1 base address register 1 (TB1BAR1)	32	RW	0000_0000h
8218h	Tx BDR 1 producer index register (TB1PIR)	32	RW	0000_0000h
821Ch	Tx BDR 1 consumer index register (TB1CIR)	32	RW	0000_0000h
8220h	Tx BDR 1 length register (TB1LENR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
82A0h	Tx BDR 1 interrupt enable register (TB1IER)	32	RW	0000_0000h
82A4h	Tx BDR 1 interrupt detect register (TB1IDR)	32	R	0000_0000h
82A8h	Tx BDR 1 interrupt coalescing register 0 (TB1ICR0)	32	RW	0000_0000h
82ACh	Tx BDR 1 interrupt coalescing register 1 (TB1ICR1)	32	RW	0000_0000h
8300h	Rx BDR 1 mode register (RB1MR)	32	RW	0000_0000h
8304h	Rx BDR 1 status register (RB1SR)	32	RW	0000_0000h
8308h	Rx BDR 1 buffer size register (RB1BSR)	32	RW	0000_0000h
830Ch	Rx BDR 1 consumer index register (RB1CIR)	32	RW	0000_0000h
8310h	Rx BDR 1 base address register 0 (RB1BAR0)	32	RW	0000_0000h
8314h	Rx BDR 1 base address register 1 (RB1BAR1)	32	RW	0000_0000h
8318h	Rx BDR 1 producer index register (RB1PIR)	32	RW	0000_0000h
8320h	Rx BDR 1 length register (RB1LENR)	32	RW	0000_0000h
8380h	Rx BDR 1 drop count register (RB1DCR)	32	R	0000_0000h
83A0h	Rx BDR 1 interrupt enable register (RB1IER)	32	RW	0000_0000h
83A4h	Rx BDR 1 interrupt detect register (RB1IDR)	32	R	0000_0000h
83A8h	Rx BDR 1 interrupt coalescing register 0 (RB1ICR0)	32	RW	0000_0000h
83ACh	Rx BDR 1 interrupt coalescing register 1 (RB1ICR1)	32	RW	0000_0000h
8400h	Tx BDR 2 mode register (TB2MR)	32	RW	0000_0000h
8404h	Tx BDR 2 status register (TB2SR)	32	RW	0000_0000h
8410h	Tx BDR 2 base address register 0 (TB2BAR0)	32	RW	0000_0000h
8414h	Tx BDR 2 base address register 1 (TB2BAR1)	32	RW	0000_0000h
8418h	Tx BDR 2 producer index register (TB2PIR)	32	RW	0000_0000h
841Ch	Tx BDR 2 consumer index register (TB2CIR)	32	RW	0000_0000h
8420h	Tx BDR 2 length register (TB2LENR)	32	RW	0000_0000h
84A0h	Tx BDR 2 interrupt enable register (TB2IER)	32	RW	0000_0000h
84A4h	Tx BDR 2 interrupt detect register (TB2IDR)	32	R	0000_0000h
84A8h	Tx BDR 2 interrupt coalescing register 0 (TB2ICR0)	32	RW	0000_0000h
84ACh	Tx BDR 2 interrupt coalescing register 1 (TB2ICR1)	32	RW	0000_0000h
8500h	Rx BDR 2 mode register (RB2MR)	32	RW	0000_0000h
8504h	Rx BDR 2 status register (RB2SR)	32	RW	0000_0000h
8508h	Rx BDR 2 buffer size register (RB2BSR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
850Ch	Rx BDR 2 consumer index register (RB2CIR)	32	RW	0000_0000h
8510h	Rx BDR 2 base address register 0 (RB2BAR0)	32	RW	0000_0000h
8514h	Rx BDR 2 base address register 1 (RB2BAR1)	32	RW	0000_0000h
8518h	Rx BDR 2 producer index register (RB2PIR)	32	RW	0000_0000h
8520h	Rx BDR 2 length register (RB2LENR)	32	RW	0000_0000h
8580h	Rx BDR 2 drop count register (RB2DCR)	32	R	0000_0000h
85A0h	Rx BDR 2 interrupt enable register (RB2IER)	32	RW	0000_0000h
85A4h	Rx BDR 2 interrupt detect register (RB2IDR)	32	R	0000_0000h
85A8h	Rx BDR 2 interrupt coalescing register 0 (RB2ICR0)	32	RW	0000_0000h
85ACh	Rx BDR 2 interrupt coalescing register 1 (RB2ICR1)	32	RW	0000_0000h
8600h	Tx BDR 3 mode register (TB3MR)	32	RW	0000_0000h
8604h	Tx BDR 3 status register (TB3SR)	32	RW	0000_0000h
8610h	Tx BDR 3 base address register 0 (TB3BAR0)	32	RW	0000_0000h
8614h	Tx BDR 3 base address register 1 (TB3BAR1)	32	RW	0000_0000h
8618h	Tx BDR 3 producer index register (TB3PIR)	32	RW	0000_0000h
861Ch	Tx BDR 3 consumer index register (TB3CIR)	32	RW	0000_0000h
8620h	Tx BDR 3 length register (TB3LENR)	32	RW	0000_0000h
86A0h	Tx BDR 3 interrupt enable register (TB3IER)	32	RW	0000_0000h
86A4h	Tx BDR 3 interrupt detect register (TB3IDR)	32	R	0000_0000h
86A8h	Tx BDR 3 interrupt coalescing register 0 (TB3ICR0)	32	RW	0000_0000h
86ACh	Tx BDR 3 interrupt coalescing register 1 (TB3ICR1)	32	RW	0000_0000h
8700h	Rx BDR 3 mode register (RB3MR)	32	RW	0000_0000h
8704h	Rx BDR 3 status register (RB3SR)	32	RW	0000_0000h
8708h	Rx BDR 3 buffer size register (RB3BSR)	32	RW	0000_0000h
870Ch	Rx BDR 3 consumer index register (RB3CIR)	32	RW	0000_0000h
8710h	Rx BDR 3 base address register 0 (RB3BAR0)	32	RW	0000_0000h
8714h	Rx BDR 3 base address register 1 (RB3BAR1)	32	RW	0000_0000h
8718h	Rx BDR 3 producer index register (RB3PIR)	32	RW	0000_0000h
8720h	Rx BDR 3 length register (RB3LENR)	32	RW	0000_0000h
8780h	Rx BDR 3 drop count register (RB3DCR)	32	R	0000_0000h
87A0h	Rx BDR 3 interrupt enable register (RB3IER)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
87A4h	Rx BDR 3 interrupt detect register (RB3IDR)	32	R	0000_0000h
87A8h	Rx BDR 3 interrupt coalescing register 0 (RB3ICR0)	32	RW	0000_0000h
87ACh	Rx BDR 3 interrupt coalescing register 1 (RB3ICR1)	32	RW	0000_0000h
8800h	Tx BDR 4 mode register (TB4MR)	32	RW	0000_0000h
8804h	Tx BDR 4 status register (TB4SR)	32	RW	0000_0000h
8810h	Tx BDR 4 base address register 0 (TB4BAR0)	32	RW	0000_0000h
8814h	Tx BDR 4 base address register 1 (TB4BAR1)	32	RW	0000_0000h
8818h	Tx BDR 4 producer index register (TB4PIR)	32	RW	0000_0000h
881Ch	Tx BDR 4 consumer index register (TB4CIR)	32	RW	0000_0000h
8820h	Tx BDR 4 length register (TB4LENR)	32	RW	0000_0000h
88A0h	Tx BDR 4 interrupt enable register (TB4IER)	32	RW	0000_0000h
88A4h	Tx BDR 4 interrupt detect register (TB4IDR)	32	R	0000_0000h
88A8h	Tx BDR 4 interrupt coalescing register 0 (TB4ICR0)	32	RW	0000_0000h
88ACh	Tx BDR 4 interrupt coalescing register 1 (TB4ICR1)	32	RW	0000_0000h
8900h	Rx BDR 4 mode register (RB4MR)	32	RW	0000_0000h
8904h	Rx BDR 4 status register (RB4SR)	32	RW	0000_0000h
8908h	Rx BDR 4 buffer size register (RB4BSR)	32	RW	0000_0000h
890Ch	Rx BDR 4 consumer index register (RB4CIR)	32	RW	0000_0000h
8910h	Rx BDR 4 base address register 0 (RB4BAR0)	32	RW	0000_0000h
8914h	Rx BDR 4 base address register 1 (RB4BAR1)	32	RW	0000_0000h
8918h	Rx BDR 4 producer index register (RB4PIR)	32	RW	0000_0000h
8920h	Rx BDR 4 length register (RB4LENR)	32	RW	0000_0000h
8980h	Rx BDR 4 drop count register (RB4DCR)	32	R	0000_0000h
89A0h	Rx BDR 4 interrupt enable register (RB4IER)	32	RW	0000_0000h
89A4h	Rx BDR 4 interrupt detect register (RB4IDR)	32	R	0000_0000h
89A8h	Rx BDR 4 interrupt coalescing register 0 (RB4ICR0)	32	RW	0000_0000h
89ACh	Rx BDR 4 interrupt coalescing register 1 (RB4ICR1)	32	RW	0000_0000h
8A00h	Tx BDR 5 mode register (TB5MR)	32	RW	0000_0000h
8A04h	Tx BDR 5 status register (TB5SR)	32	RW	0000_0000h
8A10h	Tx BDR 5 base address register 0 (TB5BAR0)	32	RW	0000_0000h
8A14h	Tx BDR 5 base address register 1 (TB5BAR1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8A18h	Tx BDR 5 producer index register (TB5PIR)	32	RW	0000_0000h
8A1Ch	Tx BDR 5 consumer index register (TB5CIR)	32	RW	0000_0000h
8A20h	Tx BDR 5 length register (TB5LENR)	32	RW	0000_0000h
8AA0h	Tx BDR 5 interrupt enable register (TB5IER)	32	RW	0000_0000h
8AA4h	Tx BDR 5 interrupt detect register (TB5IDR)	32	R	0000_0000h
8AA8h	Tx BDR 5 interrupt coalescing register 0 (TB5ICR0)	32	RW	0000_0000h
8AACh	Tx BDR 5 interrupt coalescing register 1 (TB5ICR1)	32	RW	0000_0000h
8B00h	Rx BDR 5 mode register (RB5MR)	32	RW	0000_0000h
8B04h	Rx BDR 5 status register (RB5SR)	32	RW	0000_0000h
8B08h	Rx BDR 5 buffer size register (RB5BSR)	32	RW	0000_0000h
8B0Ch	Rx BDR 5 consumer index register (RB5CIR)	32	RW	0000_0000h
8B10h	Rx BDR 5 base address register 0 (RB5BAR0)	32	RW	0000_0000h
8B14h	Rx BDR 5 base address register 1 (RB5BAR1)	32	RW	0000_0000h
8B18h	Rx BDR 5 producer index register (RB5PIR)	32	RW	0000_0000h
8B20h	Rx BDR 5 length register (RB5LENR)	32	RW	0000_0000h
8B80h	Rx BDR 5 drop count register (RB5DCR)	32	R	0000_0000h
8BA0h	Rx BDR 5 interrupt enable register (RB5IER)	32	RW	0000_0000h
8BA4h	Rx BDR 5 interrupt detect register (RB5IDR)	32	R	0000_0000h
8BA8h	Rx BDR 5 interrupt coalescing register 0 (RB5ICR0)	32	RW	0000_0000h
8BACh	Rx BDR 5 interrupt coalescing register 1 (RB5ICR1)	32	RW	0000_0000h
8C00h	Tx BDR 6 mode register (TB6MR)	32	RW	0000_0000h
8C04h	Tx BDR 6 status register (TB6SR)	32	RW	0000_0000h
8C10h	Tx BDR 6 base address register 0 (TB6BAR0)	32	RW	0000_0000h
8C14h	Tx BDR 6 base address register 1 (TB6BAR1)	32	RW	0000_0000h
8C18h	Tx BDR 6 producer index register (TB6PIR)	32	RW	0000_0000h
8C1Ch	Tx BDR 6 consumer index register (TB6CIR)	32	RW	0000_0000h
8C20h	Tx BDR 6 length register (TB6LENR)	32	RW	0000_0000h
8CA0h	Tx BDR 6 interrupt enable register (TB6IER)	32	RW	0000_0000h
8CA4h	Tx BDR 6 interrupt detect register (TB6IDR)	32	R	0000_0000h
8CA8h	Tx BDR 6 interrupt coalescing register 0 (TB6ICR0)	32	RW	0000_0000h
8CACh	Tx BDR 6 interrupt coalescing register 1 (TB6ICR1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8D00h	Rx BDR 6 mode register (RB6MR)	32	RW	0000_0000h
8D04h	Rx BDR 6 status register (RB6SR)	32	RW	0000_0000h
8D08h	Rx BDR 6 buffer size register (RB6BSR)	32	RW	0000_0000h
8D0Ch	Rx BDR 6 consumer index register (RB6CIR)	32	RW	0000_0000h
8D10h	Rx BDR 6 base address register 0 (RB6BAR0)	32	RW	0000_0000h
8D14h	Rx BDR 6 base address register 1 (RB6BAR1)	32	RW	0000_0000h
8D18h	Rx BDR 6 producer index register (RB6PIR)	32	RW	0000_0000h
8D20h	Rx BDR 6 length register (RB6LENR)	32	RW	0000_0000h
8D80h	Rx BDR 6 drop count register (RB6DCR)	32	R	0000_0000h
8DA0h	Rx BDR 6 interrupt enable register (RB6IER)	32	RW	0000_0000h
8DA4h	Rx BDR 6 interrupt detect register (RB6IDR)	32	R	0000_0000h
8DA8h	Rx BDR 6 interrupt coalescing register 0 (RB6ICR0)	32	RW	0000_0000h
8DACh	Rx BDR 6 interrupt coalescing register 1 (RB6ICR1)	32	RW	0000_0000h
8E00h	Tx BDR 7 mode register (TB7MR)	32	RW	0000_0000h
8E04h	Tx BDR 7 status register (TB7SR)	32	RW	0000_0000h
8E10h	Tx BDR 7 base address register 0 (TB7BAR0)	32	RW	0000_0000h
8E14h	Tx BDR 7 base address register 1 (TB7BAR1)	32	RW	0000_0000h
8E18h	Tx BDR 7 producer index register (TB7PIR)	32	RW	0000_0000h
8E1Ch	Tx BDR 7 consumer index register (TB7CIR)	32	RW	0000_0000h
8E20h	Tx BDR 7 length register (TB7LENR)	32	RW	0000_0000h
8EA0h	Tx BDR 7 interrupt enable register (TB7IER)	32	RW	0000_0000h
8EA4h	Tx BDR 7 interrupt detect register (TB7IDR)	32	R	0000_0000h
8EA8h	Tx BDR 7 interrupt coalescing register 0 (TB7ICR0)	32	RW	0000_0000h
8EACh	Tx BDR 7 interrupt coalescing register 1 (TB7ICR1)	32	RW	0000_0000h
8F00h	Rx BDR 7 mode register (RB7MR)	32	RW	0000_0000h
8F04h	Rx BDR 7 status register (RB7SR)	32	RW	0000_0000h
8F08h	Rx BDR 7 buffer size register (RB7BSR)	32	RW	0000_0000h
8F0Ch	Rx BDR 7 consumer index register (RB7CIR)	32	RW	0000_0000h
8F10h	Rx BDR 7 base address register 0 (RB7BAR0)	32	RW	0000_0000h
8F14h	Rx BDR 7 base address register 1 (RB7BAR1)	32	RW	0000_0000h
8F18h	Rx BDR 7 producer index register (RB7PIR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8F20h	Rx BDR 7 length register (RB7LENR)	32	RW	0000_0000h
8F80h	Rx BDR 7 drop count register (RB7DCR)	32	R	0000_0000h
8FA0h	Rx BDR 7 interrupt enable register (RB7IER)	32	RW	0000_0000h
8FA4h	Rx BDR 7 interrupt detect register (RB7IDR)	32	R	0000_0000h
8FA8h	Rx BDR 7 interrupt coalescing register 0 (RB7ICR0)	32	RW	0000_0000h
8FACh	Rx BDR 7 interrupt coalescing register 1 (RB7ICR1)	32	RW	0000_0000h
9000h	Tx BDR 8 mode register (TB8MR)	32	RW	0000_0000h
9004h	Tx BDR 8 status register (TB8SR)	32	RW	0000_0000h
9010h	Tx BDR 8 base address register 0 (TB8BAR0)	32	RW	0000_0000h
9014h	Tx BDR 8 base address register 1 (TB8BAR1)	32	RW	0000_0000h
9018h	Tx BDR 8 producer index register (TB8PIR)	32	RW	0000_0000h
901Ch	Tx BDR 8 consumer index register (TB8CIR)	32	RW	0000_0000h
9020h	Tx BDR 8 length register (TB8LENR)	32	RW	0000_0000h
90A0h	Tx BDR 8 interrupt enable register (TB8IER)	32	RW	0000_0000h
90A4h	Tx BDR 8 interrupt detect register (TB8IDR)	32	R	0000_0000h
90A8h	Tx BDR 8 interrupt coalescing register 0 (TB8ICR0)	32	RW	0000_0000h
90ACh	Tx BDR 8 interrupt coalescing register 1 (TB8ICR1)	32	RW	0000_0000h
9100h	Rx BDR 8 mode register (RB8MR)	32	RW	0000_0000h
9104h	Rx BDR 8 status register (RB8SR)	32	RW	0000_0000h
9108h	Rx BDR 8 buffer size register (RB8BSR)	32	RW	0000_0000h
910Ch	Rx BDR 8 consumer index register (RB8CIR)	32	RW	0000_0000h
9110h	Rx BDR 8 base address register 0 (RB8BAR0)	32	RW	0000_0000h
9114h	Rx BDR 8 base address register 1 (RB8BAR1)	32	RW	0000_0000h
9118h	Rx BDR 8 producer index register (RB8PIR)	32	RW	0000_0000h
9120h	Rx BDR 8 length register (RB8LENR)	32	RW	0000_0000h
9180h	Rx BDR 8 drop count register (RB8DCR)	32	R	0000_0000h
91A0h	Rx BDR 8 interrupt enable register (RB8IER)	32	RW	0000_0000h
91A4h	Rx BDR 8 interrupt detect register (RB8IDR)	32	R	0000_0000h
91A8h	Rx BDR 8 interrupt coalescing register 0 (RB8ICR0)	32	RW	0000_0000h
91ACh	Rx BDR 8 interrupt coalescing register 1 (RB8ICR1)	32	RW	0000_0000h
9200h	Tx BDR 9 mode register (TB9MR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9204h	Tx BDR 9 status register (TB9SR)	32	RW	0000_0000h
9210h	Tx BDR 9 base address register 0 (TB9BAR0)	32	RW	0000_0000h
9214h	Tx BDR 9 base address register 1 (TB9BAR1)	32	RW	0000_0000h
9218h	Tx BDR 9 producer index register (TB9PIR)	32	RW	0000_0000h
921Ch	Tx BDR 9 consumer index register (TB9CIR)	32	RW	0000_0000h
9220h	Tx BDR 9 length register (TB9LENR)	32	RW	0000_0000h
92A0h	Tx BDR 9 interrupt enable register (TB9IER)	32	RW	0000_0000h
92A4h	Tx BDR 9 interrupt detect register (TB9IDR)	32	R	0000_0000h
92A8h	Tx BDR 9 interrupt coalescing register 0 (TB9ICR0)	32	RW	0000_0000h
92ACh	Tx BDR 9 interrupt coalescing register 1 (TB9ICR1)	32	RW	0000_0000h
9300h	Rx BDR 9 mode register (RB9MR)	32	RW	0000_0000h
9304h	Rx BDR 9 status register (RB9SR)	32	RW	0000_0000h
9308h	Rx BDR 9 buffer size register (RB9BSR)	32	RW	0000_0000h
930Ch	Rx BDR 9 consumer index register (RB9CIR)	32	RW	0000_0000h
9310h	Rx BDR 9 base address register 0 (RB9BAR0)	32	RW	0000_0000h
9314h	Rx BDR 9 base address register 1 (RB9BAR1)	32	RW	0000_0000h
9318h	Rx BDR 9 producer index register (RB9PIR)	32	RW	0000_0000h
9320h	Rx BDR 9 length register (RB9LENR)	32	RW	0000_0000h
9380h	Rx BDR 9 drop count register (RB9DCR)	32	R	0000_0000h
93A0h	Rx BDR 9 interrupt enable register (RB9IER)	32	RW	0000_0000h
93A4h	Rx BDR 9 interrupt detect register (RB9IDR)	32	R	0000_0000h
93A8h	Rx BDR 9 interrupt coalescing register 0 (RB9ICR0)	32	RW	0000_0000h
93ACh	Rx BDR 9 interrupt coalescing register 1 (RB9ICR1)	32	RW	0000_0000h

53.4.6.11.2 Station interface mode register (SIMR)

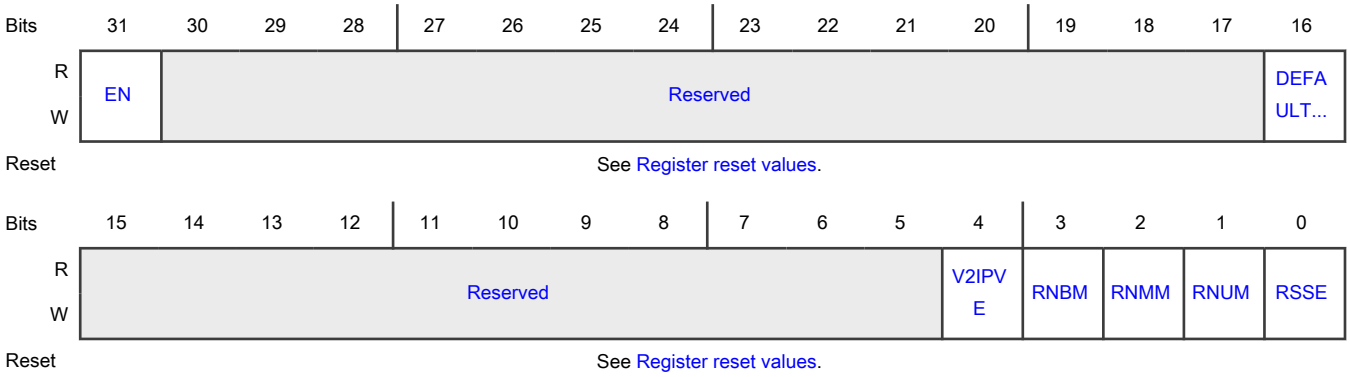
Offset

Register	Offset
SIMR	0h

Function

This is the SI mode register.

Diagram



Register reset values

Register	Reset value
SIMR	ENETC0_SI0,ENETC1_SI0: 8000_0000h ENETC1_SI1: 0000_0000h

Fields

Field	Function
31 EN	Enable station interface. This is the enable for the station interface. The port has a master enable which is ANDed with this bit to determine the final setting. 0 Station interface is disabled. Transmit descriptor rings are not processed and all received frames are dropped. Station interface counters are not updated. If received packets had started processing when the station interface was disabled, they will complete. 1 Station interface is enabled. Received frames and transmit descriptor ring entries are accepted. **NOTE** It is not safe to clear this bit while transmit rings in this SI are actively transmitting frames. Software should only disable the SI after all pending ring entries have been consumed (i.e. when PI = CI). Disabling a transmit ring (or its SI) that is actively processing BDs risks a HW-SW race hazard whereby a hardware resource becomes assigned to work on one or more ring entries only to have those entries be removed due to the ring becoming disabled. Enabled CBD rings within an SI remain operational when the SI is disabled.
30-17 —	Reserved
16 DEFAULT_RX_GROUP	Default receive group The default receive group is used when there is no match for RFS and RSS is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-5 —	Reserved
4 V2IPVE	VLAN to IPV mapping enable Maps the VLAN PCP/DEI to internal priority value (IPV). 0 Disabled. Default IPV is selected from PQOSMR[DIPV]. 1 Enabled
3 RNBM	Receive no broad-cast mode Suppresses broad-cast receive for the SI. When broad-cast receive is suppressed, the SI will not receive broad-cast packets. 0 Disabled 1 Enabled
2 RNMM	Receive no multi-cast mode Suppresses multi-cast receive for the SI. When multi-cast receive is suppressed, the SI will not receive multi-cast packets even if accepted by MAC multi-cast hash table. 0 Disabled 1 Enabled
1 RNUM	Receive no unicast-cast mode Suppresses uni-cast receive for the SI. When uni-cast receive is suppressed, the SI will not receive uni-cast packets even if accepted by MAC uni-cast address table. 0 Disabled 1 Enabled
0 RSSE	RSS classification enable 0 RSS classification is disabled 1 RSS classification is enabled This field is valid if SIPCAPR0[RSS] is set to 1.

53.4.6.11.3 Station interface status register (SISR)

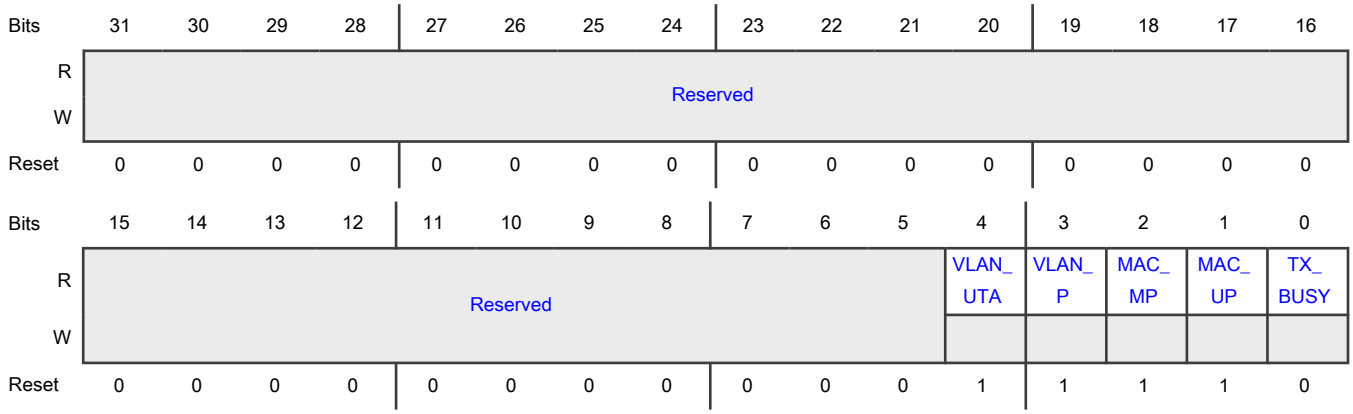
Offset

Register	Offset
SISR	4h

Function

This is the SI status register.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 VLAN_UTA	SI VLAN untagged frames accepted. If set, SI qualifies for reception of untagged frames. This bit is a reflection of the setting in PSIPVMR.
3 VLAN_P	SI VLAN promiscuous. If set, SI qualifies for reception of all VLAN tags. This bit is a reflection of the setting in PSIPVMR.
2 MAC_MP	SI MAC multicast promiscuous. If set, SI qualifies for reception of multicast frames regardless of their MAC address, except when source MAC address matches any of the receiving SI's unicast MAC addresses. This bit is a reflection of the setting in PSIPMMR.
1 MAC_UP	SI MAC unicast promiscuous. If set, SI qualifies for reception of unicast frames regardless of their MAC address. This bit is a reflection of the setting in PSIPMR.
0 TX_BUSY	Transmit busy. The station interface is busy transmitting frames or processing CBD Class 5 command. To quiesce the interface, the associated transmit rings should be disabled. 0 Idle 1 Busy

53.4.6.11.4 Station interface current time register 0 (SICTR0)

Offset

Register	Offset
SICTR0	18h

Function

SICTR0/SICTR1 indicates either the default running time (copy of TMR_DEF_CNT_L/H) or the 1588 running time (copy of TMR_CUR_TIME_L/H). The default running time is shown when TMR_CTRL[TE]=0, whereby the 1588 running time is shown when TMR_CTRL[TE]=1. Reading the low order 32-bits from SICTR0 will capture the time such that a subsequent read to SICTR1 will provide the correct 64-bit time value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURR_TIME															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURR_TIME															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURR_TIME	Time in units of nanoseconds - low order 32-bits (31-0).

53.4.6.11.5 Station interface current time register 1 (SICTR1)

Offset

Register	Offset
SICTR1	1Ch

Function

Reading the low order 32-bits from SICTR0 will capture the time such that a subsequent read to SICTR1 will provide the correct 64-bit time value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURR_TIME															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURR_TIME															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CURR_TIME	Time in units of nanoseconds - high order 32-bits (63-32).

53.4.6.11.6 Station interface port capability register 0 (SIPCAPR0)

Offset

Register	Offset
SIPCAPR0	20h

Function

This is the SI port capability register 0 which reflects the hardware capability regarding offload functions and transmit/receive rings.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															FS
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		WO	Reserved	RP	IPFLT	PSFP	RSS	Reserved	CBS	TSD	TGS	FP	RFS	Reserved	
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
SIPCAPR0	ENETC0_SI0: 0000_2E78h ENETC1_SI0,ENETC1_SI1: 0000_0E70h

Fields

Field	Function
31-17 —	Reserved
16 FS	Functional Safety Indicates whether functional safety is supported. Functional safety includes additional integrity error checks such as internal FCS and parity error checks. 0b - Not supported 1b - Supported
15-14 —	Reserved
13 WO	Wake-on-LAN Indicates whether Wake-on-LAN in low-power mode is supported. 0b - Not supported 1b - Supported
12 —	Reserved
11 RP	Rate Policing Indicates whether rate policing is supported. 0b - Not Supported 1b - Supported
10 IPFLT	Ingress Port Filtering Indicates whether ingress port filtering is supported. 0b - Not supported 1b - Supported
9 PSFP	Per-Stream Filtering and Policing (IEEE 802.1Qci) Indicates whether Per-Stream Filtering and Policing is supported.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not Supported 1b - Supported
8 RSS	Receive Side Scaling Indicates whether receive side scaling is supported. 0b - Not supported 1b - Supported
7 —	Reserved
6 CBS	Credit Based Shaping (CBS) Indicates whether credit based shaping is supported. 0b - Not Supported 1b - Supported
5 TSD	Time Specific Departure Indicates whether time specific departure is supported. 0b - Not Supported 1b - Supported
4 TGS	Time Gate Scheduling Support for time gate Scheduling (IEEE 802.1Qbv). 0b - Not supported 1b - Supported
3 FP	Frame Preemption Indicates whether frame preemption is supported. 0b - Not Supported 1b - Supported
2 RFS	Receive Flow Steering Indicates whether receive flow steering is supported. 0b - Not supported 1b - Supported
1-0 —	Reserved

53.4.6.11.7 Station interface port capability register 1 (SIPCAPR1)

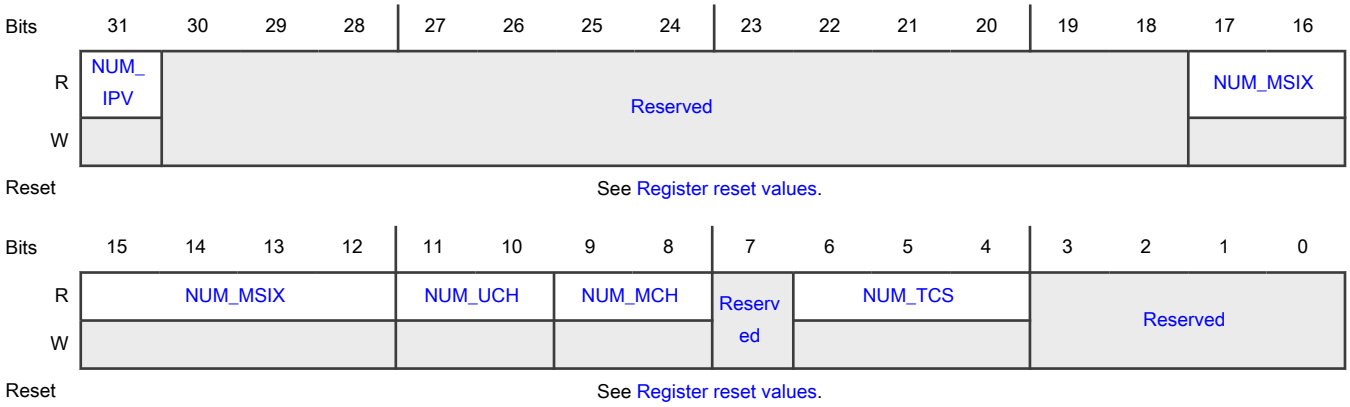
Offset

Register	Offset
SIPCAPR1	24h

Function

This is the SI port capability register 1 which reflects the hardware capability.

Diagram



Register reset values

Register	Reset value
SIPCAPR1	ENETC0_SI0: 0000_B030h ENETC1_SI0: 0001_7030h ENETC1_SI1: 0000_0030h

Fields

Field	Function
31 NUM_IPV	Indicates the number of IPVs supported. 0: 8 IPVs 1: 16 IPVs
30-18 —	Reserved
17-12 NUM_MSIX	Number of MSI-X vectors per physical/virtual function Range: 1..64

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Formula: NUM_MSIX+1
11-10 NUM_UCH	Number of unicast hash entries (1 bit per entry) per SI 00 64 unicast addresses 01 128 unicast addresses 10 256 unicast addresses 11 512 unicast addresses
9-8 NUM_MCH	Number of multicast hash entries per SI 00 64 multicast addresses 01 128 multicast addresses 10 256 multicast addresses 11 512 multicast addresses
7 —	Reserved
6-4 NUM_TCS	Number of traffic classes 0 - 1 Traffic class (0) 1 - 2 Traffic classes (0-1) ... 7 - 8 Traffic classes (0-7)
3-0 —	Reserved

53.4.6.11.8 Station interface timer status register (SITSR)

Offset

Register	Offset
SITSR	30h

Function

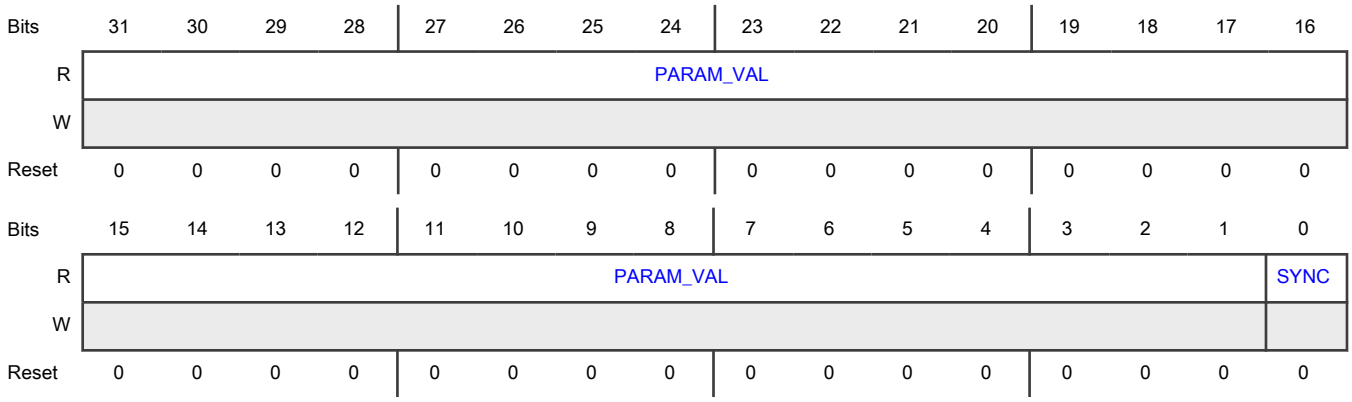
This is a read-only version of Timer function’s TMR_PARAM register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	SITSR	—
ENETC1_SI0	SITSR	—
ENETC1_SI1	—	SITSR

Diagram



Fields

Field	Function
31-1 PARAM_VAL	User specific parameter values Software specific information.
0 SYNC	Timer synchronization 0 Timer not synchronized 1 Timer synchronized

53.4.6.11.9 Station interface receive BDR group control register (SIRBGCR)

Offset

Register	Offset
SIRBGCR	38h

Function

The station interface receive BDR group control register assigns group numbers to the receive BD rings owned by the station interface. This is done by providing the total number of groups used and the number of rings per group (set once per SI and not per group within an SI).

The following rules apply:

- There is a maximum of 8 rings per group.

- Groups are numbered 0 to NUM_GROUPS-1. When one or more groups are configured to be used (NUM_GROUPS is set to a non-zero value), ring 0 is assigned to group 0 and subsequent rings are assigned to the same group in a contiguous manner till the maximum number of rings per group is reached (RINGS_PER_GROUP). Next rings are then assigned to the next higher group number following the same assignment method. The last group may have less rings if the number of rings per group is not an integer divider to the total number of rings.
- Not all rings have to be part of a group. Unassigned rings can only be reached by RFS. Rings assigned to a group can also be directly reached by RFS.
- Steering a received frame to a non existing group will result in a programming error and a dropped frame.
- Steering a received frame to a non existing ring within a group will result in a programming error and a dropped frame if the ring is outside the range of total number of rings. In all other cases, the wrong group/ring may receive the frame.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													RINGS_PER_GROUP		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														NUM_GROUPS	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-16 RINGS_PER_GROUP	Number of rings per group The number of rings per groups is RINGS_PER_GROUP + 1, e.g. a value of 7 indicates 8 rings per group.
15-2 —	Reserved
1-0 NUM_GROUPS	Number of groups Number of groups used, 0-2. A value of 0 indicates groups are not used. A value of 1 is useful for providing QoS based on IPV to BD mapping within the group. Maximum number of groups supported is indicated in SICAPR1[NUM_RX_GRP].

53.4.6.11.10 Station interface buffer cache attribute register (SIBCAR)

Offset

Register	Offset
SIBCAR	40h

Function

This is the SI buffer cache attribute register. It determines the system interface attribute setting used for reads and writes of Tx/Rx buffer descriptors and frame data.

Table 779. System Interface Read Transaction Attribute Definitions

RDSNP	RDDOMAIN[1:0]	RDCACHE[3:0]	Memory Type and System Behavior			
			Memory Type	Snoop?	Downstream cache lookup/allocate?	Description
0	b11	b0000	device	no	no / no	Non-cacheable non-coherent/register space access
0	b00	b0010	memory	no	no / no	Non-coherent copy of cacheable memory, do not look up or allocate in downstream cache, non-bufferable
0	b00	b0011	memory	no	no / no	Non-coherent copy of cacheable memory, do not look up or allocate in downstream cache, bufferable
0	b10	b0111	memory	yes	yes / yes	Coherent copy of cacheable memory, look up in downstream cache, allocate on miss
0	b00	b0111	memory	no	yes / yes	Non-coherent copy of cacheable memory, look up in downstream cache, allocate on miss
0	b10	b1011	memory	yes	yes / no	Coherent copy of cacheable memory, look up in downstream cache, no allocate on miss
0	b00	b1011	memory	no	yes / no	Non-coherent copy of cacheable memory, look up in downstream cache, no allocate on miss
all other combinations						Illegal.

Table 780. System Interface Write Transaction Attribute Definitions

WRSNP	WRDOMAIN[1:0]	WRCACHE[3:0]	Memory Type and System Behavior			
			Memory Type	Snoop?	Downstream cache lookup/allocate?	Description
0	b11	b0000	device	no	no / no	Non-cacheable non-coherent/register space access
0	b00	b0010	memory	no	no / no	Non-coherent write of cacheable memory, do not look up or allocate in downstream cache, non-bufferable
0	b00	b0011	memory	no	no / no	Non-coherent write of cacheable memory, do not look up or allocate in downstream cache, bufferable
0	b10	b1011	memory	yes	yes / yes	Coherent write of cacheable memory, look up in downstream cache, allocate on miss (partial cache line update or unknown)
0	b00	b1011	memory	no	yes / yes	Non-coherent write of cacheable memory, look up in downstream cache, allocate on miss
1	b10	b1011	memory	yes	yes / yes	Coherent write of cacheable memory, look up in downstream cache, allocate on miss (full cache line update)
0	b10	b0111	memory	yes	yes / no	Coherent write of cacheable memory, look up in downstream cache, no allocate on miss (partial cache line update or unknown)
0	b00	b0111	memory	no	yes / no	Non-coherent write of cacheable memory, look up in downstream cache, no allocate on miss
1	b10	b0111	memory	yes	yes / no	Coherent write of cacheable memory, look up in downstream cache, no allocate on miss (full cache line update)
all other combinations						Illegal.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	RDSNP	RDDOMAIN		RDCACHE				Reserved	BD_RDSNP	BD_RDDOMAIN		BD_RDCACHE			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	WRSNP	WRDOMAIN		WRCACHE				Reserved	BD_WRSNP	BD_WRDOMAIN		BD_WRCACHE			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31 —	Reserved
30 RDSNP	Read data snoop See table above for valid settings.
29-28 RDDOMAIN	Read data domain This is the domain attribute setting used when NETC reads frame data from memory for transmit. See table above for valid settings.
27-24 RDCACHE	Read data cache type This is the cache attribute setting used when NETC reads frame data from memory for transmit. See table above for valid settings.
23 —	Reserved
22 BD_RDSNP	Buffer descriptor read snoop See table above for valid settings.
21-20 BD_RDDOMAIN	Buffer descriptor read domain This is the domain attribute setting used when NETC read the buffer descriptor from memory for transmit. See table above for valid settings.
19-16 BD_RDCACHE	Buffer descriptor read cache type This is the cache attribute setting used when NETC read the buffer descriptor from memory for transmit. See table above for valid settings.
15	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14 WRSNP	Write data snoop See table above for valid settings.
13-12 WRDOMAIN	Write data domain This is the domain attribute setting used when NETC writes frame data to memory on receive. See table above for valid settings.
11-8 WRCACHE	Write data cache type This is the cache attribute setting used when NETC writes frame data to memory on receive. See table above for valid settings.
7 —	Reserved
6 BD_WRSNP	Buffer descriptor write snoop See table above for valid settings.
5-4 BD_WRDOMAIN	Buffer descriptor write domain This is the domain attribute setting used when NETC writes the buffer descriptor to memory, which includes receive and transmit BD update. See table above for valid settings.
3-0 BD_WRCACHE	Buffer descriptor write cache type This is the cache attribute setting used when NETC writes the buffer descriptor to memory, which includes receive and transmit BD update. See table above for valid settings.

53.4.6.11.11 Station interface message cache attribute register (SIMCAR)

Offset

Register	Offset
SIMCAR	44h

Function

This is the PSI-VSI message cache attribute register. It determines the system interface attribute setting used for PSI-VSI messaging reads and writes. The PSI controls message attributes for all PSI-VSI initiated messages.

See SIBCAR for attribute definitions.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	SIMCAR
ENETC1_SI0	SIMCAR	—
ENETC1_SI1	—	SIMCAR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved									MSG_RDS...	MSG_RDDOMAIN		MSG_RDCACHE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved									MSG_WRS...	MSG_WRDOMAIN		MSG_WRCACHE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-23 —	Reserved
22 MSG_RDSNP	SI messaging read data snoop See table above for valid settings.
21-20 MSG_RDDOMAIN	SI messaging read data domain This is the domain attribute setting used when NETC reads PSI-VSI message from memory during partition copy. See table above for valid settings.
19-16 MSG_RDCACHE	SI messaging read data cache type This is the cache attribute setting used when NETC reads PSI-VSI message from memory during partition copy. See table above for valid settings.
15-7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 MSG_WRSNP	SI messaging write snoop
5-4 MSG_WRDOMAIN	SI messaging write domain This is the domain attribute setting used when NETC writes PSI-VSI message during partition copy.
3-0 MSG_WRCACHE	SI messaging write cache type This is the cache attribute setting used when NETC writes PSI-VSI message during partition copy.

53.4.6.11.12 Station interface command cache attribute register (SICCAR)

Offset

Register	Offset
SICCAR	48h

Function

This is the SI command cache attribute register 2. It determines the system interface attribute setting used for reads and writes of command buffer descriptors and data.

Data and command buffer descriptor cache attributes must be set to the same value.

See SIBCAR for attribute definitions.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	CRDSNP	CRDDOMAIN		CRDCACHE				Reserved	CBD_RDS...	CBD_RDDOMAIN		CBD_RDCACHE			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	CWRSNP	CWRDOMAIN		CWRCACHE				Reserved	CBD_WRS...	CBD_WRDOMAIN		CBD_WRCACHE			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31 —	Reserved
30 CRDSNP	Read data snoop See table above for valid settings.
29-28 CRDDOMAIN	Read data domain This is the domain attribute setting used when NETC reads frame data from memory for transmit. See table above for valid settings.
27-24 CRDCACHE	Read data cache type This is the cache attribute setting used when NETC reads frame data from memory for transmit. See table above for valid settings.
23 —	Reserved
22 CBD_RDSNP	Command buffer descriptor read snoop See table above for valid settings.
21-20 CBD_RDDOMAIN	Command buffer descriptor read domain This is the domain attribute setting used when NETC read the buffer descriptor from memory for transmit. See table above for valid settings.
19-16 CBD_RDCACHE	Command buffer descriptor read cache type This is the cache attribute setting used when NETC read the buffer descriptor from memory for transmit. See table above for valid settings.
15 —	Reserved
14 CWRSNP	Write data snoop See table above for valid settings.
13-12 CWRDOMAIN	Write data domain This is the domain attribute setting used when NETC writes frame data to memory on receive. See table above for valid settings.
11-8 CWRCACHE	Write data cache type This is the cache attribute setting used when NETC writes frame data to memory on receive. See table above for valid settings.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 CBD_WRSNP	Command buffer descriptor write snoop See table above for valid settings.
5-4 CBD_WRDOMAIN	Command buffer descriptor write domain This is the domain attribute setting used when NETC writes the buffer descriptor to memory, which includes receive and transmit BD update. See table above for valid settings.
3-0 CBD_WRCACHE	Command buffer descriptor write cache type This is the cache attribute setting used when NETC writes the buffer descriptor to memory, which includes receive and transmit BD update. See table above for valid settings.

53.4.6.11.13 Station interface primary MAC address register 0 (SIPMAR0)

Offset

Register	Offset
SIPMAR0	80h

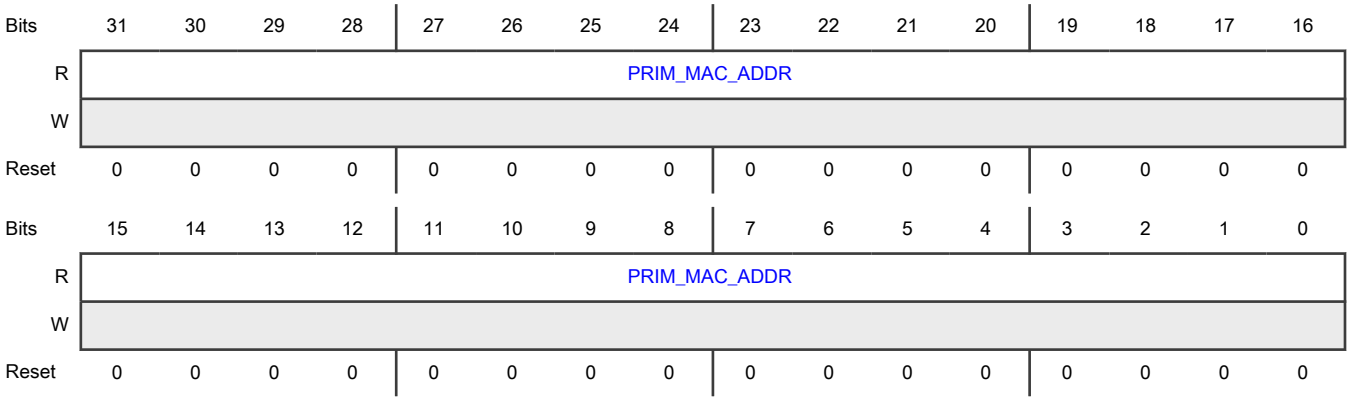
Function

This is the SI's primary MAC address register 0 as set by the port. Ingress frames addressed to this MAC address are not subject to further MAC address filtering, and are delivered to the SI only subject to VLAN filtering.

NOTE

The port will sample the IERB value when PCIe Memory Access bit is set (b1). All updates after this must be done through the port register (PMAR0 for SI 0 and PSI*a*PMAR0 for SI 1,2 .. n)

Diagram



Fields

Field	Function
31-0 PRIM_MAC_ADDR	Primary MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset (register bit offset 0-7). This field contains the 4 most significant bytes of the MAC address. For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then SIPMAR0 equals 14131211h.

53.4.6.11.14 Station interface primary MAC address register 1 (SIPMAR1)

Offset

Register	Offset
SIPMAR1	84h

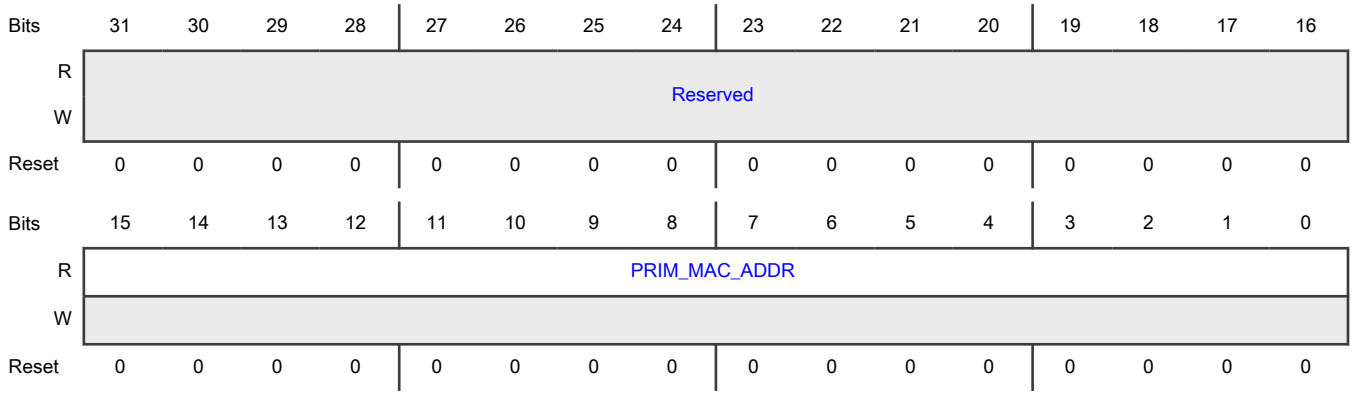
Function

This is the SI's primary MAC address register 1 as set by the port.

NOTE

The port will sample the IERB value when PCIe Memory Access bit is set (b1). All updates after this must be done through the port register (PMAR1 for SI 0 and PSI*a*PMAR1 for SI 1,2 .. n)

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 PRIM_MAC_ADDR	Primary MAC address This field is defined in network byte order (big-endian). Most significant byte of the MAC address is stored at the lowest byte offset. This field contains the 2 least significant bytes of the MAC address (least significant byte is stored in register bit offset 8-15). For example if MAC address equals 11:12:13:14:15:16 (11 being the most significant byte), then SIPMAR1 equals xxxx1615h (where x should be set to 0).

53.4.6.11.15 Station interface custom VLAN register 1 (SICVLANR1)

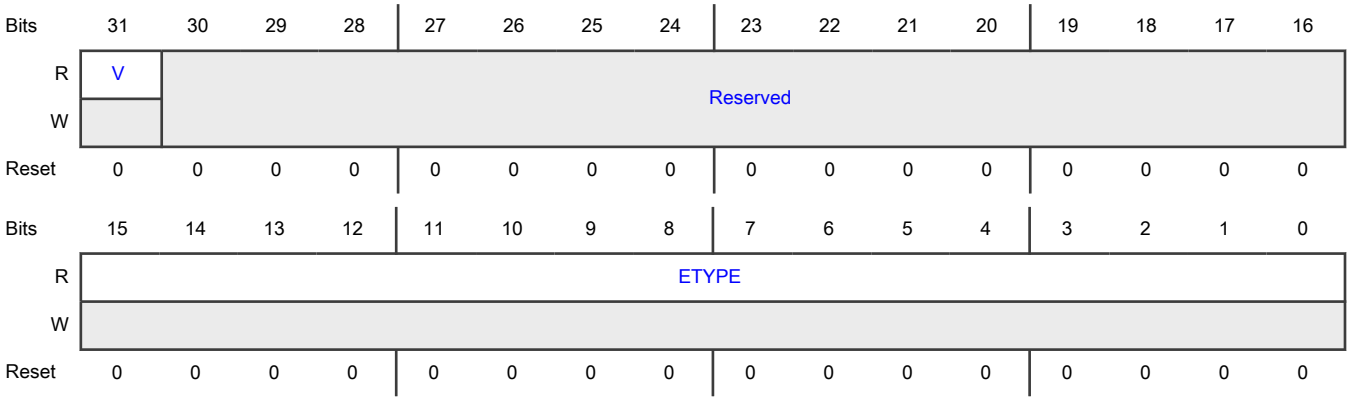
Offset

Register	Offset
SICVLANR1	90h

Function

This is the port custom VLAN tag 1 that can be recognized as a VLAN tag by the parser, reflected here as read-only values.

Diagram



Fields

Field	Function
31 V	0 not valid 1 valid
30-16 —	Reserved
15-0 ETYPE	Ethertype

53.4.6.11.16 Station interface custom VLAN register 2 (SICVLANR2)

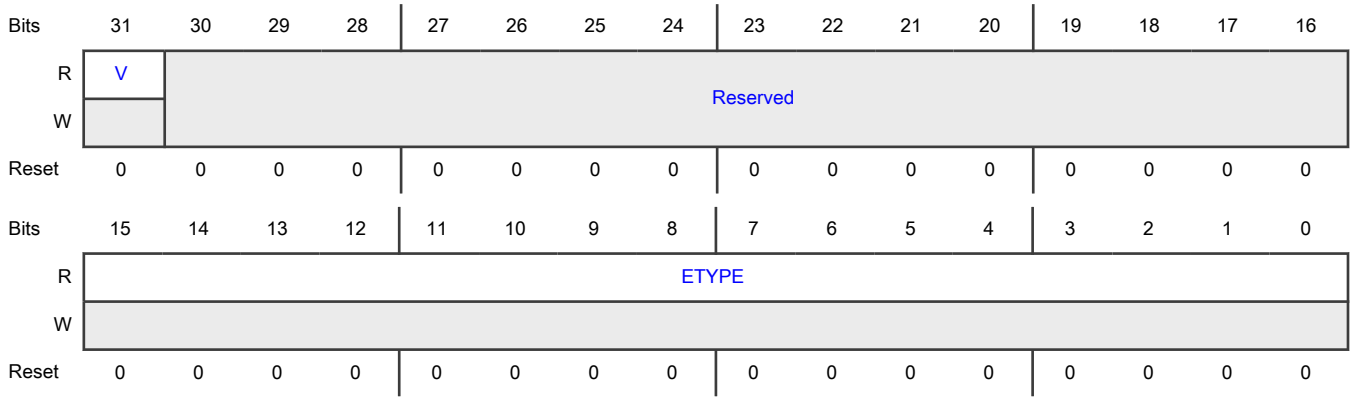
Offset

Register	Offset
SICVLANR2	94h

Function

This is the port custom VLAN tag 2 that can be recognized as a VLAN tag by the parser, reflected here as read-only values.

Diagram



Fields

Field	Function
31 V	0 not valid 1 valid
30-16 —	Reserved
15-0 ETYPE	Ethertype

53.4.6.11.17 Station interface VLAN to IPV mapping register 0 (SIVLANIPVMR0)

Offset

Register	Offset
SIVLANIPVMR0	100h

Function

This is the receive inner VLAN PCP/DEI to internal priority value (IPV) mapping register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PCP_DEI_7				PCP_DEI_6				PCP_DEI_5				PCP_DEI_4			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PCP_DEI_3				PCP_DEI_2				PCP_DEI_1				PCP_DEI_0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28: PCP_DEI_7 27-24: PCP_DEI_6 23-20: PCP_DEI_5 19-16: PCP_DEI_4 15-12: PCP_DEI_3 11-8: PCP_DEI_2 7-4: PCP_DEI_1 3-0: PCP_DEI_0	IPV value used for VLAN PCP+DEI attribute.

53.4.6.11.18 Station interface VLAN to IPV mapping register 1 (SIVLANIPVMR1)

Offset

Register	Offset
SIVLANIPVMR1	104h

Function

This is the receive inner VLAN PCP/DEI to internal priority value (IPV) mapping register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PCP_DEI_15				PCP_DEI_14				PCP_DEI_13				PCP_DEI_12			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PCP_DEI_11				PCP_DEI_10				PCP_DEI_9				PCP_DEI_8			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28: PCP_DEI_15 27-24: PCP_DEI_14 23-20: PCP_DEI_13 19-16: PCP_DEI_12 15-12: PCP_DEI_11 11-8: PCP_DEI_10 7-4: PCP_DEI_9 3-0: PCP_DEI_8	IPV value used for VLAN PCP+DEI attribute.

53.4.6.11.19 Station interface IPV to ring mapping register (SIIPVBDRMR0)

Offset

Register	Offset
SIIPVBDRMR0	150h

Function

This is the internal priority value (IPV) to BD ring (within the group) mapping register for the station interface. With the use of groups, a maximum of 8 BDRs can be mapped per group. Selecting a BDR index outside of range will result in a dropped frame and a programming error flagged.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	IPV7BDR			Reserved	IPV6BDR			Reserved	IPV5BDR			Reserved	IPV4BDR		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	IPV3BDR			Reserved	IPV2BDR			Reserved	IPV1BDR			Reserved	IPV0BDR		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 IPV7BDR	BD ring used within the group for IPV 7.
27 —	Reserved
26-24 IPV6BDR	BD ring used within the group for IPV 6.
23 —	Reserved
22-20 IPV5BDR	BD ring used within the group for IPV 5.
19 —	Reserved
18-16 IPV4BDR	BD ring used within the group for IPV 4.
15 —	Reserved
14-12 IPV3BDR	BD ring used within the group for IPV 3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 —	Reserved
10-8 IPV2BDR	BD ring used within the group for IPV 2.
7 —	Reserved
6-4 IPV1BDR	BD ring used within the group for IPV 1.
3 —	Reserved
2-0 IPV0BDR	BD ring used within the group for IPV 0.

53.4.6.11.20 Physical station interface message receive register (PSIMSGRR)

Offset

Register	Offset
PSIMSGRR	204h

Function

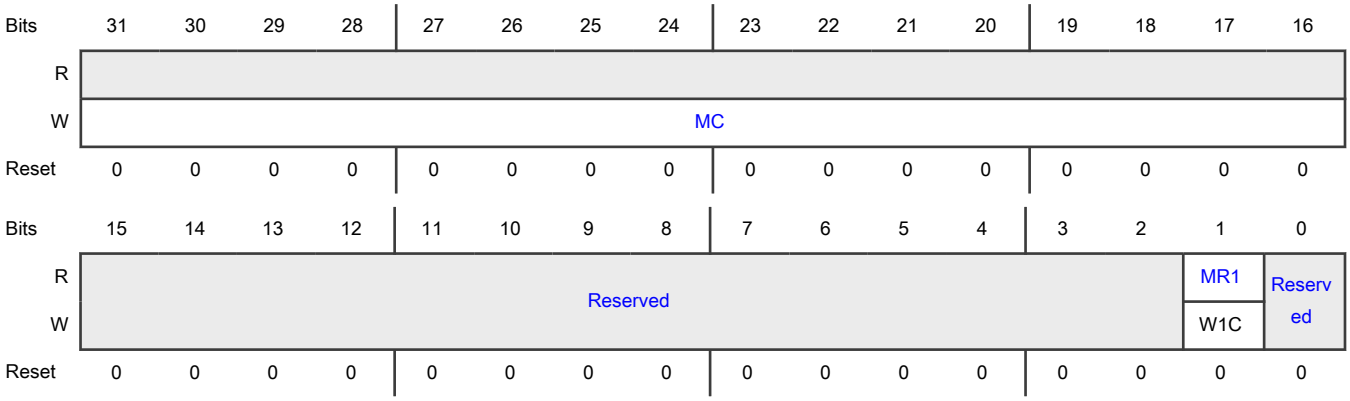
This is the message receive register for the Physical Station Interface (PSI). Messages generated by VSIs that are awaiting processing are indicated here. The register is also used to provide status back to the VSI(s) in the form of a message code.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	PSIMSGRR
ENETC1_SI0	PSIMSGRR	—
ENETC1_SI1	—	PSIMSGRR

Diagram



Fields

Field	Function
31-16 MC	Message code This field provides the result of the message back to the VSI from which it originated. It will be copied to all VSIs that have the MR bit cleared at the same time the message code is written, allowing acknowledge to multiple VSIs as long as the return code is the same. This field may provide an index value for message commands that returns an index. This field is otherwise user defined.
15-2 —	Reserved
1-1 MRn	Message received from VSI *a* When this bit is set, a message is available at address PSIV*a*MSGRCVAR0/1 from VSI *a*. Write 1 to clear and update message code and status for VSI *a*. This bit will clear automatically if VSIMSGSNDAR0 is written.
0 —	Reserved

53.4.6.11.21 Virtual station interface message send register (VSIMSGSR)

Offset

Register	Offset
VSIMSGSR	204h

Function

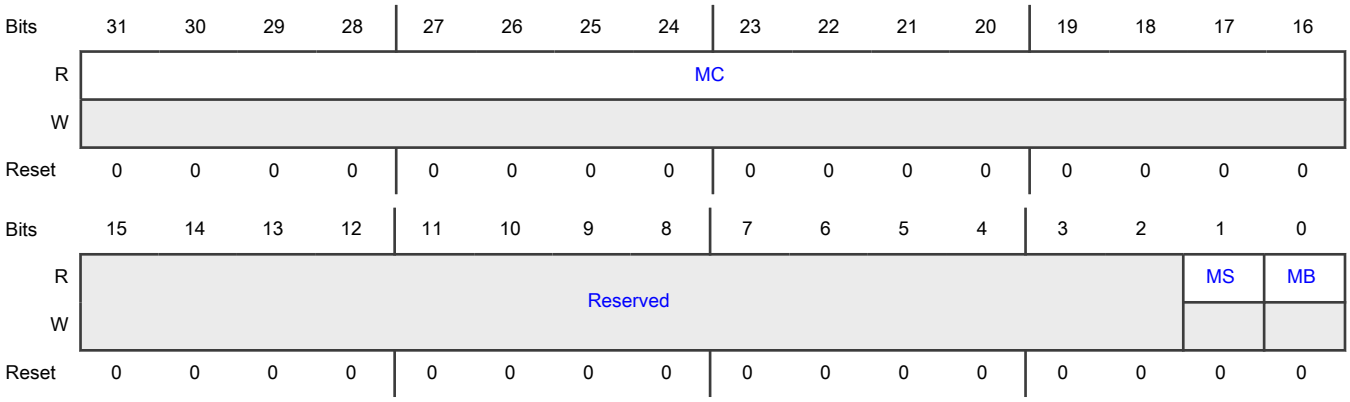
This is the message send register for the VSI to copy a message in memory to the PSI.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	VSIMSGSR
ENETC1_SI0	—	VSIMSGSR
ENETC1_SI1	VSIMSGSR	—

Diagram



Fields

Field	Function
31-16 MC	Message code This field provides additional information about the completion status of the command. If MS=0, this field provides an index value for message commands that returns an index. All other uses of this field are user defined. If MS=1 the message failed to be delivered. If MSB bit of code field is set, this indicates a user defined error type. All other values are reserved.
15-2 —	Reserved
1 MS	Message status The status bit indicates if the command was executed successfully. The message code field provides additional information depending on the status and message command and class type. Valid only when MB=0. 0 Success 1 Fail
0 MB	Message busy This bit will clear automatically and status field MS set accordingly.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0 Idle. 1 Busy. Waiting for message to be processed by PSI.

53.4.6.11.22 Physical station interface message send register (PSIMSGSR)

Offset

Register	Offset
PSIMSGSR	208h

Function

This is the message send register for the Physical Station Interface (PSI) which can provide a 16-bit user defined message code to one or more VSIs. For example, this function can provide link state changes to VSIs.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	PSIMSGSR
ENETC1_SI0	PSIMSGSR	—
ENETC1_SI1	—	PSIMSGSR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														MS1	Reserved
W															1	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 MC	Message code This field provides a user defined 16-bit message value. It will be copied to all VSIs that have the MS bit set, allowing multiple VSIs to receive the same message in one command. This field can be read to see the last message sent.
15-2 —	Reserved
1-1 MSn	Message sent to VSI *a* When this bit is set (b1), from a clear state (b0), the message code provided in the MC field will be sent to the VSI. This bit will clear when the VSI reads the message or when the VSI issues a Function Level Reset (FLR), allowing the PSI to keep track of which VSI has processed the message. Writing a 1 when the bit is already set is undefined. Writing a 0 has no effect.
0 —	Reserved

53.4.6.11.23 Virtual station interface message receive register (VSIMSGRR)

Offset

Register	Offset
VSIMSGRR	208h

Function

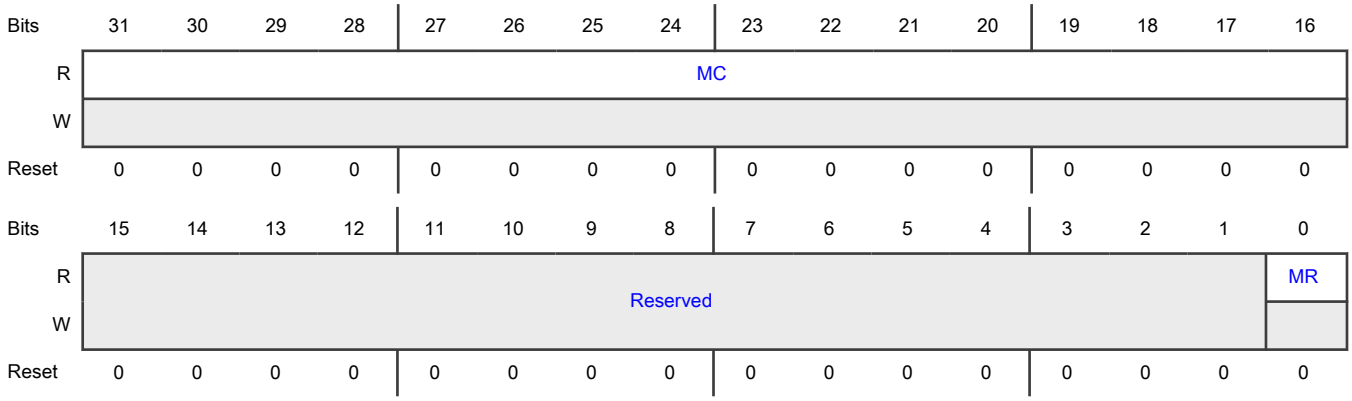
This is the message receive register for the Virtual Station Interface (VSI) which indicates if there is a pending messages from the PSI in the form of a 16-bit message code. Reading this register acknowledges to the PSI that the message was received and another message can be sent.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	VSIMSGRR
ENETC1_SI0	—	VSIMSGRR
ENETC1_SI1	VSIMSGRR	—

Diagram



Fields

Field	Function
31-16 MC	Message code This field provides a user defined 16-bit message value from the PSI.
15-1 —	Reserved
0 MR	Message Received When set, indicates that a message, provided by field MC, has been received from the PSI. Reading this register will automatically clear this bit. After a read operation, the field's value clears to 0.

53.4.6.11.24 PSI VSI 1 message receive address register 0 (PSIV1MSGRCVAR0)

Offset

Register	Offset
PSIV1MSGRCVAR0	210h

Function

This is the message receive address register 0, which determines the location where message from VSI *a* is stored. Address is 64B aligned.

NOTE

Each module instance supports a different number of registers.

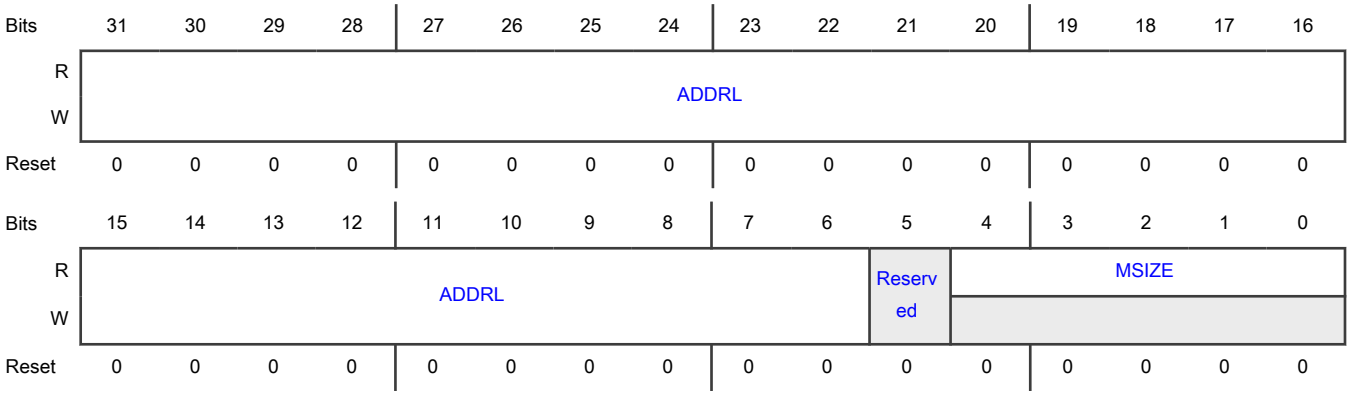
Instance	Register supported	Register not supported
ENETC0_SI0	—	PSIV1MSGRCVAR0

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
ENETC1_SI0	PSIV1MSGRCVAR0	—
ENETC1_SI1	—	PSIV1MSGRCVAR0

Diagram



Fields

Field	Function
31-6 ADDRL	Message address low Lower address bits 31-6 of message received.
5 —	Reserved
4-0 MSIZE	Message size Size of message as a multiple of 32 bytes copied from VSIMSGSNDAR0[MSIZE]. Valid when PSIMSGRR[MR] is set. Number of bytes is calculated as $2^{MSIZE+4}$, except when MSIZE equals 0. 0 1024 bytes 1 32 bytes 2 64 bytes ... 16 512 bytes 17 544 bytes ... 30 960 bytes 31 992 bytes

53.4.6.11.25 Virtual station interface message send register 0 (VSIMSGSNDAR0)

Offset

Register	Offset
VSIMSGSNDAR0	210h

Function

This is the message send address register 0. Writing this register will trigger a send of the message, located at this address, to the Physical Station Interface (PSI). Message is 64 byte aligned.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	VSIMSGSNDAR0
ENETC1_SI0	—	VSIMSGSNDAR0
ENETC1_SI1	VSIMSGSNDAR0	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ADDRL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDRL										Reserved		MSIZE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 ADDRL	Message address low Lower address bits 31-6 of message to be sent. Message is 32B address aligned.
5 —	Reserved
4-0	Message size

Table continues on the next page...

Table continued from the previous page...

Field	Function
MSIZE	Size of message as a multiple of 32 bytes. Number of bytes is calculated as $2^{MSIZE+4}$, except when MSIZE equals 0. 0 1024 bytes 1 32 bytes 2 64 bytes ... 16 512 bytes 17 544 bytes ... 30 960 bytes 31 992 bytes

53.4.6.11.26 PSI VSI 1 message receive address register 1 (PSIV1MSGRCVAR1)

Offset

Register	Offset
PSIV1MSGRCVAR1	214h

Function

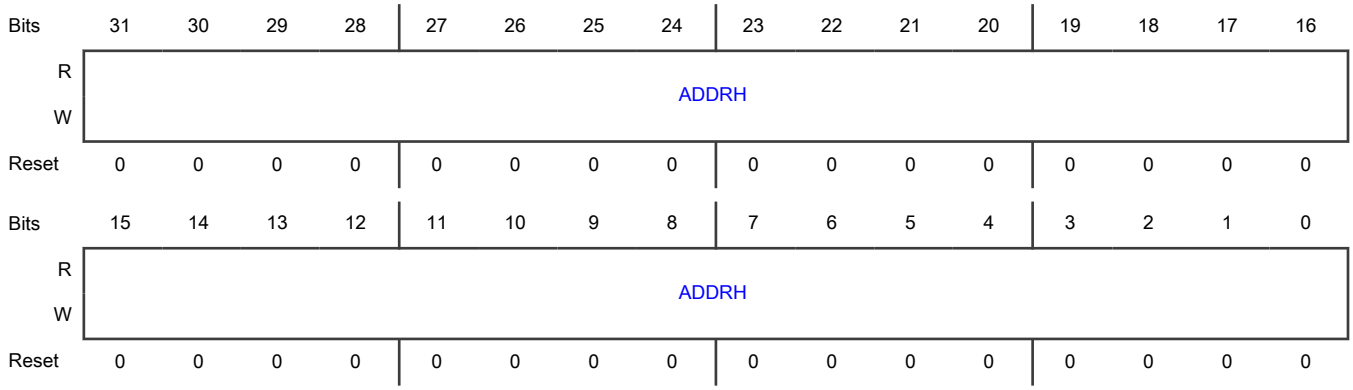
This is the message receive address register 1, which determines the location where message from VSI *a* is stored.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	PSIV1MSGRCVAR1
ENETC1_SI0	PSIV1MSGRCVAR1	—
ENETC1_SI1	—	PSIV1MSGRCVAR1

Diagram



Fields

Field	Function
31-0 ADDRH	Message address high High address bits 63-32 of message to be sent.

53.4.6.11.27 Virtual station interface message send address register 1 (VSIMSGSNDAR1)

Offset

Register	Offset
VSIMSGSNDAR1	214h

Function

This is the message send address register 1 which determines the location of the message to be sent. Address is 64B aligned.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	VSIMSGSNDAR1
ENETC1_SI0	—	VSIMSGSNDAR1
ENETC1_SI1	VSIMSGSNDAR1	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ADDRH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDRH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ADDRH	Message address high High address bits 63-32 of message to be sent.

53.4.6.11.28 Station interface receive octets counter (ifInOctets) 0 (SIROCT0)

Offset

Register	Offset
SIROCT0	300h

Function

Number of octets received (without frame errors) by the SI except preamble, SFD and CRC (if the CRC has been removed from the frame by the hardware).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ROCT_LOW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ROCT_LOW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ROCT_LOW	Counter value - low-order bits 32-0

53.4.6.11.29 Station interface receive octets counter (ifInOctets) 1 (SIROCT1)

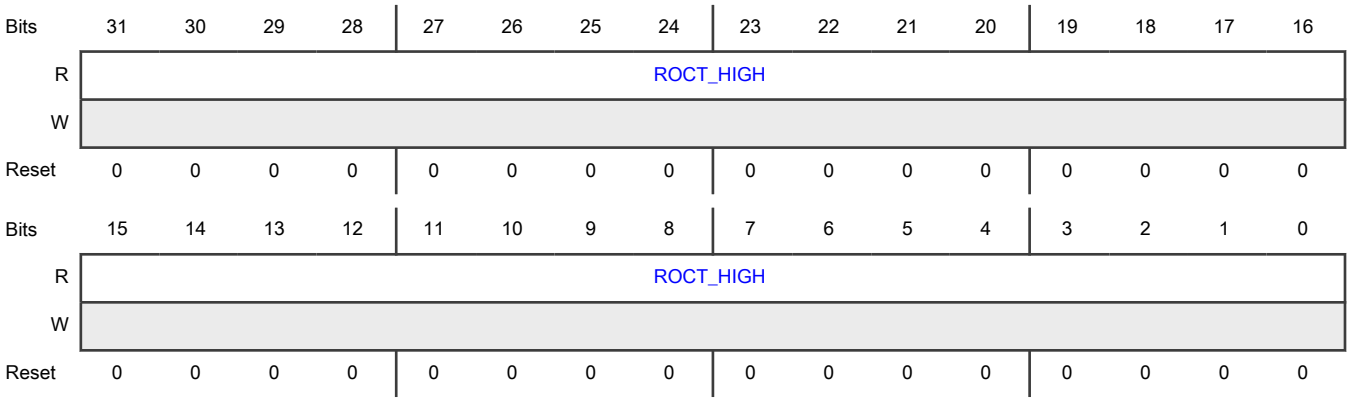
Offset

Register	Offset
SIROCT1	304h

Function

Number of octets received (without frame errors) by the SI except preamble, SFD and CRC (if the CRC has been removed from the frame by the hardware).

Diagram



Fields

Field	Function
31-0 ROCT_HIGH	Counter value - high-order bits 63-32

53.4.6.11.30 Station interface receive frame counter (aFrameReceivedOK) 0 (SIRFRM0)

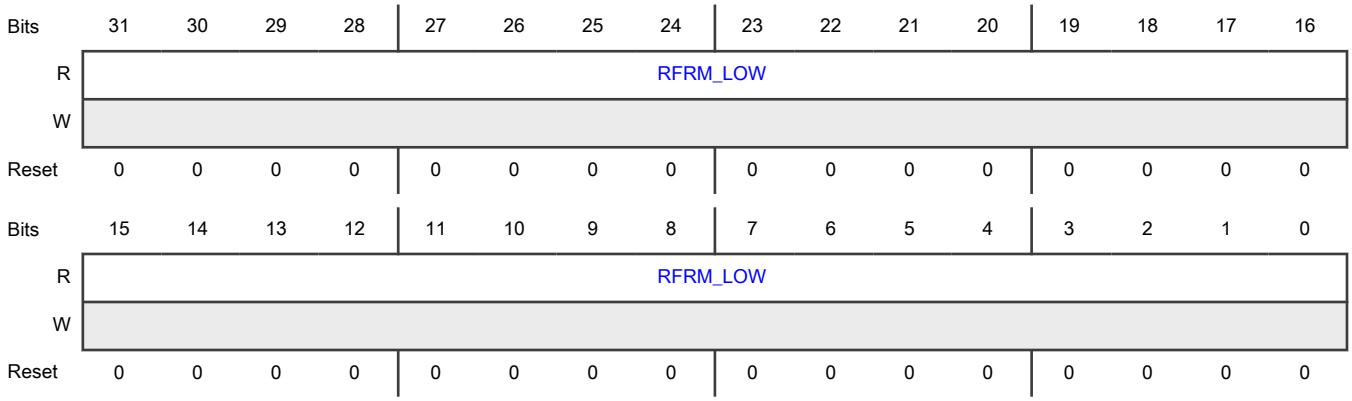
Offset

Register	Offset
SIRFRM0	308h

Function

Number of frame received by the SI without errors.

Diagram



Fields

Field	Function
31-0 RFRM_LOW	Counter value - low-order bits 31-0

53.4.6.11.31 Station interface receive frame counter (aFrameReceivedOK) 1 (SIRFRM1)

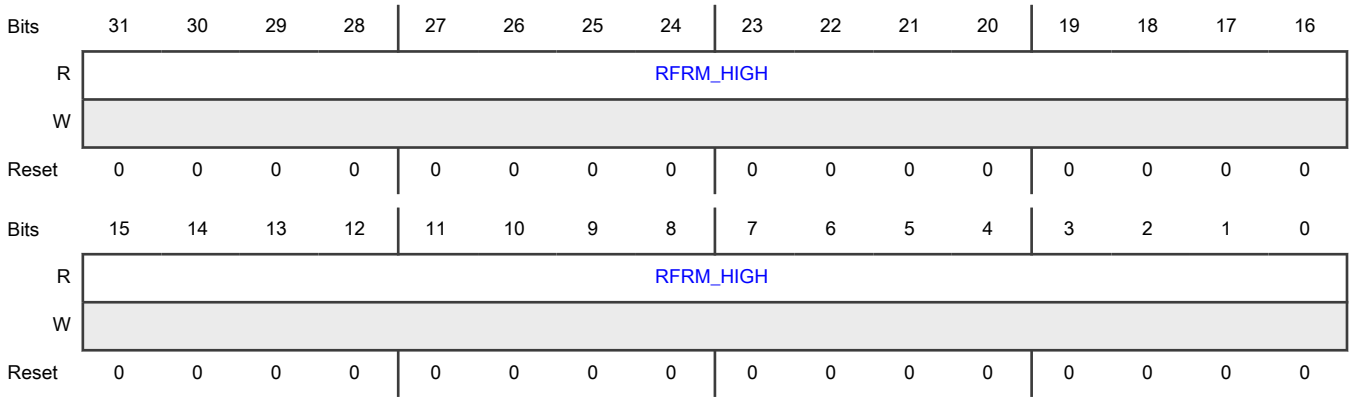
Offset

Register	Offset
SIRFRM1	30Ch

Function

Number of frames received by the SI without errors.

Diagram



Fields

Field	Function
31-0 RFRM_HIGH	Counter value - high-order bits 63-32

53.4.6.11.32 Station interface receive unicast frame counter (ifInUcastPkts) 0 (SIRUCA0)

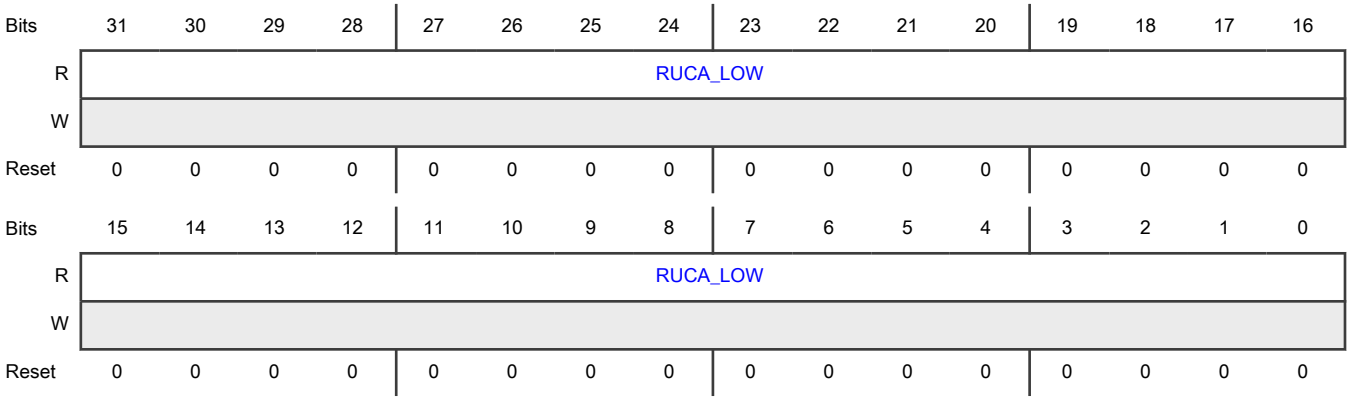
Offset

Register	Offset
SIRUCA0	310h

Function

Number of unicast frames received by the SI without errors.

Diagram



Fields

Field	Function
31-0 RUCA_LOW	Counter value - low-order bits 31-0

53.4.6.11.33 Station interface receive unicast frame counter (ifInUcastPkts) 1 (SIRUCA1)

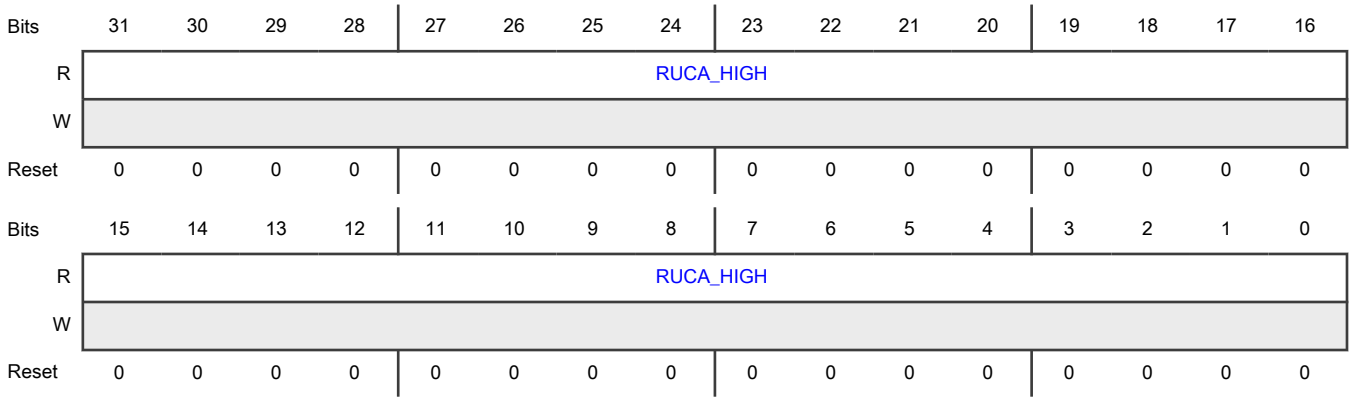
Offset

Register	Offset
SIRUCA1	314h

Function

Number of unicast frames received by the SI without errors.

Diagram



Fields

Field	Function
31-0 RUCA_HIGH	Counter value - high-order bits 63-32

53.4.6.11.34 Station interface receive multicast frame counter (ifInMulticastPkts) 0 (SIRMCA0)

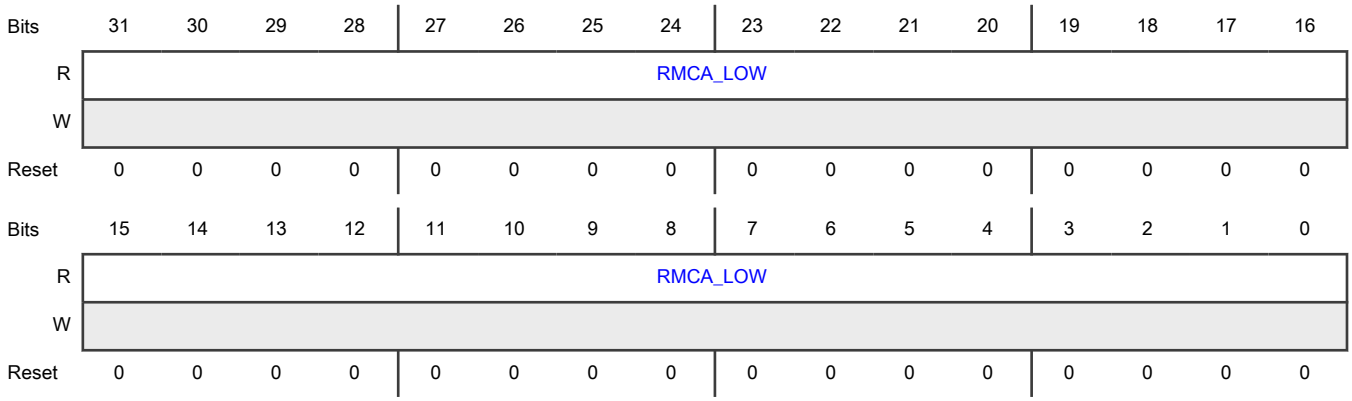
Offset

Register	Offset
SIRMCA0	318h

Function

Number of multicast frames received by the SI without errors.

Diagram



Fields

Field	Function
31-0 RMCA_LOW	Counter value - high-order bits 31-0

53.4.6.11.35 Station interface receive multicast frame counter (ifInMulticastPkts) 1 (SIRMCA1)

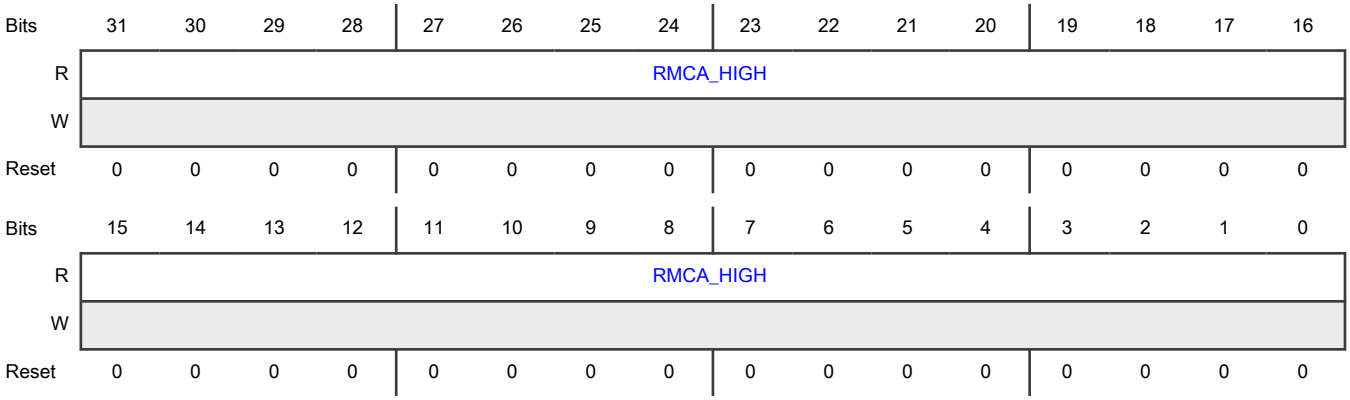
Offset

Register	Offset
SIRMCA1	31Ch

Function

Number of multicast frames received by the SI without errors.

Diagram



Fields

Field	Function
31-0 RMCA_HIGH	Counter value - high-order bits 63-32

53.4.6.11.36 Station interface transmit octets counter (ifOutOctets) 0 (SITOCT0)

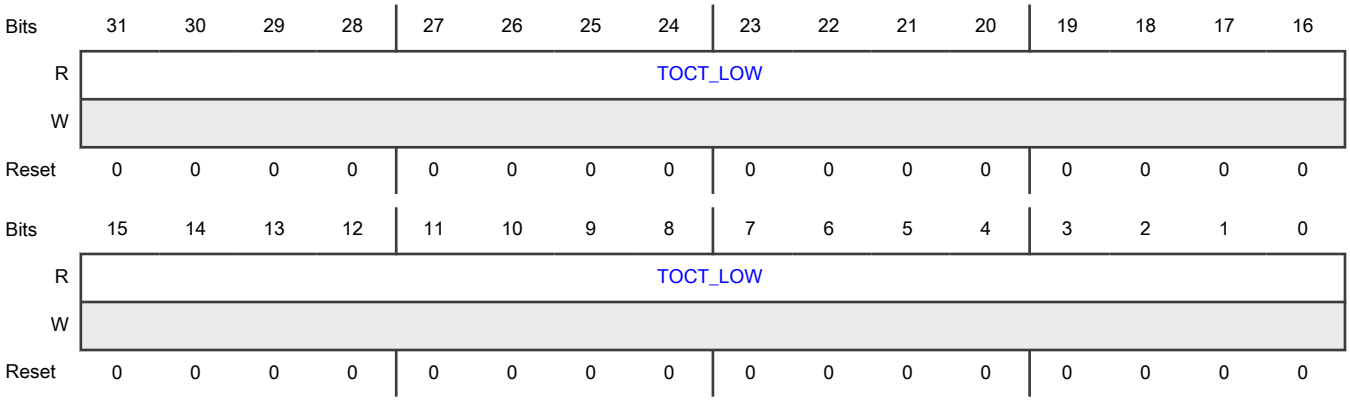
Offset

Register	Offset
SITOCT0	320h

Function

Number of octets transmitted (without errors) by the SI except preamble, SFD and CRC (if CRC has been appended to the frame by the hardware)

Diagram



Fields

Field	Function
31-0 TOCT_LOW	Counter value - low-order bits 31-0

53.4.6.11.37 Station interface transmit octets counter (ifOutOctets) 1 (SITOCT1)

Offset

Register	Offset
SITOCT1	324h

Function

Number of octets transmitted (without errors) by the SI except preamble, SFD and CRC (if CRC has been appended to the frame by the hardware).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TOCT_HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TOCT_HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TOCT_HIGH	Counter value - high-order bits 63-32

53.4.6.11.38 Station interface transmit frame counter (aFrameTransmittedOK) 0 (SITFRM0)

Offset

Register	Offset
SITFRM0	328h

Function

Number of frames transmitted by the SI without errors with the exception of frames with error (BD status code point) 0x020 (frame dropped due to port reset) or 0x0A0 (multi-bit ECC error), which will be counted in this counter.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TFRM_LOW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TFRM_LOW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TFRM_LOW	Counter value - low-order bits 31-0

53.4.6.11.39 Station interface transmit frame counter (aFrameTransmittedOK) 1 (SITFRM1)

Offset

Register	Offset
SITFRM1	32Ch

Function

Number of frames transmitted by the SI without errors with the exception of frames with error (BD status code point) 0x020 (frame dropped due to port reset) or 0x0A0 (multi-bit ECC error), which will be counted in this counter.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TFRM_HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TFRM_HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TFRM_HIGH	Counter value - high-order bits 63-32

53.4.6.11.40 Station interface transmit unicast frame counter (ifOutUcastPkts) 0 (SITUCA0)

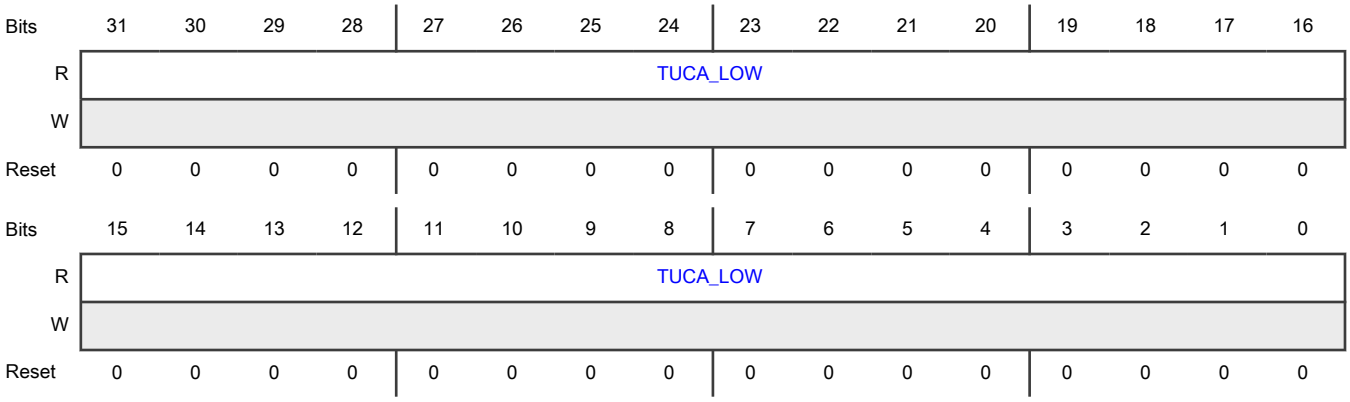
Offset

Register	Offset
SITUCA0	330h

Function

Number of unicast frames transmitted by the SI without errors with the exception of frames with error (BD status code point) 0x020 (frame dropped due to port reset) or 0x0A0 (multi-bit ECC error), which will be counted in this counter.

Diagram



Fields

Field	Function
31-0 TUCA_LOW	Counter value - low-order bits 31-0

53.4.6.11.41 Station interface transmit unicast frame counter (ifOutUcastPkts) 1 (SITUCA1)

Offset

Register	Offset
SITUCA1	334h

Function

Number of unicast frames transmitted by the SI without errors with the exception of frames with error (BD status code point) 0x020 (frame dropped due to port reset) or 0x0A0 (multi-bit ECC error), which will be counted in this counter.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TUCA_HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TUCA_HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TUCA_HIGH	Counter value - high-order bits 63-32

53.4.6.11.42 Station interface transmit multicast frame counter (ifOutMulticastPkts) 0 (SITMCA0)

Offset

Register	Offset
SITMCA0	338h

Function

Number of multicast frames transmitted by the SI without errors with the exception of frames with error (BD status code point) 0x020 (frame dropped due to port reset) or 0x0A0 (multi-bit ECC error), which will be counted in this counter.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMCA_LOW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMCA_LOW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMCA_LOW	Counter value - low-order bits 31-0

53.4.6.11.43 Station interface transmit multicast frame counter (ifOutMulticastPkts) 1 (SITMCA1)

Offset

Register	Offset
SITMCA1	33Ch

Function

Number of multicast frames transmitted by the SI without errors with the exception of frames with error (BD status code point) 0x020 (frame dropped due to port reset) or 0x0A0 (multi-bit ECC error), which will be counted in this counter.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMCA_HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMCA_HIGH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMCA_HIGH	Counter value - high-order bits 63-32

53.4.6.11.44 Station interface boot loader parameter register 0 (SIBLPR0)

Offset

Register	Offset
SIBLPR0	3F0h

Function

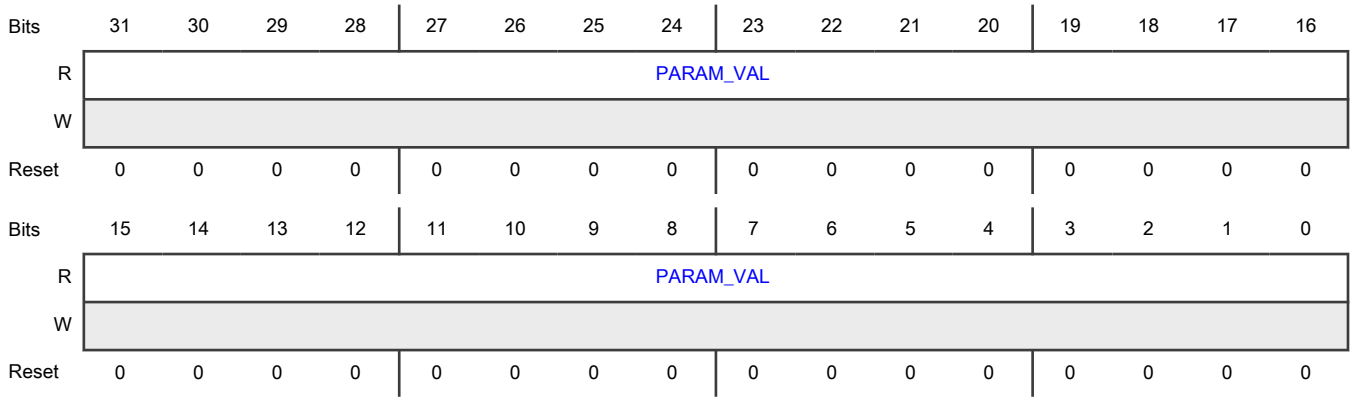
This is the SI boot loader parameter register 0. Boot loader parameters are conveyed by S/W through IERB.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	SIBLPR0
ENETC1_SI0	—	SIBLPR0
ENETC1_SI1	SIBLPR0	—

Diagram



Fields

Field	Function
31-0 PARAM_VAL	Boot loader parameter value.

53.4.6.11.45 Station interface boot loader parameter register 1 (SIBLPR1)

Offset

Register	Offset
SIBLPR1	3F4h

Function

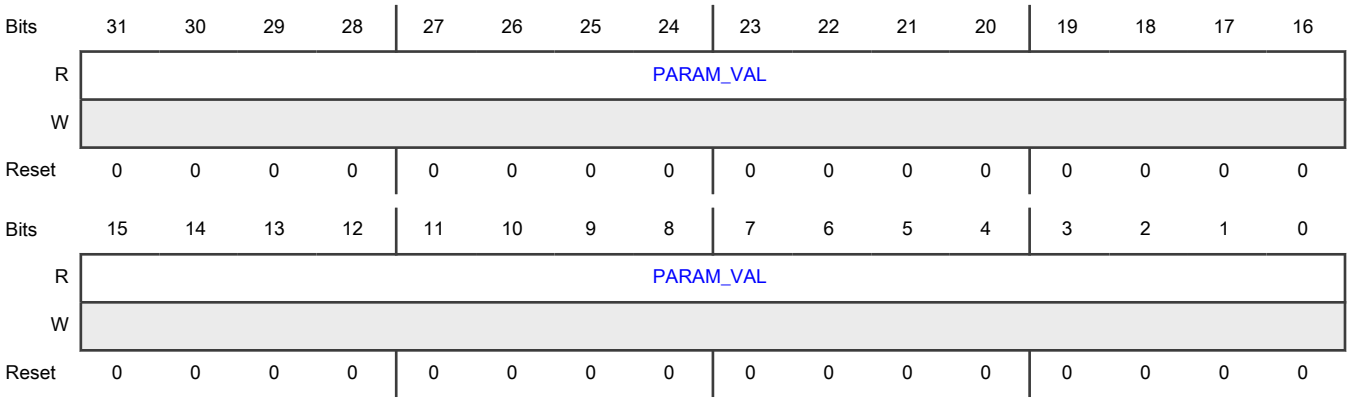
This is the SI boot loader parameter register 1. Boot loader parameters are conveyed by S/W through IERB.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	SIBLPR1
ENETC1_SI0	—	SIBLPR1
ENETC1_SI1	SIBLPR1	—

Diagram



Fields

Field	Function
31-0 PARAM_VAL	Boot loader parameter value.

53.4.6.11.46 Station interface command BDR mode register (SICBDRMR)

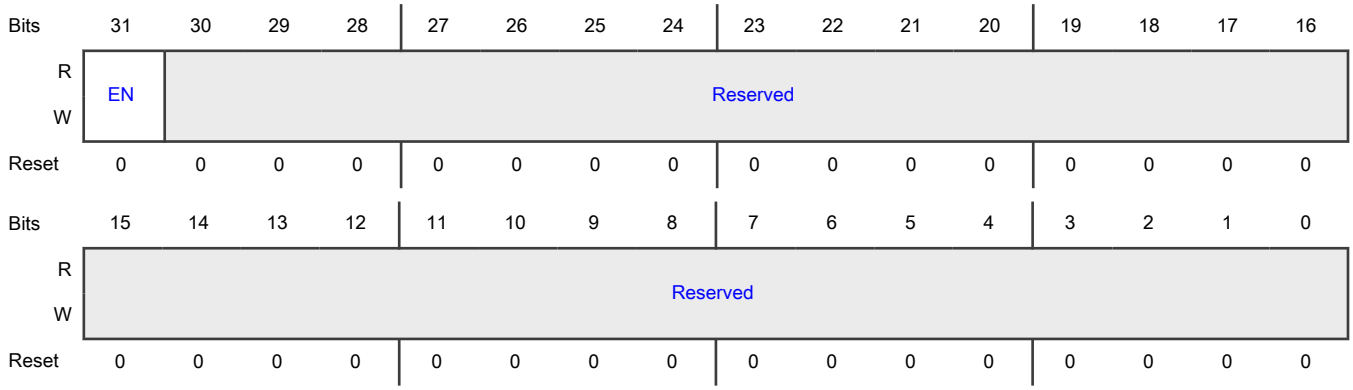
Offset

Register	Offset
SICBDRMR	800h

Function

This is the mode register for a station command buffer descriptor ring.

Diagram



Fields

Field	Function
31 EN	Enable command buffer descriptor ring 0 Disabled. 1 Enabled. When the ring is non-empty, control packets will be processed. **NOTE** When enabled, the CBD ring is active even if SIMR[EN] is disabled.
30-0 —	Reserved

53.4.6.11.47 Station interface command BDR status register (SICBDRSR)

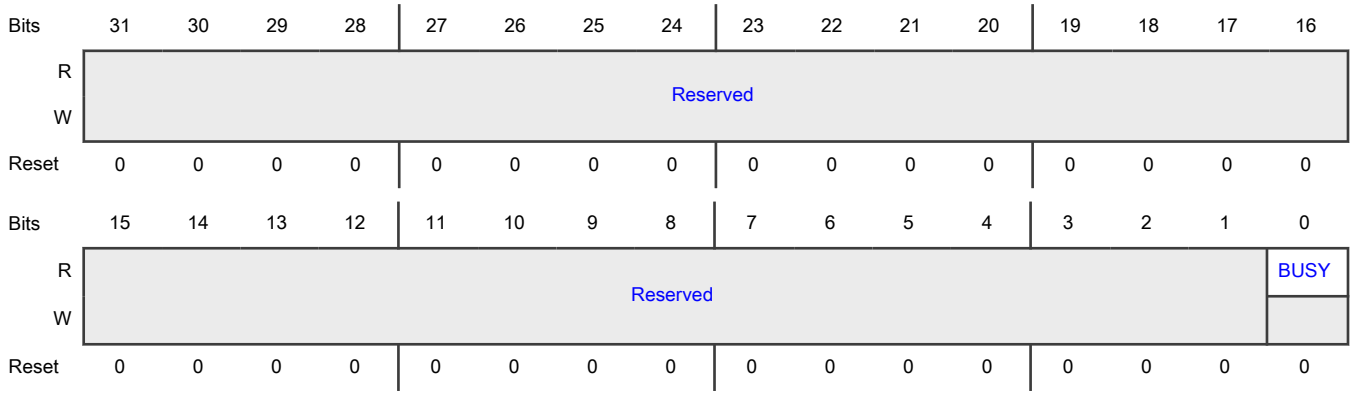
Offset

Register	Offset
SICBDRSR	804h

Function

This is the station command buffer descriptor ring status register.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 BUSY	Busy. The command BD ring is busy processing commands 0 Idle 1 Busy

53.4.6.11.48 Station interface command BDR base address register 0 (SICBDRBAR0)

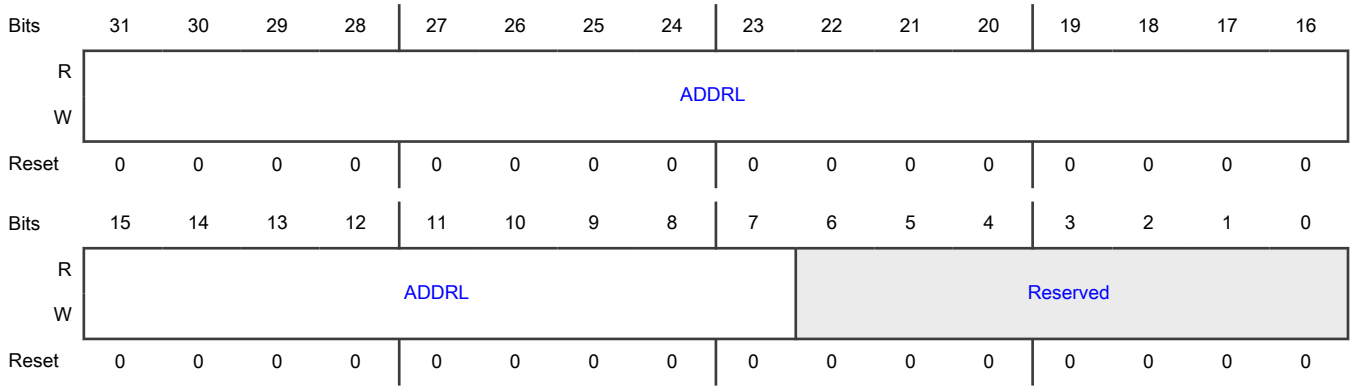
Offset

Register	Offset
SICBDRBAR0	810h

Function

This is the lower address register for the base memory address location of the station command BD ring. The address must be 128 byte aligned.

Diagram



Fields

Field	Function
31-7 ADDRL	Lower address bits 31-7.
6-0 —	Reserved

53.4.6.11.49 Station interface command BDR base address register 1 (SICBDRBAR1)

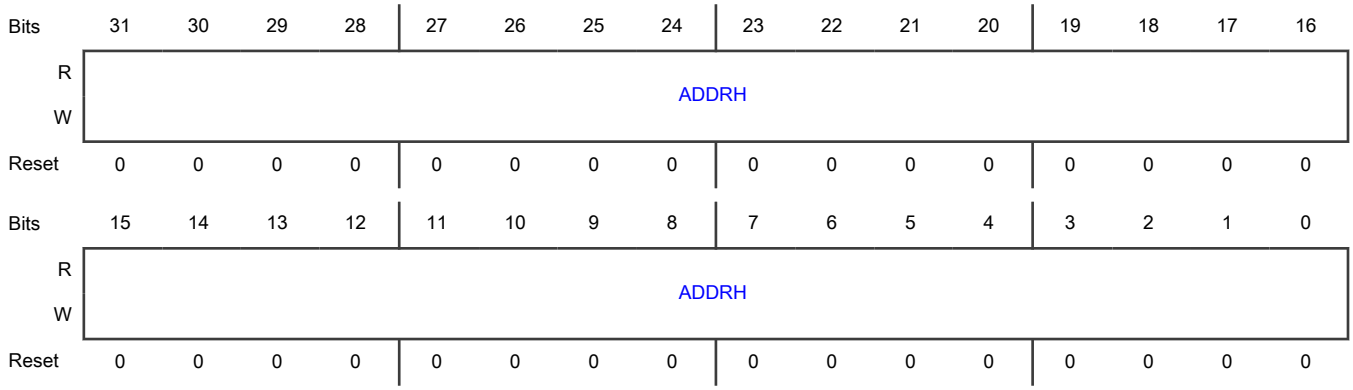
Offset

Register	Offset
SICBDRBAR1	814h

Function

This is the upper address register for the base memory address location of the station command BD ring.

Diagram



Fields

Field	Function
31-0 ADDRH	Upper address bits 63-32.

53.4.6.11.50 Station interface command BDR producer index register (SICBDRPIR)

Offset

Register	Offset
SICBDRPIR	818h

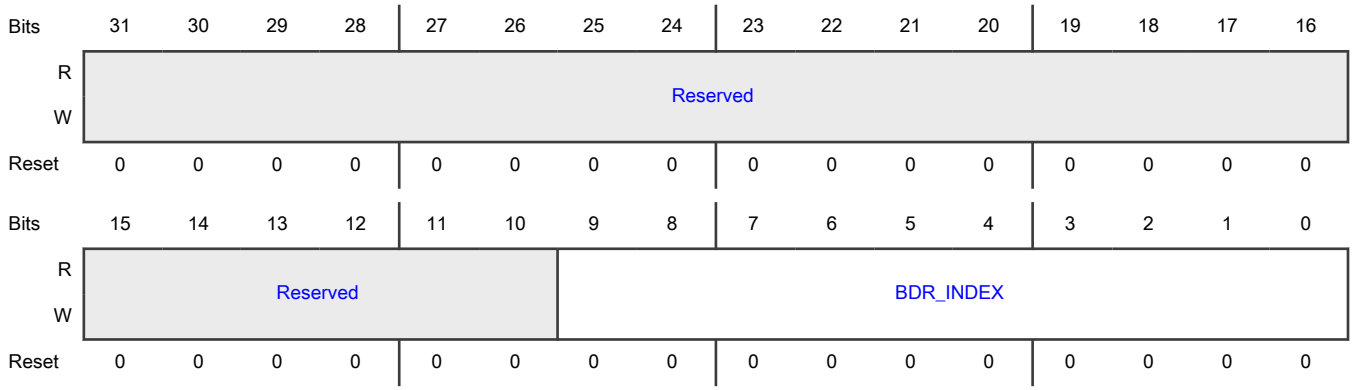
Function

This is the command BDR producer index register. When the BD ring is enabled the value of this register and the consumer index register, SICBDRCIR, determines NETC's unprocessed BDs of the ring. This value also determines how many BD's are committed to the BDR by software. Software should initialize this register while the ring is disabled to reflect an empty state or a pre-loaded BDR state.

To add one or more buffer descriptors to an enabled control ring, software should first check to make sure the ring is not full by reading SICBDRCIR. Note that maximum number of BDs allowed on a ring is one less than the ring size. A memory barrier must be used to guarantee update of the BD(s) to memory before writing the producer index register.

There is no wrap bit to indicate full/empty. One entry is used to differentiate between full and empty. When producer index + 1 equals consumer index, the ring is considered full. When the producer/consumer indices match, the ring is always considered empty.

Diagram



Fields

Field	Function
31-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-0 BDR_INDEX	Command buffer descriptor ring producer index. Range of index depends on ring size SICBDRLENR[LENGTH].

53.4.6.11.51 Station interface command BDR consumer index register (SICBDRCIR)

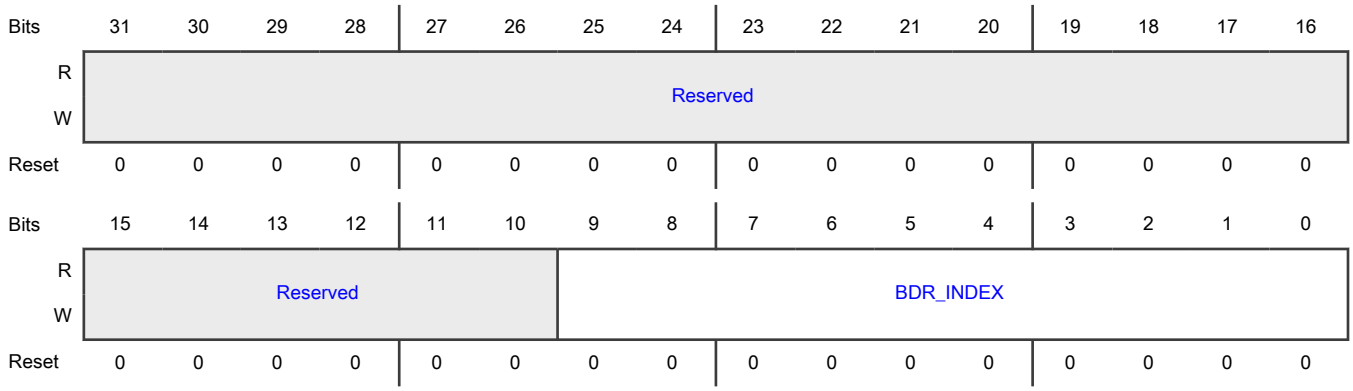
Offset

Register	Offset
SICBDRCIR	81Ch

Function

This is the command BDR consumer index register. It reflects the next BD to be processed by NETC and trails the SICBDRPIR index value. Software should only update/reset this value when the BD ring is re-initialized. Disabling the BD ring has no effect on this register. Software can calculate available entries in the ring by using the software producer index and this register value.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 BDR_INDEX	Command buffer descriptor ring consumer index. Range of index depends on ring size SICBDRLENR[LENGTH].

53.4.6.11.52 Station interface command BDR length register (SICBDRLENR)

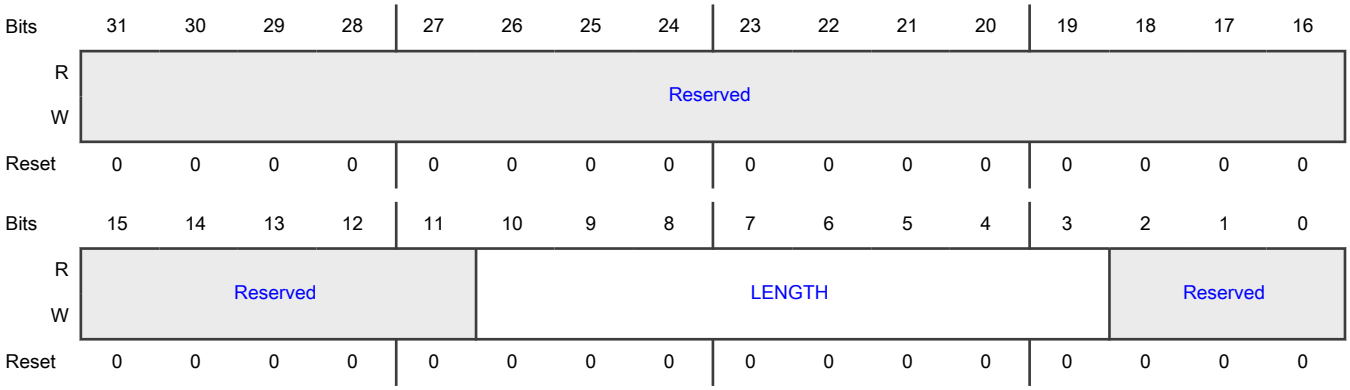
Offset

Register	Offset
SICBDRLENR	820h

Function

This is the command BDR length register. It determines the size of the transmit ring in sets of 8 BDs.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-3 LENGTH	BD ring length Size of ring in sets of 8 BDs. Maximum ring size is 1K.
2-0 —	Reserved

53.4.6.11.53 Station interface command BDR interrupt enable register (SICBDRIER)

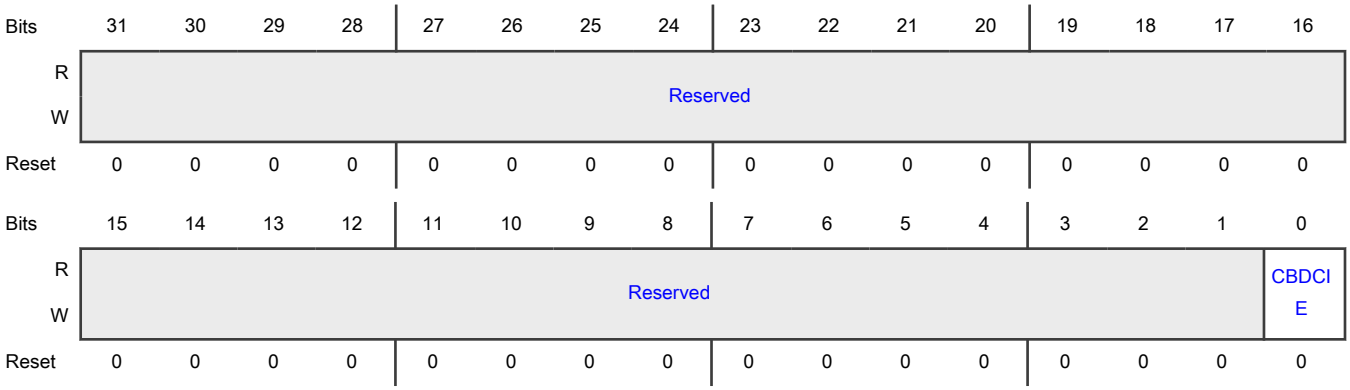
Offset

Register	Offset
SICBDRIER	8A0h

Function

This is the interrupt enable register for the station command BD ring. If the enable bit and the corresponding detect bit in SICBDRIDR is set, interrupt will be triggered for the ring.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CBDCIE	Command BD completion interrupt enable 0 Disabled 1 Enabled

53.4.6.11.54 Station interface command BDR interrupt detect register (SICBDRIDR)

Offset

Register	Offset
SICBDRIDR	8A4h

Function

This is the interrupt detect register for the station interface command buffer descriptor ring. Write 1 to clear.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															CBDC
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 CBDC	Command BD completed 0 No BD with CI=1 completed 1 Processed BD with CI=1

53.4.6.11.55 Station interface capability register 0 (SICAPR0)

Offset

Register	Offset
SICAPR0	900h

Function

This is the station interface capability register 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NUM_MAC_ADDR				Reserved				NUM_RX_BDR							
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								NUM_TX_BDR							
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
SICAPR0	ENETC0_SI0: 1004_0004h ENETC1_SI0: 100A_000Ah ENETC1_SI1: 1000_0000h

Fields

Field	Function
31-28 NUM_MAC_ADDR	Number of MAC addresses Formula: NUM_MAC_ADDR+1
27-24 —	Reserved
23-16 NUM_RX_BDR	Number of receive buffer descriptor rings assigned to the SI
15-8 —	Reserved
7-0 NUM_TX_BDR	Number of transmit buffer descriptor rings assigned to the SI

53.4.6.11.56 Station interface capability register 1 (SICAPR1)

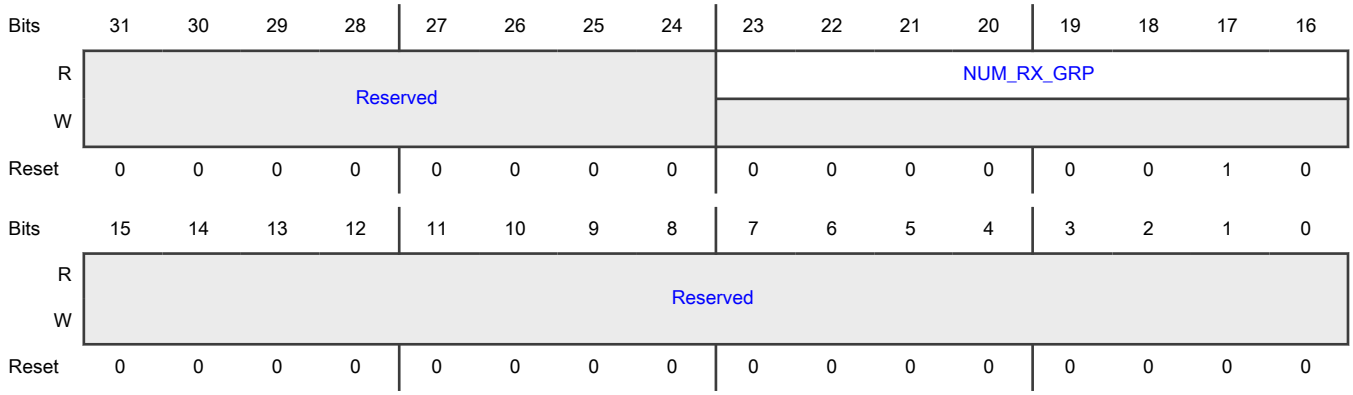
Offset

Register	Offset
SICAPR1	904h

Function

This is the station interface capability register 1.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 NUM_RX_GRP	Max number of receive buffer descriptor ring groups available to the SI
15-0 —	Reserved

53.4.6.11.57 Station interface capability register 2 (SICAPR2)

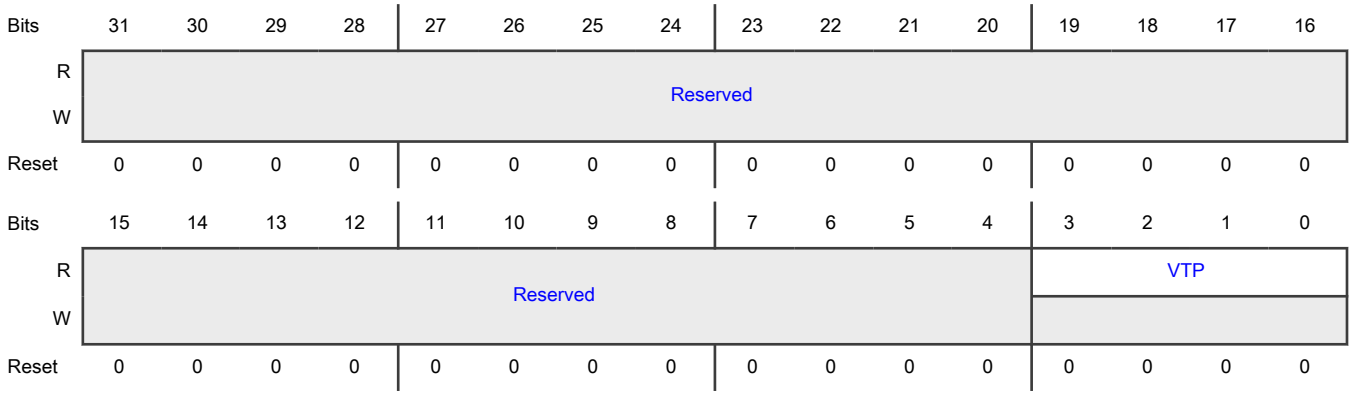
Offset

Register	Offset
SICAPR2	908h

Function

This is the station interface capability register 2.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 VTP	VLAN types permitted Bit: 0 Standard C-VLAN 0x8100 1 Standard S-VLAN 0x88A8 2 Custom VLAN as defined by CVLANR1[ETYPE] 3 Custom VLAN as defined by CVLANR2[ETYPE]

53.4.6.11.58 Physical station interface interrupt enable register (PSIIER)

Offset

Register	Offset
PSIIER	A00h

Function

This is the summary interrupt enable register for the PSI. If the enable bit and the corresponding detect bit in PSIIDR is set, interrupt will be triggered for the SI based on vector defined in SIMSIVR.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	PSIIER

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
ENETC1_SI0	PSIIER	—
ENETC1_SI1	—	PSIIER

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved														FLR1IE	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														MR1IE	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved
17-17 FLRnIE	Function level reset interrupt enable, initiated by VSI *n* 0 Disabled 1 Enabled
16-2 —	Reserved
1-1 MRnIE	Message receive interrupt enable, initiated by VSI *n* 0 Disabled 1 Enabled
0 —	Reserved

53.4.6.11.59 Virtual station interface interrupt enable register (VSIIER)

Offset

Register	Offset
VSIIER	A00h

Function

This is the summary interrupt enable register for the VSI. If the enable bit and the corresponding detect bit in VSIIDR is set, interrupt will be triggered for the SI based on vector defined in SIMSIVR.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	VSIIER
ENETC1_SI0	—	VSIIER
ENETC1_SI1	VSIIER	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						MRIE	MSIE	Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 MRIE	Message received interrupt enable 0 Disabled 1 Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 MSIE	Message sent interrupt enable 0 Disabled 1 Enabled
7-0 —	Reserved

53.4.6.11.60 Physical station interface interrupt detect register (PSIIDR)

Offset

Register	Offset
PSIIDR	A08h

Function

This is the interrupt detect register for the PSI which provides a summary for all functional events detected. Writing a 1 to any of the fields will clear the event. It is recommended that the PSIIER be cleared prior to servicing events in the PSIIDR, then re-enabled upon service routine exit. This will ensure generation of a new interrupt in the event of a set/clear collision in any of the message or FLR source bits.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	PSIIDR
ENETC1_SI0	PSIIDR	—
ENETC1_SI1	—	PSIIDR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														FLR1	RXR
W															W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														MR1	TXR
W															W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved
17-17 FLRn	Function level reset of VF initiated by VSI *n* 0 No reset 1 Reset
16 RXR	Summary of detected interrupts for all receive rings belonging to the SI 0 No interrupt detected for receive ring(s) 1 Interrupt detected for receive ring(s), see SIRXIDR Read only, clear using SIRXIDR.
15-2 —	Reserved
1-1 MRn	Message received from VSI *n* 0 No message received 1 Message received
0 TXR	Summary of detected interrupts for all transmit rings belonging to the SI 0 No interrupt detected for transmit ring(s) 1 Interrupt detected for transmit ring(s), see SITXIDR Read only, clear using SITXIDR.

53.4.6.11.61 Virtual station interface interrupt detect register (VSIIDR)

Offset

Register	Offset
VSIIDR	A08h

Function

This is the interrupt detect register for the VSI which provides a summary for all functional events detected. Writing a 1 to any of the fields will clear the event(s). It is recommended that the VSIIER be cleared prior to servicing events in the VSIIDR, then re-enabled upon service routine exit. This will ensure generation of a new interrupt in the event of a set/clear collision in any of source bits.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	—	VSIIDR
ENETC1_SI0	—	VSIIDR
ENETC1_SI1	VSIIDR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															RXR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						MR	MS	Reserved							TXR
W							W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 RXR	Summary of detected interrupts for all receive rings belonging to the SI 0 No interrupt detected for receive ring(s)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1 Interrupt detected for receive ring(s), see SIRXIDR Read only, clear using SIRXIDR.
15-10 —	Reserved
9 MR	Message received 0 No message 1 Message received from PSI, see VSIMSGRR
8 MS	Message sent 0 Message sent to PSI has not completed 1 Message sent to PSI has completed and response received
7-1 —	Reserved
0 TXR	Summary of detected interrupts for all transmit rings belonging to the SI 0 No interrupt detected for transmit ring(s) 1 Interrupt detected for transmit ring(s), see SITXIDR Read only, clear using SITXIDR.

53.4.6.11.62 Station interface transmit interrupt detect register 0 (SITXIDR0)

Offset

Register	Offset
SITXIDR0	A18h

Function

This is the transmit interrupt detect register 0 which provides a summary of events for all transmit rings belonging to the SI. See TB*a*IDR for details. Note that the actual number of bits used depends on the number of allocated transmit rings to the SI.

Writing a 1 to any of the fields will clear the event(s).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		TXF13	TXF12	TXF11	TXF10	TXF9	TXF8	TXF7	TXF6	TXF5	TXF4	TXF3	TXF2	TXF1	TXF0
W			W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		TXT13	TXT12	TXT11	TXT10	TXT9	TXT8	TXT7	TXT6	TXT5	TXT4	TXT3	TXT2	TXT1	TXT0
W			W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-16 TXFn	Summary of detected transmit frame interrupts for transmit ring n assigned to SI 0 No interrupt detected for transmit ring n 1 Frame transmit interrupt detected for transmit ring n
15-14 —	Reserved
13-0 TXTn	Summary of detected threshold interrupts for transmit ring n assigned to SI 0 No interrupt detected for transmit ring n 1 Threshold interrupt detected for transmit ring n

53.4.6.11.63 Station interface receive interrupt detect register 0 (SIRXIDR0)

Offset

Register	Offset
SIRXIDR0	A28h

Function

This is the receive interrupt detect register 0 which provides a summary of events for all receive rings belonging to the SI. See RB*a*IDR for details. Note that the actual number of bits used depends on the number of allocated transmit rings to the SI.

Writing a 1 to any of the fields will clear the event(s).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		RX13	RX12	RX11	RX10	RX9	RX8	RX7	RX6	RX5	RX4	RX3	RX2	RX1	RX0
W			W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved
13-0 RXn	Summary of detected interrupts for receive ring n assigned to SI 0 No interrupt detected for receive ring n 1 Interrupt detected for receive ring n

53.4.6.11.64 Station interface MSI-X vector register (SIMSIVR)

Offset

Register	Offset
SIMSIVR	A30h

Function

This is the MSI-X vector register which defines the vector used for address/data lookup in the MSI-X Table. The total number of vectors supported for the station interface is determined by SIPCAPR1[NUM_MSIX].

The following interrupt sources uses this vector:

- Message received from VSI
- FLR initiated by VSI

NOTE

Each module instance supports a different number of registers.

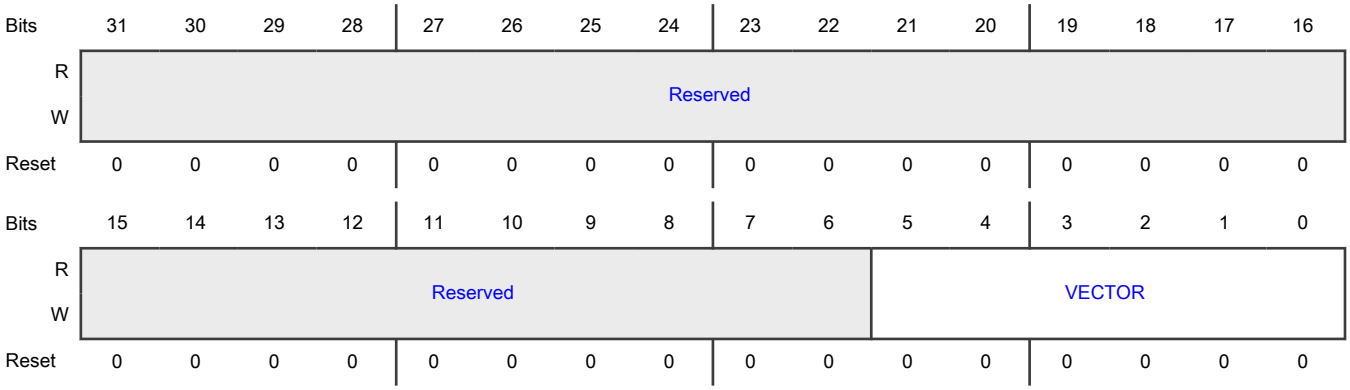
Instance	Register supported	Register not supported
ENETC0_SI0	—	SIMSIVR

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
ENETC1_SI0	SIMSIVR	—
ENETC1_SI1	SIMSIVR	—

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 VECTOR	Vector Index into MSI-X address/data table. Range: 0..SIPCAPR1[NUM_MSIX]-1 **NOTE** Specifying a vector number outside of the range will not generate an MSI-X write.

53.4.6.11.65 Station interface command MSI-X vector register (SICMSIVR)

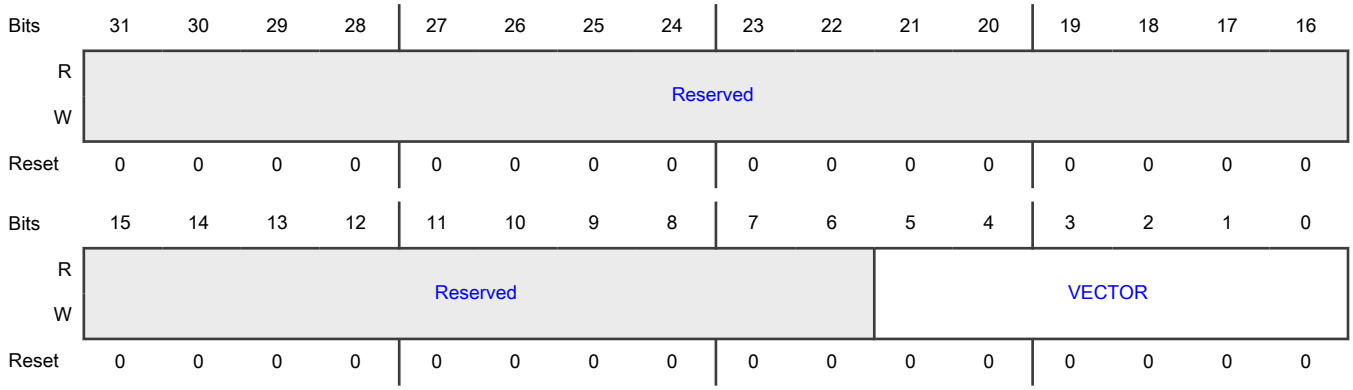
Offset

Register	Offset
SICMSIVR	A34h

Function

This is the command BD ring SI-X vector register which defines the vector used for address/data lookup in the MSI-X Table.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 VECTOR	Vector Index into MSI-X address/data table. Range: 0..SIPCAPR1[NUM_MSIX]-1 **NOTE** Specifying a vector number outside of the range will not generate an MSI-X write.

53.4.6.11.66 Station interface timer interrupt enable register (SITMRIER)

Offset

Register	Offset
SITMRIER	A40h

Function

This is the timer interrupt enable register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															SYNCE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 SYNCE	Timer synchronous state change interrupt enable 0 Disabled 1 Enabled

53.4.6.11.67 Station interface timer interrupt detect register (SITMRIDR)

Offset

Register	Offset
SITMRIDR	A44h

Function

This is the timer interrupt detect register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															SYNC W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 SYNC	Timer synchronous state change detected when set.

53.4.6.11.68 Station interface timer MSI-X vector register (SITMRMSIVR)

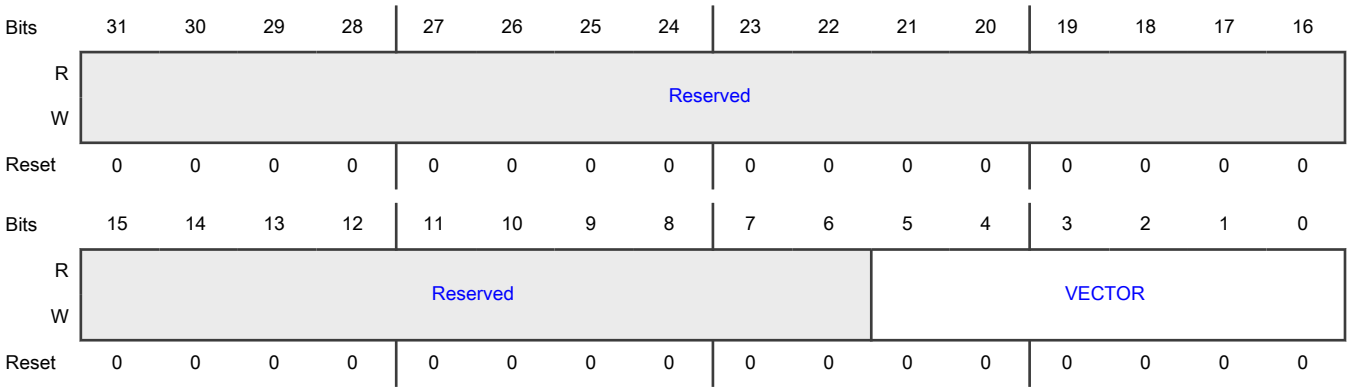
Offset

Register	Offset
SITMRMSIVR	A4Ch

Function

This is the Timer MSI-X vector register which defines the vector used for address/data lookup in the MSI-X Table.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 VECTOR	Vector Index into MSI-X address/data table. Range: 0..SIPCAPR1[NUM_MSIX]-1 **NOTE** Specifying a vector number outside of the range will not generate an MSI-X write.

53.4.6.11.69 Station interface MSI-X transmit ring a vector register (SIMSITR0VR - SIMSITR9VR)

Offset

For a = 0 to 9:

Register	Offset
SIMSITRaVR	B00h + (a × 4h)

Function

This is the MSI-X transmit ring vector register which defines the vector used for address/data lookup in the MSI-X Table. Number of vectors supported is defined in SIPCAPR1[NUM_MSIX].

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	SIMSITR0VR–SIMSITR3VR	SIMSITR4VR–SIMSITR9VR
ENETC1_SI0	SIMSITR0VR–SIMSITR9VR	—
ENETC1_SI1	SIMSITR0VR–SIMSITR9VR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										VECTOR					
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
SIMSITR0VR–SIMSITR3VR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
SIMSITR4VR–SIMSITR9VR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-6 —	Reserved
5-0 VECTOR	Vector Index into MSI-X address/data table. Range: 0..SIPCAPR1[NUM_MSIX]-1 **NOTE** Specifying a vector number outside of the range will not generate an MSI-X write.

53.4.6.11.70 Station interface MSI-X receive ring a vector register (SIMSIRR0VR - SIMSIRR9VR)

Offset

For a = 0 to 9:

Register	Offset
SIMSIRRaVR	B80h + (a × 4h)

Function

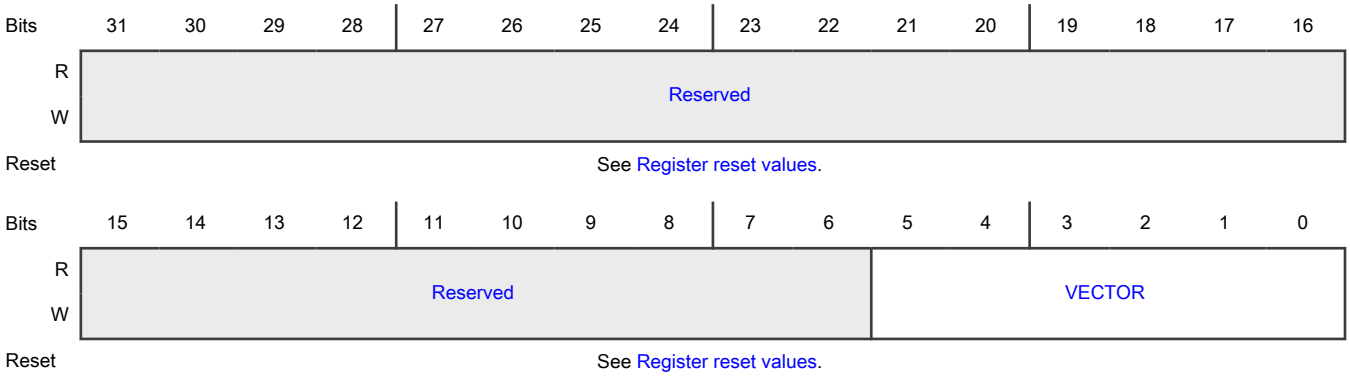
This is the MSI-X receive ring vector register which defines the vector used for address/data lookup in the MSI-X Table. Number of vectors supported is defined in SIPCAPR1[NUM_MSIX].

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	SIMSIRR0VR–SIMSIRR3VR	SIMSIRR4VR–SIMSIRR9VR
ENETC1_SI0	SIMSIRR0VR–SIMSIRR9VR	—
ENETC1_SI1	SIMSIRR0VR–SIMSIRR9VR	—

Diagram



Register reset values

Register	Reset value
SIMSIRR0VR–SIMSIRR3VR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
SIMSIRR4VR–SIMSIRR9VR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-6 —	Reserved
5-0 VECTOR	Vector Index into MSI-X address/data table. Range: 0..SIPCAPR1[NUM_MSIX]-1 **NOTE** Specifying a vector number outside of the range will not generate an MSI-X write.

53.4.6.11.71 Station interface correctable memory error configuration register (SICMECR)

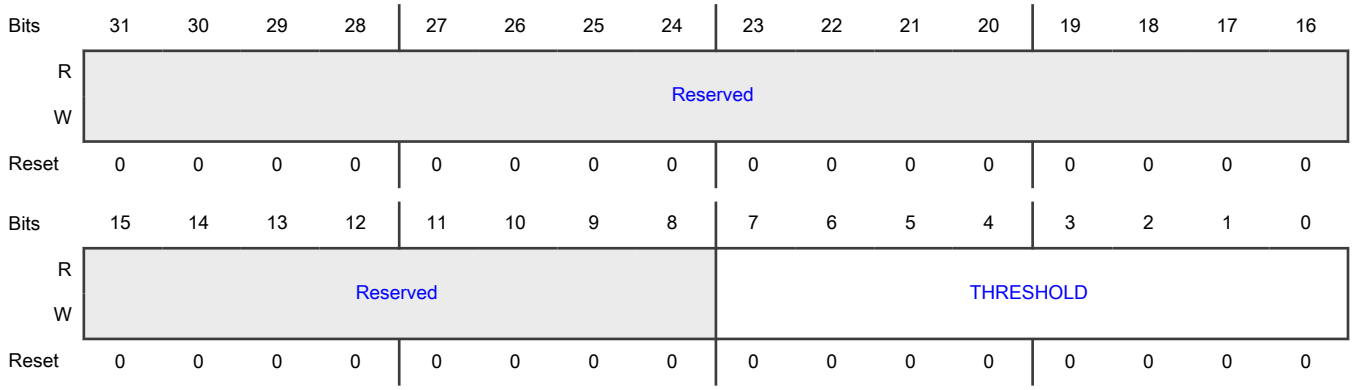
Offset

Register	Offset
SICMECR	E00h

Function

The correctable memory error configuration register can disable the event reporting to the Root Complex Event Collector. While correctable errors do not cause any harm, detecting many of these events could indicate a more severe issue. The threshold can be set to trigger the event.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 THRESHOLD	Threshold Determines how many single bit ECC errors must be detected before error status bit is set and correctable memory error is reported to PCIe Event Collector. 0 Disabled (no reporting) >0 Threshold value (1-255).

53.4.6.11.72 Station interface correctable memory error status register (SICMESR)

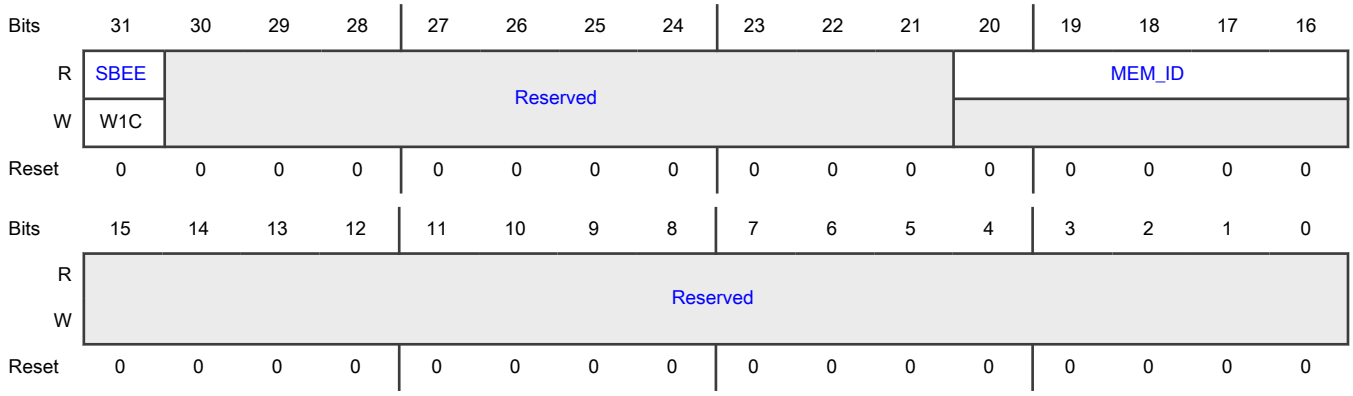
Offset

Register	Offset
SICMESR	E04h

Function

This is the correctable memory error status register.

Diagram



Fields

Field	Function
31 SBEE	Single-bit ECC error When set, a threshold number of single-bit ECC error has occurred in the memory defined by MEM_ID. Write 1 to clear.
30-21 —	Reserved
20-16 MEM_ID	Memory ID See Table 392 for list of memory identifiers.
15-0 —	Reserved

53.4.6.11.73 Station interface correctable memory error count register (SICMECTR)

Offset

Register	Offset
SICMECTR	E0Ch

Function

This is the correctable memory error count register which tracks how many events have been detected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								COUNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Count Indicates how many single-bit ECC error have been detected. After a read operation, the field's value clears to 0.

53.4.6.11.74 Station interface uncorrectable programming error configuration register (SIUPECR)

Offset

Register	Offset
SIUPECR	E10h

Function

The uncorrectable programming error configuration register can disable event reporting to the PCIe function.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RD	Reserved														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 RD	Report disable When reporting is enabled, and SIUPESR[PE] is set, an uncorrectable error is reported to the PCIe function's Advanced Error Reporting capability register, PCIE_CFC_AER_UCORR_ERR_STAT. 0 Enabled 1 Disabled
30-0 —	Reserved

53.4.6.11.75 Station interface uncorrectable programming error status register (SIUPESR)

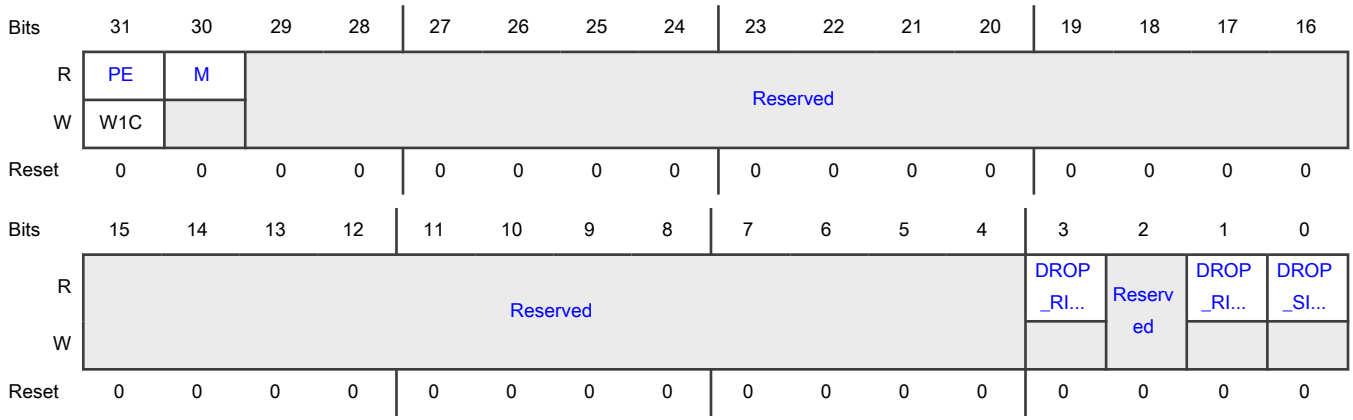
Offset

Register	Offset
SIUPESR	E14h

Function

This is the uncorrectable programming error status register.

Diagram



Fields

Field	Function
31 PE	Programming error When set, an uncorrectable programming error has occurred.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Programming errors are reported as uncorrectable non-fatal errors to Root Complex Event Collector. The order in which these scenarios are checked and reported, if multiple conditions exist, are listed below. 1. Illegal receive BD ring selection • IPV to BDR • RFS BD ring selection out of range • No receive rings mapped to SI 2. Receiving a frame to a disabled SI 3. Receiving a frame to a disabled ring Write 1 to clear.
30 M	Multiple Multiple programming error events of the same kind detected.
29-4 —	Reserved
3 DROP_RING_SEL	Non-existing receive ring error.
2 —	Reserved
1 DROP_RING_EN	Station interface ring disabled drop error.
0 DROP_SI_EN	Station interface disabled drop error.

53.4.6.11.76 Station interface uncorrectable programming error count register (SIUPECTR)

Offset

Register	Offset
SIUPECTR	E1Ch

Function

This is the uncorrectable programming error count register which tracks how many received frames have been dropped by the station interface.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COUNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 COUNT	Count Indicates how many received frames have been dropped due to uncorrectable programming errors. After a read operation, the field's value clears to 0.

53.4.6.11.77 Station interface uncorrectable non-fatal system bus error configuration register (SIUNSBECR)

Offset

Register	Offset
SIUNSBECR	E20h

Function

The uncorrectable non-fatal system bus error configuration register can disable the event reporting to the Root Complex Event Collector.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								THRESHOLD							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 THRESHOLD	Threshold Determines how many non-fatal system bus errors must be detected before error status bit is set and an uncorrectable non-fatal error message is generated to the Root Complex Event Collector. 0 Disabled (no reporting) >0 Threshold value (1-255).

53.4.6.11.78 Station interface uncorrectable non-fatal system bus error status register (SIUNSBESR)

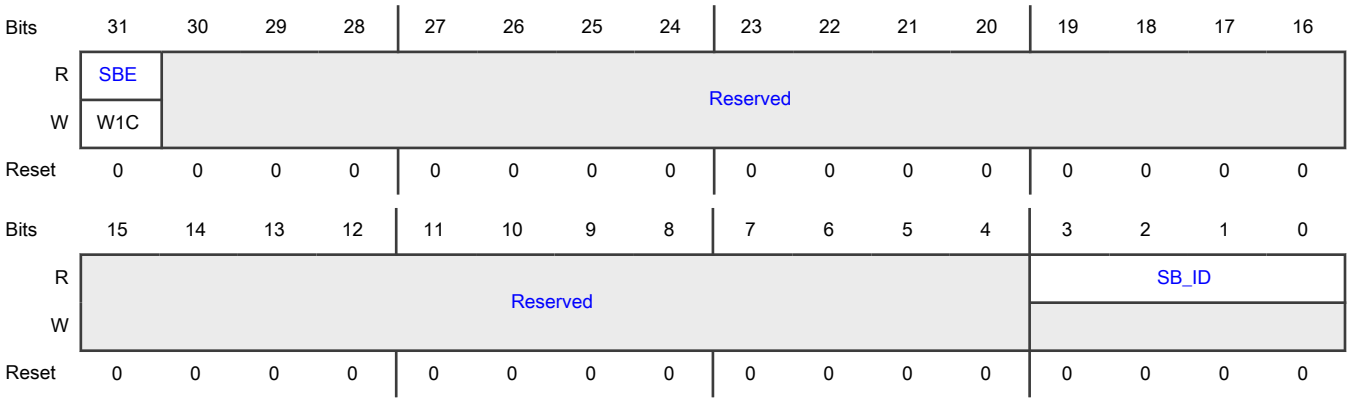
Offset

Register	Offset
SIUNSBESR	E24h

Function

This is the uncorrectable non-fatal system bus error status register.

Diagram



Fields

Field	Function
31 SBE	System bus error When set, a threshold number of system bus errors has occurred due to NETC source attempting to read/write a memory target.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The reaction may result in marking the frame being bad in the Rx BD ring or in stomping on the FCS of the frame if the error is detected while the transmission of the frame is in progress. Write 1 to clear.
30-4 —	Reserved
3-0 SB_ID	System Bus ID See Table 394 for list of system bus source identifiers.

53.4.6.11.79 Station interface uncorrectable non-fatal system bus error count register (SIUNSBECTR)

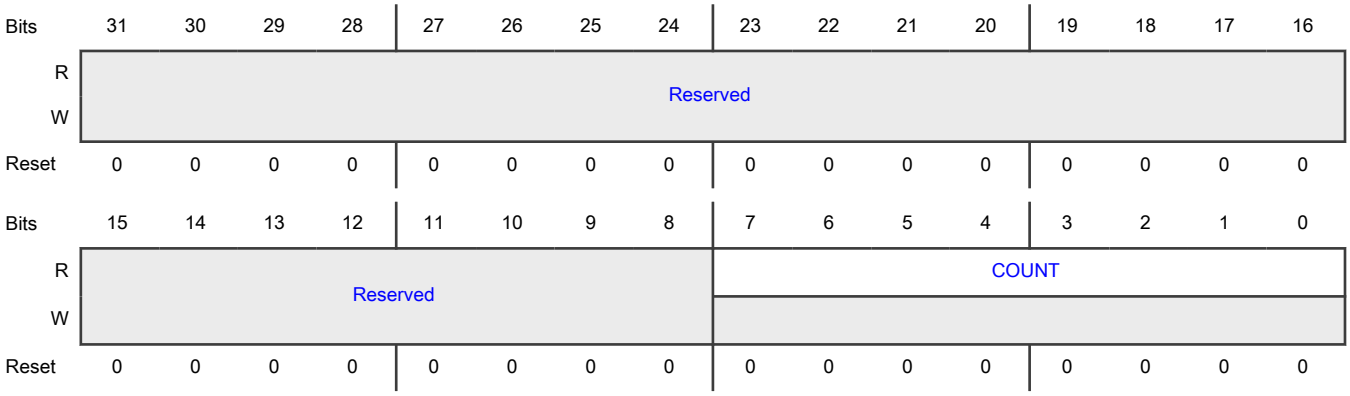
Offset

Register	Offset
SIUNSBECTR	E2Ch

Function

This is the uncorrectable non-fatal system bus error count register which tracks how many events have been detected.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0	Count

Table continues on the next page...

Table continued from the previous page...

Field	Function
COUNT	Indicates how many system bus error events have been encountered. SIUNSBESR[SB_ID] indicates the last source. After a read operation, the field's value clears to 0.

53.4.6.11.80 Station interface uncorrectable fatal system bus error configuration register (SIUFSBECR)

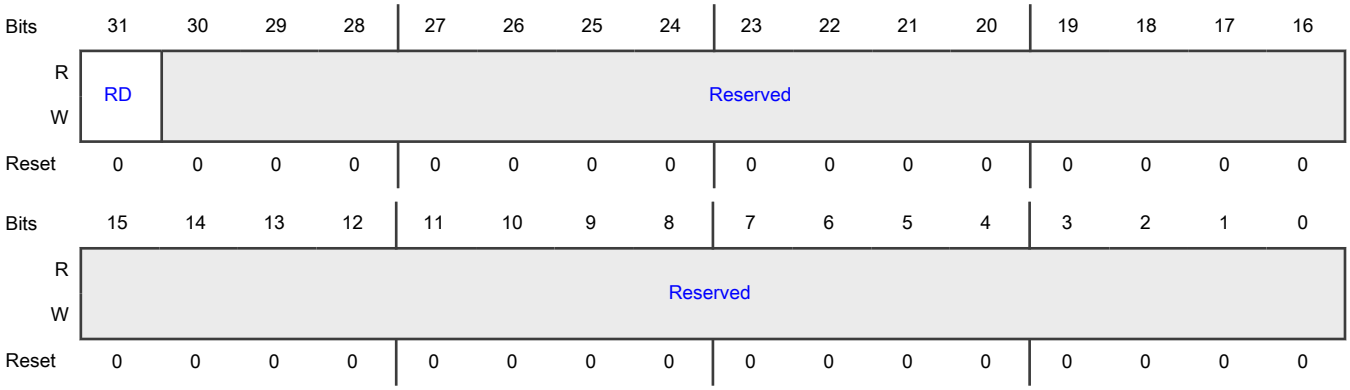
Offset

Register	Offset
SIUFSBECR	E30h

Function

The uncorrectable fatal system bus error configuration register can disable event reporting to the PCIe function.

Diagram



Fields

Field	Function
31 RD	Report disable When reporting is enabled, and SIUFSBESR[SBE] is set, an uncorrectable error is reported to the PCIe function's Advanced Error Reporting capability register, PCIE_CFC_AER_UCORR_ERR_STAT. 0 Enabled 1 Disabled
30-0 —	Reserved

53.4.6.11.81 Station interface uncorrectable fatal system bus error status register (SIUFSBESR)

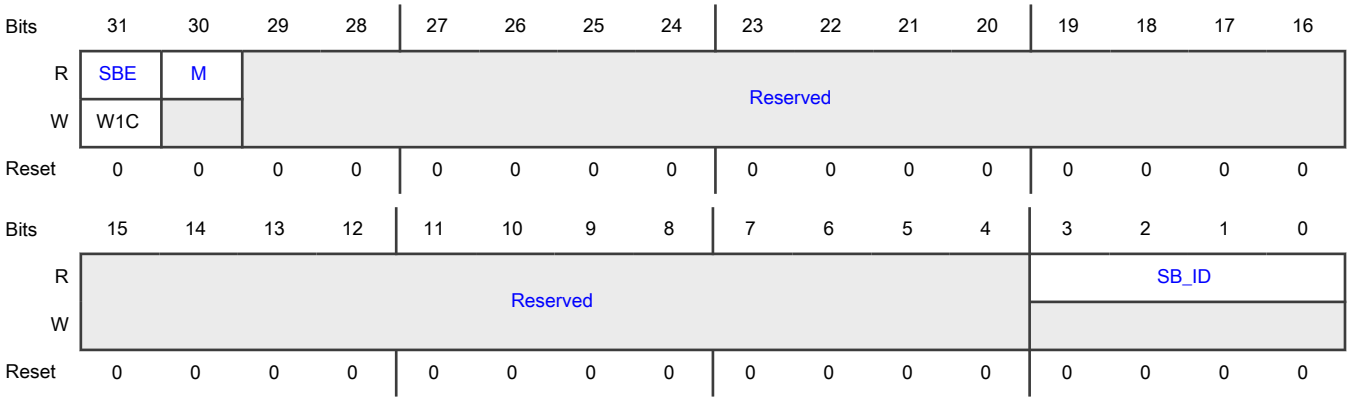
Offset

Register	Offset
SIUFSBESR	E34h

Function

This is the uncorrectable fatal system bus error status register.

Diagram



Fields

Field	Function
31 SBE	System bus error When set, a system bus error occurred when a NETC source attempted to read/write a memory target. The function has entered a safe state: • PCIe bus master enable is cleared (0b0) • Station interface is disabled (SIMR[EN]=0b0) • PSI: Pseudo MAC port is disabled (POR[TXDIS/RXDIS]=0b1) • PSI: Ethernet-MAC port is disabled (PM*a*_COMMAND_CONFIG[RX_EN/TX_EN]=0b0) For recovery, S/W should reset the function by issuing a Function Level Reset (FLR). Write 1 to clear.
30 M	Multiple Multiple system bus error events detected. The last event is captured.
29-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 SB_ID	System Bus ID See Table 394 for list of system bus source identifiers.

53.4.6.11.82 Station interface uncorrectable non-fatal memory error configuration register (SIUNMECR)

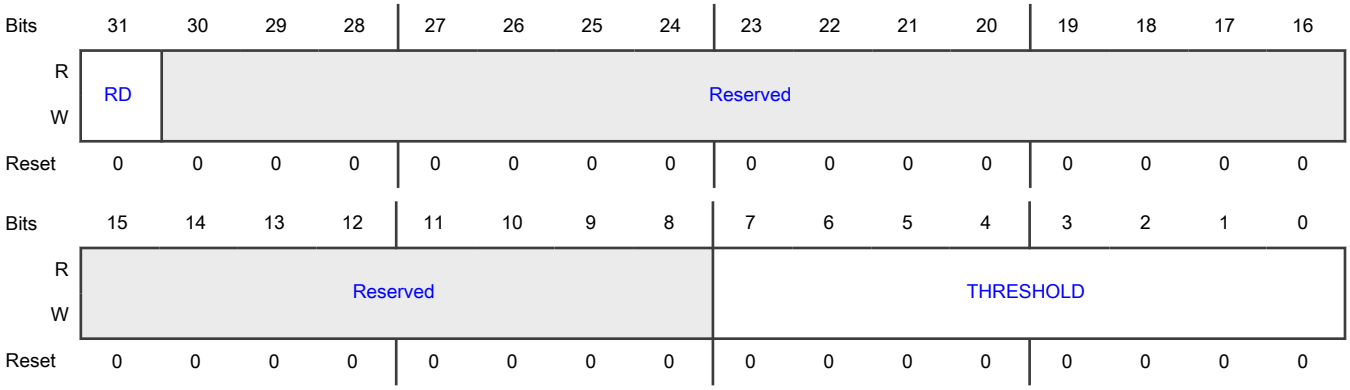
Offset

Register	Offset
SIUNMECR	E40h

Function

The uncorrectable non-fatal memory error configuration register can disable event reporting to the PCIe function.

Diagram



Fields

Field	Function
31 RD	Report disable When reporting is enabled and SIUNMESR0[MBEE] is set, an uncorrectable error is reported to the PCIe function's Advanced Error Reporting capability register, PCIE_CFC_AER_UCORR_ERR_STAT. 0 Enabled 1 Disabled
30-8 —	Reserved
7-0	Threshold

Table continues on the next page...

Table continued from the previous page...

Field	Function
THRESHOLD	Determines how many non-fatal memory errors must be detected before error status bit is set. 0 Disabled (no reporting) >0 Threshold value (1-255).

53.4.6.11.83 Station interface uncorrectable non-fatal memory error status register 0 (SIUNMESR0)

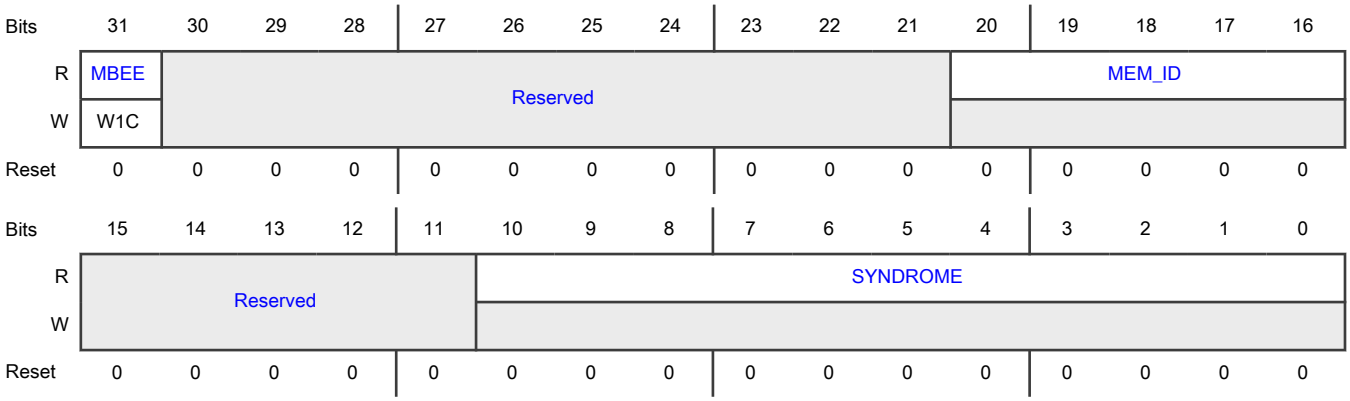
Offset

Register	Offset
SIUNMESR0	E44h

Function

This is the uncorrectable non-fatal memory error status register 0.

Diagram



Fields

Field	Function
31 MBEE	Multi-bit ECC error When set, a threshold number of multi-bit ECC errors has occurred in internal memory as defined by MEM_ID. The reaction to the error may result in marking the frame being bad in the Rx BD ring or in stomping on the FCS of the frame if the error is detected while the transmission of the frame is in progress. Write 1 to clear.
30-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20-16 MEM_ID	Memory ID See Table 392 for list of memory identifiers.
15-11 —	Reserved
10-0 SYNDROME	Syndrome Identifies the first pertinent bit position (column) in memory that caused the error.

53.4.6.11.84 Station interface uncorrectable non-fatal memory error status register 1 (SIUNMESR1)

Offset

Register	Offset
SIUNMESR1	E48h

Function

This is the uncorrectable memory error status register 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ADDR	Address Address information (row) of last ECC event.

53.4.6.11.85 Station interface uncorrectable non-fatal memory error count register (SIUNMECTR)

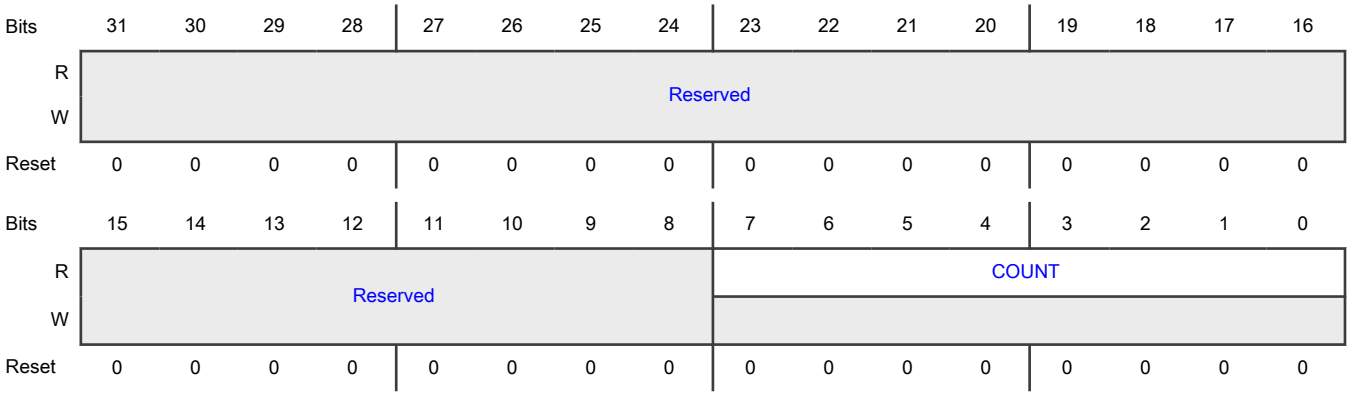
Offset

Register	Offset
SIUNMECTR	E4Ch

Function

This is the uncorrectable non-fatal memory error count register which tracks how many events have been detected.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Count Indicates how many frames have been dropped due to internal memory error. After a read operation, the field's value clears to 0.

53.4.6.11.86 Station interface uncorrectable fatal memory error configuration register (SIUFMECR)

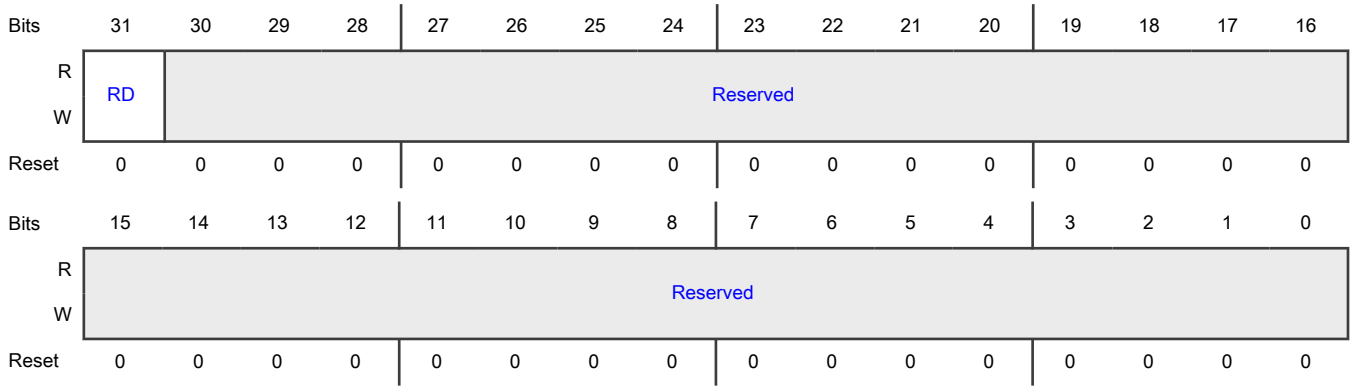
Offset

Register	Offset
SIUFMECR	E50h

Function

The uncorrectable fatal memory error configuration register can disable event reporting to the PCIe function

Diagram



Fields

Field	Function
31 RD	Report disable When reporting is enabled, and SIUFMESR0[MBEE] is set, an uncorrectable error is reported to the PCIe function's Advanced Error Reporting capability register, PCIE_CFC_AER_UCORR_ERR_STAT. 0 Enabled 1 Disabled
30-0 —	Reserved

53.4.6.11.87 Station interface uncorrectable fatal memory error status register 0 (SIUFMESR0)

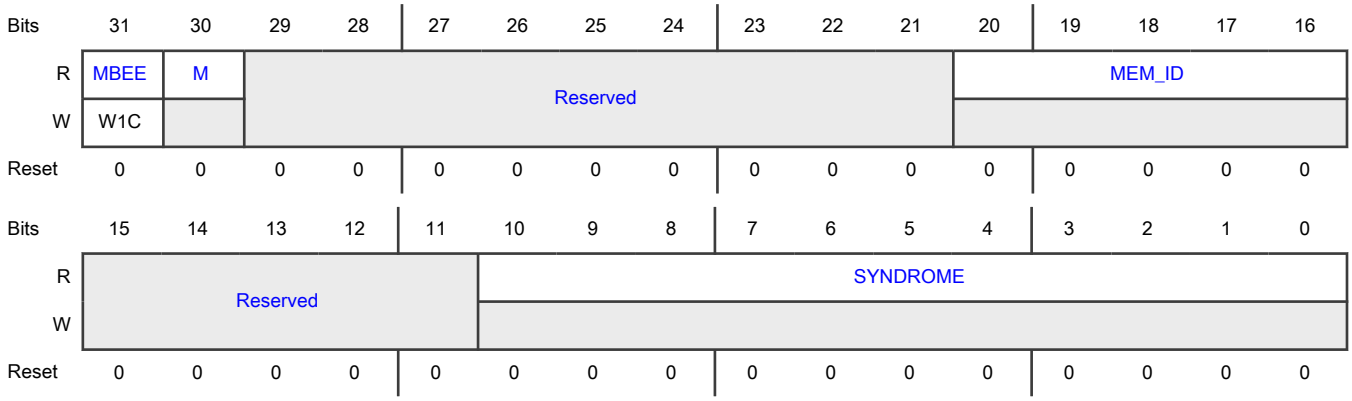
Offset

Register	Offset
SIUFMESR0	E54h

Function

This is the uncorrectable fatal memory error status register 0.

Diagram



Fields

Field	Function
31 MBEE	Multi-bit ECC error When set, a fatal multi-bit ECC error has occurred in internal memory as defined by MEM_ID. The function has entered a safe state: • PCIe bus master enable is cleared (0b0) • Station interface is disabled (SIMR[EN]=0b0) • PSI: Pseudo MAC port is disabled (POR[TXDIS/RXDIS]=0b1) • PSI: Ethernet-MAC port is disabled (PM*a*_COMMAND_CONFIG[RX_EN/TX_EN]=0b0) For recovery, S/W should reset the function by issuing a Function Level Reset (FLR). Write 1 to clear.
30 M	Multiple Multiple fatal multi-bit ECC error events detected. The last event is captured.
29-21 —	Reserved
20-16 MEM_ID	Memory ID See Table 392 for list of memory identifiers.
15-11 —	Reserved
10-0 SYNDROME	Syndrome Identifies the first pertinent bit position (column) in memory that caused the error.

53.4.6.11.88 Station interface uncorrectable fatal memory error status register 1 (SIUFMESR1)

Offset

Register	Offset
SIUFMESR1	E58h

Function

This is the uncorrectable fatal memory error status register 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ADDR	Address Address information (row) of last ECC event.

53.4.6.11.89 Station interface MAC address filter table capability register (SIMAFTCAPR)

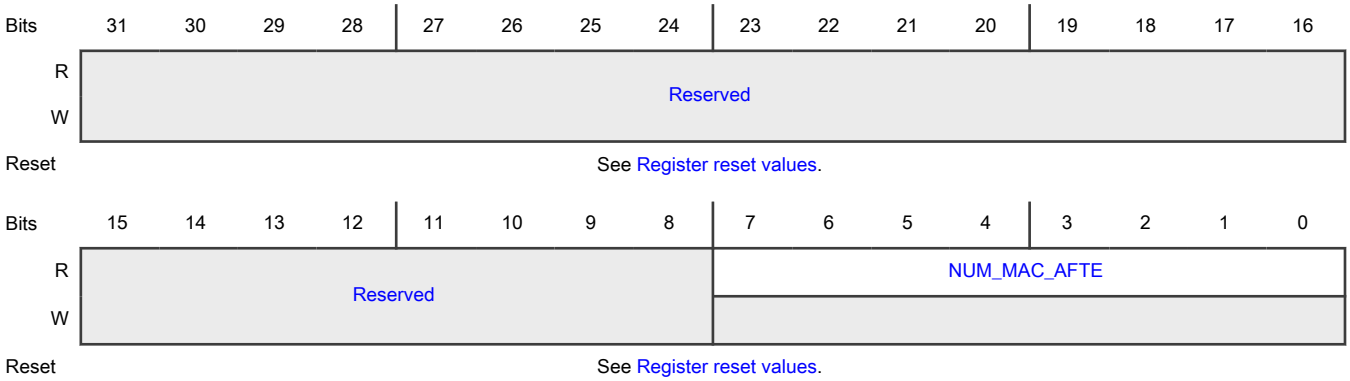
Offset

Register	Offset
SIMAFTCAPR	1000h

Function

This is the station interface MAC address filter table capability register.

Diagram



Register reset values

Register	Reset value
SIMAFTCAPR	ENETC0_SI0,ENETC1_SI0: 0000_0004h ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-8 —	Reserved
7-0 NUM_MAC_AFTE	Number of MAC address filter table entries for use by SI

53.4.6.11.90 Station interface VLAN filter table capability register (SIVFTCAPR)

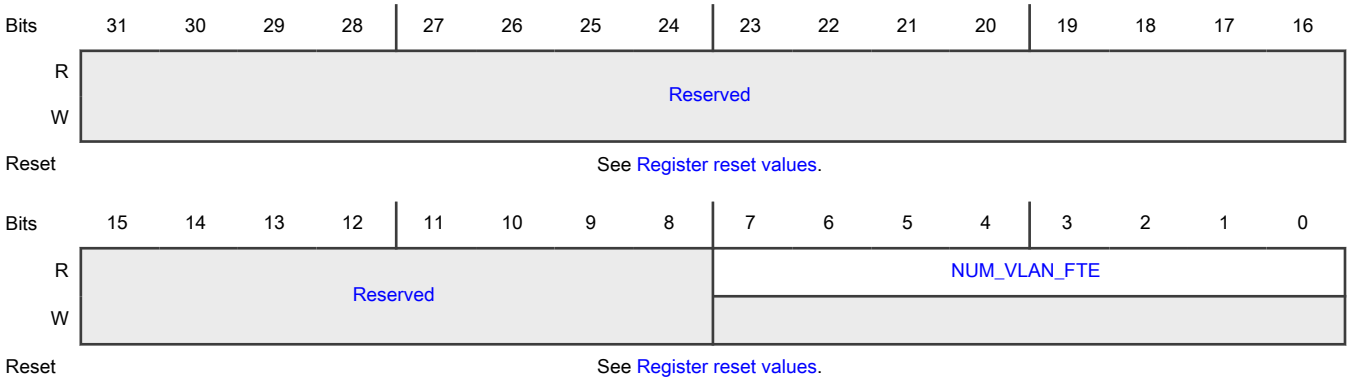
Offset

Register	Offset
SIVFTCAPR	1100h

Function

This is the station interface VLAN filter table capability register.

Diagram



Register reset values

Register	Reset value
SIVFTCAPR	ENETC0_SI0,ENETC1_SI0: 0000_0004h ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-8 —	Reserved
7-0 NUM_VLAN_FTE	Number of VLAN filter table entries for use by SI

53.4.6.11.91 Tx BDR a mode register (TB0MR - TB9MR)

Offset

For a = 0 to 9:

Register	Offset
TBaMR	8000h + (a × 200h)

Function

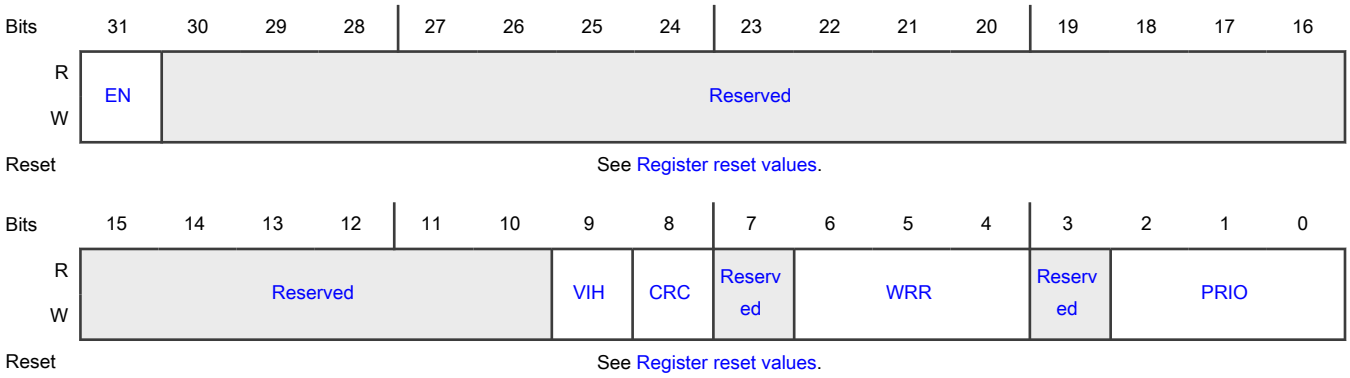
This is the mode register for a transmit buffer descriptor ring.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	TB0MR–TB3MR	TB4MR–TB9MR
ENETC1_SI0	TB0MR–TB9MR	—
ENETC1_SI1	TB0MR–TB9MR	—

Diagram



Register reset values

Register	Reset value
TB0MR–TB3MR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
TB4MR–TB9MR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31 EN	Enable transmit buffer descriptor ring 0: Disabled. 1: Enabled. When the ring is non-empty, transmit packets will be processed. **NOTE** It is not safe to clear this bit while the ring is actively transmitting frames. Software should only disable an active ring after all pending ring entries have been consumed (i.e. when PI = CI). Disabling a transmit ring that is actively processing BDs risks a HW-SW race hazard whereby a hardware resource becomes assigned to work on one or more ring entries only to have those entries be removed due to the ring becoming disabled.
30-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 VIH	VLAN Insert Hint If set (b1) then SW intends to use VLAN insertion offload by providing VLAN tag data in the Tx BD. When set, calculations performed by hardware in the transmit scheduler involving the size of the frame (e.g. credit based shaper) will account for the extra 4 bytes of VLAN information in their calculations regardless of whether descriptor-based VLAN insertion offload is used or not used.
8 CRC	User CRC provided Determines if user provides CRC32 (FCS) at the end of the frame. 0: User CRC (FCS) is not provided (calculated and appended by the hardware). Calculations performed by hardware in the transmit scheduler involving the size of the frame (e.g. credit based shaper) will account for the extra 4 bytes of FCS in their calculations. 1: User CRC (FCS) is provided; the hardware does not append any FCS to the frame. If the port is configured to pad frames to the minimum length of 64 bytes (PPMCR[TXPAD]=1 for pseudo MAC or PMa_COMMAND_CONFIG[TXP]=1 for Ethernet MAC), frames (with FCS) provided by the user must be 64 bytes or longer. If frames are shorter than 64 bytes, frames will be padded to 64 bytes and transmitted with a bad FCS. **NOTE** If this field is set to 1 (user CRC provided) and offloading of VLAN tag insertion is requested (i.e. SI-Based VLAN insertion or descriptor-based VLAN insertion), then the frame will not be transmitted as the FCS would be bad due to the frame modification. The STATUS field of host transmit descriptor is updated with Programming Error flag set to 1.
7 —	Reserved
6-4 WRR	WRR weight Weight used for arbitration when two or more rings have the same priority within the same VSI. A higher value indicates more bandwidth. 000 Weight of 1 001 Weight of 2 ... 110 Weight of 7 111 Weight of 8
3 —	Reserved
2-0 PRIO	Priority Priority of the transmit buffer descriptor ring. 000 Lowest priority 001 Next lowest priority

Table continues on the next page...

Table continued from the previous page...

Field	Function
	... 110 Next highest priority 111 Highest priority

53.4.6.11.92 Tx BDR a status register (TB0SR - TB9SR)

Offset

For a = 0 to 9:

Register	Offset
TBaSR	8004h + (a × 200h)

Function

This is the transmit buffer descriptor ring status register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	TB0SR–TB3SR	TB4SR–TB9SR
ENETC1_SI0	TB0SR–TB9SR	—
ENETC1_SI1	TB0SR–TB9SR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															SBE
W																W1C
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															BUSY
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
TB0SR–TB3SR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
TB4SR–TB9SR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-17 —	Reserved
16 SBE	System bus error A system bus error has occurred during one or more transactions related to this transmit ring, including possibly the transmit BD writeback entry itself. To avoid the possibility of referencing unreliable BD writeback contents, SBE can be checked at every observed TB*a*CIR movement. See error detect register SIUNSBESR/SIUFSBESR for more information. Write 1 to clear.
15-1 —	Reserved
0 BUSY	Busy. The transmit ring is busy processing packets 0 Idle, or programming error SIEDR[PE] is set. 1 Busy

53.4.6.11.93 Tx BDR a base address register 0 (TB0BAR0 - TB9BAR0)

Offset

For a = 0 to 9:

Register	Offset
TBaBAR0	8010h + (a × 200h)

Function

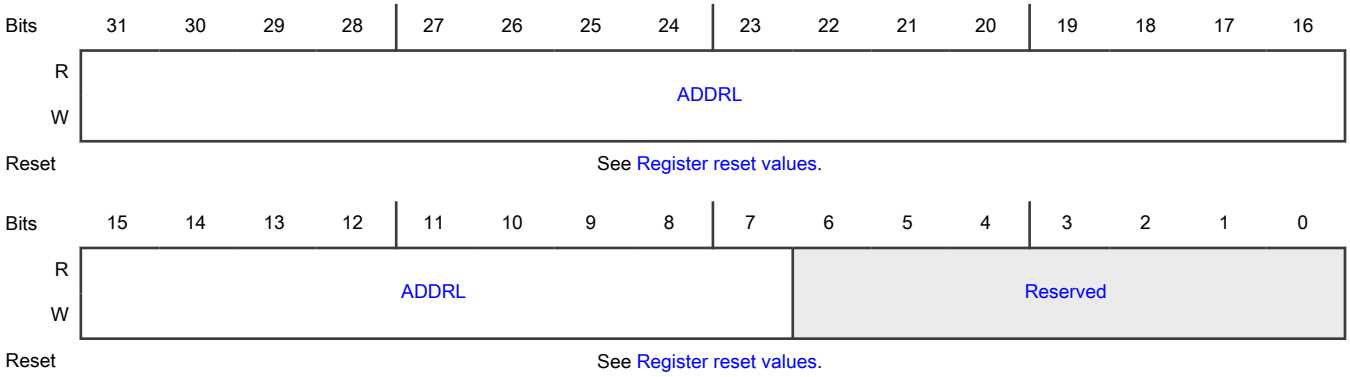
This is the lower address register for the base memory address location of the transmit ring. The address must be 128 byte aligned.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	TB0BAR0–TB3BAR0	TB4BAR0–TB9BAR0
ENETC1_SI0	TB0BAR0–TB9BAR0	—
ENETC1_SI1	TB0BAR0–TB9BAR0	—

Diagram



Register reset values

Register	Reset value
TB0BAR0–TB3BAR0	ENETC0_SI0–ENETC1_SI1: 0000_0000h
TB4BAR0–TB9BAR0	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-7 ADDRL	Lower address bits 31-7.
6-0 —	Reserved

53.4.6.11.94 Tx BDR a base address register 1 (TB0BAR1 - TB9BAR1)

Offset

For a = 0 to 9:

Register	Offset
TBaBAR1	8014h + (a × 200h)

Function

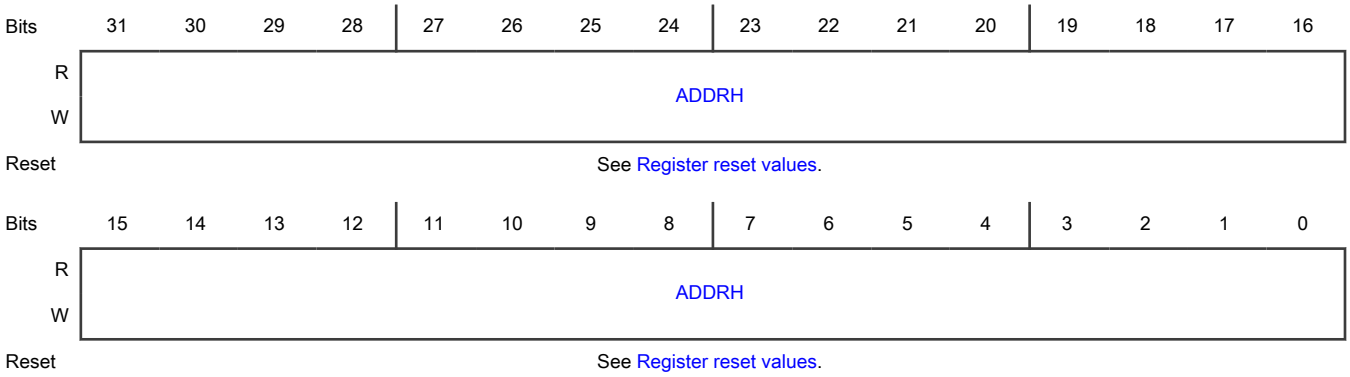
This is the upper address register for the base memory address location of the transmit ring.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	TB0BAR1–TB3BAR1	TB4BAR1–TB9BAR1
ENETC1_SI0	TB0BAR1–TB9BAR1	—
ENETC1_SI1	TB0BAR1–TB9BAR1	—

Diagram



Register reset values

Register	Reset value
TB0BAR1–TB3BAR1	ENETC0_SI0–ENETC1_SI1: 0000_0000h
TB4BAR1–TB9BAR1	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-0 ADDRH	Upper address bits 63-32.

53.4.6.11.95 Tx BDR a producer index register (TB0PIR - TB9PIR)

Offset

For a = 0 to 9:

Register	Offset
TBaPIR	8018h + (a × 200h)

Function

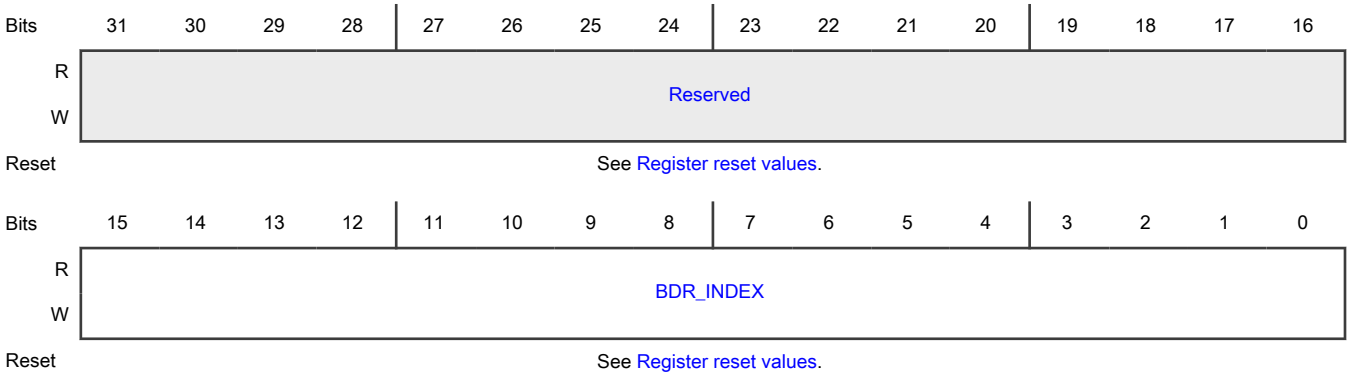
This is the transmit BDR producer index register. When the BD ring is enabled the value of this register and TB*a*CIR determines NETC's unprocessed BDs of the ring. This value also determines how many BD's are committed to the BDR by software. Software should initialize this register while the ring is disabled to reflect an empty state or a pre-loaded BDR state. Software can calculate available entries in the ring by using the software consumer index and this register value.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	TB0PIR–TB3PIR	TB4PIR–TB9PIR
ENETC1_SI0	TB0PIR–TB9PIR	—
ENETC1_SI1	TB0PIR–TB9PIR	—

Diagram



Register reset values

Register	Reset value
TB0PIR–TB3PIR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
TB4PIR–TB9PIR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-16 —	Reserved
15-0 BDR_INDEX	Transmit buffer descriptor ring producer index. Range of index depends on ring size TB*a*LENR[TBLENR].

53.4.6.11.96 Tx BDR a consumer index register (TB0CIR - TB9CIR)

Offset

For a = 0 to 9:

Register	Offset
TBaCIR	801Ch + (a × 200h)

Function

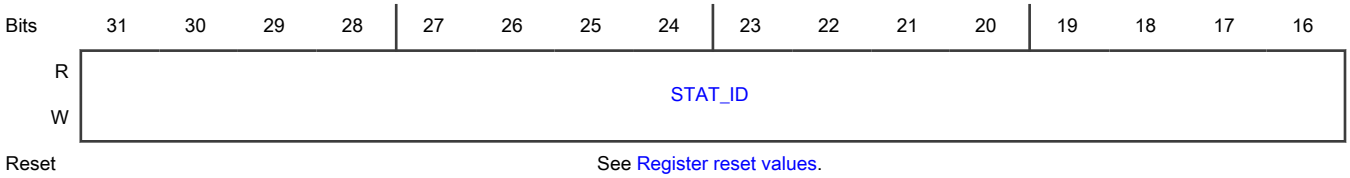
This is the transmit BDR consumer index register. It reflects the next BD to be processed by NETC and trails the TB*a*PIR index value. Software should only update/reset this value when the BD ring is re-initialized. Disabling the BD ring has no effect on this register. During operation, hardware updates the consumer index to reflect frames that have completed transmission and whose BDs are committed to memory, i.e. status/timestamp updated.

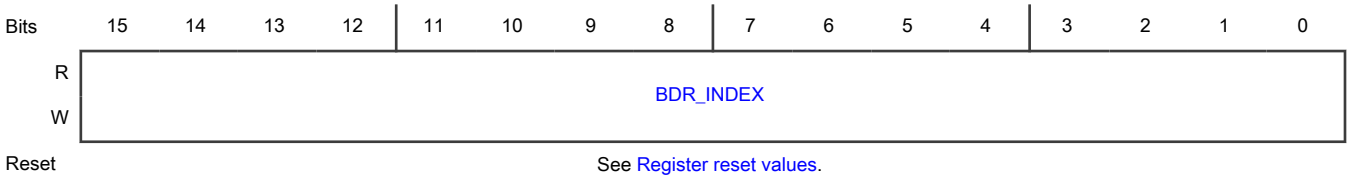
NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	TB0CIR–TB3CIR	TB4CIR–TB9CIR
ENETC1_SI0	TB0CIR–TB9CIR	—
ENETC1_SI1	TB0CIR–TB9CIR	—

Diagram





Register reset values

Register	Reset value
TB0CIR–TB3CIR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
TB4CIR–TB9CIR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-16 STAT_ID	Status identifier. Incremented each time the BDR_INDEX is updated and an error status bit was set for one of the processed BDs. Clears on read. After a read operation, the field's value clears to 0.
15-0 BDR_INDEX	Transmit buffer descriptor ring consumer index. Range of index depends on ring size TB*a*LENR[LENGTH].

53.4.6.11.97 Tx BDR a length register (TB0LENR - TB9LENR)

Offset

For a = 0 to 9:

Register	Offset
TBaLENR	8020h + (a × 200h)

Function

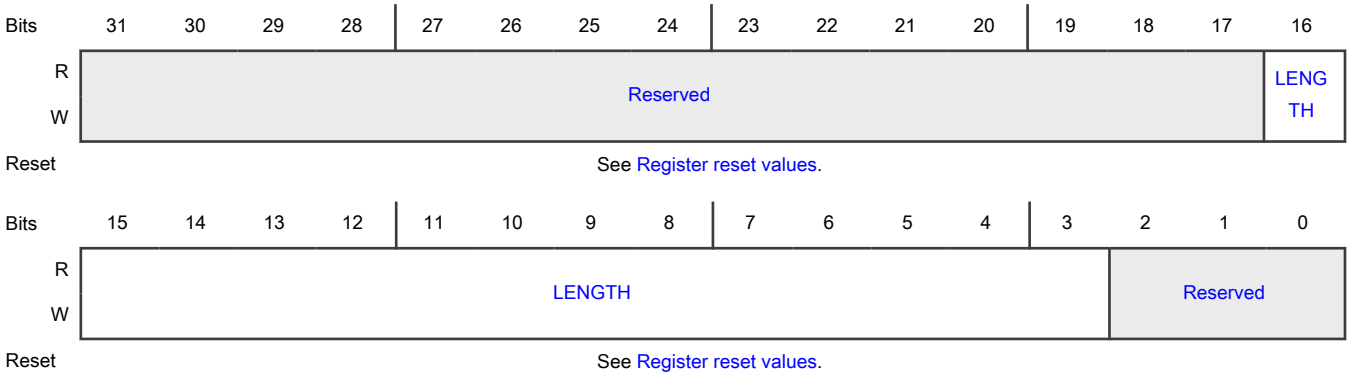
This is the transmit BDR length register. It determines the size of the transmit ring in sets of 8 BDs.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	TB0LENR–TB3LENR	TB4LENR–TB9LENR
ENETC1_SI0	TB0LENR–TB9LENR	—
ENETC1_SI1	TB0LENR–TB9LENR	—

Diagram



Register reset values

Register	Reset value
TB0LENR–TB3LENR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
TB4LENR–TB9LENR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-17 —	Reserved
16-3 LENGTH	BD ring length Size of ring in sets of 8 BDs. Maximum ring size is 64K.
2-0 —	Reserved

53.4.6.11.98 Tx BDR a interrupt enable register (TB0IER - TB9IER)

Offset

For a = 0 to 9:

Register	Offset
TBaIER	80A0h + (a × 200h)

Function

This is the interrupt enable register for the transmit buffer descriptor ring. If the interrupt is enabled while the corresponding detect bit in TB*a*IDR is set, it will trigger an interrupt event for the SI based on the MSI-X vector defined in SIMSITR*a*VR.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	TB0IER–TB3IER	TB4IER–TB9IER
ENETC1_SI0	TB0IER–TB9IER	—
ENETC1_SI1	TB0IER–TB9IER	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														TXFIE	TXTIE
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
TB0IER–TB3IER	ENETC0_SI0–ENETC1_SI1: 0000_0000h
TB4IER–TB9IER	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-2 —	Reserved
1 TXFIE	Transmit frame interrupt enable 0 Disabled 1 Enabled
0 TXTIE	Transmit threshold interrupt enable 0 Disabled 1 Enabled

53.4.6.11.99 Tx BDR a interrupt detect register (TB0IDR - TB9IDR)

Offset

For a = 0 to 9:

Register	Offset
TBaIDR	80A4h + (a × 200h)

Function

This is the interrupt detect register for the transmit buffer descriptor ring. Reading will automatically clear all events. Register SITXIDR provides a non-destructive read access. It is not necessary to clear/set the TBaIER register upon entry/exit of the service routine. A new interrupt is generated in the event of a set/clear collision in this register.

TBaIDR should not be polled if interrupt coalescing is enabled. Polling reads of TBaIDR clear the frame counter and reset the interrupt timer prematurely, preventing an interrupt from occurring.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	TB0IDR–TB3IDR	TB4IDR–TB9IDR
ENETC1_SI0	TB0IDR–TB9IDR	—
ENETC1_SI1	TB0IDR–TB9IDR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														TXF	TXT
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
TB0IDR–TB3IDR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
TB4IDR–TB9IDR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-2 —	Reserved
1 TXF	Transmit of frame 0 No transmit of frame detected with BD[FI]=1 1 Transmit of frame detected with BD[FI]=1
0 TXT	Transmit threshold 0 No threshold event detected 1 Transmit ring has transmitted at least the number of packets specified by TB*a*ICR0[ICPT] or the threshold timer has expired since last transmitted packet as specified by TB*a*ICR1[ICTT].

53.4.6.11.100 Tx BDR a interrupt coalescing register 0 (TB0ICR0 - TB9ICR0)

Offset

For a = 0 to 9:

Register	Offset
TBaICR0	80A8h + (a × 200h)

Function

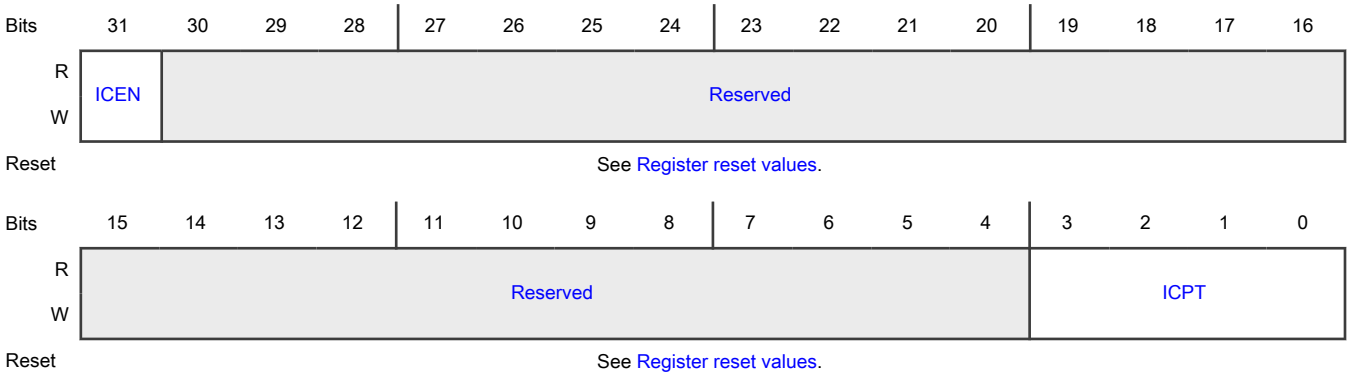
The transmit ring interrupt coalescing register enables and configures the parameters for interrupt coalescing associated with transmitted packets.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	TB0ICR0–TB3ICR0	TB4ICR0–TB9ICR0
ENETC1_SI0	TB0ICR0–TB9ICR0	—
ENETC1_SI1	TB0ICR0–TB9ICR0	—

Diagram



Register reset values

Register	Reset value
TB0ICR0–TB3ICR0	ENETC0_SI0–ENETC1_SI1: 0000_0000h
TB4ICR0–TB9ICR0	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31 ICEN	Interrupt coalescing enable 0 Interrupt coalescing is disabled 1 Interrupt coalescing is enabled. If the transmit ring threshold interrupt is enabled (TB*a*IER[TXTIE] is set), an interrupt is raised when the threshold number of packets is reached (as defined by TB*a*ICR0[ICPT]) or when the threshold timer expires (as defined by TB*a*ICR1[ICTT]).
30-4 —	Reserved
3-0 ICPT	Interrupt coalescing packet threshold While interrupt coalescing is enabled, ICEN=1, this values determines the minimum number of packets transmitted before raising an interrupt. The counter starts when ICEN is first set and one of the following events will restart the threshold counter: • Disabling transmit ring (TB*a*MR[EN]=0) • Reading TB*a*IDR • Writing a 1 to SITXIDR[TXT*a*] Settings: 0 (disabled), otherwise set to N where number of packets transmitted equals $2^{(N-1)}$ 0000 Disabled 0001 1 packet transmitted

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0010 2 packets transmitted 0011 4 packets transmitted 0100 8 packets transmitted ... 1100 2048 packets transmitted 1101 4096 packets transmitted 1110 8192 packets transmitted 1111 16384 packets transmitted **NOTE** If packet threshold is used it should be set when TB*a*ICR0[ICEN] is set (0 to 1 transition).

53.4.6.11.101 Tx BDR a interrupt coalescing register 1 (TB0ICR1 - TB9ICR1)

Offset

For a = 0 to 9:

Register	Offset
TBaICR1	80ACh + (a × 200h)

Function

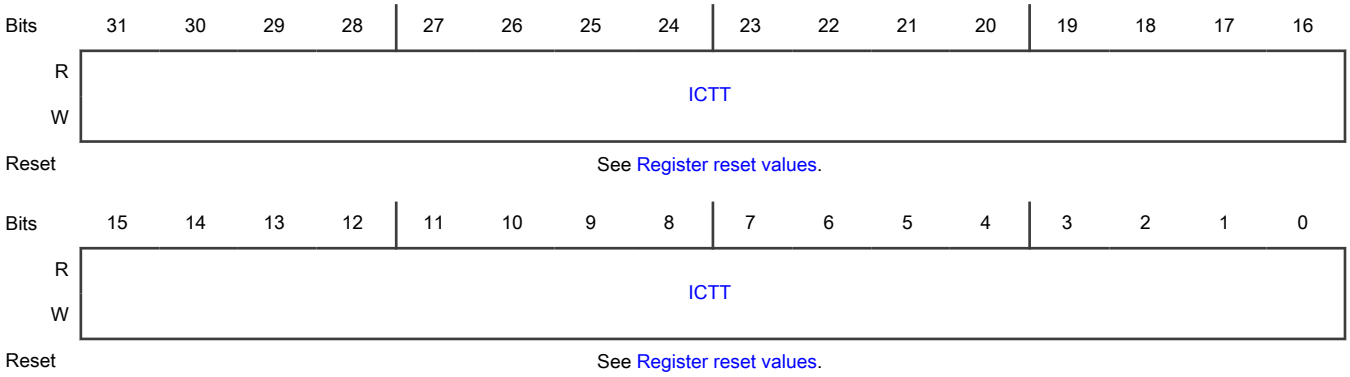
The transmit ring interrupt coalescing register enables and configures the parameters for interrupt coalescing associated with transmitted packets.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	TB0ICR1–TB3ICR1	TB4ICR1–TB9ICR1
ENETC1_SI0	TB0ICR1–TB9ICR1	—
ENETC1_SI1	TB0ICR1–TB9ICR1	—

Diagram



Register reset values

Register	Reset value
TB0ICR1–TB3ICR1	ENETC0_SI0–ENETC1_SI1: 0000_0000h
TB4ICR1–TB9ICR1	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-0 ICTT	Interrupt coalescing timer threshold, specified in units of NETC clock cycles. When interrupt coalescing is enabled, TB*a*ICR0[ICEN]=1, this value determines the maximum amount of time allowed between a first transmitted packet until the TB*a*ICR0[ICPT] threshold is reached. If the timer expires before threshold reached, the interrupt event TB*a*IDR[TXT] is asserted. The timer operates on the NETC platform clock. A register value of zero indicates the timer is disabled. One of the following conditions will reset the timer: • Disabling transmit ring (TB*a*MR[EN]=0) • Reading TB*a*IDR • Writing a 1 to SITXIDR[TXT*a*] **NOTE** If timer threshold is used it should be set when TB*a*ICR0[ICEN]=0 to avoid missing a "first packet" transmitted condition which starts the timer.

53.4.6.11.102 Rx BDR a mode register (RB0MR - RB9MR)

Offset

For a = 0 to 9:

Register	Offset
RBaMR	8100h + (a × 200h)

Function

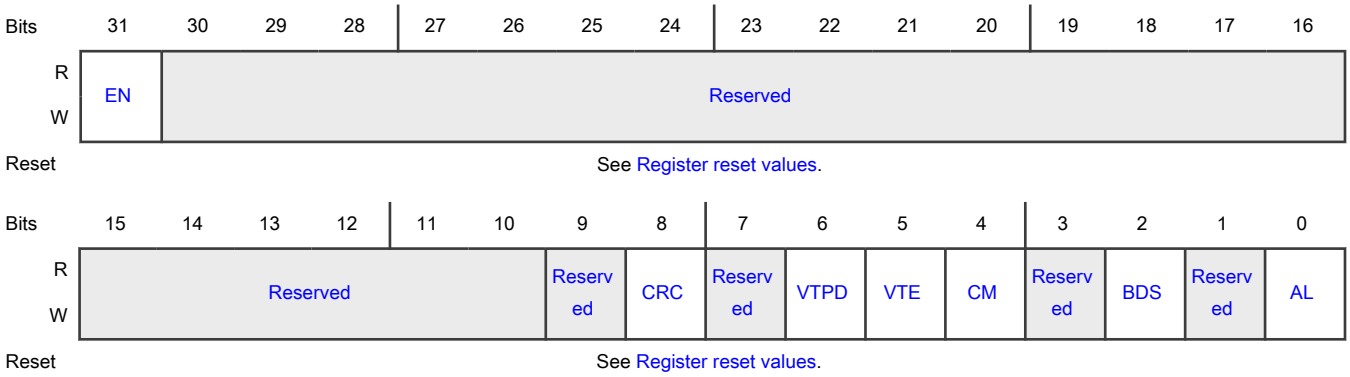
This is the mode register for a receive buffer descriptor ring.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0MR–RB3MR	RB4MR–RB9MR
ENETC1_SI0	RB0MR–RB9MR	—
ENETC1_SI1	RB0MR–RB9MR	—

Diagram



Register reset values

Register	Reset value
RB0MR–RB3MR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4MR–RB9MR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31 EN	Enable receive buffer descriptor ring 0 Disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1 Enabled.
30-10 —	Reserved
9 —	Reserved
8 CRC	Indicates if CRC32 (FCS) is to be stripped or preserved at the end of the frame. 0: CRC (FCS) is stripped 1: CRC (FCS) is preserved **NOTE** If this field is set to 1 (CRC is preserved) and offloading of VLAN tag removal is requested then the CRC of the frame delivered to the host will be bad as the CRC is not updated as a result of the frame modification.
7 —	Reserved
6 VTPD	VLAN tag presentation disable Controls whether the extracted VLAN tag is provided in the Rx BD. If set (b1) this presentation is disabled. Valid only if the VTE bit in this register is set to 1.
5 VTE	VLAN tag extract enable Controls whether the outer VLAN, as seen by the SI, is extracted. Note that this may be the second VLAN in the frame, if the actual outer VLAN matches the SI-based VLAN for this SI and the PSI has enabled removal of that tag. 0 Disabled 1 Enabled
4 CM	Congestion mode Determines the congestion scheme for the receive ring. In lossless mode, when a frame encounters lack of buffers to complete writing the frame to memory, it will back-pressure to the Ingress Congestion Manager (ICM) which in turn can start generating PAUSE frames when received frames are starting to consume too much of the internal frame memory. In lossy mode, tail drop is invoked to handle fullness of the ring. 0 Lossy 1 Lossless

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is not applicable to transmit timestamp reference responses which are identified by their Host Reason code (0x3 Timestamp Response). If a receive BD ring reaches full level (i.e. no BDs/buffers are available when a transmit timestamp reference response is ready to be delivered to the host), HTA will wait for more BDs to be added to the ring by software; the transmit timestamp reference response will not be dropped.
3 —	Reserved
2 BDS	BD Size A BD ring can use either the standard 16B or extended 32B buffer descriptor format. 0 16B buffer descriptors format 1 32B buffer descriptors format
1 —	Reserved
0 AL	Header alignment If set, an additional 2-byte alignment is applied to the header in the 1st buffer descriptor. It can be used to align the IP address in the frame. 0 No alignment added 1 +2B alignment added to frame.

53.4.6.11.103 Rx BDR a status register (RB0SR - RB9SR)

Offset

For a = 0 to 9:

Register	Offset
RBaSR	8104h + (a × 200h)

Function

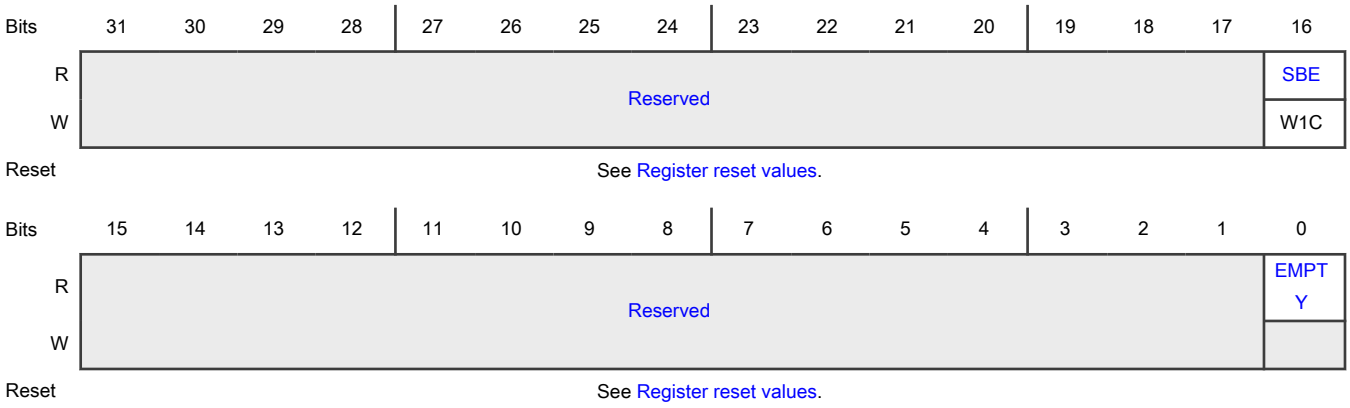
This is the status register for a receive buffer descriptor ring.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0SR–RB3SR	RB4SR–RB9SR
ENETC1_SI0	RB0SR–RB9SR	—
ENETC1_SI1	RB0SR–RB9SR	—

Diagram



Register reset values

Register	Reset value
RB0SR–RB3SR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4SR–RB9SR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-17 —	Reserved
16 SBE	System bus error A system bus error has occurred during one or more transactions related to this receive ring, including possibly the receive BD writeback entry itself. To avoid the possibility of referencing unreliable BD writeback contents, SBE can be checked at every observed RB*a*PIR movement. See error detect register SIUNSBESR/SIUFSBESR for more information. Write 1 to clear.
15-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 EMPTY	Ring is empty Indicates the ring is empty of received BDs. Valid when RB*a*MR[EN]=1.

53.4.6.11.104 Rx BDR a buffer size register (RB0BSR - RB9BSR)

Offset

For a = 0 to 9:

Register	Offset
RBaBSR	8108h + (a × 200h)

Function

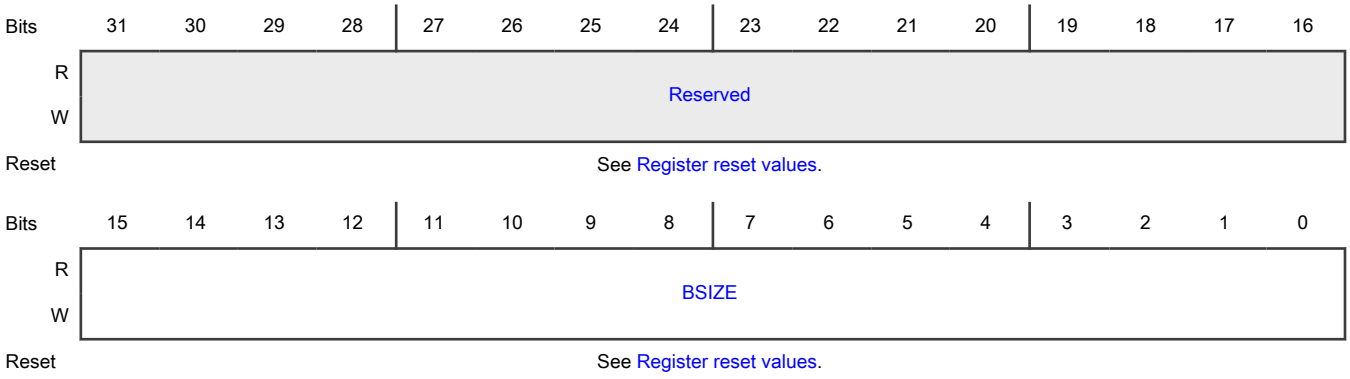
This is the buffer size register for the receive buffer descriptor ring. All buffers in the receive ring have the same size which allows the first buffer descriptor to indicate the total packet size.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0BSR–RB3BSR	RB4BSR–RB9BSR
ENETC1_SI0	RB0BSR–RB9BSR	—
ENETC1_SI1	RB0BSR–RB9BSR	—

Diagram



Register reset values

Register	Reset value
RB0BSR–RB3BSR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4BSR–RB9BSR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-16 —	Reserved
15-0 BSIZE	Buffer size Indicates the buffer size for the buffer pool used. 0x0000 64 KB 0x0080 128 bytes 0x0081 129 bytes ... 0x0100 256 bytes ... 0xFFFF 64 KB - 1 Minimum buffer size is 128 bytes. For better performance, it is recommended not to use a buffer size smaller than 256 bytes.

53.4.6.11.105 Rx BDR a consumer index register (RB0CIR - RB9CIR)

Offset

For a = 0 to 9:

Register	Offset
RBaCIR	810Ch + (a × 200h)

Function

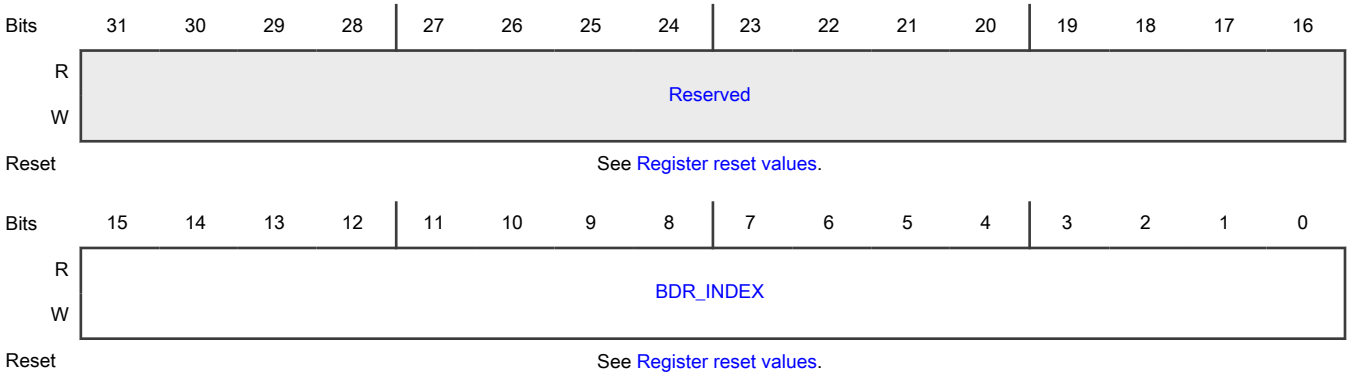
This is the receive ring consumer index register which indicates the next BD to be released to NETC for received frames. Software will initialize the buffer address in the BD and provide to hardware. Whenever the consumer index matches the value of the producer index, all BDs in the ring are considered initialized. Software writes this register during operation of the ring to indicate that any receive data occupied in the BD has been processed and it has been prepared for new data.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0CIR–RB3CIR	RB4CIR–RB9CIR
ENETC1_SI0	RB0CIR–RB9CIR	—
ENETC1_SI1	RB0CIR–RB9CIR	—

Diagram



Register reset values

Register	Reset value
RB0CIR–RB3CIR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4CIR–RB9CIR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-16 —	Reserved
15-0 BDR_INDEX	Receive buffer descriptor ring consumer index. Range of index depends on ring size RB*a*LENR[LENGTH].

53.4.6.11.106 Rx BDR a base address register 0 (RB0BAR0 - RB9BAR0)

Offset

For a = 0 to 9:

Register	Offset
RBaBAR0	8110h + (a × 200h)

Function

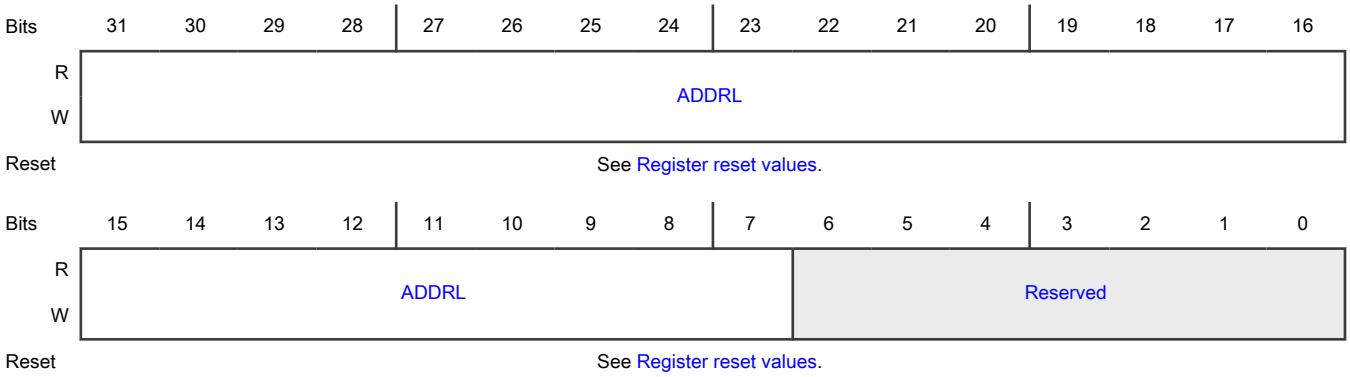
This is the lower address register for the base memory address location of the receive ring. The address must be 128 byte aligned.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0BAR0–RB3BAR0	RB4BAR0–RB9BAR0
ENETC1_SI0	RB0BAR0–RB9BAR0	—
ENETC1_SI1	RB0BAR0–RB9BAR0	—

Diagram



Register reset values

Register	Reset value
RB0BAR0–RB3BAR0	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4BAR0–RB9BAR0	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-7 ADDRL	Lower address bits 31-7.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-0 —	Reserved

53.4.6.11.107 Rx BDR a base address register 1 (RB0BAR1 - RB9BAR1)

Offset

For a = 0 to 9:

Register	Offset
RBaBAR1	8114h + (a × 200h)

Function

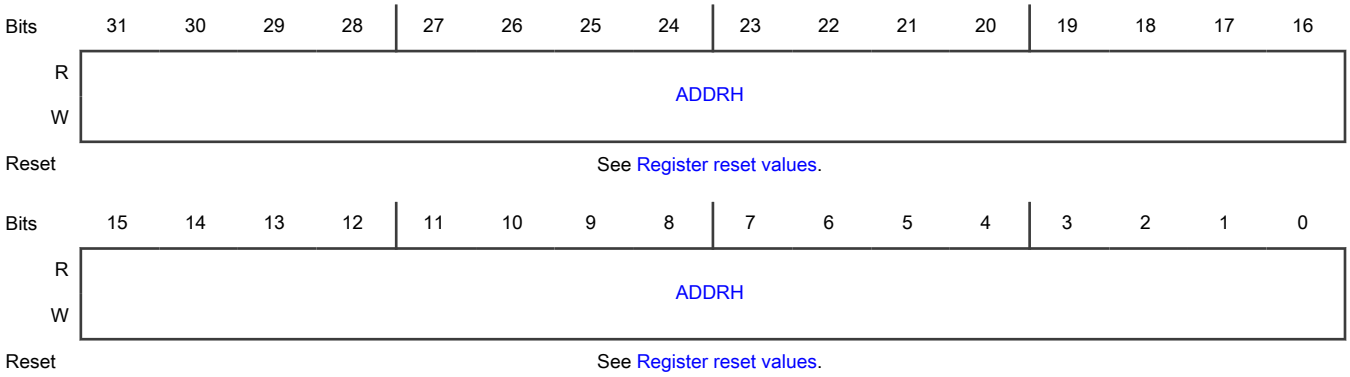
This is the upper address register for the base memory address location of the receive ring.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0BAR1–RB3BAR1	RB4BAR1–RB9BAR1
ENETC1_SI0	RB0BAR1–RB9BAR1	—
ENETC1_SI1	RB0BAR1–RB9BAR1	—

Diagram



Register reset values

Register	Reset value
RB0BAR1–RB3BAR1	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4BAR1–RB9BAR1	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-0 ADDRH	Upper address bits 63-32.

53.4.6.11.108 Rx BDR a producer index register (RB0PIR - RB9PIR)

Offset

For a = 0 to 9:

Register	Offset
RBaPIR	8118h + (a × 200h)

Function

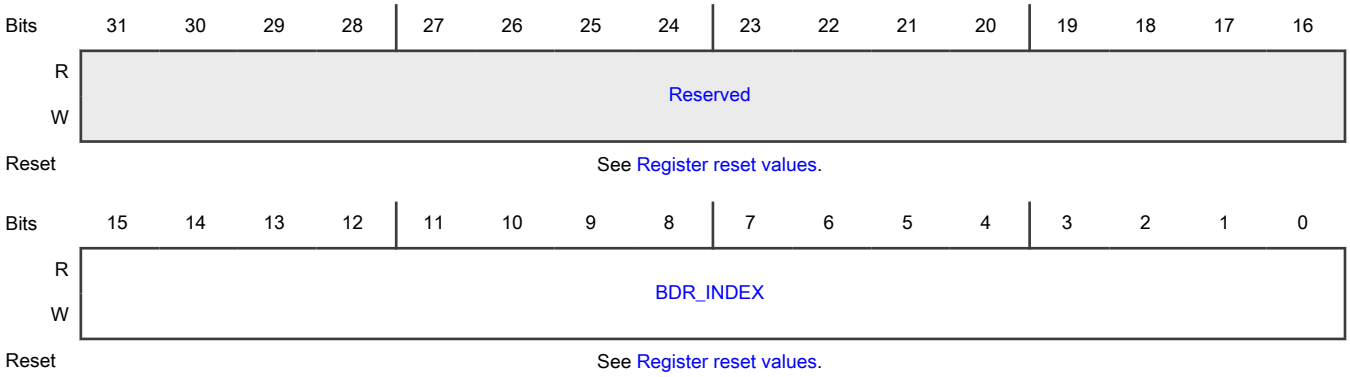
This is the producer index which indicates the BD used for the data of the next received frame. The producer index is initially configured by software but owned by hardware after the ring has been enabled. Hardware increments the index when a frame is received which may consume one or more BDs. Hardware is not allowed to increment the producer index to match the consumer index (RB*a*CIR) since ring can hold at most RB*a*LENR[LENGTH]*8-1 received BDs. Whenever the producer index matches the value of the consumer index, the ring has no unprocessed received frames and all BDs in the ring have been initialized/prepared by software, i.e. hardware owns all BDs in the ring.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0PIR–RB3PIR	RB4PIR–RB9PIR
ENETC1_SI0	RB0PIR–RB9PIR	—
ENETC1_SI1	RB0PIR–RB9PIR	—

Diagram



Register reset values

Register	Reset value
RB0PIR–RB3PIR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4PIR–RB9PIR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-16 —	Reserved
15-0 BDR_INDEX	Receive buffer descriptor ring producer index. Range of index depends on ring size RB*a*LENR[LENGTH].

53.4.6.11.109 Rx BDR a length register (RB0LENR - RB9LENR)

Offset

For a = 0 to 9:

Register	Offset
RBaLENR	8120h + (a × 200h)

Function

This is the receive BDR length register. It determines the size of the receive ring in sets of 8 BDs. The receive ring can only hold RB*a*LENR[LENGTH]*8-1 frames since no wrap bit is used to calculate full/empty conditions from consumer/producer indices.

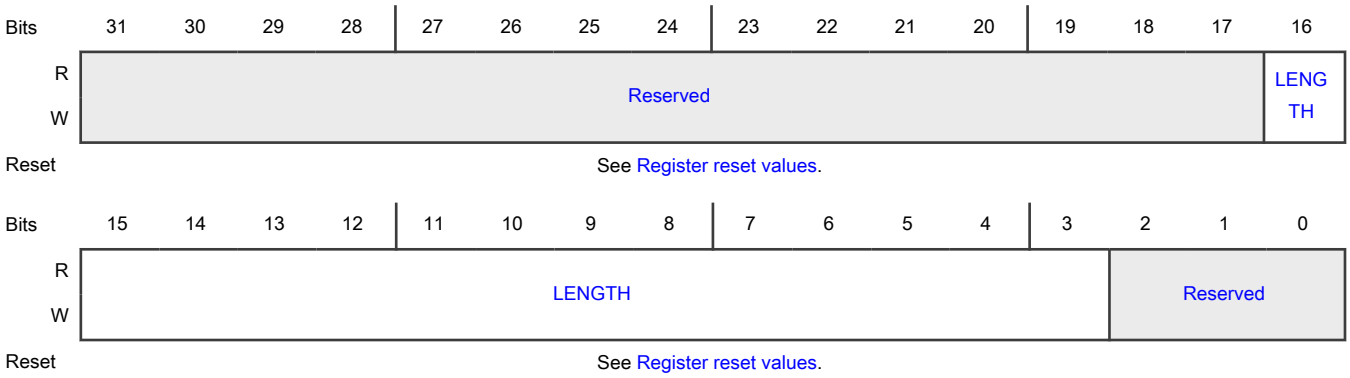
One entry is used to differentiate between full and empty ring. When producer index + 1 equals consumer index, the ring is considered full. When the producer/consumer indices match, the ring is always considered empty.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0LENR–RB3LENR	RB4LENR–RB9LENR
ENETC1_SI0	RB0LENR–RB9LENR	—
ENETC1_SI1	RB0LENR–RB9LENR	—

Diagram



Register reset values

Register	Reset value
RB0LENR–RB3LENR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4LENR–RB9LENR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-17 —	Reserved
16-3 LENGTH	BD ring length Size of ring in sets of 8 BDs. Maximum ring size is 64K. The size of the BD ring must be sufficiently large to hold a maximum sized Ethernet frame.
2-0 —	Reserved

53.4.6.11.110 Rx BDR a drop count register (RB0DCR - RB9DCR)

Offset

For a = 0 to 9:

Register	Offset
RBaDCR	8180h + (a × 200h)

Function

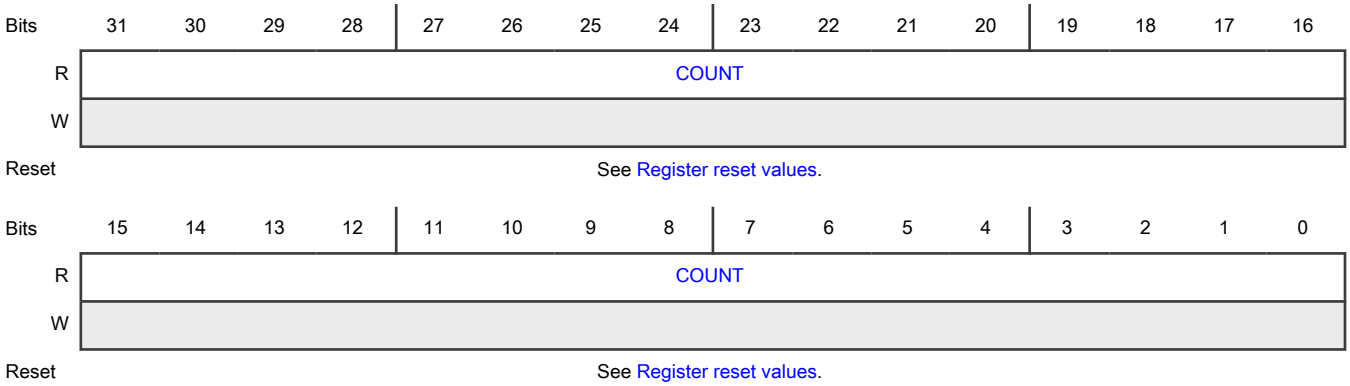
This is the receive BDR drop count register, it counts the number of frames dropped due to lack of receive buffer descriptors available.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0DCR–RB3DCR	RB4DCR–RB9DCR
ENETC1_SI0	RB0DCR–RB9DCR	—
ENETC1_SI1	RB0DCR–RB9DCR	—

Diagram



Register reset values

Register	Reset value
RB0DCR–RB3DCR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4DCR–RB9DCR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-0 COUNT	Count Number of frames dropped due to lack of receive buffer descriptors available.

53.4.6.11.111 Rx BDR a interrupt enable register (RB0IER - RB9IER)

Offset

For a = 0 to 9:

Register	Offset
RBaIER	81A0h + (a × 200h)

Function

This is the interrupt enable register for the receive buffer descriptor ring. If the interrupt is enabled while the corresponding detect bit in RB*a*IDR is set, it will trigger an interrupt for the SI based on the MSI-X vector defined in SIMSIRR*a*VR.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0IER–RB3IER	RB4IER–RB9IER
ENETC1_SI0	RB0IER–RB9IER	—
ENETC1_SI1	RB0IER–RB9IER	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															RXTIE
Reset	See Register reset values.															

Register reset values

Register	Reset value
RB0IER–RB3IER	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4IER–RB9IER	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-1 —	Reserved
0 RXTIE	Receive threshold interrupt enable 0 Disabled 1 Enabled

53.4.6.11.112 Rx BDR a interrupt detect register (RB0IDR - RB9IDR)

Offset

For a = 0 to 9:

Register	Offset
RBaIDR	81A4h + (a × 200h)

Function

This is the interrupt detect register for the receive buffer descriptor ring. Reading will automatically clear all events. Register SIRXIDR provides a non-destructive read access. It is not necessary to clear/set the RBaIER register upon entry/exit of the service routine. A new interrupt is generated in the event of a set/clear collision in this register.

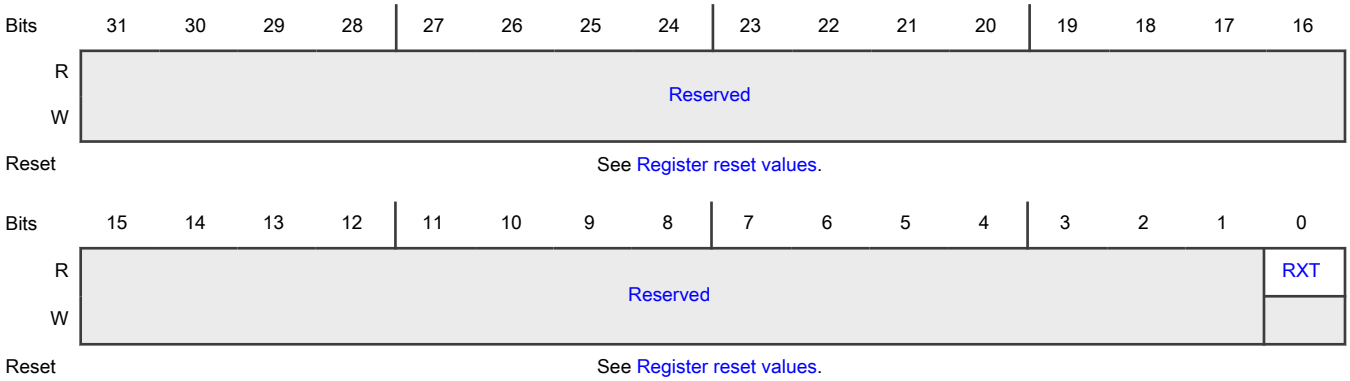
RBaIDR should not be polled if interrupt coalescing is enabled. Polling reads of RBaIDR clear the frame counter and reset the interrupt timer prematurely, preventing an interrupt from occurring.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0IDR–RB3IDR	RB4IDR–RB9IDR
ENETC1_SI0	RB0IDR–RB9IDR	—
ENETC1_SI1	RB0IDR–RB9IDR	—

Diagram



Register reset values

Register	Reset value
RB0IDR–RB3IDR	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4IDR–RB9IDR	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-1 —	Reserved
0 RXT	Receive threshold 0 No threshold event detected 1 Receive ring holds at least the number of packets specified by RB*a*ICR0[ICPT] or the threshold timer has expired since last received packet as specified by RB*a*ICR1[ICTT].

53.4.6.11.113 Rx BDR a interrupt coalescing register 0 (RB0ICR0 - RB9ICR0)

Offset

For a = 0 to 9:

Register	Offset
RBaICR0	81A8h + (a × 200h)

Function

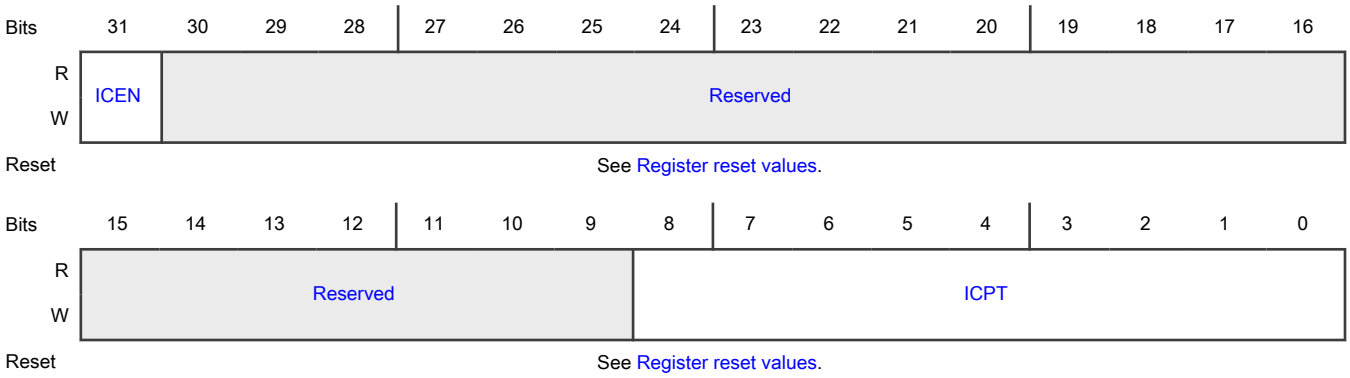
The receive ring interrupt coalescing register enables and configures the parameters for interrupt coalescing associated with received packets.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0ICR0–RB3ICR0	RB4ICR0–RB9ICR0
ENETC1_SI0	RB0ICR0–RB9ICR0	—
ENETC1_SI1	RB0ICR0–RB9ICR0	—

Diagram



Register reset values

Register	Reset value
RB0ICR0–RB3ICR0	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4ICR0–RB9ICR0	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31 ICEN	Interrupt coalescing enable 0 Interrupt coalescing is disabled 1 Interrupt coalescing is enabled. If the receive ring threshold interrupt is enabled (RB*a*IER[RXTIE] is set), an interrupt is raised when the threshold number of packets is reached (as defined by RB*a*ICR0[ICPT]) or when the threshold timer expires (as defined by RB*a*ICR1[ICTT]).
30-9 —	Reserved
8-0	Interrupt coalescing packet threshold

Table continues on the next page...

Table continued from the previous page...

Field	Function
ICPT	While interrupt coalescing is enabled, ICEN=1, this values determines the minimum number of packets received before raising an interrupt. The counter starts when ICEN is first set and one of the following events will restart the threshold counter: • Disabling receive ring (RB*a*MR[EN]=0) • Writing RB*a*ICR0 • Reading RB*a*IDR • Writing a 1 to SIRXIDR[RXT*a*] Settings: b0_0000_0000 Disabled b0_0000_0001 1 packet received b0_0000_0010 2 packets received b0_0000_0011 3 packets received ... b1_1111_1111 511 packets received **NOTE** If timer threshold is used it should be set when RB*a*ICR0[ICEN]=0 to avoid missing a "first packet" received condition which starts the timer. The threshold is based on received packets and does not include packets already present in the ring as a starting condition, therefore software should process all BDs in the ring and not just a number based on the threshold value.

53.4.6.11.114 Rx BDR a interrupt coalescing register 1 (RB0ICR1 - RB9ICR1)

Offset

For a = 0 to 9:

Register	Offset
RBaICR1	81ACh + (a × 200h)

Function

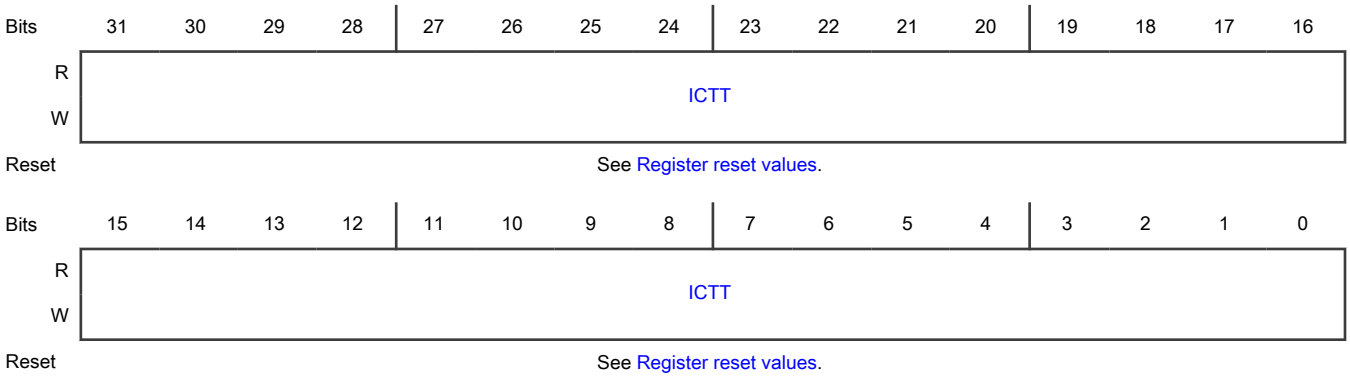
The receive ring interrupt coalescing register enables and configures the parameters for interrupt coalescing associated with received packets.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
ENETC0_SI0	RB0ICR1–RB3ICR1	RB4ICR1–RB9ICR1
ENETC1_SI0	RB0ICR1–RB9ICR1	—
ENETC1_SI1	RB0ICR1–RB9ICR1	—

Diagram



Register reset values

Register	Reset value
RB0ICR1–RB3ICR1	ENETC0_SI0–ENETC1_SI1: 0000_0000h
RB4ICR1–RB9ICR1	ENETC0_SI0: Register not supported ENETC1_SI0,ENETC1_SI1: 0000_0000h

Fields

Field	Function
31-0 ICTT	Interrupt coalescing timer threshold ,specified in units of NETC clock cycles. When interrupt coalescing is enabled, RB*a*ICR0[ICEN]=1, this value determines the maximum amount of time allowed between a first received packet until RB*a*ICR0[ICPT] threshold is reached. If the timer expires before threshold reached, the interrupt event RB*a*IDR[RXT] is asserted. The timer operates on the NETC platform clock. A register value of zero indicates the timer is disabled. One of the following conditions will reset the timer: • Disabling receive ring (RB*a*MR[EN]=0) • Reading RB*a*IDR • Writing a 1 to SIRXIDR[RXT*a*]

Table continues on the next page...

Field	Function
	NOTE If timer threshold is used it should be set when RB*a*ICR0[ICEN]=0 to avoid missing a "first packet" received condition which starts the timer.

53.4.6.12 NETC EMDIO base function register descriptions

This section describes the NETC device registers for the EMDIO base PCIe function.

The base address listed is relative to the PCIe function's base address register value can be discovered from reading the PCIe Enhanced Allocation Base Address Register Equivalent Index 0 (EA BEI 0).

53.4.6.12.1 Port memory map

EMDIO_BASE pcie ea bei 0 offset: 60BA_0000h

Offset	Register	Width (In bits)	Access	Reset value
1C00h	External MDIO configuration register (EMDIO_CFG)	32	RW	0005_1448h
1C04h	External MDIO interface control register (EMDIO_CTL)	32	RW	0000_0000h
1C08h	External MDIO interface data register (EMDIO_DATA)	32	RW	0000_0000h
1C0Ch	External MDIO register address register (EMDIO_ADDR)	32	RW	0000_0000h
1C10h	External MDIO status register (EMDIO_STAT)	32	R	0000_0000h
1C20h	PHY status configuration register (PHY_STATUS_CFG)	32	RW	0000_0000h
1C24h	PHY status control register (PHY_STATUS_CTL)	32	RW	0000_0000h
1C28h	PHY status data register (PHY_STATUS_DATA)	32	R	0000_0000h
1C2Ch	PHY status register address register (PHY_STATUS_ADDR)	32	RW	0000_0000h
1C30h	PHY status event register (PHY_STATUS_EVENT)	32	RW	0000_0000h
1C34h	PHY status mask register (PHY_STATUS_MASK)	32	RW	0000_0000h
1C40h	MDIO configuration register (MDIO_CFG)	32	R	0000_0010h

53.4.6.12.2 External MDIO configuration register (EMDIO_CFG)

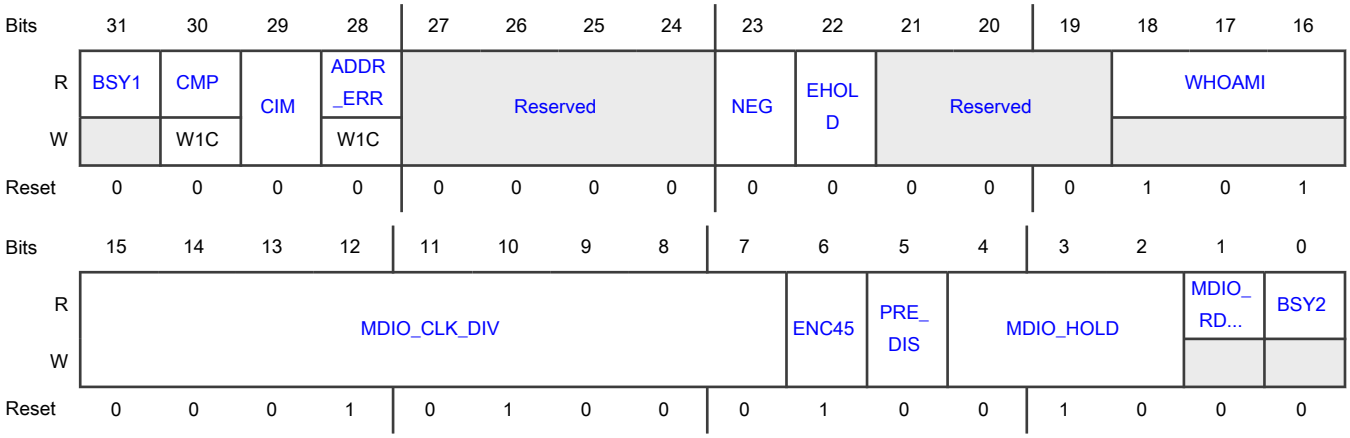
Offset

Register	Offset
EMDIO_CFG	1C00h

Function

External MDIO configuration register for access to the off-chip PHY.

Diagram



Fields

Field	Function
31 BSY1	Busy 1 Indicates whether MDIO is busy. 0b - An MDIO transaction is not occurring; software may access other MDIO registers. 1b - An MDIO transaction is occurring.
30 CMP	MDIO Command Completion Bit is cleared by writing 1b. 0b - An MDIO command completion did not occur. 1b - An MDIO command completion occurred.
29 CIM	MDIO Command Completion Interrupt Enable Mask 0b - Masked 1b - Enabled
28 ADDR_ERR	Address Error 0b - Normal 1b - Error. An access control violation has occurred. The request address used does not match the MDIO PHY's address (clause 22) or MDIO port address (clause 45) assigned.
27-24 —	Reserved
23 NEG	Negative Edge 0b - Normal operation - positive edge 1b - MDIO is driven by master on MDC negative edge (default for external MDIOs)
22	Extended HOLD

Table continues on the next page...

Table continued from the previous page...

Field	Function
EHOLD	**NOTE** For extended operation, MDIO hold time for MDIO_HOLD is specified as follows: 000b - 1 NETC cycle 001b - 9 NETC cycles 010b - 17 NETC cycles 011b - 25 NETC cycles 100b - 33 NETC cycles 101b - 41 NETC cycles 110b - 49 NETC cycles 111b - 57 NETC cycles 0b - Normal operation. MDIO hold time is specified in . 1b - Extended Operation
21-19 —	Reserved
18-16 WHOAMI	Returns the virtual port ID.
15-7 MDIO_CLK_DIV	MDIO Clock Divisor A value of 5-511 can be set to configure the MDIO clock frequency relative to NETC clock. Ratio is (2×MDIO_CLK_DIV)+1. Setting the divisor to 0 disables MDC. The minimum value that should be written to this bit field is 2, which corresponds to a clock divide ratio of 5. **NOTE** The default is 40. For external MDIO Ethernet management interface, the default is 0.
6 ENC45	Enable Clause 45 Support 0b - Clause 22 transactions are used. 1b - Clause 45 transactions are used.
5 PRE_DIS	MDIO Preamble Disable 0b - Generation of MDIO preamble is enabled (default operation). 1b - Generation of MDIO preamble is disabled
4-2 MDIO_HOLD	MDIO Hold Time 000b - 1 NETC cycle 001b - 3 NETC cycles 010b - 5 NETC cycles (default - recommended value)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - 7 NETC cycles 100b - 9 NETC cycles 101b - 11 NETC cycles 110b - 13 NETC cycles 111b - 15 NETC cycles
1 MDIO_RD_ER	MDIO Read Error 0b - No error 1b - The last read transaction received no response from a PHY; any data read should be considered invalid (for example, the PHY address does not match any PHY available on the MDIO bus).
0 BSY2	Busy 2 (same as bit 31) Indicates whether MDIO is busy. 0b - An MDIO transaction is not occurring; software may access other MDIO registers. 1b - An MDIO transaction is occurring.

53.4.6.12.3 External MDIO interface control register (EMDIO_CTL)

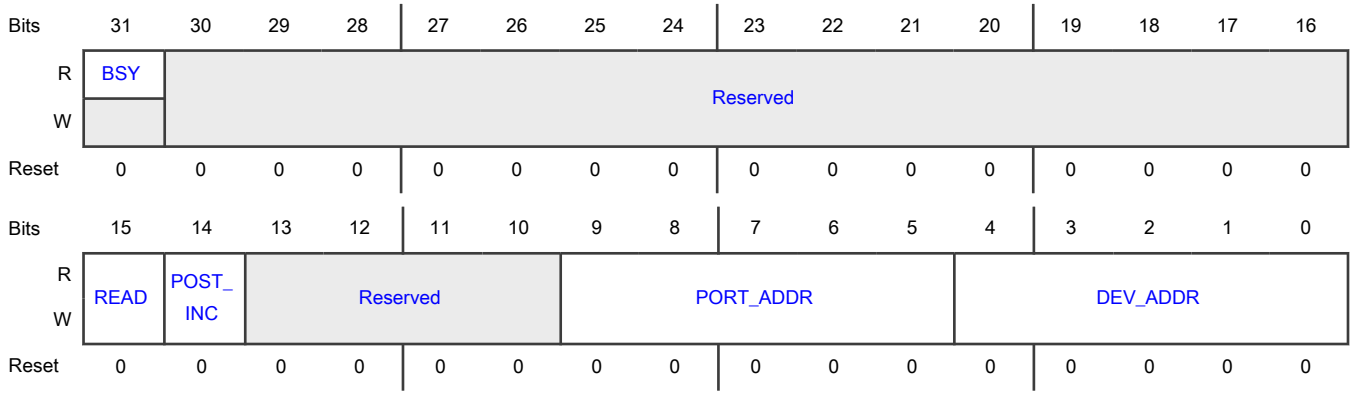
Offset

Register	Offset
EMDIO_CTL	1C04h

Function

This is the external MDIO interface control register.

Diagram



Fields

Field	Function
31 BSY	MDIO busy 0 An MDIO transaction is not occurring; software may access other MDIO registers. 1 An MDIO transaction is occurring.
30-16 —	Reserved
15 READ	MDIO read initiation. 1 A normal MDIO read transaction is initiated. Self-clearing once transaction is complete.
14 POST_INC	MDIO read with address post-increment initiation. Self-clearing once transaction is complete. 0 (default operation) 1 An MDIO read with Clause 45 address post-increment transaction is initiated (Post-incrementation is performed in the PHY internal address register.)
13-10 —	Reserved
9-5 PORT_ADDR	5-bit MDIO port address (Clause 45) / PHY address (Clause 22)
4-0 DEV_ADDR	5-bit MDIO device address (Clause 45) / register address (Clause 22)

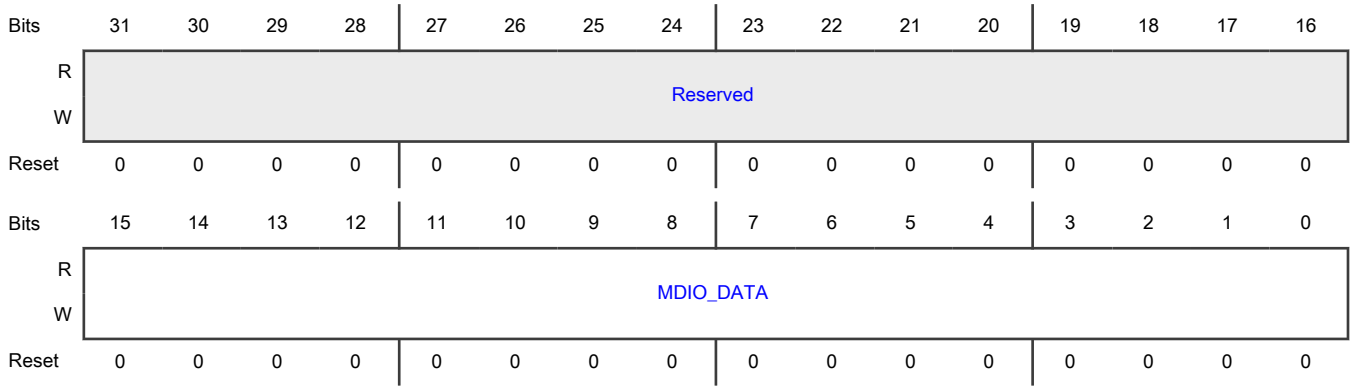
53.4.6.12.4 External MDIO interface data register (EMDIO_DATA)

Offset

Register	Offset
EMDIO_DATA	1C08h

Function
This is the external MDIO interface data register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 MDIO_DATA	16-bit MDIO data. Writing to this register initiates a write transaction to the PHY. For proper operation, the EMDIO_CTL register must have been initialized first. The EMDIO_CFG[BUSY] status bit is set immediately and cleared when the write transaction has completed. Reading this register returns data read from the PHY register specified in EMDIO_CTL after a read transaction has completed. Read transactions are initiated by writing a 1 to EMDIO_CTL[READ] or EMDIO_CTL[POST_INC]. Read data is invalid if EMDIO_CFG[BUSY] is set.

53.4.6.12.5 External MDIO register address register (EMDIO_ADDR)

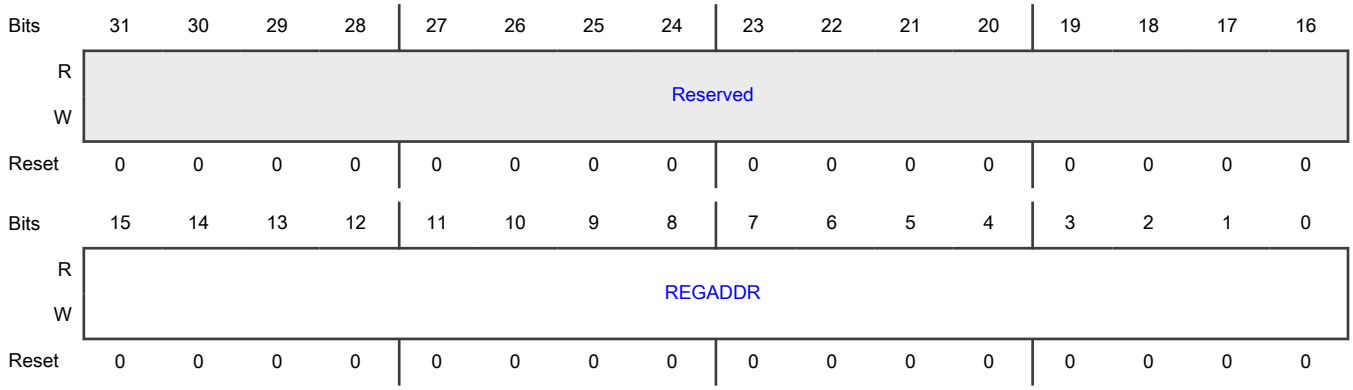
Offset

Register	Offset
EMDIO_ADDR	1C0Ch

Function

This is the external MDIO register address register, and is used to communicate with an attached Clause 45 PHY.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 REGADDR	MDIO PHY register address. Address of the register within the Clause 45 PHY device from which data is to be read or to which data is to be written. Writing to this register initiates an address-write transaction to set the PHY's internal address register to the value specified in EMDIO_ADDR[REGADDR]. For proper operation, the EMDIO_CTL register must have been initialized prior to writing to this register.

53.4.6.12.6 External MDIO status register (EMDIO_STAT)

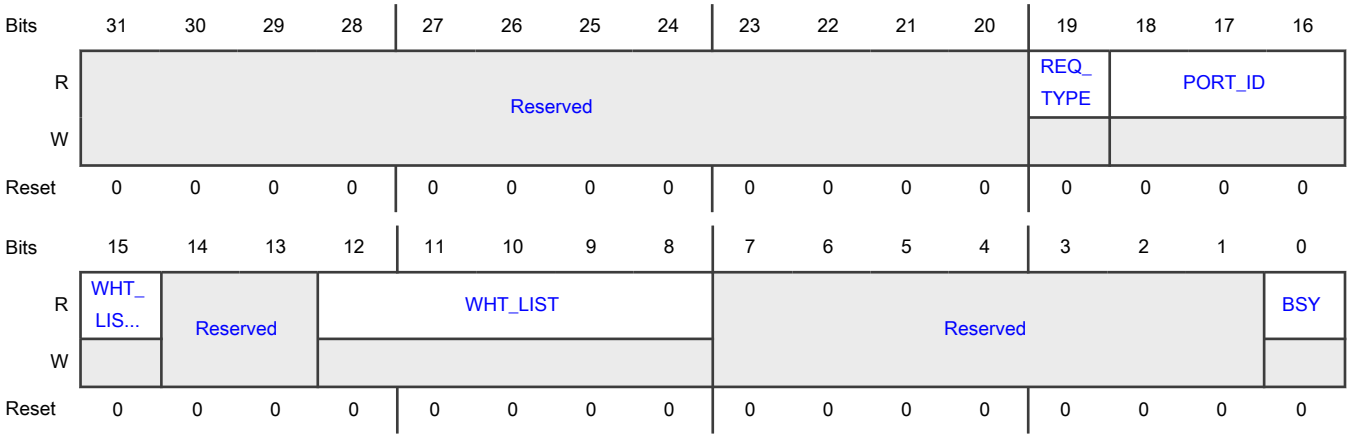
Offset

Register	Offset
EMDIO_STAT	1C10h

Function

This is the external MDIO status register.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 REQ_TYPE	Port ID Returns the request type of the active virtual port. 0 MDIO request 1 PHY polling
18-16 PORT_ID	Port ID Returns the port ID of the active virtual port. Matches the value EMDIO_CFG[WHOAMI] of active port.
15 WHT_LIST_ENA	PHY white list enable Returns the PHY white list enable of the active virtual port.
14-13 —	Reserved
12-8 WHT_LIST	PHY white list Returns the PHY white list address of the active virtual port.
7-1 —	Reserved
0 BSY	Global MDIO busy 0 An MDIO transaction is not occurring; software may access other MDIO registers. 1 An MDIO transaction is occurring.

53.4.6.12.7 PHY status configuration register (PHY_STATUS_CFG)

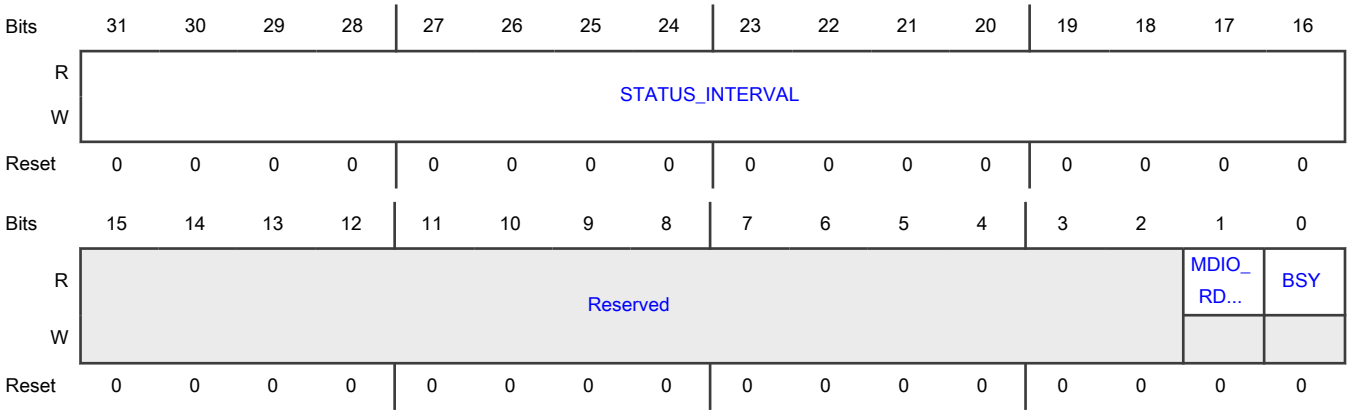
Offset

Register	Offset
PHY_STATUS_CFG	1C20h

Function

This is the PHY status configuration register.

Diagram



Fields

Field	Function
31-16 STATUS_INTERVAL	PHY status read interval Amount of time to wait between PHY status reads, in units of 1-2 ms. A value of 0 indicates auto PHY automatic status reads are disabled.
15-2 —	Reserved
1 MDIO_RD_ER	MDIO read error 1 The last read transaction received no response from a PHY; any data read should be considered invalid. (for example, The PHY address does not match any PHY available on the MDIO bus.)
0 BSY	MDIO busy 0 An MDIO transaction is not occurring; software may access other MDIO registers. 1 An MDIO transaction is occurring.

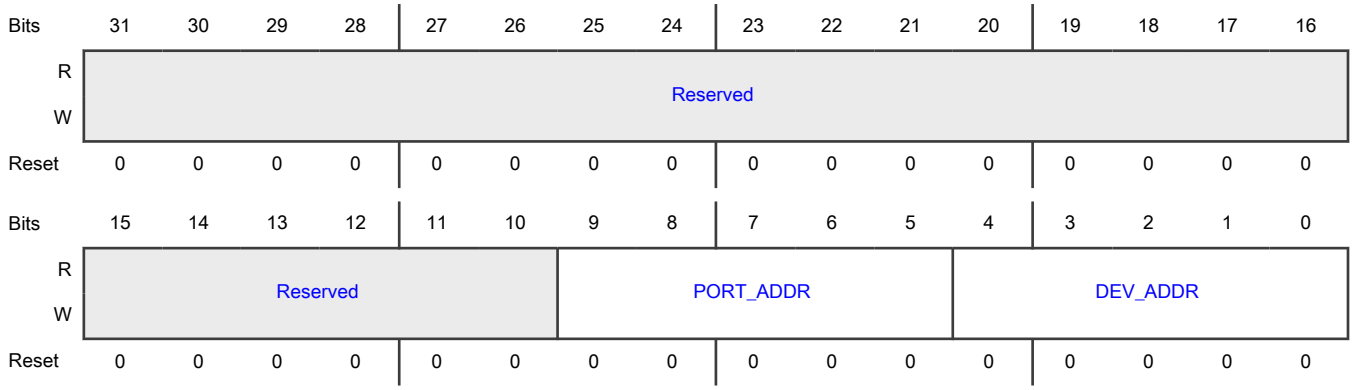
53.4.6.12.8 PHY status control register (PHY_STATUS_CTL)

Offset

Register	Offset
PHY_STATUS_CTL	1C24h

Function
This is the PHY status control register.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-5 PORT_ADDR	5-bit MDIO port address (Clause 45) / PHY address (Clause 22)
4-0 DEV_ADDR	5-bit MDIO device address (Clause 45) / register address (Clause 22)

53.4.6.12.9 PHY status data register (PHY_STATUS_DATA)

Offset

Register	Offset
PHY_STATUS_DATA	1C28h

Function
This is the PHY status data register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CURR_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MDIO_DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 CURR_CNT	Current count Current value of status interval counter.
15-0 MDIO_DATA	16-bit MDIO data Reading this register returns data read from the PHY register specified in PHY_STATUS_CTL after a read transaction has completed. Read transactions are initiated automatically based on the ms counter setting. Read data is invalid if PHY_STATUS_CFG[BUSY] is set.

53.4.6.12.10 PHY status register address register (PHY_STATUS_ADDR)

Offset

Register	Offset
PHY_STATUS_ADDR	1C2Ch

Function

This is the PHY status address register, and is used to communicate with an attached Clause 45 PHY.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REGADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 REGADDR	MDIO PHY register address. Address of the register within the Clause 45 PHY device from which data is to be read.

53.4.6.12.11 PHY status event register (PHY_STATUS_EVENT)

Offset

Register	Offset
PHY_STATUS_EVENT	1C30h

Function

This is the PHY status event register, and is used to detect transitions in the data bits. Used in conjunction with the mask register to signal interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATUS_EVENT_LH															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STATUS_EVENT_HL															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 STATUS_EVENT_LH	Status event low-to-high. Set to 1 if a 0->1 transition on a corresponding data bit has occurred. Write 1 to clear.
15-0 STATUS_EVENT_HL	Status event high-to-low. Set to 1 if a 1->0 transition on a corresponding data bit has occurred. Write 1 to clear.

53.4.6.12.12 PHY status mask register (PHY_STATUS_MASK)

Offset

Register	Offset
PHY_STATUS_MASK	1C34h

Function

This is the PHY status mask register, and is used in conjunction with the event register to signal interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STATUS_MASK_LH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	STATUS_MASK_HL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 STATUS_MASK_LH	Status mask low-to-high. If set to 1, assert an interrupt if the corresponding event bit is set.
15-0 STATUS_MASK_HL	Status high-to-low mask. If set to 1, assert an interrupt if the corresponding event bit is set.

53.4.6.12.13 MDIO configuration register (MDIO_CFG)

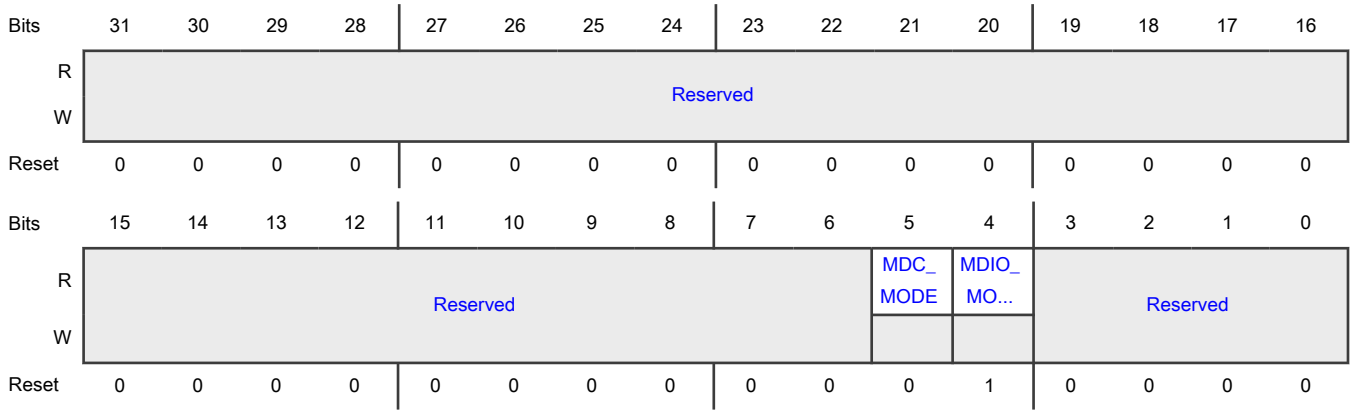
Offset

Register	Offset
MDIO_CFG	1C40h

Function

This is the MDIO configuration register.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 MDC_MODE	MDC pin mode 0 Push-pull 1 Open drain This field is writable in IERB (by pre-boot initialization), which is then immediately reflected here. It cannot be written by operational software. The hardware uses this setting for operational purposes.
4 MDIO_MODE	MDIO pin mode 0 Push-pull 1 Open drain This field is writable in IERB (by pre-boot initialization), which is then immediately reflected here. It cannot be written by operational software. The hardware uses this setting for operational purposes.
3-0 —	Reserved

53.4.6.13 Timer register descriptions

This section describes the Timer registers to support the IEEE 1588 embedded PCIe function.

The base address listed is relative to the PCIe function's base address register value discovered from reading the PCIe Enhanced Allocation Base Address Register Equivalent Index 0 (EA BEI 0).

NOTE

Timer does not return an error on invalid register accesses, but returns 0's on reads in invalid registers, and silently drops writes to invalid or read-only registers.

53.4.6.13.1 Port memory map

TMR0_BASE pcie ea bei 0 offset: 60B8_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module ID Register (TMR_ID)	32	R	0300_0201h
8h	Timer Capability Register (TMR_CAPR)	32	R	0000_0006h
20h	Timer free running time low register (TMR_FRT_L)	32	R	0000_0000h
24h	Timer free running time high register (TMR_FRT_H)	32	R	0000_0000h
28h	Timer synchronous time low register (TMR_SRT_L)	32	R	0000_0000h
2Ch	Timer synchronous time high register. (TMR_SRT_H)	32	R	0000_0000h
30h	Default ns timer counter low register (TMR_DEF_CNT_L)	32	R	0000_0000h
34h	Default ns timer counter high register (TMR_DEF_CNT_H)	32	R	0000_0000h
80h	Timer Control Register (TMR_CTRL)	32	RW	0001_0001h
84h	Timer Event Register (TMR_TEVENT)	32	RW	0000_0000h
88h	Timer event mask register (TMR_TEMASK)	32	RW	0000_0000h
94h	Timer status register (TMR_STAT)	32	R	8000_0000h
98h	Timer counter low register (TMR_CNT_L)	32	RW	0000_0000h
9Ch	Timer counter high register (TMR_CNT_H)	32	RW	0000_0000h
A0h	Timer addend register (TMR_ADD)	32	RW	0000_0000h
A4h	Timer accumulator register (TMR_ACC)	32	R	0000_0000h
A8h	Timer prescale register (TMR_PRSC)	32	RW	0000_0002h
ACh	Extended timer control register (TMR_ECTRL)	32	RW	0000_0000h
B0h	Timer offset low register (TMROFF_L)	32	RW	0000_0000h
B4h	Timer offset high register (TMROFF_H)	32	RW	0000_0000h
B8h	Alarm 1 time comparator low register (TMR_ALARM1_L)	32	RW	FFFF_FFFFh
BCh	Alarm 1 time comparator high register (TMR_ALARM1_H)	32	RW	FFFF_FFFFh
C0h	Alarm 2 time comparator low register (TMR_ALARM2_L)	32	RW	FFFF_FFFFh
C4h	Alarm 2 time comparator high register (TMR_ALARM2_H)	32	RW	FFFF_FFFFh
CCh	Timer Alarm Control Register (TMR_ALARM_CTRL)	32	RW	0000_0000h
D0h - D8h	Timer a fixed interval period register (TMR_FIPER1 - TMR_FIPER3)	32	RW	FFFF_FFFFh
DCh	Timer FIPER Control Register (TMR_FIPER_CTRL)	32	RW	0041_4141h
E0h	External trigger stamp register (TMR_ETTS1_L)	32	R	0000_0000h
E4h	External trigger stamp register (TMR_ETTS1_H)	32	R	0000_0000h
E8h	External trigger stamp register (TMR_ETTS2_L)	32	R	0000_0000h
ECh	External trigger stamp register (TMR_ETTS2_H)	32	R	0000_0000h
F0h	Timer current time low register (TMR_CUR_TIME_L)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
F4h	Timer current time high register (TMR_CUR_TIME_H)	32	R	0000_0000h
F8h	Timer parameter register (TMR_PARAM)	32	RW	0000_0000h

53.4.6.13.2 Module ID Register (TMR_ID)

Offset

Register	Offset
TMR_ID	0h

Function

The module ID register (TMR_ID) is a read-only register. The TMR_ID register is used to identify the 1588 timer IP block and revision. Currently the 1588 timer ID is 0x0300 and the major revision is 0x01, while the minor revision is 0x00.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMR_ID															
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REV_MJ								REV_MN							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-16 TMR_ID	Timer IP ID Value identifying the 1588 timer module. 0x0300 Unique identifier for the 1588 timer module
15-8 REV_MJ	Major revision Value identifies the major revision of the 1588 timer module. 02 is the initial version

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 REV_MN	Minor revision Value identifies the minor revision of the 1588 timer module. 00 is the initial version

53.4.6.13.3 Timer Capability Register (TMR_CAPR)

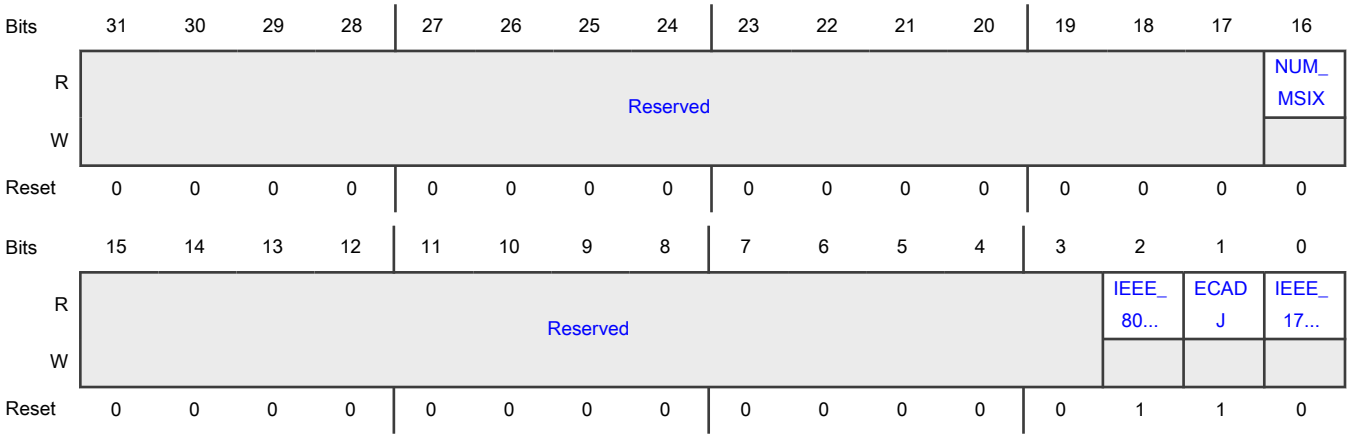
Offset

Register	Offset
TMR_CAPR	8h

Function

This is the timer capability register which reflects additional hardware capability.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 NUM_MSIX	Number of MSI-x Vectors supported. Value mirrored from IERB T *a* BCR[NUM_MSIX]. Formula: NUM_MSIX + 1
15-3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 IEEE_8021AS_REV	Support 802.1AS-2020 by providing both a ns and free-running clock to the Ethernet MACs.
1 ECADJ	Enhanced 1588 nanosecond (ns) Timer Adjustment supported
0 IEEE_1722	IEEE 1722 support 0 Not supported 1 Supported

53.4.6.13.4 Timer free running time low register (TMR_FRT_L)

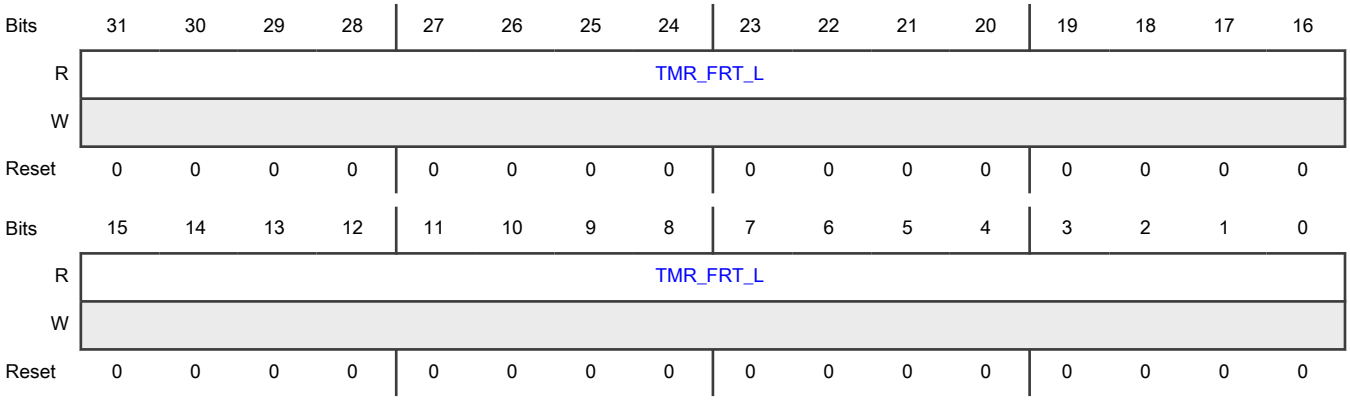
Offset

Register	Offset
TMR_FRT_L	20h

Function

This is the Timer free running time low register. Count is in 1588 clock ticks.

Diagram



Fields

Field	Function
31-0 TMR_FRT_L	Read-only copy of the free running time (lower 32b)

53.4.6.13.5 Timer free running time high register (TMR_FRT_H)

Offset

Register	Offset
TMR_FRT_H	24h

Function

This is the Timer free running time high register. Count is in 1588 clock ticks.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMR_FRT_H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMR_FRT_H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMR_FRT_H	Read-only copy of the free running time (upper 32b)

53.4.6.13.6 Timer synchronous time low register (TMR_SRT_L)

Offset

Register	Offset
TMR_SRT_L	28h

Function

This is the Timer synchronous time low register, that is a synchronized version of TMR_CNT_L.Count is in ns. A read to TMR_FRT_L captures all 64b of FRT_H/L and 64b of SRT_H/L, for an atomic read of all 4 registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMR_SRT_L															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMR_SRT_L															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMR_SRT_L	Read-only copy of the synchronous time (lower 32b)

53.4.6.13.7 Timer synchronous time high register. (TMR_SRT_H)

Offset

Register	Offset
TMR_SRT_H	2Ch

Function

This is the Timer synchronous time high register, that is a synchronized version of TMR_CNT_H. Count is in ns. A read to TMR_FRT_L captures all 64b of FRT_H/L and 64b of SRT_H/L, for an atomic read of all 4 registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMR_SRT_H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMR_SRT_H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMR_SRT_H	Read-only copy of the synchronous time (upper 32b)

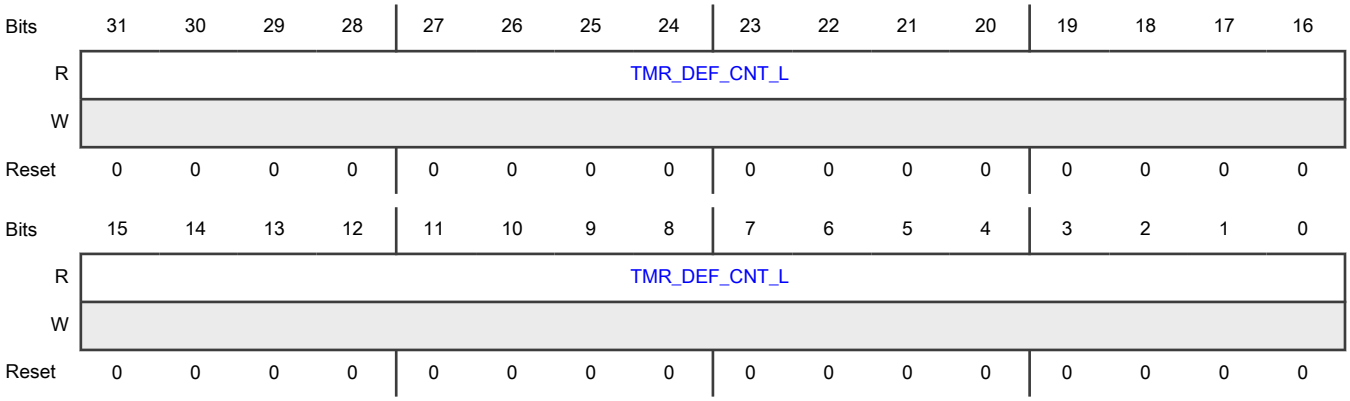
53.4.6.13.8 Default ns timer counter low register (TMR_DEF_CNT_L)

Offset

Register	Offset
TMR_DEF_CNT_L	30h

Function
This is the default ns counter low register.

Diagram



Fields

Field	Function
31-0 TMR_DEF_CNT_L	Read-only copy of the Default ns time counter (lower 32b). It is enabled when TMR_CTRL[TE] is not enabled and Default counter enable from the ierb register is asserted

53.4.6.13.9 Default ns timer counter high register (TMR_DEF_CNT_H)

Offset

Register	Offset
TMR_DEF_CNT_H	34h

Function

This is the default ns counter high register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMR_DEF_CNT_H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMR_DEF_CNT_H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMR_DEF_CNT_H	Read-only copy of the Default ns time counter (upper 32b).It is enabled when TMR_CTRL[TE] is not enabled and Default counter enable from the ierb register is asserted

53.4.6.13.10 Timer Control Register (TMR_CTRL)

Offset

Register	Offset
TMR_CTRL	80h

Function

This register is used to configure, and initialize the 1588 IP precision timer clock.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ALM1P	ALM2P	SHADOW_...	FS	PP1L	PP2L	TCLK_PERIOD									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COMP_MO...	Reserved	Reserved				ETEP2	ETEP1	COPH	CIPH	Reserved		Reserved	TE	CK_SEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31 ALM1P	Alarm1 output polarity 0 active high output 1 active low output
30 ALM2P	Alarm2 output polarity 0 active high output 1 active low output
29 SHADOW_DIS	shadow Register disable If SHADOW_DIS is high,reads to TMR_CNT_H read the upper 32 bits of the counter directly instead of the shadow copy
28 FS	FIPER start indication 0 Fiper is enabled through timer enable 1 Fiper is enabled through timer enable and alarm getting set. See FIPER_CTRL and ALARM_CTRL for more details.
27 PP1L	Fiper1 pulse loop back mode enabled 0 Trigger1 input is based upon normal external trigger input 1 Fiper1 pulse is looped back into Trigger1 input
26 PP2L	Fiper2 pulse loop back mode enabled 0 Trigger2 input is based upon normal external trigger input 1 Fiper2 pulse is looped back into Trigger2 input
25-16 TCLK_PERIOD	1588 timer reference clock period The value programmed by user in this field ideally represents the frequency of the phase adjusted clock period. The timer clock counter will increment by TCLK_PERIOD every clock cycle. When this field is being used to track time, this clock period must be equal to the clock period of the timer reference clock. For applications where user does not want the clock period to be added, they can program this field to 1 to count the clock ticks. Note: For customers sensitive to consistency in time-stamp values, it is recommended the selected frequency is fixed at a period that is an integer multiple value.
15 COMP_MODE	Mode bit to allow atomic writes to TCLK_PERIOD and TMR_ADD If drift compensation requires updates to TCLK_PERIOD and TMR_ADD, the resulting time will be inconsistent for the time it takes to do the two register writes. Setting this Compensation Mode bit makes the writes atomic. An update to TCLK_PERIOD doesn't take effect until TMR_ADD is written. That would allow writes to TMR_ADD alone, or TMR_CTRL[TCLK_PERIOD]+TMR_ADD. 0 Atomic updates to TMR_PERIOD and TMR_ADD disabled 1 Allow atomic updates to TMR_PERIOD and TMR_ADD

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 —	Reserved
13-10 —	Reserved
9 ETEP2	External trigger 2 edge polarity 0 Time stamp on the rising edge of the external trigger 1 Time stamp on the falling edge of the external trigger
8 ETEP1	External trigger 1 edge polarity 0 Time stamp on the rising edge of the external trigger 1 Time stamp on the falling edge of the external trigger
7 COPH	Generated clock (TMR_GCLK) output phase. 0 non-inverted divided clock is output 1 inverted divided clock is output
6 CIPH	External oscillator input clock phase 0 non-inverted frequency tuned timer input clock 1 inverted frequency tuned timer input clock
5-4 —	Reserved
3 —	Reserved
2 TE	1588 timer enable If not enabled, see default counter for ns time. 0 timer not enabled. Use default counter for ns time. 1 timer enabled.
1-0 CK_SEL	1588 timer reference clock source select Default is b01. The results of timer clock domain register read/write are bounded undefined if an inactive 1588 reference clock is selected. 00 External high precision timer reference clock (TMR_1588_CLK). 01 NETC system clock 10 reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11 reserved

53.4.6.13.11 Timer Event Register (TMR_TEVENT)

Offset

Register	Offset
TMR_TEVENT	84h

Function

The IEEE 1588 timer implementation requires several interrupt event signals. Software may poll this register at any time to check for pending interrupts. If an event occurs and its corresponding enable bit is set in the event mask register (TEMASK), the event also causes a hardware interrupt. A bit in the timer event register is cleared by writing a 1 to that bit position.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		ETS2_OV...	ETS1_OV...	Reserved		ETS2EN	ETS1EN	Reserved		ETS2_TH...	ETS1_TH...	Reserved		ALM2EN	ALM1EN
W			W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								PP1EN	PP2EN	PP3EN	Reserved				
W									W1C	W1C	W1C					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 ETS2_OVEN	External trigger 2 timestamp FIFO Overflow event occurred
28 ETS1_OVEN	External trigger 1 timestamp FIFO Overflow event occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-26 —	Reserved
25 ETS2EN	External trigger 2 new timestamp sample event available
24 ETS1EN	External trigger 1 new timestamp sample event available
23-22 —	Reserved
21 ETS2_THREN	External trigger 2 timestamp FIFO Threshold Level Hit
20 ETS1_THREN	External trigger 1 timestamp FIFO Threshold Level Hit
19-18 —	Reserved
17 ALM2EN	Timer ALM1 event enable
16 ALM1EN	Timer ALM2 event enable
15-8 —	Reserved
7 PP1EN	Periodic pulse event 1 enable
6 PP2EN	Periodic pulse event 2 enable
5 PP3EN	Periodic pulse event 3 enable
4-0 —	Reserved

53.4.6.13.12 Timer event mask register (TMR_TEMASK)

Offset

Register	Offset
TMR_TEMASK	88h

Function

Timer event mask register. The event mask register provides control over which possible interrupt events in the TMR_TEVENT register are permitted to participate in generating hardware interrupts to the PIC. All implemented bits in this register are R/W and cleared upon a hardware reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		ETS2_OV...	ETS1_OV...	Reserved		ETS2EN	ETS1EN	Reserved		ETS2_TH...	ETS1_TH...	Reserved		ALM2EN	ALM1EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								PP1EN	PP2EN	PP3EN	Reserved				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29 ETS2_OVEN	External trigger 2 timestamp FIFO Overflow event interrupt enabled
28 ETS1_OVEN	External trigger 1 timestamp FIFO Overflow event interrupt enabled
27-26 —	Reserved
25 ETS2EN	External trigger 2 new timestamp sample event available interrupt enable
24 ETS1EN	External trigger 1 new timestamp sample event available interrupt enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-22 —	Reserved
21 ETS2_THREN	External trigger 2 timestamp FIFO Threshold Level Hit event enable
20 ETS1_THREN	External trigger 1 timestamp FIFO Threshold Level Hit event enable
19-18 —	Reserved
17 ALM2EN	Timer ALM1 event enable
16 ALM1EN	Timer ALM2 event enable
15-8 —	Reserved
7 PP1EN	Periodic pulse event 1 enable
6 PP2EN	Periodic pulse event 2 enable
5 PP3EN	Periodic pulse event 3 enable
4-0 —	Reserved

53.4.6.13.13 Timer status register (TMR_STAT)

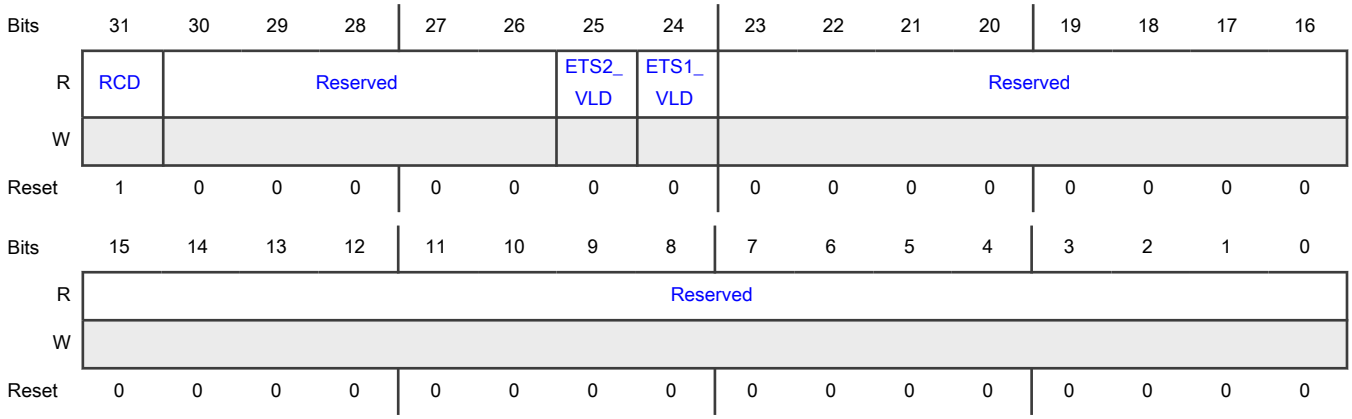
Offset

Register	Offset
TMR_STAT	94h

Function

The timer TMR_STAT register returns relevant dynamic timer status that does not generate an interrupt. This register is read only.

Diagram



Fields

Field	Function
31 RCD	Timer Reference Clock Detected The selected timer clock, TMR_CTRL[CK_SEL], is sensed as being active. When it is not active, register read/write accesses to timer clock domain registers will not be allowed. Refer to Table "Timer Clock Domain Register List" and Table "Platform Clock Domain Register List". **NOTE** This bit is not functional when the external system clock selected is not active. Access to the TMR registers in IEEE1588 clock domain will cause a hang. The bit is not functional if the external clock selected becomes inactive after RCD is set to 1. 0b - Reference Clock has not been detected as active. Registers in timer clock domain are not allowed to be accessed; reads return 0, writes are ignored. 1b - Reference Clock has been detected as active. Registers in timer clock domain are allowed to be accessed.
30-26 —	Reserved
25 ETS2_VLD	External trigger 2 Valid time-stamp 0 all valid external trigger time-stamps have been read 1 external trigger has an unread valid time-stamp value
24 ETS1_VLD	External trigger 1 Valid time-stamp 0 all valid external trigger time-stamps have been read 1 external trigger has an unread valid time-stamp value
23-0 —	Reserved

53.4.6.13.14 Timer counter low register (TMR_CNT_L)

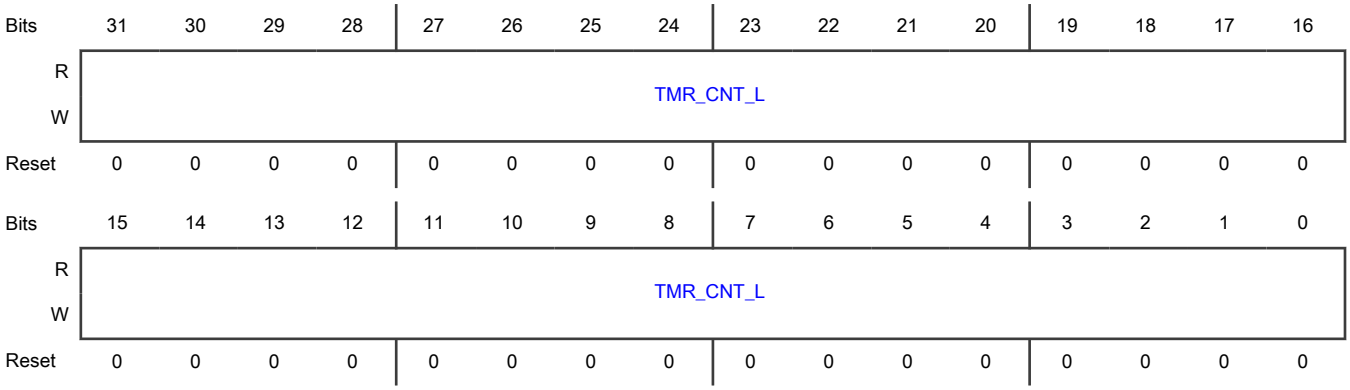
Offset

Register	Offset
TMR_CNT_L	98h

Function

If the timer module is in master mode, the timer register (TMR_CNT_H/L) represents accurate time in terms of nanoseconds. Writes to these registers will override the previous time. This is a read/write register.

Diagram



Fields

Field	Function
31-0 TMR_CNT_L	Timer counter register. Current time is calculated by adding TMROFF_H/L with the TMR_CNT_H/L counter. This register can be written through the register writes. Writes to the TMR_CNT_L register copies the written value into the shadow TMR_CNT_L register. Writes to the TMR_CNT_H register copies the values written into the shadow TMR_CNT_H register. Contents of the shadow registers are copied into the TMR_CNT_L and TMR_CNT_H registers following a write into the TMR_CNT_H register. Writes to these registers have precedence over the timer increment. The user must write to TMR_CNT_L register first. Reads from the TMR_CNT_L register copies the entire 64-bit clock time of the read enable into the TMR_CNT_H/L shadow registers. Read instruction from the TMR_CNT_H register reads the value stored in the TMR_CNT_H shadow register. The user must read the TMR_CNT_L register first to get correct 64-bit TMR_CNT_H/L counter values.

53.4.6.13.15 Timer counter high register (TMR_CNT_H)

Offset

Register	Offset
TMR_CNT_H	9Ch

Function

If the timer module is in master mode, the timer register (TMR_CNT_H/L) represents accurate time in terms of nanoseconds. Writes to these registers will override the previous time. This is a read/write register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TMR_CNT_H															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TMR_CNT_H															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMR_CNT_H	Timer counter register. Current time is calculated by adding TMROFF_H/L with the TMR_CNT_H/L counter. This register can be written through the register writes. Writes to the TMR_CNT_L register copies the written value into the shadow TMR_CNT_L register. Writes to the TMR_CNT_H register copies the values written into the shadow TMR_CNT_H register. Contents of the shadow registers are copied into the TMR_CNT_L and TMR_CNT_H registers following a write into the TMR_CNT_H register. Writes to these registers have precedence over the timer increment. The user must write to TMR_CNT_L register first. Reads from the TMR_CNT_L register copies the entire 64-bit clock time of the read enable into the TMR_CNT_H/L shadow registers. Read instruction from the TMR_CNT_H register reads the value stored in the TMR_CNT_H shadow register. The user must read the TMR_CNT_L register first to get correct 64-bit TMR_CNT_H/L counter values.

53.4.6.13.16 Timer addend register (TMR_ADD)

Offset

Register	Offset
TMR_ADD	A0h

Function

Timer addend register holds the fractional part of the timer clock period.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ADDEND															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDEND															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ADDEND	Timer addend. Set to ceiling((2^32)*FractionalFreqComp). Example: for TimerFreq = 240 MHz = 4.16667 ns = ceil(2^32*0.16667) = 0x2AAA_AAAB

53.4.6.13.17 Timer accumulator register (TMR_ACC)

Offset

Register	Offset
TMR_ACC	A4h

Function

Timer accumulator register increments by the value of the addend register every cycle. An of the accumulator is used to increment the timer counter by an additional 1 ns.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMR_ACC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMR_ACC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMR_ACC	32-bit timer accumulator register

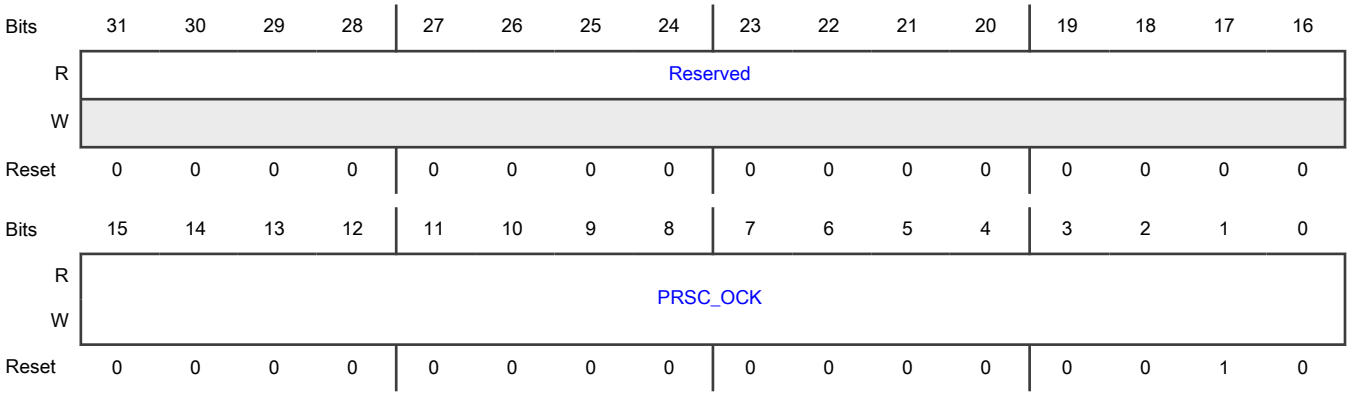
53.4.6.13.18 Timer prescale register (TMR_PRSC)

Offset

Register	Offset
TMR_PRSC	A8h

Function
TMR_PRSC Register Definition

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 PRSC_OCK	Output clock division prescale factor. Output FIPER pulse clock (TMR_GCLK) is generated by dividing the timer input clock by this number. PRSC_OCK must be an even value.

53.4.6.13.19 Extended timer control register (TMR_ECTRL)

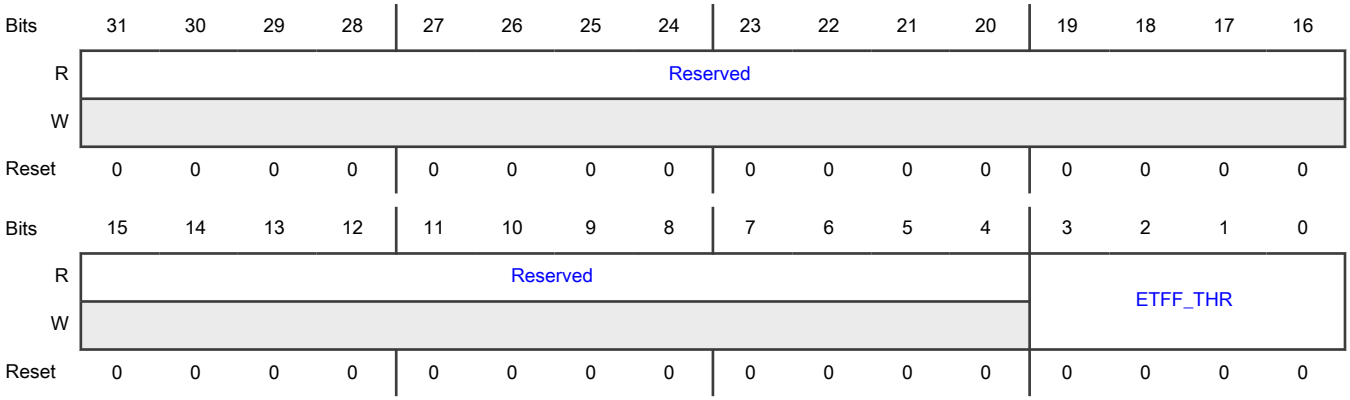
Offset

Register	Offset
TMR_ECTRL	ACh

Function

The Extended Control register provides user additional control of the Timer unit. This sections figure describes the definition for the TMR_ECTRL register.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 ETFF_THR	External trigger FIFO threshold. The External Trigger FIFO Threshold register can control when the TMR_TEVENT[ETTSn] interrupt will assert. It can be used to know when the 16-deep entry FIFO is near full, or to gather several ETTn Time-Stamps before being serviced.0000 - Set TMR_TEVENT[ETTn] when 1 or more entry is in ETTSn FIFO, clear if less0001 - Set TMR_TEVENT[ETTn] when 2 or more entries are in ETTSn FIFO, clear if less0010 - Set TMR_TEVENT[ETTn] when 3 or more entries are in ETTSn FIFO, clear if less...1110 - Set TMR_TEVENT[ETTn] when 15 or more entries are in ETTSn FIFO, clear if less1111 - Set TMR_TEVENT[ETTn] when 16 entries are in ETTSn FIFO, clear if less

53.4.6.13.20 Timer offset low register (TMROFF_L)

Offset

Register	Offset
TMROFF_L	B0h

Function

The timer offset register is used to adjust current time by adding its value to the clock counter. Value is in ns.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TMROFF_L															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TMROFF_L															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMROFF_L	Offset value of the clock counter. Offset value of the clock counter. Current time in is calculated by adding TMROFF_H/L with the timer's counter TMR_CNT_H/L register.

53.4.6.13.21 Timer offset high register (TMROFF_H)

Offset

Register	Offset
TMROFF_H	B4h

Function

The timer offset register is used to adjust current time by adding its value to the clock counter. Value is in ns.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TMROFF_H															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TMROFF_H															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMROFF_H	Offset value of the clock counter. Current time in is calculated by adding TMROFF_H/L with the timer's counter TMR_CNT_H/L register.

53.4.6.13.22 Alarm 1 time comparator low register (TMR_ALARM1_L)

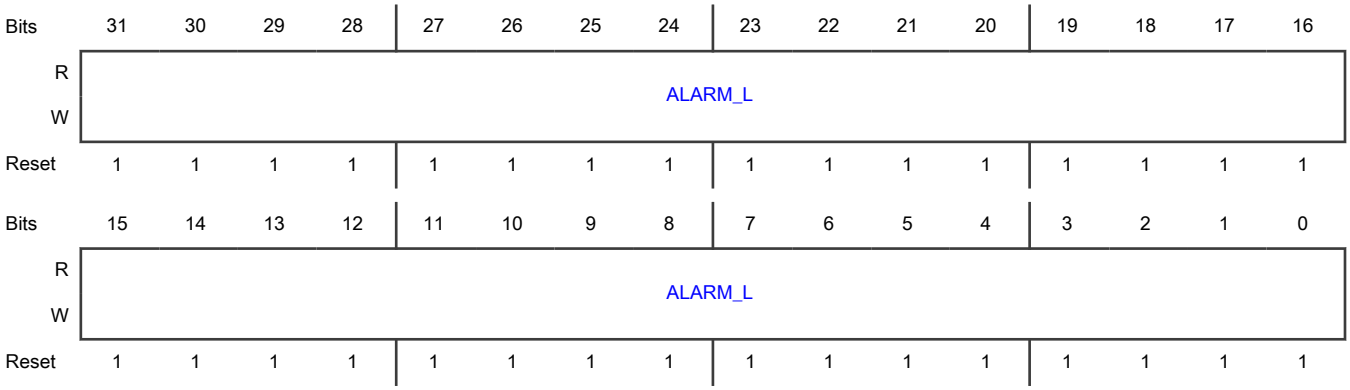
Offset

Register	Offset
TMR_ALARM1_L	B8h

Function

This register holds alarm time for comparison with the current time counter. There are two registers - one per alarm, for the 1588 timer.

Diagram



Fields

Field	Function
31-0 ALARM_L	Alarm time comparator register. The corresponding alarm event in TMR_TEVENT is set when the current time counter becomes equal to or greater than the alarm time compare value in TMR_ALARMn_L/H. By default, once the timer is enabled, the Alarm is armed and will fire if it is not deactivated. Writing the TMR_ALARMn_L register deactivates the alarm event after it has fired. Writing the TMR_ALARMn_L followed by the TMR_ALARMn_H register rearms the alarm function with the new compare value. The value programmed in this register must be an integer multiple of TMR_CTRL[TCLK_PERIOD] in order to get correct result. This register is reset to all ones to avoid false alarm after reset. In FS mode the alarm trigger is used as an indication to the fiper start down counting. Only alarm 1 supports this mode. In FS mode, alarm polarity bit should be configured to 0 (rising edge).

53.4.6.13.23 Alarm 1 time comparator high register (TMR_ALARM1_H)

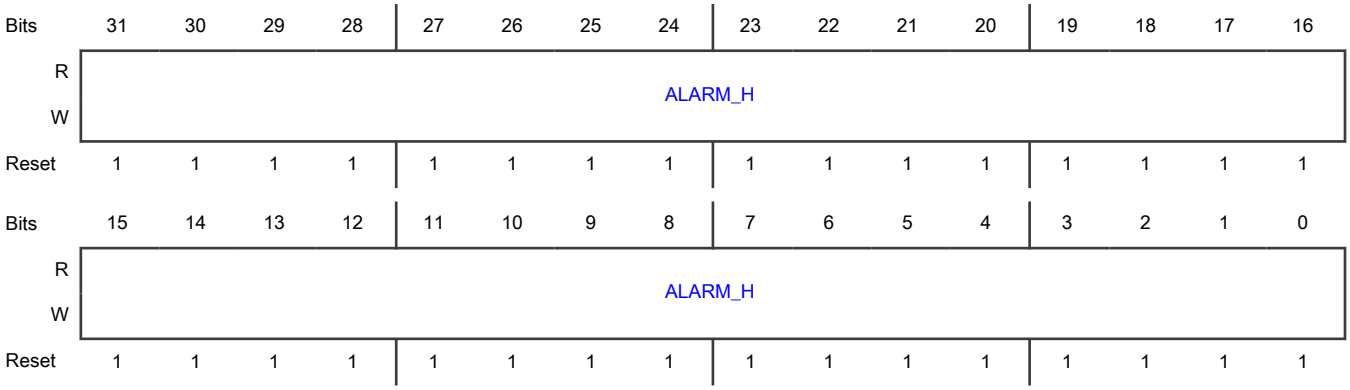
Offset

Register	Offset
TMR_ALARM1_H	BCh

Function

This register holds alarm time for comparison with the current time counter. There are two of these registers for the 1588 timer IP.

Diagram



Fields

Field	Function
31-0 ALARM_H	Alarm time comparator register. The corresponding alarm event in TMR_TEVENT is set when the current time counter becomes equal to or greater than the alarm time compare value in TMR_ALARMn_L/H. By default, once the timer is enabled, the Alarm is armed and will fire if it is not deactivated. Writing the TMR_ALARMn_L register deactivates the alarm event after it has fired. Writing the TMR_ALARMn_L followed by the TMR_ALARMn_H register rearms the alarm function with the new compare value. The value programmed in this register must be an integer multiple of TMR_CTRL[TCLK_PERIOD] in order to get correct result. This register is reset to all ones to avoid false alarm after reset. In FS mode the alarm trigger is used as an indication to the fiper start down counting. Only alarm 1 supports this mode. In FS mode, alarm polarity bit should be configured to 0 (rising edge).

53.4.6.13.24 Alarm 2 time comparator low register (TMR_ALARM2_L)

Offset

Register	Offset
TMR_ALARM2_L	C0h

Function

This register holds alarm time for comparison with the current time counter. There are two of these registers for the 1588 timer IP.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ALARM_L															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ALARM_L															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 ALARM_L	Alarm time comparator register. The corresponding alarm event in TMR_TEVENT is set when the current time counter becomes equal to or greater than the alarm time compare value in TMR_ALARMn_L/H. By default, once the timer is enabled, the Alarm is armed and will fire if it is not deactivated. Writing the TMR_ALARMn_L register deactivates the alarm event after it has fired. Writing the TMR_ALARMn_L followed by the TMR_ALARMn_H register rearms the alarm function with the new compare value. The value programmed in this register must be an integer multiple of TMR_CTRL[TCLK_PERIOD] in order to get correct result. This register is reset to all ones to avoid false alarm after reset. In FS mode the alarm trigger is used as an indication to the fiper start down counting. Only alarm 1 supports this mode. In FS mode, alarm polarity bit should be configured to 0 (rising edge).

53.4.6.13.25 Alarm 2 time comparator high register (TMR_ALARM2_H)

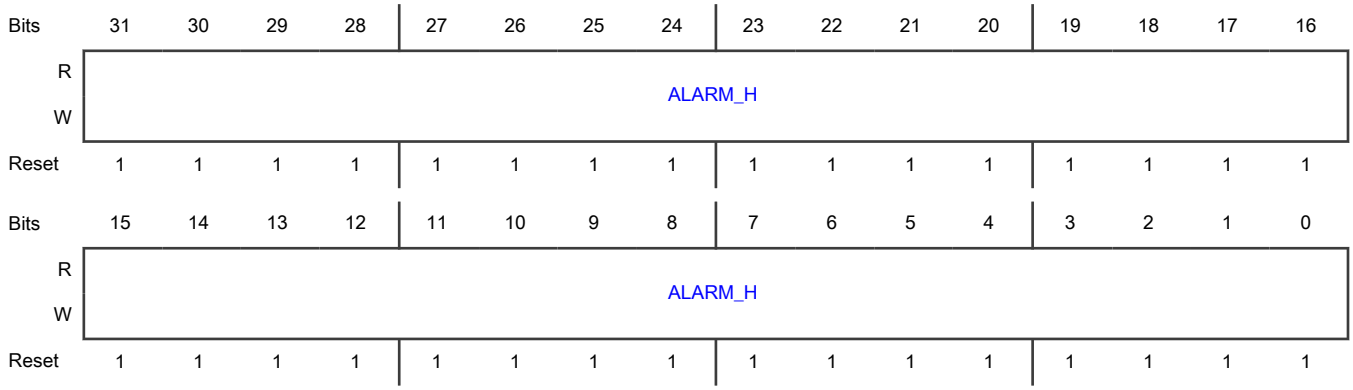
Offset

Register	Offset
TMR_ALARM2_H	C4h

Function

his register holds alarm time for comparison with the current time counter. There are two registers - one per alarm, for the 1588 timer.

Diagram



Fields

Field	Function
31-0 ALARM_H	Alarm time comparator register. The corresponding alarm event in TMR_TEVENT is set when the current time counter becomes equal to or greater than the alarm time compare value in TMR_ALARMn_L/H. By default, once the timer is enabled, the Alarm is armed and will fire if it is not deactivated. Writing the TMR_ALARMn_L register deactivates the alarm event after it has fired. Writing the TMR_ALARMn_L followed by the TMR_ALARMn_H register rearms the alarm function with the new compare value. The value programmed in this register must be an integer multiple of TMR_CTRL[TCLK_PERIOD] in order to get correct result. This register is reset to all ones to avoid false alarm after reset. In FS mode the alarm trigger is used as an indication to the fiper start down counting. Only alarm 1 supports this mode. In FS mode, alarm polarity bit should be configured to 0 (rising edge).

53.4.6.13.26 Timer Alarm Control Register (TMR_ALARM_CTRL)

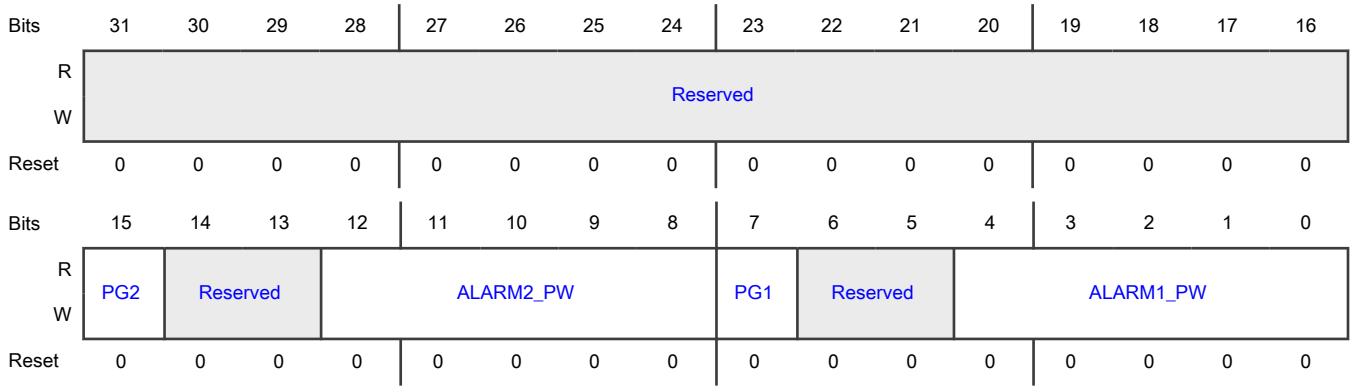
Offset

Register	Offset
TMR_ALARM_CTRL	CCh

Function

This register is used to configure, ALARM1 and ALARM2 details.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 PG2	Alarm2 pulse generation time Defines the time in relation to timer generated clock when alarm1 output will be asserted. 0 - ALARM 2 output asserted immediately. 1 - ALARM 2 output asserted synchronous to timer generated clock.
14-13 —	Reserved
12-8 ALARM2_PW	ALARM 2 pulse width selector This value selects the pulse width in number of timer generated clocks the alarm will be active for. The default value is 0. If PG2 field for alarm2 = 1, Alarm 2 output will be synchronous to timer generated clock. Else, ALARM2 will be asserted as soon as the timer count exceeds programmed value. 5'h00 : Indicates the alarm will be turned off by writing to ALARM2_L register address 1 - 31 : Number of timer generated clocks the ALARM will be active for.
7 PG1	Alarm1 pulse generation time Defines the time in relation to timer generated clock when alarm1 output will be asserted. 0 - ALARM 1 output asserted immediately. 1 - ALARM 1 output asserted synchronous to timer generated clock.
6-5 —	Reserved
4-0 ALARM1_PW	ALARM 1 pulse width selector

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This value selects the pulse width in number of timer generated clocks the alarm will be active for. The default value is 0.If PG1 field for alarm1 = 1, Alarm 1 output will be synchronous to timer generated clock. Else, ALARM1 will be asserted as soon as the timer count exceeds programmed value. 5'h00: Indicates the alarm will be turned off by writing to ALARM1_L register address 1 - 31: Number of timer generated clocks the ALARM will be active for.

53.4.6.13.27 Timer a fixed interval period register (TMR_FIPER1 - TMR_FIPER3)

Offset

Register	Offset
TMR_FIPER1	D0h
TMR_FIPER2	D4h
TMR_FIPER3	D8h

Function

This register is used to control the interval of FIPER pulses.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	FIPER															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FIPER															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 FIPER	Fixed Interval Pulse Period Controls the interval of FIPER pulses. Set to desired FIPER interval in ns - TCLK_PERIOD. This register is reset with 0xFFFF_FFFF. FIPER must be at least 4*PRSC_OCK*TIMER_CLK period. See Generating periodic pulses for more details on generating periodic pulses with FIPER.

53.4.6.13.28 Timer FIPER Control Register (TMR_FIPER_CTRL)

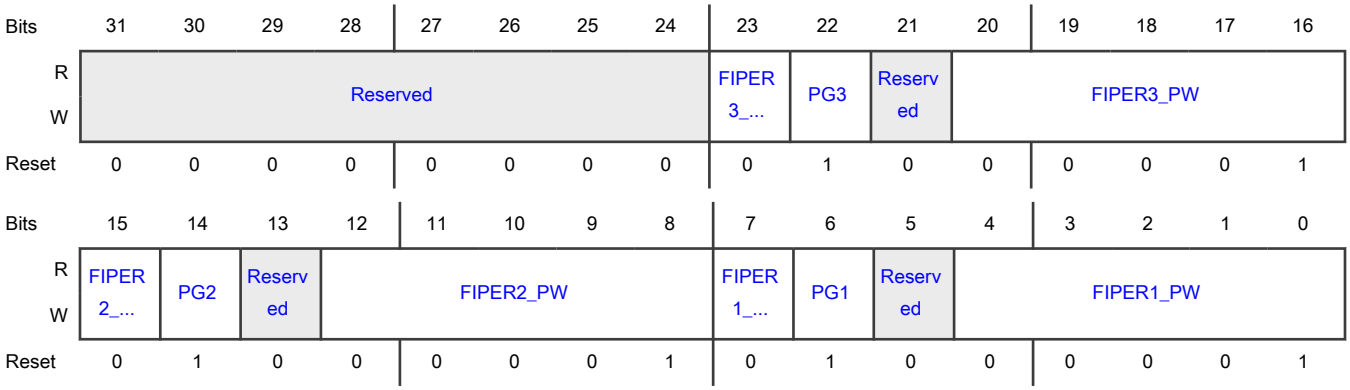
Offset

Register	Offset
TMR_FIPER_CTRL	DCh

Function

This register is used to configure, FIPER1, FIPER2, and FIPER control details.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 FIPER3_DIS	FIPER3 disable Disables FIPER 3 generation. Generation must be disabled before changing FIPER configuration. 0 - FIPER 3 is enabled, and will assert as configured 1 - FIPER 3 is disabled.
22 PG3	FIPER3 pulse generation time Selects whether FIPER3 output is asserted aligned to TMR_GCLK. FIPER3 deassertion is always aligned to TMR_GCLK 0 - FIPER 3 output asserted immediately after interval timer expires. 1 - FIPER 3 output asserted synchronous to timer generated clock. (Default)
21 —	Reserved
20-16	FIPER 3 pulse width selection

Table continues on the next page...

Table continued from the previous page...

Field	Function
FIPER3_PW	FIPER pulse width, in multiples of TMR_GCLK period. The default value is 1. Actual pulse width may be less than FIPER3_PW by up to 1 TMR_GCLK period if PG1=0. The pulse width value must be less than the FIPER interval, with recommended setting less than half of FIPER interval. For example: If PRSC_OCK = 2, timer clock is 200 MHz and FIPERn register is programmed to 200 ns, FIPERn_PW must not be > 20, and recommended value is < 10. 5'h00 : Reserved 1 - 31 : Number of timer generated clocks the FIPER will be active for.
15 FIPER2_DIS	FIPER2 disable Disables FIPER 2 generation. Generation must be disabled before changing FIPER configuration. 0 - FIPER 2 is enabled, and will assert as configured 1 - FIPER 2 is disabled.
14 PG2	FIPER2 pulse generation time Selects whether FIPER2 output is asserted aligned to TMR_GCLK. FIPER2 deassertion is always aligned to TMR_GCLK 0 - FIPER 2 output asserted immediately after interval timer expires. 1 - FIPER 2 output asserted synchronous to timer generated clock. (Default)
13 —	Reserved
12-8 FIPER2_PW	FIPER 2 pulse width selection FIPER pulse width, in multiples of TMR_GCLK period. The default value is 1. Actual pulse width may be less than FIPER2_PW by up to 1 TMR_GCLK period if PG1=0. The pulse width value must be less than the FIPER interval, with recommended setting less than half of FIPER interval. For example: If PRSC_OCK = 2, timer clock is 200 MHz and FIPERn register is programmed to 200 ns, FIPERn_PW must not be > 20, and recommended value is < 10. 5'h00 : Reserved 1 - 31 : Number of timer generated clocks the FIPER will be active for.
7 FIPER1_DIS	FIPER1 disable Disables FIPER 1 generation. Generation must be disabled before changing FIPER configuration. 0 - FIPER 1 is enabled, and will assert as configured 1 - FIPER 1 is disabled.
6 PG1	FIPER1 pulse generation select Selects whether FIPER1 output is asserted aligned to TMR_GCLK. FIPER1 deassertion is always aligned to TMR_GCLK 0 - FIPER 1 output asserted immediately after interval timer expires. 1 - FIPER 1 output asserted synchronous to timer generated clock. (Default)

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 —	Reserved
4-0 FIPER1_PW	FIPER 1 pulse width selection FIPER pulse width, in multiples of TMR_GCLK period. The default value is 1. Actual pulse width may be less than FIPER1_PW by up to 1 TMR_GCLK period if PG1=0. The pulse width value must be less than the FIPER interval, with recommended setting less than half of FIPER interval. For example: If PRSC_OCK = 2, timer clock is 200 MHz (period = 5 ns) and FIPERn register is programmed to 200 ns, FIPERn_PW must not be > 20, and recommended value is < 10. 5'h00 : Reserved 1 - 31 : Number of timer generated clocks the FIPER will be active for.

53.4.6.13.29 External trigger stamp register (TMR_ETTS1_L)

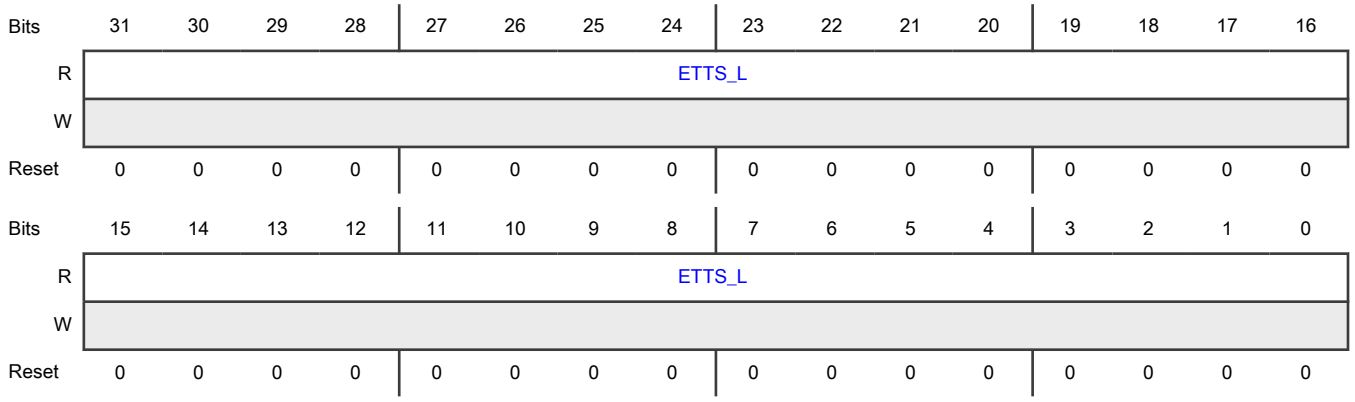
Offset

Register	Offset
TMR_ETTS1_L	E0h

Function

General purpose external trigger-stamp register (TMR_ETTSn_H/L). This register holds time at the programmable edge of the external trigger. This register is read only.

Diagram



Fields

Field	Function
31-0 ETTS_L	Time stamp field at the programmable edge of the external trigger. For time-stamping of back-2-back trigger events, the trigger falling edge can be no closer than 5 timer clocks to next trigger rising edge. There can be as many as 16 time-stamps queued up to be read. To properly read the external trigger time-stamps, user must wait till TMR_TEVENT[ETSn] bit or TMR_TEVENT[ETSn_THR] bit is set before reading the register. Note, both TMR_ETTSn_H and TMR_ETTSn_L must be read as a pair before the next time-stamp in the queue can be read. Else user will be re-reading the same value. For instance, a read to TMR_ETTS1_L, followed by another read to TMR_ETTS1_L, will result in a read of the same value; not until TMR_ETTS1_H is read, in this case, then a new read to TMR_ETTS1_L will return next time-stamp value in the queue. To read till empty, poll TMR_STAT[ETSn_VLD] bit in between reads to the TMR_ETTSn_H/L registers When it is clear, the time-stamp queue is empty. If the ETTSn time-stamp queue overflows, TMR_TEVENT[ETSn_OV] bit will be set, and newer time-stamps will be discarded until space is made available in the queue by a read of ETTSn_H/L pair. Hardware does not provide atomic access to this register. User must insure it is able to read the accurate time-stamp value before the next trigger event occurs.

53.4.6.13.30 External trigger stamp register (TMR_ETTS1_H)

Offset

Register	Offset
TMR_ETTS1_H	E4h

Function

General purpose external trigger-stamp register (TMR_ETTSn_H/L). This register holds time at the programmable edge of the external trigger. This register is read only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ETTS_H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ETTS_H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ETTS_H	Time stamp field at the programmable edge of the external trigger. For time-stamping of back-2-back trigger events, the trigger falling edge can be no closer than 5 timer clocks to next trigger rising edge. There can be as many as 16 time-stamps queued up to be read. To properly read the external trigger time-stamps, user must wait till TMR_TEVENT[ETSn] bit or TMR_TEVENT[ETSn_THR] bit is set before reading the register. Note, both TMR_ETTSn_H and TMR_ETTSn_L must be read as a pair before the next time-stamp in the queue can be read. Else user will be re-reading the same value. For instance, a read to TMR_ETTS1_L, followed by another read to TMR_ETTS1_L, will result in a read of the same value; not until TMR_ETTS1_H is read, in this case, then a new read to TMR_ETTS1_L will return next time-stamp value in the queue. To read till empty, poll TMR_STAT[ETSn_VLD] bit in between reads to the TMR_ETTSn_H/L registers When it is clear, the time-stamp queue is empty. If the ETTSn time-stamp queue overflows, TMR_TEVENT[ETSn_OV] bit will be set, and newer time-stamps will be discarded until space is made available in the queue by a read of ETTSn_H/L pair. Hardware does not provide atomic access to this register. User must ensure it is able to read the accurate time-stamp value before the next trigger event occurs.

53.4.6.13.31 External trigger stamp register (TMR_ETTS2_L)

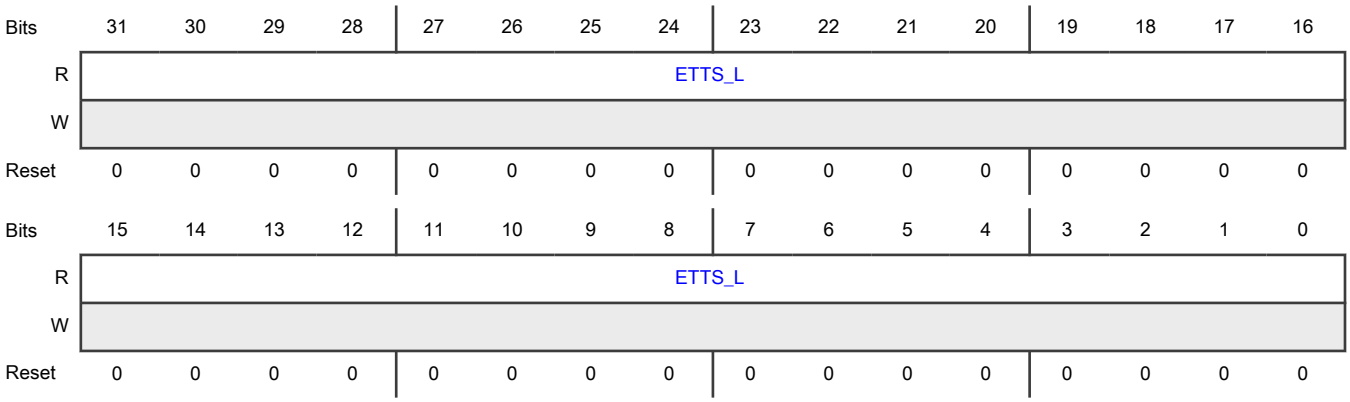
Offset

Register	Offset
TMR_ETTS2_L	E8h

Function

General purpose external trigger-stamp register (TMR_ETTSn_H/L). This register holds time at the programmable edge of the external trigger. This register is read only.

Diagram



Fields

Field	Function
31-0 ETTS_L	Time stamp field at the programmable edge of the external trigger. For time-stamping of back-2-back trigger events, the trigger falling edge can be no closer than 5 timer clocks to next trigger rising edge. There can be as many as 16 time-stamps queued up to be read. To properly read the external trigger time-stamps, user must wait till TMR_TEVENT[ETSn] bit or TMR_TEVENT[ETSn_THR] bit is set before reading the register. Note, both TMR_ETTSn_H and TMR_ETTSn_L must be read as a pair before the next time-stamp in the queue can be read. Else user will be re-reading the same value. For instance, a read to TMR_ETTS1_L, followed by another read to TMR_ETTS1_L, will result in a read of the same value; not until TMR_ETTS1_H is read, in this case, then a new read to TMR_ETTS1_L will return next time-stamp value in the queue. To read till empty, poll TMR_STAT[ETSn_VLD] bit in between reads to the TMR_ETTSn_H/L registers When it is clear, the time-stamp queue is empty. If the ETTSn time-stamp queue overflows, TMR_TEVENT[ETSn_OV] bit will be set, and newer time-stamps will be discarded until space is made available in the queue by a read of ETTSn_H/L pair. Hardware does not provide atomic access to this register. User must insure it is able to read the accurate time-stamp value before the next trigger event occurs.

53.4.6.13.32 External trigger stamp register (TMR_ETTS2_H)

Offset

Register	Offset
TMR_ETTS2_H	ECh

Function

General purpose external trigger-stamp register (TMR_ETTSn_H/L). This register holds time at the programmable edge of the external trigger. This register is read only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ETTS_H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ETTS_H															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ETTS_H	Time stamp field at the programmable edge of the external trigger. For time-stamping of back-2-back trigger events, the trigger falling edge can be no closer than 5 timer clocks to next trigger rising edge. There can be as many as 16 time-stamps queued up to be read. To properly read the external trigger time-stamps, user must wait till TMR_TEVENT[ETSn] bit or TMR_TEVENT[ETSn_THR] bit is set before reading the register. Note, both TMR_ETTSn_H and TMR_ETTSn_L must be read as a pair before the next time-stamp in the queue can be read. Else user will be re-reading the same value. For instance, a read to TMR_ETTS1_L, followed by another read to TMR_ETTS1_L, will result in a read of the same value; not until TMR_ETTS1_H is read, in this case, then a new read to TMR_ETTS1_L will return next time-stamp value in the queue. To read till empty, poll TMR_STAT[ETSn_VLD] bit in between reads to the TMR_ETTSn_H/L registers When it is clear, the time-stamp queue is empty. If the ETTSn time-stamp queue overflows, TMR_TEVENT[ETSn_OV] bit will be set, and newer time-stamps will be discarded until space is made available in the queue by a read of ETTSn_H/L pair. Hardware does not provide atomic access to this register. User must insure it is able to read the accurate time-stamp value before the next trigger event occurs.

53.4.6.13.33 Timer current time low register (TMR_CUR_TIME_L)

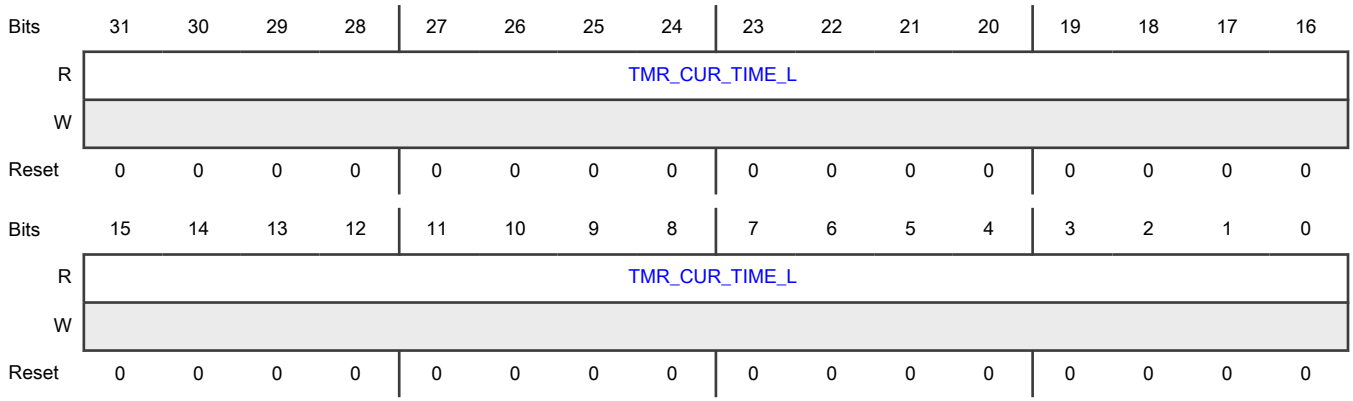
Offset

Register	Offset
TMR_CUR_TIME_L	F0h

Function

This is the timer current time low register. Count is in ns.

Diagram



Fields

Field	Function
31-0 TMR_CUR_TIME_L	Read-only copy of current time (lower 32b). This is calculated by adding TMROFF_H/L with timer's counter TMR_CNT_H/L register.

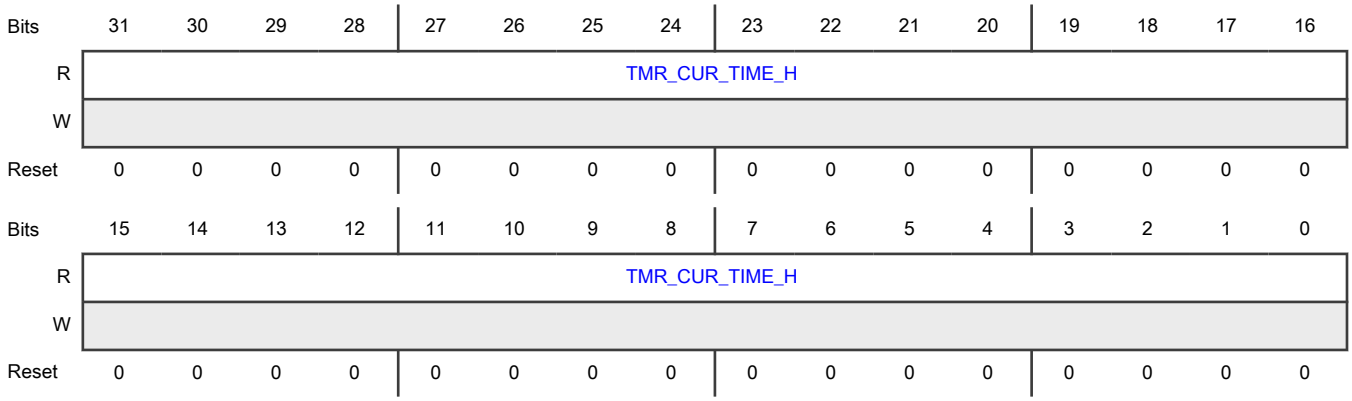
53.4.6.13.34 Timer current time high register (TMR_CUR_TIME_H)

Offset

Register	Offset
TMR_CUR_TIME_H	F4h

Function
This is the timer current time high register. Count is in ns.

Diagram



Fields

Field	Function
31-0 TMR_CUR_TIME_H	Read-only copy of current time (upper 32b). This is calculated by adding TMROFF_H/L with timer's counter TMR_CNT_H/L register.

53.4.6.13.35 Timer parameter register (TMR_PARAM)

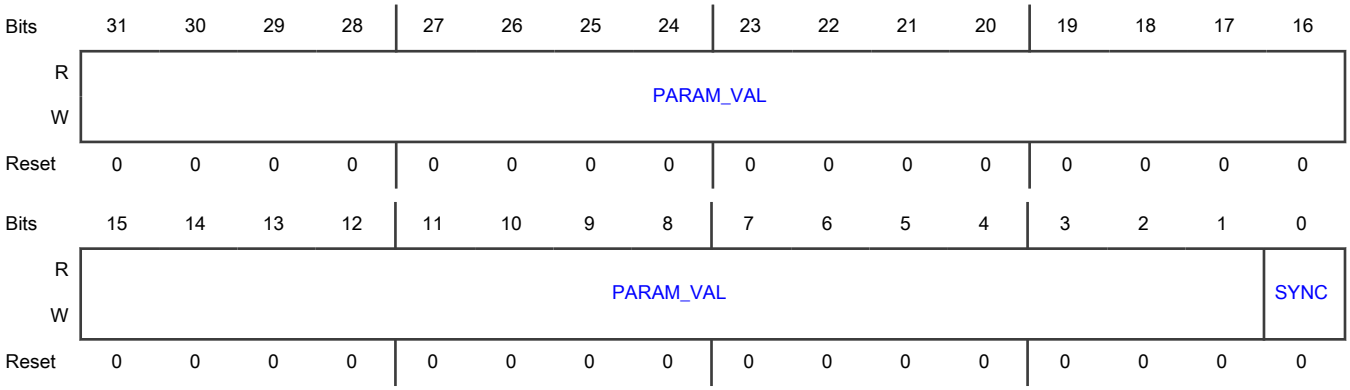
Offset

Register	Offset
TMR_PARAM	F8h

Function

This is the Timer parameter register

Diagram



Fields

Field	Function
31-1 PARAM_VAL	User specific parameter values Software specific information reflected in SI register SITSR
0 SYNC	Timer synchronization 0 Timer not synchronized 1 Timer synchronized

53.4.6.14 NETC global register descriptions

This section describes NETC global device registers which are accessible by all NETC base functions, except where noted, and is applicable to NETC as a whole.

The base address(es) listed is per function and is relative to the PCIe function's base address register value can be discovered from reading the PCIe Enhanced Allocation Base Address Register Equivalent Index 0 (EA BEI 0).

53.4.6.14.1 Global memory map

Instance	Address space	PCIe EA BEI 0 offset
EMDIO_GLOBAL	PCI_F1_BAR_0	60BB_0000h
ENETC0_GLOBAL	PCI_F3_BAR_0	60B2_0000h
ENETC1_GLOBAL	PCI_F4_BAR_0	60B6_0000h
SW0_GLOBAL	PCI_F2_BAR_0	60A8_0000h
TMR0_GLOBAL	PCI_F0_BAR_0	60B9_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Shared memory capability register (SMCAPR)	32	R	0000_1FB0h
4h	Shared memory depletion threshold register (SMDTR)	32	R	0000_0040h
8h	Shared memory available count register (SMACR)	32	R	0000_1F81h
10h	Shared memory count low watermark register (SMCLWMR)	32	R	0000_1F81h
14h	Shared memory buffer unassigned count register (SMBUCR)	32	R	0000_002Fh
18h	Shared memory buffer unassigned count high watermark register (SMBUCHWMR)	32	R	0000_002Fh
1Ch	Shared memory loss count register (SMLCR)	32	R	0000_0000h
20h	Hash bucket table capability register (HBTCAPR)	32	R	3200_0C00h
24h	Hash bucket table operational register 0 (HBTOR0)	32	R	0000_0000h
2Ch	Hash bucket table operational register 2 (HBTOR2)	32	R	0000_0100h
40h	Shared memory ENETC receive buffer capability register (SMERBCAPR)	32	R	0000_05D2h
44h	Shared memory ENETC receive buffer operational register 0 (SMERBOR0)	32	R	0000_0000h
48h	Shared memory ENETC receive buffer operational 1 (SMERBOR1)	32	R	0000_0000h
100h	PCE 0 operational register (PCE0OR)	32	R	0017_0000h
104h	Replication Forwarding Engine 0 operational register (RFE0OR)	32	R	0010_0000h
164h	NETC clock register (NETCCLKR)	32	R	0000_00F0h
200h	HTA 0 capability register (HTA0CAPR)	32	R	0000_0808h
204h	HTA 0 receive frame count operational register (HTA0RFCOR)	32	R	0000_0000h
208h	HTA 0 high priority byte count operational register (HTA0HPBCOR)	32	R	0000_0000h
20Ch	HTA 0 low priority byte count operational register (HTA0LPBCOR)	32	R	0000_0000h
224h	HTA 0 transmit frame count operational register (HTA0TFCOR)	32	R	0000_0000h
300h	Root complex 0 system bus read latency average register (RC0SBRLAR)	32	RW	0000_0000h
304h	Root complex 0 system bus read latency high watermark register (RC0SBRLHWMR)	32	R	0000_0000h
308h	Root complex 0 system bus write latency average register (RC0SBWLAR)	32	RW	0000_0000h
30Ch	Root complex 0 system bus write latency high watermark register (RC0SBWLHWMR)	32	R	0000_0000h
BF8h	IP block revision register 0 (IPBRR0)	32	R	0000_0300h
BFCh	IP block revision register 1 (IPBRR1)	32	R	0001_0100h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
D00h - D04h	Function boot loader parameter register a (FBLPR0 - FBLPR1)	32	R	0000_0000h
E20h	EMDIO uncorrectable fatal system bus error configuration register (EMDIOUFSBECR)	32	RW	0000_0000h
E20h	Timer uncorrectable fatal system bus error configuration register (TUFSBECR)	32	RW	0000_0000h
E24h	EMDIO uncorrectable fatal system bus error status register (EMDIOUFSBESR)	32	RW	0000_0000h
E24h	Timer uncorrectable fatal system bus error status register (TUFSBESR)	32	RW	0000_0000h

53.4.6.14.2 Shared memory capability register (SMCAPR)

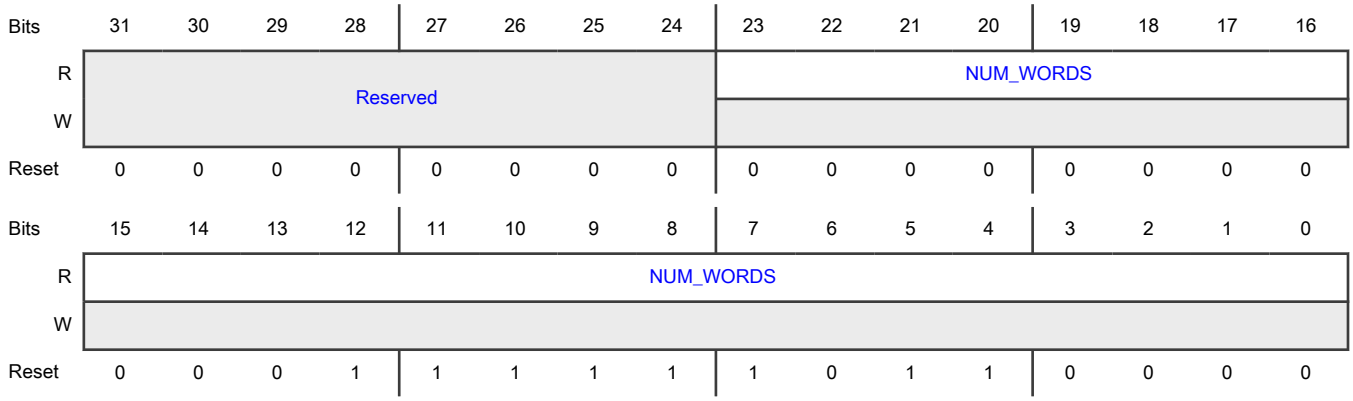
Offset

Register	Offset
SMCAPR	0h

Function

This is the shared memory capability register.

Diagram



Fields

Field	Function
31-24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23-0 NUM_WORDS	Total amount of words in common memory available for free list to NETC buffers, frame descriptors and hash tables. **NOTE** Value derived from IERB as follows: CMCAPR[NUM_WORDS] - HBTMAR[NUM_WORDS] - GHTEMCAPR[NUM_WORDS] - S*a*ltMAR[NUM_WORDS] - E*a*ltMAR[NUM_WORDS]

53.4.6.14.3 Shared memory depletion threshold register (SMDTR)

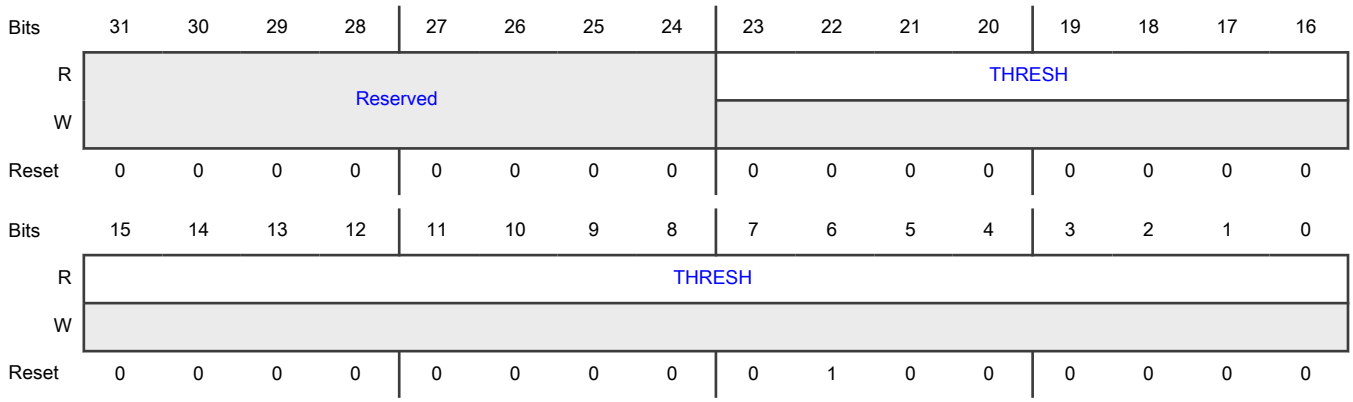
Offset

Register	Offset
SMDTR	4h

Function

This is the shared memory depletion threshold register.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 THRESH	Shared memory depletion threshold in Words.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This sets the minimum amount of free memory that should be maintained in the datapath sub system, and when the amount of shared memory falls below this threshold, a depletion indication is asserted, which may trigger "intelligent drop" of frames from the ingress queues in the ICM. Setting this field to 0 disables dropping of frames by Ingress Congestion Manager (ICM) due to internal shared memory depletion. This value is writable in IERB (by pre-boot initialization), and is immediately reflected here.

53.4.6.14.4 Shared memory available count register (SMACR)

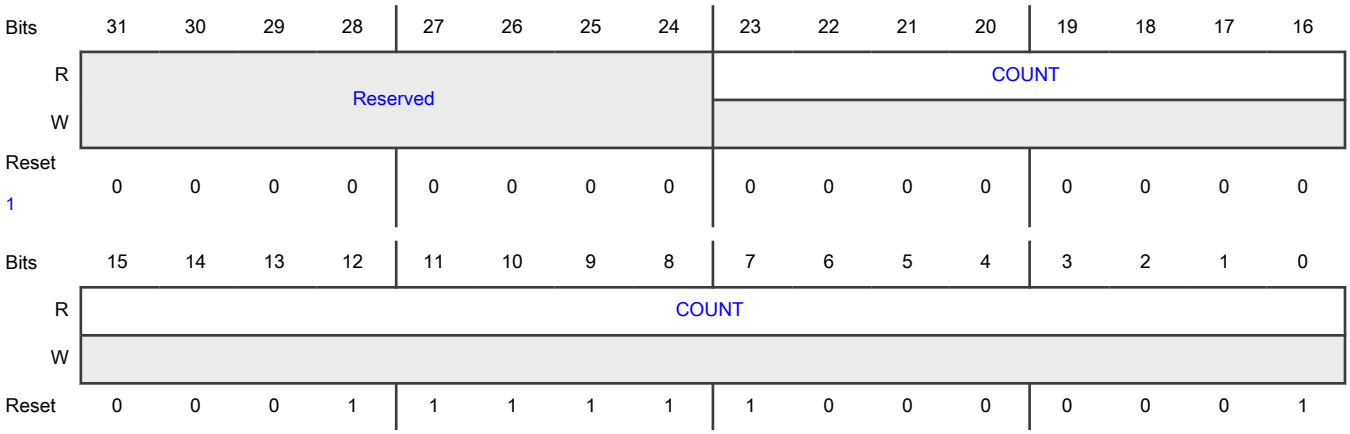
Offset

Register	Offset
SMACR	8h

Function

This is the shared memory available count register.

Diagram



1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-24 —	Reserved
23-0 COUNT	Shows the current amount of available shared memory in words.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Reset value is defined by SMCAPR - SMBUCR.

53.4.6.14.5 Shared memory count low watermark register (SMCLWMR)

Offset

Register	Offset
SMCLWMR	10h

Function

This is the shared memory count low watermark register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								WATERMARK							
W																
Reset [1]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WATERMARK															
W																
Reset	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	1

1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-24 —	Reserved
23-0 WATERMARK	Shows the low watermark for shared memory in words since the last read of this register. When register is read, value is reset to SMACR.

53.4.6.14.6 Shared memory buffer unassigned count register (SMBUCR)

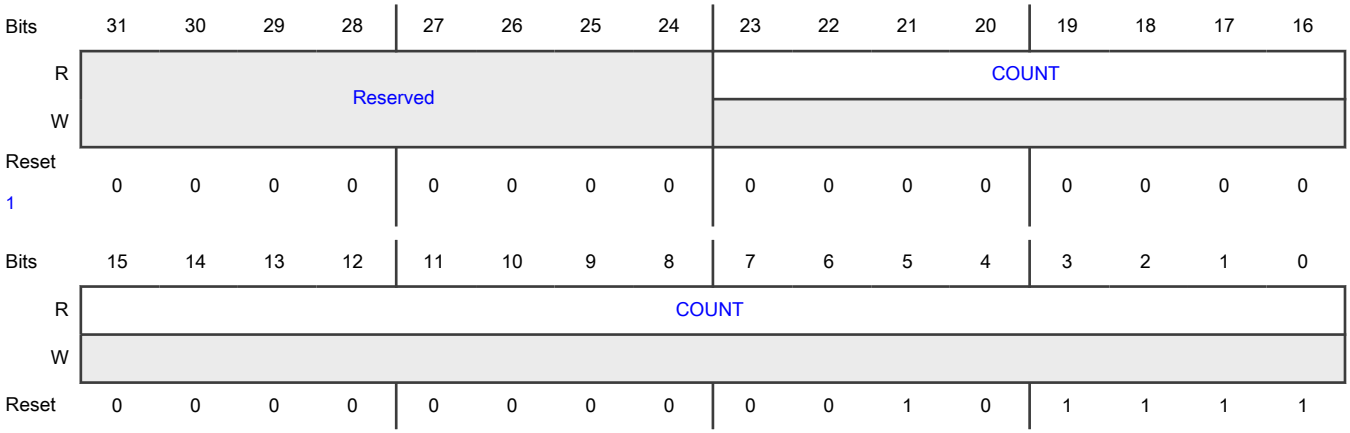
Offset

Register	Offset
SMBUCR	14h

Function

This is the shared memory buffer unassigned count register.

Diagram



1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-24 —	Reserved
23-0 COUNT	Shows the current amount of unassigned Shared memory buffer, in words. This tracks buffer memory from the shared memory partition that has been acquired (i.e. is no longer Free) but has not yet been assigned to switch frame buffering or ENETC frame buffering. This includes all buffer memory used by frames that are in transit between functions, for example in small queues between the receive MAC and the switch or ENETC queues, as well as the shared memory used for hash table entries. Unlike the other budgets, there is no allocation limit for the unassigned buffer memory. In effect, unassigned buffer memory gets to use whatever allocation is left over in the shared memory partition. It is important to note that the datapath sub-system always requires some amount of unassigned buffer memory for in-transit functions, therefore the sum of the switch, ENETC Rx, and hash table allocations must be less than the total available shared buffer memory.

53.4.6.14.7 Shared memory buffer unassigned count high watermark register (SMBUCHWMR)

Offset

Register	Offset
SMBUCHWMR	18h

Function

This is the shared memory buffer unassigned count high watermark register which shows the high watermark for unassigned memory.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								WATERMARK							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1																

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WATERMARK															
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1

1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-24 —	Reserved
23-0 WATERMARK	Shows the high watermark for unassigned memory since the last read of this register, in words. After a read, this register is set equal to SMBUCR.

53.4.6.14.8 Shared memory loss count register (SMLCR)

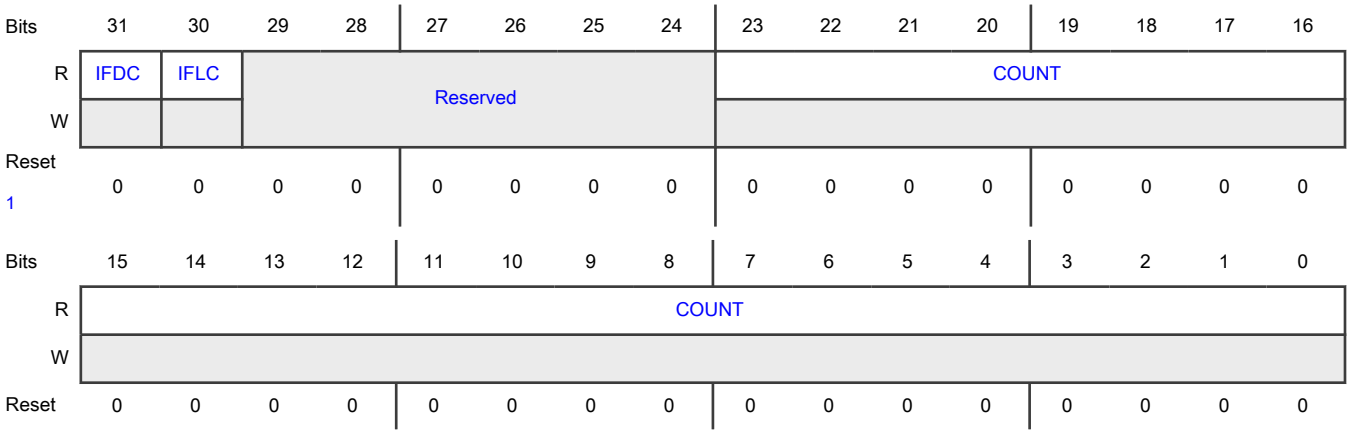
Offset

Register	Offset
SMLCR	1Ch

Function

This is the shared memory loss count register. Read-only count of the amount of shared memory in words (IMUs) lost due to multi-bit ECC memory error.

Diagram



1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31 IFDC	Indeterminate frame descriptor corruption Number of words lost due to frame's key meta data (reference count or number of IMUs) corruption. After a read operation, the field's value clears to 0.
30 IFLC	Indeterminate free list corruption Number of words lost due to free list corruption. Indeterminate number of words lost due to corruption to one of the free list. 1 = one or more IMU free lists has been disabled due to multi-bit ECC error corruption of its link pointers. After a read operation, the field's value clears to 0.
29-24 —	Reserved
23-0 COUNT	Determinate number of lost words. This is a determinate count incremented due to buffer pointer corruption.

53.4.6.14.9 Hash bucket table capability register (HBTCAPR)

Offset

Register	Offset
HBTCAPR	20h

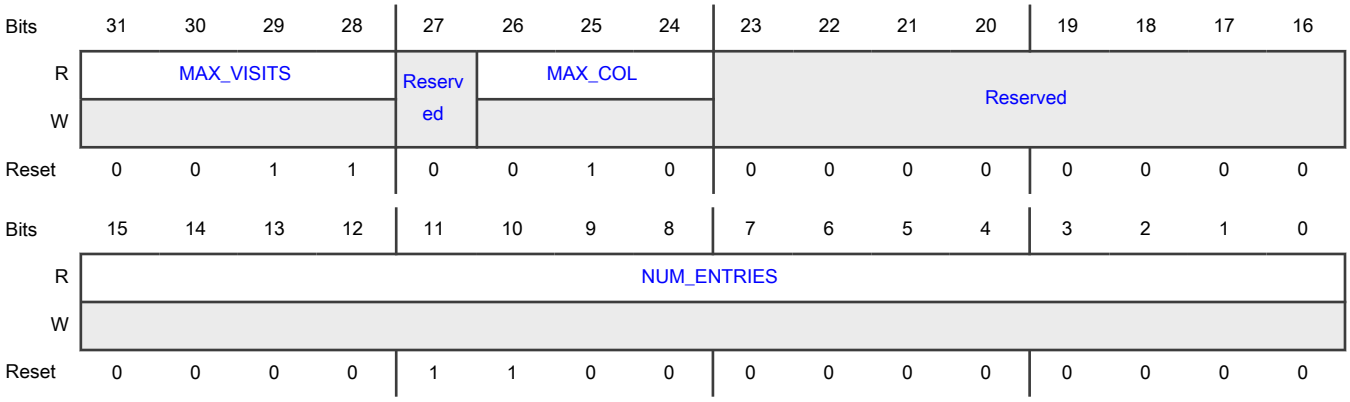
Function

This is the hash bucket table capability register.

NOTE

A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-28 MAX_VISITS	Specifies the maximum number of hash entries visited during a table management search command. Range: 64..1K Formula: 64*(n+1) n=0..15 Reset value determined by IERB register HBTCR.
27 —	Reserved
26-24 MAX_COL	Specifies the maximum exact match hash collisions chain allowed. Range: 1..3 Formula: MAX_COL+1 Entries are not allowed to be added if the hash collisions chain has reached the limit specified in this field. Reset value determined by IERB register HBTCR.
23-16 —	Reserved
15-0 NUM_ENTRIES	Specifies the total number of allocate bucket entries for use.

53.4.6.14.10 Hash bucket table operational register 0 (HBTOR0)

Offset

Register	Offset
HBTOR0	24h

Function

This is the hash bucket table operational register 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HWM_ENTRIES															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1																

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NUM_ENTRIES															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-16 HWM_ENTRIES	Specifies the high watermark of used buckets Register is reset to NUM_ENTRIES after read.
15-0 NUM_ENTRIES	Specifies the amount of entries used

53.4.6.14.11 Hash bucket table operational register 2 (HBTOR2)

Offset

Register	Offset
HBTOR2	2Ch

Function

This is the hash bucket table operational register 2.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved												HWM_COL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1																

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RUN_AVG_INT								RUN_AVG_FRACT							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-20 —	Reserved
19-16 HWM_COL	Length of the longest hash collision chain noticed since the last read. Range: 0..8 After a read operation, the field's value clears to 0.
15-8 RUN_AVG_INT	The integer portion of the running average length of hash lookup. This is to determine the exponential moving average latency of our hash lookup for the various tables being used. After a read operation, the field is set to 1.
7-0 RUN_AVG_FRACT	The fractional portion of the running average length of hash lookup. This is to determine the exponential moving average latency of our hash lookup for the various tables being used. After a read operation, the field's value clears to 0.

53.4.6.14.12 Shared memory ENETC receive buffer capability register (SMERBCAPR)

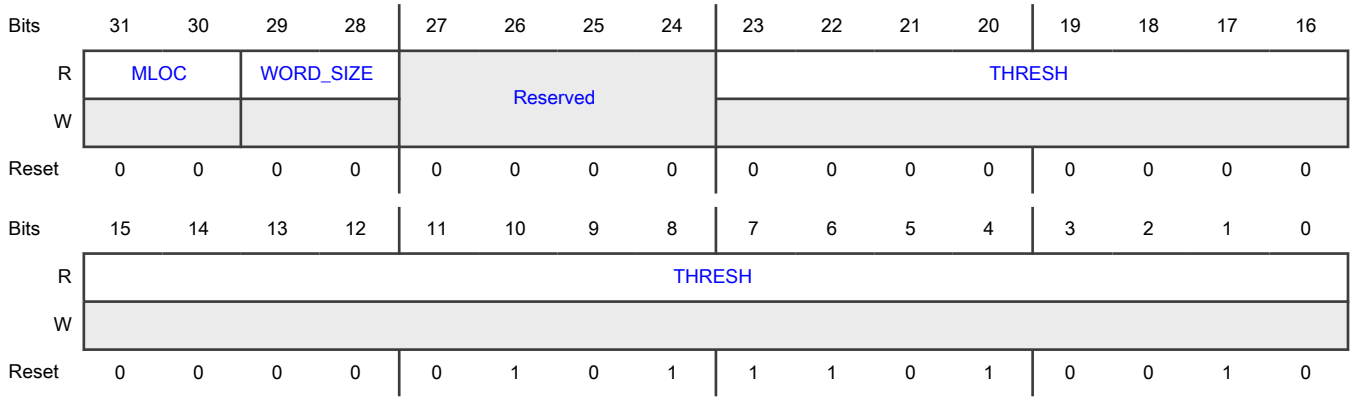
Offset

Register	Offset
SMERBCAPR	40h

Function

This is the shared memory ENETC receive buffer capability register.

Diagram



Fields

Field	Function
31-30 MLOC	Indicates memory location 0: Common memory 1-3: Reserved
29-28 WORD_SIZE	Word size in bytes. 0: 24 Bytes 1-3: reserved
27-24 —	Reserved
23-0 THRESH	Threshold in words of receive buffer memory used by all ENETC functions. If this threshold is exceeded, smart-drop from ICM queues may be triggered. Reset value from IERB ERSMBAR.

53.4.6.14.13 Shared memory ENETC receive buffer operational register 0 (SMERBOR0)

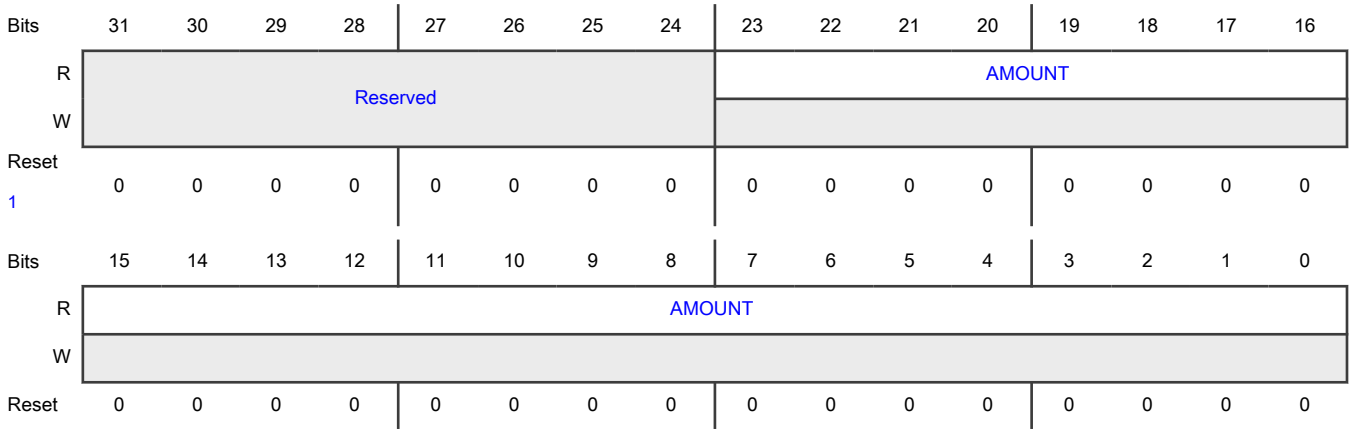
Offset

Register	Offset
SMERBOR0	44h

Function

This is the shared memory ENETC receive buffer operational register 0.

Diagram



1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-24 —	Reserved
23-0 AMOUNT	Number of words in use for buffers and frame descriptors.

53.4.6.14.14 Shared memory ENETC receive buffer operational 1 (SMERBOR1)

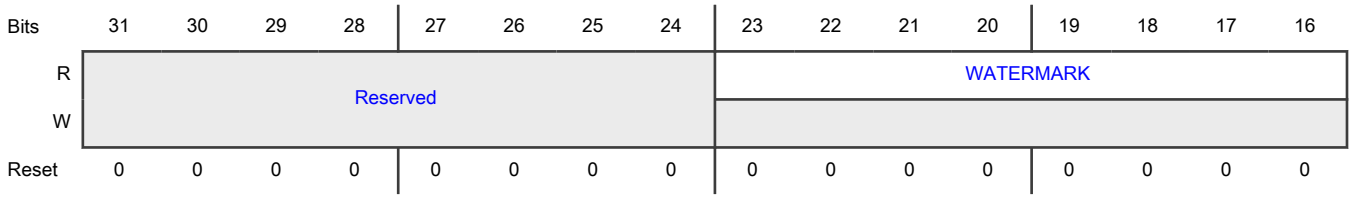
Offset

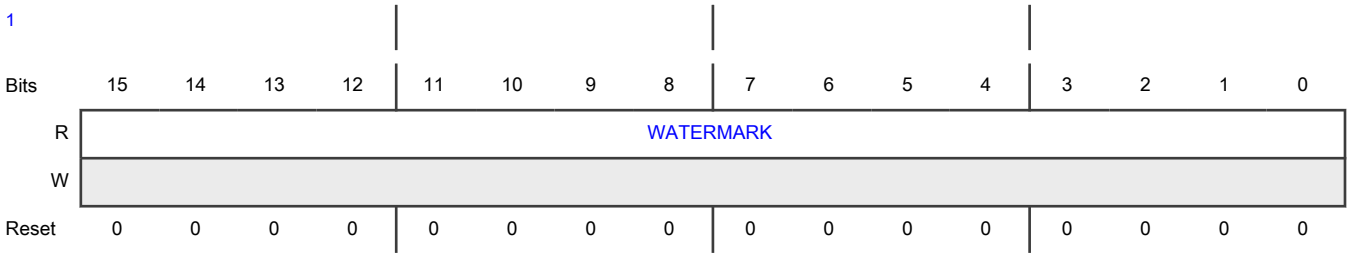
Register	Offset
SMERBOR1	48h

Function

This is the shared memory ENETC receive buffer operational register 1.

Diagram





1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-24 —	Reserved
23-0 WATERMARK	High watermark of words in use by buffers and frame descriptors since the last read. **NOTE** Value reset to AMOUNT after a read.

53.4.6.14.15 PCE 0 operational register (PCE0OR)

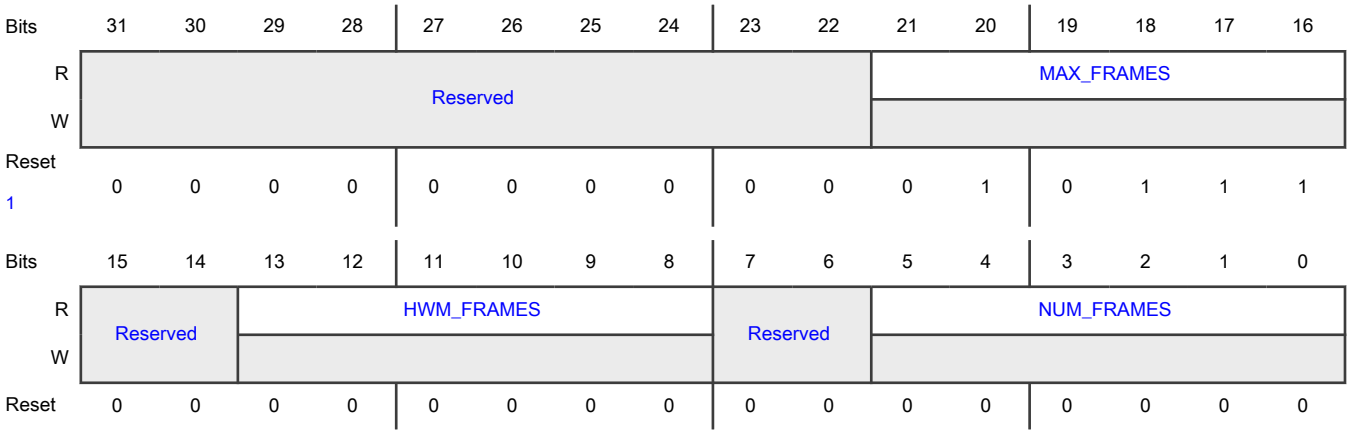
Offset

Register	Offset
PCE0OR	100h

Function

This is the Parse Classifier Engine (PCE) block operational register. The PCE realizes the Ingress Packet Processing (Switch) and Ingress Port Processing (ENETC) functional blocks.

Diagram



1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-22 —	Reserved
21-16 MAX_FRAMES	Maximum number of concurrent frames that can be processed by the PCE block. There are 8 threads per multi-threaded hardware engine. One or more per multi-threaded hardware engines is instantiated inside the PCE hardware block depending on the requirements e.g. number of ports, port speeds. Total number of threads available is equal to 8 multiply by the number of engines. One thread is reserved for processing table management commands. The remaining threads are used to process received frames, with each thread handling a single frame.
15-14 —	Reserved
13-8 HWM_FRAMES	High watermark of concurrent frames being processed since the last read of this register. **NOTE** When register is read, value is equal to NUM_FRAMES.
7-6 —	Reserved
5-0 NUM_FRAMES	Number of active frames currently being processed by the Parse Classifier Engine.

53.4.6.14.16 Replication Forwarding Engine 0 operational register (RFE0OR)

Offset

Register	Offset
RFE0OR	104h

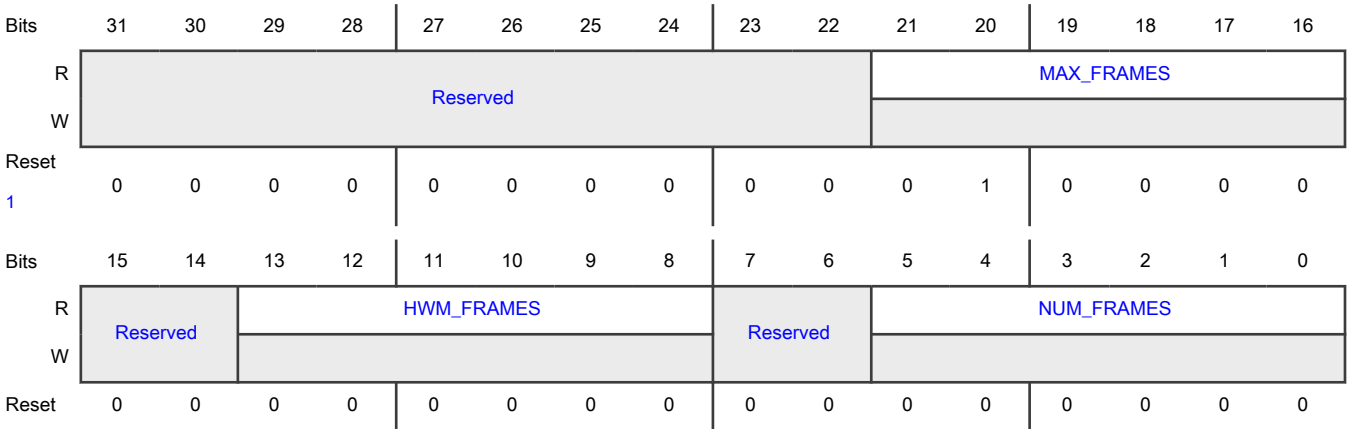
Function

This is the Replication Forwarding Engine (RFE) operational register. RFE realizes the Replication and Egress Packet Processing functional block.

NOTE

A write to this read-only register will return an error.

Diagram



1. Not affected by FLR, requires soft reset.

Fields

Field	Function
31-22 —	Reserved
21-16 MAX_FRAMES	Maximum number of concurrent frames that can be processed by the RFE block. There are 16 threads per multi-threaded hardware engine, where each thread is handling a single frame. One or more per multi-threaded hardware engines is instantiated inside the RFE hardware block depending on the requirements e.g. number of ports, port speeds. Total number of threads available is equal to 16 multiply by the number of engines.
15-14 —	Reserved
13-8 HWM_FRAMES	High watermark of concurrent frames being processed since the last read of this register. **NOTE** When register is read, value is equal to NUM_FRAMES.
7-6 —	Reserved
5-0 NUM_FRAMES	Number of active frames currently being processed by the Replication and Forwarding Engine.

53.4.6.14.17 NETC clock register (NETCCLKR)

Offset

Register	Offset
NETCCLKR	164h

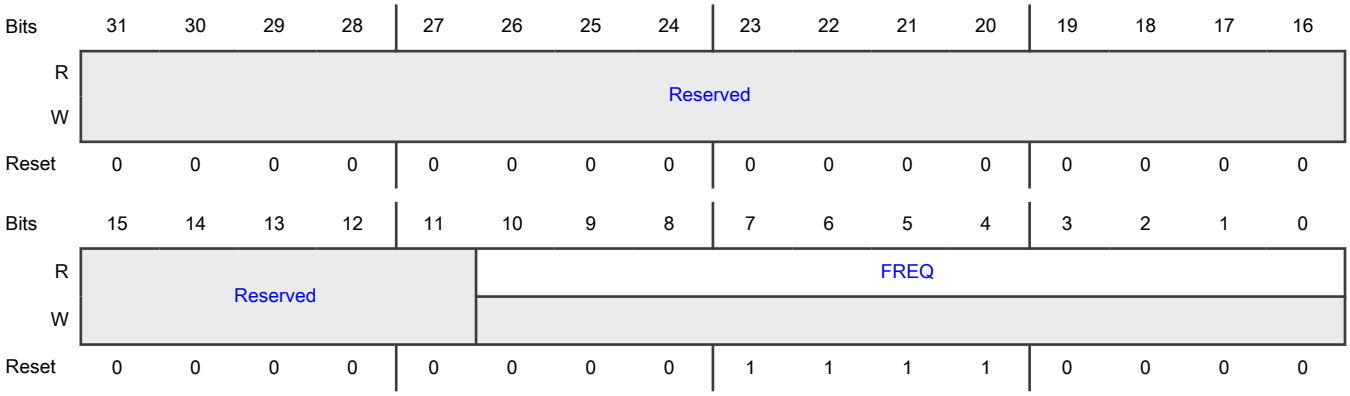
Function

This is the NETC clock register.

NOTE

A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-0 FREQ	Frequency This is the NETC clock frequency in MHz.

53.4.6.14.18 HTA 0 capability register (HTA0CAPR)

Offset

Register	Offset
HTA0CAPR	200h

Function

This is the HTA capability register.

The HTA hardware block realizes the HTA Transmit functional block and the HTA Receive functional block for the ENETC instances. The HTA hardware block is implemented using multi-threaded hardware engine(s) where each engine implements a finite state machine (FSM) which operates on a plurality of receive and transmit threads with each thread handling a single frame. One or more per multi-threaded hardware engines is instantiated inside the HTA hardware block depending on the requirements e.g. number of ports, port speeds. There are 8 receive threads and 8 transmit threads per multi-threaded hardware engine.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAX_TX_FRAMES								MAX_RX_FRAMES							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 MAX_TX_FRAMES	Maximum number of Tx frames the HTA hardware block can process concurrently.
7-0 MAX_RX_FRAMES	Maximum number of Rx frames the HTA hardware block can process concurrently.

53.4.6.14.19 HTA 0 receive frame count operational register (HTA0RFCOR)

Offset

Register	Offset
HTA0RFCOR	204h

Function

This is the HTA receive frame count operational register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LP_HWM								LP_COUNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HP_HWM								HP_COUNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 LP_HWM	Low watermark of concurrent Rx frames being processed since the last read of this register. **NOTE** When register is read, value is equal to LP_COUNT.
23-16 LP_COUNT	Number of active low priority tier Rx frames currently being processed by the host transfer engine.
15-8 HP_HWM	High watermark of concurrent Rx frames being processed by HTA since the last read of this register. **NOTE** When register is read, value is equal to HP_COUNT.
7-0 HP_COUNT	Number of active high priority tier Rx frames currently being processed by the host transfer engine.

53.4.6.14.20 HTA 0 high priority byte count operational register (HTA0HPBCOR)

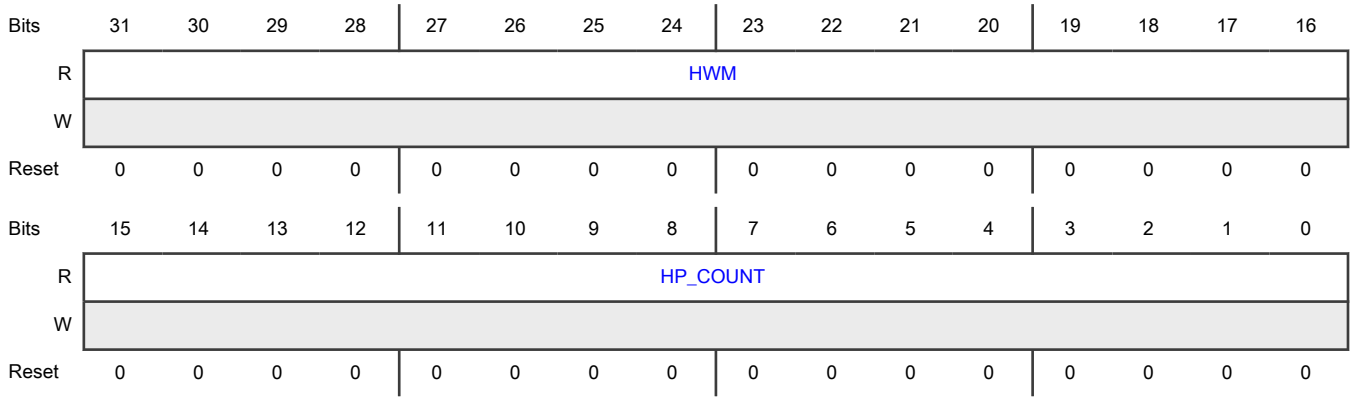
Offset

Register	Offset
HTA0HPBCOR	208h

Function

This is the HTA high priority byte count operational register.

Diagram



Fields

Field	Function
31-16 HWM	High watermark of concurrent frames being processed since the last read of this register. **NOTE** When register is read, value is equal to HP_COUNT. After a read operation, the field's value clears to 0.
15-0 HP_COUNT	Amount of DMA bytes in progress for all high priority tier HTA threads. After a read operation, the field's value clears to 0.

53.4.6.14.21 HTA 0 low priority byte count operational register (HTA0LPBCOR)

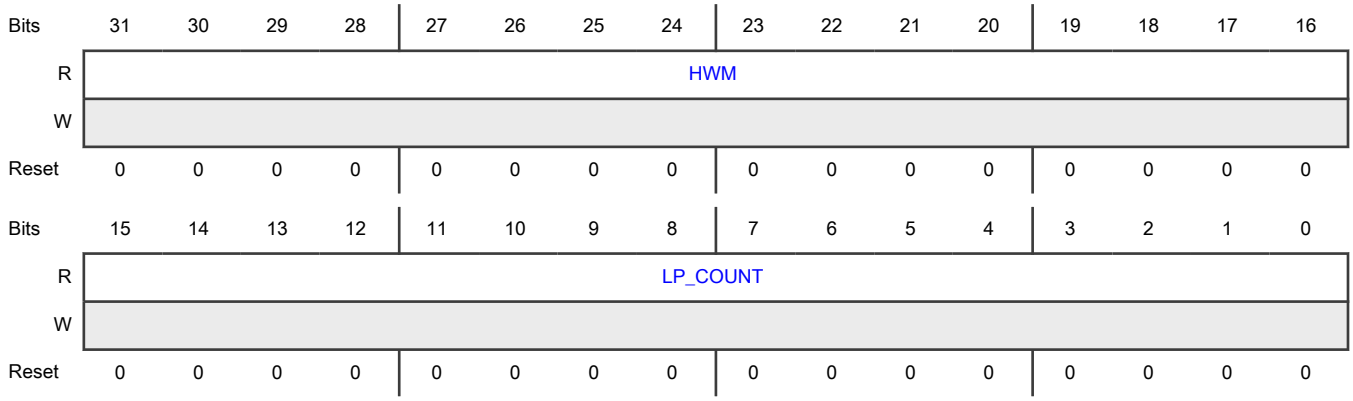
Offset

Register	Offset
HTA0LPBCOR	20Ch

Function

This is the HTA low priority byte count operational register.

Diagram



Fields

Field	Function
31-16 HWM	High watermark of concurrent frames being processed since the last read of this register. **NOTE** When register is read, value is equal to LP_COUNT. After a read operation, the field's value clears to 0.
15-0 LP_COUNT	Amount of DMA bytes in progress for all low priority tier HTA threads. After a read operation, the field's value clears to 0.

53.4.6.14.22 HTA 0 transmit frame count operational register (HTA0TFCOR)

Offset

Register	Offset
HTA0TFCOR	224h

Function

This is the HTA transmit frame count operational register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LP_HWM								LP_COUNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HP_HWM								HP_COUNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 LP_HWM	Low watermark of concurrent Tx frames being processed since the last read of this register. **NOTE** When register is read, value is equal to LP_COUNT.
23-16 LP_COUNT	Number of active low priority tier Tx frames currently being processed by the host transfer engine.
15-8 HP_HWM	High watermark of concurrent Tx frames being processed by HTA since the last read of this register. **NOTE** When register is read, value is equal to HP_COUNT.
7-0 HP_COUNT	Number of active high priority tier Tx frames currently being processed by the host transfer engine.

53.4.6.14.23 Root complex 0 system bus read latency average register (RC0SBRLAR)

Offset

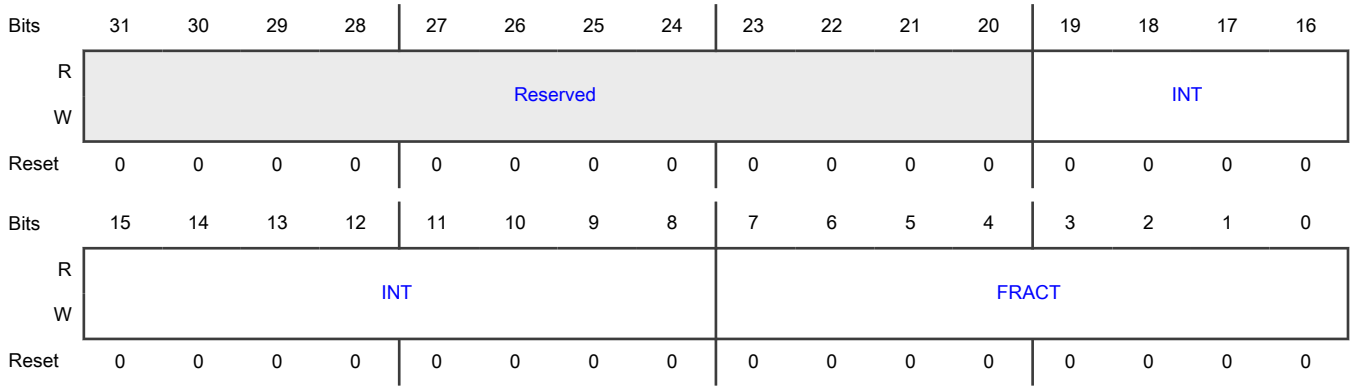
Register	Offset
RC0SBRLAR	300h

Function

The Exponentially Weighted Moving Average (EWMA) is calculated with every latency sample: EWMAnew = (1 - a) * EWMAcurrent + Sample * a; (where a = 1/256). The latency samples are 12-bit integer values (0 to 4095 NETC clock cycles) and a power of 2 weight (1/256). The actual implementation is as follows: Avg_latency_20_bit = Avg_latency_20_bit - (Avg_latency_20_bit >> 8) + sample_12_bit.

Note that after reset, the average value is 0, following which a sufficient number of samples must be taken to arrive at a statistically significant average value reading. To speed up the process of arriving at a significant average value, the average can be initialized to an expected value by writing to this register.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-8 INT	Integer portion of the latency value.
7-0 FRACT	Fractional portion of the latency value.

53.4.6.14.24 Root complex 0 system bus read latency high watermark register (RC0SBRLHWMR)

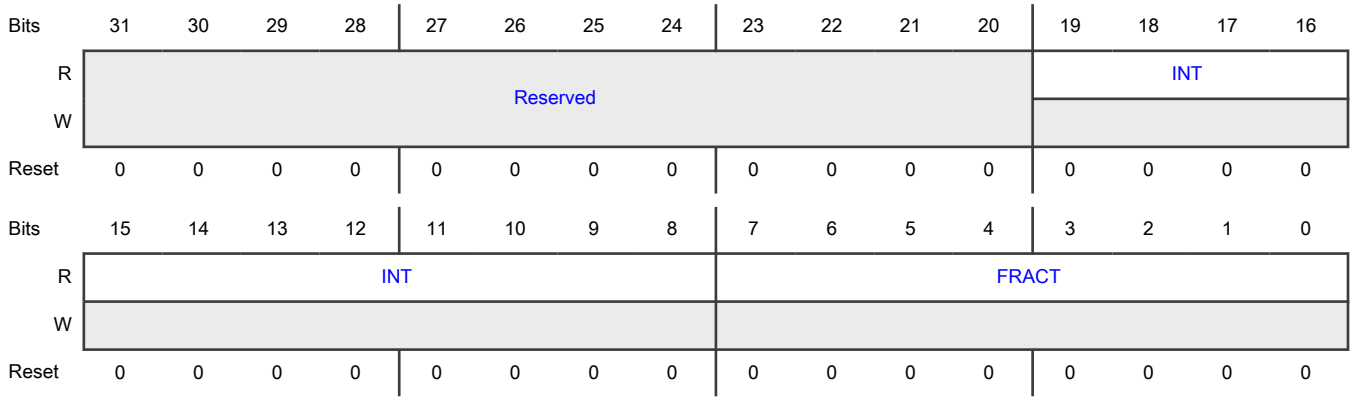
Offset

Register	Offset
RC0SBRLHWMR	304h

Function

A read of this register resets the high watermark.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-8 INT	Integer portion of the latency value.
7-0 FRACT	Fractional portion of the latency value.

53.4.6.14.25 Root complex 0 system bus write latency average register (RC0SBWLAR)

Offset

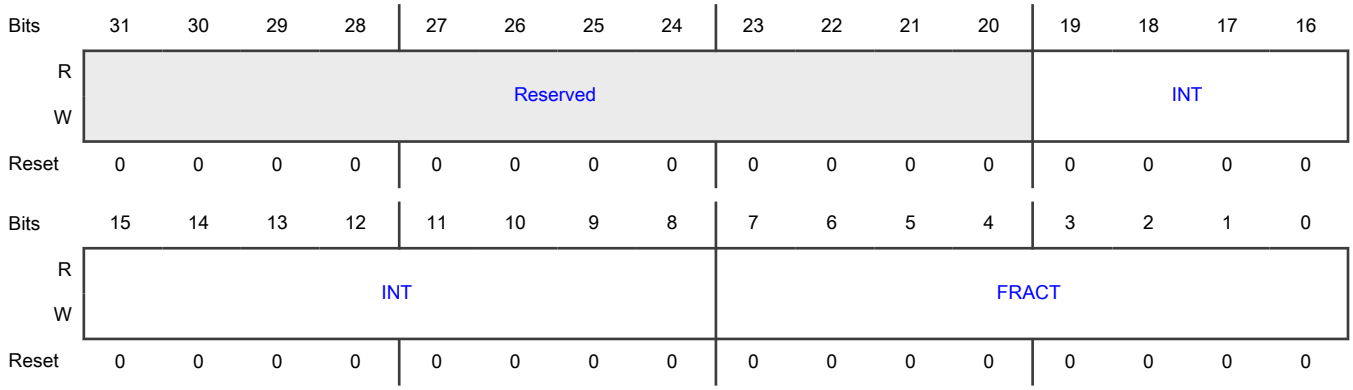
Register	Offset
RC0SBWLAR	308h

Function

The Exponentially Weighted Moving Average (EWMA) is calculated with every latency sample: EWMAnew = (1 - a) * EWMAcurrent + Sample * a; (where a = 1/256). The latency samples are 12-bit integer values (0 to 4095 NETC clock cycles) and a power of 2 weight (1/256). The actual implementation is as follows: Avg_latency_20_bit = Avg_latency_20_bit - (Avg_latency_20_bit >> 8) + sample_12_bit.

Note that after reset, the average value is 0, following which a sufficient number of samples must be taken to arrive at a statistically significant average value reading. To speed up the process of arriving at a significant average value, the average can be initialized to an expected value by writing to this register.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-8 INT	Integer portion of the latency value.
7-0 FRACT	Fractional portion of the latency value.

53.4.6.14.26 Root complex 0 system bus write latency high watermark register (RC0SBWLHWMR)

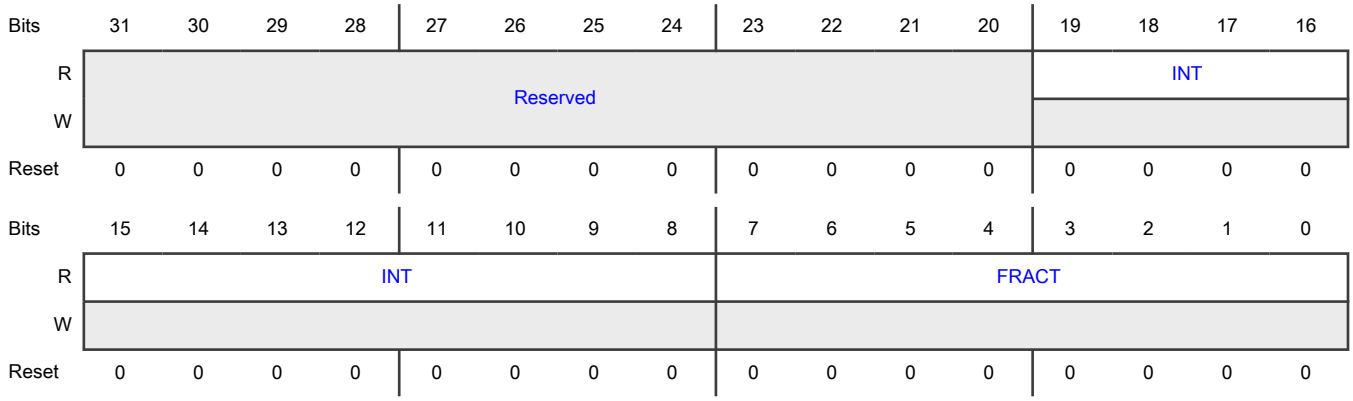
Offset

Register	Offset
RC0SBWLHWMR	30Ch

Function

A read of this register resets the high watermark.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-8 INT	Integer portion of the latency value.
7-0 FRACT	Fractional portion of the latency value.

53.4.6.14.27 IP block revision register 0 (IPBRR0)

Offset

Register	Offset
IPBRR0	BF8h

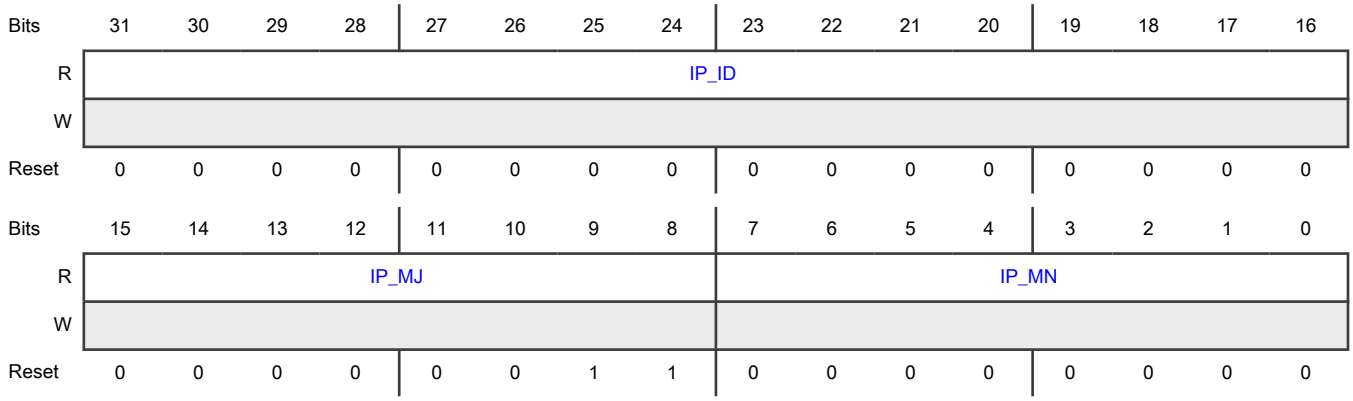
Function

The IP block revision register 0 (IPBRR0) is used to identify the NETC revision.

NOTE

A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-16 IP_ID	IP block ID
15-8 IP_MJ	Major revision
7-0 IP_MN	Minor revision

53.4.6.14.28 IP block revision register 1 (IPBRR1)

Offset

Register	Offset
IPBRR1	BFCh

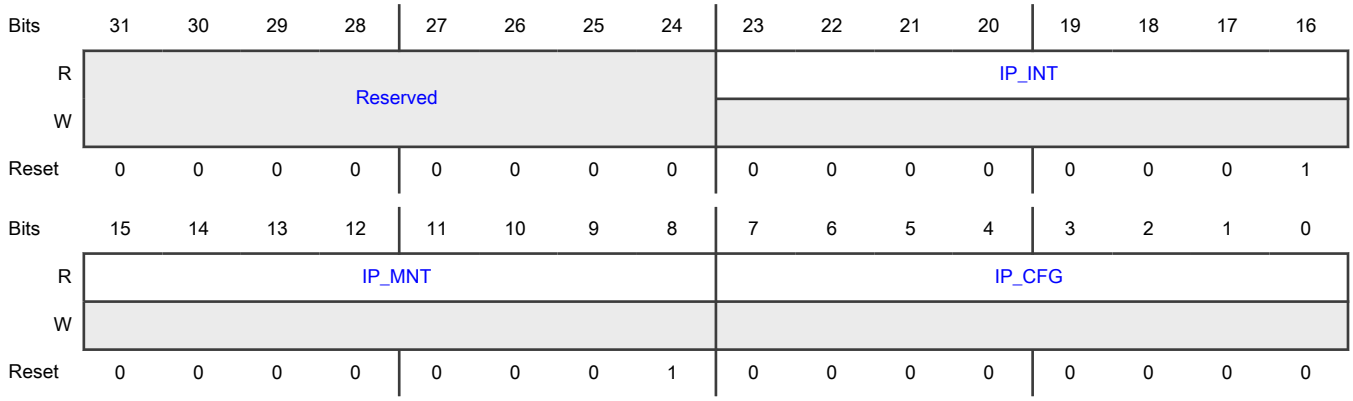
Function

The IP block revision register 1 (IPBRR1) is used to identify NETC's configuration.

NOTE

A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 IP_INT	IP block integration options Bit 23-16: Reserved
15-8 IP_MNT	IP block maintenance version
7-0 IP_CFG	IP block configuration options. Used for debug purposes to identify the capabilities of the present network controller. Bits 7-0: Reserved

53.4.6.14.29 Function boot loader parameter register a (FBLPR0 - FBLPR1)

Offset

Register	Offset
FBLPR0	D00h
FBLPR1	D04h

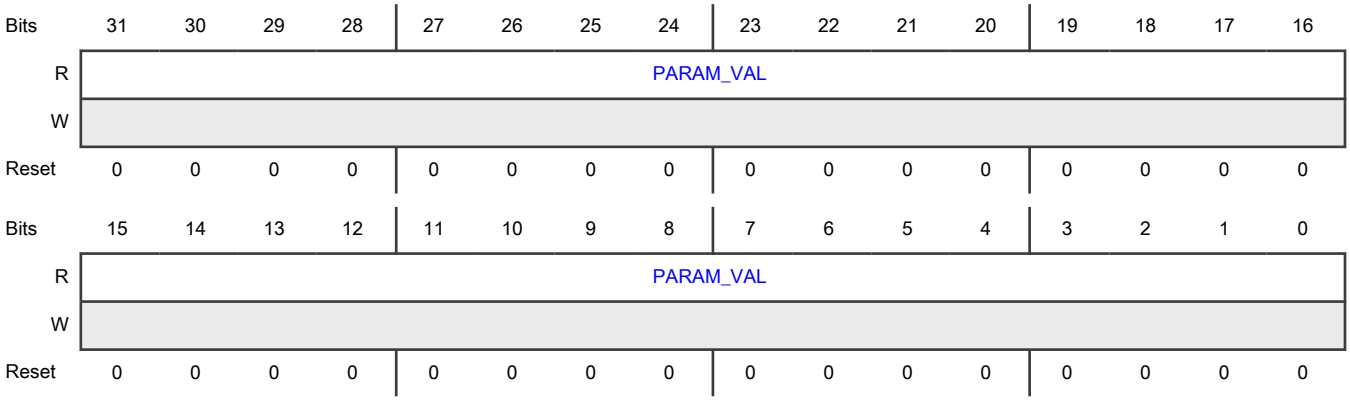
Function

This is the NETC function boot loader parameter register 0. Boot loader parameters are conveyed by S/W through IERB.

NOTE

A write to this read-only register will return an error.

Diagram



Fields

Field	Function
31-0 PARAM_VAL	Boot loader parameter value

53.4.6.14.30 EMDIO uncorrectable fatal system bus error configuration register (EMDIOUFSBECR)

Offset

Register	Offset
EMDIOUFSBECR	E20h

Function

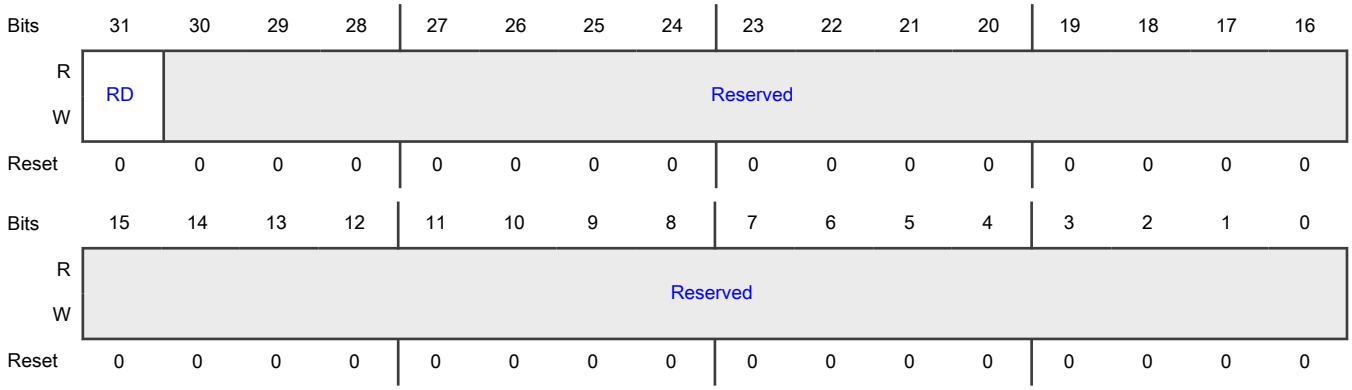
The EMDIO uncorrectable fatal system bus error configuration register can disable event reporting to the PCIe function.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
EMDIO_GLOBAL	EMDIOUFSBECR	—
ENETC0_GLOBAL	—	EMDIOUFSBECR
ENETC1_GLOBAL	—	EMDIOUFSBECR
SW0_GLOBAL	—	EMDIOUFSBECR
TMR0_GLOBAL	—	EMDIOUFSBECR

Diagram



Fields

Field	Function
31 RD	Report disable When reporting is enabled, and EMDIOUFSBESR[SBE] is set, an uncorrectable error is reported to the PCIe function's Advanced Error Reporting capability register, PCIE_CFC_AER_UCORR_ERR_STAT. 0 Enabled 1 Disabled
30-0 —	Reserved

53.4.6.14.31 Timer uncorrectable fatal system bus error configuration register (TUFSBECR)

Offset

Register	Offset
TUFSBECR	E20h

Function

The timer uncorrectable fatal system bus error configuration register can disable event reporting to the PCIe function.

NOTE

Each module instance supports a different number of registers.

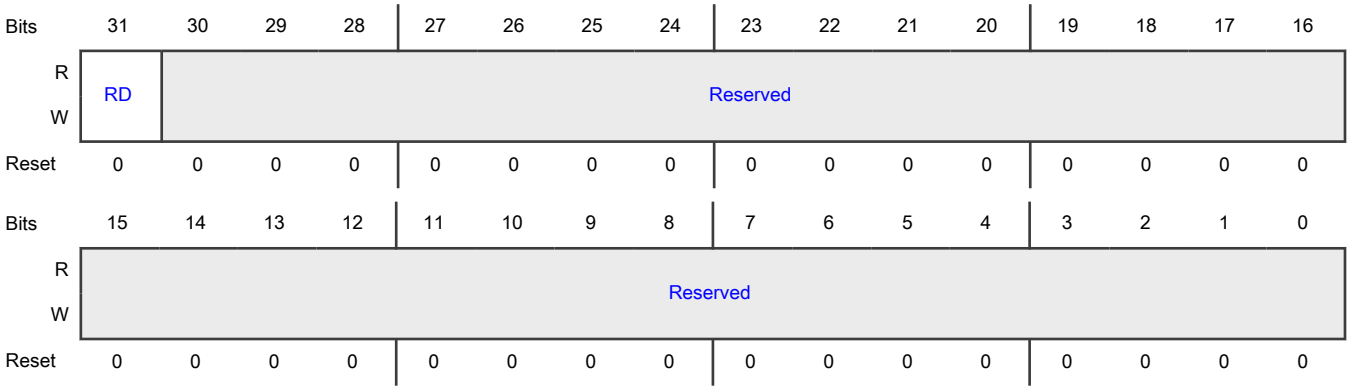
Instance	Register supported	Register not supported
EMDIO_GLOBAL	—	TUFSBECR
ENETC0_GLOBAL	—	TUFSBECR

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
ENETC1_GLOBAL	—	TUFSBECR
SW0_GLOBAL	—	TUFSBECR
TMR0_GLOBAL	TUFSBECR	—

Diagram



Fields

Field	Function
31 RD	Report disable When reporting is enabled, and TUFSBESR[SBE] is set, an uncorrectable error is reported to the PCIe function's Advanced Error Reporting capability register, PCIE_CFC_AER_UCORR_ERR_STAT. 0 Enabled 1 Disabled
30-0 —	Reserved

53.4.6.14.32 EMDIO uncorrectable fatal system bus error status register (EMDIOUFSBESR)

Offset

Register	Offset
EMDIOUFSBESR	E24h

Function

This is the EMDIO uncorrectable fatal system bus error status register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
EMDIO_GLOBAL	EMDIOUFSBESR	—
ENETC0_GLOBAL	—	EMDIOUFSBESR
ENETC1_GLOBAL	—	EMDIOUFSBESR
SW0_GLOBAL	—	EMDIOUFSBESR
TMR0_GLOBAL	—	EMDIOUFSBESR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBE	M	Reserved													
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SB_ID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBE	System bus error When set, a fatal system bus error has occurred for EMDIO MSI-X when attempting to write memory target. The function has entered a safe state: • PCIe bus master enable is cleared (0b0) For recovery, S/W may have to reconfigure MSI-X table for the function. Write 1 to clear.
30 M	Multiple Multiple fatal system bus error events detected. The last event is captured.
29-4 —	Reserved
3-0 SB_ID	System Bus ID See Table 394 for list of system bus source identifiers.

53.4.6.14.33 Timer uncorrectable fatal system bus error status register (TUFSBESR)

Offset

Register	Offset
TUFSBESR	E24h

Function

This is the timer uncorrectable fatal system bus error status register.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
EMDIO_GLOBAL	—	TUFSBESR
ENETC0_GLOBAL	—	TUFSBESR
ENETC1_GLOBAL	—	TUFSBESR
SW0_GLOBAL	—	TUFSBESR
TMR0_GLOBAL	TUFSBESR	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBE	M	Reserved													
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SB_ID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBE	System bus error When set, a fatal system bus errors has occurred as a result of the timer attempting to:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• write MSI-X message to memory The function has entered a safe state: • PCIe bus master enable is cleared (0b0) For recovery, S/W may have to reconfigure MSI-X table for the function. Write 1 to clear.
30 M	Multiple Multiple fatal system bus error events detected. The last event is captured.
29-4 —	Reserved
3-0 SB_ID	System Bus ID See Table 394 for list of system bus source identifiers.

53.4.6.15 RevMII MDIO Register Map

This section provides a detailed description of RevMII MDIO registers. Registers are not memory mapped but accessible from:

- External RevMII MDIO slave interface
- Internal MAC MDIO interface

Reserved bits should be written with 0 and ignored on read. Unused registers read zero and a write has no effect.

53.4.6.15.1 REVMII MDIO register descriptions

This section describes the RevMII MDIO specific registers for one or more ports.

NOTE

LaBCR[MDIO_PHYAD_PRTAD] determines the PHY address of the slave MDIO interface.

53.4.6.15.1.1 REVMII memory map

Instance	Address space	MDIO PHY Address
ENETC0_REVMII_MAC	LINK4_REVMII_MDIO	1Fh
ENETC0_REVMII_PHY	LINK4_REVMII_MDIO	0h
SW_REVMII_MAC2	LINK2_REVMII_MDIO	1Fh
SW_REVMII_MAC3	LINK3_REVMII_MDIO	1Fh
SW_REVMII_PHY2	LINK2_REVMII_MDIO	0h
SW_REVMII_PHY3	LINK3_REVMII_MDIO	0h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control register (CONTROL)	16	RW	1140h
1h	Status register (STATUS)	16	R	9169h
2h	PHY identifier upper register (PHY_ID_U)	16	R	0083h
3h	PHY identifier lower register (PHY_ID_L)	16	R	E400h
4h	AN Advertisement (AN_ADV)	16	RW	0E01h
5h	AN link partner ability (AN_LP_ABIL)	16	R	4E01h
6h	AN Expansion (AN_EXP)	16	R	006Fh
7h	AN Next Page transmit register (AN_NEXT_TX)	16	RW	0000h
8h	AN Link Partner Received Next Page (AN_LP_RX_NEXT_PG)	16	R	0000h
9h	MASTER-SLAVE Control Register (MS_CTL)	16	RW	See section
Ah	MASTER-SLAVE Status Register (MS_STA)	16	R	See section
Fh	Extended Status (XTND_STAT)	16	R	2000h
10h	Scratch register (SCRATCH)	16	RW	0000h

53.4.6.15.1.2 Control register (CONTROL)

Offset

Register	Offset
CONTROL	0h

Function

The RevMII Control Register contains control bits used to enable/disable functions in the PHY, and to initiate commands such as reset.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RST	LOOPBACK	SPEEDLSB	AUTO	PD	ISOLATE	RESTART...	FD	COLL_TE...	SPEEDMSB	Reserved					
W																
Reset	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0

Fields

Field	Function
15 RST	Reset Self-Clearing Read / Write Reset Command Register. When set to 1, a synchronous reset pulse is generated which resets all the RevMII state machines. Should be set to 0 (default) for normal operation. 0b - Normal operation 1b - PHY reset
14 LOOPBACK	Loopback Enable loopback This bit is RO when accessed on the PHY-side by the internal eMAC MDIO and RW when accessed on the MAC-side by an external MDIO. 0b - Disable loopback mode 1b - Enable loopback mode
13 SPEEDLSB	Speed selection LSB Has no meaning if CONTROL[12]=1
12 AUTO	Auto-Negotiate Enable Enables Auto-Negotiate process
11 PD	Power down When set to a 1, TX data is gated off. If the protocol select is RGMII, TXC is also gated off.

Bit 6	Bit 13	Speed selection
1	1	Reserved
1	0	1000 Mb/s
0	1	100 Mb/s
0	0	10 Mb/s

NOTE

Speed selection is done using the combination of CONTROL[SPEEDMSB] and CONTROL[SPEEDLSB] bits.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - Disable powerdown mode
1b - Enable powerdown mode</td></tr>
<tr><td>10
ISOLATE</td><td>Isolate
This bit has no effect</td></tr>
<tr><td>9
RESTART_AN</td><td>Restart Auto-Negotiate
Restart auto-negotiation
If AN enabled, recalculate CONTROL[SPEED]. Self-clearing - reads as 0 if AN is disabled or AN process complete.</td></tr>
<tr><td>8
FD</td><td>Full Duplex
Duplex mode.
0b - Reserved, only Full duplex mode is supported
1b - Full duplex</td></tr>
<tr><td>7
COLL_TEST</td><td>Collision test
Collision test.
This bit has no effect</td></tr>
<tr><td>6
SPEEDMSB</td><td>Speed selection MSB
Has no meaning if CONTROL[12]=1
<table><tr><th>Bit 6</th><th>Bit 13</th><th>Speed selection</th></tr><tr><td>1</td><td>1</td><td>Reserved</td></tr><tr><td>1</td><td>0</td><td>1000 Mb/s</td></tr><tr><td>0</td><td>1</td><td>100 Mb/s</td></tr><tr><td>0</td><td>0</td><td>10 Mb/s</td></tr></table>
NOTE
Speed selection is done using the combination of CONTROL[SPEEDMSB] and CONTROL[SPEEDLSB] bits.</td></tr>
<tr><td>5-0
—</td><td>Reserved</td></tr>
</table>

53.4.6.15.1.3 Status register (STATUS)

Offset

Register	Offset
STATUS	1h

Function

The RevMII Status Register contains status bits on the operation of the PHY.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MODE							EXT_STAT	UNI	PREAMBLE	AN_COMP	REM_FAU...	AN_ABIL	LINKST	JABBER	EXT_CAP
W																
Reset	1	0	0	1	0	0	0	1	0	1	1	0	1	0	0	1

Fields

Field	Function
15-9 MODE	Current speed/duplex

Bit	RO State	Description	Supported
15	0b1	100Base-T4	Supported
14	0b0	100Base-TX FD	Not supported
13	0b0	100Base-TX HD	Not supported
12	0b1	10Mbs FD	Supported
11	0b0	10Mbs HD	Not supported
10	0b0	100Base-T2 FD	Not supported
9	0b0	100Base-T2 HD	Not supported

8 EXT_STAT	Extended status Extended register status Read Only bit always set to 1 to indicate that the PHY implements an extended status register. 0b - Basic register set status 1b - Extended register set status
7 UNI	Unidirectional ability Unidirectional ability

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - PHY able to transmit from media independent interface only when the PHY has determined that a valid link has been established 1b - PHY able to transmit from media independent interface regardless of whether the PHY has determined that a valid link has been established
6 PREAMBLE	Preamble status Preamble status 0b - PHY will not accept management frames with preamble suppressed 1b - PHY will accept management frames with preamble suppressed
5 AN_COMP	Auto-Negotiation Complete Set to 1 if AN enabled and AN advertisement (AN_ADV/AN_NEXT_TX/MS_CTL registers) and partner ability (AN_LP_ABIL/AN_LP_RX_NEXT_PG/MS_STA registers) have at least 1 speed capability bit set in common 0b - Auto-Negotiation process not completed 1b - Auto-Negotiation process completed
4 REM_FAULT	Remote fault Remote fault Read Only bit always set to 0.
3 AN_ABIL	AN Ability Auto-Negotiation Ability Read Only bit always set to 1 to indicate that the PHY implements an extended status register. 0b - Link is not able to perform Auto-Negotiation 1b - Link is not able to perform Auto-Negotiation
2 LINKST	Link status Read-only set to 1 after read if AN complete or AN disabled. LL=once cleared, stays 0 until register read After a read operation, the field's bits are set to 1. 0b - Link is NOT valid. If the link synchronization is lost a 0 is latched which is cleared only after a register read access 1b - Link is valid
1 JABBER	Jabber detect Jabber detect Read Only bit always set to 0.
0 EXT_CAP	Extended Capability Extended register capability

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Read Only bit set to 1 to indicate that the PHY supports extended registers 0b - Basic register set capabilities 1b - Extended register set capabilities

53.4.6.15.1.4 PHY identifier upper register (PHY_ID_U)

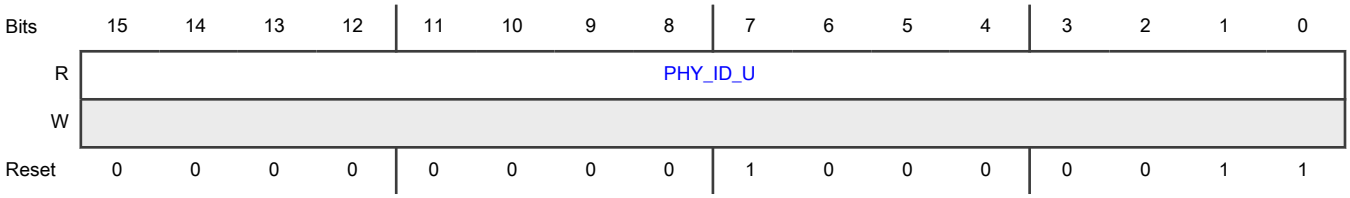
Offset

Register	Offset
PHY_ID_U	2h

Function

The PHY identifier upper register contains the upper half of the 32-bit PHY identifier.

Diagram



Fields

Field	Function
15-0 PHY_ID_U	PHY identifier Upper OUI[3:18]

53.4.6.15.1.5 PHY identifier lower register (PHY_ID_L)

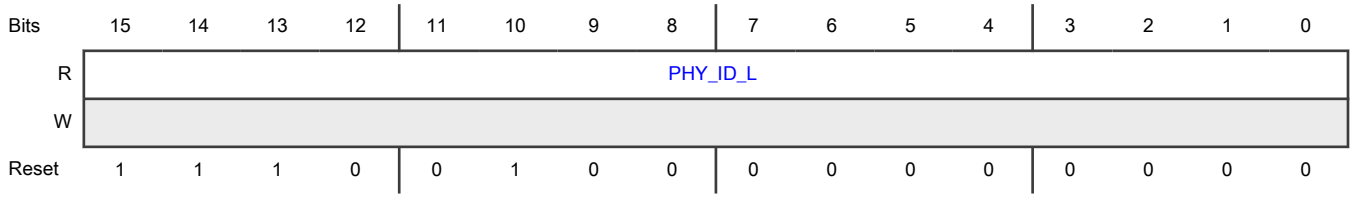
Offset

Register	Offset
PHY_ID_L	3h

Function

The PHY identifier lower register contains the lower half of the 32-bit PHY identifier.

Diagram



Fields

Field	Function
15-0 PHY_ID_L	PHY identifier lower 15:10 OUI[19:24] 9:4 Manufacturer's Model Number 3:0 Revision Number

53.4.6.15.1.6 AN Advertisement (AN_ADV)

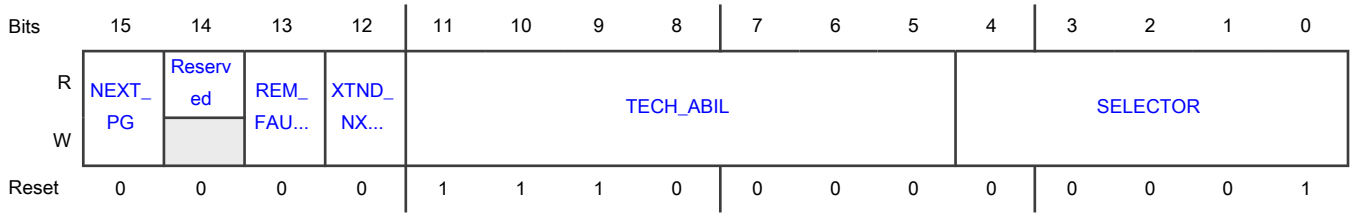
Offset

Register	Offset
AN_ADV	4h

Function

Auto Negotiation Advertisement

Diagram



Fields

Field	Function
15 NEXT_PG	Next page Next page
14 —	Reserved. Write as 0, ignore on read.

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 REM_FAULT	Remote Fault Remote Fault
12 XTND_NXTP	Extended Next Page Extended Next Page
11-5 TECH_ABIL	Technology Ability Field

Bit	Description
11	Asymmetric PAUSE operation for full duplex links
10	PAUSE operation for full duplex links
9	100Base-T4
8	100Base-TX FD
7	100Base-TX
6	10Base-T FD
5	10Base-T

4-0 SELECTOR	Selector Field Selector Field

53.4.6.15.1.7 AN link partner ability (AN_LP_ABIL)

Offset

Register	Offset
AN_LP_ABIL	5h

Function

Auto Negotiation link partner ability

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NEXT_PG	ACK	REM_FAU...	XTND_NX...	TECH_ABIL								SELECTOR			
W																
Reset	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	1

Fields

Field	Function
15 NEXT_PG	Next page Next page
14 ACK	Acknowledge Acknowledge Read-only set to 1 when rest of register is updated
13 REM_FAULT	Remote Fault Remote Fault
12 XTND_NXTP	Extended Next Page Extended Next Page
11-5 TECH_ABIL	Technology Ability Field
4-0 SELECTOR	Selector Field Selector Field

Bit	Description
11	Asymmetric PAUSE operation for full duplex links
10	PAUSE operation for full duplex links
9	100Base-T4
8	100Base-TX FD
7	100Base-TX
6	10Base-T FD
5	10Base-T

53.4.6.15.1.8 AN Expansion (AN_EXP)

Offset

Register	Offset
AN_EXP	6h

Function

Auto Negotiation Expansion

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved									RX_NEXT...	RX_NEXT...	DET_FAU...	LP_NEXT...	NEXT_PG...	PAGE_RC...	LP_AN_A...
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1

Fields

Field	Function
15-7 —	Reserved.
6 RX_NEXT_PG_ABL	Receive Next Page Location Able Receive Next Page Location Able
5 RX_NEXT_PG_LOC	Received Next Page Storage Location Received Next Page Storage Location
4 DET_FAULT	Parallel Detection Fault Parallel Detection Fault
3 LP_NEXT_PG_ABL	Link Partner Next Page able Link Partner Next Page able
2 NEXT_PG_ABL	Next Page Able Next Page Able
1 PAGE_RCVD	Page Received Page Received

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 LP_AN_ABLE	Link Partner Auto-Negotiation Able Link Partner Auto-Negotiation Able

53.4.6.15.1.9 AN Next Page transmit register (AN_NEXT_TX)

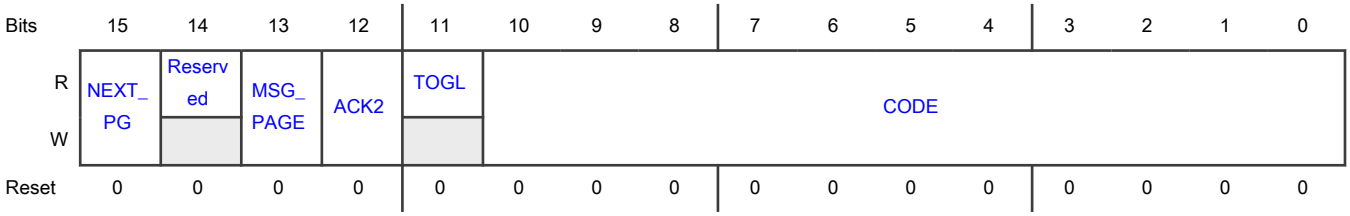
Offset

Register	Offset
AN_NEXT_TX	7h

Function

AN Next Page transmit register

Diagram



Fields

Field	Function
15 NEXT_PG	Next page Next page
14 —	Reserved. Write as 0, ignore on read.
13 MSG_PAGE	Message Page Message Page
12 ACK2	Acknowledge 2 Acknowledge 2
11 TOGL	Toggle Toggle
10-0 CODE	Message/Unformatted Code Message/Unformatted Code

53.4.6.15.1.10 AN Link Partner Received Next Page (AN_LP_RX_NEXT_PG)

Offset

Register	Offset
AN_LP_RX_NEXT_PG	8h

Function

AN Link Partner Received Next Page

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NEXT_PG	ACK	MSG_PAGE	ACK2	TOGL	CODE										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 NEXT_PG	Next page Next page
14 ACK	Acknowledge Acknowledge
13 MSG_PAGE	Message Page Message Page
12 ACK2	Acknowledge 2 Acknowledge 2
11 TOGL	Toggle Toggle TBD
10-0 CODE	Message/Unformatted Code Message/Unformatted Code

53.4.6.15.1.11 MASTER-SLAVE Control Register (MS_CTL)

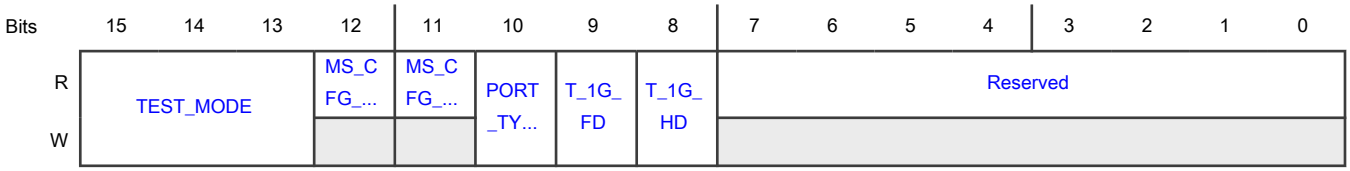
Offset

Register	Offset
MS_CTL	9h

Function

MASTER-SLAVE Control Register

Diagram



See Register reset values.

Register reset values

Register	Reset value
MS_CTL	ENETC0_REVMII_MAC: 1200h ENETC0_REVMII_PHY: 1A00h SW_REVMII_MAC2,SW_REVMII_MAC3: 1200h SW_REVMII_PHY2,SW_REVMII_PHY3: 1A00h

Fields

Field	Function
15-13 TEST_MODE	Test Mode Test Mode
12 MS_CFG_ENABLE	MASTER-SLAVE Manual Config Enable MASTER-SLAVE Manual Config Enable Must be set to 1 (only manual config supported)
11 MS_CFG_VALUE	MASTER-SLAVE Config Value PHY access reads this bit as a 1 MAC access reads this bit as a 0
10 PORT_TYPE	Port type Port type

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 T_1G_FD	1000Base-T Full Duplex 1000Base-T Full Duplex
8 T_1G_HD	1000Base-T Half Duplex 1000Base-T Half Duplex
7-0 —	Reserved. Write as 0, ignore on read.

53.4.6.15.1.12 MASTER-SLAVE Status Register (MS_STA)

Offset

Register	Offset
MS_STA	Ah

Function

MASTER-SLAVE Status Register

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MS_CFG_...	MS_CFG_...	LOC_RX_...	REM_RX_...	LP_1GT_...	LP_1GT_...	Reserved		IDLE_ERR_CNT							
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
MS_STA	ENETC0_REVMII_MAC: 3800h ENETC0_REVMII_PHY: 7800h SW_REVMII_MAC2,SW_REVMII_MAC3: 3800h SW_REVMII_PHY2,SW_REVMII_PHY3: 7800h

Fields

Field	Function
15	MASTER-SLAVE Manual Configuration Fault

Table continues on the next page...

Table continued from the previous page...

Field	Function
MS_CFG_FAULT	Read-only set to 1 if PHY MS_CTL[11]=MAC MS_CTL[11]) (both PHY and MAC are master or both slave) Latch high, clear on read or reset This bit will always read 0, since PHY MS_CTL[11] is RO 1 and MAC MS_CTL is RO 0
14 MS_CFG_RES	MASTER-SLAVE configuration resolution Read-only set to MS_CTL[11]
13 LOC_RX_STAT	Local receiver status Local receiver status
12 REM_RX_STAT	Remote receiver status Remote receiver status
11 LP_1GT_FD	LP 1000T FD LP 1000T FD Read-only set to MS_CTL[9]
10 LP_1GT_HD	LP 1000T HD LP 1000T HD
9-8 —	Reserved. Write as 0, ignore on read.
7-0 IDLE_ERR_CNT	Idle Error Count Idle Error Count

53.4.6.15.1.13 Extended Status (XTND_STAT)

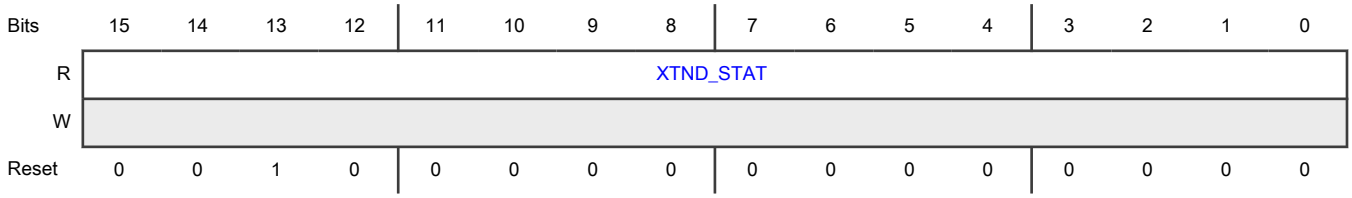
Offset

Register	Offset
XTND_STAT	Fh

Function

The Extended Status Register is reserved, as this device does not implement the optional extended status registers.

Diagram



Fields

Field	Function
15-0 XTND_STAT	Extended Status Reserved. Always 0x2000

Bit	Description
15	1000Base-X FD
14	1000Base-X HD
13	1000Base-T FD
12	1000Base-T HD
10:0	Reserved

53.4.6.15.1.14 Scratch register (SCRATCH)

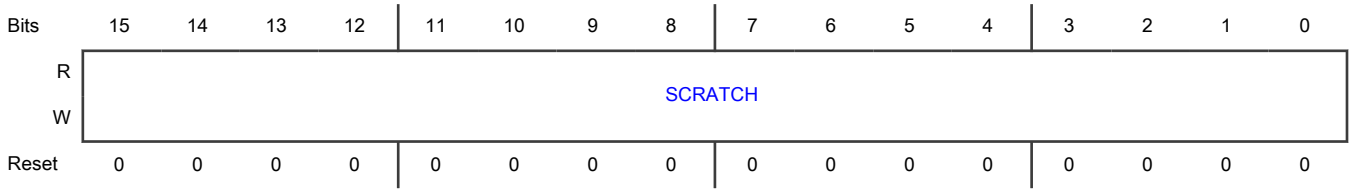
Offset

Register	Offset
SCRATCH	10h

Function

The scratch register contains a read/writeable scratch register that can be used to test MDIO register access.

Diagram



Fields

Field	Function
15-0 SCRATCH	Scratch register Scratch register

Chapter 54
Message Interrupt (MSGINTR)

54.1 Introduction

The Message Interrupt (MSGINTR) module converts memory mapped write transactions to interrupt signals.

Writes to the MSIIR register will simultaneously generate an interrupt signal and set a bit in the MSIR register associated with each interrupt so that multiple “messages” can be sent and processed separately.

There will be three sets of MSIIR/MSIR registers and three associated interrupts.

54.1.1 Overview

The block diagram of MSGINTR is shown below as would be instantiated within a system. The APB interface and the interrupt outputs are clocked at "apb_clk" rate.

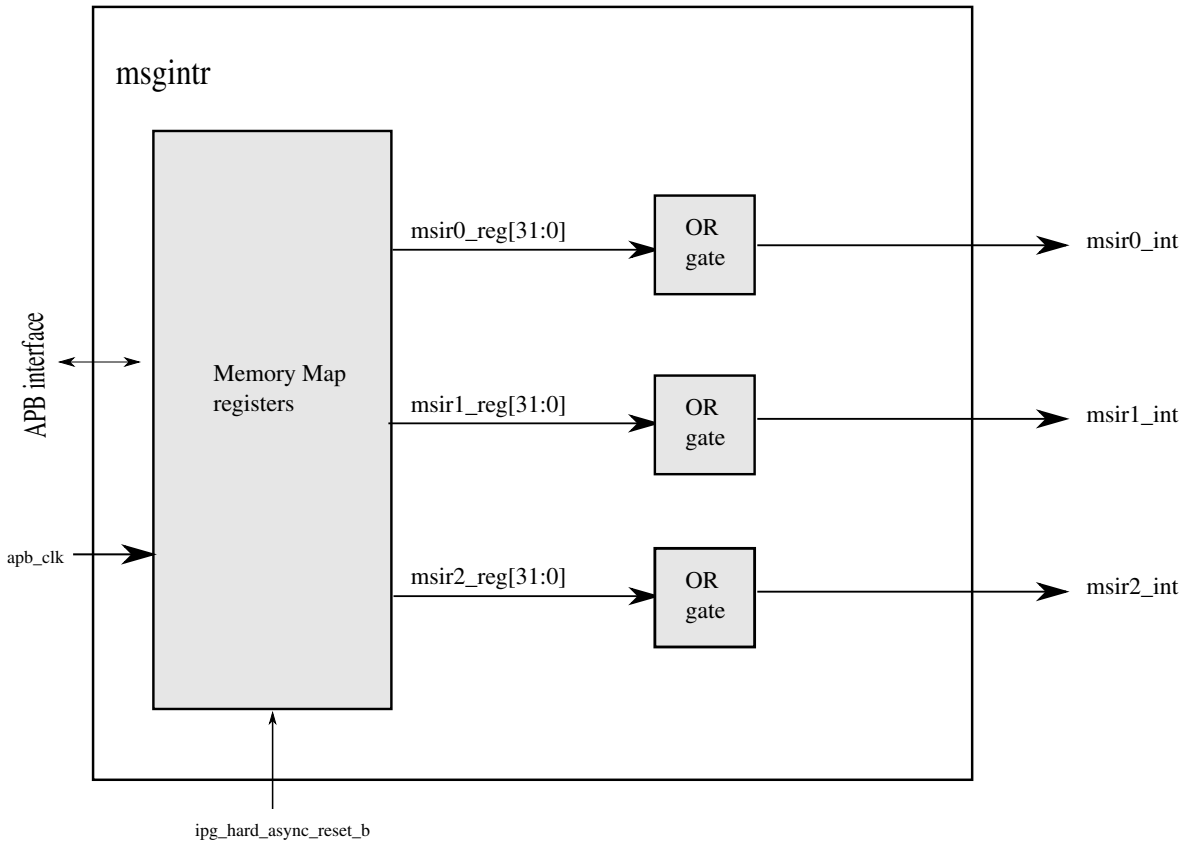


Figure 369. MSGINTR Block Diagram

54.2 External Signal Description

There are no external signals for MSGINTR block.

54.3 Memory Map

This section provides a detailed description of all accessible MSGINTR memory and registers. Reserved bits should be written with 0 and ignored on read. Unused registers read zero and a write has no effect.

MSGINTR uses a contiguous 64KB memory page as shown in Memory Map.

54.4 MSGINTR register descriptions

The table shows the memory map of MSGINTR resources.

54.4.1 MSGINTR memory map

MSGINTR1 base address: 428A_0000h

MSGINTR2 base address: 428B_0000h

MSGINTR3 base address: 428C_0000h

MSGINTR4 base address: 428D_0000h

MSGINTR5 base address: 428E_0000h

MSGINTR6 base address: 428F_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Message Signaled Interrupt Index Register 0 (MSIIR0)	32	RW	0000_0000h
4h	Message Signaled Interrupt Register 0 (MSIR0)	32	R	0000_0000h
8h	Message Signaled Interrupt Index Register 1 (MSIIR1)	32	RW	0000_0000h
Ch	Message Signaled Interrupt Register 1 (MSIR1)	32	R	0000_0000h
10h	Message Signaled Interrupt Index Register 2 (MSIIR2)	32	RW	0000_0000h
14h	Message Signaled Interrupt Register 2 (MSIR2)	32	R	0000_0000h

54.4.2 Message Signaled Interrupt Index Register 0 (MSIIR0)

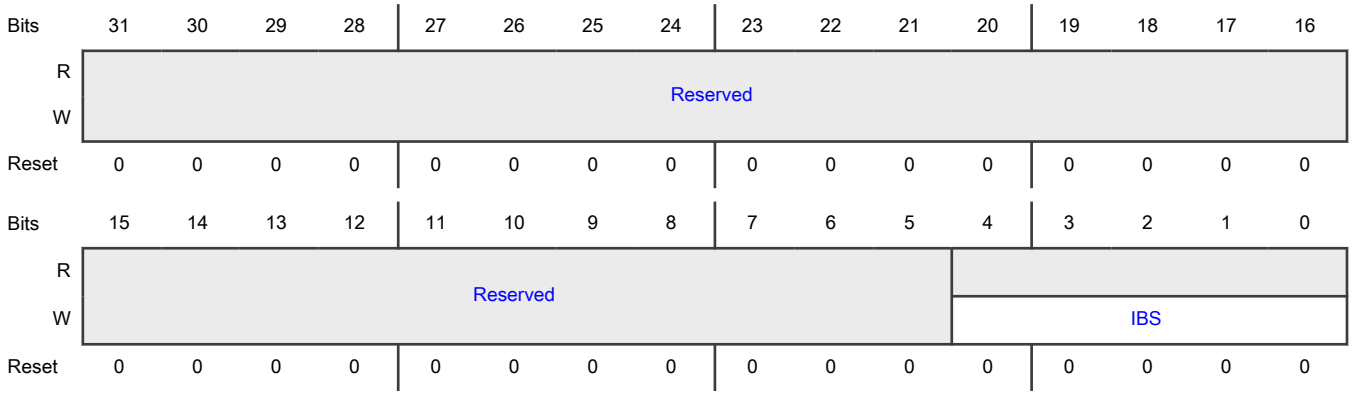
Offset

Register	Offset
MSIIR0	0h

Function

Writing the Interrupt Bit Select (IBS) field of the MSIIR0 register generates an interrupt to the processor and sets the corresponding bit of the MSIR0 registers.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 IBS	Interrupt Bit Select. Selects the bit to set in the MSIR0 register. 00000 Set field SH0 (bit 0) 00001 Set field SH1 (bit 1) 00010 Set field SH2 (bit 2) -- 11111 Set field SH31 (bit 31)

54.4.3 Message Signaled Interrupt Register 0 (MSIR0)

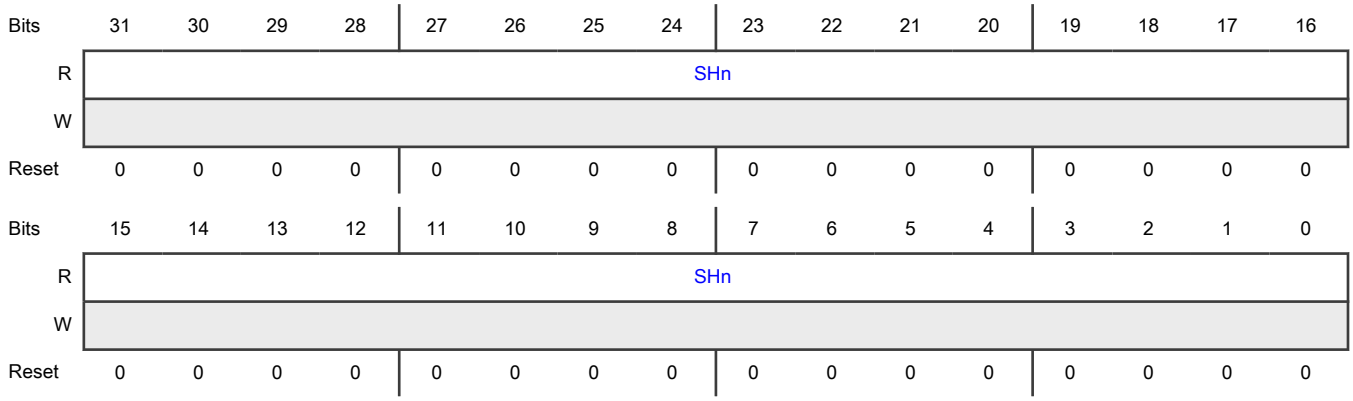
Offset

Register	Offset
MSIR0	4h

Function

This shared message signaled interrupt register indicates which of the up to 32 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; writes to the register have no effect.

Diagram



Fields

Field	Function
31-0 SHn	Message sharer n has a pending interrupt.

54.4.4 Message Signaled Interrupt Index Register 1 (MSIIR1)

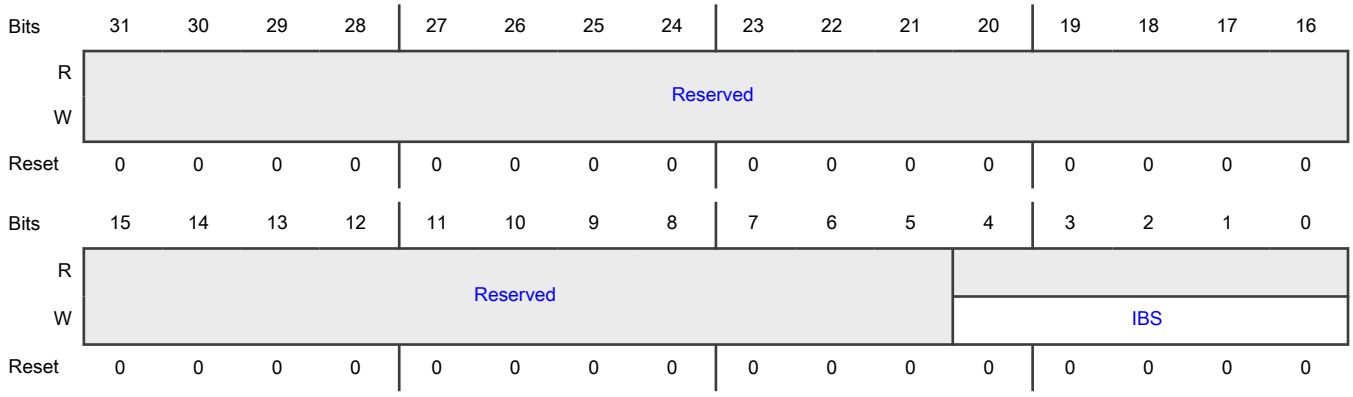
Offset

Register	Offset
MSIIR1	8h

Function

Writing the Interrupt Bit Select (IBS) field of the MSIIR1 register generates an interrupt to the processor and sets the corresponding bit of the MSIR1 registers.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 IBS	Interrupt Bit Select. Selects the bit to set in the MSIR1 register. 00000 Set field SH0 (bit 0) 00001 Set field SH1 (bit 1) 00010 Set field SH2 (bit 2) -- 11111 Set field SH31 (bit 31)

54.4.5 Message Signaled Interrupt Register 1 (MSIR1)

Offset

Register	Offset
MSIR1	Ch

Function

This shared message signaled interrupt register indicates which of the up to 32 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; writes to the register have no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SHn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SHn															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHn	Message sharer n has a pending interrupt.

54.4.6 Message Signaled Interrupt Index Register 2 (MSIIR2)

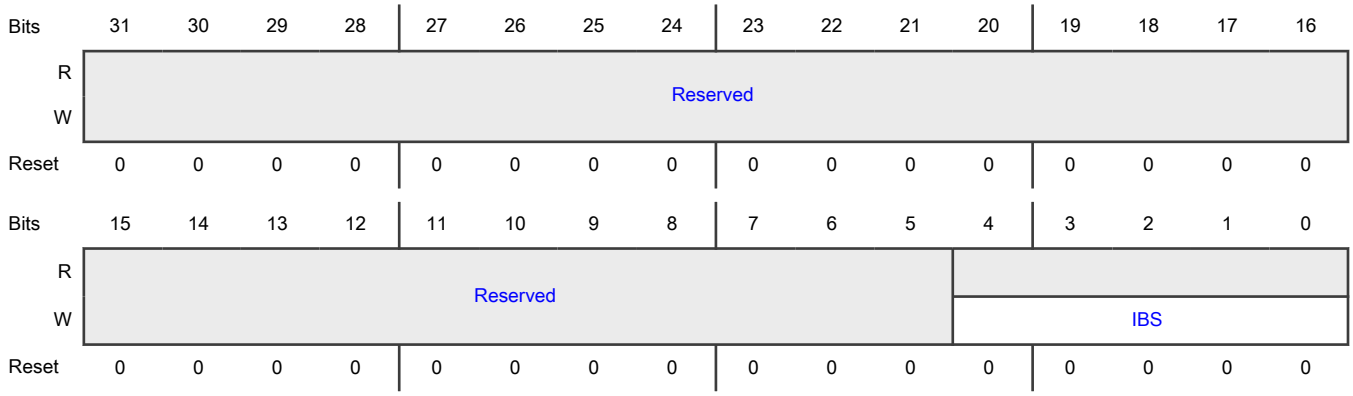
Offset

Register	Offset
MSIIR2	10h

Function

Writing the Interrupt Bit Select (IBS) field of the MSIIR2 register generates an interrupt to the processor and sets the corresponding bit of the MSIR2 registers.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 IBS	Interrupt Bit Select. Selects the bit to set in the MSIR2 register. 00000 Set field SH0 (bit 0) 00001 Set field SH1 (bit 1) 00010 Set field SH2 (bit 2) -- 11111 Set field SH31 (bit 31)

54.4.7 Message Signaled Interrupt Register 2 (MSIR2)

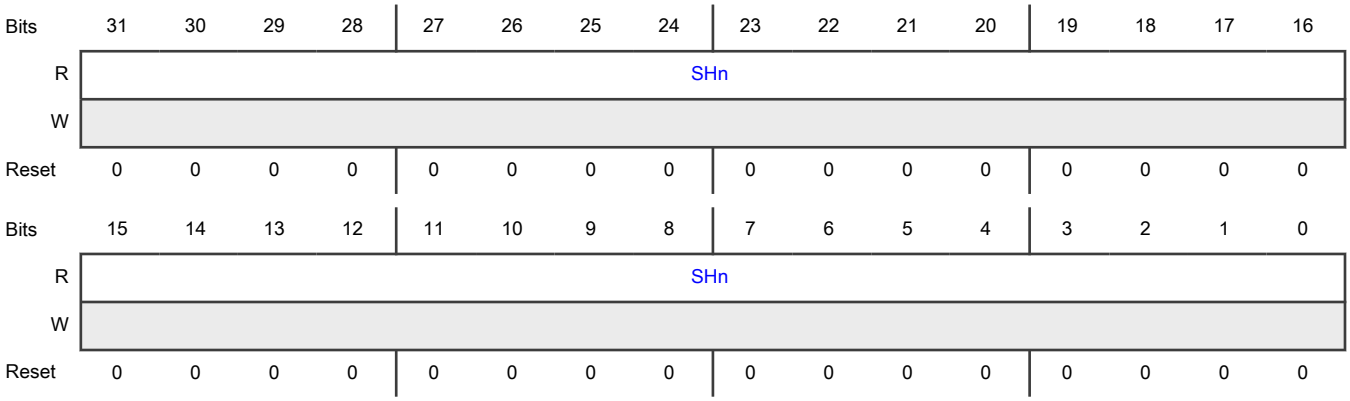
Offset

Register	Offset
MSIR2	14h

Function

This shared message signaled interrupt register indicates which of the up to 32 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; writes to the register have no effect.

Diagram



Fields

Field	Function
31-0 SHn	Message sharer n has a pending interrupt.

54.5 Functional Description

MSGINTR converts a memory write transaction to an interrupt event. It has 3 sets of MSIIR/MSIR registers.

A write to the MSIIRn[IBS] will set the bit (indexed by IBS) in the MSIRn register and trigger the interrupt associated with MSIRn. A read to the MSIIRn will return 0s and has no effect.

A read to MSIRn will clear all bits in the MSIRn register and hence de-assert the associated interrupt.

Chapter 55
EtherCAT Controller (eCAT)

55.1 Chip-specific eCAT Information

Table 781. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

NOTE

Only two ethernet ports are supported in this chip.

55.2 Overview

An EtherCAT® Subordinate Controller (ESC) takes care of the EtherCAT communication as an interface between the EtherCAT fieldbus and slave application.

55.2.1 Block diagram

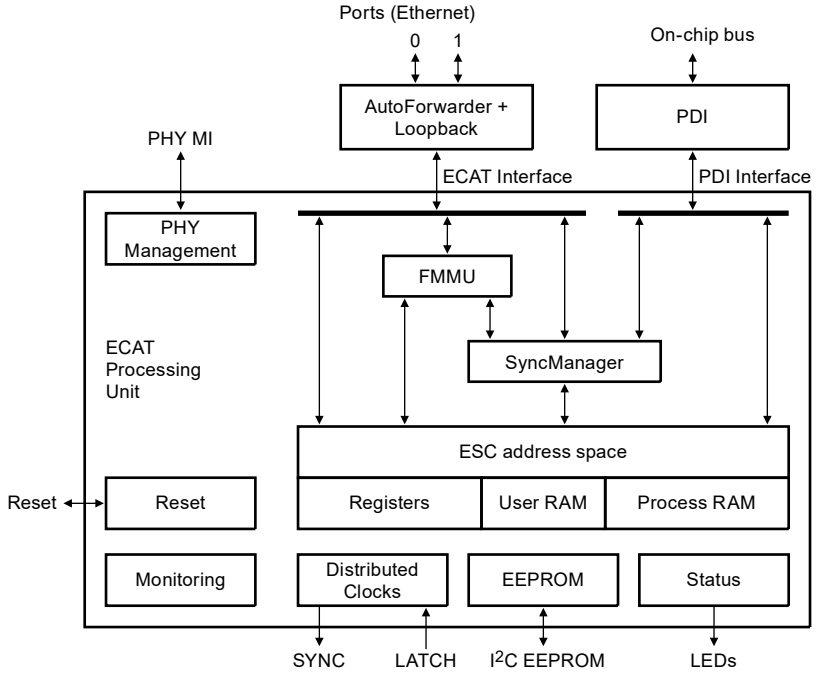


Figure 370. ESC block diagram

The block diagram above shows the general functionality of EtherCAT.

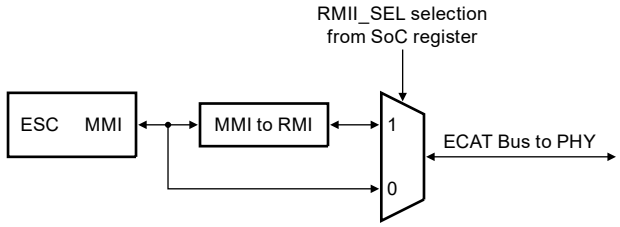


Figure 371. ESC wrapper

55.2.2 Features

This sections lists the features of the EtherCAT block.

- 2 EtherCAT Ports
- 8 FMMUs
- 8 Sync Managers
- Process Data RAM 8Kbyte
- User RAM 128Byte
- 64 bits Distributed clocks
- Process data interface is deactivated (no PDI)

55.3 Functional description

EtherCAT supports the following functions:

EtherCAT protocol

- Ethernet frames with Ether type 88A4h
- EtherCAT frames encapsulated in UDP/IP
- EtherCAT frames with VLAN Tag
- Normal Ethernet frames
- **Addressing modes**
  - Device addressing
  - Logical addressing
- **Working counter**
  - Counting the number of read/write from/to the device.
- **EtherCAT command type**
  - Processing the command that master requests slaves to address each addressing mode

Loop control

- Loop control and loop state in eCAT

Shadow buffer

- Shadow buffers function when register is written using frame access.

Circulating frames

- Prevention of circulating frames during the failure

FIFO size reduction

- RX FIFO size reduction because of reduction of propagation delay

Ethernet physical layer

- MII
- RMII
- Back-to-Back MII connection
- MII management interface
- Read/write of the PHY register via MII management interface
- PHY address offset
- Automatic TX clock shift compensation

FMMU

- Mapping between logical address and physical address

SyncManager

- Buffer mode
- Mailbox mode
- Watchdog,Interrupt and latch event generation when a buffer is completely and successfully written or read.
- Repeating mailbox communication
- SyncManager deactivation by the PDI

Distributed clocks

- Clock Synchronization considering propagation delay and drift compensation
- Generation of synchronous output signals (SYNC0 and 1 signals): (1)Cyclic mode (2)Single shot mode;(3)Cyclic acknowledge mode;(4)Single shot acknowledge mode
- Precise time stamping of input events (LATCH0 and 1 signals):(1)Single event mode (2)Continuous mode (3)SyncManager event mode (for debugging)
- Generation of synchronous interrupts
- Exclusive control for the SYNC and LATCH signals of the ECAT and PDI
- Communication Timing: (1) Free run (2)Synchronized to output event (3)Synchronized to SYNC signal

EtherCAT state machine

- Control of state machine / Indication of the status and error code

SII EEPROM

- SII EEPROM commands
- SII EEPROM error indication
- SII EEPROM access interface
- EEPROM size selection

Interrupt

- AL event request (PDI interrupt)
- ECAT event request (ECAT interrupt)

Watchdog

- Process data watchdog
- PDI watchdog

LED signals

- RUN LED signal
- ERR LED signal
- STATE LED and STATE_RUN LED signals
- LINK/ACT LED signals
- RUN/ERR LED override

Process data interface (PDI)

- On-chip bus

ESC reset

- ESC reset from the master or PDI

55.3.1 EtherCAT controller function blocks

- **EtherCAT interfaces (Ethernet)**

  The EtherCAT interfaces or ports connect the ESC to other EtherCAT slaves and the master. The MAC layer is integral part of the ESC. The physical layer may be Ethernet . For Ethernet ports, external Ethernet PHYs connect to the MII/RMII ports of the ESC. Transmission speed for EtherCAT is fixed to 100 Mbit/s with Full Duplex communication. Link state and communication status are reported to the Monitoring device.

  EtherCAT slaves support 2 ports, the logical ports are numbered 0-1, formerly they were denoted by A-B

- **EtherCAT processing unit**

  The EtherCAT Processing Unit (EPU) receives, analyses, and processes the EtherCAT data stream. It is logically located between port 0 and port 1. The main purpose of the EtherCAT Processing unit is to enable and coordinate access to the internal registers and the memory space of the ESC, which can be addressed both from the EtherCAT master and from the local application via the PDI. Data exchange between master and slave application is comparable to a dual-ported memory (process memory), enhanced by special functions e.g. for consistency checking (SyncManager) and data mapping (FMMU). The EtherCAT Processing Units contains the main function blocks of EtherCAT slaves besides Auto-Forwarding, Loop-back function, and PDI.

- **Auto-forwarder**

  The Auto-Forwarder receives the Ethernet frames, performs frame checking and forwards it to the Loop-back function. Time stamps of received frames are generated by the Auto-Forwarder.

- **Loop-back function**

  The Loop-back function forwards Ethernet frames to the next logical port if there is either no link at a port, or if the port is not available, or if the loop is closed for that port. The Loop-back function of port 0 forwards the frames to the EtherCAT Processing Unit. The loop settings can be controlled by the EtherCAT master.

- **FMMU**

  Fieldbus Memory Management Units are used for bitwise mapping of logical addresses to physical addresses of the ESC.

- **SyncManager**

  SyncManagers are responsible for consistent data exchange and mailbox communication between EtherCAT master and slaves. The communication direction can be configured for each SyncManager. Read or write transactions may generate events for the EtherCAT master and an attached Controller respectively. The SyncManagers are responsible for the main difference between and ESC and a dual-ported memory, because they map addresses to different buffers and block accesses depending on the SyncManager state. This is also a fundamental reason for bandwidth restrictions of the PDI.

- **Monitoring**

  The Monitoring unit contains error counters and watchdogs. The watchdogs are used for observing communication and returning to a safe state in case of an error. Error counters are used for error detection and analysis.

- **PHY management**

  The PHY Management unit communicates with Ethernet PHYs via the MII management interface. This is either used by the master or by the slave. The MII management interface is used by the ESC itself for optionally restarting auto negotiation after receive errors with the enhanced link detection mechanism, and for the optional MI link detection and configuration feature.

- **Distributed clock**

  Distributed Clocks (DC) allow for precisely synchronized generation of output signals and input sampling, as well as time stamp generation of events. The synchronization may span the entire EtherCAT network.

- **Memory**

  An EtherCAT slave can have an address space of up to 64Kbyte. The first block of 4 Kbyte (0x0000-0x0FFF) is used for registers and user memory. The memory space from address 0x1000 onwards is used as the process memory (up to 60 Kbyte). The size of process memory depends on the device. The ESC address range is directly addressable by the EtherCAT master and an attached Controller.

- **Process Data Interface (PDI) or Application interface:**

  — Process data interface is deactivated (no PDI)

- **SII EEPROM**

  One non-volatile memory is needed for EtherCAT Slave Information (ESI) storage, typically an I2C EEPROM.

- **Status/LEDs**

The Status block provides ESC and application status information. It controls external LEDs like the application RUN LED/ERR LED and port Link/Activity LEDs.

55.3.2 EtherCAT protocol

EtherCAT uses standard IEEE 802.3 Ethernet frames, thus a standard network controller can be used and no special hardware is required on master side.

EtherCAT has a reserved EtherType of 0x88A4 that distinguishes it from other Ethernet frames. Thus, EtherCAT can run in parallel to other Ethernet protocols.

EtherCAT does not require the IP protocol, however it can be encapsulated in IP/UDP. The EtherCAT Subordinate Controller processes the frame in hardware. Thus, communication performance is independent from processor power.

An EtherCAT frame is subdivided into the EtherCAT frame header followed by one or more EtherCAT datagrams. At least one EtherCAT datagram has to be in the frame. Only EtherCAT frames with Type 1 in the EtherCAT Header are currently processed by the EtherCAT Controllers. The ESC also support IEEE802.1Q VLAN Tags, although the VLAN Tag contents are not evaluated by the ESC.

If the Ethernet frame size falls below 64 byte, padding bytes have to be added until this size is reached. Otherwise the EtherCAT frame is exactly as large as the sum of all EtherCAT datagrams plus EtherCAT frame header.

The figure below shows how an ethernet frame containing EtherCAT data is assembled.

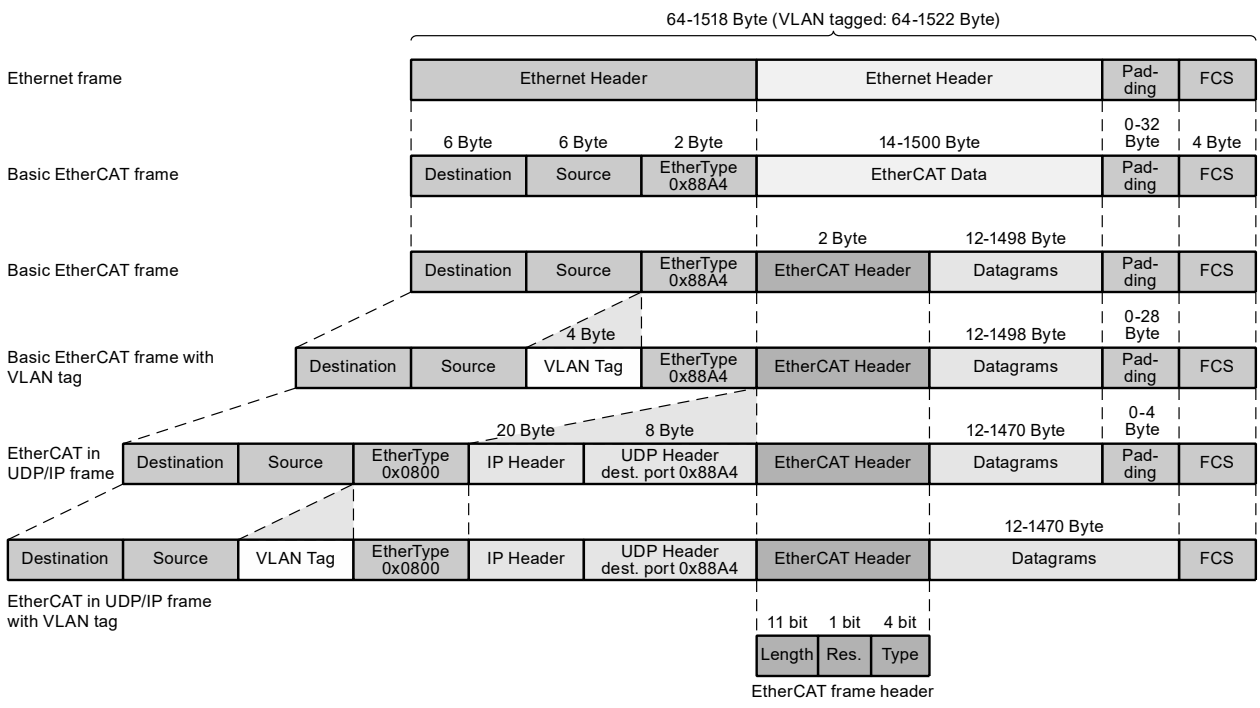


Figure 372. EtherCAT frame with EtherCAT data

Table 782. EtherCAT frame header

Field	Data Type	Description
Length	11 bits	Length of the EtherCAT datagrams (excl. FCS)
Reserved	1 bit	Reserved, 0

Table continues on the next page...

Table 782. EtherCAT frame header (continued)

Field	Data Type	Description
Type	4 bit	Protocol type. Only EtherCAT commands (Type = 0x1) are supported by ESC.

55.3.2.1 EtherCAT datagram

The below figure shows the structure of an EtherCAT datagram.

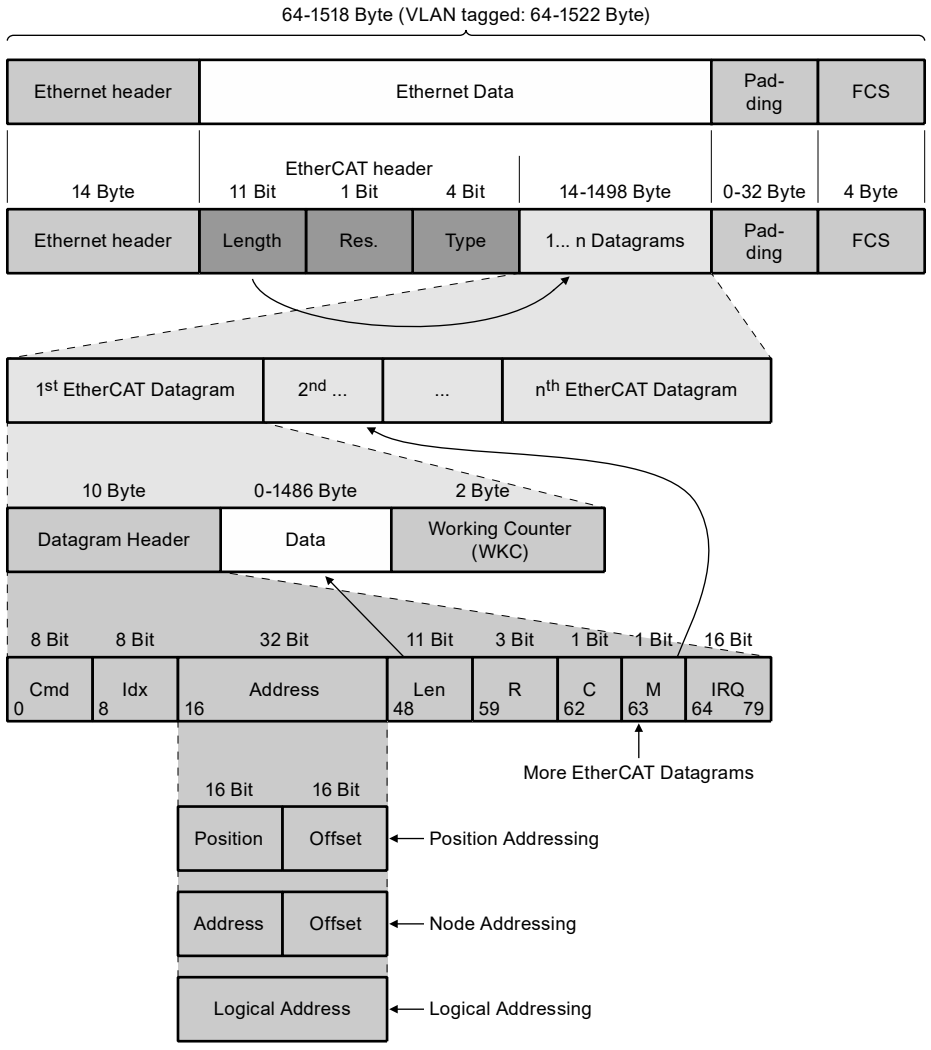


Figure 373. EtherCAT datagram

Table 783. EtherCAT frame header

Field	Data Type	Description
Cmd	BYTE	EtherCAT Command Type
Idx	BYTE	The index is a numeric identifier used by the master for identification of

Table continues on the next page...

Table 783. EtherCAT frame header (continued)

Field	Data Type	Description
		duplicates/lost datagrams. It shall not be changed by EtherCAT slaves
Address	BYTE[4]	Address (Auto Increment, Configured Station Address, or Logical Address, see 2.3)
Len	11 bits	Length of the following data within this datagram
R	3 bits	Reserved, 0
C	1 bit	Circulating frame (see 3.5): • 0: Frame is not circulating • 1: Frame has circulated once
M	1 bit	More EtherCAT datagrams • 0: Last EtherCAT datagram • 1: More EtherCAT datagrams will follow
IRQ	WORD	EtherCAT Event Request registers of all slaves combined with a logical OR
Data	BYTE[n]	Read/Write Data
WKC	WORD	Working Counter

55.3.2.2 EtherCAT addressing modes

Two addressing modes of EtherCAT devices are supported within one segment: device addressing and logical addressing. Three device addressing modes are available: auto increment addressing, configured station address, and broadcast. EtherCAT devices can have up to two configured station addresses, one is assigned by the master (Configured Station Address), the other one is stored in the SII EEPROM and can be changed by the slave application (Configured Station Alias address). The EEPROM setting for the Configured Station Alias address is only taken over at the first EEPROM loading after power-on or reset

Table 784. EtherCAT frame header

Mode	Field	Data Type	Description
Auto Increment Address	Position	WORD	Each slave increments Position. Slave is addressed if Position = 0.
	Offset	WORD	Local register or memory address of the ESC
Configured Station Address	Address	WORD	Slave is addressed if Address matches Configured Station Address or Configured Station Alias (if enabled).
	Offset	WORD	Local register or memory address of the ESC

Table continues on the next page...

Table 784. EtherCAT frame header (continued)

Mode	Field	Data Type	Description
Broadcast	Position	WORD	Each slave increments Position (not used for addressing)
	Offset	WORD	Local register or memory address of the ESC
Logical Address	Address	DWORD	Logical Address (configured by FMMUs) Slave is addressed if FMMU configuration matches Address.

55.3.2.2.1 Device addressing

The device can be addressed via Device Position Address (Auto Increment address), by Node Address (Configured Station Address/Configured Station Alias), or by a Broadcast.

- Position Address / Auto Increment Address: The datagram holds the position address of the addressed slave as a negative value. Each slave increments the address. The slave which reads the address equal zero is addressed and will execute the appropriate command at receive. Position Addressing should only be used during start-up of the EtherCAT system to scan the fieldbus and later only occasionally to detect newly attached slaves. Using Position addressing is problematic if loops are closed temporarily due to hot connecting or link problems. Position addresses are shifted in this case, and e.g., a mapping of error register values to devices becomes impossible, thus the faulty link cannot be localized
- Node Address / Configured Station Address and Configured Station Alias: The configured Station Address is assigned by the master during start up and cannot be changed by the EtherCAT slave. The Configured Station Alias address is stored in the SII EEPROM and can be changed by the EtherCAT slave. The Configured Station Alias has to be enabled by the master. The appropriate command action will be executed if Node Address matches with either Configured Station Address or Configured Station Alias. Node addressing is typically used for register access to individual and already identified devices.
- Broadcast: Each EtherCAT slave is addressed. Broadcast addressing is used e.g. for initialization of all slaves and for checking the status of all slaves if they are expected to be identical.

Each slave device has a 16 bit local address space (address range 0x0000:0x0FFF is dedicated for EtherCAT registers, address range 0x1000:0xFFFF is used as process memory) which is addressed via the Offset field of the EtherCAT datagram. The process memory address space is used for application communication (e.g. mailbox access).

55.3.2.2.2 Logical addressing

All devices read from and write to the same logical 4 Gbyte address space (32 bit address field within the EtherCAT datagram). A slave uses a mapping unit (FMMU, Fieldbus Memory Management Unit) to map data from the logical process data image to its local address space. During start up the master configures the FMMUs of each slave. The slave knows which parts of the logical process data image have to be mapped to which local address space using the configuration information of the FMMUs. Logical Addressing supports bit wise mapping. Logical Addressing is a powerful mechanism to reduce the overhead of process data communication, thus it is typically used for accessing process data.

55.3.2.3 Working counter

Every EtherCAT datagram ends with a 16 Bit Working Counter (WKC). The Working Counter counts the number of devices that were successfully addressed by this EtherCAT datagram. Successfully means that the ESC is addressed and the addressed memory is accessible (e.g., protected SyncManager buffer). EtherCAT Controllers increment the Working Counter in hardware. Each datagram should have an expected Working Counter value calculated by the master. The master can check the valid processing of EtherCAT datagrams by comparing the Working Counter with the expected value.

The Working Counter is increased if at least one byte/one bit of the whole multi-byte datagram was successfully read and/or written. For a multi-byte datagram, you cannot tell from the Working Counter value if all or only one byte was successfully read and/or written. This allows reading separated register areas using a single datagram by ignoring unused bytes.

The Read-Multiple-Write commands ARMW and FRMW are either treated like a read command or like a write command, depending on the address match.

Table 785. Working counter increment

Command	Data Type	Increment
Read command	No success	no change
	Successful read	+1
Write command	No success	no change
	Successful write	+1
ReadWrite command	No success	no change
	Successful read	+1
	Successful write	+2
	Successful read and write	+3

55.3.2.4 EtherCAT command types

All supported EtherCAT Command types are listed in the below table. For ReadWrite operations, the Read operation is performed before the Write operation

Table 786. EtherCAT frame header

CMD	Abbr.	Name	Description
0	NOP	No Operation	Slave ignores command
1	APRD	Auto Increment Read	Slave increments address. Slave puts read data into the EtherCAT datagram if received address is zero.
2	APWR	Auto Increment Write	Slave increments address. Slave writes data into memory location if received address is zero.
3	APRW	Auto Increment Read Write	Slave increments address. Slave puts read data into the EtherCAT datagram and writes the data into the same memory location if received address is zero.
4	FPRD	Configured Address Read	Slave puts read data into the EtherCAT datagram if address matches with one of its configured addresses

Table continues on the next page...

Table 786. EtherCAT frame header (continued)

CMD	Abbr.	Name	Description
5	FPWR	Configured Address Write	Slave writes data into memory location if address matches with one of its configured addresses
6	FPRW	Configured Address Read Write	Slave puts read data into the EtherCAT datagram and writes data into the same memory location if address matches with one of its configured addresses
7	BRD	Broadcast Read	All slaves put logical OR of data of the memory area and data of the EtherCAT datagram into the EtherCAT datagram. All slaves increment position field.
8	BWR	Broadcast Write	All slaves write data into memory location. All slaves increment position field.
9	BRW	Broadcast Read Write	All slaves put logical OR of data of the memory area and data of the EtherCAT datagram into the EtherCAT datagram, and write data into memory location. BRW is typically not used. All slaves increment position field.
10	LRD	Logical Memory Read	Slave puts read data into the EtherCAT datagram if received address matches with one of the configured FMMU areas for reading.
11	LWR	Logical Memory Write	Slaves writes data to into memory location if received address matches with one of the configured FMMU areas for writing.
12	LRW	Logical Memory Read Write	Slave puts read data into the EtherCAT datagram if received address matches with one of the configured FMMU areas for reading. Slaves writes data to into memory location if received address matches with one of

Table continues on the next page...

Table 786. EtherCAT frame header (continued)

CMD	Abbr.	Name	Description
			the configured FMMU areas for writing.
13	ARMW	Auto Increment Read Multiple Write	Slave increments address. Slave puts read data into the EtherCAT datagram if received address is zero, otherwise slave writes the data into memory location.
14	FRMW	Configured Read Multiple Write	Slave puts read data into the EtherCAT datagram if address matches with one of its configured addresses, otherwise slave writes the data into memory location.
15-255	-	reserved	-

55.3.2.5 UDP/IP

The following header fields are evaluated by an ESC to detect an EtherCAT frame encapsulated in UDP/IP:

All other fields if the IP and UDP header are not evaluated, and the UDP checksum is not checked. Since EtherCAT frames are processed on the fly, the UDP checksum cannot be updated by an ESC when the frame content is modified. Instead, the ESC clears the UDP checksum for any EtherCAT frame (regardless of DL Control register 0x0100[0]), which indicates that the checksum is not used. The UDP checksum is forwarded without modification for non-EtherCAT frames if DL Control register 0x0100[0]=0.

Table 787. EtherCAT UDP/IP encapsulation

Field	Expected value for EtherCAT
EtherType	0x800(IP)
IP version	4
IP Header length	5
IP Protocol	0x11(UDP)
UDP destination port	0x88A4

55.3.3 Frame processing

The IP Core slave controllers only support Direct Mode addressing: neither a MAC address nor an IP address is assigned to the ESC, they process EtherCAT frames with any MAC or IP address.

It is not possible to use unmanaged switches between these ESCs or between master and the first slave, because source and destination MAC addresses are not evaluated or exchanged by the ESCs. Only the source MAC address is modified when using the default settings, so outgoing and incoming frames can be distinguished by the master.

NOTE

Attaching an ESC directly to an office network will result in network flooding, since the ESC will reflect any frame especially broadcast frames back into the network (broadcast storm).

The frames are processed by the ESC on the fly, i.e., they are not stored inside the ESC. Data is read and written as the bits are passing the ESC. The forwarding delay is minimized to achieve fast cycle times. The forwarding delay is defined by the receive FIFO size and the EtherCAT Processing Unit delay. A transmit FIFO is omitted to reduce delay times.

The ESCs support EtherCAT, UDP/IP, and VLAN tags. EtherCAT frames and UDP/IP frames containing EtherCAT datagrams are processed. Frames with VLAN tags are processed by the ESCs, the VLAN settings are ignored and the VLAN tag is not modified. The source MAC address is changed for every frame passing the EtherCAT Processing Unit (SOURCE_MAC[1] is set to 1 locally administered address). This helps to distinguish between frames transmitted by the master and frames received by the master

55.3.3.1 Loop control and loop state

Each port of an ESC can be in one of two states: open or closed. If a port is open, frames are transmitted to other ESCs at this port, and frames from other ESCs are received. A port which is closed will not exchange frames with other ESCs, instead, the frames are forwarded internally to the next logical port, until an open port is reached. The loop state of each port can be controlled by the master (ESC DL Control register 0x0100). The ESCs supports four loop control settings, two manual configurations, and two automatic modes:

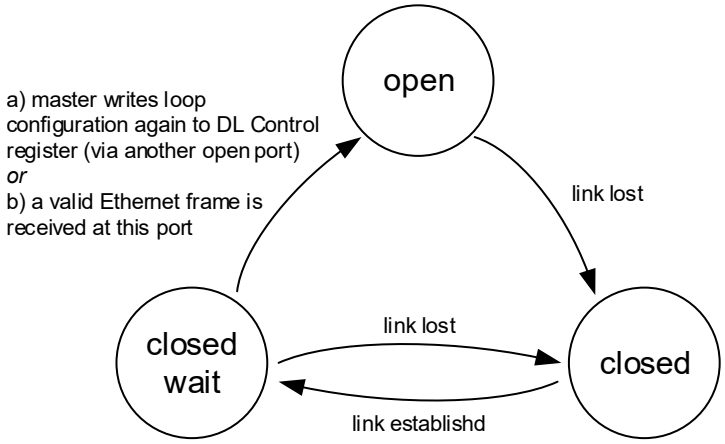


Figure 374. Auto close loop state transitions

- Manual open: The port is open regardless of the link state. If there is no link, outgoing frames will be lost.
- Manual close: The port is closed regardless of the link state. No frames will be sent out or received at this port, even if there is a link with incoming frames.
- Auto: The loop state of each port is determined by the link state of the port. The loop is open if there is a link, and it is closed without a link.
- Auto close (manual open): The port is closed depending on the link state, i.e., if the link is lost, the loop will be closed (auto close). If the link is established, the loop will not be automatically opened, instead, it will remain closed (closed wait state). Typically, the port has to be opened by the master explicitly by writing the loop configuration again to the ESC DL Control register 0x0100. This write access has to enter the ESC via a different open port. There is an additional fallback option for opening the port: if a valid Ethernet frame is received from the external link at the closed port in Auto close mode, it will also be opened after the CRC is received correctly. The content of the frame is not evaluated.

A port is considered open if the port is available, i.e., it is enabled in the configuration, and one of the following conditions is met:

- The loop setting in the DL Control register is Auto and there is an active link at the port.
- The loop setting in the DL Control register is Auto close and there is an active link at the port and the DL Control register was written again after the link was established
- The loop setting in the DL Control register is Auto close and there is an active link at the port and a valid frame was received at this port after the link was established.
- The loop setting in the DL control register is Always open

A port is considered closed if one of the following conditions is met:

- The port is not available or not enabled in the configuration.
- The loop setting in the DL Control register is Auto and there is no active link at the port.
- The loop setting in the DL Control register is Auto close and there is no active link at the port or the DL Control register was not written again after the link was established
- The loop setting in the DL Control register is Always closed

Table 788. Registers for loop control and loop/link status

Register Address	Name	Description
0x0100[15:8]	ESC DL Control	Loop control/loop setting
0x0110[15:4]	ESC DL Status	Loop and link status
0x0518:0x051B	PHY Port status	PHY Management link status

55.3.3.2 Frame Processing Order

The frame processing order of EtherCAT Subordinate Controllers depends on the number of ports (logical port numbers are used):

Table 789. Frame Processing Order

Number of Ports	Frame Processing Order
1	0--> EtherCAT Processing Unit -->0
2	0--> EtherCAT Processing Unit -->1 / 1-->0

The direction through an ESC including the EtherCAT Processing Unit is called "processing" direction, other directions without passing the EtherCAT Processing Unit are called "forwarding" direction. Ports which are not implemented behave similar to closed ports, the frame is forwarded to the next port. Frame processing in general is shown in the below figure:

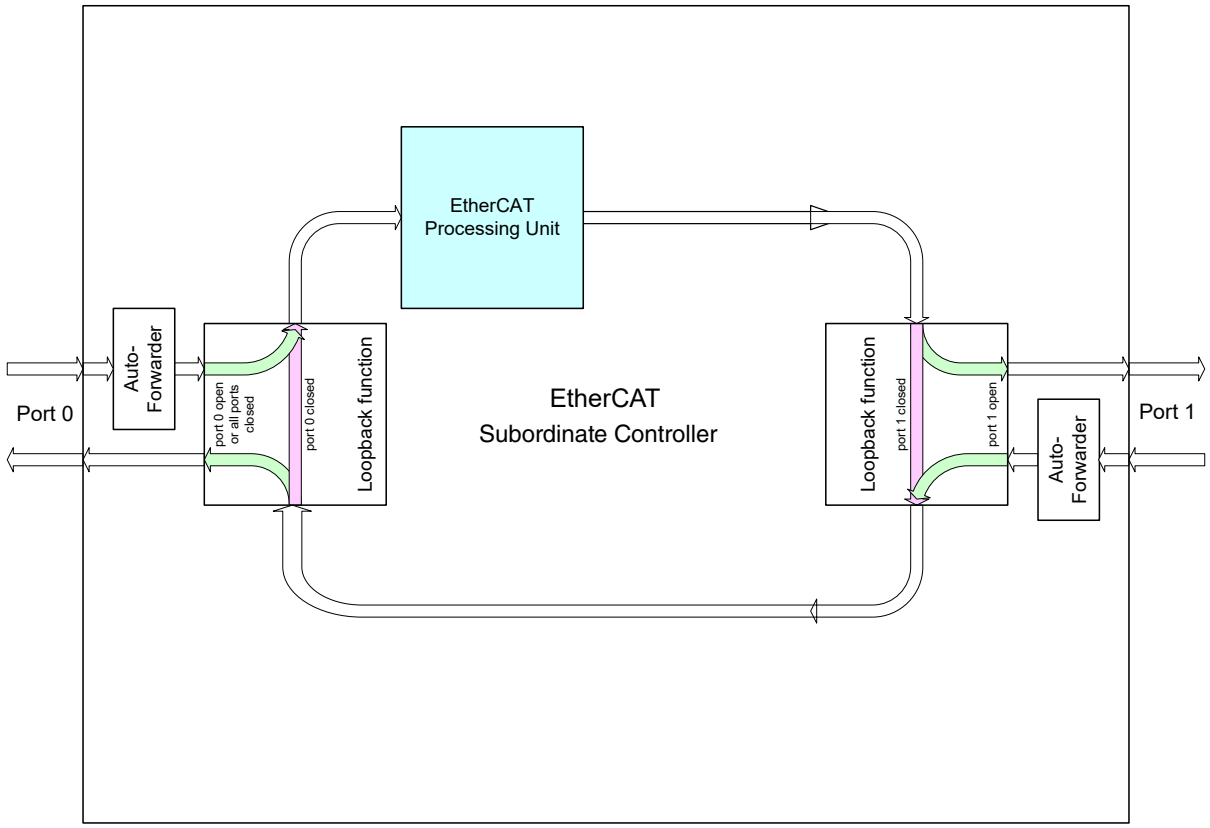


Figure 375. Frame Processing

Example Port Configuration with Ports 0, 1

If there are only ports 0, 1, a frame received at port 0 goes via the Auto-Forwarder and the Loopback function to the EtherCAT Processing Unit which processes it. Then, the frame is sent to port 1. If port 1 is closed, the frame is forwarded by the Loopback function . If port 1 is open, the frame is sent out at port 1. When the frame comes back into port 1, it is handled by the Auto-Forwarder . Then it is handled by the Loopback function and sent out at port 0 - back to the master.

55.3.3.3 Permanent ports and bridge port

The EtherCAT ports of an ESC are typically permanent ports, which are directly available after Power- On. Permanent ports are initially configured for Auto mode, i.e., they are opened after the link is established. This Bridge port becomes available if the EEPROM is loaded successfully, and it is closed initially, i.e., it has to be opened (or set to Auto mode) explicitly by the EtherCAT master.

55.3.3.4 Shadow buffer for register write operations

The ESCs have shadow buffers for write operations to registers (0x0000 to 0x0F7F). During a frame, write data is stored in the shadow buffers. If the frame is received correctly, the values of the shadow buffers are transferred into the effective registers. Otherwise, the values of the shadow buffers are not taken over. As a consequence of this behavior, registers take their new value shortly after the FCS of an EtherCAT frame is received. SyncManagers also change the buffers after the frame was received correctly.

User and Process Memory do not have shadow buffers. Accesses to these areas are taking effect directly. If a SyncManager is configured to User Memory or Process Memory, write data will be placed in the memory, but the buffer will not change in case of an error.

55.3.3.5 Circulating frames

The ESCs incorporate a mechanism for prevention of circulating frames. This mechanism is very important for proper watchdog functionality. This is an example network with a link failure between slave 1 and slave 2:

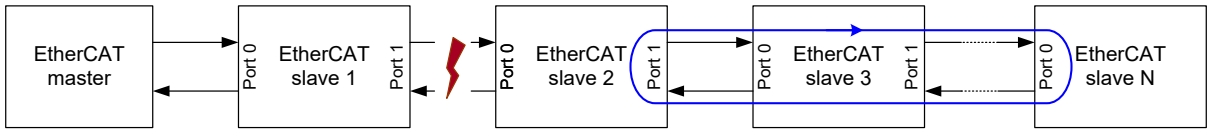


Figure 376. Circulating frames

Both slave 1 and slave 2 detect the link failure and close their ports (port 1 at slave 1 and port 0 at slave 2). A frame currently traveling through the ring at the right side of slave 2 might start circulating. If such a frame contains output data, it might trigger the built-in watchdog of the ESCs, so the watchdog never expires, although the EtherCAT master cannot update the outputs anymore. To prevent this, a slave with loop closed at port 0 and loop control for port 0 set to Auto or Auto close (ESC DL Control register 0x0100) will do the following inside the EtherCAT Processing Unit:

- If the Circulating bit of the EtherCAT datagram is 0, set the Circulating bit to 1
- If the Circulating bit is 1, do not process the frame and destroy it

The result is that circulating frames are detected and destroyed. Since the ESCs do not store the frames for processing, a fragment of the frame will still circulate triggering the Link/Activity LEDs. Nevertheless, the fragment is not processed.

55.3.3.5.1 Unconnected port 0

Port 0 must not be left intentionally unconnected (slave hardware or topology) because of the circulating frame prevention. All frames will be dropped after they have passed an automatically closed Port 0 for the second time, and this can prohibit any EtherCAT communication.

In redundancy operation, only one Port 0 is automatically closed, so the communication remains active.

55.3.3.6 Non-EtherCAT protocols

If non-EtherCAT protocols are used, the forwarding rule in the ESC DL Control register (0x0100[0]) has to be set to forward non-EtherCAT protocols. Otherwise they are destroyed by the eCAT.

55.3.3.7 Special functions of port 0

Port 0 of each EtherCAT is characterized by some special functions in contrast to port 1:

- Port 0 leads to the master, i.e., port 0 is the upstream port.
- The link state of Port 0 influences the Circulating Frame bit, and frames are dropped at port 0 if the bit is set and the link is automatically closed.
- Port 0 loop state is open if all ports are closed (either automatically or manually).

55.3.4 Physical layer common features

EtherCAT supports two basic types of physical layers, Ethernet . It requires 100 Mbit/s links with full duplex communication.

The Ethernet-based physical layer uses standard Ethernet physical layer devices (PHY) according to IEEE 802.3. It supports different flavors like cable (100BaseTX), optical (100BaseFX), and EtherCAT P (100BaseTX-like with power supply).

55.3.4.1 Link status

The link status of each port is available in the ESC DL Status register (0x0110:0x0111), most important are the "Communication established" bits 15, 13, 11, and 9. Additional link information is available in the PHY Port status register (0x0518:0x051B) if MI link detection and configuration is used. All other status bits are mainly for debugging purposes. The link status bits are described in the following table

If all ports are closed (either manually or automatically, e.g., because no port has a communication link), port 0 is automatically opened as the recovery port. Reading and writing via this port is possible, although the DL status register reflects the correct status. This can be used to correct erroneous DL control register settings or to fix LINK_MII polarity configuration.

55.3.4.2 Selecting standard/enhanced link detection

Some ESCs distinguish between standard and enhanced link detection. Enhanced link detection provides additional security mechanisms regarding link establishment and surveillance. Using enhanced link detection is recommended for Ethernet PHY ports. Some ESCs only support global Enhanced Link Detection configuration for all ports, some support port-wise configuration.

After power-on, enhanced link detection is enabled by default. It is disabled or remains enabled after the SII EEPROM is loaded according to the EEPROM setting (register 0x0141). An invalid EEPROM content will also disable enhanced link detection.

The EEPROM setting for enhanced Link detection is only taken over at the first EEPROM loading after power-on or reset. Changing the EEPROM and manually reloading it will not affect the enhanced link detection enable status (register 0x0110[2]), even if the EEPROM could not be read initially. Registers used for Enhanced link detection are listed in the below table.

Table 790. Registers for enhanced link detection

Register Address	Name	Description
0x0141[1]	ESC Configuration	Enable/disable Enhanced link detection for all ports
0x0141[7:4]	ESC Configuration	Enable/disable Enhanced link detection port-wise
0x0110[2]	ESC DL Status	Enhanced link detection status

NOTE

Some of these register bits are set via SII EEPROM/IP Core configuration.

55.3.4.3 FIFO size reduction

The ESCs incorporate a receive FIFO (RX FIFO) for decoupling receive clock and processing clock. The FIFO size is programmable by the EtherCAT master (ESC DL Control register 0x0100). Some ESCs support a default value for the FIFO size loaded from the SII EEPROM. The FIFO size values determine a reduction of the FIFO size, the FIFO cannot be disabled completely. The FIFO size can be reduced considering these three factors:

- Accuracy of the receiver's clock source
- Accuracy of the sender's clock source
- Maximum frame size

The default FIFO size is sufficient for maximum Ethernet Frames and default Ethernet clock source accuracy (100 ppm). If the clock accuracy is 25 ppm or better, the FIFO size can be reduced to the minimum. If the FIFO size was accidentally reduced too much, a short 64 Byte frame should be sent for resetting the FIFO size to the default value, since a smaller frame is not utilizing the FIFO as much as a larger frame.

The FIFO size can be reduced to minimum if both sender and receiver have 25 ppm accuracy of their clock sources, even with maximum frame size.

Since 25 ppm clock accuracy can typically not be guaranteed for the entire life-time of a clock source, the actual clock deviation has to be measured on a regular basis for FIFO size reduction. If a slave does not support Distributed Clocks or the actual deviation is larger than 25 ppm, the FIFO size of all neighbors and the slave itself cannot be reduced. The actual deviation can be measured using Distributed Clocks:

- Compare DC Receive Times over a period of time for slaves which only support DC Receive Times. Do not use this method if both slaves which are compared support DC Time Loop, since the measured deviation will approximate zero if the DC control loop has settled, but the actual deviation determining the FIFO size might be larger than 25 ppm.
- Compare calculated deviation from register Speed Counter Diff (0x0932:0x0933) for adjacent slaves with DC Time Loop support after the DC control loop has settled (i.e., System Time Difference 0x092C:0x092F is at its minimum).

NOTE

Be careful with FIFO size reduction at the first slave if off-the-shelf network interface cards without 25 ppm accuracy are used by master.

Table 791. Registers for FIFO size reduction

Register Address	Name	Description
0x0100[18:16]	ESC DL Control	Current FIFO size setting

55.3.5 Ethernet physical layer

ESCs with Ethernet Physical Layer support use the MII interface, some do also support the RMII interface. Since RMII PHY include FIFOs, they increase the forwarding delay of an EtherCAT slave device as well as the jitter. MII is recommended due to these reasons.

55.3.5.1 PHY reset and link partner notification/loop closing

The main principle of EtherCAT operation in case of link errors is disabling unreliable links by closing loops. This is automatically performed by the ESCs. The ESCs rely on the LINK_MII signal from the PHYs for detecting the link state.

It is crucial that a PHY does not establish a link while the ESC is not operating. Otherwise, the communication partner would also detect a physical link, causing it to open the communication link. Subsequently, all frames will get lost because the ESC is not operating.

So at least the following requirements have to be fulfilled, otherwise frames will be lost:

- ESC in reset state --> PHY disabled

The recommended solution for this issue is to enable the PHY together with the ESC by using the ESCs global reset signal for the PHY, too. If the ESC has individual PHY reset outputs, they should be used instead. This solution prevents the PHYs from establishing a link while the ESCs are not operating. The individual PHY reset signals of an ESC have this functionality:

- Keep the PHY in reset while the ESC is in Reset
- If an FX link is used, and Enhanced link detection is active, a reset for the PHY and the transceiver is applied when too many RX_ERR are received
- The PHY reset is extended to a certain time

55.3.6 MII interface

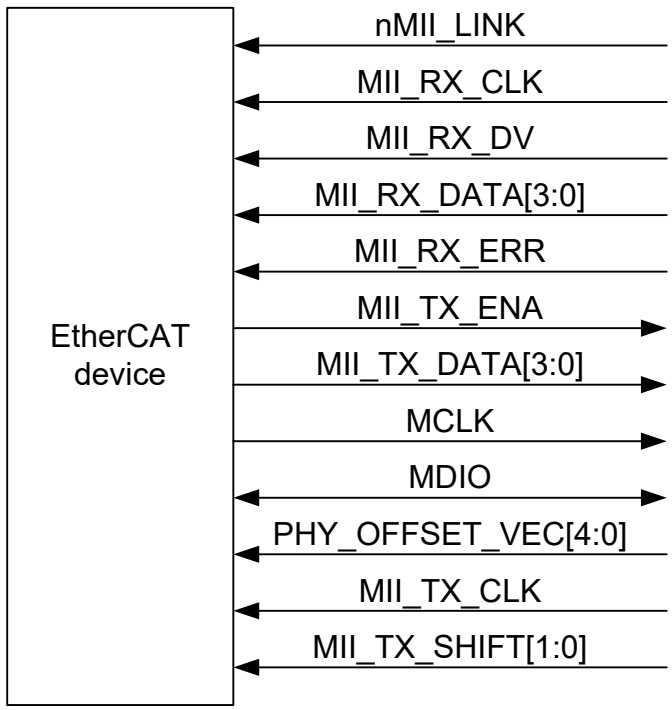


Figure 377. ECAT interrupt masking

Table 792. MII interface signals

Signal	Direction at PHY	Description
nMII_LINK	IN	Input signal provided by the PHY if a 100Mbit/s(Full duplex) link is established (alias Link_MII)
MII_RX_CLK	IN	Receive Clock
MII_RX_DV	IN	Receive data valid
MII_RX_DATA[3:0]	IN	Receive data (alias RXD)
MII_RX_ERR	IN	Receive error (alias RX_ER)
MII_TX_ENA	OUT	Transmit enable (alias TX_EN)
MII_TX_DATA[3:0]	OUT	Transmit data (alias TXD)
MCLK	OUT	Management Interface clock (alias MCLK)
MDIO	BIDIR	Management Interface data (alias MDIO)
PHY_OFFSET_VEC[4:0]	IN	Configuration: PHY address offset
MII_TX_CLK	IN	Transmit Clock for automatic TX Shift compensation
MII_TX_SHIFT[1:0]	IN	Manual TX Shift compensation with additional registers

55.3.6.1 Unused MII port

If an ESC MII interface is not used, LINK_MII has to be tied to a logic value which indicates no link, and RX_CLK, RXD, RX_ER, and especially RX_DV have to be tied to GND. The TX outputs can be left unconnected, unless they are used for ESC configuration.

55.3.7 RMII interface

For more details about the RMII interface, refer to the RMII Specification, available from the RMII consortium.

55.3.7.1 Unused RMII ports

If an ESC RMII interface is not used, LINK_MII has to be tied to a logic value which indicates no link, and RXD, RX_ER, and especially CRS_DV have to be tied to GND. The TX signals can be left unconnected, unless they are used for ESC configuration.

55.3.8 Link detection

The ESCs support different mechanisms for link detection, and some of them are optional. The major goal of the link detection is to detect a link loss very fast. This is required to maintain communication in the rest of the network by appropriately closing communication loops.

The following link status sources are available for Ethernet PHYs:

- LINK_MII signal from an Ethernet PHY
- MI link detection and configuration (MII management interface)
- RX_ERR signal used by Enhanced Link Detection

ECAT support a LINK_MII signal for fast link detection at each Ethernet MII port. EtherCAT IP Core additionally support link detection and configuration via the MII management interface.

The MI link detection is generally not fast enough for EtherCAT when a link loss happens. For many PHYs, the LINK_MII signal is fast enough, but others need Enhanced Link Detection (RX_ERR).

The results of the different link status sources are evaluated inside the ESC according to the DL Control loop setting (register 0x0100):

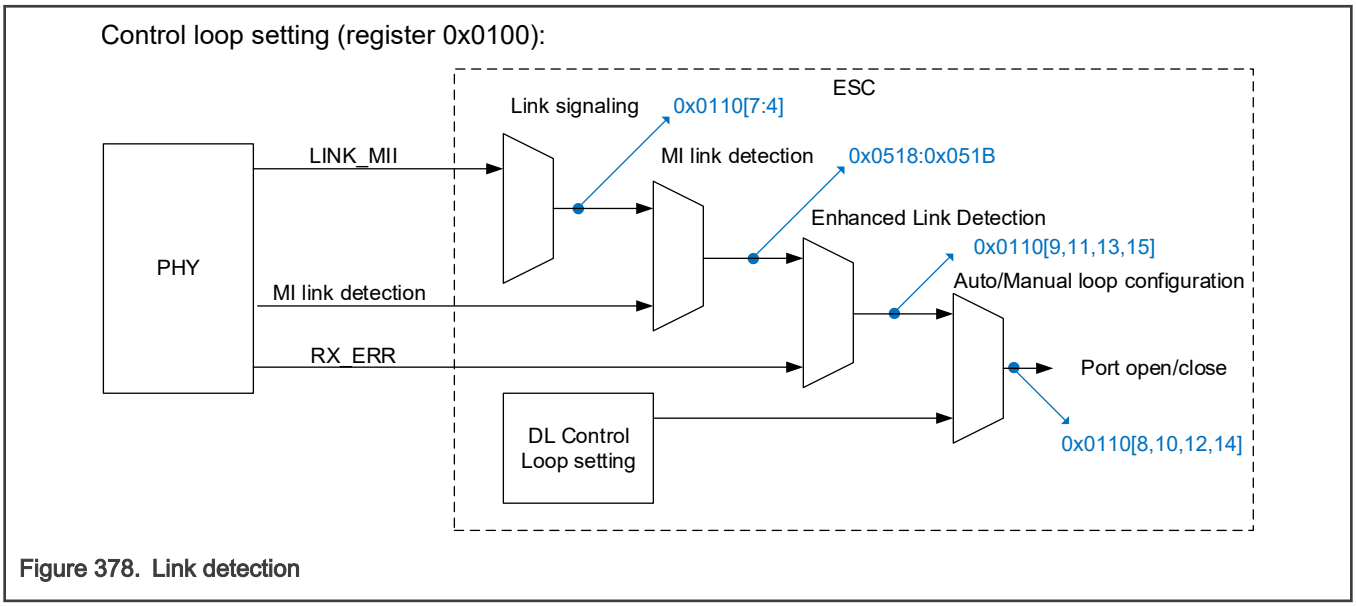


Figure 378. Link detection

The result of the MI link detection is available in the PHY Port Status register 0x0518:0x051B, it is combined with the previous link result.

Enhanced link detection overrides the previous result, if it is enabled and too many RX_ERR have been detected. The result is reflected in the ESC DL Status register (0x0110[15,13,11,9]- Communication established), and it is also used for LINKACT LED.

Finally, the DL Control settings select if the link result is evaluated (and how). The loop state of each port is reflected in the ESC DL Status register (0x0110[14,12,10,8] - Loop open/closed).

Table 793. Registers used for ethernet link detection

Register Address	Name	Description
0x0100:0x0111	ESC DL Control	Link Configuration (manual/auto)
0x0110:0x0111	ESC DL Status	Link Status (LINK_MII)
0x0300:0x0307	RX Error Counter	RX_ERR counter for Enhanced Link Detection
0x0518:0x051B	PHY Port Status	MI Link Detection results

55.3.8.1 Standard link detection

The results of the different sources for Standard MII link detection are combined by a specific logic. The main combinations are shown in the following table.

Table 794. Ethernet link detection combination

LINK_MI	In-band signaling	MI link	MI no link	Link result	Description
no	no	no	no	no link	
yes	no	no	no	link	
no	yes	no	no	link	In-band signaling is used without LINK_MII
yes	yes	no	no	link	LINK_MII is used while In-band signaling is available
yes(one or both)		yes	no	link	MI link detection found valid link, status of LINK_MII/in-band signaling is accepted
don't care		no	yes	no link	MI link detection found invalid link status, either local or at link partner, overrides other status
link down event		yes	no	no link	A link down event (edge) overrides MI link detection
no	no	yes	no	link	Test only, because link-loss reaction time is too slow: MI link is used

Table continues on the next page...

Table 794. Ethernet link detection combination (continued)

LINK_MI	In-band signaling	MI link	MI no link	Link result	Description
					without LINK_MII and without in-band signaling

NOTE

NOTE: "MI link" means that MI link detection successfully read the PHY, and the result is OK. "MI no link" means that MI link detection successfully read the PHY, but the result is not OK. If MI link detection could not properly read from the PHY, the result is undefined.

55.3.8.1.1 LINK_MII signal

The LINK_MII signal used for link detection is typically an LED output signal of the Ethernet PHY. If available, LINK_MII should be connected to a combined signal indicating a 100 Mbit/s Full Duplex link. If such a signal is not available, a signal indicating a 100 Mbit/s link (speed LED) might be used. If only a Link signal is available (link LED), this might be used. Never use (combined) activity signals, e.g., Link/Act LED outputs, because the link state will toggle upon activity.

The main advantage of using a dedicated link signal instead of reading out MII management interface registers is the fast reaction time in case of a link loss. This is crucial for redundancy operation, since only one lost frame is tolerated. The EtherCAT port of an ESC which loses a link has to be closed as fast as possible to maintain EtherCAT communication at the other ports and to reduce the number of lost frames.

55.3.8.1.2 MI link detection and configuration

The EtherCAT IP Core supports link detection and PHY configuration by using the MII management interface. Initially, the PHY configuration is checked and updated if necessary. Afterwards, the link status of each Ethernet port is cyclically polled. PHY accesses of the EtherCAT master are inferred upon request.

The MI Link Configuration mechanism configures the Ethernet PHYs to use Auto negotiation and advertise only 100BASE-TX Full-Duplex connections. ESCs support using Gigabit Ethernet PHYs, which are restricted to 100 Mbit/s links by the MI Link Configuration. ESCs which support FX operation natively are configuring the FX ports to use fixed 100BASE-TX Full-Duplex connections without Auto negotiation

Depending on the physical layer configuration and the supported ESC features (e.g., TX vs. FX), the MI Link Detection will check if the link characteristics fulfill EtherCAT requirements (Auto negotiation, Speed, Duplex, etc.). If all conditions are met, an MI Link is detected.

Since the MI Link Detection does not solely rely on the PHY link status bit (register 1[2]), the local PHY and the remote PHY may indicate a link, but the ESC refuses it because it does not fulfill EtherCAT requirements. The current MI Link Detection state is reflected in the MI Interface registers (PHY Port Status 0x0518:0x051B).

MI Link Detection and Configuration must not be used without link detection via LINK_MII signal , because link loss reaction time would otherwise be to slow for redundancy operation. Enhanced Link Detection might not be a suitable solution in this case if too few RX_ERR are issued to the ESC before the PHY takes down the link

The MI Link Detection and Configuration checks the management communication with Ethernet PHYs. If communication is not possible - e.g. because no PHY is configured for the expected PHY address - the results are ignored. Take care of proper PHY address configuration to prevent erroneous behavior.

NOTE

Proper PHY address settings and PHY address offset configuration is crucial for MI Link Detection and Configuration.

55.3.8.1.3 Enhanced link detection

For Ethernet, the enhanced MII link detection feature is a feature of link error detection and reaction. This has to be distinguished from the actual link detection, which tells the ESC if a physical link is available (i.e., the LINK_MII signal or the MI link detection and configuration mechanism).

Enhanced MII link detection will additionally disconnect a link if at least 32 RX errors (RX_ER) occur in a fixed interval of time (~10 µs). The local loop is closed and the link partner is informed by restarting the Auto-Negotiation mechanism via the MII Management Interface. This informs the link partner of the error condition, and the link partner will close the loop.

The ESC keeps the port closed until the link goes down during Auto-Negotiation and comes up again (the port remains closed if the link does not go down).

The availability of Enhanced MII Link Detection depends on a supported PHY address configuration, otherwise it has to be disabled.

55.3.9 MII Management Interface (MI)

Most EtherCAT Subordinate Controllers with MII/RMII ports use the MII management interface for communication with the Ethernet PHYs. The MII management interface can be used by the EtherCAT master - or the local µController if supported by the eCAT. Enhanced MII link detection uses the management interface for configuration and restarting auto negotiation after communication errors occurred (TX PHYs only). Some ESC can make use of the MII management interface for link detection and PHY configuration. For fast link detection, the ESC require to use a separate signal (LINK_MII).

The ESCs support a shared MII Management Interface for all PHYs, i.e., MCLK and MDIO are connected to the ESC and to all PHYs.

55.3.9.1 PHY addressing/phy address offset

Proper PHY address configuration is crucial for Enhanced Link Detection and MI Link Detection and configuration, because the ESC itself needs to relate logical ports to the corresponding PHY addresses. The EtherCAT master can access the Ethernet PHYs using any address by means of the PHY address register 0x0513.

Typically, the logical port numbers match with the PHY addresses, i.e. the EtherCAT master and the ESC itself use PHY address 0 for accessing the PHY at port 0 (PHY address 1 for the PHY at port 1 and so on).

Depending on the ESC, there are two options for configuring the PHY addresses:

- PHY address offset: The ESC accesses a PHY at logical port x with the address "x + PHY address offset". Depending on the ESC only some or all possible offsets are configurable. The EtherCAT master uses the PHY addresses 0-3 to access the logical ports 0-3. These addresses are incremented by the PHY address offset inside the ESC. If the master uses PHY addresses 4-31, these addresses are also incremented by the PHY address offset inside the ESC.
- Individual PHY address: The ESC accesses a PHY at each logical port with an individual PHY address. The EtherCAT master uses the PHY addresses 0-3 to access the logical ports 0-3. These addresses are replaced with the individual addresses inside the ESC. If the master uses PHY addresses 4-31, these addresses are not translated.

Depending on the ESC the PHY address configuration is a strapped at power-up, configured in advance (IP Core) or configured by signals.

Typically, the PHY address offset should be 0, and the logical port numbers match with the PHY addresses. Some Ethernet PHYs associate a special function with PHY address 0, e.g., address 0 is a broadcast PHY address. In these cases, PHY address 0 cannot be used. Instead, a PHY address offset different from 0 should be selected, preferably an offset which is supported by the ESC. If PHY addresses are chosen which are not supported by the ESC, Enhanced Link Detection and MI Link Detection and Configuration cannot be used and have to be disabled (the PHY address offset should be 0 in these cases). Nevertheless, the EtherCAT master can communicate with the PHYs using the actual PHY addresses, and EtherCAT communication is possible anyway- using the LINK_MII signal. It is recommended that the PHY addresses are selected to be equal to the logical port number plus 1 in this case.

If the PHY address offset configuration of an ESC reflects the actual PHY address settings, the EtherCAT master can use addresses 0-3 in PHY address register 0x0513 for accessing the PHYs of logical ports 0-3, regardless of the PHY address offset.

Table 795. PHY Address configuration matches PHY address settings

Logical Port	Configured address of the PHY		PHY address register value used by EtherCAT master
	PHY address offset = 0	PHY address offset = 16	
0	0	16	0
1	1	17	1
2	2	18	2
3	3	19	3
none	4-15	20-31	4-15
none	16-31	0-15	16-31

If the actual PHY address settings differ from the PHY address configuration of the ESC, the EtherCAT master has to use the actual PHY address mapping, i.e., PHY addresses 1-4 for accessing the PHYs of logical ports 0-3.

Table 796. PHY Address configuration does not match actual PHY address settings

Logical Port	Configured address of the PHY	PHY address register value used by EtherCAT master
0	1	1
1	2	2
2	3	3
3	4	4
none	0, 5-31	0, 5-13

NOTE

PHY address offset is 0 in this case (recommended).

55.3.9.2 Logical interface

The MI of the ESC is typically controlled by EtherCAT via the MI registers

Table 797. PHY address configuration does not match actual PHY address settings

Register Address	Description
0x0510:0x0511	MII Management Control/Status
0x0512	PHY Address
0x0513	PHY Register Address
0x0514:0x0515	PHY Data

The MI supports two commands: write to one PHY register or read one PHY register.

55.3.9.2.1 MI read/write example

The following steps have to be performed for a PHY register access:

- Check if the Busy bit of the MI Status register is cleared and the MI is not busy.
- Write PHY address to PHY Address register.

- Write PHY register number to be accessed into PHY Register Address register (0-31).
- Write command only: put write data into PHY Data register (1 word/2 byte).
- Issue command by writing to Control register. For read commands, write 1 into Command Register Read 0x0510[8]. For write commands, write 1 into Write Enable bit 0x0510[0] and also 1 into Command Register Write 0x0510[9]. Both bits have to be written in one frame. The Write enable bit realizes a write protection mechanism. It is valid for subsequent MI commands issued in the same frame and self- clearing afterwards.
- The command is executed after the EOF, if the EtherCAT frame had no errors.
- Wait until the Busy bit of the MI Status register is cleared.
- Check the Error bits of the MI Status register. The command error bit is cleared with a valid command or by clearing the command register. The read error bit indicates a read error, e.g., a wrong PHY address. It is cleared by writing to the register.
- Read command only: Read data is available in PHY Data register.

NOTE

NOTE: The Command register bits are self-clearing. Manually clearing the command register will also clear the status information.

55.3.9.2.2 MI Interface Assignment to ECAT/PDI

The EtherCAT master controls the MI Interface (default) if the MII Management PDI Access State register 0x0517[0] is not set. The EtherCAT master can prevent PDI control over the MI Interface, and it can force the PDI to release the MI Interface control. After power-on, the PDI can take over MI Interface control without any master transactions.

55.3.9.2.3 MI Protocol

Each MI access begins with a Preamble of "Ones"(32 without preamble suppression, less if both ESC and PHY support preamble suppression), followed by a Start-of-Frame (01) and the Operation Code (01 for write and 10 for read operations). Then the PHY address (5 bits) and the PHY register address (5 bits) are transmitted to the PHY. After a Turnaround (10 for write and Z0 for read operations - Z means MDIO is high impedance), two bytes of data follow. The transfer finishes after the second data byte and at least one IDLE cycle.

55.3.9.2.4 Timing Specifications

Table 798. MII Management Interface timing characteristics

Parameter	Description
t_{clk}	MDC Period
t_{Write}	Write access time
t_{Read}	Read Access time

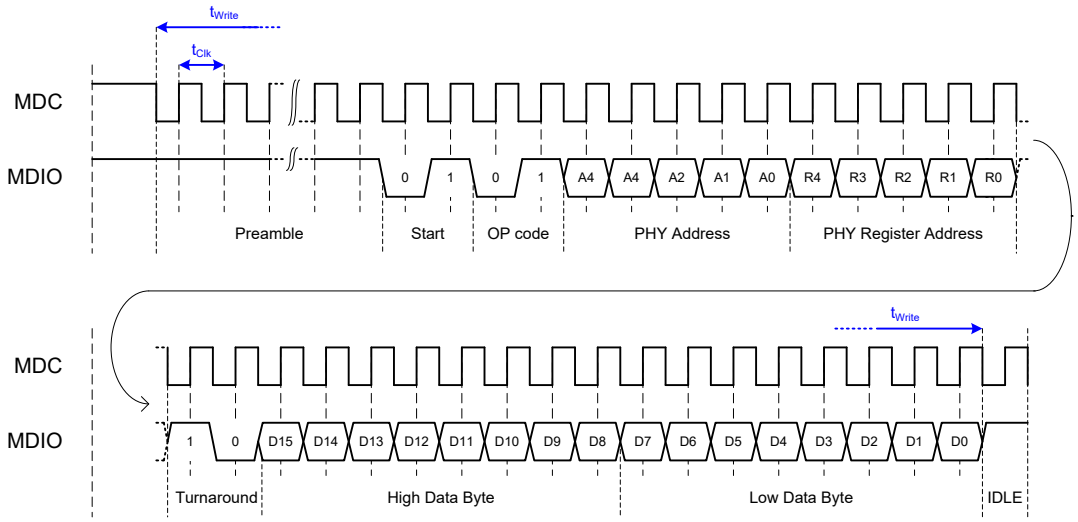


Figure 379. Write Access

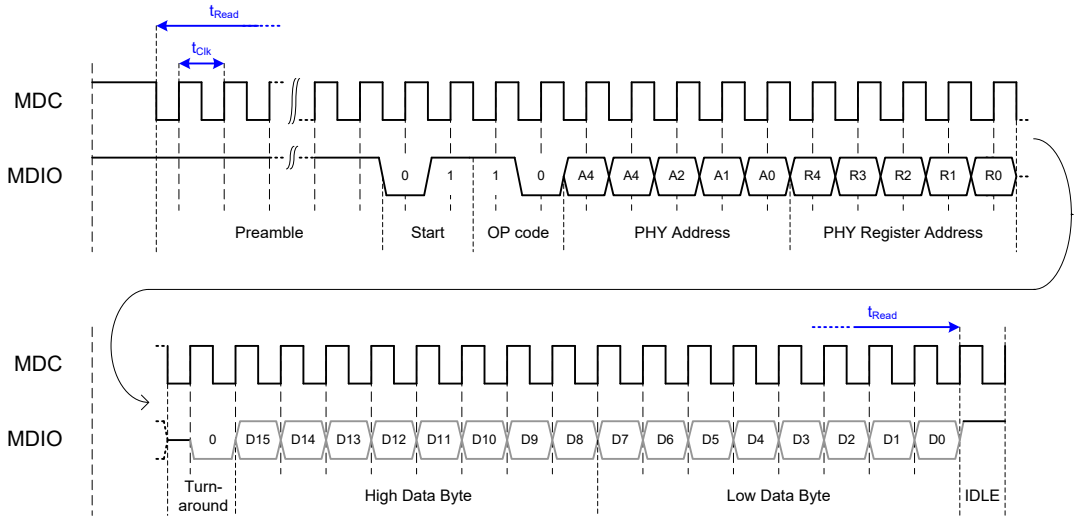


Figure 380. Read Access

55.3.10 MII management example schematic

The MII management interface is a shared bus for all PHYs. The example schematic shows the connection between ESC and PHYs. Take care of proper PHY address configuration

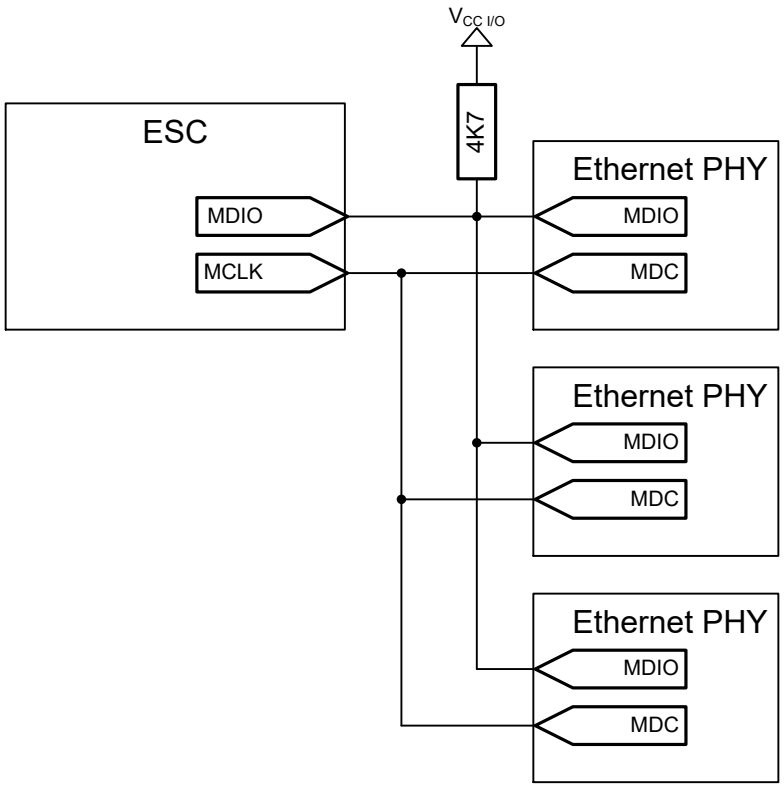


Figure 381. MII management example schematic

55.3.11 Fieldbus Memory Management Units (FMMU)

Fieldbus Memory Management Units (FMMU) convert logical addresses into physical addresses by the means of internal address mapping. Thus, FMMUs allow to use logical addressing for data segments that span several slave devices: one datagram addresses data within several arbitrarily distributed ESCs. Each FMMU channel maps one continuous logical address space to one continuous physical address space of the slave. The FMMUs support bit wise mapping, the number of supported FMMUs depends on the ESC. The access type supported by an FMMU is configurable to be either read, write, or read/write.

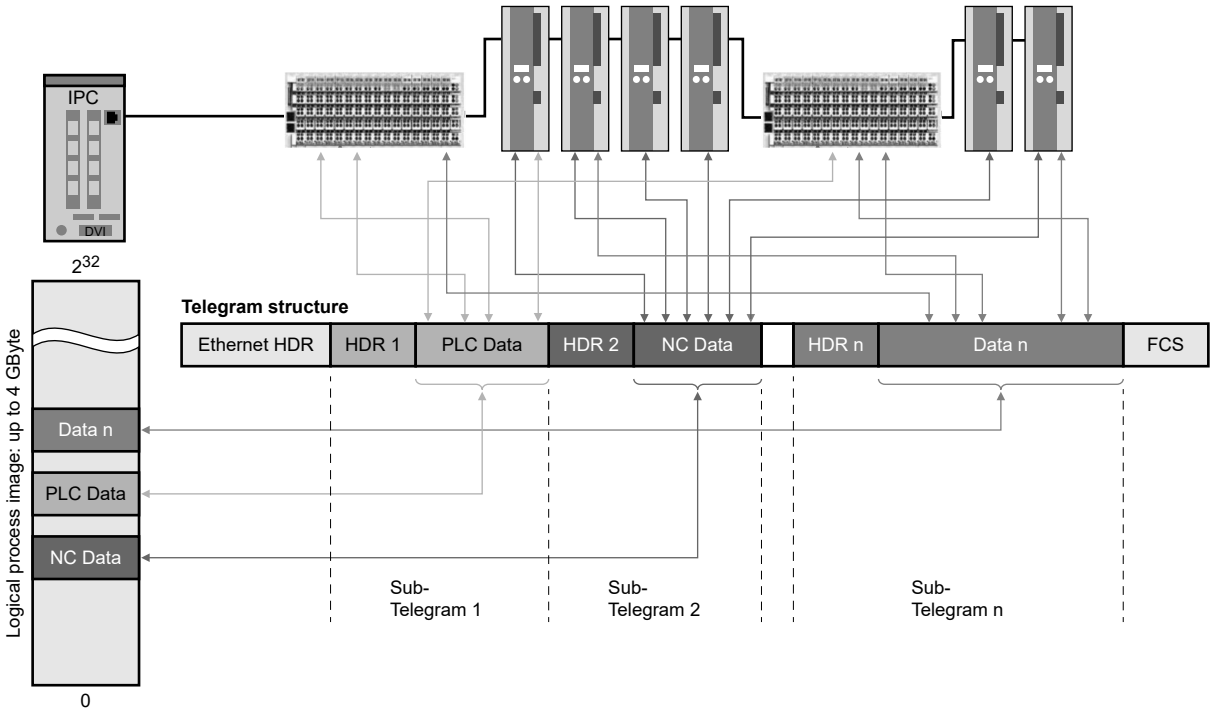


Figure 382. FMMU mapping

NOTE

FMMU configuration registers start at address 0x0600.

Additional FMMU Characteristics:

- Each logical address byte can at most be mapped either by one FMMU(read) plus one FMMU(write), or by one FMMU(read/write). If two or more FMMUs (with the same direction -read or write) are configured for the same logical byte, the FMMU with the lower number (lower configuration address space) is used, the other ones are ignored.
- One or more FMMUs may point to the same physical memory, all of them are used. Collisions cannot occur.
- It is the same to use one read/write FMMU or two FMMUs - one read, the other one write-for the same logical address.
- A read/write FMMU cannot be used together with SyncManagers, since independent read and write SyncManagers cannot be configured to use the same (or overlapping) physical address range.
- Bit-wise reading is supported at any address. Bits which are not mapped to logical addresses are not changed in the EtherCAT datagram. E.g., this allows for mapping bits from several ESCs into the same logical byte.
- A frame/datagram addressing a logical address space which is not configured in the ESC will not change data in the ESC, and no data from the ESC is placed in the frame/datagram.

55.3.12 SyncManager

The memory of an ESC can be used for exchanging data between the EtherCAT master and a local application (on a µController attached to the PDI) without any restrictions. Using the memory for communication like this has some drawbacks which are addressed by the SyncManagers inside the ESCs:

- Data consistency is not guaranteed. Semaphores have to be implemented in software for exchanging data in a coordinated way.
- Data security is not guaranteed. Security mechanisms have to be implemented in software.

- Both EtherCAT master and application have to poll the memory in order to find out when the access of the other side has finished.

SyncManagers enable consistent and secure data exchange between the EtherCAT master and the local application, and they generate interrupts to inform both sides of changes.

SyncManagers are configured by the EtherCAT master. The communication direction is configurable, as well as the communication mode (Buffered Mode and Mailbox Mode). SyncManagers use a buffer located in the memory area for exchanging data. Access to this buffer is controlled by the hardware of the SyncManagers.

A buffer has to be accessed beginning with the start address, otherwise the access is denied. After accessing the start address, the whole buffer can be accessed, even the start address again, either as a whole or in several strokes. A buffer access finishes by accessing the end address, the buffer state changes afterwards and an interrupt or a watchdog trigger pulse is generated (if configured). The end address cannot be accessed twice inside a frame.

Two communication modes are supported by SyncManagers:

- Buffered Mode

  The buffered mode allows both sides, EtherCAT master and local application, to access the communication buffer at any time. The consumer always gets the latest consistent buffer which was written by the producer, and the producer can always update the content of the buffer. If the buffer is written faster than it is read out, old data will be dropped.

  The buffered mode is typically used for cyclic process data.

- Mailbox Mode

  The mailbox mode implements a handshake mechanism for data exchange, so that no data will be lost. Each side, EtherCAT master or local application, will get access to the buffer only after the other side has finished its access. At first, the producer writes to the buffer. Then, the buffer is locked for writing until the consumer has read it out. Afterwards, the producer has write access again, while the buffer is locked for the consumer.

  The mailbox mode is typically used for application layer protocols.

The SyncManagers accept buffer changes caused by the master only if the FCS of the frame is correct, thus, buffer changes take effect shortly after the end of the frame. The configuration registers for SyncManagers are located beginning at register address 0x0800.

Table 799. SyncManager Register overview

Description	Register Address Offset
Physical Start Address	0x0:0x1
Length	0x2:0x3
Control Register	0x4
Status Register	0x5
Activate	0x6
PDI Control	0x7

55.3.12.1 Buffered mode

The buffered mode allows writing and reading data simultaneously without interference. If the buffer is written faster than it is read out, old data will be dropped. The buffered mode is also known as 3-buffer-mode.

Physically, 3 buffers of identical size are used for buffered mode. The start address and size of the first buffer is configured in the SyncManager configuration. The addresses of this buffer have to be used by the master and the local application for reading/writing the data. Depending on the SyncManager state, accesses to the first buffer's (0) address range are redirected to one of the 3 buffers. The memory used for buffers 1 and 2 cannot be used and should be taken into account for configuring other SyncManagers.

One buffer of the three buffers is allocated to the producer (for writing), one buffer to the consumer (for reading), and the third buffer keeps the last consistently written data of the producer. As an example, Figure 24 demonstrates a configuration with start address 0x1000 and Length 0x100. The other buffers shall not be read or written. Access to the buffer is always directed to addresses in the range of buffer 0.

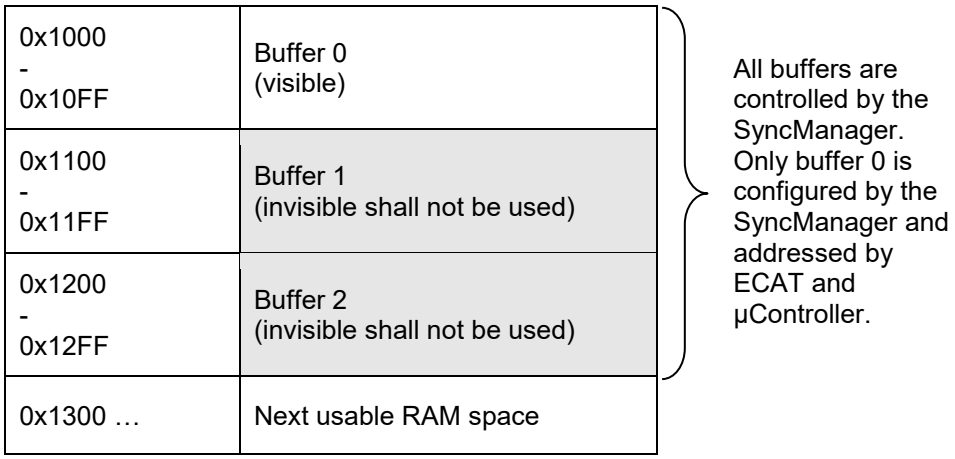


Figure 383. SyncManager buffer allocation

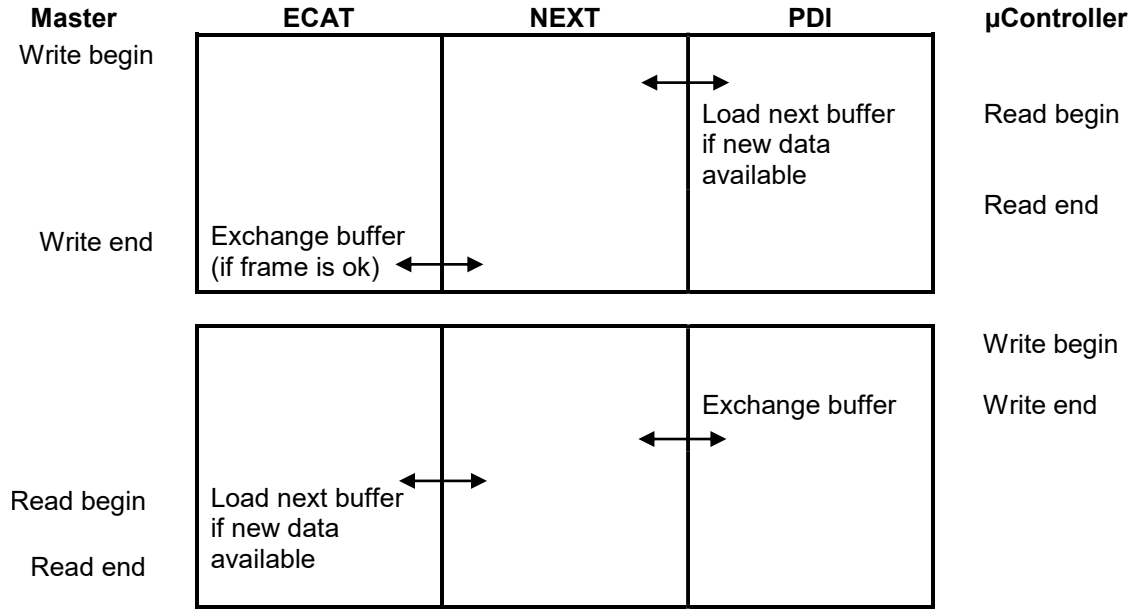


Figure 384. SyncManager mailbox interaction

The Status register of the SyncManager reflects the current state. The last written buffer is indicated (informative only, access redirection is performed by the ESC), as well as the interrupt states. If the SyncManager buffer was not written before, the last written buffer is indicated to be 3 (start/empty).

55.3.12.2 Mailbox mode

The mailbox mode only allows alternating reading and writing. This assures all data from the producer reaches the consumer. The mailbox mode uses just one buffer of the configured size.

At first, after initialization/activation, the buffer (mailbox, MBX) is writeable. Once it is written completely, write access is blocked, and the buffer can be read out by the other side. After it was completely read out, it can be written again. The time it takes to read or write the mailbox does not matter.

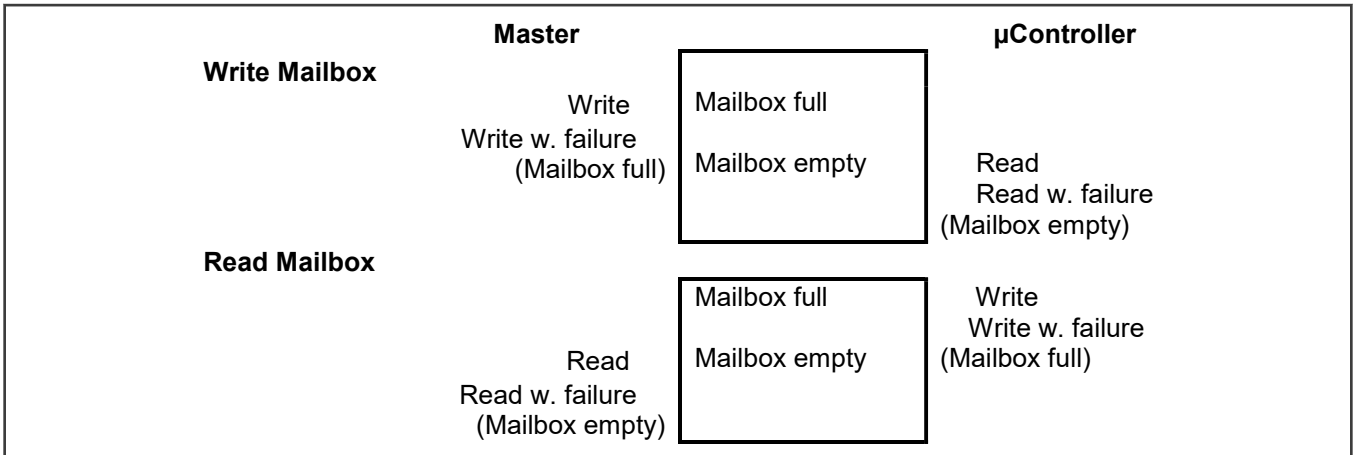


Figure 385. SyncManager mailbox interaction

55.3.12.2.1 Mailbox communication protocols

This is only a brief overview of the mailbox communication protocols.

- **Ethernet over EtherCAT (EoE):** Defines a standard way to exchange or tunnel standard Ethernet Frames over EtherCAT.
- **CAN application layer over EtherCAT (CoE)** Defines a standard way to access a CAN application layer object dictionary and to exchange CAN application layer Emergency and PDO messages on an event driven path.
- **File Access over EtherCAT (FoE)** Defines a standard way to download and upload firmware and other 'files'.
- **Servo Profile over EtherCAT (SoE)** Defines a standard way to access the IEC 61800-7 identifier.
- **Vendor specific Profile over EtherCAT (VoE)** A vendor specific protocol follows after a VoE header that identifies the vendor and a vendor specific type.
- **ADS over EtherCAT (AoE)** AoE defines a standard way to exchange Automation Device Specification (ADS) messages over EtherCAT.

The content of an EtherCAT mailbox header is shown in below figure.

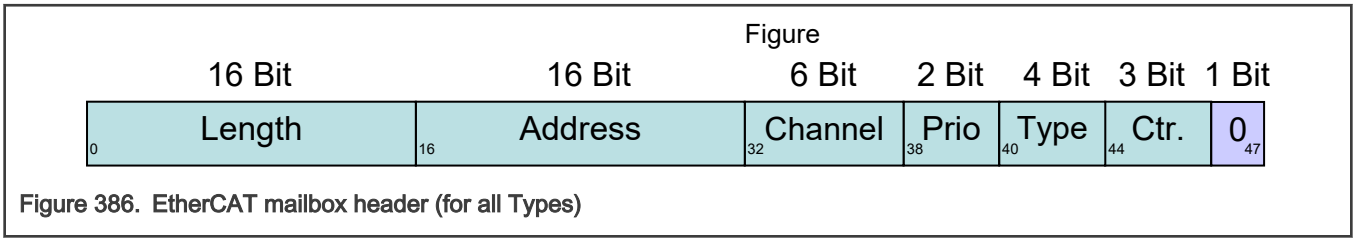


Figure 386. EtherCAT mailbox header (for all Types)

Table 800. EtherCAT mailbox header

Field	Data Type	Value/Description
Length	WORD	Number of Bytes of this mailbox command excluding the mailbox header
Address	WORD	Station Address of originator
Channel	6 bit	0 - reserved for future use
Priority	2 bit	Priority between 0 (lowest) and 3 (highest)
Type	4 bit	Mailbox protocol types:

Table continues on the next page...

Table 800. EtherCAT mailbox header (continued)

Field	Data Type	Value/Description
		• 0x0: Error • 0x1: Vendor specific (AoE -ADS over EtherCAT) • 0x2: EoE (Ethernet over EtherCAT) • 0x3: CoE (CAN application layer over EtherCAT) • 0x4: FoE (File access over EtherCAT) • 0x5: SoE (Servo profile over EtherCAT) • 0xF: Vendor specific (VoE)
Ctr.	3 bit	Sequence number that is used for detection of duplicated frames
Reserved	1 bit	0

55.3.12.3 Interrupt and watchdog trigger generation, latch event generation

Interrupts can be generated when a buffer was completely and successfully written or read. A watchdog trigger signal can be generated to rewind (trigger) the Process Data watchdog used for Digital I/O after a buffer was completely and successfully written. Interrupt and watchdog trigger generation are configurable. The SyncManager Status register reflects the current buffer state.

For debugging purposes it is also possible to trigger Distributed Clock Latch events upon successful buffer accesses (for some ESCs).

55.3.12.4 Single byte buffer length / watchdog trigger for digital output PDI

If a SyncManager is configured for a length of 1 byte (or even 0), the buffer mechanism is disabled, i.e., read/write accesses to the configured address will pass the SyncManager without interference. The additional buffers 1 and 2 are not used in buffered mode, and alternation of write/read accesses is not necessary for mailbox mode. Consistency is not an issue for a single byte buffer. Nevertheless, watchdog triggering is still possible if the buffer length is 1 byte (interrupt generation as well).

NOTE

For some ESCs in Mailbox mode, watchdog and interrupt generation are depending on the alternation of write and read accesses, although the write/read accesses itself are executed without interference. I.e., Buffered mode should be used for single byte buffers for watchdog generation.

Watchdog trigger generation with single byte SyncManagers is used for Digital Outputs, because the outputs are only driven with output data if the Process Data watchdog is triggered. One SyncManager has to be configured for each byte of the Digital Output register (0x0F00:0x0F03) which is used for outputs. The SyncManagers have to be configured like this:

- Buffered Mode (otherwise the process data watchdog will not be wound up with some ESCs upon the second and following writes, because the Digital I/O PDI does not read the addresses)
- Length of 1 byte
- EtherCAT write / PDI read
- Watchdog Trigger enabled

NOTE

A SyncManager with length 0 behaves like a disabled SyncManager. It does not interfere accesses nor generate interrupt or watchdog trigger signals

55.3.12.5 Repeating mailbox communication

Acknowledge bits in the SyncManager Activation register (offset 0x06[1]) and the PDI Control register (offset 0x07[1]) are used in mailbox mode for retransmissions of buffers from a slave to the master. If a mailbox read frame gets lost/broken on the way back to the master, the master can toggle the Repeat Request bit. The slave polls this bit or receives an interrupt (SyncManager activation register changed, register 0x0220[4]) and writes the last buffer again to the SyncManager. Then the PDI toggles the Repeat Acknowledge bit in the PDI Control register. The master will read out this bit and read the buffer content. Communication resumes afterwards. This mechanism is shown in Figure 28 for a mailbox write service. The Mailbox confirmation is lost on its way from the slave to the master and has to be repeated again.

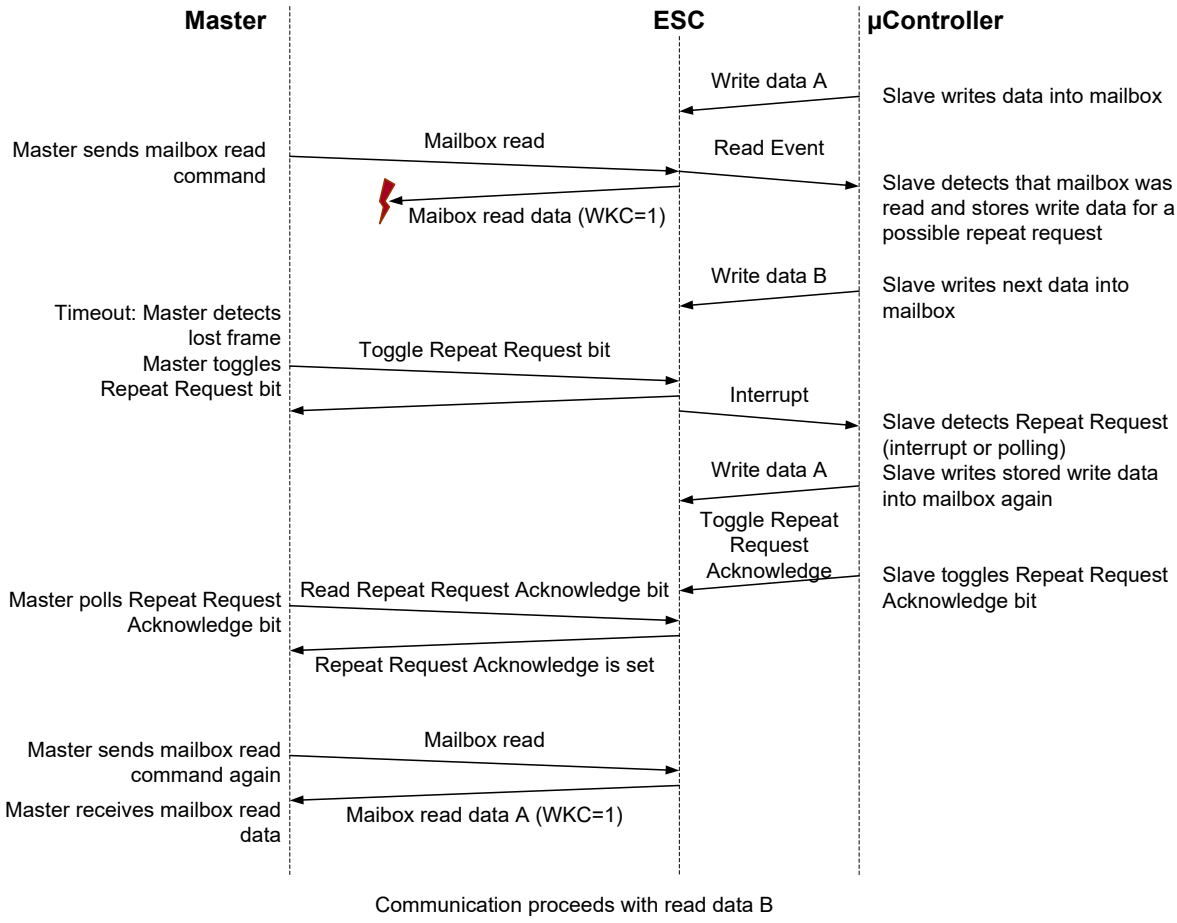


Figure 387. Handling of a repeat request with read mailbox

55.3.12.6 SyncManager deactivation by the PDI

A SyncManager can be deactivated by the PDI to inform the master of local problems (typically used in buffered mode only). The master can detect SyncManager deactivation by checking the Working Counter, which is not incremented if a deactivated SyncManager buffer is accessed. If a SyncManager is deactivated by the PDI (PDI Control register 0x7[0]=1), the state of the SyncManager is reset, interrupts are cleared and the SyncManager has to be written first after re-activation. The entire SyncManager buffer area is read/write protected while the SyncManager is deactivated by the PDI.

55.3.13 SII EEPROM

EtherCAT Subordinate Controllers use a mandatory NVRAM (typically a serial EEPROM with I2C interface) to store EtherCAT Slave Information (ESI). EEPROM sizes from 1 Kbit up to 4 Mbit are supported, depending on the ESC.

The EtherCAT IP Core supports omitting the serial I2C EEPROM if a µController with read/write access to an NVRAM (e.g., the one which contains the µController's program and data, or the FPGA configuration EPPROM) is used to emulate the EEPROM transactions.

The EEPROM structure is shown in the below figure. The ESI uses word addressing.

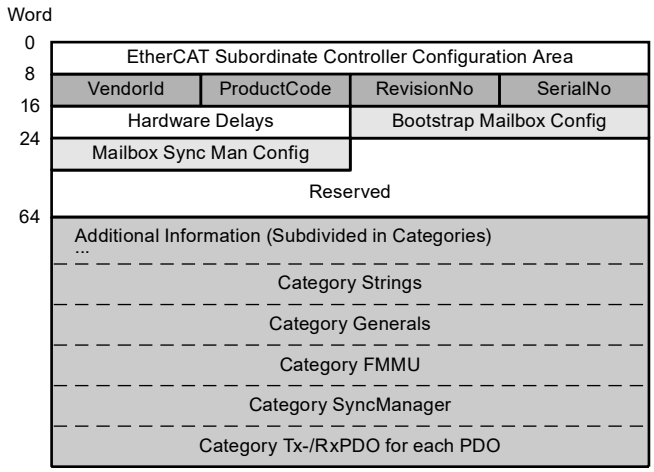


Figure 388. SII EEPROM layout

55.3.13.1 SII EEPROM Content

The ESC Configuration Area (EEPROM word addresses 0 to 7) is automatically read by the ESC after power-on or reset. It contains the PDI configuration, DC settings, and the Configured Station Alias.

The consistency of the ESC Configuration data is secured with a checksum.

The EtherCAT master can invoke reloading the EEPROM content. In this case the Configured Station Alias register 0x0012:0x0013 and ESC Configuration register bits 0x0141[1,4,5,6,7] (enhanced link detection) are not transferred into the registers, they are only transferred at the initial EEPROM loading after power-on or reset.

The ESC Configuration Area is shown in the below table.

Table 801. ESC Configuration Area

Word Address	Parameter	Description	Register Address
0x0	PDI Control/ ESC Configuration	Initialization value for PDI Control register (EEPROM ADR 0x0000[9] is also mapped to register 0x0110[2])	0x0140:0x0141
0x1	PDI Configuration	Initialization value for PDI Configuration register	0x0150:0x0151
0x2	Pulse Length of SYNC Signals	Initialization value for Pulse Length of SYNC Signals register	0x0982:0x0983

Table continues on the next page...

Table 801. ESC Configuration Area (continued)

0x3	Extended PDI Configuration	Initialization value for extended PDI Configuration register	0x0152:0x0153
0x4	Configured Station Alias	Initialization value for Configured Station Alias Address register	0x0012:0x0013
0x5	Reserved	Reserved, shall be zero	-
0x6	Reserved	Reserved, shall be zero	-
0x7	Checksum	Low byte contains remainder of division of word 0 to word 6 as unsigned number divided by the polynomial x8+x2+x+1(initial value 0xFF). NOTE: For debugging purposes it is possible to disable the checksum validation with a checksum value of 0x88A4. Never use this for production!	-

NOTE: Reserved words or reserved bits of the ESC Configuration Area should be filled with 0.

An excerpt of the SII EEPROM content following the ESC Configuration area is shown in below table.

Table 802. SII EEPROM Content Excerp

Word Address	Parameter
0x0	PDI Control
0x1	PDI Configuration
0x2	Pulse Length of SYNC Signals
0x3	Extended PDI Configuration
0x4	Configured Station Alias
0x5	Extended ESC configuration 1
0x6	Extended ESC configuration 2
0x7	Extended ESC configuration 1
0x8:0x9	Vendor ID
0xA:0xB	Product Code
0xC:0xD	Revision Number
0xE:0xF	Serial Number
0x10:0x13	Reserved
0x14	Bootstrap Receive Mailbox Offset
0x15	Bootstrap Receive Mailbox Size
0x16	Bootstrap Send Mailbox Offset
0x17	Bootstrap Send Mailbox Size
0x18	Standard Receive Mailbox Offset

Table continues on the next page...

Table 802. SII EEPROM Content Excerp (continued)

0x19	Standard Receive Mailbox Size
0x1A	Standard Send Mailbox Offset
0x1B	Standard Send Mailbox Size
0x1C	Mailbox Protocol
0x1D:0x1F	Reserved
0x20:0x22	Vendor specifc
0x23:0x27	Reserved
0x28:0x2F	Additional ESC configuration
0x30:0x3D	Reserved
0x3E	Size
0x3F	Version
0x40	Category header
0x41	Category word size
0x42	Category data

55.3.13.2 SII EEPROM logical interface

The SII EEPROM interface of the eCAT is either controlled by EtherCAT or by the PDI. Initially, EtherCAT has EEPROM interface access, but it can transfer access to the PDI.

Table 803. SII EEPROM interface register overview

Register Address	Description
0x0500	EEPROM Configuration
0x0501	EEPROM PDI Access State
0x0502:0x0503	EEPROM Control/Status
0x0504:0x0507	EEPROM Address
0x0508:0x050F	EEPROM Data

The EEPROM interface supports three commands: write to one EEPROM address (1 Word), read from EEPROM (2 or 4 Words, depending on eCAT), or reload eCAT configuration from EEPROM.

55.3.13.2.1 SII EEPROM errors

The ESC retries reading the EEPROM after power-on or reset once if an error has occurred (missing acknowledge, wrong checksum). If reading the ESC Configuration Area fails twice, the Error Device Information bit is set, and the PDI Operational bit in the ESC DL Status register (0x0110[0]) remains clear and the EEPROM_Loaded signal (if available) remains inactive. The process memory is not accessible until the ESC Configuration Area is loaded successfully.

All registers initialized by the ESC Configuration Area keep their values in case of an error. This is also true for the Error Device Information bit as well as the PDI Operational bit. Only if the EEPROM was loaded/reloaded successfully, the registers take over

the new values (except for Configured Station Alias register 0x0012:0x0013 and ESC Configuration register bits 0x0141[1,4,5,6,7] – enhanced link detection).

The SII EEPROM interface has these error status bits in register EEPROM Control/Status (0x0502:0x0503):

Table 804. SII EEPROM interface errors

Bit	Name	Description
11	Checksum Error	ECAT Configuration Area checksum is wrong (after device initialization or EEPROM reload). Registers initialized with EEPROM values keep their value. Reason: CRC error Solution: Check CRC
12	Error Device Information	ECAT Configuration not loaded Reasons: Checksum error, acknowledge error, EEPROM missing Solution: Check other error bits
13	Error Acknowledge/ Command	Missing Acknowledge or invalid command Reason: a) Missing acknowledge from EEPROM chip (see below). EEPROM chip is busy performing operations internally or EEPROM chip is not available. b) Invalid command issued Solution: a) Retry access. EEPROM device does not acknowledge if it is internally busy. b) Use valid commands
14	Error Write Enable	Write without Write Enable (ECAT control only): Reason: ECAT issued a write command without Write Enable bit set (0x0502[0]) Solution: Set Write Enable bit in the same frame as the write command

55.3.13.2.1.1 Missing acknowledge

Missing acknowledges from the EEPROM chip are a common issue, especially if a fast PDI uses the EEPROM interface. E.g., a write access to the EEPROM with missing acknowledge may look like this:

1. ECAT/PDI issue write command (first command)
2. ESC is busy transferring the write data to the EEPROM chip.
3. ESC is not busy anymore. EEPROM chip is internally busy transferring data from input buffer to storage area.
4. ECAT/PDI issue a second command.
5. ESC is busy transferring the write data to the EEPROM chip. EEPROM chip does not acknowledge any access until internal transfer has finished (may take up to several ms).
6. ESC is not busy anymore. Error Acknowledge/Command bit is set. (ESC has to re-issue the second command after EEPROM chip is finished and the command is acknowledged).
7. EEPROM chip finishes internal transfer.
8. ESC re-issues the second command, the command is acknowledged and executed successful.

This is also possible for a read access, because some EEPROM chips require an idle period between any two accesses. During this idle period, they do not acknowledge any access.

55.3.13.2.2 SII EEPROM interface assignment to ECAT/PDI

The EtherCAT master controls the EEPROM interface (default) if EEPROM configuration register 0x0500[0]=0 and EEPROM PDI Access register 0x0501[0]=0, otherwise PDI controls the EEPROM interface. These access rights should be checked by both sides before using the EEPROM Interface.

A typical EEPROM interface control hand-over is as follows:

1. ECAT assigns EEPROM interface to PDI by writing 0x0500[0]=1
2. If PDI wishes to access EEPROM, it takes over EEPROM control by writing 0x0501[0]=1.
3. PDI issues EEPROM commands.
4. After PDI has finished EEPROM commands, PDI releases EEPROM control by writing 0x0501[0]=0.
5. ECAT may take back the EEPROM interface by writing 0x0500[0]=0
6. ECAT checks EEPROM control by reading 0x0501
7. ECAT issues EEPROM commands.

If the PDI does not release EEPROM control (e.g. because of a software failure), ECAT can force releasing the access:

1. ECAT writes 0x02 to register 0x0500 (as the result, 0x0501[0] is cleared)
2. ECAT writes 0x00 to register 0x0500
3. ECAT has control over EEPROM interface

55.3.13.2.3 Read/write/reload example

The following steps have to be performed for an SII EEPROM read or write access:

1. Check if the Busy bit of the EEPROM Status register is cleared (0x0502[15]=0) and the EEPROM interface is not busy, otherwise wait until the EEPROM interface is not busy anymore.
2. Check if the Error bits of the EEPROM Status register are cleared. If not, write “000” to the command register (register 0x0502 bits [10:8]).
3. Write EEPROM word address to EEPROM Address register.
4. Write command only: put write data into EEPROM Data register (1 word/2 byte only).
5. Issue command by writing to Control register.
    a. For a read command, write 001 into Command Register 0x0502[10:8].
    b. For a write command, write 1 into Write Enable bit 0x0502[0] and 010 into Command Register 0x0502[10:8]. Both bits have to be written in one frame. The Write enable bit realizes a write protection mechanism. It is valid for subsequent EEPROM commands issued in the same frame and self-clearing afterwards. The Write enable bit needs not to be written from PDI if it controls the EEPROM interface.
    c. For a reload command, write 100 into Command Register 0x0502[10:8].
6. The command is executed after the EOF if the EtherCAT frame had no errors. With PDI control, the command is executed immediately.
7. Wait until the Busy bit of the EEPROM Status register is cleared.
8. Check the Error bits of the EEPROM Status register. The Error bits are cleared by clearing the command register. Retry command (back to step 5) if EEPROM acknowledge was missing. If necessary, wait some time before retrying to allow slow EEPROMs to store the data internally.
9. a) For a Read command: Read data is available in EEPROM Data registers (2 or 4 Words, depending on eCAT – check register 0x0502[6]).

b) For a Reload command: ESC configuration is reloaded into appropriate registers.

NOTE: The Command register bits are self-clearing. Manually clearing the command register will also clear the status information.

55.3.13.3 SII EEPROM electrical interface (I2C)

The SII EEPROM Interface is intended to be a point-to-point interface between the ESC and the I²C EEPROM. If other I²C masters are required to access the I²C bus, the ESC must be held in reset state (e.g. for in-circuit-programming of the EEPROM), otherwise access collisions are possible.

The SII EEPROM interface has the following signals1:

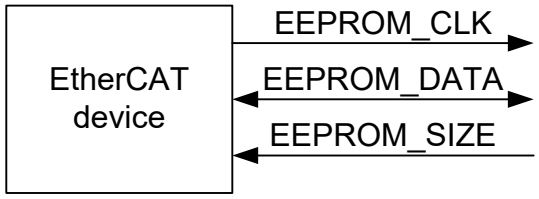


Figure 389. I²C EEPROM signals

Table 805. I²C EEPROM signals

Signal	Direction	Description
EEPROM_CLK	OUT	I²C clock
EEPROM_DATA	BIDIR	I²C data
EEPROM_SIZE	IN	EEPROM size configuration

Both EEPROM_CLK and EEPROM_DATA must have a pull-up resistor (4.7 kΩ recommended for ESCs), either integrated into the ESC or connected externally.

55.3.13.3.1 Addressing

EtherCAT and ESCs use word addressing when accessing the EEPROM, although the I²C interface actually uses byte addressing. The lowest address bit A[0] is added internally by the EEPROM interface controller of the ESCs. I.e., the EEPROM address register (0x0504:0x0507) reflects the physical EEPROM address bits A[18:1] (higher address bits are reserved/are zero).

SII EEPROM word 0 is typically located at I²C address 0, i.e., the I²C device address has to be set to

0. Some ESCs support an offset in the I²C base address.

55.3.13.3.2 EEPROM size

Depending on the EEPROM size, one out of two EEPROM algorithms has to be selected with the EEPROM_SIZE configuration signal. Smaller EEPROMs need only one address byte, larger ones need two address bytes:

Table 806. EEPROM size

EEPROM Size	Address Bytes	Max. I²C Address Bits	EEPROM_SIZE signal
1 Kbit – 16 Kbit	1	11	0
32 Kbit – 4 Mbit	2	19	1

55.3.13.3.3 I²C access protocol

Each EEPROM access begins with a Start condition and ends with a Stop condition. Data is transferred byte-wise, and each byte is acknowledged by the recipient.

The Start condition is a falling edge on EEPROM_DATA while EEPROM_CLK is high, the Stop condition is a rising edge on EEPROM_DATA while EEPROM_CLK is high. In all other cases, EEPROM_DATA has to remain stable while EEPROM_CLK is high, as this indicates valid data. A byte transfer is acknowledged in an additional bit, which is driven low by the recipient of the byte transfer if it acknowledges the byte.

NOTE: If the EEPROM does not acknowledge an access (Ack bit=high), it might be busy internally. Especially if the EERPOM interface is handled by a µController via the PDI, this situation may come up, because many µControllers can write to the EEPROM interface much faster than many EEPROMs can transfer the data from their input registers into their NVRAM.

The first byte of an I²C access is the Control Byte (Bit 7/MSB is transferred first):

Table 807. I²C control byte

Bit	Description
0	Read/Write access: 0: Write 1: Read
[3:1]	Chip Select Bits/Highest Address Bits
[7:4]	Control Code: 1010

Depending on the access, either read data will follow or additional address bytes and write data. This is described in the following chapters.

The EEPROM has an internal byte pointer, which is incremented automatically after each data byte transfer.

55.3.14 Watchdogs

The ESCs support up to two internal watchdogs (WD), a Process Data watchdog used for monitoring process data accesses, and a PDI watchdog monitoring PDI activity. The timeout for both watchdogs can be configured individually, but they share a single Watchdog Divider (WD_DIV, register 0x0400:0x0401). The watchdog timeout is calculated from the Watchdog Divider settings multiplied with the Watchdog Time settings for PDI (WD_PDI, register 0x0410:0x0411) or Process Data (WD_PD, register 0x0420:0x0421). Base time unit is 40 ns. The Watchdog timeout jitters, the jitter depends on the Watchdog Divider settings. i.e., selecting smaller Watchdog Divider settings results in smaller jitter.

The following equations are used for a quick estimation of the watchdog timeout (they are not exact in terms of nanoseconds):

$t_{WD\_Div} = (W_{D\_DIV} + 2) * 40ns$

$t_{WD\_PDI} = [t_{WD\_Div} * W_{D\_PDI} ; t_{WD\_Div} * W_{D\_PDI} + t_{WD\_Div}]$

$t_{WD\_PD} = [t_{WD\_Div} * W_{D\_PD} ; t_{WD\_Div} * W_{D\_PD} + t_{WD\_Div}]$

55.3.14.1 Process data watchdog

The Process Data watchdog is rewound (triggered) by a write access to a SyncManager buffer area if the SyncManager is configured to generate a watchdog trigger signal (SyncManager Control register 0x0804[6] for SyncManager 0, etc.). The watchdog trigger signal is generated after the buffer was completely and successfully written (similar to the Interrupt Write of a SyncManager). The Process Data watchdog can be disabled by setting the Process Data Watchdog Time to 0. A timeout of the Process Data watchdog has these consequences:

- Watchdog Status register for Process Data (0x0440[0]) reflects the watchdog status.
- The Digital I/O PDI takes back digital output data, either by not driving the signals anymore or by driving them low (ESC and configuration dependent).
- The Watchdog Counter Process Data (0x0442) is incremented.

55.3.14.2 PDI watchdog

The PDI watchdog is rewound (triggered) by any correct read or write access by the PDI. The PDI watchdog can be disabled by setting the PDI Watchdog Time to 0. A timeout of the PDI watchdog has these consequences:

- ESC DL Status register (0x0110[1]) reflects the watchdog status. This can be mapped to the ECAT Interrupt to inform the master.
- The Watchdog Counter PDI (0x0443) is incremented.

NOTE

The Digital I/O PDI only triggers the PDI watchdog upon input events

55.3.15 EtherCAT state machine

The EtherCAT State machine (ESM) is responsible for the coordination of master and slave applications at start up and during operation. State changes are typically initiated by requests of the master. They are acknowledged by the local application after the associated operations have been executed. Unsolicited state changes of the local application are also possible.

Simple devices without a µController can be configured to use EtherCAT State Machine emulation. These devices simply accept and acknowledge any state change automatically.

There are four states an EtherCAT slave shall support, plus one optional state:

- Init (state after Reset)
- Pre-Operational
- Safe-Operational
- Operational
- Bootstrap (optional)

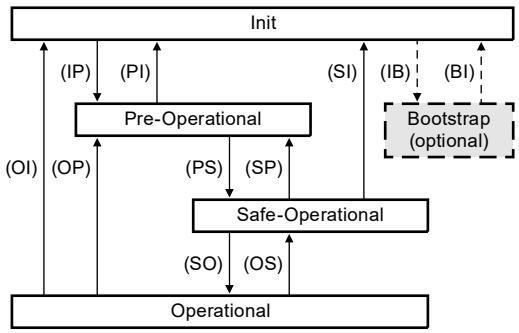


Figure 390. EtherCAT state machine

Each state defines required services. Before a state change is confirmed by the slave all services required for the requested state have to be provided or stopped respectively

55.3.16 LED Signals (Indicators)

55.3.16.1 RUN LED

The AL status is displayed with the RUN LED (green). The RUN output of an ESC is controlled by the AL status register (0x0130) and supports the following states, which are automatically translated into blink codes:

Table 808. RUN LED state indication

RUN LED	Description
Off	The device is in state INIT
Blinking (slow)	The device is in state PRE-OPERATIONAL
Single Flash	The device is in state SAFE-OPERATIONAL
On	The device is in state OPERATIONAL
Flickering (fast)	The device is in state BOOTSTRAP or loading the SII EEPROM*

- Some ESCs support RUN LED flickering while the SII EEPROM is loaded.

55.3.16.1.1 RUN LED override

Some ESCs support optional RUN LED outputs by overriding the state indication of the RUN LED. The output can be set by master or local application. This can be used e.g. for locating a specific slave by forcing the RUN LED to indicate a triple flash (Device Identification).

The RUN LED override is disabled automatically by a following ESM state change.

Table 809. Registers for RUN LED control

Register Address	Name	Description
0x0130	AL Status register	After a state change, the RUN LED indicates the current ESM state.
0x0138	RUN LED Override	Direct control of the RUN LED from ECAT and/or PDI

55.3.16.2 ERR LED

The ERR LED indicates local errors and application errors. It is either connected to the application controller or to the eCAT (optional eCAT feature). If it is connected to the eCAT, some errors are automatically indicated by the eCAT, other error states are detected by the application controller and indicated by writing to the ERR LED Override register 0x0139.

The following ERR LED states can be automatically generated by an eCAT, without interaction of an application controller. The support of individual error states is eCAT specific.

Table 57:

Table 810. Automatic eCAT ERR LED state indication

ERR LED	Description	Examples
Off	No error	
Flickering (fast)	SII EEPROM loading error	no SII EEPROM device, SII CRC error
Blinking (slow)	Invalid hardware configuration	ESC pin sharing violation

Table continues on the next page...

Table 810. Automatic eCAT ERR LED state indication (continued)

Single Flash	AL Status register Error Indication bit 0x0130[4] is set while device emulation is disabled	Local application controller sets Error Indication bit. NOTE: E.g., if the µController makes a state change with Error Indication bit 0x0130[4] set after a Process Data Watchdog Timeout, it has to manually set the ERR LED to Double Flash again (otherwise the ESC would generate a Single Flash due to the Error Indication bit automatically).
Double Flash	Process Data Watchdog timeout (edge detected) while device is OPERATIONAL	master is disconnected/not sending process data any more
On	PDI Watchdog timeout (edge detected) or Build-In self-test error	local application controller failure or self-test failure

NOTE: Do not confuse the application ERR LED with the port receive error LEDs (PERR(x)) supported by some eCATs.

55.3.16.2.1 ERR LED override

If the ESC supports the ERR LED, the local application (and the master) is able to control the ERR LED via the ERR LED override register.

The ERR LED override is disabled automatically if an error is detected by the ESC which is associated with an ERR LED blink code.

Table 811. Registers for ERR LED control

Register Address	Name	Description
0x0139	ERR LED Override	Direct control of the ERR LED from ECAT and/or PDI

55.3.16.3 STATE LED and STATE_RUN LED signal

The STATE LED is a bicolor-LED combining RUN and ERR LED. Since the RUN LED part of the STATE LED must be turned off while the ERR LED part is active, the RUN and ERR LED signals cannot be simply combined to drive the bicolor LED. Some ESCs support a STATE_RUN signal, which is turned off while ERR LED is on, so STATE_RUN and ERR signals can be used to drive the bicolor STATE LED. Otherwise the logical combination “RUN and not(ERR)” has to be used to control the RUN LED part of the STATE LED.

55.3.16.4 LINKACT LED

The Link/Activity state of each port is displayed with the LINKACT LED (green).

Table 812. LINKACT LED states

LINKACT LED	Description
Off	No link
Blinking	Link and activity
On	Link without activity

It is highly recommended to use the LINKACT LED signals of the eCATs for driving visible LEDs, instead of the Link/Activity LED signals of the PHY, because the ESC signals reflect the actual link/activity state of the device – not only the state of the PHYs –, and the ESC signals adhere to the ETG.1300 EtherCAT Indicator and Labeling Specification.

The LINKACT LED indicates the link status after evaluation of the Enhanced link detection plus MI link detection and configuration, it corresponds with the ESC DL Status register (0x0110[15,13,11,9] – Communication established.

55.3.16.5 Port Error LED (PERR)

Some eCATs support port receive error indicators PERR(x), which display physical layer RX errors. The PERR(x) LEDs are not part of the ETG.1300 EtherCAT Indicator and Labeling Specification. They are only intended for testing and debugging. The PERR(x) LEDs flash once if a physical layer RX error occurs.

55.3.17 Distributed Clocks (DC)

The Distributed Clocks (DC) unit of EtherCAT Subordinate Controllers supports the following features:

- Clock synchronization between the slaves (and the master)
- Generation of synchronous output signals (SyncSignals)
- Precise time stamping of input events (LatchSignals)
- Generation of synchronous interrupts
- Synchronous Digital Output updates
- Synchronous Digital Input sampling

55.3.17.1 Clocks synchronization

DC clock synchronization enables all EtherCAT devices (master and slaves) to share the same EtherCAT System Time. The EtherCAT devices can be synchronized to each other, and consequently, the local applications are synchronized as well.

For system synchronization all slaves are synchronized to one Reference Clock. Typically, the first ESC with Distributed Clocks capability after the master within one segment holds the reference time (System Time). This System Time is used as the reference clock to synchronize the DC slave clocks of other devices and of the master. The propagation delays, local clock sources drift, and local clock offsets are taken into account for the clock synchronization.

The ESCs can generate SyncSignals for local applications to be synchronized to the EtherCAT System Time. SyncSignals can be used directly (e.g., as interrupts) or for Digital Output updating/Digital Input sampling. Additionally, LatchSignals can be time stamped with respect to the EtherCAT System Time.

Definition of the System Time:

- Beginning on January, 1st 2000 at 0:00h
- Base unit is 1 ns
- 64 bit value (enough for more than 500 years)
- Lower 32 bits span over 4.2 seconds (typically enough for communication and time stamping). Some ESCs only have 32 bit DCs, which are compatible with 64 bit DCs.

Definition of the Reference Clock:

One EtherCAT device will be used as a Reference Clock. Typically, the Reference Clock is the first ESC with DC capability between master and all the slaves to be synchronized (DC slaves). The Reference Clock might be adjusted to a global reference clock, e.g. to an IEEE 1588 grandmaster clock. The reference clock provides the System Time.

Each DC slave holds a copy of the Reference Clock, which is calculated from the local clock and the local offset. The Reference Clock has a local clock, too.

Definition of the Master Clock:

The Reference Clock is typically initialized by the EtherCAT master using the master clock to deliver the System Time according to the System Time definition. The EtherCAT master clock is typically bound to a global clock reference (RTC or the master PC, IEEE1588, GPS, etc.), which is either directly available to the master or indirectly by an EtherCAT slave providing access to the reference.

Propagation Delay:

The propagation delay between Reference Clock and slave clock has to be taken into account when the System Time is distributed to the slaves.

Offset:

The offset between local clock and Reference Clock has two reasons: the propagation delay from the ESC holding the Reference Clock to the device with the slave clock, and initial differences of the local times resulting from different times at which the ESCs have been powered up. This offset is compensated locally in each slave. The ESC holding the Reference Clock derives the System Time from its local time by adding a local offset. This offset represents the difference between local time (started at power-up) and master time (starting on January, 1st 2000 at 0:00h).

Drift:

Since Reference Clock and DC slaves are typically not sourced by the same clock source (e.g., a quartz), their clock sources are subject to small deviations of the clock periods. The result is that one clock is running slightly faster than the other one, their local clocks are drifting apart.

55.3.17.1.1 ESC classification regarding DC support

Three classes of ESCs are distinguished regarding Distributed Clocks support:

- Slaves supporting only propagation delay measurement: Mandatory for ESCs with 3 or more ports. Local clock and receive time stamps are supported.
- Slaves without Distributed Clocks support: Slaves with max. 2 ports do not have to support DC features. Processing/ forwarding delay of such slaves is treated like a wire delay by the surrounding DC capable slaves.

55.3.17.2 Propagation delay measurement

Since each slave introduces a small processing/forwarding delay in each direction (within the device and also in the PHY), as well as the cable between the ESCs has a delay, the propagation delay between Reference Clock and the respective slave clock has to be considered for the synchronization of the slave clocks.

1. For measuring the propagation delay, the master sends a broadcast write to register DC Receive Time Port 0 (at least the first byte).
2. Each slave device stores the time of its local clock when the first bit of the Ethernet preamble of the frame was received, separately for each port (Receive Time Port 0-3 registers).
3. The master reads all time stamps and calculates the delay times with respect to the topology. The delay time between Reference Clock and an individual slave is written to the slave's System Time Delay register (0x0928:0x092B).

The receive time registers are used to sample the receive time of a specific frame (a broadcast write to Receive Time Port 0 register).

The clocks must not be synchronized for the delay measurement, only local clock values are used. Since the local clocks of the slaves are not synchronized, there is no relation between the Receive Times of different slaves. So the propagation delay calculation has to be based on receive time differences between the ports of a slave.

Devices with two ports do not need to support Distributed Clocks at all, their delay is treated as an additional "wire delay" between the surrounding DC-capable slaves. Devices with more than 2 ports have to support at least propagation delay measurements (DC Receive Times).

Some eCATs use the broadcast write to Receive Time Port 0 register as an indicator to latch the receive times of the next frame at all ports other than port 0 (if port 0 is open). Thus, another frame which is still traveling around the ring might trigger the measurement, and the receive times do no correlate. For these eCATs, the ring has to be empty before the broadcast write is issued.

Table 813. Registers for propagation delay measurement

Register Address	Name	Description
0x0900:0x0903	Receive Time Port 0	Local time when receiving frame on Port 0
0x0904:0x0907	Receive Time Port 1	Local time when receiving frame on Port 1
0x0908:0x09B	Reserved	Reserved
0x090C:0x090F	Reserved	Reserved
0x0918:0x091F	Receive Time ECAT Processing Unit	Local time when receiving frame at the ECAT Processing Unit
0x0936	Receive Time Latch Mode	ET1200: Receive time latching in forwarding or reverse mode

55.3.17.2.1 Propagation delay measurement in reverse mode

For redundancy operation, it is necessary to perform propagation delay measurements either in forwarding mode (master connected to port 0), or in reverse mode (master connected to port 1). For ET1200, care has to be taken because the measurement principle is slightly different for these two ESC types: DC Receive Time Latch Mode register 0x0936 has to be used. As ET1100 and IP Core do not require register 0x0936, the following rules can be used for propagation delay measurement in a mixed ET1200/ET1100/IP Core environment (ESC20 does not support propagation delay measurement in reverse mode).

- In forwarding mode, the master should write 0x00 to register 0x0936 of all slaves, and perform the delay measurement afterwards.
- In reverse mode, the master should write 0x01 to register 0x0936 of all slaves, and perform the delay measurement afterwards.

55.3.17.2.2 Propagation delay measurement example

The propagation delay between the local device and the Reference Clock device is calculated for the network example shown in the below figure. The example assumes that slave A is the Reference Clock. The loops of slave D and F are closed internally. The wire delays are assumed to be symmetrical, and the processing and forwarding delays are assumed to be identical for all ESCs.

Parameters used for propagation delay calculation are listed in the below table.

Table 814. Parameters for propagation delay calculation

Parameter	Description
t_{Px}	Processing delay of slave x (through EtherCAT Processing Unit, x=A-F)
t_{Fx}	Forwarding delay of slave x (alongside EtherCAT Processing Unit, x=A-F)
t_{xy}	Propagation delay from slave x to slave y (x/y=A-F)
t_{Wxy}	Wire propagation delay between slaves x and y (assumed to be symmetrical in both directions, x/y=A-F)
t_{x0}, t_{x1}, t_{x2}	Receive Time Port 0/1/2 values of slave x (time when first preamble bit is detected, x=A-F), measured with a write access to DC Receive Time 0 register.
t_P	Processing delay (through EtherCAT Processing Unit) if all slaves are identical

Table continues on the next page...

Table 814. Parameters for propagation delay calculation (continued)

t_F	Forwarding delay (alongside EtherCAT Processing Unit) if all slaves are identical
t_{Diff}	Difference between Processing delay and forwarding delay tDiff = tP – tF if all slaves are identical. ESC specific information, part of the ESI.
t_{Ref_x}	Propagation delay from Reference Clock (slave A) to slave x

- **Propagation delay between Slave C and D**

  The propagation delays between slave C and D (t_{CD} and t_{DC}) consist of a processing delay and the wire delay:

  $t_{CD} = t_{PC} + t_{WCD}$ $t_{DC} = t_{PD} + t_{WCD}$

  Assuming that the processing delays of slave C and D are identical ($t_P = t_{PC} = t_{PD}$): $t_{CD} = t_{DC} = t_P + t_{WCD}$

  The two Receive Times of slave C have the following relation: $t_{C1} = t_{C0} + t_{CD} + t_{DC}$

  So the propagation delays between slave C and D are $t_{CD} = t_{DC} = (t_{C1} - t_{C0}) / 2$

- **Propagation delay between Slave B and C**

  The propagation delays between slave B and C (t_{BC} and t_{CB}) are calculated as follows: $t_{BC} = t_{PB} + t_{WBC}$

  $t_{CB} = t_{FC} + t_{WBC}$

  Assuming that the processing delays of slaves B, C and D are identical ($t_P = t_{PB} = t_{PC} = t_{PD}$), and the difference between forwarding and processing delay of slave C is $t_{Diff} = t_{PC} - t_{FC}$:

  $t_{BC} = t_P + t_{WBC}$

  $t_{CB} = t_{BC} - t_{Diff}$

  The Receive Times (port 0 and 1) of slave B have the following relation: $t_{B1} = t_{B0} + t_{BC} + t_{CD} + t_{DC} + t_{CB}$

  So the propagation delay between slave B and C is $2*t_{BC} - t_{Diff} = (t_{B1} - t_{B0}) - (t_{C1} - t_{C0})$

  $t_{BC} = ((t_{B1} - t_{B0}) - (t_{C1} - t_{C0}) + t_{Diff}) / 2$

  And for the other direction:

  $t_{CB} = ((t_{B1} - t_{B0}) - (t_{C1} - t_{C0}) - t_{Diff}) / 2$

- **Propagation delays between slave E and F**

  The propagation delays between slave E and F are calculated like the delays between slave C and D: $t_{EF} = t_{PE} + t_{WEF}$

  $t_{FE} = t_{PF} + t_{WEF}$

  Assuming that the processing delays of slave E and F are identical ($t_P = t_{PE} = t_{PF}$): $t_{EF} = t_{FE} = (t_{E1} - t_{E0}) / 2$

- **Propagation delay between Slave B and E**

  The propagation delays between slave B and E (t_{BE} and t_{EB}) are calculated as follows: $t_{BE} = t_{FB} + t_{WBE}$

  $t_{EB} = t_{FE} + t_{WBE}$

  Assuming that the processing delays of slaves B to F are identical ($t_P = t_{Px}$), and the difference between forwarding and processing delay of these slaves is $t_{Diff} = t_{Px} - t_{Fx}$:

  $t_{BE} = t_{EB} = t_P - t_{Diff} + t_{WBE}$

  The Receive Times Port 1 and 2 of slave B have the following relation: $t_{B2} = t_{B1} + t_{BE} + t_{EF} + t_{FE} + t_{EB}$

  So the propagation delay between slave B and E is $2*t_{BE} = (t_{B2} - t_{B1}) - t_{EF} - t_{FE}$

  $t_{BE} = t_{EB} = ((t_{B2} - t_{B1}) - (t_{E1} - t_{E0})) / 2$

- **Propagation delay between slave A and B**

The propagation delays between slave A and B are calculated as follows: $t_{AB} = t_{PA} + t_{WAB}$

$t_{BA} = t_{FB} + t_{WAB}$

Assuming that the processing delays of all slaves are identical (tP = tPx), and the difference between forwarding and processing delay of these slaves is tDiff = tPx - tFx :

$t_{AB} = t_{P} + t_{WAB}$

$t_{BA} = t_{AB} - t_{Diff}$

The Receive Times of slave A have the following relation: $t_{A1} = t_{A0} + t_{AB} + (t_{B1} - t_{B0}) + (t_{B2} - t_{B1}) + t_{BA}$

So the propagation delay between slave A and B is $t_{AB} = ((t_{A1} - t_{A0}) - (t_{B2} - t_{B0}) + t_{Diff}) / 2$

And for the other direction:

$t_{BA} = ((t_{A1} - t_{A0}) - (t_{B2} - t_{B0}) - t_{Diff}) / 2$

- **Summary of Propagation Delay Calculation between Slaves**

  $t_{AB} = ((t_{A1} - t_{A0}) - (t_{B2} - t_{B0}) + t_{Diff}) / 2$ $t_{BA} = ((t_{A1} - t_{A0}) - (t_{B2} - t_{B0}) - t_{Diff}) / 2$ $t_{BC} = ((t_{B1} - t_{B0}) - (t_{C1} - t_{C0}) + t_{Diff}) / 2$ $t_{CB} = ((t_{B1} - t_{B0}) - (t_{C1} - t_{C0}) - t_{Diff}) / 2$ $t_{CD} = t_{DC} = (t_{C1} - t_{C0}) / 2$

  $t_{EF} = t_{FE} = (t_{E1} - t_{E0}) / 2$

  $t_{BE} = t_{EB} = ((t_{B2} - t_{B1}) - (t_{E1} - t_{E0})) / 2$

- **Propagation Delays between Reference Clock and Slave Clocks**

  The System Time Delay register of each slave clock takes the propagation delay from the Reference Clock to the slave. This delay is calculated like this:

  $t_{Ref\_B} = t_{AB}$

  $t_{Ref\_C} = t_{AB} + t_{BC}$

  $t_{Ref\_D} = t_{AB} + t_{BC} + t_{CD}$

  $t_{Ref\_E} = t_{AB} + t_{BC} + t_{CD} + t_{DC} + t_{CB} + t_{BE}$

  $t_{Ref\_F} = t_{AB} + t_{BC} + t_{CD} + t_{DC} + t_{CB} + t_{BE} + t_{EF}$

55.3.17.3 Offset compensation

The local time of each device is a free-running clock which typically will not have the same time as the Reference Clock. To achieve the same absolute System Time in all devices, the offset between the Reference Clock and every slave device's clock is calculated by the master. The offset time is written to register System Time Offset to adjust the local time for every individual device. Small offset errors are eliminated by the drift compensation after some time, but this time might become extremely high for large offset errors – especially for 64 bit DCs.

Each DC slave calculates its local copy of the System time using its local time and the local offset value:

$t_{Local\ copy\ of\ System\ Time} = t_{Local\ time} + t_{Offset}$

This time is used for SyncSignal generation and time stamping of LatchSignals. It is also provided to the PDI for use by µControllers.

The System Time of the Reference Clock is bound to the master clock by calculating the difference and compensating it using the System Time Offset of the Reference Clock.

Registers used for Offset Compensation are listed in below table.

Table 815. Registers for offset compensation

Register Address	Name	Description

Table continues on the next page...

Table 815. Registers for offset compensation (continued)

0x0910:0x0917	System Time	Local copy of System Time (read from PDI)
0x0918:0x091F	Receive Time ECAT Processing Unit	Local Time (tLocal time)
0x0920:0x0927	System Time Offset	Difference between local time and System Time (tOffset)

55.3.17.4 Resetting the Time Control Loop

Before starting drift compensation, the internal filters of the Time Control Loop must be reset. Their current status is typically unknown, and they can have negative impact on the settling time. The filters are reset by writing the Speed Counter Start value to the Speed Counter Start register (0x0930:0x0931). Writing the current value of the register again is sufficient to reset the filters.

Registers used for resetting the Time Control Loop filters are listed in below table.

Table 816. Registers for Resetting the Time Control Loop

Register Address	Name	Description
0x0930:0x0931	Speed Counter Start	Bandwidth for adjustment of local copy of System Time Writing a value resets the internal filters.

55.3.17.5 Drift compensation

After the delay time between the Reference Clock and the slave clocks has been measured, and the offset between both clocks has been compensated, the natural drift of every local clock (emerging from quartz variations between the Reference Clock's quartz and the local quartz) is compensated by the time control loop which is integrated into each ESC.

For drift compensation, the master distributes the System Time from the Reference Clock to all slave clocks periodically. The ARMW or FRMW commands can be used for this purpose. The time control loop of each slave takes the lower 32 bit of the System Time received from the Reference Clock and compares it to its local copy of the System Time. For this difference, the propagation delay has to be taken into account:

Δt = (tLocal time + tOffset – tPropagation delay) – tReceived System Time

If Δt is positive, the local time is running faster than the System time, and has to be slowed down. If Δt is negative, the local time is running slower than the System time, and has to speed up. The time control loop adjusts the speed of the local clock.

For a fast compensation of the static deviations of the clock speeds, the master should initially send many ARMW/FRMW commands (e.g. 15,000) for drift compensation in separate frames after initialization of the propagation delays and offsets. The control loops compensate the static deviations and the distributed clocks are synchronized. Afterwards, the drift compensation frames are sent periodically for compensation of dynamic clock drifts.

NOTE: The System Time Offset allows fast compensation of differences between local copy of the system time and the System Time, the drift compensation is very slow. Thus, shortly before drift compensation is started, the offset should be roughly compensated using the System Time Offset register. Otherwise settling time might become very high.

NOTE: Δt must not exceed 230 ns (~ 1 second), otherwise stability is not guaranteed. For fast settling times, Δt should be as low as possible.

55.3.17.6 Reference between DC registers/functions and clocks

Table 817. Reference between DC registers/functions and clocks

Register Address	Name	Referring to clock

Table continues on the next page...

Table 817. Reference between DC registers/functions and clocks (continued)

0x0900:0x090F	Receive Time Port n	Local Time ($t_{Local\ time}$)
0x0918:0x091F	Receive Time ECAT Processing Unit	Local Time ($t_{Local\ time}$)
0x0910:0x0917	System Time	Local copy of System Time ($t_{Local\ time} + t_{Offset}$)
0x0990:0y0997	SYNC0 Start Time	Local copy of System Time ($t_{Local\ time} + t_{Offset}$)
0x0998:0x099F	NEXT SYNC1 Pulse	Local copy of System Time ($t_{Local\ time} + t_{Offset}$)
0x09B0:0x09B7	Latch0 Time Positive Edge	Local copy of System Time ($t_{Local\ time} + t_{Offset}$)
0x09B8:0x09BF	Latch0 Time Negative Edge	Local copy of System Time ($t_{Local\ time} + t_{Offset}$)
0x09C0:0x09C7	Latch1 Time Positive Edge	Local copy of System Time ($t_{Local\ time} + t_{Offset}$)
0x09C8:0x09CF	Latch1 Time Negative Edge	Local copy of System Time ($t_{Local\ time} + t_{Offset}$)
0x09F0:0x09F3	EtherCAT Buffer Change Event Time	Local Time ($t_{Local\ time}$)
0x09F8:0x09FB	PDI Buffer Start Event Time	Local Time ($t_{Local\ time}$)
0x09FC:0x09FF	PDI Buffer Change Event Time	Local Time ($t_{Local\ time}$)

55.3.17.7 When is synchronization established?

There are two possibilities to detect if DC synchronization of a slave is established:

- Read System Time Difference (0x92C:0x092F):

If the difference is below an application specific threshold, DC has locked.

Advantage: Can be read using a single BRD command for the entire network: if the upper N bits are zero, synchronization is established.

Recommended if an EtherCAT slave is the reference clock. If the master is the reference clock, the threshold has to be increased to accomplish for the master jitter, which could make this solution unusable.

- Read Speed Counter Difference (0x0932:0x0933):

If the value is stable (within an application-specific range), DC has locked. Disadvantage: Loss of lock is recognized late.

55.3.17.8 Clock synchronization initialization example

The initialization procedure of clock synchronization including propagation delay measurement, offset compensation, filter reset, and drift compensation is shown in the following. After initialization, all DC slaves are synchronized with the Reference Clock.

1. Master reads the DL Status register of all slaves and calculates the network topology.
2. Master sends a broadcast write to Receive Time Port 0 register (at least first byte). All slaves latch the local time of the first preamble bit of this frame at all ports and at the ECAT Processing Unit. Some ESCs need the EtherCAT network to be free of frames before the broadcast write is sent.
3. Master waits until the broadcast write frame has returned.
4. Master reads all Receive Time Port 0-3 registers (depending on the topology and the Receive Time ECAT Processing Unit register (0x0918:0x091F) which contains the upper 32 bits of the receive times.

5. Master calculates individual propagation delays and writes them to System Time Delay registers of the slaves. Possible overruns of the 32 bit Receive Times have to be checked and taken into account.
6. Master sets System Time Offset register of the Reference Clock so that the Reference Clock is bound to the master time. The offset for the Reference Clock is master time minus Receive Time ECAT Processing Unit (local time) of the Reference Clock.
7. Master calculates System Time offsets for all DC slaves and writes them to the System Time Offset registers. The offset of each slave is Receive Time ECAT Processing Unit from Reference Clock minus Receive Time ECAT Processing Unit from each DC slave.
8. Master resets all slaves' Time Control Loop filters by writing to the Speed Counter Start register (0x0930:0x0931).
9. For static drift compensation, the master sends many separate ARMW or FRMW drift compensation frames (e.g., 15,000 frames) to distribute the System Time of the Reference Clock to all DC slaves.
10. For dynamic drift compensation, the master sends ARMW or FRMW commands periodically to distribute the System Time of the Reverence Clock to all DC slaves. The rate of the drift compensation commands depends on the acceptable maximum deviation.

55.3.17.9 SyncSignals and LatchSignals

ESCs with Distributed Clocks support generation of SyncSignals and time stamping of LatchSignals. The SyncSignals can be used internally for

- Interrupt generation (mapping to AL Event Request register 0x0220:0x0223 and PDI IRQ)
- PDI Digital Output Update events
- PDI Digital Input Latch events

The SyncSignals can also be directly mapped to output signals (SYNC[1:0]) for use by external devices, e.g., as interrupt signals (less jitter than PDI IRQ, no interrupt source decoding).

The Latch Event unit supports time stamping of up to two LatchSignals (LATCH[1:0], rising and falling edge separately), and time stamping of SyncManager events for debugging purposes.

55.3.17.9.1 Interface

The Distributed Clocks unit has the following external signals (depending on the ESC and the ESC configuration):

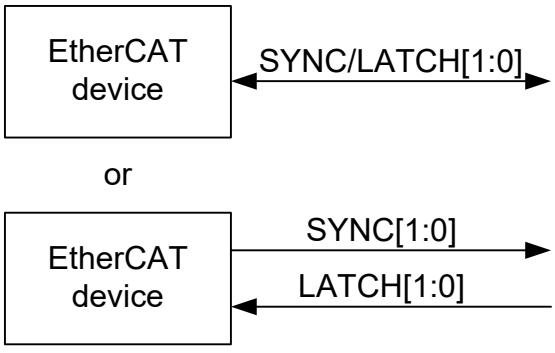


Figure 391. Distributed clocks signals

Table 818. Distributed clocks signals

Signal	Direction	Description
SYNC/LATCH[1:0]	IN/OUT	Combined SyncSignals / LatchSignals

Table continues on the next page...

Table 818. Distributed clocks signals (continued)

Signal	Direction	Description
SYNC[1:0]	OUT	SyncSignals (also named SYNC0/ SYNC1)
LATCH[1:0]	IN	LatchSignals (also named LATCH0/ LATCH1)

Not all of these signals might be available depending on the ESC and its hardware configuration.

55.3.17.9.2 Configuration

The mapping of Distributed Clocks SyncSignals and LatchSignals to the external SYNC/LATCH[1:0] signals is controlled by the setting of the Sync/Latch PDI Configuration register 0x0151. The SYNC[1:0] driver characteristics are also selected in this register. The SyncSignals are internally available for interrupt generation and Digital I/O synchronization regardless of the Sync/Latch PDI Configuration. The mapping of SyncSignals to the AL Event Request register is also controlled by the Sync/Latch PDI Configuration register 0x0151.

The length of a SyncSignal pulse is defined in the DC Pulse Length of SYNC Signals register (0x0982:0x0983). A value of 0 selects acknowledged modes.

Some ESCs support power saving options (partly disabling DC units) controlled by two bits of the ESC Configuration register (0x0141[3:2]), others have individual configuration options for each SyncSignal/LatchSignal.

The Sync/Latch signals are not driven (high-impedance) by some ESCs until the SII EEPROM is successfully loaded. Take care of proper SyncSignal usage while the EEPROM is not loaded (e.g. pull-down/pull-up resistors).

55.3.17.9.3 SyncSignal generation

The DC Cyclic Unit / Sync Unit supports the generation of a base SyncSignal SYNC0 and a dependent SyncSignal SYNC1. The SyncSignals can both be used internally and externally of the ESC. SyncSignals can be generated at a specific System Time. Four operation modes are supported: cyclic generation, single shot, cyclic acknowledge, and single shot acknowledge mode. The acknowledged modes are typically used for interrupt generation. The interrupts have to be acknowledged by a µController.

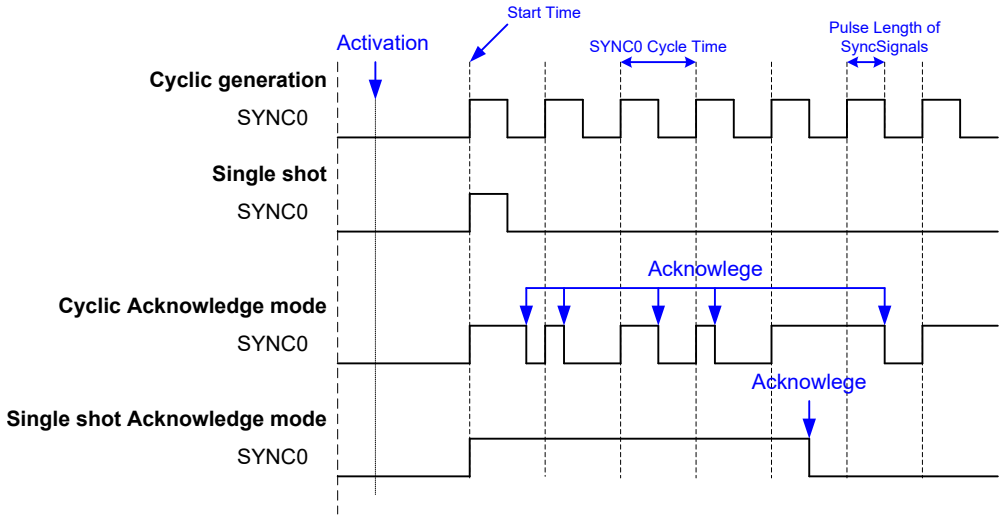


Figure 392. SyncSignal Generation modes

The SyncSignal operation mode is selected by the configuration of the Pulse Length and the SYNC0 Cycle Time, according to the following table:

Table 819. SyncSignal Generation mode selection

Pulse Length of SYNC Signals (0x0982:0x0983)	SYNC0 Cycle Time (0x09A0:0x09A3)	
	> 0	= 0
> 0	Cyclic Generation	Single Shot
= 0	Cyclic Acknowledge	Single Shot Acknowledge

The cycle time of the SYNC0 signal is configured in the SYNC0 Cycle Time register (0x09A0:0x09A3), the start time is set in the Start Time Cyclic Operation register (0x0990:0x0997). After the Sync Unit is activated and the output of the SYNC0/1 signals is enabled (DC Activation register 0x0981), the Sync Unit waits until the start time is reached and generates the first SYNC0 pulse.

Some ESCs support additional activation options like auto-activation when the Start Time is written, or 64 bit extension if only 32 bit of the Start Time is written. Other options are to detect invalid Start Times and provide debug output of SyncSignals.

Internally, the SyncSignals are generated with an update rate of 100 MHz (10 ns update cycle). The jitter of the internal SyncSignal generation in comparison to the local copy of the System Time is 12 ns.

The registers used for SyncSignal Generation are shown in below table.

Table 820. Registers for SyncSignal generation

Register Address	Name	Description
0x0141[3:2]	ESC Configuration	Enable/Disable DC Units (power saving)
0x0151	Sync/Latch PDI Configuration	Configuration of SYNC/LATCH[1:0] pins
0x0980[0]	Cyclic Unit Control	Assignment of cyclic function to EtherCAT or PDI
0x0981	Activation	Activation of cyclic function and SYNC pins
0x0982:0x0983	Pulse Length of SYNC signals	Length of SYNC impulse length
0x0984	Activation Status	Activation status of SYNC0/SYNC1
0x098E	SYNC0 Status	Status of SYNC0 signal
0x098F	SYNC1 Status	Status of SYNC1 signal
0x0990:0y0997	SYNC0 Start Time	Start System time of cyclic operation
0x0998:0x099F	NEXT SYNC1 Pulse	System Time of next Sync1 Pulse
0x09A0:0x09A3	SYNC0 Cycle Time	Cycle Time of SYNC0
0x09A4:0x09A7	SYNC1 Cycle Time	Cycle Time of SYNC1

Some of these registers are set via SII EEPROM/IP Core configuration, or they are not available in specific ESCs.

55.3.17.9.3.1 Cyclic generation

In Cyclic Generation mode, the Sync unit generates isochronous SyncSignals after the Start Time. The generation ends if the Cyclic Unit is deactivated or SYNC0/1 generation is deactivated. The Cycle times are determined by the SYNC0/1 Cycle Time registers. The Pulse Length of the SYNC signals has to be greater than 0. If the Pulse Length is greater than the Cycle Time, the SyncSignal will always be activated after the Start Time.

55.3.17.9.3.2 Single Shot mode

In Single Shot mode (SYNC0 Cycle Time set to 0), only one SyncSignal pulse is generated after the Start Time is reached. Another pulse can only be generated by deactivating the Cyclic Unit (0x0981[0]=0), reprogramming the Start Time, and reactivation of the Cyclic Unit.

55.3.17.9.3.3 Cyclic Acknowledge mode

The Cyclic Acknowledge mode is typically used for generation of isochronous interrupts. The acknowledged modes are selected by setting the Pulse Length of SYNC Signals to 0 (0x0982:0x0983). Each SyncSignal pulse remains active until it is acknowledged – typically by a µController – by reading the appropriate SYNC0 or SYNC1 Status register (0x098E, 0x098F). The first pulse is generated after the Start Time is reached, following pulses are generated when the next regular SYNC0/1 event would occur.

55.3.17.9.3.4 Single Shot Acknowledge mode

In Single Shot Acknowledge mode (both Pulse Length of SYNC Signals and SYNC0 Cycle Time are 0), only one pulse is generated when the Start Time is reached. The pulse remains active until it is acknowledged by reading the appropriate SYNC0/1 Status registers. Another pulse can only be generated by deactivating the Cyclic Unit (0x0981[0]=0), reprogramming the Start Time, and reactivation of the Cyclic Unit.

55.3.17.9.3.5 SYNC1 generation

The second SyncSignal (SYNC1) depends on SYNC0, it can be generated with a predefined delay after SYNC0 pulses. The delay is configured in the SYNC1 Cycle Time register (0x09A4:0x09A7).

If the SYNC1 Cycle Time is larger than the SYNC0 Cycle Time, it will be generated as follows: when the Start Time Cyclic Operation is reached, a SYNC0 pulse is generated. The SYNC1 pulse is generated after the SYNC0 pulse with a delay of SYNC1 Cycle Time. The next SYNC1 pulse is generated when the next SYNC0 pulse was generated, plus the SYNC1 Cycle Time.

Some example configurations are shown in the following figure:

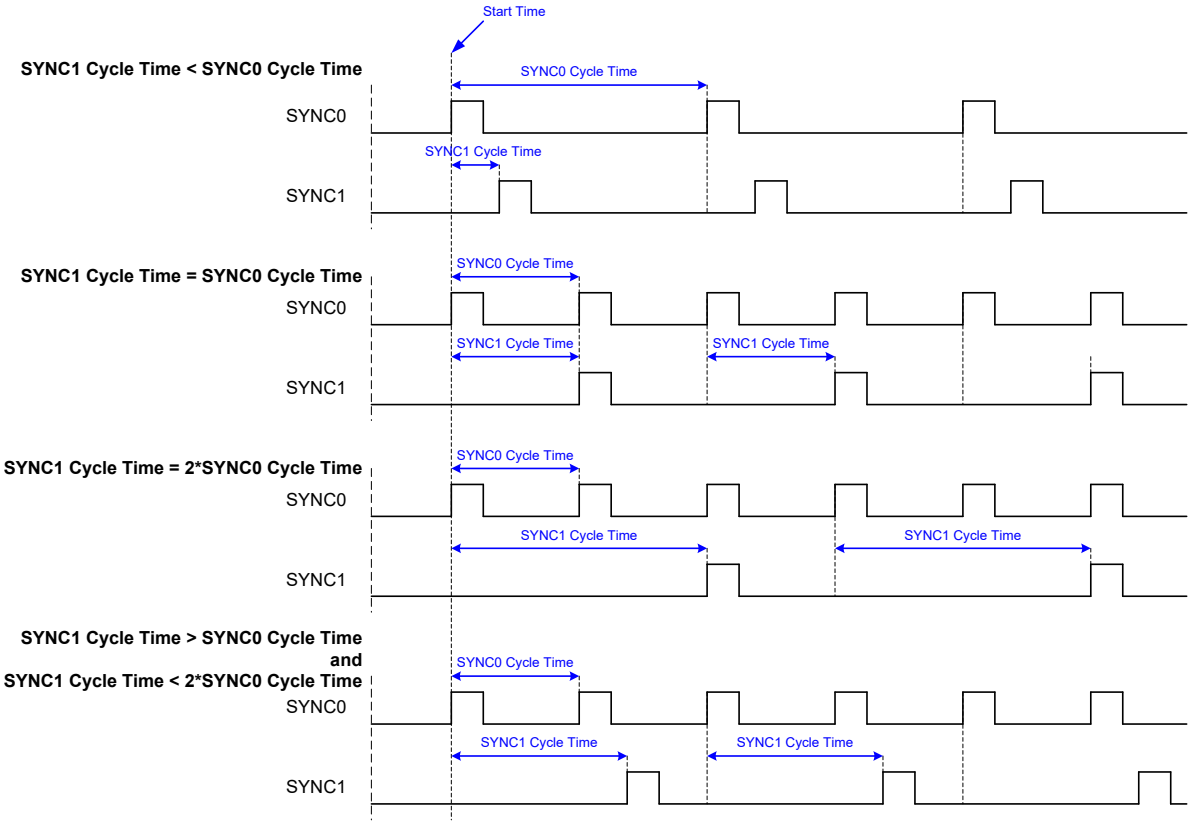


Figure 393. SYNC0/1 cycle time examples

NOTE: If The SYNC1 Cycle Time is 0, SYNC1 reflects SYNC0.

55.3.17.9.3.6 SyncSignal initialization example

The SyncSignal generation is initialized with the following procedure:

1. Enable DC SYNC Out Unit in ESC Configuration register (0x0141[2]=1; specific ESCs only)
2. Set SYNC/Latch PDI Configuration register (0x0151, initialized by SII EEPROM) to SYNC0/1 output with appropriate driver settings.
3. Set Pulse Length register (0x0982:0x0983, initialized by EEPROM) to pulse length of SYNC signals. Select a value > 0 ns for cyclic repetition of the SyncSignals
4. Assign Sync Unit to ECAT or PDI (0x0980, part of ESI)
5. Set cycle time of SYNC0 signal (0x09A0:0x09A3) and for SYNC1 signal (0x09A4:0x09A7)
6. Set Start Time of Cyclic Operation (0x0990:0x0997) to a time later than the time the cyclic generation will be activated (end of activation frame; e.g., read the System Time and add the time for writing Start Time and Activation). For 32 bit DCs, the SyncSignal generation will start at worst after a turn-over of the System Time (~ 4 s), but with 64 bit DCs, SyncSignal generation may start in hundreds of years.
7. Activate Cyclic Operation (0x0981[0]=1) to start cyclic generation of SyncSignals and activate SYNC0/1 generation (0x0981[2:1]=0x3). The Sync Unit waits until the Start Time of Cyclic Operation is reached for the generation of the first SYNC0 pulse.

Register Start Time of Cyclic Operation and register Next SYNC1 pulse can be read to get the time of the next output event. In the acknowledged modes, the Sync0/1 Status registers (0x098E:0x098F) give the status of the SyncSignals. The SyncSignals are acknowledged by reading the SYNC0/1 Status registers.

55.3.17.9.4 LatchSignals

The DC Latch Unit enables time stamping of LatchSignal events for two external signals, LATCH0 and LATCH1. Both rising edge and falling edge time stamps are recorded. Additionally, time stamping of SyncManager events is possible with some ESCs.

LatchSignals are sampled with a sample rate of 100 MHz, the corresponding time stamp has an internal jitter of 11 ns.

The state of the LatchSignals can be read from the Latch Status registers (0x09AE:0x09AF) – if supported by the ESC.

The DC Latch Unit supports two modes: single event or continuous mode, configured in the Latch0/1 Control registers (0x09A8:0x09A8).

The registers used for LatchSignal event time stamping are shown in below table.

Table 821. Registers for latch input events

Register Address	Name	Description
0x0141[3:2]	ESC Configuration	Enable/Disable DC Units (power saving)
0x0151	Sync/Latch PDI Configuration	Configuration of SYNC/LATCH[1:0] pins
0x0980[5:4]	Cyclic Unit Control	Assignment of cyclic function to EtherCAT or PDI
0x09A8	Latch0 Control	Latch unit configuration for Latch0
0x09A9	Latch1 Control	Latch unit configuration for Latch1
0x09AE	Latch0 Status	Latch status of Latch0
0x09AF	Latch1 Status	Latch status Latch1
0x09B0:0x09B7	Latch0 Time Positive Edge	System time at positive edge Latch0
0x09B8:0x09BF	Latch0 Time Negative Edge	System time at negative edge Latch0
0x09C0:0x09C7	Latch1 Time Positive Edge	System time at positive edge Latch1
0x09C8:0x09CF	Latch1 Time Negative Edge	System time at negative edge Latch1
0x09F0:0x09F3	EtherCAT Buffer Change Event Time	Local time at beginning of frame causing ECAT SyncManager buffer change event
0x09F8:0x09FB	PDI Buffer Start Event Time	Local time at PDI SyncManager buffer start event
0x09FC:0x09FF	PDI Buffer Change Event Time	Local time at PDI SyncManager buffer change event

Some of these registers are set via SII EEPROM/IP Core configuration, or they are not available in specific ESCs.

55.3.17.9.4.1 Single Event Mode

In single event mode, only the timestamps of the first rising and the first falling edge of the LatchSignals are recorded. The Latch Status registers (0x09AE:0x09AF) contain information about the events which already have occurred. The Latch Time registers (0x09B0 to 0x09CF) contain the time stamps.

Each event is acknowledged by reading the corresponding Latch Time register. After reading the Latch Time register, the Latch unit is waiting for the next event. Latch events are mapped to the AL Event Request register in single event mode.

55.3.17.9.4.2 Continuous Mode

In continuous mode, each event is stored in the Latch Time registers. At reading, the time stamp of the last event is read. The Latch Status registers (0x09AE:0x09AF) do not reflect the latch event states in continuous mode.

55.3.17.9.4.3 SyncManager Event

Some ESCs support debugging of SyncManager interactions with time stamps for buffer events. The last event can be read out at the SyncManager Event Time registers (0x09F0:0x09FF), if the SyncManager is configured appropriately.

55.3.17.9.4.4 ECAT or PDI Control

The SyncSignal unit and the two LatchSignal units of the Distributed Clocks entity can be assigned independently by the master to be controlled either by ECAT or a local µController (PDI) using the Cyclic Unit Control register 0x0980. With PDI control, a µController can e.g. set up cyclic interrupts for itself.

55.3.17.9.5 System time PDI controlled

Sometimes Distributed Clocks of different EtherCAT networks have to be synchronized. One solution is master-master communication, the other one is based on a physical device which is present in both EtherCAT networks. One of the networks contains the DC Reference Clock (DC source), the other one – DC destination – is synchronized to the Reference Clock in the DC source network.

Some ESCs support such a synchronization by a different functionality of the System Time register (0x0910:0x0913). In normal operation mode, a write access initiated by the EtherCAT master to the System Time register triggers the synchronization: the written value is compared to the local copy of the System Time, and the difference is fed into the control loop. If the System Time is PDI controlled, the PDI writes the System Time register, and the written value is compared to the DC Latch0 Time Positive Edge register (0x09B0:0x09B3). This feature makes the accuracy of the synchronization independent of the µController/PDI response times.

The following figures illustrate how the System Time is transferred from the DC source to the DC destination. ESC 1 and ESC 2 are located in different EtherCAT networks. The EtherCAT network of ESC 1 contains the Reference Clock, the network of ESC 2 will become synchronized to this Reference Clock. ESC 2 is the “reference clock” of its EtherCAT network. There are two options for synchronization, which has to be performed on a regular basis.

The first option is to let the µController generate a trigger pulse for ESC 1 and 2. The time of the rising edge is stored in the Latch0 Time Positive Edge register both in ESC 1 and 2. Afterwards, the µController reads this time from ESC 1 and writes it into the System Time register of ESC 2. The difference of the Latch0 times is used to feed the control loop

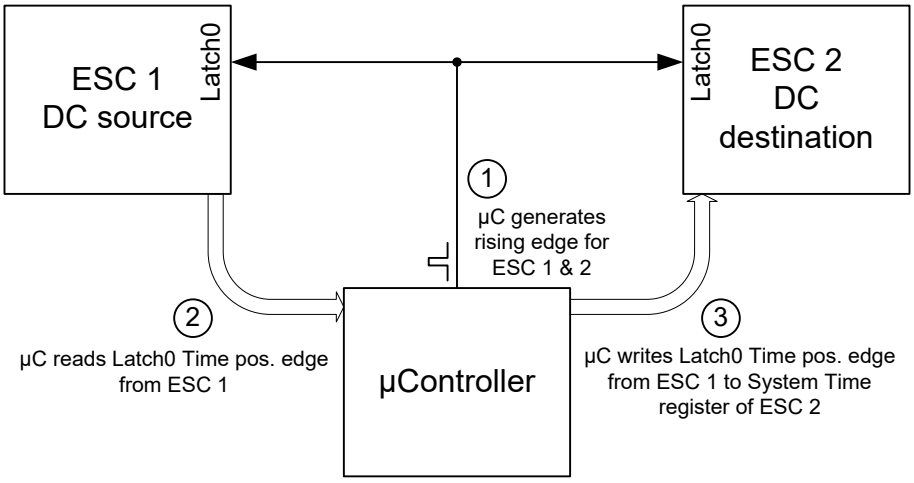


Figure 394. System time PDI controlled with three steps

The second option uses a SyncSignal output of ESC 1 to trigger Latch0 at ESC 2 and an interrupt at the µController. Upon receiving an interrupt, the µController writes the time of the SyncSignal pulse to the System Time register of ESC 2. The µController has to calculate the time of the SyncSignal based upon Start Time Cyclic Operation and SYNC Cycle Time configuration of ESC 1 from interrupt to interrupt. The advantage of the second solution is less communication, the disadvantages are more calculation overhead and error detection/troubleshooting.

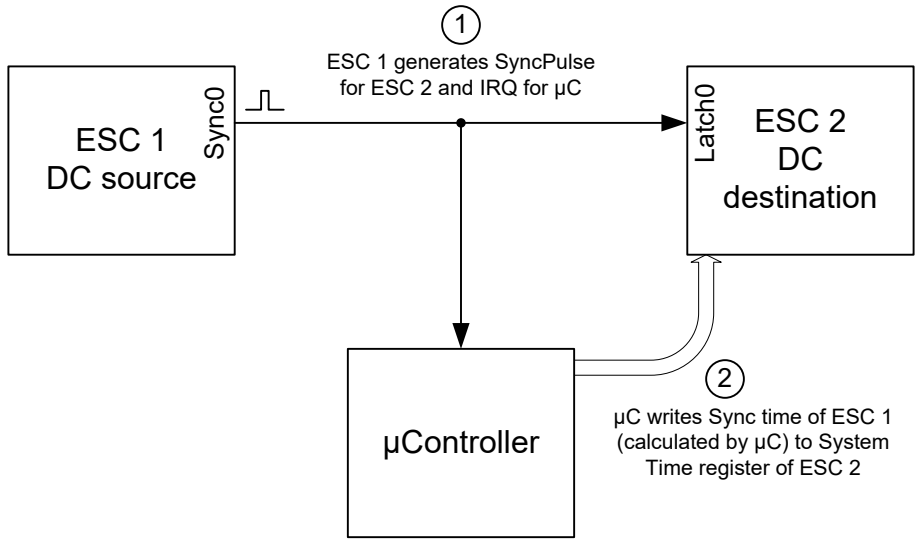


Figure 395. System time PDI controlled with two steps

55.3.17.9.6 Offset compensation

The local time of each device is a free-running clock which typically will not have the same time as the Reference Clock. To achieve the same absolute System Time in all devices, the offset between the Reference Clock and every slave device's clock is calculated by the master. The offset time is written to register System Time Offset to adjust the local time for every individual device. Small offset errors are eliminated by the drift compensation after some time, but this time might become extremely high for large offset errors – especially for 64 bit DCs.

Each DC slave calculates its local copy of the System time using its local time and the local offset value:

$t_{Local\ copy\ of\ System\ Time} = t_{Local\ time} + t_{Offset}$

This time is used for SyncSignal generation and time stamping of LatchSignals. It is also provided to the PDI for use by µControllers.

The System Time of the Reference Clock is bound to the master clock by calculating the difference and compensating it using the System Time Offset of the Reference Clock.

Registers used for Offset Compensation are listed in below table.

Table 822. Registers for offset compensation

Register Address	Name	Description
0x0910:0x0917	System Time	Local copy of System Time (read from PDI)
0x0918:0x091F	Receive Time ECAT Processing Unit	Local Time (tLocal time)
0x0920:0x0927	System Time Offset	Difference between local time and System Time (tOffset)

55.3.18 PDI selection and configuration

55.3.18.1 PDI selection and configuration

Typically, the PDI selection and configuration is part of the eCAT Configuration Area of the SII EEPROM. Some eCATs (e.g. IP Core) have the PDI selected and configured at power-on time. In this case, the eCAT Configuration Area should reflect the actual settings, although they are not evaluated by the eCAT itself. For the other eCATs, the PDI becomes active after the SII EEPROM is successfully loaded. All PDI pins are inactive (high impedance) until then (as well as the DC Sync/Latch signals). Some eCATs and PDIs provide an EEPROM_Loaded signal, which indicates that the EEPROM is successfully loaded and the PDI can be used. Attach a pull-down resistor to the EEPROM_Loaded pin, because it is also not driven (high impedance) until the EEPROM is successfully loaded. The PDI of an IP Core is active after reset is released, which enables e.g. EEPROM emulation by a µController. Take care of Digital Output signals and DC SyncSignals while the EEPROM is not loaded to achieve proper output behavior.

55.3.18.2 General purpose I/O

Some eCATs support general purpose inputs, outputs, or both. General Purpose I/O is much different from Digital I/O, because of missing consistency and security features, and no bit-write support.

55.3.18.2.1 General purpose inputs

The general purpose inputs are directly mapped into the General Purpose Input registers. Consistency of the general purpose inputs is not provided.

55.3.18.2.2 General purpose output

The general purpose output signals reflect the values of the General Purpose Output register without watchdog protection. The General Purpose Output register can be written both by ECAT and PDI. The general purpose outputs are intended e.g. for application specific LED outputs. General purpose outputs are updated at the end of an EtherCAT frame or at the end of a PDI access. Consistency of the general purpose outputs is not provided, and they are also not watchdog-secured. Additionally, they do not support bit-wise modification with FMMUs.

55.3.19 Clocking

Table 823. Primary core clocks

Clocks	Description
CLK25	period: 40ns Core clock (25MHz)
CLK50	period: 20ns phase shift: 0ns Clock reference: CLK25 Core clock(50MHz) for RMII only. Phase shift is rising edge to rising edge.
CLK100	period: 10ns phase shift: 0ns Clock reference: CLK25 Core clock(100MHz).Phase shift is rising edge to rising edge.
MII_RX_CLK0-1	period: 40ns

Table continues on the next page...

Table 823. Primary core clocks (continued)

Clocks	Description
	MII receive reference clock (25 MHz). IEEE802.3 requirement. Each clock is asynchronous to any other clock.
MII_TX_CLK0-1	Input clock. Transmit clock port 0 for automatic TX Shift configuration. Set to 0 for manual TX Shift configuration.
PDI_OPB_CLK	period: 40ns/N phase shift: 0ns Clock reference: CLK25 OPB bus clock. Multiple of 25 MHz (N=1, 2, 3, etc.). Phase shift is rising edge to rising edge.

55.3.20 Reset

This sections describes the reset signals of the block.

Table 824. Primary core reset

Signal	Clock Reference	Description
nRESET	none	Asynchronous reset input
EtherCAT_IPCore_inst/ nRESET_CHAIN[1]	CLK25	Internal reset, asynchronously asserted, synchronously deasserted.
PDI_EMULATION	CLK25	Disable internal register 0x0120 if PDI emulation is enabled. Write value is directly routed to 0x0130.

55.3.21 Interrupts

ESCs support two types of interrupts: AL Event Requests targeted at a µController, and ECAT event requests targeted at the EtherCAT master. Additionally, the Distributed Clocks SyncSignals can be used as interrupts for a µController as well.

The following table lists the module interrupts.

Table 825. Interrupt signals

Condition	Interrupt Signal Name	Direction	Description
OPB	PDI_OPB_IRQ	O	Interrupt

55.3.21.1 AL event request (PDI interrupt)

AL Event Requests can be signaled to a µController using the PDI Interrupt Request signal (IRQ/SPI_IRQ, etc.). For IRQ generation, the AL Event Request register (0x0220:0x0223) is combined with the AL Event Mask register (0x0204:0x0207) using a logical AND operation, then all resulting bits are combined (logical OR) into one interrupt signal. The output driver characteristics of the IRQ signal are configurable using the SYNC/LATCH PDI configuration register (0x0151). The AL Event Mask register allows for selecting the interrupts which are relevant for the µController and handled by the application.

The DC SyncSignals can be used for interrupt generation in two ways:

- The DC SYNC signals are mapped into the AL Event Request Register (configured with SYNC/LATCH PDI Configuration register 0x0151.3/7). In this case, all interrupts from the ESC to the µController are combined into one IRQ signal, and the Distributed Clocks LATCH0/1 inputs can still be used. The IRQ signal has a jitter of ~40 ns.
- The DC SyncSignals are directly connected to µController interrupt inputs. The µController can react on DC SyncSignal interrupts faster (without reading AL Request register), but it needs more interrupt inputs. The jitter of the SyncSignals is ~12 ns. The DC Latch functions are only available for one Latch input or not at all (if both DC SYNC outputs are used).

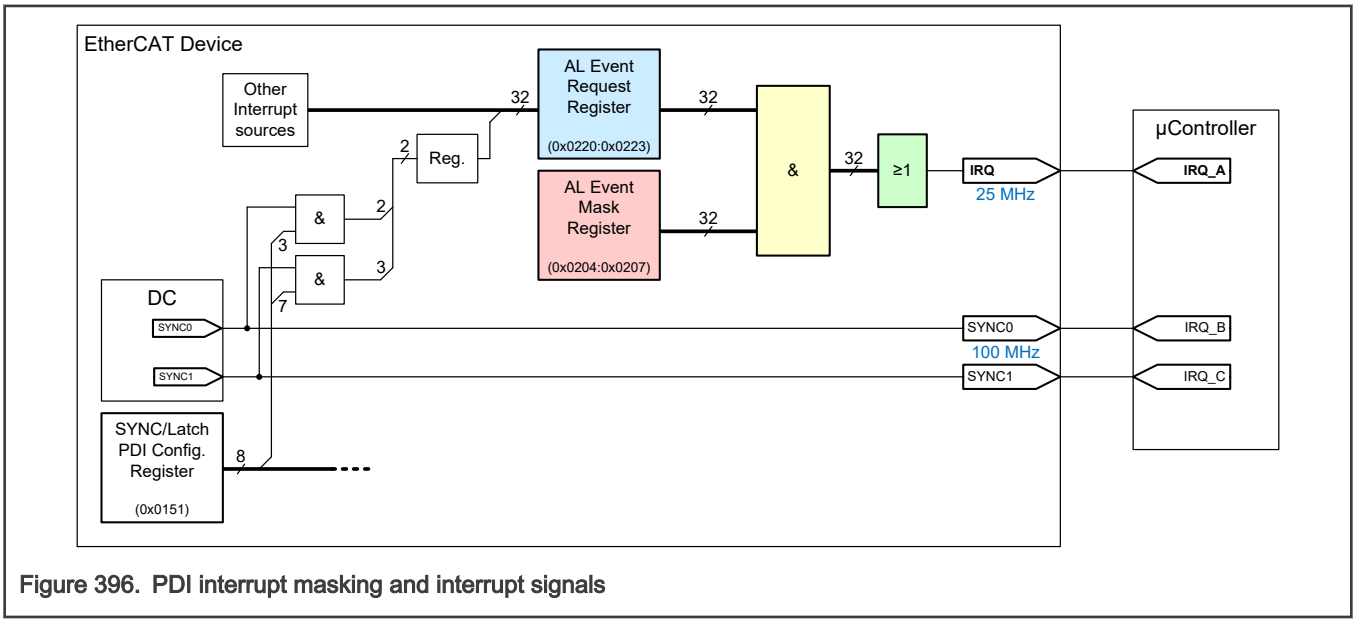


Figure 396. PDI interrupt masking and interrupt signals

Table 826. Registers for AL event request configuration

Name	Description
PDI Configuration (0x0150)	IRQ driver characteristics, depending on PDI
SYNC/LATCH PDI Configuration (0x0151)	Mapping DC SyncSignals to Interrupts
AL Event Mask (0x0204:0x0207)	Mask register
AL Event Request (0x0220:0x0223)	Pending Interrupts
SyncManager Control (0x0804 + N*8)	Mapping SyncManager Interrupts

55.3.21.2 ECAT event request (ECAT Interrupt)

ECAT event requests are used to inform the EtherCAT master of slave events. ECAT events make use of the IRQ field inside EtherCAT datagrams. The ECAT Event Request register (0x0210:0x0211) is combined with the ECAT Event Mask register (0x0200:0x0201) using a logical AND operation. The resulting interrupt bits are combined with the incoming ECAT IRQ field using a logical OR operation, and written into the outgoing ECAT IRQ field. The ECAT Event Mask register allows for selecting the interrupts which are relevant for the EtherCAT master and handled by the master application. NOTE: The master cannot distinguish which slave (or even more than one) was the origin of an interrupt.

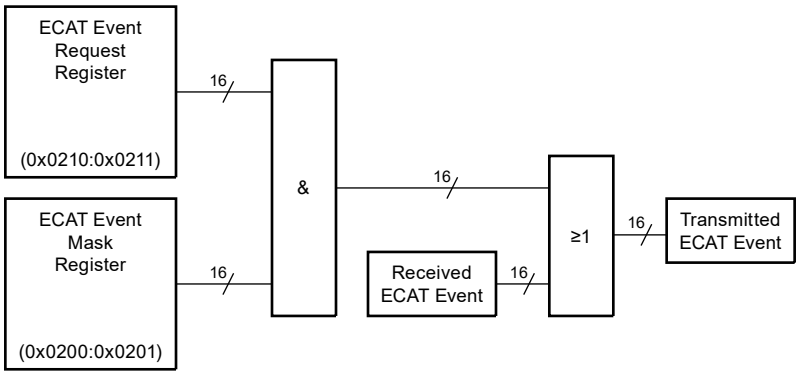


Figure 397. ECAT interrupt masking

Table 827. Registers for ECAT request configuration

Name (Register Address)	Description
ECAT Event Mask (0x0200:0x0201)	Mask Register
ECAT Event Request (0x0210:0x0211)	Pending Interrupts
SyncManager control (0x0804 + N*8)	Mapping SyncManager Interrupts

55.4 External signals

This sections deals with the external signals of the block.

55.4.1 Signal descriptions

Table 828. Signals

Signal			Description
ECAT_RX_CLK_0	RX clk	Input	Rx clk for ESC port0
ECAT_RX_DV_0	RX data valid signal	Input	RX data valid for port0
ECAT_RX_ER_0	RX data error signal	Input	RX data error for port0
ECAT_RX_DATA0_0	RX data 0	Input	RX data 0 for port0
ECAT_RX_DATA1_0	RX data 1	Input	RX data 1 for port0
ECAT_RX_DATA2_0	RX data 2	Input	RX data 2 for port0
ECAT_RX_DATA3_0	RX data 3	Input	RX data 3 for port0
ECAT_TX_DATA0_0	TX data 0	Output	TX data 0 for port0
ECAT_TX_DATA1_0	TX data 1	Output	TX data 1 for port0
ECAT_TX_DATA2_0	TX data 2	Output	TX data 2 for port0
ECAT_TX_DATA3_0	TX data 3	Output	TX data 3 for port0
ECAT_TX_CLK_0	TX clk of port 0	Input	TX clk of port 0

Table continues on the next page...

Table 828. Signals (continued)

Signal			Description
ECAT_TX_EN_0	Transmit enable	Output	Transmit enable for port0
ECAT_LINK_0	Link indictor	Input	Input signal provided by the PHY if a 100 Mbit/s (Full Duplex) link is established for port 0
ECAT_RX_CLK_1	RX clk	Input	Rx clk for ESC port1
ECAT_RX_DV_1	RX data valid signal	Input	RX data valid for port1
ECAT_RX_ER_1	RX data error signal	Input	RX data error for port1
ECAT_RX_DATA0_1	RX data 0	Input	RX data 0 for port1
ECAT_RX_DATA1_1	RX data 1	Input	RX data 1 for port1
ECAT_RX_DATA2_1	RX data 2	Input	RX data 2 for port1
ECAT_RX_DATA3_1	RX data 3	Input	RX data 3 for port1
ECAT_TX_DATA0_1	TX data 0	Output	TX data 0 for port1
ECAT_TX_DATA1_1	TX data 1	Output	TX data 1 for port1
ECAT_TX_DATA2_1	TX data 2	Output	TX data 2 for port1
ECAT_TX_DATA3_1	TX data 3	Output	TX data 3 for port1
ECAT_TX_CLK_1	TX clk of port 1	Input	TX clk of port 1
ECAT_TX_EN_1	Transmit enable	Output	Transmit enable for port1
ECAT_LINK_1	Link indicator	Input	Input signal provided by the PHY if a 100 Mbit/s (Full Duplex) link is established for port 1
RAM_RESET	RAM reset	Output	RAM reset
RAM_RD_ENA	RAM read enable	Output	RAM read enable
RAM_WR_DATA[7:0]	RAM write data	Output	RAM write data
RAM_RD_DATA[7:0]	RAM read data	Input	RAM read data
RAM_RD_ADR[15:0]	RAM read address	Output	RAM read address
RAM_WR_ADR[15:0]	RAM write address	Output	RAM write address
RAM_WR_ENA	RAM write enable	Output	RAM write enable

Table continues on the next page...

Table 828. Signals (continued)

Signal			Description
RMII_SEL[0]	port 0 selection	Input	0: Port0 is MII port 1: Port0 is RMII port
RMII_SEL[1]	port 1 selection	Input	0: Port1 is MII port 1: Port1 is RMII port
RESET_OUT	Reset out	Output	Reset by ECAT (reset register 0x0040), active high. RESET_OUT has to trigger nRESET, which clears RESET_OUT.
NRESET	Reset	Input	System hard reset
ips_rdata	IPS read data	Output	IPS read data
ips_xfr_err	IPS transfer error	Output	IPS transfer error
ips_addr	address	Input	IPS Address
ips_module_en	IPS module enable	Input	IPS module enable
ips_rwb	IPS read/ write	Input	IPS read/write selection
ipg_clk	IPS clk	Input	System Clk
CLK25	CLK25	Input	25 MHz clock signal from PLL (rising edge synchronous with rising edge of CLK100)
CLK50	CLK50	Input	50M clk input
CLK100	CLK100	Input	100M clk input
MDIO	Management Interface data (alias MDIO)	Bidir	Management Interface data (alias MDIO)
MDIO_DATA_IN	Management Interface data Input	Input	Management Interface data Input
MDIO_DATA_OUT	Management Interface data Output	Output	Management Interface data Output
MDIO_DATA_ENA	Management Interface data enable	output	Management Interface data enable
MCLK	Management Interface Clock	Output	Management Interface Clock
PHY_OFFSET	PHY_OFFSET	Input	PHY_OFFSET

Table continues on the next page...

Table 828. Signals (continued)

Signal			Description
PHY_OFFSET_VEC[4:0]	Ethernet PHY Address Offset	Input	Ethernet PHY Address Offset
PROM_CLK	EEPROM I2C Clock	Output	EEPROM I2C Clock
PROM_DATA	EEPROM I2C Data	InOut	EEPROM I2C Data
PROM_DATA_IN	EEPROM I2C Data Input	Input	EEPROM I2C Data Input
PROM_DATA_OUT	EEPROM I2C Data Output	Output	EEPROM I2C Data Output
PROM_DATA_ENA	EEPROM I2C Data Enable	Output	EEPROM I2C Data Enable
PROM_SIZE	EEPROM size configuration	Input	EEPROM size configuration
LINK_ACT	Link/Activity LED	Output	Link/Activity LED
DEV_STATE	LED State	Output	LED State
LED_RUN	RUN LED	Output	RUN LED
LED_ERR	RUN Error	Output	RUN Error
LED_STATE_RUN	RUN pin of dual-color STATE LEDs	Output	RUN pin of dual-color STATE LEDs
SYNC_OUT0	Distributed Clocks SyncSignal output0	Output	Distributed Clocks SyncSignal output0
SYNC_OUT1	Distributed Clocks SyncSignal output1	Output	Distributed Clocks SyncSignal output1
LATCH_IN0	Distributed Clocks LatchSignal input0	Input	Distributed Clocks LatchSignal input0
LATCH_IN1	Distributed Clocks LatchSignal input1	Input	Distributed Clocks LatchSignal input1

Table continues on the next page...

Table 828. Signals (continued)

Signal			Description
SYSTEMTIME_OUT[63:0]	SystemTime out signal	Output	SystemTime out signal
PDI_OPB_IRQ	OPB interrupt	Output	EtherCAT System interrupt

55.5 Initialization

55.5.1 Power-on sequence

Table 829. EtherCAT power-on sequence

No.	Step	Result
1	Power-On	Voltages reach proper levels ASICs only: Power-on values are sampled
3	PLL locks	Clocks are generated properly
4	Release RESET	EtherCAT operation begins. Process memory is not accessible until the SII EEPROM is loaded, as well as any function depending on ESC Configuration data. IP Core: PDI is operational; others: PDI is not operational until EEPROM is loaded.
5	Links are established	EtherCAT communication begins, master can access ESC registers
6	Loading EtherCAT EEPROM	Only upon successful EEPROM loading: • ESC Configuration registers initialized • Register bit 0x0110[0] turns to 1 • Process Data RAM becomes accessible • Some PDIs: EEPROM_Loaded signal is driven high • SC is in Init state Steps 5 and 6 are executed in parallel
7	Example: Master proceeds to Operational state	EtherCAT proceeds to Operational state

NOTE

The PDI signals are not driven until the ESC EEPROM is loaded successfully, especially the EEPROM_Loaded signal is not driven and needs a pull-down resistor if it is used.

55.6 Memory map and registers

This section describes the module memory map and registers.

55.6.1 ETHERCAT register descriptions

55.6.1.1 ETHERCAT memory map

ECAT base address: 42A8_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Type (TYPE)	8	R	D8h
1h	Revision (REVISION)	8	R	00h
2h	Build (BUILD)	16	R	0000h
4h	FMMUs supported (FMMUS_SUPPORTED)	8	R	08h
5h	SyncManagers supported (SYNCMANAGERS_SUPPORTED)	8	R	08h
6h	RAM Size (RAM_SIZE)	8	R	08h
7h	Port configuration (PORT_DESCRIPTOR)	8	R	0Fh
8h	Register ESC Features supported (ESC_FEATURES_SUPPORTED)	16	R	01CCh
10h	Configured Station Address (CONFIGURED_STATION_ADDRESS)	16	RW	0000h
10h	Configured Station Address (CONFIGURED_STATION_ADDRESS_PDI)	16	R	0000h
12h	Configured Station Alias (CONFIGURED_STATION_ALIAS)	16	R	0000h
12h	Configured Station Alias (CONFIGURED_STATION_ALIAS_PDI)	16	RW	0000h
20h	Register Write Enable (REGISTER_WRITE_ENABLE)	8	RW	00h
20h	Register Write Enable (REGISTER_WRITE_ENABLE_PDI)	8	R	00h
21h	Register Write Protection (REGISTER_WRITE_PROTECTION)	8	RW	00h
21h	Register Write Protection (REGISTER_WRITE_PROTECTION_PDI)	8	R	00h
30h	ESC Write Enable (ESC_WRITE_ENABLE)	8	RW	00h
30h	ESC Write Enable (ESC_WRITE_ENABLE_PDI)	8	R	00h
31h	ESC Write Protection (ESC_WRITE_PROTECTION)	8	RW	00h
31h	ESC Write Protection (ESC_WRITE_PROTECTION_PDI)	8	R	00h
40h	ESC Reset ECAT WRITE (ESC_RESET_ECAT_WRITE)	8	RW	00h
40h	ESC Reset ECAT WRITE (ESC_RESET_ECAT_WRITE_PDI)	8	R	00h
41h	ESC Reset PDI WRITE (ESC_RESET_PDI_WRITE)	8	R	00h
41h	ESC Reset PDI WRITE (ESC_RESET_PDI_WRITE_PDI)	8	RW	00h
100h	ESC DL Control (ESC_DL_CONTROL)	32	RW	0007_C001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
100h	ESC DL Control (ESC_DL_CONTROL_PDI)	32	R	0007_C001h
108h	Physical Read Write Offset (PHYSICAL_READ_WRITE_OFFSET)	16	RW	0000h
108h	Physical Read Write Offset (PHYSICAL_READ_WRITE_OFFSET_PDI)	16	R	0000h
110h	ESC DL Status (ESC_DL_STATUS)	16	R	5A36h
120h	AL Control (AL_CONTROL)	16	RW	0001h
120h	AL Control (AL_CONTROL_PDI)	16	R	0001h
130h	AL Status (AL_STATUS)	16	R	0001h
130h	AL Status (AL_STATUS_PDI)	16	RW	0001h
134h	AL Status Code (AL_STATUS_CODE)	16	R	0000h
134h	AL Status Code (AL_STATUS_CODE_PDI)	16	RW	0000h
138h	RUN LED Override (RUN_LED_OVERRIDE)	8	RW	00h
139h	ERR LED Override (ERR_LED_OVERRIDE)	8	RW	00h
140h	PDI Control (PDI_CONTROL)	8	R	00h
141h	ESC Configuration (ESC_CONFIGURATION)	8	R	FEh
14Eh	PDI Information (PDI_INFORMATION)	16	R	0000h
150h	Register PDI On-chip bus configuration (PDI_ON_CHIP_BUS_CONFIGURATION)	8	R	84h
151h	PDI Configuration Sync Latch 1 and 0 PDI Configuration (SYNC_LATCH_1_AND_0_PDI_CONFIGURATION)	8	R	EEh
152h	Register PDI On-chip bus extended configuration. (PDI_ON_CHIP_BUS_EXTENDED_CONFIGURATION)	16	R	0000h
200h	ECAT Event Mask (ECAT_EVENT_MASK)	16	RW	0000h
200h	ECAT Event Mask (ECAT_EVENT_MASK_PDI)	16	R	0000h
204h	PDI AL Event Mask (PDI_AL_EVENT_MASK)	32	R	00FF_FF0Fh
204h	PDI AL Event Mask (PDI_AL_EVENT_MASK_PDI)	32	RW	00FF_FF0Fh
210h	ECAT Event Request (ECAT_EVENT_REQUEST)	16	R	0004h
220h	AL Event request (AL_EVENT_REQUEST)	32	R	0000_0020h
300h - 302h	RX Error Counter (RX_ERROR_COUNTER_PORT0 - RX_ERROR_COUNTER_PORT1)	16	RW	0000h
300h - 302h	RX Error Counter (RX_ERROR_COUNTER_PORT0_PDI - RX_ERROR_COUNTER_PORT1_PDI)	16	R	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
308h - 309h	Forwarded RX Error Counter (FORWARDED_RX_ERROR_COUNTER_PORT0 - FORWARDED_RX_ERROR_COUNTER_PORT1)	8	RW	00h
308h - 309h	Forwarded RX Error Counter (FORWARDED_RX_ERROR_COUNTER_PORT0_PDI - FORWARDED_RX_ERROR_COUNTER_PORT1_PDI)	8	R	00h
30Ch	ECAT Processing Unit Error Counter (ECAT_PROCESSING_UNIT_ERROR_COUNTER)	8	RW	00h
30Ch	ECAT Processing Unit Error Counter (ECAT_PROCESSING_UNIT_ERROR_COUNTER_PDI)	8	R	00h
30Dh	PDI Error counter (PDI_ERROR_COUNTER)	8	RW	00h
30Dh	PDI Error counter (PDI_ERROR_COUNTER_PDI)	8	R	00h
30Eh	ASYNCHRONOUS_SYNCHRONOUS_MICROCONTROLLER_PDI_ERROR_CODE. (ASYNCHRONOUS_SYNCHRONOUS_MICROCONTROLLER)	8	R	00h
310h - 311h	Lost Link Counter (LOST_LINK_COUNTER_PORT0 - LOST_LINK_COUNTER_PORT1)	8	RW	00h
310h - 311h	Lost Link Counter (LOST_LINK_COUNTER_PORT0_PDI - LOST_LINK_COUNTER_PORT1_PDI)	8	R	00h
400h	Watchdog Divider (WATCHDOG_DIVIDER)	16	RW	09C2h
400h	Watchdog Divider (WATCHDOG_DIVIDER_PDI)	16	R	09C2h
410h	Register Watchdog Time PDI (WATCHDOG_TIME_PDI)	16	RW	03E8h
410h	Register Watchdog Time PDI (WATCHDOG_TIME_PDI_PDI)	16	R	03E8h
420h	Regsister Watchdog Time Process Data (WATCHDOG_TIME_PROCESS_DATA)	16	RW	03E8h
420h	Regsister Watchdog Time Process Data (WATCHDOG_TIME_PROCESS_DATA_PDI)	16	R	03E8h
440h	Watchdog Status Process Data (WATCHDOG_STATUS_PROCESS_DATA)	16	R	0000h
442h	Watchdog Counter Process Data (WATCHDOG_COUNTER_PROCESS_DATA)	8	RW	00h
442h	Watchdog Counter Process Data (WATCHDOG_COUNTER_PROCESS_DATA_PDI)	8	R	00h
443h	Watchdog Counter PDI (WATCHDOG_COUNTER_PDI)	8	RW	00h
443h	Watchdog Counter PDI (WATCHDOG_COUNTER_PDI_PDI)	8	R	00h
500h	EEPROM Configuration (EEPROM_CONFIGURATION)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
500h	EEPROM Configuration (EEPROM_CONFIGURATION_PDI)	8	R	00h
501h	EEPROM PDI Access State (REGISTER_EEPROM_PDI_ACCESS_STATE)	8	R	00h
501h	EEPROM PDI Access State (REGISTER_EEPROM_PDI_ACCESS_STATE_PDI)	8	RW	00h
502h	Register EEPROM Control/Status (EEPROM_CONTROL_STATUS)	16	RW	9440h
502h	Register EEPROM Control/Status (EEPROM_CONTROL_STATUS_PDI)	16	RW	9440h
504h	EEPROM Address (EEPROM_ADDRESS)	32	RW	0000_0000h
508h	EEPROM Data (EEPROM_DATA)	64	RW	0000_0000_0000_0000h
510h	MII Management Control/Status (MII_MANAGEMENT_CONTROL_OR_STATUS)	16	RW	0002h
510h	MII Management Control/Status (MII_MANAGEMENT_CONTROL_OR_STATUS_PDI)	16	RW	0002h
512h	PHY Address (PHY_ADDRESS)	8	RW	00h
513h	PHY Register Address (PHY_REGISTER_ADDRESS)	8	RW	00h
514h	PHY Data (PHY_DATA)	16	RW	0000h
516h	MII Management ECAT Access State (MII_MANAGEMENT_ECAT_ACCESS_STATE)	8	RW	00h
516h	MII Management ECAT Access State (MII_MANAGEMENT_ECAT_ACCESS_STATE_PDI)	8	R	00h
517h	MII Management PDI Access State (MII_MANAGEMENT_PDI_ACCESS_STATE)	8	RW	00h
517h	MII Management PDI Access State (MII_MANAGEMENT_PDI_ACCESS_STATE_PDI)	8	RW	00h
518h - 519h	PHY Port (PHY_PORT0_STATUS - PHY_PORT1_STATUS)	8	RW	00h
600h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS0)	32	RW	0000_0000h
600h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS0_PDI)	32	R	0000_0000h
604h	Register Length FMMU (FMMU_LENGTH0)	16	RW	0000h
604h	Register Length FMMU (FMMU_LENGTH0_PDI)	16	R	0000h
606h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT0)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
606h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT0_PDI)	8	R	00h
607h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT0)	8	RW	00h
607h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT0_PDI)	8	R	00h
608h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS0)	16	RW	0000h
608h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS0_PDI)	16	R	0000h
60Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT0)	8	RW	00h
60Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT0_PDI)	8	R	00h
60Bh	Register Type FMMU y (FMMU_TYPE0)	8	RW	00h
60Bh	Register Type FMMU y (FMMU_TYPE0_PDI)	8	R	00h
60Ch	Register Activate FMMU (FMMU_ACTIVATE0)	8	RW	00h
60Ch	Register Activate FMMU (FMMU_ACTIVATE0_PDI)	8	R	00h
610h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS1)	32	RW	0000_0000h
610h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS1_PDI)	32	R	0000_0000h
614h	Register Length FMMU (FMMU_LENGTH1)	16	RW	0000h
614h	Register Length FMMU (FMMU_LENGTH1_PDI)	16	R	0000h
616h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT1)	8	RW	00h
616h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT1_PDI)	8	R	00h
617h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT1)	8	RW	00h
617h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT1_PDI)	8	R	00h
618h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS1)	16	RW	0000h
618h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS1_PDI)	16	R	0000h
61Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT1)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
61Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT1_PDI)	8	R	00h
61Bh	Register Type FMMU y (FMMU_TYPE1)	8	RW	00h
61Bh	Register Type FMMU y (FMMU_TYPE1_PDI)	8	R	00h
61Ch	Register Activate FMMU (FMMU_ACTIVATE1)	8	RW	00h
61Ch	Register Activate FMMU (FMMU_ACTIVATE1_PDI)	8	R	00h
620h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS2)	32	RW	0000_0000h
620h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS2_PDI)	32	R	0000_0000h
624h	Register Length FMMU (FMMU_LENGTH2)	16	RW	0000h
624h	Register Length FMMU (FMMU_LENGTH2_PDI)	16	R	0000h
626h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT2)	8	RW	00h
626h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT2_PDI)	8	R	00h
627h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT2)	8	RW	00h
627h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT2_PDI)	8	R	00h
628h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS2)	16	RW	0000h
628h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS2_PDI)	16	R	0000h
62Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT2)	8	RW	00h
62Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT2_PDI)	8	R	00h
62Bh	Register Type FMMU y (FMMU_TYPE2)	8	RW	00h
62Bh	Register Type FMMU y (FMMU_TYPE2_PDI)	8	R	00h
62Ch	Register Activate FMMU (FMMU_ACTIVATE2)	8	RW	00h
62Ch	Register Activate FMMU (FMMU_ACTIVATE2_PDI)	8	R	00h
630h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS3)	32	RW	0000_0000h
630h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS3_PDI)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
634h	Register Length FMMU (FMMU_LENGTH3)	16	RW	0000h
634h	Register Length FMMU (FMMU_LENGTH3_PDI)	16	R	0000h
636h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT3)	8	RW	00h
636h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT3_PDI)	8	R	00h
637h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT3)	8	RW	00h
637h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT3_PDI)	8	R	00h
638h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS3)	16	RW	0000h
638h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS3_PDI)	16	R	0000h
63Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT3)	8	RW	00h
63Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT3_PDI)	8	R	00h
63Bh	Register Type FMMU y (FMMU_TYPE3)	8	RW	00h
63Bh	Register Type FMMU y (FMMU_TYPE3_PDI)	8	R	00h
63Ch	Register Activate FMMU (FMMU_ACTIVATE3)	8	RW	00h
63Ch	Register Activate FMMU (FMMU_ACTIVATE3_PDI)	8	R	00h
640h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS4)	32	RW	0000_0000h
640h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS4_PDI)	32	R	0000_0000h
644h	Register Length FMMU (FMMU_LENGTH4)	16	RW	0000h
644h	Register Length FMMU (FMMU_LENGTH4_PDI)	16	R	0000h
646h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT4)	8	RW	00h
646h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT4_PDI)	8	R	00h
647h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT4)	8	RW	00h
647h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT4_PDI)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
648h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS4)	16	RW	0000h
648h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS4_PDI)	16	R	0000h
64Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT4)	8	RW	00h
64Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT4_PDI)	8	R	00h
64Bh	Register Type FMMU y (FMMU_TYPE4)	8	RW	00h
64Bh	Register Type FMMU y (FMMU_TYPE4_PDI)	8	R	00h
64Ch	Register Activate FMMU (FMMU_ACTIVATE4)	8	RW	00h
64Ch	Register Activate FMMU (FMMU_ACTIVATE4_PDI)	8	R	00h
650h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS5)	32	RW	0000_0000h
650h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS5_PDI)	32	R	0000_0000h
654h	Register Length FMMU (FMMU_LENGTH5)	16	RW	0000h
654h	Register Length FMMU (FMMU_LENGTH5_PDI)	16	R	0000h
656h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT5)	8	RW	00h
656h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT5_PDI)	8	R	00h
657h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT5)	8	RW	00h
657h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT5_PDI)	8	R	00h
658h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS5)	16	RW	0000h
658h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS5_PDI)	16	R	0000h
65Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT5)	8	RW	00h
65Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT5_PDI)	8	R	00h
65Bh	Register Type FMMU y (FMMU_TYPE5)	8	RW	00h
65Bh	Register Type FMMU y (FMMU_TYPE5_PDI)	8	R	00h
65Ch	Register Activate FMMU (FMMU_ACTIVATE5)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
65Ch	Register Activate FMMU (FMMU_ACTIVATE5_PDI)	8	R	00h
660h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS6)	32	RW	0000_0000h
660h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS6_PDI)	32	R	0000_0000h
664h	Register Length FMMU (FMMU_LENGTH6)	16	RW	0000h
664h	Register Length FMMU (FMMU_LENGTH6_PDI)	16	R	0000h
666h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT6)	8	RW	00h
666h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT6_PDI)	8	R	00h
667h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT6)	8	RW	00h
667h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT6_PDI)	8	R	00h
668h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS6)	16	RW	0000h
668h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS6_PDI)	16	R	0000h
66Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT6)	8	RW	00h
66Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT6_PDI)	8	R	00h
66Bh	Register Type FMMU y (FMMU_TYPE6)	8	RW	00h
66Bh	Register Type FMMU y (FMMU_TYPE6_PDI)	8	R	00h
66Ch	Register Activate FMMU (FMMU_ACTIVATE6)	8	RW	00h
66Ch	Register Activate FMMU (FMMU_ACTIVATE6_PDI)	8	R	00h
670h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS7)	32	RW	0000_0000h
670h	Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS7_PDI)	32	R	0000_0000h
674h	Register Length FMMU (FMMU_LENGTH7)	16	RW	0000h
674h	Register Length FMMU (FMMU_LENGTH7_PDI)	16	R	0000h
676h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT7)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
676h	Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT7_PDI)	8	R	00h
677h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT7)	8	RW	00h
677h	Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT7_PDI)	8	R	00h
678h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS7)	16	RW	0000h
678h	Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS7_PDI)	16	R	0000h
67Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT7)	8	RW	00h
67Ah	Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT7_PDI)	8	R	00h
67Bh	Register Type FMMU y (FMMU_TYPE7)	8	RW	00h
67Bh	Register Type FMMU y (FMMU_TYPE7_PDI)	8	R	00h
67Ch	Register Activate FMMU (FMMU_ACTIVATE7)	8	RW	00h
67Ch	Register Activate FMMU (FMMU_ACTIVATE7_PDI)	8	R	00h
800h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS0)	16	RW	0000h
800h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS0_PDI)	16	R	0000h
802h	Register Length SyncManager (SYNCMANAGER_LENGTH0)	16	RW	0000h
802h	Register Length SyncManager (SYNCMANAGER_LENGTH0_PDI)	16	R	0000h
804h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER0)	8	RW	00h
804h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER0_PDI)	8	R	00h
805h	Register Status Register SyncManager (SYNCMANAGER_STATUS0)	8	R	30h
806h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE0)	8	RW	00h
806h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE0_PDI)	8	R	00h
807h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL0)	8	R	00h
807h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL0_PDI)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
808h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS1)	16	RW	0000h
808h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS1_PDI)	16	R	0000h
80Ah	Register Length SyncManager (SYNCMANAGER_LENGTH1)	16	RW	0000h
80Ah	Register Length SyncManager (SYNCMANAGER_LENGTH1_PDI)	16	R	0000h
80Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER1)	8	RW	00h
80Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER1_PDI)	8	R	00h
80Dh	Register Status Register SyncManager (SYNCMANAGER_STATUS1)	8	R	30h
80Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE1)	8	RW	00h
80Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE1_PDI)	8	R	00h
80Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL1)	8	R	00h
80Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL1_PDI)	8	RW	00h
810h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS2)	16	RW	0000h
810h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS2_PDI)	16	R	0000h
812h	Register Length SyncManager (SYNCMANAGER_LENGTH2)	16	RW	0000h
812h	Register Length SyncManager (SYNCMANAGER_LENGTH2_PDI)	16	R	0000h
814h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER2)	8	RW	00h
814h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER2_PDI)	8	R	00h
815h	Register Status Register SyncManager (SYNCMANAGER_STATUS2)	8	R	30h
816h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE2)	8	RW	00h
816h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE2_PDI)	8	R	00h
817h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL2)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
817h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL2_PDI)	8	RW	00h
818h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS3)	16	RW	0000h
818h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS3_PDI)	16	R	0000h
81Ah	Register Length SyncManager (SYNCMANAGER_LENGTH3)	16	RW	0000h
81Ah	Register Length SyncManager (SYNCMANAGER_LENGTH3_PDI)	16	R	0000h
81Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER3)	8	RW	00h
81Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER3_PDI)	8	R	00h
81Dh	Register Status Register SyncManager (SYNCMANAGER_STATUS3)	8	R	30h
81Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE3)	8	RW	00h
81Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE3_PDI)	8	R	00h
81Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL3)	8	R	00h
81Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL3_PDI)	8	RW	00h
820h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS4)	16	RW	0000h
820h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS4_PDI)	16	R	0000h
822h	Register Length SyncManager (SYNCMANAGER_LENGTH4)	16	RW	0000h
822h	Register Length SyncManager (SYNCMANAGER_LENGTH4_PDI)	16	R	0000h
824h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER4)	8	RW	00h
824h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER4_PDI)	8	R	00h
825h	Register Status Register SyncManager (SYNCMANAGER_STATUS4)	8	R	30h
826h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE4)	8	RW	00h
826h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE4_PDI)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
827h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL4)	8	R	00h
827h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL4_PDI)	8	RW	00h
828h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS5)	16	RW	0000h
828h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS5_PDI)	16	R	0000h
82Ah	Register Length SyncManager (SYNCMANAGER_LENGTH5)	16	RW	0000h
82Ah	Register Length SyncManager (SYNCMANAGER_LENGTH5_PDI)	16	R	0000h
82Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER5)	8	RW	00h
82Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER5_PDI)	8	R	00h
82Dh	Register Status Register SyncManager (SYNCMANAGER_STATUS5)	8	R	30h
82Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE5)	8	RW	00h
82Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE5_PDI)	8	R	00h
82Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL5)	8	R	00h
82Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL5_PDI)	8	RW	00h
830h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS6)	16	RW	0000h
830h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS6_PDI)	16	R	0000h
832h	Register Length SyncManager (SYNCMANAGER_LENGTH6)	16	RW	0000h
832h	Register Length SyncManager (SYNCMANAGER_LENGTH6_PDI)	16	R	0000h
834h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER6)	8	RW	00h
834h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER6_PDI)	8	R	00h
835h	Register Status Register SyncManager (SYNCMANAGER_STATUS6)	8	R	30h
836h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE6)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
836h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE6_PDI)	8	R	00h
837h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL6)	8	R	00h
837h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL6_PDI)	8	RW	00h
838h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS7)	16	RW	0000h
838h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS7_PDI)	16	R	0000h
83Ah	Register Length SyncManager (SYNCMANAGER_LENGTH7)	16	RW	0000h
83Ah	Register Length SyncManager (SYNCMANAGER_LENGTH7_PDI)	16	R	0000h
83Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER7)	8	RW	00h
83Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER7_PDI)	8	R	00h
83Dh	Register Status Register SyncManager (SYNCMANAGER_STATUS7)	8	R	30h
83Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE7)	8	RW	00h
83Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE7_PDI)	8	R	00h
83Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL7)	8	R	00h
83Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL7_PDI)	8	RW	00h
840h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS8)	16	RW	0000h
840h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS8_PDI)	16	R	0000h
842h	Register Length SyncManager (SYNCMANAGER_LENGTH8)	16	RW	0000h
842h	Register Length SyncManager (SYNCMANAGER_LENGTH8_PDI)	16	R	0000h
844h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER8)	8	RW	00h
844h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER8_PDI)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
845h	Register Status Register SyncManager (SYNCMANAGER_STATUS8)	8	R	30h
846h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE8)	8	RW	00h
846h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE8_PDI)	8	R	00h
847h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL8)	8	R	00h
847h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL8_PDI)	8	RW	00h
848h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS9)	16	RW	0000h
848h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS9_PDI)	16	R	0000h
84Ah	Register Length SyncManager (SYNCMANAGER_LENGTH9)	16	RW	0000h
84Ah	Register Length SyncManager (SYNCMANAGER_LENGTH9_PDI)	16	R	0000h
84Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER9)	8	RW	00h
84Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER9_PDI)	8	R	00h
84Dh	Register Status Register SyncManager (SYNCMANAGER_STATUS9)	8	R	30h
84Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE9)	8	RW	00h
84Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE9_PDI)	8	R	00h
84Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL9)	8	R	00h
84Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL9_PDI)	8	RW	00h
850h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS10)	16	RW	0000h
850h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS10_PDI)	16	R	0000h
852h	Register Length SyncManager (SYNCMANAGER_LENGTH10)	16	RW	0000h
852h	Register Length SyncManager (SYNCMANAGER_LENGTH10_PDI)	16	R	0000h
854h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER10)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
854h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER10_PDI)	8	R	00h
855h	Register Status Register SyncManager (SYNCMANAGER_STATUS10)	8	R	30h
856h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE10)	8	RW	00h
856h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE10_PDI)	8	R	00h
857h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL10)	8	R	00h
857h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL10_PDI)	8	RW	00h
858h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS11)	16	RW	0000h
858h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS11_PDI)	16	R	0000h
85Ah	Register Length SyncManager (SYNCMANAGER_LENGTH11)	16	RW	0000h
85Ah	Register Length SyncManager (SYNCMANAGER_LENGTH11_PDI)	16	R	0000h
85Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER11)	8	RW	00h
85Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER11_PDI)	8	R	00h
85Dh	Register Status Register SyncManager (SYNCMANAGER_STATUS11)	8	R	30h
85Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE11)	8	RW	00h
85Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE11_PDI)	8	R	00h
85Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL11)	8	R	00h
85Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL11_PDI)	8	RW	00h
860h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS12)	16	RW	0000h
860h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS12_PDI)	16	R	0000h
862h	Register Length SyncManager (SYNCMANAGER_LENGTH12)	16	RW	0000h
862h	Register Length SyncManager (SYNCMANAGER_LENGTH12_PDI)	16	R	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
864h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER12)	8	RW	00h
864h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER12_PDI)	8	R	00h
865h	Register Status Register SyncManager (SYNCMANAGER_STATUS12)	8	R	30h
866h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE12)	8	RW	00h
866h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE12_PDI)	8	R	00h
867h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL12)	8	R	00h
867h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL12_PDI)	8	RW	00h
868h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS13)	16	RW	0000h
868h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS13_PDI)	16	R	0000h
86Ah	Register Length SyncManager (SYNCMANAGER_LENGTH13)	16	RW	0000h
86Ah	Register Length SyncManager (SYNCMANAGER_LENGTH13_PDI)	16	R	0000h
86Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER13)	8	RW	00h
86Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER13_PDI)	8	R	00h
86Dh	Register Status Register SyncManager (SYNCMANAGER_STATUS13)	8	R	30h
86Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE13)	8	RW	00h
86Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE13_PDI)	8	R	00h
86Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL13)	8	R	00h
86Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL13_PDI)	8	RW	00h
870h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS14)	16	RW	0000h
870h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS14_PDI)	16	R	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
872h	Register Length SyncManager (SYNCMANAGER_LENGTH14)	16	RW	0000h
872h	Register Length SyncManager (SYNCMANAGER_LENGTH14_PDI)	16	R	0000h
874h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER14)	8	RW	00h
874h	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER14_PDI)	8	R	00h
875h	Register Status Register SyncManager (SYNCMANAGER_STATUS14)	8	R	30h
876h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE14)	8	RW	00h
876h	Register Activate SyncManager (SYNCMANAGER_ACTIVATE14_PDI)	8	R	00h
877h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL14)	8	R	00h
877h	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL14_PDI)	8	RW	00h
878h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS15)	16	RW	0000h
878h	Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS15_PDI)	16	R	0000h
87Ah	Register Length SyncManager (SYNCMANAGER_LENGTH15)	16	RW	0000h
87Ah	Register Length SyncManager (SYNCMANAGER_LENGTH15_PDI)	16	R	0000h
87Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER15)	8	RW	00h
87Ch	Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER15_PDI)	8	R	00h
87Dh	Register Status Register SyncManager (SYNCMANAGER_STATUS15)	8	R	30h
87Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE15)	8	RW	00h
87Eh	Register Activate SyncManager (SYNCMANAGER_ACTIVATE15_PDI)	8	R	00h
87Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL15)	8	R	00h
87Fh	Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL15_PDI)	8	RW	00h
900h	Distributed Clocks Receive Times (RECEIVE_TIMES)	32	RW	6568_7445h
900h	Distributed Clocks Receive Times (RECEIVE_TIMES_PDI)	32	R	6568_7445h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
904h	Distributed Clocks Receive Time Port 1 (RECEIVE_TIME_PORT_1)	32	R	5441_4372h
910h	Register System Time (SYSTEM_TIME)	64	RW	0000_0000_0001_7818h
910h	Register System Time (SYSTEM_TIME_PDI)	64	RW	0000_0000_0001_7818h
918h	Distributed Clocks Register Receive Time ECAT Processing Unit (RECEIVE_TIME_ECAT_PROCESSING_UNIT)	64	R	5645_2D52_542D_4254h
920h	Register System Time Offset (SYSTEM_TIME_OFFSET)	64	RW	0000_0000_0000_0000h
928h	Register System Time Delay (SYSTEM_TIME_DELAY)	32	RW	0000_0000h
92Ch	Register System Time Difference (SYSTEM_TIME_DIFFERENCE)	32	R	0000_0000h
930h	Register Speed Counter Start (SPEED_COUNTER_START)	16	RW	1000h
932h	Register Speed Counter Diff (SPEED_COUNTER_DIFF)	16	R	0000h
934h	Register System Time Difference Filter Depth (SYSTEM_TIME_DIFFERENCE_FILTER_DEPTH)	8	RW	04h
935h	Register Speed Counter Filter Depth (SPEED_COUNTER_FILTER_DEPTH)	8	RW	0Ch
980h	Register Cyclic Unit Control (CYCLIC_UNIT_CONTROL)	8	RW	10h
980h	Register Cyclic Unit Control (CYCLIC_UNIT_CONTROL_PDI)	8	R	10h
981h	Register Activation register (UNIT_ACTIVATION_REGISTER)	8	RW	00h
982h	Register Pulse Length of SyncSignals (UNI_PULSE_LENGTH_OF_SYNCSIGNALS)	16	R	000Ah
984h	Register Activation Status (UNIT_ACTIVATION_STATUS)	8	R	00h
98Eh	Register SYNC0 Status (UNIT_SYNC0_STATUS)	8	R	00h
98Fh	Register SYNC1 Status (UNIT_SYNC1_STATUS)	8	R	00h
990h	Register Start Time Cyclic Operation (UNIT_START_TIME_CYCLIC_OPERATION)	64	RW	0000_0000_0000_0000h
998h	Register Next SYNC1 Pulse (UNIT_NEXT_SYNC1_PULSE)	64	R	0000_0000_0000_0000h
9A0h	Register SYNC0 Cycle Time (UNIT_SYNC0_CYCLE_TIME)	32	RW	0000_0000h
9A4h	Register SYNC1 Cycle Time (UNIT_SYNC1_CYCLE_TIME)	32	RW	0000_0000h
9A8h	Register Latch0 Control (LATCH0_CONTROL)	8	RW	00h
9A9h	Register Latch1 Control (LATCH1_CONTROL)	8	RW	00h
9AEh	Register Latch0 Status (LATCH0_STATUS)	8	R	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9AFh	Register Latch1 Status (LATCH1_STATUS)	8	R	00h
9B0h	Register Latch0 Time Positive Edge (LATCH0_TIME_POSITIVE_EDGE)	64	R	0000_0000_0000_0000h
9B8h	Register Latch0 Time Negative Edge (LATCH0_TIME_NEGATIVE_EDGE)	64	R	0000_0000_0000_0000h
9C0h	Register Latch1 Time Positive Edge (LATCH1_TIME_POSITIVE_EDGE)	64	R	0000_0000_0000_0000h
9C8h	Register Latch1 Time Negative Edge (LATCH1_TIME_NEGATIVE_EDGE)	64	R	0000_0000_0000_0000h
9F0h	Register EtherCAT Buffer Change Event Time (ETHERCAT_BUFFER_CHANGE_EVENT_TIME)	32	R	0000_0000h
9F8h	Register PDI Buffer Start Event Time (PDI_BUFFER_START_EVENT_TIME)	32	R	0000_0000h
9FCh	Register PDI Buffer Change Event Time (PDI_BUFFER_CHANGE_EVENT_TIME)	32	R	0000_0000h
E00h	Register Product ID IP Core (PRODUCT_ID_IP_CORE)	64	R	3038_3131_5452_584Dh
E08h	Register Vendor ID IP Core (VENDOR_ID_IP_CORE)	64	R	EEEE_EEEE_EEEE_EEEEh
F10h	Register General Purpose Outputs (GENERAL_PURPOSE_OUTPUTS)	16	RW	0000h
F18h	Register General Purpose Inputs (GENERAL_PURPOSE_INPUTS)	16	R	0000h

55.6.1.2 Type (TYPE)

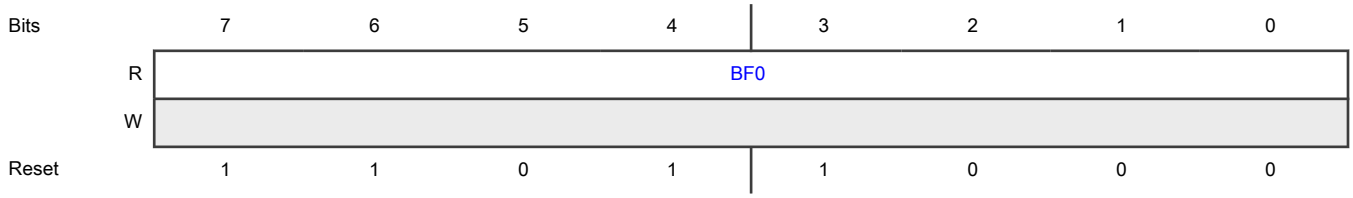
Offset

Register	Offset
TYPE	0h

Function

Provides information on type

Diagram



Fields

Field	Function
7-0 BF0	Type of EtherCAT controller Bit field access for ECAT: r/- Bit field access for PDI: r/-

55.6.1.3 Revision (REVISION)

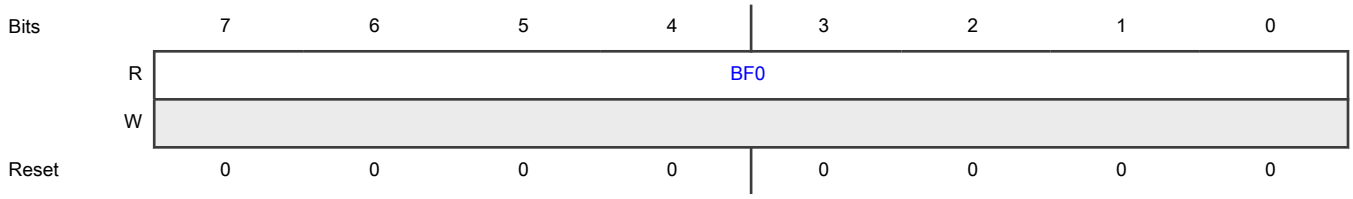
Offset

Register	Offset
REVISION	1h

Function

Provides information on revision

Diagram



Fields

Field	Function
7-0 BF0	Revision of EtherCAT controller. IP Core: major version X (version X.Y.Z) Bit field access for ECAT: r/- Bit field access for PDI: r/-

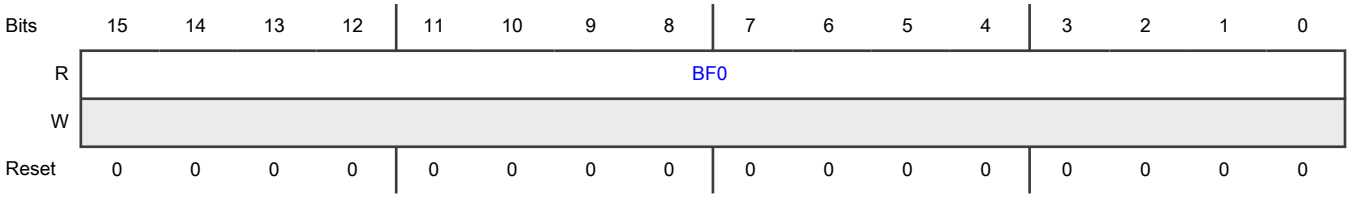
55.6.1.4 Build (BUILD)

Offset

Register	Offset
BUILD	2h

Function
Provides information on build

Diagram



Fields

Field	Function
15-0 BF0	Build of EtherCAT controller EtherCAT IP Core (version X.Y.Z): [3:0] maintenance version Z [7:4] minor version Y [15:8] patch level/ development build: ----0x00: Original release ----0x01-0x0F: Patch level of original release ----0x10-0xFF: Development build Bit field access for ECAT: r/- Bit field access for PDI: r/-

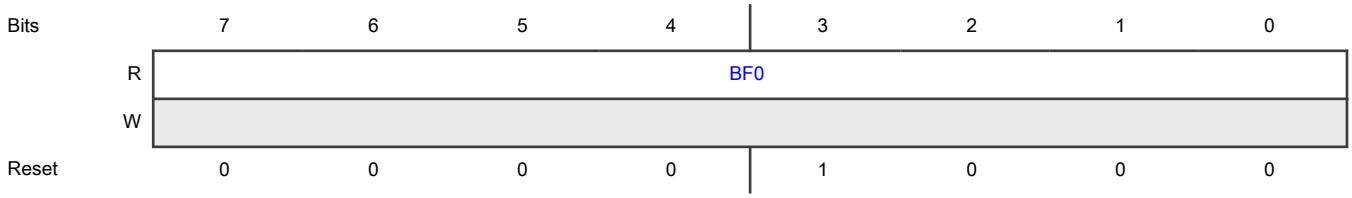
55.6.1.5 FMMUs supported (FMMUS_SUPPORTED)

Offset

Register	Offset
FMMUS_SUPPORTED	4h

Function
Specifies number of FMMU channels supported

Diagram



Fields

Field	Function
7-0	Number of supported FMMU channels (or entities)
BF0	Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration.

55.6.1.6 SyncManagers supported (SYNCMANAGERS_SUPPORTED)

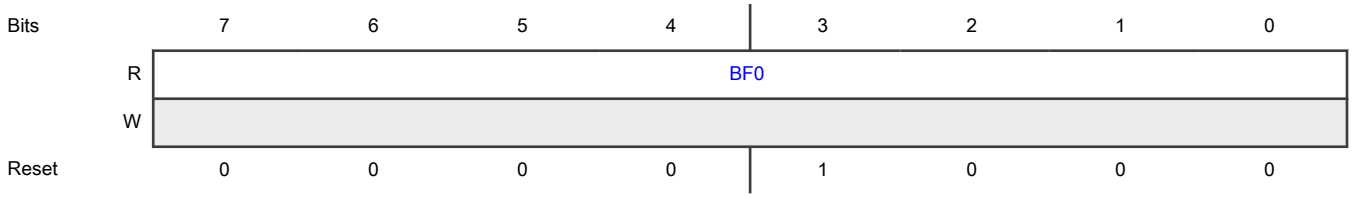
Offset

Register	Offset
SYNCMANAGERS_SUPPORTED	5h

Function

Specifies number of SyncManager channels supported

Diagram



Fields

Field	Function
7-0	Number of supported SyncManager channels (or entities)
BF0	Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration.

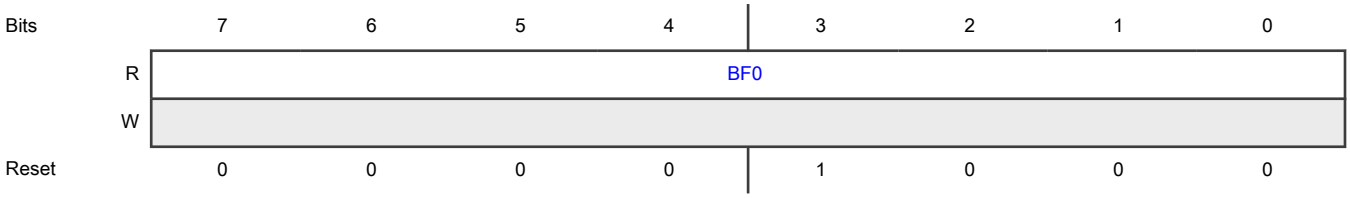
55.6.1.7 RAM Size (RAM_SIZE)

Offset

Register	Offset
RAM_SIZE	6h

Function
Specifies process data RAM size

Diagram



Fields

Field	Function
7-0 BF0	Process Data RAM size supported in KByte Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration.

55.6.1.8 Port configuration (PORT_DESCRIPTOR)

Offset

Register	Offset
PORT_DESCRIPTOR	7h

Function

- 00: Not implemented
- 01: Not configured (SII EEPROM)
- 10: Reserved
- 11: MII / RMII

Diagram

Bits	7	6	5	4	3	2	1	0
R	BF6		BF4		BF2		BF0	
W								
Reset	0	0	0	0	1	1	1	1

Fields

Field	Function
7-6 BF6	Reserved
5-4 BF4	Reserved
3-2 BF2	Port 1 Bit field access for ECAT: r/- Bit field access for PDI: r/- The reset value for bit is ESC and ESC configuration dep.
1-0 BF0	Port 0 Bit field access for ECAT: r/- Bit field access for PDI: r/- The reset value for bit is ESC and ESC configuration dep.

55.6.1.9 Register ESC Features supported (ESC_FEATURES_SUPPORTED)

Offset

Register	Offset
ESC_FEATURES_SUPPORTED	8h

Function
ESC features supported

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF12				BF11	BF10	BF9	BF8	BF7	BF6	Reserved	Reserved	BF3	BF2	BF1	BF0
W																
Reset	0	0	0	0	0	0	0	1	1	1	0	0	1	1	0	0

Fields

Field	Function
15-12 BF12	Reserved Bit field access for ECAT: r/- Bit field access for PDI: r/-
11 BF11	Fixed FMMU/SyncManager configuration Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Variable configuration 1b - Fixed configuration (refer to documentation of supporting ESCs)
10 BF10	EtherCAT read/write command support (BRW, APRW, FPRW): Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Supported 1b - Not supported
9 BF9	EtherCAT LRW command support: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Supported 1b - Not supported
8 BF8	Enhanced DC SYNC Activation **NOTE** This feature refers to registers 0x981[7:3] and 0x0984 Bit field access for ECAT: r/- Bit field access for PDI: r/- The reset value for bit depends on version 0b - Not available 1b - Available

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 BF7	Separate Handling of FCS Errors: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Not supported 1b - Supported, frames with wrong FCS and additional nibble will be counted separately in Forwarded RX Error Counter
6 BF6	Enhanced Link Detection MII: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Not available 1b - Available
5 —	Reserved
4 —	Reserved
3 BF3	Distributed Clocks (width): Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration. 0b - 32 bit 1b - 64 bit
2 BF2	Distributed Clocks: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration. 0b - Not available 1b - Available
1 BF1	Unused register access: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - allowed 1b - not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 BF0	FMMU Operation: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Bit oriented 1b - Byte oriented

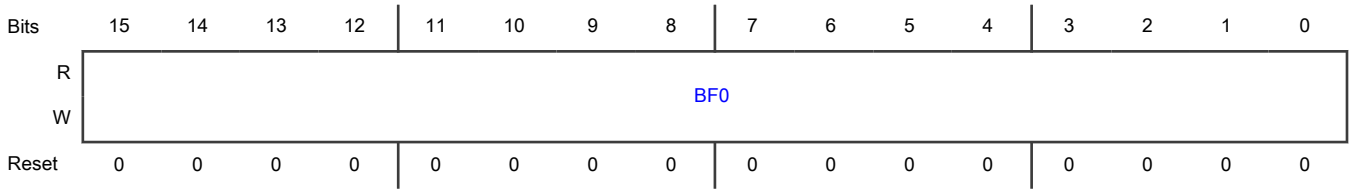
55.6.1.10 Configured Station Address (CONFIGURED_STATION_ADDRESS)

Offset

Register	Offset
CONFIGURED_STATION_ADDRESS	10h

Function
Configured station Address

Diagram



Fields

Field	Function
15-0 BF0	Address used for node addressing (FPRD/FPWR/FPRW/FRMW commands). Bit field access for ECAT: r/w Bit field access for PDI: r/-

55.6.1.11 Configured Station Address (CONFIGURED_STATION_ADDRESS_PDI)

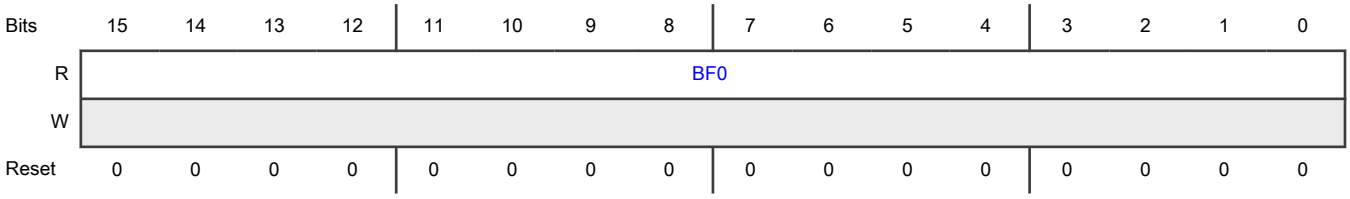
Offset

Register	Offset
CONFIGURED_STATION_ADDRESS_PDI	10h

Function

Configured station Address

Diagram



Fields

Field	Function
15-0 BF0	Address used for node addressing (FPRD/FPWR/FPRW/FRMW commands). Bit field access for ECAT: r/w Bit field access for PDI: r/-

55.6.1.12 Configured Station Alias (CONFIGURED_STATION_ALIAS)

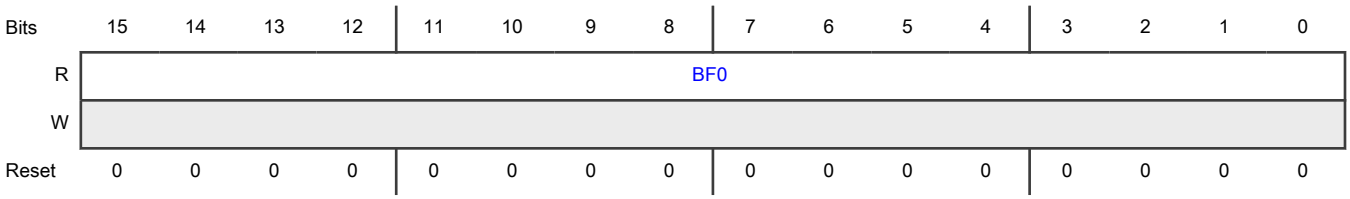
Offset

Register	Offset
CONFIGURED_STATION_ALIAS	12h

Function

Configured Station Alias

Diagram



Fields

Field	Function
15-0 BF0	Alias Address used for node addressing (FPRD/FPWR/FPRW/FRMW commands). The use of this alias is activated by Register DL Control Bit 0x0100[24].

Table continues on the next page...

Field	Function
	NOTE EEPROM value is only transferred into this register at first EEPROM load after power-on or reset. Bit field access for ECAT: r/- Bit field access for PDI: r/w Reset for bit is 0 until first EEPROM load, then EEPROM word 4

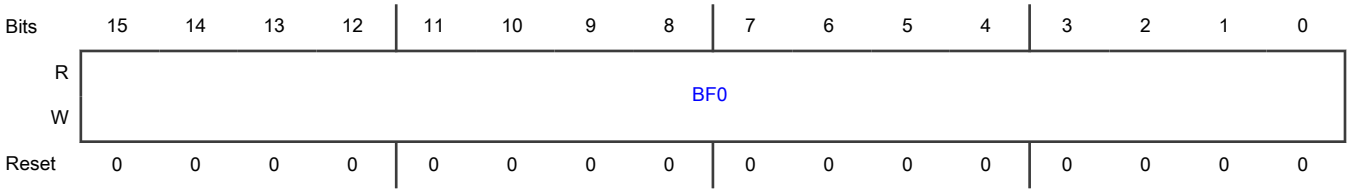
55.6.1.13 Configured Station Alias (CONFIGURED_STATION_ALIAS_PDI)

Offset

Register	Offset
CONFIGURED_STATION_ALIAS_PDI	12h

Function
Configured Station Alias

Diagram



Fields

Field	Function
15-0 BF0	Alias Address used for node addressing (FPRD/FPWR/FPRW/FRMW commands). The use of this alias is activated by Register DL Control Bit 0x0100[24]. **NOTE** EEPROM value is only transferred into this register at first EEPROM load after power-on or reset. Bit field access for ECAT: r/- Bit field access for PDI: r/w Reset for bit is 0 until first EEPROM load, then EEPROM word 4

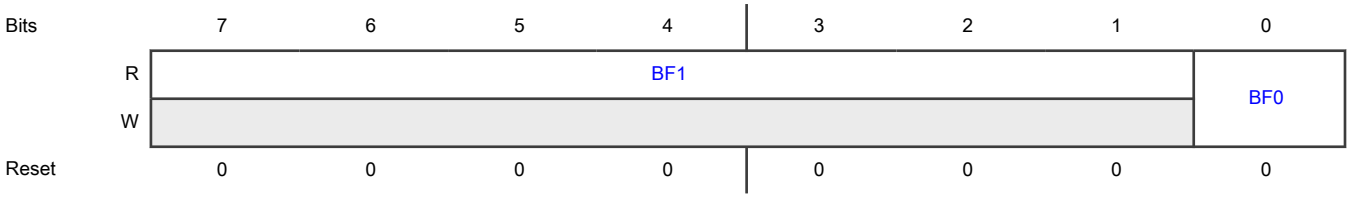
55.6.1.14 Register Write Enable (REGISTER_WRITE_ENABLE)

Offset

Register	Offset
REGISTER_WRITE_ENABLE	20h

Function
Register Write Enable

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	Register Write Enable. If register write protection is enabled, this register has to be written in the same Ethernet frame (value does not matter) before other writes to this station are allowed. This bit is self-clearing at the beginning of the next frame (SOF), or if Register Write Protection is disabled. Bit field access for ECAT: r/w Bit field access for PDI: r/-

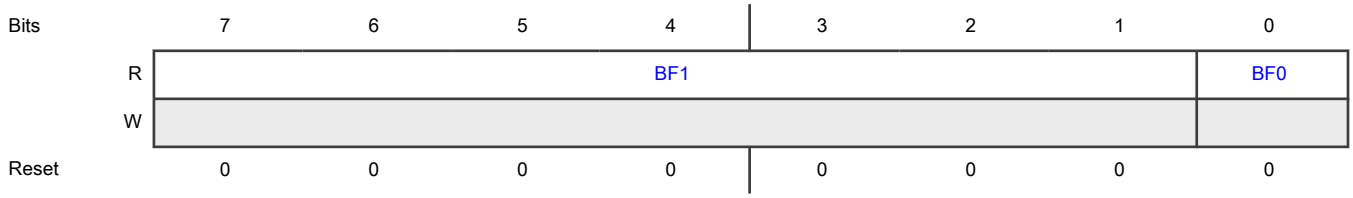
55.6.1.15 Register Write Enable (REGISTER_WRITE_ENABLE_PDI)

Offset

Register	Offset
REGISTER_WRITE_ENABLE_PDI	20h

Function
Register Write Enable

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	Register Write Enable. If register write protection is enabled, this register has to be written in the same Ethernet frame (value does not matter) before other writes to this station are allowed. This bit is self-clearing at the beginning of the next frame (SOF), or if Register Write Protection is disabled. Bit field access for ECAT: r/w Bit field access for PDI: r/-

55.6.1.16 Register Write Protection (REGISTER_WRITE_PROTECTION)

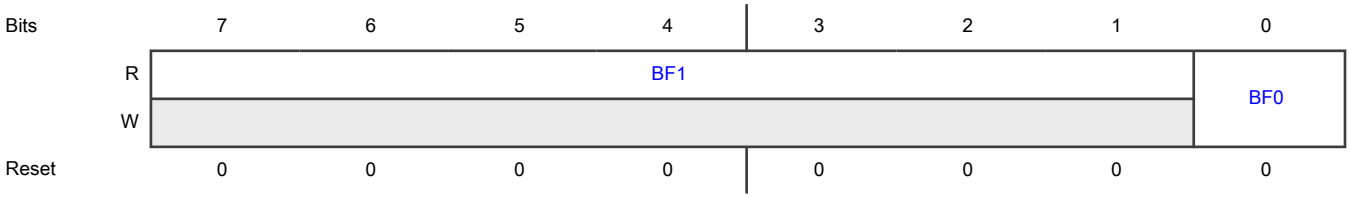
Offset

Register	Offset
REGISTER_WRITE_PROTECTION	21h

Function

Register Write Protection

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	Register write protection. Registers 0x0000:0x0F7F are write-protected, except for 0x0020 and 0x0030. Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - Protection disabled 1b - Protection enabled

55.6.1.17 Register Write Protection (REGISTER_WRITE_PROTECTION_PDI)

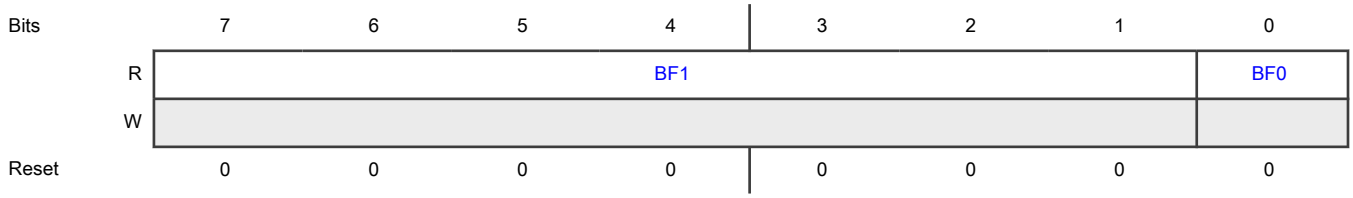
Offset

Register	Offset
REGISTER_WRITE_PROTECTION_PDI	21h

Function

Register Write Protection

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	Register write protection. Registers 0x0000:0x0F7F are write-protected, except for 0x0020 and 0x0030. Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - Protection disabled 1b - Protection enabled

55.6.1.18 ESC Write Enable (ESC_WRITE_ENABLE)

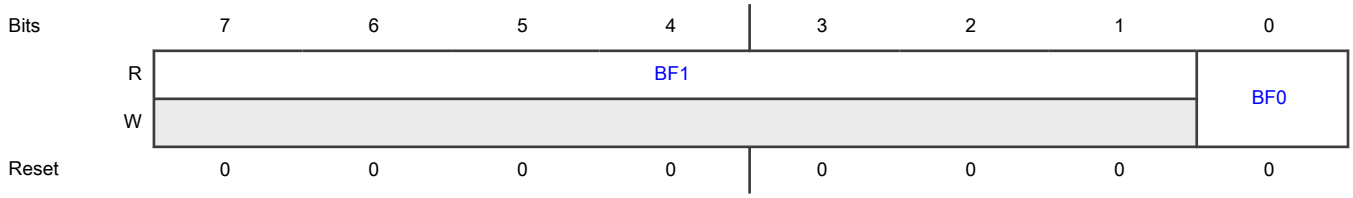
Offset

Register	Offset
ESC_WRITE_ENABLE	30h

Function

ESC Write Enable

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	ESC Write Enable If ESC write protection is enabled, this register has to be written in the same Ethernet frame (value does not matter) before other writes to this station are allowed. This bit is self-clearing at the beginning of the next frame (SOF), or if ESC Write Protection is disabled. Bit field access for ECAT: r/w Bit field access for PDI: r/-

55.6.1.19 ESC Write Enable (ESC_WRITE_ENABLE_PDI)

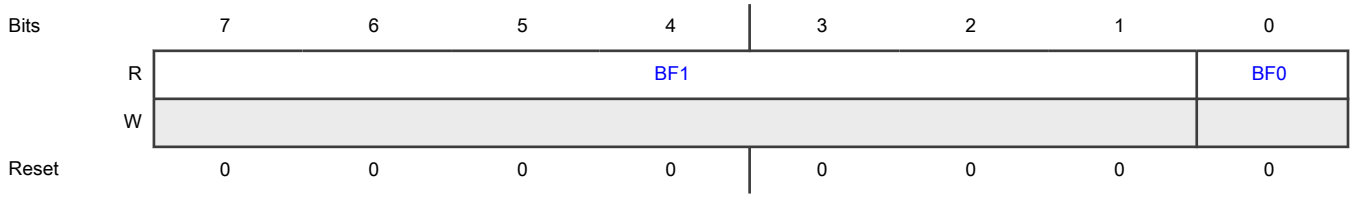
Offset

Register	Offset
ESC_WRITE_ENABLE_PDI	30h

Function

ESC Write Enable

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	ESC Write Enable If ESC write protection is enabled, this register has to be written in the same Ethernet frame (value does not matter) before other writes to this station are allowed. This bit is self-clearing at the beginning of the next frame (SOF), or if ESC Write Protection is disabled. Bit field access for ECAT: r/w Bit field access for PDI: r/-

55.6.1.20 ESC Write Protection (ESC_WRITE_PROTECTION)

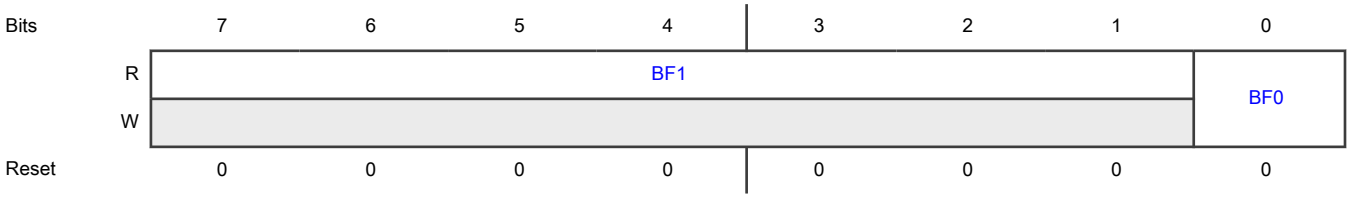
Offset

Register	Offset
ESC_WRITE_PROTECTION	31h

Function

ESC Write Protection

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	Write protect: All areas are write-protected, except for 0x0030. Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - Protection disabled 1b - Protection enabled

55.6.1.21 ESC Write Protection (ESC_WRITE_PROTECTION_PDI)

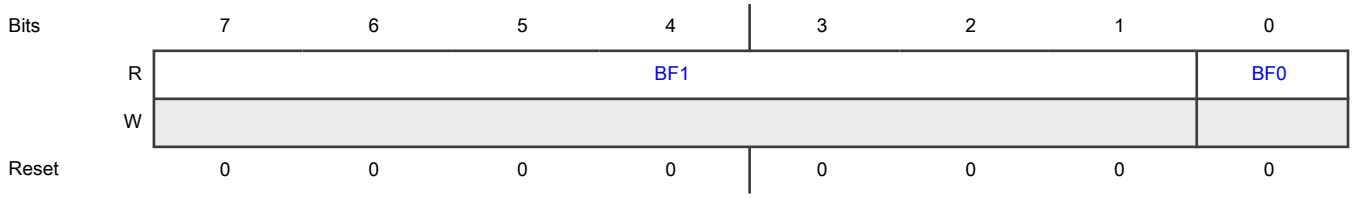
Offset

Register	Offset
ESC_WRITE_PROTECTION_PDI	31h

Function

ESC Write Protection

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	Write protect: All areas are write-protected, except for 0x0030. Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - Protection disabled 1b - Protection enabled

55.6.1.22 ESC Reset ECAT WRITE (ESC_RESET_ECAT_WRITE)

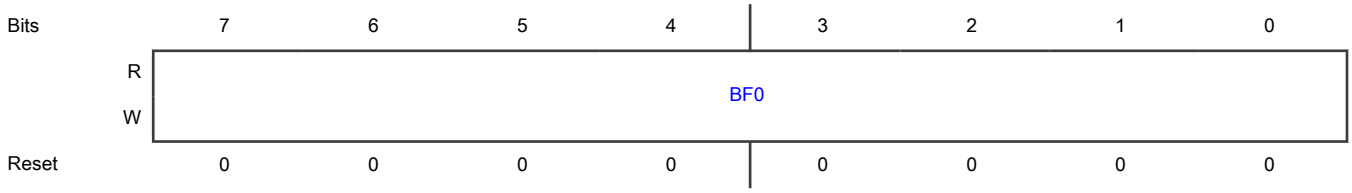
Offset

Register	Offset
ESC_RESET_ECAT_WRITE	40h

Function

ESC Reset ECAT WRITE

Diagram



Fields

Field	Function
7-0 BF0	A reset is asserted after writing 0x52 (‘R’), 0x45 (‘E’) and 0x53 (‘S’) in this register with 3 consecutive frames. Bit field access for ECAT: r/w Bit field access for PDI: r/-

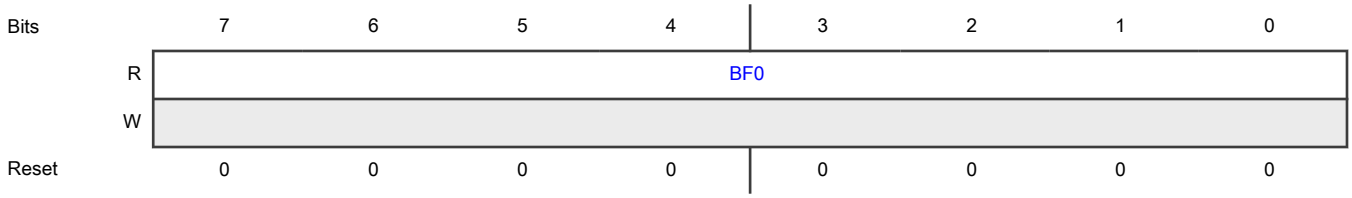
55.6.1.23 ESC Reset ECAT WRITE (ESC_RESET_ECAT_WRITE_PDI)

Offset

Register	Offset
ESC_RESET_ECAT_WRITE_PDI	40h

Function
ESC Reset ECAT WRITE

Diagram



Fields

Field	Function
7-0 BF0	A reset is asserted after writing 0x52 (‘R’), 0x45 (‘E’) and 0x53 (‘S’) in this register with 3 consecutive frames. Bit field access for ECAT: r/w Bit field access for PDI: r/-

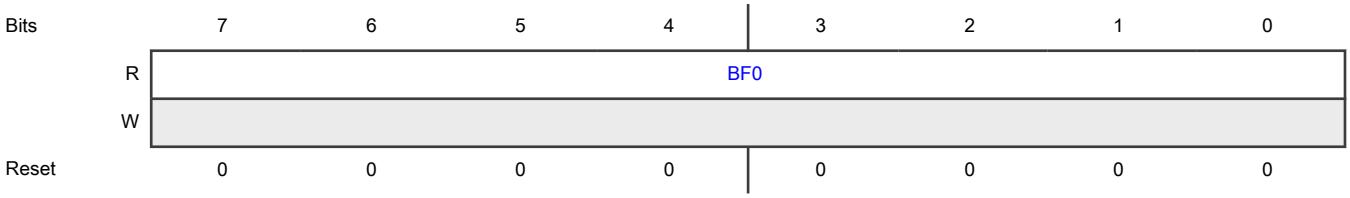
55.6.1.24 ESC Reset PDI WRITE (ESC_RESET_PDI_WRITE)

Offset

Register	Offset
ESC_RESET_PDI_WRITE	41h

Function
ESC Reset PDI WRITE

Diagram



Fields

Field	Function
7-0 BF0	A reset is asserted after writing 0x52 (‘R’), 0x45 (‘E’) and 0x53 (‘S’) in this register with 3 consecutive commands. Bit field access for ECAT: r/- Bit field access for PDI: r/w

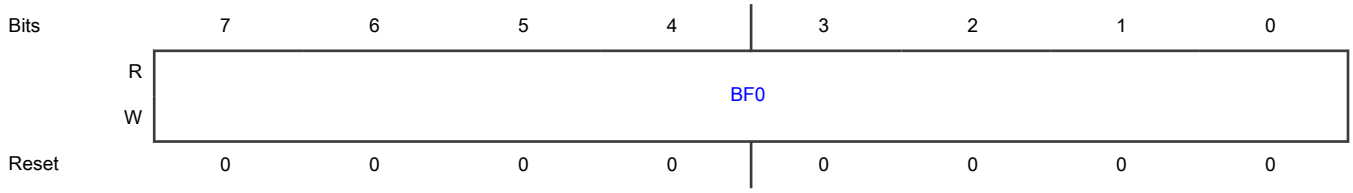
55.6.1.25 ESC Reset PDI WRITE (ESC_RESET_PDI_WRITE_PDI)

Offset

Register	Offset
ESC_RESET_PDI_WRITE_PDI	41h

Function
ESC Reset PDI WRITE

Diagram



Fields

Field	Function
7-0 BF0	A reset is asserted after writing 0x52 (‘R’), 0x45 (‘E’) and 0x53 (‘S’) in this register with 3 consecutive commands. Bit field access for ECAT: r/- Bit field access for PDI: r/w

55.6.1.26 ESC DL Control (ESC_DL_CONTROL)

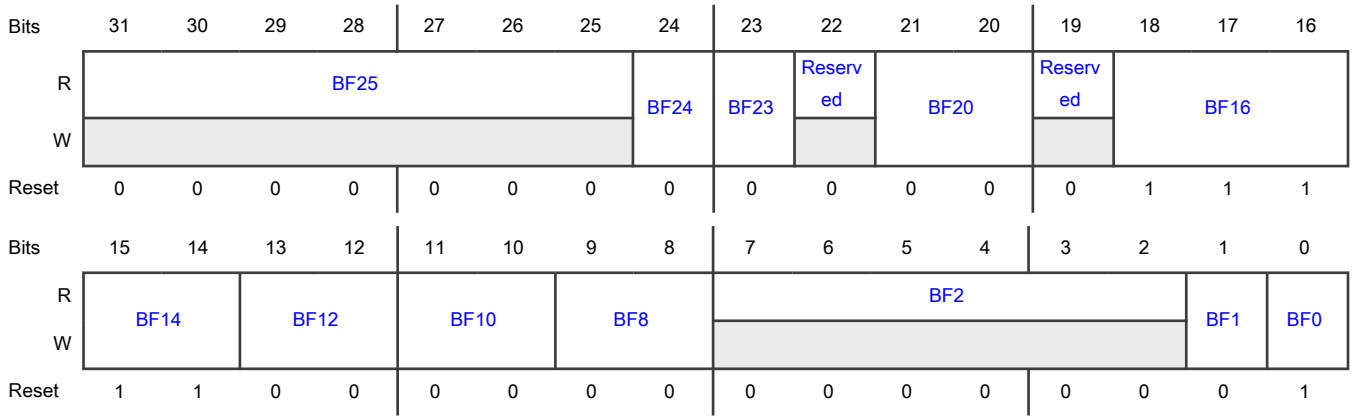
Offset

Register	Offset
ESC_DL_CONTROL	100h

Function

The possibility of RX FIFO Size reduction depends on the clock source accuracy of the ESC and of every connected EtherCAT/ Ethernet devices (master, slave, etc.). RX FIFO Size of 7 is sufficient for 100ppm accuracy, FIFO Size 0 is possible with 25ppm accuracy (frame size of 1518/1522 Byte).

Diagram



Fields

Field	Function
31-25 BF25	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
24 BF24	Station alias: Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - Ignore Station Alias 1b - Alias can be used for all configured address command types (FPRD, FPWR, …)
23 BF23	Reserved, write 0 Bit field access for ECAT: r/w Bit field access for PDI: r/- Reset for bit is 0, later EEPROM word 5[7]
22 —	Reserved
21-20 BF20	Reserved, write 0 Bit field access for ECAT: r/w Bit field access for PDI: r/- Reset for bit is 0, later EEPROM word 5[5:4]
19 —	Reserved

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>18-16
BF16</td><td>RX FIFO Size (ESC delays start of forwarding until FIFO is at least half full).
RX FIFO Size/RX delay reduction** :
<table><tr><th>Value</th><th>MII</th></tr><tr><td>0</td><td>-40ns(-80ns***)</td></tr><tr><td>1</td><td>-40ns(-80ns***)</td></tr><tr><td>2</td><td>-40ns(-80ns***)</td></tr><tr><td>3</td><td>-40ns</td></tr><tr><td>4</td><td>no change</td></tr><tr><td>5</td><td>no change</td></tr><tr><td>6</td><td>no change</td></tr><tr><td>7</td><td>default</td></tr></table>
NOTE
EEPROM value is only taken over at first EEPROM load after power-on or reset
Bit field access for ECAT: r/w
Bit field access for PDI: r/-
** The possibility of RX FIFO Size reduction depends on the clock source accuracy of the eCAT and of every connected EtherCAT/Ethernet devices. RX FIFO Size of 7 is sufficient for 100ppm accuracy, FIFO Size 0 is possible with 25ppm accuracy (frame size of 1518/1522 Byte).</td></tr>
<tr><td>15-14
BF14</td><td>Reserved</td></tr>
<tr><td>13-12
BF12</td><td>Reserved</td></tr>
<tr><td>11-10
BF10</td><td>Loop Port 1:
* Loop configuration changes are delayed until the end of a currently received or transmitted frame at the port.
Bit field access for ECAT: r/w*
Bit field access for PDI: r/-
00b - Auto
01b - Auto Close
10b - Open
11b - Closed</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 BF8	Loop Port 0: **NOTE** Loop open means sending/receiving over this port is enabled, loop closed means sending/receiving is disabled and frames are forwarded to the next open port internally. Auto: loop closed at link down, opened at link up Auto Close: loop closed at link down, opened with writing 01 again after link up (or receiving a valid Ethernet frame at the closed port) Open: loop open regardless of link state Closed: loop closed regardless of link state * Loop configuration changes are delayed until the end of a currently received or transmitted frame at the port. Bit field access for ECAT: r/w* Bit field access for PDI: r/- 00b - Auto 01b - Auto Close 10b - Open 11b - Closed
7-2 BF2	Reserved, Write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Temporary use of settings in 0x0100:0x0103[8:15]: Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - permanent use 1b - use for about 1 second, then revert to previous settings
0 BF0	Forwarding Rule Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - EtherCAT frames are processed, non-EtherCAT frames are forwarded without processing or modification. The source MAC address is not changed for any frame. 1b - EtherCAT frames are processed, non-EtherCAT frames are destroyed. The source MAC address is changed by the Processing Unit for every frame (SOURCE_MAC[1] is set to 1 – locally administered address).

55.6.1.27 ESC DL Control (ESC_DL_CONTROL_PDI)

Offset

Register	Offset
ESC_DL_CONTROL_PDI	100h

Function

** The possibility of RX FIFO Size reduction depends on the clock source accuracy of the ESC and of every connected EtherCAT/Ethernet devices (master, slave, etc.). RX FIFO Size of 7 is sufficient for 100ppm accuracy, FIFO Size 0 is possible with 25ppm accuracy (frame size of 1518/1522 Byte).

*** Reduction by 80 ns for IP Core since V3.0.0/V3.00c only, otherwise reduction by 40 ns

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BF25							BF24	BF23	Reserved	BF20		Reserved	BF16		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF14		BF12		BF10		BF8		BF2						BF1	BF0
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-25 BF25	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
24 BF24	Station alias: Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - Ignore Station Alias 1b - Alias can be used for all configured address command types (FPRD, FPWR, …)
23 BF23	Reserved, write 0 Bit field access for ECAT: r/w

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>Bit field access for PDI: r/-
Reset for bit is 0, later EEPROM word 5[7]</td></tr>
<tr><td>22
—</td><td>Reserved</td></tr>
<tr><td>21-20
BF20</td><td>Reserved, write 0
Bit field access for ECAT: r/w
Bit field access for PDI: r/-
Reset for bit is 0, later EEPROM word 5[5:4]</td></tr>
<tr><td>19
—</td><td>Reserved</td></tr>
<tr><td>18-16
BF16</td><td>RX FIFO Size (ESC delays start of forwarding until FIFO is at least half full).
RX FIFO Size/RX delay reduction** :
<table><tr><th>Value</th><th>MII</th></tr><tr><td>0</td><td>-40ns(-80ns***)</td></tr><tr><td>1</td><td>-40ns(-80ns***)</td></tr><tr><td>2</td><td>-40ns(-80ns***)</td></tr><tr><td>3</td><td>-40ns</td></tr><tr><td>4</td><td>no change</td></tr><tr><td>5</td><td>no change</td></tr><tr><td>6</td><td>no change</td></tr><tr><td>7</td><td>default</td></tr></table>
NOTE
EEPROM value is only taken over at first EEPROM load after power-on or reset
Bit field access for ECAT: r/w
Bit field access for PDI: r/-
Reset value for bit is IP Core since V2.4.3/V2.04d: 7, later EEPROM word 5[11:9] inverted</td></tr>
<tr><td>15-14
BF14</td><td>Reserved</td></tr>
<tr><td>13-12
BF12</td><td>Reserved</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-10 BF10	Loop Port 1: * Loop configuration changes are delayed until the end of a currently received or transmitted frame at the port. Bit field access for ECAT: r/w* Bit field access for PDI: r/- 00b - Auto 01b - Auto Close 10b - Open 11b - Closed
9-8 BF8	Loop Port 0: **NOTE** Loop open means sending/receiving over this port is enabled, loop closed means sending/receiving is disabled and frames are forwarded to the next open port internally. Auto: loop closed at link down, opened at link up Auto Close: loop closed at link down, opened with writing 01 again after link up (or receiving a valid Ethernet frame at the closed port) Open: loop open regardless of link state Closed: loop closed regardless of link state * Loop configuration changes are delayed until the end of a currently received or transmitted frame at the port. Bit field access for ECAT: r/w* Bit field access for PDI: r/- 00b - Auto 01b - Auto Close 10b - Open 11b - Closed
7-2 BF2	Reserved, Write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Temporary use of settings in 0x0100:0x0103[8:15]: Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - permanent use 1b - use for about 1 second, then revert to previous settings

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 BF0	Forwarding Rule Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - EtherCAT frames are processed, non-EtherCAT frames are forwarded without processing or modification. The source MAC address is not changed for any frame. 1b - EtherCAT frames are processed, non-EtherCAT frames are destroyed. The source MAC address is changed by the Processing Unit for every frame (SOURCE_MAC[1] is set to 1 – locally administered address).

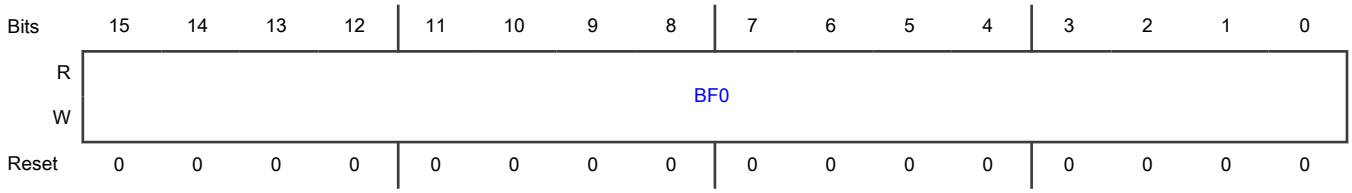
55.6.1.28 Physical Read Write Offset (PHYSICAL_READ_WRITE_OFFSET)

Offset

Register	Offset
PHYSICAL_READ_WRITE_OFFSET	108h

Function
Register Physical Read Write Offset

Diagram



Fields

Field	Function
15-0 BF0	This register is used for ReadWrite commands in Device Addressing mode (FPRW, APRW, BRW). The internal read address is directly taken from the offset address field of the EtherCAT datagram header, while the internal write address is calculated by adding the Physical Read/Write Offset value to the offset address field. Internal read address = ADR, internal write address = ADR + R/W-Offset Bit field access for ECAT: r/w Bit field access for PDI: r/-

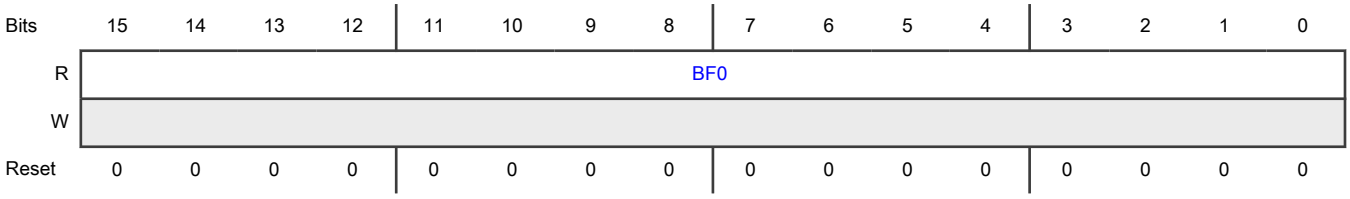
55.6.1.29 Physical Read Write Offset (PHYSICAL_READ_WRITE_OFFSET_PDI)

Offset

Register	Offset
PHYSICAL_READ_WRITE_OFFSET_PDI	108h

Function
Register Physical Read Write Offset

Diagram



Fields

Field	Function
15-0 BF0	This register is used for ReadWrite commands in Device Addressing mode (FPRW, APRW, BRW). The internal read address is directly taken from the offset address field of the EtherCAT datagram header, while the internal write address is calculated by adding the Physical Read/Write Offset value to the offset address field. Internal read address = ADR, internal write address = ADR + R/W-Offset Bit field access for ECAT: r/w Bit field access for PDI: r/-

55.6.1.30 ESC DL Status (ESC_DL_STATUS)

Offset

Register	Offset
ESC_DL_STATUS	110h

Function
Register ESC DL Status

Decoding port state in ESC DL Status register 0x0111(typical modes only)

Register	Port1	Port0
0x55	No link, closed	No link, closed
0x56	No link, closed	Link, open
0x59	Link, open	No link, closed
0x5A	Link, open	Link, open
0x65	No link, closed	No link, closed
0x66	No link, closed	No link, open
0x69	Link, open	No link, closed
0x6A	Link, open	Link, open
0x95	No link, closed	No link, closed
0x96	No link, closed	Link, open
0x99	Link, open	No link, closed
0x9A	Link, open	Link, open
0xA5	No link, closed	No link, closed
0xA6	No link, closed	Link, open
0xA9	Link, open	No link, closed
0xAA	Link, open	Link, open
0xD5	No link, closed	No link, closed
0xD6	No link, closed	Link, open
0xD9	Link, open	No link, closed
0xDA	Link, open	Link, open

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF15	BF14	BF13	BF12	BF11	BF10	BF9	BF8	BF7	BF6	BF5	BF4	BF3	BF2	BF1	BF0
W																
Reset	0	1	0	1	1	0	1	0	0	0	1	1	0	1	1	0

Fields

Field	Function
15 BF15	Reserved
14 BF14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 BF13	Reserved
12 BF12	Reserved
11 BF11	Communication on Port 1: Read access note: Reading DL Status register from ECAT clears ECAT Event Request 0x0210[2]. Avoid reading DL Status register from PDI. Bit field access for ECAT: r*/- Bit field access for PDI: r/- 0b - No stable communication 1b - Communication established
10 BF10	Loop Port 1 Read access note: Reading DL Status register from ECAT clears ECAT Event Request 0x0210[2]. Avoid reading DL Status register from PDI. Bit field access for ECAT: r*/- Bit field access for PDI: r/- 0b - Open 1b - Closed
9 BF9	Communication on Port 0: Read access note: Reading DL Status register from ECAT clears ECAT Event Request 0x0210[2]. Avoid reading DL Status register from PDI. Bit field access for ECAT: r*/- Bit field access for PDI: r/- 0b - No stable communication 1b - Communication established
8 BF8	Loop Port 0: Read access note: Reading DL Status register from ECAT clears ECAT Event Request 0x0210[2]. Avoid reading DL Status register from PDI. Bit field access for ECAT: r*/- Bit field access for PDI: r/- 0b - Open 1b - Closed
7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
BF7	
6 BF6	Reserved
5 BF5	Physical link on Port 1: Read access note: Reading DL Status register from ECAT clears ECAT Event Request 0x0210[2]. Avoid reading DL Status register from PDI. Bit field access for ECAT: r*/- Bit field access for PDI: r/- 0b - No link 1b - Link detected
4 BF4	Physical link on Port 0: Read access note: Reading DL Status register from ECAT clears ECAT Event Request 0x0210[2]. Avoid reading DL Status register from PDI. Bit field access for ECAT: r*/- Bit field access for PDI: r/- 0b - No link 1b - Link detected
3 BF3	Reserved Read access note: Reading DL Status register from ECAT clears ECAT Event Request 0x0210[2]. Avoid reading DL Status register from PDI. Bit field access for ECAT: r*/- Bit field access for PDI: r/-
2 BF2	Enhanced Link detection: EEPROM value is only transferred into this register at first EEPROM load after power-on or reset Read access note: Reading DL Status register from ECAT clears ECAT Event Request 0x0210[2]. Avoid reading DL Status register from PDI. Bit field access for ECAT: r*/- Bit field access for PDI: r/- Reset for bit is IP Core with feature: 1 until first EEPROM load, then 0 if EEPROM word 0[9]=0 and EEPROM word 0[15:12]=0x0, else 1 0b - Deactivated for all ports 1b - Activated for at least one port
1	PDI Watchdog Status:

Table continues on the next page...

Table continued from the previous page...

Field	Function
BF1	Read access note: Reading DL Status register from ECAT clears ECAT Event Request 0x0210[2]. Avoid reading DL Status register from PDI. Bit field access for ECAT: r*/- Bit field access for PDI: r/- 0b - Watchdog expired 1b - Watchdog reloaded
0 BF0	Register ESC DL Status Read access note: *Reading DL Status register from ECAT clears ECAT Event Request 0x0210[2]. Avoid reading DL Status register from PDI. Bit field access for ECAT: r*/- Bit field access for PDI: r/- 0b - PDI operational/EEPROM loaded correctly: 1b - EEPROM loaded correctly, PDI operational (access to Process Data RAM)

55.6.1.31 AL Control (AL_CONTROL)

Offset

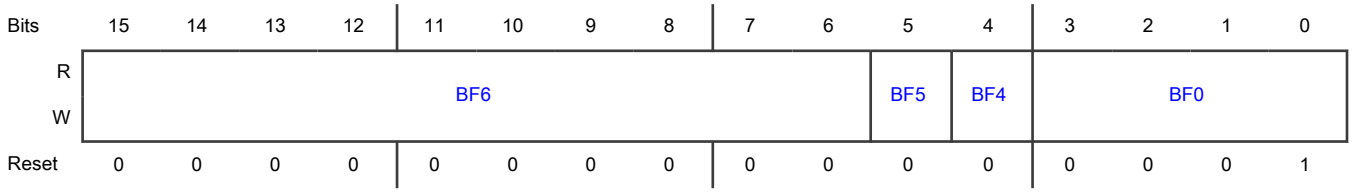
Register	Offset
AL_CONTROL	120h

Function

NOTE

AL Control register behaves like a mailbox if Device Emulation is off (0x0141[0]=0): The PDI has to read/write* the AL Control register after ECAT has written it. Otherwise ECAT cannot write again to the AL Control register. After Reset, AL Control register can be written by ECAT. (Regarding mailbox functionality, both low and high byte of the AL Control register trigger read/write functions, e.g., reading 0x0121 is sufficient to make this register writable again).

Diagram



Fields

Field	Function
15-6 BF6	Reserved, write 0 Bit field access for ECAT: r/(w) Bit field access for PDI: r/-
5 BF5	Device Identification: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - No request 1b - Device Identification request
4 BF4	Error Ind Ack Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - No Ack of Error Ind in AL status register 1b - Ack of Error Ind in AL status register
3-0 BF0	Initiate State Transition of the Device State Machine: 1: Request Init State 3: Request Bootstrap State 2: Request Pre-Operational State 4: Request Safe-Operational State 8: Request Operational State Bit field access for ECAT: r/(w) Bit field access for PDI: r/-

55.6.1.32 AL Control (AL_CONTROL_PDI)

Offset

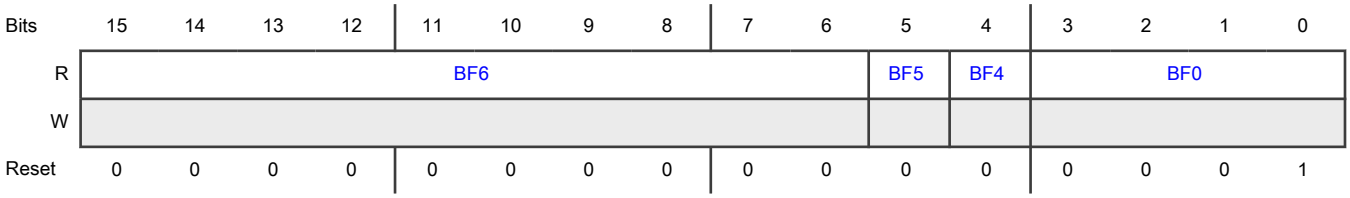
Register	Offset
AL_CONTROL_PDI	120h

Function

NOTE

AL Control register behaves like a mailbox if Device Emulation is off (0x0141[0]=0): The PDI has to read/write* the AL Control register after ECAT has written it. Otherwise ECAT cannot write again to the AL Control register. After Reset, AL Control register can be written by ECAT. (Regarding mailbox functionality, both low and high byte of the AL Control register trigger read/write functions, e.g., reading 0x0121 is sufficient to make this register writable again).

Diagram



Fields

Field	Function
15-6 BF6	Reserved, write 0 Bit field access for ECAT: r/(w) Bit field access for PDI: r/-
5 BF5	Device Identification: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - No request 1b - Device Identification request
4 BF4	Error Ind Ack Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - No Ack of Error Ind in AL status register 1b - Ack of Error Ind in AL status register
3-0 BF0	Initiate State Transition of the Device State Machine: 1: Request Init State 3: Request Bootstrap State 2: Request Pre-Operational State 4: Request Safe-Operational State 8: Request Operational State Bit field access for ECAT: r/(w) Bit field access for PDI: r/-

55.6.1.33 AL Status (AL_STATUS)

Offset

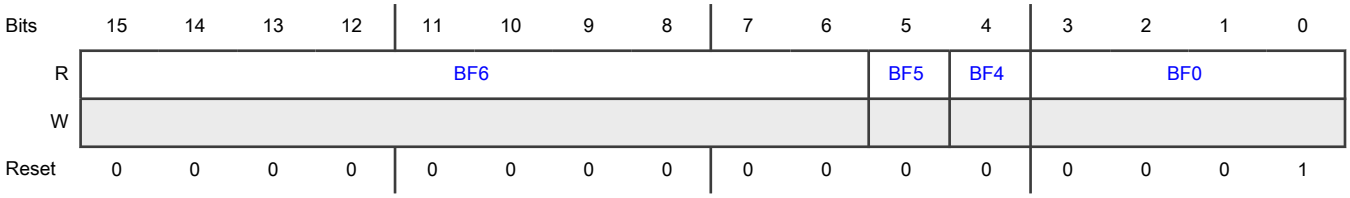
Register	Offset
AL_STATUS	130h

Function

NOTE

AL Status register is only writable from PDI if Device Emulation is off (0x0141[0]=0), otherwise AL Status register will reflect AL Control register values. Avoid reading AL Status register from PDI.

Diagram



Fields

Field	Function
15-6 BF6	Reserved, write 0 Bit field access for ECAT: r*/-Bit field access for PDI: r/(w). * Reading AL Status from ECAT clears ECAT Event Request 0x0210[3].
5 BF5	Device Identification: Bit field access for ECAT: r*/-Bit field access for PDI: r/(w). * Reading AL Status from ECAT clears ECAT Event Request 0x0210[3]. 0b - Device Identification not valid 1b - Device Identification loaded
4 BF4	Error Ind: Bit field access for ECAT: r*/-Bit field access for PDI: r/(w). * Reading AL Status from ECAT clears ECAT Event Request 0x0210[3]. 0b - Device is in State as requested or Flag cleared by command 1b - Device has not entered requested State or changed State as result of a local action
3-0 BF0	Actual State of the Device State Machine: 1: Init State3: Bootstrap State 2: Pre-Operational State 4: Safe-Operational State 8: Operational StateBit field access for ECAT: r*/-Bit field access for PDI: r/(w). * Reading AL Status from ECAT clears ECAT Event Request 0x0210[3].

55.6.1.34 AL Status (AL_STATUS_PDI)

Offset

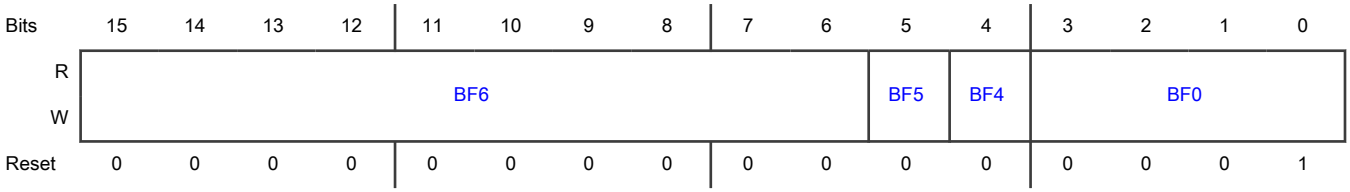
Register	Offset
AL_STATUS_PDI	130h

Function

NOTE

AL Status register is only writable from PDI if Device Emulation is off (0x0141[0]=0), otherwise AL Status register will reflect AL Control register values. Avoid reading AL Status register from PDI.

Diagram



Fields

Field	Function
15-6 BF6	Reserved, write 0 Bit field access for ECAT: r*/-Bit field access for PDI: r/(w). * Reading AL Status from ECAT clears ECAT Event Request 0x0210[3].
5 BF5	Device Identification: Bit field access for ECAT: r*/-Bit field access for PDI: r/(w). * Reading AL Status from ECAT clears ECAT Event Request 0x0210[3]. 0b - Device Identification not valid 1b - Device Identification loaded
4 BF4	Error Ind: Bit field access for ECAT: r*/-Bit field access for PDI: r/(w). * Reading AL Status from ECAT clears ECAT Event Request 0x0210[3]. 0b - Device is in State as requested or Flag cleared by command 1b - Device has not entered requested State or changed State as result of a local action
3-0 BF0	Actual State of the Device State Machine: 1: Init State3: Bootstrap State 2: Pre-Operational State 4: Safe-Operational State 8: Operational StateBit field access for ECAT: r*/-Bit field access for PDI: r/(w). * Reading AL Status from ECAT clears ECAT Event Request 0x0210[3].

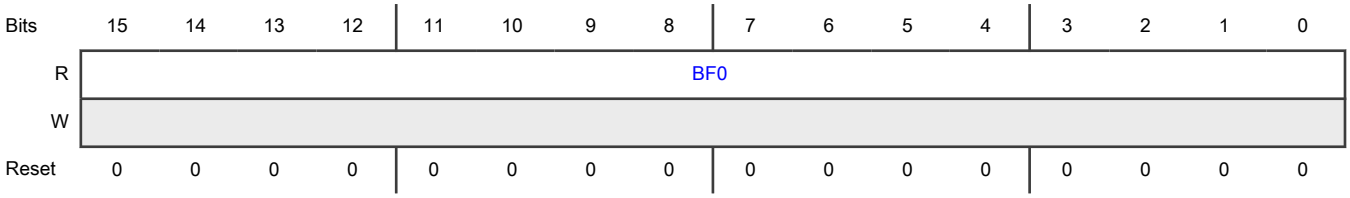
55.6.1.35 AL Status Code (AL_STATUS_CODE)

Offset

Register	Offset
AL_STATUS_CODE	134h

Function
Register AL Status Code

Diagram



Fields

Field	Function
15-0 BF0	AL Status Code Bit field access for ECAT: r/- Bit field access for PDI: r/w

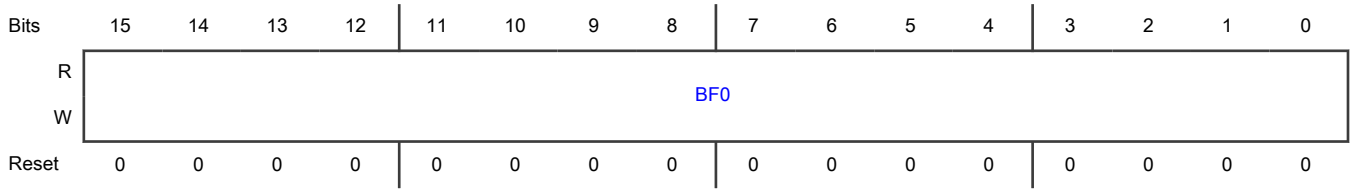
55.6.1.36 AL Status Code (AL_STATUS_CODE_PDI)

Offset

Register	Offset
AL_STATUS_CODE_PDI	134h

Function
Register AL Status Code

Diagram



Fields

Field	Function
15-0 BF0	AL Status Code Bit field access for ECAT: r/- Bit field access for PDI: r/w

55.6.1.37 RUN LED Override (RUN_LED_OVERRIDE)

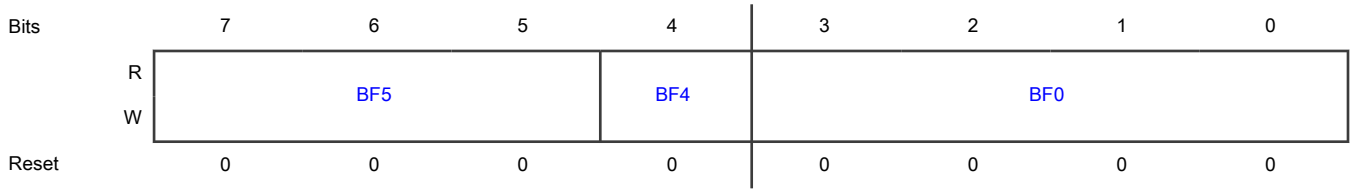
Offset

Register	Offset
RUN_LED_OVERRIDE	138h

Function

NOTE: Changes to AL Status register (0x0130) with valid values will disable RUN LED Override (0x0138[4]=0). The value read in this register always reflects current LED output.

Diagram



Fields

Field	Function
7-5 BF5	Reserved, write 0 Bit field access for ECAT: r/w Bit field access for PDI: r/w
4 BF4	Enable Override: Bit field access for ECAT: r/w

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>Bit field access for PDI: r/w
0b - Override disabled
1b - Override enabled</td></tr>
<tr><td>3-0
BF0</td><td>LED code and AL Status
<table><tr><th>LED Code</th><th>Description</th><th>AL Status</th></tr><tr><td>0x0</td><td>Off</td><td>Init(1)</td></tr><tr><td>0x1</td><td>Flash 1x</td><td>SafeOp(4)</td></tr><tr><td>0x2-0xC</td><td>Flash 2x - 12x</td><td>-</td></tr><tr><td>0xD</td><td>Blinking</td><td>PreOp(2)</td></tr><tr><td>0xE</td><td>Flickering</td><td>Bootstrap (3)</td></tr><tr><td>0xF</td><td>On</td><td>Operational (8)</td></tr></table>Bit field access for ECAT: r/w
Bit field access for PDI: r/w</td></tr>
</table>

55.6.1.38 ERR LED Override (ERR_LED_OVERRIDE)

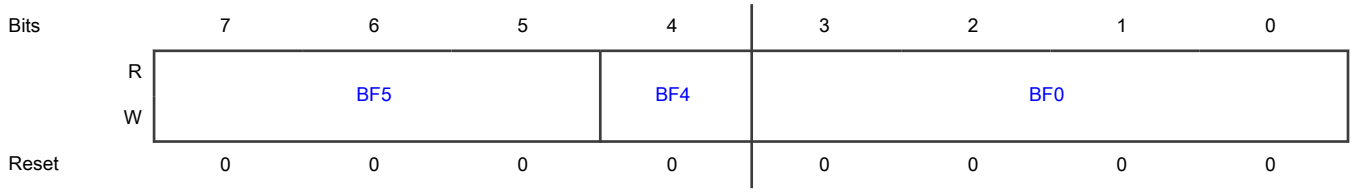
Offset

Register	Offset
ERR_LED_OVERRIDE	139h

Function

NOTE: New error conditions will disable ERR LED Override (0x0139[4]=0). The value read in this register always reflects current LED output.

Diagram



Fields

Field	Function
7-5 BF5	Reserved, write 0 Bit field access for ECAT: r/w Bit field access for PDI: r/w
4 BF4	Enable Override: Bit field access for ECAT: r/w Bit field access for PDI: r/w 0b - Override disabled 1b - Override enabled
3-0 BF0	LED code **LED Code / Description** 0x0 — Off 0x1-0xC — Flash 1x - 12x 0xD — Blinking 0xE — Flickering 0xF — On Bit field access for ECAT: r/w Bit field access for PDI: r/w

55.6.1.39 PDI Control (PDI_CONTROL)

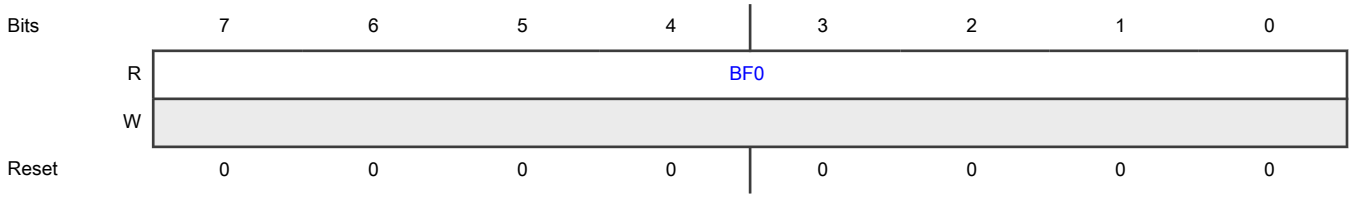
Offset

Register	Offset
PDI_CONTROL	140h

Function

Register PDI Control

Diagram



Fields

Field	Function
7-0 BF0	Process data interface: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration. 0000_0000b - Interface deactivated (no PDI) 0000_0001b - 4 Digital Input 0000_0010b - 4 Digital Output 0000_0011b - 2 Digital Input and 2 Digital Output 0000_0100b - Digital I/O 0000_0101b - SPI Slave 0000_0110b - Oversampling I/O 0000_0111b - Reserved 0000_1000b - 16 Bit asynchronous Microcontroller interface 0000_1001b - 8 Bit asynchronous Microcontroller interface 0000_1010b - 16 Bit synchronous Microcontroller interface 0000_1011b - 8 Bit synchronous Microcontroller interface 0001_0000b - 32 Digital Input and 0 Digital Output 0001_0001b - 24 Digital Input and 8 Digital Output 0001_0010b - 16 Digital Input and 16 Digital Output 0001_0011b - 8 Digital Input and 24 Digital Output 0001_0100b - 0 Digital Input and 32 Digital Output 1000_0000b - On-chip bus. Others: Reserved

55.6.1.40 ESC Configuration (ESC_CONFIGURATION)

Offset

Register	Offset
ESC_CONFIGURATION	141h

Function

NOTE

EEPROM values of bits 1, 4, 5, 6, and 7 are only transferred into this register at first EEPROM load after power-on or reset.

Diagram

Bits	7	6	5	4	3	2	1	0
R	BF7	BF6	BF5	BF4	BF3	BF2	BF1	BF0
W								
Reset	1	1	1	1	1	1	1	0

Fields

Field	Function
7 BF7	Reserved
6 BF6	Reserved
5 BF5	Enhanced Link port 1: Bit field access for ECAT: r/- Bit field access for PDI: r/- 1, later EEPROM word 0 0b - disabled (if bit 1=0) 1b - enabled
4 BF4	Enhanced Link port 0: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset for bit is 1, later EEPROM word 0 0b - disabled (if bit 1=0) 1b - enabled
3 BF3	Distributed Clocks Latch In Unit: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration. 0b - disabled (power saving) 1b - enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 BF2	Distributed Clocks SYNC Out Unit: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration. 0b - disabled (power saving) 1b - enabled
1 BF1	Enhanced Link detection all ports: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset for bit is 1, later EEPROM word 0 0b - disabled (if bits [7:4]=0) 1b - enabled at all ports (overrides bits [7:4])
0 BF0	Device emulation (control of AL status): Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset for bit is 1 with Digital I/O PDI, PDI_EMULATION pin with µC/On-chip bus 0b - AL status register has to be set by PDI 1b - AL status register will be set to value written to AL control register

55.6.1.41 PDI Information (PDI_INFORMATION)

Offset

Register	Offset
PDI_INFORMATION	14Eh

Function

Offset address for this register

was not found in RTL code

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF4												BF3	BF2	BF1	BF0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-4 BF4	Reserved Bit field access for ECAT: r/- Bit field access for PDI: r/-
3 BF3	PDI configuration invalid: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - PDI configuration ok 1b - PDI configuration invalid
2 BF2	PDI active: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - PDI not active 1b - PDI active
1 BF1	ESC configuration area loaded from EEPROM: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - not loaded 1b - loaded
0 BF0	PDI function Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration. 0b - Disabled 1b - Enabled

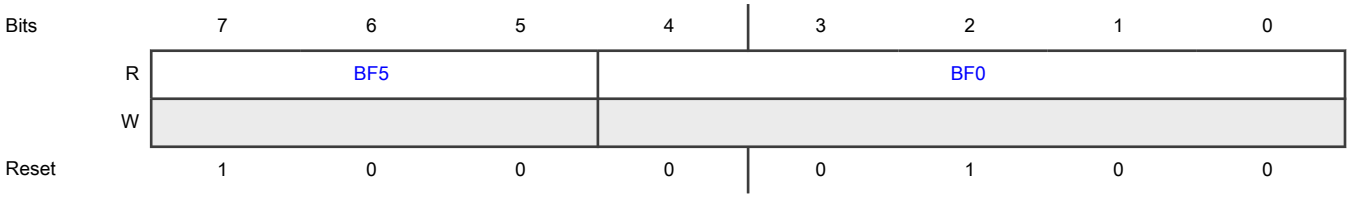
55.6.1.42 Register PDI On-chip bus configuration (PDI_ON_CHIP_BUS_CONFIGURATION)

Offset

Register	Offset
PDI_ON_CHIP_BUS_CONFIGURATION	150h

Function
Register PDI On-chip bus configuration

Diagram



Fields

Field	Function
7-5 BF5	On-chip bus Bit field access for ECAT: r/-Bit field access for PDI: r/-Reset value for bit depends on the configuration. 000b - Intel® Avalon® 001b - AXI® 010: Xilinx® PLB v4.6 100b - Xilinx OPB
4-0 BF0	On-chip bus clock: Bit field access for ECAT: r/-Bit field access for PDI: r/-Reset value for bit depends on the configuration. 0_0000b - asynchronous 0_0001b-1_1111b - synchronous multiplication factor (N * 25 MHz)

55.6.1.43 PDI Configuration Sync Latch 1 and 0 PDI Configuration (SYNC_LATCH_1_AND_0_PDI_CONFIGURATION)

Offset

Register	Offset
SYNC_LATCH_1_AND_0_PDI_CONFIGURATION	151h

Function

* The IP Core has concurrent SYNC[1:0] outputs and LATCH[1:0] inputs, independent of this configuration.

Diagram

Bits	7	6	5	4	3	2	1	0
R	BF7	BF6	BF4		BF3	BF2	BF0	
W								
Reset	1	1	1	0	1	1	1	0

Fields

Field	Function
7 BF7	SYNC1 mapped to AL Event Request register 0x0220[3]: Bit field access for ECAT: r/-Bit field access for PDI: r/-Reset value for bit depends on the configuration. 0b - Disabled 1b - Enabled
6 BF6	SYNC1/LATCH1 configuration* Bit field access for ECAT: r/-Bit field access for PDI: r/- 0b - LATCH1 input 1b - SYNC1 output
5-4 BF4	SYNC1 output driver/polarity: Bit field access for ECAT: r/-Bit field access for PDI: r/- 00b - Push-Pull active low 01b - Open Drain (active low) 10b - Push-Pull active high 11b - Open Source (active high)
3 BF3	SYNC0 mapped to AL Event Request register 0x0220[2]: Bit field access for ECAT: r/-Bit field access for PDI: r/-Reset for bit depends on configuration 0b - Disabled 1b - Enabled
2 BF2	SYNC0/LATCH0 configuration*: Bit field access for ECAT: r/-Bit field access for PDI: r/- 0b - LATCH0 Input 1b - SYNC0 Output
1-0 BF0	SYNC0 output driver/polarity: Bit field access for ECAT: r/-Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Push-Pull active low 01b - Open Drain (active low) 10b - Push-Pull active high 11b - Open Source (active high)

55.6.1.44 Register PDI On-chip bus extended configuration. (PDI_ON_CHIP_BUS_EXTENDED_CONFIGURATION)

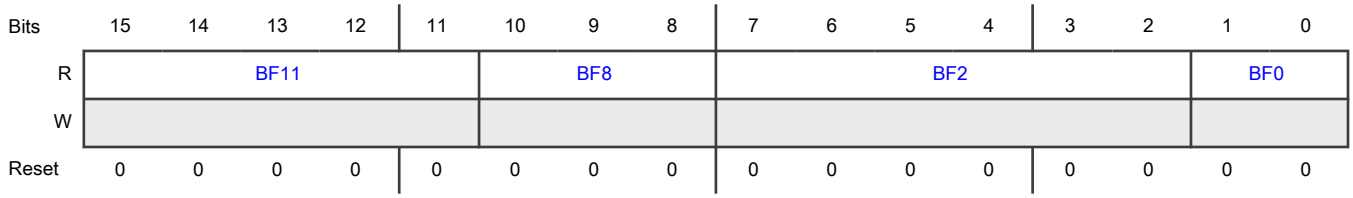
Offset

Register	Offset
PDI_ON_CHIP_BUS_EXTENDED_CONFIGURATION	152h

Function

Register PDI On-chip bus extended configuration has PDI number 0x80.

Diagram



Fields

Field	Function
15-11	Reserved
BF11	Bit field access for ECAT: r/-Bit field access for PDI: r/-Reset value for bit depends on the configuration.
10-8	On-chip bus sub-type for AXI:
BF8	Bit field access for ECAT: r/-Bit field access for PDI: r/-Reset value for bit depends on the configuration. 000b - AXI3 001b - AXI4 010b - AXI4 LITE
7-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
BF2	Bit field access for ECAT: r/-Bit field access for PDI: r/-Reset value for bit depends on the configuration.
1-0 BF0	Read prefetch size (in cycles of PDI width): Bit field access for ECAT: r/-Bit field access for PDI: r/-Reset value for bit depends on the configuration. 00b - 4 cycles 01b - 1 cycle (typical) 10b - 2 cycles 11b - Reserved

55.6.1.45 ECAT Event Mask (ECAT_EVENT_MASK)

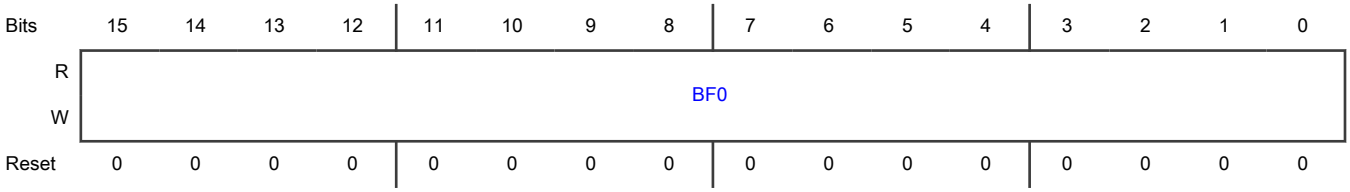
Offset

Register	Offset
ECAT_EVENT_MASK	200h

Function

Register ECAT Event Mask

Diagram



Fields

Field	Function
15-0 BF0	ECAT Event masking of the ECAT Event Request Events for mapping into ECAT event field of EtherCAT frames: Bit field access for ECAT: r/wBit field access for PDI: r/- 0000_0000_0000_0000b - Corresponding ECAT Event Request register bit is not mapped 0000_0000_0000_0001b - Corresponding ECAT Event Request register bit is mapped

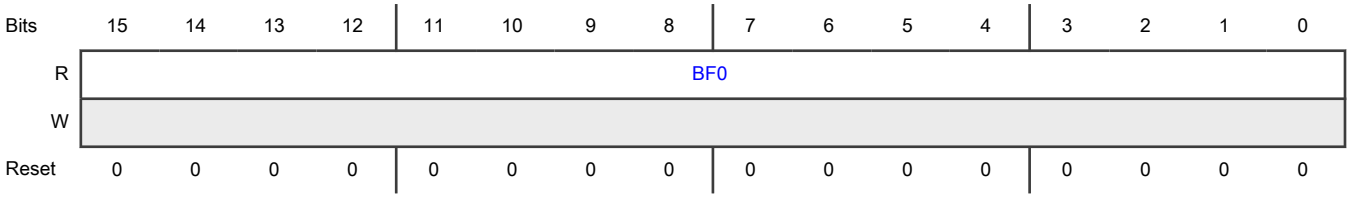
55.6.1.46 ECAT Event Mask (ECAT_EVENT_MASK_PDI)

Offset

Register	Offset
ECAT_EVENT_MASK_PDI	200h

Function
Register ECAT Event Mask

Diagram



Fields

Field	Function
15-0 BF0	ECAT Event masking of the ECAT Event Request Events for mapping into ECAT event field of EtherCAT frames: Bit field access for ECAT: r/wBit field access for PDI: r/- 0000_0000_0000_0000b - Corresponding ECAT Event Request register bit is not mapped 0000_0000_0000_0001b - Corresponding ECAT Event Request register bit is mapped

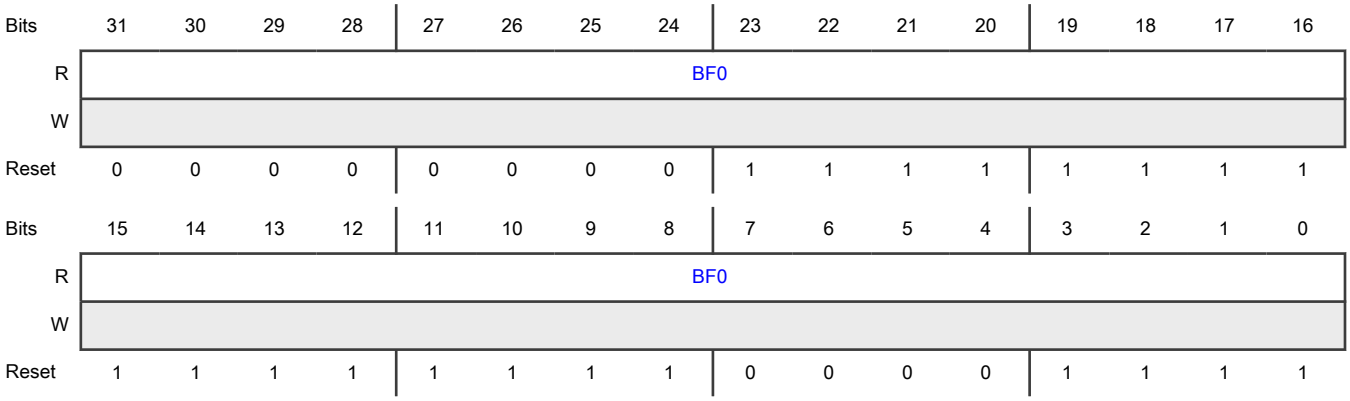
55.6.1.47 PDI AL Event Mask (PDI_AL_EVENT_MASK)

Offset

Register	Offset
PDI_AL_EVENT_MASK	204h

Function
Register PDI AL Event Mask

Diagram



Fields

Field	Function
31-0 BF0	AL Event masking of the AL Event Request register Events for mapping to PDI IRQ signal: Bit field access for ECAT: r/-Bit field access for PDI: r/wReset for bit is 0x00FF:0xFF0F 0000_0000_0000_0000_0000_0000_0000_0000b - Corresponding AL Event Request register bit is not mapped 0000_0000_0000_0000_0000_0000_0000_0001b - Corresponding AL Event Request register bit is mapped

55.6.1.48 PDI AL Event Mask (PDI_AL_EVENT_MASK_PDI)

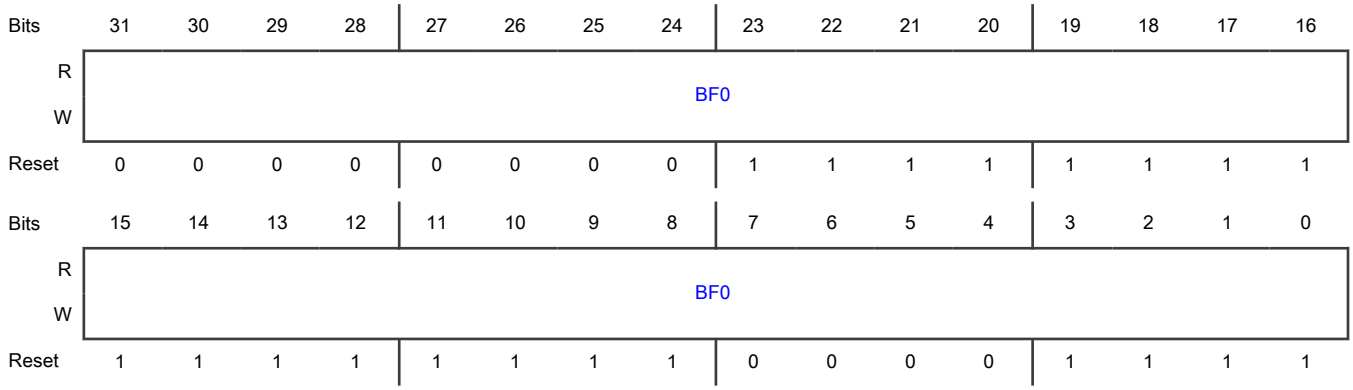
Offset

Register	Offset
PDI_AL_EVENT_MASK_PDI	204h

Function

Register PDI AL Event Mask

Diagram



Fields

Field	Function
31-0 BF0	AL Event masking of the AL Event Request register Events for mapping to PDI IRQ signal: Bit field access for ECAT: r/-Bit field access for PDI: r/wReset for bit is 0x00FF:0xFF0F 0000_0000_0000_0000_0000_0000_0000_0000b - Corresponding AL Event Request register bit is not mapped 0000_0000_0000_0000_0000_0000_0000_0001b - Corresponding AL Event Request register bit is mapped

55.6.1.49 ECAT Event Request (ECAT_EVENT_REQUEST)

Offset

Register	Offset
ECAT_EVENT_REQUEST	210h

Function

Register ECAT Event Request

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF12				BF11	BF10	BF9	BF8	BF7	BF6	BF5	BF4	BF3	BF2	BF1	BF0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
15-12 BF12	Reserved Bit field access for ECAT: r/- Bit field access for PDI: r/-
11 BF11	Mirrors values of each SyncManager Status: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No Sync Channel 7 event 1b - Sync Channel 7 event pending
10 BF10	Mirrors values of each SyncManager Status: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No Sync Channel 6 event 1b - Sync Channel 6 event pending
9 BF9	Mirrors values of each SyncManager Status: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No Sync Channel 5 event 1b - Sync Channel 5 event pending
8 BF8	Mirrors values of each SyncManager Status: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No Sync Channel 4 event 1b - Sync Channel 4 event pending
7 BF7	Mirrors values of each SyncManager Status: Bit field access for ECAT: r/- Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No Sync Channel 3 event 1b - Sync Channel 3 event pending
6 BF6	Mirrors values of each SyncManager Status: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No Sync Channel 2 event 1b - Sync Channel 2 event pending
5 BF5	Mirrors values of each SyncManager Status: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No Sync Channel 1 event 1b - Sync Channel 1 event pending
4 BF4	Mirrors values of each SyncManager Status: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No Sync Channel 0 event 1b - Sync Channel 0 event pending
3 BF3	AL Status event: (Bit is cleared by reading out AL Status 0x0130:0x0131 from ECAT) Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No change in AL Status 1b - AL Status change
2 BF2	DL Status event: (Bit is cleared by reading out DL Status 0x0110:0x0111 from ECAT) Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No change in DL Status 1b - DL Status change
1 BF1	Reserved Bit field access for ECAT: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Bit field access for PDI: r/-
0 BF0	DC Latch event: (Bit is cleared by reading DC Latch event times from ECAT for ECAT-controlled Latch Units, so that Latch 0/1 Status 0x09AE:0x09AF indicates no event) Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No change on DC Latch Inputs 1b - At least one change on DC Latch Inputs

55.6.1.50 AL Event request (AL_EVENT_REQUEST)

Offset

Register	Offset
AL_EVENT_REQUEST	220h

Function
Register AL Event Request

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BF16															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF15	BF14	BF13	BF12	BF11	BF10	BF9	BF8	BF7	BF6	BF5	BF4	BF3	BF2	BF1	BF0
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Fields

Field	Function
31-16	Reserved
BF16	Bit field access for ECAT: r/- Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 BF15	Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No SyncManager 7 interrupt 1b - SyncManager 7 interrupt pending
14 BF14	Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No SyncManager 6 interrupt 1b - SyncManager6 interrupt pending
13 BF13	Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No SyncManager 5 interrupt 1b - SyncManager 5 interrupt pending
12 BF12	Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No SyncManager4 interrupt 1b - SyncManager 4 interrupt pending
11 BF11	Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No SyncManager 3 interrupt 1b - SyncManager 3 interrupt pending
10 BF10	Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No SyncManager2 interrupt 1b - SyncManager 2 interrupt pending
9 BF9	Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No SyncManager 1 interrupt 1b - SyncManager 1 interrupt pending
8 BF8	SyncManager interrupts (SyncManager register offset 0x5, bit [0] or [1]): Bit field access for ECAT: r/- Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No SyncManager 0 interrupt 1b - SyncManager 0 interrupt pending
7 BF7	Reserved Bit field access for ECAT: r/- Bit field access for PDI: r/-
6 BF6	Watchdog Process Data: (Bit is cleared by reading Watchdog Status Process Data 0x0440 from PDI) Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Has not expired 1b - Has expired
5 BF5	EEPROM Emulation: (Bit is cleared by acknowledging the command in EEPROM Control/Status register 0x0502:0x0503[10:8] from PDI) Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No command pending 1b - EEPROM command pending
4 BF4	SyncManager activation register (SyncManager register offset 0x6) changed: (Bit is cleared by reading SyncManager Activation registers 0x0806 etc. from PDI) Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No change in any SyncManager 1b - At least one SyncManager changed
3 BF3	State of DC SYNC1 (if register 0x0151[7]=1): (Bit is cleared by reading of SYNC1 status 0x098F from PDI, use only in Acknowledge mode) Bit field access for ECAT: r/- Bit field access for PDI: r/-
2 BF2	State of DC SYNC0 (if register 0x0151[3]=1): (Bit is cleared by reading SYNC0 status 0x098E from PDI, use only in Acknowledge mode) Bit field access for ECAT: r/- Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 BF1	DC Latch event: (Bit is cleared by reading DC Latch event times from PDI, so that Latch 0/1 Status 0x09AE:0x09AF indicates no event. Available if Latch Unit is PDI-controlled) Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No change on DC Latch Inputs 1b - At least one change on DC Latch Inputs
0 BF0	AL Control event: (Bit is cleared by reading AL Control register 0x0120:0x0121 from PDI) Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No AL Control Register change 1b - AL Control Register has been written3

55.6.1.51 RX Error Counter (RX_ERROR_COUNTER_PORT0 - RX_ERROR_COUNTER_PORT1)

Offset

Register	Offset
RX_ERROR_COUNTER_PORT0	300h
RX_ERROR_COUNTER_PORT1	302h

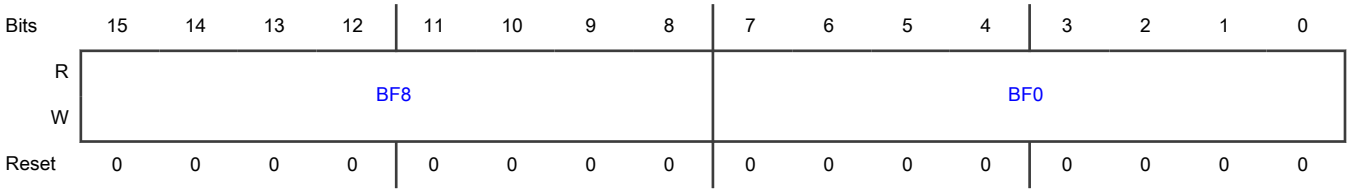
Function

Errors are only counted if the corresponding port is enabled.

NOTE

Error Counters 0x0300-0x030B are cleared if one of the implemented RX Error counters 0x0300-0x030B is written (preferably 0x0300). Write value is ignored (write 0). Errors are only counted if the loop of the port is open.

Diagram



Fields

Field	Function
15-8 BF8	RX Error counter of Port y (counting is stopped when 0xFF is reached). Bit field access for ECAT: r/ w(clr). Bit field access for PDI: r/-
7-0 BF0	Invalid frame counter of Port y (counting is stopped when 0xFF is reached). Bit field access for ECAT: r/ w(clr). Bit field access for PDI: r/-

55.6.1.52 RX Error Counter (RX_ERROR_COUNTER_PORT0_PDI - RX_ERROR_COUNTER_PORT1_PDI)

Offset

Register	Offset
RX_ERROR_COUNTER_PORT0_PDI	300h
RX_ERROR_COUNTER_PORT1_PDI	302h

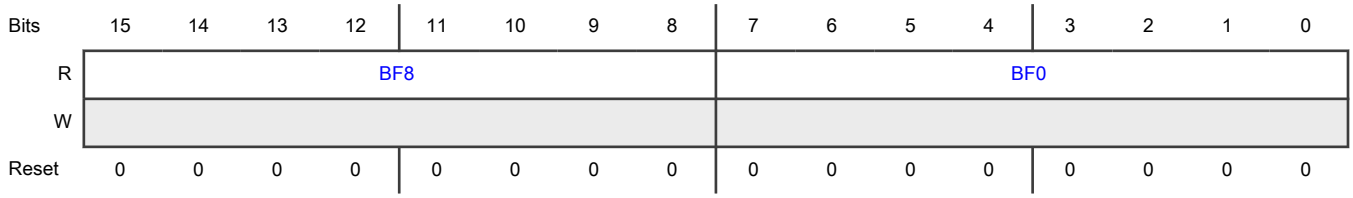
Function

Errors are only counted if the corresponding port is enabled.

NOTE

Error Counters 0x0300-0x030B are cleared if one of the implemented RX Error counters 0x0300-0x030B is written (preferably 0x0300). Write value is ignored (write 0). Errors are only counted if the loop of the port is open.

Diagram



Fields

Field	Function
15-8 BF8	RX Error counter of Port y (counting is stopped when 0xFF is reached). Bit field access for ECAT: r/ w(clr). Bit field access for PDI: r/-
7-0 BF0	Invalid frame counter of Port y (counting is stopped when 0xFF is reached). Bit field access for ECAT: r/ w(clr). Bit field access for PDI: r/-

55.6.1.53 Forwarded RX Error Counter (FORWARDED_RX_ERROR_COUNTER_PORT0 - FORWARDED_RX_ERROR_COUNTER_PORT1)

Offset

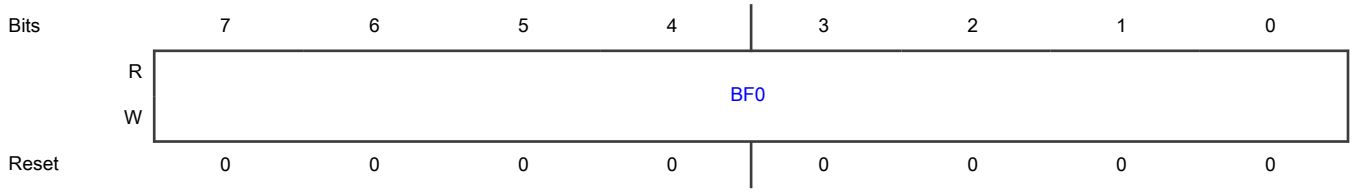
Register	Offset
FORWARDED_RX_ERROR_COUNTER_PORT0	308h
FORWARDED_RX_ERROR_COUNTER_PORT1	309h

Function

NOTE

Error Counters 0x0300-0x030B are cleared if one of the implemented RX Error counters 0x0300-0x030B is written (preferably 0x0300). Write value is ignored (write 0). Errors are only counted if the loop of the port is open.

Diagram



Fields

Field	Function
7-0 BF0	Forwarded error counter of Port y (counting is stopped when 0xFF is reached). Bit field access for ECAT: r/ w(clr) Bit field access for PDI: r/-

55.6.1.54 Forwarded RX Error Counter (FORWARDED_RX_ERROR_COUNTER_PORT0_PDI - FORWARDED_RX_ERROR_COUNTER_PORT1_PDI)

Offset

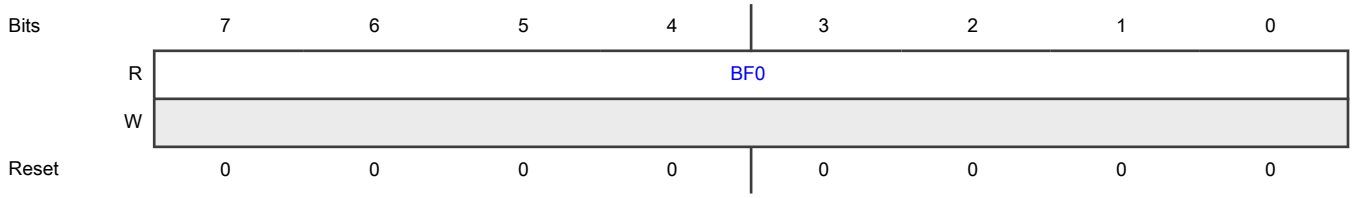
Register	Offset
FORWARDED_RX_ERROR_COUNTER_PORT0_PDI	308h
FORWARDED_RX_ERROR_COUNTER_PORT1_PDI	309h

Function

NOTE

Error Counters 0x0300-0x030B are cleared if one of the implemented RX Error counters 0x0300-0x030B is written (preferably 0x0300). Write value is ignored (write 0). Errors are only counted if the loop of the port is open.

Diagram



Fields

Field	Function
7-0 BF0	Forwarded error counter of Port y (counting is stopped when 0xFF is reached). Bit field access for ECAT: r/ w(clr) Bit field access for PDI: r/-

55.6.1.55 ECAT Processing Unit Error Counter (ECAT_PROCESSING_UNIT_ERROR_COUNTER)

Offset

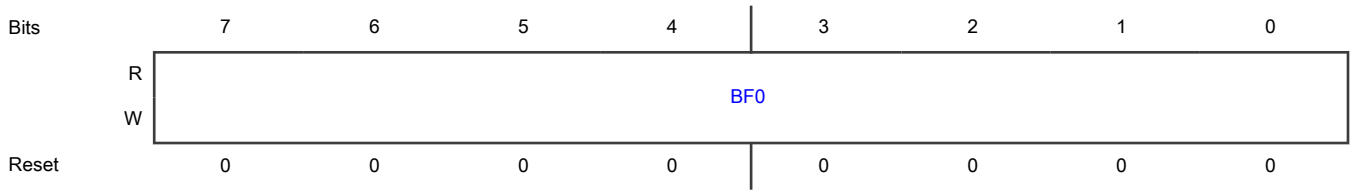
Register	Offset
ECAT_PROCESSING_UNIT_ERROR_COUNTER	30Ch

Function

NOTE

Error Counter 0x030C is cleared if error counter 0x030C is written. Write value is ignored (write 0).

Diagram



Fields

Field	Function
7-0 BF0	ECAT Processing Unit error counter (counting is stopped when 0xFF is reached). Counts errors of frames passing the Processing Unit. Bit field access for ECAT: r/ w(clr) Bit field access for PDI: r/-

55.6.1.56 ECAT Processing Unit Error Counter (ECAT_PROCESSING_UNIT_ERROR_COUNTER_PDI)

Offset

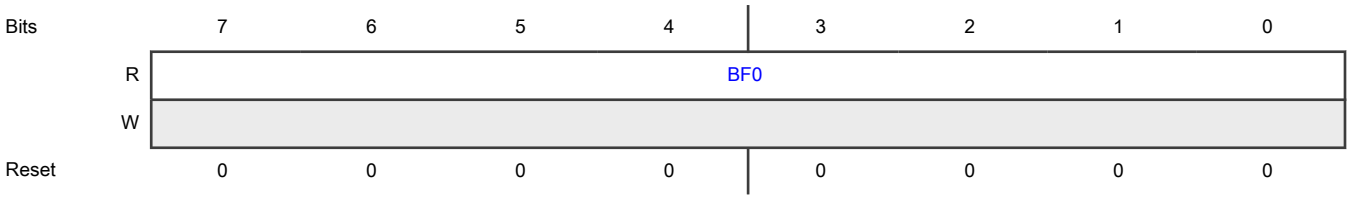
Register	Offset
ECAT_PROCESSING_UNIT_ERROR_COUNTER_PDI	30Ch

Function

NOTE

Error Counter 0x030C is cleared if error counter 0x030C is written. Write value is ignored (write 0).

Diagram



Fields

Field	Function
7-0 BF0	ECAT Processing Unit error counter (counting is stopped when 0xFF is reached). Counts errors of frames passing the Processing Unit. Bit field access for ECAT: r/ w(clr) Bit field access for PDI: r/-

55.6.1.57 PDI Error counter (PDI_ERROR_COUNTER)

Offset

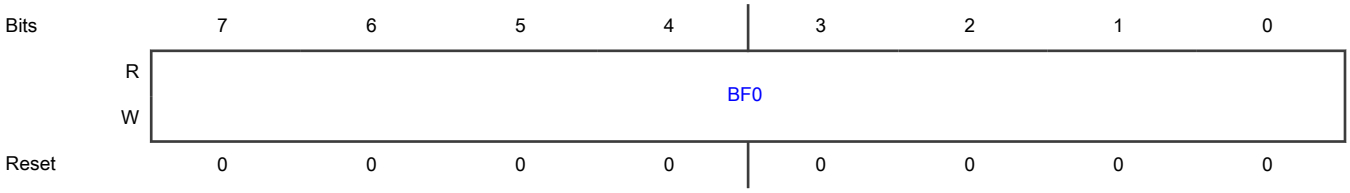
Register	Offset
PDI_ERROR_COUNTER	30Dh

Function

NOTE

Error Counter 0x030D and Error Code 0x030E are cleared if error counter 0x030D is written. Write value is ignored (write 0).

Diagram



Fields

Field	Function
7-0 BF0	PDI Error counter (counting is stopped when 0xFF is reached). Counts if a PDI access has an interface error. Bit field access for ECAT: r/ w(clr) Bit field access for PDI: r/-

55.6.1.58 PDI Error counter (PDI_ERROR_COUNTER_PDI)

Offset

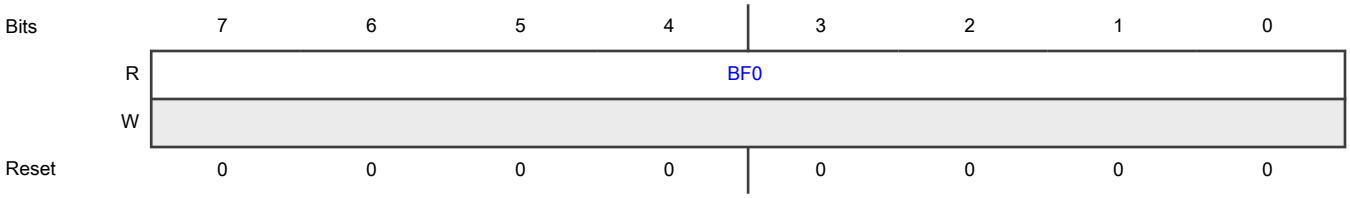
Register	Offset
PDI_ERROR_COUNTER_PDI	30Dh

Function

NOTE

Error Counter 0x030D and Error Code 0x030E are cleared if error counter 0x030D is written. Write value is ignored (write 0).

Diagram



Fields

Field	Function
7-0 BF0	PDI Error counter (counting is stopped when 0xFF is reached). Counts if a PDI access has an interface error. Bit field access for ECAT: r/ w(clr) Bit field access for PDI: r/-

55.6.1.59 ASYNCHRONOUS_SYNCHRONOUS_MICROCONTROLLER_PDI_ERROR_CODE. (ASYNCHRONOUS_SYNCHRONOUS_MICROCONTROLLER)

Offset

Register	Offset
ASYNCHRONOUS_SYNCHRONOUS_MICROCONTROLLER	30Eh

Function

Reasons for last PDI Error.

NOTE

Error Counter 0x030D and Error Code 0x030E are cleared if error counter 0x030D is written. Write value is ignored (write 0).

Diagram

Bits	7	6	5	4	3	2	1	0
R	BF4				BF3	BF2	BF1	BF0
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-4 BF4	reserved Bit field access for ECAT: r/- Bit field access for PDI: r/-
3 BF3	Addressing error for a write access (odd address without BHE) Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - no error 1b - error detected
2 BF2	Addressing error for a read access (odd address without BHE) Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - no error 1b - error detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 BF1	Busy violation during write access Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - no error 1b - error detected
0 BF0	Busy violation during read access Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - no error 1b - error detected

55.6.1.60 Lost Link Counter (LOST_LINK_COUNTER_PORT0 - LOST_LINK_COUNTER_PORT1)

Offset

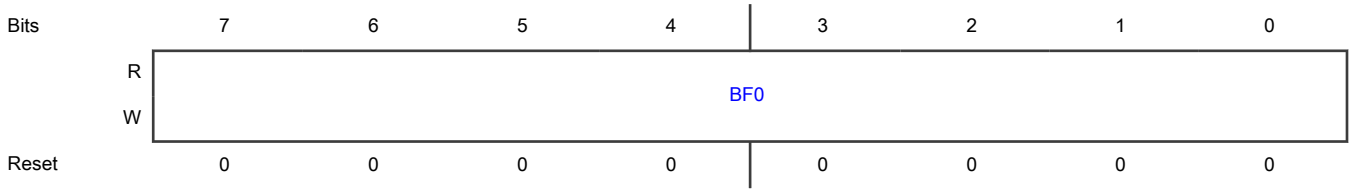
Register	Offset
LOST_LINK_COUNTER_PORT0	310h
LOST_LINK_COUNTER_PORT1	311h

Function

NOTE

Lost Link Counters 0x0310-0x0313 are cleared if one of the implemented Lost Link Counters 0x0310-0x0313 is written (preferably 0x0310). Write value is ignored (write 0).

Diagram



Fields

Field	Function
7-0 BF0	Lost Link counter of Port y (counting is stopped when 0xff is reached). Counts only if port is open and loop is Auto. Bit field access for ECAT: r/ w(clr) Bit field access for PDI: r/-

55.6.1.61 Lost Link Counter (LOST_LINK_COUNTER_PORT0_PDI - LOST_LINK_COUNTER_PORT1_PDI)

Offset

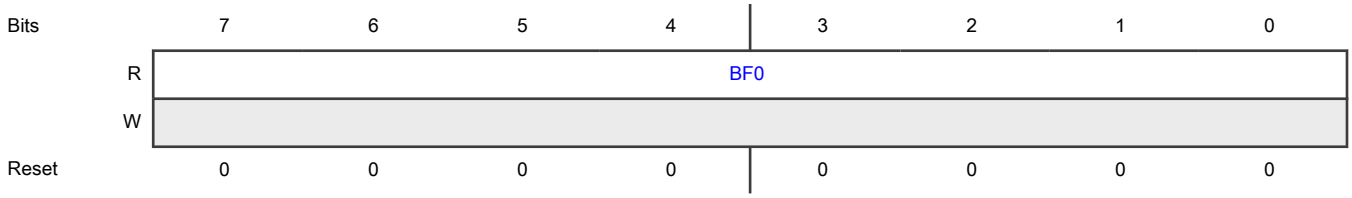
Register	Offset
LOST_LINK_COUNTER_PORT0_PDI	310h
LOST_LINK_COUNTER_PORT1_PDI	311h

Function

NOTE

Lost Link Counters 0x0310-0x0313 are cleared if one of the implemented Lost Link Counters 0x0310-0x0313 is written (preferably 0x0310). Write value is ignored (write 0).

Diagram



Fields

Field	Function
7-0	Lost Link counter of Port y (counting is stopped when 0xff is reached).

Table continues on the next page...

Field	Function
BF0	Counts only if port is open and loop is Auto. Bit field access for ECAT: r/ w(clr) Bit field access for PDI: r/-

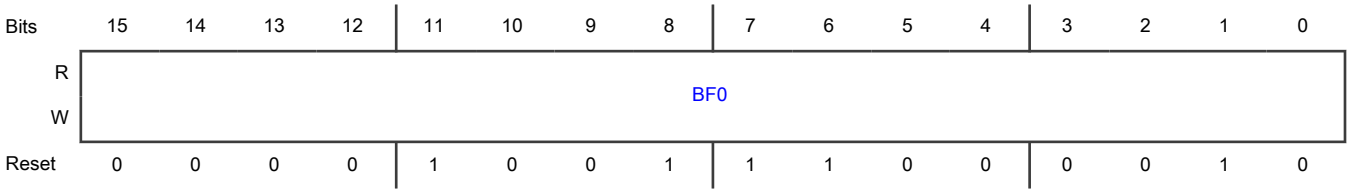
55.6.1.62 Watchdog Divider (WATCHDOG_DIVIDER)

Offset

Register	Offset
WATCHDOG_DIVIDER	400h

Function
Register Watchdog Divider

Diagram



Fields

Field	Function
15-0 BF0	Watchdog divider: Number of 25 MHz tics (minus 2) that represent the basic watchdog increment. (Default value is 100µs = 2498) Bit field access for ECAT: r/w Bit field access for PDI: r/-

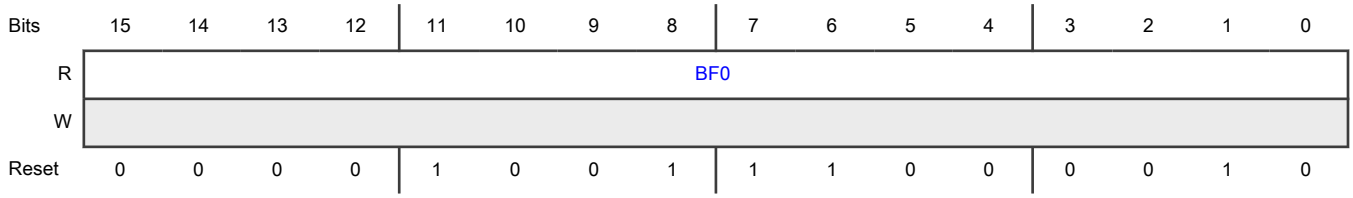
55.6.1.63 Watchdog Divider (WATCHDOG_DIVIDER_PDI)

Offset

Register	Offset
WATCHDOG_DIVIDER_PDI	400h

Function
Register Watchdog Divider

Diagram



Fields

Field	Function
15-0 BF0	Watchdog divider: Number of 25 MHz tics (minus 2) that represent the basic watchdog increment. (Default value is 100µs = 2498) Bit field access for ECAT: r/w Bit field access for PDI: r/-

55.6.1.64 Register Watchdog Time PDI (WATCHDOG_TIME_PDI)

Offset

Register	Offset
WATCHDOG_TIME_PDI	410h

Function

Watchdog is disabled if Watchdog time is set to 0x0000. Watchdog starts counting again with every PDI access.

Diagram

Fields

Field	Function
15-0 BF0	Watchdog Time PDI: number or basic watchdog increments (Default value with Watchdog divider 100µs means 100ms Watchdog) Bit field access for ECAT: r/wBit field access for PDI: r/-

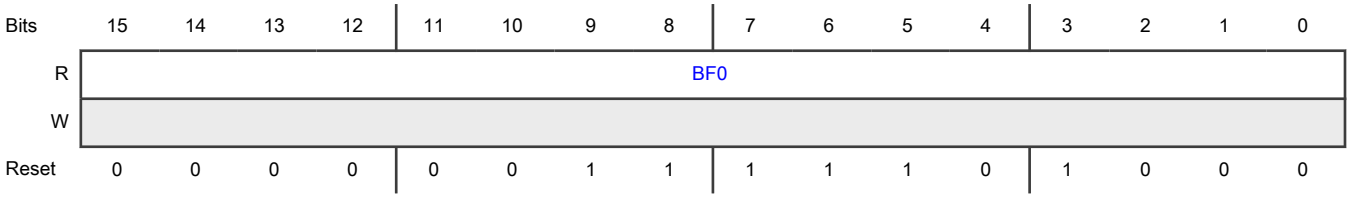
55.6.1.65 Register Watchdog Time PDI (WATCHDOG_TIME_PDI_PDI)

Offset

Register	Offset
WATCHDOG_TIME_PDI_PDI	410h

Function
Watchdog is disabled if Watchdog time is set to 0x0000. Watchdog starts counting again with every PDI access.

Diagram



Fields

Field	Function
15-0 BF0	Watchdog Time PDI: number or basic watchdog increments (Default value with Watchdog divider 100µs means 100ms Watchdog) Bit field access for ECAT: r/wBit field access for PDI: r/-

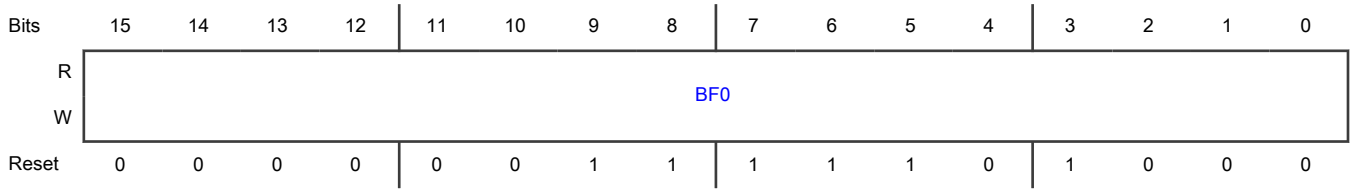
55.6.1.66 Regsister Watchdog Time Process Data (WATCHDOG_TIME_PROCESS_DATA)

Offset

Register	Offset
WATCHDOG_TIME_PROCESS_DATA	420h

Function
There is one Watchdog for all SyncManagers. Watchdog is disabled if Watchdog time is set to 0x0000. Watchdog starts counting again with every write access to SyncManagers with Watchdog Trigger Enable Bit set.

Diagram



Fields

Field	Function
15-0 BF0	Watchdog Time Process Data Number of basic watchdog increments (Default value with Watchdog divider 100µs means 100ms Watchdog)Bit field access for ECAT: r/wBit field access for PDI: r/-

55.6.1.67 Regsister Watchdog Time Process Data (WATCHDOG_TIME_PROCESS_DATA_PDI)

Offset

Register	Offset
WATCHDOG_TIME_PROCESS_DATA_PDI	420h

Function

There is one Watchdog for all SyncManagers. Watchdog is disabled if Watchdog time is set to 0x0000. Watchdog starts counting again with every write access to SyncManagers with Watchdog Trigger Enable Bit set.

Diagram

Fields

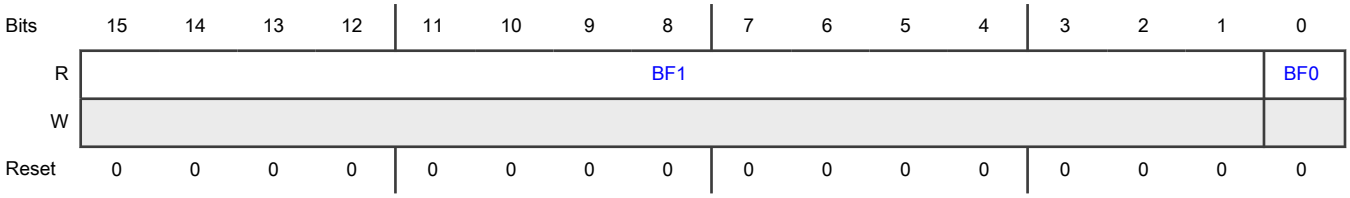
Field	Function
15-0 BF0	Watchdog Time Process Data Number of basic watchdog increments (Default value with Watchdog divider 100µs means 100ms Watchdog)Bit field access for ECAT: r/wBit field access for PDI: r/-

55.6.1.68 Watchdog Status Process Data (WATCHDOG_STATUS_PROCESS_DATA)

Offset

Register	Offset
WATCHDOG_STATUS_PROCESS_DATA	440h

Diagram



Fields

Field	Function
15-1	Reserved
BF1	Bit field access for ECAT: r/- Bit field access for PDI: r/-
0	Watchdog Status of Process Data (triggered by SyncManagers)
BF0	Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Watchdog Process Data expired 1b - Watchdog Process Data is active or disabled

55.6.1.69 Watchdog Counter Process Data (WATCHDOG_COUNTER_PROCESS_DATA)

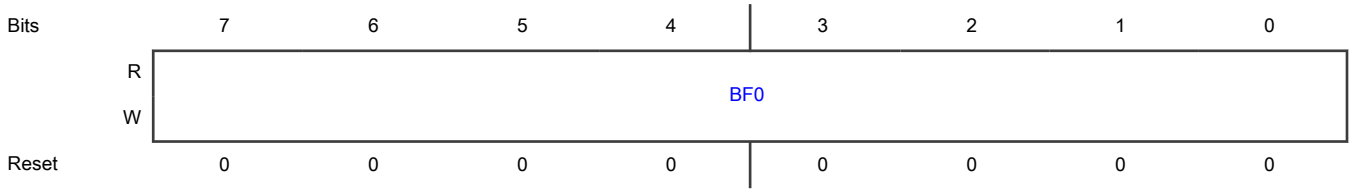
Offset

Register	Offset
WATCHDOG_COUNTER_PROCESS_DATA	442h

Function

NOTE: Watchdog Counters 0x0442-0x0443 are cleared if one of the Watchdog Counters 0x0442-0x0443 is written. Write value is ignored (write 0).

Diagram



Fields

Field	Function
7-0 BF0	Watchdog Counter Process Data (counting is stopped when 0xFF is reached). Counts if Process Data Watchdog expires. Bit field access for ECAT: r/ w(clr)Bit field access for PDI: r/-

55.6.1.70 Watchdog Counter Process Data (WATCHDOG_COUNTER_PROCESS_DATA_PDI)

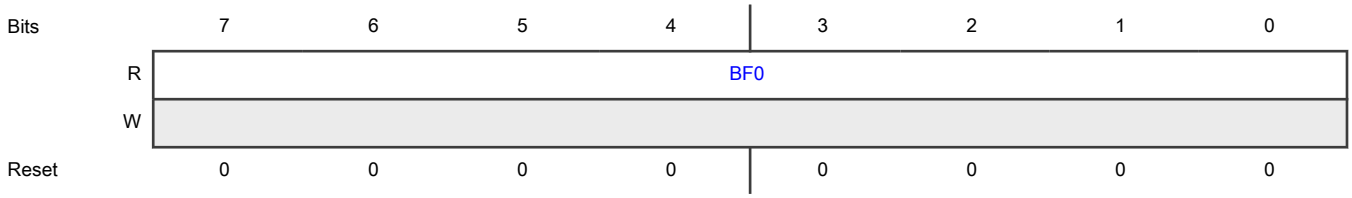
Offset

Register	Offset
WATCHDOG_COUNTER_PROCESS_DATA_PDI	442h

Function

NOTE: Watchdog Counters 0x0442-0x0443 are cleared if one of the Watchdog Counters 0x0442-0x0443 is written. Write value is ignored (write 0).

Diagram



Fields

Field	Function
7-0 BF0	Watchdog Counter Process Data (counting is stopped when 0xFF is reached). Counts if Process Data Watchdog expires. Bit field access for ECAT: r/ w(clr)Bit field access for PDI: r/-

55.6.1.71 Watchdog Counter PDI (WATCHDOG_COUNTER_PDI)

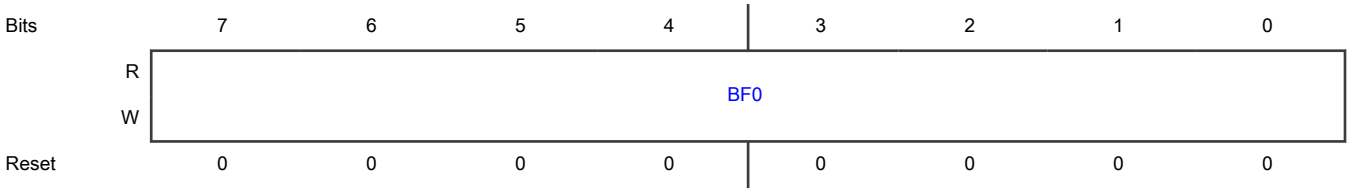
Offset

Register	Offset
WATCHDOG_COUNTER_PDI	443h

Function

NOTE: Watchdog Counters 0x0442-0x0443 are cleared if one of the Watchdog Counters 0x0442-0x0443 is written. Write value is ignored (write 0).

Diagram



Fields

Field	Function
7-0	Watchdog PDI counter (counting is stopped when 0xFF is reached).
BF0	Counts if PDI Watchdog expires.Bit field access for ECAT: r/ w(clr)Bit field access for PDI: r/-

55.6.1.72 Watchdog Counter PDI (WATCHDOG_COUNTER_PDI_PDI)

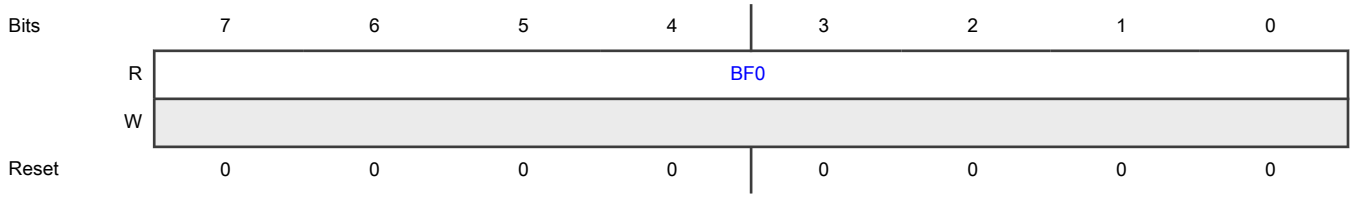
Offset

Register	Offset
WATCHDOG_COUNTER_PDI_PDI	443h

Function

NOTE: Watchdog Counters 0x0442-0x0443 are cleared if one of the Watchdog Counters 0x0442-0x0443 is written. Write value is ignored (write 0).

Diagram



Fields

Field	Function
7-0 BF0	Watchdog PDI counter (counting is stopped when 0xFF is reached). Counts if PDI Watchdog expires.Bit field access for ECAT: r/ w(clr)Bit field access for PDI: r/-

55.6.1.73 EEPROM Configuration (EEPROM_CONFIGURATION)

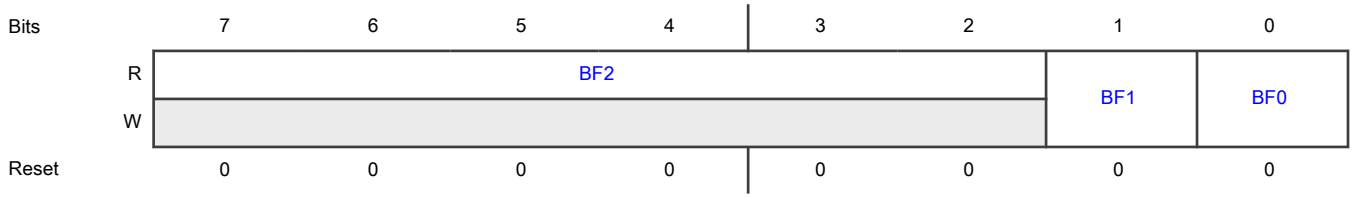
Offset

Register	Offset
EEPROM_CONFIGURATION	500h

Function

EtherCAT controls the SII EEPROM interface if EEPROM configuration register 0x0500[0]=0 and EEPROM PDI Access register 0x0501[0]=0,otherwise PDI controls the EEPROM interface. In EEPROM emulation mode, the PDI executes pending EEPROM commands. The PDI has access to some registers while the EEPROM Interface is busy.

Diagram



Fields

Field	Function
7-2 BF2	Reserved, write 0 Bit field access for ECAT: r/-Bit field access for PDI: r/-
1 BF1	Force ECAT access Bit field access for ECAT: r/wBit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Do not change Bit 0x0501[0] 1b - Reset Bit 0x0501[0] to 0
0 BF0	EEPROM control is offered to PDI Bit field access for ECAT: r/wBit field access for PDI: r/- 0b - no 1b - yes (PDI has EEPROM control)

55.6.1.74 EEPROM Configuration (EEPROM_CONFIGURATION_PDI)

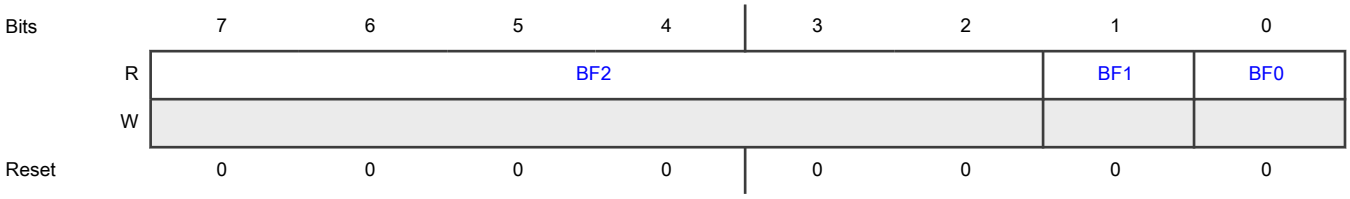
Offset

Register	Offset
EEPROM_CONFIGURATION_PDI	500h

Function

EtherCAT controls the SII EEPROM interface if EEPROM configuration register 0x0500[0]=0 and EEPROM PDI Access register 0x0501[0]=0,otherwise PDI controls the EEPROM interface.In EEPROM emulation mode, the PDI executes pending EEPROM commands. The PDI has access to some registers while the EEPROM Interface is busy.

Diagram



Fields

Field	Function
7-2 BF2	Reserved, write 0 Bit field access for ECAT: r/-Bit field access for PDI: r/-
1 BF1	Force ECAT access Bit field access for ECAT: r/wBit field access for PDI: r/- 0b - Do not change Bit 0x0501[0] 1b - Reset Bit 0x0501[0] to 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 BF0	EEPROM control is offered to PDI Bit field access for ECAT: r/wBit field access for PDI: r/- 0b - no 1b - yes (PDI has EEPROM control)

55.6.1.75 EEPROM PDI Access State (REGISTER_EEPROM_PDI_ACCESS_STATE)

Offset

Register	Offset
REGISTER_EEPROM_PDI_ACCESS_STATE	501h

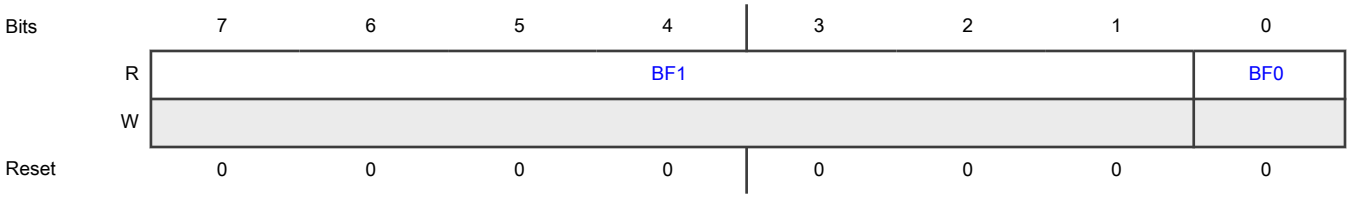
Function

EtherCAT controls the SII EEPROM interface if EEPROM configuration register 0x0500[0]=0 and EEPROM PDI Access register 0x0501[0]=0,otherwise PDI controls the EEPROM interface. In EEPROM emulation mode, the PDI executes pending EEPROM commands. The PDI has access to some registers while the EEPROM Interface is busy.

NOTE

r/(w): write access is only possible if (0x0500[0]=1 or 0x0501[0]=1) and 0x0500[1]=0.

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/-Bit field access for PDI: r/-
0 BF0	Access to EEPROM: Bit field access for ECAT: r/-Bit field access for PDI: r/(w) 0b - PDI releases EEPROM access 1b - PDI takes EEPROM access (PDI has EEPROM control)

55.6.1.76 EEPROM PDI Access State (REGISTER_EEPROM_PDI_ACCESS_STATE_PDI)

Offset

Register	Offset
REGISTER_EEPROM_PDI_ACCESS_STATE_PDI	501h

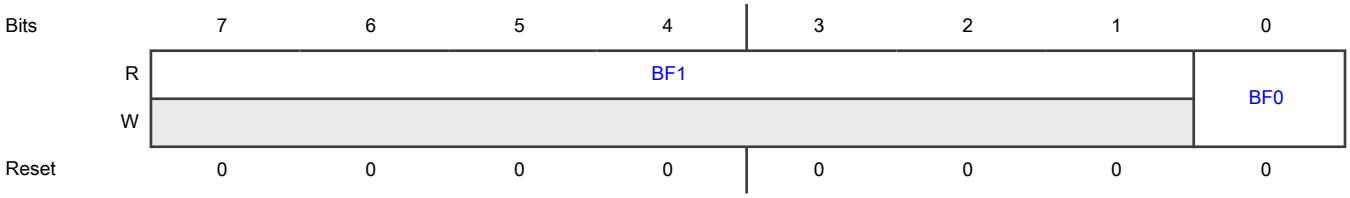
Function

EtherCAT controls the SII EEPROM interface if EEPROM configuration register 0x0500[0]=0 and EEPROM PDI Access register 0x0501[0]=0,otherwise PDI controls the EEPROM interface. In EEPROM emulation mode, the PDI executes pending EEPROM commands. The PDI has access to some registers while the EEPROM Interface is busy.

NOTE

r/(w): write access is only possible if (0x0500[0]=1 or 0x0501[0]=1) and 0x0500[1]=0.

Diagram



Fields

Field	Function
7-1	Reserved, write 0
BF1	Bit field access for ECAT: r/-Bit field access for PDI: r/-
0	Access to EEPROM:
BF0	Bit field access for ECAT: r/-Bit field access for PDI: r/(w) 0b - PDI releases EEPROM access 1b - PDI takes EEPROM access (PDI has EEPROM control)

55.6.1.77 Register EEPROM Control/Status (EEPROM_CONTROL_STATUS)

Offset

Register	Offset
EEPROM_CONTROL_STATUS	502h

Function

EtherCAT controls the SII EEPROM interface if EEPROM configuration register 0x0500[0]=0 and EEPROM PDI Access register 0x0501[0]=0, otherwise PDI controls the EEPROM interface. In EEPROM emulation mode, the PDI executes pending EEPROM commands. The PDI has access to some registers while the EEPROM Interface is busy.

NOTE

r/(w): write access depends upon the assignment of the EEPROM interface (ECAT/PDI). Write access is blocked if EEPROM interface is busy (0x0502[15]=1).

NOTE

r/[w]: EEPROM emulation only: write access is possible if EEPROM interface is busy (0x0502[15]=1). PDI acknowledges pending commands by writing a 1 into the corresponding command register bits (0x0502[10:8]). General/temporary errors can be indicated by writing a 1 into the error bit 0x0502[13], CRC errors for Reload command can be indicated by writing a 1 into the error bit 0x0502[11]. Acknowledging clears AL Event Request 0x0220[5].

- 1. Write Enable bit 0 is self-clearing at the SOF of the next frame, Command bits [10:8] are self-clearing after the command is executed (EEPROM Busy ends). Writing “000” to the command register will also clear the error bits [14:13]. Command bits [10:8] are ignored if Error Acknowledge/Command is pending (bit 13).
- 2. Error bits are cleared by writing “000” (or any valid command) to Command Register Bits [10:8].

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF15	BF14	BF13	BF12	BF11	BF8			BF7	BF6	BF5	BF1				BF0
W																
Reset	1	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0

Fields

Field	Function
15 BF15	Busy: **NOTE** r/(w): write access depends upon the assignment of the EEPROM interface (ECAT/PDI). Write access is blocked if EEPROM interface is busy (0x0502[15]=1).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE r/[w]: EEPROM emulation only: write access is possible if EEPROM interface is busy (0x0502[15]=1). PDI acknowledges pending commands by writing a 1 into the corresponding command register bits (0x0502[10:8]). General/temporary errors can be indicated by writing a 1 into the error bit 0x0502[13], CRC errors for Reload command can be indicated by writing a 1 into the error bit 0x0502[11]. Acknowledging clears AL Event Request 0x0220[5]. • 1. Write Enable bit 0 is self-clearing at the SOF of the next frame, Command bits [10:8] are self-clearing after the command is executed(EEPROM Busy ends). Writing "000" to the command register will also clear the error bits[14:13]. Command bits [10:8] are ignored if Error Acknowledge/Command is pending (bit 13). • 2. Error bits are cleared by writing "000" (or any valid command) to Command Register Bits [10:8]. Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - EEPROM Interface is idle 1b - EEPROM Interface is busy
14 BF14	Error Write Enable3: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No error 1b - Write Command without Write enable
13 BF13	Error Acknowledge/Command3: EEPROM emulation only: PDI writes 1 if a temporary failure has occurred. Bit field access for ECAT: r/- Bit field access for PDI: r/- r/[w] 0b - No error 1b - Missing EEPROM acknowledge or invalid command
12 BF12	EEPROM loading status: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - EEPROM loaded, device information ok 1b - EEPROM not loaded, device information not available (EEPROM loading in progress or finished with a failure)
11	Checksum Error in ESC Configuration Area:

Table continues on the next page...

Table continued from the previous page...

Field	Function
BF11	EEPROM emulation only: PDI writes 1 if a CRC failure has occurred for a reload command. Bit field access for ECAT: r/- Bit field access for PDI: r/- r/[w] 0b - Checksum ok 1b - Checksum error
10-8 BF8	Command register Write: Initiate command. Read: Currently executed command Commands: Others: Reserved/invalid commands (do not issue) EEPROM emulation only: after execution, PDI writes command value to indicate operation is ready. Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) r/[w] 000b - No command/EEPROM idle (clear error bits) 001b - Read 010b - Write 100b - Reload
7 BF7	Selected EEPROM Algorithm: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset for bit is dependent on PROM_SIZE and features 0b - 1 address byte (1Kbit to 16Kbit EEPROMs) 1b - 2 address bytes (32Kbit to 4 Mbit EEPROMs)
6 BF6	Supported number of EEPROM read bytes: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - 4 Bytes 1b - 8 Bytes
5 BF5	EEPROM emulation: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Normal operation (I2C interface used) 1b - PDI emulates EEPROM (I2C not used)
4-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	ECAT write enable2 This bit is always 1 if PDI has EEPROM control. Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - Write requests are disabled 1b - Write requests are enabled

55.6.1.78 Register EEPROM Control/Status (EEPROM_CONTROL_STATUS_PDI)

Offset

Register	Offset
EEPROM_CONTROL_STATUS_PDI	502h

Function

EtherCAT controls the SII EEPROM interface if EEPROM configuration register 0x0500[0]=0 and EEPROM PDI Access register 0x0501[0]=0, otherwise PDI controls the EEPROM interface. In EEPROM emulation mode, the PDI executes pending EEPROM commands. The PDI has access to some registers while the EEPROM Interface is busy.

NOTE

r/(w): write access depends upon the assignment of the EEPROM interface (ECAT/PDI). Write access is blocked if EEPROM interface is busy (0x0502[15]=1). r/[w]: EEPROM emulation only: write access is possible if EEPROM interface is busy (0x0502[15]=1). PDI acknowledges pending commands by writing a 1 into the corresponding command register bits (0x0502[10:8]). General/temporary errors can be indicated by writing a 1 into the error bit 0x0502[13], CRC errors for Reload command can be indicated by writing a 1 into the error bit 0x0502[11]. Acknowledging clears AL Event Request 0x0220[5].

- 1. ESC20: configurable with pin EEPROM SIZE, but not readable in this register.
- 2. Write Enable bit 0 is self-clearing at the SOF of the next frame, Command bits [10:8] are self-clearing after the command is executed (EEPROM Busy ends). Writing “000” to the command register will also clear the error bits [14:13]. Command bits [10:8] are ignored if Error Acknowledge/Command is pending (bit 13).
- 3. Error bits are cleared by writing “000” (or any valid command) to Command Register Bits [10:8].

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF15	BF14	BF13	BF12	BF11	BF8			BF7	BF6	BF5	BF1				BF0
W																
Reset	1	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0

Fields

Field	Function
15 BF15	Busy: r/(w): write access depends upon the assignment of the EEPROM interface (ECAT/PDI). Write access is blocked if EEPROM interface is busy (0x0502[15]=1). **NOTE** r/[w]: EEPROM emulation only: write access is possible if EEPROM interface is busy (0x0502[15]=1). PDI acknowledges pending commands by writing a 1 into the corresponding command register bits (0x0502[10:8]). General/temporary errors can be indicated by writing a 1 into the error bit 0x0502[13], CRC errors for Reload command can be indicated by writing a 1 into the error bit 0x0502[11]. Acknowledging clears AL Event Request 0x0220[5]. • 1. ESC20: configurable with pin EEPROM SIZE, but not readable in this register. • 2 Write Enable bit 0 is self-clearing at the SOF of the next frame, Command bits [10:8] are self-clearing after the command is executed(EEPROM Busy ends). Writing "000" to the command register will also clear the error bits[14:13]. Command bits [10:8] are ignored if Error Acknowledge/Command is pending (bit 13). • 3 lll Error bits are cleared by writing "000" (or any valid command) to Command Register Bits [10:8]. Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - EEPROM Interface is idle 1b - EEPROM Interface is busy
14 BF14	Error Write Enable3: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No error 1b - Write Command without Write enable
13 BF13	Error Acknowledge/Command3: EEPROM emulation only: PDI writes 1 if a temporary failure has occurred. Bit field access for ECAT: r/- Bit field access for PDI: r/- r/[w]

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No error 1b - Missing EEPROM acknowledge or invalid command
12 BF12	EEPROM loading status: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - EEPROM loaded, device information ok 1b - EEPROM not loaded, device information not available (EEPROM loading in progress or finished with a failure)
11 BF11	Checksum Error in ESC Configuration Area: EEPROM emulation only: PDI writes 1 if a CRC failure has occurred for a reload command. Bit field access for ECAT: r/- Bit field access for PDI: r/- r/[w] 0b - Checksum ok 1b - Checksum error
10-8 BF8	Command register Write: Initiate command. Read: Currently executed command Commands: Others: Reserved/invalid commands (do not issue) EEPROM emulation only: after execution, PDI writes command value to indicate operation is ready. Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) r/[w] 000b - No command/EEPROM idle (clear error bits) 001b - Read 010b - Write 100b - Reload
7 BF7	Selected EEPROM Algorithm: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset for bit is dependent on PROM_SIZE and features 0b - 1 address byte (1Kbit to 16Kbit EEPROMs) 1b - 2 address bytes (32Kbit to 4 Mbit EEPROMs)

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 BF6	Supported number of EEPROM read bytes: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - 4 Bytes 1b - 8 Bytes
5 BF5	EEPROM emulation: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration. 0b - Normal operation (I2C interface used) 1b - PDI emulates EEPROM (I2C not used)
4-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	ECAT write enable2 This bit is always 1 if PDI has EEPROM control. Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - Write requests are disabled 1b - Write requests are enabled

55.6.1.79 EEPROM Address (EEPROM_ADDRESS)

Offset

Register	Offset
EEPROM_ADDRESS	504h

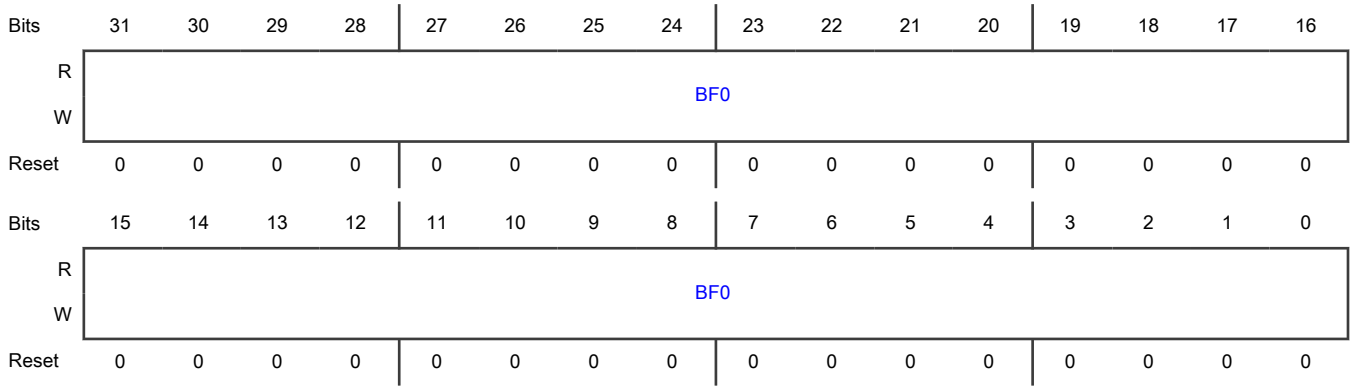
Function

EtherCAT controls the SII EEPROM interface if EEPROM configuration register 0x0500[0]=0 and EEPROM PDI Access register 0x0501[0]=0,otherwise PDI controls the EEPROM interface. In EEPROM emulation mode, the PDI executes pending EEPROM commands. The PDI has access to some registers while the EEPROM Interface is busy.

NOTE

r/(w): write access depends upon the assignment of the EEPROM interface (ECAT/PDI). Write access is blocked if EEPROM interface is busy (0x0502[15]=1).

Diagram



Fields

Field	Function
31-0 BF0	EEPROM Address 0: First word (=16 bit) 1: Second word ... Actually used EEPROM Address bits: [9:0]: EEPROM size up to 16Kbit [17:0]: EEPROM size 32 Kbit - 4Mbit [31:0]: EEPROM Emulation **NOTE** r/(w): write access depends upon the assignment of the EEPROM interface (ECAT/PDI). Write access is blocked if EEPROM interface is busy (0x0502[15]=1). Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w)

55.6.1.80 EEPROM Data (EEPROM_DATA)

Offset

Register	Offset
EEPROM_DATA	508h

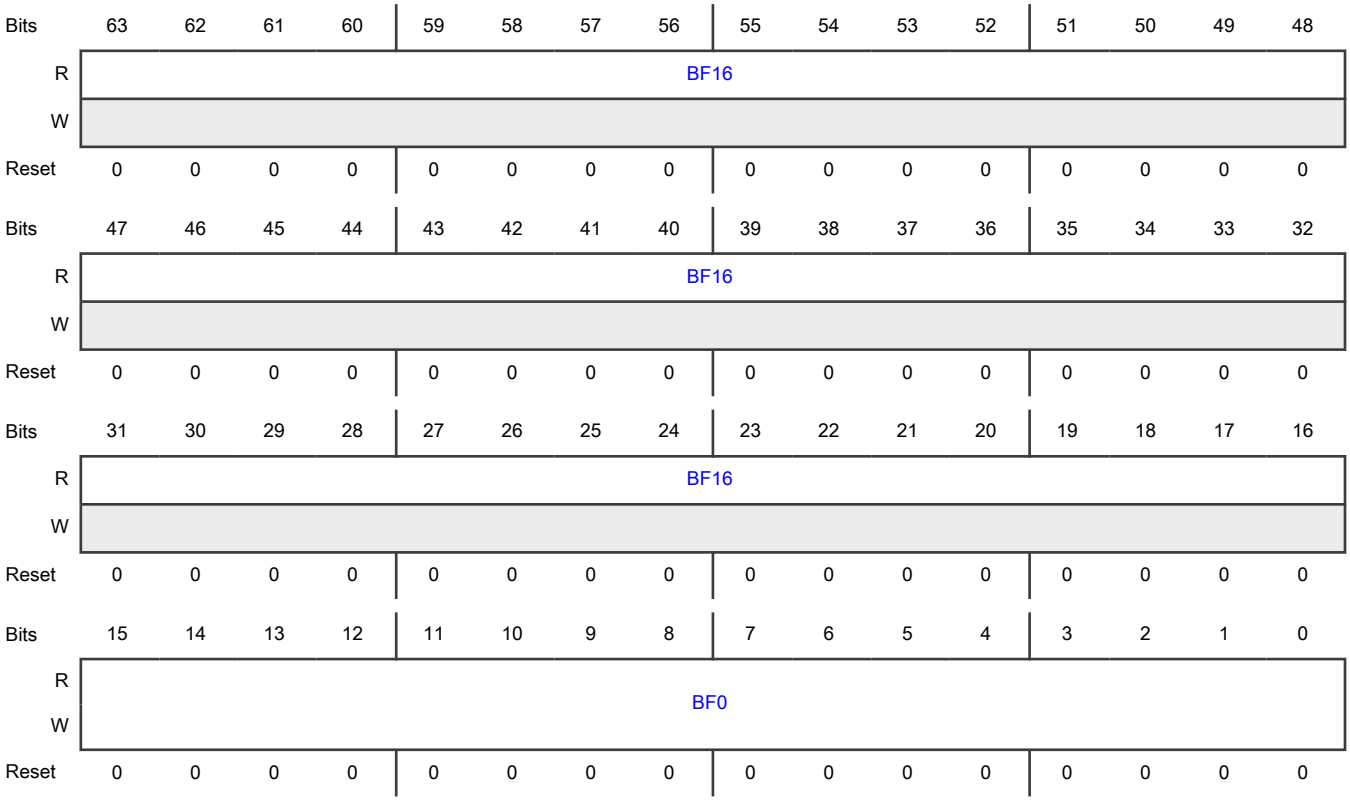
Function

EtherCAT controls the SII EEPROM interface if EEPROM configuration register 0x0500[0]=0 and EEPROM PDI Access register 0x0501[0]=0,otherwise PDI controls the EEPROM interface.In EEPROM emulation mode, the PDI executes pending EEPROM commands. The PDI has access to some registers while the EEPROM Interface is busy.

NOTE

r/(w): write access depends upon the assignment of the EEPROM interface (ECAT/PDI). Write access is blocked if EEPROM interface is busy (0x0502[15]=1). r/[w]: write access for EEPROM emulation if read or reload command is pending. See the following information for further details:

Diagram



Fields

Field	Function
63-16 BF16	EEPROM Read data (data read from EEPROM, higher bytes) Bit field access for ECAT: r/- Bit field access for PDI: r/- r/[w]
15-0 BF0	EEPROM Write data (data to be written to EEPROM) or EEPROM Read data (data read from EEPROM,. lower bytes) Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) r/[w]

55.6.1.81 MII Management Control/Status (MII_MANAGEMENT_CONTROL_OR_STATUS)

Offset

Register	Offset
MII_MANAGEMENT_CONTROL_OR_STATUS	510h

Function

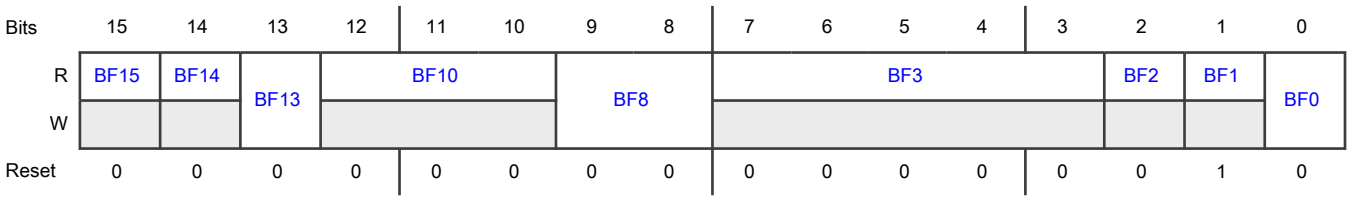
ECAT controls the MII management interface if MII Management PDI Access register 0x0517[0]=0 or if 0x0517 is not available,otherwise PDI controls the MII management interface.

NOTE

r/ (w): write access depends on assignment of MI (ECAT/PDI). Write access is blocked if Management interface is busy (0x0510[15]=1).

* Write enable bit 0 is self-clearing at the SOF of the next frame, Command bits [9:8] are self-clearing after the command is executed (Busy ends). Writing “00” to the command register will also clear the error bits [14:13]. The Command bits are cleared after the command is executed.

Diagram



Fields

Field	Function
15 BF15	Busy: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - MII Management Interface is idle 1b - MII Management Interface is busy
14 BF14	Command error: Cleared by executing a valid command or by writing “00” to Command register bits [9:8]. Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Last Command was successful 1b - Invalid command or write command without Write Enable
13	Read error:

Table continues on the next page...

Table continued from the previous page...

Field	Function
BF13	Cleared by writing to register 0x0511. Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - No read error 1b - Read error occurred (PHY or register not available)
12-10 BF10	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
9-8 BF8	Command register*: Write: Initiate command. Read: Currently executed command Commands: Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 00b - No command/MI idle (clear error bits) 01b - Read 10b - Write 11b - Reserved/invalid command (do not issue)
7-3 BF3	PHY address of port 0 IP Core since V3.0.0/3.00c: PHY address of port 0-3 depending on 0x0512[7] Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration.
2 BF2	MI link detection and configuration: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration. 0b - Disabled for all ports 1b - Enabled for at least one MII port, refer to PHY Port Status (0x0518 ff.) for details
1 BF1	Management Interface can be controlled by PDI (registers 0x0516-0x0517): Bit field access for ECAT: r/- Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Only ECAT control 1b - PDI control possible
0 BF0	Write enable*: This bit is always 1 if PDI has MI control. ET1100-0000/-0001 exception: Bit is not always 1 if PDI has MI control, and bit is writable by PDI. Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - Write disabled 1b - Write enabled

55.6.1.82 MII Management Control/Status (MII_MANAGEMENT_CONTROL_OR_STATUS_PDI)

Offset

Register	Offset
MII_MANAGEMENT_CONTROL_OR_STATUS_PDI	510h

Function

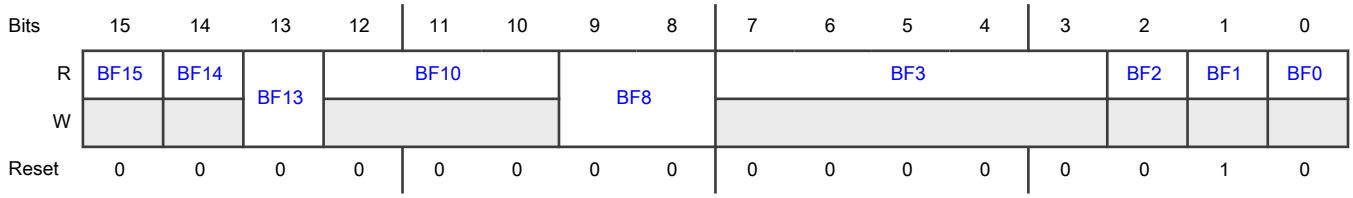
ECAT controls the MII management interface if MII Management PDI Access register 0x0517[0]=0 or if 0x0517 is not available,otherwise PDI controls the MII management interface.

NOTE

r/ (w): write access depends on assignment of MI (ECAT/PDI). Write access is blocked if Management interface is busy (0x0510[15]=1).

* Write enable bit 0 is self-clearing at the SOF of the next frame, Command bits [9:8] are self-clearing after the command is executed (Busy ends). Writing “00” to the command register will also clear the error bits [14:13]. The Command bits are cleared after the command is executed.

Diagram



Fields

Field	Function
15 BF15	Busy: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - MII Management Interface is idle 1b - MII Management Interface is busy
14 BF14	Command error: Cleared by executing a valid command or by writing “00” to Command register bits [9:8]. Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Last Command was successful 1b - Invalid command or write command without Write Enable
13 BF13	Read error: Cleared by writing to register 0x0511. Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - No read error 1b - Read error occurred (PHY or register not available)
12-10 BF10	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
9-8 BF8	Command register*: Write: Initiate command. Read: Currently executed command Commands: Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 00b - No command/MI idle (clear error bits)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Read 10b - Write 11b - Reserved/invalid command (do not issue)
7-3 BF3	PHY address of port 0 IP Core since V3.0.0/3.00c: PHY address of port 0-3 depending on 0x0512[7] Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration.
2 BF2	MI link detection and configuration: Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration. 0b - Disabled for all ports 1b - Enabled for at least one MII port, refer to PHY Port Status (0x0518 ff.) for details
1 BF1	Management Interface can be controlled by PDI (registers 0x0516-0x0517): Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Only ECAT control 1b - PDI control possible
0 BF0	Write enable*: This bit is always 1 if PDI has MI control. ET1100-0000/-0001 exception: Bit is not always 1 if PDI has MI control, and bit is writable by PDI. Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - Write disabled 1b - Write enabled

55.6.1.83 PHY Address (PHY_ADDRESS)

Offset

Register	Offset
PHY_ADDRESS	512h

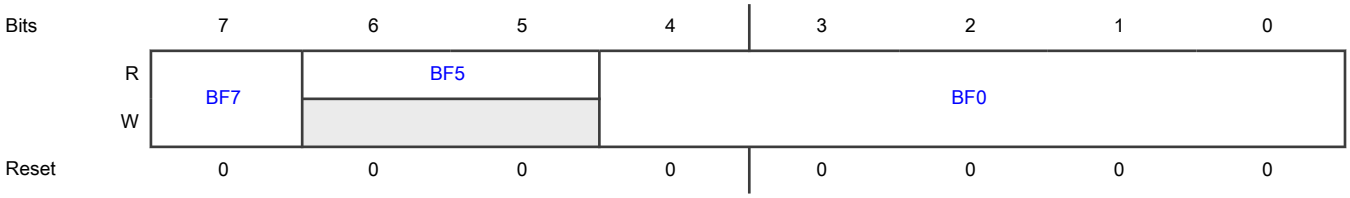
Function

ECAT controls the MII management interface if MII Management PDI Access register 0x0517[0]=0 or if 0x0517 is not available, otherwise PDI controls the MII management interface.

NOTE

r/(w): write access depends on assignment of MI (ECAT/PDI). Write access is blocked if Management interface is busy (0x0510[15]=1).

Diagram



Fields

Field	Function
7 BF7	Show configured PHY address of port 0-3 in register 0x0510[7:3]. This is used if the PHY addresses are not consecutive. Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - Register 0x0510[7:3] shows PHY address of port 0 (this is also the PHY address offset, if the PHY addresses are consecutive) 1b - Register 0x0510[7:3] shows PHY address of port 0x0512[4:0] (valid values 0-3)
6-5 BF5	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
4-0 BF0	PHY Address Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w)

55.6.1.84 PHY Register Address (PHY_REGISTER_ADDRESS)

Offset

Register	Offset
PHY_REGISTER_ADDRESS	513h

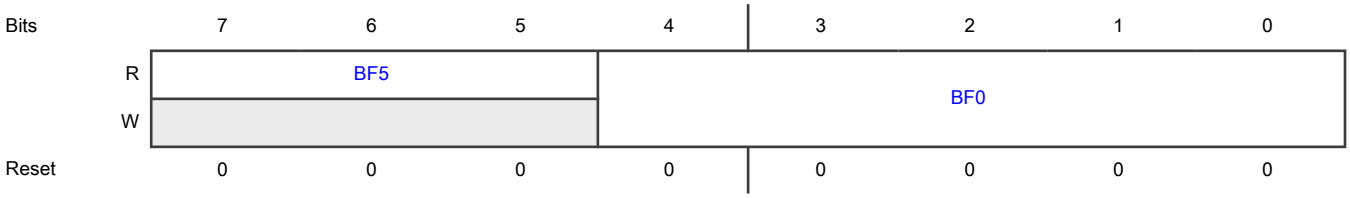
Function

ECAT controls the MII management interface if MII Management PDI Access register 0x0517[0]=0 or if 0x0517 is not available, otherwise PDI controls the MII management interface.

NOTE

r/(w): write access depends on assignment of MI (ECAT/PDI). Write access is blocked if Management interface is busy (0x0510[15]=1).

Diagram



Fields

Field	Function
7-5 BF5	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
4-0 BF0	Address of PHY Register that shall be read/written Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w)

55.6.1.85 PHY Data (PHY_DATA)

Offset

Register	Offset
PHY_DATA	514h

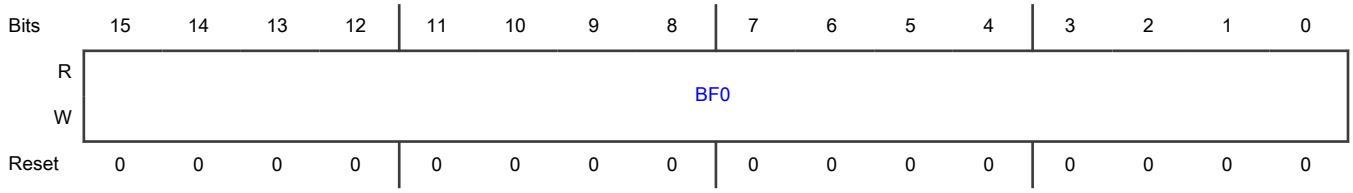
Function

ECAT controls the MII management interface if MII Management PDI Access register 0x0517[0]=0 or if 0x0517 is not available, otherwise PDI controls the MII management interface.

NOTE

r/(w): write access depends on assignment of MI (ECAT/PDI). Write access is blocked if Management interface is busy (0x0510[15]=1).

Diagram



Fields

Field	Function
15-0 BF0	PHY Read/Write Data Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w)

55.6.1.86 MII Management ECAT Access State (MII_MANAGEMENT_ECAT_ACCESS_STATE)

Offset

Register	Offset
MII_MANAGEMENT_ECAT_ACCESS_STATE	516h

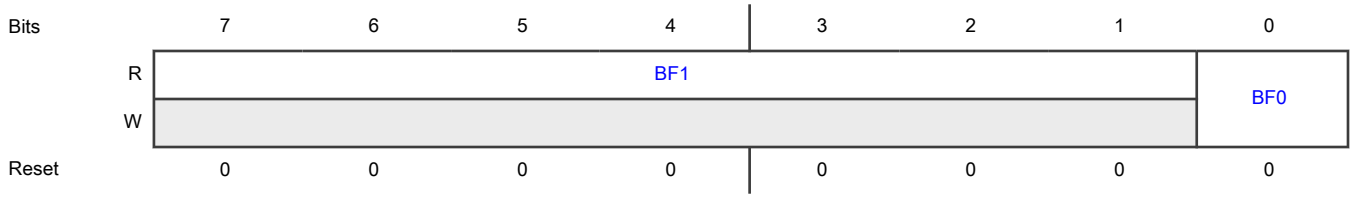
Function

ECAT controls the MII management interface if MII Management PDI Access register 0x0517[0]=0 or if 0x0517 is not available, otherwise PDI controls the MII management interface.

NOTE

r/(w): write access is only possible if 0x0517[0]=0.

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 BF0	Access to MII management: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - ECAT enables PDI takeover of MII management interface 1b - ECAT claims exclusive access to MII management interface

55.6.1.87 MII Management ECAT Access State (MII_MANAGEMENT_ECAT_ACCESS_STATE_PDI)

Offset

Register	Offset
MII_MANAGEMENT_ECAT_ACCESS_STATE_PDI	516h

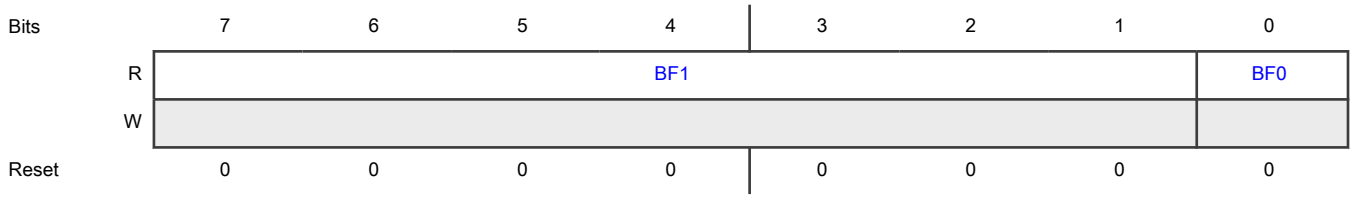
Function

ECAT controls the MII management interface if MII Management PDI Access register 0x0517[0]=0 or if 0x0517 is not available, otherwise PDI controls the MII management interface.

NOTE

r/(w): write access is only possible if 0x0517[0]=0.

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	Access to MII management: Bit field access for ECAT: r/(w) Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - ECAT enables PDI takeover of MII management interface 1b - ECAT claims exclusive access to MII management interface

55.6.1.88 MII Management PDI Access State (MII_MANAGEMENT_PDI_ACCESS_STATE)

Offset

Register	Offset
MII_MANAGEMENT_PDI_ACCESS_STATE	517h

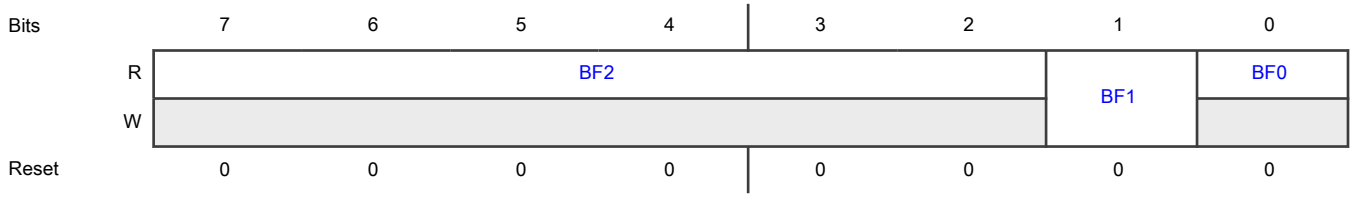
Function

ECAT controls the MII management interface if MII Management PDI Access register 0x0517[0]=0 or if 0x0517 is not available, otherwise PDI controls the MII management interface.

NOTE

r/(w): assigning access to PDI (bit 0 = 1) is only possible if 0x0516[0]=0 and 0x0517[1]=0. The SII EEPROM must be loaded (0x0110[0]=1) as well.

Diagram



Fields

Field	Function
7-2 BF2	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Force PDI Access State: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - Do not change Bit 0x0517[0] 1b - Reset Bit 0x0517[0] to 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 BF0	Access to MII management: Bit field access for ECAT: r/- Bit field access for PDI: r/(w) 0b - ECAT has access to MII management 1b - PDI has access to MII management

55.6.1.89 MII Management PDI Access State (MII_MANAGEMENT_PDI_ACCESS_STATE_PDI)

Offset

Register	Offset
MII_MANAGEMENT_PDI_ACCESS_STATE_PDI	517h

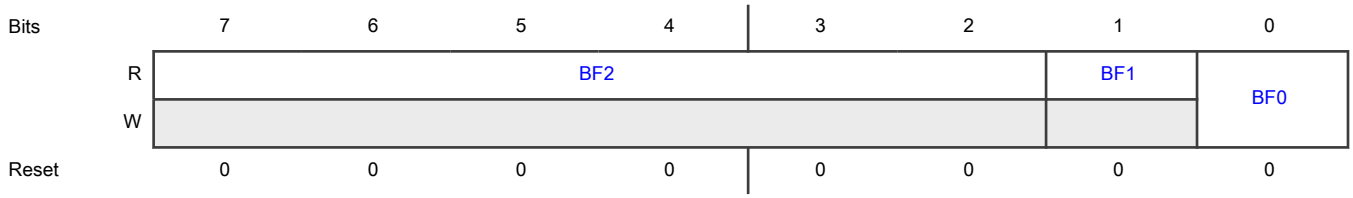
Function

ECAT controls the MII management interface if MII Management PDI Access register 0x0517[0]=0 or if 0x0517 is not available, otherwise PDI controls the MII management interface. .

NOTE

r/(w): assigning access to PDI (bit 0 = 1) is only possible if 0x0516[0]=0 and 0x0517[1]=0. The SII EEPROM must be loaded (0x0110[0]=1) as well.

Diagram



Fields

Field	Function
7-2 BF2	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Force PDI Access State: Bit field access for ECAT: r/(w) Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Do not change Bit 0x0517[0] 1b - Reset Bit 0x0517[0] to 0
0 BF0	Access to MII management: Bit field access for ECAT: r/- Bit field access for PDI: r/(w) 0b - ECAT has access to MII management 1b - PDI has access to MII management

55.6.1.90 PHY Port (PHY_PORT0_STATUS - PHY_PORT1_STATUS)

Offset

Register	Offset
PHY_PORT0_STATUS	518h
PHY_PORT1_STATUS	519h

Function

Register PHY Port y (port number y=0 to 3) Status (0x0518+y) ECAT controls the MII management interface if MII Management PDI Access register 0x0517[0]=0 or if 0x0517 is not available, otherwise PDI controls the MII management interface.

NOTE

r/(w): write access depends on assignment of MI (ECAT/PDI).

Diagram

Bits	7	6	5	4	3	2	1	0
R	BF6		BF5	BF4	BF3	BF2	BF1	BF0
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-6	Reserved
BF6	Bit field access for ECAT: r/- Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 BF5	PHY configuration updated Cleared by writing any value to at least one of the PHY Port y Status registers. Bit field access for ECAT: r/(w/clr) Bit field access for PDI: r/(w/clr) 0b - No update 1b - PHY configuration was updated
4 BF4	Link partner error: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No error detected 1b - Link partner error
3 BF3	Read error: Cleared by writing any value to at least one of the PHY Port y Status registers. Bit field access for ECAT: r/(w/clr) Bit field access for PDI: r/(w/clr) 0b - No read error occurred 1b - A read error has occurred
2 BF2	Link status error: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No error 1b - Link error, link inhibited
1 BF1	Link status (100 Mbit/s, Full Duplex, Auto negotiation): Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No link 1b - Link detected
0 BF0	Physical link status (PHY status register 1.2): Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - No physical link 1b - Physical link detected

55.6.1.91 Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS0 - FMMU_LOGICAL_START_ADDRESS7)

Offset

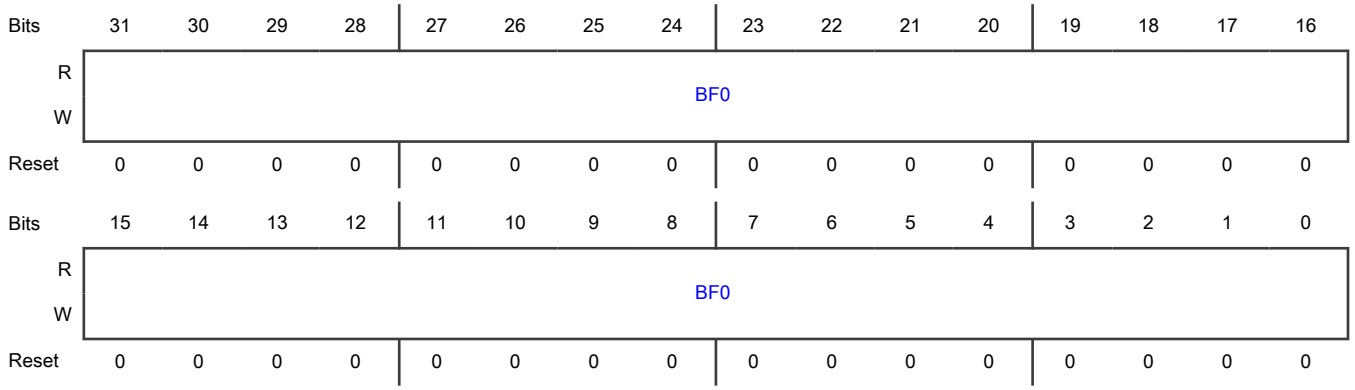
For a = 0 to 7:

Register	Offset
FMMU_LOGICAL_START_ADDRESSa	600h + (a × 10h)

Function

Register Logical Start address FMMU

Diagram



Fields

Field	Function
31-0	Logical start address within the EtherCAT Address Space.
BF0	Bit field access for ECAT: r/w

55.6.1.92 Register Logical Start address FMMU (FMMU_LOGICAL_START_ADDRESS0_PDI - FMMU_LOGICAL_START_ADDRESS7_PDI)

Offset

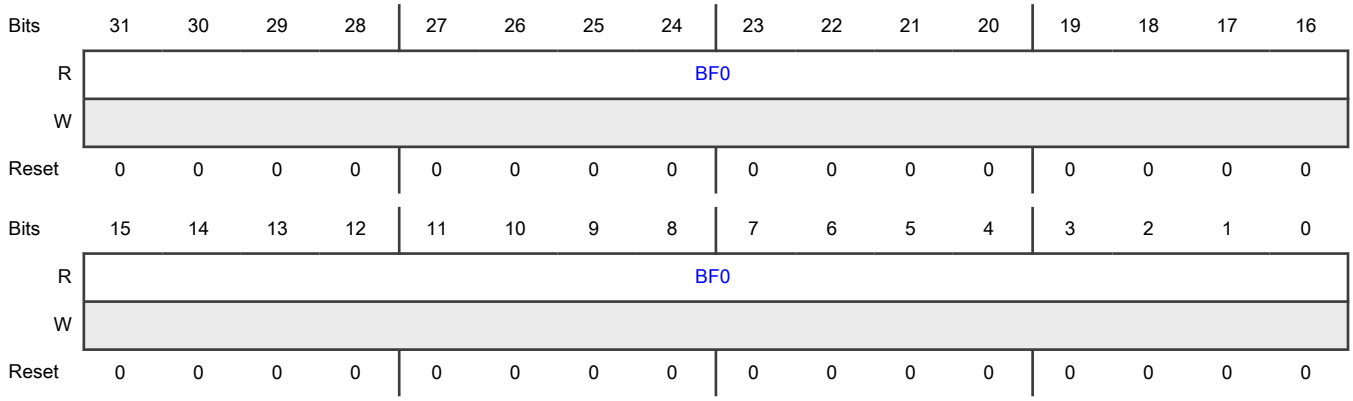
For a = 0 to 7:

Register	Offset
FMMU_LOGICAL_START_ADDRESSa_PDI	600h + (a × 10h)

Function

Register Logical Start address FMMU

Diagram



Fields

Field	Function
31-0 BF0	Logical start address within the EtherCAT Address Space. Bit field access for PDI: r/-

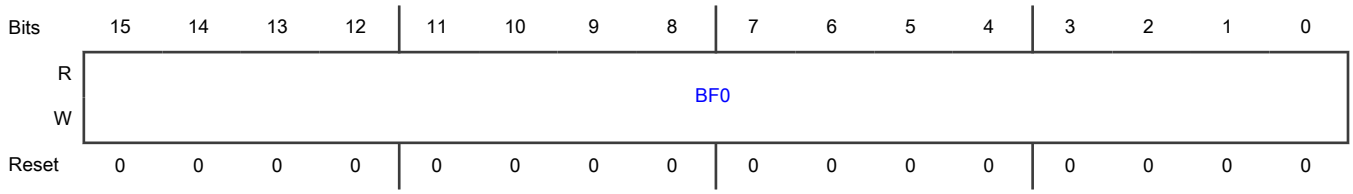
55.6.1.93 Register Length FMMU (FMMU_LENGTH0 - FMMU_LENGTH7)

Offset

For a = 0 to 7:

Register	Offset
FMMU_LENGTHa	604h + (a × 10h)

Diagram



Fields

Field	Function
15-0 BF0	Offset from the first logical FMMU byte to the last FMMU byte + 1 Example: If two bytes are used, then this parameter shall contain 2) Bit field access for ECAT: r/w

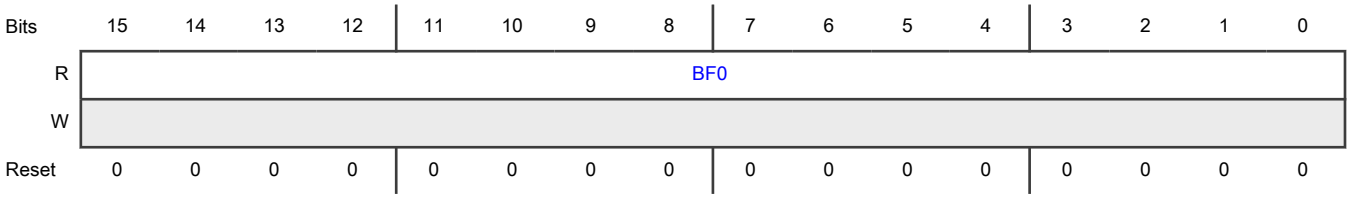
55.6.1.94 Register Length FMMU (FMMU_LENGTH0_PDI - FMMU_LENGTH7_PDI)

Offset

For a = 0 to 7:

Register	Offset
FMMU_LENGTHa_PDI	604h + (a × 10h)

Diagram



Fields

Field	Function
15-0 BF0	Offset from the first logical FMMU byte to the last FMMU byte + 1 Example: If two bytes are used, then this parameter shall contain 2) Bit field access for PDI: r/-

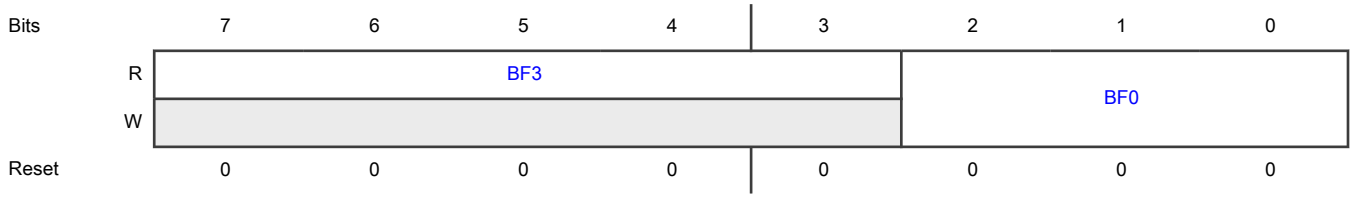
55.6.1.95 Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT0 - FMMU_LOGICAL_START_BIT7)

Offset

For a = 0 to 7:

Register	Offset
FMMU_LOGICAL_START_BITa	606h + (a × 10h)

Diagram



Fields

Field	Function
7-3 BF3	Reserved, write 0 Bit field access for ECAT: r/-
2-0 BF0	Logical starting bit that shall be mapped (bits are counted from least significant bit 0 to most significant bit 7) Bit field access for ECAT: r/w

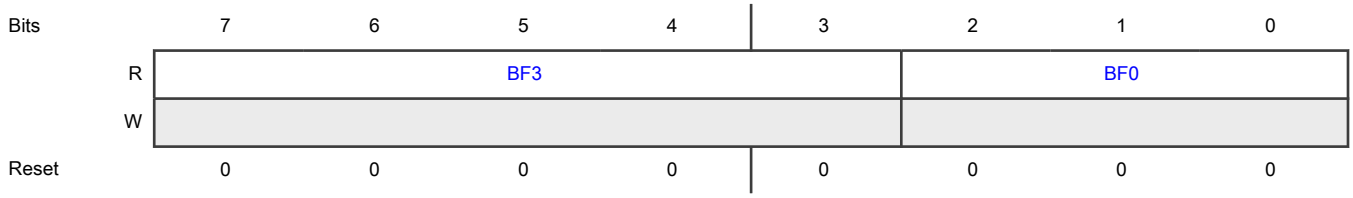
55.6.1.96 Register Start bit FMMU y in logical address space (FMMU_LOGICAL_START_BIT0_PDI - FMMU_LOGICAL_START_BIT7_PDI)

Offset

For a = 0 to 7:

Register	Offset
FMMU_LOGICAL_START_BITa_PDI	606h + (a × 10h)

Diagram



Fields

Field	Function
7-3 BF3	Reserved, write 0 Bit field access for PDI: r/-
2-0 BF0	Logical starting bit that shall be mapped (bits are counted from least significant bit 0 to most significant bit 7) Bit field access for PDI: r/-

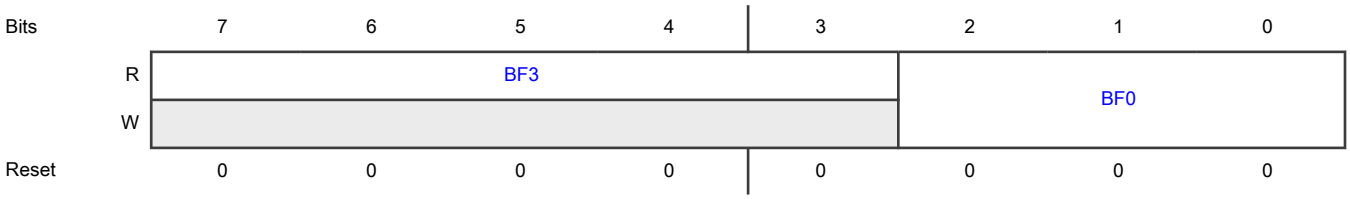
55.6.1.97 Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT0 - FMMU_LOGICAL_STOP_BIT7)

Offset

For a = 0 to 7:

Register	Offset
FMMU_LOGICAL_STOP_BITa	607h + (a × 10h)

Diagram



Fields

Field	Function
7-3 BF3	Reserved, write 0 Bit field access for ECAT: r/-
2-0 BF0	Last logical bit that shall be mapped (bits are counted from least significant bit 0 to most significant bit 7) Bit field access for ECAT: r/w

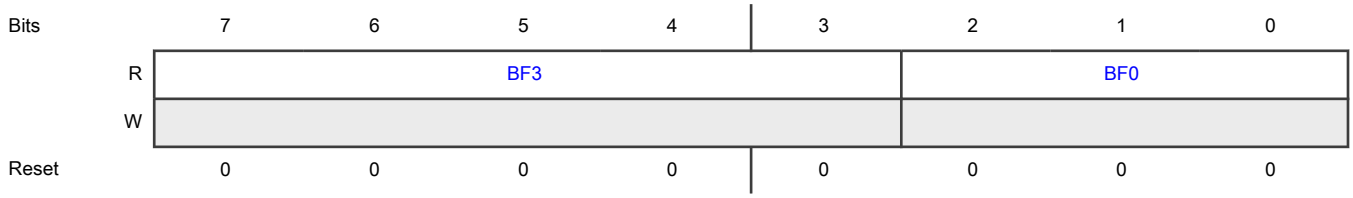
55.6.1.98 Register Stop bit FMMU y in logical address space (FMMU_LOGICAL_STOP_BIT0_PDI - FMMU_LOGICAL_STOP_BIT7_PDI)

Offset

For a = 0 to 7:

Register	Offset
FMMU_LOGICAL_STOP_BITa_PDI	607h + (a × 10h)

Diagram



Fields

Field	Function
7-3 BF3	Reserved, write 0 Bit field access for PDI: r/-
2-0 BF0	Last logical bit that shall be mapped (bits are counted from least significant bit 0 to most significant bit 7) Bit field access for PDI: r/-

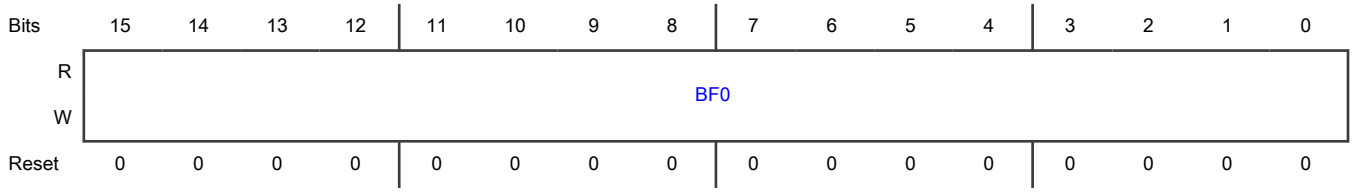
55.6.1.99 Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS0 - FMMU_PHYSICAL_START_ADDRESS7)

Offset

For a = 0 to 7:

Register	Offset
FMMU_PHYSICAL_START_ADDRESSa	608h + (a × 10h)

Diagram



Fields

Field	Function
15-0 BF0	Physical Start Address (mapped to logical Start address) Bit field access for ECAT: r/w

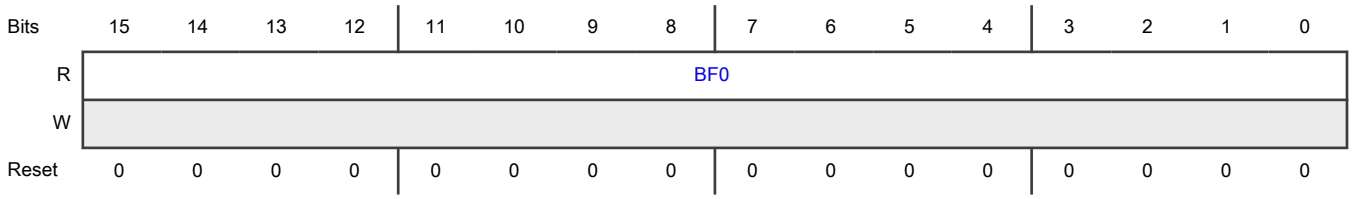
55.6.1.100 Register Physical Start address FMMU (FMMU_PHYSICAL_START_ADDRESS0_PDI - FMMU_PHYSICAL_START_ADDRESS7_PDI)

Offset

For a = 0 to 7:

Register	Offset
FMMU_PHYSICAL_START_ADDRESSa_PDI	608h + (a × 10h)

Diagram



Fields

Field	Function
15-0	Physical Start Address (mapped to logical Start address)
BF0	Bit field access for PDI: r/-

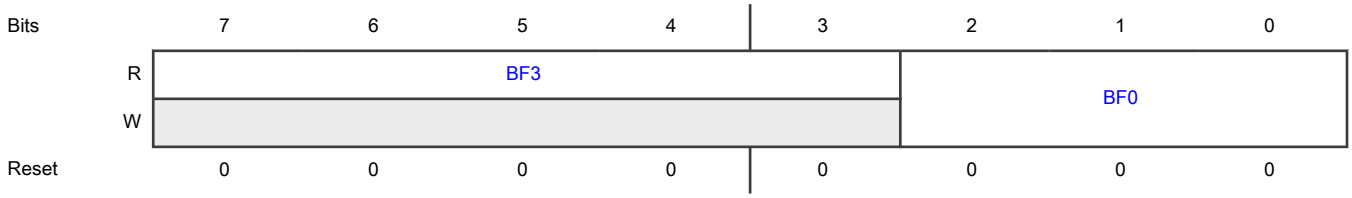
55.6.1.101 Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT0 - FMMU_PHYSICAL_START_BIT7)

Offset

For a = 0 to 7:

Register	Offset
FMMU_PHYSICAL_START_BITa	60Ah + (a × 10h)

Diagram



Fields

Field	Function
7-3 BF3	Reserved, write 0 Bit field access for ECAT: r/-
2-0 BF0	Physical starting bit as target of logical start bit mapping (bits are counted from least significant bit 0 to most significant bit 7) Bit field access for ECAT: r/w

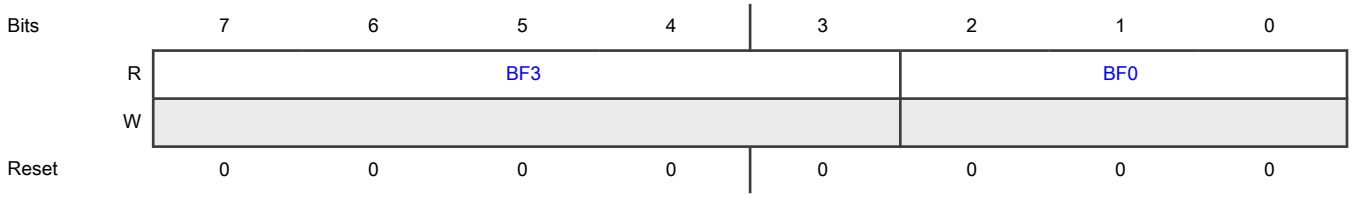
55.6.1.102 Register Physical Start bit FMMU (FMMU_PHYSICAL_START_BIT0_PDI - FMMU_PHYSICAL_START_BIT7_PDI)

Offset

For a = 0 to 7:

Register	Offset
FMMU_PHYSICAL_START_BITa_PDI	60Ah + (a × 10h)

Diagram



Fields

Field	Function
7-3 BF3	Reserved, write 0 Bit field access for PDI: r/-
2-0 BF0	Physical starting bit as target of logical start bit mapping (bits are counted from least significant bit 0 to most significant bit 7) Bit field access for PDI: r/-

55.6.1.103 Register Type FMMU y (FMMU_TYPE0 - FMMU_TYPE7)

Offset

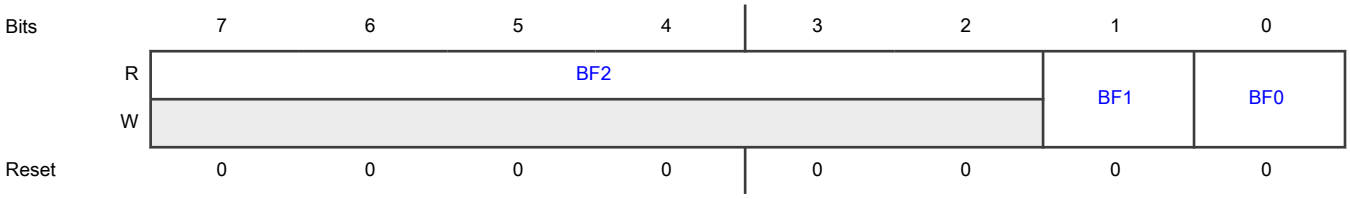
For a = 0 to 7:

Register	Offset
FMMU_TYPEa	60Bh + (a × 10h)

Function

Register Type FMMU y

Diagram



Fields

Field	Function
7-2 BF2	Reserved, write 0 Bit field access for ECAT: r/-
1 BF1	Bit field access for ECAT: r/w 0b - Ignore mapping for write accesses 1b - Use mapping for write accesses
0 BF0	Bit field access for ECAT: r/w 0b - Ignore mapping for read accesses 1b - Use mapping for read accesses

55.6.1.104 Register Type FMMU y (FMMU_TYPE0_PDI - FMMU_TYPE7_PDI)

Offset

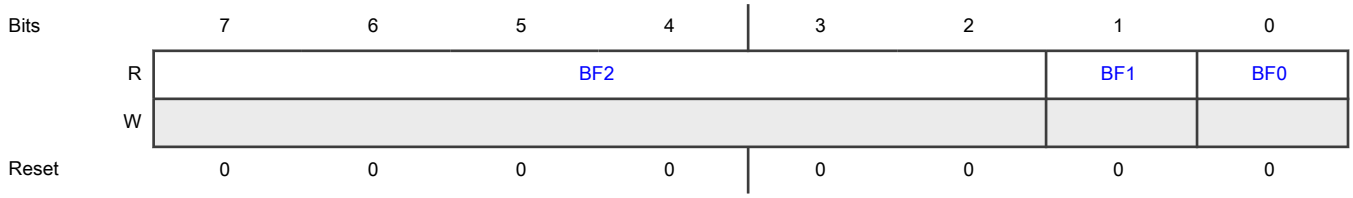
For a = 0 to 7:

Register	Offset
FMMU_TYPEa_PDI	60Bh + (a × 10h)

Function

Register Type FMMU y

Diagram



Fields

Field	Function
7-2 BF2	Reserved, write 0 Bit field access for PDI: r/-
1 BF1	Bit field access for PDI: r/- 0b - Ignore mapping for write accesses 1b - Use mapping for write accesses
0 BF0	Bit field access for PDI: r/- 0b - Ignore mapping for read accesses 1b - Use mapping for read accesses

55.6.1.105 Register Activate FMMU (FMMU_ACTIVATE0 - FMMU_ACTIVATE7)

Offset

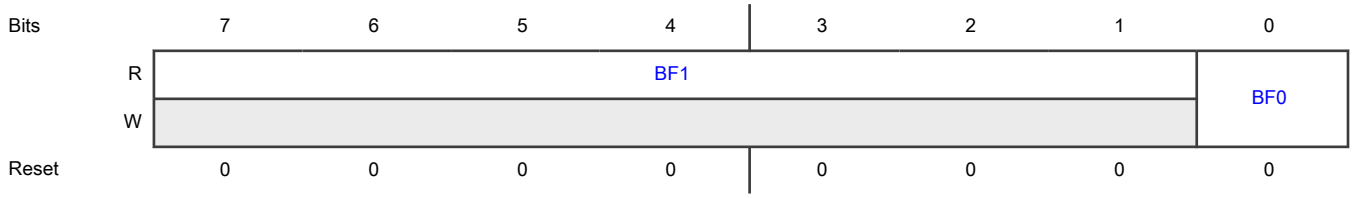
For a = 0 to 7:

Register	Offset
FMMU_ACTIVATEa	60Ch + (a × 10h)

Function

Register Activate FMMU

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for ECAT: r/-
0 BF0	Bit field access for ECAT: r/w 0b - FMMU deactivated 1b - FMMU activated. FMMU checks logically addressed blocks to be mapped according to configured mapping

55.6.1.106 Register Activate FMMU (FMMU_ACTIVATE0_PDI - FMMU_ACTIVATE7_PDI)

Offset

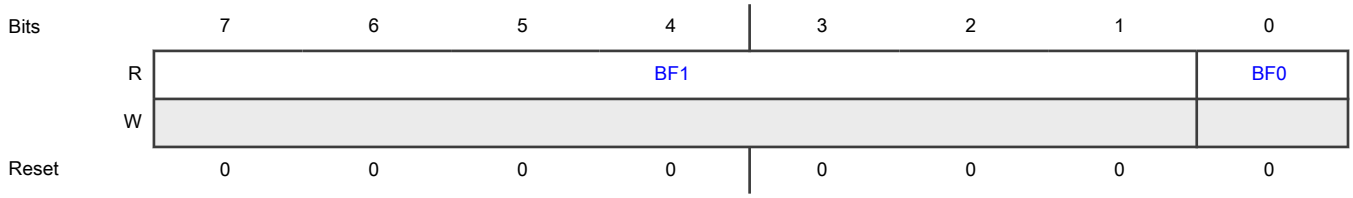
For a = 0 to 7:

Register	Offset
FMMU_ACTIVATEa_PDI	60Ch + (a × 10h)

Function

Register Activate FMMU

Diagram



Fields

Field	Function
7-1 BF1	Reserved, write 0 Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 BF0	Bit field access for PDI: r/- 0b - FMMU deactivated 1b - FMMU activated. FMMU checks logically addressed blocks to be mapped according to configured mapping

55.6.1.107 Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS0 - SYNCMANAGER_PHYSICAL_START_ADDRESS15)

Offset

For a = 0 to 15:

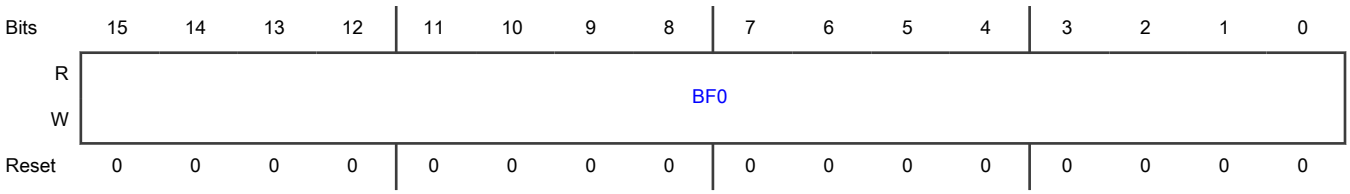
Register	Offset
SYNCMANAGER_PHYSICAL_START_ADDRESSa	800h + (a × 8h)

Function

NOTE

r/(w): Register can only be written if SyncManager is disabled (+0x6[0] = 0).

Diagram



Fields

Field	Function
15-0 BF0	First byte that will be handled by SyncManager Bit field access for ECAT: r/(w) Bit field access for PDI: r/-

55.6.1.108 Register physical Start Address SyncManager (SYNCMANAGER_PHYSICAL_START_ADDRESS0_PDI - SYNCMANAGER_PHYSICAL_START_ADDRESS15_PDI)

Offset

For a = 0 to 15:

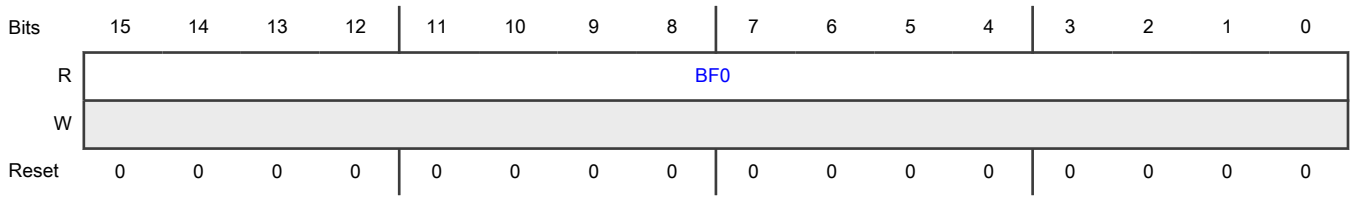
Register	Offset
SYNCMANAGER_PHYSICAL_START_ADDRESSa_PDI	800h + (a × 8h)

Function

NOTE

r/(w): Register can only be written if SyncManager is disabled (+0x6[0] = 0).

Diagram



Fields

Field	Function
15-0 BF0	First byte that will be handled by SyncManager Bit field access for ECAT: r/(w) Bit field access for PDI: r/-

55.6.1.109 Register Length SyncManager (SYNCMANAGER_LENGTH0 - SYNCMANAGER_LENGTH15)

Offset

For a = 0 to 15:

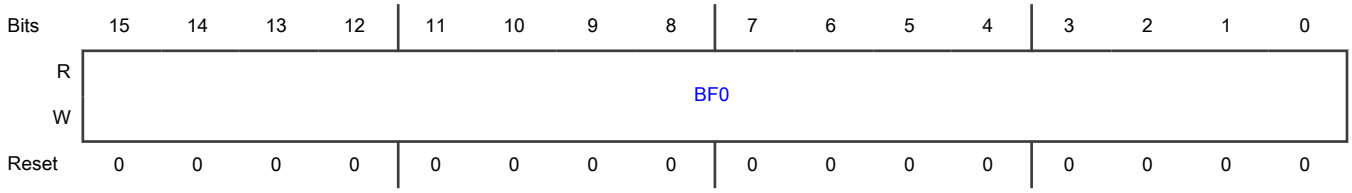
Register	Offset
SYNCMANAGER_LENGTHa	802h + (a × 8h)

Function

NOTE

r/(w): Register can only be written if SyncManager is disabled (+0x6[0] = 0).

Diagram



Fields

Field	Function
15-0 BF0	Number of bytes assigned to SyncManager (shall be greater than 1 otherwise SyncManager is not activated. If set to 1, only Watchdog Trigger is generated if configured) Bit field access for ECAT: r/(w) Bit field access for PDI: r/-

55.6.1.110 Register Length SyncManager (SYNCMANAGER_LENGTH0_PDI - SYNCMANAGER_LENGTH15_PDI)

Offset

For a = 0 to 15:

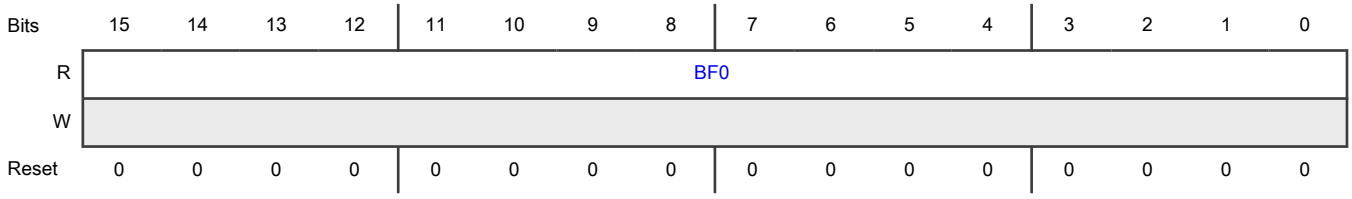
Register	Offset
SYNCMANAGER_LENGTHa_PDI	802h + (a × 8h)

Function

NOTE

r/(w): Register can only be written if SyncManager is disabled (+0x6[0] = 0).

Diagram



Fields

Field	Function
15-0	Number of bytes assigned to SyncManager (shall be greater than 1

Table continues on the next page...

Field	Function
BF0	otherwise SyncManager is not activated. If set to 1, only Watchdog Trigger is generated if configured) Bit field access for ECAT: r/(w) Bit field access for PDI: r/-

55.6.1.111 Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER0 - SYNCMANAGER_CONTROL_REGISTER15)

Offset

For a = 0 to 15:

Register	Offset
SYNCMANAGER_CONTROL_REGISTERa	804h + (a × 8h)

Function

NOTE

r/(w): Register can only be written if SyncManager is disabled (+0x6[0] = 0).

Diagram

Bits	7	6	5	4	3	2	1	0
R	BF7	BF6	BF5	BF4	BF2		BF0	
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 BF7	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
6 BF6	Watchdog Trigger Enable: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - Disabled 1b - Enabled
5	Interrupt in AL Event Request Register:

Table continues on the next page...

Table continued from the previous page...

Field	Function
BF5	Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - Disabled 1b - Enabled
4 BF4	Interrupt in ECAT Event Request Register: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - Disabled 1b - Enabled
3-2 BF2	Direction: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 00b - Read: ECAT read access, PDI write access. 01b - Write: ECAT write access, PDI read access. 10b - Reserved 11b - Reserved
1-0 BF0	Operation Mode: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 00b - Buffered (3 buffer mode) 01b - Reserved 10b - Mailbox (Single buffer mode) 11b - Reserved

55.6.1.112 Register Control Register SyncManager (SYNCMANAGER_CONTROL_REGISTER0_PDI - SYNCMANAGER_CONTROL_REGISTER15_PDI)

Offset

For a = 0 to 15:

Register	Offset
SYNCMANAGER_CONTROL_REGISTERa_PDI	804h + (a × 8h)

Function

NOTE

r/(w): Register can only be written if SyncManager is disabled (+0x6[0] = 0).

Diagram

Bits	7	6	5	4	3	2	1	0
R	BF7	BF6	BF5	BF4	BF2		BF0	
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 BF7	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
6 BF6	Watchdog Trigger Enable: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - Disabled 1b - Enabled
5 BF5	Interrupt in AL Event Request Register: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - Disabled 1b - Enabled
4 BF4	Interrupt in ECAT Event Request Register: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 0b - Disabled 1b - Enabled
3-2 BF2	Direction: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 00b - Read: ECAT read access, PDI write access.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Write: ECAT write access, PDI read access. 10b - Reserved 11b - Reserved
1-0 BF0	Operation Mode: Bit field access for ECAT: r/(w) Bit field access for PDI: r/- 00b - Buffered (3 buffer mode) 01b - Reserved 10b - Mailbox (Single buffer mode) 11b - Reserved

55.6.1.113 Register Status Register SyncManager (SYNCMANAGER_STATUS0 - SYNCMANAGER_STATUS15)

Offset

For a = 0 to 15:

Register	Offset
SYNCMANAGER_STATUSa	805h + (a × 8h)

Function

Register Status Register SyncManager

Diagram

Bits	7	6	5	4	3	2	1	0
R	BF7	BF6	BF4		BF3	BF2	BF1	BF0
W								
Reset	0	0	1	1	0	0	0	0

Fields

Field	Function
7 BF7	Write buffer in use (opened) Bit field access for ECAT: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Bit field access for PDI: r/-
6 BF6	Read buffer in use (opened) Bit field access for ECAT: r/- Bit field access for PDI: r/-
5-4 BF4	Buffered mode: buffer status (last written buffer): Mailbox mode: reserved Bit field access for ECAT: r/- Bit field access for PDI: r/- 00b - 1st buffer 01b - 2nd buffer 10b - 3rd buffer 11b - (no buffer written)
3 BF3	Mailbox mode: mailbox status: Buffered mode: reserved Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Mailbox empty 1b - Mailbox full
2 BF2	Reserved Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Interrupt Read: **NOTE** This interrupt is signalled to the writing side if enabled in the SM Control register . Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Interrupt cleared after first byte of buffer was written 1b - Interrupt after buffer was completely and successfully read
0 BF0	Interrupt Write:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This interrupt is signalled to the reading side if enabled in the SM Control register . Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Interrupt cleared after first byte of buffer was read 1b - Interrupt after buffer was completely and successfully written

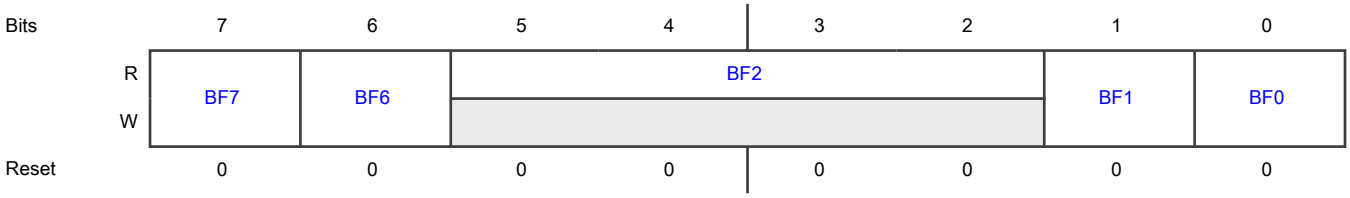
55.6.1.114 Register Activate SyncManager (SYNCMANAGER_ACTIVATE0 - SYNCMANAGER_ACTIVATE15)

Offset

For a = 0 to 15:

Register	Offset
SYNCMANAGER_ACTIVATEa	806h + (a × 8h)

Diagram



Fields

Field	Function
7 BF7	Latch Event PDI: Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - No 1b - Generate Latch events when PDI issues a buffer exchange or when PDI accesses buffer start address
6 BF6	Latch Event ECAT: Bit field access for ECAT: r/w Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No 1b - Generate Latch event when EtherCAT master issues a buffer exchange
5-2 BF2	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Repeat Request: A toggle of Repeat Request means that a mailbox retry is needed (primarily used in conjunction with ECAT Read Mailbox) Bit field access for ECAT: r/w Bit field access for PDI: r/-
0 BF0	SyncManager Enable/Disable: Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - Disable: Access to Memory without SyncManager control 1b - Enable: SyncManager is active and controls Memory area set in configuration

55.6.1.115 Register Activate SyncManager (SYNCMANAGER_ACTIVATE0_PDI - SYNCMANAGER_ACTIVATE15_PDI)

Offset

For a = 0 to 15:

Register	Offset
SYNCMANAGER_ACTIVATEa_PDI	806h + (a × 8h)

Diagram

Bits	7	6	5	4	3	2	1	0
R	BF7	BF6	BF2				BF1	BF0
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 BF7	Latch Event PDI: Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - No 1b - Generate Latch events when PDI issues a buffer exchange or when PDI accesses buffer start address
6 BF6	Latch Event ECAT: Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - No 1b - Generate Latch event when EtherCAT master issues a buffer exchange
5-2 BF2	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Repeat Request: A toggle of Repeat Request means that a mailbox retry is needed (primarily used in conjunction with ECAT Read Mailbox) Bit field access for ECAT: r/w Bit field access for PDI: r/-
0 BF0	SyncManager Enable/Disable: Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - Disable: Access to Memory without SyncManager control 1b - Enable: SyncManager is active and controls Memory area set in configuration

55.6.1.116 Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL0 - SYNCMANAGER_PDI_CONTROL15)

Offset

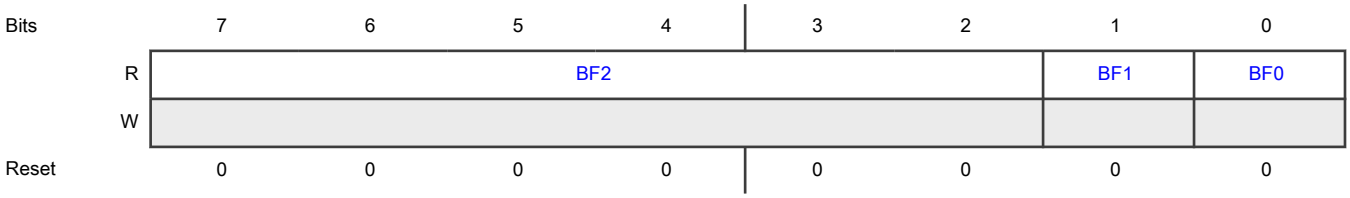
For a = 0 to 15:

Register	Offset
SYNCMANAGER_PDI_CONTROLa	807h + (a × 8h)

Function

Throughout this chapter, y specifies SyncManager (y=0x0 to 0xF).

Diagram



Fields

Field	Function
7-2 BF2	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Repeat Ack: If this is set to the same value as that set by Repeat Request, the PDI acknowledges the execution of a previous set Repeat request. Bit field access for ECAT: r/- Bit field access for PDI: r/w
0 BF0	Deactivate SyncManager: Read Deactivate SyncManager: Write **NOTE** Writing 1 is delayed until the end of the frame, which is currently processed

Read	Description
0	Normal operation, SyncManager activated.

Field	Function
	Read / **Description** 1 / SyncManager deactivated and reset. SyncManager locks access to Memory area. **Write** / **Description** 0 / Activate SyncManager 1 / Request SyncManager deactivation Bit field access for ECAT: r/- Bit field access for PDI: r/w

55.6.1.117 Register PDI Control SyncManager (SYNCMANAGER_PDI_CONTROL0_PDI - SYNCMANAGER_PDI_CONTROL15_PDI)

Offset

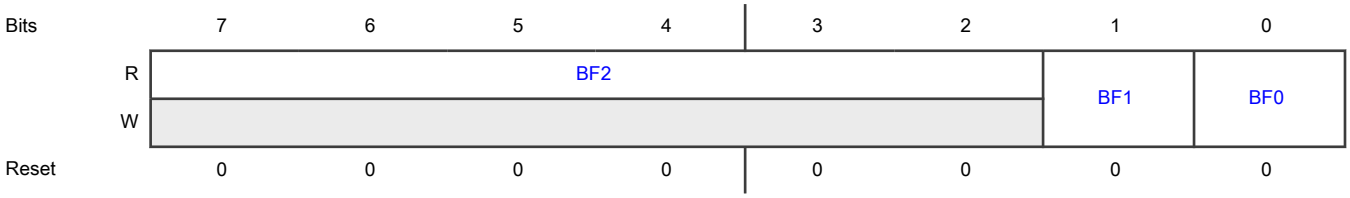
For a = 0 to 15:

Register	Offset
SYNCMANAGER_PDI_CONTROLa_PDI	807h + (a × 8h)

Function

Throughout this chapter, y specifies SyncManager (y=0x0 to 0xF).

Diagram



Fields

Field	Function
7-2	Reserved, write 0
BF2	Bit field access for ECAT: r/- Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>1
BF1</td><td>Repeat Ack:
If this is set to the same value as that set by Repeat Request, the PDI acknowledges the execution of a previous set Repeat request.
Bit field access for ECAT: r/-
Bit field access for PDI: r/w</td></tr>
<tr><td>0
BF0</td><td>Deactivate SyncManager:
Read
Deactivate SyncManager: Write
NOTE
Writing 1 is delayed until the end of the frame, which is currently processed
<table>
<tr><th>Read</th><th>Description</th></tr>
<tr><td>0</td><td>Normal operation, SyncManager activated.</td></tr>
<tr><td>1</td><td>SyncManager deactivated and reset. SyncManager locks access to Memory area.</td></tr>
</table>
<table>
<tr><th>Write</th><th>Description</th></tr>
<tr><td>0</td><td>Activate SyncManager</td></tr>
<tr><td>1</td><td>Request SyncManager deactivation</td></tr>
</table>
Bit field access for ECAT: r/-
Bit field access for PDI: r/w</td></tr>
</table>

55.6.1.118 Distributed Clocks Receive Times (RECEIVE_TIMES)

Offset

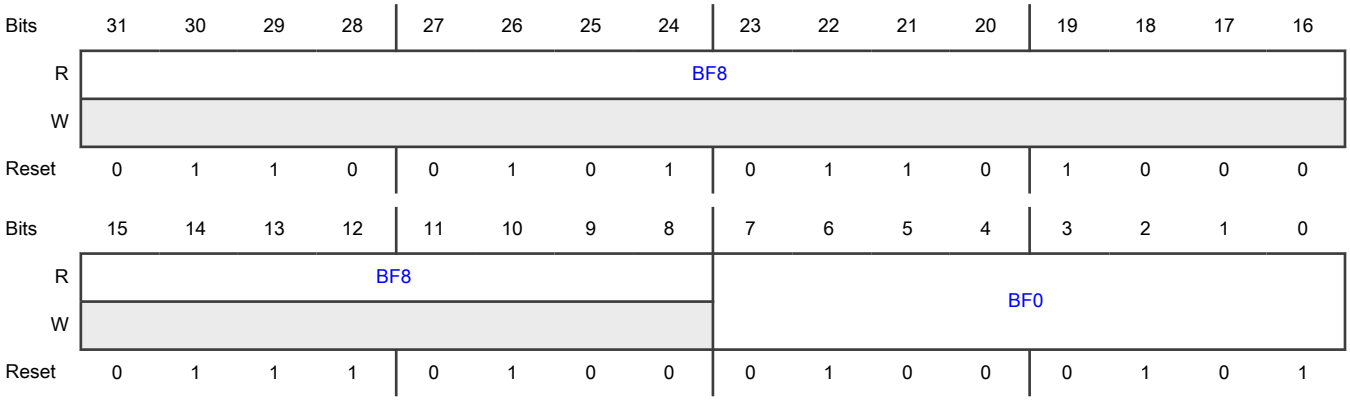
Register	Offset
RECEIVE_TIMES	900h

Function

NOTE

The time stamps cannot be read in the same frame in which this register was written.

Diagram



Fields

Field	Function
31-8 BF8	Local time at the beginning of the last receive frame containing a write access to register 0x0900. Bit field access for ECAT: r/- Bit field access for PDI: r/-
7-0 BF0	Write A write access to register 0x0900 with BWR or FPWR latches the local time at the beginning of the receive frame (start first bit of preamble) at each port. Read: Local time at the beginning of the last receive frame containing a write access to this register. Write : A write access latches the local time at the beginning of the receive frame at port 0. It enables the time stamping at the other ports. NOTE: FPWR requires an address match for accessing this register like any FPWR command. All write commands with address match will increment the working counter (e.g., APWR), but they will not trigger receive time latching Bit field access for ECAT: r/w (special function) Bit field access for PDI: r/-

55.6.1.119 Distributed Clocks Receive Times (RECEIVE_TIMES_PDI)

Offset

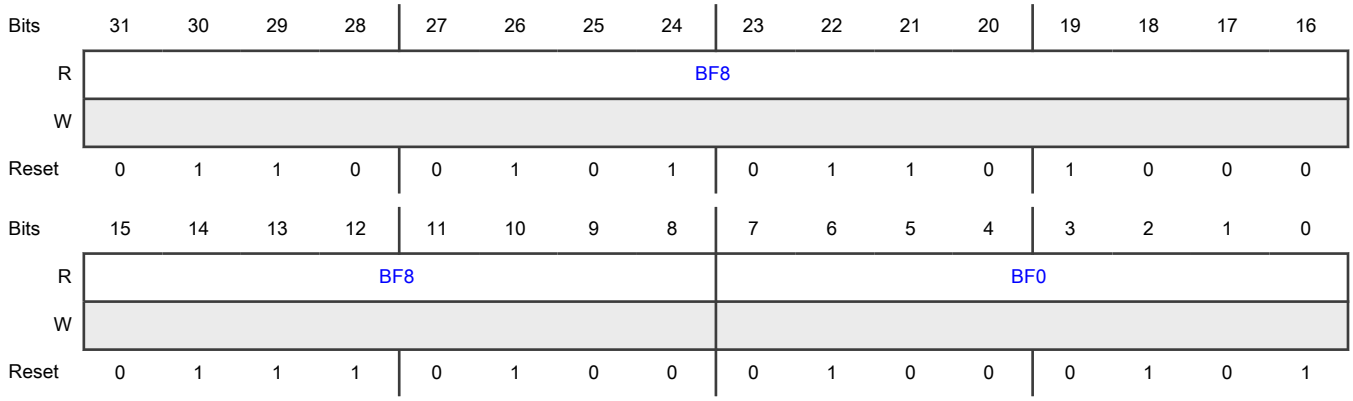
Register	Offset
RECEIVE_TIMES_PDI	900h

Function

NOTE

The time stamps cannot be read in the same frame in which this register was written.

Diagram



Fields

Field	Function
31-8 BF8	Local time at the beginning of the last receive frame containing a write access to register 0x0900. Bit field access for ECAT: r/- Bit field access for PDI: r/-
7-0 BF0	Write A write access to register 0x0900 with BWR or FPWR latches the local time at the beginning of the receive frame (start first bit of preamble) at each port. Read: Local time at the beginning of the last receive frame containing a write access to this register. Write : A write access latches the local time at the beginning of the receive frame at port 0. It enables the time stamping at the other ports. NOTE: FPWR requires an address match for accessing this register like any FPWR command. All write commands with address match will increment the working counter (e.g., APWR), but they will not trigger receive time latching Bit field access for ECAT: r/w (special function) Bit field access for PDI: r/-

55.6.1.120 Distributed Clocks Receive Time Port 1 (RECEIVE_TIME_PORT_1)

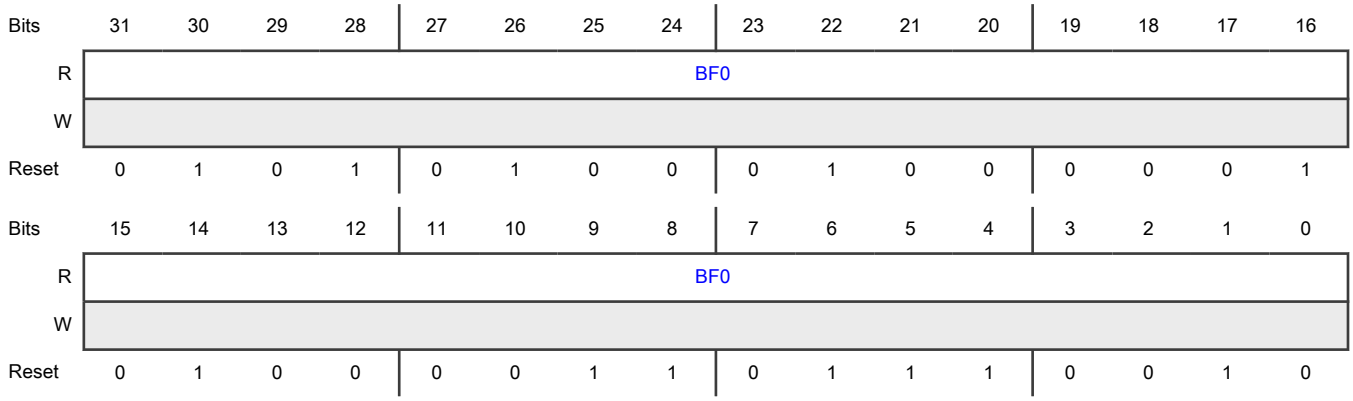
Offset

Register	Offset
RECEIVE_TIME_PORT_1	904h

Function

Register Receive Time Port 1

Diagram



Fields

Field	Function
31-0 BF0	Local time at the beginning of a frame (start first bit of preamble) received at port 1 containing a BWR or FPWR to register 0x0900. Bit field access for ECAT: r/- Bit field access for PDI: r/-

55.6.1.121 Register System Time (SYSTEM_TIME)

Offset

Register	Offset
SYSTEM_TIME	910h

Function

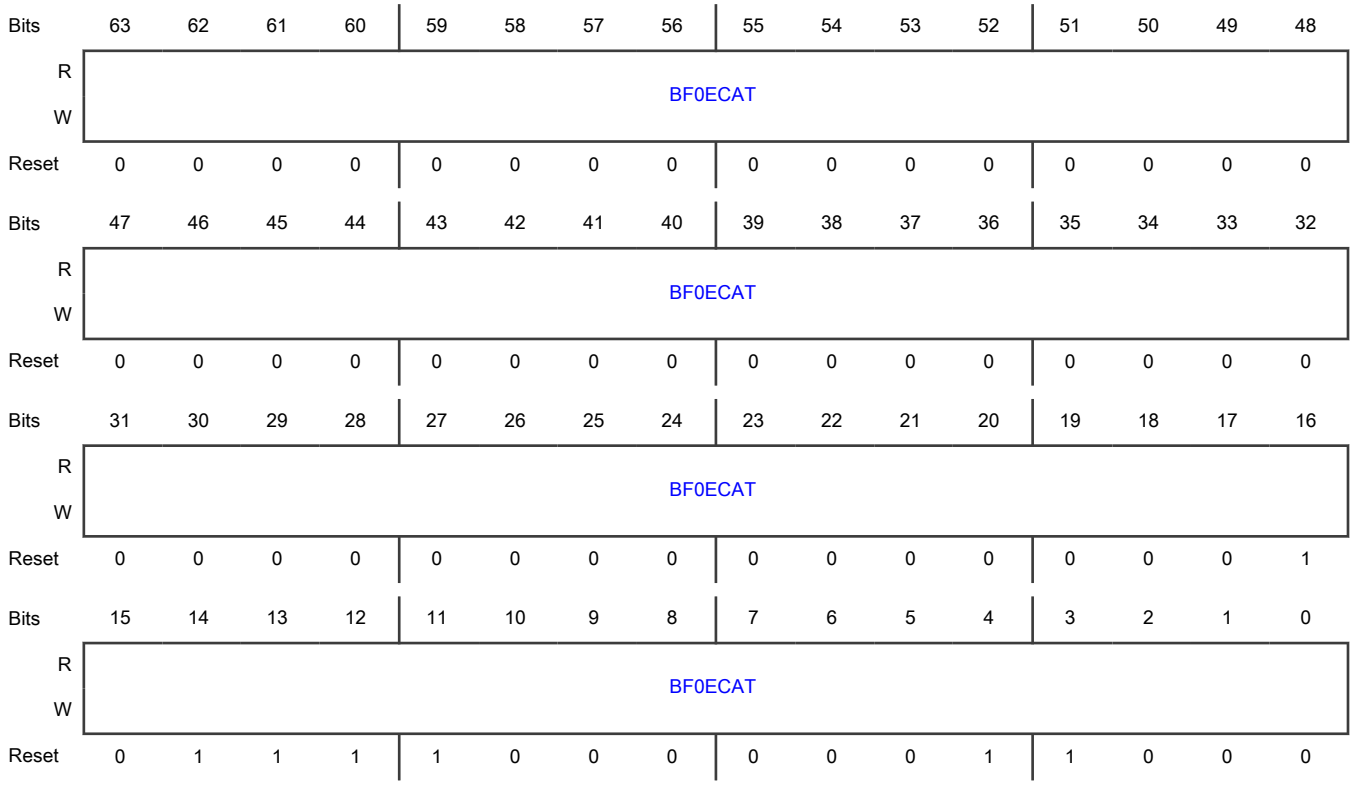
Time Loop Control unit is usually assigned to ECAT. Write access to Time Loop Control registers by PDI (and not ECAT) is only possible with explicit IP Core configuration.

NOTE

Write access to this register depends upon ESC configuration (typically ECAT, PDI only with explicit ESC configuration: System Time PDI-controlled)

.

Diagram



Fields

Field	Function
63-0 BF0ECAT	ECAT read access Local copy of the System Time when the frame passed the reference clock (i.e., including System Time Delay). Time latched at beginning of the frame (Ethernet SOF delimiter) Bit field access for ECAT: r Bit field access for PDI: nil

55.6.1.122 Register System Time (SYSTEM_TIME_PDI)

Offset

Register	Offset
SYSTEM_TIME_PDI	910h

Function

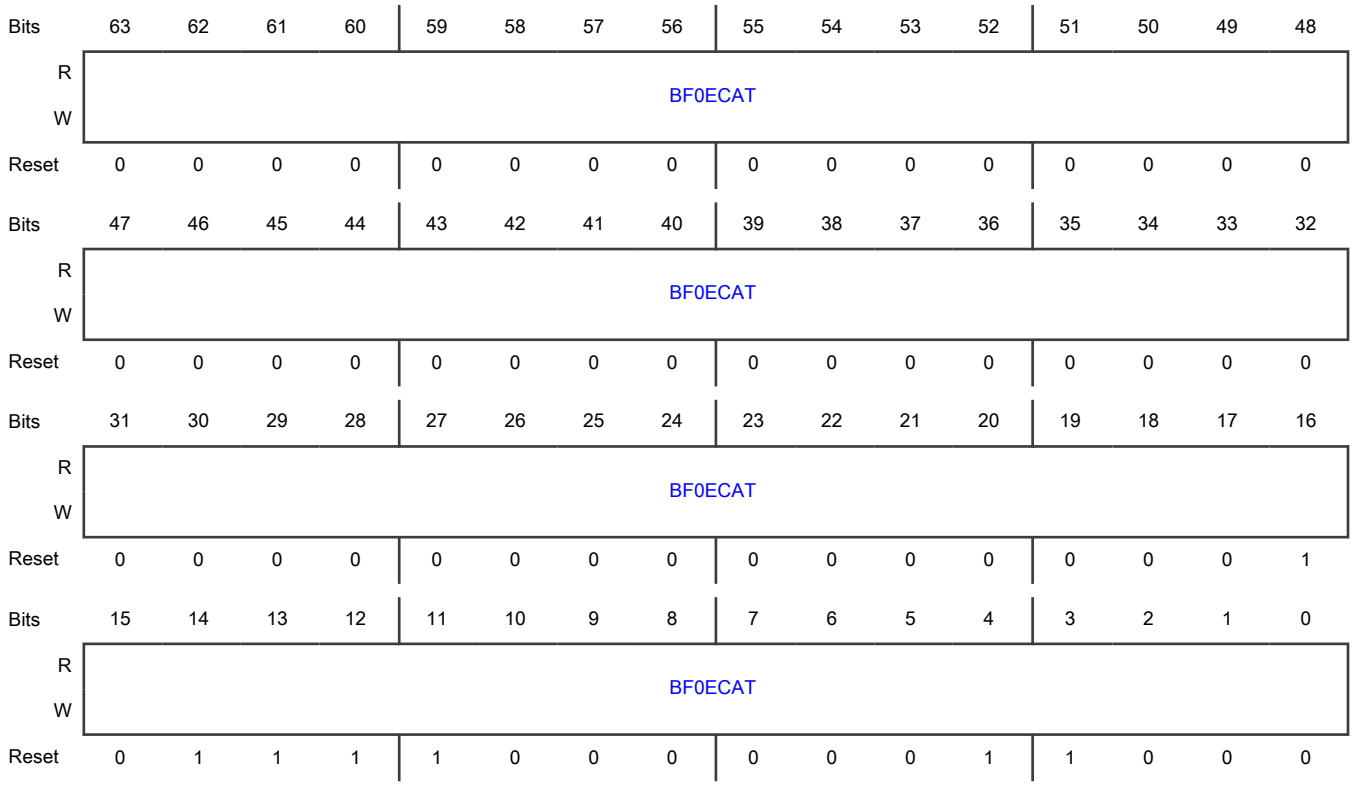
Time Loop Control unit is usually assigned to ECAT. Write access to Time Loop Control registers by PDI (and not ECAT) is only possible with explicit IP Core configuration.

NOTE

Write access to this register depends upon ESC configuration (typically ECAT, PDI only with explicit ESC configuration: System Time PDI-controlled)

.

Diagram



Fields

Field	Function
63-0 BF0ECAT	PDI read access Local copy of the System Time. Time latched when reading first byte (0x0910) Bit field access for PDI: r Bit field access for ECAT: nil

55.6.1.123 Distributed Clocks Register Receive Time ECAT Processing Unit (RECEIVE_TIME_ECAT_PROCESSING_UNIT)

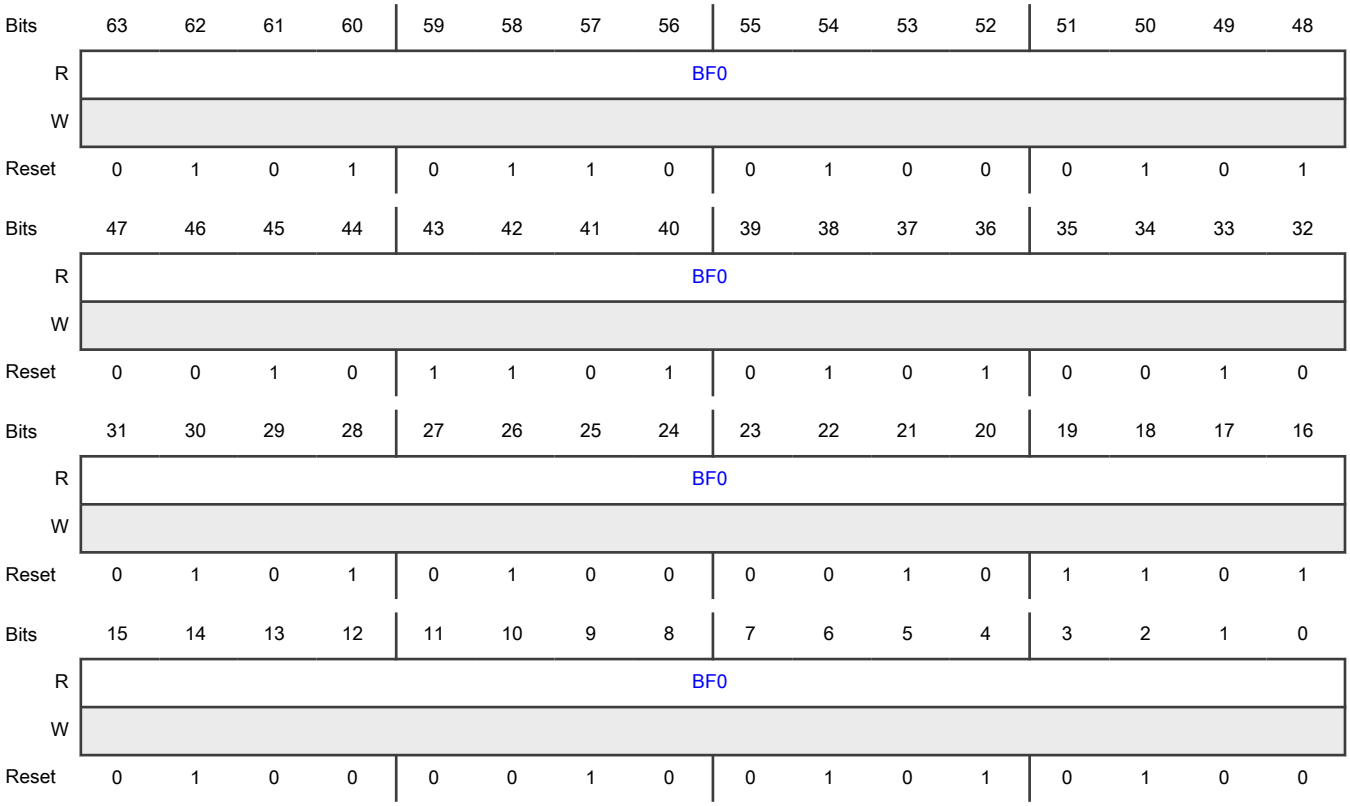
Offset

Register	Offset
RECEIVE_TIME_ECAT_PROCESSING_UNIT	918h

Function

Register Receive Time ECAT Processing Unit

Diagram



Fields

Field	Function
63-0 BF0	Local time at the beginning of a frame (start first bit of preamble) received at the ECAT Processing Unit containing a write access to register 0x0900 NOTE: E.g., if port 0 is open, this register reflects the Receive Time Port 0 as a 64 Bit value. Any valid EtherCAT write access to register 0x0900 triggers latching, not only BWR/FPWR commands as with register 0x0900. Bit field access for ECAT: r/- Bit field access for PDI: r/-

55.6.1.124 Register System Time Offset (SYSTEM_TIME_OFFSET)

Offset

Register	Offset
SYSTEM_TIME_OFFSET	920h

Function

NOTE

Write access to this register depends upon eCAT configuration (typically ECAT, PDI only with explicit ESC configuration: System Time PDI-controlled). Reset internal system time difference filter and speed counter filter by writing Speed Counter Start (0x0930:0x0931) after changing this value.

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R W	BF0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R W	BF0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 BF0	Difference between local time and System Time. Offset is added to the local time. Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w)

55.6.1.125 Register System Time Delay (SYSTEM_TIME_DELAY)

Offset

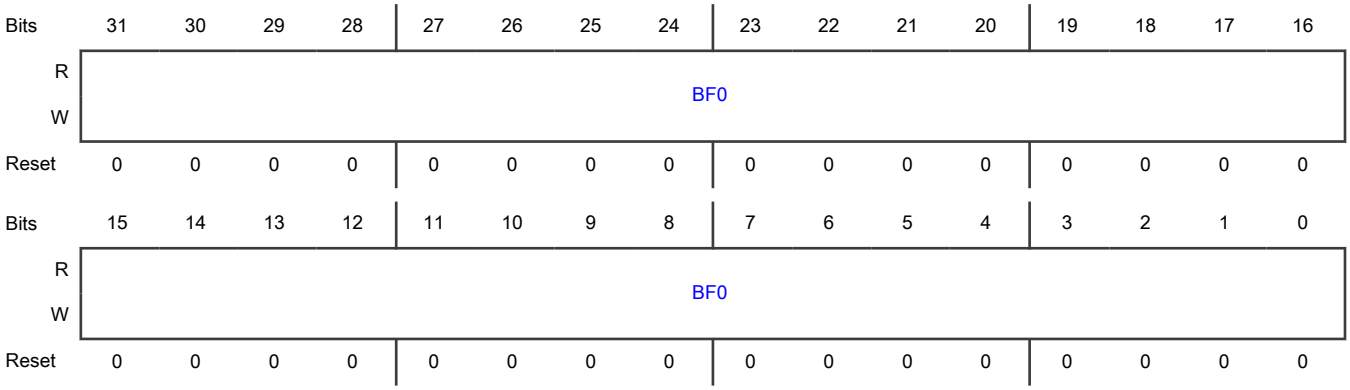
Register	Offset
SYSTEM_TIME_DELAY	928h

Function

NOTE

Write access to this register depends upon eCAT configuration (typically ECAT, PDI only with explicit eCAT configuration: System Time PDI-controlled). Reset internal system time difference filter and speed counter filter by.writing Speed Counter Start (0x0930:0x0931) after changing this value

Diagram



Fields

Field	Function
31-0 BF0	Delay between Reference Clock and the eCAT Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w)

55.6.1.126 Register System Time Difference (SYSTEM_TIME_DIFFERENCE)

Offset

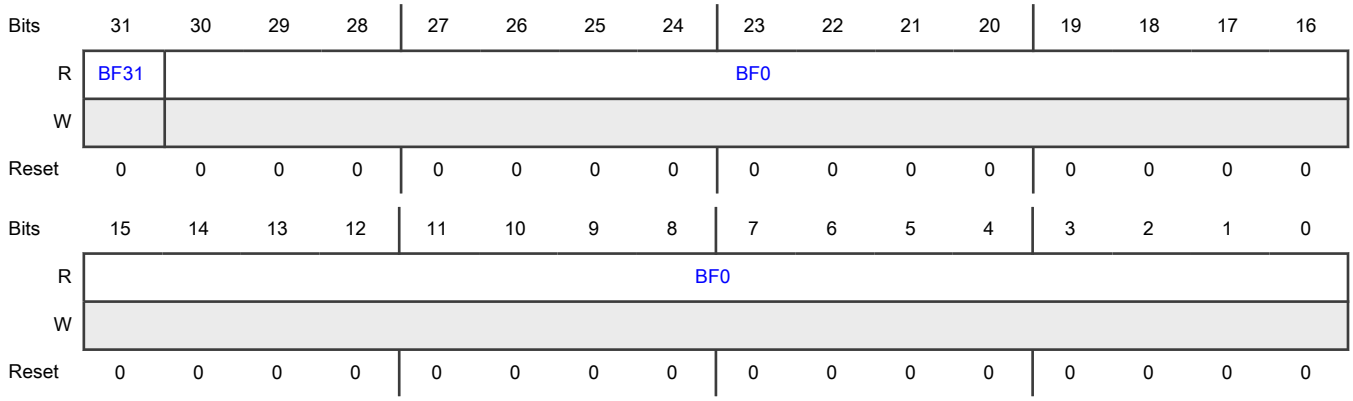
Register	Offset
SYSTEM_TIME_DIFFERENCE	92Ch

Function

NOTE

Register bits [31:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value.

Diagram



Fields

Field	Function
31 BF31	Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Local copy of System Time less than received System Time 1b - Local copy of System Time greater than or equal to received System Time
30-0 BF0	Mean difference between local copy of system Time and received System Time values *Difference = Received System Time – local copy of System Time* Bit field access for ECAT: r/- Bit field access for PDI: r/-

55.6.1.127 Register Speed Counter Start (SPEED_COUNTER_START)

Offset

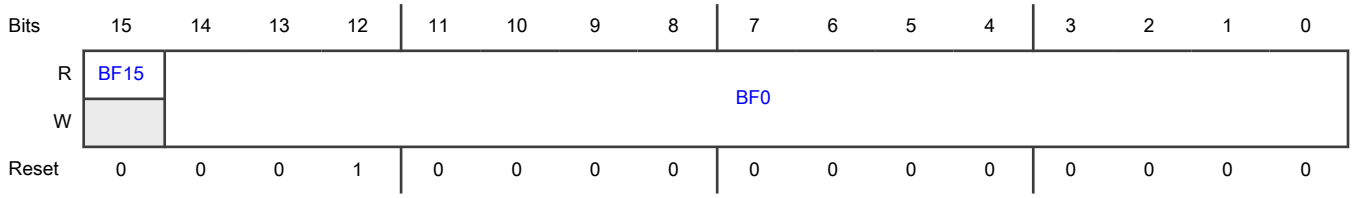
Register	Offset
SPEED_COUNTER_START	930h

Function

NOTE

Write access to this register depends upon eCAT configuration (typically ECAT, PDI only with explicit ESC configuration: System Time PDI-controlled).

Diagram



Fields

Field	Function
15 BF15	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
14-0 BF0	Bandwidth for adjustment of local copy of system Time (larger values → smaller bandwidth and smoother adjustment) A write access resets System Time Difference (0x092C:0x092F) and Speed Counter Diff (0x0932:0x0933). Valid values: 0x0080 to 0x3FFF Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w)

55.6.1.128 Register Speed Counter Diff (SPEED_COUNTER_DIFF)

Offset

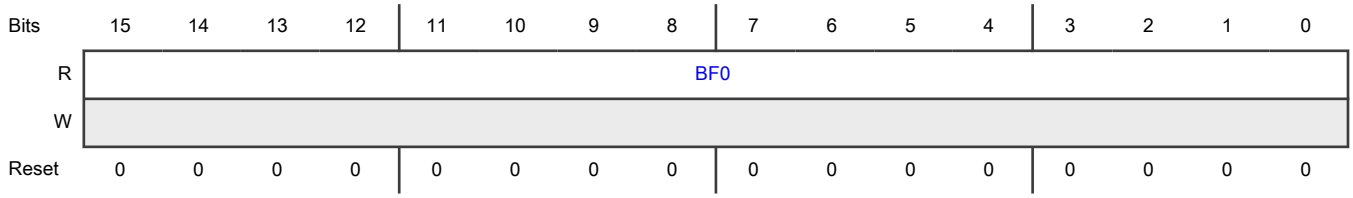
Register	Offset
SPEED_COUNTER_DIFF	932h

Function

NOTE

Calculate the clock deviation after System Time Difference has settled at a low value as follows: A / (B * C), where A = Speed Counter Diff, B = 5(Speed Counter Start + Speed Counter Diff + 2) and C = Speed Counter Start - Speed Counter Diff + 2

Diagram



Fields

Field	Function
15-0 BF0	Representation of the deviation between local clock period and Reference Clock's clock period (representation: two's complement) Range: ±(Speed Counter Start – 0x7F) Bit field access for ECAT: r/- Bit field access for PDI: r/-

55.6.1.129 Register System Time Difference Filter Depth (SYSTEM_TIME_DIFFERENCE_FILTER_DEPTH)

Offset

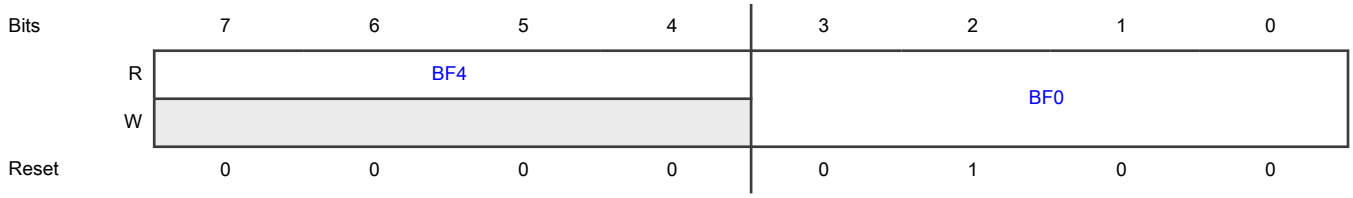
Register	Offset
SYSTEM_TIME_DIFFERENCE_FILTER_DEPTH	934h

Function

NOTE

Write access to this register depends upon eCAT configuration (typically ECAT, PDI only with explicit ESC configuration: System Time PDI-controlled).

Diagram



Fields

Field	Function
7-4 BF4	Reserved, write 0 Bit field access for ECAT: r/-

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Bit field access for PDI: r/-
3-0 BF0	Filter depth for averaging the received System Time deviation IP Core since V2.2.0/V2.02a: A write access resets System Time Difference (0x092C:0x092F) Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w)

55.6.1.130 Register Speed Counter Filter Depth (SPEED_COUNTER_FILTER_DEPTH)

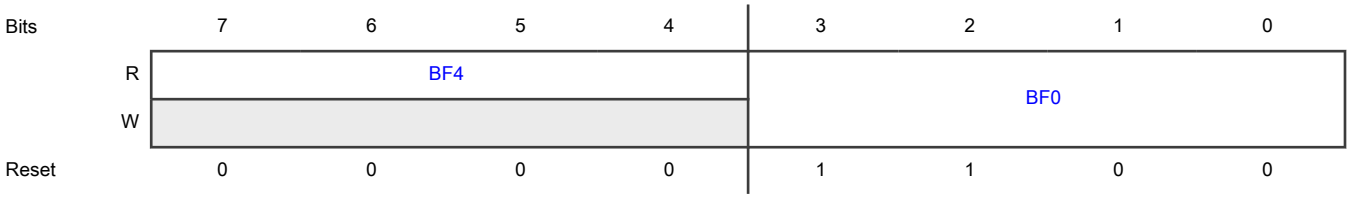
Offset

Register	Offset
SPEED_COUNTER_FILTER_DEPTH	935h

Function

NOTE: Write access to this register depends upon eCAT configuration (typically ECAT, PDI only with explicit ESC configuration: System Time PDI-controlled).

Diagram



Fields

Field	Function
7-4 BF4	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
3-0 BF0	Filter depth for averaging the clock period deviation IP Core since V2.2.0/V2.02a: A write access resets the internal speed counter filter. Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w)

55.6.1.131 Register Cyclic Unit Control (CYCLIC_UNIT_CONTROL)

Offset

Register	Offset
CYCLIC_UNIT_CONTROL	980h

Function
Register Cyclic Unit Control

Diagram

Bits	7	6	5	4	3	2	1	0
R	BF6		BF5	BF4	BF1			BF0
W								
Reset	0	0	0	1	0	0	0	0

Fields

Field	Function
7-6 BF6	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
5 BF5	Latch In unit 1: **NOTE** Latch interrupt is routed to ECAT/PDI depending on this setting Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - ECAT-controlled 1b - PDI-controlled
4 BF4	Latch In unit 0: NOTE: Latch interrupt is routed to ECAT/PDI depending on this setting. Always 1 (PDI-controlled) if System Time is PDI-controlled. Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - ECAT-controlled 1b - PDI-controlled

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	SYNC out unit control: Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - ECAT-controlled 1b - PDI-controlled

55.6.1.132 Register Cyclic Unit Control (CYCLIC_UNIT_CONTROL_PDI)

Offset

Register	Offset
CYCLIC_UNIT_CONTROL_PDI	980h

Function
Register Cyclic Unit Control

Diagram

Bits	7	6	5	4	3	2	1	0
R	BF6		BF5	BF4	BF1			BF0
W								
Reset	0	0	0	1	0	0	0	0

Fields

Field	Function
7-6 BF6	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
5 BF5	Latch In unit 1: **NOTE** Latch interrupt is routed to ECAT/PDI depending on this setting

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - ECAT-controlled 1b - PDI-controlled
4 BF4	Latch In unit 0: NOTE: Latch interrupt is routed to ECAT/PDI depending on this setting. Always 1 (PDI-controlled) if System Time is PDI-controlled. Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - ECAT-controlled 1b - PDI-controlled
3-1 BF1	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	SYNC out unit control: Bit field access for ECAT: r/w Bit field access for PDI: r/- 0b - ECAT-controlled 1b - PDI-controlled

55.6.1.133 Register Activation register (UNIT_ACTIVATION_REGISTER)

Offset

Register	Offset
UNIT_ACTIVATION_REGISTER	981h

Function

NOTE

Write to this register depends upon setting of 0x0980[0].

Diagram

Bits	7	6	5	4	3	2	1	0
R W	BF7	BF6	BF5	BF4	BF3	BF2	BF1	BF0
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 BF7	SyncSignal debug pulse (Vasily bit): This bit is self-clearing, always read 0. All pulses are generated at the same time, the cycle time is ignored. The configured pulse length is used. Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - Deactivated 1b - Immediately generate one ping only on SYNC0-1 according to 0x0981[2:1] for debugging.
6 BF6	Near future configuration (approx.): Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - 1/2 DC width future (2^31ns or 2^63ns) 1b - ~2.1 sec. future (2^31ns)
5 BF5	Start Time plausibility check: Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - Disabled. SyncSignal generation if Start Time is reached. 1b - Immediate SyncSignal generation if Start Time is outside near future (see 0x0981[6])
4 BF4	Extension of Start Time Cyclic Operation (0x0990:0x0993): Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - No extension 1b - Extend 32 bit written Start Time to 64 bit
3 BF3	Auto-activation by writing Start Time Cyclic Operation (0x0990:0x0997): Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - Disabled 1b - Auto-activation enabled. 0x0981[0] is set automatically after Start Time is written

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 BF2	SYNC1 generation: Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - Deactivated 1b - SYNC1 pulse is generated
1 BF1	SYNC0 generation: Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - Deactivated 1b - SYNC0 pulse is generated
0 BF0	Sync Out Unit activation: Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - Deactivated 1b - Activated

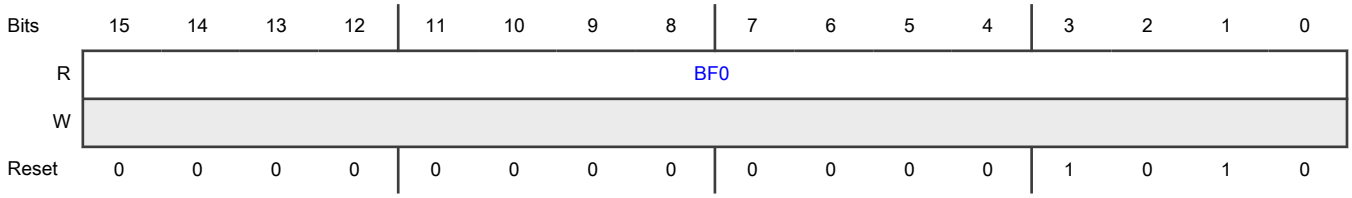
55.6.1.134 Register Pulse Length of SyncSignals (UNI_PULSE_LENGTH_OF_SYNCSIGNALS)

Offset

Register	Offset
UNI_PULSE_LENGTH_OF_SYNCSIGNALS	982h

Function
Register Pulse Length of SyncSignals

Diagram



Fields

Field	Function
15-0	Pulse length of SyncSignals (in Units of 10ns)
BF0	Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset for bit depends on configuration 0000_0000_0000_0000b - Acknowledge mode: SyncSignal will be cleared by reading SYNC[1:0] Status register

55.6.1.135 Register Activation Status (UNIT_ACTIVATION_STATUS)

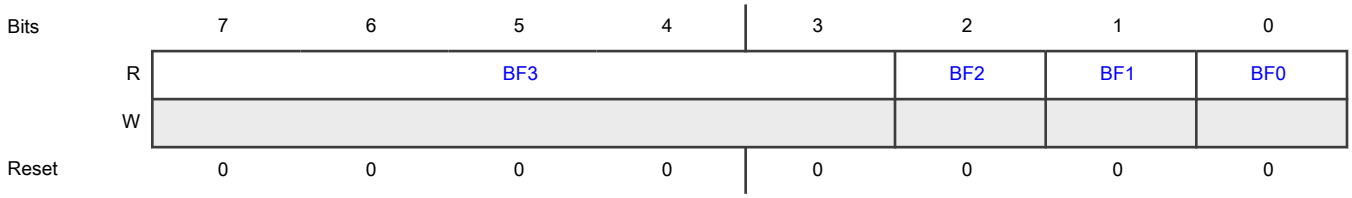
Offset

Register	Offset
UNIT_ACTIVATION_STATUS	984h

Function

Register Activation Status

Diagram



Fields

Field	Function
7-3	Reserved
BF3	Bit field access for ECAT: r/- Bit field access for PDI: r/-

Table continues on the next page...

Table continued from the previous page...

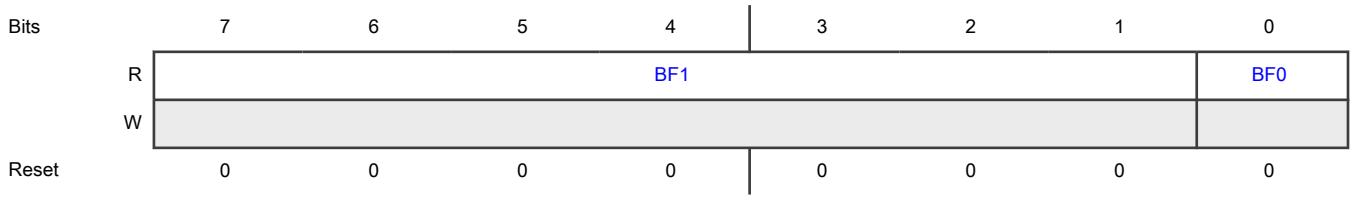
Field	Function
2 BF2	Start Time Cyclic Operation (0x0990:0x0997) plausibility check result when Sync Out Unit was activated: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Start Time was within near future 1b - Start Time was out of near future (0x0981[6])
1 BF1	SYNC1 activation state: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - First SYNC1 pulse is not pending 1b - First SYNC1 pulse is pending
0 BF0	SYNC0 activation state: Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - First SYNC0 pulse is not pending 1b - First SYNC0 pulse is pending

55.6.1.136 Register SYNC0 Status (UNIT_SYNC0_STATUS)

Offset

Register	Offset
UNIT_SYNC0_STATUS	98Eh

Diagram



Fields

Field	Function
7-1 BF1	Reserved Bit field access for ECAT: r/-

Table continues on the next page...

Table continued from the previous page...

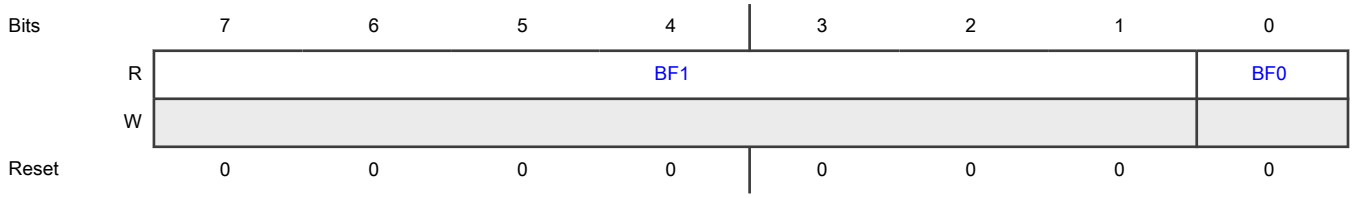
Field	Function
	Bit field access for PDI: r/-
0 BF0	SYNC0 state for Acknowledge mode. SYNC0 in Acknowledge mode is cleared by reading this register from PDI, use only in Acknowledge mode Bit field access for ECAT: r/- Bit field access for PDI: r/-

55.6.1.137 Register SYNC1 Status (UNIT_SYNC1_STATUS)

Offset

Register	Offset
UNIT_SYNC1_STATUS	98Fh

Diagram



Fields

Field	Function
7-1 BF1	Reserved Bit field access for ECAT: r/- Bit field access for PDI: r/-
0 BF0	SYNC1 state for Acknowledge mode. SYNC1 in Acknowledge mode is cleared by reading this register from PDI, use only in Acknowledge mode Bit field access for ECAT: r/- Bit field access for PDI: r/-

55.6.1.138 Register Start Time Cyclic Operation (UNIT_START_TIME_CYCLIC_OPERATION)

Offset

Register	Offset
UNIT_START_TIME_CYCLIC_OPERATION	990h

Function

NOTE

Register bits [63:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value. Write to this register depends upon setting of 0x0980[0]. Write value is taken over when 0x0981[0] has a transition to 1. Auto-activation (0x0981[3]=1): 0x0981[0] is set automatically after this register is written. Extension of Start Time (0x0981[4]=1): upper 32 bits are automatically extended after writing this register if only lower 32 bits are written within one frame.

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R W	BF0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R W	BF0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0	Write: Start time (System time) of cyclic operation in ns Read: System time of next SYNC0 pulse in ns

Table continues on the next page...

Field	Function
BF0	Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w)

55.6.1.139 Register Next SYNC1 Pulse (UNIT_NEXT_SYNC1_PULSE)

Offset

Register	Offset
UNIT_NEXT_SYNC1_PULSE	998h

Function

NOTE

Register bits [63:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value.

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 BF0	System time of next SYNC1 pulse in ns Bit field access for ECAT: r/- Bit field access for PDI: r/-

55.6.1.140 Register SYNC0 Cycle Time (UNIT_SYNC0_CYCLE_TIME)

Offset

Register	Offset
UNIT_SYNC0_CYCLE_TIME	9A0h

Function

NOTE

Write to this register depends upon setting of 0x0980[0].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BF0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BF0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BF0	Time between two consecutive SYNC0 pulses in ns. Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0000_0000_0000_0000_0000_0000_0000_0000b - Single shot mode, generate only one SYNC0 pulse.

55.6.1.141 Register SYNC1 Cycle Time (UNIT_SYNC1_CYCLE_TIME)

Offset

Register	Offset
UNIT_SYNC1_CYCLE_TIME	9A4h

Function

NOTE

Write to this register depends upon setting of 0x0980[0].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BF0	Time between SYNC0 pulse and SYNC1 pulse in ns Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w)

55.6.1.142 Register Latch0 Control (LATCH0_CONTROL)

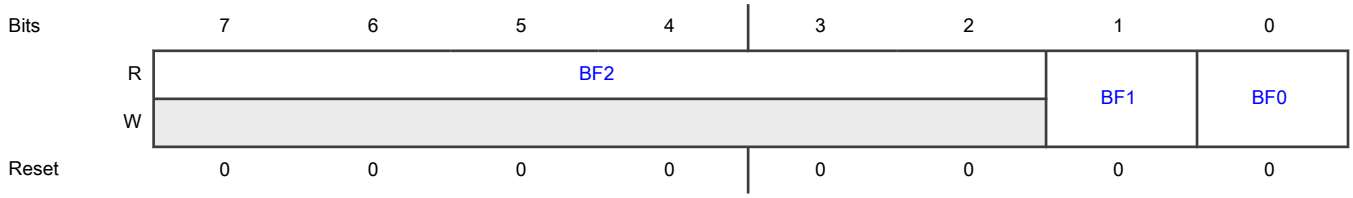
Offset

Register	Offset
LATCH0_CONTROL	9A8h

Function

NOTE: Write access depends upon setting of 0x0980[4].

Diagram



Fields

Field	Function
7-2 BF2	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Latch0 negative edge: Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - Continuous Latch active 1b - Single event (only first event active)
0 BF0	Latch0 positive edge: Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - Continuous Latch active 1b - Single event (only first event active)

55.6.1.143 Register Latch1 Control (LATCH1_CONTROL)

Offset

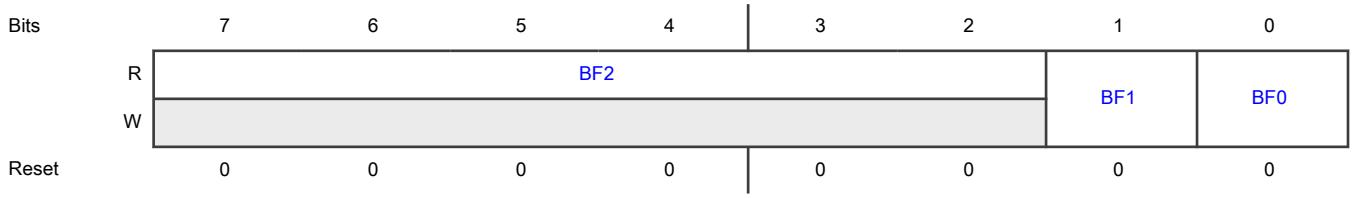
Register	Offset
LATCH1_CONTROL	9A9h

Function

NOTE

Write access depends upon setting of 0x0980[5].

Diagram



Fields

Field	Function
7-2 BF2	Reserved, write 0 Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Latch1 negative edge: Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - Continuous Latch active 1b - Single event (only first event active)
0 BF0	Latch1 positive edge: Bit field access for ECAT: r/(w) Bit field access for PDI: r/(w) 0b - Continuous Latch active 1b - Single event (only first event active)

55.6.1.144 Register Latch0 Status (LATCH0_STATUS)

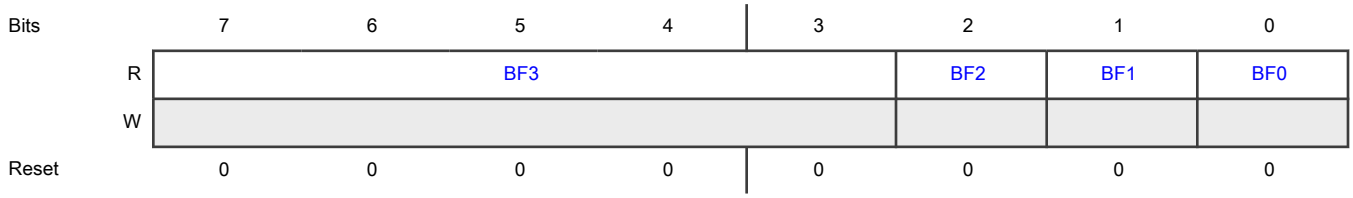
Offset

Register	Offset
LATCH0_STATUS	9AEh

Function

Register Latch0 Status

Diagram



Fields

Field	Function
7-3 BF3	Reserved Bit field access for ECAT: r/- Bit field access for PDI: r/-
2 BF2	Latch0 pin state Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Event Latch0 negative edge. Flag cleared by reading out Latch0 Time Negative Edge. Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Negative edge not detected or continuous mode 1b - Negative edge detected in single event mode only
0 BF0	Event Latch0 positive edge. Flag cleared by reading out Latch0 Time Positive Edge Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Positive edge not detected or continuous mode 1b - Positive edge detected in single event mode only.

55.6.1.145 Register Latch1 Status (LATCH1_STATUS)

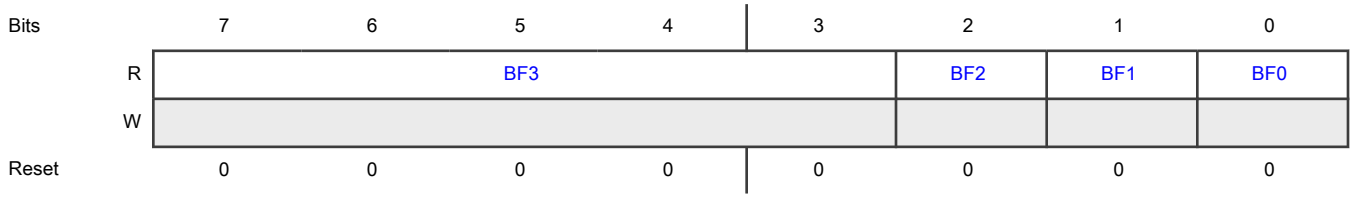
Offset

Register	Offset
LATCH1_STATUS	9AFh

Function

Register Latch1 Status

Diagram



Fields

Field	Function
7-3 BF3	Reserved Bit field access for ECAT: r/- Bit field access for PDI: r/-
2 BF2	Latch1 pin state Bit field access for ECAT: r/- Bit field access for PDI: r/-
1 BF1	Event Latch1 negative edge. Flag cleared by reading out Latch1 Time Negative Edge. Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Negative edge not detected or continuous mode 1b - Negative edge detected in single event mode only.
0 BF0	Event Latch1 positive edge. Flag cleared by reading out Latch1 Time Positive Edge. Bit field access for ECAT: r/- Bit field access for PDI: r/- 0b - Positive edge not detected or continuous mode 1b - Positive edge detected in single event mode only.

55.6.1.146 Register Latch0 Time Positive Edge (LATCH0_TIME_POSITIVE_EDGE)

Offset

Register	Offset
LATCH0_TIME_POSITIVE_EDGE	9B0h

Function

NOTE

Register bits [63:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value. Reading this register from ECAT clears Latch0 Status 0x09AE[0], if 0x0980[4]=0. Writing to this register from ECAT is not possible.

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 BF0	System time at the positive edge of the Latch0 signal. Bit field access for ECAT: r(ack)/- Bit field access for PDI: r/-

55.6.1.147 Register Latch0 Time Negative Edge (LATCH0_TIME_NEGATIVE_EDGE)

Offset

Register	Offset
LATCH0_TIME_NEGATIVE_EDGE	9B8h

Function

NOTE

Register bits [63:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value. Reading this register from ECAT clears Latch0 Status 0x09AE[1] if 0x0980[4]=0. Writing to this register from ECAT is not possible.

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 BF0	System time at the negative edge of the Latch0 signal. Bit field access for ECAT: r(ack)/- Bit field access for PDI: r/-

55.6.1.148 Register Latch1 Time Positive Edge (LATCH1_TIME_POSITIVE_EDGE)

Offset

Register	Offset
LATCH1_TIME_POSITIVE_EDGE	9C0h

Function

NOTE

Register bits [63:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value. Reading this register from ECAT clears Latch0 Status 0x09AF[0] if 0x0980[5]=0. Writing to this register from ECAT is not possible.

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 BF0	System time at the positive edge of the Latch1 signal. Bit field access for ECAT: r(ack)/- Bit field access for PDI: r/-

55.6.1.149 Register Latch1 Time Negative Edge (LATCH1_TIME_NEGATIVE_EDGE)

Offset

Register	Offset
LATCH1_TIME_NEGATIVE_EDGE	9C8h

Function

NOTE

Register bits [63:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value. Reading this register from ECAT clears Latch0 Status 0x09AF[1] if 0x0980[5]=0. Writing to this register from ECAT is not possible.

Diagram

Bits	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BF0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
63-0 BF0	System time at the negative edge of the Latch1 signal. Bit field access for ECAT: r(ack)/- Bit field access for PDI: r/-

55.6.1.150 Register EtherCAT Buffer Change Event Time (ETHERCAT_BUFFER_CHANGE_EVENT_TIME)

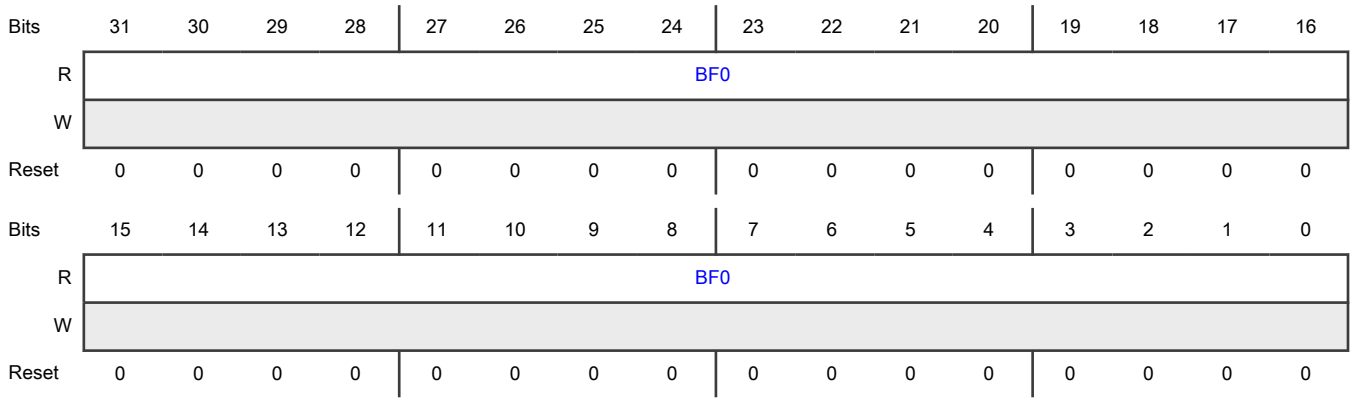
Offset

Register	Offset
ETHERCAT_BUFFER_CHANGE_EVENT_TIME	9F0h

Function

NOTE: Register bits [31:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value.

Diagram



Fields

Field	Function
31-0 BF0	Local time at the beginning of the frame which causes at least one SyncManager to assert an ECAT event Bit field access for ECAT: r/- Bit field access for PDI: r/-

55.6.1.151 Register PDI Buffer Start Event Time (PDI_BUFFER_START_EVENT_TIME)

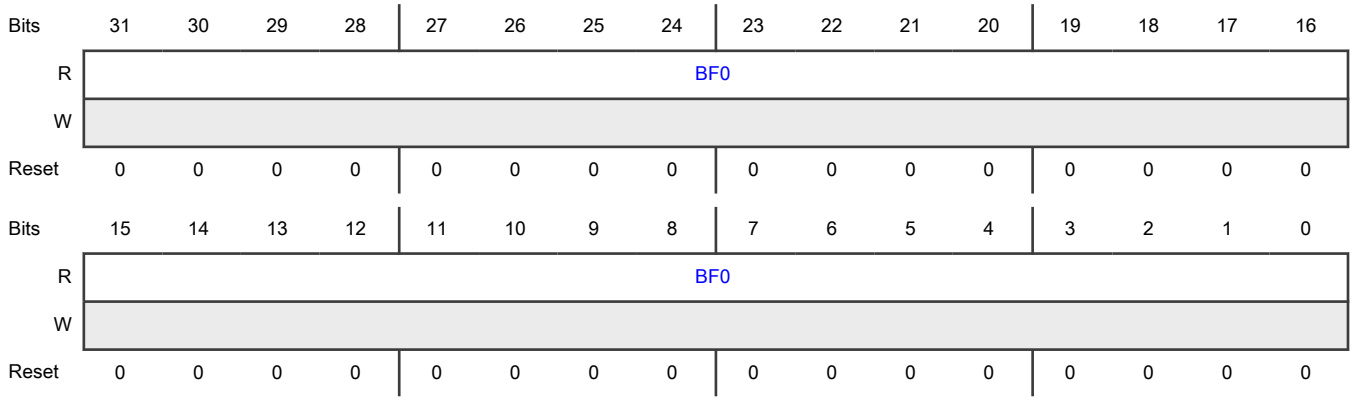
Offset

Register	Offset
PDI_BUFFER_START_EVENT_TIME	9F8h

Function

NOTE: Register bits [31:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value.

Diagram



Fields

Field	Function
31-0 BF0	Local time when at least one SyncManager asserts a PDI buffer start event Bit field access for ECAT: r/- Bit field access for PDI: r/-

55.6.1.152 Register PDI Buffer Change Event Time (PDI_BUFFER_CHANGE_EVENT_TIME)

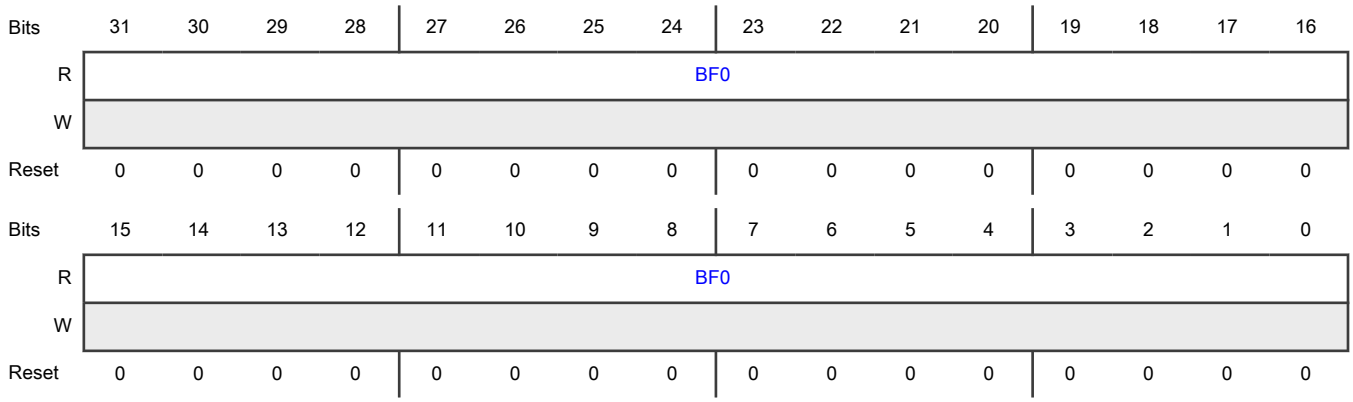
Offset

Register	Offset
PDI_BUFFER_CHANGE_EVENT_TIME	9FCh

Function

NOTE: Register bits [31:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value.

Diagram



Fields

Field	Function
31-0 BF0	Local time when at least one SyncManager asserts a PDI buffer change event Bit field access for ECAT: r/- Bit field access for PDI: r/-

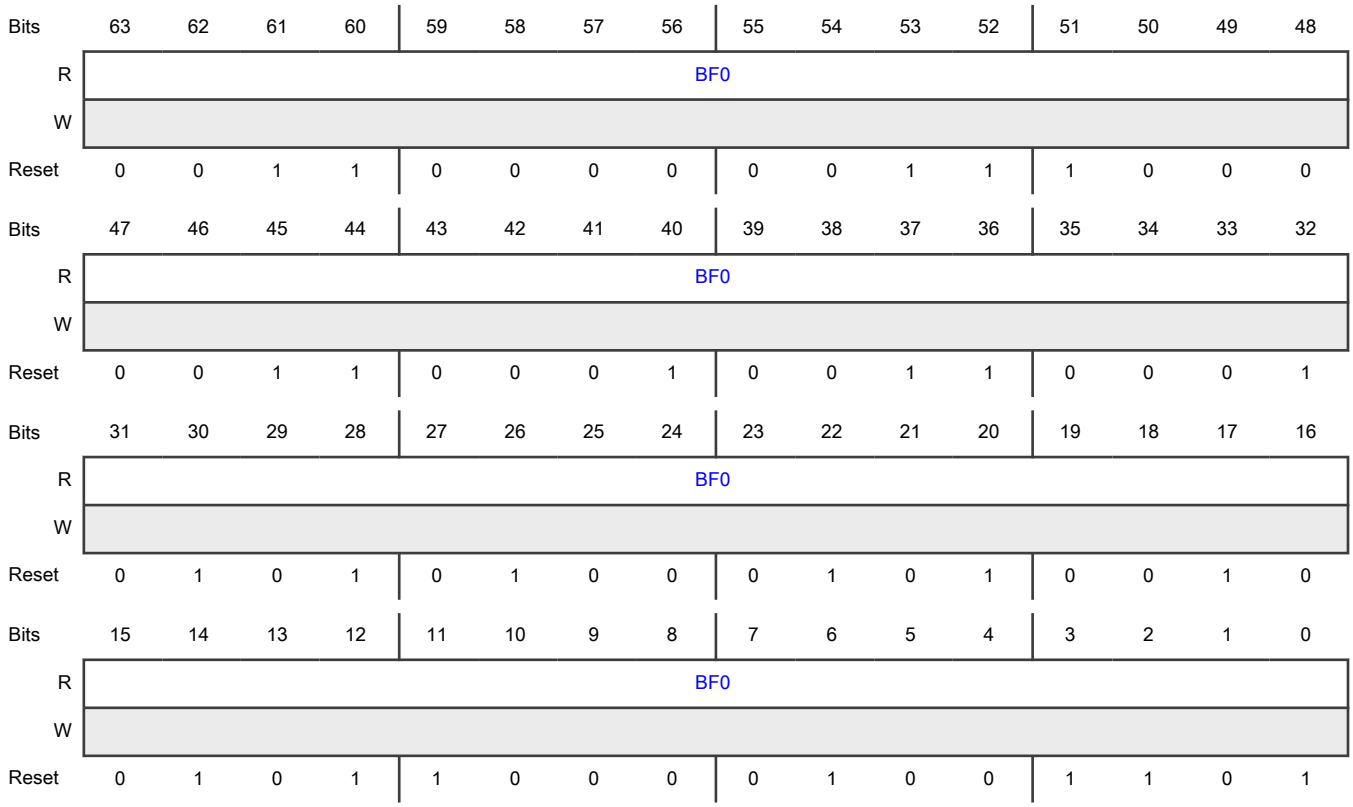
55.6.1.153 Register Product ID IP Core (PRODUCT_ID_IP_CORE)

Offset

Register	Offset
PRODUCT_ID_IP_CORE	E00h

Function
ESC Specific Register

Diagram



Fields

Field	Function
63-0 BF0	Product ID Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset value for bit depends on the configuration.

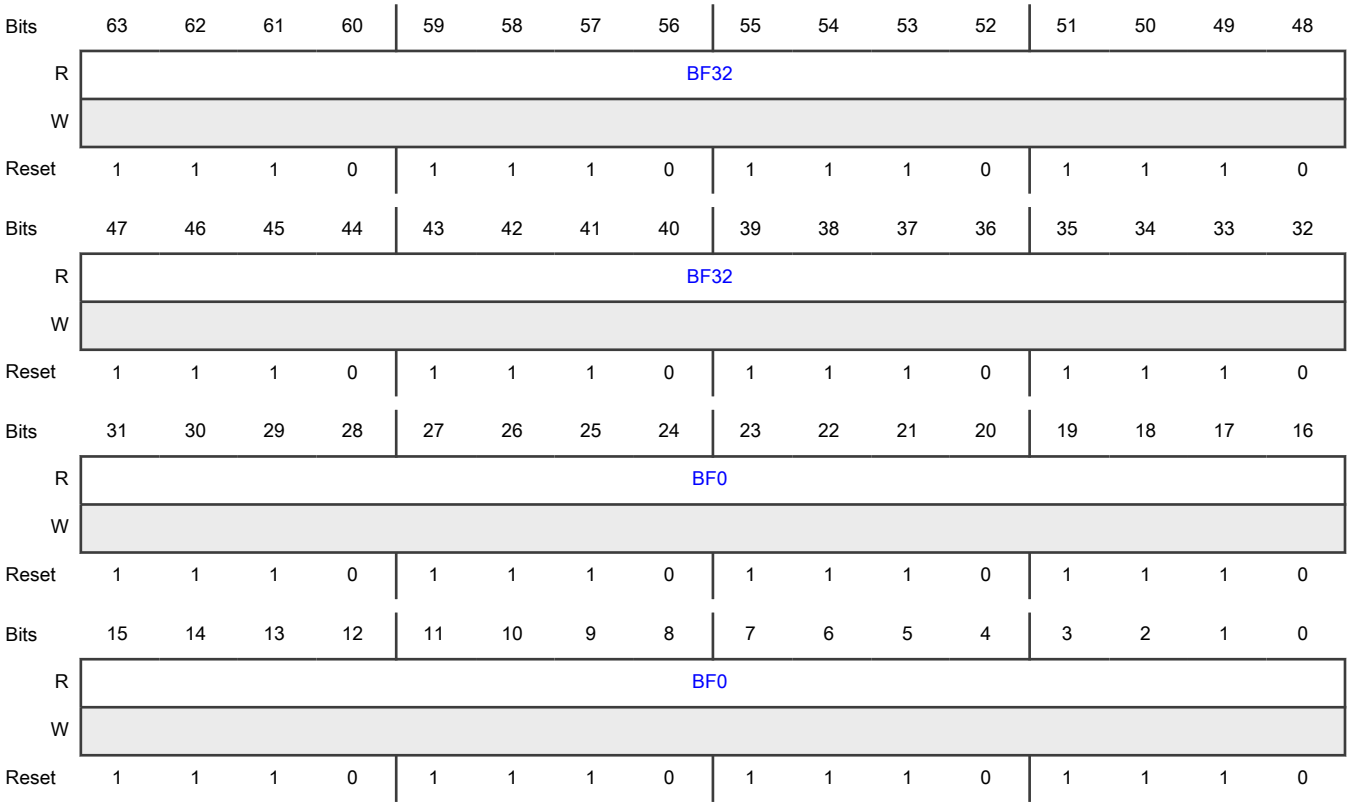
55.6.1.154 Register Vendor ID IP Core (VENDOR_ID_IP_CORE)

Offset

Register	Offset
VENDOR_ID_IP_CORE	E08h

Function
ESC Specific Register

Diagram



Fields

Field	Function
63-32 BF32	Reserved Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset for bit depends on license file
31-0 BF0	Vendor ID [23:0] Company [31:24] Department NOTE: Test Vendor IDs have [31:28]=0xE Bit field access for ECAT: r/- Bit field access for PDI: r/- Reset for bit depends on license file

55.6.1.155 Register General Purpose Outputs (GENERAL_PURPOSE_OUTPUTS)

Offset

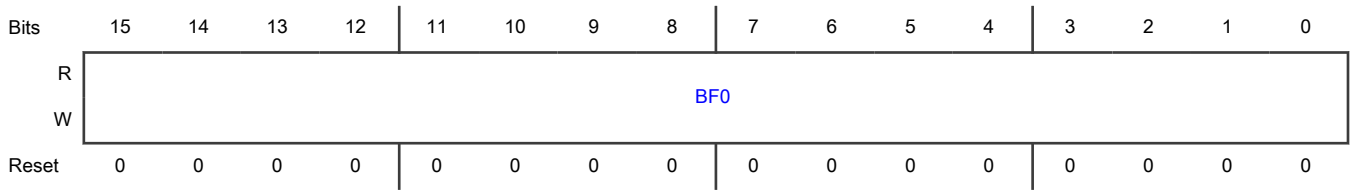
Register	Offset
GENERAL_PURPOSE_OUTPUTS	F10h

Function

NOTE

Register size depends on PDI setting and/or device configuration

Diagram



Fields

Field	Function
15-0 BF0	General Purpose Output Data Bit field access for ECAT: r/w Bit field access for PDI: r/w

55.6.1.156 Register General Purpose Inputs (GENERAL_PURPOSE_INPUTS)

Offset

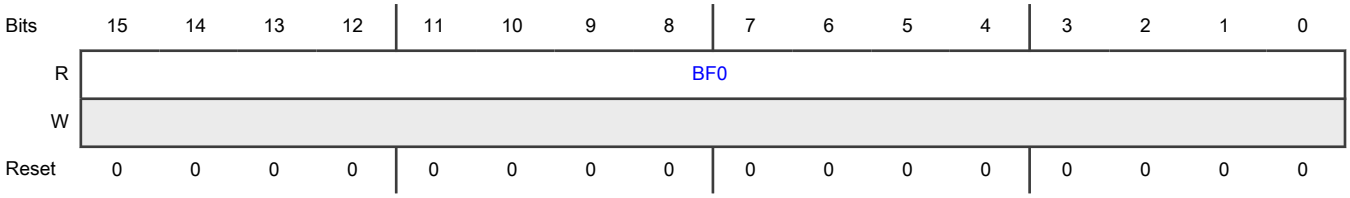
Register	Offset
GENERAL_PURPOSE_INPUTS	F18h

Function

NOTE

Register size depends on PDI setting and/or device configuration

Diagram



Fields

Field	Function
15-0 BF0	General Purpose Input Data Bit field access for ECAT: r/- Bit field access for PDI: r/-

Chapter 56
Universal Serial Bus Controller (USB)

56.1 Chip-specific USB information

Table 830. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

56.2 Overview

USB controller provides high-performance USB functionality that conforms to the *Universal Serial Bus Specification*, Rev. 2.0.

NOTE

The document uses the term "OTG" throughout. USB core hardware does not support the automatic OTG function by hardware. It is a dual-role (Host and Device mode) core, which you can implement in software with the hardware assist.

The USB controller consists of two independent USB controller cores: two On-The-Go (OTG) controller cores. Each controller core supports UTMI interfaces according to its feature. See Features for more details. Both the controller cores are single-port cores. For the OTG cores, there is only one port. The port can be used as either a downstream or an upstream port.

56.2.1 Block diagram

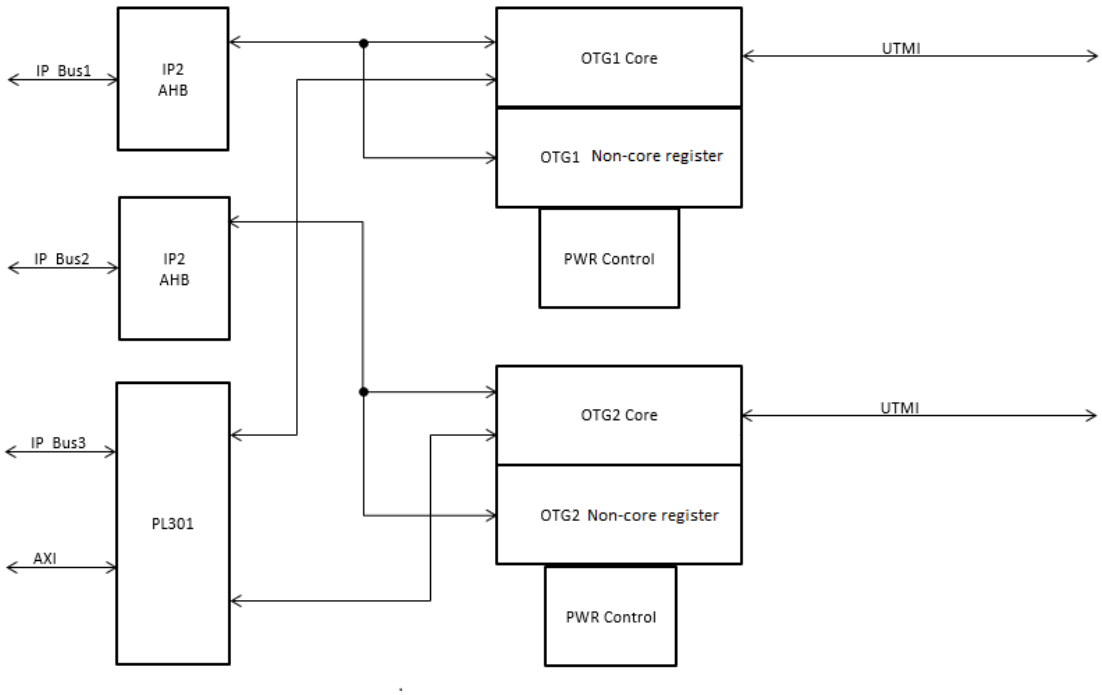


Figure 398. USB block diagram

56.2.2 Features

There are two USB 2.0 controller cores in this chip:

- Controller Core 0 is also named 'OTG1 Core'; its connected port is named 'OTG1 port'.
- Controller Core 1 is also named 'OTG2 Core'; its connected port is named 'OTG2 port'.
- Supports USB 2.0 Controller Core 0
  - High-speed/full-speed/low-speed (HS/FS/LS) OTG core
  - HS/FS/LS UTMI compliant interface
  - HS, FS, and LS operation in Host mode (with UTMI transceiver)
  - HS and FS operation in Peripheral mode (with UTMI transceiver)
  - OTG signaling, session request protocol, and host negotiation protocol
  - 8 bidirectional endpoints
  - Support charger detection
- USB 2.0 Controller Core 1
  - HS/FS/LS OTG core
  - HS/FS/LS UTMI compliant interface
  - HS, FS, and LS operation in Host mode (with UTMI transceiver)
  - HS and FS operation in Peripheral mode (with UTMI transceiver)

  - Hardware support for OTG signaling, session request protocol, and host negotiation protocol
  - 8 bidirectional endpoints
- Low-Power mode with local and remote wake-up capability
- Embedded DMA controller in each core

56.3 Functional description

These sections describe the functionality of the various building blocks of USB.

56.3.1 Submodule sections

56.3.1.1 USB 2.0 controller core 0

USB 2.0 Controller 0 is an instantiation of an EHCI-compatible core that supports HS, FS, and LS operation.

In Host mode, this controller core supports HS, FS, and LS operation. In Device mode, it supports HS and FS operation.

56.3.1.1.1 Host mode

The controller supports direct connection of an HS, FS, or LS device with on-chip UTMI transceiver. The system does not have a separate transaction translator block. The DMA and protocol engine blocks include the transaction translator function, which is normally associated with an USB 2.0 HS hub to support connection to FS and LS devices.

56.3.1.1.2 Peripheral (device) mode

- 8 bidirectional endpoints
- HS or FS operation
- Support of HNP and SRP
- Remote wake-up capability

56.3.1.2 USB 2.0 controller core 1

USB 2.0 Controller Core 1 is an instantiation of EHCI-compatible core which supports HS/FS/LS operation USB 2.0 Controller Core 1 is identical as Core 0.

In Host mode, this controller core supports HS, FS, and LS operation. In Device mode, it supports HS and FS operation.

56.3.2 Operation sections

56.3.2.1 Host operational model

The general operational model is for the enhanced interface host controller hardware and enhanced interface host controller driver (a general term for system software). A separate section discusses each significant operational feature of the EHCI HC. Each section presents the operational model requirements for the host controller hardware. You can also find recommended system software operational models presented for features where appropriate.

56.3.2.1.1 Host controller initialization

After initial power-on or HCReset (hardware or through HCReset field in USB Command (USBCMD), all of the operational registers are at their default values. After a hardware reset, only the operational registers not contained in the Auxiliary power well are at their default values.

The following table describes the default values of operational registers.

Table 831. Default values of operational register space

Operational register	Default value (after reset)
USB Command (USBCMD)	0008_0000h (0008_0B00h, if *Asynchronous Schedule Park Capability is 1*)
USB Status (USBSTS)	0000_1000h
Interrupt Enable (USBINTR)	0000_0000h
USB Frame Index (FRINDEX)	0000_0000h
Frame List Base Address (PERIODICLISTBASE)	Undefined
Next Asynch. Address (ASYNCLISTADDR)	Undefined
Configure Flag (CONFIGFLAG)	0000_0000h
Port Status & Control (PORTSC1)	0000_2000h (write 1 to w/*PPC*); 0000_3000h (write 0 to w/*PPC*)

To initialize the host controller, perform the following steps:

- Write the appropriate value to Interrupt Enable (USBINTR) to enable the appropriate interrupts.
- Write the base address of the Periodic Frame List to Frame List Base Address (PERIODICLISTBASE). If no work items are in the periodic schedule, all elements of the periodic frame list should have their T-Bits set to one.
- Write USB Command (USBCMD) to set the desired interrupt threshold, frame list size (if applicable) and turn the host controller ON through setting USBCMD[RS].

At this point, the host controller is up and running and the port registers begin reporting device connects, and so on. System software can enumerate a port through the reset process (where the port is in the enabled state). At this point, the port is active with SOFs occurring down the enabled ports, but the schedules have not enabled. To communicate with devices through the asynchronous schedule, you must write Next Asynch. Address (ASYNCLISTADDR) with the address of a control or bulk queue head. Then enable the asynchronous schedule by writing 1 to USBCMD[ASE] in USBCMD. To communicate with devices through the periodic schedule, system software must enable the periodic schedule by writing 1 to USBCMD[PSE] in USBCMD.

NOTE

It is possible to turn on the schedules before resetting the first port (and enabling).

When writing USBCMD, system software must ensure preserving the appropriate fields, depending on the intended operation.

56.3.2.1.2 Port routing and control

The EHCI specification defines that a USB 2.0 host controller (HC) consists of one HS host controller, which implements the EHCI programming interface and 0 to N USB 1.1 companion HCs. Companion host controllers (CHCs) may be implementations of either Universal or Open host controller specifications. Use this configuration to deliver the required full USB 2.0-defined port capability; for example, LS, FS, and HS capability for every port.

NOTE

The USB controllers neither require nor support companion controllers to support FS and LS device. USB controllers support FA and LS devices by emulating HS hub functionality. Therefore, there is no port routing present in the controller. See Embedded transaction translator function for more information.

The following figure illustrates a simple block diagram of the port routing logic and its relationship to the high-speed and CHCs within an USB 2.0 HC.

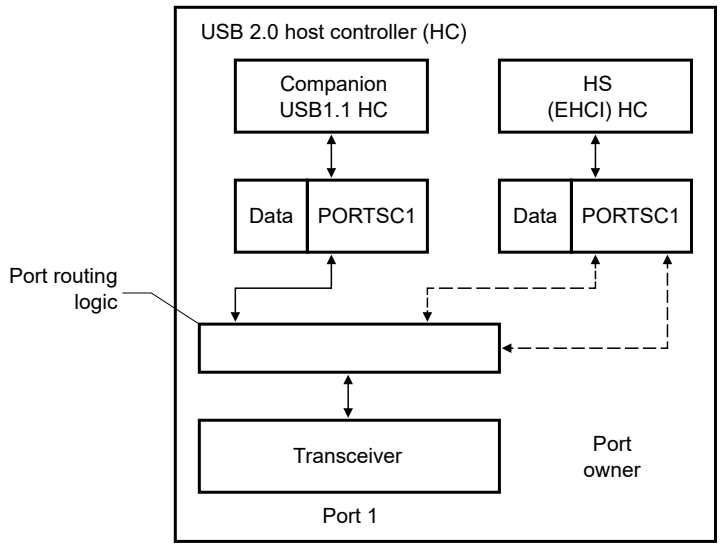


Figure 399. Example USB 2.0 Host Controller Port Routing Block Diagram

There exists one transceiver per physical port. Each host controller block has its own port status and control registers. The EHCI controller has port status and control registers for every port. Each CHC has only the port control and status registers to operate. Either the EHCI HC or one companion HC controls each transceiver. Routing logic lies among the transceiver, the port status and control registers.[6]

You can control the port routing logic from signals originating in the EHCI HC. The EHCI HC has a global routing policy control field and per-port ownership control fields. CONFIGFLAG[CF] is the global routing policy control. At power on or reset, the default routing policy follows the companion controllers (if they exist). If the system does not include a driver for the EHCI HC and the HC includes companion controllers, then the ports still work in FS and LS mode (assuming the system includes a driver for the companion controllers). In general, when the EHCI owns the ports, the CHCs' port registers do not see a connect indication from the transceiver. Similarly, when a CHC owns a port, the EHCI controller's port registers do not see a connect indication from the transceiver. The following sections describe the details on the rules for the port routing logic. If the implementation includes companion controllers, implement the USB 2.0 HC as a multi-function PCI device. The CHCs' function numbers must be less than the EHCI HC function number. The EHCI HC must be a larger function number with respect to the CHCs associated with this EHCI HC. If a PCI device implementation contains only an EHCI controller (that is, no companion controllers or other PCI functions), then the EHCI HC must function as 0, in accordance with the PCI Specification. HCSPARAMS[N_CC] in Host Controller Structural Parameters (HCSPARAMS) indicates whether the controller implementation includes CHCs. When HCSPARAMS[N_CC] has a non-zero value, there exists CHCs. If HCSPARAMS[N_CC] has a 0 value, then the HC implementation does not include CHCs. While exposing the HC root ports to attachment of FS or LS devices, the ports always fail the HS chirp during reset without enabling the ports. System software can notify the user of the illegal condition. This type of implementation requires connection of an USB 2.0 hub to a root port to provide FS and LS device connectivity.

System software uses information in the HC capability registers to determine routing of the ports to the CHCs. See HCSPARAMS for more information.

56.3.2.1.2.1 Port routing control through EHCI CONFIGFLAG[CF]

The USB 2.0 host controller has either a single CHC or an EHCI HC for each port. Two mechanisms in the EHCI HC control the port routing logic: a host controller global flag and per-port control. Use Configured Flag (CF) to globally set the policy of the routing logic. Each port register has a Port Owner control which allows the EHCI Driver to explicitly control the routing of individual ports. Whenever CF transitions from 0 to 1 (this transition is only available under program control) the port routing unconditionally routes all of the port registers to the EHCI HC (all Port Owner bits go to 0). While CF is 1, the EHCI Driver controls individual ports' routing through Port Owner control. Likewise, whenever CF transitions from 1 to 0 (as a result of Aux

[6] It is not recommended to implement the routing logic in the 480 MHz clock domain of the transceiver.

power application, HCRESET, or software writing 0 to CF-bit), the port routing unconditionally routes all of the port registers to the appropriate companion HC. The default value for the EHCI HC's CF bit (after Aux power application or HCRESET) is 0.

The *view* of the port depends on the current owner. A Universal or Open CHC will see port register bits consistent with the appropriate specification. Port bit definitions that are required for EHCI HCs are not visible to CHCs.

The following table summarizes the default routing for all the ports, based on the value of the EHCI HC's CONFIGFLAG[CF].

Table 832. Default port routing depending on EHCI HC CONFIGFLAG[CF]

HS CF	Default port ownership	Explanation
0B	Companion HCs	The companion host controllers own the ports. The system supports only FS and LS devices in the system. The exact port assignments are implementation dependent. The ports behave only as FS and LS ports in this configuration
1B	EHCI HC	The EHCI HC has default ownership over all of the ports. The routing logic inhibits device connect events from reaching the companion HCs' port status and control registers when the port owner is EHCI HC. The EHCI HC has access to the additional port status and control bits defined in this specification (see Port Status & Control (PORTSC1)). The EHCI HC can temporarily release control of the port to a companion HC by changing the *PortOwner* in PORTSC1 to 1.

56.3.2.1.2.2 Port routing control through portOwner and disconnect event

Manipulating the port routing through the CF-bit is an extreme process and not intended to be used during normal operation. The normal mode of port ownership transferal is on the granularity of individual ports using the Port Owner bit in the EHCI HC's USB_PORTSC1 register (for hand-offs from EHCI to companion host controllers). Individual port ownership is returned to the EHCI controller when the port registers a device disconnect. When the disconnect is detected, the port routing logic immediately returns the port ownership to the EHCI controller. The companion host controller port register detects the device disconnect and operates normally.

Under normal operating conditions (assuming all HC drivers loaded and operational and the EHCI *CF-bit* is set to one), the typical port enumeration sequence proceeds as illustrated below:

- Initial condition is that EHCI is port owner. A device is connected causing the port to detect a connect, set the port connect change bit and issue a port-change interrupt (if enabled).
- EHCI Driver identifies the port with the new connect change bit asserted and sends a change report to the hub driver. Hub driver issues a GetPortStatus() request and identifies the connect change. It then issues a request to clear the connect change, followed by a request to reset and enable the port.
- When the EHCI Driver receives the request to reset and enable the port, it first checks the value reported by the LineStatus bits in the USB_PORTSC1 register. If they indicate the attached device is a full-speed device (for example, D+ is asserted), then the EHCI Driver sets the PortReset control bit to one (and sets the PortEnable bit to zero) which begins the reset-process. Software times the duration of the reset, then terminates reset signaling by writing zero to the port reset bit. The reset process is actually complete when software reads zero in the PortReset bit. The EHCI Driver checks the PortOwner bit in the USB_PORTSC1 register. If set to one, the connected device is a high-speed device and EHCI Driver (root hub emulator) issues a change report to the hub driver and the hub driver continues to enumerate the attached device.
- At the time the EHCI Driver receives the port reset and enable request the LineStatus bits might indicate a low-speed device. Additionally, when the port reset process is complete, the PortEnable field may indicate that a full-speed device is attached. In either case the EHCI driver sets the PortOwner bit in the USB_PORTSC1 register to one to release port ownership to a companion host controller.
- When the EHCI Driver sets PortOwner bit to one, the port routing logic makes the connection state of the transceiver available to the companion host controller port register and removes the connection state from the EHCI HC port. The

EHCI USB_PORTSC1 register observes and reports a disconnect event through the disconnect change bit. The EHCI Driver detects the connection status change (either by polling or by port change interrupt) and then sends a change report to the hub driver. When the hub driver requests that port-state, the EHCI Driver responds with a reset complete change set to one, a connect change set to one and a connect status set to zero. This information is derived directly from the EHCI port register. This allows the hub driver to assume the device was disconnected during reset. It acknowledges the change bits and wait for the next change event. While the EHCI controller does not own the port, it simply remains in a state where the port reports no device connected. The device-connect evaluation circuitry of the companion HC activates and detects the device, the companion Driver detects the connection and enumerates the port.

When a port is routed to a companion HC, it remains under the control of the companion HC until the device is disconnected from the root port (ignoring for now the scenario where EHCI's CF-bit transitions from1b to 0b). When a disconnect occurs, the disconnect event is detected by both the companion HC port control and the EHCI port ownership control. On the event, the port ownership is returned immediately to the EHCI controller. The companion HC stack detects the disconnect and acknowledges as it would in an ordinary standalone implementation. Subsequent connects is detected by the EHCI port register and the process repeats.

56.3.2.1.2.3 Example port routing state machine

The following figure illustrates an example of how to manage the port ownership. The following sections describe the entry conditions to each state.

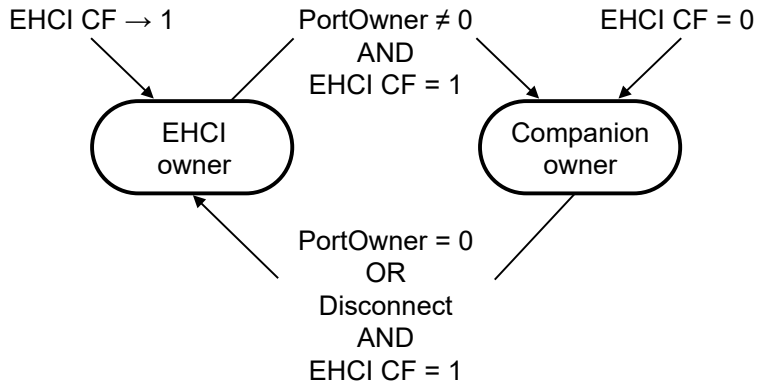


Figure 400. Port owner handoff state machine

56.3.2.1.2.3.1 EHCI HC owner

Entry to this state occurs when one of the following events occur:

- When EHCI HC's CONFIGFLAG[CF] in CONFIGFLAG transitions from 0 to 1. This signals the fact that the system has a host controller driver for the EHCI HC and that all ports in the USB 2.0 host controller must default route to the EHCI controller.
- When a companion HC owns the port and there is no connection of the device with the port. There is notification of the disconnect to the EHCI port routing control logic, and returns port routing to the EHCI controller. The connection state of the companion HC goes immediately to the disconnected state (with appropriate side effect to connect change, enable and enable change). The companion HC driver acknowledges the disconnect by setting the connect status change bit to 0. This allows the companion HC's driver to interact with the port completely through the disconnect process.
- When system software writes 0 to PortOwner in PORTSC1. This allows you to take ownership of a port from a companion host controller. When this occurs, the routing logic to the companion HC effectively signals a disconnect to the companion HC's port status and control register.

56.3.2.1.2.3.2 Companion HC owner

Entry to this state occurs whenever one of the following events occur:

- When the PortOwner field transitions from zero to one.
- When the HS-mode HC's Configure Flag (CF) is equal to zero.

On entry to this state, the routing logic allows the companion HC port register to detect a device connect. Normal port enumeration proceeds.

56.3.2.1.2.4 Port power

The Port Power Control (PPC) bit in the USB_HCSPARAMS register indicates whether the USB 2.0 host controller has port power control (see HCSPARAMS register). When this bit is zero, then the host controller does not support software control of port power switches. When in this configuration, the port power is always available and the companion host controllers must implement functionality consistent with port power always on. When the *PPC* bit is one, then the host controller implementation includes port power switches. Each available switch has an output enable, which is referred to in this discussion as PortPowerOutputEnable (PPE). PPE is controlled based on the state of the combination bits PPC bit, EHCI Configured (CF)-bit and individual Port Power (PP) bits.

The following table describes the summary behavioral model.

Table 833. Port Power Enable Control Rules

CF	CHC[1] (PP)	EHC[2] (PP)	Owner	PPE[3]	Description
0	0	X	CHC	0	When the EHCI controller is not configured, the port is owned by the companion host controller. When the companion HC's port power select is off, then the port power is off.
0	1	X	CHC	1	Similar to previous entry. When the companion HC's port power select is on, then the port power is on.
1	0	0	CHC	0	Port owner has port power turned off, the power to port is off.
1	0	0	EHC	0	Port owner has port power turned off, the power to port is off.
1	0	1	EHC	1	Port owner has port power on, so power to port is on.
1	0	1	CHC	1	If either HC has port power turned on, the power to the port is on.
1	1	0	EHC	1	If either HC has port power turned on, the power to the port is on.
1	1	0	CHC	1	Port owner has port power on, so power to port is on.
1	1	1	CHC	1	Port owner has port power on, so power to port is on.
1	1	1	EHC	1	Port owner has port power on, so power to port is on.

1. CHC (Companion Host Controller).

2. EHC (EHCI Host Controller).
3. PPE (Port Power Enable). This bit actually turns on the port power switch (if one exists).

56.3.2.1.2.5 Port reporting over-current

Host controllers are by definition power providers on USB. Whether the ports are considered high- or low-powered is a platform implementation issue. Each EHCI USB_PORTSC1 register has an over-current status and over-current change bit. The functionality of these bits are specified in the USB Specification Revision 2.0.

The over current detection and limiting logic usually resides outside the host controller logic. This logic may be associated with one or more ports. When this logic detects an over-current condition it is made available to both the companion and EHCI ports. The effect of an over-current status on a companion host controller port is beyond the scope of this document.

The over-current condition effects the following bits in the USB_PORTSC1 register on the EHCI port:

- Over-current Active bits are set to one. When the over-current condition goes away, the Over-current Active bit transitions from one to zero.
- Over-current Change bits are set to one. On every transition of the Over-current Active bit the host controller sets the Over-current Change bit to one. Software sets the Over-current Change bit to zero by writing one to this bit.
- Port Enabled/Disabled bit is set to zero. When this change bit gets set to one, then the Port Change Detect bit in the USB_USBSTS register is set to one.
- Port Power (PP) bits may optionally be set to zero. There is no requirement in USB that a power provider shut off power in an over current condition. It is sufficient to limit the current and leave power applied. When the Over-current Change bit transitions from zero to one, the host controller also sets the Port Change Detect bit in the USB_USBSTS register to one. In addition, if the Port Change Interrupt Enable bit in the USB_USBINTR register is one, then the host controller issues an interrupt to the system. Refer to Table 834 for summary behavior for over-current detection when the host controller is halted (suspended from a device component point of view).

56.3.2.1.3 Suspend/resume-host operational model

The EHCI HC provides an equivalent suspend and resume model as that defined for individual ports in a USB 2.0 Hub. Control mechanisms are provided to allow system software to suspend and resume individual ports. The mechanisms allow the individual ports to be resumed completely through software initiation. Other control mechanisms are provided to parameterize the host controller's response (or sensitivity) to external resume events. In this discussion, host-initiated, or software initiated resumes are called Resume Events/Actions. Bus-initiated resume events are called wake-up events. The classes of wake-up events are:

- Remote-wake-up enabled device asserts resume signaling. In similar kind to USB 2.0 Hubs, EHCI controllers must always respond to explicit device resume signaling and wake-up the system (if necessary).
- Port connect and disconnect and over-current events. Sensitivity to these events can be turned on or off by using the per-port control bits in the USB_PORTSC1 registers.

Selective suspend is a feature supported by every USB_PORTSC1 register. It is used to place specific ports into a suspend mode. This feature is used as a functional component for implementing the appropriate power management policy implemented in a particular operating system. When system software intends to suspend the entire bus, it should selectively suspend all enabled ports, then shut off the host controller by setting the Run/Stop bit in the USB_USBCMD register to zero. The EHCI sub-block can then be placed into a lower device state through the PCI power management interface (see Appendix A, Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 0.95, November 2000, Intel Corporation. http://www.intel.com).

When a wake event occurs, the system resumes operation and system software eventually set the Run/Stop bit to one and resume the suspended ports. Software must not set the Run/Stop bit to one until it is confirmed that the clock to the host controller is stable. This is usually confirmed in a system implementation in that all of the clocks in the system are stable before the Arm platform is restarted. So, by definition, if software is running, clocks in the system are stable and the Run/Stop bit in the USB_USBCMD register can be set to one. Minimum system software delays are also defined in the PCI Power Management Specification. Refer to PCI Power Management Specification for more information.

56.3.2.1.3.1 Port suspend/resume

System software places individual ports into suspend mode by writing one into the appropriate USB_PORTSC1 Suspend bit. You must only set the Suspend bit when the port is in the enabled state (Port Enabled bit is 1) and the EHCI is the port owner (PortOwner bit is 0).

The host controller may evaluate the Suspend bit immediately or wait until a micro-frame or frame boundary occurs. If evaluated immediately, the port is not suspended until the current transaction (if one is executing) completes. Therefore, there may be several micro-frames of activity on the port until the host controller evaluates the Suspend bit. The host controller must evaluate the Suspend bit at least every frame boundary.

System software can initiate a resume on a selectively suspended port by writing one to the Force Port Resume bit. Software should not attempt to resume a port unless the port reports that it is in the suspended state (see USB_nPORTSC1). If system software sets Force Port Resume bit to one when the port is not in the suspended state, the resulting behavior is undefined. In order to assure proper USB device operation, software must wait for at least 10 ms after a port indicates that it is suspended (Suspend bit is one) before initiating a port resume through the Force Port Resume bit. When Force Port Resume bit is one, the host controller sends resume signaling down the port. System software times the duration of the resume (nominally 20 ms) then sets the Force Port Resume bit to zero. When the host controller receives the write to transition Force Port Resume to zero, it completes the resume sequence as defined in the USB specification, and sets both the Force Port Resume and Suspend bits to zero. Software-initiated port resumes do not affect the Port Change Detect bit in the USB_USBSTS register nor do they cause an interrupt if the Port Change Interrupt Enable bit in the USB_USBINTR register is one. An external USB event may also initiate a resume. The wake events are defined above. When a wake event occurs on a suspended port, the resume signaling is detected by the port and the resume is reflected downstream within 100 µsec. The port's Force Port Resume bit is set to one and the Port Change Detect bit in the USB_USBSTS register is set to one. If the Port Change Interrupt Enable bit in the USB_USBINTR register is one the host controller issues a hardware interrupt.

System software observes the resume event on the port, delays a port resume time (nominally 20 ms), then terminates the resume sequence by writing zero to the Force Port Resume bit in the port. The host controller receives the write of zero to Force Port Resume, terminates the resume sequence and sets Force Port Resume and Suspend port bits to zero. Software can determine that the port is enabled (not suspended) by sampling the USB_PORTSC1 register and observing that the Suspend and Force Port Resume bits are zero. Software must ensure that the host controller is running (that is HCHalted bit in the USB_USBSTS register is zero), before terminating a resume by writing zero to a port's Force Port Resume bit. If HCHalted is one when Force Port Resume is set to zero, then SOFs do not occur down the enabled port and the device returns to suspend mode in a maximum of 10 msec.

The table below summarizes the wake-up events. Whenever a resume event is detected, the Port Change Detect bit in the USB_USBSTS register is set to one. If the Port Change Interrupt Enable bit is one in the USB_USBINTR register, the host controller generates an interrupt on the resume event. Software acknowledges the resume event interrupt by clearing the Port Change Detect status bit in the USB_USBSTS register.

Table 834. Behavior During Wake-up Events

Port Status and Signaling Type	Signaled Port Response	Device State	
		D0	Not D0
Port disabled, resume K-State received	No Effect	N/A	N/A
Port suspended, resume K-State received	Resume reflected downstream on signaled port. Force Port Resume status bit in USB_PORTSC1 register is set to one. Port Change Detect bit in USB_USBSTS register set to one.	[1], [2]	[2]
Port is enabled, disabled or suspended, and the port's WKDSCNNT_E bit is one. A disconnect is detected.	Depending in the initial port state, the USB_ PORTSC1 Connected Enable status bits are set to zero, and the Connect Change status bit is set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [2]	[2]

Table continues on the next page...

Table 834. Behavior During Wake-up Events (continued)

Port is enabled, disabled or suspended, and the port's WKDSCNNT_E bit is zero. A disconnect is detected.	Depending on the initial port state, the USB_PORTSC1 Connect and Enable status bits are set to zero, and the Connect Change status bit is set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]
Port is not connected and the port's WKCNNT_E bit is one. A connect is detected.	USB_PORTSC1 Connect Status and Connect Status Change bits are set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [2]	[2]
Port is not connected and the port's WKCNNT_E bit is zero. A connect is detected.	USB_PORTSC1 Connect Status and Connect Status Change bits are set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]
Port is connected and the port's WKOC_E bit is one. An over-current condition occurs.	USB_PORTSC1 Over-current Active, Over-current Change bits are set to one. If Port Enable/Disable bit is one, it is set to zero. Port Change Detect bit in the USB_USBSTS register is set to one	[1], [2]	[2]
Port is connected and the port's WKOC_E bit is zero. An over-current condition occurs.	USB_PORTSC1 Over-current Active, Over-current Change bits are set to one. If Port Enable/Disable bit is one, it is set to zero. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]

[1] Hardware interrupt issued if Port Change Interrupt Enable bit in the USB_USBINTR register is one.

[2] PME# asserted if enabled (Note: PME Status must always be set to one).

[3] PME# not asserted.

56.3.2.1.4 Schedule traversal rules

The host controller executes transactions for devices using a simple, shared-memory schedule. The schedule is comprised of a few data structures, organized into two distinct lists. The data structures are designed to provide the maximum flexibility required by USB, minimize memory traffic and hardware / software complexity.

System software maintains two schedules for the host controller: a periodic schedule and an asynchronous schedule. The root of the periodic schedule is the USB_PERIODICLISTBASE register (see USB_nPERIODICLISTBASE) DEVICEADDR. The USB_PERIODICLISTBASE register is the physical memory base address of the periodic frame list. The periodic frame list is an array of physical memory pointers. The objects referenced from the frame list must be valid schedule data structures as defined in Host data structures. In each micro-frame, if the periodic schedule is enabled (see Periodic scheduling threshold) then the host controller must execute from the periodic schedule before executing from the asynchronous schedule. It only executes from the asynchronous schedule after it encounters the end of the periodic schedule. The host controller traverses the periodic schedule by constructing an array offset reference from the USB_PERIODICLISTBASE and the USB_FRINDEX registers (see the following figure). It fetches the element and begins traversing the graph of linked schedule data structures.

The end of the periodic schedule is identified by a next link pointer of a schedule data structure having its T-bit set to one. When the host controller encounters a T-Bit set to one during a horizontal traversal of the periodic list, it interprets this as an End-Of-Periodic-List mark. This causes the host controller to cease working on the periodic schedule and transitions immediately to traversing the asynchronous schedule. After the transition, the host controller executes from the asynchronous schedule until the end of the micro-frame.

The following figure illustrates the derivation of pointer into frame list array.

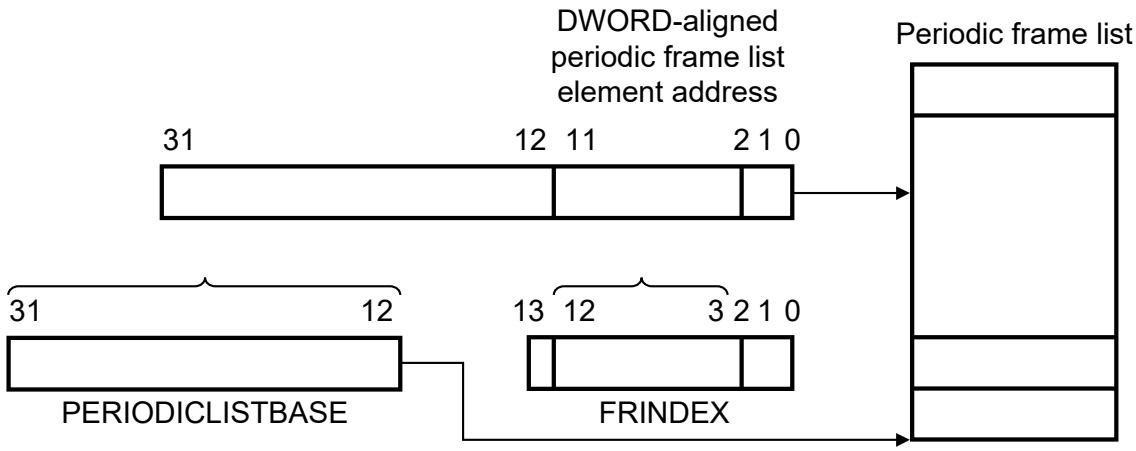


Figure 401. Derivation of Pointer into Frame List Array

When the host controller determines that it is the time to execute from the asynchronous list, it uses the operational register USB_ASYNCLISTADDR to access the asynchronous schedule, see the figure below.

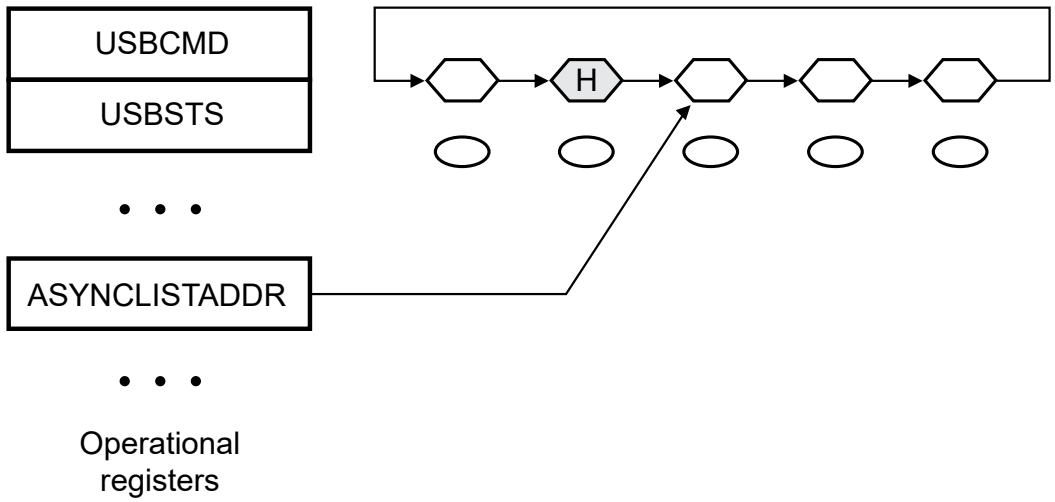


Figure 402. General Format of Asynchronous Schedule List

The USB_ASYNCLISTADDR register contains a physical memory pointer to the next queue head. When the host controller makes a transition to executing the asynchronous schedule, it begins by reading the queue head referenced by the USB_ASYNCLISTADDR register. Software must set queue head horizontal pointer T-bits to zero for queue heads in the asynchronous schedule. See Asynchronous schedule for complete operational details.

56.3.2.1.4.1 Example - preserving micro-frame integrity

One of the requirements of a USB host controller is to maintain Frame Integrity. This means that the HC must preserve the micro-frame boundaries. For example, SOF packets must be generated on time (within the specified allowable jitter), and High-speed EOF1,2 thresholds must be enforced. The end of micro-frame timing points EOF1 and EOF2 are clearly defined in the USB Specification Revision 2.0. One implication of this responsibility is that the HC must ensure that it does not start transactions that do not complete before the end of the micro-frame. More precisely, no transactions should be started by the host controller, which do not complete in their entirety before the EOF1 point. In order to enforce this rule, the host controller must check each transaction before it starts to ensure that it completes before the end of the micro-frame.

So, what exactly needs to be involved in this check? Fundamentally, the transaction data payload, plus bit stuffing, plus transaction overhead must be taken into consideration. It is possible to be extremely accurate on how much time the next transaction takes. Take OUTs for an example. The host controller must fetch all of the OUT data from memory in order to send it onto the USB bus. A host controller implementation could pre-fetch all of the OUT data, and pre-compute the actual number of bits in the token and data packets. In addition, the system knows the depth of the target endpoint, so it could closely estimate turnaround time for handshake. In addition, the host controller knows the size of a handshake packet. Pre-computing effects of bit stuffing and summing up the other overhead numbers can allow the host controller to know exactly whether there is enough bus time, before EOF1 to complete the OUT transaction. To accomplish this particular approach takes an inordinate amount of time and hardware complexity.

The alternative is to make a reasonable guess whether the next transaction can be started. An example approximation algorithm is described below. This example algorithm relies on the EHCI policy that periodic transactions are scheduled first in the micro-frame. It is a reasonable assumption that software never over-commits the micro-frame to periodic transactions greater than the specification allowable 80%. In the available remaining 20% bandwidth, the host controller has some ability (in this example) to decide whether or not to execute a transaction. The result of this algorithm is that sometimes, under some circumstances a transaction is not executed that could have been executed. However, under all circumstances, a transaction is never started unless there is enough time in the frame to complete the transaction.

56.3.2.1.4.1.1 Transaction fit - a best-fit approximation algorithm

A curve is calculated which represents the latest start time for every packet size, at which software schedules the start of a periodic transaction. This curve is the 80 % bandwidth curve. Another curve is calculated which is the absolute, latest permitted start time for every packet size. This curve represents the absolute latest time, that a transaction of each packet size can be started and completed, in the micro-frame. A plot of these two curves is illustrated in Figure 403. The plot Y-axis represents the number of byte-times left in a frame.

The space between the 80 % and the Last Start plots is bandwidth reclamation area. In this algorithm, the host controller may skip transactions during this time if it is prudent.

The Best-Fit Approximation method plots a function ($f(x)$) between the 80 % and Last Start curves. The function $f(x)$ adds a constant to every transaction's maximum packet size and the result compared with the number of bytes left in the frame. The constant represents an approximation of the effects of bit stuffing and protocol overhead. The host controller starts transactions whose results land above the function curve. The host controller will not start transactions whose results land below the function curve.

The following figure illustrates the Best-Fit Approximation.

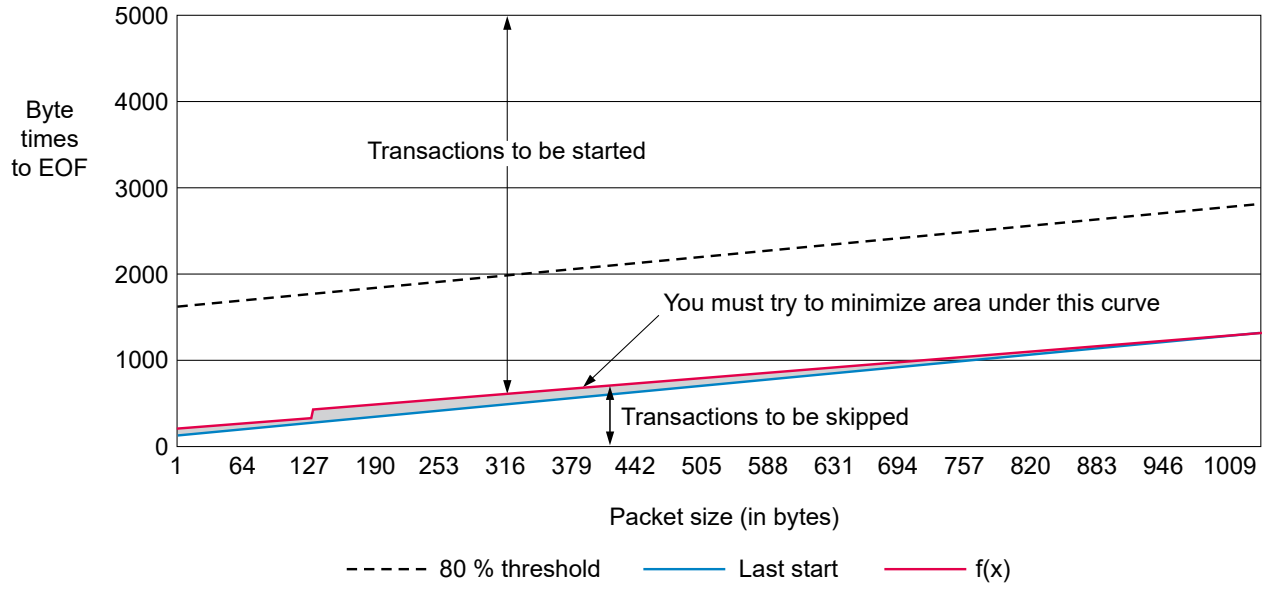


Figure 403. Best Fit Approximation

The LastStart line was calculated in this example to assume the absolute worst-case bus overhead per transaction. The particular transaction used is a start-split, zero-length OUT transaction with a handshake. Summaries of the component parts are listed in the table below. The component times were derived from the protocol timings defined in the USB Specification Revision 2.0.

Table 835. Example Worse-case Transaction Timing Components

Component	Bit time	Byte Time	Explanation
Split Token	76	9.5	Split token as defined in USB core specification. Includes sync, token, EOP, and so on.
Host 2 Host IPG	88	11	Number of bit times required between consecutive host packets.
Token	67	8.375	Token as defined in USB core specification. Includes sync, token, EOP, and so on.
Host 2 Host IPG	88	11	Token as defined in USB core specification. Includes sync, token, EOP, and so on.
Data Packet (0 data bytes)	66.7	8.34	Zero-length data packet. Includes sync, PID, crc16, EOP, and so on.
Turnaround time	721	90.125	Time for packet initiator (Host) to see the beginning of a response to a transmitted packet.
Handshake packet	48	6	Handshake packet as defined in USB core specification. Includes sync, PID, EOP, and so on.
		144	Total

The exact details of the function (f(x)) are up to the particular implementation. However, it should be obvious that the goal is to minimize the area under the curve between the approximation function and the Last Start curve, without dipping below the LastStart line, while at the same time keeping the check as simple as possible for hardware implementation. The f(x) in Figure 403 was constructed using the following pseudo-code test on each transaction size data point. This algorithm assumes that the host controller keeps track of the remaining bits in the frame.

```
Algorithm CheckTransactionWillFit (MaximumPacketSize, HC_BytesLeftInFrame)
Begin
Local Temp = MaximumPacketSize + 192
Local rvalue = TRUE
If MaximumPacketSize >= 128 then
     Temp += 128
End If
If Temp > HC_BytesLeftInFrame then
     Rvalue = FALSE
End If
Return rvalue
End
```

This algorithm takes two inputs, the current maximum packet size of the transaction and the hardware counter of the number of bytes left in the current micro-frame. It unconditionally adds a simple constant of 192 to the maximum packet size to account for a first-order effect of transaction overhead and bit stuffing. If the transaction size is greater than or equal to 128 bytes, then an additional constant of 128 is added to the running sum to account for the additional worst-case bit stuffing of payloads larger than 128. An inflection point was inserted at 128 because the *f*(*x*) plot was getting close to the LastStart line.

56.3.2.1.5 Periodic schedule frame boundaries vs bus frame boundaries

The USB Specification Revision 2.0 requires that the frame boundaries (SOF frame number changes) of the high-speed bus and the full- and low-speed bus(s) below USB 2.0 Hubs be strictly aligned. Super-imposed on this requirement is that USB 2.0 Hubs manage full- and low-speed transactions through a micro-frame pipeline (see start- (SS) and complete- (CS) splits illustrated in the following figure). A simple, direct projection of the frame boundary model into the host controller interface schedule architecture creates tension (complexity for both hardware and software) between the frame boundaries and the scheduling mechanisms required to service the full- and low-speed transaction translator periodic pipelines.

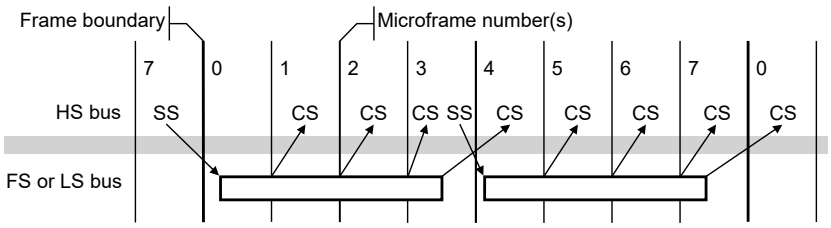


Figure 404. Frame Boundary Relationship between HS bus and FS/LS Bus

The simple projection, as the above figure illustrates, introduces frame-boundary wrap conditions for scheduling on both the beginning and end of a frame. In order to reduce the complexity for hardware and software, the host controller is required to implement one micro-frame phase shift for its view of frame boundaries. The phase shift eliminates the beginning of frame and frame-wrap scheduling boundary conditions.

The implementation of this phase shift requires that the host controller use one register value for accessing the periodic frame list and another value for the frame number value included in the SOF token. These two values are separate, but tightly coupled. The periodic frame list is accessed through the Frame List Index Register (USB_FRINDEX) documented in FRINDEX and initially illustrated in Schedule traversal rules. Bits FRINDEX[2:0], represent the micro-frame number. The SOF value is coupled to the value of FRINDEX[13:3]. Both FRINDEX[13:3] and the SOF value are increment based on FRINDEX[2:0]. It is required that the SOF value be delayed from the FRINDEX value by one micro-frame. The one micro-frame delay yields host controller periodic schedule and bus frame boundary relationship as illustrated in the following figure. This adjustment allows software to trivially schedule the periodic start and complete-split transactions for full-and low-speed periodic endpoints, using the natural alignment of the periodic schedule interface. The reasons for selecting this phase-shift are beyond the scope of this specification.

The following figure illustrates how periodic schedule data structures relate to schedule frame boundaries and bus frame boundaries. To aid the presentation, two terms are defined: The host controller's view of the 1 msec boundaries is called H-Frames. The high-speed bus's view of the 1 msec boundaries is called B-Frames.

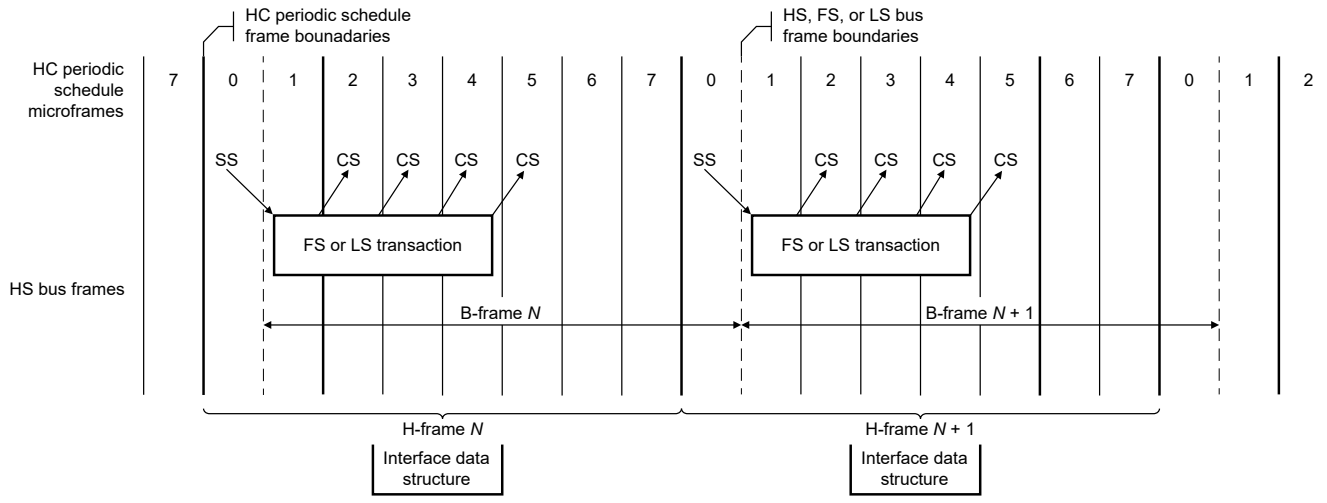


Figure 405. Relationship of Periodic Schedule Frame Boundaries to Bus Frame Boundaries

H-Frame boundaries for the host controller correspond to increments of FRINDEX[13:3]. Micro-frame numbers for the H-Frame are tracked by FRINDEX[2:0]. B-Frame boundaries are visible on the high-speed bus through changes in the SOF token's frame number. Micro-frame numbers on the high-speed bus are only derived from the SOF token's frame number (that is the high-speed bus sees eight SOFs with the same frame number value). H-Frames and B-Frames have the fixed relationship (that is B-Frames lag H-Frames by one micro-frame time) illustrated in the figure above. The host controller's periodic schedule is naturally aligned to H-Frames. Software schedules transactions for full- and low-speed periodic endpoints relative the H-Frames. The result is these transactions execute on the high-speed bus at exactly the right time for the USB 2.0 Hub periodic pipeline. As described in FRINDEX, the SOF Value can be implemented as a shadow register (in this example, called SOFV), which lags the FRINDEX register bits [13:3] by one micro-frame count. This lag behavior can be accomplished by incrementing FRINDEX[13:3] based on carry-out on the 7 to 0 increment of FRINDEX[2:0] and incrementing SOFV based on the transition of 0 to 1 of FRINDEX[2:0].

Software is allowed to write to FRINDEX. FRINDEX provides the requirements that software should adhere when writing a new value in FRINDEX.

The table below illustrates the required relationship between the value of FRINDEX and the value of SOFV.

Table 836. Operation of FRINDEX and SOFV (SOF Value Register)

Current			Next		
FRINDEX[F]	SOFV	FRINDEX[mF]	FRINDEX[F]	SOFV	FRINDEX[mF]
N	N	111b	N+1	N	000b
N+1	N	000b	N+1	N+1	001b
N+1	N+1	001b	N+1	N+1	010b
N+1	N+1	010b	N+1	N+1	011b
N+1	N+1	011b	N+1	N+1	100b
N+1	N+1	100b	N+1	N+1	101b
N+1	N+1	101b	N+1	N+1	110b
N+1	N+1	110b	N+1	N+1	111b

NOTE

Where [F] = [13:3]; [μF] = [2:0]

56.3.2.1.6 Periodic schedule

The periodic schedule traversal is enabled or disabled through the Periodic Schedule Enable bit in the USB_USBCMD register. If the Periodic Schedule Enable bit is set to zero, then the host controller simply does not try to access the periodic frame list through the USB_PERIODICLISTBASE register. Likewise, when the Periodic Schedule Enable bit is one, then the host controller does use the USB_PERIODICLISTBASE register to traverse the periodic schedule.

The host controller will not react to modifications to the Periodic Schedule Enable immediately. In order to eliminate conflicts with split transactions, the host controller evaluates the Periodic Schedule Enable bit only when FRINDEX[2:0] is zero. System software must not disable the periodic schedule if the schedule contains an active split transaction work item that spans the 000b micro-frame. These work items must be removed from the schedule before the Periodic Schedule Enable bit is written to zero.

The Periodic Schedule Status bit in the USB_USBSTS register indicates status of the periodic schedule. System software enables (or disables) the periodic schedule by writing one (or zero) to the Periodic Schedule Enable bit in the USB_USBCMD register. Software then can poll the Periodic Schedule Status bit to determine when the periodic schedule has made the desired transition.

Software must not modify the Periodic Schedule Enable bit unless the value of the Periodic Schedule Enable bit equals that of the Periodic Schedule Status bit.

The periodic schedule is used to manage all isochronous and interrupt transfer streams. The base of the periodic schedule is the periodic frame list. Software links schedule data structures to the periodic frame list to produce a graph of scheduled data structures. The graph represents an appropriate sequence of transactions.

The following figure illustrates isochronous transfers (using Isochronous transaction descriptors (iTDs) and Split-transaction isochronous transfer descriptors(siTDs)) with a period of one are linked directly to the periodic frame list. Interrupt transfers (are managed with queue heads) and isochronous streams with periods other than one are linked following the period-one iTD/siTDs. Interrupt queue heads are linked into the frame list ordered by poll rate. Longer poll rates are linked first (for example, closest to the periodic frame list), followed by shorter poll rates, with queue heads with a poll rate of one, on the very end.

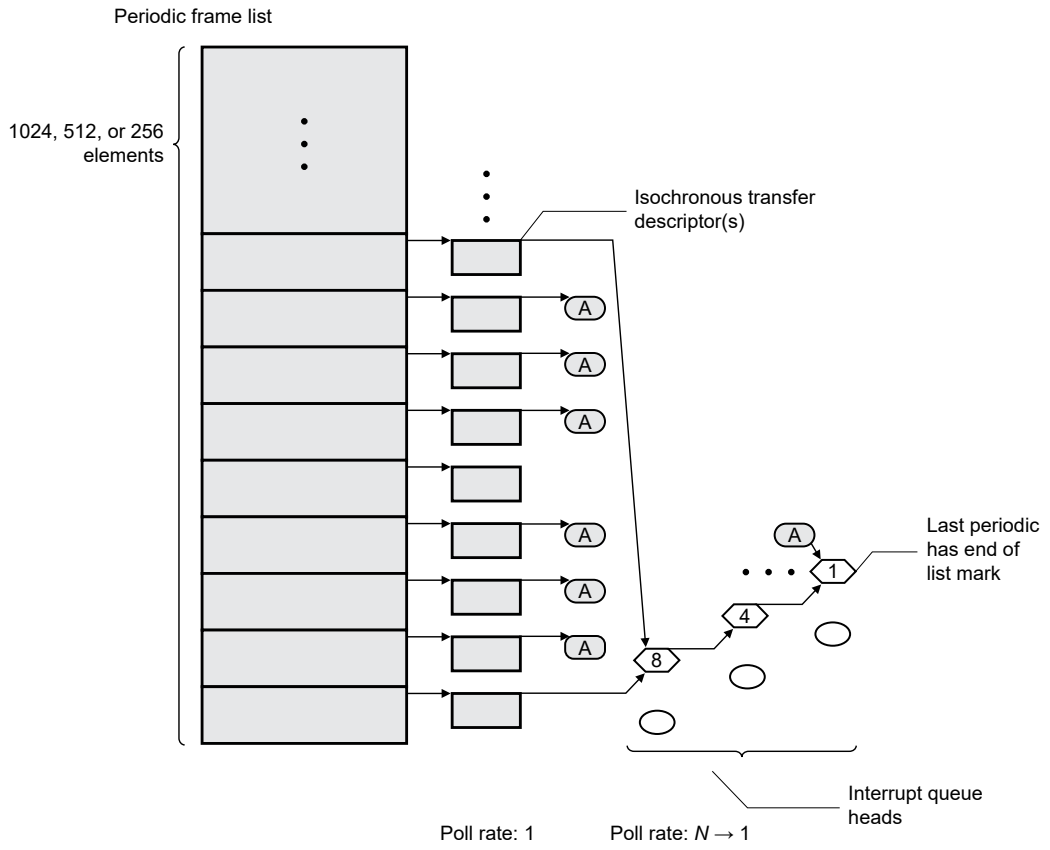


Figure 406. Example Periodic Schedule

56.3.2.1.7 Managing isochronous transfers using iTDs

The structure of an iTD is presented in Isochronous (high-speed) transfer descriptor (iTD). The four distinct sections to an iTD:

- The first field is the Next Link Pointer. This field is for schedule linkage purposes only.
- Transaction description array. This area is an eight-element array. Each element represents control and status information for one micro-frame's worth of transactions for a single high-speed isochronous endpoint.
- The buffer page pointer array is a 7-element array of physical memory pointers to data buffers. These are 4 K aligned pointers to physical memory.
- Endpoint capabilities. This area utilizes the unused low-order 12 bits of the buffer page pointer array. The fields in this area are used across all transactions executed for this iTD, including endpoint addressing, transfer direction, maximum packet size and high-bandwidth multiplier.

56.3.2.1.7.1 Host controller operational model for iTDs

The host controller uses FRINDEX register bits [12:3] to index into the periodic frame list. This means that the host controller visits each frame list element eight consecutive times before incrementing to the next periodic frame list element. Each iTD contains eight transaction descriptions, which map directly to FRINDEX register bits [2:0]. Therefore, each transaction descriptor corresponds to one micro-frame. Each iTD can span 8 micro-frames worth of transactions. When the host controller fetches an iTD, it uses FRINDEX register bits [2:0] to index into the transaction description array. If the active bit in the Status field of the indexed transaction description is set to zero, the host controller ignores the iTD and follows the Next pointer to the next schedule data structure.

When the indexed active bit is one, the host controller continues to parse the iTD. It stores the indexed transaction description and the general endpoint information (device address, endpoint number, maximum packet size, and so on.). It also uses the Page Select (PG) field to index the buffer pointer array, storing the selected buffer pointer and the next sequential buffer pointer. For example, if PG field is 0, then the host controller stores Page 0 and Page 1.

The host controller constructs a physical data buffer address by concatenating the current buffer pointer (as selected using the current transaction description's PG field) and the transaction description's Transaction Offset field. The host controller uses the endpoint addressing information and I/O-bit to execute a transaction to the appropriate endpoint. When the transaction is complete, the host controller clears the active bit and writes back any additional status information to the Status field in the currently selected transaction description.

The data buffer associated with the iTD must be virtually contiguous memory. Seven page pointers are provided to support eight high-bandwidth transactions regardless of the starting packet's offset alignment into the first page. A starting buffer pointer (physical memory address) is constructed by concatenating the page pointer (for example, page 0 pointer) selected by the active transaction descriptions' PG (for example, value: 00B) field with the transaction offset field. As the transaction moves data, the host controller must detect when an increment of the current buffer pointer crosses a page boundary. When this occurs the host controller simply replaces the current buffer pointer's page portion with the next page pointer (for example, page 1 pointer) and continues to move data. The size of each bus transaction is determined by the value in the Maximum Packet Size field. An iTD supports high-bandwidth pipes through the Mult (multiplier) field. When the Mult field is 1, 2, or 3, the host controller executes the specified number of Maximum Packet sized bus transactions for the endpoint in the current micro-frame. In other words, the Mult field represents a transaction count for the endpoint in the current micro-frame. If the Mult field is zero, the operation of the host controller is undefined. The transfer description is used to service all transactions indicated by the Mult field.

For OUT transfers, the value of the Transaction X Length field represents the total bytes to be sent during the micro-frame. The Mult field must be set by software to be consistent with Transaction X Length and Maximum Packet Size. The host controller sends the bytes in Maximum Packet Size'd portions. After each transaction, the host controller decrements its local copy of Transaction X Length by Maximum Packet Size. The number of bytes the host controller sends is always Maximum Packet Size or Transaction X Length, whichever is less. The host controller advances the transfer state in the transfer description, updates the appropriate record in the iTD and moves to the next schedule data structure. The maximum sized transaction supported is 3 x 1024 bytes.

For IN transfers, the host controller issues Mult transactions. It is assumed that software has properly initialized the iTD to accommodate all of the possible data. During each IN transaction, the host controller must use Maximum Packet Size to detect packet babble errors. The host controller keeps the sum of bytes received in the Transaction X Length field. After all transactions for the endpoint have completed for the micro-frame, Transaction X Length contains the total bytes received. If the final value of Transaction X Length is less than the value of Maximum Packet Size, then less data than was allowed for was received from the associated endpoint. This short packet condition does not set the USBINT bit in the USB_USBSTS register to one. The host controller will not detect this condition. If the device sends more than Transaction X Length or Maximum Packet Size bytes (whichever is less), then the host controller sets the Babble Detected bit to one and set the Active bit to zero. Note, that the host controller is not required to update the iTD field Transaction X Length in this error scenario. If the Mult field is greater than one, then the host controller automatically executes the value of Mult transactions. The host controller will not execute all Mult transactions if:

- The endpoint is an OUT and Transaction X Length goes to zero before all the Mult transactions have executed (ran out of data), or
- The endpoint is an IN and the endpoint delivers a short packet, or an error occurs on a transaction before Mult transactions have been executed. The end of micro-frame may occur before all of the transaction opportunities have been executed. When this happens, the transfer state of the transfer description is advanced to reflect the progress that was made, the result written back to the iTD and the host controller proceeds to processing the next micro-frame. Refer to Appendix D for a table summary of the host controller required behavior for all the high-bandwidth transaction cases.

56.3.2.1.7.2 Software operational model for iTDs

A client buffer request to an isochronous endpoint may span 1 to N micro-frames. When N is larger than one, system software may have to use multiple iTDs to read or write data with the buffer (if N is larger than eight, it must use more than one iTD).

The following figure illustrates the simple model of how a client buffer is mapped by system software to the periodic schedule (that is the periodic frame list and a set of iTDs). On the right is the client description of its request. The description includes a buffer base address plus additional annotations to identify which portions of the buffer should be used with each bus transaction. In the middle is the iTD data structures used by the system software to service the client request. Each iTD can be initialized to service up to 24 transactions, organized into eight groups of up to three transactions each. Each group maps to one micro-frame's worth of transactions. The EHCI controller does not provide per-transaction results within a micro-frame. It treats the per-micro-frame transactions as a single logical transfer. On the left is the host controller's frame list. System software establishes references from the appropriate locations in the frame list to each of the appropriate iTDs. If the buffer is large, then system software can use a small set of iTDs to service the entire buffer. System software can activate the transaction description records (contained in each iTD) in any pattern required for the particular data stream.

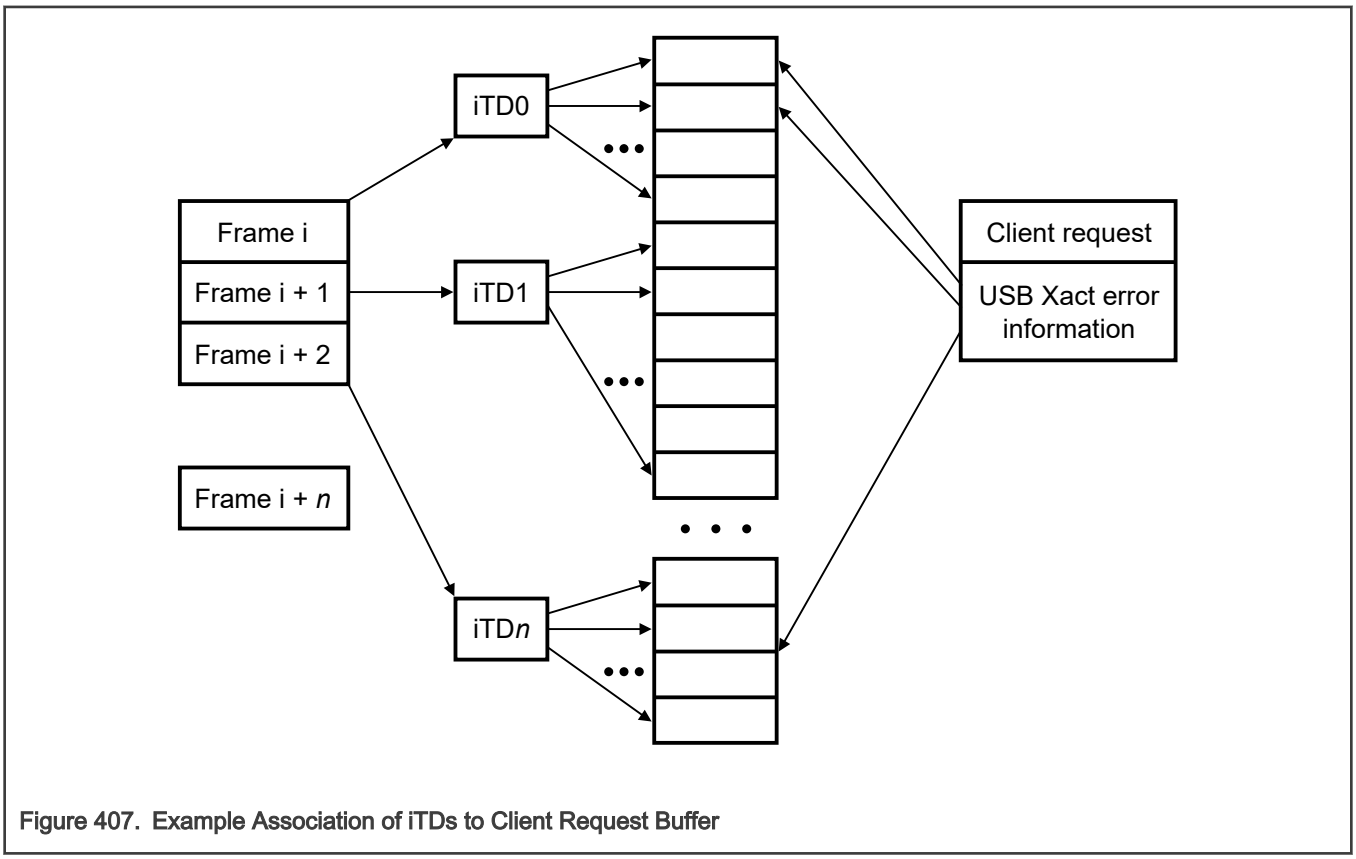


Figure 407. Example Association of iTDs to Client Request Buffer

As noted above, the client request includes a pointer to the base of the buffer and offsets into the buffer to annotate which buffer sections are to be used on each bus transaction that occurs on this endpoint. System software must initialize each transaction description in an iTD to ensure it uses the correct portion of the client buffer. For example, for each transaction description, the PG field is set to index the correct physical buffer page pointer and the Transaction Offset field is set relative to the correct buffer pointer page (for example, the same one referenced by the PG field). When the host controller executes a transaction it selects a transaction description record based on FRINDEX[2:0]. It then uses the current Page Buffer Pointer (as selected by the PG field) and concatenates to the transaction offset field. The result is a starting buffer address for the transaction. As the host controller moves data for the transaction, it must watch for a page wrap condition and properly advance to the next available Page Buffer Pointer. System software must not use the Page 6 buffer pointer in a transaction description where the length of the transfer wraps a page boundary. Doing so yields undefined behavior. The host controller hardware is not required to 'alias' the page selector to Page zero. USB 2.0 isochronous endpoints can specify a period greater than one. Software can achieve the appropriate scheduling by linking iTDs into the appropriate frames (relative to the frame list) and by setting appropriate transaction description elements active bits to one.

56.3.2.1.7.2.1 Periodic scheduling threshold

The Isochronous Scheduling Threshold field in the USB_HCCPARAMS capability register is an indicator to system software as to how the host controller pre-fetches and effectively caches schedule data structures. It is used by system software when adding isochronous work items to the periodic schedule. The value of this field indicates to system software the minimum distance it can update isochronous data (relative to the current location of the host controller execution in the periodic list) and still have the host controller process them.

The iTD and siTD data structures each describe 8 micro-frames worth of transactions. The host controller is allowed to cache one (or more) of these data structures in order to reduce memory traffic. Three basic caching models that account for the fact the isochronous data structures span 8 micro-frames. The three caching models are: no caching, micro-frame caching and frame caching.

When software is adding new isochronous transactions to the schedule, it always performs a read of the USB_FRINDEX register to determine the current frame and micro-frame the host controller is currently executing. Of course, there is no information about where in the micro-frame the host controller is, so a constant uncertainty-factor of one micro-frame has to be assumed. Combining the knowledge of where the host controller is executing with the knowledge of the caching model allows the definition of simple algorithms for how closely software can reliably work to the executing host controller.

No caching is indicated with a value of zero in the Isochronous Scheduling Threshold field. The host controller may pre-fetch data structures during a periodic schedule traversal (per micro-frame) but always dumps any accumulated schedule state at the end of the micro-frame. At the appropriate time relative to the beginning of every micro-frame, the host controller always begins schedule traversal from the frame list. Software can use the value of the USB_FRINDEX register (plus the constant 1 uncertainty-factor) to determine the approximate position of the executing host controller. When no caching is selected, software can add an isochronous transaction as near as 2 micro-frames in front of the current executing position of the host controller.

Frame caching is indicated with a non-zero value in bit [7] of the Isochronous Scheduling Threshold field. In the frame-caching model, system software assumes that the host controller caches one (or more) isochronous data structures for an entire frame (8 micro-frames). Software uses the value of the USB_FRINDEX register (plus the constant 1 uncertainty) to determine the current micro-frame/frame (assume modulo 8 arithmetic in adding the constant 1 to the micro-frame number). For any current frame N, if the current micro-frame is 0 to 6, then software can safely add isochronous transactions to Frame N + 1. If the current micro-frame is 7, then software can add isochronous transactions to Frame N + 2.

Micro-frame caching is indicated with a non-zero value in the least-significant 3 bits of the Isochronous Scheduling Threshold field. System software assumes the host controller caches one or more periodic data structures for the number of micro-frames indicated in the Isochronous Scheduling Threshold field. For example, if the count value were 2, then the host controller keeps a window of 2 micro-frames worth of state (current micro-frame, plus the next) on-chip. On each micro-frame boundary, the host controller releases the current micro-frame state and begins accumulating the next micro-frame state.

56.3.2.1.8 Asynchronous schedule

The Asynchronous schedule traversal is enabled or disabled through the Asynchronous Schedule Enable bit in the USB_USBCMD register. If the Asynchronous Schedule Enable bit is set to zero, then the host controller simply does not try to access the asynchronous schedule through the USB_ASYNCLISTADDR register. Likewise, when the Asynchronous Schedule Enable bit is one, then the host controller does use the USB_ASYNCLISTADDR register to traverse the asynchronous schedule. Modifications to the Asynchronous Schedule Enable bit are not necessarily immediate. Rather the new value of the bit is taken into consideration the next time the host controller needs to use the value of the USB_ASYNCLISTADDR register to get the next queue head.

The Asynchronous Schedule Status bit in the USB_USBSTS register indicates status of the asynchronous schedule. System software enables (or disables) the asynchronous schedule by writing one (or zero) to the Asynchronous Schedule Enable bit in the USB_USBCMD register. Software then can poll the Asynchronous Schedule Status bit to determine when the asynchronous schedule has made the desired transition. Software must not modify the Asynchronous Schedule Enable bit unless the value of the Asynchronous Schedule Enable bit equals that of the Asynchronous Schedule Status bit.

The asynchronous schedule is used to manage all Control and Bulk transfers. Control and Bulk transfers are managed using queue head data structures. The asynchronous schedule is based at the USB_ASYNCLISTADDR register. The default value of the USB_ASYNCLISTADDR register after reset is undefined and the schedule is disabled when the Asynchronous Schedule Enable bit is zero.

Software may only write this register with defined results when the schedule is disabled. For example, Asynchronous Schedule Enable bit in the USB_USBCMD and the Asynchronous Schedule Status bit in the USB_USBSTS register are zero. System software enables execution from the asynchronous schedule by writing a valid memory address (of a queue head) into this register. Then software enables the asynchronous schedule by setting the Asynchronous Schedule Enable bit to one. The asynchronous schedule is actually enabled when the Asynchronous Schedule Status bit is one.

When the host controller begins servicing the asynchronous schedule, it begins by using the value of the USB_ASYNCLISTADDR register. It reads the first referenced data structure and begins executing transactions and traversing the linked list as appropriate. When the host controller completes processing the asynchronous schedule, it retains the value of the last accessed queue head's horizontal pointer in the USB_ASYNCLISTADDR register. Next time the asynchronous schedule is accessed, this is the first data structure that is serviced. This provides round-robin fairness for processing the asynchronous schedule.

A host controller completes processing the asynchronous schedule when one of the following events occur:

- The end of a micro-frame occurs.
- The host controller detects an empty list condition (see Empty asynchronous schedule detection)
- The schedule has been disabled through the Asynchronous Schedule Enable bit in the USB_USBCMD register.

The queue heads in the asynchronous list are linked into a simple circular list as shown in Figure 402. Queue head data structures are the only valid data structures that may be linked into the asynchronous schedule. An isochronous transfer descriptor (iTD or siTD) in the asynchronous schedule yields undefined results.

The maximum packet size field in a queue head is sized to accommodate the use of this data structure for all non-isochronous transfer types. The USB Specification, Revision 2.0 specifies the maximum packet sizes for all transfer types and transfer speeds. System software should always parameterize the queue head data structures according to the core specification requirements.

56.3.2.1.8.1 Adding queue heads to asynchronous schedule

This is a software requirement section. There are two independent events for adding queue heads to the asynchronous schedule. The first is the initial activation of the asynchronous list. The second is inserting a new queue head into an activated asynchronous list.

Activation of the list is simple. System software writes the physical memory address of a queue head into the USB_ASYNCLISTADDR register, then enables the list by setting the Asynchronous Schedule Enable bit in the USB_USBCMD register to one.

When inserting a queue head into an active list, software must ensure that the schedule is always coherent from the host controllers' point of view. This means that the system software must ensure that all queue head pointer fields are valid. For example, qTD pointers have T-Bits set to one or reference valid Queue element transfer descriptors (qTDs) and the Horizontal Pointer references a valid queue head data structure. The following algorithm represents the functional requirements:

```
InsertQueueHead (pQHeadCurrent, pQueueHeadNew)
          --
          -- Requirement: all inputs must be properly initialized.
          --
          -- pQHeadCurrent is a pointer to a queue head that is
          -- already in the active list
          -- pQHeadNew is a pointer to the queue head to be added
          --
          -- This algorithm links a new queue head into an existing
          -- list
          --
          pQueueHeadNew.HorizontalPointer  =  pQueueHeadCurrent.HorizontalPointer
          pQueueHeadCurrent.HorizontalPointer =  physicalAddressOf(pQueueHeadNew)
End InsertQueueHead
```

56.3.2.1.8.2 Removing queue heads from asynchronous schedule

This is a software requirement section. There are two independent events for removing queue heads from the asynchronous schedule. The first is shutting down (deactivating) the asynchronous list. The second is extracting a single queue head from an activated list. Software deactivates the asynchronous schedule by setting the Asynchronous Schedule Enable bit in the USB_USBCMD register to zero. Software can determine when the list is idle when the Asynchronous Schedule Status bit in the USB_USBSTS register is zero. The normal mode of operation is that software removes queue heads from the asynchronous schedule without shutting it down. Software must not remove an active queue head from the schedule. Software should first deactivate all active qTDs, wait for the queue head to go inactive, then remove the queue head from the asynchronous list. Software removes a queue head from the asynchronous list through the following algorithm. As illustrated, the unlinking is quite easy. Software merely must ensure all of the link pointers reachable by the host controller are kept consistent.

```
UnlinkQueueHead (pQHeadPrevious, pQueueHeadToUnlink, pQHeadNext)
          --
          -- Requirement: all inputs must be properly initialized.
          --
          -- pQHeadPrevious is a pointer to a queue head that
          -- references the queue head to remove
          -- pQHeadToUnlink is a pointer to the queue head to be
          -- removed
          -- pQheadNext is a pointer to a queue head still in the
          -- schedule. Software provides this pointer with the
          -- following strict rules:
          --          if the host software is one queue head, then
          --          pQHeadNext must be the same as
          --          QueueheadToUnlink.HorizontalPointer. If the host
          --          software is unlinking a consecutive series of
          --          queue heads, QHeadNext must be set by software to
          --          the  queue head remaining in the schedule.
          --
          -- This algorithm unlinks a queue head from a circular list
          --
          pQueueHeadPrevious.HorizontalPointer =  pQueueHeadToUnlink.HorizontalPointer
                              pQueueHeadToUnlink.HorizontalPointer = pQHeadNext
End UnlinkQueueHead
```

If software removes the queue head with the H-bit set to one, it must select another queue head still linked into the schedule and set its H-bit to one. This should be completed before removing the queue head. The requirement is that software keep one queue head in the asynchronous schedule, with its H-bit set to one. At the point software has removed one or more queue heads from the asynchronous schedule, it is unknown whether the host controller has a cached pointer to them. Similarly, it is unknown how long the host controller might retain the cached information, as it is implementation dependent and may be affected by the actual dynamics of the schedule load. Therefore, once software has removed a queue head from the asynchronous list, it must retain the coherency of the queue head (link pointers, and so on). It cannot disturb the removed queue heads until it knows that the host controller does not have a local copy of a pointer to any of the removed data structures.

The method software uses to determine when it is safe to modify a removed queue head is to handshake with the host controller. The handshake mechanism allows software to remove items from the asynchronous schedule, then execute a simple, lightweight handshake that is used by software as a key that it can free (or reuse) the memory associated the data structures it has removed from the asynchronous schedule.

The handshake is implemented with three bits in the host controller. The first bit is a command bit (Interrupt on Async Advance Doorbell bit in the USB_USBCMD register) that allows software to inform the host controller that something has been removed from its asynchronous schedule. The second bit is a status bit (Interrupt on Async Advance bit in the USB_USBSTS register) that the host controller sets after it has released all on-chip state that may potentially reference one of the data structures just removed. When the host controller sets this status bit to one, it also sets the command bit to zero. The third bit is an interrupt enable (Interrupt on Async Advance bit in the USB_USBINTR register) that is matched with the status bit. If the status bit is one and the interrupt enable bit is one, then the host controller asserts a hardware interrupt.

The figure below illustrates a general example. In this example, consecutive queue heads (B and C) are unlinked from the schedule using the algorithm above. Before the unlink operation, the host controller has a copy of queue head A.

The unlink algorithm requires that as software unlinks each queue head, the unlinked queue head is loaded with the address of a queue head that remains in the asynchronous schedule.

When the host controller observes that doorbell bit being set to one, it makes a note of the local reachable schedule information. In this example, the local reachable schedule information includes both queue heads (A and B). It is sufficient that the host controller can set the status bit (and clear the doorbell bit) as soon as it has traversed beyond current reachable schedule information (that is traversed beyond queue head (B) in this example). The following figure illustrates the generic queue head unlink scenario.

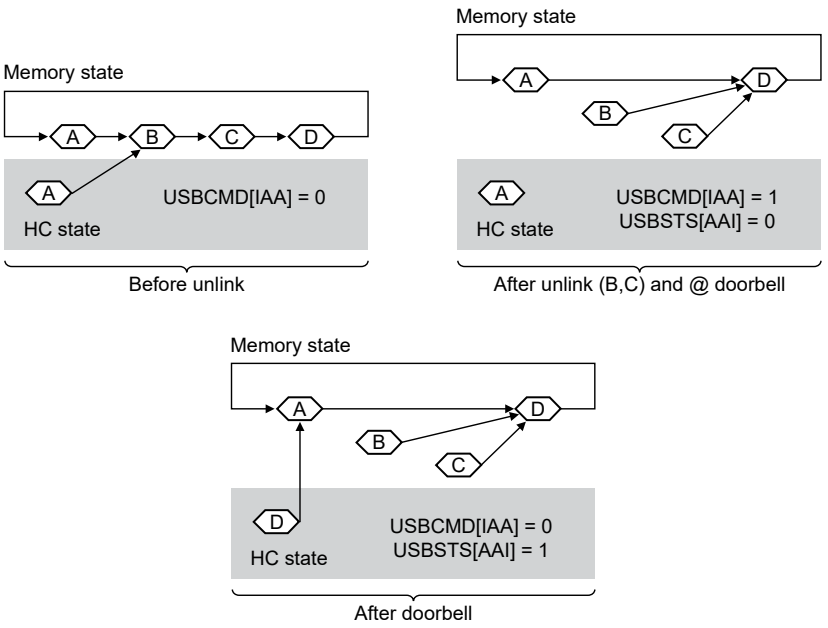


Figure 408. Generic Queue Head Unlink Scenario

Alternatively, a host controller implementation is allowed to traverse the entire asynchronous schedule list (for example, observed the head of the queue (twice)) before setting the Advance on Async status bit to one.

Software may re-use the memory associated with the removed queue heads after it observes the Interrupt on Async Advance status bit is set to one, following assertion of the doorbell. Software should acknowledge the Interrupt on Async Advance status as indicated in the USB_USBSTS register, before using the doorbell handshake again.

56.3.2.1.8.3 Empty asynchronous schedule detection

The Enhanced Host Controller Interface uses two bits to detect when the asynchronous schedule is empty. The queue head data structure (see Queue head) defines an *H-bit* in the queue head, which allows software to mark a queue head as being the *head* of the reclaim list. The Enhanced Host Controller Interface also keeps a 1-bit flag in the USB_USBSTS register (*Reclamation*) that is set to zero when the Enhanced Interface Host Controller observes a queue head with the H-bit set to one. The reclamation flag in the status register is set to one when any USB transaction from the asynchronous schedule is executed (or whenever the asynchronous schedule starts, see Asynchronous schedule traversal: *Start event*).

If the Enhanced Host Controller Interface ever encounters an *H-bit* of one and a *Reclamation* bit of zero, the EHCI controller simply stops traversal of the asynchronous schedule.

An example illustrating the H-bit in a schedule is shown in the following figure.

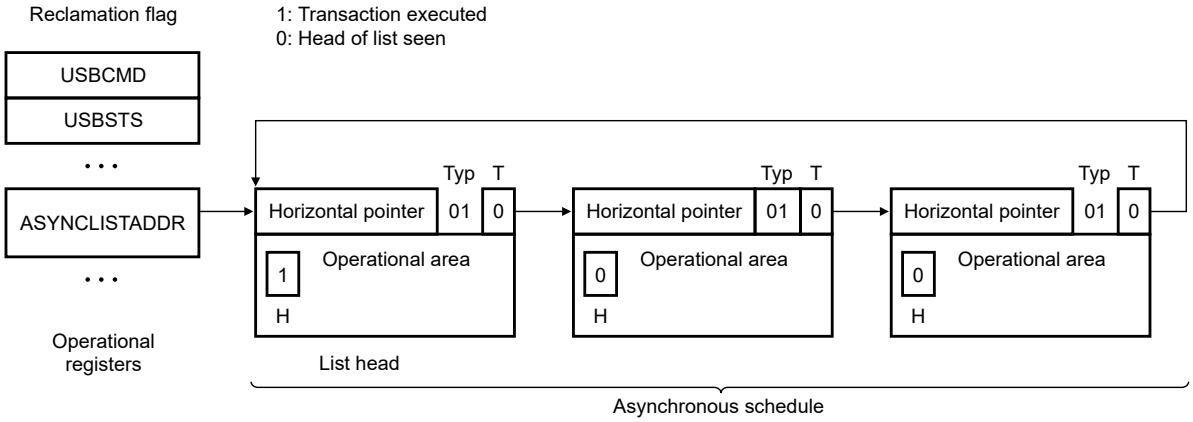


Figure 409. Asynchronous Schedule List w/Annotation to Mark Head of List

Software must ensure there is at most one queue head with the *H-bit* set to one, and that it is always coherent with respect to the schedule.

56.3.2.1.8.4 Restarting asynchronous schedule before EOF

There are many situations where the host controller will detect an empty list *long* before the end of the micro-frame. It is important to remember that under many circumstances the schedule traversal has stopped due to Nak/Nyet responses from all endpoints.

An example of particular interest is when a start-split for a bulk endpoint occurs early in the micro-frame. Given the EHCI simple traversal rules, the complete-split for that transaction may Nak/Nyet out very quickly. If it is the only item in the schedule, then the host controller ceases traversal of the Asynchronous schedule very early in the micro-frame. In order to provide reasonable service to this endpoint, the host controller should issue the complete-split before the end of the current micro-frame, instead of waiting until the next micro-frame. When the reason for host controller idling asynchronous schedule traversal is because of empty list detection, it is mandatory the host controller implement a 'waking' method to resume traversal of the asynchronous schedule. An example method is described below.

56.3.2.1.8.4.1 Example method for restarting asynchronous Schedule traversal

The reason for idling the host controller when the list is empty is to keep the host controller from unnecessarily occupying too much memory bandwidth. The question is: *how long should the host controller stay idle before restarting*?

The answer in this example is based on deriving a manifest constant, which is the amount of time the host controller will stay idle before restarting traversal. In this example, the manifest constant is called *AsyncSchedSleepTime*, and has a value of 10 µsec. The value is derived based on the analysis in Example derivation for *AsyncSchedSleepTime*. The traversal algorithm is simple:

- Traverse the Asynchronous schedule until either an End-Of-micro-Frame event occurs, or an empty list is detected. If the event is an End-of-micro-Frame, go attempt to traverse the Periodic schedule. If the event is an empty list, then set a sleep timer and go to a *schedule sleep* state.
- When the sleep timer expires, set working context to the Asynchronous Schedule start condition and go to *schedule active* state. The start context allows the HC to reload *Nakcnt* fields, and so on. So the HC has a chance to run for more than one iteration through the schedule.

This process simply repeats itself each micro-frame. The figure below illustrates a sample state machine to manage the active and sleep states of the Asynchronous Schedule traversal policy. There are three states: Actively traversing the Asynchronous schedule, Sleeping, and Not Active. The last two are similar in terms of interaction with the Asynchronous schedule, but the Not Active state means that the host controller is busy with the Periodic schedule or the Asynchronous schedule is not enabled. The Sleeping state is specifically a special state where the host controller is just waiting for a period of time before resuming execution of the Asynchronous schedule.

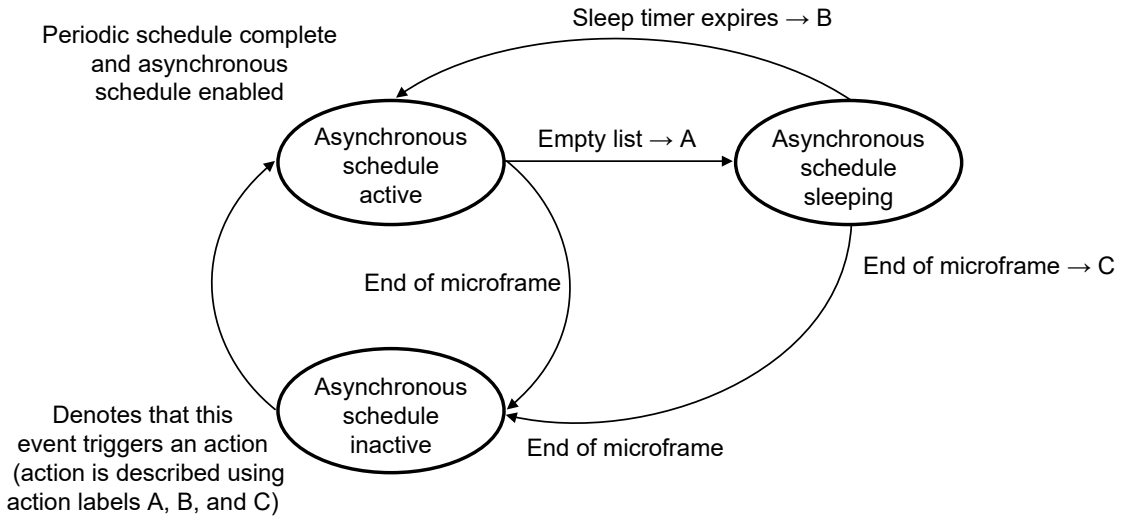


Figure 410. Example State Machine for Managing Asynchronous Schedule Traversal

The actions referred to in the figure above are defined in the following table.

Table 837. Asynchronous Schedule SM Transition Actions

Action	Action Description Label
A	On detection of the empty list, the host controller sets the *AsynchronousTraversalSleepTimer* to *AsyncSchedSleepTime*.
B	When the *AsynchronousTraversalSleepTimer* expires, the host controller sets the Reclamation bit in the USBSTS register to one, and moves the Nak Counter reload state machine to WaitForListHead (see Nak count reload control).
C	The host controller cancels the sleep timer (*AsynchronousTraversalSleepTimer*).

56.3.2.1.8.4.2 Async sched not active

This is the initial state of the traversal state machine after a host controller reset. The traversal state machine does not leave this state when the *Asynchronous Schedule Enable* bit in the USB_USBCMD register is zero.

This state is entered from Async Sched Active or Async Sched Sleeping states when the end-of-micro-frame event is detected.

56.3.2.1.8.4.3 Async sched active

This state is entered from the Async Sched Not Active state when the periodic schedule is not active. It is also entered from the Async Sched Sleeping states when the *AsynchronousTraversalSleepTimer* expires. On every transition into this state, the host controller sets the *Reclamation* bit in the USB_USBSTS register to one.

While in this state, the host controller continually traverses the asynchronous schedule until either the end of micro-frame or an empty list condition is detected.

56.3.2.1.8.4.4 Async sched sleeping

The state is entered from the Async Sched Active state when a schedule empty condition is detected. On entry to this state, the host controller sets the *AsynchronousTraversalSleepTimer* to *AsyncSchedSleepTime*.

56.3.2.1.8.4.5 Example derivation for AsyncSchedSleepTime

The derivation is based on analysis of what work the host controller could be doing next. It assumes the host controller does not keep any state about what work is possibly pending in the asynchronous schedule. The schedule could contain any mix of the possible combinations of high- full- or low-speed control and bulk requests.

The table below summarizes some of the typical 'next transactions' that could be in the schedule, and the amount of time (for example *footprint*, or *wall clock*) the transaction takes to complete.

Table 838. Typical Low-/Full-speed Transaction Times

Transaction Attributes		Footprint (time)	Description
Speed	HS	11.9 ms	Maximum foot print for a worst-case, full-sized bulk data transaction.
Size	512	9.45 ms	Maximum footprint for an approximate best-case, full-sized bulk data transaction.
Type	Bulk		
Speed	FS	~50 ms	Approximate typical for full-sized bulk data. An 8-byte low-speed is about 2x, or between 90 and 100 ms.
Size	64		
Type	Bulk		
Speed	FS	~12 ms	Approximate typical for 8-byte bulk/control (that is setup)
Size	8		
Type	Cntrl		

A *AsyncSchedSleepTime* value of 10 µs provides a reasonable relaxation of the system memory load and still provides a good level of service for the various transfer types and payload sizes. For example, say we detect an empty list after issuing a start-split for a 64-byte full-speed bulk request. Assuming this is the only thing in the list, the host controller gets the results of the full-speed transaction from the hub during the fifth complete-split request. If the full-speed transaction was an IN and it nak'd, the 10 µs sleep period would allow the host controller to get the NAK results on the first complete-split.

56.3.2.1.8.5 Asynchronous schedule traversal: *Start event*

Once the HC has *idled* itself through the empty schedule detection (Section 0), it will naturally *activate* and begin processing from the Periodic Schedule at the beginning of each micro-frame. In addition, it may have idled itself early in a micro-frame. When this occurs (idles early in the micro-frame) the HC must occasionally *re-activate* during the micro-frame and traverse the asynchronous schedule to determine whether any progress can be made. The requirements and method for this restart are described in Restarting asynchronous schedule before EOF . Asynchronous schedule *Start Event*s are defined to be:

- Whenever the host controller transitions from the periodic schedule to the asynchronous schedule. If the periodic schedule is disabled and the asynchronous schedule is enabled, then the beginning of the micro-frame is equivalent to the transition from the periodic schedule, or
- The asynchronous schedule traversal restarts from a sleeping state (see Restarting asynchronous schedule before EOF).

56.3.2.1.8.6 Reclamation status bit (USBSTS Register)

The operation of the empty asynchronous schedule detection feature (see Empty asynchronous schedule detection) depends on the proper management of the *Reclamation* bit in the USB_USBSTS register. The host controller tests for an empty schedule just after it fetches a new queue head while traversing the asynchronous schedule (see Fetch queue head).

The host controller is required to set the *Reclamation* bit to one whenever an asynchronous schedule traversal *Start Event*, as documented in Asynchronous schedule traversal: *Start event*, occurs. The *Reclamation* bit is also set to one whenever the host controller executes a transaction while traversing the asynchronous schedule (see Execute transaction). The host controller sets the *Reclamation* bit to zero whenever it finds a queue head with its *H-bit* set to one. Software should only set a queue head's *H-bit* if the queue head is in the asynchronous schedule. If software sets the *H-bit* in an interrupt queue head to one, the resulting behavior is undefined. The host controller may set the *Reclamation* bit to zero when executing from the periodic schedule.

56.3.2.1.9 Operational model for Nak counter

This section describes the operational model for the *NakCnt* field defined in a queue head. See Queue head initialization for more information. Software should not use this feature for interrupt queue heads. This rule is not required to be enforced by the host controller.

USB protocol has built-in flow control through the Nak response by a device. There are several scenarios, beyond the Ping feature, where an endpoint may naturally Nak or Nyet the majority of the time. An example is the host controller management of the split transaction protocol for control and bulk endpoints. All bulk endpoints (High- or Full-speed) are serviced through the same asynchronous schedule. The time between the *Start-split* transaction and the first *Complete-split* transaction could be very short (that is like when the endpoint is the only one in the asynchronous schedule). The hub NYETs (effectively Naks) the *Complete-split* transaction until the classic transaction is complete. This could result in the host controller thrashing memory, repeatedly fetching the queue head and executing the transaction to the Hub, which does not complete until after the transaction on the classic bus completes.

The two component fields in a queue head to support the throttling feature: a counter field (*NakCnt*), and a counter reload field (*RL*). *NakCnt* is used by the host controller as one of the criteria to determine whether or not to execute a transaction to the endpoint. The two operational modes associated with this counter:

- Not Used- This mode is set when the *RL* field is zero. The host controller ignores the *NakCnt* field for any execution of transactions through a queue head with an *RL* field of zero. Software must use this selection for interrupt endpoints.
- Nak Throttle Mode- This mode is selected when the *RL* field is non-zero. In this mode, the value in the *NakCnt* field represents the maximum number of Nak or Nyet responses the host controller tolerates on each endpoint. In this mode, the HC decrements the *NakCnt* field based on the token/handshake criteria listed in the table below. The host controller must reload *NakCnt* when the endpoint successfully moves data (for example, policy to reward device for moving data).

The following table describes the NakCnt field adjustment rules.

Table 839. NakCnt Field Adjustment Rules

Token	Handshake	
	Handshake NAK	NYET
IN/PING	decrement *NakCnt*	N/A (protocol error)
OUT	decrement *NakCnt*	No Action[1] Start
Split	decrement *NakCnt*	N/A (protocol error)
Complete Split	No Action	Decrement *NakCnt*

1. Recommended behavior on this response is to reload *NakCnt*

In summary, system software enables the counter by setting the reload field (*RL*) to a non-zero value. The host controller may execute a transaction if *NakCnt* is non-zero. The host controller does not execute a transaction if *NakCnt* is zero. The reload mechanism is described in detail in Nak count reload control .

NOTE

When all queue heads in the Asynchronous Schedule either exhausts all transfers or all NakCnt's go to zero, then the host controller detects an empty Asynchronous Schedule and idle schedule traversal (see Empty asynchronous schedule detection).

Any time the host controller begins a new traversal of the Asynchronous Schedule, a *Start Event* is assumed, see Asynchronous schedule traversal: *Start event*. Every time a Start-Event occurs, the Nak Count reload procedure is enabled.

56.3.2.1.9.1 Nak count reload control

When the host controller reaches the *Execute Transaction* state for a queue head (meaning that it has an active operational state), it checks to determine whether the *NakCnt* field should be reloaded from *RL* (see Execute transaction). If the answer is yes, then *RL* is copied into *NakCnt*. After the reload or if the reload is not active, the host controller evaluates whether to execute the transaction.

The host controller must reload nak counters (*NakCnt* see Queue head) in queue heads during the first pass through the reclamation list after an asynchronous schedule Start Event (see Asynchronous schedule traversal: *Start event* for the definition of the Start Event). The Asynchronous Schedule should have at most one queue head marked as the head (see Figure 409).

The following figure illustrates an example state machine that satisfies the operational requirements of the host controller detecting the first pass through the Asynchronous Schedule. This state machine is maintained internal to the host controller and is only used to gate reloading of the nak counter during the queue head traversal state: Execute Transaction (see the figure below). The host controller does not perform the nak counter reload operation if the RL field (see Queue head) is set to zero.

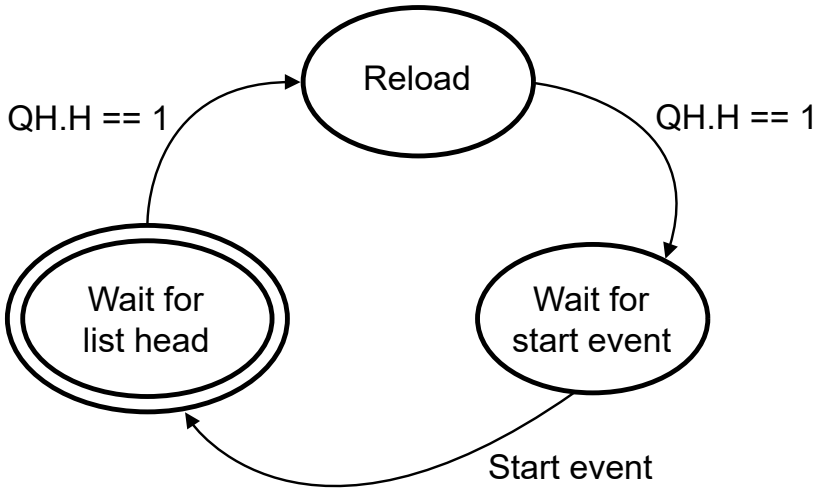


Figure 411. Example HC State Machine for Controlling Nak Counter Reloads

56.3.2.1.9.1.1 Wait for list head

This is the initial state. The state machine enters this state from Wait for Start Event when a start event as defined in Asynchronous schedule traversal: *Start event* occurs. The purpose of this state is to wait for the first observation of the head of the Asynchronous Schedule. This occurs when the host controller fetches a queue head whose *H-bit* is set to one.

56.3.2.1.9.1.2 Do reload

This state is entered from the Wait for List Head state when the host controller fetches a queue head with the *H-bit* set to one. While in this state, the host controller performs nak counter reloads for every queue head visited that has a non-zero nak reload value (*RL*) field.

56.3.2.1.9.1.3 Wait for start event

This state is entered from the *Do Reload* state when a queue head with the *H-bit* set to one is fetched. While in this state, the host controller does not perform nak counter reloads.

56.3.2.1.10 Managing control/bulk/interrupt transfers through queue heads

This section presents an overview of how the host controller interacts with queuing data structures.

Queue heads use the Queue Element Transfer Descriptor (qTD) structure. One queue head is used to manage the data stream for one endpoint. The queue head structure contains static endpoint characteristics and capabilities. It also contains a working area from where individual bus transactions for an endpoint are executed (see Overlay area defined in Table 891). Each qTD represents one or more bus transactions, which is defined in the context of this specification as a *transfer*.

The general processing model for the host controller's use of a queue head is simple:

- read a queue head,
- execute a transaction from the overlay area,
- write back the results of the transaction to the overlay area,
- move to the next queue head.

If the host controller encounters errors during a transaction, the host controller sets one (or more) of the error reporting bits in the queue head's *Status* field. The *Status* field accumulates all errors encountered during the execution of a qTD (for example, the error bits in the queue head *Status* field are 'sticky' until the transfer (qTD) has completed). This state is always written back to the source qTD when the transfer is complete. On transfer (for example, buffer or halt conditions) boundaries, the host controller must auto-advance (without software intervention) to the next qTD. Additionally, the hardware must be able to halt the queue so no additional bus transactions occurs for the endpoint and the host controller does not advance the queue.

An example host controller operational state machine of a queue head traversal is illustrated in the following figure. This state machine is a model for how a host controller should traverse a queue head. The host controller must be able to advance the queue from the *Fetch QH* state in order to avoid all hardware/software race conditions. This simple mechanism allows software to simply link qTDs to the queue head and *activate* them, then the host controller always *find* them if/when they are reachable. The figure below illustrates the Host Controller Queue Head Traversal State Machine.

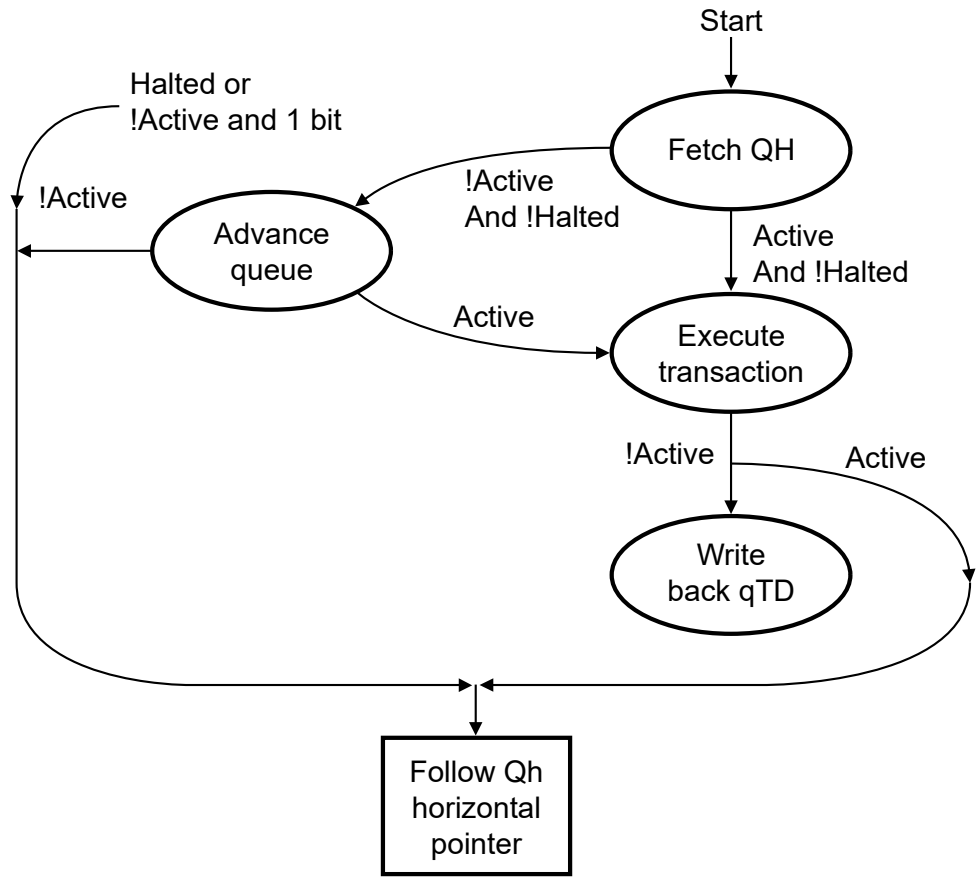


Figure 412. Host Controller Queue Head Traversal State Machine

This traversal state machine applies to all queue heads, regardless of transfer type or whether split transactions are required. The following sections describe each state. Each state description describes the entry criteria. The Execute Transaction state (see Execute transaction) describes the basic requirements for all endpoints. Split transactions for asynchronous transfers and Split transaction interrupt describe details of the required extensions to the Execute Transaction state for endpoints requiring split transactions.

NOTE

Prior to software placing a queue head into either the periodic or asynchronous list, software must ensure the queue head is properly initialized. Minimally, the queue head should be initialized to the following (see Section Queue Head for layout of a queue head):

Valid static endpoint state.

- For the very first use of a queue head, software may zero-out the queue head transfer overlay, then set the *Next qTD Pointer* field value to reference a valid qTD.

56.3.2.1.10.1 Fetch queue head

A queue head can be referenced from the physical address stored in the ASYNCLISTADDR Register. Additionally, it may be referenced from the *Next LinkPointer* field of an iTD, siTD, FSTN or another Queue Head. If the referencing link pointer has the *Typ* field set to indicate a queue head, it is assumed to reference a queue head structure as defined in Queue head.

While in this state, the host controller performs operations to implement empty schedule detection (see Empty asynchronous schedule detection) and Nak Counter reloads (see Operational model for Nak counter). After the queue head has been fetched, the host controller conducts the following queries for empty schedule detection:

- If queue head is not an interrupt queue head (that is *S-mask* is zero), and
- The *H-bit* is one, and
- The *Reclamation* bit in the USBSTS register is zero.

When these criteria are met, the host controller stops traversing the asynchronous list (as described in Empty asynchronous schedule detection). When the criteria are not met, the host controller continues schedule traversal. If the queue head is not an interrupt and the *H-bit* is one and the *Reclamation* bit is one, then the host controller sets the *Reclamation* bit in the USBSTS register to zero before completing this state. The operations for reloading of the Nak Counter are described in detail in Operational model for Nak counter.

This state is complete when the queue head has been read on-chip.

56.3.2.1.10.2 Advance queue

To advance the queue, the host controller must find the next qTD, adjust pointers, perform the overlay and write back the results to the queue head.

This state is entered from the FetchQHD state if the overlay *Active* and *Halt* bits are set to zero. On entry to this state, the host controller determines which next pointer to use to fetch a qTD, fetches a qTD and determines whether or not to perform an overlay.

NOTE

If the *I-bit* is one and the *Active* bit is zero, the host controller immediately skips processing of this queue head, exits this state and uses the horizontal pointer to the next schedule data structure. If the field *Bytes to Transfer* is not zero and the *T-bit* in the *Alternate Next qTD Pointer* is set to zero, then the host controller uses the *Alternate Next qTD Pointer*. Otherwise, the host controller uses the *NextqTD Pointer*. If *NextqTD Pointer's T-bit* is set to one, then the host controller exits this state and uses the horizontal pointer to the next schedule data structure.

Using the selected pointer the host controller fetches the referenced qTD. If the fetched qTD has its *Active* bit set to one, the host controller moves the pointer value used to reach the qTD (*Next* or *Alternate Next*) to the *Current qTD Pointer* field, then performs the overlay. If the fetched qTD has its *Active* bit set to zero, the host controller aborts the queue advance and follows the queue head's horizontal pointer to the next schedule data structure.

The host controller performs the overlay based on the following rules:

- The value of the data toggle (*dt*) field in the overlay area depends on the value of the *data toggle control* (*dtc*) bit (see Table 893).
- If the *EPS* field indicates the endpoint is a high-speed endpoint, the *Ping* state field is preserved by the host controller. The value of this field is not changed as a result of the overlay.
- *C-prog-mask* field is set to zero (field from incoming qTD is ignored, as is the current contents of the overlay area).
- *Frame Tag* field is set to zero (field from incoming qTD is ignored, as is the current contents of the overlay area).
- *NakCnt* field in the overlay area is loaded from the *RL* field in the queue head's Static Endpoint State.
- All other areas of the overlay are set by the incoming qTD.

The host controller exits this state when it has committed the write to the queue head.

56.3.2.1.10.3 Execute transaction

The host controller enters this state from the Fetch Queue Head state only if the *Active* bit in *Status* field of the queue head is set to one.

On entry to this state, the host controller executes a few pre-operations, then checks some pre-condition criteria before committing to executing a transaction for the queue head.

The pre-operations performed and the pre-condition criteria depend on whether the queue head is an interrupt endpoint. The host controller can determine that a queue head is an interrupt queue head when the queue head's *S-mask* field contains a non-zero value. It is the responsibility of software to ensure the *S-mask* field is appropriately initialized based on the transfer type. There are

other criteria that must be met if the *EPS* field indicates that the endpoint is a low- or full-speed endpoint, see Split transactions for asynchronous transfers and Split transaction interrupt .

56.3.2.1.10.3.1 Interrupt transfer pre-condition criteria

If the queue head is for an interrupt endpoint (for example, non-zero *S-mask* field), then the FRINDEX[2:0] field must identify a bit in the *S-mask* field that has one in it. For example, an *S-mask* value of 00100000b would evaluate to true only when FRINDEX[2:0] is equal to 101b. If this condition is met then the host controller considers this queue head for a transaction.

56.3.2.1.10.3.2 Asynchronous transfer pre-operations and pre-condition criteria

If the queue head is not for an interrupt endpoint (for example, zero *S-mask* field), then the host controller performs one pre-operation and then evaluates one pre-condition criteria.

The pre-operation is:

Checks the Nak counter reload state (Operational model for Nak counter). It may be necessary for the host controller to reload the Nak Counter field. The reload is performed at this time.

The pre-condition evaluated is:

- Whether or not the *NakCnt* field has been reloaded, the host controller checks the value of the *NakCnt* field in the queue head. If *NakCnt* is non-zero, or if the *Reload Nak Counter* field is zero, then the host controller considers this queue head for a transaction.

56.3.2.1.10.3.3 Transfer type independent pre-operations

Regardless of the transfer type, the host controller always performs at least one pre-operation and evaluates one pre-condition. The pre-operation is:

- A host controller internal transaction (down) counter *qHTransactionCounter* is loaded from the queue head's *Mult* field. A host controller implementation is allowed to ignore this for queue heads on the asynchronous list. It is mandatory for interrupt queue heads. Software should ensure that the *Mult* field is set appropriately for the transfer type.

The pre-conditions evaluated are:

- The host controller determines whether there is enough time in the micro-frame to complete this transaction (see Transaction fit - a best-fit approximation algorithm for an example evaluation method). If there is not enough time to complete the transaction, the host controller exits this state.
- If the value of *qHTransactionCounter* for an interrupt endpoint is zero, then the host controller exits this state.

When the pre-operations are complete and pre-conditions are met, the host controller sets the *Reclamation* bit in the USBSTS register to one and then begins executing one or more transactions using the endpoint information in the queue head. The host controller iterates *qHTransactionCounter* times in this state executing transactions. After each transaction is executed, *qHTransactionCounter* is decremented by one. The host controller exits this state when one of the following events occurs:

- The *qHTransactionCounter* decrements to zero, or
- The endpoint responds to the transaction with any handshake other than an ACK,[4] or
- The transaction experiences a transaction error, or
- The *Active* bit in the queue head goes to zero, or
- There is not enough time in the micro-frame left to execute the next transaction (see Transaction fit - a best-fit approximation algorithm), for example method for implementing the frame boundary test).

> **NOTE**
>
> For a high-bandwidth interrupt OUT endpoint, the host controller may optionally immediately retry the transaction if it fails.

The results of each transaction is recorded in the on-chip overlay area. If data was successfully moved during the transaction, the transfer state in the overlay area is advanced. To advance queue head's transfer state, the *Total Bytes to Transfer* field is decremented by the number of bytes moved in the transaction, the data toggle bit (*dt*) is toggled, the current page offset is

advanced to the next appropriate value (for example, advanced by the number of bytes successfully moved), and the *C_Page* field is updated to the appropriate value (if necessary). See Buffer pointer list use for data streaming with qTDs .

NOTE

The *Total Bytes To Transfer* field may be zero when all the other criteria for executing a transaction are met. When this occurs, the host controller executes zero-length transaction to the endpoint. If the *PID_Code* field indicates an IN transaction and the device delivers data, the host controller detects a packet babble condition, set the *babble* and *halted* bits in the *Status* field, set the *Active* bit to zero, write back the results to the source qTD, then exit this state.

In the event an IN token receives a data PID mismatch response, the host controller must ignore the received data (for example not advance the transfer state for the bytes received). Additionally, if the endpoint is an interrupt IN, then the host controller must record that the transaction occurred (for example, decrement *qHTransactionCounter*). It is recommended (but not required) the host controller continue executing transactions for this endpoint if the resultant value of *qHTransactionCounter* is greater than one.

If the response to the IN bus transaction is a Nak (or Nyet) and *RL* is non-zero, *NakCnt* is decremented by one. If *RL* is zero, then no write-back by the host controller is required (for a transaction receiving a Nak or Nyet response and the value of *CErr* did not change). Software should set the *RL* field to zero if the queue head is an interrupt endpoint. Host controller hardware is not required to enforce this rule or operation.

After the transaction has finished and the host controller has completed the post processing of the results (advancing the transfer state and possibly *NakCnt*, the host controller writes back the results of the transaction to the queue head's overlay area in main memory).

The number of bytes moved during an IN transaction depends on how much data the device endpoint delivers. The maximum number of bytes a device can send is *MaximumPacket Size*. The number of bytes moved during an OUT transaction is either *Maximum Packet Length* bytes or *Total Bytes to Transfer*, whichever is less.

If there was a transaction error during the transaction, the transfer state (as defined above) is not advanced by the host controller. The *CErr* field is decremented by one and the status field is updated to reflect the type of error observed. Transaction errors are summarized in Transaction error .

The following events cause the host controller to clear the *Active* bit in the queue head's overlay status field. When the *Active* bit transitions from one to zero, the transfer in the overlay is considered complete. The reason for the transfer completion (clearing the *Active* bit) determines the next state.

- *CErr* field decrements to 0. When this occurs, the *Halted* bit is set to 1 and *Active* is set to 0. This results in the hardware not advancing the queue and the pipe halts. You must intercede to recover.
- The device responds to the transaction with a STALL PID. When this occurs, the *Halted* bit is set to 1 and the *Active* bit is set to 0. This results in the hardware not advancing the queue and the pipe halts. Software must intercede to recover.
- The *Total Bytes to Transfer* field is 0 after the transaction completes.
  - For a 0-length transaction, it was 0 before the transaction was started. When this condition occurs, the *Active* bit is set to 0.
- The PID code is an IN, and the number of bytes moved during the transaction is less than the *Maximum Packet Length*. When this occurs, the *Active* bit is set to 0 and a short packet condition exists. The short-packet condition is detected during the Advance Queue state. Refer to Split transactions for additional rules for managing low- and full-speed transactions.

With the exception of a NAK response (when *RL* field is zero), the host controller always writes the results of the transaction back to the overlay area in main memory. This includes when the transfer completes. For a high-speed endpoint, the queue head information written back includes minimally the following fields: The *PID Code* field indicates an IN and the device sends more than the expected number of bytes (for example *Maximum Packet Length* or *Total Bytes to Transfer* bytes, whichever is less) (for example a packet babble). This results in the host controller setting the *Halted* bit to one.

- NakCnt, dt, Total Bytes to Transfer, C_Page, Status, CERR, and Current Offset

For a low- or full-speed device the queue head information written back also includes the fields:

- C-prog-mask, FrameTag and S-bytes.

The duration of this state depends on the time it takes to complete the transaction(s) and the status write to the overlay is committed.

56.3.2.1.10.3.4 Halting a queue head

A halted endpoint is defined only for the transfer types that are managed through queue heads (control, bulk and interrupt). The following events indicate that the endpoint has reached a condition where no more activity can occur without intervention from the driver:

- An endpoint may return a STALL handshake during a transaction,
- A transaction had three consecutive error conditions, or
- A Packet Babble error occurs on the endpoint.

When any of these events occur (for a queue head) the Host Controller halts the queue head and set the USBERRINT status bit in the USB_n_USBSTS register to one. To halt the queue head, the *Active* bit is set to zero and the *Halted* bit is set to one. There may be other error status bits that are set when a queue is halted. The host controller always writes back the overlay area to the source qTD when the transfer is complete, regardless of the reason (normal completion, short packet or halt). The host controller does not advance the transfer state on a transaction that results in a *Halt* condition (for example no updates necessary for *Total Bytes to Transfer*, *C_Page*, *Current Offset,* and *dt*). The host controller must update *CErr* as appropriate. When a queue head is halted, the *USB Error Interrupt* bit in the USB_n_USBSTS register is set to one. If the *USB Error Interrupt Enable* bit in the USB_n_USBINTR register is set to one, a hardware interrupt is generated at the next interrupt threshold.

56.3.2.1.10.3.5 Asynchronous schedule park mode

Asynchronous Schedule Park mode is a special execution mode that can be enabled by system software, where the host controller is permitted to execute more than one bus transaction from a high-speed queue head in the Asynchronous schedule before continuing horizontal traversal of the Asynchronous schedule. This feature has no effect on queue heads or other data structures in the Periodic schedule. This feature is similar in intent as the *Mult* feature that is used in the Periodic schedule. Where-as the *Mult* feature is a characteristic that is tunable for each endpoint; park-mode is a policy that is applied to all high-speed queue heads in the asynchronous schedule. It is essentially the specification of an iterator for consecutive bus transactions to the same endpoint. All of the rules for managing bus transactions and the results of those as defined in Execute transaction apply. This feature merely specifies how many consecutive times the host controller is permitted to execute from the same queue head before moving to the next queue head in the Asynchronous List. This feature should allow the host controller to attain better bus utilization for those devices that are capable of moving data at maximum rate, while at the same time providing a fair service to all endpoints.

A host controller exports its capability to support this feature to system software by setting the *Asynchronous Schedule Park Capability* bit in the USB_n_HCCPARAMs register to one. This information keys system software that the *Asynchronous Schedule Park Mode Enable* and *Asynchronous Schedule Park Mode Count* fields in the USB_n_USBCMD register are modifiable. System software enables the feature by writing a one to the *Asynchronous Schedule Park Mode Enable* bit.

When park-mode is not enabled (for example *Asynchronous Schedule Park Mode Enable* bit in the USB_n_USBCMD register is zero), the host controller must not execute more than one bus transaction per high-speed queue head, per traversal of the asynchronous schedule. When park-mode is enabled, the host controller must not apply the feature to a queue head whose *EPS* field indicates a Low/Full-speed device (for example only one bus transaction is allowed from each Low/Full-speed queue head per traversal of the asynchronous schedule). Park-mode may only be applied to queue heads in the Asynchronous schedule whose *EPS* field indicates that it is a high-speed device.

The host controller must apply park mode to queue heads whose *EPS* field indicates a high-speed endpoint. The maximum number of consecutive bus transactions a host controller may execute on a high-speed queue head is determined by the value in the *Asynchronous Schedule Park Mode Count* field in the USB_n_USBCMD register. Software must not set *Asynchronous Schedule Park Mode Enable* bit to one and also set *Asynchronous Schedule Park Mode Count* field to zero. The resulting behavior is not defined. An example behavioral example describes the operational requirements for the host controller implementing park-mode. This feature does not affect how the host controller handles the bus transaction as defined in Execute transaction . It only effects how many consecutive bus transactions for the current queue head can be executed. All boundary conditions, error detection and reporting applies as usual. This feature is similar in concept to the use of the *Mult* field for high-bandwidth Interrupt for queue heads in the Periodic Schedule.

The host controller effectively loads an internal down-counter *PM-Count* from *Asynchronous Schedule Park Mode Count* when *Asynchronous Schedule Park Mode Enable* bit is one, and a high-speed queue head is first fetched and meets all the criteria for executing a bus transaction. After the bus transaction, *PM-Count* is decremented. The host controller may continue to execute bus transactions from the current queue head until *PM-Count* goes to zero, an error is detected, the buffer for the current transfer is exhausted or the endpoint responds with a flow-control or STALL handshake.

The following table summarizes the responses that effect whether the host controller continues with another bus transaction for the current queue head.

Table 840. Actions for Park Mode, based on Endpoint Response and Residual Transfer State

PID	Endpoint Response	Transfer State after Transaction		Action
		PM-Count	Bytes to Transfer	
IN	DATA[0,1] w/ Maximum Packet sized data	Not zero	Not Zero	Allowed to perform another bus transaction.[1,2]
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't care	Move to next QH.
	DATA[0,1] w/ short packet	Don't care	Don't care	Retire qTD and move to next QH.
	NAK	Don't care	Don't care	Move to next QH.
	STALL, XactErr	Don't care	Don't care	Move to next QH.
OUT	ACK	Not zero	Not Zero	Allowed to perform another bus transaction.[2]
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't' care	Move to next QH.
	NYET, NAK	Don't care	Don't care	Move to next QH.
	STALL, XactErr	Don't care	Don't care	Move to next QH
PING	ACK	Not Zero	Not Zero	Allowed to perform another bus transaction. [2]
	NAK	Don't care	Don't care	Move to next QH
	STALL, XactErr	Don't care	Don't care	Move to next QH

1. The host controller may continue to execute bus transactions from the current high-speed queue head (if *PM-Count* is not equal to zero), if a PID mismatch is detected (for example expected DATA1 and received DATA0, or visa-versa).
2. This specification does not *require* that the host controller execute another bus transaction when *PM-Count* is non-zero. Implementations are encouraged to make appropriate complexity and performance trade-offs.

56.3.2.1.10.4 Write back qTD

This state is entered from the Execute Transaction state when the *Active* bit is set to zero. The source data for the write-back is the transfer results area of the queue head overlay area (see Table 840). The host controller uses the *Current qTD Pointer* field as the target address for the qTD. The queue head transfer result area is written back to the transfer result area of the target qTD. This state is also referred to as: qTD retirement. The fields that must be written back to the source qTD include *Total Bytes to Transfer*, *Cerr*, and *Status*.

The duration of this state depends on when the qTD write-back is committed.

56.3.2.1.10.5 Follow queue head horizontal pointer

The host controller must use the horizontal pointer in the queue head to the next schedule data structure when any of the following conditions exist:

- If the *Active* bit is one on exit from the Execute Transaction state, or
- When the host controller exits the Write Back qTD state, or
- If the Advance Queue state fails to advance the queue because the target qTD is not active, or
- If the *Halted* bit is one on exit from the Fetch QH state.

There is no functional requirement that the host controller wait until the current transaction is complete before using the horizontal pointer to read the next linked data structure. However, it must wait until the current transaction is complete before executing the next data structure.

56.3.2.1.10.6 Buffer pointer list use for data streaming with qTDs

A qTD has an array of buffer pointers, which is used to reference the data buffer for a transfer. This specification requires that the buffer associated with the transfer be *virtually contiguous*. This means: if the buffer spans more than one physical page, it must obey the following rules (the figure below illustrates an example):

- The first portion of the buffer must begin at some offset in a page and extend through the end of the page.
- The remaining buffer cannot be allocated in small chunks scattered around memory. For each 4 K chunk beyond the first page, each buffer portion matches to a full 4 K page. The final portion, which may only be large enough to occupy a portion of a page, must start at the top of the page and be contiguous within that page.

The buffer pointer list in the qTD is long enough to support a maximum transfer size of 20 KB. This case occurs when all five buffer pointers are used and the first offset is zero. A qTD handles a 16 KB buffer with any starting buffer alignment.

The host controller uses the field *C_Page* field as an index value to determine which buffer pointer in the list should be used to start the current transaction. The host controller uses a different buffer pointer for each physical page of the buffer. This is always true, even if the buffer is physically contiguous.

The host controller must detect when the current transaction spans a page boundary and automatically move to the next available buffer pointer in the page pointer list. The next available pointer is reached by incrementing *C_Page* and pulling the next page pointer from the list. Software must ensure there are sufficient buffer pointers to move the amount of data specified in the *Bytes to Transfer* field.

The following figure illustrates a nominal example of how System software would initialize the buffer pointers list and the *C_Page* field for a transfer size of 16383 bytes. *C_Page* is set to zero. The upper 20-bits of Page 0 references the start of the physical page. *Current Offset* (the lower 12-bits of queue head Dword 7) holds the offset in the page for example 2049 (for example 4096-2047). The remaining page pointers are set to reference the beginning of each subsequent 4 K page.

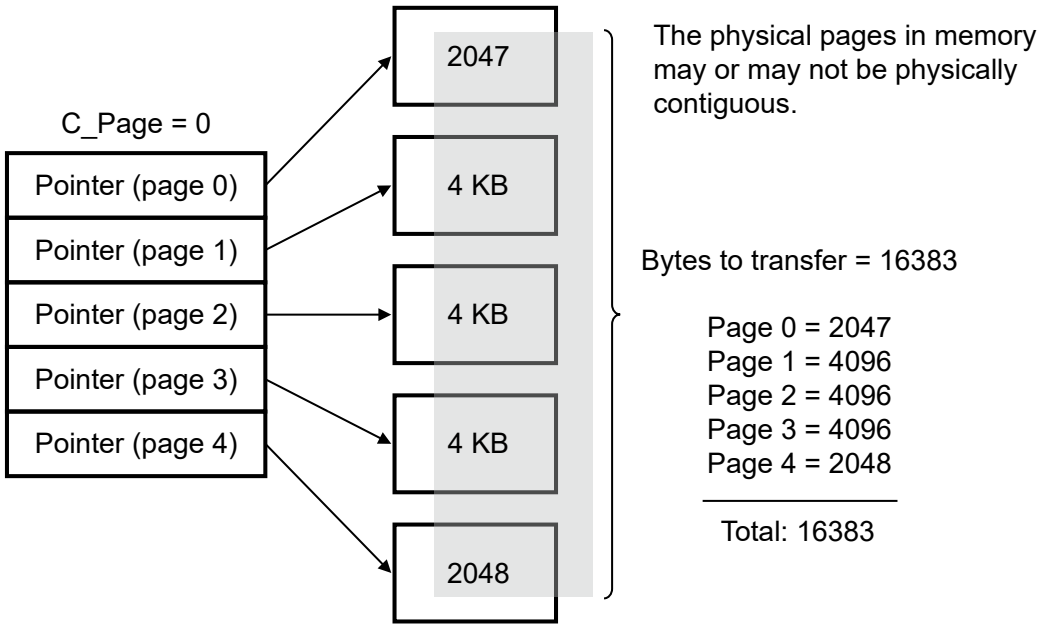


Figure 413. Example Mapping of qTD Buffer Pointers to Buffer Pages

For the first transaction on the qTD (assuming a 512-byte transaction), the host controller uses the first buffer pointer (page 0 because *C_Page* is set to zero) and concatenates the *Current Offset* field. The 512 bytes are moved during the transaction, the *Current Offset* and *Total Bytes to Transfer* are adjusted by 512 and written back to the queue head working area.

During the 4th transaction, the host controller needs 511 bytes in page 0 and one byte in page 1. The host controller increments *C_Page* (to 1) and use the page 1 pointer to move the final byte of the transaction. After the 4th transaction, the active page pointer is the page 1 pointer and *Current Offset* has rolled to one, and both are written back to the overlay area. The transactions continue for the rest of the buffer, with the host controller automatically moving to the next page pointer (that is *C_Page*) when necessary. The three conditions for how the host controller handles *C_Page*:

- The current transaction does not span a page boundary. The value of *C_Page* is not adjusted by the host controller.
- The current transaction does span a page boundary. The host controller must detect the page cross condition and advance to the next buffer while streaming data to/from the USB.
- The current transaction completes on a page boundary (that is the last byte moved for the current transaction is the last byte in the page for the current page pointer). The host controller must increment *C_Page* before writing back status for the transaction.

NOTE

The only valid adjustment the host controller may make to *C_Page* is to increment by one.

56.3.2.1.10.7 Adding interrupt queue heads to the periodic schedule

The link path(s) from the periodic frame list to a queue head establishes in which frames a transaction can be executed for the queue head. Queue heads are linked into the periodic schedule so they are polled at the appropriate rate. System software sets a bit in a queue head's *S-Mask* to indicate which micro-frame with-in 1 msec period a transaction should be executed for the queue head. Software must ensure that all queue heads in the periodic schedule have *S-Mask* set to a non-zero value. An *S-mask* with zero value in the context of the periodic schedule yields undefined results.

If the desired poll rate is greater than one frame, system software can use a combination of queue head linking and *S-Mask* values to spread interrupts of equal poll rates through the schedule so that the periodic bandwidth is allocated and managed in the most efficient manner possible. Some examples are illustrated in the following table.

Table 841. Example Periodic Reference Patterns for Interrupt Transfers with 2ms Poll Rate

Frame # Reference Sequence	Description
0, 2, 4, 6, 8, and so on *S-Mask* = 01h	A queue head for the *bInterval* of 2 msec (16 micro-frames) is linked into the periodic schedule so that it is reachable from the periodic frame list locations indicated in the previous column. In addition, the *S-Mask* field in the queue head is set to 01h, indicating that the transaction for the endpoint should be executed on the bus during micro-frame 0 of the frame.
0, 2, 4, 6, 8, and so on *S-Mask* = 02h	Another example of a queue head with a *bInterval* of 2 msec is linked into the periodic frame list at exactly the same interval as the previous example. However, the *S-Mask* is set to 02h indicating that the transaction for the endpoint should be executed on the bus during micro-frame 1 of the frame.

56.3.2.1.10.8 Managing transfer complete interrupts from queue heads

The host controller sets an interrupt to be signaled at the next interrupt threshold when the completed transfer (qTD) has an *Interrupt on Complete* (*IOC*) bit set to one, or whenever a transfer (qTD) completes with a short packet. If system software needs multiple qTDs to complete a client request (that is like a control transfer) the intermediate qTDs do not require interrupts. System software may only need a single interrupt to notify it that the complete buffer has been transferred. System software may set IOC's to occur more frequently. A motivation for this may be that it wants early notification so that interface data structures can be re-used in a timely manner.

56.3.2.1.11 Ping control

USB 2.0 defines an addition to the protocol for high-speed devices called Ping. Ping is required for all USB 2.0 High-speed bulk and control endpoints. Ping is not allowed for a split-transaction stream. This extension to the protocol eliminates the bad side-effects of Naking OUT endpoints. The *Status* field has a *Ping State* bit, which the host controller uses to determine the *next* actual PID it uses in the next transaction to the endpoint (see the table below).

The Ping State bit is only managed by the host controller for queue heads that meet the following criteria:

- Queue head is not an interrupt and
- *EPS* field equals High-Speed and
- *PIDCode* field equals OUT

The following table illustrates the state transition table for the host controller's responsibility for maintaining the PING protocol. Refer to Chapter 8 in the USB Specification Revision 2.0 for detailed description on the Ping protocol.

Table 842. Ping Control State Transition Table

Event			
Current	Host	Device	Next
Do Ping	PING	Nak	Do Ping
Do Ping	PING	Ack	Do OUT
Do Ping	PING	XactErr [1]	Do Ping
Do Ping	PING	Stall	N/C [2] Do
OUT	OUT	Nak	Do Ping

Table continues on the next page...

Table 842. Ping Control State Transition Table (continued)

Do OUT	OUT	Nyet	Do Ping
Do OUT	OUT	Ack	Do OUT
Do OUT	OUT	XactErr[1]	Do Ping
Do OUT	OUT	Stall	N/C [2]

1. Transaction Error (XactErr) is any time the host misses the handshake.
2. No transition change required for the Ping State bit. The Stall handshake results in the endpoint being halted (for example Active set to zero and Halt set to one). Software intervention is required to restart queue. 3 A Nyet response to an OUT means that the device has accepted the data, but cannot receive any more at this time. Host must advance the transfer state and additionally, transition the Ping State bit to Do Ping. The Ping State bit has the following encoding:

Table 843. Ping State bit Encoding

Value	Meaning
0B	Do OUT The host controller uses an OUT PID during the next bus transaction to this endpoint.
1B	Do Ping The host controller uses a PING PID during the next bus transaction to this endpoint.

The defined ping protocol (see USB 2.0 Specification, Chapter 8) allows the host to be *imprecise* on the initialization of the ping protocol (that is start in *Do OUT* when we don't know whether there is space on the device or not). The host controller manages the *Ping State* bit. System software sets the initial value in the queue head when it initializes a queue head. The host controller preserves the *Ping State* bit across all queue advancements. This means that when a new qTD is written into the queue head overlay area, the previous value of the *Ping State* bit is preserved.

56.3.2.1.12 Split transactions

USB 2.0 defines extensions to the bus protocol for managing USB 1.x data streams through USB 2.0 Hubs. This section describes how the host controller uses the interface data structures to manage data streams with full- and low-speed devices, connected below USB 2.0 hub, utilizing the split transaction protocol. Refer to USB 2.0 Specification for the complete definition of the split transaction protocol. Full- and Low-speed devices are enumerated identically as high-speed devices, but the transactions to the Full- and Low-speed endpoints use the split-transaction protocol on the high-speed bus. The split transaction protocol is an encapsulation of (or wrapper around) the Full- or Low-speed transaction. The high-speed wrapper portion of the protocol is addressed to the USB 2.0 Hub and Transaction Translator below which the Full- or Low-speed device is attached.

The EHCI interface uses dedicated data structures for managing full-speed isochronous data streams (see Split transaction isochronous transfer descriptor (siTD)). Control, Bulk and Interrupt are managed using the queuing data structures (see Queue head). The interface data structures need to be programmed with the device address and the Transaction Translator number of the USB 2.0 Hub operating as the Low-/Full-speed host controller for this link. The following sections describe the details of how the host controller must process and manage the split transaction protocol.

56.3.2.1.12.1 Split transactions for asynchronous transfers

A queue head in the asynchronous schedule with an *EPS* field indicating a full-or low-speed device indicates to the host controller that it must use split transactions to stream data for this queue head. All full-speed bulk and full-, low-speed control are managed through queue heads in the asynchronous schedule.

Software must initialize the queue head with the appropriate device address and port number for the transaction translator that is serving as the full/low-speed host controller for the links connecting the endpoint. Software must also initialize the split transaction state bit (*SplitXState*) to Do-Start-Split. Finally, if the endpoint is a control endpoint, then system software must set the *Control Transfer Type* (*C*) bit in the queue head to one. If this is not a control transfer type endpoint, the *C* bit must be initialized by software

to be zero. This information is used by the host controller to properly set the Endpoint Type (ET) field in the split transaction bus token. When the *C* bit is zero, the split transaction token's ET field is set to indicate a bulk endpoint. When the *C* bit is one, the split transaction token's ET field is set to indicate a control endpoint. Refer to Chapter 8 of USB Specification Revision 2.0 for details.

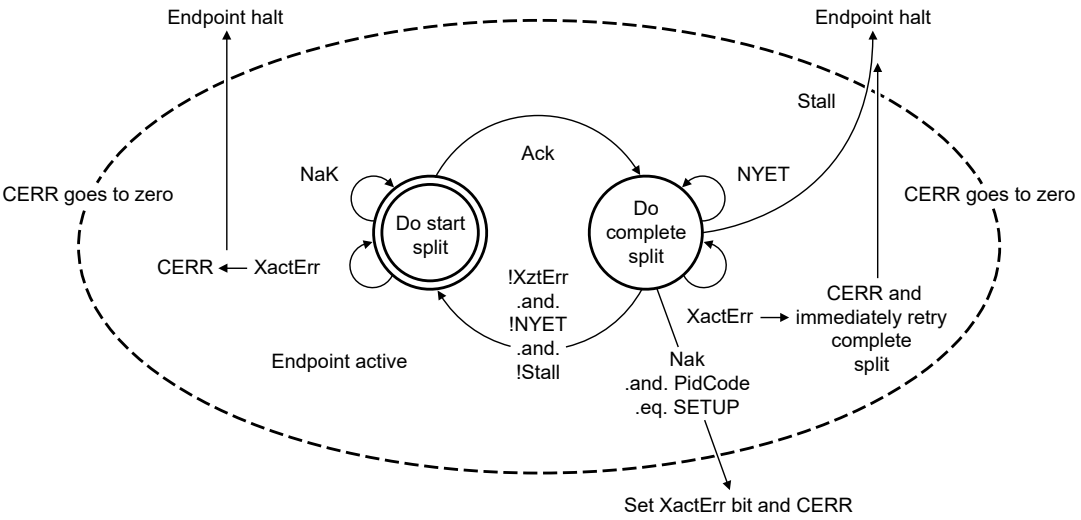


Figure 414. Host Controller Asynchronous Schedule Split-Transaction State Machine

56.3.2.1.12.1.1 Asynchronous - do start split

This is the state which software must initialize a full- or low-speed asynchronous queue head. This state is entered from the Do Complete Split state only after a complete-split transaction receives a valid response from the transaction translator that is not a Nyet handshake.

For queue heads in this state, the host controller executes a start-split transaction to the appropriate transaction translator. If the bus transaction completes without an error and *PidCode* indicates an IN or OUT transaction, then the host controller reloads the error counter (*CErr*). If it is a successful bus transaction and the *PidCode* indicates a SETUP, the host controller does not reload the error counter. If the transaction translator responds with a Nak, the queue head is left in this state, and the host controller proceeds to the next queue head in the asynchronous schedule.

If the host controller times out the transaction (no response, or bad response) the host controller decrements *Cerr* and proceeds to the next queue head in the asynchronous schedule.

56.3.2.1.12.1.2 Asynchronous - do complete split

This state is entered from the Do Start Split state only after a start-split transaction receives an Ack handshake from the transaction translator.

For queue heads in this state, the host controller executes a complete-split transaction to the appropriate transaction translator. If the transaction translator responds with a Nyet handshake, the queue head is left in this state, the error counter is reset and the host controller proceeds to the next queue head in the asynchronous schedule. When a Nyet handshake is received for a bus transaction where the queue head's *PidCode* indicates an IN or OUT, the host controller reloads the error counter (*CErr*). When a Nyet handshake is received for a complete-split bus transaction where the queue head's *PidCode* indicates a SETUP, the host controller must not adjust the value of *CErr*.

Independent of *PIDCode*, the following responses have the effects:

- Transaction Error (XactErr). Timeout or data CRC failure, and so on. The error counter (*Cerr*) is decremented by one and the complete split transaction is *immediately* retried (if possible). If there is not enough time in the micro-frame to execute the retry, the host controller MUST ensure that the next time the host controller begins executing from the Asynchronous schedule, it must begin executing from this queue head. If another start-split (for some other endpoint) is sent to the transaction translator before the complete-split is really completed, the transaction translator could dump the results (which were never delivered to the host). This is why the core specification states the retries must be immediate. A

method to accomplish this behavior is to not advance the asynchronous schedule. When the host controller returns to the asynchronous schedule in the next micro-frame, the first transaction from the schedule is the retry for this endpoint.

If *Cerr* went to zero, the host controller must halt the queue.

- NAK. The target endpoint Nak'd the full- or low-speed transaction. The state of the transfer is not advanced and the state is exited. If the *PidCode* is a SETUP, then the Nak response is a protocol error. The *XactErr* status bit is set to one and the *CErr* field is decremented.
- STALL. The target endpoint responded with a STALL handshake. The host controller sets the *halt* bit in the status byte, retires the qTD but does not attempt to advance the queue.

If the *PidCode* indicates an IN, then any of following responses are expected:

- DATA0/1. On reception of data, the host controller ensures the PID matches the expected data toggle and checks CRC. If the packet is *good*, the host controller advances the state of the transfer, for example move the data pointer by the number of bytes received, decrement *BytesToTransfer* field by the number of bytes received, and toggle the *dt* bit. The host controller then exit this state. The response and advancement of transfer may trigger other processing events, such as retirement of the qTD and advancement of the queue.

If the data sequence PID does not match the expected, the data is ignored, the transfer state is not advanced and this state is exited. If the *PidCode* indicates an OUT/SETUP, then any of following responses are expected:

- ACK. The target endpoint accepted the data, so the host controller must advance the state of the transfer. The *Current Offset* field is incremented by *Maximum Packet Length* or *Bytes to Transfer*, whichever is less. The field *Bytes To Transfer* is decremented by the same amount and the data toggle bit (*dt*) is toggled. The host controller then exit this state.
- Advancing the transfer state may cause other processing events such as retirement of the qTD and advancement of the queue (see Managing control/bulk/interrupt transfers through queue heads).

56.3.2.1.12.2 Split transaction interrupt

Split-transaction Interrupt-IN/OUT endpoints are managed through the same data structures used for high-speed interrupt endpoints. They both co-exist in the periodic schedule. Queue heads/qTDs offer the set of features required for reliable data delivery, which is characteristic to interrupt transfer types. The split-transaction protocol is managed completely within this defined functional transfer framework. For example, for a high-speed endpoint, the host controller visits a queue head, execute a high-speed transaction (if criteria are met) and advance the transfer state (or not) depending on the results of the entire transaction. For low- and full-speed endpoints, the details of the *execution* phase are different (that is takes more than one bus transaction to complete), but the remainder of the operational framework is intact. This means that the transfer advancement, and so on, occurs as defined in Managing control/bulk/interrupt transfers through queue heads, but only occurs on the completion of a split transaction.

56.3.2.1.12.2.1 Split transaction scheduling mechanisms for interrupt

Full- and low-speed Interrupt queue heads have an *EPS* field indicating full- or low-speed and have a non-zero *S-mask* field. The host controller can detect this combination of parameters and assume the endpoint is a periodic endpoint. Low- and full-speed interrupt queue heads require the use of the split transaction protocol. The host controller sets the Endpoint Type (ET) field in the split token to indicate the transaction is an interrupt. These transactions are managed through a transaction translator's periodic pipeline. Software should not set these fields to indicate the queue head is an interrupt unless the queue head is used in the periodic schedule.

System software manages the per/transaction translator periodic pipeline by budgeting and scheduling exactly during which micro-frames the start-splits and complete-splits for each endpoint occurs. The characteristics of the transaction translator are such that the high-speed transaction protocol must execute during explicit micro-frames, or the data or response information in the pipeline is lost.

The following figure illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule and queue head data structure. The S and ^{C}X labels indicate micro-frames where software can schedule start-splits and complete splits (respectively).

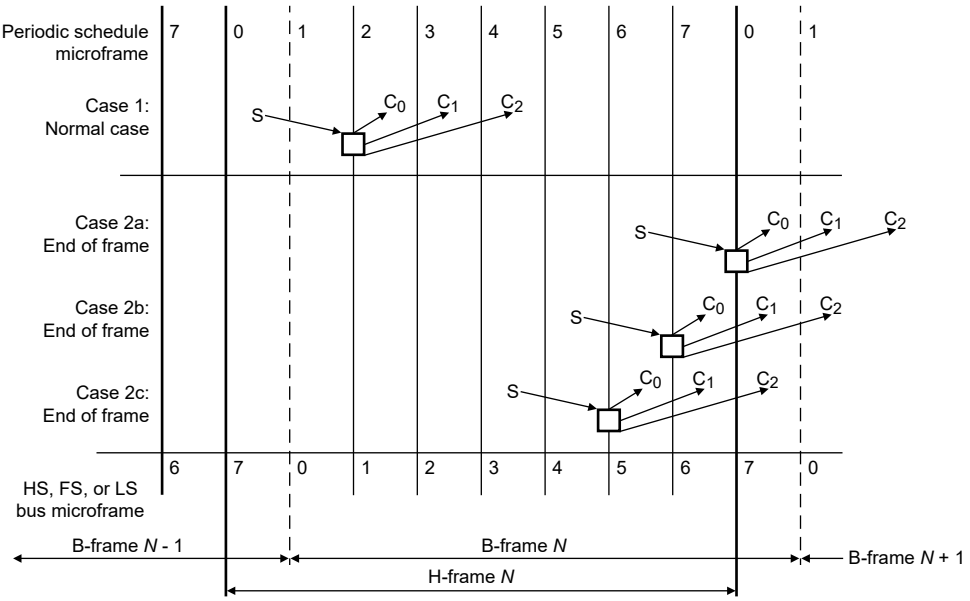


Figure 415. Split Transaction, Interrupt Scheduling Boundary Conditions

The scheduling cases are:

- Case 1: The normal scheduling case is where the entire split transaction is completely bounded by a frame (*H-Frame* in this case).
- Case 2a through Case 2c: The USB 2.0 Hub pipeline rules states clearly, when and how many complete-splits must be scheduled to account for earliest to latest execution on the full/low-speed link. The complete-splits may span the *H-Frame* boundary when the start-split is in micro-frame 4 or later. When this occurs, the *H-Frame* to *B-Frame* alignment requires that the queue head be reachable from consecutive periodic frame list locations. System software cannot build an efficient schedule that satisfies this requirement unless it uses FSTNs.

The figure below illustrates the general layout of the periodic schedule.

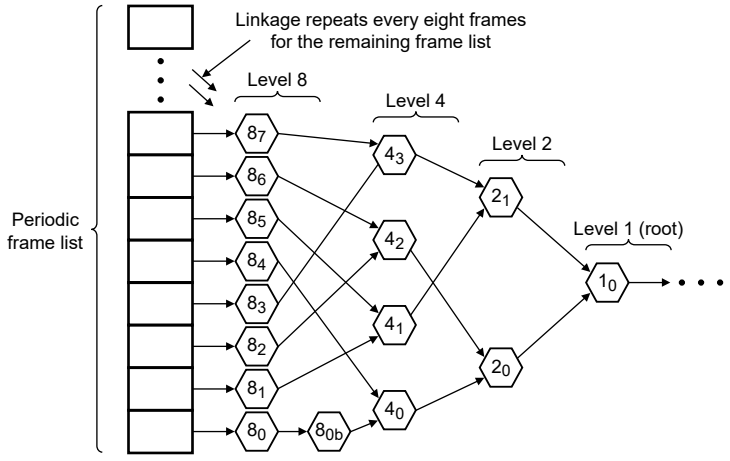


Figure 416. General Structure of EHCI Periodic Schedule Utilizing Interrupt Spreading

The periodic frame list is effectively the leaf level a binary tree, which is always traversed leaf to root. Each level in the tree corresponds to a 2^N poll rate. Software can efficiently manage periodic bandwidth on the USB by *spreading* interrupt queue heads

that have the same poll rate requirement across all the available paths from the frame list. For example, system software can schedule eight poll rate 8 queue heads and account for them once in the high-speed bus bandwidth allocation.

When an endpoint is allocated an execution footprint that spans a frame boundary, the queue head for the endpoint must be reachable from consecutive locations in the frame list. An example would be if 8_{0b} where such an endpoint. Without additional support on the interface, to get 8_{0b} reachable at the correct time, software would have to link 8_1 to 8_{0b}. It would then have to move 4_1 and everything linked after into the same path as 4_0. This upsets the integrity of the binary tree and disallows the use of the spreading technique.

FSTN data structures are used to preserve the integrity of the binary-tree structure and enable the use of the spreading technique. Host controller operational model for FSTNs defines the hardware and software operational model requirements for using FSTNs.

The following queue head fields are initialized by system software to instruct the host controller when to execute portions of the split-transaction protocol:

- *SplitXState*. This is single bit residing in the *Status* field of a queue head (see Table 889). This bit is used to track the current state of the split transaction.
- *Frame S-mask*. This is a bit-field where-in system software sets a bit corresponding to the micro-frame (within an *H-Frame*) that the host controller should execute a start-split transaction. This is always qualified by the value of the *SplitXState* bit in the *Status* field of the queue head. For example, referring to Figure 415, case one, the *S-mask* would have a value of 00000001b indicating that if the queue head is traversed by the host controller, and the *SplitXState* indicates Do_Start, and the current micro-frame as indicated by FRINDEX[2:0] is 0, then execute a start-split transaction.
- *Frame C-mask.* This is a bit-field where system software sets one or more bits corresponding to the micro-frames (within an *H-Frame*) that the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the *SplitXState* bit in the *Status* field of the queue head. For example, referring to Figure 415, case one, the *C-mask* would have a value of 00011100b indicating that if the queue head is traversed by the host controller, and the *SplitXState* indicates Do_Complete, and the current micro-frame as indicated by FRINDEX[2:0] is 2, 3, or 4, then execute a complete-split transaction. It is software's responsibility to ensure that the translation between *H-Frames* and *B-Frames* is correctly performed when setting bits in *S-mask* and *C-mask*

56.3.2.1.12.2.2 Host controller operational model for FSTNs

The FSTN data structure is used to manage Low/Full-speed interrupt queue heads that need to be reached from consecutive frame list locations (that is boundary cases 2a through 2c). An FSTN is essentially a *back pointer*, similar in intent to the back pointer field in the siTD data structure (see siTD back link pointer).

This feature provides software a simple primitive to save a schedule position, redirect the host controller to traverse the necessary queue heads in the previous frame, then restore the original schedule position and complete normal traversal.

The four components to the use of FSTNs:

- FSTN data structure.
- A *Save Place* indicator. This is always an FSTN with its *Back Path Link Pointer.T-bit* set to zero.
- A *Restore* indicator. This is always an FSTN with its *Back Path Link Pointer.T-bit* set to one.
- Host controller FSTN traversal rules.

When the host controller encounters an FSTN during micro-frames 2 through 7 it simply follows the node's *Normal Path Link Pointer* to access the next schedule data structure.

NOTE

The FSTN's *Normal Path Link Pointer.T-bit* may set to one, which the host controller must interpret as the end of periodic list mark.

When the host controller encounters a *Save-Place* FSTN in micro-frames 0 or 1, it saves the value of the *Normal Path Link Pointer* and set an internal flag indicating that it is executing in *Recovery Path* mode. *Recovery Path* mode modifies the host controller's rules for how it traverses the schedule and limits which data structures is considered for execution of bus transactions. The host controller continues executing in *Recovery Path* mode until it encounters a *Restore* FSTN or it determines that it has reached the end of the micro-frame (see details in the list below).

The rules for schedule traversal and limited execution while in *Recovery Path* mode are:

- Always follow the *Normal Path Link Pointer* when it encounters an FSTN that is a *Save-Place* indicator. The host controller must not recursively follow *Save-Place* FSTNs. Therefore, while executing in *Recovery Path* mode, it must never follow an FSTN's *Back Path Link Pointer*.
- Do not process an siTD or, iTD data structure. Simply follow its *Next Link Pointer*.
- Do not process a QH (Queue Head) whose *EPS* field indicates a high-speed device. Simply follow its *Horizontal Link Pointer*.
- When a QH's *EPS* field indicates a Full/Low-speed device, the host controller considers only it for execution if its *SplitXState* is DoComplete (note: this applies whether the *PID Code* indicates an IN or an OUT). See Execute transaction and Tracking split transaction progress for interrupt transfers for a complete list of additional conditions that must be met in general for the host controller to issue a bus transaction.
  - The host controller must not execute a Start-split transaction while executing in *Recovery Path* mode. See Periodic isochronous - do complete split for special handling when in *Recovery Path* mode.
- Stop traversing the *recovery path* when it encounters an FSTN that is a *Restore* indicator. The host controller unconditionally uses the saved value of the *Save-Place* FSTN's *Normal Path Link Pointer* when returning to the normal path traversal. The host controller must clear the context of executing a *Recovery Path* when it restores schedule traversal to the *Save-Place* FSTN's *Normal Path Link Pointer*.
- If the host controller determines that there is not enough time left in the micro-frame to complete processing of the periodic schedule, it abandons traversal of the recovery path, and clears the context of executing a recovery path. The result is that at the start of the next consecutive micro-frame, the host controller starts traversal at the frame list.

An example traversal of a periodic schedule that includes FSTNs is illustrated in the following figure.

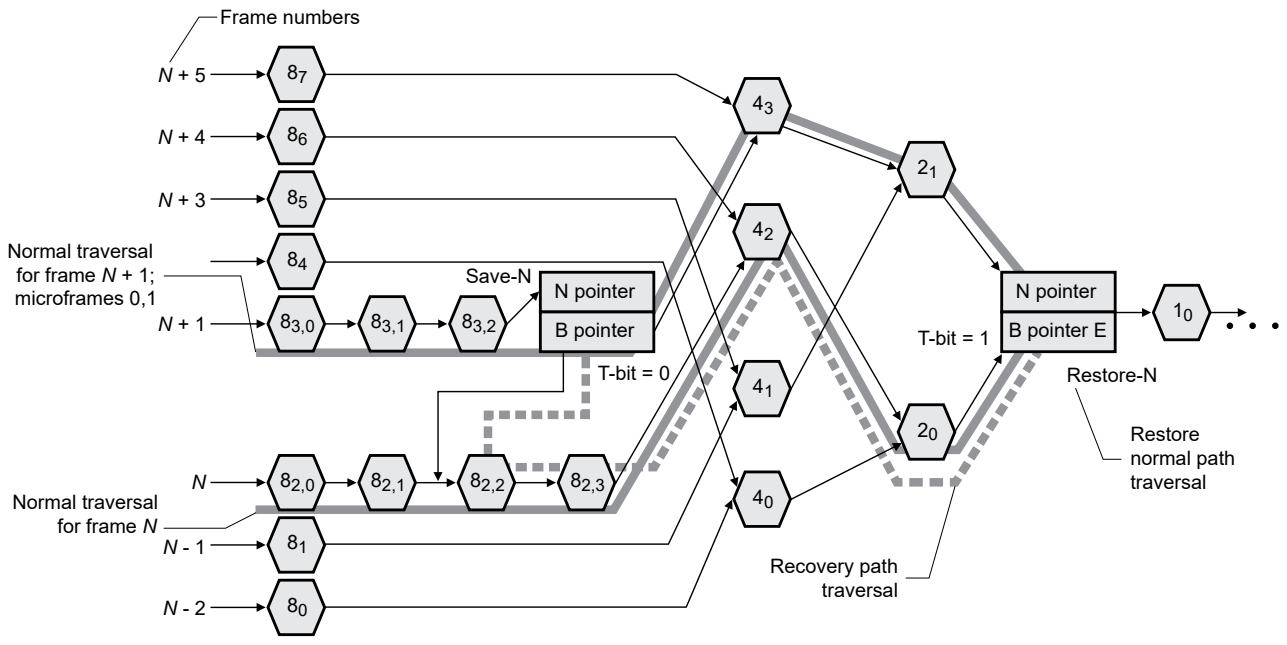


Figure 417. Example Host Controller Traversal of Recovery Path via FSTNs

In frame N+1 (micro-frames 0 and 1), when the host controller encounters Save-Path FSTN (Save-N), it observes that Save-N.Back Path Link Pointer.T-bit is zero (definition of a Save-Path indicator). The host controller saves the value of Save-N.Normal Path Link Pointer and follows Save-N.Back Path Link Pointer. At the same time, it sets an internal flag indicating that it is now in *Recovery Path* mode (the recovery path is annotated in the figure above with a large dashed line). The host controller continues traversing data structures on the recovery path and executing only those bus transactions as noted above, on the recovery path until it reaches Restore FSTN (Restore-N). Restore-N.Back Path Link Pointer.T-bit is set to one (definition

of a Restore indicator), so the host controller exits *Recovery Path* mode by clearing the internal *Recovery Path* mode flag and commences (restores) schedule traversal using the saved value of the *Save-Place* FSTN's *Normal Path Link Pointer* (for example Save-N.Normal Path Link Pointer). The nodes traversed during these micro-frames include: {$8_{3.0}$, $8_{3.1}$, $8_{3.2}$, Save-A, $8_{2.2}$, $8_{2.3}$, 4_2, 2_0, Restore-N, 4_3, 2_1, Restore-N, 1_0 …}. The nodes on the recovery-path are in bold. In frame N (micro-frames 0-7), for this example, the host controller traverses all of the schedule data structures utilizing the *Normal Path Link Pointers* in any FSTNs it encounters. This is because the host controller has not yet encountered a *Save-Place* FSTN so it not executing in *Recovery Path* mode. When it encounters the *Restore* FSTN, (Restore-N), during micro-frames 0 and 1, it uses Restore-N.Normal Path Link Pointer to traverse to the next data structure (that is normal schedule traversal). This is because the host controller must use a Restore FSTN's *Normal Path Link Pointer* when not executing in a *Recovery-Path* mode. The nodes traversed during frame N include: {$8_{2.0}$, $8_{2.1}$, $8_{2.2}$, $8_{2.3}$, 4_2, 2_0, Restore-N, 1_0 …}.

In frame N+1 (micro-frames 2-7), when the host controller encounters Save-Path FSTN Save-N, it unconditionally follows Save-N.Normal Path Link Pointer. The nodes traversed during these micro-frames include: {$8_{3.0}$, $8_{3.1}$, $8_{3.2}$, Save-A, 4_3, 2_1, Restore-N, 1_0 …}.

56.3.2.1.12.2.3 Software operational model for FSTNs

Software must create a consistent, coherent schedule for the host controller to traverse. When using FSTNs, system software must adhere to the following rules:

- Each *Save-Place* indicator requires a matching *Restore* indicator.
  - The *Save-Place* indicator is an FSTN with a valid *Back Path Link Pointer and T-bit* equal to zero.
    - *Back Path Link Pointer.Typ* field must be set to indicate the referenced data structure is a queue head. The *Restore* indicator is an FSTN with its *Back Path Link Pointer.T-bit* set to one.
  - A *Restore* FSTN may be matched to one or more *Save-Place* FSTNs. For example, if the schedule includes a poll-rate 1 level, then system software only needs to place a *Restore* FSTN at the beginning of this list in order to match all possible *Save-Place* FSTNs.
- If the schedule does not have elements linked at a poll-rate level of one, and one or more *Save-Place* FSTNs are used, then System Software must ensure the *Restore* FSTN's *Normal Path Link Pointer*'s *T-bit* is set to one, as this is used to mark the end of the periodic list.
- When the schedule does have elements linked at a poll rate level of one, a *Restore* FSTN must be the first data structure on the poll rate one list. All traversal paths from the frame list converge on the poll-rate one list. System software must ensure that *Recovery Path* mode is exited before the host controller is allowed to traverse the poll rate level one list.
- A *Save-Place* FSTN's *Back Path Link Pointer* must reference a queue head data structure. The referenced queue head must be reachable from the previous frame list location. In other words, if the *Save-Place* FSTN is reachable from frame list offset N, then the FSTN's *Back Path Link Pointer* must reference a queue head that is reachable from frame list offset N-1.

Software should make the schedule as efficient as possible. What this means in this context is that software should have no more than one *Save-Place* FSTN reachable in any single frame. Note there is times when two (or more, depending on the implementation) could exist as full/low-speed footprints change with bandwidth adjustments. This could occur, for example when a bandwidth re-balance causes system software to move the *Save-Place* FSTN from one poll rate level to another. During the transition, software must preserve the integrity of the previous schedule until the new schedule is in place.

56.3.2.1.12.2.4 Tracking split transaction progress for interrupt transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where data is lost. For interrupt-IN transfers, data is lost when it makes it into the USB 2.0 hub, but the USB 2.0 host system is unable to get it from the USB 2.0 Hub and into the system before it expires from the transaction translator pipeline. When a lost data condition is detected, the queue must be halted, thus signaling system software to recover from the error. A data-loss condition exists whenever a start-split is issued, accepted, and successfully executed by the USB 2.0 Hub, but the complete-splits get unrecoverable errors on the high-speed link, or the complete-splits do not occur at the correct times. One reason complete-splits might not occur at the right time would be due to host-induced system hold-offs that cause the host controller to miss bus transactions because it cannot get timely access to the schedule in system memory.

The same condition can occur for an interrupt-OUT, but the result is not an endpoint halt condition, but rather effects only the progress of the transfer. The queue head has the following fields to track the progress of each split transaction. These fields are used to keep incremental state about which (and when) portions have been executed.

- *C-prog-mask*. This is an eight-bit bit-vector where the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in-order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur due to system hold-offs where the host controller cannot get to the memory-based schedule. *C-prog-mask* is a simple bit-vector that the host controller sets one of the *C-prog-*mask bits for each complete-split executed. The bit position is determined by the micro-frame number in which the complete-split was executed. The host controller always checks *C-prog-mask* before executing a complete-split transaction. If the previous complete-splits have not been executed then it means one (or more) have been skipped and data has potentially been lost.
- *FrameTag*. This field is used by the host controller during the complete-split portion of the split transaction to tag the queue head with the frame number (*H-Frame* number) when the next complete split must be executed.
- *S-bytes*. This field can be used to store the number of data payload bytes sent during the start-split (if the transaction was an OUT). The *S-bytes* field must be used to accumulate the data payload bytes received during the complete-splits (for an IN).

56.3.2.1.12.2.5 Split transaction execution state machine for interrupt

In the following presentation, all references to micro-frame are in the context of a micro-frame within an *H-Frame.* As with asynchronous Full- and Low-speed endpoints, a split-transaction state machine is used to manage the split transaction sequence. Aside from the fields defined in the queue head for scheduling and tracking the split transaction, the host controller calculates one internal mechanism that is also used to manage the split transaction. The internal calculated mechanism is:

- *cMicroFrameBit* is a single-bit encoding of the current micro-frame number. It is an eight-bit value calculated by the host controller at the beginning of every micro-frame. It is calculated from the three least significant bits of the *FRINDEX* register (that is, *cMicroFrameBit* = (1 shifted-left(*FRINDEX*[2:0])). The *cMicroFrameBit* has at most one bit asserted, which always corresponds to the current micro-frame number. For example, if the current micro-frame is 0, then *cMicroFrameBit* will equal 00000001b. The variable *cMicroFrameBit* is used to compare against the *S-mask* and *C-mask* fields to determine whether the queue head is marked for a start - or complete-split transaction for the current micro-frame.

The following figure illustrates the state machine for managing a complete interrupt split transaction. There are two phases to each split transaction. The first is a single start-split transaction, which occurs when the *SplitXState* is at Do_Start and the single bit in *cMicroFrameBit* has a corresponding bit active in *QH.S-mask*. The transaction translator does not acknowledge the receipt of the periodic start-split, so the host controller unconditionally transitions the state to Do_Complete. Due to the available jitter in the transaction translator pipeline, there will be more than one complete-split transaction scheduled by software for the Do_Complete state. This translates simply to the fact that there are multiple bits set to a one in the *QH.C-mask* field.

The host controller keeps the queue head in the Do_Complete state until the split transaction is complete (see definition below), or an error condition triggers the *three-strikes-rule* (for example, after the host tries the same transaction three times, and each encounters an error, the host controller will stop retrying the bus transaction and halt the endpoint, thus requiring system software to detect the condition and perform system-dependent recovery).

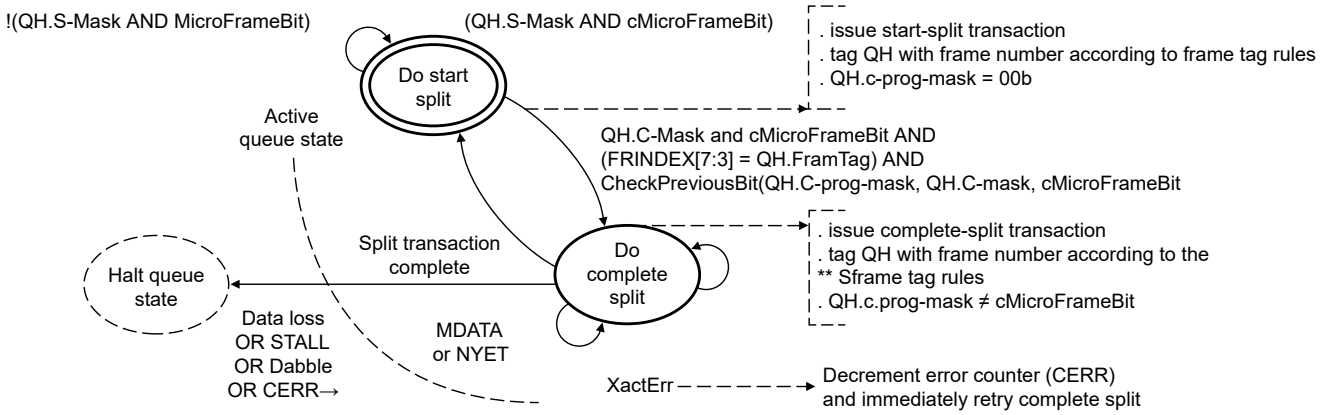


Figure 418. Split Transaction State Machine for Interrupt

See Previous Section for the frame tag management rules.

Periodic Interrupt - Do Start Split

This is the state software must initialize a full- or low-speed interrupt queue head *StartXState* bit. This state is entered from the Do_Complete Split state only after the split transaction is complete. This occurs when one of the following events occur: The transaction translator responds to a complete-split transaction with one of the following:

- NAK. A NAK response is a propagation of the full- or low-speed endpoint's NAK response.
- ACK. An ACK response is a propagation of the full- or low-speed endpoint's ACK response. Only occurs on an OUT endpoint.
- DATA 0/1. Only occurs for INs. Indicates that this is the last of the data from the endpoint for this split transaction.
- ERR. The transaction on the low-/full-speed link below the transaction translator had a failure (for example, timeout, bad CRC).
- NYET (and Last). The host controller issued the last complete-split and the transaction translator responded with a NYET handshake. This means that the start-split was not correctly received by the transaction translator, so it never executed a transaction to the full- or low-speed endpoint, see Section Periodic isochronous - do complete split for the definition of 'Last'.

Each time the host controller visits a queue head in this state (once within the Execute Transaction state), it performs the following test to determine whether to execute a start-split.

- *QH.S-mask* is bit-wise anded with *cMicroFrameBit*.

If the result is non-zero, then the host controller will issue a start-split transaction. If the *PIDCode* field indicates an IN transaction, the host controller must zero-out the *QH.S-bytes* field. After the split-transaction has been executed, the host controller sets up state in the queue head to track the progress of the complete-split phase of the split transaction. Specifically, it records the expected frame number into *QH.FrameTag* field (see Section), set *C-prog-mask* to zero (00h), and exits this state. Note that the host controller must not adjust the value of *CErr* as a result of completion of a start-split transaction.

Periodic Interrupt - Do Complete Split

This state is entered unconditionally from the Do Start Split state after a start-split transaction is executed on the bus. Each time the host controller visits a queue head in this state (once within the Execute Transaction state), it checks to determine whether a complete-split transaction should be executed now.

There are four tests to determine whether a complete-split transaction should be executed.

- Test A. *cMicroFrameBit* is bit-wise anded with *QH.C-mask* field. A non-zero result indicates that software scheduled a complete-split for this endpoint, during this micro-frame.
- Test B. *QH.FrameTag* is compared with the current contents of *FRINDEX*[7:3]. An equal indicates a match.

- Test C. The complete-split progress bit vector is checked to determine whether the previous bit is set, indicating that the previous complete-split was appropriately executed. An example algorithm for this test is provided below:

```
Algorithm Boolean CheckPreviousBit(QH.C-prog-mask, QH.C-mask, cMicroFrameBit)
Begin
-- Return values:
-- TRUE - no error
-- FALSE - error
--
Boolean rvalue = TRUE;
previousBit = cMicroframeBit logical-rotate-right(1)
-- Bit-wise anding previousBit with C-mask indicates
-- whether there was an intent
-- to send a complete split in the previous micro-frame. So,
-- if the
-- 'previous bit' is set in C-mask, check C-prog-mask to
-- make sure it
-- happened.
If (previousBit bitAND QH.C-mask)then
                              If not(previousBit bitAND QH.C-prog-mask) then
                    rvalue = FALSE;
          End if
End If
-- If the C-prog-mask already has a one in this bit position,
-- then an aliasing
-- error has occurred. It will probably get caught by the
-- FrameTag Test, but
-- at any rate it is an error condition that as detectable here
-- should not allow
-- a transaction to be executed.
If (cMicroFrameBit bitAND QH.C-prog-mask) then
 rvalue = FALSE;
End if
return (rvalue)
End Algorithm
```

- Test D. Check to see if a start-split should be executed in this micro-frame. Note this is the same test performed in the Do Start Split state (see Section Periodic isochronous - do start split). Whenever it evaluates to TRUE and the controller is NOT processing in the context of a *Recovery Path* mode, it means a start-split should occur in this micro-frame. Test D and Test A evaluating to TRUE at the same time is a system software error. Behavior is undefined.

If (A .and. B .and. C .and. not(D)) then the host controller will execute a complete-split transaction. When the host controller commits to executing the complete-split transaction, it updates *QH.C-prog-mask* by bit-ORing with *cMicroFrameBit*. On completion of the complete-split transaction, the host controller records the result of the transaction in the queue head and sets *QH.FrameTag* to the expected *H-Frame* number (see Section). The effect to the state of the queue head and thus the state of the transfer depends on the response by the transaction translator to the complete-split transaction. The following responses have the effects (note that any responses that result in decrementing of the *CErr* will result in the queue head being halted by the host controller if the result of the decrement is zero):

- NYET (and Last). On each NYET response, the host controller checks to determine whether this is the last complete-split for this split transaction. Last is defined in this context as the condition where all of the scheduled complete-splits have been executed. If it is the last complete-split (with a NYET response), then the transfer state of the queue head is not advanced (never received any data) and this state exited. The transaction translator must have responded to all the complete-splits with NYETs, meaning that the start-split issued by the host controller was not received. The start-split should be retried at the next poll period.
- The test for whether this is the Last complete split can be performed by XOR *QH.C-mask* with *QH.C-prog-mask*. If the result is all zeros, then all complete-splits have been executed. When this condition occurs, the *XactErr* status bit is set to a one, and the *CErr* field is decremented.

- NYET (and not Last). See above description for testing for Last. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for *C-prog-mask* and *FrameTag*) and stay in this state. The host controller must not adjust *CErr* on this response.
- Transaction Error (XactErr). Timeout, data CRC failure, and so on. The *CErr* field is decremented and the *XactErr* bit in the *Status* field is set to a one. The complete split transaction is *immediately* retried (if *Cerr* is non-zero). If there is not enough time in the micro-frame to complete the retry and the endpoint is an IN, or *CErr* is decremented to a zero from a one, the queue is halted. If there is not enough time in the micro-frame to complete the retry and the endpoint is an OUT and *CErr* is not zero, then this state is exited (that is, return to Do Start Split). This results in a retry of the entire OUT split transaction, at the next poll period. Refer to Chapter 11 Hubs (specifically the section full- and low-speed Interrupts) in the USB Specification Revision 2.0 for detailed requirements on why these errors must be immediately retried.
- ACK. This can only occur if the target endpoint is an OUT. The target endpoint ACK'd the data and this response is a propagation of the endpoint ACK up to the host controller. The host controller must advance the state of the transfer. The *Current Offset* field is incremented by *Maximum Packet Length* or *Bytes to Transfer*, whichever is less. The field *Bytes To Transfer* is decremented by the same amount. And the data toggle bit (*dt*) is toggled. The host controller will then exit this state for this queue head. The host controller must reload *CErr* with maximum value on this response. Advancing the transfer state may cause other process events such as retirement of the qTD and advancement of the queue (see Section Managing control/bulk/interrupt transfers through queue heads).
- MDATA. This response will only occur for an IN endpoint. The transaction translator responded with zero or more bytes of data and an MDATA PID. The incremental number of bytes received is accumulated in *QH.S-bytes*. The host controller must not adjust *CErr* on this response.
- DATA0/1. This response may only occur for an IN endpoint. The number of bytes received is added to the accumulated byte count in *QH.S-bytes*. The state of the transfer is advanced by the result and the host controller will exit this state for this queue head.
- Advancing the transfer state may cause other processing events such as retirement of the qTD and advancement of the queue (see Section Managing control/bulk/interrupt transfers through queue heads).
- If the data sequence PID does not match the expected, the entirety of the data received in this split transaction is ignored, the transfer state is not advanced and this state is exited.
- NAK. The target endpoint Nak'd the full- or low-speed transaction. The state of the transfer is not advanced, and this state is exited. The host controller must reload *CErr* with maximum value on this response.
- ERR. There was an error during the full- or low-speed transaction. The ERR status bit is set to a one, *Cerr* is decremented, the state of the transfer is not advanced, and this state is exited.
- STALL. The queue is halted (an exit condition of the Execute Transaction state). The status field bits: *Active* bit is set to zero and the *Halted* bit is set to a one and the qTD is retired. Responses which are not enumerated in the list or which are received out of sequence are illegal and may result in undefined host controller behavior. The other possible combinations of tests A, B, C, and D may indicate that data or response was lost. The table below lists the possible combinations and the appropriate action.

Table 844. Interrupt IN/OUT Do Complete Split State Execution Criteria

Condition	Action	Description
not(A) not(D)	Ignore QHD	Neither a start nor complete-split is scheduled for the current micro-frame. Host controller should continue walking the schedule.
A not(C)	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	Progress bit check failed. These mean a complete-split has been missed. There is the possibility of lost data. If *PIDCode* is an IN, then the Queue head must be halted.

Table continues on the next page...

Table 844. Interrupt IN/OUT Do Complete Split State Execution Criteria (continued)

		If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect *CERR*. In either case, set the *Missed Micro-frame* bit in the status field to a one.
A not(B) C	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	*QH.FrameTag* test failed. This means that exactly one or more *H-Frames* have been skipped. This means complete-splits and have missed. There is the possibility of lost data. If *PIDCode* is an IN, then the Queue head must be halted. If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect *CERR*. In either case, set the *Missed Micro-frame* bit in the status field to a one.
A B C not(D)	Execute complete-split	This is the non-error case where the host controller executes a complete-split transaction.
D	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	This is a degenerate case where the start-split was issued, but all of the complete-splits were skipped and all possible intervening opportunities to detect the missed data failed to fire. If *PIDCode* is an IN, then the Queue head must be halted. If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect *CERR*. In either case, set the *Missed Micro-frame* bit in the status field to a one. Note: When executing in the context of a *Recovery Path* mode, the host controller is allowed to process the queue head and take the actions indicated above, or it may wait until the queue head is visited in the normal processing mode. Regardless, the host controller must not execute a start-split in the context of an executing in a *Recovery Path* mode.

Managing QH.FrameTag Field

The *QH.FrameTag* field in a queue head is completely managed by the host controller. The rules for setting *QH.FrameTag* are simple:

- Rule 1: If transitioning from Do Start Split to Do Complete Split and the current value of *FRINDEX*[2:0] is 6 *QH.FrameTag* is set to *FRINDEX*[7:3] + 1. This accommodates split transactions whose start-split and complete-splits are in different *H-Frames* (case 2a, see Figure 415).
- Rule 2: If the current value of *FRINDEX*[2:0] is 7, *QH.FrameTag* is set to *FRINDEX*[7:3] + 1. This accommodates staying in Do Complete Split for cases 2a, 2b, and 2c (Figure 415).
- Rule 3: If transitioning from Do_Start Split to Do Complete Split and the current value of *FRINDEX*[2:0] is not 6, or currently in Do Complete Split and the current value of (*FRINDEX*[2:0]) is not 7, *FrameTag* is set to *FRINDEX*[7:3]. This accommodates all other cases (Figure 415).

56.3.2.1.12.2.6 Rebalancing the periodic schedule

System software must occasionally adjust a periodic queue head's S-mask and C-mask fields during operation. This need occurs when adjustments to the periodic schedule create a new bandwidth budget and one or more queue head's are assigned new execution footprints (that is, new S-mask and C-mask values).

It is imperative that System software must not update these masks to new values in the midst of a split transaction. In order to avoid any race conditions with the update, the EHCI HC provides a simple assist to system software. System software sets the *Inactivate-on-next-Transaction* (*I*) bit to a one to signal the host controller that it intends to update the S-mask and C-mask on this queue head. System software will then wait for the host controller to observe the *I-bit* is a one and transition the *Active* bit to a zero. The rules for how and when the host controller sets the *Active* bit to zero are enumerated below:

- If the *Active* bit is a zero, no action is taken. The host controller does not attempt to advance the queue when the *I-bit* is a one.
- If the *Active* bit is a one and the *SplitXState* is DoStart (regardless of the value of *S-mask*), the host controller will simply set *Active* bit to a zero. The host controller is not required to write the transfer state back to the *current* qTD. Note that if the *S-mask* indicates that a start-split is scheduled for the current micro-frame, the host controller must not issue the start-split bus transaction. It must set the *Active* bit to zero.

System software must save transfer state before setting the *I-bit* to a one. This is required so that it can correctly determine what transfer progress (if any) occurred after the *I-bit* was set to a one and the host controller executed its final bus-transaction and set *Active* to a zero.

After system software has updated the S-mask and C-mask, it must then reactivate the queue head. Because the *Active* bit and the *I-bit* cannot be updated with the same write, system software needs to use the following algorithm to coherently re-activate a queue head that has been stopped via the *I-bit*.

1. Set the *Halted* bit to a one, then
2. Set the *I-bit* to a zero, then
3. Set the *Active* bit to a one and the *Halted* bit to a zero in the same write.

Setting the *Halted* bit to a one inhibits the host controller from attempting to advance the queue between the time the *I-bit* goes to a zero and the *Active* bit goes to a one.

56.3.2.1.12.3 Split transaction isochronous

Full-speed isochronous transfers are managed using the split-transaction protocol through a USB 2.0 transaction translator in a USB2.0 Hub. The EHCI controller utilizes siTD data structure to support the special requirements of isochronous split-transactions. This data structure uses the scheduling model of isochronous TDs (iTD, Section Isochronous (high-speed) transfer descriptor (iTD)) (see Section Managing isochronous transfers using iTDs for the operational model of iTDs) with the contiguous data feature provided by queue heads. This simple arrangement allows a single isochronous scheduling model and adds the additional feature that all data received from the endpoint (per split transaction) must land into a contiguous buffer.

56.3.2.1.12.3.1 Split transaction scheduling mechanisms for isochronous

Full-speed isochronous transactions are managed through a transaction translator's periodic pipeline. As with full- and low-speed interrupt, system software manages each transaction translator's periodic pipeline by budgeting and scheduling exactly during which micro-frames the start-splits and complete-splits for each full-speed isochronous endpoint occur. The requirements described in Section Split transaction scheduling mechanisms for interrupt apply. The following figure illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule. The ^{S}X and ^{C}X labels indicate micro-frames where software can schedule start- and complete-splits (respectively). The *H-Frame* boundaries are marked with a large, solid bold vertical line. The *B-Frame* boundaries are marked with a large, bold, dashed line. The bottom of the figure illustrates the relationship of an siTD to the *H-Frame*.

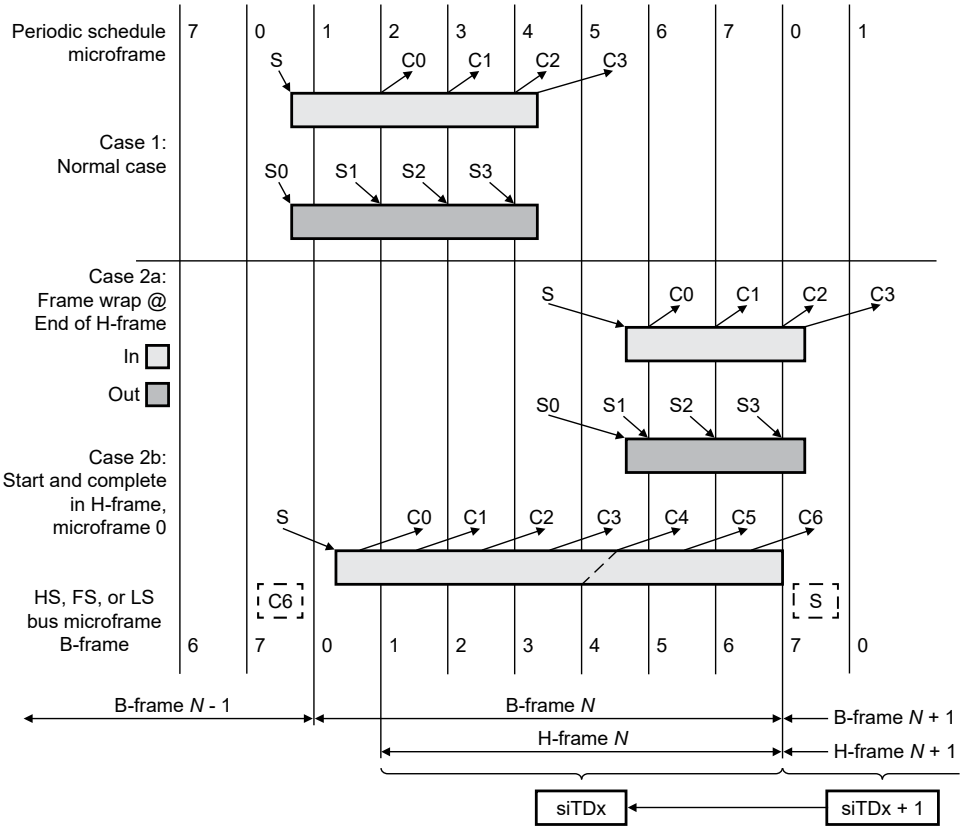


Figure 419. Split Transaction, Isochronous Scheduling Boundary Conditions

When the endpoint is an isochronous OUT, there are only start-splits, and no complete-splits. When the endpoint is an isochronous IN, there is at most one start-split and one to *N* complete-splits. The scheduling boundary cases are:

- *Case 1*: The entire split transaction is completely bounded by an *H-Frame.* For example: the start-splits and complete-splits are all scheduled to occur in the same *H-Frame*.
- *Case 2a*: This boundary case is where one or more (at most two) complete-splits of a split transaction IN are scheduled across an *H-Frame* boundary. This can only occur when the split transaction has the possibility of moving data in *B-Frame,* micro-frames 6 or 7 (*H-Frame* micro-frame 7 or 0). When an *H-Frame* boundary wrap condition occurs, the scheduling of the split transaction spans more than one location in the periodic list.(For example, it takes two siTDs in adjacent periodic frame list locations to fully describe the scheduling for the split transaction.)
- Although the scheduling of the split transaction may take two data structures, all of the complete-splits for each full-speed IN isochronous transaction must use only one data pointer. For this reason, siTDs contain a back pointer, the use of which is described below.
- Software must never schedule full-speed isochronous OUTs across an *H-Frame* boundary.
- *Case 2b*: This case can only occur for a very large isochronous IN. It is the only allowed scenario where a start-split and complete-split for the same endpoint can occur in the same micro-frame. Software must enforce this rule by scheduling the large transaction first. Large is defined to be anything larger than 579 bytes maximum packet size.

A subset of the same mechanisms employed by full- and low-speed interrupt queue heads are employed in siTDs to schedule and track the portions of isochronous split transactions. The following fields are initialized by system software to instruct the host controller when to execute portions of the split transaction protocol.

- *SplitXState*. This is a single bit residing in the *Status* field of an siTD (see Figure 420). This bit is used to track the current state of the split transaction. The rules for managing this bit are described in Section Split transaction execution state machine for interrupt.

- *Frame S-mask*. This is a bit-field where-in system software sets a bit corresponding to the micro-frame (within an *H-Frame*) that the host controller should execute a start-split transaction. This is always qualified by the value of the *SplitXState* bit. For example, referring to the IN example in Figure 419, case one, the *S-mask* would have a value of 00000001b indicating that if the siTD is traversed by the host controller, and the *SplitXState* indicates Do Start Split, and the current micro-frame as indicated by USB_n_FRINDEX[2:0] is 0, then execute a start-split transaction.
- *Frame C-mask*. This is a bit-field where system software sets one or more bits corresponding to the micro-frames (within an *H-Frame*) that the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the *SplitXState* bit. For example, referring to the IN example in Figure 419, case one, the *C-mask* would have a value of 00111100b indicating that if the siTD is traversed by the host controller, and the *SplitXState* indicates Do Complete Split, and the current micro-frame as indicated by *USB_n_FRINDEX*[2:0] is 2, 3, 4, or 5, then execute a complete-split transaction.
- *Back Pointer*. This field in a siTD is used to complete an IN split-transaction using the previous *H-Frame*'s siTD. This is only used when the scheduling of the complete-splits span an *H-Frame* boundary.

There exists a one-to-one relationship between a high-speed isochronous split transaction (including all start- and complete-splits) and one full-speed isochronous transaction. An siTD contains (amongst other things) buffer state and split transaction scheduling information. An siTD's buffer state always maps to one full-speed isochronous data payload. This means that for any full-speed transaction payload, a single siTD's data buffer must be used. This rule applies to both IN an OUTs. An siTD's scheduling information usually also maps to one high-speed isochronous split transaction. The exception to this rule is the *H-Frame* boundary wrap cases mentioned above.

The siTD data structure describes at most, one frame's worth of high-speed transactions and that description is strictly bounded within a frame boundary. The figure below illustrates some examples. On the top are examples of the full-speed transaction footprints for the boundary scheduling cases described above. In the middle are time-frame references for both the *B-Frames* (HS/FS/LS Bus) and the *H-Frames*. On the bottom is illustrated the relationship between the scope of an siTD description and the time references. Each *H-Frame* corresponds to a single location in the periodic frame list. The implication is that each siTD is reachable from a single periodic frame list location at a time.

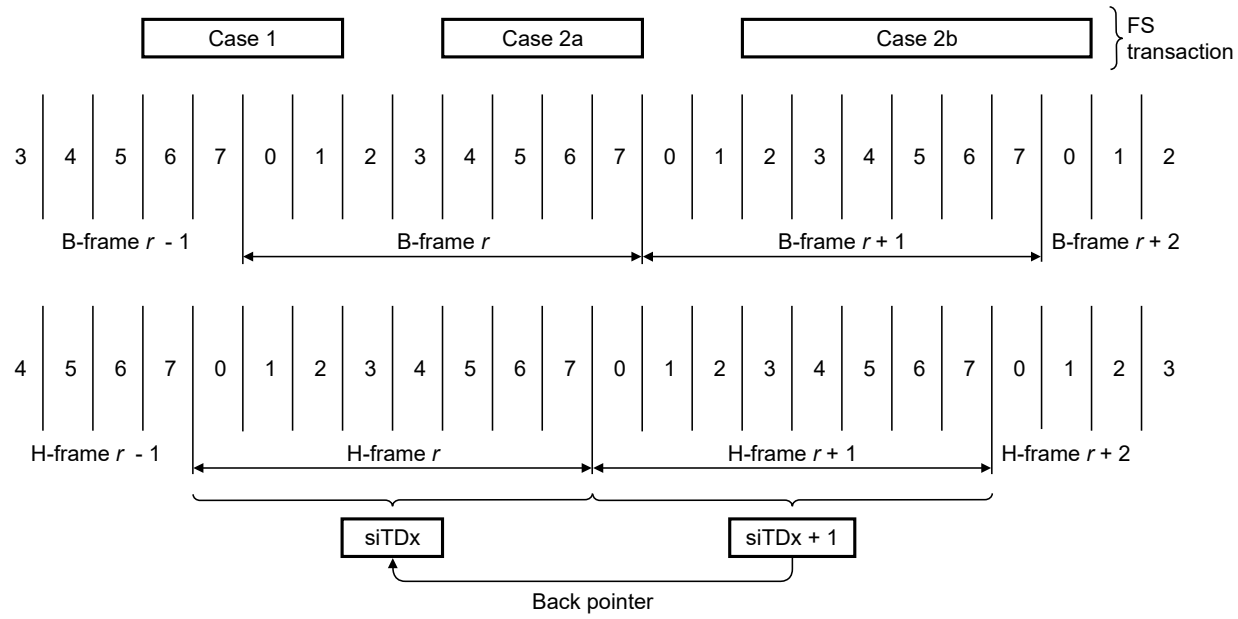


Figure 420. siTD Scheduling Boundary Examples

Each case is described below:

- *Case 1*: One siTD is sufficient to describe and complete the isochronous split transaction because the whole isochronous split transaction is tightly contained within a single *H-Frame*.

- *Case 2a, 2b*: Although both INs and OUTs can have these footprints, OUTs always take only one siTD to schedule. However, INs (for these boundary cases) require two siTDs to complete the scheduling of the isochronous split transaction $siTD_X$ is used to always issue the start-split and the first *N* complete-splits. The full-speed transaction (for these cases) can deliver data on the full-speed bus segment during micro-frame 7 of $H\text{-}Frame_{Y+1}$, or micro-frame 0 of $H\text{-}Frame_{Y+2}$. The complete splits are scheduled using $siTD_{X+2}$ (not shown). The complete-splits to extract this data must use the buffer pointer from $siTD_{X+1}$. The only way for the host controller to reach $siTD_{X+1}$ from $H\text{-}Frame_{Y+2}$is to use $siTD_{X+2}$'s back pointer. The host controller rules for when to use the back pointer are described is Section Periodic isochronous - do complete split .

Software must apply the following rules when calculating the schedule and linking the schedule data structures into the periodic schedule:

- Software must ensure that an isochronous split-transaction is started so that it will complete before the end of the *B-Frame*.
- Software must ensure that for a single full-speed isochronous endpoint, there is never a start-split and complete-split in *H-Frame, micro-frame 1*. This is mandated as a rule so that case 2a and case 2b can be discriminated. According to the core USB specification, the long isochronous transaction illustrated in Case 2b, could be scheduled so that the start-split was in micro-frame 1 of *H-Frame* N and the last complete-split would need to occur in micro-frame 1 of H-Frame N+1. However, it is impossible to discriminate between cases 2a and case 2b, which has significant impact on the complexity of the host controller.

56.3.2.1.12.3.2 Tracking split transaction progress for isochronous transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where device to host data is lost. Isochronous endpoints do not employ the concept of a halt on error, however the host is required to identify and report per-packet errors observed in the data stream. This includes schedule traversal problems (skipped micro-frames), timeouts, and corrupted data received.

In similar kind to interrupt split-transactions, the portions of the split transaction protocol must execute in the micro-frames they are scheduled. The queue head data structure used to manage full- and low-speed interrupt has several mechanisms for tracking when portions of a transaction have occurred. Isochronous transfers use siTDs, for their transfers, and the data structures are only reachable via the schedule in the exact micro-frame in which they are required (so all the mechanism employed for tracking in queue heads is not required for siTDs). Software has the option of reusing siTD several times in the complete periodic schedule. However, it must ensure that the results of split transaction N are consumed and the siTD reinitialized (activated) before the host controller gets back to the siTD (in a future micro-frame).

Split-transaction isochronous OUTs utilize a low-level protocol to indicate which portions of the split transaction data have arrived. Control over the low-level protocol is exposed in an siTD via the fields *Transaction Position* (*TP*) and *Transaction Count* (*T-count*). If the entire data payload for the OUT split transaction is larger than 188 bytes, there will be more than one start-split transaction, each of which require proper annotation. If host hold-offs occur, then the sequence of annotations received from the host will not be complete, which is detected and handled by the transaction translator. See Section Periodic isochronous - do start split for a description on how these fields are used during a sequence of start-split transactions.

The fields *siTD.T-Count* and *siTD.TP* are used by the host controller to drive and sequence the transaction position annotations. It is the responsibility of system software to properly initialize these fields in each siTD. Once the budget for a split-transaction isochronous endpoint is established, *S-mask*, *T-Count*, and *TP* initialization values for all the siTD associated with the endpoint are constant. They remain constant until the budget for the endpoint is recalculated by software and the periodic schedule adjusted.

For IN-endpoints, the transaction translator simply annotates the response data packets with enough information to allow the host controller to identify the last data. As with split transaction Interrupt, it is the host controller's responsibility to detect when it has missed an opportunity to execute a complete-split. The following field in the siTD is used to track and detect errors in the execution of a split transaction for an IN isochronous endpoint.

- *C-prog-mask*. This is an eight-bit bit-vector where the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in-order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur due to system hold-offs where the host controller cannot get to the memory-based schedule. *C-prog-mask* is a simple bit-vector that the host controller sets a bit for each complete-split executed. The bit position is determined by the

micro-frame (USB_n_FRINDEX[2:0]) number in which the complete-split was executed. The host controller always checks *C-prog-mask* before executing a complete-split transaction. If the previous complete-splits have not been executed, then it means one (or more) have been skipped and data has potentially been lost. System software is required to initialize this field to zero before setting an siTD's *Active* bit to a one.

If a transaction translator returns with the final data before all of the complete-splits have been executed, the state of the transfer is advanced so that the remaining complete-splits are not executed. Refer to Section Asynchronous - do complete split for a description on how the state of the transfer is advanced. It is important to note that an IN siTD is retired based solely on the responses from the Transaction Translator to the complete-split transactions. This means, for example, that it is possible for a transaction translator to respond to a complete-split with an MDATA PID. The number of bytes in the MDATA's data payload could cause the siTD field *Total Bytes to Transfer* to decrement to zero. This response can occur, before all of the scheduled complete-splits have been executed. In other interface, data structures (for example, high-speed data streams through queue heads), the transition of *Total Bytes to Transfer* to zero signals the end of the transfer and results in setting of the *Active* bit to zero. However, in this case, the result has not been delivered by the Transaction Translator and the host must continue with the next complete-split transaction to extract the residual transaction state. This scenario occurs because of the pipeline rules for a Transaction Translator (see Chapter 11 of the Universal Serial Bus Revision 2.0). In summary the periodic pipeline rules require that on a micro-frame boundary, the Transaction Translator will hold the final two bytes received (if it has not seen an End Of Packet (EOP)) in the full-speed bus pipe stage and gives the remaining bytes to the high-speed pipeline stage. At the micro-frame boundary, the Transaction Translator could have received the entire packet (including both CRC bytes) but not received the packet EOP. In the next micro-frame, the Transaction Translator will respond with an MDATA and send all of the data bytes (with the two CRC bytes being held in the full-speed pipeline stage). This could cause the siTD to decrement its *Total Bytes to Transfer* field to zero, indicating it has received all expected data. The host must still execute one more (scheduled) complete-split transaction in order to extract the results of the full-speed transaction from the Transaction Translator (for example, the Transaction Translator may have detected a CRC failure, and this result must be forwarded to the host).

If the host experiences hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous OUT, then the protocol to the transaction translator will not be consistent and the transaction translator will detect and react to the problem. Likewise, for host hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous IN, the *C-prog-mask* is used by the host controller to detect errors. However, if the host experiences a hold-off that causes it to skip all of an siTD, or an siTD expires during a host hold off (for example, a hold-off occurs and the siTD is no longer reachable by the host controller in order for it to report the hold-off event), then system software must detect that the siTDs have not been processed by the host controller (that is, state not advanced) and report the appropriate error to the client driver.

56.3.2.1.12.3.3 Split transaction execution state machine for isochronous

In the following presentation, all references to micro-frame are in the context of a micro-frame within an *H-Frame.*

If the *Active* bit in the *Status* byte is a zero, the host controller will ignore the siTD and continue traversing the periodic schedule. Otherwise the host controller will process the siTD as specified below. A split transaction state machine is used to manage the split-transaction protocol sequence. The host controller uses the fields defined in Section Tracking split transaction progress for interrupt transfers , plus the variable *cMicroFrameBit* defined in Section Split transaction execution state machine for interrupt to track the progress of an isochronous split transaction. The figure below illustrates the state machine for managing an siTD through an isochronous split transaction. Bold, dotted circles denote the state of the *Active* bit in the *Status* field of a siTD. The Bold, dotted arcs denote the transitions between these states. Solid circles denote the states of the split transaction state machine and the solid arcs denote the transitions between these states. Dotted arcs and boxes reference actions that take place either as a result of a transition or from being in a state.

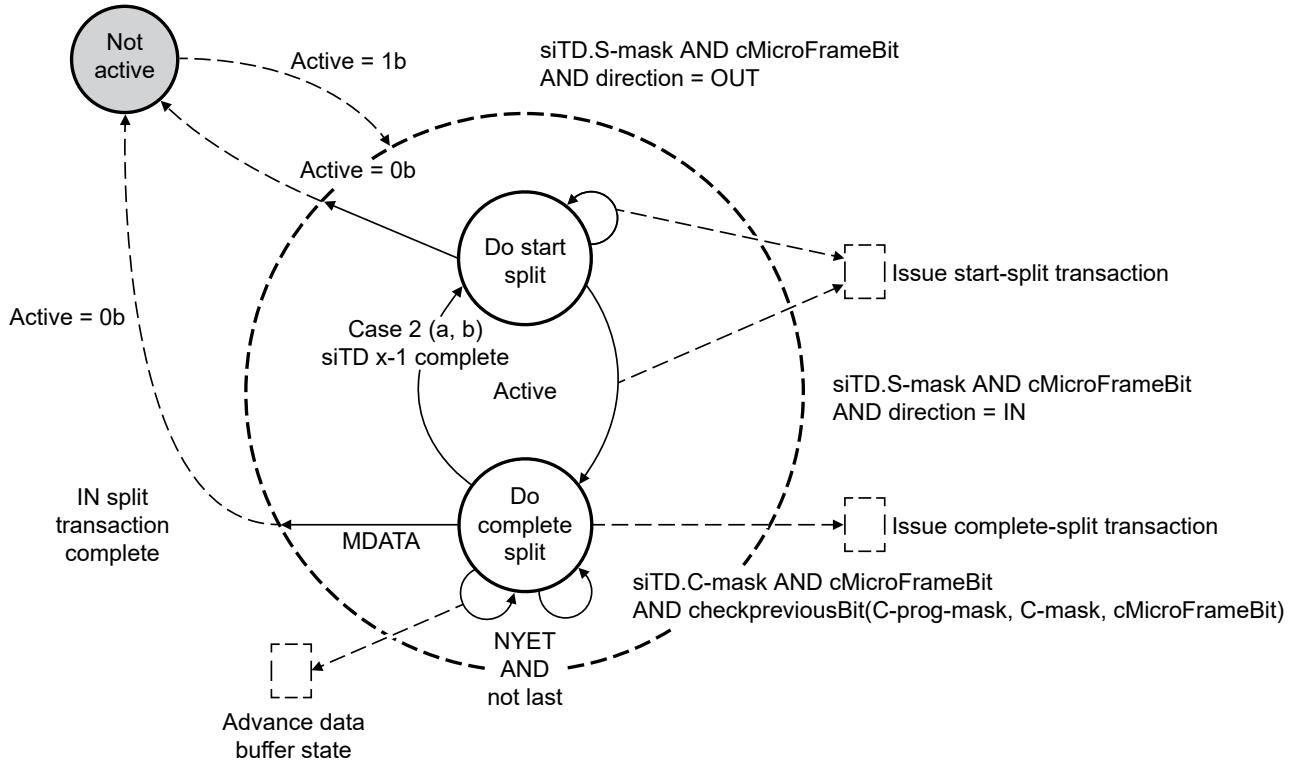


Figure 421. Split Transaction State Machine for Isochronous

56.3.2.1.12.3.4 Periodic isochronous - do start split

Isochronous split transaction OUTs use only this state. An siTD for a split-transaction isochronous IN is either initialized to this state, or the siTD transitions to this state from Do Complete Split when a case 2a (IN) or 2b scheduling boundary isochronous split-transaction completes.

Each time the host controller reaches an active siTD in this state, it checks the *siTD.S-mask* against *cMicroFrameBit*. If there is a one in the appropriate position, the siTD will execute a start-split transaction. By definition, the host controller cannot *reach* an siTD at the wrong time. If the *I/O* field indicates an IN, then the start-split transaction includes only the extended token plus the full-speed token. Software must initialize the *siTD.Total Bytes To Transfer* field to the number of bytes expected. This is usually the maximum packet size for the full-speed endpoint. The host controller exits this state when the start-split transaction is complete.

The remainder of this section is specific to an isochronous OUT endpoint (that is, the *I/O* field indicates an OUT). When the host controller executes a start-split transaction for an isochronous OUT it includes a data payload in the start-split transaction. The memory buffer address for the data payload is constructed by concatenating *siTD.Current Offset* with the page pointer indicated by the page selector field (*siTD.P*). A zero in this field selects Page 0 and a 1 selects Page 1. During the start-split for an OUT, if the data transfer crosses a page boundary during the transaction, the host controller must detect the page cross, update the *siTD.P*-bit from a zero to a one, and begin using the *siTD.Page 1* with *siTD.Current Offset* as the memory address pointer. The field *siTD.TP* is used to annotate each start-split transaction with the indication of which part of the split-transaction data the current payload represents (ALL, BEGIN, MID, END). In all cases the host controller simply uses the value in *siTD.TP* to mark the start-split with the correct transaction position code.

T-Count is always initialized to the number of start-splits for the current frame. *TP* is always initialized to the first required transaction position identifier. The scheduling boundary case (see Figure 420) is used to determine the initial value of *TP*. The initial cases are summarized in the following table.

Table 845. Initial Conditions for OUT siTD's TP and T-count Fields

Case	T-count	TP	Description
1, 2a	=1	ALL	When the OUT data payload is less than (or equal to) 188 bytes, only one start-split is required to move the data. The one start-split must be marked with an ALL.
1, 2a	!=1	BEGIN	When the OUT data payload is greater than 188 bytes more than one start-split must be used to move the data. The initial start-split must be marked with a BEGIN.

After each start-split transaction is complete, the host controller updates *T-Count* and *TP* appropriately so that the next start-split is correctly annotated.

The table below illustrates all of the *TP* and *T-count* transitions, which must be accomplished by the host controller.

.

Table 846. Transaction Position (TP)/Transaction Count (T-Count) Transition Table

TP	T-count next	TP next	Description
ALL	0	N/A	Transition from ALL, to done.
BEGIN	1	END	Transition from BEGIN to END. Occurs when *T-count* starts at 2.
BEGIN	!=1	MID	Transition from BEGIN to MID. Occurs when *T-count* starts at greater than 2.
MID	!=1	MID	*TP* stays at MID while *T-count* is not equal to 1 (that is, greater than 1). This case can occur for any of the scheduling boundary cases where the *T-count* starts greater than 3.
MID	1	END	Transition from MID to END. This case can occur for any of the scheduling boundary cases where the *T-count* starts greater than 2.

The start-split transactions do not receive a handshake from the transaction translator, so the host controller always advances the transfer state in the siTD after the bus transaction is complete. To advance the transfer state the following operations take place:

- The *siTD.Total Bytes To Transfer* and the *siTD.Current Offset* fields are adjusted to reflect the number of bytes transferred.
- The *siTD.P* (page selector) bit is updated appropriately.
- The *siTD.TP* and *siTD.T-count* fields are updated appropriately as defined in Table 846.

These fields are then written back to the memory based siTD. The *S-mask* is fixed for the life of the current budget. As mentioned above, *TP* and *T-count* are set specifically in each siTD to reflect the data to be sent from this siTD. Therefore, regardless of the value of *S-mask*, the actual number of start-split transactions depends on *T-count* (or equivalently, *Total Bytes to Transfer*). The host controller must set the *Active* bit to a zero when it detects that all of the schedule data has been sent to the bus. The preferred method is to detect when *T-Count* decrements to zero as a result of a start-split bus transaction. Equivalently, the host controller can detect when *Total Bytes to Transfer* decrements to zero. Either implementation must ensure that if the initial condition is *Total Bytes to Transfer* equal to zero and *T-count* is equal to a one, then the host controller will issue a single start-split, with a zero-length data payload. Software must ensure that *TP*, *T-count*, and *Total Bytes to Transfer* are set to deliver the appropriate number of bus transactions from each siTD. An inconsistent combination will yield undefined behavior.

If the host experiences hold-offs that cause the host controller to skip, start-split transactions for an OUT transfer, the state of the transfer will not progress appropriately. The transaction translator will observe protocol violations in the arrival of the start-splits for the OUT endpoint (that is, the transaction position annotation will be incorrect as received by the transaction translator).

Example scenarios are described in Section Split transaction for isochronous - processing examples.

A host controller implementation can optionally track the progress of an OUT split transaction by setting appropriate bits in the *siTD.C-prog-mask* as it executes each scheduled start-split. The *checkPreviousBit*() algorithm defined in Periodic isochronous - do complete split can be used prior to executing each start-split to determine whether start-splits were skipped. The host controller can use this mechanism to detect missed micro-frames. It can then set the siTD's *Active* bit to zero and stop execution of this siTD. This saves on both memory and high-speed bus bandwidth.

56.3.2.1.12.3.5 Periodic isochronous - do complete split

This state is only used by a split-transaction isochronous IN endpoint. This state is entered unconditionally from the Do Start State after a start-split transaction is executed for an IN endpoint. Each time the host controller visits an siTD in this state, it conducts a number of tests to determine whether it should execute a complete-split transaction. The individual tests are listed below. The sequence they are applied depends on which micro-frame the host controller is currently executing which means that the tests might not be applied until after the siTD referenced from the back pointer has been fetched.

- Test A. *cMicroFrameBit* is bit-wise anded with *siTD.C-mask* field. A non-zero result indicates that software scheduled a complete-split for this endpoint, during this micro-frame. This test is always applied to a newly fetched siTD that is in this state.
- Test B. The *siTD.C-prog-mask* bit vector is checked to determine whether the previous complete splits have been executed. An example algorithm is below (this is slightly different than the algorithm used in Section Periodic isochronous - do start split). The sequence in which this test is applied depends on the current value of USB_n_FRINDEX[2:0]. If USB_n_FRINDEX[2:0] is 0 or 1, it is not applied until the back pointer has been used. Otherwise it is applied immediately.

```
Algorithm Boolean CheckPreviousBit(siTD.C-prog-mask, siTD.C-mask, cMicroFrameBit)
Begin
          Boolean rvalue = TRUE;
          previousBit = cMicroFrameBit rotate-right(1)
          -- Bit-wise anding previousBit with C-mask indicates whether there was an intent
          -- to send a complete split in the previous micro-frame. So, if the
          -- 'previous bit' is set in C-mask, check C-prog-mask to make sure it
          -- happened.
          if previousBit bitAND siTD.C-mask then
                    if not (previousBit bitAND siTD.C-prog-mask) then
                              rvalue = FALSE
                    End if
          End if
          Return rvalue
End Algorithm
```

If Test A is true and USB_n_FRINDEX[2:0] is zero or one, then this is a case 2a or 2b scheduling boundary (see Figure 419). See Section Periodic isochronous - do start split for details in handling this condition.

If Test A and Test B evaluate to true, then the host controller will execute a complete-split transaction using the transfer state of the current siTD. When the host controller commits to executing the complete-split transaction, it updates *QH.C-prog-mask* by bit-ORing with *cMicroFrameBit*. The transfer state is advanced based on the completion status of the complete-split transaction. To advance the transfer state of an IN siTD, the host controller must:

- Decrement the number of bytes received from *siTD.Total Bytes To Transfer,*
- Adjust *siTD.Current Offset* by the number of bytes received,
- Adjust *siTD.P* (page selector) field if the transfer caused the host controller to use the next page pointer, and
- Set any appropriate bits in the *siTD.Status* field, depending on the results of the transaction.

Note that if the host controller encounters a condition where *siTD.Total Bytes To Transfer* is zero, and it receives more data, the host controller must not write the additional data to memory. The *siTD.Status.Active* bit must be set to zero and the *siTD.Status.Babble Detected* bit must be set to a one. The fields *siTD.Total Bytes To Transfer*, *siTD.Current Offset*, and *siTD.P* (page selector) are not required to be updated as a result of this transaction attempt.

The host controller must accept (assuming good data packet CRC and sufficient room in the buffer as indicated by the value of *siTD.Total Bytes To Transfer*) MDATA and DATA0/1 data payloads up to and including 192 bytes. A host controller implementation may optionally set *siTD.Status Active* to a zero and *siTD.Status.Babble Detected* to a one when it receives and MDATA or DATA0/1 with a data payload of more than 192 bytes. The following responses have the noted effects:

- ERR. The full-speed transaction completed with a time-out or bad CRC and this is a reflection of that error to the host. The host controller sets the *ERR* bit in the *siTD.Status* field and sets the *Active* bit to a zero.
- Transaction Error (XactErr). The complete-split transaction encounters a Timeout, CRC16 failure, and so on. The *siTD.Status* field *XactErr* field is set to a one and the complete-split transaction must be retried immediately. The host controller must use an internal error counter to count the number of retries as a counter field is not provided in the siTD data structure. The host controller will not retry more than two times. If the host controller exhausts the retries or the end of the micro-frame occurs, the *Active* bit is set to zero.
- DATAx (0 or 1). This response signals that the final data for the split transaction has arrived. The transfer state of the siTD is advanced and the *Active* bit is set to a zero. If the *Bytes To Transfer* field has not decremented to zero (including the reception of the data payload in the DATAx response), then less data than was expected, or allowed for was actually received. This *short packet* event does not set the USBINT status bit in the USBSTS register to a one. The host controller will not detect this condition.
- NYET (and Last). On each NYET response, the host controller also checks to determine whether this is the last complete-split for this split transaction. Last was defined in Section Periodic isochronous - do complete split . If it is the last complete-split (with a NYET response), then the transfer state of the siTD is not advanced (never received any data) and the *Active* bit is set to a zero. No bits are set in the *Status* field because this is essentially a skipped transaction. The transaction translator must have responded to all the scheduled complete-splits with NYETs, meaning that the start-split issued by the host controller was not received. This result should be interpreted by system software as if the transaction was completely skipped. The test for whether this is the last complete split can be performed by XORing C-mask with C-prog-mask. A zero result indicates that all complete-splits have been executed.
- MDATA (and Last). See above description for testing for Last. This can only occur when there is an error condition. Either there has been a babble condition on the full-speed link, which delayed the completion of the full-speed transaction, or software set up the *S-mask* and/or *C-mask*s incorrectly. The host controller must set *XactErr* bit to a one and the *Active* bit is set to a zero.
- NYET (and not Last). See above description for testing for Last. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for *C-prog-mask*) and stay in this state.
- MDATA (and not Last). The transaction translator responds with an MDATA when it has partial data for the split transaction. For example, the full-speed transaction data payload spans from micro-frame X to X+1 and during micro-frame X, the transaction translator will respond with an MDATA and the data accumulated up to the end of micro-frame X. The host controller advances the transfer state to reflect the number of bytes received.

If Test A succeeds, but Test B fails, it means that one or more of the complete-splits have been skipped. The host controller sets the *Missed Micro-Frame* status bit and sets the *Active* bit to a zero.

56.3.2.1.12.3.6 Complete-split for scheduling boundary cases 2a, 2b

Boundary cases 2a and 2b (INs only) (see Figure 419) require that the host controller use the transaction state context of the previous siTD to finish the split transaction. The table below enumerates the transaction state fields.

Table 847. Summary siTD Split Transaction State

Buffer State	Status	Execution Progress
Total Bytes To Transfer P (page select) Current Offset	All bits in the status field	C-prog-mask

Table continues on the next page...

Table 847. Summary siTD Split Transaction State (continued)

TP (transaction position) T-count (transaction count)		

NOTE

TP and *T-count* are used only for Host to Device (OUT) endpoints.

If software has budgeted the schedule of this data stream with a frame wrap case, then it must initialize the *siTD.Back Pointer* field to reference a valid siTD and will have the *siTD.Back Pointer.T-bit* in the *siTD.Back Pointer*

field set to a zero. Otherwise, software must set the *siTD.Back Pointer.T-bit* in the *siTD.Back Pointer* field to a one. The host controller's rules for interpreting when to use the *siTD.Back Pointer* field are listed below. These rules apply only when the siTD's *Active* bit is a one and the *SplitXState* is Do Complete Split.

- When cMicroFrameBit is a 1h and the siTDX.Back Pointer.T-bit is a zero, or
- If cMicroFrameBit is a 2h and siTDX.S-mask[0] is a zero

When either of these conditions apply, then the host controller must use the transaction state from $siTD_{X-1}$.

In order to access $siTD_{X-1}$, the host controller reads on-chip the siTD referenced from $siTD_X$.*Back Pointer*.

The host controller must save the entire state from $siTD_X$ while processing $siTD_{X-1}$. This is to accommodate for case 2b processing. The host controller must not recursively walk the list of *siTD.Back Pointers*.

If $siTD_{X-1}$ is active (*Active* bit is a one and *SplitXStat* is Do Complete Split), then both Test A and Test B are applied as described above. If these criteria to execute a complete-split are met, the host controller executes the complete split and evaluates the results as described above. The transaction state (see Table 847) of $siTD_{X-1}$ is appropriately advanced based on the results and written back to memory. If the resultant state of $siTD_{X-1}$'s *Active* bit is a one, then the host controller returns to the context of $siTD_X$, and follows its next pointer to the next schedule item. No updates to $siTD_X$ are necessary.

If $siTD_{X-1}$ is active (*Active* bit is a one and *SplitXStat* is Do Start Split), then the host controller must set *Active* bit to a zero and *Missed Micro-Frame* status bit to a one and the resultant status written back to memory.

If $siTD_{X-1}$*'s Active* bit is a zero, (because it was zero when the host controller first visited $siTD_{X-1}$ via $siTD_X$*'s* back pointer, it transitioned to zero as a result of a detected error, or the results of $siTD_{X-1}$*'s* complete-split transaction transitioned it to zero), then the host controller returns to the context of $siTD_X$ and transitions its *SplitXState* to Do Start Split. The host controller then determines whether the case 2b start split boundary condition exists (that is, if *cMicroframeBit* is a 1b and $siTD_X$.*S-mask*[0] is a 1b). If this criterion is met the host controller immediately executes a start-split transaction and appropriately advances the transaction state of $siTD_X$, then follows $siTD_X$.*Next Pointer* to the next schedule item. If the criterion is not met, the host controller simply follows $siTD_X$.*Next Pointer* to the next schedule item. Note that in the case of a 2b boundary case, the split-transaction of $siTD_{X-1}$ will have its *Active* bit set to zero when the host controller returns to the context of $siTD_X$. Also, note that software should not initialize an siTD with *C-mask* bits 0 and 1 set to a one and an *S-mask* with bit zero set to a one. This scheduling combination is not supported and the behavior of the host controller is undefined.

56.3.2.1.12.3.7 Split transaction for isochronous - processing examples

There is an important difference between how the hardware/software manages the isochronous split transaction state machine and how it manages the asynchronous and interrupt split transaction state machines. The asynchronous and interrupt split transaction state machines are encapsulated within a single queue head. The progress of the data stream depends on the progress of each split transaction. In some respects, the split-transaction state machine is sequenced via the Execute Transaction queue head traversal state machine (see Figure 412).

Isochronous is a pure time-oriented transaction/data stream. The interface data structures are optimized to efficiently describe transactions that need to occur at specific times. The isochronous split-transaction state machine must be managed across these time-oriented data structures. This means that system software must correctly describe the scheduling of split-transactions across more than one data structure.

Then the host controller must make the appropriate state transitions at the appropriate times, in the correct data structures.

For example, the table below illustrates a couple of frames worth of scheduling required to schedule a case 2a full-speed isochronous data stream.

Table 848. Example Case 2a - Software Scheduling siTDs for an IN Endpoint

siTDX		Micro-Frames								Initial SplitXState
#	Masks	0	1	2	3	4	5	6	7	
X	S-Mask	-	-	-	-	1	-	-	-	Do Start Split
	C-Mask	1	1	-	-	-	-	1	1	
X+1	S-Mask	-	-	-	-	1	-	-	-	Do Complete Split
	C-Mask	1	1					1	1	
X+2	S-Mask	-	-	-	-	1	-	-	-	Do Complete Split
	C-Mask	1	1					1	1	
X+3	S-Mask	Repeats previous pattern								Do Complete Split
	C-Mask									

This example shows the first three siTDs for the transaction stream. Because this is the case-2a frame-wrap case, *S-masks* of all siTDs for this endpoint have a value of 10h (a one bit in micro-frame 4) and *C-mask* value of C3h (one-bits in micro-frames 0,1, 6 and 7). Additionally, software ensures that the *Back Pointer* field of each siTD references the appropriate siTD data structure (and the *Back PointerT-bits* are set to zero).

The initial *SplitXState* of the first siTD is Do Start Split. The host controller will visit the first siTD eight times during frame X. The C-mask bits in micro-frames 0 and 1 are ignored because the state is Do Start Split. During micro-frame 4, the host controller determines that it can run a start-split (and does) and changes *SplitXState* to Do Complete Split. During micro-frames 6 and 7, the host controller executes complete-splits. Notice the siTD for frame X+1 has its *SplitXState* initialized to Do Complete Split. As the host controller continues to traverse the schedule during *H-Frame* X+1, it will visit the second siTD eight times. During micro-frames 0 and 1 it will detect that it must execute complete-splits.

During *H-Frame* X+1, micro-frame 0, the host controller detects that $siTD_{X+1}$'s *Back Pointer.T-bit* is a zero, saves the state of $siTD_{X+1}$ and fetches $siTD_X$. It executes the complete split transaction using the transaction state of $siTD_X$. If the $siTD_X$ split transaction is complete, siTD's *Active* bit is set to zero, and results written back to $siTD_X$. The host controller retains the fact that $siTD_X$ is retired and transitions the *SplitXState* in the $siTD_{X+1}$ to Do Start Split. At this point, the host controller is prepared to execute the start-split for $siTD_{X+1}$ when it reaches micro-frame 4. If the split-transaction completes early (transaction-complete is defined in Section Periodic isochronous - do complete split), that is, before all the scheduled complete-splits have been executed, the host controller will transition $siTD_X$.*SplitXState* to Do Start Split early and naturally skip the remaining scheduled complete-split transactions. For this example, $siTD_{X+1}$ does not receive a DATA0 response until *H-Frame* X+2, micro-frame 1.

During *H-Frame X+2,* micro-frame 0, the host controller detects that $siTD_{X+2}$'s *Back Pointer.T-bit* is a zero, saves the state of $siTD_{X+2}$ and fetches $siTD_{X+1}$. As described above, it executes another split transaction, receives an MDATA response, updates the transfer state, but does not modify the *Active* bit. The host controller returns to the context of $siTD_{X+2}$, and traverses its next pointer without any state change updates to $siTD_{X+2}$. S

During *H-Frame X+2*, micro-frame 1, the host controller detects $siTD_{X+2}$'s *S-mask[0]* is a zero, saves the state of $siTD_{X+2}$ and fetches $siTD_{X+1}$. It executes another complete-split transaction, receives a DATA0 response, updates the transfer state and sets the *Active* bit to a zero. It returns to the state of $siTD_{X+2}$ and changes its *SplitXState* to Do Start Split. At this point, the host controller is prepared to execute start-splits for $siTD_{X+2}$ when it reaches micro-frame 4.

56.3.2.1.13 Host controller pause

When the host controller's *HCHalted* bit in the USBSTS register is a zero, the host controller is sending SOF (Start OF Frame) packets down all enabled ports. When the schedules are enabled, the EHCI HC will access the schedules in main memory each micro-frame. This constant pinging of main memory is known to create Arm platform power management problems for mobile systems. Specifically, mobile systems aggressively manage the state of the Arm platform, based on recent history usage. In the more aggressive power saving modes, the Arm platform can disable its caches. Current PC architectures assume that bus-controller accesses to main memory must be cache-coherent. So, when bus controllers are busy touching memory, the Arm platform power management software can detect this activity over time and inhibit the transition of the Arm platform into its lowest power savings mode. USB controllers are bus controllers and the frequency at which they access their memory-based schedules keeps the Arm platform power management software from placing the Arm platform into its lowest power savings state.

USB Host controllers do not access main memory when they are suspended. However, there are a variety of reasons why placing the USB controllers into suspend will not work, but they are beyond the scope of this document. The base requirement is that the USB controller needs to be kept out of main memory, while at the same time, the USB bus is kept from going into suspend.

EHCI controllers provide a large-grained mechanism that can be manipulated by system software to change the memory access pattern of the host controller. System software can manipulate the schedule enable bits in the USBCMD register to turn on/off the scheduling traversal. A software heuristic can be applied to implement an on/off duty cycle that allows the USB to make reasonable progress and allow the Arm platform power management to get the Arm platform into its lowest power state. This method is not intended to be applied at all times to throttle USB, but should only be applied in very specific configurations and usage loads. For example, when only a keyboard or mouse is attached to the USB, the heuristic could detect times when the USB is attempting to move data only very infrequently and can adjust the duty cycle to allow the Arm platform to reach its low power state for longer periods of time. Similarly, it could detect increases in the USB load and adjust the duty cycle appropriately, even to the point where the schedules are never disabled. The assumption here is that the USB is moving data and the Arm platform will be required to process the data streams.

It is suggested that in order to provide a complete solution for the system, the companion host controllers should also provide a similar method to allow system software to inhibit the companion host controller from accessing its shared memory based data structures (schedule lists or otherwise).

56.3.2.1.14 Port test modes - host operational model

EHCI HCs must implement the port test modes Test J_State, Test K_State, Test_Packet, Test Force_Enable, and Test SE0_NAK as described in the USB Specification Revision 2.0. The system is only allowed to test ports that are owned by the EHCI controller (for example, *CF-bit* is a one and *PortOwner* bit is a zero). System software is allowed to have at most one port in test mode at a time. Placing more than one port in test mode will yield undefined results. The required, per port test sequence is (assuming the *CF-bit* in the USB_n_CONFIGFLAG register is a one):

- Disable the periodic and asynchronous schedules by setting the *Asynchronous Schedule Enable* and *Periodic Schedule Enable* bits in the USBCMD register to a zero.
- Place all enabled root ports into the suspended state by setting the *Suspend* bit in each appropriate USB_n_PORTSC register to a one.
- Set the *Run/Stop* bit in the USBCMD register to a zero and wait for the *HCHalted* bit in the USBSTS register, to transition to a one. Note that an EHCI HC implementation may optionally allow port testing with the *Run/Stop* bit set to a one. However, all host controllers must support port testing with *Run/Stop* set to a zero and *HCHalted* set to a one.
- Set the *Port Test Control* field in the port under test PORTSC register to the value corresponding to the desired test mode. If the selected test is Test_Force_Enable, then the *Run/Stop* bit in the USBCMD register must then be transitioned back to one, in order to enable transmission of SOFs out of the port under test.
- When the test is complete, system software must ensure the host controller is halted (*HCHalted* bit is a one) then it terminates and exits test mode by setting *HCReset* to a one.

56.3.2.1.15 Interrupts-host operational model

The EHCI HC hardware provides interrupt capability based on a number of sources. There are several general groups of interrupt sources:

- Interrupts as a result of executing transactions from the schedule (success and error conditions),
- Host controller events (Port change events, and so on), and
- Host Controller error events

All transaction-based sources are maskable through the Host Controller's Interrupt Enable register (USBINTR). Additionally, individual transfer descriptors can be marked to generate an interrupt on completion. This section describes each interrupt source and the processing that occurs in response to the interrupt.

During normal operation, interrupts may be immediate or deferred until the next interrupt threshold occurs. The interrupt threshold is a tunable parameter via the *Interrupt Threshold Control* field in the USBCMD register. The value of this register controls when the host controller will generate an interrupt on behalf of normal transaction execution. When a transaction completes during an interrupt interval period, the interrupt signaling the completion of the transfer will not occur until the interrupt threshold occurs. For example, the default value is eight micro-frames. This means that the host controller will not generate interrupts any more frequently than once every eight micro-frames.

Section Host system error details effects of a host system error.

If an interrupt has been scheduled to be generated for the current interrupt threshold interval, the interrupt is not signaled until after the status for the last complete transaction in the interval has been written back to host memory. This may sometimes result in the interrupt not being signaled until the next interrupt threshold.

Initial interrupt processing is the same, regardless of the reason for the interrupt. When an interrupt is signaled by the hardware, Arm platform control is transferred to host controller's USB interrupt handler. The precise mechanism to accomplish the transfer is OS specific. For this discussion it is just assumed that control is received. When the interrupt handler receives control, its first action is to reads the USBSTS (USB Status Register). It then acknowledges the interrupt by clearing all of the interrupt status bits by writing ones to these bit positions. The handler then determines whether the interrupt is due to schedule processing or some other event. After acknowledging the interrupt, the handler (via an OS-specific mechanism), schedules a deferred procedure call (DPC) which will execute later. The DPC routine processes the results of the schedule execution. The precise mechanisms used are beyond the scope of this document.

Note: the host controller is not required to de-assert a currently active interrupt condition when software sets the interrupt enables (in the USBINR register) to a zero. The only reliable method software should use for acknowledging an interrupt is by transitioning the appropriate status bits in the USBSTS register from a one to a zero.

56.3.2.1.15.1 Transfer/transaction based interrupts

These interrupt sources are associated with transfer and transaction progress. They are all dependent on the next interrupt threshold.

56.3.2.1.15.1.1 Transaction error

A transaction error is any error that caused the host controller to think that the transfer did not complete successfully. The table below lists the events/responses that the host can observe as a result of a transaction. The effects of the error counter and interrupt status are summarized in the following paragraphs. Most of these errors set the *XactErr* status bit in the appropriate interface data structure.

There is a small set of protocol errors that relate only when executing a queue head and fit under the umbrella of a WRONG PID error that are significant to explicitly identify. When these errors occur, the *XactErr* status bit in the queue head is set, and the *CErr* field is decremented. When the *PIDCode* indicates a SETUP, the following responses are protocol errors and result in *XactErr* bit being set to a one and the *CErr* field being decremented.

- *EPS* field indicates a high-speed device and it returns a Nak handshake to a SETUP.
- *EPS* field indicates a high-speed device and it returns a Nyet handshake to a SETUP.
- *EPS* field indicates a low- or full-speed device and the complete-split receives a Nak handshake.

Table 849. Summary of Transaction Errors

Event / Result	Queue Head/qTD/iTD/siTD Side-effects		USB Status Register (USBSTS)
	Cerr	Status Field	USBERRINT
CRC	-1	XactErr set to a one.	1[1]
Timeout	-1	XactErr set to a one.	1[1]
Bad PID[2]	-1	XactErr set to a one.	1[1]
Babble	N/A	Section Serial bus babble	1
Buffer Error	N/A	Section Data buffer error	

1. If occurs in a queue head, then *USBERRINT* is asserted only when *CErr* counts down from a one to a zero. In addition the queue is halted, see Halting a queue head .
2. The host controller received a response from the device, but it could not recognize the PID as a valid PID.

56.3.2.1.15.1.2 Serial bus babble

When a device transmits more data on the USB than the host controller is expecting for this transaction, it is defined to be babbling. In general, this is called a *Packet Babble*. When a device sends more data than the *Maximum Length* number of bytes, the host controller sets the *Babble Detected* bit to a one and halts the endpoint if it is using a queue head (see Halting a queue head). *Maximum Length* is defined as the minimum of *Total Bytes to Transfer* and *Maximum Packet Size*. The *CErr* field is not decremented for a packet babble condition (only applies to queue heads). A babble condition also exists if IN transaction is in progress at High-speed EOF2 point. This is called a frame babble. A frame babble condition is recorded into the appropriate schedule data structure. In addition, the host controller must disable the port to which the frame babble is detected.

The *USBERRINT* bit in the USB_n_USBSTS register is set to a one and if the *USB Error Interrupt Enable* bit in the USB_n_USBINTR register is a one, then a hardware interrupt is signaled to the system at the next interrupt threshold. The host controller must never start an OUT transaction that will babble across a micro-frame EOF.

NOTE

When a host controller detects a data PID mismatch, it must either: disable the packet babble checking for the duration of the bus transaction or do packet babble checking based solely on *Maximum Packet Size*. The USB core specification defines the requirements on a data receiver when it receives a data PID mismatch (for example, expects a DATA0 and gets a DATA1 or the opposite way). In summary, it must ignore the received data and respond with an ACK handshake, in order to advance the transmitter's data sequence.

The EHCI interface allows System software to provide buffers for a Control, Bulk, or Interrupt IN endpoint that are not an even multiple of the maximum packet size specified by the device. Whenever a device misses an ACK for an IN endpoint, the host and device are out of synchronization with respect to the progress of the data transfer. The host controller may have advanced the transfer to a buffer that is less than maximum packet size. The device will re-send its maximum packet size data packet, with the original data PID, in response to the next IN token. In order to properly manage the bus protocol, the host controller must disable the packet babble check when it observes the data PID mismatch.

56.3.2.1.15.1.3 Data buffer error

This event indicates that an overrun of incoming data or an underrun of outgoing data has occurred for this transaction. This would generally be caused by the host controller not being able to access required data buffers in memory within necessary latency requirements. These conditions are not considered transaction errors, and do not effect the error count in the queue head. When these errors do occur, the host controller records the fact the error occurred by setting the *Data Buffer Error* bit in the queue head, iTD, or siTD.

If the data buffer error occurs on a non-isochronous IN, the host controller will not issue a handshake to the endpoint. This will force the endpoint to resend the same data (and data toggle) in response to the next IN to the endpoint.

If the data buffer error occurs on an OUT, the host controller must corrupt the end of the packet so that it cannot be interpreted by the device as a good data packet. Simply truncating the packet is not considered acceptable. An acceptable implementation option is to 1's complement the CRC bytes and send them. There are other options suggested in the Transaction Translator section of the USB Specification Revision 2.0.

56.3.2.1.15.1.4 USB interrupt (Interrupt on completion (IOC))

Transfer Descriptors (iTDs, siTDs, and queue heads (qTDs)) contain a bit that can be set to cause an interrupt on their completion. The completion of the transfer associated with that schedule item causes the USB Interrupt (USBINT) bit in the USB_n_USBSTS register to be set to a one. In addition, if a short packet is encountered on an IN transaction associated with a queue head, then this event also causes USBINT to be set to a one. If the USB Interrupt Enable bit in the USB_n_USBINTR register is set to a one, a hardware interrupt is signaled to the system at the next interrupt threshold. If the completion is because of errors, the *USBERRINT* bit in the USB_n_USBSTS register is also set to a one.

56.3.2.1.15.1.5 Short packet

Reception of a data packet that is less than the endpoint's Max Packet size during Control, Bulk, or Interrupt transfers signals the completion of the transfer. Whenever a short packet completion occurs during a queue head execution, the *USBINT* bit in the USB_n_USBSTS register is set to a one. If the *USB Interrupt Enable* bit is set in the USB_n_USBINTR register, a hardware interrupt is signaled to the system at the next interrupt threshold.

56.3.2.1.15.2 Host controller event interrupts

These interrupt sources are independent of the interrupt threshold (with the one exception being the Interrupt on Async Advance, see Section Interrupt on async advance).

56.3.2.1.15.2.1 Port change events

Port registers contain status and status change bits. When the status change bits are set to a one, the host controller sets the *Port Change Detect* bit in the USBSTS register to a one. If the *Port Change Interrupt Enable* bit in the USB_n_USBINTR register is a one, then the host controller will issue a hardware interrupt. The port status change bits include:

- Connect Status Change
- Port Enable/Disable Change
- Over-current Change
- Force Port Resume

56.3.2.1.15.2.2 Frame list rollover

This event indicates that the host controller has wrapped the frame list. The current programmed size of the frame list effects how often this interrupt occurs. If the frame list size is 1024, then the interrupt will occur every 1024 milliseconds, if it is 512, then it will occur every 512 milliseconds, and so on. When a frame list rollover is detected, the host controller sets the *Frame List Rollover* bit in the USB.USBSTS register to a one. If the *Frame List Rollover Enable* bit in the USB.USBINTR register is set to a one, the host controller issues a hardware interrupt. This interrupt is not delayed to the next interrupt threshold.

56.3.2.1.15.2.3 Interrupt on async advance

This event is used for deterministic removal of queue heads from the asynchronous schedule. Whenever the host controller advances the on-chip context of the asynchronous schedule, it evaluates the value of the *Interrupt on Async Advance Doorbell* bit in the USB.USBCMD register. If it is a one, it sets the *Interrupt on Async Advance* bit in the USB.USBSTS register to a one. If the *Interrupton Async Advance Enable* bit in the USB.USBINTR register is a one, the host controller issues a hardware interrupt at the next interrupt threshold. A detailed explanation of this feature is described in Section Removing queue heads from asynchronous schedule .

56.3.2.1.15.2.4 Host system error

The host controller is a bus controller and any interaction between the host controller and the system may experience errors. The type of host error may be catastrophic to the host controller (such as a Controller Abort) making it impossible for the host controller to continue in a coherent fashion. In the presence of non-catastrophic host errors, such as parity errors, the host controller could potentially continue operation. The recommended behavior for these types of errors is to escalate it to a catastrophic error and halt the host controller. Host-based error must result in the following actions:

- The *Run/Stop* bit in the USB.USBCMD register is set to a zero.
- The following bits in the USB.USBSTS register are set:
  - *Host System Error* bit is to a one.
  - *HCHalted* bit is set to a one.
- If the *Host System Error Enable* bit in the USB.USBINTR register is a one, then the host controller will issue a hardware interrupt. This interrupt is not delayed to the next interrupt threshold. The following table summarizes the required actions taken on the various host errors.

Table 850. Summary Behavior of EHCI HC on host system errors

Cycle Type	Controller Abort	Target Abort	Data Phase Parity
Frame list pointer fetch (read)	Fatal	Fatal	Fatal [o]
siTD fetch (read)	Fatal	Fatal	Fatal [o]
siTD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
iTD fetch (read)	Fatal	Fatal	Fatal [o]
iTD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
qTD fetch (read)	Fatal	Fatal	Fatal [o]
qHD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
Data write	Fatal [o]	Fatal [o]	Fatal [o]
Data read	Fatal	Fatal	Fatal [o]

Potentially, a host controller implementation could continue operation without a halt. However, the recommended behavior is to halt the host controller.

NOTE

After a *Host System Error*, Software must reset the host controller through *HCReset* in the USB.USBCMD register before re-initializing and restarting the host controller.

56.3.2.2 EHCI deviation

For the purposes a dual-role Host/Device controller with support for On-The-Go applications, it is necessary to deviate from the EHCI specification. Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 0.95, November 2000, Intel Corporation. http://www.intel.com. Device operation and On-The-Go operation is not specified in the EHCI and therefore the implementation supported in this core is proprietary. The Host mode operation of the core is near EHCI compatible with few minor differences documented in this section.

The particulars of the deviations occur in the areas summarized here:

- Embedded Transaction Translator - Allows direct attachment of FS and LS devices in Host mode without the need for a companion controller.
- Device operation - In Host mode, the device operational registers are generally disabled and thus device mode is mostly transparent when in Host mode. However, there are a couple exceptions documented in the following sections.
- Embedded design interface - This core does not have a PCI Interface and therefore the PCI configuration registers described in the EHCI specification are not applicable.
- On-The-Go Operation - This design includes an On-The-Go controller for Port #1.

56.3.2.2.1 Embedded transaction translator function

The OTG controller supports directly connected full and low speed devices without requiring a companion controller by including the capabilities of a USB 2.0 high speed hub transaction translator. Although there is no separate Transaction Translator block in the system, the transaction translator function normally associated with a high speed hub has been implemented within the DMA and Protocol engine blocks. The embedded transaction translator function is an extension to EHCI interface, but makes use of the standard data structures and operational models that exist in the EHCI specification to support full and low speed devices.

56.3.2.2.1.1 Capability registers

The following additions have been added to the capability registers to support the embedded Transaction Translator Function:

- N_TT added to USB.HCSPARAMS - Host Control Structural Parameters
- N_PTT added to USB.HCSPARAMS - Host Control Structural Parameters

56.3.2.2.1.2 Operational registers

The following additions have been added to the operational registers to support the embedded TT:

- Addition of two-bit Port Speed (PSPD) to the PORTSC1 register.

56.3.2.2.1.3 Discovery-EHCI deviation

In a standard EHCI controller design, the EHCI HC driver detects a Full speed (FS) or Low speed (LS) device by noting if the port enable bit is set after the port reset operation. The port enable will only be set in a standard EHCI controller implementation after the port reset operation and when the host and device negotiate a High-Speed connection (that is, Chirp completes successfully).

Because this controller has an embedded Transaction Translator, the port enable will always be set after the port reset operation regardless of the result of the host device chirp result, and the resulting port speed will be indicated by the PSPD field in USB.PORTSCx.

Therefore, the standard EHCI HC driver requires an alteration to handle directly connected Full and Low speed devices or hubs.

The change is a fundamental one in that is summarized in the following table.

Table 851. Summary of EHCI

Standard EHCI	EHCI with embedded Transaction Translator
After port enable bit is set following a connection and reset sequence, the device/hub is assumed to be HS.	After port enable bit is set following a connection and reset sequence, the device/hub speed is noted from USB.PORTSCx.
FS and LS devices are assumed to be downstream from an HS hub thus, all port-level control is performed through the Hub Class to the nearest Hub.	FS and LS device can be either downstream from an HS hub or directly attached. When the FS/LS device is downstream from an HS hub, then port-level control is done using the Hub Class through

Table continues on the next page...

Table 851. Summary of EHCI (continued)

	the nearest Hub. When a FS/LS device is directly attached, then port-level control is accomplished using USB.PORTSCx.
FS and LS devices are assumed to be downstream from an HS hub with HubAddr=X. [where HubAddr > 0 and HubAddr is the address of the Hub where the bus transitions from HS to FS/LS (that is, Split target hub)]	FS and LS device can be either downstream from an HS hub with HubAddr = X [HubAddr > 0] or directly attached [where HubAddr = 0 and HubAddr is the address of the Root Hub where the bus transitions from HS to FS/LS (that is, Split target hub is the root hub)]

56.3.2.2.1.4 Data structures

The same data structures used for FS/LS transactions though an HS hub are also used for transactions through the Root Hub with sm embedded Transaction Translator. Here it is demonstrated how the Hub Address and Endpoint Speed fields should be set for directly attached FS/LS devices and hubs:

1. QH (for direct attach FS/LS) - Async. (Bulk/Control Endpoints) Periodic (Interrupt)
   - Hub Address = 0
   - Transactions to direct attached device/hub.
     - QH.EPS = Port Speed
   - Transactions to a device downstream from direct attached FS hub.
     - QH.EPS = Downstream Device Speed

     NOTE

     When QH.EPS = 01 (LS) and PORTSCx.PSPD = 00 (FS), a LS-pre-pid will be sent before the transmitting LS traffic.

     Maximum Packet Size must be less than or equal 64 or undefined behaviour may result.
2. siTD (for direct attach FS) - Periodic (ISO Endpoint)
   - All FS ISO transactions:
     - Hub Address = 0
     - siTD.EPS = 00 (full speed)
       - Maximum Packet Size must less than or equal to 1023 or undefined behaviour may result.

56.3.2.2.1.5 Operational model

The operational models are well defined for the behavior of the Transaction Translator (see USB 2.0 specification. Universal Serial Bus Specification, Revision 2.0, April 2000, Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips. http://www.usb.org) and for the EHCI controller moving packets between system memory and a USB-HS hub. Because the embedded Transaction Translator exists within the host controller, there is no physical bus between EHCI HC driver and the USB FS/LS bus. These sections will briefly discuss the operational model for how the EHCI and Transaction Translator operational models are combined without the physical bus between. The following sections assume that the reader is familiar with both the EHCI and USB 2.0 Transaction Translator operational models.

56.3.2.2.1.5.1 Micro-frame pipeline

The EHCI operational model uses the concept of H-frames and B-frames to describe the pipeline between the Host (H) and the Bus (B). The embedded Transaction Translator shall use the same pipeline algorithms specified in the USB 2.0 specification for a Hub-based Transaction Translator.

All periodic transfers always begin at B-frame 0 (after SOF) and continue until the stored periodic transfers are complete. As an example of the micro-frame pipeline implemented in the embedded Transaction Translator, all periodic transfers that are tagged in EHCI to execute in H-frame 0 will be ready to execute on the bus in B-frame 0.

It is important to note that when programming the S-mask and C-masks in the EHCI data structures to schedule periodic transfers for the embedded Transaction Translator, the EHCI HC driver must follow the same rules specified in EHCI for programming the S-mask and C-mask for downstream Hub-based Transaction Translators.

Once periodic transfers are exhausted, any stored asynchronous transfer will be moved. Asynchronous transfers are opportunistic in that they shall execute whenever possible and their operation is not tied to H-frame and B-frame boundaries with the exception that an asynchronous transfer cannot babble through the SOF (start of B-frame 0.)

56.3.2.2.1.5.2 Split state machines

The start and complete split operational model differs from EHCI slightly because there is no bus medium between the EHCI controller and the embedded Transaction Translator. Where a start or complete-split operation would occur by requesting the split to the HS hub, the start/complete split operation is simple an internal operation to the embedded Transaction Translator. The following table summarizes the conditions where handshakes are emulated from internal state instead of actual handshakes to HS split bus traffic.

Table 852. Summary of the Conditions of Handshakes[1]

Condition	Emulate TT Response
Start-Split: All asynchronous buffers full.	NAK
Start-Split: All periodic buffers full.	ERR
Start-Split: Success for start of Async. Transaction.	ACK
Start-Split: Start Periodic Transaction.	No Handshake (Ok)
Complete-Split: Failed to find transaction in queue.	Bus Time Out
Complete-Split: Transaction in Queue is Busy.	NYET
Complete-Split: Transaction in Queue is Complete.	[Actual Handshake from LS/FS device]

1. The un-shaded cells represent Start-Splits and the shaded cells represent Complete-Splits

56.3.2.2.1.5.3 Asynchronous transaction scheduling and buffer management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

- Sequencing is provided & a packet length estimator ensures no full-speed/low-speed packet babbles into SOF time.
- Transaction tracking for 2 data pipes.

56.3.2.2.1.5.3.1 USB 2.0 - 11.17.3

- Sequencing is provided & a packet length estimator ensures no full-speed/low-speed packet babbles into SOF time.

56.3.2.2.1.5.3.2 USB 2.0 - 11.17.4

- Transaction tracking for 2 data pipes.

56.3.2.2.1.5.4 Periodic transaction scheduling and buffer management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

- Abort of pending start-splits

  - EOF (and not started in micro-frames 6)
  - Idle for more than 4 micro-frames
- Abort of pending complete-splits
  - EOF
  - Idle for more than 4 micro-frames
- Transaction tracking for up to 16 data pipes.
- Complete-split transaction searching.

NOTE

There is no data schedule mechanism for these transactions other than the micro-frame pipeline. The embedded TT assumes the number of packets scheduled in a frame does not exceed the frame duration (1 ms) or else undefined behavior may result.

56.3.2.2.1.5.4.1 USB 2.0 - 11.18.6.[1-2]

- Abort of pending start-splits
  - EOF (and not started in micro-frames 6)
  - Idle for more than 4 micro-frames
- Abort of pending complete-splits
  - EOF
  - Idle for more than 4 micro-frames

56.3.2.2.1.5.4.2 USB 2.0 - 11.18.[7-8]

- Transaction tracking for up to 16 data pipes.
- Complete-split transaction searching.

NOTE

There is no data schedule mechanism for these transactions other than the micro-frame pipeline. The embedded TT assumes the number of packets scheduled in a frame does not exceed the frame duration (1 ms) or else undefined behavior may result.

56.3.2.2.1.5.5 Multiple transaction translators

The maximum number of embedded Transaction Translators that is currently supported is one as indicated by the N_TT field in the HCSPARAMS register.

56.3.2.2.2 Device operation

The co-existence of a device operational controller within the host controller has little effect on EHCI compatibility for host operation except as noted in this section.

56.3.2.2.2.1 USBMODE

EHCI HC driver cannot start EHCI host operations until there is USB Device Mode (USBMODE) configuration for host operation. This is because the dual-role controller is not initially in either a host or device mode.

56.3.2.2.2.2 Non-zero fields the register file

Some of the reserved fields and reserved addresses in the capability registers and operational register have use in Device mode. Adhere to the following:

- Write operations to all EHCI reserved fields (some of which are device fields) by writing 0 to the operation registers. You must adhere to this EHCI requirement of the device controller driver (DCD).
- Read operations by the host controller must properly mask EHCI reserved fields (some of which are device fields). This is because Host mode does not define fields that are exclusive to device.

56.3.2.2.2.3 SOF interrupt

This SOF Interrupt used for device mode is shared as a free running 125us interrupt for Host mode. EHCI does not specify this interrupt but it has been added for convenience and as a potential software time base. See USB_nUSBSTS and USB_nUSBINTR registers.

56.3.2.2.3 Embedded design interface

This is an Embedded USB Host Controller as defined by the EHCI specification and thus does not implement the PCI configuration registers.

56.3.2.2.3.1 Frame adjust register

Given that the optional PCI configuration registers are not included in this implementation, there is no corresponding bit level timing adjustments like is provided by the Frame Adjust register in the PCI configuration registers. Starts of micro-frames are timed precisely to 125 us using the transceiver clock as a reference clock. That is, a 60 MHz transceiver clock for 8-bit physical interfaces & full-speed serial interfaces or 30 MHz transceiver clock for 16-bit physical interfaces.

56.3.2.2.4 Miscellaneous variations from EHCI

56.3.2.2.4.1 Programmable physical interface behaviour

This design supports multiple Physical interfaces which can operate in differing modes when the core is configured with software programmable Physical Interface Modes. Software programmability allows the selection of the Physical interface part during the board design phase instead of during the chip design phase. The control bits for selecting the Physical Interface operating mode have been added to the USB_nPORTSC1 register providing a capability that is not defined by EHCI.

56.3.2.2.4.2 Discovery

56.3.2.2.4.2.1 Port reset

The port connect methods specified by EHCI require setting the port reset bit in the USB_nPORTSC1 register for a duration of 10 ms. Due to the complexity required to support the attachment of devices that are not high speed there are counter already present in the design that can count the 10ms reset pulse to alleviate the requirement of the software to measure this duration. Therefore, the basic connection is then summarized as the following:

- [Port Change Interrupt] Port connect change occurs to notify the host controller driver that a device has attached.
- Software shall write a '1' to reset the device.
- Software shall write a '0' to reset the device after 10 ms.
  - This step, which is necessary in a standard EHCI design, may be omitted with this implementation. Should the EHCI HC driver attempt to write a '0' to the reset bit while a reset is in progress, the write will simple be ignored and the reset will continue until completion.
- [Port Change Interrupt] Port enable change occurs to notify the host controller that the device in now operational and at this point the port speed has been determined.

56.3.2.2.4.2.2 Port speed detection

After the port change interrupt indicates that a port is enabled, the EHCI stack should determine the port speed. Unlike the EHCI implementation, which will re-assign the port owner for any device that does not connect at High-Speed, this host

controller supports direct attach of non High-Speed devices. Therefore, the following differences are important regarding port speed detection:

- Port Owner is read-only and always reads 0.
- A 2-bit Port Speed indicator has been added to PORTSC to provide the current operating speed of the port to the host controller driver.
- A 1-bit High Speed indicator has been added to PORTSC to signify that the port is in High-Speed vs. Full/Low Speed - *This information is redundant with the 2-bit Port Speed indicator above.*

56.3.2.2.4.3 Port test mode

Port Test Control mode behaves fully as described in EHCI. An alternate host controller driver procedure is not necessary or supported.

56.3.2.3 Device operational model

The function of the device operation is to transfer a request in the memory image to and from the USB. Using a set of linked list transfer descriptors, pointed to by a queue head, the device controller will perform the data transfers. The following sections explain the use of the device controller from the DCD point-of-view and further describe how specific USB bus events relate to status changes in the device controller programmer's interface.

56.3.2.3.1 Device controller initialization

After hardware reset, the device is disabled until the Run/Stop bit is set to a '1'. In the disabled state, the pull-up on the USB D+ is not active which prevents an attach event from occurring. At a minimum, it is necessary to have the queue heads setup for endpoint zero before the device attach occurs. Shortly after the device is enabled, a USB reset will occur followed by setup packet arriving at endpoint 0. A Queue head must be prepared so that the device controller can store the incoming setup packet.

In order to initialize a device, the software should perform the following steps:

- Set Controller Mode in the USB.USBMODE register to device mode.

  NOTE

  Transitioning from Host mode to device mode requires a device controller reset before modifying USB.USBMODE.

- Allocate and Initialize device queue heads in system memory.
  - Minimum: Initialize device queue heads 0 Tx & 0 Rx.

    NOTE

    All device queue heads for control endpoints must be initialized before the endpoint is enabled. Non-Control device queue heads before the endpoint can be used.

  - For information on device queue heads, refer to section Device data structures.
- Configure USB.ENDPOINTLISTADDR Pointer.
  - For additional information on USB.ENDPOINTLISTADDR, refer to the register table.
- Enable the microprocessor interrupt associated with the USB core.
  - Recommended: enable all device interrupts including: USBINT, USBERRINT, Port Change Detect, USB Reset Received, DCSuspend.
  - For a list of available interrupts refer to the USB_nUSBINTR and the USB_nUSBSTS register tables.
- Set Run/Stop bit to Run Mode.
  - After the Run bit is set and the device is connected to a host, a Bus Reset will be issued by host downstream port. The DCD must monitor the reset event and adjust the software state as described in the Bus Reset section of the Port State and Control section below.

NOTE

Endpoint 0 is designed as a control endpoint only and does not need to be configured using ENDPTCTRL0 register.

It is also not necessary to prime Endpoint 0 initially because the first packet received will always be a setup packet. The contents of the first setup packet will require a response in accordance with USB device framework (Chapter 9) command set.

56.3.2.3.2 Port state and control

From a chip or system reset, the device controller enters the *powered* state. A transition from the *powered* state to the *attach* state occurs when the Run/Stop bit is set to a '1'. After receiving a reset on the bus, the port will enter the *defaultFS or defaultHS* state in accordance with the reset protocol described in Appendix C.2 of the USB Specification Rev. 2.0. The following state diagram depicts the state of a USB 2.0 device.

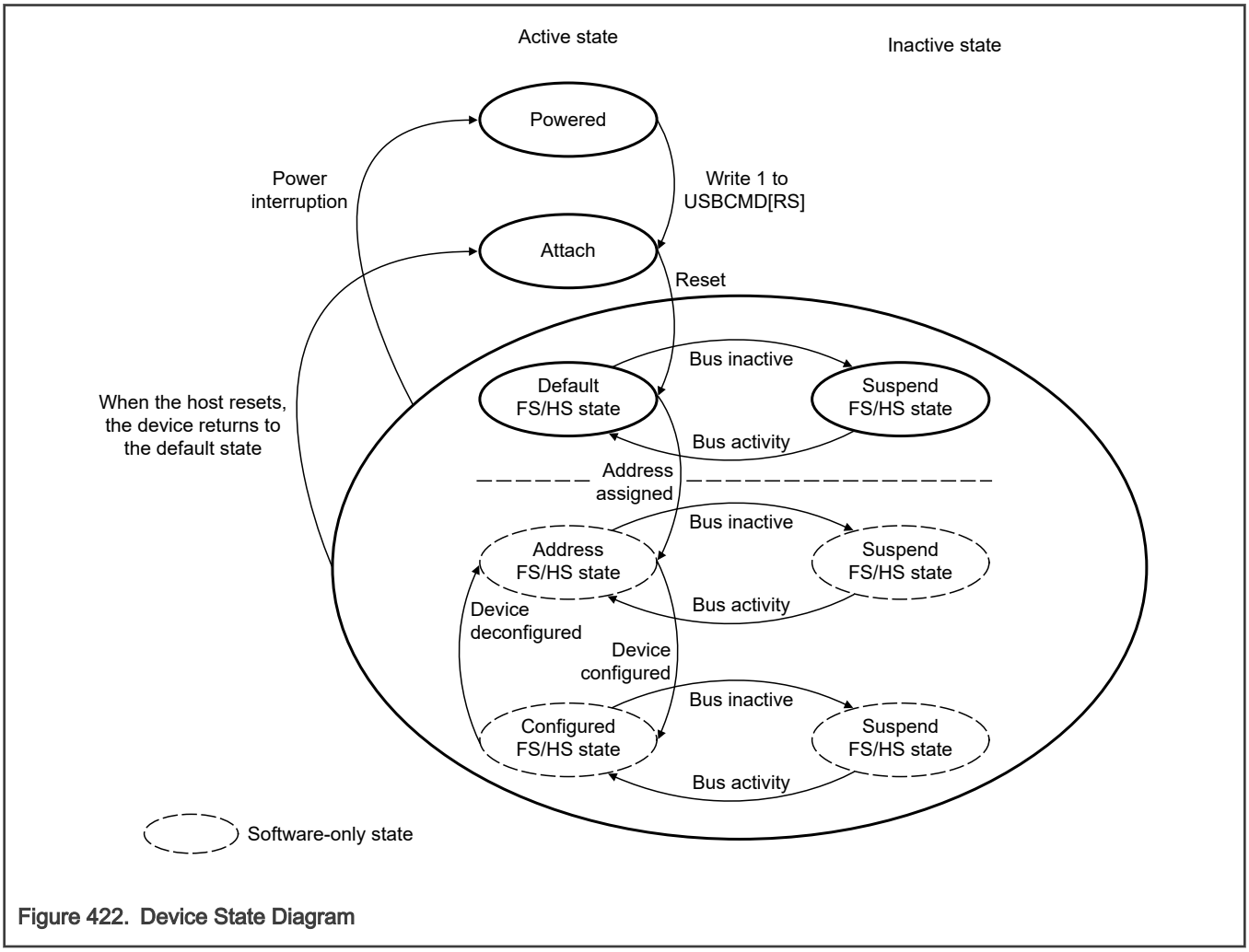


Figure 422. Device State Diagram

States *powered, attach, defaultFS/HS, suspendFS/HS* are implemented in the device controller and are communicated to the DCD using the following status bits:

The following table describes the Device Controller State Information Bits.

Table 853. Device Controller State Information Bits

Bit	Register
DCSuspend	USBSTS

Table continues on the next page...

Table 853. Device Controller State Information Bits (continued)

USB Reset Received	USBSTS
Port Change Detect	USBSTS
High-Speed Port	PORTSC1

It is the responsibility of the DCD to maintain a state variable to differentiate between the *DefaultFS/HS* state and the *Address/Configured* states. Change of state from *Default* to *Address* and the *Configured* states is part of the enumeration process described in the device framework section of the USB 2.0 Specification.

As a result of entering the *Address* state, the device address register (DEVICEADDR) must be programmed by the DCD.

Entry into the *Configured* indicates that all endpoints to be used in the operation of the device have been properly initialized by programming the USB_UOG_ENDPTCTRLx registers and initializing the associated queue heads.

56.3.2.3.2.1 Bus reset

A bus reset is used by the host to initialize downstream devices. When a bus reset is detected, the device controller will renegotiate its attachment speed, reset the device address to 0, and notify the DCD by interrupt (assuming the USB Reset Interrupt Enable is set). After a reset is received, all endpoints (except endpoint 0) are disabled, and any primed transactions will be cancelled by the device controller. The concept of priming will be clarified below, but the DCD must perform the following tasks when a reset is received:

Clear all setup token semaphores by reading the USB_nENDPTSTAT register and writing the same value back to the USB_nENDPTSTAT register.

Clear all the endpoint complete status bits by reading the USB_nPTCOMPLETE register and writing the same value back to the USB_nENDPTCOMPLETE register.

Cancel all primed status by waiting until all bits in the USB_nENDPTPRIME are 0 and then writing 0xFFFFFFFF to USB_nENDPTFLUSH.

Read the reset bit in the USB_nPORTSC1 register and make sure that it is still active. A USB reset will occur for a minimum of 3 ms and the DCD must reach this point in the reset cleanup before end of the reset occurs, otherwise a hardware reset of the device controller is recommended (rare.)

- A hardware reset can be performed by writing a one to the device controller reset bit in the USBCMD reset. Note: a hardware reset will cause the device to detach from the bus by clearing the Run/Stop bit. Thus, the DCD must completely re-initialize the device controller after a hardware reset.

Free all allocated dTDs because they will no longer be executed by the device controller. If this is the first time the DCD is processing a USB reset event, then it is likely that no dTDs have been allocated.

At this time, the DCD may release control back to the OS because no further changes to the device controller are permitted until a Port Change Detect is indicated.

After a Port Change Detect, the device has reached the default state and the DCD can read the USB_nPORTSC1 to determine if the device is operating in FS or HS mode. At this time, the device controller has reached normal operating mode and DCD can begin enumeration according to the USB Chapter 9 - Device Framework.

NOTE

The device DCD may use the FS/HS mode information to determine the bandwidth mode of the device.

In some applications, it may not be possible to enable one or more pipes while in FS mode. *Beyond the data rate issue, there is no difference in DCD operation between FS and HS modes.*

56.3.2.3.2.2 Suspend/resume

The details of suspend and resume are explained in these sections.

56.3.2.3.2.2.1 Suspend

Suspend Description

In order to conserve power, USB devices automatically enter the suspended state when the device has observed no bus traffic for a specified period. When suspended, the USB device maintains any internal status, including its address and configuration. Attached devices must be prepared to suspend at any time they are powered, regardless of if they have been assigned a non-default address, are configured, or neither. Bus activity may cease due to the host entering a suspend mode of its own. In addition, a USB device shall also enter the suspended state when the hub port it is attached to is disabled.

A USB device exits suspend mode when there is bus activity. A USB device may also request the host to exit suspend mode or selective suspend by using electrical signaling to indicate remote wake-up. The ability of a device to signal remote wake-up is optional. If the USB device is capable of remote wake-up signaling, the device must support the ability of the host to enable and disable this capability. When the device is reset, remote wake-up signaling must be disabled.

Suspend Operational Model

The device controller moves into the suspend state when suspend signaling is detected or activity is missing on the upstream port for more than a specific period. After the device controller enters the suspend state, the DCD is notified by an interrupt (assuming *DC Suspend Interrupt* is enabled). When the *DCSuspend* bit in the USB_nPORTSC1 is set to a '1', the device controller is suspended.

DCD response when the device controller is suspended is application specific and may involve switching to low power operation.

Information on the bus power limits in suspend state can be found in USB 2.0 specification.

NOTE

Review system level clocking issues defined in section (Ref: Signals-Clocking) for the clocking requirements of a suspended device controller.

56.3.2.3.2.2.2 Resume

If the device controller is suspended, its operation is resumed when any non-idle signaling is received on its upstream facing port. In addition, the device can signal the system to resume operation by forcing resume signaling to the upstream port. Resume signaling is sent upstream by writing a '1' to the Resume bit in the USB_nPORTSC1 while the device is in suspend state. Sending resume signal to an upstream port should cause the host to issue resume signaling and bring the suspended bus segment (one more devices) back to the active condition.

NOTE

Before resume signaling can be used, the host must enable it by using the Set Feature command defined in device framework (chapter 9) of the USB 2.0 Specification.

56.3.2.3.3 Managing endpoints

The USB 2.0 specification defines an endpoint, also called a device endpoint or an address endpoint as a uniquely addressable portion of a USB device that can source or sink data in a communications channel between the host and the device. The endpoint address is specified by the combination of the endpoint number and the endpoint direction.

The channel between the host and an endpoint at a specific device represents a data pipe. Endpoint 0 for a device is always a *control* type data channel used for device discovery and enumeration. Other types of endpoints support by USB include *bulk, interrupt*, and *isochronous*. Each endpoint type has specific behavior related to packet response and error handling. More detail on endpoint operation can be found in the USB 2.0 specification.

The USB OTG device controller hardware supports up to 8 endpoint numbers.

Each endpoint direction is essentially independent and can be configured with differing behavior in each direction. For example, the DCD can configure endpoint 1-IN to be a bulk endpoint and endpoint 1-OUT to be an isochronous endpoint. This helps to conserve the total number of endpoints required for device operation. The only exception is that control endpoints must use both directions on a single endpoint number to function as a control endpoint. Endpoint 0 is, for example, is always a control endpoint and uses the pair of directions.

Each endpoint direction requires a *queue head* allocated in memory. To support the 8 endpoint numbers, 16 *queue heads* are required. The operation of an endpoint and use of *queue heads* are described later in this document.

56.3.2.3.3.1 Endpoint initialization

After hardware reset, all endpoints except endpoint zero are uninitialized and disabled. The DCD must configure and enable each endpoint by writing to configuration bit in the USB_UOG_ENDPTCTRLx register. Each 32-bit USB_UOG_ENDPTCTRLx is split into an upper and lower half. The lower half of USB_UOG_ENDPTCTRLx is used to configure the receive or OUT endpoint and the upper half is likewise used to configure the corresponding transmit or IN endpoint. Control endpoints must be configured the same in both the upper and lower half of the USB_UOG_ENDPTCTRLx register otherwise the behavior is undefined. The following table shows how to construct a configuration word for endpoint initialization. The following table shows the fields and values for the Device Controller Endpoint initialization.

Table 854. Device Controller Endpoint Initialization

Field	Value
Data Toggle Reset	1
Data Toggle Inhibit	0
Endpoint Type	00 Control 01 Isochronous 10 Bulk 11 Interrupt
Endpoint Stall	0

56.3.2.3.3.2 Stalling

There are two occasions where the device controller may need to return to the host a STALL.

The first occasion is the functional stall, which is a condition set by the DCD as described in the USB 2.0 device framework. A functional stall is only used on non-control endpoints and can be enabled in the device controller by setting the endpoint stall bit in the USB_UOG_ENDPTCTRLx register associated with the given endpoint and the given direction. In a functional stall condition, the device controller will continue to return STALL responses to all transactions occurring on the respective endpoint and direction until the endpoint stall bit is cleared by the DCD.

A protocol stall, unlike a function stall, is used on control endpoints is automatically cleared by the device controller at the start of a new control transaction (setup phase). When enabling a protocol stall, the DCD should enable the stall bits (both directions) as a pair. A single write to the USB_UOG_ENDPTCTRLx register can ensure that both stall bits are set at the same instant.

NOTE

Any write to the USB_UOG_ENDPTCTRLx register during operational mode must preserve the endpoint type field (that is, perform a read-modify-write).

The following table shows the response matrix for the Device Controller Stall.

Table 855. Device Controller Stall Response Matrix

USB Packet	Endpoint Stall Bit.	Effect on STALL bit.	USB Response
SETUP packet received by a non-control endpoint.	N/A	None.	STALL

Table continues on the next page...

Table 855. Device Controller Stall Response Matrix (continued)

IN/OUT/PING packet received by a non-control endpoint.	'1'	None.	STALL
IN/OUT/PING packet received by a non-control endpoint.	'0'	None.	ACK/ NAK/ NYET
SETUP packet received by a control endpoint.	N/A	Cleared	ACK
IN/OUT/PING packet received by a control endpoint	'1'	None	STALL
IN/OUT/PING packet received by a control endpoint.	'0'	None.	ACK/ NAK/ NYET

56.3.2.3.3.3 Data toggle

Data toggle is a mechanism to maintain data coherency between host and device for any given data pipe. For more information on data toggle, refer to the USB 2.0 specification.

56.3.2.3.3.3.1 Data toggle reset

The DCD may reset the data toggle state bit and cause the data toggle sequence to reset in the device controller by writing a '1' to the data toggle reset bit in the USB_UOG_ENDPTCTRLx register. This should only be necessary when configuring/ initializing an endpoint or returning from a STALL condition.

56.3.2.3.3.3.2 Data toggle inhibit

NOTE

This feature is for test purposes only and should never be used during normal device controller operation.

Setting the *data toggle Inhibit bit* active ('1') causes the device controller to ignore the data toggle pattern that is normally sent and accept all incoming data packets regardless of the data toggle state.

In normal operation, the device controller checks the DATA0/DATA1 bit against the data toggle to determine if the packet is valid. If Data PID does not match the data toggle state bit maintained by the device controller for that endpoint, the Data toggle is considered not valid. If the data toggle is not valid, the device controller assumes the packet was already received and discards the packet (not reporting it to the DCD). To prevent the host controller from re-sending the same packet, the device controller will respond to the error packet by acknowledging it with either an ACK or NYET response.

56.3.2.3.3.3.3 Priming transmit endpoints

Priming a transmit endpoint will cause the device controller to fetch the device transfer descriptor (dTD) for the transaction pointed to by the device queue head (dQH). After the dTD is fetched, it will be stored in the dQH until the device controller completes the transfer described by the dTD. Storing the dTD in the dQH allows the device controller to fetch the operating context needed to handle a request from the host without the need to follow the linked list, starting at the dQH when the host request is received.

After the device has loaded the dTD, the leading data in the packet is stored in a FIFO in the device controller. This FIFO is split into virtual channels so that the leading data can be stored for any endpoint up to the maximum number of endpoints configured at device synthesis time.

After a priming request is complete, an endpoint state of primed is indicated in the USB_UOG_ENDPTSTATUS register. For a primed transmit endpoint, the device controller can respond to an IN request from the host and meet the stringent bus turnaround time of High Speed USB.

Because only the leading data is stored in the device controller FIFO, it is necessary for the device controller to begin filling in behind leading data after the transaction starts. The FIFO must be sized to account for the maximum latency that can be incurred by the system memory bus. More information about FIFO sizing is presented in section.

56.3.2.3.3.3.4 Priming receive endpoints

Priming receive endpoints is identical to priming of transmit endpoints from the point of view of the DCD. At the device controller, the major difference in the operational model is that there is no data movement of the leading packet data simply because the data is to be received from the host.

Note as part of the architecture, the FIFO for the receive endpoints is not partitioned into multiple channels like the transmit FIFO. Thus, the size of the RX FIFO does not scale with the number of endpoints.

56.3.2.3.4 Operational model For packet transfers

All transactions on the USB bus are initiated by the host and in turn, the device must respond to any request from the host within the turnaround time stated in the USB 2.0 Specification. At USB 1.1 Full or Low Speed rates, this turnaround time was significant and the USB 1.1 device controllers were architected so that the device controller could access main memory or interrupt a host protocol processor in order to respond to the USB 1.1 transaction. The architecture of the USB 2.0 device controller must be different because same methods will not meet USB 2.0 High-speed turnaround time requirements by simply increasing clock rate.

A USB host will send requests to the device controller in an order that cannot be precisely predicted as a single pipeline, so it is not possible to prepare a single packet for the device controller to execute. However, the order of packet requests is predictable when the endpoint number and direction is considered. For example, if endpoint 3 (transmit direction) is configured as a bulk pipe, then we can expect the host will send IN requests to that endpoint. This device controller is architected in such a way that it can prepare packets for each endpoint/direction in anticipation of the host request. The process of preparing the device controller to send or receive data in response to host initiated transaction on the bus is referred to as "priming" the endpoint. This term will be used throughout the following documentation to describe the device controller operation so the DCD can be architected properly use priming. Further, note that the term "flushing" is used to describe the action of clearing a packet that was queued for execution.

56.3.2.3.4.1 Interrupt/bulk endpoint operational model

The behaviors of the device controller for interrupt and bulk endpoints are identical. All valid IN and OUT transactions to bulk pipes will handshake with a NAK unless the endpoint had been primed. Once the endpoint has been primed, data delivery will commence.

A dTD will be retired by the device controller when the packets described in the transfer descriptor have been completed. Each dTD describes N packets to be transferred according to the USB Variable Length transfer protocol. The formula and table on the following page describe how the device controller computes the number and length of the packets to be sent/received by the USB vary according to the total number of bytes and maximum packet length.

With Zero Length Termination (ZLT) = 0

N = INT(Number Of Bytes/Max. Packet Length) + 1

With Zero Length Termination (ZLT) = 1

N = MAXINT(Number Of Bytes/Max. Packet Length)

Table 856. Variable Length Transfer Protocol Example (ZLT = 0)

Bytes (dTD)	Max. Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	

Table continues on the next page...

Table 856. Variable Length Transfer Protocol Example (ZLT = 0) (continued)

512	256	3	256	256	0
512	512	2	512	0	

Table 857. Variable Length Transfer Protocol Example (ZLT = 1)

Bytes (dTD)	Max. Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	
512	256	2	256	256	
512	512	1	512		

NOTE

The MULT field in the dQH must be set to "00" for bulk, interrupt, and control endpoints.

TX-dTD is complete when:

- All packets described dTD were successfully transmitted. *** Total bytes in dTD will equal zero when this occurs.

RX-dTD is complete when:

- All packets described in dTD were successfully received. *** Total bytes in dTD will equal zero when this occurs.
- A short packet (number of bytes < maximum packet length) was received. *** This is a successful transfer completion; DCD must check Total Bytes in dTD to determine the number of bytes that are remaining. From the total bytes remaining in the dTD, the DCD can compute the actual bytes received.
- A long packet was received (number of bytes > maximum packet size) OR (total bytes received > total bytes specified). *** This is an error condition. The device controller will discard the remaining packet, and set the Buffer Error bit in the dTD. In addition, the endpoint will be flushed and the USBERR interrupt will become active.

On the successful completion of the packet(s) described by the dTD, the active bit in the dTD will be cleared and the next pointer will be followed when the Terminate bit is clear. When the Terminate bit is set, the device controller will flush the endpoint/direction and cease operations for that endpoint/direction.

On the unsuccessful completion of a packet (see long packet above), the dQH will be left pointing to the dTD that was in error. In order to recover from this error condition, the DCD must properly reinitialize the dQH by clearing the active bit, and update the nextTD pointer before attempting to re-prime the endpoint.

NOTE

All packet level errors such as a missing handshake or CRC error will be retried automatically by the device controller.

There is no required interaction with the DCD for handling such errors.

56.3.2.3.4.1.1 Interrupt/bulk endpoint bus response matrix

The table below shows the response matrix for Interrupt/Bulk Endpoint Bus.

Table 858. Interrupt/Bulk Endpoint Bus Response Matrix

	Stall	Not Primed	Primed	Underflow	Overflow
Setup	Ignore	Ignore	Ignore	N/A	N/A

Table continues on the next page...

Table 858. Interrupt/Bulk Endpoint Bus Response Matrix (continued)

In	STALL	NAK	Transmit	BS Error	N/A
Out	STALL	NAK	Receive + NYET/ACK	N/A	NAK
Ping	STALL	NAK	ACK	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

NOTE

BS Error = Force Bit Stuff Error

NYET/ACK - NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.

SYSERR - System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

56.3.2.3.4.2 Control endpoint operation model

This section details the setup phase, data phase, status phase, and the control endpoint bus response matrix.

56.3.2.3.4.2.1 Setup phase

All requests to a control endpoint begin with a setup phase followed by an optional data phase and a required status phase. The device controller will always accept the setup phase unless the setup lockout is engaged.

The setup lockout will engage so that future setup packets are ignored. Lockout of setup packets ensures that while software is reading the setup packet stored in the queue head, that data is not written as it is being read potentially causing an invalid setup packet.

The setup lockout mechanism can be disabled and a new tripwire type semaphore will ensure that the setup packet payload is extracted from the queue head without being corrupted be an incoming setup packet. This is the preferred behavior because ignoring repeated setup packets due to long software interrupt latency would be a compliance issue.

- Disable Setup Lockout by writing 1 to Setup Lockout Mode (SLOM) in USB_nUSBMODE. (once at initialization). Setup lockout is not necessary when using the tripwire as described below.

NOTE

Leaving the Setup Lockout Mode As 0 will result in pre-2.3 hardware behavior.

- After receiving an interrupt and inspecting UBS_nENDSETUPSTAT to determine that a setup packet was received on a particular pipe:
  1. Write 1 to clear corresponding bit UBS_nENDSETUPSTAT.
  2. Write 1 to Setup Tripwire (SUTW) in USB_nUSBCMD register.
  3. Duplicate contents of dQH.SetupBuffer into local software byte array.
  4. Read Setup TripWire (SUTW) in USB_nUSBCMD register. (if set - continue; if cleared - goto 2)
  5. Write 0 to clear Setup Tripwire (SUTW) in USB nUSBCMD register.
  6. Process setup packet using local software byte array copy and execute status/handshake phases.

NOTE

After receiving a new setup packet the status and/or handshake phases may still be pending from a previous control sequence. These should be flushed & deallocated before linking a new status and/or handshake dTD for the most recent setup packet.

56.3.2.3.4.2.2 Data phase

Following the setup phase, the DCD must create a device transfer descriptor for the data phase and prime the transfer.

After priming the packet, the DCD must verify a new setup packet has not been received by reading the USB.ENDPTSETUPSTAT register immediately verifying that the prime had completed. A prime will complete when the associated bit in the USB_nENDPTPRIME register is zero and the associated bit in the USB_nENDPTSTAT register is a one. If a prime fails, that is, the USB_nENDPTPRIME bit goes to zero and the USB_nENDPTSTAT bit is not set, then the prime has failed. This can only be due to improper setup of the dQH, dTD or a setup arriving during the prime operation. If a new setup packet is indicated after the ENDPTPRIME bit is cleared, then the transfer descriptor can be freed and the DCD must reinterpret the setup packet.

Should a setup arrive after the data stage is primed, the device controller will automatically clear the prime status (USB_nENDPTSTAT) to enforce data coherency with the setup packet.

NOTE

- The MULT field in the dQH must be set to "00" for bulk, interrupt, and control endpoints.
- Error handling of data phase packets is the same as bulk packets described previously.

56.3.2.3.4.2.3 Status phase

Similar to the data phase, the DCD must create a transfer descriptor (with byte length equal zero) and prime the endpoint for the status phase. The DCD must also perform the same checks of the USB.ENDPTSETUPSTAT as described above in the data phase.

NOTE

- The MULT field in the dQH must be set to 00 for bulk, interrupt, and control endpoints.
- Error handling of data phase packets is the same as bulk packets described previously.

56.3.2.3.4.2.4 Control endpoint bus response matrix

Shown in the following table is the device controller response to packets on a control endpoint according to the device controller state.

The table below shows the response matrix for the Control Endpoint Bus.

Table 859. Control Endpoint Bus Response Matrix

Token Type	Endpoint State					Setup Lockout
	Stall	Not Primed	Primed	Underflow	Overflow	
Setup	ACK	ACK	ACK	N/A	SYSERR	
In	STALL	NAK	Transmit	BS Error	N/A	N/A
Out	STALL	NAK	Receive + NYET/ACK	N/A	NAK	N/A
Ping	STALL	NAK	ACK	N/A	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore

BS Error = Force Bit Stuff Error

NYET/ACK - NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.

SYSERR - System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

56.3.2.3.4.3 Isochronous endpoint operational model

Isochronous endpoints are used for real-time scheduled delivery of data and their operational model is significantly different than the host throttled Bulk, Interrupt, and Control data pipes. Real time delivery by the device controller will be accomplished by the following:

- Exactly MULT Packets per (micro) Frame are transmitted/received. Note: MULT is a two-bit field in the device Queue Head. The variable length packet protocol is not used on isochronous endpoints.
- NAK responses are not used. Instead, zero length packets are sent in response to an IN request to an unprimed endpoints. For unprimed RX endpoints, the response to an OUT transaction is to ignore the packet within the device controller.
- Prime requests always schedule the transfer described in the dTD for the next (micro) frame. If the ISO-dTD is still active after that frame, then the ISO-dTD will be held ready until executed or canceled by the DCD.

An EHCI compatible host controller uses the periodic frame list to schedule data exchanges to Isochronous endpoints. The operational model for device mode does not use such a data structure. Instead, the same dTD used for Control/Bulk/Interrupt endpoints is also used for isochronous endpoints. The difference is in the handling of the dTD.

The first difference between bulk and ISO-endpoints is that priming an ISO-endpoint is a delayed operation such that an endpoint will become primed only after a SOF is received. After the DCD writes the prime bit, the prime bit will be cleared as usual to indicate to software that the device controller completed a priming the dTD for transfer. Internal to the design, the device controller hardware masks that prime start until the next frame boundary. This behavior is hidden from the DCD but occurs so that the device controller can match the dTD to a specific (micro)frame.

Another difference with isochronous endpoints is that the transaction must wholly complete in a (micro)frame. Once an ISO transaction is started in a (micro)frame, it will retire the corresponding dTD when MULT transactions occur or the device controller finds a fulfillment condition.

The transaction error bit set in the status field indicates a fulfillment error condition. When a fulfillment error occurs, the frame after the transfer failed to complete wholly, the device controller will force retire the ISO-dTD and move to the next ISO-dTD.

It is important to note that fulfillment errors are only caused due to partially completed packets. If no activity occurs to a primed ISO-dTD, the transaction will stay primed indefinitely. This means it is up to software discard transmit ISO-dTDs that pile up from a failure of the host to move the data.

Finally, the last difference with ISO packets is in the data level error handling. When a CRC error occurs on a received packet, the packet is not retried similar to bulk and control endpoints. Instead, the CRC is noted by setting the *Transaction Error* bit and the data is stored as usual for the application software to sort out.

- TX Packet Retired
  - MULT counter reaches zero.
  - Fulfillment Error [*Transaction Error* bit is set]
    - # Packets Occurred > 0 AND # Packets Occurred < MULT

    NOTE

    For TX-ISO, MULT Counter can be loaded with a lesser value in the dTD Multiplier Override field in hardware versions 2.3 and later. If the Multiplier Override is zero, the MULT Counter is initialized to the Multiplier in the QH.
- RX Packet Retired:
  - MULT counter reaches zero.
  - Non-MDATA Data PID is received**
    - ** Exit criteria only valid in hardware version 2.3 or later. Previous to hardware version 2.3, any PID sequence that did not match the MULT field exactly would be flagged as a transaction error due to PID mismatch or fulfillment error.
  - Overflow Error:

- Packet received is > maximum packet length. [*Buffer Error* bit is set]
- Packet received exceeds total bytes allocated in dTD. [*Buffer Error* bit is set]

— Fulfillment Error [*Transaction Error* bit is set]

- # Packets Occurred > 0 AND # Packets Occurred < MULT

— CRC Error [*Transaction Error* bit is set]

NOTE

For ISO, when a dTD is retired, the next dTD is primed for the next frame. For continuous (micro)frame to (micro)frame operation the DCD should ensure that the dTD linked-list is out ahead of the device controller by at least two (micro)frames.

56.3.2.3.4.3.1 Isochronous pipe synchronization

When it is necessary to synchronize an isochronous data pipe to the host, the (micro) frame number (USB_UOG_FRINDEX register) can be used as a marker. To cause a packet transfer to occur at a specific (micro) frame number [N], the DCD should interrupt on SOF during frame N-1. When the USB_UOG_FRINDEX=N-1, the DCD must write the prime bit. The device controller will prime the isochronous endpoint in (micro) frame N-1 so that the device controller will execute delivery during (micro) frame N.

NOTE

Priming an endpoint towards the end of (micro) frame N-1 will not guarantee delivery in (micro) frame N. The delivery may actually occur in (micro) frame N+1 if device controller does not have enough time to complete the prime before the SOF for packet N is received.

56.3.2.3.4.3.2 Isochronous endpoint bus response matrix

The following table shows the response matrix for the Isochronous Endpoint Bus.

Table 860. Isochronous Endpoint Bus Response Matrix

	Stall	Not Primed	Primed	Underflow	Overflow
Setup	STALL	STALL	STALL	N/A	N/A
In	NULL Packet	NULL Packet	Transmit	BS Error	N/A
Out	Ignore	Ignore	Receive	N/A	Drop Packet
Ping	Ignore	Ignore	Ignore	Ignore	Ignore
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

1. BS Error = Force Bit Stuff Error

   NULL Packet = Zero Length Packet

56.3.2.3.5 Managing queue heads

The following figure shows the End Point Queue Head.

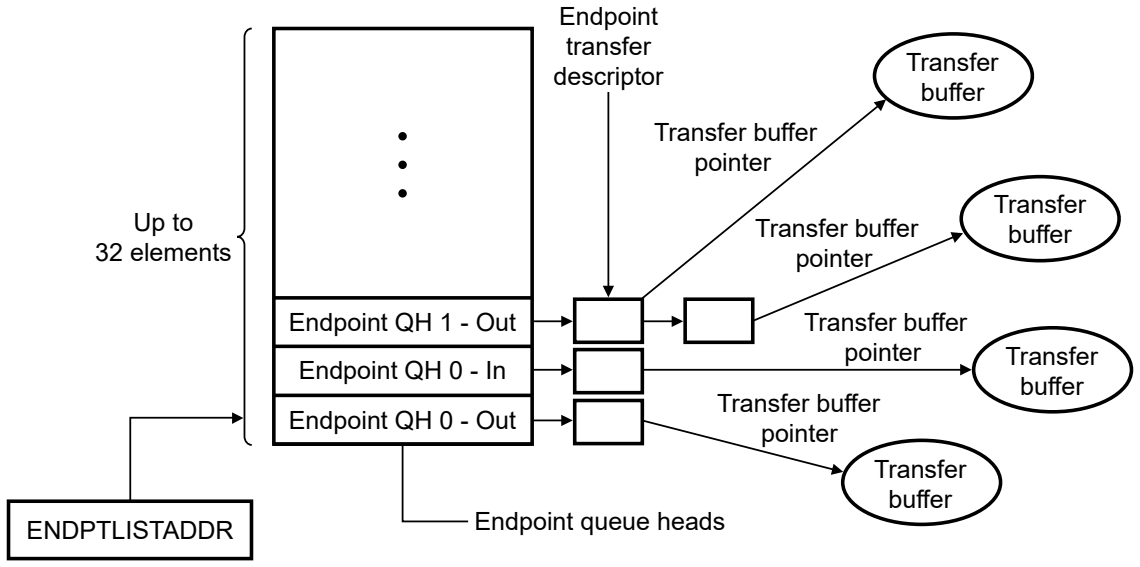


Figure 423. End Point Queue Head Diagram

The device queue head (dQH) points to the linked list of transfer tasks, each depicted by the device Transfer Descriptor (dTD). An area of memory pointed to by USB.ENDPOINTLISTADDR contains a group of all dQH's in a sequential list as shown in Figure 423. The even elements in the list of dQH's are used for receive endpoints (OUT/SETUP) and the odd elements are used for transmit endpoints (IN/INTERRUPT). Device transfer descriptors are linked head to tail starting at the queue head and ending at a terminate bit. Once the dTD has been retired, it will no longer be part of the linked list from the queue head. Therefore, software is required to track all transfer descriptors because pointers will no longer exist within the queue head once the dTD is retired (see section Software link pointers).

In addition to the current and next pointers and the dTD overlay examined in section Operational model For packet transfers, the dQH also contains the following parameters for the associated endpoint: Multiplier, Maximum Packet Length, Interrupt On Setup. The complete initialization of the dQH including these fields is demonstrated in the next section.

56.3.2.3.5.1 Queue head initialization

One device queue head must be initialized for each active endpoint. To initialize a device queue head:

- Write the wMaxPacketSize field as required by the USB Chapter 9 or application specific protocol.
- Write the multiplier field to 0 for control, bulk, and interrupt endpoints. For ISO endpoints, set the multiplier to 1, 2, or 3 as required bandwidth and in conjunction with the USB Chapter 9 protocol.

NOTE

In FS mode, the multiplier field can only be 1 for ISO endpoints.

- Write the next dTD Terminate bit field to 1.
- Write the Active bit in the status field to 0.
- Write the Halt bit in the status field to 0.

NOTE

The DCD must only modify dQH if the associated endpoint is not primed and there are no outstanding dTD's.

56.3.2.3.5.2 Operational model for setup transfers

As discussed in section Control endpoint operation model, setup transfer requires special treatment by the DCD. A setup transfer does not use a dTD but instead stores the incoming data from a setup packet in an 8-byte buffer within the dQH.

Upon receiving notification of the setup packet, the DCD should handle the setup transfer as demonstrated here:

1. Copy setup buffer contents from dQH - RX to software buffer.
2. Acknowledge setup backup by writing a "1" to the corresponding bit in ENDPTSETUPSTAT.

NOTE

- The acknowledge must occur before continuing to process the setup packet.
- After the acknowledge has occurred, the DCD must not attempt to access the setup buffer in the dQH - RX. Only the local software copy should be examined.

3. Check for pending data or status dTD's from previous control transfers and flush if any exist as discussed in section Flushing/de-priming an endpoint.
4. Decode setup packet and prepare data phase [optional] and status phase transfer as required by the USB Chapter 9 or application specific protocol.

NOTE

It is possible for the device controller to receive setup packets before previous control transfers complete. Existing control packets in progress must be flushed and the new control packet completed.

56.3.2.3.6 Managing transfers with transfer descriptors

These sections detail how to build, execute, and complete transfers with transfer descriptors.

56.3.2.3.6.1 Software link pointers

It is necessary for the DCD software to maintain head and tail pointers for the linked list of dTDs for each respective queue head. This is necessary because the dQH only maintains pointers to the current working dTD and the next dTD to be executed. The operations described in next section for managing dTD will assume the DCD can use reference the head and tail of the dTD linked list. The following figure shows the Software Link Pointers.

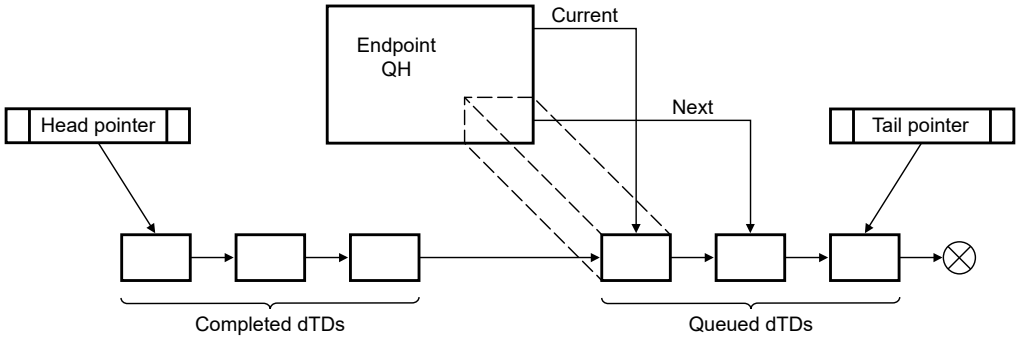


Figure 424. Software Link Pointers

NOTE

To conserve memory, the reserved fields at the end of the dQH can be used to store the Head & Tail pointers, but it still remains the responsibility of the DCD to maintain the pointers.

56.3.2.3.6.2 Building a transfer descriptor

Before a transfer can be executed from the linked list, a dTD must be built to describe the transfer. Use the following procedure for building dTDs.

Allocate 8-DWord dTD block of memory aligned to 8-DWord boundaries. Example: bit address 4:0 would be equal to "00000"

Write the following fields:

1. Initialize first 7 DWords to 0.
2. Set the terminate bit to 1.
3. Fill in total bytes with transfer size.
4. Set the interrupt on complete if desired.
5. Initialize the status field with the active bit set to 1 and all remaining status bits set to 0.
6. Fill in buffer pointer page 0 and the current offset to point to the start of the data buffer.
7. Initialize buffer pointer page 1 through page 4 to be one greater than each of the previous buffer pointer.

56.3.2.3.6.3 Executing a transfer descriptor

To safely add a dTD, the DCD must be follow this procedure which will handle the event where the device controller reaches the end of the dTD list at the same time a new dTD is being added to the end of the list.

Determine whether the link list is empty: Check DCD driver to see if pipe is empty (internal representation of linked-list should indicate if any packets are outstanding).

- Case 1: Link list is empty
    1. Write dQH next pointer AND dQH terminate bit to 0 as a single DWord operation.
    2. Clear active & halt bit in dQH (in case set from a previous error).
    3. Prime endpoint by writing 1 to correct bit position in ENDPTPRIME. Software should check that the dTD/dQH content in external memory is updated before setting the bit in ENDPTPRIME register.
- Case 2: Link list is not empty
    1. Add dTD to end of linked list.
    2. Read correct prime bit in ENDPTPRIME - if 1 DONE.
    3. Set ATDTW bit in USBCMD register to 1.
    4. Read correct status bit in ENDPTSTAT. (store in tmp. variable for later)
    5. Read ATDTW bit in USBCMD register.
        - If 0 goto 3.
        - If 1 continue to 6.
    6. Write ATDTW bit in USBCMD register to 0.
    7. If status bit read in (3) is 1 DONE.
    8. If status bit read in (3) is 0, then Goto Case 1: Step 1.

56.3.2.3.6.4 Transfer completion

After a dTD has been initialized and the associated endpoint primed the device controller will execute the transfer upon the host-initiated request. The DCD will be notified with a USB interrupt if the Interrupt On Complete bit was set or alternately, the DCD can poll the endpoint complete register to find when the dTD had been executed. After a dTD has been executed, DCD can check the status bits to determine success or failure.

NOTE

Multiple dTD can be completed in a single endpoint complete notification. After clearing the notification, DCD must search the dTD linked list and retire all dTDs that have finished (Active bit cleared).

By reading the status fields of the completed dTDs, the DCD can determine if the transfers completed successfully. Success is determined with the following combination of status bits:

- Active = 0

- Halted = 0
- Transaction Error = 0
- Data Buffer Error = 0

Should any combination other than the one shown above exist, the DCD must take proper action. Transfer failure mechanisms are indicated in the Device error matrix.

In addition to checking the status bit the DCD must read the Transfer Bytes field to determine the actual bytes transferred. When a transfer is complete, the Total Bytes transferred is decremented by the actual bytes transferred. For Transmit packets, a packet is only complete after the actual bytes reaches zero, but for receive packets, the host may send fewer bytes in the transfer according to the USB variable length packet protocol.

56.3.2.3.6.5 Flushing/de-priming an endpoint

It is necessary for the DCD to flush to de-prime one more endpoints on a USB device reset or during a broken control transfer. There may also be application specific requirements to stop transfers in progress. The following procedure can be used by the DCD to stop a transfer in progress:

1. Write a '1' to the corresponding bit(s) in ENDPTFLUSH.
2. Wait until all bits in ENDPTFLUSH are '0'.
   - Software note: this operation may take a large amount of time depending on the USB bus activity. It is not desirable to have this wait loop within an interrupt service routine.
3. Read ENDPTSTAT register to ensure that for all endpoints commanded to be flushed, that the corresponding bits are now '0'. If the corresponding bits are '1' after step #2 has finished, then the flush failed as described in the following:
   - Explanation: In very rare cases, a packet is in progress to the particular endpoint when commanded flush using ENDPTFLUSH register. A safeguard is in place to refuse the flush to ensure that the packet in progress completes successfully. The DCD may need to repeatedly flush any endpoints that fail to flush be repeating steps 1-3 until each endpoint is successfully flushed.

56.3.2.3.6.6 Device error matrix

The following table summarizes packet errors that are not automatically handled by the Device Controller.

Table 861. Device Error Matrix

Error	Direction	Packet Type	Data Buffer Error Bit	Transaction Error Bit
Overflow **	RX	Any	1	0
ISO Packet Error	RX	ISO	0	1
ISO Fulfillment Error	Both	ISO	0	1

Notice that the device controller handles all errors on Bulk/Control/Interrupt Endpoints except for a data buffer overflow. However, for ISO endpoints, errors packets are not retried and errors are tagged as indicated. The table below describes the errors.

Table 862. Error Descriptions

Error	Description
Overflow	Number of bytes received exceeded max. packet size or total buffer length. ** This error will also set the Halt bit in the dQH and if there are dTDs remaining in the linked list for the endpoint, then those will not be executed.

Table continues on the next page...

Table 862. Error Descriptions (continued)

Error	Description
ISO Packet Error	CRC Error on received ISO packet. Contents not guaranteed to be correct.
ISO Fulfillment Error	Host failed to complete the number of packets defined in the dQH mult field within the given (micro)frame. For scheduled data delivery the DCD may need to readjust the data queue because a fulfillment error will cause Device Controller to cease data transfers on the pipe for one (micro)frame. During the "dead" (micro)frame, the Device Controller reports error on the pipe and primes for the following frame.

56.3.2.3.7 Servicing interrupts

The interrupt service routine must consider that there are high-frequency, low-frequency operations, and error operations and order accordingly.

56.3.2.3.7.1 High-frequency interrupts

High frequency interrupts in particular should be handled in the order below. The most important of these are listed first because the DCD must acknowledge a setup buffer in the timeliest manner possible.

The table below describes the High frequency interrupt events.

Table 863. High Frequency Interrupt Events

Execution Order	Interrupt	Action
1a	USB Interrupt - USB.ENDPTSETUPSTATUS	Copy contents of setup buffer and acknowledge setup packet (as indicated in Figure 423 shows the End Point Queue Head). Process setup packet according to USB 2.0 Chapter 9 or application specific protocol.
1b	USB Interrupt[1] - USB.ENDPTCOMPLETE	Handle completion of dTD as indicated in Figure 423 shows the End Point Queue Head.
2	SOF Interrupt	Action as deemed necessary by application. This interrupt may not have a use in all applications.

1. It is likely that multiple interrupts to stack up on any call to the Interrupt Service Routine AND during the Interrupt Service Routine.

56.3.2.3.7.2 Low-frequency interrupts

The low frequency interrupts can be handled in any order because they do not occur often in comparison to the high-frequency interrupts.

The table below shows the Low frequency interrupt events.

Table 864. Low Frequency Interrupt Events

Interrupt	Action
Port Change	Change software state information.
Sleep Enable (Suspend)	Change software state information. Low power handling as necessary.
Reset Received	Change software state information. Abort pending transfers.

56.3.2.3.7.3 Error interrupts

Error interrupts will be least frequent and should be placed last in the interrupt service routine.

The following table shows the error interrupt events.

Table 865. Error Interrupt Events

Interrupt	Action
USB Error Interrupt	This error is redundant because it combines USB Interrupt and an error status in the dTD. The DCD will more aptly handle packet-level errors by checking dTD status field upon receipt of USB Interrupt (w/ USB.ENDPTCOMPLETE).
System Error	Unrecoverable error. Immediate Reset of core; free transfers buffers in progress and restart the DCD.

56.3.3 Modes

The USB has two main modes of operation: Normal mode and Low-Power mode. Each USB OTG controller core can operate in High Speed operation (480 Mbps), Full Speed operation (12 Mbps) and Low Speed operation (1.5 Mbps).

This chapter explains the operation modes.

56.3.3.1 Normal mode

The OTG controller core can operate in Host mode and Device (Peripheral) mode.

Each USB controller core has its corresponding port, which can work in one or more interface modes.

NOTE

Each controller supports only the interface type listed below. Selecting a different interface type in the PORTSC_PTS field results in unpredictable behavior and may cause the system to hang.

- OTG1 port
  - This port supports on-chip UTMI transceiver only.
- OTG2 port
  - This port supports on-chip UTMI transceiver only.

56.3.3.2 Low-power mode

Each USB controller core has a low-power mode (Suspend mode) to save power consumption.

As described in the USB 2.0 specification, the device can go into the Suspend state after it sees a constant Idle state on the upstream facing port. The OTG controller core enters Suspend mode after 3 ms of inactivity on the port when it is in Device Operation mode. Host controllers, including the OTG controller in Host mode, do not suspend automatically but can be placed in Suspend mode by software.

Either the local Arm platform or the remote USB Host/Peripheral can initiate a wake-up sequence to resume USB communication. For details about Suspend/Resume, see Power control.

56.3.3.2.1 Power control

The USB controller supports suspend and wake-up functionality. The power control block allows for placing the transceiver in USB low power mode when USB bus is IDLE, and supports local and remote wake-up to bring the transceiver out of USB low power mode when needed. Additionally, the power control block can wake-up the Arm platform from core sleep mode by generating an interrupt.

56.3.3.2.1.1 Entering low power suspend mode

In Host operation mode, low power suspend mode is entered as follows:

1. Clear the ASE and PSE bits in USB_USBCMD, and wait until the AS and PS bits in USB_USBSTS become "0."
2. Set the "SUSPEND" bit in USB_PORTSC1
3. Set VBUSVALID_TO_SESSVALID in USBPHYx_USB1_VBUS_DETECT
4. Set the "PHCD" bit in USB_PORTSC1
5. Set all PWD bits in USBPHYx_PWD
6. Set CLKGATE in USBPHYx_CTRL

NOTE

Step 4,5,6 shall be done in atomic operation. That is, interrupt should be disabled during these three steps.

For device operation mode, low power suspend mode is entered as follows:

1. After Host drive is IDLE for 3ms, an SLI interrupt is issued (the "DCSUSPEND" or "SLI" bit in USB_USBSTS)
2. Set VBUSVALID_TO_SESSVALID in USBPHYx_USB1_VBUS_DETECT
3. Set the "PHCD" bit on USB_PORTSC1
4. Set all PWD bits in USBPHYx_PWD
5. Set CLKGATE in USBPHYx_CTRL

NOTE

Step 3,4,5 shall be done in atomic operation. That is, interrupt should be disabled during these three steps.

56.3.3.2.1.2 Wake-up events

The power control block monitors the USB bus when the USB core is in the USB suspend state.

The core monitors a number of wake-up conditions, depending on whether it is on Host or Device mode. Upon detection of a wake-up condition, an interrupt will be generated to Arm platform if the related wake-up interrupt enable bit is set.

USB wake-up interrupt also re-activates the Arm platform clocks if they were stopped during the suspend.

56.3.3.2.1.2.1 Host mode events

The host controller wakes up on the following events:

- Remote Wake-Up Request

A peripheral can request the host to reactivate the bus by driving wake-up signaling on the DM/DP lines. The power control block sends a wake-up request to the USB core during the detection of a J-K transition on DM/DP line.

- Wake-Up On Overcurrent

While enabling Wake-Up On Overcurrent (WKOC bit in the USB core register PORTSC1 is set '1'), the power control block sends a wake-up request to the USB core during the detection of an overcurrent.

- Wake-Up On Disconnect

While enabling Wake-Up On Disconnect (WKDC bit in the USB core register PORTSC1 is set '1'), the power control block sends a wake-up request to the USB core during the detection of a disconnection event (J-SE0/K-SE0 transition on DM/DP line).

- Wake-Up On Connect

While enabling a Wake-Up On Connect (WKCN bit in the USB core register PORTSC1 is set '1'), the power control sub-block sends a wake-up request to the USB core during the detection of the connection event (SE0-J/SE0-K transition on DM/DP line).

For a detailed description of register bits WKOC, WKDC, WKCN, see USB_nPORTSC1.

56.3.4 Interrupts

56.3.4.1 USB core interrupts

Each USB core uses one dedicated vector in the Interrupt Table. In the Interrupt section, you can find the vector numbers for each core.

USB cores control all interrupt sources except wake-up interrupts. Refer to USB_nUSBINTR for details.

56.3.4.2 USB wake-up interrupts

Each USB Core has an associated wake-up interrupt. The wake-up interrupts are generated outside of the USB controller cores, but using the same vector as the corresponding USB controller cores interrupt. These interrupts are generated by the Power Control blocks which run on the 32 kHz standby clock. The wake-up interrupt is designed to work even when the USB and Arm platform clocks are disabled, such that a wake-up condition on the USB bus can re-activate the Arm platform clocks.

Because the wake-up interrupt is generated and cleared on a 32 kHz clock, this interrupt request responds very slowly to clear actions. For this reason, the software must disable the wake-up interrupt to clear the request flag. Disabling the interrupt masks the request instantaneously as this is clocked by the Arm platform clock. The software should wait for at least three 32 kHz clock cycles before re-enabling this interrupt to allow sufficient time for the request flag to clear. Because this interrupt is only used during low power modes of the USB, it is sufficient to enable the wake-up interrupt just prior to entering the USB suspend mode.

56.3.5 Clocking

The following table describes the clock sources for USB. Please see the clock control chapter for clock setting, configuration, and gating information.

Table 866. USB Clocks

Clock	Description
ipg_ahb_clk	AHB bus clock
ipg_clk_s	Peripheral access clock
ipg_clk_s_pl301	PL301 peripheral clock

56.4 External signals

The following table describes the external signals of USB. See the chip-specific information to know the external signals supported by your chip.

Table 867. USB External Signals

Signal	Description
OTGn_DN	DN OTG Signal
OTGn_DP	DP OTG Signal
OTGn_ID	ID signal
OTGn_OC	OTG External input for VBUS overcurrent detection
OTGn_PWR	To control PMIC to supply VBUS voltage

Table continues on the next page...

Table 867. USB External Signals (continued)

Signal	Description
USB2_VBUS	USB 5 V signal
USB2_TXRTUNE	Transmitter resistor tune pin

56.5 Initialization

This section provides the information required to initialize the USB controller properly. For more details on how to enable and disable, see Host controller initialization and Device controller initialization.

56.6 Application information

This section describes the detailed application knowledge for OTG1 and OTG2 ports.

56.6.1 Register interface

There are three categories of configuration, control, and status registers: identification, capability, and operational registers.

NOTE

USB control registers support only DWORD (32-bit) access.

- Use the identification registers that contain the complete set of the hardware configuration parameters to declare the peripheral interface presence.
- Static, read-only capability registers define the software limits, restrictions, and capabilities of the host/device controller.
- Operational registers are dynamic control or status registers that may be read only, read/write, or read/write-to-clear. The following sections define the use of these registers.

Complementary nature of host and device control show up by EHCI registers listed alongside device registers.

The following table describes the Interface register sets.

Table 868. Interface register sets

Offset	Register Set	Explanation
000h-07Ch	Identification registers	Use the identification registers that include a table of the hardware configuration parameters to declare the peripheral interface presence.
100h-124h	Capability registers	Capability registers specify the limits, restrictions, and capabilities of a host/device controller implementation. Use these values as parameters to the host or DCD.
080h-0FCh 140h-1FCh	Operational registers	System software uses operational registers to control and monitor the operational state of the host/device controller.

56.6.1.1 Configuration, control, and status register set

The following table describes the Device/Host capability registers.

NOTE

Depending on implementation, "*x*" can have the following values: UOG1, UOG2.

Table 869. Device/Host Capability Registers

Offset	Size (Bytes)	Mnemonic	Register Name	Device Mode	Host Mode
000h	4	USB_*x*_ID	Identification Register	O	O
004h	4	USB_*x*_HWGENERAL	General Hardware Parameters	O	O
008h	4	USB_*x*_HWHOST	Host Hardware Parameters	X	O
00Ch	4	USB_*x*_HWDEVICE	Device Hardware Parameters	O	X
010h	4	USB_*x*_HWTXBUF	TX Buffer Hardware Parameters	O	O
014h	4	USB_*x*_HWRXBUF	RX Buffer Hardware Parameters	O	O
018-07Fh		-	Reserved		
080h	4	USB_*x*_GPTIMER0LD	General Purpose Timer #0 Load Register	O	O
084h	4	USB_*x*_GPTIMER0CTRL	General Purpose Timer #0 Control Register	O	O
088h	4	USB_*x*_GPTIMER1LD	General Purpose Timer #1 Load Register	O	O
08Ch	4	USB_*x*_GPTIMER1CTRL	General Purpose Timer #1 Control Register	O	O
090h	4	USB_*x*_SBUSCFG	System Bus Interface Configuration Register	O	O
094-09Fh		-	Reserved		
100h	1	USB_*x*_CAPLENGTH	Capability Register Length	O	O
101h		-	Reserved		
102h	2	USB_*x*_HCIVERSION	Host Controller Interface Version Number	X	O
104h	4	USB_*x*_HCSPARAMS	Host Controller Structural Parameters	X	O
108h	4	USB_*x*_HCCPARAMS	Host Controller Capability Parameters	X	O
10C-11Fh		-	Reserved		
120h	2	USB_*x*_DCIVERSION	Device Controller Interface Version Number	O	X
122h	2	-	Reserved		
124h	4	USB_*x*_DCCPARAMS	Device Controller Capability Parameters	O	X
128-13Fh		-	Reserved		
140h	4	USB_*x*_USBCMD	USB Command Register	O	O

Table continues on the next page...

Table 869. Device/Host Capability Registers (continued)

Offset	Size (Bytes)	Mnemonic	Register Name	Device Mode	Host Mode
144h	4	USB_*x*_USBSTS	USB Status Register	O	O
148h	4	USB_*x*_USBINTR	USB Interrupt Enable Register	O	O
14Ch	4	USB_*x*_FRINDEX	USB Frame Index	O	O
150h	4	-	Reserved		
154h	4	USB_*x*_PERIODICLISTBASE	Frame List Base Address	X	O
		USB_*x*_DEVICEADDR	USB Device Address	O	X
158h	4	USB_*x*_ASYNCLISTADDR	Next Asynchronous List Address	X	O
	4	USB_*x*_ENDPOINTLISTADDR	Address at Endpoint list in memory	O	X
15Ch	4	-	Reserved		
160h	4	USB_*x*_BURSTSIZE	Programmable Burst Size	O	O
164h	4	USB_*x*_TXFILLTUNING	Host Transmit Pre-Buffer Packet Tuning	X	O
168h	4	-	Reserved		
170h	4	-	Reserved		
178h	4	USB_*x*_ENDPTNAK	Endpoint NAK register	O	X
17Ch	4	USB_*x*_ENDPTNAKEN	Endpoint NAK Enable register	O	X
180h	4	USB_*x*_CONFIGFLAG	Configured Flag Register	X	O
184h	4	USB_*x*_PORTSC1	Port Status/Control Register 1	O	O
188-1A3h		-	Reserved		
1A4h	4	USB_*x*_OTGSC	On-The-Go Status/Control Register (OTG only)	O	O
1A8h	4	USB_*x*_USBMODE	USB Controller Operating Mode	O	O
1ACh	4	USB_*x*_ENDPTSETUPSTAT	Endpoint Setup Status	O	X
1B0h	4	USB_*x*_ENDPTPRIME	Endpoint Initialization	O	X
1B4h	4	USB_*x*_ENDPTFLUSH	Endpoint De-Initialization	O	X

Table continues on the next page...

Table 869. Device/Host Capability Registers (continued)

Offset	Size (Bytes)	Mnemonic	Register Name	Device Mode	Host Mode
1B8h	4	USB_*x*_ENDPTSTATUS	Endpoint Status	O	X
1BCh	4	USB_*x*_ENDPTCOMPLETE	Endpoint Complete	O	X
1C0 1C4 ... 1DCh	64	USB_*x*_ENDPTCTRL0 USB_*x*_ENDPTCTRL1 .... USB_*x*_ENDPTCTRL7	Endpoint Control Register 0-7	O	X

NOTE

"O" means the register is available in host/device operation mode.

"X" means the register is reserved in host/device operation mode.

56.6.1.2 Identification registers

Identification registers are used to declare the target interface presence and include a table of the hardware configuration parameters.

56.6.1.3 OTG operations

56.6.1.3.1 Register bits

Register interface has behavior descriptions for Device mode and Host mode. However, for OTG operations, it is necessary to perform tasks independent of the Controller mode.

NOTE

The only way to transit the controller mode out of host or device mode is with the controller reset bit. Therefore, it is also necessary for the OTG tasks to be performed independent of a controller reset as well as independent of the controller mode.

The following figure shows the controller mode.

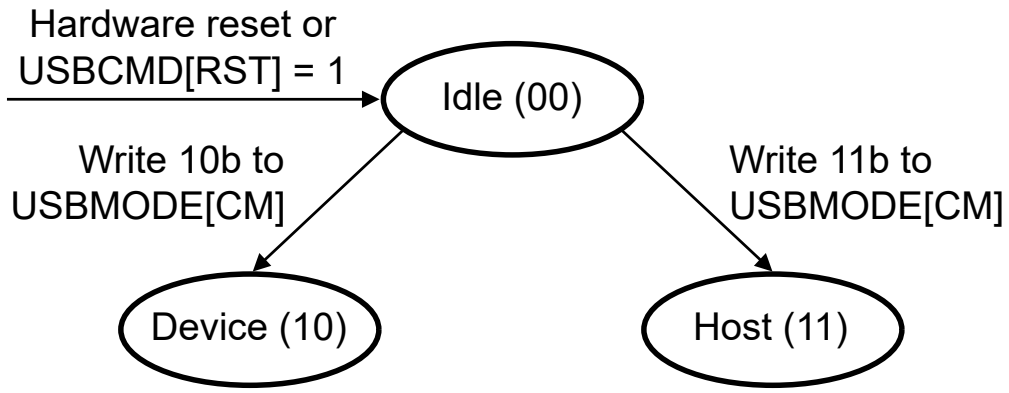


Figure 425. Controller Mode

To this end, listed below are the register bits that are used for OTG operations, which are independent of the controller mode and are also not affected by a write to the reset bit in the USBCMD register:

All Identification Registers

All Device/Host Capability Registers

OTGSC: All bits

PORTSC1:

- Physical Interface Select
- Physical Interface Serial Select
- Physical Interface Data Width
- Physical Interface Low Power
- Physical Interface Wake Signals
- Port Indicators
- Port Power

56.6.2 Host data structures

This section defines the interface data structures used to communicate control, status, and data between Host Controller Driver (HCD) (software) and the Enhanced Host Controller (hardware). The data structure definitions in this chapter support a 32-bit memory buffer address space. The interface consists of a Periodic Schedule, Periodic Frame List, Asynchronous Schedule, Isochronous Transaction Descriptors, Split-transaction Isochronous Transfer Descriptors, Queue Heads, and Queue Element Transfer Descriptors.

The periodic frame list is the root of all periodic (isochronous and interrupt transfer type) transfers for the host controller. The asynchronous list is the root for all the bulk and control transfers. Isochronous data streams are managed using Isochronous Transaction Descriptors. Isochronous split-transaction data streams are managed with Split-transaction Isochronous Transfer Descriptors. All Interrupt, Control, and Bulk data streams are managed via queue heads and Queue Element Transfer Descriptors. These data structures are optimized to reduce the total memory footprint of the schedule and to reduce (on average) the number of memory accesses needed to execute a USB transaction.

Note that software must ensure that no interface data structure reachable by the EHCI HC spans a 4 K-page boundary.

The data structures defined in this section are (from the host controller's perspective) a mix of read-only and read/writeable fields. The host controller must preserve the read-only fields on all data structure writes.

56.6.2.1 Periodic frame list

This schedule is for all periodic transfers (isochronous and interrupt). The periodic schedule is referenced from the operational registers space using the PERIODICLISTBASE address register and the FRINDEX register. The periodic schedule is based on an array of pointers called the Periodic Frame List. The PERIODICLISTBASE address register is combined with the FRINDEX register to produce a memory pointer into the frame list. The Periodic Frame List implements a sliding (1 msec) window of work over time.

The following figure shows the organization of periodic schedule.

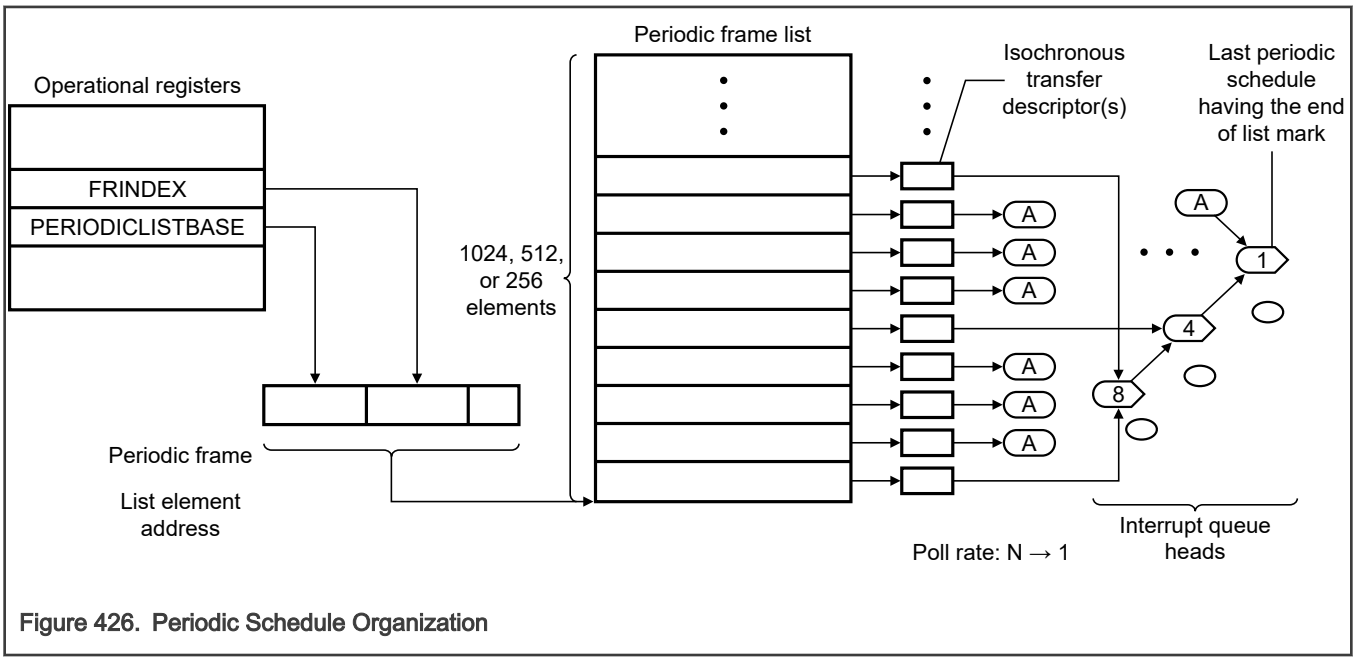


Figure 426. Periodic Schedule Organization

Split transaction Interrupt, Bulk, and Control are also managed using queue heads and queue element transfer descriptors.

The periodic frame list is a 4 K-page aligned array of Frame List Link pointers. The length of the frame list may be programmable. The programmability of the periodic frame list is exported to system software via the HCCPARAMS register. If non-programmable, the length is 1024 elements. If programmable, the length can be selected by system software as one of 8, 16, 32, 64, 128, 256, 512, or 1024 elements. An implementation must support all three sizes. Programming the size (that is, the number of elements) is accomplished by system software writing the appropriate value into Frame List Size field in the USBCMD register.

Frame List Link pointers direct the host controller to the first work item in the frame's periodic schedule for the current micro-frame. The link pointers are aligned on DWord boundaries within the Frame List.

The table below illustrates the format of the Frame list element pointer.

Table 870. Format of Frame List Element Pointer

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Frame List Link Pointer																											0		Typ			03-00 H

Frame List Link pointers always reference memory objects that are 32-byte aligned. The referenced object may be an isochronous transfer descriptor for high-speed devices, a split-transaction isochronous transfer descriptor (for full-speed isochronous endpoints), or a queue head (used to support high-, full- and low-speed interrupt). System software should not place non-periodic schedule items into the periodic schedule. The least significant bits in a frame list pointer are used to key the host controller as to the type of object the pointer is referencing.

The least significant bit is the T-Bit (bit 0). When this bit is set to a one, the host controller never uses the value of the frame list pointer as a physical memory pointer. The Typ field is used to indicate the exact type of data structure being referenced by this pointer. The value encodings are.

Table 871. Typ Field Value Definitions

Value	Meaning
00b	Isochronous Transfer Descriptor
01b	Queue Head
10b	Split Transaction Isochronous Transfer Descriptor.
11b	Frame Span Traversal Node.

56.6.2.2 Asynchronous list queue head pointer

The Asynchronous Transfer List (based at the USB_ASYNCLISTADDR register) is where all of the control and bulk transfers are managed. Host controllers use this list only when it reaches the end of the periodic list, the periodic list is disabled, or the periodic list is empty.

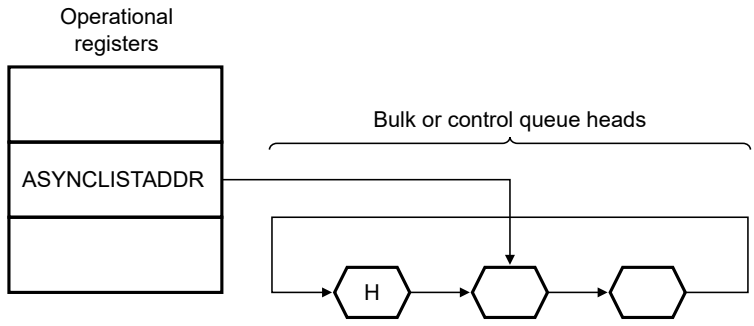


Figure 427. Asynchronous Schedule Organization

The Asynchronous list is a simple circular list of queue heads. The USB_ASYNCLISTADDR register is simply a pointer to the next queue head. This implements a pure round-robin service for all queue heads linked into the asynchronous list.

56.6.2.3 Isochronous (high-speed) transfer descriptor (iTD)

The format of an isochronous transfer descriptor is shown in the table below. This structure is used only for high-speed isochronous endpoints. All other transfer types should use queue structures. Isochronous TDs must be aligned on a 32-byte boundary.

Table 872. Isochronous Transaction Descriptor (iTD)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Next Link Pointer																											0		Typ		T	03-00H

Table continues on the next page...

Table 872. Isochronous Transaction Descriptor (iTD) (continued)

Status	Transaction 0 Length	IOC	PG*	Transaction 0 Offset*	07-04H
Status	Transaction 1 Length	IOC	PG*	Transaction 1 Offset*	0B-08H
Status	Transaction 2 Length	IOC	PG*	Transaction 2 Offset*	0F-0CH
Status	Transaction 3 Length	IOC	PG*	Transaction 3 Offset*	13-10H
Status	Transaction 4 Length	IOC	PG*	Transaction 4 Offset*	17-14H
Status	Transaction 5 Length	IOC	PG*	Transaction 5 Offset*	1B-18H
Status	Transaction 6 Length	IOC	PG*	Transaction 6 Offset*	1F-1CH
Status	Transaction 7 Length	IOC	PG*	Transaction 7 Offset*	23-20H

Buffer Pointer (Page 0)	EndPt	R	Device Address	27-24H
Buffer Pointer (Page 1)	I/O	Maximum Packet Size		2B-28H
Buffer Pointer (Page 2)	-		Mult	2F-2CH
Buffer Pointer (Page 3)	-			33-30H

Table continues on the next page...

Table 872. Isochronous Transaction Descriptor (iTD) (continued)

Buffer Pointer (Page 4)	-	37-34H
Buffer Pointer (Page 5)	-	3B-38H
Buffer Pointer (Page 6)	-	3F-3CH

Host controller read/write Host controller read-only

These fields may be modified by the host controller if the I/O field indicates an OUT.

56.6.2.3.1 Next link pointer

The first DWord of an iTD is a pointer to the next schedule data structure. The following table describes the Next Schedule Element pointer field.

Table 873. Next Schedule Element Pointer

Bit	Description
31-5 Link Pointer (LP)	These bits correspond to memory address signals [31:5], respectively. This field points to another Isochronous Transaction Descriptor (iTD/siTD) or Queue Head (QH).
4-3 Reserved	These bits are reserved and their value has no effect on operation. Software should initialize this field to zero.
2-1 QH/(s)iTD Select (Typ)	This field indicates to the Host Controller whether the item referenced is an iTD, siTD, or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor 11b FSTN (frame span traversal node)
0 Terminate (T)	1= Link Pointer field is not valid. 0= Link Pointer field is valid.

56.6.2.3.2 iTD transaction status and control list

DWords 1 through 8 are eight slots of transaction control and status. Each transaction description includes:

- Status results field
- Transaction length (bytes to send for OUT transactions and bytes received for IN transactions).
- Buffer offset. The PG and Transaction X Offset fields are used with the buffer pointer list to construct the starting buffer address for the transaction.

The host controller uses the information in each transaction description plus the endpoint information contained in the first three DWords of the Buffer Page Pointer list, to execute a transaction on the USB.

The following table describes iTD Transaction Status and Control fields.

Table 874. iTD Transaction Status and Control

Bit	Description	
31-28 Status	This field records the status of the transaction executed by the host controller for this slot. This field is a bit vector with the following encoding:	
	Bit	Definition
	31	Active. Set to one by software to enable the execution of an isochronous transaction by the Host Controller. When the transaction associated with this descriptor is completed, the Host Controller sets this bit to zero indicating that a transaction for this element should not be executed when it is next encountered in the schedule.
	30	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). If an overrun condition occurs, no action is necessary.
	29	Babble Detected. Set to one by the Host Controller during status update when "babble" is detected during the transaction generated by this descriptor.
	28	Transaction Error (XactErr). Set to one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). This bit may only be set for isochronous IN transactions.
27-16 Transaction X Length	For an OUT, this field is the number of data bytes the host controller sends during the transaction. The host controller is not required to update this field to reflect the actual number of bytes transferred during the transfer. For an IN, the initial value of the endpoint to deliver. During the status update, the host controller writes back the field is the number of bytes the host expects the number of bytes successfully received. The value in this register is the actual byte count (0‡zero length data, 1‡one byte, 2‡two bytes, and so on). The maximum value this field may contain is 0xC00 (3072).	
15 Interrupt On Complete (IOC)	If this bit is set to one, it specifies that when this transaction completes, the Host Controller should issue an interrupt at the next interrupt threshold.	
14-12 Page Select (PG)	These bits are set by software to indicate which of the buffer page pointers the offset field in this slot should be concatenated to produce the starting memory address for this transaction. The valid range of values for this field is 0 to 6.	
11-0 Transaction X Offset	This field is a value that is an offset, expressed in bytes, from the beginning of a buffer. This field is concatenated onto the buffer page pointer indicated in the adjacent *PG* field to produce the starting buffer address for this transaction.	

56.6.2.3.3 iTD buffer page pointer list (plus)

DWords 9-15 of an isochronous transaction descriptor are nominally page pointers (4 K aligned) to the data buffer for this transfer descriptor. This data structure requires the associated data buffer to be contiguous (relative to virtual memory), but allows the physical memory pages to be non-contiguous. Seven page pointers are provided to support the expression of eight isochronous transfers. The seven pointers allow for 3 (transactions) * 1024 (maximum packet size) * 8 (transaction records) (24576 bytes) to be moved with this data structure, regardless of the alignment offset of the first page.

Because each pointer is a 4 K aligned page pointer, the least significant 12 bits in several of the page pointers are used for other purposes.

The tables below illustrate the field descriptions.

Table 875. iTD Buffer Pointer Page 0 (Plus)

Bit	Description
31-12 Buffer Pointer (Page 0)	This is a 4 K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11-8 Endpoint Number (Endpt)	This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.
7 Reserved	Bit reserved for future use and should be initialized by software to zero.
6-0 Device Address	This field selects the specific device serving as the data source or sink.

Table 876. iTD Buffer Pointer Page 1 (Plus)

Bit	Description
31-12 Buffer Pointer (Page 1)	This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11 Direction (I/O)	0 = OUT; 1 = IN. This field encodes whether the high-speed transaction should use an IN or OUT PID.
10-0 Maximum Packet Size	This directly corresponds to the maximum packet size of the associated endpoint *(wMaxPacketSize)*. This field is used for high-bandwidth endpoints where more than one transaction is issued per transaction description (per micro-frame). This field is used with the *Multi* field to support high-bandwidth pipes. This field is also used for all IN transfers to detect packet babble. Software should not set a value larger than 1024 (400h). Any value larger yields undefined results.

Table 877. iTD Buffer Pointer Page 2 (Plus)

Bit	Description
31-12 Buffer Pointer	This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11-2 Reserved	This bit reserved for future use and should be set to zero.
1-0 Multi	This field is used to indicate to the host controller the number of transactions that should be executed per transaction description (per micro-frame). The valid values are:

Table continues on the next page...

Table 877. iTD Buffer Pointer Page 2 (Plus) (continued)

Bit	Description
	Value Meaning 00b Reserved. A zero in this field yields undefined results. 01b One transaction to be issued for this endpoint per micro-frame. 10b Two transactions to be issued for this endpoint per micro-frame. 11b Three transactions to be issued for this endpoint per micro-frame.

Table 878. iTD Buffer Pointer Page 3-6

Bit	Description
31-12 Buffer Pointer	This is a 4 K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11-0 Reserved	These bits reserved for future use and should be set to zero.

56.6.2.4 Split transaction isochronous transfer descriptor (siTD)

All Full-speed isochronous transfers through the internal transaction translator are managed using the siTD data structure. This data structure satisfies the operational requirements for managing the split transaction protocol.

The following table shows the Split Transaction Isochronous Transfer Descriptor (siTD).

Table 879. Split Transaction Isochronous Transfer Descriptor

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Next Link Pointer																											0		Typ		T	03-00
I/O	Port Number				-	Hub Addr										Reserved				EndPt		-	Device Address									07-04[1]
Reserved																μFrame C-mask							μFrame S-mask									0B-08[1]
ioc	P	Reserved				Total Bytes to Transfer										μFrame C-prog-mask							Status									0F-0C[2]
Buffer Pointer (Page 0)																				Current Offset												13-10[2]
Buffer Pointer (Page 1)																				Reserved							TP		T-count			17-14[2]

Table continues on the next page...

Table 879. Split Transaction Isochronous Transfer Descriptor (continued)

Back Pointer	0	T	1B-18

1. 04-0B: Static Endpoint State
2. 0C-13: Transfer results

Host controller read/write Host controller read-only

56.6.2.4.1 Next link pointer

DWord0 of a siTD is a pointer to the next schedule data structure.

The following table describes the Next Link Pointer fields.

Table 880. Next Link Pointer

Bit	Description
31-5	Next Link Pointer (LP). This field contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved. These bits must be written as zeros.
2-1	QH/(s)iTD Select (Typ). This field indicates to the Host Controller whether the item referenced is an iTD/siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor 11b FSTN (frame span traversal node)
0	Terminate (T). 1 = Link Pointer field is not valid. 0 = Link Pointer is valid.

56.6.2.4.2 siTD endpoint capabilities/characteristics

DWords 1 and 2 specify static information about the full-speed endpoint, the addressing of the parent Companion Controller, and micro-frame scheduling control.

The tables below describe the Endpoint and transaction translator characteristics and micro-frame schedule control fields.

Table 881. Endpoint and Transaction Translator Characteristics

Bit	Description
31	Direction (I/O).0 = OUT; 1 = IN. This field encodes whether the full-speed transaction should be an IN or OUT.
30-24	Port Number. This field is the port number of the recipient Transaction Translator.

Table continues on the next page...

Table 881. Endpoint and Transaction Translator Characteristics (continued)

Bit	Description
23	Reserved. Bit reserved and should be set to zero.
22-16	Hub Address. This field holds the device address of the Companion Controllers' hub.
15-12	Reserved. Field reserved and should be set to zero.
11-8	Endpoint Number (Endpt). This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.
7	Reserved. Bit is reserved for future use. It should be set to zero.
6-0	Device Address. This field selects the specific device serving as the data source or sink.

Table 882. Micro-frame Schedule Control

Bit	Description
31-16	Reserved. This field reserved for future use. It should be set to zero.
15-8	Split Completion Mask (mFrame C-Mask). This field (along with the *Active* and *SplitX- state* fields in the *Status* byte) is used to determine during which micro-frames the host controller should execute complete-split transactions. When the criteria for using this field is met, an all zeros value has undefined behavior. The host controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position where the m*Frame C-Mask* field is a one, then this siTD is a candidate for transaction execution. There may be more than one bit in this mask set.
7-0	Split Start Mask (mFrame S-mask). This field (along with the *Active* and *SplitX-state* fields in the *Status* byte) is used to determine during which micro-frames the host controller should execute start-split transactions. The host controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position where the m*Frame S-mask* field is a one, then this siTD is a candidate for transaction execution. An all zeros value in this field, in combination with existing periodic frame list has undefined results.

56.6.2.4.3 siTD transfer state

DWords 3-6 are used to manage the state of the transfer. The following table describes siTD transfer state fields.

Table 883. siTD Transfer Status and Control

Bit	Description
31	Interrupt On Complete (ioc). 0 = Do not interrupt when transaction is complete. 1 = Do interrupt when transaction is complete. When the host controller determines that the split transaction has completed it asserts a hardware interrupt at the next interrupt threshold.
30	Page Select (P). Used to indicate which data page pointer should be concatenated with the *CurrentOffset* field to construct a data buffer pointer (0 selects *Page 0* pointer and 1 selects *Page 1*). The host controller is not required to write this field back when the siTD is retired (*Active* bit transitioned from a one to a zero).

Table continues on the next page...

Table 883. siTD Transfer Status and Control (continued)

Bit	Description
29-26	Reserved. This field reserved for future use and should be set to zero.
25-16	Total Bytes To Transfer. This field is initialized by software to the total number of bytes expected in this transfer. Maximum value is 1023 (3FFh)
15-8	µFrame Complete-split Progress Mask (C-prog-Mask). This field is used by the host controller to record which split-completes has been executed.
7-0: Status—This field records the status of the transaction executed by the host controller for this slot. It is a bit vector with the encoding shown in the following rows.	
7	Active. Set to one by software to enable the execution of an isochronous split transaction by the Host Controller.
6	ERR. Set to a one by the Host Controller when an ERR response is received from the Companion Controller.
5	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). In the case of an under run, the Host Controller transmits an incorrect CRC (thus invalidating the data at the endpoint). If an overrun condition occurs, no action is necessary.
4	Babble Detected. Set to a one by the Host Controller during status update when" babble" is detected during the transaction generated by this descriptor.
3	Transaction Error (XactErr). Set to a one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). This bit is set only for IN transactions.
2	Missed Micro-Frame. The host controller detected that a host-induced hold- off caused the host controller to miss a required complete-split transaction.
1	Split Transaction State (SplitXstate). The bit encodings are: Value Meaning 00b Do Start Split. This value directs the host controller to issue a Start split transaction to the endpoint when a match is encountered in the S-mask. 01b Do Complete Split. This value directs the host controller to issue a Complete split transaction to the endpoint when a match is encountered in the C-mask.
0	Reserved. Bit reserved for future use and should be set to zero.

56.6.2.4.4 siTD buffer pointer list (plus)

DWords 4 and 5 are the data buffer page pointers for the transfer. This structure supports one physical page cross. The most significant 20 bits of each DWord in this section are the 4 K (page) aligned buffer pointers.

The least significant 12 bits of each DWord are used as additional transfer state. The following table describes the siTD buffer pointer fields.

Table 884. Buffer Page Pointer List (plus)

Bit	Description
31-12	Buffer Pointer List. Bits [31:12] of DWords 4 and 5 are 4 K page aligned physical memory addresses. These bits correspond to physical address bits [31:12] respectively. The lower 12 bits in each pointer are defined and used as specified below. The field *P* (see siTD transfer state) specifies the *current* active pointer.
Bits 11-0 (Page 0)	Current Offset—The 12 least significant bits of the Page 0 pointer are the current byte offset for the current page pointer (as selected with the page indicator bit (*P* field). The host controller is not required to write this field back when the siTD is retired (*Active* bit transitioned from a one to a zero).
Bits 11-0 (Page 1)—The least significant bits of the Page 1 pointer are split into three subfields as shown in the following rows.	
11-5 (Page 1)	Reserved
4-3 (Page 1)	Transaction position (TP). This field is used with T-count to determine whether to send *all*, *first*, *middle*, or *last* with each outbound transaction payload. System software must initialize this field with the appropriate starting value. The host controller must correctly manage this state during the lifetime of the transfer. The bit encodings are: Value Meaning 00b All. The entire full-speed transaction data payload is in this transaction (that is, less than or equal to 188 bytes). 01b Begin. This is the first data payload for a full- speed that is greater than 188 bytes. 10B Mid. This is the *middle* payload for a full-speed OUT transaction that is larger than 188 bytes. 11b End. This is the *last* payload for a full-speed OUT transaction that was larger than 188 bytes.
2-0 (Page 1)	Transaction count (T-Count). Software initializes this field with the number of OUT start-splits this transfer requires. Any value larger than 6 is undefined.

56.6.2.4.5 siTD back link pointer

DWord 6 of a siTD is simply another schedule link pointer. This pointer is always zero, or references a siTD, and it cannot reference any other schedule data structure.

The following table describes the siTD back link pointer fields.

Table 885. siTD Back Link Pointer

Bit	Description
31-5	siTD Back Pointer. This field is a physical memory pointer to a siTD.
4-1	Reserved. This field is reserved for future use. It should be set to zero.
0	Terminate (T). 1 = siTD Back Pointer field is not valid. 0 = siTD Back Pointer field is valid.

56.6.2.5 Queue element transfer descriptor (qTD)

This data structure is only used with a queue head. It describes one or more USB transactions to transfer up to 20480 (5*4096) bytes. The structure contains two structure pointers used for queue advancement, a DWord of transfer state, and a five-element array of data buffer pointers. It is 32 bytes and must be physically contiguous.

The buffer associated with this transfer must be virtually contiguous. The buffer may start on any byte boundary; however, for optimal utilization of on-chip buses it is recommended to align the buffers on a 32-byte boundary. A separate buffer pointer list element must be used for each physical page in the buffer, regardless of whether the buffer is physically contiguous.

Host controller updates (host controller writes) to stand-alone qTDs only occur during transfer retirement. References in the following bit field definitions of updates to the qTD are to the qTD portion of a queue head.

The following table shows the queue element transfer descriptor data structure.

Table 886. Queue element transfer descriptor data structure

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Next qTD Pointer																											0				T	03-00
Alternate Next qTD Pointer																											0				T	07-04
dt	Total Bytes to Transfer															ioc	C_Page			Cerr		PID Code		Status								0B-08[1]
Buffer Pointer (page 0)																				Current Offset												0F-0C[1]
Buffer Pointer (page 1)																				Reserved												13-10
Buffer Pointer (page 2)																				Reserved												17-14
Buffer Pointer (page 3)																				Reserved												1B-18
Buffer Pointer (page 4)																				Reserved												1F-1C

1. 08-0F: Transfer Results

Host controller read/write Host controller read-only

Queue Element Transfer Descriptors must be aligned on 32-byte boundaries.

56.6.2.5.1 Next qTD pointer

The first DWord of an element transfer descriptor is a pointer to another transfer element descriptor. The following table describes Next qTD pointer fields.

Table 887. qTD Next Element Transfer Pointer (DWord 0)

Bit	Description
31-5	Next Transfer Element Pointer. This field contains the physical memory address of the next qTD to be processed. The field corresponds to memory address signals[31:5], respectively.
4-1	Reserved
0	Terminate (T). 1= pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Host Controller that there are no more valid entries in the queue.

56.6.2.5.2 Alternate next qTD pointer

The second DWord of a queue element transfer descriptor is used to support hardware-only advance of the data stream to the next transfer descriptor on short packet. To be more explicit the host controller always uses this pointer when the current qTD is retired due to short packet. The following table describes the TD Alternate Next Element Transfer Pointer field descriptions.

Table 888. TD Alternate Next Element Transfer Pointer (DWord 1)

Bit	Description
31-5	Alternate Next Transfer Element Pointer. This field contains the physical memory address of the next qTD to be processed in the event that the current qTD execution encounters a short packet (for an IN transaction). The field corresponds to memory address signals [31:5], respectively.
4-1	Reserved
0	Terminate (T). 1= pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Host Controller that there are no more valid entries in the queue.

56.6.2.5.3 qTD token

The third DWord of a queue element transfer descriptor contains most of the information the host controller requires to execute a USB transaction (the remaining endpoint-addressing information is specified in the queue head).

NOTE

The field descriptions forward reference fields defined in the queue head. Where necessary, these forward references are preceded with a QH notation.

The following table describes the TD Token fields.

Table 889. TD Token (DWord 2)

Bit	Description
31 Data Toggle	This is the data toggle sequence bit. The use of this bit depends on the setting of the *Data Toggle Control* bit in the queue head.

Table continues on the next page...

Table 889. TD Token (DWord 2) (continued)

Bit	Description	
30-16 Total Bytes to Transfer	This field specifies the total number of bytes to be moved with this transfer descriptor. This field is decremented by the number of bytes actually moved during the transaction, only on the successful completion of the transaction. The maximum value software may store in this field is 5 * 4K (5000H). This is the maximum number of bytes 5 page pointers can access. If the value of this field is zero when the host controller fetches this transfer descriptor (and the active bit is set), the host controller executes a zero-length transaction and retires the transfer descriptor. It is not a requirement for OUT transfers that *Total Bytes To Transfer* be an even multiple of QHD.Maximum Packet Length. If software builds such a transfer descriptor for an OUT transfer, the last transaction is always less than QHD.Maximum Packet Length. Although it is possible to create a transfer up to 20K this assumes the 1st offset into the first page is 0. When the offset cannot be predetermined, crossing past the 5th page can be guaranteed by limiting the total bytes to 16K**. Therefore, the maximum recommended transfer is 16 K(4000H).	
15 Interrupt On Complete (IOC)	If this bit is set to a one, it specifies that when this qTD is completed, the Host Controller should issue an interrupt at the next interrupt threshold.	
14-12 Current Page (C_Page)	This field is used as an index into the qTD buffer pointer list. Valid values are in the range 0H to 4H. The host controller is not required to write this field back when the qTD is retired.	
11-10 Error Counter (CERR)	This field is a 2-bit down counter that keeps track of the number of consecutive Errors detected while executing this qTD. If this field is programmed with a non-zero value during set-up, the Host Controller decrements the count and writes it back to the qTD if the transaction fails. If the counter counts from one to zero, the Host Controller marks the qTD inactive, sets the *Halted* bit to a one, and error status bit for the error that caused *CERR* to decrement to zero. An interrupt is generated if the *USB Error Interrupt Enable* bit in the USBINTR register is set to a one. If HCD programs this field to zero during set-up, the Host Controller does not count errors for this qTD and there is no limit on the retries of this qTD. Note that write-backs of intermediate execution state are to the queue head overlay area, not the qTD. Transaction Error - Decrement Data Buffer Error - No Decrement[3] Stalled - No Decrement[1] Babble Detected - No Decrement[1] No Error - No Decrement[2]	
	Error	Decrement Counter
	1	Detection of Babble or Stall automatically halts the queue head. Thus, count is not decremented
	2	If the EPS field indicates a HS device or the queue head is in the Asynchronous Schedule (and *PIDCode* indicates an IN or OUT) and a bus transaction completes and the host controller does not detect a transaction error, then the host controller should reset *CERR* to extend the total number of errors for this transaction. For example, *CERR* should be reset with maximum value (3) on each successful

Table continues on the next page...

Table 889. TD Token (DWord 2) (continued)

Bit	Description	
		completion of a transaction. The host controller must never reset this field if the value at the start of the transaction is 00b. See Split transaction interrupt for CERR adjustment rules when the EPS field indicates a FS or LS device and the queue head is in the Periodic Schedule. See Asynchronous - do complete split for CERR adjustment rules when the EPS field indicates a FS or LS device, the queue head is in the Asynchronous schedule and the *PIDCode* indicates a SETUP.
	3	Data buffer errors are host problems. They don't count against the device's retries.
		NOTE Software must not program CERR to a value of zero when the EPS field is programmed with a value indicating a Full- or Low-speed device. This combination could result in undefined behavior.
9-8 PID Code	This field is an encoding of the token, which should be used for transactions associated with this transfer descriptor. Encodings are:	
	00b	OUT Token generates token (E1H)
	01b	IN Token generates token (69H)
	10b	SETUP Token generates token (2DH) (undefined if endpoint is an interrupt, the queue head is non-zero) transfer type, for example, µ*Frame S-mask* field in.
	11b	Reserved
7-0 Status	This field is used by the Host Controller to communicate individual command execution states back to HCD. This field contains the status of the last transaction performed on this qTD. The bit encodings are:	
	Bit	Status Field Description
	7	Active. Set to one by software to enable the execution of transactions by the Host Controller.
	6	Halted. Set to one by the Host Controller during status updates to indicate that a serious error has occurred at the device/endpoint addressed by this qTD. This can be caused by babble, the error counter counting down to zero, or reception of the STALL handshake from the device during a transaction. Any time that a transaction results in the Halted bit being set to a one, the Active bit is also set to zero.
	5	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). If an overrun condition occurs, the Host Controller forces a timeout condition on the USB, invalidating the transaction at the source. If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.

Table continues on the next page...

Table 889. TD Token (DWord 2) (continued)

Bit	Description	
	4	Babble Detected. Set to a one by the Host Controller during status update when" babble" is detected during the transaction. In addition to setting this bit, the Host Controller also sets the *Halted* bit to a one. Because "babble" is considered a fatal error for the transfer, setting the Halted bit to a one insures that no more transactions occur because of this descriptor.
	3	Transaction Error (XactErr). Set to a one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
	2	Missed Micro-Frame. This bit is ignored unless the *QH.EPS* field indicates a full- or low-speed endpoint and the queue head is in the periodic list. This bit is set when the host controller detected that a host-induced hold-off caused the host controller to miss a required complete-split transaction. If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
	1	Split Transaction State (SplitXstate). This bit is ignored by the host controller unless the *QH.EPS* field indicates a full- or low-speed endpoint. When a Full- or Low-speed device, the host controller uses this bit to track the state of the split-transaction. The functional requirements of the host controller for managing this state bit and the split transaction protocol depends on whether the endpoint is in the periodic or asynchronous schedule. The bit encodings are: Value Meaning 0b Do Start Split. This value directs the host controller to issue a Start split transaction to the endpoint. 1b Do Complete Split. This value directs the host controller to issue a Complete split transaction to the endpoint.
	0	Ping State (P)/ERR. If the *QH.EPS* field indicates a High-speed device and the *PID_Code* indicates an OUT endpoint, then this is the state bit for the Ping protocol. The bit encodings are: Value Meaning 0b Do OUT. This value directs the host controller to issue an OUT PID to the endpoint. 1b Do Ping. This value directs the host controller to issue a PING PID to the endpoint. If the *QH.EPS* field does not indicate a High-speed device, then this field is used as an error indicator bit. It is set to a one by the host controller whenever a periodic split-transaction receives an ERR handshake.

56.6.2.5.4 qTD buffer page pointer list

The last five DWords of a queue element transfer descriptor is an array of physical memory address pointers. These pointers reference the individual pages of a data buffer.

System software initializes Current Offset field to the starting offset into the current page, where current page is selected through the value in the C_Page field.

The following table describes the qTD Buffer Pointer(s) (DWords 3-7) fields.

Table 890. qTD Buffer Pointer(s) (DWords 3-7)

Bit	Description
31-12	Buffer Pointer List. Each element in the list is a 4 K page aligned physical memory address. The lower 12 bits in each pointer are reserved (except for the first one), as each memory pointer must reference the start of a 4 K page. The field C_Page specifies the current active pointer. When the transfer element descriptor is fetched, the starting buffer address is selected using C_Page (similar to an array index to select an array element). If a transaction spans a 4K buffer boundary, the host controller must detect the page-span boundary in the data stream, increment C_Page and advance to the next buffer pointer in the list, and conclude the transaction through the new buffer pointer.
11-0	Current Offset (Reserved). This field is reserved in all pointers except the first one (for example Page 0). The host controller should ignore all reserved bits. For the page 0 current offset interpretation, this field is the byte offset into the current page (as selected by C_Page). The host controller is not required to write this field back when the qTD is retired. Software should ensure the Reserved fields are initialized to zero.

56.6.2.6 Queue head

The table located in this section shows the Queue Head structure layout.

The following table shows the queue head structure layout.

Table 891. Queue Head Structure Layout

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Queue Head Horizontal Link Pointer																											0		Typ		T	03-00
RL				C	Maximum Packet Length											H	dtc	EP		EndPt		I	Device Address									07-04[1]
Mult		Port Number[2]								Hub Addr[2]						μFrame C-mask[2]								μFrame S-mask								0B-08[1]
Current qTD Pointer																											0					0F-0C
Next qTD Pointer																											0				T	13-10[3]
Alternate Next qTD pointer																											NakCnt				T	17-14[4]
dt	Total Bytes to Transfer															ioc	C_Page			Cerr		PID Code		Status								1B-18

Table continues on the next page...

Table 891. Queue Head Structure Layout (continued)

Buffer Pointer (Page 0)	Current Offset		1F-1C
Buffer Pointer (Page 1)	Reserved	C-prog-mask[2]	23-20
Buffer Pointer (Page 2)	S-bytes[2]	FrameTag[2]	27-24[4]
Buffer Pointer (Page 3)	Reserved		2B-28
Buffer Pointer (Page 4)	Reserved		2F-2C[3]

1. 04-0B: Static endpoint state.
2. These fields are used exclusively to support split transactions to USB 2.0 hubs
3. 10-2F: Transfer overlay.
4. 14-27: Transfer results.

Host controller read/write Host controller read-only

56.6.2.6.1 Queue head horizontal link pointer

The first DWord of a Queue Head contains a link pointer to the next data object to be processed after any required processing in this queue has been completed, as well as the control bits defined below.

This pointer may reference a queue head or one of the isochronous transfer descriptors. It must not reference a queue element transfer descriptor.

The following table describes the Queue head DWord 0 fields.

Table 892. Queue Head DWord 0

Bit	Description
31-5	Queue Head Horizontal Link Pointer (QHLP). This field contains the address of the next data object to be processed in the horizontal list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	QH/(s)iTD Select (Typ). This field indicates to the hardware whether the item referenced by the link pointer is an iTD, siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor)

Table continues on the next page...

Table 892. Queue Head DWord 0 (continued)

	11b FSTN (frame span traversal node)
0	Terminate (T). 1=Last QH (pointer is invalid). 0=Pointer is valid. If the queue head is in the context of the periodic list, a one bit in this field indicates to the host controller that this is the end of the periodic list. This bit is ignored by the host controller when the queue head is in the Asynchronous schedule. Software must ensure that queue heads reachable by the host controller always have valid horizontal link pointers.

56.6.2.6.2 Queue head endpoint capabilities/characteristics

The second and third DWords of a Queue Head specifies static information about the endpoint. This information does not change over the lifetime of the endpoint. There are three types of information in this region:

- Endpoint Characteristics. These are the USB endpoint characteristics including addressing, maximum packet size, and endpoint speed.
- Endpoint Capabilities. These are adjustable parameters of the endpoint. They effect how the endpoint data stream is managed by the host controller.
- Split Transaction Characteristics. This data structure is used to manage full- and low-speed data streams for bulk, control, and interrupt via split transactions to USB2.0 Hub Transaction Translator. There are additional fields used for addressing the hub and scheduling the protocol transactions (for periodic).

The host controller must not modify the bits in this region.

The following table describes the Endpoint characteristics: Queue head DWord 1 fields.

Table 893. Endpoint Characteristics: Queue Head DWord 1

Bit	Description	
31-28	Nak Count Reload (RL). This field contains a value, which is used by the host controller to reload the Nak Counter field.	
27	Control Endpoint Flag (C). If the *QH.EPS* field indicates the endpoint is not a high-speed device, and the endpoint is a control endpoint, then software must set this bit to a one. Otherwise, it should always set this bit to zero.	
26-16	Maximum Packet Length. This directly corresponds to the maximum packet size of the associated endpoint (*wMaxPacketSize*). The maximum value this field may contain is 0x400 (1024).	
15	Head of Reclamation List Flag (H). This bit is set by System Software to mark a queue head as being the head of the reclamation list.	
14	Data Toggle Control (DTC). This bit specifies where the host controller should get the initial data toggle on an overlay transition. 0b Ignore DT bit from incoming qTD. Host controller preserves DT bit in the queue head. 1b Initial data toggle comes from incoming qTD DT bit. Host controller replaces DT bit in the queue head from the DT bit in the qTD.	
13-12	Endpoint Speed (EPS). This is the speed of the associated endpoint. Bit combinations are:	
	Value	Meaning
	00b	Full-Speed (12 Mbits/sec)

Table continues on the next page...

Table 893. Endpoint Characteristics: Queue Head DWord 1 (continued)

	01b	Low-Speed (1.5 Mbits/sec)
	10b	High-Speed (480 Mbits/sec)
	11b	Reserved
	This field must not be modified by the host controller.	
11-8	Endpoint Number (Endpt). This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.	
7	Inactivate on Next Transaction (I). This bit is used by system software to request that the host controller set the Active bit to zero. See Rebalancing the periodic schedule , for full operational details. This field is only valid when the queue head is in the Periodic Schedule and the *EPS* field indicates a Full or Low-speed endpoint. Setting this bit to one when the queue head is in the Asynchronous Schedule or the *EPS* field indicates a high-speed device yields undefined results.	
6-0	Device Address. This field selects the specific device serving as the data source or sink.	

The table below describes the Endpoint capabilities: Queue head DWord 2 field descriptions.

Table 894. Endpoint Capabilities: Queue Head DWord 2

Bit	Description
31-30	High-Bandwidth Pipe Multiplier (Mult). This field is a multiplier used to key the host controller as the number of successive packets the host controller may submit to the endpoint in the current execution. The host controller makes the simplifying assumption that software properly initializes this field (regardless of location of queue head in the schedules or other run time parameters). The valid values are: Value Meaning 00b Reserved. A zero in this field yields undefined results. 01b One transaction to be issued for this endpoint per micro-frame. 10b Two transactions to be issued for this endpoint per micro-frame. 11b Three transactions to be issued for this endpoint per micro-frame.
29-23	Port Number. This field is ignored by the host controller unless the *EPS* field indicates a full- or low-speed device. The value is the port number identifier on the USB 2.0 Hub (for hub at device address *Hub Addr* below), below which the full- or low-speed device associated with this endpoint is attached. This information is used in the split-transaction protocol.
22-16	Hub Addr. This field is ignored by the host controller unless the *EPS* field indicates a full-or low-speed device. The value is the USB device address of the USB 2.0 Hub below which the full- or low-speed device associated with this endpoint is attached. This field is used in the split-transaction protocol.
15-8	Split Completion Mask (μFrame C-Mask). This field is ignored by the host controller unless the *EPS* field indicates this device is a low- or full-speed device and this queue head is in the periodic list. This field (along with the *Active* and *SplitX-state* fields) is used to determine during which micro-frames the host controller should execute a complete-split transaction. When the criteria for using this field are met, a zero value in this field has undefined behavior. This field is used by the host controller to match against the three low-order bits of the

Table continues on the next page...

Table 894. Endpoint Capabilities: Queue Head DWord 2 (continued)

	FRINDEX register. If the FRINDEX register bits decode to a position where the µ*Frame C- Mask* field is a one, then this queue head is a candidate for transaction execution. There may be more than one bit in this mask set.
7-0	Interrupt Schedule Mask (µFrame S-mask). This field is used for all endpoint speeds. Software should set this field to a zero when the queue head is on the asynchronous schedule. A non-zero value in this field indicates an interrupt endpoint. The host controller uses the value of the three low-order bits of the FRINDEX register as an index into a bit position in this bit vector. If the µ*Frame S-mask* field has a one at the indexed bit position then this queue head is a candidate for transaction execution. If the *EPS* field indicates the endpoint is a high-speed endpoint, then the transaction executed is determined by the *PID_Code* field contained in the execution area. This field is also used to support split transaction types: Interrupt (IN/OUT). This condition is true when this field is non-zero and the *EPS* field indicates this is either a full- or low-speed device. A zero value in this field, in combination with existing in the periodic frame list has undefined results.

56.6.2.6.3 Transfer overlay-queue head

The nine DWords in this area represent a transaction working space for the host controller. The general operational model is that the host controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it follows the Queue Head Horizontal Link Pointer to the next queue head. The host controller will never follow the Next Transfer Queue Element or Alternate Queue Element pointers unless it is actively attempting to advance the queue. For the duration of the transfer, the host controller keeps the incremental status of the transfer in the overlay area. When the transfer is complete, the results are written back to the original queue element.

The DWord3 of a Queue Head contains a pointer to the source qTD currently associated with the overlay. The host controller uses this pointer to write back the overlay area into the source qTD after the transfer is complete.

The following table describes the current qTD link pointer field descriptions.

Table 895. Current qTD Link Pointer

Bit	Description
31-5	Current Element Transaction Descriptor Link Pointer. This field contains the address Of the current transaction being processed in this queue and corresponds to memory address signals [31:5], respectively.
4-0	Reserved (R). These bits are ignored by the host controller when using the value as an address to write data. The actual value may vary depending on the usage.

The DWords 4-11 of a queue head are the transaction overlay area. This area has the same base structure as a Queue Element Transfer Descriptor. The queue head utilizes the reserved fields of the page pointers to implement tracking the state of split transactions.

This area is characterized as an overlay because when the queue is advanced to the next queue element, the source queue element is merged onto this area. This area serves as execution cache for the transfer.

The table below describes the Host-controller rules for bits in overlay.

Table 896. Host-Controller Rules for Bits in Overlay (DWords 5, 6, 8 and 9)

DWord	Bit	Description
5	4-1	Nak Counter (NakCnt)µRW. This field is a counter the host controller decrements whenever a transaction for the endpoint associated with this queue head results in a Nak or Nyet response. This counter is reloaded from *RL* before a transaction is executed during the first pass of the reclamation list (relative to an Asynchronous List Restart condition). It is also loaded from RL during an overlay.

Table continues on the next page...

Table 896. Host-Controller Rules for Bits in Overlay (DWords 5, 6, 8 and 9) (continued)

6	31	Data Toggle. The *Data Toggle Control* controls whether the host controller preserves this bit when an overlay operation is performed.
6	15	Interrupt On Complete (IOC). The IOC control bit is always inherited from the source qTD when the overlay operation is performed.
6	11-10	Error Counter (C_ERR). This two-bit field is copied from the qTD during the overlay and written back during queue advancement.
6	0	Ping State (P)/ERR. If the *EPS* field indicates a high-speed endpoint, then this field should be preserved during the overlay operation.
8	7-0	Split-transaction Complete-split Progress (C-prog-mask). This field is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.
9	4-0	Split-transaction Frame Tag (Frame Tag). This field is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.
9	11-5	S-bytes. Software must ensure that the *S-bytes* field in a *qTD* is zero before activating the *qTD*. This field is used to keep track of the number of bytes sent or received during an IN or OUT split transaction.

56.6.2.7 Periodic frame span traversal node (FSTN)

This data structure is to be used only for managing Full- and Low-speed transactions that span a Host-frame boundary. See Host controller operational model for FSTNs for full operational details. Software must not use an FSTN in the Asynchronous Schedule. An FSTN in the Asynchronous schedule results in undefined behavior. Software must not use the FSTN feature with a host controller whose USB_HCIVERSION register indicates a revision implementation below 0096h. FSTNs are not defined for implementations before 0.96 and their use yields undefined results.

Table 897. Frame Span Traversal Node Structure Layout

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Normal Path Link Pointer																											0		Typ		T	03-00
Back Path Link Pointer																											0		Typ[1]		T	07-04

1. Must be set to indicate a queue head

Host controller read/write Host controller read-only

56.6.2.7.1 FSTN normal path pointer

The first DWord of an FSTN contains a link pointer to the next schedule object. This object can be of any valid periodic schedule data type.

The following table describes the FSTN normal path pointer fields.

Table 898. FSTN Normal Path Pointer Field Descriptions

Bit	Description
31-5	Normal Path Link Pointer (NPLP). This field contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	QH/(s)iTD/FSTN Select (Typ). This field indicates to the Host Controller whether the item referenced is an iTD/siTD, a QH or an FSTN. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor) 11b FSTN (Frame Span Traversal Node)
0	Terminate (T). 1 = Link Pointer field is not valid. 0 = Link Pointer is valid.

56.6.2.7.2 FSTN back path link pointer

The second DWord of an FTSN node contains a link pointer to a queue head. If the T-bit in this pointer is zero, then this FSTN is a Save-Place indicator. Its Typ field must be set by software to indicate the target data structure is a queue head. If the T-bit in this pointer is set to one, then this FSTN is the Restore indicator. When the T-bit is one, the host controller ignores the Typ field.

The following table describes the FSTN back path link pointer fields.

Table 899. FSTN Back Path Link Pointer Field Descriptions

Bit	Description
31-5	Back Path Link Pointer (BPLP). This field contains the address of a Queue Head. This field corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	Typ. Software must ensure this field is set to indicate the target data structure is a Queue Head. Any other value in this field yields undefined results.
0	Terminate (T). 1=Link Pointer field is not valid (that is the host controller must not use bits [31:5] as a valid memory address). This value also indicates that this FSTN is a Restore indicator. 0=Link Pointer is valid (that is the host controller may use bits [31:5] (in combination with the CTRLDSSEGMENT register if applicable) as a valid memory address). This value also indicates that this FSTN is a Save-Place indicator.

56.6.3 Device data structures

This section defines the interface data structures used to communicate control, status, and data between DCD Software and the Device Controller. The data structure definitions in this chapter support a 32-bit memory buffer address space. The interface consists of device Queue Heads and Transfer Descriptors.

NOTE

Software must ensure that no interface data structure reachable by the Device Controller spans a 4K-page boundary.

The data structures defined in the chapter are (from the device controller's perspective) a mix of read-only and read/ writable fields. The device controller must preserve the read-only fields on all data structure writes.

The figure below shows the organization of the EndPoint Queue Head.

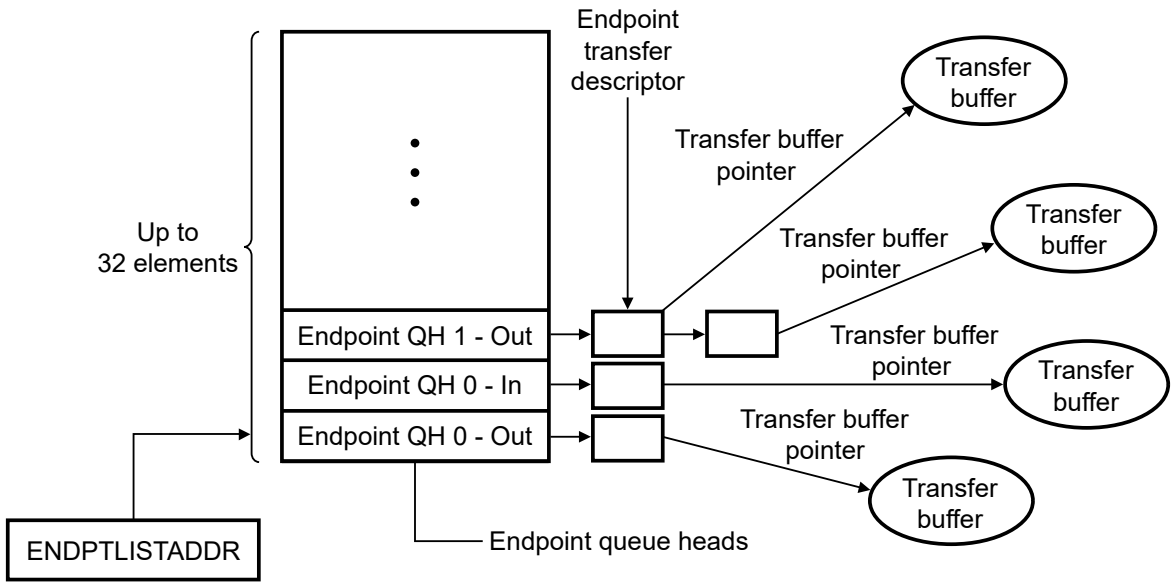


Figure 428. EndPoint Queue Head Organization

Endpoint queue heads are arranged in an array in a continuous area of memory pointed to by the USB.ENDPOINTLISTADDR pointer. The even-numbered device queue heads in the list support receive endpoints (OUT/SETUP) and the odd-numbered queue heads in the list are used for transmit endpoints (IN/INTERRUPT). That is, the device queue heads in the list with even offset support receiver endpoints (OUT) and the queue heads in the list with odd offset are used for transmit endpoint(IN). The device controller will index into this array based upon the endpoint number received from the USB bus. All information necessary to respond to transactions for all primed transfers is contained in this list so the Device Controller can readily respond to incoming requests without having to traverse a linked list.

NOTE

The Endpoint Queue Head List must be aligned to a 2k boundary.

56.6.3.1 Endpoint queue head (dQH)

The device Endpoint Queue Head (dQH) manages all transfers for a given endpoint. Data structure in the dQH has 48-byte data structure. It must have 64-byte boundaries.Priming of an endpoint includes copying the dTD (device transfer descriptor) into the overlay area of the dQH, which starts at the nextTD pointer DWord and continues through the end of the buffer pointers DWords. After a transfer is complete, there is updation of the dTD status DWord in the dTD pointed to by the currentTD pointer. While a packet is in progress, there is use of the overlay area of the dQH as a staging area for the dTD, so that the Device Controller can access needed information with little, minimal latency.

Table 900. Endpoint Queue Head (dQH)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Mult		zlt	0		Maximum Packet Length											ios	0														
Current dTD Pointer																											0				
Next dTD Pointer																											0				T [1]
0	Total Bytes															ioc	0			MultO		0		Status							
Buffer Pointer (Page 0)																				Current Offset											
Buffer Pointer (Page 1)																				Reserved											
Buffer Pointer (Page 2)																				Reserved											
Buffer Pointer (Page 3)																				Reserved											
Buffer Pointer (Page 4)[1]																				Reserved											
Reserved																															
Set-up Buffer Bytes 3…0																															
Set-up Buffer Bytes 7…4																															

1. Transfer overlay starts at T and continues through Buffer Pointer (Page 4).

Host controller read/write Host controller read-only

56.6.3.1.1 Endpoint capabilities/characteristics

This DWord specifies static information about the endpoint, in other words, this information does not change over the lifetime of the endpoint. Device Controller software should not modify this information when enabling the corresponding endpoint.

Table 901 describes the endpoint capabilities.

Table 901. Endpoint capabilities/characteristics

Bit	Description

Table continues on the next page...

Table 901. Endpoint capabilities/characteristics (continued)

31-30	Mult. Use this field to indicate the number of packets executed per transaction description as follows: 00 - Execute N Transactions as per the demonstration by the USB variable length packet protocol where you can compute N by using the maximum packet length (dQH) and the total bytes field (dTD) 01 Execute 1 transaction. 10 Execute 2 transactions. 11 Execute 3 transactions. **NOTE** Non-ISO endpoints must set Mult="00." ISO endpoints must set Mult="01," "10," or "11" as per need.
29	Zero-length termination select. Use this bit to terminate transfers in which the total transfer length is multiple. This bit is not relevant for Isochronous 0 - Enable zero-length packet to terminate transfers equal to a multiple of the maximum packet length. (default). 1 - Disable the zero-length packet on transfers that are equal in length to a multiple maximum packet length.
28-27	Reserved. These bit reserved for future use and should be set to 0.
26-16	Maximum packet length. This directly corresponds to the maximum packet size of the associated endpoint (wMaxPacketSize). The maximum value this field may contain is 0x400 (1024).
15	Interrupt on setup (IOS). Use this bit to control type endpoints to indicate whether USBINT is set in response to receiving a setup.
14-0	Reserved. Bits reserved for future use and should be set to 0.

56.6.3.1.2 Transfer overlay-endpoint queue head

The seven DWords in the overlay area represent a transaction working space for the device controller. The general operational model is that the device controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it will not read the associated endpoint.

After an endpoint is readied, the dTD will be copied into this queue head overlay area by the device controller. Until there is completion of a transfer, you must not write the queue head overlay area or the associated transfer descriptor. When the transfer is complete, the device controller will write the results back to the original transfer descriptor and advance the queue.

See Endpoint transfer descriptor (dTD) for a description of the overlay fields.

56.6.3.1.3 Current dTD pointer

The device controller uses the current dTD pointer to locate the transfer in progress. This word is for Device Controller (hardware) use only, and DCD software should not modify it.

The following table describes the dTD Pointer.

Table 902. Next dTD Pointer

Bit	Description
31-5	Current dTD. This field is a pointer to the dTD that is represented in the transfer overlay area. The device controller modifies this field to next dTD pointer during endpoint priming or queue advance.
4-0	Reserved. Bit reserved for future use and should be set to zero.

56.6.3.1.4 Set-up buffer

There is a dedicated buffer for the 8-byte data that follows the PID set-up.

NOTE

Each endpoint has a TX and an RX dQH association with it. Use only the RX queue head for receiving setup data packets.

The following table describes the multiple mode control.

Table 903. Multiple mode control (HCCPARAMS)

DWord	Bits	Description
1	31-0	Setup Buffer 0. This buffer contains bytes 3 to 0 of an incoming setup buffer packet, and the device controller writes it for you to read.
2	31-0	Setup Buffer 1. This buffer contains bytes 7 to 4 of an incoming setup buffer packet, and the device controller writes it for you to read.

56.6.3.2 Endpoint transfer descriptor (dTD)

The dTD describes to the chip controller where and how much data to send or receive for a given transfer. The DCD should not attempt to modify any field in an active dTD except the Next Like Pointer. Modify this as per the discussion in Managing transfers with transfer descriptors.

The following table shows the Endpoint Transfer Descriptor (dTD).

Table 904. Endpoint Transfer Descriptor (dTD)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Next Link Pointer																											0				T
0	Total Bytes															ioc	0			MultO		0	Status								
Buffer Pointer (Page 0)																				Current Offset											
Buffer Pointer (Page 1)																				0	Frame Number										
Buffer Pointer (Page 2)																				Reserved											
Buffer Pointer (Page 3)																				Reserved											
Buffer Pointer (Page 4)																				Reserved											

Host controller read/write Host controller read-only

The following table describes the dTD Pointer.

Table 905. Next dTD Pointer

Bit	Description
31-5	Next Transfer Element Pointer. This field contains the physical memory address of the next dTD for processing. The field corresponds to memory address signals [31:5], respectively.
4-1	Reserved. Bits reserved for future use and should be set to 0.
0	Terminate (T). 1 = Pointer is invalid. 0 = Pointer is valid (points to a valid transfer element descriptor). This bit indicates to the Device Controller that there are no more valid entries in the queue.

The following table describes the dTD Token.

Table 906. dTD Token

Bit	Description
31	Reserved. Bit reserved for future use and should be set to 0.
30-16	Total Bytes. This field specifies the total number of bytes for removal with this transfer descriptor. Decrement to this field happens by the number of bytes actually moved during the transaction and only on the successful completion of the transaction. The maximum value software may store in the field is 5*4K (5000H). This is the maximum number of bytes which five page pointers can access. Although it is possible to create a transfer up to 20K, this assumes the first offset into the first page is 0. When you cannot predetermine the offset, you can guarantee crossing past the fifth page by limiting the total bytes to 16K**. Therefore, the maximum recommended transfer is 16K (4000H). If the value of the field is 0 when the host controller fetches this transfer descriptor (and the active bit is set), the device controller executes a zero-length transaction and retires the transfer descriptor. It is not a requirement for IN transfers that Total Bytes To Transfer be an even multiple of *Maximum Packet Length*. If you build such a transfer descriptor for an IN transfer, the last transaction will always be less than *Maximum Packet Length*.
15	Interrupt On Complete (IOC). Use this bit to indicate if you need to set USBINT in response to device controller finishing with this dTD.
14-12	Reserved. Bits reserved for future use and should be set to 0.
11-10	Multiplier Override (MultO). Use this field to transmit ISOs (that is, ISO-IN) to override the multiplier in the QH. This field must be 0 for all packet types that are not transmit-ISO. Example: if QH.multiplier = 3; Maximum packet size = 8; Total Bytes = 15; MultiO = 0 [default] Three packets are sent: {Data2(8); Data1(7); Data0(0)} if QH.multiplier = 3; Maximum packet size = 8; Total Bytes = 15; MultiO = 2 Two packets are sent: {Data1(8); Data0(7)} For maximal efficiency, you should compute MultO = greatest integer of (Total Bytes / Max. Packet Size) except for the case when Total Bytes = 0; then MultO should be 1. Note: Non-ISO and non-TX endpoints must set MultO = "00."

Table continues on the next page...

Table 906. dTD Token (continued)

9-8	Reserved. Bits reserved for future use and should be set to zero.
7-0	Status. Device Controller uses this field to communicate individual command execution states back to the Device Controller software. This field contains the status of the last transaction performed on this qTD. The bit encodings are: Bit Status Field Description 7 Active. 6 Halted. 5 Data Buffer Error. 3 Transaction Error. 4, 2, 0 Reserved.

The following table describes the dTD Buffer Page Pointer List.

Table 907. dTD Buffer Page Pointer List

Bit	Description
31-12	Buffer Pointer. Selects the page offset in memory for the packet buffer. Non-virtual memory systems will typically set the buffer pointers to a series of incrementing integers.
0,11-0	Current Offset. Offset into the 4kb buffer where the packet is to begin.
1,10-0	Frame Number. Written by the device controller to indicate the frame number in which a packet finishes. Use this mostly to correlate relative completion times of packets on an ISO endpoint.

56.7 Register descriptions

There are two kinds of registers in the USB module: USB core (USBC) registers and USB noncore (USBNC) registers.

USBC registers are used to control USB core functions and are more independent of USB features. Each USB controller core has its own core registers.

USBNC registers are additional to USBC registers and are more dependent on USB features. i.MX series products vary in noncore registers.

For detailed descriptions of USBC registers, refer to Register interface.

Section USBNC memory map describes only the USBNC registers.

NOTE

- For reserved bits, preserve the value when writing (read its reset value, then write this value back).

56.7.1 USBC memory map

56.7.1.1 USBC register descriptions

56.7.1.1.1 USBC memory map

USB_OTG1 base address: 42C8_0000h

USB_OTG2 base address: 42C9_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Identification (ID)	32	R	E4A1_FA05h
4h	Hardware General (HWGENERAL)	32	R	0000_0015h
8h	Host Hardware Parameters (HWHOST)	32	R	1002_0001h
Ch	Device Hardware Parameters (HWDEVICE)	32	R	0000_0011h
10h	TX Buffer Hardware Parameters (HWTXBUF)	32	R	8008_0B08h
14h	RX Buffer Hardware Parameters (HWRXBUF)	32	R	0000_0808h
80h	General Purpose Timer #0 Load (GPTIMER0LD)	32	RW	0000_0000h
84h	General Purpose Timer #0 Controller (GPTIMER0CTRL)	32	RW	0000_0000h
88h	General Purpose Timer #1 Load (GPTIMER1LD)	32	RW	0000_0000h
8Ch	General Purpose Timer #1 Controller (GPTIMER1CTRL)	32	RW	0000_0000h
90h	System Bus Config (SBUSCFG)	32	RW	0000_0002h
100h	Capability Registers Length (CAPLENGTH)	8	R	40h
102h	Host Controller Interface Version (HCIVERSION)	16	R	0100h
104h	Host Controller Structural Parameters (HCSPARAMS)	32	R	0001_0011h
108h	Host Controller Capability Parameters (HCCPARAMS)	32	R	0000_0006h
120h	Device Controller Interface Version (DCIVERSION)	16	R	0001h
124h	Device Controller Capability Parameters (DCCPARAMS)	32	R	0000_0188h
140h	USB Command (USBCMD)	32	RW	0008_0000h
144h	USB Status (USBSTS)	32	RW	0000_0080h
148h	Interrupt Enable (USBINTR)	32	RW	0000_0000h
14Ch	USB Frame Index (FRINDEX)	32	RW	0000_0000h
154h	Device Address (DEVICEADDR)	32	RW	0000_0000h
154h	Frame List Base Address (PERIODICLISTBASE)	32	RW	0000_0000h
158h	Next Asynch. Address (ASYNCLISTADDR)	32	RW	0000_0000h
158h	Endpoint List Address (ENDPTLISTADDR)	32	RW	0000_0000h
160h	Programmable Burst Size (BURSTSIZE)	32	RW	0000_0808h
164h	TX FIFO Fill Tuning (TXFILLTUNING)	32	RW	0000_0000h
178h	Endpoint NAK (ENDPTNAK)	32	RW	0000_0000h
17Ch	Endpoint NAK Enable (ENDPTNAKEN)	32	RW	0000_0000h
180h	Configure Flag (CONFIGFLAG)	32	R	0000_0001h
184h	Port Status & Control (PORTSC1)	32	RW	1C00_0004h
1A4h	On-The-Go Status & control (OTGSC)	32	RW	0020_2F20h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1A8h	USB Device Mode (USBMODE)	32	RW	0000_5000h
1ACh	Endpoint Setup Status (ENDPTSETUPSTAT)	32	RW	0000_0000h
1B0h	Endpoint Prime (ENDPTPRIME)	32	RW	0000_0000h
1B4h	Endpoint Flush (ENDPTFLUSH)	32	RW	0000_0000h
1B8h	Endpoint Status (ENDPTSTAT)	32	R	0000_0000h
1BCh	Endpoint Complete (ENDPTCOMPLETE)	32	RW	0000_0000h
1C0h	Endpoint Control0 (ENDPTCTRL0)	32	RW	0080_0080h
1C4h	Endpoint Control 1 (ENDPTCTRL1)	32	RW	0000_0000h
1C8h	Endpoint Control 2 (ENDPTCTRL2)	32	RW	0000_0000h
1CCh	Endpoint Control 3 (ENDPTCTRL3)	32	RW	0000_0000h
1D0h	Endpoint Control 4 (ENDPTCTRL4)	32	RW	0000_0000h
1D4h	Endpoint Control 5 (ENDPTCTRL5)	32	RW	0000_0000h
1D8h	Endpoint Control 6 (ENDPTCTRL6)	32	RW	0000_0000h
1DCh	Endpoint Control 7 (ENDPTCTRL7)	32	RW	0000_0000h

56.7.1.1.2 Identification (ID)

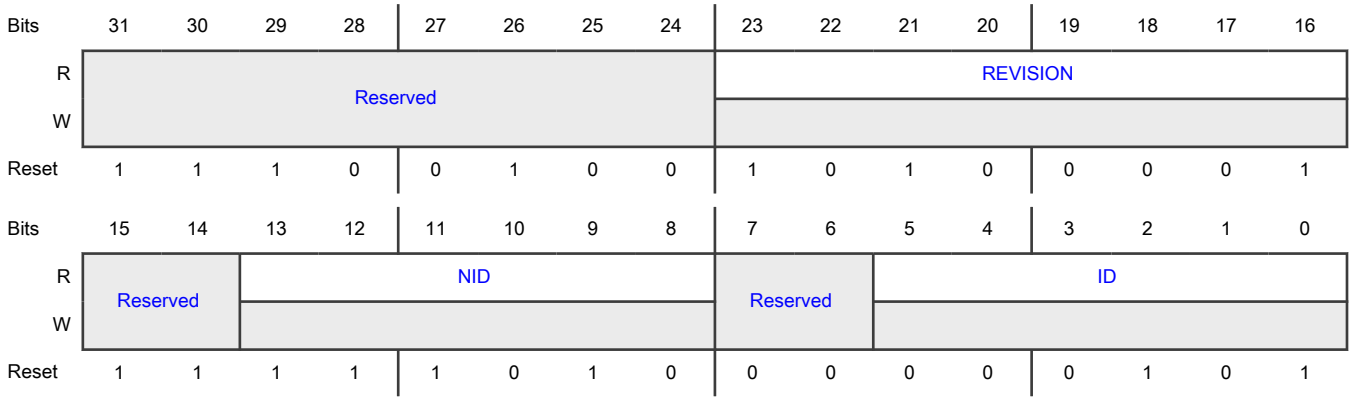
Offset

Register	Offset
ID	0h

Function

Identifies the USB 2.0 High-Speed core and its revision.

Diagram



Fields

Field	Function
31-24 —	- Reserved
23-16 REVISION	REVISION Revision number of the controller core.
15-14 —	- Reserved
13-8 NID	NID Complement version of ID
7-6 —	- Reserved
5-0 ID	ID Configuration number. This number is set to 0x05 and indicates that the peripheral is USB 2.0 High-Speed core.

56.7.1.1.3 Hardware General (HWGENERAL)

Offset

Register	Offset
HWGENERAL	4h

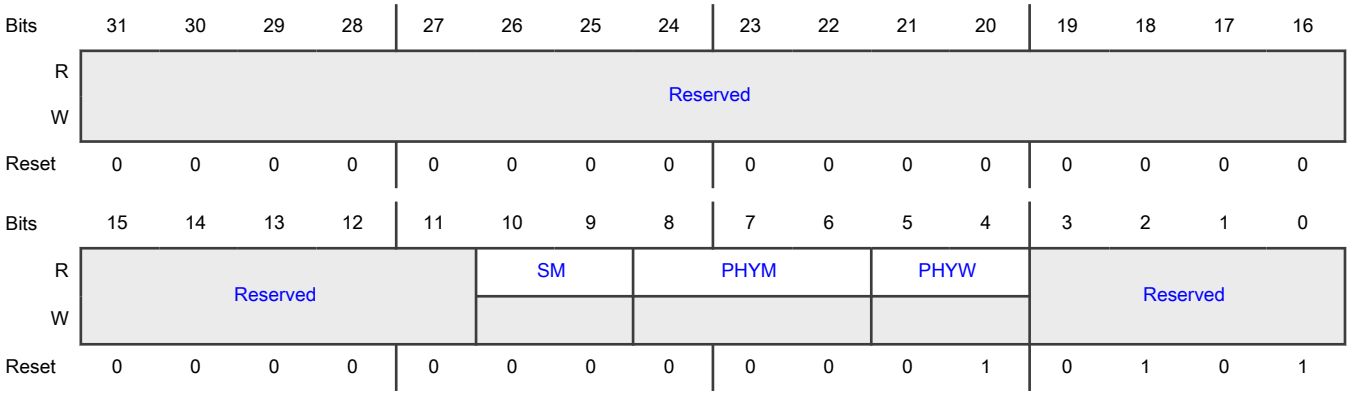
Function

General hardware parameters as defined in System Level Issues and Core Configuration.

NOTE

The reset value could vary from instance to instance. See the detail in field description and ignore reset value in summary table in this case!

Diagram



Fields

Field	Function
31-11 —	- Reserved
10-9 SM	SM Serial interface mode capability 00b - No Serial Engine, always use parallel signalling. 01b - Serial Engine present, always use serial signalling for FS/LS. 10b - Software programmable - Reset to use parallel signalling for FS/LS 11b - Software programmable - Reset to use serial signalling for FS/LS
8-6 PHYM	PHYM Transceiver type 000b - UTMI/UMTI+ 001b - ULPI DDR 010b - ULPI 011b - Serial Only 100b - Software programmable - reset to UTMI/UTMI+ 101b - Software programmable - reset to ULPI DDR 110b - Software programmable - reset to ULPI 111b - Software programmable - reset to Serial
5-4	PHYW

Table continues on the next page...

Table continued from the previous page...

Field	Function
PHYW	Data width of the transceiver connected to the controller core. 00b - 8 bit wide data bus (Software non-programmable) 01b - 16 bit wide data bus (Software non-programmable) 10b - Reset to 8 bit wide data bus (Software programmable) 11b - Reset to 16 bit wide data bus (Software programmable)
3-0 —	- Reserved

56.7.1.1.4 Host Hardware Parameters (HWHOST)

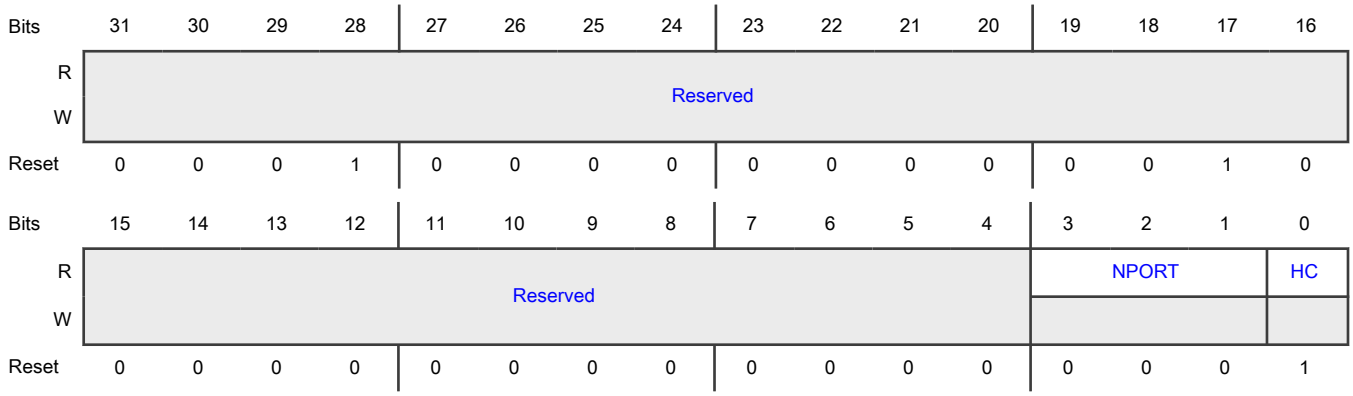
Offset

Register	Offset
HWHOST	8h

Function

Host Hardware Parameters defined in configuration file.

Diagram



Fields

Field	Function
31-4 —	- Reserved
3-1	NPORT

Table continues on the next page...

Table continued from the previous page...

Field	Function
NPORT	The Number of downstream ports supported by the host controller is (NPORT+1). **NOTE** When these bits value is '000', it indicates a single-port host controller.
0 HC	HC Host Capable. Indicating whether host operation mode is supported or not. 0b - Not supported 1b - Supported

56.7.1.1.5 Device Hardware Parameters (HWDEVICE)

Offset

Register	Offset
HWDEVICE	Ch

Function

NOTE
This register is only available in OTG core.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										DEVEP					DC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

Fields

Field	Function
31-6	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
5-1 DEVEP	DEVEP Device Endpoint Number
0 DC	DC Device Capable. Indicating whether device operation mode is supported or not. 0b - Not supported 1b - Supported

56.7.1.1.6 TX Buffer Hardware Parameters (HWTXBUF)

Offset

Register	Offset
HWTXBUF	10h

Function

TX Buffer Hardware Parameters defined in configuration file.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								TXCHANADD							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								TXBURST							
W																
Reset	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0

Fields

Field	Function
31-24 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 TXCHANADD	TXCHANADD TX FIFO Buffer size is: (2^TXCHANADD) * 4 Bytes. These bits are set to '08h', so buffer size is 256*4 Bytes. For the OTG controller operating in device mode, this is the FIFO buffer size per endpoint. As the OTG controller has 8 TX endpoint, there are 8 of these buffers. For the OTG controller operating in Host mode, or for Host-only controller, the entire buffer memory is used as a single TX buffer. Therefore, there is only 1 of this buffer
15-8 —	- Reserved
7-0 TXBURST	TXBURST Default burst size for memory to TX buffer transfer. This is reset value of TXPBURST fields in BURSTSIZE.

56.7.1.1.7 RX Buffer Hardware Parameters (HWRXBUF)

Offset

Register	Offset
HWRXBUF	14h

Function

RX Buffer Hardware Parameters defined in configuration file.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXADD								RXBURST							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-16 —	- Reserved
15-8 RXADD	RXADD Buffer total size for all receive endpoints is (2^RXADD). RX Buffer size is: (2^RXADD) * 4 Bytes. These bits are set to '08h', so buffer size is 256*4 Bytes. There is a single Receive FIFO buffer in the USB controller. The buffer is shared for all endpoints for the OTG controller in device mode.
7-0 RXBURST	RXBURST Default burst size for memory to RX buffer transfer. This is reset value of RXPBURST bits in USB core register USB_n_BURSTSIZE. Please see BURSTSIZE .

56.7.1.1.8 General Purpose Timer #0 Load (GPTIMER0LD)

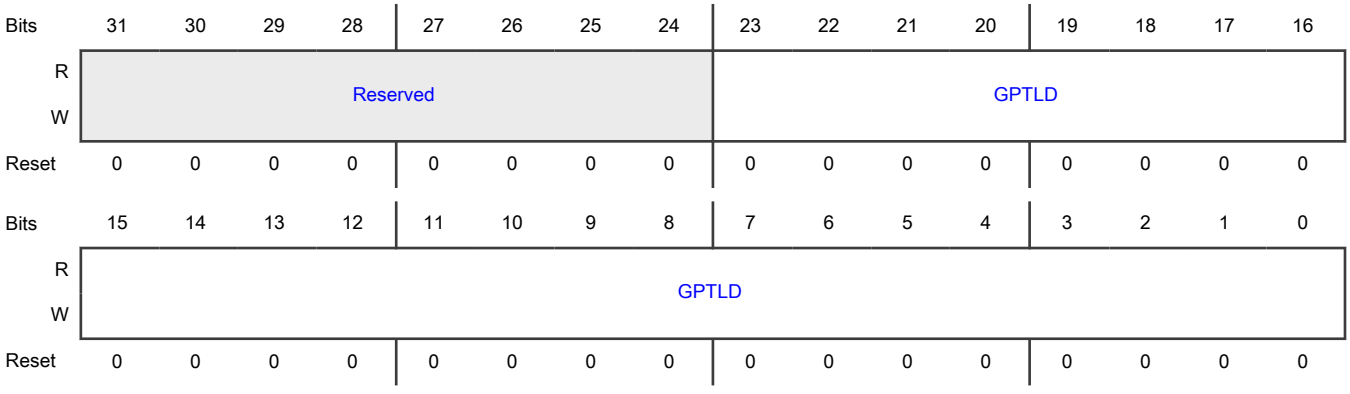
Offset

Register	Offset
GPTIMER0LD	80h

Function

This register controls load value of the count timer in register n_GPTIMER0CTRL. Please see GPTIMER0CTRL .

Diagram



Fields

Field	Function
31-24 —	- Reserved
23-0 GPTLD	GPTLD General Purpose Timer Load Value These bit fields are loaded to GPTCNT bits when GPTRST bit is set '1b'. This value represents the time in microseconds minus 1 for the timer duration. Example: for a one millisecond timer, load 1000-1=999 or 0x0003E7. **NOTE** Max value is 0xFFFFFF or 16.777215 seconds.

56.7.1.1.9 General Purpose Timer #0 Controller (GPTIMER0CTRL)

Offset

Register	Offset
GPTIMER0CTRL	84h

Function

This register contains the control for this countdown timer and a data field can be queried to determine the running count value. This timer has granularity on 1 us and can be programmed to a little over 16 seconds. There are two counter modes which are described in the register table below. When the timer counter value transitions to zero, an interrupt could be generated if enable.

Interrupt status bit is TI0 bit in n_USBSTS register (See USBSTS), interrupt enable bit is TIE0 bit in n_USBINTR register. (See USBINTR .)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GPTRUN	GPTRST	Reserved					GPTMODE	GPTCNT							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPTCNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 GPTRUN	GPTRUN General Purpose Timer Run GPTCNT bits are not effected when setting or clearing this bit. 0b - Stop counting 1b - Run
30 GPTRST	GPTRST General Purpose Timer Reset 0b - No action 1b - Load counter value from GPTLD bits in n_GPTIMER0LD
29-25 —	- Reserved
24 GPTMODE	GPTMODE General Purpose Timer Mode In one shot mode, the timer will count down to zero, generate an interrupt, and stop until the counter is reset by software; In repeat mode, the timer will count down to zero, generate an interrupt and automatically reload the counter value from GPTLD bits to start again. 0b - One Shot Mode 1b - Repeat Mode
23-0 GPTCNT	GPTCNT General Purpose Timer Counter. This field is the count value of the countdown timer.

56.7.1.1.10 General Purpose Timer #1 Load (GPTIMER1LD)

Offset

Register	Offset
GPTIMER1LD	88h

Function

This register controls load value of the count timer in register n_GPTIMER1CTRL. Please see GPTIMER1CTRL .

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								GPTLD							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPTLD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	- Reserved
23-0 GPTLD	GPTLD General Purpose Timer Load Value These bit fields are loaded to GPTCNT bits when GPTRST bit is set '1b'. This value represents the time in microseconds minus 1 for the timer duration. Example: for a one millisecond timer, load 1000-1=999 or 0x0003E7. **NOTE** Max value is 0xFFFFFF or 16.777215 seconds.

56.7.1.1.11 General Purpose Timer #1 Controller (GPTIMER1CTRL)

Offset

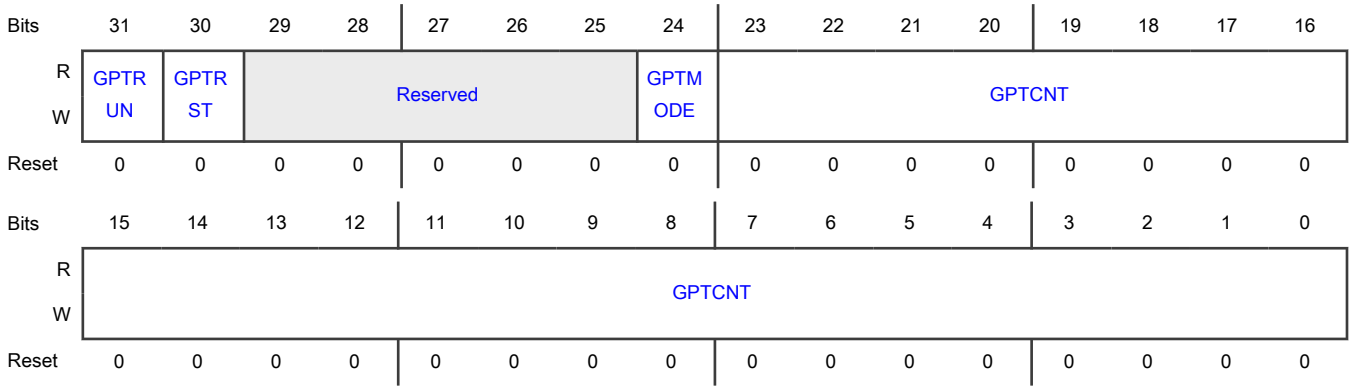
Register	Offset
GPTIMER1CTRL	8Ch

Function

This register contains the control for this countdown timer and a data field can be queried to determine the running count value. This timer has granularity on 1 us and can be programmed to a little over 16 seconds. There are two counter modes which are described in the register table below. When the timer counter value transitions to zero, an interrupt could be generated if enable.

Interrupt status bit is TI1 bit in USB_n_USBSTS register (See USBSTS), interrupt enable bit is TIE1 bit in n_USBINTR register (See USBINTR).

Diagram



Fields

Field	Function
31 GPTRUN	GPTRUN General Purpose Timer Run GPTCNT bits are not effected when setting or clearing this bit. 0b - Stop counting 1b - Run
30 GPTRST	GPTRST General Purpose Timer Reset 0b - No action 1b - Load counter value from GPTLD bits in USB_n_GPTIMER0LD
29-25 —	- Reserved
24 GPTMODE	GPTMODE General Purpose Timer Mode In one shot mode, the timer will count down to zero, generate an interrupt, and stop until the counter is reset by software. In repeat mode, the timer will count down to zero, generate an interrupt and automatically reload the counter value from GPTLD bits to start again. 0b - One Shot Mode 1b - Repeat Mode
23-0 GPTCNT	GPTCNT General Purpose Timer Counter. This field is the count value of the countdown timer.

56.7.1.1.12 System Bus Config (SBUSCFG)

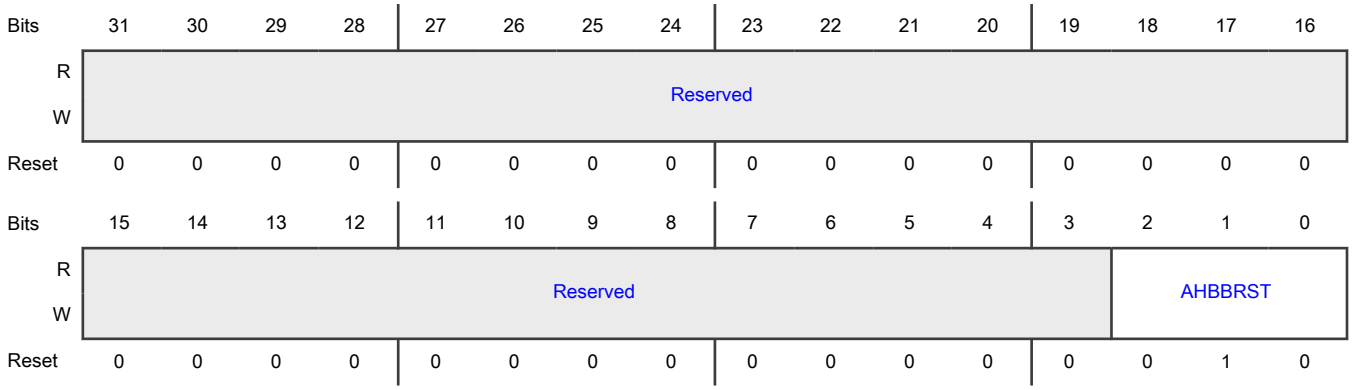
Offset

Register	Offset
SBUSCFG	90h

Function

This register defines the system bus configuration.

Diagram



Fields

Field	Function
31-3 —	- Reserved
2-0 AHBBRST	AHBBRST AHB controller interface burst configuration These bits control AHB controller transfer type sequence (or priority). **NOTE** This register overrides n_BURSTSIZE register when its value is not zero. 000b - Incremental burst of unspecified length only 001b - INCR4 burst, then single transfer 010b - INCR8 burst, INCR4 burst, then single transfer 011b - INCR16 burst, INCR8 burst, INCR4 burst, then single transfer 100b - Reserved, don't use 101b - INCR4 burst, then incremental burst of unspecified length 110b - INCR8 burst, INCR4 burst, then incremental burst of unspecified length 111b - INCR16 burst, INCR8 burst, INCR4 burst, then incremental burst of unspecified length

56.7.1.1.13 Capability Registers Length (CAPLENGTH)

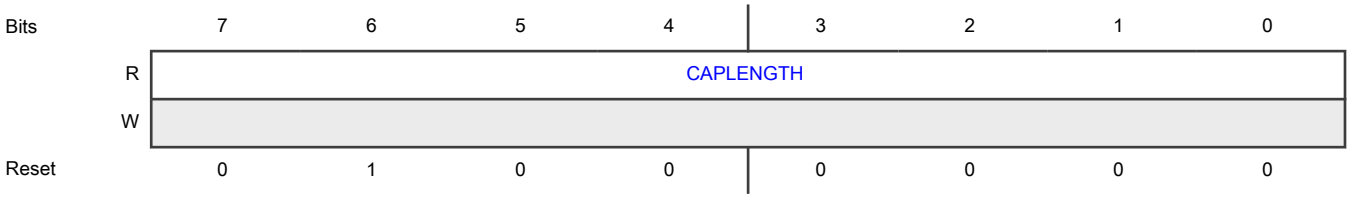
Offset

Register	Offset
CAPLENGTH	100h

Function

The Capability Registers Length register contains the address offset to the Operational registers relative to the CAPLENGTH register.

Diagram



Fields

Field	Function
7-0 CAPLENGTH	CAPLENGTH These bits are used as an offset to add to register base to find the beginning of the Operational Register. Default value is '40h'.

56.7.1.1.14 Host Controller Interface Version (HCIVERSION)

Offset

Register	Offset
HCIVERSION	102h

Function

This is a 2-byte register containing a BCD encoding of the EHCI revision number supported by this host controller. The most significant byte of this register represents a major revision and the least significant byte is the minor revision.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HCIVERSION															
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 HCIVERSION	HCIVERSION Host Controller Interface Version Number Default value is '10h', which means EHCI rev1.0.

56.7.1.1.15 Host Controller Structural Parameters (HCSPARAMS)

Offset

Register	Offset
HCSPARAMS	104h

Function

The following figure shows the port steering logic capabilities of Host Control Structural Parameters (n_HCSPARAMS).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				N_TT				N_PTT				Reserved			PI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	N_CC				N_PCC				Reserved			PPC	N_PORTS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

Fields

Field	Function
31-28	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
27-24 N_TT	N_TT Number of Transaction Translators (N_TT). Default value '0000b' This field indicates the number of embedded transaction translators associated with the USB2.0 host controller. These bits would be set to '0001b' for Multi-Port Host, and '0000b' for Single-Port Host.
23-20 N_PTT	N_PTT Number of Ports per Transaction Translator (N_PTT). Default value '0000b' This field indicates the number of ports assigned to each transaction translator within the USB2.0 host controller. These bits would be set to equal N_PORTS for Multi-Port Host, and '0000b' for Single-Port Host.
19-17 —	- Reserved
16 PI	PI Port Indicators (P INDICATOR) This bit indicates whether the ports support port indicator control. When set to one, the port status and control registers include a read/writeable field for controlling the state of the port indicator This bit is "1b" in all controller core.
15-12 N_CC	N_CC Number of Companion Controller (N_CC). This field indicates the number of companion controllers associated with this USB2.0 host controller. These bits are '0000b' in all controller core. 0000b - There is no internal Companion Controller and port-ownership hand-off is not supported. 0001b - There are internal companion controller(s) and port-ownership hand-offs is supported.
11-8 N_PCC	N_PCC Number of Ports per Companion Controller This field indicates the number of ports supported per internal Companion Controller. It is used to indicate the port routing configuration to the system software. For example, if N_PORTS has a value of 6 and N_CC has a value of 2 then N_PCC could have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, etc. In the previous example, the N_PCC could have been 4, where the first 4 are routed to companion controller 1 and the last two are routed to companion controller 2. The number in this field must be consistent with N_PORTS and N_CC. These bits are '0000b' in all controller core.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-5 —	- Reserved
4 PPC	PPC Port Power Control This field indicates whether the host controller implementation includes port power control. A one indicates the ports have port power switches. A zero indicates the ports do not have port power switches. The value of this field affects the functionality of the Port Power field in each port status and control register
3-0 N_PORTS	N_PORTS Number of downstream ports. This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register. Valid values are in the range of 1h to Fh. A zero in this field is undefined. These bits are always set to '0001b' because all controller cores are Single-Port Host.

56.7.1.1.16 Host Controller Capability Parameters (HCCPARAMS)

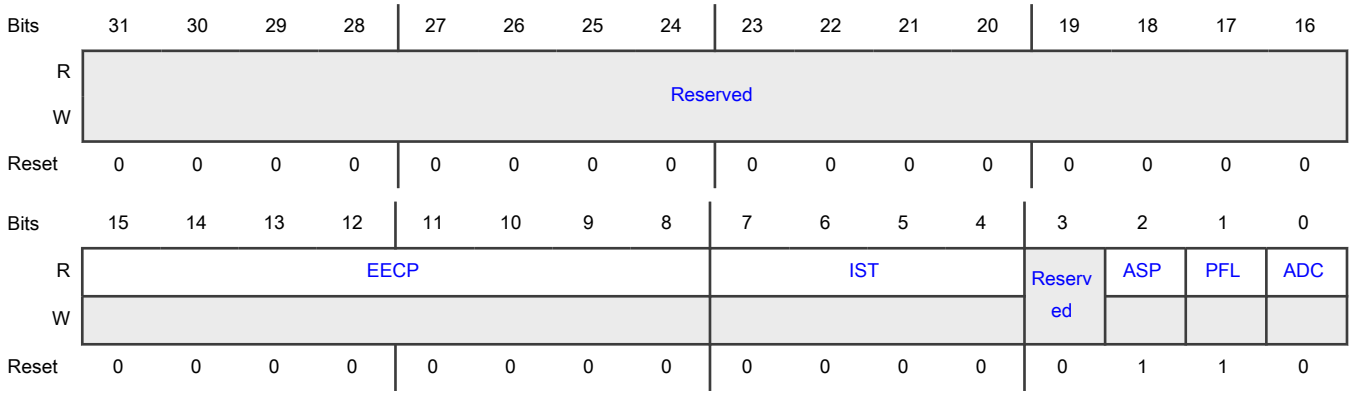
Offset

Register	Offset
HCCPARAMS	108h

Function

This register identifies multiple mode control (time-base bit functionality), addressing capability.

Diagram



Fields

Field	Function
31-16 —	- Reserved
15-8 EECP	EECP EHCI Extended Capabilities Pointer. This field indicates the existence of a capabilities list. A value of 00h indicates no extended capabilities are implemented. A non-zero value in this register indicates the offset in PCI configuration space of the first EHCI extended capability. The pointer value must be 40h or greater if implemented to maintain the consistency of the PCI header defined for this class of device. **NOTE** These bits are set '00h' in all controller core.
7-4 IST	IST Isochronous Scheduling Threshold. This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule. When bit [7] is zero, the value of the least significant 3 bits indicates the number of micro-frames a host controller can hold a set of isochronous data structures (one or more) before flushing the state. When bit [7] is a one, then host software assumes the host controller may cache an isochronous data structure for an entire frame. These bits are set '00h' in all controller core.
3 —	- Reserved
2 ASP	ASP Asynchronous Schedule Park Capability If this bit is set to a one, then the host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule. The feature can be disabled or enabled and set to a specific level by using the *Asynchronous Schedule Park Mode Enable* and *Asynchronous Schedule Park Mode Count* fields in the USBCMD register. **NOTE** ASP bit reset value: '00b' for OTG controller core, '11b' for Host-only controller core.
1 PFL	PFL Programmable Frame List Flag If this bit is set to zero, then the system software must use a frame list length of 1024 elements with this host controller. The USBCMD register Frame List Size field is a read-only register and must be set to zero. If set to a one, then the system software can specify and use a smaller frame list and configure the host controller via the USBCMD register Frame List Size field. The frame list must always be aligned on a 4K-page boundary. This requirement ensures that the frame list is always physically contiguous. This bit is set '1b' in all controller core.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 ADC	ADC 64-bit Addressing Capability This bit is set '0b' in all controller core, no 64-bit addressing capability is supported.

56.7.1.1.17 Device Controller Interface Version (DCIVERSION)

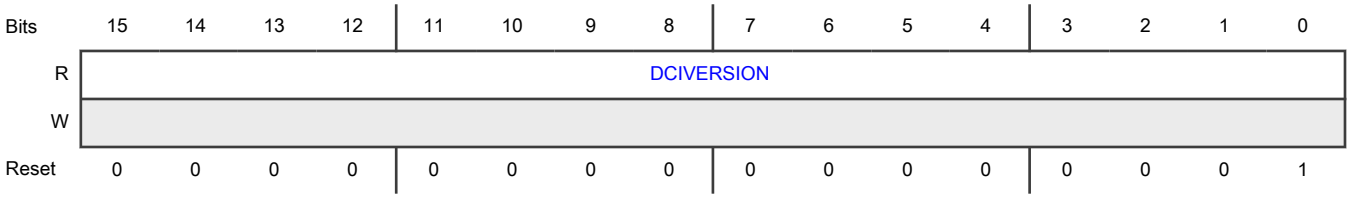
Offset

Register	Offset
DCIVERSION	120h

Function

This register indicates the two-byte BCD encoding of the device controller interface version number.

Diagram



Fields

Field	Function
15-0 DCIVERSION	DCIVERSION Device Controller Interface Version Number Default value is '01h', which means rev0.1.

56.7.1.1.18 Device Controller Capability Parameters (DCCPARAMS)

Offset

Register	Offset
DCCPARAMS	124h

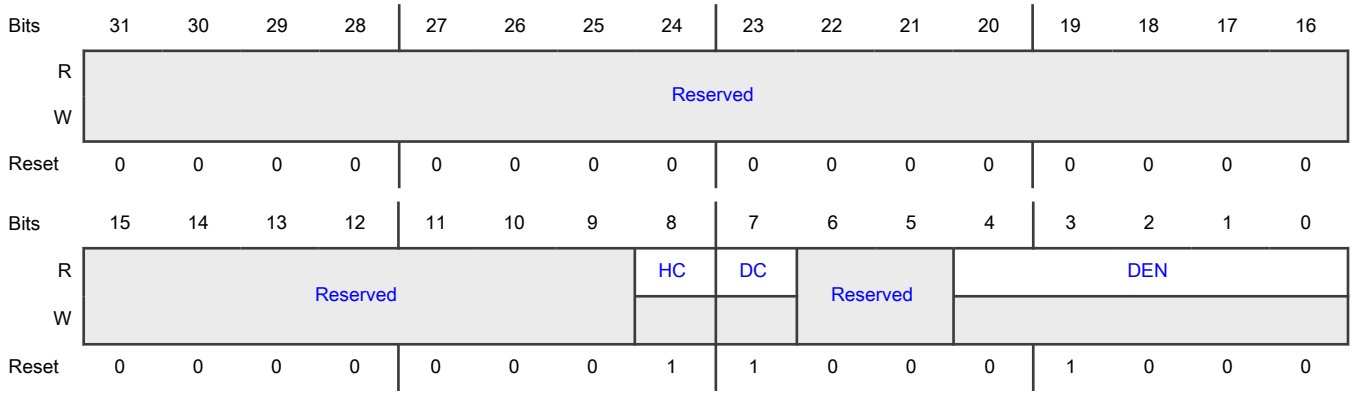
Function

These fields describe the overall device capability of the controller.

NOTE

This register is only available in OTG controller core.

Diagram



Fields

Field	Function
31-9 —	- Reserved
8 HC	HC Host Capable When this bit is 1, this controller is capable of operating as an EHCI compatible USB 2.0 host controller.
7 DC	DC Device Capable When this bit is 1, this controller is capable of operating as a USB 2.0 device.
6-5 —	- Reserved
4-0 DEN	DEN Device Endpoint Number This field indicates the number of endpoints built into the device controller. If this controller is not device capable, then this field will be zero. Valid values are 0 - 15.

56.7.1.1.19 USB Command (USBCMD)

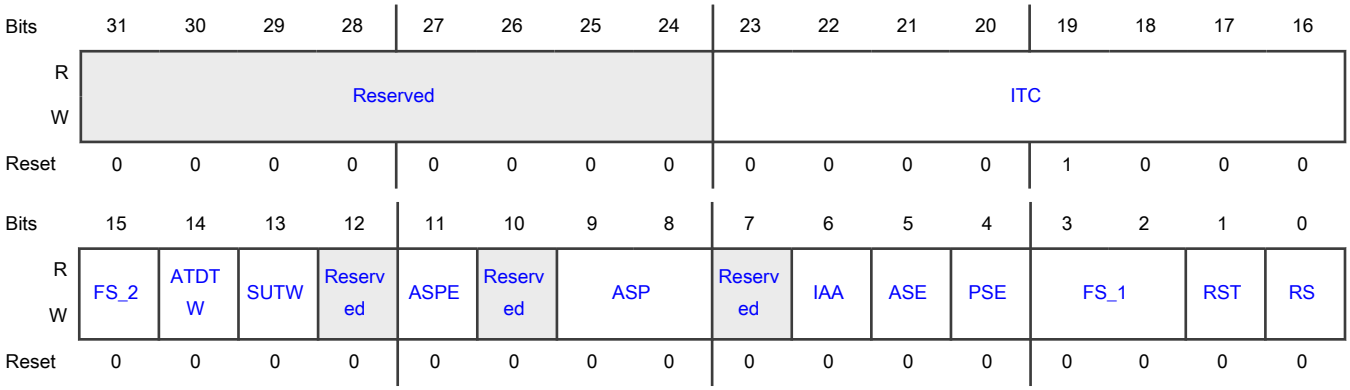
Offset

Register	Offset
USBCMD	140h

Function

Indicates the serial bus host/device controller with the command to execute. Writing to the register causes a command to be executed.

Diagram



Fields

Field	Function
31-24 —	- Reserved
23-16 ITC	ITC Interrupt Threshold Control -Read/Write. The system software uses this field to set the maximum rate at which the host/device controller will issue interrupts. ITC contains the maximum interrupt interval measured in micro-frames. Valid values are shown below. Value Maximum Interrupt Interval 0000_0000b - Immediate (no threshold) 0000_0001b - 1 micro-frame 0000_0010b - 2 micro-frames 0000_0100b - 4 micro-frames 0000_1000b - 8 micro-frames 0001_0000b - 16 micro-frames 0010_0000b - 32 micro-frames 0100_0000b - 64 micro-frames
15 FS_2	FS_2 Frame List Size - (Read/Write or Read Only). [host mode only] This field is Read/Write only if Programmable Frame List Flag in the HCCPARAMS registers is set to one. This field specifies the size of the frame list that controls which bits in the Frame Index Register should be used for the Frame List Current index.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field is made up from USBCMD bits 15, 3 and 2. Value Meaning 0b000 - 1024 elements (4096 bytes) Default value 0b001 - 512 elements (2048 bytes) 0b010 - 256 elements (1024 bytes) 0b011 - 128 elements (512 bytes) 0b100 - 64 elements (256 bytes) 0b101 - 32 elements (128 bytes) 0b110 - 16 elements (64 bytes) 0b111 - 8 elements (32 bytes)
14 ATDTW	ATDTW Add dTD TripWire - Read/Write. [device mode only] This bit is used as a semaphore to ensure proper addition of a new dTD to an active (primed) endpoint's linked list. This bit is set and cleared by software. This bit would also be cleared by hardware when state machine is hazard region for which adding a dTD to a primed endpoint may go unrecognized.
13 SUTW	SUTW Setup TripWire - Read/Write. [device mode only] This bit is used as a semaphore to ensure that the setup data payload of 8 bytes is extracted from a QH by the DCD without being corrupted. If the setup lockout mode is off (SLOM bit in USB core register n_USBMODE, see USBMODE) then there is a hazard when new setup data arrives while the DCD is copying the setup data payload from the QH for a previous setup packet. This bit is set and cleared by software. This bit would also be cleared by hardware when a hazard detected.
12 —	- Reserved
11 ASPE	ASPE Asynchronous Schedule Park Mode Enable - Read/Write. If the *Asynchronous Park Capability* bit in the HCCPARAMS register is a one, then this bit defaults to a 1h and is R/W. Otherwise the bit must be a zero and is RO. Software uses this bit to enable or disable Park mode. When this bit is one, Park mode is enabled. When this bit is a zero, Park mode is disabled. **NOTE** ASPE bit reset value: '0b' for OTG controller .

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 —	- Reserved
9-8 ASP	ASP Asynchronous Schedule Park Mode Count - Read/Write. If the *Asynchronous Park Capability* bit in the HCCPARAMS register is a one, then this field defaults to 3h and is R/W. Otherwise it defaults to zero and is Read-Only. It contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the Asynchronous schedule before continuing traversal of the Asynchronous schedule. Valid values are 1h to 3h. Software must not write a zero to this bit when *Park Mode Enable* is a one as this will result in undefined behavior. This field is set to 3h in all controller core.
7 —	- Reserved
6 IAA	IAA Interrupt on Async Advance Doorbell - Read/Write. This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to ring the doorbell. When the host controller has evicted all appropriate cached schedule states, it sets the Interrupt on Async Advance status bit in the USBSTS register. If the Interrupt on Sync Advance Enable bit in the USBINTR register is one, then the host controller will assert an interrupt at the next interrupt threshold. The host controller sets this bit to zero after it has set the Interrupt on Sync Advance status bit in the USBSTS register to one. Software should not write a one to this bit when the asynchronous schedule is inactive. Doing so will yield undefined results. This bit is only used in Host mode. Writing a one to this bit when device mode is selected will have undefined results.
5 ASE	ASE Asynchronous Schedule Enable - Read/Write. Default 0b. This bit controls whether the host controller skips processing the Asynchronous Schedule. Only the host controller uses this bit. Values Meaning 0b - Do not process the Asynchronous Schedule. 1b - Use the ASYNCLISTADDR register to access the Asynchronous Schedule.
4 PSE	PSE Periodic Schedule Enable- Read/Write. Default 0b. This bit controls whether the host controller skips processing the Periodic Schedule. Only the host controller uses this bit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Values Meaning 0b - Do not process the Periodic Schedule 1b - Use the PERIODICLISTBASE register to access the Periodic Schedule.
3-2 FS_1	FS_1 See description at bit 15
1 RST	RST Controller Reset (RESET) - Read/Write. Software uses this bit to reset the controller. This bit is set to zero by the Host/Device Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register. Host operation mode: When software writes a one to this bit, the Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports. Software should not set this bit to a one when the HCHalted bit in the USBSTS register is a zero. Attempting to reset an actively running host controller will result in undefined behavior. Device operation mode: When software writes a one to this bit, the Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Writing a one to this bit when the device is in the attached state is not recommended, because the effect on an attached host is undefined. In order to ensure that the device is not in an attached state before initiating a device controller reset, all primed endpoints should be flushed and the USBCMD Run/Stop bit should be set to 0.
0 RS	RS Run/Stop (RS) - Read/Write. Default 0b. 1=Run. 0=Stop. Host operation mode: When set to '1b', the Controller proceeds with the execution of the schedule. The Controller continues execution as long as this bit is set to a one. When this bit is set to 0, the Host Controller completes the current transaction on the USB and then halts. The HC Halted bit in the status register indicates when the Controller has finished the transaction and has entered the stopped state. Software should not write a one to this field unless the controller is in the Halted state (that is, HCHalted in the USBSTS register is a one). Device operation mode: Writing a one to this bit will cause the controller to enable a pull-up on D+ and initiate an attach event. This control bit is not directly connected to the pull-up enable, as the pull-up will become disabled upon transitioning into high-speed mode. Software should use this bit to prevent an attach event before the controller has been properly initialized. Writing a 0 to this will cause a detach event.

56.7.1.1.20 USB Status (USBSTS)

Offset

Register	Offset
USBSTS	144h

Function

Indicates various states of the host/device controller and any pending interrupts. This register does not indicate status resulting from a transaction on the serial bus.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved						TI1	TI0	Reserved							NAKI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AS	PS	RCL	HCH	Reserved	ULPII	Reserved	SLI	SRI	URI	AAI	SEI	FRI	PCI	UEI	UI
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	- Reserved
25 TI1	TI1 General Purpose Timer Interrupt 1(GPTINT1)--R/WC. This bit is set when the counter in the GPTIMER1CTRL register transitions to zero, writing a one to this bit will clear it.
24 TI0	TI0 General Purpose Timer Interrupt 0(GPTINT0)--R/WC. This bit is set when the counter in the GPTIMER0CTRL register transitions to zero, writing a one to this bit clears it.
23-17 —	- Reserved
16	NAKI

Table continues on the next page...

Table continued from the previous page...

Field	Function
NAKI	NAK Interrupt Bit--RO. This bit is set by hardware when for a particular endpoint both the TX/RX Endpoint NAK bit and corresponding TX/RX Endpoint NAK Enable bit are set. This bit is automatically cleared by hardware when all Enabled TX/RX Endpoint NAK bits are cleared.
15 AS	AS Asynchronous Schedule Status - Read Only. This bit reports the current real status of the Asynchronous Schedule. When set to zero the asynchronous schedule status is disabled and if set to one the status is enabled. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled (1) or disabled (0). Only used in the host operation mode.
14 PS	PS Periodic Schedule Status - Read Only. This bit reports the current real status of the Periodic Schedule. When set to zero the periodic schedule is disabled, and if set to one the status is enabled. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled (1) or disabled (0). Only used in the host operation mode.
13 RCL	RCL Reclamation - Read Only. This is a read-only status bit used to detect an empty asynchronous schedule. Only used in the host operation mode.
12 HCH	HCH HCHalted - Read Only. This bit is a zero whenever the Run/Stop bit is a one. The Controller sets this bit to one after it has stopped executing because of the Run/Stop bit being set to 0, either by software or by the Controller hardware (for example, an internal error). Only used in the host operation mode. Default value is '0b' for OTG core . This is because OTG core is not operating as host in default. Please see CM bit in USB_n_USBMODE register. **NOTE** HCH bit reset value: '0b' for OTG controller core .
11	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
10 ULPII	ULPII ULPI Interrupt - R/WC. This bit will be set '1b' by hardware when there is an event completion in ULPI viewport. This bit is usable only if the controller support UPLI interface mode.
9 —	- Reserved
8 SLI	SLI DCSuspend - R/WC. When a controller enters a suspend state from an active state, this bit will be set to a one. The device controller clears the bit upon exiting from a suspend state. Only used in device operation mode.
7 SRI	SRI SOF Received - R/WC. When the device controller detects a Start Of (micro) Frame, this bit will be set to a one. When a SOF is extremely late, the device controller will automatically set this bit to indicate that an SOF was expected. Therefore, this bit will be set roughly every 1ms in device FS mode and every 125ms in HS mode and will be synchronized to the actual SOF that is received. Because the device controller is initialized to FS before connect, this bit will be set at an interval of 1ms during the prelude to connect and chirp. In Host mode, this bit will be set every 125us and can be used by host controller driver as a time base. Software writes a 1 to this bit to clear it.
6 URI	URI USB Reset Received - R/WC. When the device controller detects a USB Reset and enters the default state, this bit will be set to a one. Software can write a 1 to this bit to clear the USB Reset Received status bit. Only used in device operation mode.
5 AAI	AAI Interrupt on Async Advance - R/WC. System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a one to the Interrupt on Async Advance Doorbell bit in the n_USBCMD register. This status bit indicates the assertion of that interrupt source. Only used in host operation mode.
4	SEI

Table continues on the next page...

Table continued from the previous page...

Field	Function
SEI	System Error- R/WC. This bit is will be set to '1b' when an Error response is seen to a read on the system interface.
3 FRI	FRI Frame List Rollover - R/WC. The Host Controller sets this bit to a one when the Frame List Index rolls over from its maximum value to zero. The exact value at which the rollover occurs depends on the frame list size. For example. If the frame list size (as programmed in the Frame List Size field of the USB_n_USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX [13] toggles. Similarly, if the size is 512, the Host Controller sets this bit to a one every time FHINDEX [12] toggles. Only used in host operation mode.
2 PCI	PCI Port Change Detect - R/WC. The Host Controller sets this bit to a one when on any port a Connect Status occurs, a Port Enable/Disable Change occurs, or the Force Port Resume bit is set as the result of a J-K transition on the suspended port. The Device Controller sets this bit to a one when the port controller enters the full or high-speed operational state. When the port controller exits the full or high-speed operation states due to Reset or Suspend events, the notification mechanisms are the USB Reset Received bit and the DCSuspend bits respectively.
1 UEI	UEI USB Error Interrupt (USBERRINT) - R/WC. When completion of a USB transaction results in an error condition, this bit is set by the Host/Device Controller. This bit is set along with the USBINT bit, if the TD on which the error interrupt occurred also had its interrupt on complete (IOC) bit set. The device controller detects resume signaling only.
0 UI	UI USB Interrupt (USBINT) - R/WC. This bit is set by the Host/Device Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set. This bit is also set by the Host/Device Controller when a short packet is detected. A short packet is when the actual number of bytes received was less than the expected number of bytes.

56.7.1.1.21 Interrupt Enable (USBINTR)

Offset

Register	Offset
USBINTR	148h

Function

Enables interrupts to software. An interrupt is generated when a bit is set and the corresponding interrupt source is active. The USB Status register (n_USBSTS) still shows interrupt sources even if they are disabled by the n_USBINTR register, allowing polling of interrupt events by the software.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved						TIE1	TIE0	Reserved				UPIE	UAIE	Reserved	NAKE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved					ULPIE	Reserved	SLE	SRE	URE	AAE	SEE	FRE	PCE	UEE	UE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	- Reserved
25 TIE1	TIE1 General Purpose Timer #1 Interrupt Enable When this bit is one and the TI1 bit in n_USBSTS register is a one the controller will issue an interrupt.
24 TIE0	TIE0 General Purpose Timer #0 Interrupt Enable When this bit is one and the TI0 bit in n_USBSTS register is a one the controller will issue an interrupt.
23-20 —	- Reserved
19 UPIE	UPIE USB Host Periodic Interrupt Enable When this bit is one, and the UPI bit in the n_USBSTS register is one, host controller will issue an interrupt at the next interrupt threshold.
18 UAIE	UAIE USB Host Asynchronous Interrupt Enable When this bit is one, and the UAI bit in the n_USBSTS register is one, host controller will issue an interrupt at the next interrupt threshold.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 —	- Reserved
16 NAKE	NAKE NAK Interrupt Enable When this bit is one and the NAKI bit in n_USBSTS register is a one the controller will issue an interrupt.
15-11 —	- These bits are reserved and should be set to zero.
10 ULPIE	ULPIE ULPI Interrupt Enable When this bit is one and the UPLII bit in n_USBSTS register is a one the controller will issue an interrupt. This bit is usable only if the controller support UPLI interface mode.
9 —	- Reserved
8 SLE	SLE Sleep Interrupt Enable When this bit is one and the SLI bit in n_n_USBSTS register is a one the controller will issue an interrupt. Only used in device operation mode.
7 SRE	SRE SOF Received Interrupt Enable When this bit is one and the SRI bit in n_USBSTS register is a one the controller will issue an interrupt.
6 URE	URE USB Reset Interrupt Enable When this bit is one and the URI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in device operation mode.
5 AAE	AAE Async Advance Interrupt Enable When this bit is one and the AAI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in host operation mode.
4 SEE	SEE System Error Interrupt Enable When this bit is one and the SEI bit in n_USBSTS register is a one the controller will issue an interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Only used in host operation mode.
3 FRE	FRE Frame List Rollover Interrupt Enable When this bit is one and the FRI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in host operation mode.
2 PCE	PCE Port Change Detect Interrupt Enable When this bit is one and the PCI bit in n_USBSTS register is a one the controller will issue an interrupt.
1 UEE	UEE USB Error Interrupt Enable When this bit is one and the UEI bit in n_USBSTS register is a one the controller will issue an interrupt.
0 UE	UE USB Interrupt Enable When this bit is one and the UI bit in n_USBSTS register is a one the controller will issue an interrupt.

56.7.1.1.22 USB Frame Index (FRINDEX)

Offset

Register	Offset
FRINDEX	14Ch

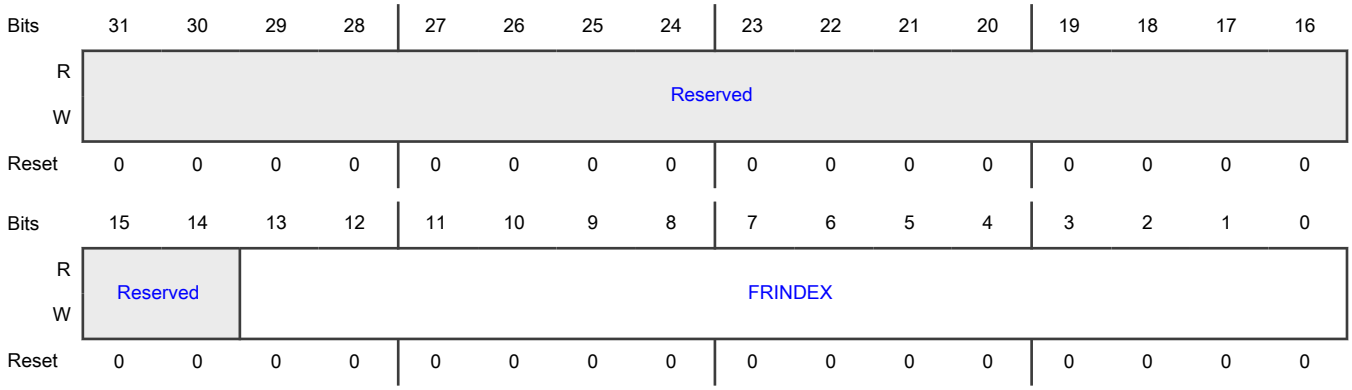
Function

This register is used by the host controller to index the periodic frame list. The register updates every 125 microseconds (once each micro-frame). Bits [N: 3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index depends on the size of the frame list as set by system software in the Frame List Size field in the n_USBCMD register.

This register must be written as a DWord. Byte writes produce undefined results. This register cannot be written unless the Host Controller is in the 'Halted' state as indicated by the HCHalted bit. A write to this register while the Run/Stop hit is set to a one produces undefined results. Writes to this register also affect the SOF value.

In device mode this register is read only and, the device controller updates the FRINDEX [13:3] register from the frame number indicated by the SOF marker. Whenever a SOF is received by the USB bus, FRINDEX [13:3] will be checked against the SOF marker. If FRINDEX [13:3] is different from the SOF marker, FRINDEX [13:3] will be set to the SOF value and FRINDEX [2:0] will be set to zero (that is, SOF for 1 ms frame). If FRINDEX [13:3] is equal to the SOF value, FRINDEX [2:0] will be increment (that is, SOF for 125 us micro-frame.).

Diagram



Fields

Field	Function
31-14 —	- Reserved
13-0 FRINDEX	FRINDEX Frame Index. The value, in this register, increments at the end of each time frame (micro-frame). Bits [N: 3] are used for the Frame List current index. This means that each location of the frame list is accessed 8 times (frames or micro-frames) before moving to the next index. The following illustrates values of N based on the value of the Frame List Size field in the USBCMD register, when used in Host mode. USBCMD [Frame List Size] Number Elements N In device mode the value is the current frame number of the last frame transmitted. It is not used as an index. In either mode bits 2:0 indicate the current microframe. The bit field values description below is represented as (Frame List Size) Number Elements N. 00_0000_0000_0000b - (1024) 12 00_0000_0000_0001b - (512) 11 00_0000_0000_0010b - (256) 10 00_0000_0000_0011b - (128) 9 00_0000_0000_0100b - (64) 8 00_0000_0000_0101b - (32) 7 00_0000_0000_0110b - (16) 6 00_0000_0000_0111b - (8) 5

56.7.1.1.23 Device Address (DEVICEADDR)

Offset

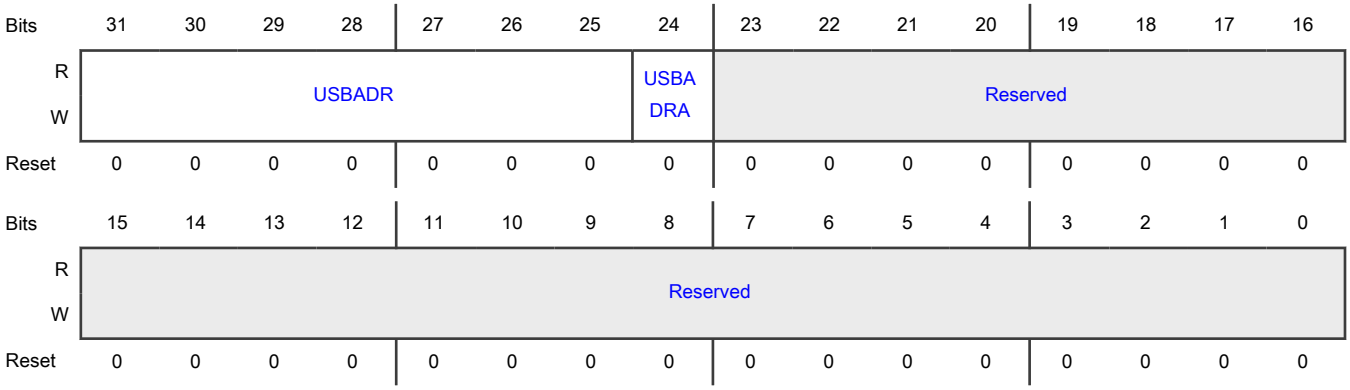
Register	Offset
DEVICEADDR	154h

Function

Device Controller only

The upper seven bits of this register represent the device address. After any controller reset or a USB reset, the device address is set to the default address (0). The default address will match all incoming addresses. Software shall reprogram the address after receiving a SET_ADDRESS descriptor.

Diagram



Fields

Field	Function
31-25 USBADR	USBADR Device Address. These bits correspond to the USB device address
24 USBADRA	USBADRA Device Address Advance. Default=0. When this bit is '0', any writes to USBADR are instantaneous. When this bit is written to a '1' at the same time or before USBADR is written, the write to the USBADR field is staged and held in a hidden register. After an IN occurs on endpoint 0 and is ACKed, USBADR will be loaded from the holding register. Hardware will automatically clear this bit on the following conditions: 1) IN is ACKed to endpoint 0. (USBADR is updated from staging register). 2) OUT/SETUP occur to endpoint 0. (USBADR is not updated). 3) Device Reset occurs (USBADR is reset to 0).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE After the status phase of the SET_ADDRESS descriptor, the DCD has 2 ms to program the USBADR field. This mechanism will ensure this specification is met when the DCD can not write of the device address within 2ms from the SET_ADDRESS status phase. If the DCD writes the USBADR with USBADRA=1 after the SET_ADDRESS data phase (before the prime of the status phase), the USBADR will be programmed instantly at the correct time and meet the 2ms USB requirement.
23-0 —	- Reserved

56.7.1.1.24 Frame List Base Address (PERIODICLISTBASE)

Offset

Register	Offset
PERIODICLISTBASE	154h

Function

Host Controller only

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. HCD loads this register prior to starting the schedule execution by the Host Controller. The memory structure referenced by this physical memory pointer is assumed to be 4-Kbyte aligned. The contents of this register are combined with the Frame Index Register (USB_n_FRINDEX) to enable the Host Controller to step through the Periodic Frame List in sequence.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BASEADR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BASEADR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 BASEADR	BASEADR Base Address (Low). These bits correspond to memory address signals [31:12], respectively. Only used by the host controller.
11-0 —	- Reserved

56.7.1.1.25 Next Asynch. Address (ASYNCLISTADDR)

Offset

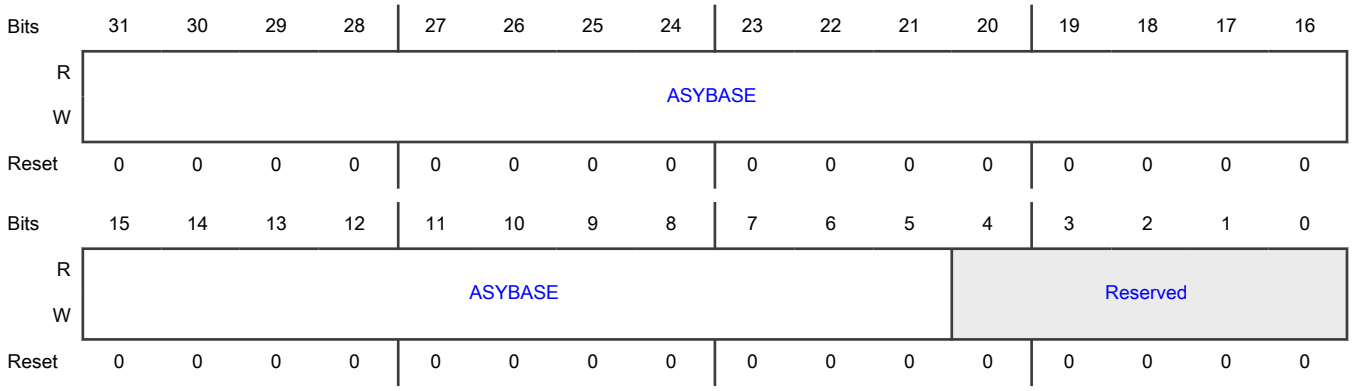
Register	Offset
ASYNCLISTADDR	158h

Function

Host Controller only

This 32-bit register contains the address of the next asynchronous queue head to be executed by the host. Bits [4:0] of this register cannot be modified by the system software and will always return a zero when read.

Diagram



Fields

Field	Function
31-5 ASYBASE	ASYBASE Link Pointer Low (LPL).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (QH). Only used by the host controller.
4-0 —	- Reserved

56.7.1.1.26 Endpoint List Address (ENDPTLISTADDR)

Offset

Register	Offset
ENDPTLISTADDR	158h

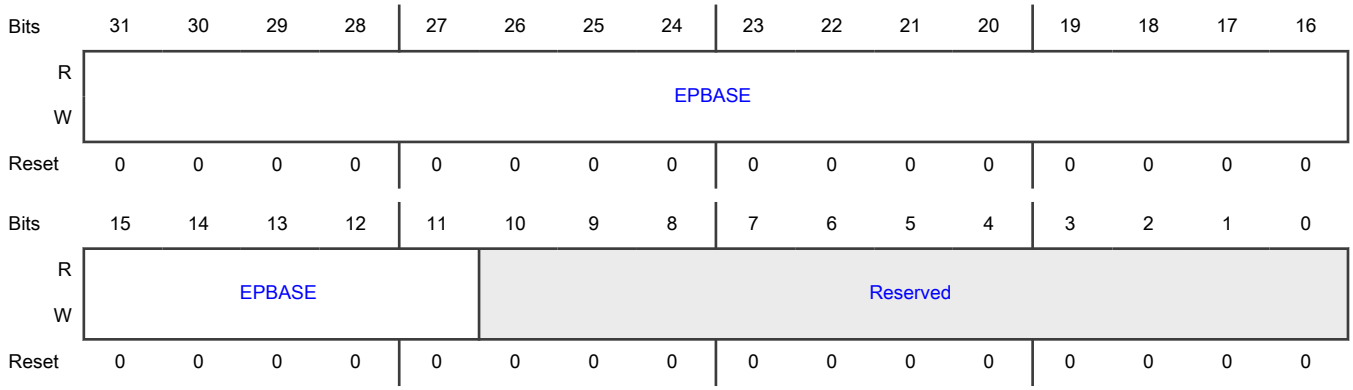
Function

Device Controller only

In device mode, this register contains the address of the top of the endpoint list in system memory. Bits [10:0] of this register cannot be modified by the system software and will always return a zero when read.

The memory structure referenced by this physical memory pointer is assumed 64-byte.

Diagram



Fields

Field	Function
31-11 EPBASE	EPBASE Endpoint List Pointer(Low). These bits correspond to memory address signals [31:11], respectively. This field will reference a list of up to 32 Queue Head (QH) (that is, one queue head per endpoint & direction).

Table continues on the next page...

Table continued from the previous page...

Field	Function
10-0 —	- Reserved

56.7.1.1.27 Programmable Burst Size (BURSTSIZE)

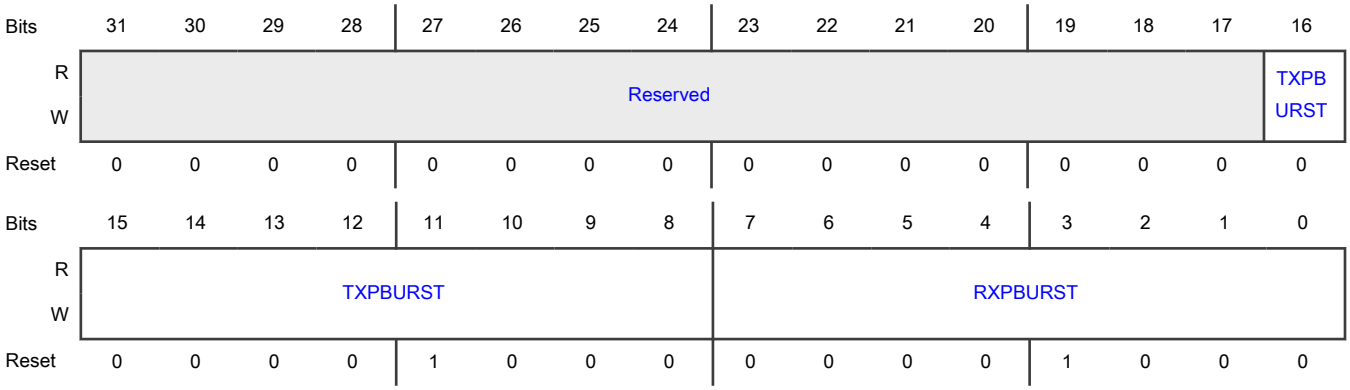
Offset

Register	Offset
BURSTSIZE	160h

Function

This register is used to control the burst size used during data movement on the AHB controller interface. This register is ignored if AHBBRST bits in SBUSCFG register is non-zero value.

Diagram



Fields

Field	Function
31-17 —	- Reserved
16-8 TXPBURST	TXPBURST Programmable TX Burst Size. Default value is determined by TXBURST bits in n_HWTXBUF. This register represents the maximum length of a the burst in 32-bit words while moving data from system memory to the USB bus.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 RXPBURST	RXPBURST Programmable RX Burst Size. Default value is determined by TXBURST bits in n_HWRXBUF. This register represents the maximum length of a the burst in 32-bit words while moving data from the USB bus to system memory.

56.7.1.1.28 TX FIFO Fill Tuning (TXFILLTUNING)

Offset

Register	Offset
TXFILLTUNING	164h

Function

The fields in this register control performance tuning associated with how the host controller posts data to the TX latency FIFO before moving the data onto the USB bus. The specific areas of performance include the how much data to post into the FIFO and an estimate for how long that operation should take in the target system.

Definitions:

T_0 = Standard packet overhead

T_1 = Time to send data payload

T_{ff} = Time to fetch packet into TX FIFO up to specified level.

T_s = Total Packet Flight Time (send-only) packet

$T_s = T_0 + T_1$

T_p = Total Packet Time (fetch and send) packet

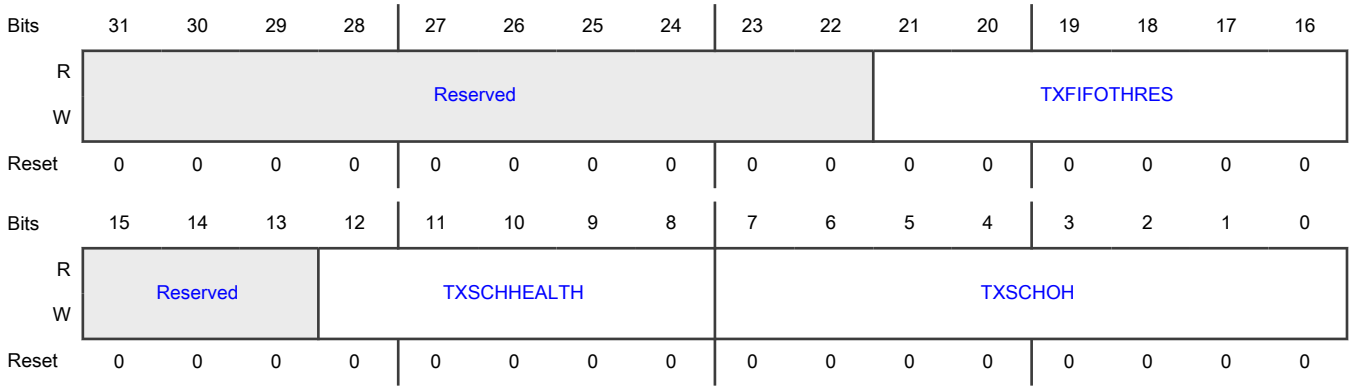
$T_p = T_{ff} + T_0 + T_1$

Upon discovery of a transmit (OUT/SETUP) packet in the data structures, host controller checks to ensure T_p remains before the end of the [micro]frame. If so it proceeds to pre-fill the TX FIFO. If at anytime during the pre-fill operation the time remaining the [micro]frame is < T_s then the packet attempt ceases and the packet is tried at a later time. Although this is not an error condition and the host controller will eventually recover, a mark will be made the scheduler health counter to note the occurrence of a "back-off" event. When a back-off event is detected, the partial packet fetched may need to be discarded from the latency buffer to make room for periodic traffic that will begin after the next SOF. Too many back-off events can waste bandwidth and power on the system bus and thus should be minimized (not necessarily eliminated). Back-offs can be minimized with use of the n_TSCHHEALTH (T_{ff}) described below.

NOTE

The reset value could vary from instance to instance. Please see the detail in bit field description and ignore reset value in summary table in this case!

Diagram



Fields

Field	Function
31-22 —	- Reserved
21-16 TXFIFOTHRES	TXFIFOTHRES FIFO Burst Threshold. (Read/Write) This register controls the number of data bursts that are posted to the TX latency FIFO in Host mode before the packet begins on to the bus. The minimum value is 2 and this value should be a low as possible to maximize USB performance. A higher value can be used in systems with unpredictable latency and/or insufficient bandwidth where the FIFO may underrun because the data transferred from the latency FIFO to USB occurs before it can be replenished from system memory. This value is ignored if the Stream Disable bit in USB_n_USBMODE register is set.
15-13 —	- Reserved
12-8 TXSCHHEALTH	TXSCHHEALTH Scheduler Health Counter. (Read/Write To Clear) This register increments when the host controller fails to fill the TX latency FIFO to the level programmed by TXFIFOTHRES before running out of time to send the packet before the next Start-Of-Frame. This health counter measures the number of times this occurs to provide feedback to selecting a proper TXSCHOH. Writing to this register will clear the counter and this counter will max. at 31.
7-0 TXSCHOH	TXSCHOH Scheduler Overhead. (Read/Write) [Default = 0] This register adds an additional fixed offset to the schedule time estimator described above as Tff. As an approximation, the value chosen for this register should limit the number of back-off events captured in the TXSCHHEALTH to less than 10 per second in a highly utilized bus. Choosing a value that is too high for this register is not desired as it can needlessly reduce USB utilization. The time unit represented in this register is 1.267us when a device is connected in High-Speed Mode. The time unit represented in this register is 6.333us when a device is connected in Low/Full Speed Mode. Default value is '08h' for OTG controller core .

56.7.1.1.29 Endpoint NAK (ENDPTNAK)

Offset

Register	Offset
ENDPTNAK	178h

Function
Endpoint sends NAK handshake for the IN/OUT/PING token.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								EPTN							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								EPRN							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	- Reserved
23-16 EPTN	EPTN TX Endpoint NAK - R/WC. Each TX endpoint has 1 bit in this field. The bit is set when the device sends a NAK handshake on a received IN token for the corresponding endpoint. Bit [N] - Endpoint #[N], N is 0-7
15-8 —	- Reserved
7-0 EPRN	EPRN RX Endpoint NAK - R/WC. Each RX endpoint has 1 bit in this field. The bit is set when the device sends a NAK handshake on a received OUT or PING token for the corresponding endpoint. Bit [N] - Endpoint #[N], N is 0-7

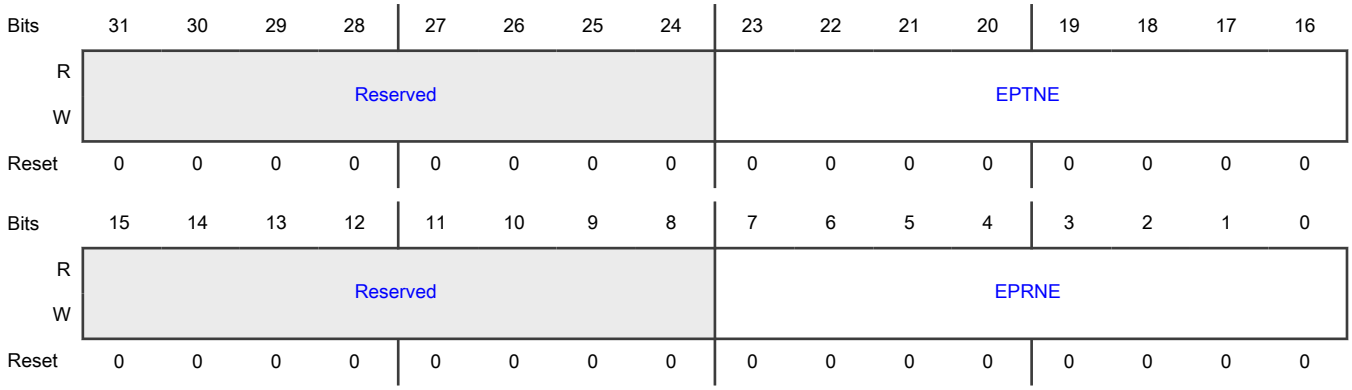
56.7.1.1.30 Endpoint NAK Enable (ENDPTNAKEN)

Offset

Register	Offset
ENDPTNAKEN	17Ch

Function
Endpoint NAK indication enable.

Diagram



Fields

Field	Function
31-24 —	- Reserved
23-16 EPTNE	EPTNE TX Endpoint NAK Enable - R/W. Each bit is an enable bit for the corresponding TX Endpoint NAK bit. If this bit is set and the corresponding TX Endpoint NAK bit is set, the NAK Interrupt bit is set. Bit [N] - Endpoint #[N], N is 0-7
15-8 —	- Reserved
7-0 EPRNE	EPRNE RX Endpoint NAK Enable - R/W. Each bit is an enable bit for the corresponding RX Endpoint NAK bit. If this bit is set and the corresponding RX Endpoint NAK bit is set, the NAK Interrupt bit is set. Bit [N] - Endpoint #[N], N is 0-7

56.7.1.1.31 Configure Flag (CONFIGFLAG)

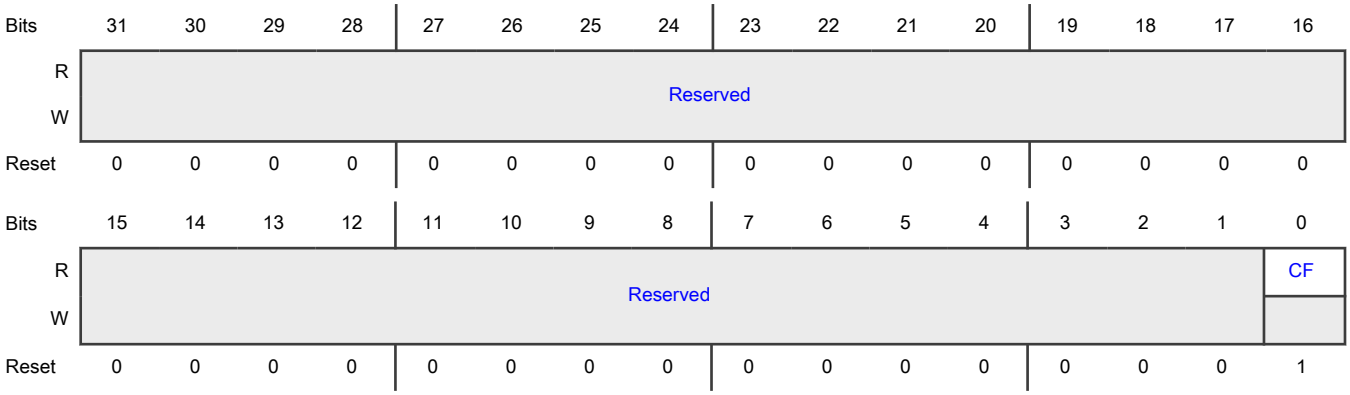
Offset

Register	Offset
CONFIGFLAG	180h

Function

This register is not used in this implementation. A read from this register returns a constant of a 00000001h to indicate that all port routings default to this host controller.

Diagram



Fields

Field	Function
31-1 —	- Reserved
0 CF	CF Configure Flag Host software sets this bit as the last action in its process of configuring the Host Controller. This bit controls the default port-routing control logic. 0b - Port routing control logic default-routes each port to an implementation dependent classic host controller. 1b - Port routing control logic default-routes all ports to this host controller.

56.7.1.1.32 Port Status & Control (PORTSC1)

Offset

Register	Offset
PORTSC1	184h

Function

Host Controller

A host controller could implement one to eight port status and control registers. The number is determined by N_PORTs bits in HWSPARAMs register (please see HCSPARAMS). Software could read this parameter register to determine how many ports need service.

All controller cores on this product are Single-Port, so there is only one port status and control register for each controller core.

This register is only reset by power on reset or controller core reset. The initial conditions of a port are:

- No device connected
- Port disabled

If the port supports power control, this state remains until port power is supplied (by software).

Device Controller

A controller operating in device mode has only port register one (PORTSC1) and it does not support power control in that mode. Port control in device mode is only used for status port reset, suspend, and current connect status. It is also used to initiate test mode or force signaling and allows software to put the PHY into low power suspend mode and disable the PHY clock.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PTS_1		STS	PTW	PSPD		PTS_2	PFSC	PHCD	WKOC	WKDC	WKCN	PTC			
W																
Reset	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PIC		PO	PP	LS		HSP	PR	SUSP	FPR	OCC	OCA	PEC	PE	CSC	CCS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-30 PTS_1	PTS_1 Bit field {bit25, bit31, bit30}: "000b" UTMI/UTMI+ "001b" Reserved "010b" ULPI "011b" Serial/USB 1.1 PHY/IC-USB (FS Only) "100b" HSIC Parallel Transceiver Select (bit25, bit31, bit30). For OTG core, it's Read-Only. For Host1/Host2/Host3 core, it's Read/Write.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE All USB port interface modes are listed in this field description, but not all are supported. For detail feature of each controller core, please see Features . The behaviour is unknown when unsupported interface mode is selected.
29 STS	STS Serial Transceiver Select 1 Serial Interface Engine is selected 0 Parallel Interface signals is selected Serial Interface Engine can be used in combination with UTMI+/ULPI physical interface to provide FS/LS signaling instead of the parallel interface signals. When this bit is set '1b', serial interface engine will be used instead of parallel interface signals. This bit has no effect unless PTS bits is set to select UTMI+/ULPI interface. The Serial/USB1.1 PHY/IC-USB will use the serial interface engine for FS/LS signaling regardless of this bit value.
28 PTW	PTW Parallel Transceiver Width This bit has no effect if serial interface engine is used. 0b - Select the 8-bit UTMI interface [60MHz] 1b - Select the 16-bit UTMI interface [30MHz]
27-26 PSPD	PSPD Port Speed - Read Only. Indicates the speed at which the port is operating. 00b - Full Speed 01b - Low Speed 10b - High Speed 11b - Undefined
25 PTS_2	PTS_2 See description at bits 31-30
24 PFSC	PFSC Port Force Full Speed Connect - Read/Write. Default = 0b. When this bit is set to '1b', the port will be forced to only connect at Full Speed, It disables the chirp sequence that allows the port to identify itself as High Speed. 0b - Normal operation

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Forced to full speed
23 PHCD	PHCD PHY Low Power Suspend - Clock Disable (PLPSCD) - Read/Write. Default = 0b. When this bit is set to '1b', the PHY clock is disabled. Reading this bit will indicate the status of the PHY clock. **NOTE** The PHY clock cannot be disabled if it is being used as the system clock. In device mode, The PHY can be put into Low Power Suspend when the device is not running (USBCMD Run/Stop=0b) or the host has signalled suspend (PORTSC1 SUSPEND=1b). PHY Low power suspend will be cleared automatically when the host initials resume. Before forcing a resume from the device, the device controller driver must clear this bit. In Host mode, the PHY can be put into Low Power Suspend when the downstream device has been put into suspend mode or when no downstream device is connected. Low power suspend is completely under the control of software. 0b - Enable PHY clock 1b - Disable PHY clock
22 WKOC	WKOC Wake on Over-current Enable (WKOC_E) - Read/Write. Default = 0b. Writing this bit to a one enables the port to be sensitive to over-current conditions as wake-up events. This field is zero if *Port Power(* PORTSC1 *)* is zero.
21 WKDC	WKDC Wake on Disconnect Enable (WKDSCNNT_E) - Read/Write. Default=0b. Writing this bit to a one enables the port to be sensitive to device disconnects as wake-up events. This field is zero if *Port Power(* PORTSC1 *)* is zero or in device mode.
20 WKCN	WKCN Wake on Connect Enable (WKCNNT_E) - Read/Write. Default=0b. Writing this bit to a one enables the port to be sensitive to device connects as wake-up events. This field is zero if *Port Power(* PORTSC1 *)* is zero or in device mode.
19-16 PTC	PTC Port Test Control - Read/Write. Default = 0000b. Refer to Port test mode for the operational model for using these test modes and the USB Specification Revision 2.0, Chapter 7 for details on each test mode. The FORCE_ENABLE_FS and FORCE ENABLE_LS are extensions to the test mode support specified in the EHCI specification. Writing the PTC field to any of the FORCE_ENABLE_{HS/FS/LS} values will force the port into the connected and enabled state at the selected speed. Writing the PTC field back to TEST_MODE_DISABLE will allow the port state machines to progress normally from that point.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE *Low speed operations are not supported as a peripheral device.* Any other value than zero indicates that the port is operating in test mode. Value Specific Test 0000b - TEST_MODE_DISABLE 0001b - J_STATE 0010b - K_STATE 0011b - SE0 (host) / NAK (device) 0100b - Packet 0101b - FORCE_ENABLE_HS 0110b - FORCE_ENABLE_FS 0111b - FORCE_ENABLE_LS 1000b-1111b - Reserved
15-14 PIC	PIC Port Indicator Control - Read/Write. Default = Ob. Writing to this field has no effect if the P_INDICATOR bit in the HCSPARAMS register is a zero. Refer to the USB Specification Revision 2.0 for a description on how these bits are to be used. This field is zero if *Port Power* is zero. Bit Value Meaning 00b - Port indicators are off 01b - Amber 10b - Green 11b - Undefined
13 PO	PO Port Owner-Read/Write. Default = 0. This bit unconditionally goes to a 0 when the configured bit in the CONFIGFLAG register makes a 0 to 1 transition. This bit unconditionally goes to 1 whenever the Configured bit is zero System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes a one to this bit when the attached device is not a high-speed device. A one in this bit means that an internal companion controller owns and controls the port. Port owner handoff is not supported in all controller cores, therefore this bit will always be 0.
12 PP	PP Port Power (PP)-Read/Write or Read Only.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The function of this bit depends on the value of the Port Power Switching (PPC) field in the HCSPARAMS register. The behavior is as follows: PPC PP Operation 0 1b *Read Only - Host controller does not have port power control switches. Each port is hard-wired to power.* 1 1b/0b - *Read/Write. OTG controller requires port power control switches. This bit represents the current setting of the switch (0=off, 1=on). When power is not available on a port (that is, PP equals a 0), the port is non-functional and will not report attaches, detaches, etc.* When an over-current condition is detected on a powered port and PPC is a one, the PP bit in each affected port may be transitional by the host controller driver from a one to a zero (removing power from the port). This feature is implemented in all controller cores (PPC = 1).
11-10 LS	LS Line Status-Read Only. These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. In Host mode, the use of linestate by the host controller driver is not necessary (unlike EHCI), because the port controller state machine and the port routing manage the connection of LS and FS. In device mode, the use of linestate by the device controller driver is not necessary. The encoding of the bits are: Bits [11:10] Meaning 00b - SE0 01b - K-state 10b - J-state 11b - Undefined
9 HSP	HSP High-Speed Port - Read Only. Default = 0b. When the bit is one, the host/device connected to the port is in high-speed mode and if set to zero, the host/device connected to the port is not in a high-speed mode. **NOTE** HSP is redundant with PSPD(bit 27, 26) but remained for compatibility.
8 PR	PR Port Reset - Read/Write or Read Only. Default = 0b. In Host Mode: Read/Write. 1=Port is in Reset. 0=Port is not in Reset. Default 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When software writes a one to this bit the bus-reset sequence as defined in the USB Specification Revision 2.0 is started. *This bit will automatically change to zero after the reset sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the reset duration is timed in the driver.* In Device Mode: This bit is a read only status bit. Device reset from the USB bus is also indicated in the USBSTS register. This field is 0 if *Port Power (* PORTSC1 *)* is 0.
7 SUSP	SUSP Suspend - Read/Write or Read Only. Default = 0b. 1=Port in suspend state. 0=Port not in suspend state. In Host Mode: Read/Write. Port Enabled Bit and Suspend bit of this register define the port states as follows: Bits [Port Enabled, Suspend] Port State 0x Disable 10 Enable 11 Suspend When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB. The host controller will unconditionally set this bit to zero when software sets the *Force Port Resume* bit to zero. The host controller ignores a write of zero to this bit. If host software sets this bit to a one when the port is not enabled (that is, *Port enabled* bit is a zero) the results are undefined. This field is 0 if *Port Power (* PORTSC1 *)* is 0 in Host mode. In Device Mode: Read Only. In device mode this bit is a read only status bit.
6 FPR	FPR Force Port Resume -Read/Write. 1= Resume detected/driven on port. 0=No resume (K-state) detected/driven on port. Default = 0. In Host Mode: Software sets this bit to one to drive resume signaling. The Host Controller sets this bit to one if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to a one because a J-to-K transition is detected, the *Port Change Detect* bit in the USBSTS register is also set to one. *This bit will automatically change to zero after the resume sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the resume duration is timed in the driver.*

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Note that when the Host controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains a one. This bit will remain a one until the port has switched to the high-speed idle. Writing a zero has no effect because the port controller will time the resume operation, clear the bit the port control state switches to HS or FS idle. This field is zero if *Port Power(* PORTSC1 *)* is zero in Host mode. This bit is not-EHCI compatible. In Device mode: After the device has been in Suspend State for 5ms or more, software must set this bit to one to drive resume signaling before clearing. The Device Controller will set this bit to one if a J-to-K transition is detected while the port is in the Suspend state. The bit will be cleared when the device returns to normal operation. Also, when this bit will be cleared because a K-to-J transition detected, the *Port Change Detect* bit in the USBSTS register is also set to one.
5 OCC	OCC Over-current Change-R/WC. Default=0. This bit is set '1b' by hardware when there is a change to Over-current Active. Software can clear this bit by writing a one to this bit position.
4 OCA	OCA Over-current Active-Read Only. Default 0. This bit will automatically transition from one to zero when the over current condition is removed. 0b - This port does not have an over-current condition. 1b - This port currently has an over-current condition
3 PEC	PEC Port Enable/Disable Change-R/WC. 1=Port enabled/disabled status has changed. 0 = No change. Default = 0. In Host Mode: For the root hub, this bit is set to a one only when a port is disabled due to disconnect on the port or due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification). Software clears this by writing a one to it. This field is 0 if *Port Power (* PORTSC1 *)* is 0. In Device mode: The device port is always enabled, so this bit is always '0b'.
2 PE	PE Port Enabled/Disabled-Read/Write. 1=Enable. 0=Disable. Default 0. In Host Mode:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a one to this field. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by the host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events. When the port is disabled, (0b) downstream propagation of data is blocked except for reset. This field is 0 if *Port Power(* PORTSC1 *)* is 0 in Host mode. In Device Mode: The device port is always enabled, so this bit is always '1b'.
1 CSC	CSC Connect Status Change-R/WC. 1 =Change in Current Connect Status. 0=No change. Default 0. In Host Mode: Indicates a change has occurred in the port's Current Connect Status. The host/device controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be 'setting' an already-set bit (that is, the bit will remain set). Software clears this bit by writing a one to it. This field is 0 if *Port Power (* PORTSC1 *)* is 0 in Host mode. In Device Mode: This bit is undefined in device controller mode.
0 CCS	CCS Current Connect Status-Read Only. In Host Mode: 1=Device is present on port. 0=No device is present. Default = 0. This value reflects the current state of the port, and may not correspond directly to the event that caused the *Connect Status Change* bit (Bit 1) to be set. This field is 0 if *Port Power (* PORTSC1 *)* is 0 in Host mode. In Device Mode: 1=Attached. 0=Not Attached. Default=0. A one indicates that the device successfully attached and is operating in either high speed or full speed as indicated by the High Speed Port bit in this register. A zero indicates that the device did not attach successfully or was forcibly disconnected by the software writing a zero to the Run bit in the USBCMD register. It does not state the device being disconnected or suspended.

56.7.1.1.33 On-The-Go Status & control (OTGSC)

Offset

Register	Offset
OTGSC	1A4h

Function

This register is available only in OTG controller core. It has four sections:

- OTG Interrupt enables (Read/Write)
- OTG Interrupt status (Read/Write to Clear)
- OTG Status inputs (Read Only)
- OTG Controls (Read/Write)

The status inputs are debounced using a 1 ms time constant. Values on the status inputs that do not persist for more than 1 ms does not cause an update of the status input register, or cause an OTG interrupt.

See also USBMODE.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	DPIE	EN_1MS	BSEIE	BSVIE	ASVIE	AVVIE	IDIE	Reserved	DPIS	STATUS_...	BSEIS	BSVIS	ASVIS	AVVIS	IDIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	DPS	TOG_1MS	BSE	BSV	ASV	AVV	ID	Reserved		IDPU	DP	OT	Reserved	VC	VD
W																
Reset	0	0	1	0	1	1	1	1	0	0	1	0	0	0	0	0

Fields

Field	Function
31 —	- Reserved
30 DPIE	DPIE Data Pulse Interrupt Enable
29 EN_1MS	EN_1MS 1 millisecond timer Interrupt Enable - Read/Write
28 BSEIE	BSEIE B Session End Interrupt Enable - Read/Write. Setting this bit enables the B session end interrupt.
27 BSVIE	BSVIE B Session Valid Interrupt Enable - Read/Write. Setting this bit enables the B session valid interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 ASVIE	ASVIE A Session Valid Interrupt Enable - Read/Write. Setting this bit enables the A session valid interrupt.
25 AVVIE	AVVIE A VBus Valid Interrupt Enable - Read/Write. Setting this bit enables the A VBus valid interrupt.
24 IDIE	IDIE USB ID Interrupt Enable - Read/Write. Setting this bit enables the USB ID interrupt.
23 —	- Reserved
22 DPIS	DPIS Data Pulse Interrupt Status - Read/Write to Clear. This bit is set when data bus pulsing occurs on DP or DM. Data bus pulsing is only detected when USBMODE.CM = Host (11) and PORTSC1(0)[PP] = 0. Software must write a one to clear this bit.
21 STATUS_1MS	STATUS_1MS 1 millisecond timer Interrupt Status - Read/Write to Clear. This bit is set once every millisecond. Software must write a one to clear this bit.
20 BSEIS	BSEIS B Session End Interrupt Status - Read/Write to Clear. This bit is set when VBus has fallen below the B session end threshold. Software must write a one to clear this bit.
19 BSVIS	BSVIS B Session Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the B session valid threshold. Software must write a one to clear this bit.
18 ASVIS	ASVIS A Session Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the A session valid threshold.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Software must write a one to clear this bit.
17 AVVIS	AVVIS A VBus Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the VBus valid threshold on an A device. Software must write a one to clear this bit.
16 IDIS	IDIS USB ID Interrupt Status - Read/Write. This bit is set when a change on the ID input has been detected. Software must write a one to clear this bit.
15 —	- Reserved
14 DPS	DPS Data Bus Pulsing Status - Read Only. A '1' indicates data bus pulsing is being detected on the port.
13 TOG_1MS	TOG_1MS 1 millisecond timer toggle - Read Only. This bit toggles once per millisecond.
12 BSE	BSE B Session End - Read Only. Indicates VBus is below the B session end threshold.
11 BSV	BSV B Session Valid - Read Only. Indicates VBus is above the B session valid threshold.
10 ASV	ASV A Session Valid - Read Only. Indicates VBus is above the A session valid threshold.
9 AVV	AVV A VBus Valid - Read Only. Indicates VBus is above the A VBus valid threshold.
8	ID

Table continues on the next page...

Table continued from the previous page...

Field	Function
ID	USB ID - Read Only. 0 = A device, 1 = B device
7-6 —	- Reserved
5 IDPU	IDPU ID Pullup - Read/Write This bit provide control over the ID pull-up resistor; 0 = off, 1 = on [default]. When this bit is 0, the ID input will not be sampled.
4 DP	DP Data Pulsing - Read/Write. Setting this bit causes the pullup on DP to be asserted for data pulsing during SRP.
3 OT	OT OTG Termination - Read/Write. This bit must be set when the OTG device is in device mode, this controls the pulldown on DM.
2 —	- Reserved
1 VC	VC VBUS Charge - Read/Write. Setting this bit causes the VBus line to be charged. This is used for VBus pulsing during SRP.
0 VD	VD VBUS_Discharge - Read/Write. Setting this bit causes VBus to discharge through a resistor.

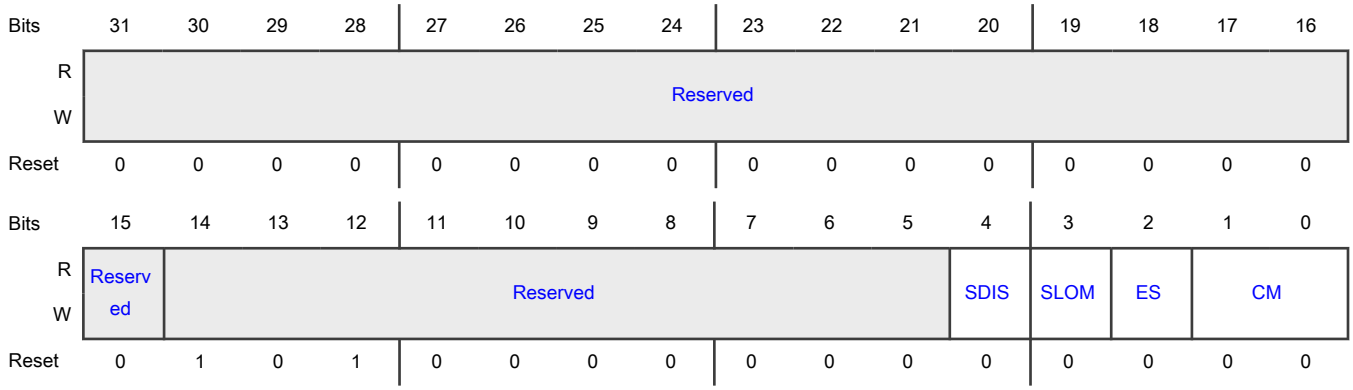
56.7.1.1.34 USB Device Mode (USBMODE)

Offset

Register	Offset
USBMODE	1A8h

Function
USB Mode Selection

Diagram



Fields

Field	Function
31-16 —	- Reserved
15 —	- Reserved
14-5 —	- Reserved
4 SDIS	SDIS Stream Disable Mode. (0 - Inactive [default]; 1 - Active) Device Mode: Setting to a '1' disables double priming on both RX and TX for low bandwidth systems. This mode ensures that when the RX and TX buffers are sufficient to contain an entire packet that the standard double buffering scheme is disabled to prevent overruns/underruns in bandwidth limited systems. Note: In High Speed Mode, all packets received are responded to with a NYET handshake when stream disable is active. Host Mode: Setting to a '1' ensures that overruns/underruns of the latency FIFO are eliminated for low bandwidth systems where the RX and TX buffers are sufficient to contain the entire packet. Enabling stream disable also has the effect of ensuring the TX latency is filled to capacity before the packet is launched onto the USB. **NOTE** Time duration to pre-fill the FIFO becomes significant when stream disable is active. See TXFILLTUNING and TXTTFILLTUNING [MPH Only] to characterize the adjustments needed for the scheduler when using this feature. **NOTE** The use of this feature substantially limits of the overall USB performance that can be achieved.
3	SLOM

Table continues on the next page...

Table continued from the previous page...

Field	Function
SLOM	Setup Lockout Mode. In device mode, this bit controls behavior of the setup lock mechanism. See Control endpoint operation model . 0b - Setup Lockouts On (default); 1b - Setup Lockouts Off. DCD requires use of Setup Data Buffer Tripwire in USBCMD.
2 ES	ES Endian Select - Read/Write. This bit can change the byte alignment of the transfer buffers to match the host microprocessor. The bit fields in the microprocessor interface and the data structures are unaffected by the value of this bit because they are based upon the 32-bit word. Bit Meaning 0b - Little Endian [Default] 1b - Big Endian
1-0 CM	CM Controller Mode - R/WO. Controller mode is defaulted to the proper mode for host only and device only implementations. For those designs that contain both host & device capability, the controller defaults to an idle state and needs to be initialized to the desired operating mode after reset. For combination host/device controllers, this register can only be written once after reset. If it is necessary to switch modes, software must reset the controller by writing to the *RESET* bit in the USBCMD register before reprogramming this register. For OTG controller core, reset value is '00b'. 00b - Idle [Default for combination host/device] 01b - Reserved 10b - Device Controller [Default for device only controller] 11b - Host Controller [Default for host only controller]

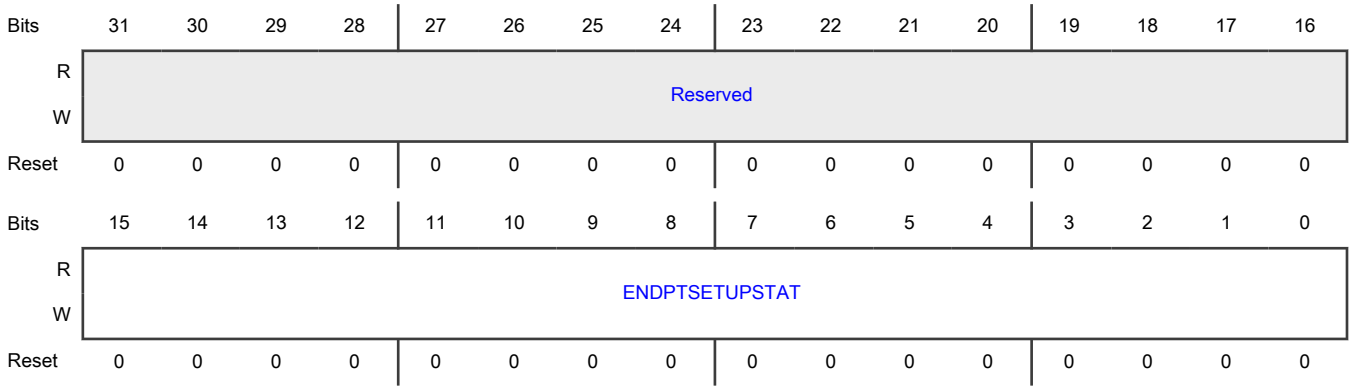
56.7.1.1.35 Endpoint Setup Status (ENDPTSETUPSTAT)

Offset

Register	Offset
ENDPTSETUPSTAT	1ACh

Function
Endpoint Setup Status

Diagram



Fields

Field	Function
31-16 —	- Reserved
15-0 ENDPTSETUPSTAT	ENDPTSETUPSTAT Setup Endpoint Status. For every setup transaction that is received, a corresponding bit in this register is set to one. Software must clear or acknowledge the setup transfer by writing a one to a respective bit after it has read the setup data from Queue head. The response to a setup packet as in the order of operations and total response time is crucial to limit bus time outs while the setup lock our mechanism is engaged. See Managing endpoints in the Device Operational Model. This register is only used in device mode.

56.7.1.1.36 Endpoint Prime (ENDPTPRIME)

Offset

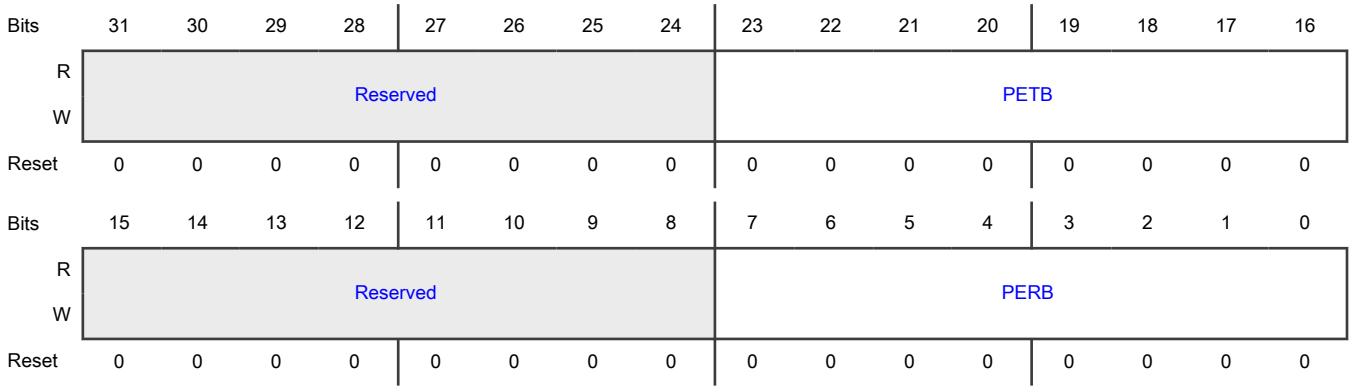
Register	Offset
ENDPTPRIME	1B0h

Function

This register is only used in device mode.

When software sets the prime bit for a given endpoint, the device controller loads the transfer descriptor, pointed to by the queue head, such that the endpoint is ready to transmit or receive when the host sends a request (IN/OUT token). The endpoint will NAK all requests from the host until the endpoint is primed. The controller will automatically re-prime the endpoint with a new transfer descriptor when one is found via the next_dtd pointer of the current transfer descriptor. Hence, the prime bit must only be set by software when a descriptor is added to the queue head.

Diagram



Fields

Field	Function
31-24 —	- Reserved
23-16 PETB	PETB Prime Endpoint Transmit Buffer - R/WS. For each endpoint a corresponding bit is used to request that a buffer is prepared for a transmit operation in order to respond to a USB IN/INTERRUPT transaction. Software should write a one to the corresponding bit when posting a new transfer descriptor to an endpoint queue head. Hardware automatically uses this bit to begin parsing for a new transfer descriptor from the queue head and prepare a transmit buffer. Hardware clears this bit when the associated endpoint(s) is (are) successfully primed. **NOTE** These bits are momentarily set by hardware during hardware re-priming operations when a dTD is retired, and the dQH is updated. PETB[N] - Endpoint #N, N is in 0..7
15-8 —	- Reserved
7-0 PERB	PERB Prime Endpoint Receive Buffer - R/WS. For each endpoint, a corresponding bit is used to request a buffer prepare for a receive operation for when a USB host initiates a USB OUT transaction. Software should write a one to the corresponding bit whenever posting a new transfer descriptor to an endpoint queue head. Hardware automatically uses this bit to begin parsing for a new transfer descriptor from the queue head and prepare a receive buffer. Hardware clears this bit when the associated endpoint(s) is (are) successfully primed. **NOTE** These bits are momentarily set by hardware during hardware re-priming operations when a dTD is retired, and the dQH is updated. PERB[N] - Endpoint #N, N is in 0..7

56.7.1.1.37 Endpoint Flush (ENDPTFLUSH)

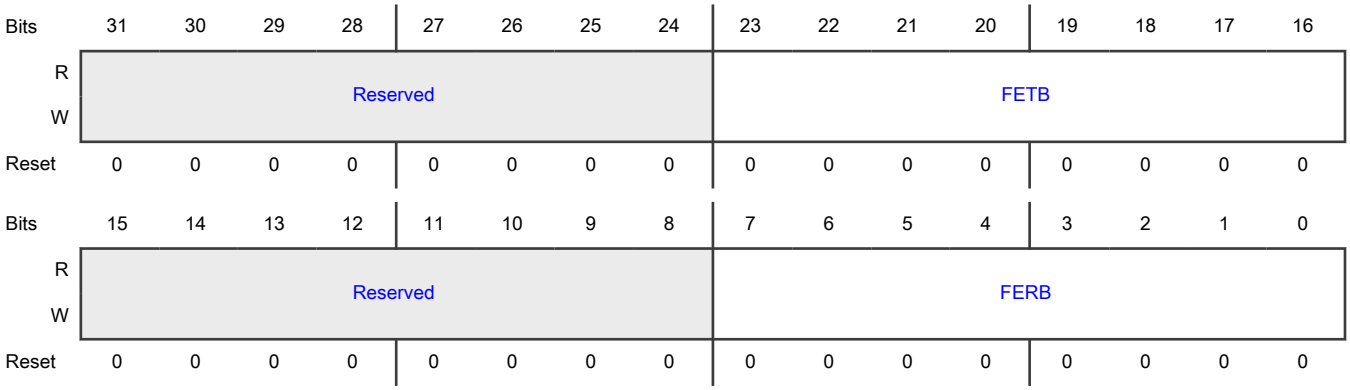
Offset

Register	Offset
ENDPTFLUSH	1B4h

Function

This register is only used in device mode.

Diagram



Fields

Field	Function
31-24 —	- Reserved
23-16 FETB	FETB Flush Endpoint Transmit Buffer - R/WS. Writing one to a bit(s) in this register causes the associated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer continues until completion. Hardware clears this register after the endpoint flush operation is successful. FETB[N] - Endpoint #N, N is in 0..7
15-8 —	- Reserved
7-0 FERB	FERB Flush Endpoint Receive Buffer - R/WS. Writing one to a bit(s) causes the assocUOGiated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer continues until completion. Hardware clears this register after the endpoint flush operation is successful. FERB[N] - Endpoint #N, N is in 0..7\

56.7.1.1.38 Endpoint Status (ENDPTSTAT)

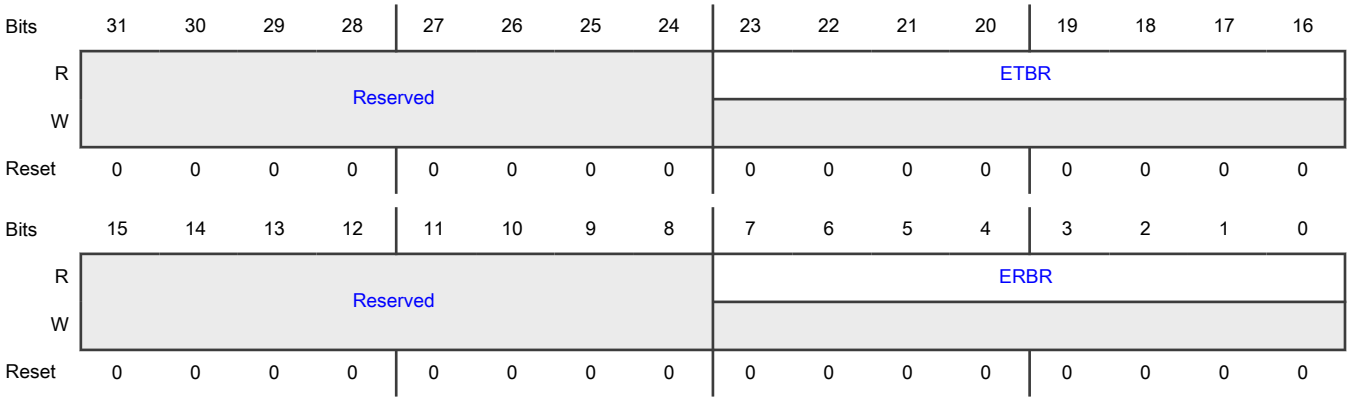
Offset

Register	Offset
ENDPTSTAT	1B8h

Function

This register is only used in device mode.

Diagram



Fields

Field	Function
31-24 —	- Reserved
23-16 ETBR	ETBR Endpoint Transmit Buffer Ready -- Read Only. One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to one by the hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. There is always a delay between setting a bit in the ENDPTPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register. **NOTE** These bits are momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated. ETBR[N] - Endpoint #N, N is in 0..7
15-8 —	- Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 ERBR	ERBR Endpoint Receive Buffer Ready -- Read Only. One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to a one by the hardware as a response to receiving a command from a corresponding bit in the ENDPRIME register. There is always a delay between setting a bit in the ENDPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register. **NOTE** These bits are momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated. ERBR[N] - Endpoint #N, N is in 0..7

56.7.1.1.39 Endpoint Complete (ENDPTCOMPLETE)

Offset

Register	Offset
ENDPTCOMPLETE	1BCh

Function

This register is only used in device mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								ETCE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								ERCE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
23-16 ETCE	ETCE Endpoint Transmit Complete Event - R/WC. Each bit indicates a transmit event (IN/INTERRUPT) occurred and software should read the corresponding endpoint queue to determine the endpoint status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit is set simultaneously with the *USBINT*. Writing one clears the corresponding bit in this register. ETCE[N] - Endpoint #N, N is in 0..7
15-8 —	- Reserved
7-0 ERCE	ERCE Endpoint Receive Complete Event - RW/C. Each bit indicates a received event (OUT/SETUP) occurred and software should read the corresponding endpoint queue to determine the transfer status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit is set simultaneously with the *USBINT*. Writing one clears the corresponding bit in this register. ERCE[N] - Endpoint #N, N is in 0..7

56.7.1.1.40 Endpoint Control0 (ENDPTCTRL0)

Offset

Register	Offset
ENDPTCTRL0	1C0h

Function

Every Device implements Endpoint 0 as a control endpoint.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	Reserved			TXT		Reserved	TXS
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	Reserved			RXT		Reserved	RXS
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	- Reserved
23 TXE	TXE TX Endpoint Enable 1 Enabled Endpoint0 is always enabled.
22-20 —	- Reserved
19-18 TXT	TXT TX Endpoint Type - Read/Write 00 - Control Endpoint0 is fixed as a Control End Point.
17 —	- Reserved
16 TXS	TXS TX Endpoint Stall - Read/Write 0 End Point OK [Default] 1 End Point Stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues returning STALL until the bit is cleared by software or it is automatically cleared upon receipt of a new SETUP request. After receiving a SETUP request, this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. **NOTE** There is a slight delay (50 clocks max.) between the endptsetupstat being cleared and hardware continuing to clear this bit. In most systems it is unlikely the DCD software will observe this delay. However, should the dcd observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a newsetup has been received by checking the associated endptsetupstat bit.
15-8 —	- Reserved
7 RXE	RXE RX Endpoint Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1 Enabled Endpoint0 is always enabled.
6-4 —	- Reserved
3-2 RXT	RXT RX Endpoint Type - Read/Write 00 Control Endpoint0 is fixed as a Control End Point.
1 —	- Reserved
0 RXS	RXS RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues returning STALL until the bit is cleared by software or it is automatically cleared upon receipt of a new SETUP request. After receiving a SETUP request, this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. **NOTE** There is a slight delay (50 clocks max.) between the endptsetupstat being cleared and hardware continuing to clear this bit. In most systems it is unlikely the dcd software will observe this delay. However, should the dcd observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a newsetup has been received by checking the associated endptsetupstat bit.

56.7.1.1.41 Endpoint Control 1 (ENDPTCTRL1)

Offset

Register	Offset
ENDPTCTRL1	1C4h

Function

This is endpoint control register for endpoint 1 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	- Reserved
23 TXE	TXE TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TXR TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TXI TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
19-18 TXT	TXT TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TXD TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TXS TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15-8 —	- Reserved
7 RXE	RXE RX Endpoint Enable 0 Disabled [Default] 1 Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	An Endpoint should be enabled only after it has been configured.
6 RXR	RXR RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RXI RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 —	- Reserved.
3-2 RXT	RXT RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
1 RXD	RXD RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RXS RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

56.7.1.1.42 Endpoint Control 2 (ENDPTCTRL2)

Offset

Register	Offset
ENDPTCTRL2	1C8h

Function

This is endpoint control register for endpoint 2 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	- Reserved
23 TXE	TXE TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TXR TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TXI TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 —	- Reserved
19-18 TXT	TXT TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TXD TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TXS TX Endpoint Stall - Read/Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15-8 —	- Reserved
7 RXE	RXE RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RXR RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RXI RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 —	- Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-2 RXT	RXT RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
1 RXD	RXD RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RXS RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

56.7.1.1.43 Endpoint Control 3 (ENDPTCTRL3)

Offset

Register	Offset
ENDPTCTRL3	1CCh

Function

This is endpoint control register for endpoint 3 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	- Reserved
23 TXE	TXE TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TXR TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TXI TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
19-18 TXT	TXT TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TXD TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TXS TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15-8 —	- Reserved
7 RXE	RXE RX Endpoint Enable 0 Disabled [Default] 1 Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	An Endpoint should be enabled only after it has been configured.
6 RXR	RXR RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RXI RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 —	- Reserved.
3-2 RXT	RXT RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
1 RXD	RXD RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RXS RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

56.7.1.1.44 Endpoint Control 4 (ENDPTCTRL4)

Offset

Register	Offset
ENDPTCTRL4	1D0h

Function

This is endpoint control register for endpoint 4 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	- Reserved
23 TXE	TXE TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TXR TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TXI TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 —	- Reserved
19-18 TXT	TXT TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TXD TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TXS TX Endpoint Stall - Read/Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15-8 —	- Reserved
7 RXE	RXE RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RXR RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RXI RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 —	- Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-2 RXT	RXT RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
1 RXD	RXD RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RXS RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

56.7.1.1.45 Endpoint Control 5 (ENDPTCTRL5)

Offset

Register	Offset
ENDPTCTRL5	1D4h

Function

This is endpoint control register for endpoint 5 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	- Reserved
23 TXE	TXE TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TXR TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TXI TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
19-18 TXT	TXT TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TXD TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TXS TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15-8 —	- Reserved
7 RXE	RXE RX Endpoint Enable 0 Disabled [Default] 1 Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	An Endpoint should be enabled only after it has been configured.
6 RXR	RXR RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RXI RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 —	- Reserved.
3-2 RXT	RXT RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
1 RXD	RXD RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RXS RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

56.7.1.1.46 Endpoint Control 6 (ENDPTCTRL6)

Offset

Register	Offset
ENDPTCTRL6	1D8h

Function

This is endpoint control register for endpoint 6 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	- Reserved
23 TXE	TXE TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TXR TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TXI TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 —	- Reserved
19-18 TXT	TXT TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TXD TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TXS TX Endpoint Stall - Read/Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15-8 —	- Reserved
7 RXE	RXE RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RXR RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RXI RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 —	- Reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-2 RXT	RXT RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
1 RXD	RXD RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RXS RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

56.7.1.1.47 Endpoint Control 7 (ENDPTCTRL7)

Offset

Register	Offset
ENDPTCTRL7	1DCh

Function

This is endpoint control register for endpoint 7 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	- Reserved
23 TXE	TXE TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TXR TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TXI TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20	-

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Reserved
19-18 TXT	TXT TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TXD TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TXS TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15-8 —	- Reserved
7 RXE	RXE RX Endpoint Enable 0 Disabled [Default] 1 Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	An Endpoint should be enabled only after it has been configured.
6 RXR	RXR RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RXI RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 —	- Reserved.
3-2 RXT	RXT RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
1 RXD	RXD RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RXS RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

56.7.2 USBNC memory map

56.7.2.1 USBNC register descriptions

NOTE

- For reserved bits, please preserve the value when writing (read its reset value, then write this value back)

56.7.2.1.1 USBNC memory map

USBNC_OTG1 base address: 42C8_0200h

USBNC_OTG2 base address: 42C9_0200h

Offset	Register	Width (In bits)	Access	Reset value
0h	USB OTG Control 1 Register (CTRL1)	32	RW	3000_1000h
4h	USB OTG Control 2 Register (CTRL2)	32	RW	5F00_0000h
10h	USB Host HSIC Control Register (HSIC_CTRL)	32	RW	1000_4084h

56.7.2.1.2 USB OTG Control 1 Register (CTRL1)

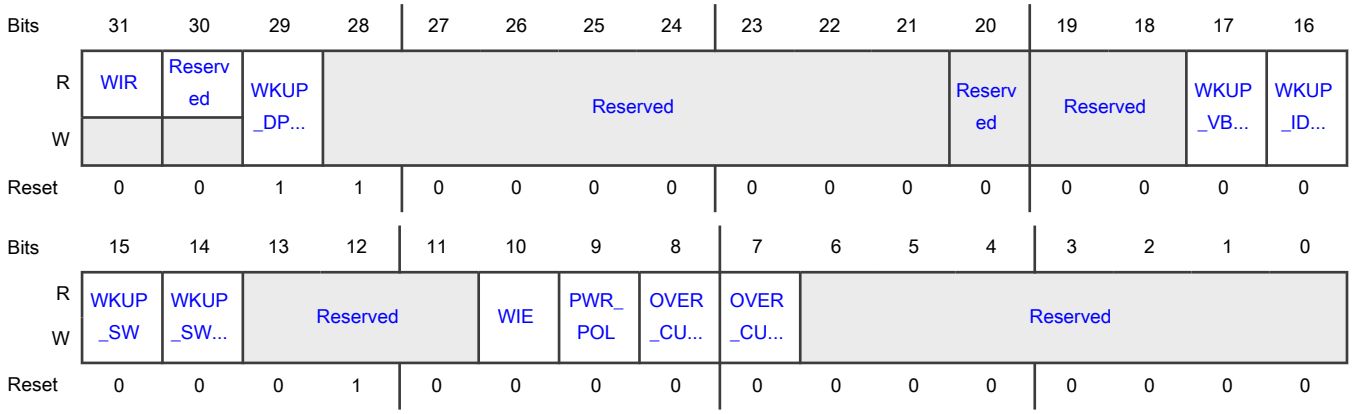
Offset

Register	Offset
CTRL1	0h

Function

The USB Control 1 register controls the integration specific features of the USB OTG1 and OTG2 module. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control and wake-up functionality.

Diagram



Fields

Field	Function
31 WIR	WIR Wake-up Interrupt Request This bit indicates that a wake-up interrupt request is received on the OTG1 port. This bit is cleared by disabling the wake-up interrupt (clearing bit "OWIE"). 0b - No wake-up interrupt request received 1b - Wake-up Interrupt Request received
30 —	Reserved
29 WKUP_DPDM_EN	Wake-up on DP/DM change enable 0b - DP/DM changes wake-up to be disabled only when VBUS is 0. 1b - (Default) DP/DM changes wake-up to be enabled, it is for device only.
28-21 —	Reserved
20 —	- Reserved
19-18 —	- Reserved
17 WKUP_VBUS_EN	WKUP_VBUS_EN Wake-up on VBUS change enable 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 WKUP_ID_EN	WKUP_ID_EN Wake-up on ID change enable 0b - Disable 1b - Enable
15 WKUP_SW	WKUP_SW Software Wake-up 0b - Inactive 1b - Force wake-up
14 WKUP_SW_EN	WKUP_SW_EN Software Wake-up Enable 0b - Disable 1b - Enable
13-11 —	- Reserved
10 WIE	WIE Wake-up Interrupt Enable This bit enables or disables the OTG1 wake-up interrupt. Disabling the interrupt also clears the Interrupt request bit. Wake-up interrupt enable should be turned off after receiving a wake-up interrupt and turned on again prior to going in suspend mode 0b - Interrupt Disabled 1b - Interrupt Enabled
9 PWR_POL	PWR_POL Power Polarity This bit should be set according to PMIC Power Pin polarity. 0b - PMIC Power Pin is Low active. 1b - PMIC Power Pin is High active.
8 OVER_CUR_POL	OVER_CUR_POL Polarity of Overcurrent The polarity of OTG1/OTG2 port overcurrent event 0b - High active (high on this signal represents an overcurrent condition) 1b - Low active (low on this signal represents an overcurrent condition)
7	OVER_CUR_DIS

Table continues on the next page...

Table continued from the previous page...

Field	Function
OVER_CUR_DIS	Disable Overcurrent Detection 0b - Enables overcurrent detection 1b - Disables overcurrent detection
6-0 —	- Reserved

56.7.2.1.3 USB OTG Control 2 Register (CTRL2)

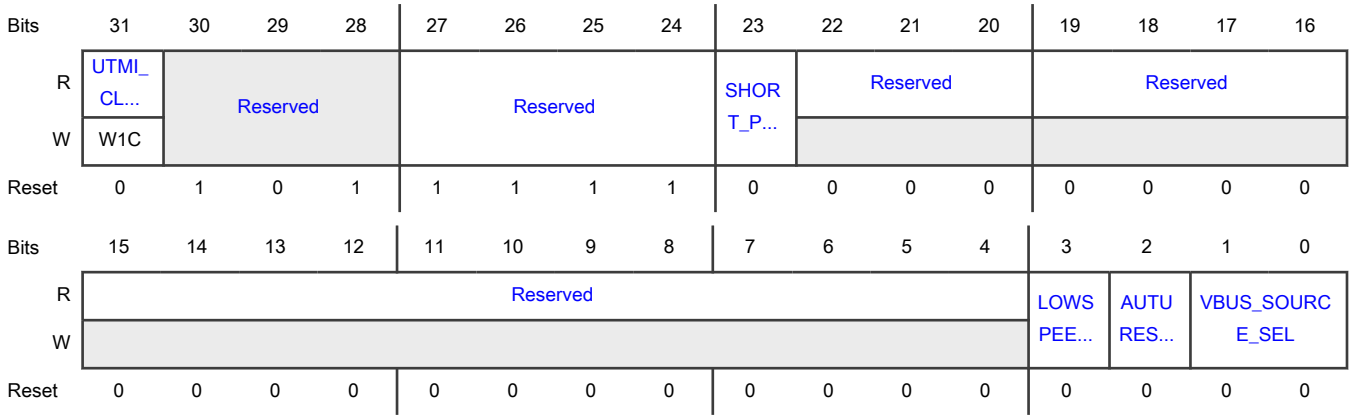
Offset

Register	Offset
CTRL2	4h

Function

The USB Control 2 register controls the integration specific features of the USB module. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control and wakeup functionality.

Diagram



Fields

Field	Function
31 UTMI_CLK_VLD	UTMI_CLK_VLD Indicate whether the UTMI clock to the USB PHY is valid. Write 1 to clear 0b - Default

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-28 —	Reserved
27-24 —	Reserved 1111b - Default
23 SHORT_PKT_EN	Short Packet Interrupt When enabled, USB controller issues an interrupt as a short packet is received in device mode, even if IoC of QTD is not set. 0b - Default
22-20 —	Reserved 000b - Default
19-4 —	Reserved
3 LOWSPEED_EN	LOWSPEED_EN Set if AUTURESUME_EN is set and works on low speed. 0b - Default
2 AUTURESUME_EN	Auto Resume Enable It is for UTMI PHY host mode only (UH core has no this feature since HSIC PHY does not support auto resume). To set USB, it will send resume with 32 KHz clock when it detects remote wakeup from device. This feature is useful if device drive a very short remote wakeup (minimul value is 1 ms for USB2 spec) and system 24 MHz is off during suspend mode. 0b - Default
1-0 VBUS_SOURCE_SEL	VBUS_SOURCE_SEL VBUS source select when detect VBUS wakeup event, it is for UTMI PHY only (UH core has no such feature). 00b - vbus_valid 01b - sess_valid 10b - sess_valid 11b - sess_valid

56.7.2.1.4 USB Host HSIC Control Register (HSIC_CTRL)

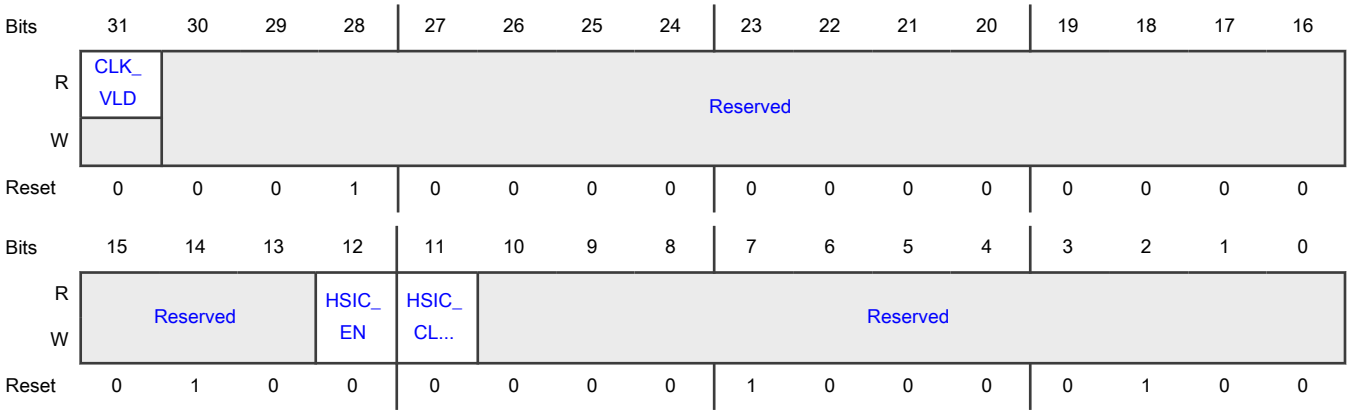
Offset

Register	Offset
HSIC_CTRL	10h

Function

The USB Host HSIC control register controls Host high speed IC configuration. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control.

Diagram



Fields

Field	Function
31 CLK_VLD	CLK_VLD Indicating whether Host HSIC clock is valid. 0b - Invalid 1b - Valid
30-13 —	- Reserved
12 HSIC_EN	HSIC_EN Host HSIC enable 0b - Disabled 1b - Enabled
11 HSIC_CLK_ON	HSIC_CLK_ON Force Host HSIC module 480M clock on, even when in Host is in suspend mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Inactive 1b - Active
10-0 —	- Reserved

Chapter 57
USB Device Charger Detection Module (USBDCD)

57.1 Chip-specific USBDCD information

Table 908. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

57.2 Overview

USBDCD works with the USB transceiver to detect whether the USB device is attached to a Charging Port, either a Dedicated Charging Port (DCP) or a Charging Downstream Port (CDP). System software coordinates the detection activities of the module and controls an off-chip integrated circuit that performs the battery charging.

The use of the term "USB transceiver" in this module documentation applies to the USB physical layer instance on the chip, whether it is an Full Speed (FS)/Low Speed (LS) only transceiver or an High Speed (HS)/FS/LS capable PHY.

57.2.1 Block diagram

The following figure is a high-level block diagram of the module.

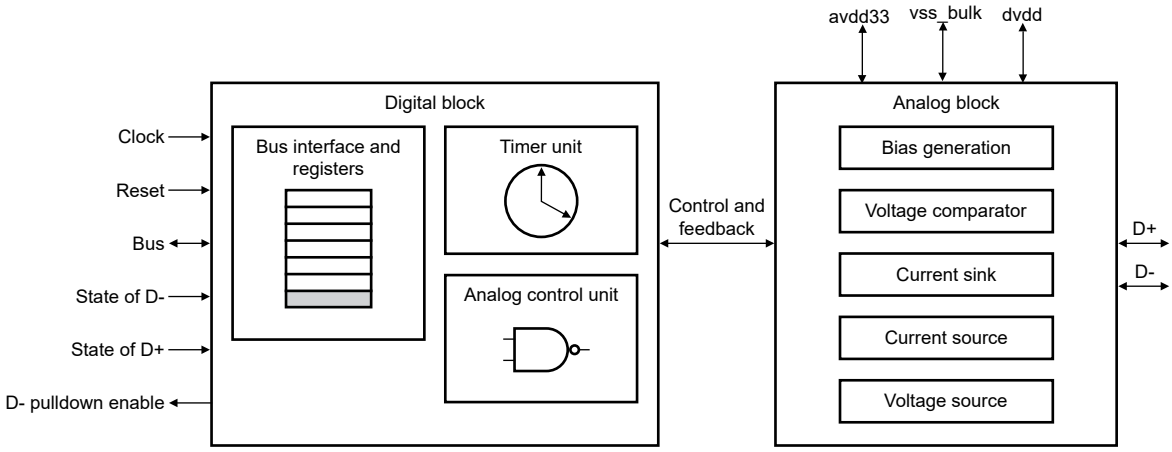


Figure 429. Block diagram

USBDCD consists of two main blocks:

- A Digital block provides the programming interface (memory-mapped registers) and includes the Timer unit and the Analog control unit.
- An Analog block provides the circuitry for the physical detection of the charger that includes the Bias generation, Voltage source, Current source, Current sink, and Voltage comparator circuitry. This block also contains necessary muxes between the source/sink circuits and the D- and D+ pins.

57.2.2 Features

- Compliant with the latest industry-standard specifications: *USB Battery Charging Specification, Revisions 1.1 and 1.2*
- Supports programmable timing parameters:
  - By default set to values required by the industry standards to allow easy configuration. You only need to set the clock frequency before enabling the module.
  - Updates of the standards are programmable.

57.3 Functional description

Several hardware components are involved in the sequence of detecting the presence of the charging port and type of the charging port. System software coordinates this activity. This collection of interacting hardware and software is called USB battery charging subsystem. The following figure shows USBDCD as a component of the subsystem.

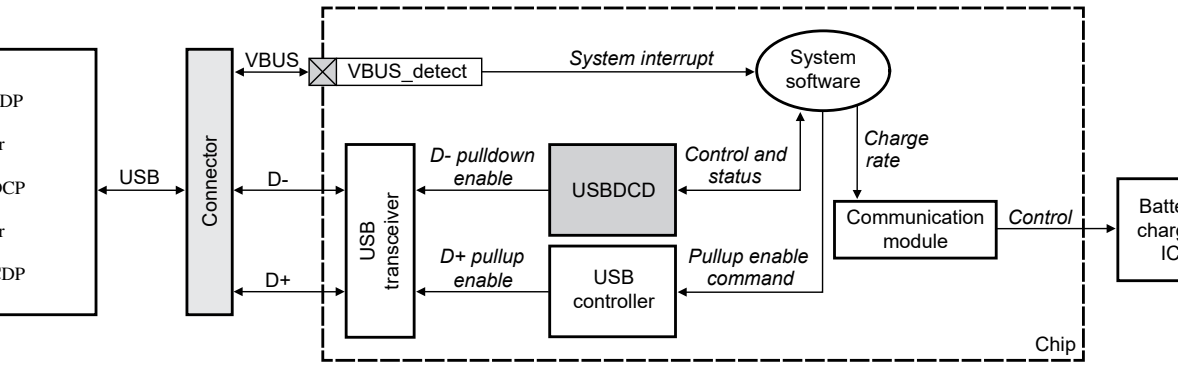


Figure 430. USB battery charging subsystem

The following table describes the components.

Table 909. USB battery charger subsystem components

<table>
<tr><th>Component</th><th>Description</th></tr>
<tr><td>Battery Charger IC</td><td>The external battery charger IC regulates the charge rate to the rechargeable battery. System software is responsible for communicating the appropriate charge rates.

<table><tr><th>Charger</th><th>Maximum current drawn</th></tr><tr><td>Standard Downstream Port (SDP)</td><td>up to 500 mA (I_{CFG_MAX})[1]</td></tr><tr><td>CDP</td><td>up to 1500 mA (I_{DEV_CHG})</td></tr><tr><td>DCP</td><td>up to 1500 mA (I_{DEV_CHG})</td></tr></table>
1. If a USB SDP host has suspended the USB device, the current drawn by the device has to follow the rules of the Dead Battery Provision in the USB Battery Charging Specification, Rev. 1.2.</td></tr>
<tr><td>Comm Module</td><td>A communications module on the device can be used to control the charge rate of the battery charger IC.</td></tr>
<tr><td>System software</td><td>Coordinates the detection activities of the subsystem.</td></tr>
<tr><td>USB Controller</td><td>The D+ pullup enable control signal plays a role during the charger type detection phase. System software must issue a command to the USB controller to assert this signal. After this pullup is enabled, the device is considered to be connected to the USB bus. The host then attempts to enumerate it.

NOTE
In general, the USB controller is used only for USB device applications when using USBDCD. For USB host applications, USBDCD must be disabled.</td></tr>
<tr><td>USB Transceiver</td><td>The USB transceiver contains the pullup resistor for the USB D+ signal and the pulldown resistors for the USB D+ and D– signals. The D+ pullup and the D– pulldown are both used during the charger detection sequence in BC1.1, but it is not used during charger detection in BC1.2. The USB transceiver also outputs the digital state of the D+ and D– signals from the USB bus.

The pullup and pulldown enable signals are controlled by other modules during the charger detection sequence in BC1.1: The D+ pullup enable is output from the USB controller and is under software control. USBDCD controls the D– pulldown enable.</td></tr>
<tr><td>USBDCD</td><td>Detects whether the device has been plugged into either an SDP, a CDP, or a DCP.</td></tr>
<tr><td>VBUS_detect</td><td>This interrupt pin connected to the USB VBUS signal detects when the device has been plugged into or unplugged from the USB bus. If the system requires waking up from a low-power mode on being plugged into the USB port, this interrupt should also be a low-power wakeup source. If this pin multiplexes other functions, such as GPIO, the pin can be configured as an interrupt so that the USB plug or unplug event can be detected.</td></tr>
</table>

57.3.1 Modes of operation

The operating modes of the USBDCD module are shown in the following table.

Table 910. Module modes and their conditions

Module mode	Description	Conditions when used
Enabled for Charger Detection	The module performs the charger detection sequence.	System software must enable the module only if all of the following conditions are true: • The system is using a rechargeable battery or is otherwise capable of being powered up by an SDP or a Charging Port. • The device is being used in a USB device application. • The device is attached to the USB cable.
Enabled for Signal Overrides	Module bias circuits are enabled for special signaling on DP/DM pins.	See SIGNAL_OVERRIDE[PS].
Disabled	The module is not active and is held in a low-power state.	System software must disable the module when either of the following conditions is true: • The charger detect sequence is complete. • The conditions for being enabled are not met.
Powered Off	The digital supply voltage dvdd is removed.	Low system performance requirements allow putting the device into a very low-power stop mode.

Operating mode transitions are shown in the following table.

Table 911. Entering and exiting module modes

Module mode	Entering	Exiting	Mode after exiting
Enabled	Set CONTROL[START].	Set CONTROL[SR]. [1]	Disabled
Disabled	Take *either* of the following actions: • Set CONTROL[SR].[1] • Reset the module. By default, the module is disabled.	Set CONTROL[START].	Enabled
Powered Off	Perform the following actions: 1. Put the device into very low-power stop mode. 2. Adjust the supply voltages.	Perform the following actions: 1. Restore the supply voltages. 2. Take the device out of very low-power stop mode.	Disabled

1. The effect of setting the CONTROL[SR] field is immediate; that is, the module is disabled even if the sequence has not completed.

57.3.2 The charger detection sequence

The following figure illustrates the charger detection sequence in a simplified timing diagram based on the *USB Battery Charging Specification, Rev. 1.2*.

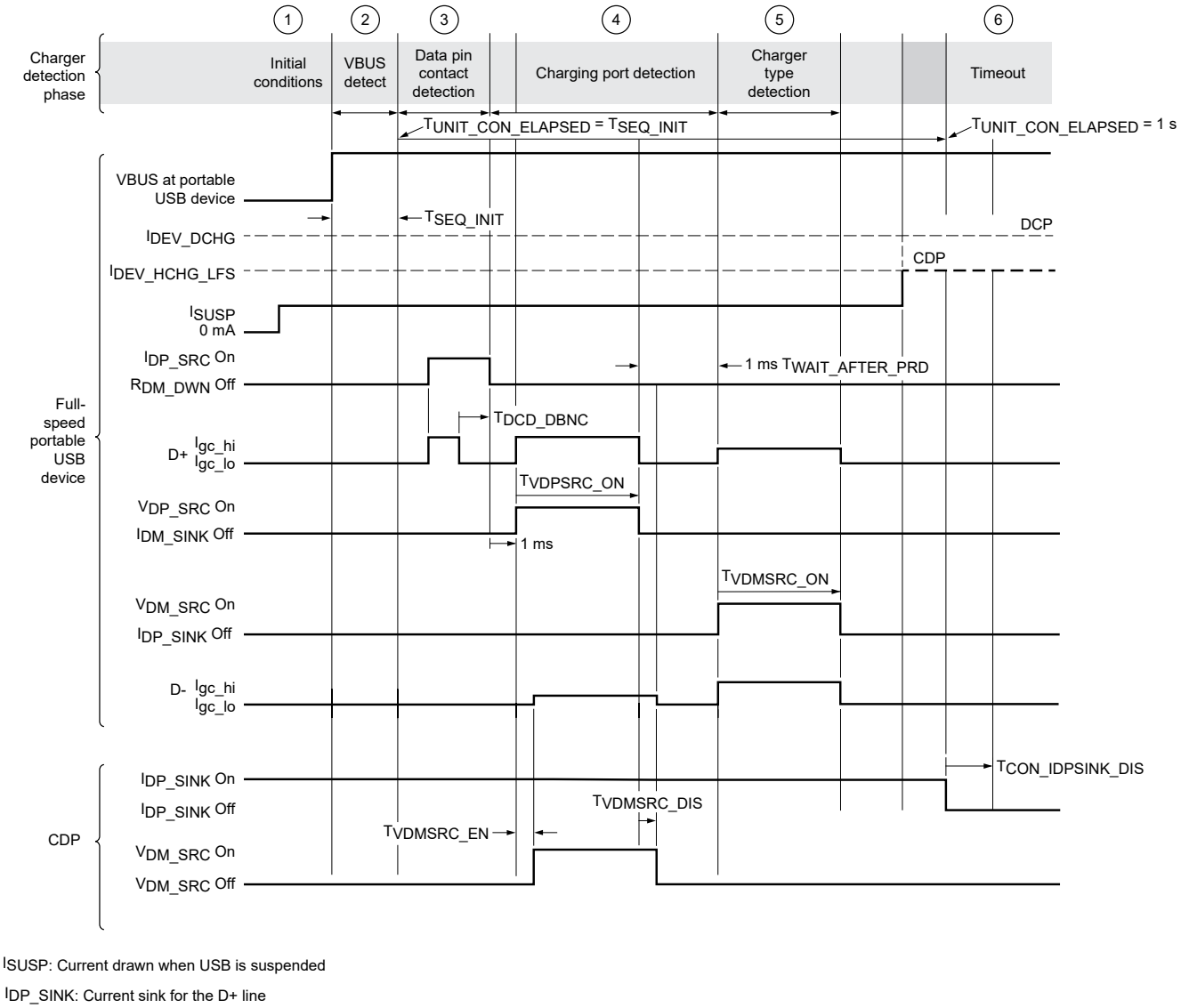


Figure 431. FS charger detection timing for BC1.2

Timing parameter values used in this module for BC1.2 are listed in the following table.

Table 912. Timing parameters for the charger detection sequence for BC1.2

Parameter	USB Battery Charging Spec	Module default	Module programmable range
T_{DCD_DBNC} [1]	10 ms min (no max)	10 ms	0– 1023 ms
T_{VDPSRC_ON} [1]	40 ms min (no max)	40 ms	0 –1023 ms
$T_{WAIT\_AFTER\_PRD}$	N/A	1 ms	0– 1023 ms
T_{VDMSRC_ON}	40 ms	40 ms	0 –1023 ms

Table continues on the next page...

Table 912. Timing parameters for the charger detection sequence for BC1.2 (continued)

Parameter	USB Battery Charging Spec	Module default	Module programmable range
T_{SEQ_INIT}	N/A	16 ms	0 –1023 ms
T_{UNIT_CON} [1]	1 s	N/A	N/A
T_{VDMSRC_EN} [1]	1– 20 ms	From the USB host	N/A
T_{VDMSRC_DIS} [1]	0 – 20 ms	From the USB host	N/A
$T_{CON\_IDPSINK\_DIS}$ [1]	0 – 10 ms	From the USB host	N/A

1. This parameter is defined by the *USB Battery Charging Specification, Rev. 1.2*.

The following figure illustrates the charger detection sequence in a simplified timing diagram based on the *USB Battery Charging Specification, Rev. 1.1*.

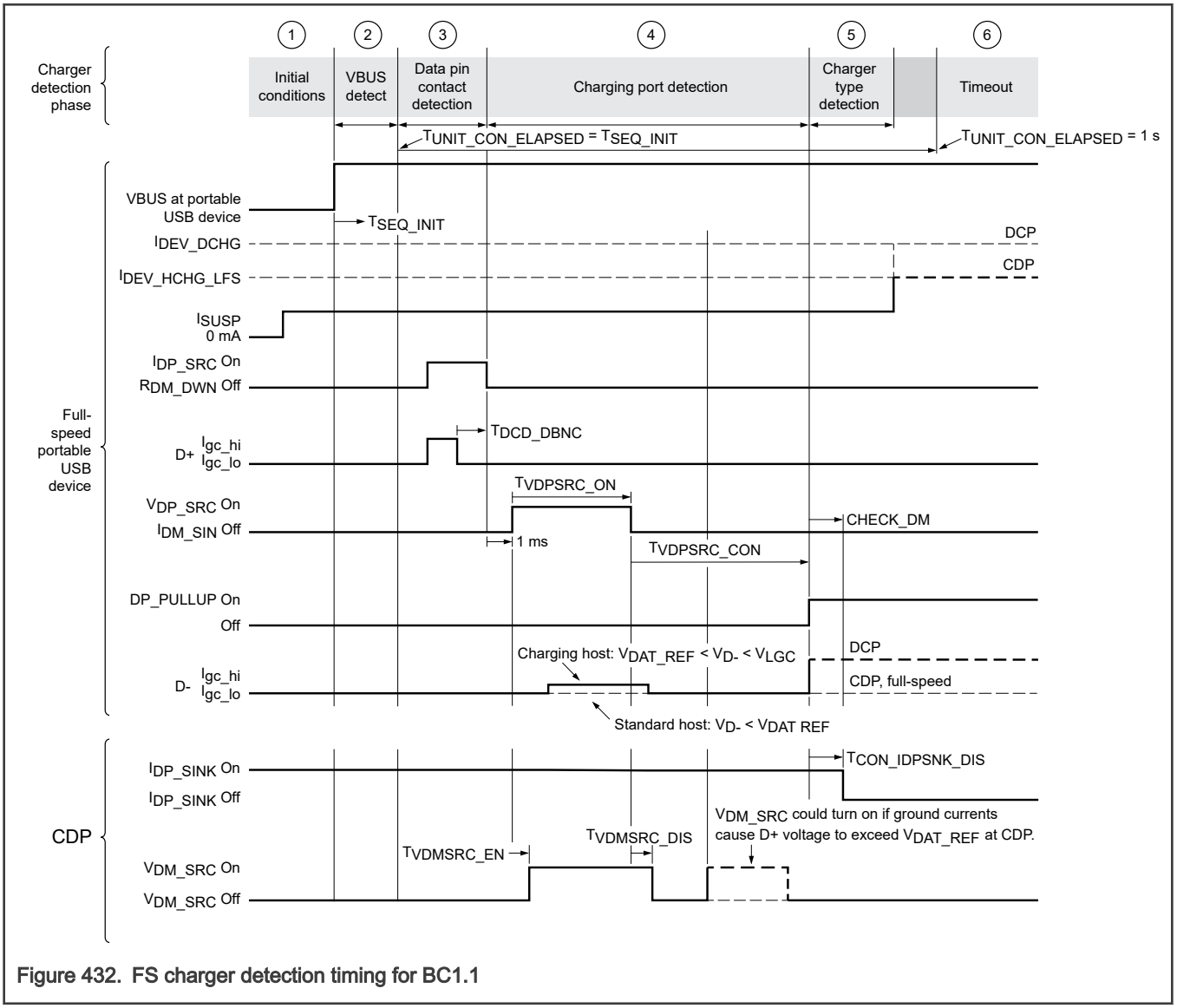


Figure 432. FS charger detection timing for BC1.1

Timing parameter values used in this module for BC1.1 are listed in the following table.

Table 913. Timing parameters for the charger detection sequence for BC1.1

Parameter	USB Battery Charging Spec	Module default	Module programmable range
T_{DCD_DBNC} [1]	10 ms min (no max)	10 ms	0– 1023 ms
T_{VDPSRC_ON} [1]	40 ms min (no max)	40 ms	0 –1023 ms
T_{VDPSRC_CON} [1]	40 ms min (no max)	40 ms	0 –1023 ms
CHECK_DM	N/A	1 ms	0– 15 ms
T_{SEQ_INIT}	N/A	16 ms	0 –1023 ms
T_{UNIT_CON} [1]	1 s	N/A	N/A
T_{VDMSRC_EN} [1]	1– 20 ms	From the USB host	N/A
T_{VDMSRC_DIS} [1]	0 –20 ms	From the USB host	N/A
$T_{CON\_IDPSINK\_DIS}$ [1]	0– 20 ms	From the USB host	N/A

1. This parameter is defined by the *USB Battery Charging Specification, Rev. 1.1*.

The following table provides an overview description of the charger detection sequence shown in the preceding figure.

Table 914. Overview of the charger detection sequence

Phase		Overview description	Full description
1	Initial conditions	Initial system conditions that need to be met before the detection sequence is initiated.	Initial system conditions
2	VBUS detection	System software detects contact of the VBUS signal with the system interrupt pin VBUS_detect.	VBUS contact detection
3	Data pin contact detection	The USBDCD module detects that the USB data pins D+ and D– have made contact with the USB port.	Data pin contact detection
4	Charging port detection	The USBDCD module detects if the port is an SDP or either type of charging port, that is CDP or DCP.	Charging port detection
5	Charger type detection	The USBDCD module detects the type of charging port, if applicable.	Charger type detection
6	Sequence timeout	The USBDCD module did not finish the detection sequence within the timeout interval. The sequence will continue until halted by software.	Charger detection sequence timeout

57.3.2.1 Initial system conditions

The USBDCD module is intended for use with USB device applications using a rechargeable battery. The module does not have support for interfacing with ACA or ACA-Dock equipment as defined in the *USB Battery Charging Specification, Revision 1.2*. It cannot be used with USB applications that are embedded host or OTG.

In addition, before the USBDCD module's charger detection sequence can be initiated, the system must be:

- Powered-up and in run mode. The USBDCD instantiation of this module for the High-Speed port does not directly depend on the VBUS voltage and can operate as long as the avdd33 supply is in a valid range.
- Recently plugged into a USB port.
- Drawing not more than 2.5 mA total system current from the USB bus, except as allowed by the *USB 2.0 Connect Timing Update ECN*.

Examples of allowable precursors to this set of initial conditions include:

- A powered-down device is subsequently powered-up upon being plugged into the USB bus.
- A device in a low-power mode subsequently enters run mode upon being plugged into the USB bus.

57.3.2.2 VBUS contact detection

Once the device is plugged into a USB port, the VBUS_detect system interrupt is triggered. System software must do the following to initialize the module and start the charger detection sequence:

1. Restore power if the module is powered-off.
2. Set CONTROL[SR] to initiate a software reset.
3. Configure the USBDCD module by programming the CLOCK register and the timing parameters as needed.
4. Set CONTROL[IE] to enable interrupts, or clear CONTROL[IE] if the software polling method is used.
5. Program CONTROL[BC12] based on which revision of the USB Battery Charging Specification is needed.
6. Set CONTROL[START] to start the charger detection sequence.

57.3.2.3 Data pin contact detection

The module must ensure that the data pins have made contact because the detection sequence depends upon the state of the USB D+ and D- signals. USB plugs and receptacles are designed such that when the plug is inserted into the receptacle, the power pins make contact before the data pins make contact. See the following figure.

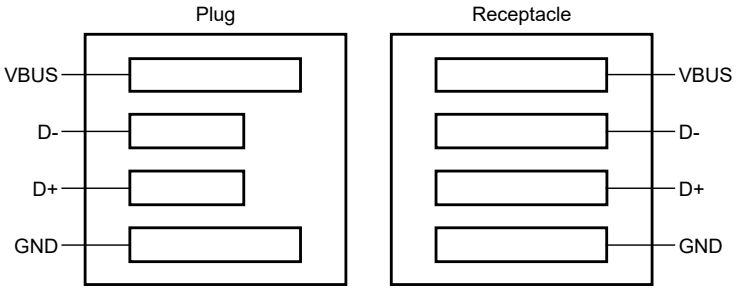


Figure 433. Relative pin positions in USB plugs and receptacles

As a result, when a portable USB device is attached to an upstream port, the portable USB device detects VBUS before the data pins have made contact. The time between power pins and data pins making contact depends on how fast the plug is inserted into the receptacle. Delays of several hundred milliseconds are possible.

57.3.2.3.1 Debouncing the data pin contact

When system software has initiated the charger detection sequence, as described in Initial system conditions, the USBDCD module turns on the I_{DP_SRC} current source and enables the R_{DM_DWN} pulldown resistor. If the data pins have not made contact, the D+ line remains high. After the data pins make contact, the D+ line goes low and debouncing begins.

After the D+ line goes low, the module continuously samples the D+ line over the duration of the T_{DCD_DBNC} debounce time interval. By default, T_{DCD_DBNC} is 10 ms, but it can be programmed in the TIMER0[TDCD_DBNC] field. See the description of TIMER0 for details.

When it has remained low for the entire interval, the debouncing is complete. However, if the D+ line returns high during the debounce interval, the module waits until the D+ line goes low again to restart the debouncing. This cycle repeats until either of the following happens:

- The data pin contact has been successfully debounced (see Success in detecting data pin contact (phase completion)).
- A timeout occurs (see Charger detection sequence timeout).

57.3.2.3.2 Success in detecting data pin contact (phase completion)

After successfully debouncing the D+ state, the module does the following:

- Updates the STATUS register to reflect phase completion (See Table 920 for field values.)
- Directly proceeds to the next step in the sequence: detection of a charging port (See Charging port detection.)

57.3.2.4 Charging port detection

After it detects that the data pins have made contact, the module waits for a fixed delay of 1 ms, and then attempts to detect whether it is plugged into a charging port. The module connects the following analog units to the USB D+ or D– lines during this phase:

- The voltage source V_{DP_SRC} connects to the D+ line
- The current sink I_{DM_SINK} connects to the D– line
- The voltage comparator connects to the USB D– line, comparing it to the voltage V_{DAT_REF}.

After a time of T_{VDPSRC_ON}, the module samples the D– line. The T_{VDPSRC_ON} parameter is programmable and defaults to 40 ms. After sampling the D– line, the module disconnects the voltage source, current sink, and comparator.

The next steps in the sequence depend on the voltage on the D– line as determined by the voltage comparator. See the following table.

Table 915. Sampling D– in the charging port detection phase

If the voltage on D- is...	Then...	See...
Below V_{DAT_REF}	The port is an SDP that does not support using charging currents above I_{CFG_MAX}.	SDP
Above V_{DAT_REF} but below V_{LGC}	The port is a charging port.	Charging port
Above V_{LGC}	This is an error condition.	Error in charging port detection

57.3.2.4.1 SDP

As part of the charger detection handshake with a standard USB host, the module does the following without waiting for the interval ($T_{WAIT\_AFTER\_PRD}$ or T_{VDPSRC_CON}) to elapse:

- Updates the STATUS register to reflect that an SDP is detected with SEQ_RES = 01b. See Table 920 for field values.
- Sets CONTROL[IF].

- Generates an interrupt if enabled in CONTROL[IE].

At this point, control is passed to system software via the interrupt. The rest of the sequence, which detects the type of charging port, is not applicable, so software must perform the following steps:

1. Read the STATUS register.
2. Set CONTROL[IACK] to acknowledge the interrupt.
3. Set CONTROL[SR] to issue a software reset to the module.
4. Disable the module.
5. Communicate the appropriate charge rate to the external battery charger IC; see Table 909.

57.3.2.4.2 Charging port

As part of the charger detection handshake with any type of USB host, the module waits until the interval ($T_{WAIT\_AFTER\_PRD}$ or T_{VDPSRC_CON}) has elapsed before it does the following:

For *USB Battery Charging Specification, Rev. 1.2*:

- Enables V_{DM_SRC}.
- Updates the STATUS register to reflect that a charging port is detected with SEQ_RES = 10b. See Table 920 for field values.
- At this point the detection sequence progresses to Charger type detection without needing software interaction.

For *USB Battery Charging Specification, Rev. 1.1*:

- Updates the STATUS register to reflect that a charging port is detected with SEQ_RES = 10b. See Table 920 for field values.
- Sets CONTROL[IF].
- Generates an interrupt if enabled in CONTROL[IE].
- At this point, control is passed to system software via the interrupt. Software must:
  1. Read the STATUS register.
  2. Set CONTROL[IACK] to acknowledge the interrupt.
  3. Issue a command to the USB controller to pullup the USB D+ line.
  4. Wait for the module to complete the final phase of the sequence. See Charger type detection.

57.3.2.4.3 Error in charging port detection

For this error condition, the module does the following:

- Updates the STATUS register to reflect the error with SEQ_RES = 00b. See Table 920 for field values.
- Sets CONTROL[IF].
- Generates an interrupt if enabled in CONTROL[IE].

Note that in this case the module does not wait for the interval ($T_{WAIT\_AFTER\_PRD}$ or T_{VDPSRC_CON}) to elapse.

At this point, control has been passed to system software via the interrupt. The rest of the sequence (detecting the type of charging port) is not applicable, so software should:

1. Read the STATUS register.
2. Set CONTROL[IACK] to acknowledge the interrupt.
3. Set CONTROL[SR] to issue a software reset to the module.
4. Disable the module.

57.3.2.5 Charger type detection

For *USB Battery Charging Specification, Rev. 1.2*:

After the USBDCD module enables the V_{DM_SRC}, the module starts the T_{VDMSRC_ON} timer counting down the time interval programmed into the TIMER2[T_{VDMSRC_ON}] field.

Once the T_{VDMSRC_ON} timer is elapsed, the module samples the USB D+ line to determine the type of charger. See the following table.

Table 916. Sampling D+ in the charger type detection phase (BC1.2)

If the voltage on D+ is...	Then...	See...
High (D+>V_{DAT_REF})	The port is a DCP. [1]	DCP
Low (D+<V_{DAT_REF})	The port is a CDP. [2]	CDP

1. In a DCP, the D+ and D– lines are shorted together through a small resistor.
2. In a CDP, the D+ and D– lines are not shorted.

For *USB Battery Charging Specification, Rev. 1.1*:

After software enables the D+ pullup resistor, the module is notified automatically (via internal signaling) to start the CHECK_DM timer counting down the time interval programmed in the TIMER2[CHECK_DM] field.

After the CHECK_DM time is elapsed, the module samples the USB D– line to determine the type of charger. See the following table.

Table 917. Sampling D– in the charger type detection phase (BC1.1)

If the voltage on D– is...	Then...	See...
High	The port is a DCP. [1]	DCP
Low	The port is a CDP. [2]	CDP

57.3.2.5.1 DCP

For a DCP, the module does the following:

- Updates the STATUS register to reflect that a dedicated charging port is detected with SEQ_RES = 11b. See Table 920 for field values.
- Sets CONTROL[IF].
- Generates an interrupt if enabled in CONTROL[IE] field.

The *USB Battery Charging Specification, Rev. 1.2* indicates that additionally if the detection sequence determines an attachment to a DCP, then the USB device should signal on the D+ pin by either enabling the D+ pullup resistor or enabling V_{DP_SRC}.

At this point, control has been passed to system software via the interrupt. Software should:

1. Read the STATUS register.
2. Disable the USB controller to prevent transitions on the USB D+ or D– lines from causing spurious interrupt or wakeup events to the system.
3. Set CONTROL[IACK] to acknowledge the interrupt.
4. Set CONTROL[SR] to issue a software reset to the module.
5. Disable the module.

6. Enable signaling on the D+ pin. This can be done by enabling the Transceiver D+ pullup resistor through configuration of the register fields in the USB controller instance for this USB port. For USBHSDCD instantiation of USBDCD, this signaling can alternatively be done by enabling V_{DP_SRC} through setting the SIGNAL_OVERRIDE[PS] field to a value of 010b.
7. Communicate the appropriate charge rate to the external battery charger IC; see Table 909.

57.3.2.5.2 CDP

For a CDP, the module does the following:

- Updates the STATUS register to reflect that a CDP is detected with SEQ_RES = 10b. See Table 920 for field values.
- Sets CONTROL[IF].
- Generates an interrupt if enabled in CONTROL[IE].

At this point, control is passed to system software via the interrupt. Software should:

1. Read the STATUS register.
2. Set CONTROL[IACK] to acknowledge the interrupt.
3. Set CONTROL[SR] to issue a software reset to the module.
4. Disable the module.
5. Communicate the appropriate charge rate to the external battery charger IC; see Table 909.

57.3.2.6 Charger detection sequence timeout

The maximum time allowed to connect according to the *USB Battery Charging Specification* is one second. If the Unit Connection Timer reaches the one second limit and the sequence is still running as indicated by the STATUS[ACTIVE] field, the module does the following:

- Updates the STATUS register to reflect that a timeout error has occurred. See Table 920 for field values.
- Sets CONTROL[IF].
- Generates an interrupt if enabled in CONTROL[IE].
- The detection sequence continues until explicitly halted by software setting CONTROL[SR].
- The Unit Connection Timer continues counting. See the description of TIMER0.

At this point, control is passed to the system software via the interrupt, which has two options: ignore the interrupt and allow more time for the sequence to complete, or halt the sequence. To halt the sequence, software must:

1. Read the STATUS register.
2. Set CONTROL[IACK] to acknowledge the interrupt.
3. Set CONTROL[SR] to issue a software reset to the module.
4. Disable the module.

This timeout function is also useful in case software does not realize that the USB device is unplugged from USB port during the charger detection sequence. If the interrupt occurs but the V_{BUS_DETECT} input is low, software can disable and reset the module.

System software might allow the sequence to run past the timeout interrupt under these conditions:

1. The USB Battery Charging Spec is amended to allow more time. In this case, software should poll TIMER0[$T_{UNITCON}$] periodically to track elapsed time after 1s; or
2. For debug purposes.

Note that the $T_{UNITCON}$ register field will stop incrementing when it reaches its maximum value so it will not roll over to zero and start counting up again.

57.3.2.7 Dead Battery Provision signaling

If the system needs to operate under the Dead Battery Provision of the *USB Battery Charging Specification, Revision 1.2*, the USBDCD module must be configured to signal V_{DP_SRC} on the USB_DP pin.

For information on Dead Battery Provision conditions and signaling, see the Dead or weak battery later in this chapter.

57.3.3 Clocking

The following table describes the clock sources for the USBDCD module. Refer to the chip's clocking chapter for clock setting, configuration, and gating information.

Note these clocks share same name with the USB PHY.

Table 918. USBDCD clocks

Clock name	Description
MODULE_CLK	Peripheral clock
MODULE_CLK_S	Peripheral access clock

57.3.4 Resets

There are two ways to reset various register contents in this module: hardware resets and a software reset.

57.3.4.1 Hardware resets

Hardware resets originate at the system or device level and propagate down to the individual module level. They include start up reset, low-voltage reset, and all other hardware reset sources.

Hardware resets cause the register contents to be restored to their default state as listed in the register descriptions.

57.3.4.2 Software reset

A software reset re-initializes the module's status information, but leaves configuration information unchanged. The software reset allows software to prepare the module without needing to reprogram the same configuration each time the USB device is plugged into a USB port.

Setting CONTROL[SR] initiates a software reset. The following table shows all register fields that are reset to their default values by a software reset.

Table 919. Software reset and register fields affected

Register	Fields affected	Fields not affected
CONTROL[1]	IF	IE, START
STATUS	All	None
CLOCK	None	All
TIMER*n*	TUNITCON	All other

1. CONTROL[SR] and CONTROL[IACK] are self-clearing.

A software reset also returns all internal logic, timers, and counters to their reset states. If the module is already active (STATUS[ACTIVE] = 1b), a software reset stops the sequence.

NOTE

Software must always initiate a software reset before starting the sequence to ensure the module is in a known state.

57.3.5 Interrupts

The USBDCD module has an interrupt to alert system software of certain events, which are listed in the following table. All events except the Phase Complete event for the Data Pin Detection phase can trigger an interrupt.

Table 920. Events triggering an interrupt by sequence phase

Sequence phase	Event	Event description	STATUS fields [1]	Phase description
Data Pin Detection	Phase Complete	The module has detected data pin contact. *No interrupt occurs*: CONTROL[IF] = 0b.	ERR = 0b SEQ_STAT = 01b SEQ_RES = 00b TO = 0b	VBUS contact detection
Charging Port Detection	Phase Complete	The module has completed the process of identifying if the USB port is a charging port or not. **NOTE:** An interrupt is triggered at completion of this phase for BC 1.1 operation whether an SDP or a Charging port is detected. For BC 1.2 operation, if an SDP is detected an interrupt is triggered but if a Charging port is detected the sequence goes directly to the Charger Type Detection phase.	ERR = 0b SEQ_STAT = 10b SEQ_RES = 01b or 10b TO = 0b	Charging port detection
	Error	The module cannot identify the type of port because the D– line is above the USB's VLGC threshold.	ERR = 1b SEQ_STAT = 10b SEQ_RES = 00b TO = 0b	Error in charging port detection
Charger Type Detection	Phase Complete	The module has completed the process of identifying the charger type detection. **NOTE:** The ERR flag always reads zero because no known error conditions are checked during this phase.	ERR = 0b SEQ_STAT = 11b SEQ_RES = 11b or 10b TO = 0b	Charger type detection
Sequence Timeout	Error	The timeout interval from the time the USB device attaches to a USB port until it connects has elapsed.	ERR = 1b SEQ_STAT = last value [2] SEQ_RES = last value [2] TO = 1b	Charger detection sequence timeout

1. See the description of the Status register for register information.
2. The SEQ_STAT and SEQ_RES fields retain the values held at the time of the timeout error.

57.3.5.1 Interrupt handling

Software reads which event caused the interrupt from the STATUS register during the interrupt service routine.

An interrupt is generated only if CONTROL[IE] is set. The CONTROL[IF] field is always set under interrupt conditions, even if CONTROL[IE] is cleared. In this case, software can poll CONTROL[IF] to determine if an interrupt condition is pending.

Writes to CONTROL[IF] are ignored. To reset CONTROL[IF], set CONTROL[IACK] to acknowledge the interrupt. Writing to CONTROL[IACK] when CONTROL[IF] is cleared has no effect.

57.4 External signals

This section describes the module signals. The following table shows a summary of module signals that interface with the pins of the device.

Table 921. Signal descriptions

Signal	Description	I/O
USB_DM	USB D– analog data signal. The analog block interfaces directly to the D– signal on the USB bus.	I/O
USB_DP	USB D+ analog data signal. The analog block interfaces directly to the D+ signal on the USB bus.	I/O
avdd33 [1]	3.3 V regulated analog supply	POWER
vss_bulk	Digital/Analog ground	POWER
dvdd	Core voltage digital supply	POWER

1. Voltage must be 3.3 V ± 10% for full functionality of the module. That is, the charger detection function does not work when this voltage is below 3.0 V, and the CONTROL[START] field should not be set.

NOTE

The transceiver module also interfaces to the USB_DM and USB_DP signals. Both modules and the USB host/hub use these signals as bidirectional, tristate signals.

Information about the signal integrity aspects of the lines including shielding, isolated return paths, input or output impedance, packaging, suggested external components, ESD, and other protections can be found in the USB 2.0 Specification and in Application information.

57.5 Initialization

This module is designed for minimal configuration while retaining significant programmability. To initialize USBDCD:

1. Initialize the CLOCK register to the actual system clock frequency, unless the default value already matches the system requirements.
2. Reset ANACTRL[DEV_PULLDOWN] to 0b when this module is in use for the proper operation of the USBHSDCD instantiation of USBDCD.

   For more information, see the "USB PHY Analog Control Register (ANACTRL)" section of the "Universal Serial Bus 2.0 Integrated PHY (USB-PHY)" chapter.
3. Do not modify other registers, as they default to the values that comply with the USB Battery Charging Specification.
4. Configure timing parameters to obtain flexibility as per the requirements of a specific system.

All module configuration must occur *before* initiating the charger detection sequence. Configuration changes made *after* setting CONTROL[START] result in undefined behavior.

57.6 Application information

This section provides application information.

57.6.1 External pullups

Any external pullups applied to the USB D+ or D- data lines must be capable of being disabled to prevent incorrect pullup values or incorrect operation of the USB subsystem.

57.6.2 Dead or weak battery

According to the *USB Battery Charging Specification, Revision 1.2*, a USB device with a dead or weak battery that is attached to an SDP can draw charging current from VBUS without connecting for several minutes, until the battery is charged to the point that the USB device can connect. The conditions for this operation are referred to as the Dead Battery Provision.

The *USB Battery Charging Specification, Revision 1.2* limits this charging condition to 100 mA for a duration of 45 minutes. The later *USB 2.0 Connect Timing Update ECN* modifies this charging condition to 500 mA for a duration of 2 minutes. Customers designing systems to take advantage of the Dead Battery Provision should study both specifications closely.

The USBDCD module is compatible with systems that do not check the strength of the battery. Therefore, this module assumes that the battery is good, so the USB device must immediately connect to the USB bus by pulling the USB_DP pin high after the USBDCD module has determined that the device is attached to an SDP or CDP. (By definition, a DCP does not support connection but still requires a signaling event, see DCP.)

The module is also compatible with systems that have other circuitry to check the strength of the battery. In these systems, if it is known that the battery is weak or dead, software can delay connecting to the USB while charging using the Dead Battery Provision. Once the battery is charged to the good battery threshold, software must then connect to the USB host by pulling the USB_DP pin high.

While using the Dead Battery Provision, the USB_DP pin must signal by enabling V_{DP_SRC}. On this product that signaling can be done most simply by setting the SIGNAL_OVERRIDE[PS] field to value 10b.

57.6.3 Handling unplug events

If the device is unplugged from the USB bus during the charger detection sequence, the contents of the STATUS register must be ignored and the USBDCD module must get a Software Reset, as described in Software reset.

57.7 USBDCD register descriptions

57.7.1 USBDCD memory map

USBHSDCD1 base address: 42CA_0800h

USBHSDCD2 base address: 42CB_0800h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CONTROL)	32	RW	0001_0000h
4h	Clock (CLOCK)	32	RW	0000_00C1h
8h	Status (STATUS)	32	R	0000_0000h
Ch	Signal Override (SIGNAL_OVERRIDE)	32	RW	0000_0000h
10h	TIMER0 (TIMER0)	32	RW	0010_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
14h	TIMER1 (TIMER1)	32	RW	000A_0028h
18h	TIMER2_BC11 (TIMER2_BC11)	32	RW	0028_0001h
18h	TIMER2_BC12 (TIMER2_BC12)	32	RW	0001_0028h

57.7.2 Control (CONTROL)

Offset

Register	Offset
CONTROL	0h

Function

Controls various global functions, such as software reset, initiating the charger detection sequence, BC12 compatibility, and interrupt enablement.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						0	0	0						BC12	IE
W							SR	START								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved							IF	0							0
W																IACK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25 SR	Software Reset Determines whether a software reset is performed. 0b - Do not perform a software reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Perform a software reset.
24 START	Start Change Detection Sequence Determines whether the charger detection sequence is initiated. 0b - Do not start the sequence. Writes of this value have no effect. 1b - Initiate the charger detection sequence. If the sequence is already running, writes of this value have no effect.
23-18 —	Reserved
17 BC12	Battery Charging Revision 1.2 Compatibility BC1.2 compatibility. This field cannot be changed after start detection. 0b - Compatible with BC1.1 (default) 1b - Compatible with BC1.2
16 IE	Interrupt Enable Enables/disables interrupts to the system. 0b - Disable interrupts to the system. 1b - Enable interrupts to the system.
15-9 —	Reserved
8 IF	Interrupt Flag Determines whether an interrupt is pending. 0b - No interrupt is pending. 1b - An interrupt is pending.
7-1 —	Reserved
0 IACK	Interrupt Acknowledge Determines whether the interrupt is cleared. 0b - Do not clear the interrupt. 1b - Clear the IF field (interrupt flag).

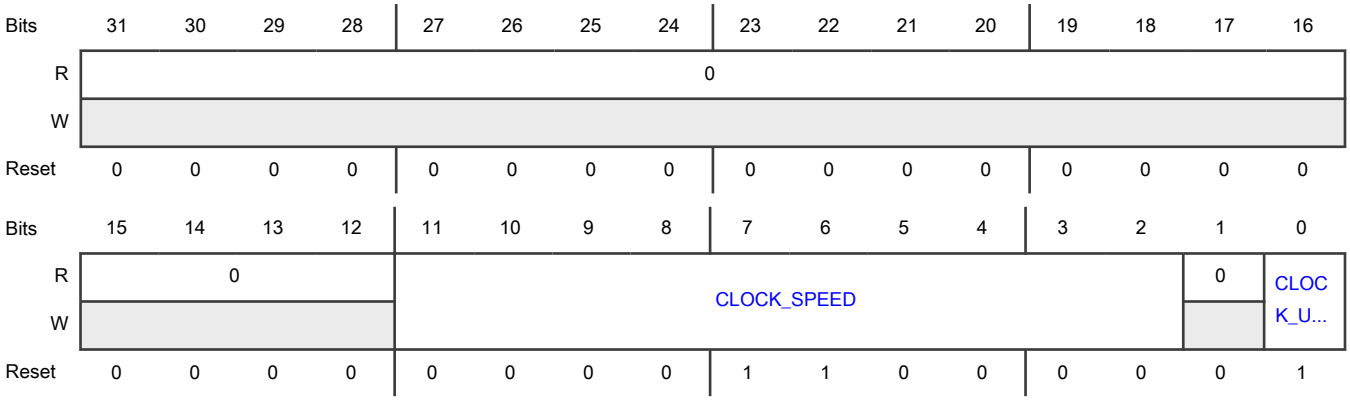
57.7.3 Clock (CLOCK)

Offset

Register	Offset
CLOCK	4h

Function
Controls the clock speed.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-2 CLOCK_SPEED	Numerical Value of Clock Speed in Binary The unit of measure is programmed in CLOCK_UNIT. The valid range is from 1 to 1023 when the clock unit is MHz and 4 to 1023 when the clock unit is kHz. Examples with CLOCK_UNIT = 1b: • For 48 MHz: 00110000b (48) • For 24 MHz: 00011000b (24) Examples with CLOCK_UNIT = 0b: • For 100 kHz: 01100100b (100) • For 500 kHz: 000111110100b (500)
1 —	Reserved
0	Unit of Measurement Encoding for Clock Speed

Table continues on the next page...

Table continued from the previous page...

Field	Function
CLOCK_UNIT	Specifies the unit of measure for the clock speed. 0b - kHz Speed (between 4 kHz and 1023 kHz) 1b - MHz Speed (between 1 MHz and 1023 MHz)

57.7.4 Status (STATUS)

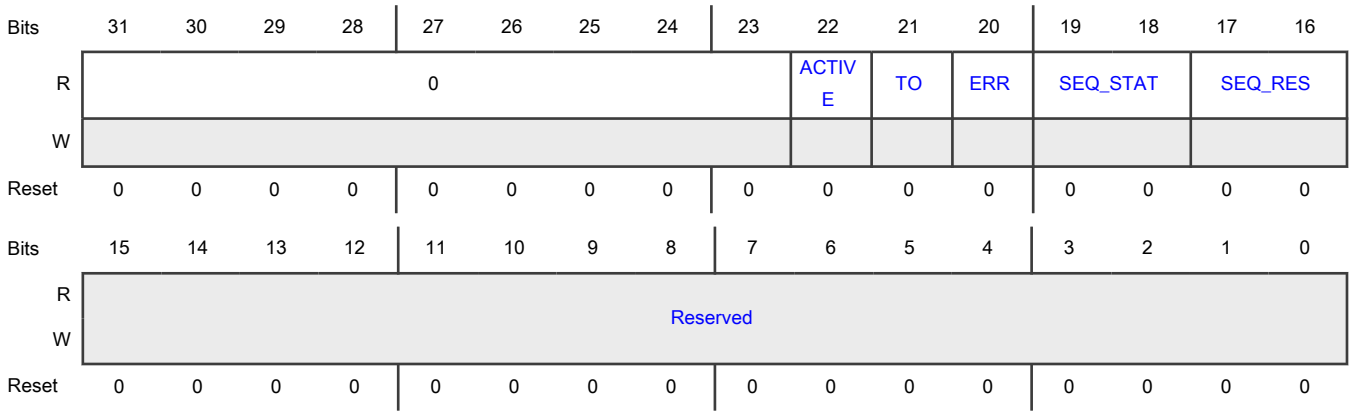
Offset

Register	Offset
STATUS	8h

Function

Stores the current state of the module for system software monitoring.

Diagram



Fields

Field	Function
31-23 —	Reserved
22 ACTIVE	Active Status Indicator Indicates whether the sequence is running. 0b - The sequence is not running. 1b - The sequence is running.
21	Timeout Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
TO	Indicates whether the detection sequence is passed the timeout threshold. 0b - The detection sequence is not running for over 1 s. 1b - It is over 1 s since the data pin contact was detected and debounced.
20 ERR	Error Flag Indicates whether there is an error in the detection sequence. 0b - No sequence errors. 1b - Error in the detection sequence. See STATUS[SEQ_STAT]to determine the phase in which the error occurred.
19-18 SEQ_STAT	Charger Detection Sequence Status Indicates the status of the charger detection sequence. 00b - The module is either not enabled, or the module is enabled but the data pins have not yet been detected. 01b - Data pin contact detection is complete. 10b - Charging port detection is complete. 11b - Charger type detection is complete.
17-16 SEQ_RES	Charger Detection Sequence Results Reports how the charger detection is attached. 00b - No results to report. 01b - Attached to an SDP. Must comply with USB 2.0 by drawing only 2.5 mA (max) until connected. 10b - Attached to a charging port. The exact meaning depends on the STATUS[SEQ_STAT] field (value 0: Attached to either a CDP or a DCP. The charger type detection has not completed. value 1: Attached to a CDP. The charger type detection has completed.) 11b - Attached to a DCP.
15-0 —	Reserved **NOTE** Fields do not always read as 0.

57.7.5 Signal Override (SIGNAL_OVERRIDE)

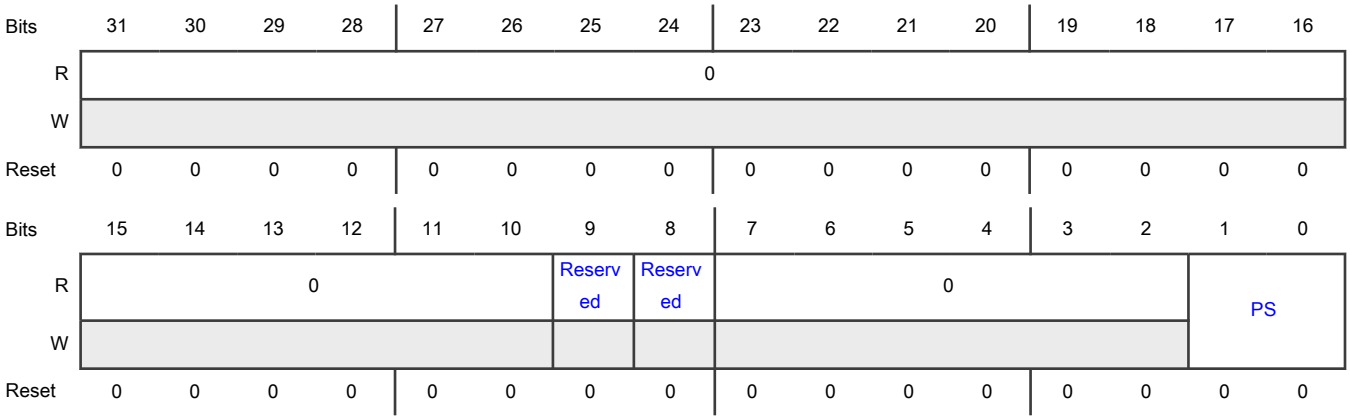
Offset

Register	Offset
SIGNAL_OVERRIDE	Ch

Function

Provides a way for the customer to enable signaling required by the USB BC Rev. 1.2 Specification after the battery charger detection sequences is completed.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved Reserved, not for customer use. The value of this field may change during module operation.
8 —	Reserved Reserved, not for customer use. The value of this field may change during module operation.
7-2 —	Reserved
1-0 PS	Phase Selection Enables specified voltage and current source circuits on the USB_DP and USB_DM pins. Use of the override value below enables the charger detection bias circuits without needing to set CONTROL[START]. Customers may set this field to 10b for required signaling if attached to a Dedicated Charging Port, or during operation under the Dead Battery Provision. 00b - No overrides. Field must remain at this value during normal USB data communication to prevent unexpected conditions on USB_DP and USB_DM pins. (Default) 01b - Reserved, not for customer use. 10b - Enables VDP_SRC voltage source for the USB_DP pin and IDM_SINK current source for the USB_DM pin. 11b - Reserved, not for customer use.

57.7.6 TIMER0 (TIMER0)

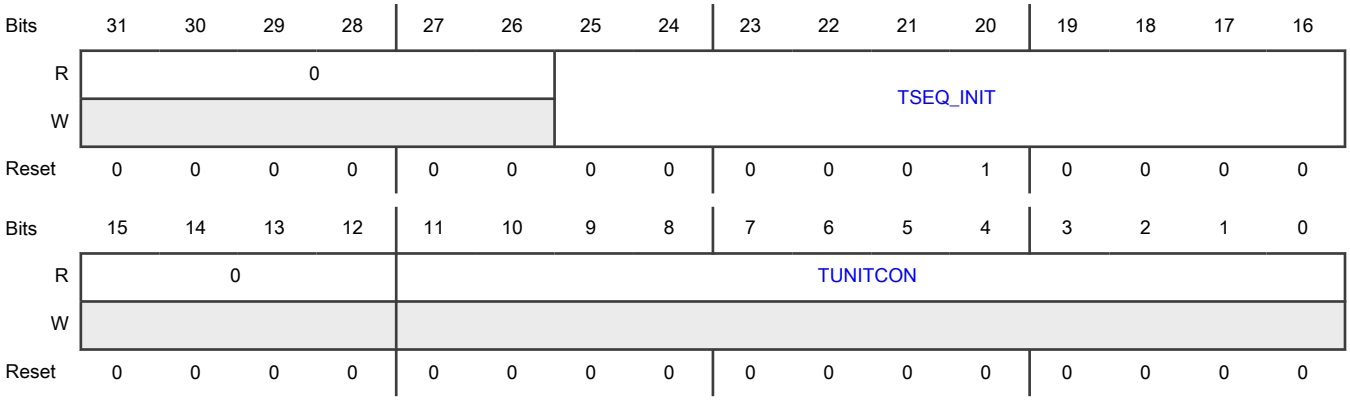
Offset

Register	Offset
TIMER0	10h

Function

Represents the system latency in ms using the TSEQ_INIT field.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 TSEQ_INIT	Sequence Initiation Time TSEQ_INIT represents the system latency (in ms) measured from the time VBUS goes active to the time system software initiates the charger detection sequence in the USBDCD module. When software sets CONTROL[START], the Unit Connection Timer (TUNITCON) is initialized with the value of TSEQ_INIT. Valid values are 0-1023, but the USB Battery Charging Specification requires the entire sequence, including TSEQ_INIT, to be completed in 1 s or less. 00_0000_0000b-11_1111_1111b - 0 ms - 1023 ms
15-12 —	Reserved
11-0 TUNITCON	Unit Connection Timer Elapse (in ms) Displays the amount of elapsed time since the event of setting CONTROL[START] plus the value of TSEQ_INIT. The timer is automatically initialized with the value of TSEQ_INIT before starting to count.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This timer enables compliance with the maximum time allowed to connect T_{UNIT_CON} under the USB Battery Charging Specification. If the timer reaches the one second limit, the module triggers an interrupt and sets the error flag STATUS[ERR]. The timer continues counting throughout the charger detection sequence, even when control has been passed to software. As long as the module is active, the timer continues to count until it reaches the maximum value of FFFh (4095 ms). The timer does not roll over to zero. A software reset clears the timer.

57.7.7 TIMER1 (TIMER1)

Offset

Register	Offset
TIMER1	14h

Function

Contains timing parameters.

NOTE

Register values can be written that are not compliant with the USB Battery Charging Specification, so care should be taken when overwriting the default values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						TDCD_DBNC									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						TVDPSRC_ON									
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0

Fields

Field	Function
31-26 —	Reserved
25-16	Time Period to Debounce D+ Signal

Table continues on the next page...

Table continued from the previous page...

Field	Function
TDCD_DBNC	Sets the time period (ms) to debounce the D+ signal during the data pin contact detection phase. See Debouncing the data pin contact. Valid values are 1-1023, but the USB Battery Charging Specification requires a minimum value of 10 ms. 00_0000_0001b-11_1111_1111b - 1 ms - 1023 ms
15-10 —	Reserved
9-0 TVDPSRC_ON	Time Period Comparator Enabled This timing parameter is used after detection of the data pin. See Charging port detection. Valid values are 1-1023, but the USB Battery Charging Specification requires a minimum value of 40 ms. 00_0000_0001b-11_1111_1111b - 1 ms - 1023 ms

57.7.8 TIMER2_BC11 (TIMER2_BC11)

Offset

Register	Offset
TIMER2_BC11	18h

Function

TIMER2_BC11 contains timing parameters for *USB Battery Charging Specification, Rev 1.1.*

NOTE

Register values can be written that are not compliant with the USB Battery Charging Specification, so care should be taken when overwriting the default values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								TVDPSRC_CON							
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												CHECK_DM			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-26 —	Reserved
25-16 TVDPSRC_CON	Time Period Before Enabling D+ Pullup Sets the time period (ms) that the module waits after charging port detection before system software must enable the D+ pullup to connect to the USB host. Valid values are 1-1023, but the USB Battery Charging Specification requires a minimum value of 40 ms. 00_0000_0001b-11_1111_1111b - 1 ms - 1023 ms
15-4 —	Reserved
3-0 CHECK_DM	Time Before Check of D- Line Sets the amount of time (in ms) that the module waits after the device connects to the USB bus until checking the state of the D- line to determine the type of charging port. See Charger type detection. Valid values are 1-15 ms. 0001b-1111b - 1 ms - 15 ms

57.7.9 TIMER2_BC12 (TIMER2_BC12)

Offset

Register	Offset
TIMER2_BC12	18h

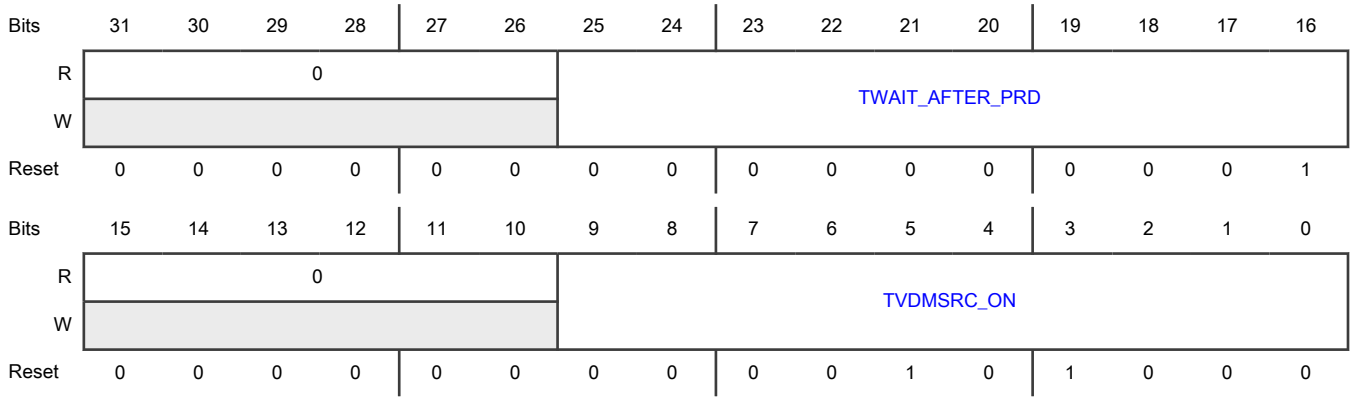
Function

Contains timing parameters for *USB Battery Charging Specification, Rev 1.2.*

NOTE

Register values can be written that are not compliant with the USB Battery Charging Specification, so care should be taken when overwriting the default values.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 TWAIT_AFTER_PRD	TWAIT_AFTER_PRD Sets the amount of time (in ms) that the module waits after primary detection before start to secondary detection. Valid values are 1-1023ms. 00_0000_0001b-11_1111_1111b - 1 ms - 1023 ms
15-10 —	Reserved
9-0 TVDMSRC_ON	TVDMSRC_ON Sets the amount of time (in ms) that the module enables the V_{DM_SRC}. Valid values are 0-40ms. 00_0000_0000b-00_0010_1000b - 0 ms - 40 ms

Chapter 58
Universal Serial Bus 2.0 PHY (USB-PHY)

58.1 Chip-specific USBPHY information

Table 922. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

58.2 Overview

USBPHY interfaces with the USB host and device system at a low-speed (LS) rate of 1.5 Mbit/s, full-speed (FS) rate of 12 Mbit/s, or USB 2.0 high-speed (HS) rate of 480 Mbit/s. The following sections describe the external and internal interfaces, major blocks, and programmable registers that comprise the integrated USBPHY.

58.2.1 Block diagram

The integrated PHY provides a standard USB transceiver macrocell interface (UTMI). The DP and DN pins (as shown in the following figure) connect directly to a USB connector.

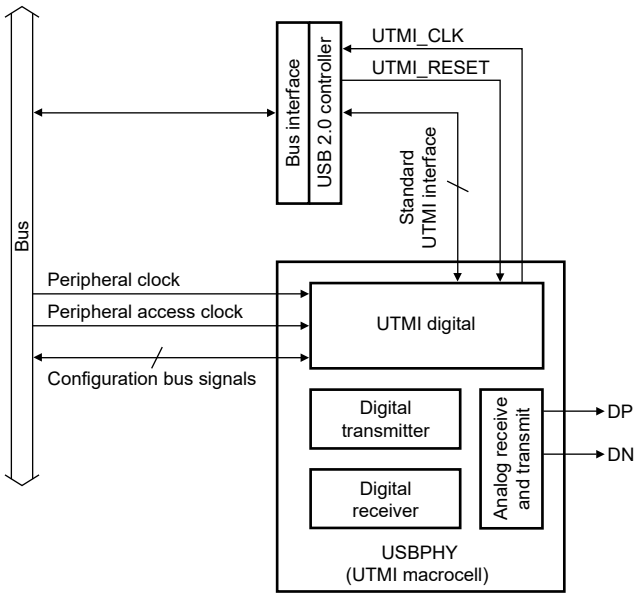


Figure 434. Block diagram

58.2.2 Features

- Supports HS and FS in Device mode
- Supports HS, FS, and LS in Host mode
- Complies with the UTMI+ standard, having a 16-bit data interface to the USB controller
- Provides an integrated 480 MHz PLL
- Provides an integrated USBDCD (see the "USB Device Charger Detection Module (USBDCD)" chapter) with features of *USB Battery Charging Specification, Revision 1.2*

58.3 Functional description

This section covers the USBPHY core components.

- The digital portions of USBPHY include the UTMI, digital transmitter, digital receiver, and programmable registers.
- The analog transceiver comprises an analog receiver and analog transmitter, as shown in Figure 435.

58.3.1 UTMI

The UTMI block handles the LineState bits, reset buffering, suspend distribution, transceiver speed selection, transceiver termination selection, and clock generation logic for the clocks derived from the PHY's USB PLL.

The PLL in USBPHY supplies a 480 MHz clock to the UTMI digital block that divides the clock to generate the 30 MHz clock (UTMI_CLK) used in the interface.

58.3.2 Digital transmitter

The digital transmitter receives a 16-bit transmit data from the USB controller and handles the TX_VALID, TX_VALIDH, and TX_READY handshake. Additionally, it contains the transmit serializer that converts 16-bit parallel words, at a frequency of 30 MHz, to a single bitstream at 480 Mbit/s for HS, 12 Mbit/s for FS, or 1.5 Mbit/s for LS. The digital transmitter does this while implementing the bit-stuffing algorithm and NRZI encoder that you use to remove the DC component from the serial bitstream. The output of this encoder is sent to the LS, FS, or HS drivers in the analog transceiver section's transmitter block.

58.3.3 Digital receiver

The digital receiver receives the raw serial bitstream from the LS differential transceiver, FS differential transceiver, and 480 MHz sampled data from the HS differential transceiver. As the phase of the USB host transmitter shifts relative to the local PLL, the receiver section's HS DLL tracks these changes to give an appropriate sample of the incoming 480 Mbit/s bitstream. Because this sample point shifts relative to the PLL phase that the digital logic uses, a rate-matching elastic buffer is provided to cross this clock domain boundary.

After the bitstream is in the local clock domain, an NRZI decoder and a bit unstuffer restore the original payload data bitstream and pass it to a deserializer and holding register. The RX state machine handles the RX_READY and RX_VALIDH signals, and a handshake with the USB controller. The handshake is not interlocked, with no RX_READY signal coming from the controller, which must take each 16-bit value that the PHY presents. The RX state machine provides an RX_ACTIVE signal to the controller that indicates whether it is inside a valid packet (synchronization detected, for example).

58.3.4 Analog receiver

The following figure illustrates the analog receiver that comprises differential receivers—two single-ended ones, a 9X, and a 480 MHz HS data sampling module.

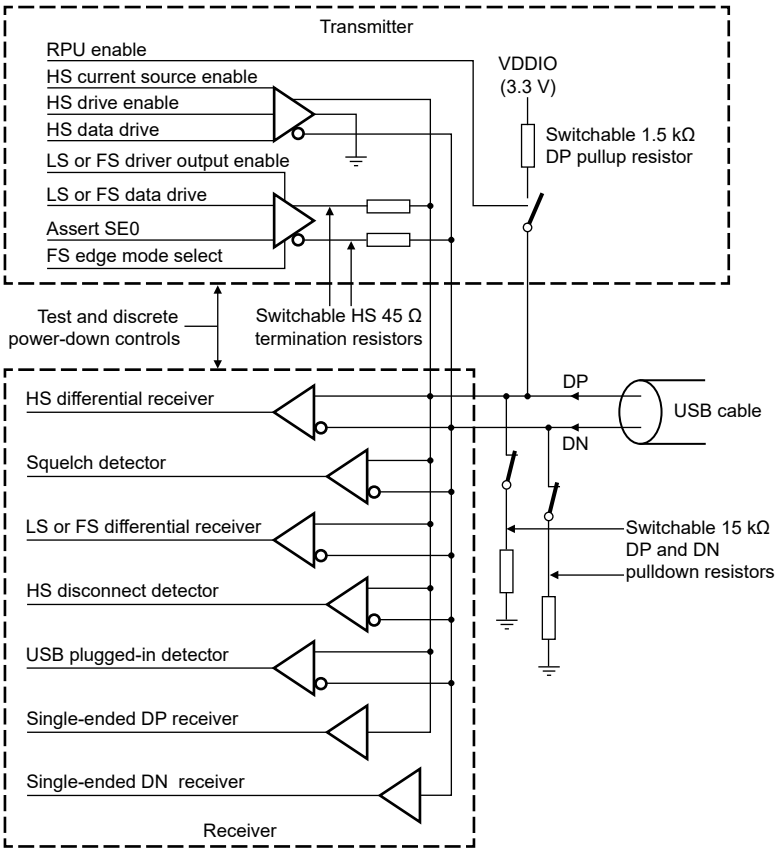


Figure 435. Analog transceiver block diagram

58.3.4.1 HS differential receiver

The HS differential receiver is both a differential analog receiver and threshold comparator. Its output is high (1b) if the differential signal is greater than a 0 V threshold. Otherwise, its output is low (0b). The purpose of the HS differential receiver is to discriminate the ± 400 mV differential voltage resulting from the HS drivers' current flow into the dual 45 Ω terminations found on each pin of the differential pair. The envelope or squelch detector (see Squelch detector) ensures that the differential signal has sufficient magnitude to be valid. The HS differential receiver tolerates up to 500 mV of common mode offset.

58.3.4.2 Squelch detector

The squelch detector is a differential analog receiver and threshold comparator. Its output is high (1b) if the differential magnitude is less than a nominal 100 mV threshold. Otherwise, its output is low (0b). The purpose of the squelch detector is to invalidate the HS differential receiver if the incoming signal is too low for appropriate reception.

58.3.4.3 LS or FS differential receiver

The LS or FS differential receiver is both a differential analog receiver and threshold comparator. The crossover voltage lies between 1.3 V and 2.0 V. Its output is 1 if the DP line is above the crossover point and the DN line is below the crossover point. The digital receiver section decodes the receiver data into the J or K state, according to the speed.

58.3.4.4 HS disconnect detector

The HS disconnect detector is a differential analog receiver and threshold comparator. Its output is high when the differential magnitude is greater than a nominal 575 mV threshold. Otherwise, its output is low.

58.3.4.5 USB plugged-in detector

When operating in Device mode, USBPHY provides the following methods for the local USB device to detect the attachment of a cable to a remote USB host or hub. You must deploy only one of these methods at a time when waiting to identify the cable attachment event, and disable all methods during USB data communication:

- USB plugged-in detector: This method looks for both DP and DN to be high. A large pair of on-chip pullup resistors (200 kΩ) hold both the DP and DN pins high when the USB cable is not attached. The USB plugged-in detector signals a 0 in this case. When operating in Device mode, the upstream port in the host or hub interface contains a 15 kΩ pulldown resistor that could easily override the 200 kΩ pullup resistor. When plugged in, at least one signal in the pair is low, which forces the plugged-in detector's output to become high.

  CTRL[ENDEVPLUGINDETECT] controls the USB plugged-in detector, and you can observe the results through STATUS[DEVPLUGIN_STATUS].

- Data contact detect: This method functions according to *USB Battery Charging Specification, Revision 1.2*. See the "USB Device Charger Detection (USBDCD)" chapter that describes the usage of this method.

58.3.4.6 Single-ended DP receiver

The single-ended DP receiver output is high whenever the DP input is above its nominal 1.8 V threshold.

58.3.4.7 Single-ended DN receiver

The single-ended DN receiver output is high whenever the DN input is above its nominal 1.8 V threshold.

58.3.5 Analog transmitter

The analog transmitter comprises two differential drivers: one for HS signaling and another for FS signaling. It also contains the switchable 1.5 kΩ pullup resistor (see Figure 436).

58.3.5.1 Switchable HS 45 Ω termination resistors

The HS current mode differential signaling requires a 90 Ω differential termination at each end of the USB cable. This results from switching in 45 Ω terminating resistors from each signal line to ground at each end of the cable.

Because each signal is terminated in parallel with 45 Ω at each end, each driver sees a 22.5 Ω load. This load impedance is too low for FS signaling levels, generating the need for switchable HS terminating resistors. Switchable trimming resistors are provided to tune the actual termination resistance for each device, as shown in Figure 436.

TX[TXCAL45DP], for example, allows one of the trimming resistor values to be placed in parallel with the 45 Ω terminator on the DP signal.

58.3.5.2 LS or FS differential driver

The LS or FS differential drivers are essentially a pair of low-impedance pullup and pulldown devices that are switched in a differential mode for most LS or FS signaling. One output is driven high while the other output is driven low to signal the "J" state or the "K" state. Both outputs are driven low for the LS or FS "SE0" state.

58.3.5.3 HS differential driver

The HS differential driver receives a 17.78 mA current from the constant current source (IREF) and essentially steers it down either the DP signal or DN signal, or alternatively to the ground. This current produces an approximately 400 mV drop across the 22.5 Ω termination that the driver sees when it is steered onto one of the signal lines. The approximately 17.78 mA current source is referenced back to the integrated voltage-band-gap (Vbg) circuit.

58.3.5.4 Switchable 1.5 kΩ DP pullup resistor

This chip contains a switchable 1.5 kΩ pullup resistor on the DP signal. You switch the register on to indicate to the host or hub controller that an FS-capable device is on the USB cable, powered on, and ready. You switch this resistor off at POR so that the host does not recognize a USB device until the processor software enables the announcement of an FS device.

58.3.5.5 Switchable 15 kΩ DP and DN pulldown resistors

This chip contains switchable 15 kΩ pulldown resistors on both DP and DN signals. You enable them in Host mode to indicate to the device controller that a host is present.

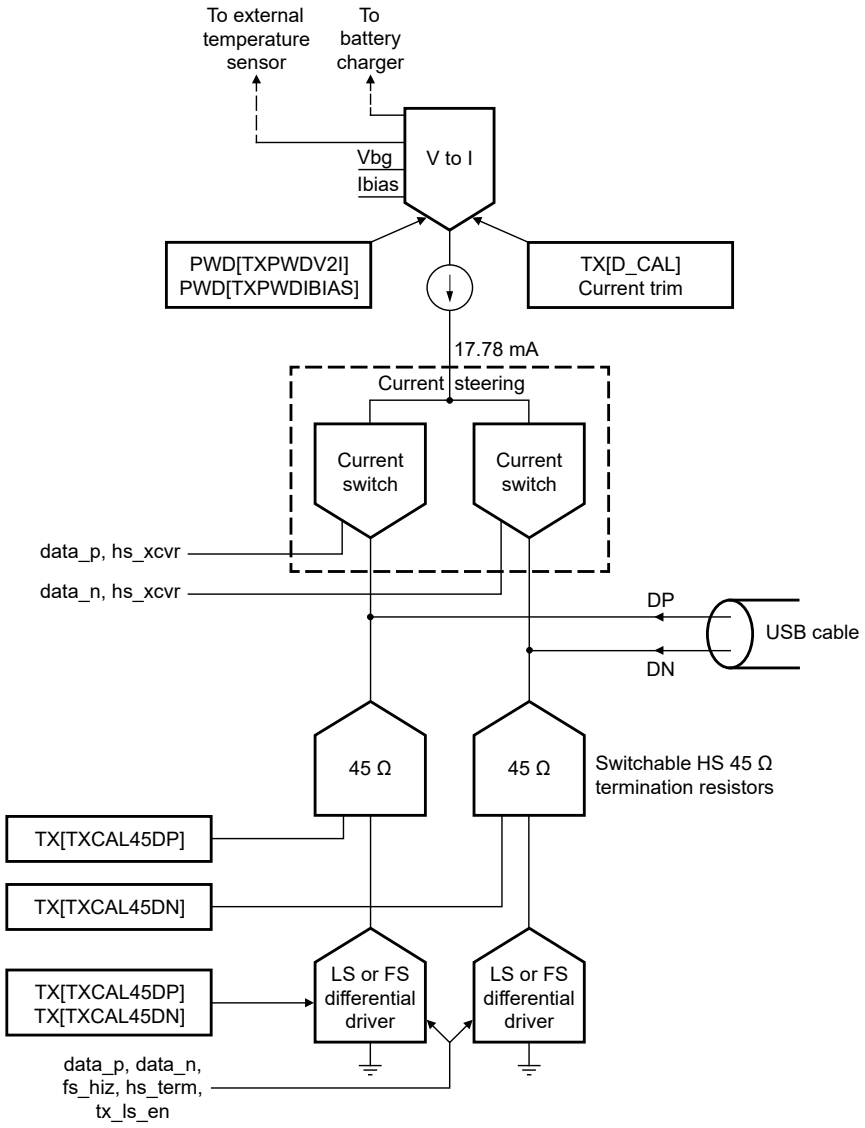


Figure 436. USBPHY transmitter block diagram

58.3.6 Low-Power mode

USBPHY includes a low-power mode (Suspend mode) to save power consumption.

When the USB bus is idle, you can place the transceiver in Low-Power mode (suspend it), after which the clocks to USBPHY can be stopped. The 32 kHz low-power clock must remain active because you need it for wake-up detection.

The USB HS controller monitors USBPHY's entry and exit into Low-Power mode:

- If USB.PORTSC1[SUSP] is 1, the UTMI interface is driven to the FS or LS state.
- If USB.PORTSC1[PHCD] is 1, the UTMI signal "SuspendM" goes low.
- If CTRL[ENAUTOCLR_PHY_PWD] and CTRL[ENAUTOCLR_CLKGATE] are 1, the UTMI clock and PHY transmitter are automatically enabled when the "SuspendM" signal goes high.

USBPHY enters Low-Power mode under the control of the driver software. See the "Universal Serial Bus Controller (USB)" chapter for details.

58.3.7 Clocking

The following table describes the clock sources for USBPHY. See the chip's clocking chapter for clock setting, configuration, and gating information.

Table 923. USBPHY clocks

Clock name	Description
Peripheral clock	Free running IPS bus clock for register configuration
Peripheral access clock	Gated IPS bus clock

58.3.8 Reset

Table 924. USBPHY resets

Reset	How it works
Chip	USBPHY's logic and registers are reset to their default states.
UTMI	UTMI_RESET signal from the USB controller resets the UTMI state machines, FIFO, and so on.
Software	A soft reset (SFTRST) of USBPHY's USBPHY Powerdown (PWD), USBPHY Transmitter Control (TX), USBPHY Receiver Control (RX), USBPHY General Control (CTRL), STATUS[OTGID_STATUS], DEBUG0_STATUS[UTMI_RXERROR_FAIL_COUNT], USBPHY Debug (DEBUG), and UTMI Debug Status 1 (DEBUG1) registers and fields takes place when you write 1 to CTRL[SFTRST]. This also resets state machines in the PHY. Write 0 to CTRL[SFTRST] to release the PHY from reset. **NOTE** A soft reset can take multiple clock periods to complete; therefore, do not write 1 to CTRL[CLKGATE] when writing 1 to CTRL[SFTRST]. The reset process gates the clocks automatically.

58.3.9 Interrupts

Table 925. USBPHY interrupts

Flag or event	Interrupt enable	Description
CTRL[DEVPLUGIN_IRQ]	CTRL[ENIRQDEVPLUGIN]	Indicates nonstandard resistive plugged-in detection
CTRL[RESUME_IRQ]	CTRL[ENIRQRESUMEDETECT]	Indicates that the host is sending a "resume" signal after a "suspend" signal
CTRL[WAKEUP_IRQ]	CTRL[ENIRQWAKEUP]	Indicates that the USB controller has stopped driving the "SuspendM" signal in its active-low state to the PHY

Table continues on the next page...

Table 925. USBPHY interrupts (continued)

Flag or event	Interrupt enable	Description
CTRL[OTG_ID_CHG_IRQ]	CTRL[ENOTG_ID_CHG_IRQ]	Indicates that the value of the OTG ID pin has changed
CTRL[HOSTDISCONDETECT_IRQ]	CTRL[ENIRQHOSTDISCON]	Indicates that the host has detected disconnection of the device in HS mode
USBDCD interrupt	USBDCD interrupt enable	See the "Interrupts" section in the "USB Device Charger Detection Module (USBDCD)" chapter for details
DP changes	USB.CTRL1[WKUP_DPDM_EN]	Wake-up on DP changes—only for the USB device
DN changes	USB.CTRL1[WKUP_DPDM_EN]	Wake-up on DN changes—only for the USB device
ID changes	USB.CTRL1[WKUP_ID_EN]	Wake-up on ID changes
VBUS valid	USB.CTRL1[WKUP_VBUS_EN]	Wake-up on VBUS valid
Session valid	USB.CTRL1[WKUP_VBUS_EN]	Wake-up on session valid

58.4 External signals

Table 926. USBPHY external signals

Signal	Description	Direction
DP	D+ pin	I/O
DN	D- pin	I/O
USB_ID	ID pin	Input
VBUS	VBUS pin	Power

58.5 Initialization

Perform the following procedure to initialize the internal 480 MHz USB PLL clock:

1. Enable the reference clock for USB PLL.
2. Enable the USBPHY PLL regulator.
3. Release the PHY from reset.
4. Power up the PLL.
5. Configure PLL_SIC[PLL_DIV_SEL] appropriately, according to the selected external reference. The system oscillator ranges from 16 MHz to 32 MHz.

58.6 Application information

58.6.1 Recommended register configuration for USB certification

USBPHY has programmability to adjust several transceiver parameters. You can:

- Adjust the TX HS driver termination impedance, FS driver output impedance, and HS driver output currents by using USBPHY Transmitter Control (TX).

- Adjust the HS RX thresholds for the envelope detector or squelch comparator and the host disconnect comparator by using USBPHY Receiver Control (RX).

The default values of the transceiver parametric trim fields are centered with no changes needed for USB certification testing when the chip is used with boards optimized for signal integrity on the HS USB port.

It may be useful to make changes to the parametric trim fields in cases where external components are inserted between the USB DP and DN pins, or compromises are made on USB DP and DN signal routing.

58.6.2 Register macro usage

A read-modify-write (RMW) operation involves updating a field without disrupting the contents of the remaining fields in the register. In this operation, the CPU reads the register, modifies the target field, and then writes the results back to the register. Because of the initial register read, this is an expensive operation in terms of CPU cycles.

To address this issue, you can implement some USBPHY registers as a group, including registers that can be used to either set, clear, or toggle (SCT) individual fields of the primary register. When writing to an SCT register, all fields that are 1 perform the associated operation on the primary register, while all fields that are 0 are not affected. The SCT registers always read back 0, and must have the write-only access type. You cannot implement SCT registers if the primary register has a read-only access type.

With this architecture, you can update one or more fields using only register writes. First, all bits of the target fields are cleared by a write to the associated clear register, and then the desired value of the target fields is written to the set register. This sequence of two writes is referred to as a clear-set (CS) operation.

A CS operation does have a potential drawback—whenever a field is modified, USBPHY sees a value of 0 before you write the final value. For most fields, passing through the 0 state is not a problem. Nonetheless, this behavior is something you must consider when using a CS operation.

Also, you do not require a CS operation for fields that are 1-bit wide. While the CS operation works in this case, it is more appropriate to set or clear the target bit (that is, one write instead of two). A simple set or clear operation is also atomic, while a CS operation is not.

NOTE

All macros for SCT are not atomic. For registers that do not provide hardware support for this functionality, these macros are implemented as a sequence of RMW operations. You must remember this when an atomic operation is required because an unexpected behavior may take place if an interrupt occurs during a critical update sequence.

58.7 USBPHY register descriptions

USBPHY does not generate transfer errors when accessing unimplemented registers in its register space.

Also, the SCT feature is implemented for some registers (see Register macro usage for information about this feature).

58.7.1 USBPHY memory map

USBPHY1 base address: 42CA_0000h

USBPHY2 base address: 42CB_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	USBPHY Powerdown (PWD)	32	RW	001E_1C00h
4h	USBPHY Powerdown (PWD_SET)	32	RW	001E_1C00h
8h	USBPHY Powerdown (PWD_CLR)	32	RW	001E_1C00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
Ch	USBPHY Powerdown (PWD_TOG)	32	RW	001E_1C00h
10h	USBPHY Transmitter Control (TX)	32	RW	1008_0807h
14h	USBPHY Transmitter Control (TX_SET)	32	RW	1008_0807h
18h	USBPHY Transmitter Control (TX_CLR)	32	RW	1008_0807h
1Ch	USBPHY Transmitter Control (TX_TOG)	32	RW	1008_0807h
20h	USBPHY Receiver Control (RX)	32	RW	0000_0000h
24h	USBPHY Receiver Control (RX_SET)	32	RW	0000_0000h
28h	USBPHY Receiver Control (RX_CLR)	32	RW	0000_0000h
2Ch	USBPHY Receiver Control (RX_TOG)	32	RW	0000_0000h
30h	USBPHY General Control (CTRL)	32	RW	See section
34h	USBPHY General Control (CTRL_SET)	32	RW	See section
38h	USBPHY General Control (CTRL_CLR)	32	RW	See section
3Ch	USBPHY General Control (CTRL_TOG)	32	RW	See section
40h	USBPHY Status (STATUS)	32	R	See section
50h	USBPHY Debug (DEBUG)	32	RW	7F18_0000h
54h	USBPHY Debug (DEBUG_SET)	32	RW	7F18_0000h
58h	USBPHY Debug (DEBUG_CLR)	32	RW	7F18_0000h
5Ch	USBPHY Debug (DEBUG_TOG)	32	RW	7F18_0000h
60h	UTMI Debug Status 0 (DEBUG0_STATUS)	32	R	0000_0000h
70h	UTMI Debug Status 1 (DEBUG1)	32	RW	0000_1000h
74h	UTMI Debug Status 1 (DEBUG1_SET)	32	RW	0000_1000h
78h	UTMI Debug Status 1 (DEBUG1_CLR)	32	RW	0000_1000h
7Ch	UTMI Debug Status 1 (DEBUG1_TOG)	32	RW	0000_1000h
80h	USBPHY Version (VERSION)	32	R	0500_0000h
A0h	USBPHY PLL Control and Status (PLL_SIC)	32	RW	00D1_2000h
A4h	USBPHY PLL Control and Status (PLL_SIC_SET)	32	RW	00D1_2000h
A8h	USBPHY PLL Control and Status (PLL_SIC_CLR)	32	RW	00D1_2000h
ACh	USBPHY PLL Control and Status (PLL_SIC_TOG)	32	RW	00D1_2000h
C0h	USBPHY VBUS Detect Control (USB1_VBUS_DETECT)	32	RW	0070_0004h
C4h	USBPHY VBUS Detect Control (USB1_VBUS_DETECT_SET)	32	RW	0070_0004h
C8h	USBPHY VBUS Detect Control (USB1_VBUS_DETECT_CLR)	32	RW	0070_0004h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
CCh	USBPHY VBUS Detect Control (USB1_VBUS_DETECT_TOG)	32	RW	0070_0004h
D0h	USBPHY VBUS Detector Status (USB1_VBUS_DET_STAT)	32	R	0000_0001h
E0h	USBPHY Charger Detect Control (USB1_CHRG_DETECT)	32	RW	8018_0000h
E4h	USBPHY Charger Detect Control (USB1_CHRG_DETECT_SET)	32	RW	8018_0000h
E8h	USBPHY Charger Detect Control (USB1_CHRG_DETECT_CLR)	32	RW	8018_0000h
ECh	USBPHY Charger Detect Control (USB1_CHRG_DETECT_TOG)	32	RW	8018_0000h
F0h	USBPHY Charger Detect Status (USB1_CHRG_DET_STAT)	32	R	0000_0000h
100h	USBPHY Analog Control (ANACTRL)	32	RW	8200_0402h
104h	USBPHY Analog Control (ANACTRL_SET)	32	RW	8200_0402h
108h	USBPHY Analog Control (ANACTRL_CLR)	32	RW	8200_0402h
10Ch	USBPHY Analog Control (ANACTRL_TOG)	32	RW	8200_0402h
110h	USBPHY Loopback Control and Status (USB1_LOOPBACK)	32	RW	0055_0000h
114h	USBPHY Loopback Control and Status (USB1_LOOPBACK_SET)	32	RW	0055_0000h
118h	USBPHY Loopback Control and Status (USB1_LOOPBACK_CLR)	32	RW	0055_0000h
11Ch	USBPHY Loopback Control and Status (USB1_LOOPBACK_TOG)	32	RW	0055_0000h
120h	USBPHY Loopback Packet Number Selection (USB1_LOOPBACK_HSFSCNT)	32	RW	0004_0010h
124h	USBPHY Loopback Packet Number Selection (USB1_LOOPBACK_HSFSCNT_SET)	32	RW	0004_0010h
128h	USBPHY Loopback Packet Number Selection (USB1_LOOPBACK_HSFSCNT_CLR)	32	RW	0004_0010h
12Ch	USBPHY Loopback Packet Number Selection (USB1_LOOPBACK_HSFSCNT_TOG)	32	RW	0004_0010h
130h	USBPHY Trim Override Enable (TRIM_OVERRIDE_EN)	32	RW	0000_007Fh
134h	USBPHY Trim Override Enable (TRIM_OVERRIDE_EN_SET)	32	RW	0000_007Fh
138h	USBPHY Trim Override Enable (TRIM_OVERRIDE_EN_CLR)	32	RW	0000_007Fh
13Ch	USBPHY Trim Override Enable (TRIM_OVERRIDE_EN_TOG)	32	RW	0000_007Fh

58.7.2 USBPHY Powerdown (PWD)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

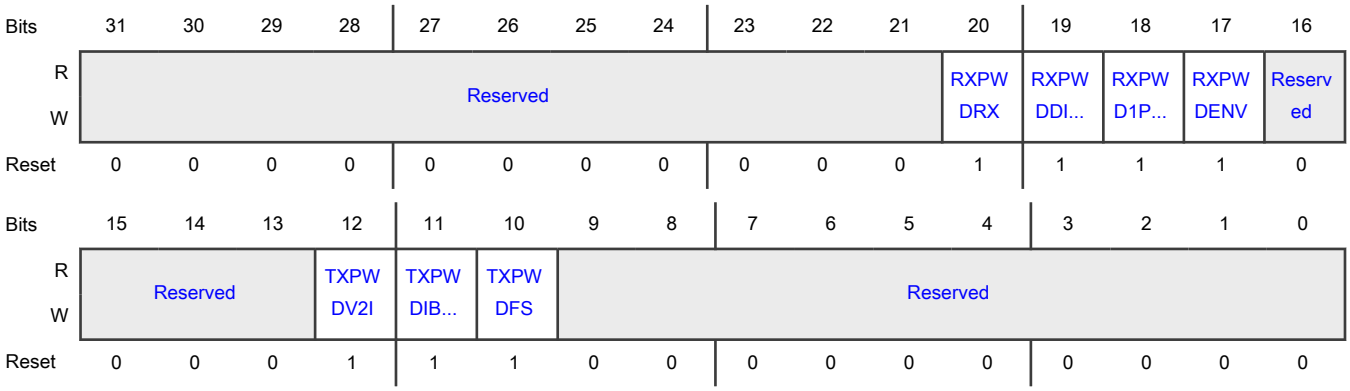
Register	Offset	Description
PWD	0h	USBPHY Powerdown
PWD_SET	4h	Writing 1 to a bit in this register ensures that the corresponding bit in PWD is 1
PWD_CLR	8h	Writing 1 to a bit in this register ensures that the corresponding bit in PWD is 0
PWD_TOG	Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in PWD

Function

Provides an overall control of the PHY power state.

You must enable the PHY clocks before programming this register.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 RXPWDRX	Receiver Powerdown Disables the HS RX differential receiver, HS envelope detector, and HS host disconnect comparator along with their RX current mirror bias circuits. The overall control of the RX current bias block is shared with the battery charge circuit, but this field powers down the other aforementioned RX circuits (the entire USBPHY receiver block) unconditionally, except for the FS differential receiver. This field becomes 0 automatically in case of a wake-up event when USB is suspended while CTRL[ENAUTOCLR_PHY_PWD] is 1. Additionally, this field becomes 1 automatically at the falling edge of the signal from the USB controller that drives CTRL[UTMI_SUSPENDM] to enter the Suspend state. 0b - Enable for normal operation

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Disable or power down RX circuits
19 RXPWDDIFF	HS Receiver Powerdown Disables the USB HS RX differential receiver that is used for data reception in HS operation. This field becomes 0 automatically in case of a wake-up event when USB is suspended while CTRL[ENAUTOCLR_PHY_PWD] is 1. Additionally, this field becomes 1 automatically at the falling edge of the signal from the USB controller that drives CTRL[UTMI_SUSPENDM] to enter the Suspend state. 0b - Enable for normal operation 1b - Disable or power down
18 RXPWD1PT1	FS Receiver Powerdown Disables the USB FS RX differential receiver that is used for data reception in both FS and LS operations. This field does not affect the bias and operation of the FS differential receiver. This field becomes 0 automatically in case of a wake-up event when USB is suspended while CTRL[ENAUTOCLR_PHY_PWD] is 1. Additionally, this field becomes 1 automatically at the falling edge of the signal from the USB controller that drives CTRL[UTMI_SUSPENDM] to enter the Suspend state. 0b - Enable for normal operation 1b - Disable or power down
17 RXPWDENV	Receiver Envelope Powerdown Disables the USB HS RX envelope detector that generates the squelch signal to qualify the HS RX data. The host disconnect comparator is used in Host mode to check for a disconnection from the remote device. This field becomes 0 automatically in case of a wake-up event when USB is suspended while CTRL[ENAUTOCLR_PHY_PWD] is 1. Additionally, this field becomes 1 automatically at the falling edge of the signal from the USB controller that drives CTRL[UTMI_SUSPENDM] to enter the Suspend state. 0b - Enable for normal operation 1b - Disable or power down
16-13 —	Reserved
12 TXPWDV2I	USBPHY TX V-I Converter and Current Mirror Powerdown Disables the USBPHY TX V-I converter that provides currents used in the HS TX drivers. The converter depends on both USBPHY's bias module and the bias current block. This field becomes 0 automatically in case of a wake-up event when USB is suspended while CTRL[ENAUTOCLR_PHY_PWD] is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Additionally, this field becomes 1 automatically at the falling edge of the signal from the USB controller that drives CTRL[UTMI_SUSPENDM] to enter the Suspend state. 0b - Enable for normal operation 1b - Disable or power down
11 TXPWDIBIAS	Transmitter Bias Powerdown Disables the USBPHY current bias block for the transmitter that provides mirrored currents to track USBPHY's bias module reference current generator for several functions in the TX, RX, and detector sections. Control of the current bias block is shared with the battery charge circuit. Writing 1 to this field does not power down the circuit unless the value of the corresponding field in the battery charge control is also 1 for power down. The field must be 1 only when the USB is in the Suspend bus state. This effectively powers down the entire USB transmit path. The field must become 1 only when the USB is in Suspend mode. This effectively powers down the entire USB transmit path. This field becomes 0 automatically in case of a wake-up event when USB is suspended while CTRL[ENAUTOCLR_PHY_PWD] is 1. Additionally, this field becomes 1 automatically at the falling edge of the signal from the USB controller that drives CTRL[UTMI_SUSPENDM] to enter the Suspend state. **NOTE** These circuits are shared with the battery charge circuit. Writing 1 to this field does not power down these circuits, unless the corresponding field in the battery charger is also configured as 1 for a power down. 0b - Enable for normal operation 1b - Disable or power down
10 TXPWDFS	FS Transmitter Powerdown Disables the USB FS TX drivers that must have their bias enabled during USB data communication, including resume and bus reset signaling. They are used in HS, FS, and LS because pulldown of the FS TX drivers provides local HS termination. If this field is 1, it turns off the current starvation sources and puts the drivers into high-impedance output. This field becomes 0 in case of a wake-up event if USB is suspended when CTRL[ENAUTOCLR_PHY_PWD] is 1. Additionally, this field becomes 1 automatically at the falling edge of the signal from the USB controller that drives CTRL[UTMI_SUSPENDM] to enter the Suspend state. 0b - Provide bias to enable for normal operation 1b - Disable or power down
9-0 —	Reserved

58.7.3 USBPHY Transmitter Control (TX)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
TX	10h	USBPHY Transmitter Control
TX_SET	14h	Writing 1 to a bit in this register ensures that the corresponding bit in TX is 1
TX_CLR	18h	Writing 1 to a bit in this register ensures that the corresponding bit in TX is 0
TX_TOG	1Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in TX

Function

Allows adjustment of transmit parametric values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				Reserved				Reserved		Reserved	Reserved	TXCAL45DP			
Reset	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved	Reserved	TXCAL45DN				Reserved				D_CAL			
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31-29 —	Reserved
28-24 —	The value of this reserved bit must remain 10000b.
23-22 —	Reserved
21	The value of this reserved bit must remain 0b.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
20 —	Reserved
19-16 TXCAL45DP	Transmit Calculation 45 Ω DP Specifies the trim value of the 45 Ω DP resistance. Decode to trim the nominal 45 Ω series termination resistance to the DP output pin. Resistance is centered at value 0111b, with the maximum resistance being 0000b. 0000b - +19.95% 0001b - +17.35% 0010b - +14.85% 0011b - +12.46% 0100b - +9.07% 0101b - +5.87% 0110b - +2.85% 0111b - 0% 1000b - –2.70% 1001b - –5.25% 1010b - –7.67% 1011b - –9.98% 1100b - –12.17% 1101b - –14.25% 1110b - –18.14% 1111b - –21.68%
15-14 —	Reserved
13 —	The value of this reserved bit must remain 0b.
12 —	Reserved
11-8 TXCAL45DN	Transmit Calculation 45 Ω DN Specifies the trim value of the 45 Ω DN resistance. Decode to trim the nominal 45 Ω series termination resistance to the DN output pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The resistance is centered at value 0110b, with the maximum resistance being 0000b. 0000b - +19.95% 0001b - +17.35% 0010b - +14.85% 0011b - +12.46% 0100b - +9.07% 0101b - +5.87% 0110b - +2.85% 0111b - 0% 1000b - –2.70% 1001b - –5.25% 1010b - –7.67% 1011b - –9.98% 1100b - –12.17% 1101b - –14.25% 1110b - –18.14% 1111b - –21.68%
7-4 —	The value of this reserved bit must remain 0b.
3-0 D_CAL	HS Transmit Output Current Trim Specifies the HS transmit output current trim value. Decode to trim the nominal 17.78 mA current source for the HS transmit drivers on the DP and DN pins. This current is directly proportional to the amplitude of the HS TX eye diagram. 0000b - +20.30% 0001b - +17.60% 0010b - +14.80% 0011b - +12.60% 0100b - +8.79% 0101b - +6.04% 0110b - +2.75% 0111b - 0% 1000b - –2.75% 1001b - –5.49%

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1010b - –7.69% 1011b - –10.40% 1100b - –12.60% 1101b - –14.30% 1110b - –18.10% 1111b - –22.00%

58.7.4 USBPHY Receiver Control (RX)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
RX	20h	USBPHY Receiver Control
RX_SET	24h	Writing 1 to a bit in this register ensures that the corresponding bit in RX is 1
RX_CLR	28h	Writing 1 to a bit in this register ensures that the corresponding bit in RX is 0
RX_TOG	2Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in RX

Function

Allows adjustment of receive parametric values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved									RXDBYPA...	Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved									DISCONADJ			Reserved	ENVADJ		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-23 —	Reserved
22 RXDBYPASS	Differential Receiver Bypass Bypasses the FS differential receiver. If this field is 0, normal operation continues. If this field is 1, the output of the DP single-ended receiver is used instead of the FS differential receiver. 0b - Operate normally 1b - Bypass
21-7 —	Reserved
6-4 DISCONADJ	Disconnect Adjustment Adjusts the trip point (trip-level voltage) for the disconnect detector. 000b - 0.56875 V 001b - 0.55000 V 010b - 0.58125 V 011b - 0.60000 V 1xxb - Reserved
3 —	Reserved
2-0 ENVADJ	Envelope Adjustment Adjusts the trip point for the envelope detector. The field values are trip-level voltages in nominal DC settings to indicate the effect of changing these fields. AC values measured during compliance testing are a bit higher. 000b - 0.1000 V 001b - 0.1125 V 010b - 0.1250 V 011b - 0.0875 V 1xxb - Reserved

58.7.5 USBPHY General Control (CTRL)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
CTRL	30h	USBPHY General Control
CTRL_SET	34h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL is 1
CTRL_CLR	38h	Writing 1 to a bit in this register ensures that the corresponding bit in CTRL is 0
CTRL_TOG	3Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in CTRL

Function

Handles OTG and host controls.

This register also includes interrupt enables, connectivity detect enables, and results.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SFTRST	CLKGATE	UTMI_SU...	HOST_FO...	OTG_ID_...	Reserved		FSDLL_R...	Reserved		Reserved	ENAUTOC...	ENAUTOC...	AUTORES...	WAKEUP_...	ENIRQWA...
W																
Reset	1	1	0	0	u	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ENUTMIL...	ENUTMIL...	Reserved	DEVPLUG...	ENIRQDE...	RESUME_...	ENIRQRE...	RESUMEI...	ENOTGID...	OTG_ID_...	DEVPLUG...	ENDEVPL...	HOSTDIS...	ENIRQHO...	ENHOSTD...	ENOTG_I...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SFTRST	Software Reset Soft-resets USBPHY Powerdown (PWD), USBPHY Transmitter Control (TX), USBPHY Receiver Control (RX), and USBPHY General Control (CTRL). 0b - Release from reset 1b - Reset
30 CLKGATE	Clock Gating Gates UTMI clocks. Write 1 to this field to save power when the USB is not actively used. Retain the configuration state when the clock is gated. **NOTE** This field can automatically become 0 in case of a wake-up event when USB is suspended, if ENAUTOCLR_CLKGATE is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Run clocks 1b - Gate clocks
29 UTMI_SUSPENDM	UTMI Suspend Mode Indicates a power-down state. UTMI_SUSPENDM is negative logic, as the UTMI specification requires it to be. If this field is 0, all powerdown fields in USBPHY Powerdown (PWD) are enabled. If this field is 0, the USB controller enters Suspend mode. 0b - Suspended 1b - Not suspended
28 HOST_FORCE_LS_SE0	FS EOP LS Timing Works with DEBUG[HOST_RESUME_DEBUG] and forces the next FS packet that is transmitted to have an EOP with LS timing. This field is used in Host mode for the resume sequence. After the packet is transferred, the field becomes 0. You can use this function to force the LS SE0 or use UTMI_SUSPENDM to trigger this event when existing the Suspend state. 0b - Do not force the next FS packet 1b - Force the next FS packet
27 OTG_ID_VALUE	ID Value Indicates the results of the USB_ID pin while monitoring the cable plugged into the Micro- or Mini-AB receptacle. If this field is 0, the ID resistance is less than Ra_Plug_ID, indicating host (A) side. If this field is 1, the ID resistance is greater than Rb_Plug_ID, indicating device (B) side. The functioning of this field is similar to that of STATUS[OTGID_STATUS]; however, this field includes debounce and system clock synchronization logic to filter the glitches on the USB ID pad. 0b - Lesser ID resistance than Ra_Plug_ID 1b - Greater ID resistance than Rb_Plug_ID
26-25 —	Reserved
24 FSDLL_RST_EN	FSDLL Reset Enable Enables resetting of the FSDLL lock detection logic at the end of each TX packet. 0b - Disables 1b - Enables
23-22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21 —	Reserved
20 ENAUTOCLR_PHY_PWD	PHY PWD Auto Clear Enable Enables automatic writing of 0 to the fields in USBPHY Powerdown (PWD) in case of a wake-up event while USB is suspended. You must write 1 to this field if auto wake-up is required without software interaction. 0b - Disables 1b - Enables
19 ENAUTOCLR_CLKGATE	Clock Gating Auto Clear Enable Enables automatic writing of 0 to CLKGATE in case of a wake-up event while USB is suspended. You must write 1 to this field if auto wake-up is required without software interaction. 0b - Disables 1b - Enables
18 AUTORESUME_EN	Auto Resume Enable Enables the auto resume feature. If this field is 1, USBPHY uses a 32 kHz clock to send resume signaling to respond to the USB device remote wakeup (for Host mode only). The field functionality is useful when PLL is off and the reference clock is powered down. 0b - Disables 1b - Enables
17 WAKEUP_IRQ	Wake-Up Interrupt Specifies whether a wake-up event exists. Reset this field by writing 1 to the clear address space, and not by a general write. 0b - No wake-up event exists 1b - Wake-up event exists
16 ENIRQWAKEUP	Wake-Up Interrupt Enable Enables interrupts for wake-up events. 0b - Disables 1b - Enables
15 ENUTMILEVEL3	UTMI Level 3 Enable Enables UTMI+ level 3 operation for USBPHY. You must write 1 to this field if an embedded host use case needs to support an external FS hub with an LS device connected. 0b - Disables

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enables
14 ENUTMILEVEL2	UTMI Level 2 Enable Enables UTMI+ level 2 operation for USBPHY. You must write 1 to this field if an embedded host use case needs to support an LS device. 0b - Disables 1b - Enables
13 —	The value of this reserved bit must remain 0b.
12 DEVPLUGIN_IRQ	Device Plug-In Interrupt Indicates that the device is connected. Reset this field by writing 1 to the SCT clear address space, and not by a general write. 0b - Not connected 1b - Connected
11 ENIRQDEVPLUGIN	Device Plug-In Interrupt Enable Enables an interrupt for the detection of connectivity to the USB line. 0b - Disables 1b - Enables
10 RESUME_IRQ	Interrupt Resume Specifies whether the host is sending a wake-up signal after suspend signaling. This field also becomes 1 after a reset during Suspend mode. Use this field to wake up the USB device from Suspend mode for either the host resume or host reset case. You must reset this field by writing 1 to the clear address space, and not by a general write. 0b - No resume interrupt 1b - Resume interrupt
9 ENIRQRESUMEDETECT	Resume Detection Interrupt Enable Enables an interrupt for detection of a non-J state on the USB line. You must write 1 to this field only after the USB device has entered Suspend mode. 0b - Disables 1b - Enables
8 RESUMEIRQSTICKY	RESUME_IRQ Sticky Specifies until when RESUME_IRQ remains 1 during the resume or reset state signaling period. 0b - Remains 1 during the wake-up period 1b - Remains 1 until you write 0 to it

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 ENOTGIDDETECT	Enable Internal OTG ID Detector Enables the circuit to detect resistance of the Mini-AB ID pin. 0b - Disables 1b - Enables
6 OTG_ID_CHG_IRQ	OTG ID Change Interrupt Indicates that the value of the OTG ID pin has changed. 0b - No ID change interrupt 1b - ID change interrupt
5 DEVPLUGIN_POLARITY	Device Plug-In Polarity Specifies whether to trip the interrupt when the USB device is plugged in or when it is unplugged. If this field is 0, it trips the interrupt when the USB cable from the host to the device is plugged in. If this field is 1, it trips the interrupt if the USB cable from the host to the device is unplugged. 0b - Plugged in 1b - Unplugged
4 ENDEVPLUGINDETECT	Nonstandard Resistive Plugged-In Detection Enable Controls connection of nominal 200 kΩ pullup resistors with both the DP and DN pins as a method of detecting, in Device mode, when a USB cable is attached. This field must remain 0 for normal USB data communication, or when using USBHSDCD for battery charger detection, according to *USB Battery Charger Specification Revision 1.2* or any other detection mechanism for USB cable plugin. STATUS[DEVPLUGIN_STATUS] reports the results of this detection method. 0b - Disables 1b - Enables
3 HOSTDISCONDETECT_IRQ	Host Disconnect Detection Interrupt Specifies whether the device is disconnected in HS Host mode. Reset this field by writing 1 to the SCT clear address space, and not by a general write. 0b - Connected 1b - Disconnected
2 ENIRQHOSTDISCON	Host Disconnect Interrupt Enable Enables an interrupt for detection of device disconnection when in HS Host mode. You must write 1 to this field after writing 1 to ENDEVPLUGINDETECT. 0b - Disables 1b - Enables

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 ENHOSTDISCONDETECT	Host Disconnect Detection Enable Enables the HS disconnect detector for Host mode. This signal allows the override of detection enabling, which the UTMI controller normally performs. The UTMI controller enables this circuit whenever the host sends a start-of-frame packet. Write 1 to this field when an HS device is connected. This is suggested during a bus reset. Ensure that this field is not 1 at the end of resume signaling; otherwise, a wrong disconnect status may be detected. 0b - Disables 1b - Enables
0 ENOTG_ID_CHG_IRQ	ID Change Interrupt Enable Enables the interrupt for ID change. 0b - Disables 1b - Enables

58.7.6 USBPHY Status (STATUS)

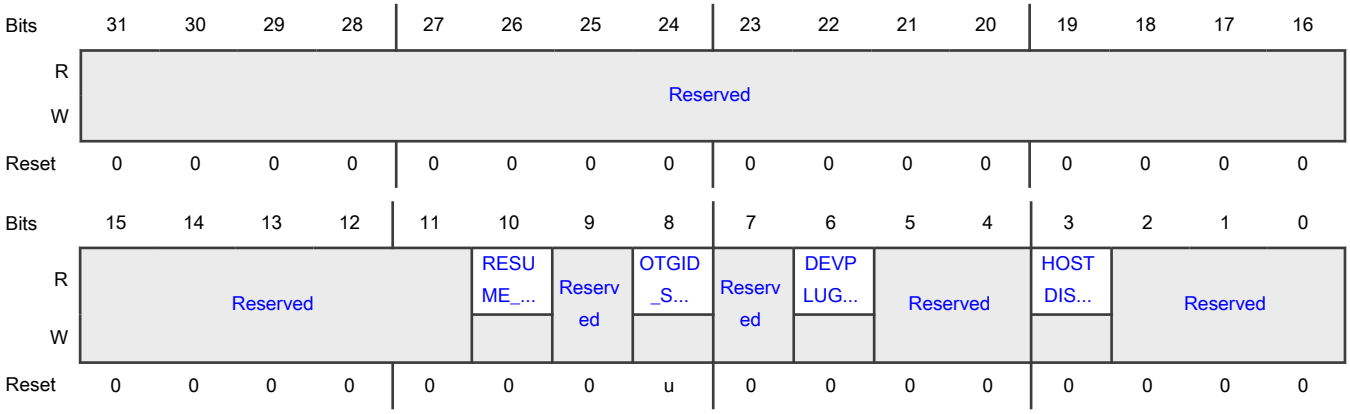
Offset

Register	Offset
STATUS	40h

Function

Holds results of IRQ and other detects.

Diagram



Fields

Field	Function
31-11 —	Reserved
10 RESUME_STATUS	Resume Status Indicates whether the line state is in J state. This field provides a valid indication of resume signaling only when operating in FS or HS mode, after being in the Suspend bus state. This status field may toggle during USB data packet communication. Also, it does not indicate resume signaling when operating as a host that is directly attached to an LS device. 0b - Is in J state 1b - Is not in J state
9 —	Reserved
8 OTGID_STATUS	OTG ID Status Indicates the results of the USB_ID pin on the USB cable plugged into the local Micro-AB or Mini-AB receptacle. If this field is 0, the ID resistance to ground is less than Ra_Plug_ID, indicating host (A) side. If this field is 1, the ID resistance is greater than Rb_Plug_ID, indicating device (B) side. 0b - Lesser ID resistance 1b - Greater ID resistance
7 —	Reserved
6 DEVPLUGIN_STATUS	Status Indicator for Nonstandard Resistive Plugged-In Detection Indicates whether the device is connected on the DP and DN lines using the nonstandard resistive plugged-in detection method that CTRL[ENDEVPLUGINDETECT] controls. When a USB cable attached to a remote host is attached to the local device, the 15 kΩ host pulldowns override the high value resistors used in this detection method. 0b - No attachment detected 1b - Cable attachment detected
5-4 —	Reserved
3 HOSTDISCONDETECT_STATUS	Host Disconnect Detection Status Detects at the local host (downstream) port that the USB cable is disconnected when in HS mode. 0b - Do not detect 1b - Detect

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-0 —	Reserved

58.7.7 USBPHY Debug (DEBUG)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
DEBUG	50h	USBPHY Debug
DEBUG_SET	54h	Writing 1 to a bit in this register ensures that the corresponding bit in DEBUG is 1
DEBUG_CLR	58h	Writing 1 to a bit in this register ensures that the corresponding bit in DEBUG is 0
DEBUG_TOG	5Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in DEBUG

Function

Debugs USBPHY.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	CLKGATE	HOST_RE...	SQUELCHRESETLENGTH				ENSQUEL...	Reserved				SQUELCHRESETCOUNT			
Reset	0	1	1	1	1	1	1	1	0	0	0	1	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			ENTX2RX...	TX2RXCOUNT				Reserved		ENHSTPULLDOWN		HSTPULLDOWN		Reserved	OTGIDPI...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 CLKGATE	Clock Gating Gates the test clocks. To save power while the USB is not being actively used, retain the value of 1 in this field. The configuration state is maintained when the clock is gated. 0b - Run clocks 1b - Gate clocks
29 HOST_RESUME_DEBUG	Host Resume Selects the method used to generate the LS SE0 at the end of host resume signaling. The default method triggers from a change to the UTMI_SUSPENDM signal from the USB controller to the PHY. The other method depends on programming CTRL[HOST_FORCE_LS_SE0] at an appropriate time. If this field is 0, it generates LS SE0 at the end of host resume based on CTRL[HOST_FORCE_LS_SE0]. If this field is 1, it generates LS SE0 at the end of host resume based on the state of CTRL[UTMI_SUSPENDM]. 0b - Based on CTRL[HOST_FORCE_LS_SE0] 1b - Based on CTRL[UTMI_SUSPENDM]
28-25 SQUELCHRESETLENGTH	Squelch Reset Length Indicates the number of 480 MHz clock intervals used for the delay time. The changes you make to this field are used only for a specific HS RX debug mode that ENSQUELCHRESET enables. This field adjusts the delay time between the detection of valid HS RX data and release of squelch for that debug mode. The field value indicates the number of 480 MHz clock intervals used for the delay time.
24 ENSQUELCHRESET	Enable Squelch Reset Enables squelch reset. Let the value of this field be 1 to allow normal operation of the squelch state machine for qualification of HS receive data. Writing 0 to this field freezes the functional mode squelch state machine in its reset state, and must be done only for a specific HS RX debug mode. 0b - Disables 1b - Enables
23-21 —	Reserved
20-16 SQUELCHRESETCOUNT	Squelch Reset Count Adjusts the delay time, in 480 MHz clock intervals, between the detection of squelch to the reset of HS RX state machines. Any changes that you make to this field must only be used for a specific HS RX debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-13 —	Reserved
12 ENTX2RXCOUNT	Enable Countdown from TX to RX Packets for Debug Enables a countdown to transition between the TX and RX packets for debug. 0b - Disables 1b - Enables
11-8 TX2RXCOUNT	Set Countdown Delay Value from TX to RX Packets for Debug Specifies the delay between the end of transmit to the beginning of receive packets for debug. This is a Johnson counter value and therefore counts to 8.
7-6 —	Reserved
5-4 ENHSTPULLDOWN	Enable Host Pulldown Overdrive Mode Selects whether to override the control of DP and DN pins. This field selects Host Pulldown Overdrive mode. Write 1b to bit 5 to override the control of the DP 15 kΩ pulldown, which the USB controller normally performs. Write 1b to bit 4 to override the control of the DN 15 kΩ pulldown, which the USB controller normally performs. Write 00b to both the bits to disable Host Pulldown Overdrive mode and return the control of the pulldown resistors to the USB controller. When in Host Pulldown Overdrive mode, HSTPULLDOWN further controls the connection of the individual pulldown resistors. Both pulldown resistors are also unconditionally enabled when ANACTRL[DEV_PULLDOWN] becomes 1 without considering the value of this field. 00b - Disable Host Pulldown Overdrive mode 01b - Override the control of DN 15 kΩ pulldown 10b - Override the control of DP 15 kΩ pulldown 11b - Override the control of DP and DN 15 kΩ pulldown
3-2 HSTPULLDOWN	Select DP and DN Pulldown Resistors in Host Pulldown Overdrive Mode Selects whether to connect pulldown resistors on the DP and DN pins if Host Pulldown Overdrive mode is enabled through ENHSTPULLDOWN. Write 1b to bit 3 of this field to connect the 15 kΩ pulldown on the DP line, and write 1b to bit 2 to connect the 15 kΩ pulldown on the DN line. Write 00b to both the bits to disconnect the resistors in Host Pulldown Override mode. 00b - Disconnect the resistors 01b - Connect 15 kΩ pulldown on DN 10b - Connect 15 kΩ pulldown on DP 11b - Connect 15 kΩ pulldown on DP and DN

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 —	Reserved
0 OTGIDPIOLOCK	Hold OTG_ID Holds OTG ID value. After the OTG ID status is acquired from STATUS[OTGID_STATUS], use this field to hold the OTG ID value. This is to save power for the comparators that are used to determine the ID status.

58.7.8 UTMI Debug Status 0 (DEBUG0_STATUS)

Offset

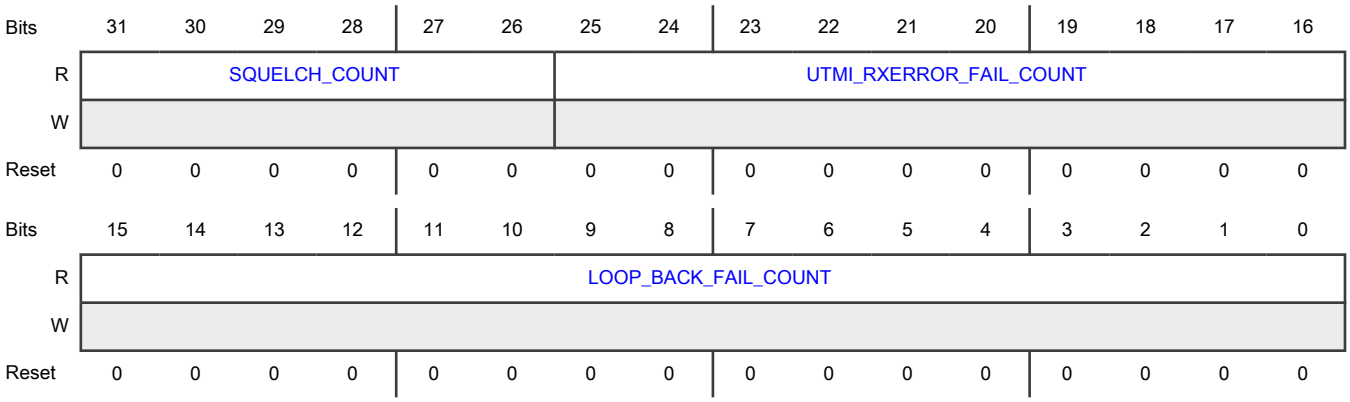
Register	Offset
DEBUG0_STATUS	60h

Function

Holds multiple views for counters and status of state machines.

This field works with DEBUG1[DBG_ADDRESS] to select the function to view. The default settings used to count errors are described in this register.

Diagram



Fields

Field	Function
31-26 SQUELCH_COUNT	Squelch Count Indicates the running count of the squelch reset instead of the normal end for HS RX.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-16 UTMI_RXERROR_FAIL_COUNT	UTMI Receive Error Fail Count Indicates the running count of UTMI_RXERROR.
15-0 LOOP_BACK_FAIL_COUNT	Loopback Fail Count Indicates the running count of the failed pseudo-random generator loopback. Each time it enters Test mode, the counter goes to 900D and counts up for every detected packet failure in digital and analog loopback tests.

58.7.9 UTMI Debug Status 1 (DEBUG1)

Offset

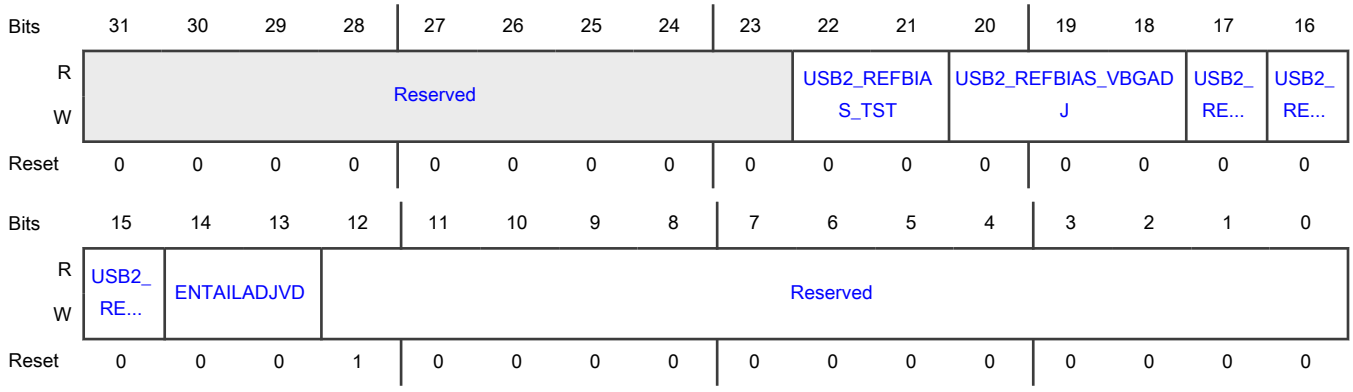
This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
DEBUG1	70h	UTMI Debug Status 1
DEBUG1_SET	74h	Writing 1 to a bit in this register ensures that the corresponding bit in DEBUG1 is 1
DEBUG1_CLR	78h	Writing 1 to a bit in this register ensures that the corresponding bit in DEBUG1 is 0
DEBUG1_TOG	7Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in DEBUG1

Function

Selects the muxing of USBPHY Debug (DEBUG) to be indicated in UTMI Debug Status 0 (DEBUG0_STATUS).

Diagram



Fields

Field	Function
31-23 —	Reserved
22-21 USB2_REFBIAS_TST	Bias Current Control Adjustment Specifies the main current trim for the USBPHY bias module current reference generator. Use this field to trim the PHY's current reference when TRIM_OVERRIDE_EN[TRIM_REFBIAS_TST_OVERRIDE] is 1. If TRIM_OVERRIDE_EN[TRIM_REFBIAS_TST_OVERRIDE] becomes 0, the current trim values in TRIM_OVERRIDE_EN[TRIM_USB2_REFBIAS_TST] are used instead. 00b - ≈ 10 µA reference current; nominal 01b - ≈ 0.9x compared to nominal 10b - ≈ 0.8x compared to nominal 11b - ≈ 1.1x compared to nominal
20-18 USB2_REFBIAS_VBGADJ	Bandgap Voltage Adjustment Specifies the voltage adjustment trim on USBPHY bias module bandgap generator. Use this field to trim the PHY's bandgap voltage when TRIM_OVERRIDE_EN[TRIM_REFBIAS_VBGADJ_OVERRIDE] is 1. If TRIM_OVERRIDE_EN[TRIM_REFBIAS_VBGADJ_OVERRIDE] becomes 0, the voltage trim values in TRIM_OVERRIDE_EN[TRIM_USB2_REFBIAS_VBGADJ] are used instead. 000b - Nominal bandgap voltage; flattest temperature coefficient 001b - ≈ +10 mV compared to nominal 010b - ≈ +20 mV compared to nominal 011b - ≈ +30 mV compared to nominal; most-positive temperature coefficient 100b - ≈ -10 mV compared to nominal 101b - ≈ -20 mV compared to nominal 110b - ≈ -30 mV compared to nominal 111b - ≈ -40 mV compared to nominal; most-negative temperature coefficient
17 USB2_REFBIAS_LOWPWR	Reference Bias Low Power Configuration Configures the bias currents in the bandgap reference amplifier in the USBPHY bias module. If this field is 0, the bandgap amplifier for a nominal bias current is configured. If this field is 1, the bandgap amplifier for 50% of the nominal bias current is configured. The field must remain 0 for normal operation. Writing 1 is only intended for lab characterization purposes. 0b - Nominal bias current 1b - 50% of nominal bias current
16	Bandgap Voltage Status Comparator Powerdown

Table continues on the next page...

Table continued from the previous page...

Field	Function
USB2_REFBIAS_PWDVBGUP	Disables the VBGUP comparator to save its bias current after the bandgap voltage is valid and stable. The VBGUP comparator monitors the USBPHY bias module bandgap voltage to check that the voltage is in the valid range. If this field is 1, MISC[VBGUP] is also forced to become 1. You cannot detect a bandgap voltage problem if this field is 1, and the PHY does not automatically restore the VBGUP comparator if such a problem occurs. Therefore, it is not recommended to make changes to this field. 0b - Enables 1b - Disables
15 USB2_REFBIAS_SELFBIASOFF	Self-Bias Off for Reference Bias Amplifiers and Comparators Selects the local bias sources to use for the amplifiers and comparators in the USBPHY bias module. If this field is 0, self-bias is used for bias reference amplifier and comparator circuits. If this field is 1, the current reference bias is used for amplifier and comparator circuits. Self-bias is required at reference generator startup. After the bandgap voltage and reference current generator are stable, bias from the reference generator's own regulated currents can be used instead. You must let the value of this field be 0 during a ramp of the 1.8 V power domain and during any time that the USBPHY bias reference circuit transitions from the disabled to enabled state. Switching to the current reference bias after the bandgap voltage is stable may somehow improve bandgap voltage stability, but could cause problems if the 1.8 V power is interrupted. You must monitor MISC[VBGUP]; intervention is mandatory if the value of this field changes. You must also note the interaction of USB2_REFBIAS_PWDVBGUP and MISC[VBGUP]. 0b - Self-bias 1b - Current reference bias
14-13 ENTAILADJVD	HS RX Squelch Rise Time Delay Trim Adjusts the squelch rise time delay for the squelch signal, from the HS RX envelope detector, when TRIM_OVERRIDE_EN[TRIM_ENV_TAIL_ADJ_VD_OVERRIDE] is 1. If TRIM_OVERRIDE_EN[TRIM_ENV_TAIL_ADJ_VD_OVERRIDE] becomes 0, the squelch rise time trim values in TRIM_OVERRIDE_EN[TRIM_USB_REG_ENV_TAIL_ADJ_VD] are used instead. 00b - Squelch rising edge delay is nominal 01b - +20% delay compared to nominal 10b - -20% delay compared to nominal 11b - -40% delay compared to nominal
12-0 —	The value of this reserved bit must remain 0b.

58.7.10 USBPHY Version (VERSION)

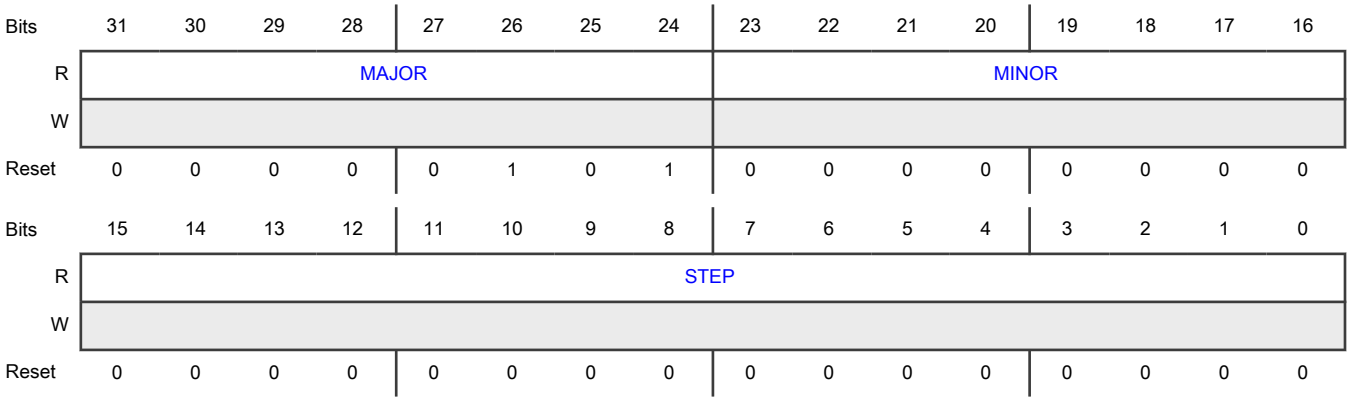
Offset

Register	Offset
VERSION	80h

Function

Indicates the version of USBPHY.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Indicates the "Major" field of the USBPHY version.
23-16 MINOR	Minor Indicates the "Minor" field of the USBPHY version.
15-0 STEP	Step Indicates the stepping of the USBPHY version.

58.7.11 USBPHY PLL Control and Status (PLL_SIC)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
PLL_SIC	A0h	USBPHY PLL Control and Status

Table continues on the next page...

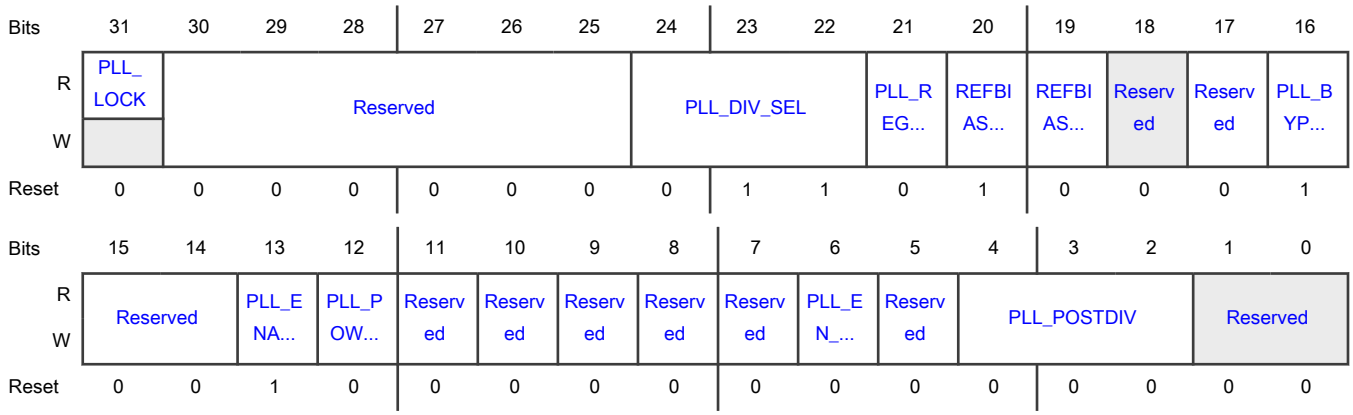
Table continued from the previous page...

Register	Offset	Description
PLL_SIC_SET	A4h	Writing 1 to a bit in this register ensures that the corresponding bit in PLL_SIC is 1
PLL_SIC_CLR	A8h	Writing 1 to a bit in this register ensures that the corresponding bit in PLL_SIC is 0
PLL_SIC_TOG	ACh	Writing 1 to a bit in this register inverts the value of the corresponding bit in PLL_SIC

Function

Configures the 480 MHz USBPHY PLL.

Diagram



Fields

Field	Function
31 PLL_LOCK	PLL Lock Indicates the current status of the USB PLL lock. 0b - Not locked 1b - Locked
30-25 —	Reserved
24-22 PLL_DIV_SEL	PLL Divider Selection Controls the USB PLL feedback loop divider. The valid range for divider values is 13–240. (Fout = Fin × div_select). The USB PLL produces a 480 MHz output clock. This field allows the use of different frequency signals for the PLL reference clock input connected to the OSCCLK signal from the system oscillator. If TRIM_OVERRIDE_EN[TRIM_DIV_SEL_OVERRIDE] is 1, the USB PLL uses this register value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - Divide by 13 001b - Divide by 15 010b - Divide by 16 011b - Divide by 20 100b - Divide by 22 101b - Divide by 25 110b - Divide by 30 111b - Divide by 240
21 PLL_REG_ENABLE	PLL Regulator Enable Enables the USB PLL regulator. You must set this field to 15 µs before selecting PLL_POWER to avoid glitches on the PLL output clock. 0b - Disables 1b - Enables
20 REFBIAS_PWD	Reference Bias Powerdown Enables reference bias powerdown. This field is used only when REFBIAS_PWD_SEL is 1. 0b - Disables 1b - Enables
19 REFBIAS_PWD_SEL	Reference Bias Powerdown Selection Selects PLL_POWER or REFBIAS_PWD to control the reference bias. 0b - PLL_POWER 1b - REFBIAS_PWD
18 —	Reserved
17 —	The value of this reserved bit must remain 0b.
16 PLL_BYPASS	PLL Bypass Bypasses the USB PLL. 0b - Do not bypass 1b - Bypass
15-14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
13 PLL_ENABLE	PLL Enable Enables the clock output from the USB PLL (the PLL powerup control also controls the PLL output enable signal). USBPHY disables the PLL output before PLL powerdown, and enables the PLL output after PLL powerup. You must only set it when initializing the PLL. 0b - Disables 1b - Enables
12 PLL_POWER	PLL Power Enables the USB PLL (USBPHY also controls PLL powerup and powers it down when the USB is suspended and not in use). 0b - Disables 1b - Enables
11 —	The value of this reserved bit must remain 0b for normal operation.
10 —	The value of this reserved bit must remain 0b.
9 —	The value of this reserved bit must remain 0b.
8 —	The value of this reserved bit must remain 0b.
7 —	The value of this reserved bit must remain 0b.
6 PLL_EN_USB_CLKS	PLL USB Clocks Enable Enables the USB clock output from USBPHY PLL. 0b - Disables 1b - Enables
5 —	The value of this reserved bit must remain 0b.
4-2 PLL_POSTDIV	PLL Post-Divider Output Value Configuration Specifies the integer divide values for PLL post divider. 000b - Disable the output of PLL post divider

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - Divide value is 1 010b - Divide value is 2 011b - Divide value is 3 100b - Divide value is 4 101b - Divide value is 5 110b - Divide value is 6 111b - Reserved
1-0 —	Reserved

58.7.12 USBPHY VBUS Detect Control (USB1_VBUS_DETECT)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
USB1_VBUS_DETECT	C0h	USBPHY VBUS Detect Control
USB1_VBUS_DETECT_SET	C4h	Writing 1 to a bit in this register ensures that the corresponding bit in USB1_VBUS_DETECT is 1
USB1_VBUS_DETECT_CLR	C8h	Writing 1 to a bit in this register ensures that the corresponding bit in USB1_VBUS_DETECT is 0
USB1_VBUS_DETECT_TOG	CCh	Writing 1 to a bit in this register inverts the value of the corresponding bit in USB1_VBUS_DETECT

Function

Defines controls for USB VBUS detect and some additional out-of-band signaling functions.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EN_CHAR...	0			Reserved	DISCHAR...	0		PWRUP_CMPS				Reserved	VBUSVAL...	0	
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			ID_OVER...	ID_OVER...	VBUS_SOURCE_SEL		VBUSVAL...	VBUSVAL...	AVALID_...	BVALID_...	SESSEND...	VBUS_OV...	VBUSVALID_THRESH		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 EN_CHARGER_RESISTOR	Charger Resistor Enable Enables a nominal 125 kΩ pullup on DP and a nominal 375 kΩ pulldown on DN. This allows resistive battery charger detection according to *USB Battery Charging Specification Revision 1.0*, which is now obsolete. With this method, you perform the detection of the DN and DP pin states by using USB1_CHRG_DET_STAT[DN_STATE] and USB1_CHRG_DET_STAT[DP_STATE], respectively. This field must remain 0 for normal USB data communication, or when using the USBHSDCD module for battery charger detection, according to *USB Battery Charger Specification Revision 1.2* or any other detection mechanism for USB cable plugin. 0b - Disables 1b - Enables
30-28 —	Reserved
27 —	Reserved
26 DISCHARGE_VBUS	VBUS Discharge Resistor Controller Enables (controls) a nominal 22 kΩ resistor between the USB1_VBUS pin and the ground that can be used to accelerate the fall of the VBUS signal at the end of a session. 0b - Disables 1b - Enables
25-23 —	Reserved
22-20 PWRUP_CMPS	VBUS_VALID Comparator Enable Enables (powers up) the comparator used for the VBUS_VALID detector. You can reset this field to 0 to save power if the internal VBUS_VALID comparator is not used. The default value is 111b. 000b - Disable the VBUS_VALID comparator 001b - Enable the SESS_VALID comparator 010b - Enable 3V detection
19 —	Reserved
18 VBUSVALID_TO_SESSVALID	VBUS_VALID Comparator Selection Controls the comparator that is used to report the VBUS_VALID results of the VBUS_VALID comparator and session valid comparators, in USB1_VBUS_DETECT[ID_OVERRIDE_EN].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 0, the VBUS_VALID comparator for the VBUS_VALID results is used. If this field is 1, the session end comparator for the VBUS_VALID results is used. The session end threshold is > 0.8 V and < 4.0 V. The VBUS_VALID comparator is the most accurate and has a programmable threshold that VBUSVALID_THRESH defines. The session valid comparator may be useful in systems using nonstandard VBUS voltages. The mux selection in this field happens before any VBUS_VALID selection controlled by VBUSVALID_SEL and VBUS_SOURCE_SEL. 0b - VBUS_VALID comparator for the VBUS_VALID results 1b - Session end comparator for the VBUS_VALID results
17-13 —	Reserved
12 ID_OVERRIDE	ID Pin Status Local Override Value Specifies the ID pin status local override value. If ID_OVERRIDE_EN is 1, the value of this field is used for the IDDIG signal and also reported in STATUS[OTGID_STATUS]. This field has no impact if ID_OVERRIDE_EN becomes 0. See the description of STATUS[OTGID_STATUS] to understand the state assignment convention of the ID pin value.
11 ID_OVERRIDE_EN	Enable Local ID Pin Status Override Allows local override of the ID pin detection status. If this field is 0, USBPHY's ID pin detector for ID status is used. If this field is 1, the value of ID_OVERRIDE is used for the reported ID pin status. 0b - Do not allow override 1b - Allow override
10-9 VBUS_SOURCE_SEL	VBUS_VALID Source Selection Selects the source of the VBUS_VALID signal reported to the USB controller. If this field is 0b, by default, the VBUS_VALID comparator results for the signal reported to the USB controller are used. If this field is 1b or 10b, the session valid comparator results for the signal reported to the USB controller are used. The VBUS_VALID source selections in this field take effect only if both VBUSVALID_SEL and VBUS_OVERRIDE_EN are 0. This field does not impact the value of USB1_VBUS_DET_STAT[VBUS_VALID]. 00b - VBUS_VALID comparator results 01b - Session valid comparator results

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Session valid comparator result 11b - Reserved; do not use
8 VBUSVALID_SEL	VBUS_VALID Source Selection Selects the source of the VBUS_VALID signal reported to the USB controller. If this field is 0, by default, the VBUS_VALID comparator results for the signal reported to the USB controller are used. If this field is 1, the VBUS_VALID_3V detector results for the signal reported to the USB controller are used. The VBUS_VALID source selection in this field only takes effect if VBUS_VALID is 0 . This field does not impact the value of USB1_VBUS_DET_STAT[VBUS_VALID]. 0b - VBUS_VALID comparator results 1b - VBUS_VALID_3V detector results
7 VBUSVALID_OVERRIDE	Override Value for the VBUS_VALID Signal Provides the value of USB1_VBUS_DET_STAT[VBUS_VALID] reported to the USB controller if the value of VBUS_OVERRIDE_EN is 1. This field does not impact the value of USB1_VBUS_DET_STAT[VBUS_VALID]. 0b - Overridden to 0 1b - Overridden to 1
6 AVALID_OVERRIDE	Override Value for A-Device Session Valid Provides the value for USB1_VBUS_DET_STAT[AVALID] if the value of VBUS_OVERRIDE_EN is 1. 0b - Overridden to 0 1b - Overridden to 1
5 BVALID_OVERRIDE	Override Value for B-Device Session Valid Provides the value for USB1_VBUS_DET_STAT[BVALID] if the value of VBUS_OVERRIDE_EN is 1. 0b - Overridden to 0 1b - Overridden to 1
4 SESSEND_OVERRIDE	Override Value for SESSEND Provides the value for USB1_VBUS_DET_STAT[SESSEND] if the value of VBUS_OVERRIDE_EN is 1. 0b - Overridden to 0 1b - Overridden to 1
3 VBUS_OVERRIDE_EN	VBUS Detect Signal Override Enable Allows you to override the results from VBUS_VALID and session valid comparators using the values in USB1_VBUS_DETECT[7:4].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 0, by default, the results of the internal VBUS_VALID and session valid comparators for VBUS_VALID, AVALID, BVALID, and SESSEND are used. If this field is 1, the override values for VBUS_VALID, AVALID, BVALID, and SESSEND are used. The VBUS_VALID, AVALID, BVALID, and SESSEND signals sent to the USB controller are affected by these bit selections. The values reported for USB1_VBUS_DET_STAT[AVALID], USB1_VBUS_DET_STAT[BVALID], and USB1_VBUS_DET_STAT[SESSEND] are also affected but the value of USB1_VBUS_DET_STAT[VBUS_VALID] is not affected. This override method may be useful if you do not perform VBUS detection using the internal VBUS_VALID or session end comparators. 0b - Results of VBUS_VALID and session valid comparators for VBUS_VALID, AVALID, BVALID, and SESSEND 1b - Override values for VBUS_VALID, AVALID, BVALID, and SESSEND
2-0 VBUSVALID_THRESH	VBUSVALID Threshold Sets the threshold for the VBUSVALID comparator, which is the most accurate method to determine the presence of 5 V, and includes hysteresis to minimize the need for software debounce of the detection. This comparator has ~50 mV of hysteresis to prevent chattering at the comparator trip point. 000b - 4.0 V 001b - 4.1 V 010b - 4.2 V 011b - 4.3 V 100b - 4.4 V 101b - 4.5 V 110b - 4.6 V 111b - 4.7 V

58.7.13 USBPHY VBUS Detector Status (USB1_VBUS_DET_STAT)

Offset

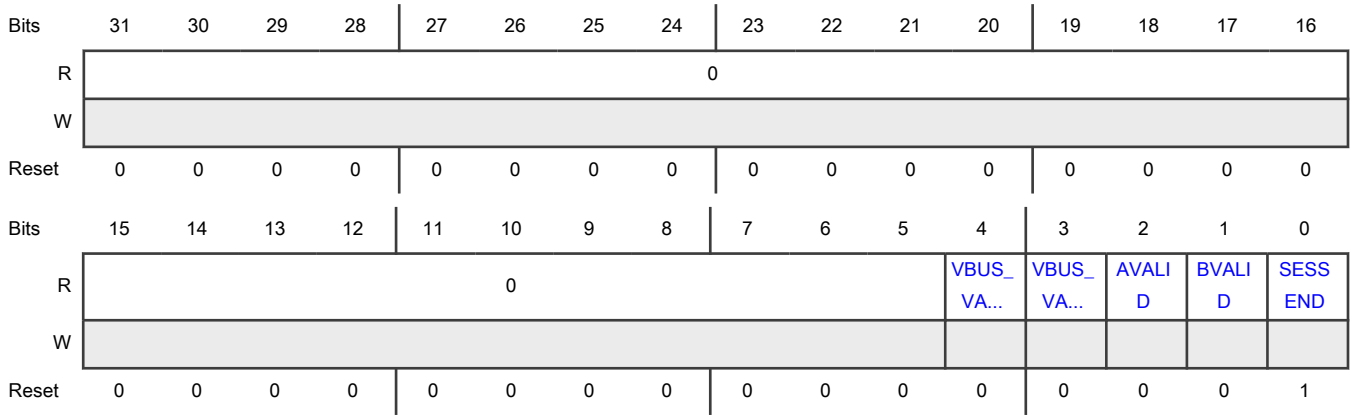
Register	Offset
USB1_VBUS_DET_STAT	D0h

Function

Allows status observation for USB VBUS detect functions.

The APB clock synchronizes the values reported in this register.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 VBUS_VALID_3V	VBUS_VALID_3V Detector Status Indicates the status of the VBUS_VALID_3V detector. If this field is 0, the VBUS voltage is below the VBUS_VALID_3V threshold. If this field is 1, the VBUS voltage is above the VBUS_VALID_3V threshold. The VBUS_VALID_3V detector has a lower threshold for the voltage on the USB1_VBUS pin than either the session valid or VBUS_VALID comparators. This signal may be useful for applications using a nonstandard VBUS voltage. 0b - Below threshold 1b - Above threshold
3 VBUS_VALID	VBUS Voltage Status Indicates the result of VBUS_VALID detection for the USB1_VBUS pin. If this field is 0, the VBUS voltage is below the comparator threshold. If this field is 1, the VBUS voltage is above the comparator threshold. USB1_VBUS_DETECT[VBUSVALID_TO_SESSVALID] selects the VBUS_VALID comparator used. You cannot overwrite the VBUS_VALID source, which is not affected by the values of USB1_VBUS_DETECT[VBUSVALID_OVERRIDE] and USB1_VBUS_DETECT[VBUS_OVERRIDE_EN]. 0b - Below threshold 1b - Above threshold
2 AVALID	A-Device Session Valid Status Indicates A-device session valid status that the session valid comparator determines. If this field is 0, the VBUS voltage is below the session valid threshold.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 1, the VBUS voltage is above the session valid threshold. The voltage on the USB1_VBUS pin determines the default value of this field. You can overwrite this field by using USB1_VBUS_DETECT[AVALID_OVERRIDE] and USB1_VBUS_DETECT[VBUS_OVERRIDE_EN]. 0b - Below threshold 1b - Above threshold
1 BVALID	B-Device Session Valid Status Indicates B-device session valid status that the session valid comparator determines. If this field is 0, the VBUS voltage is below the session valid threshold. If this field is 1, the VBUS voltage is above the session valid threshold. The voltage on the USB1_VBUS pin determines the default value of this field. You can overwrite this field by using USB1_VBUS_DETECT[BVALID_OVERRIDE] and USB1_VBUS_DETECT[VBUS_OVERRIDE_EN]. 0b - Below threshold 1b - Above threshold
0 SESSEND	Session End Indicator Indicates session end status, the value inverted from the session valid comparator. If this field is 0, the VBUS voltage is above the session valid threshold. If this field is 1, the VBUS voltage is below the session valid threshold. The voltage on the USB1_VBUS pin determines the default value of this field. You can overwrite this field by using USB1_VBUS_DETECT[SESSEND_OVERRIDE] and USB1_VBUS_DETECT[VBUS_OVERRIDE_EN]. 0b - Above threshold 1b - Below threshold

58.7.14 USBPHY Charger Detect Control (USB1_CHRG_DETECT)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
USB1_CHRG_DETECT	E0h	USBPHY Charger Detect Control
USB1_CHRG_DETECT_SET	E4h	Writing 1 to a bit in this register ensures that the corresponding bit in USB1_CHRG_DETECT is 1

Table continues on the next page...

Table continued from the previous page...

Register	Offset	Description
USB1_CHRG_DETECT_CLR	E8h	Writing 1 to a bit in this register ensures that the corresponding bit in USB1_CHRG_DETECT is 0
USB1_CHRG_DETECT_TOG	ECh	Writing 1 to a bit in this register inverts the value of the corresponding bit in USB1_CHRG_DETECT

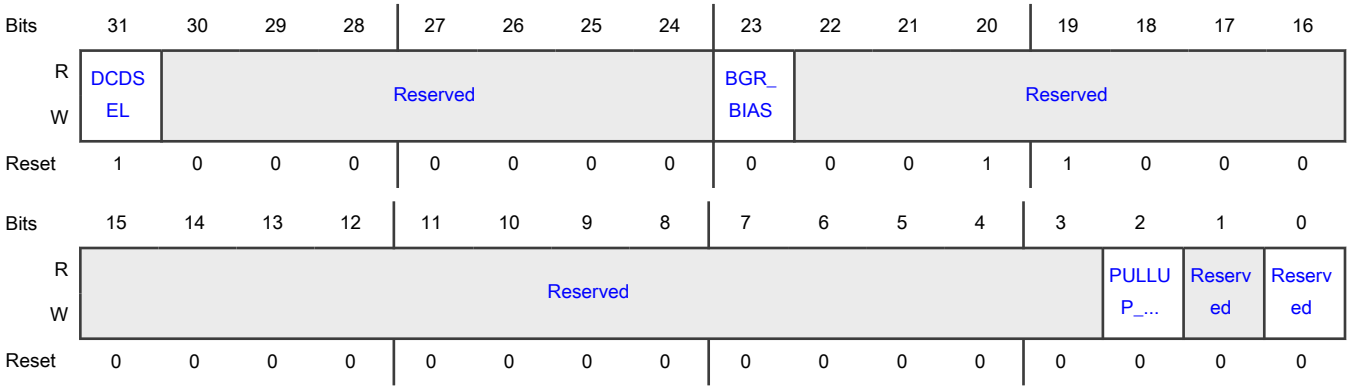
Function

Defines controls for USB battery charging detection functions.

USBDCD controls the USB battery charging detection functions (see the "USB Device Charger Detection Module (USBDCD)" chapter for more information).

You must not change the values of this field except for characterization tasks.

Diagram



Fields

Field	Function
31 DCDSEL	DCD Selection Selects the control source for battery charging detection. If this field is 0, USBPHY Charger Detect Control (USB1_CHRG_DETECT) controls the BC 1.2 functionality. If this field is 1, fields and state machines in USBHSDCD control the BC 1.2 functionality (see the "USB Device Charger Detection Module (USBDCD)" chapter for more information). You can control several of the USBPHY analog circuits associated with the USB battery charging v1.2 functions by either using control fields in USBPHY Charger Detect Control (USB1_CHRG_DETECT) or by using the USBHSDCD module's registers. Use of USBHSDCD is recommended for software simplification, for device mode charger detection because USBHSDCD can auto-sequence through the detection states. 0b - Fields in the USB1_CHRG_DETECT register 1b - Fields and state machines in the USBHSDCD module

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-24 —	Reserved
23 BGR_BIAS	BGR Bias Selects the bias method deployed for the voltage and current sources used for the analog circuits associated with USB battery charging detection in the USBHSDCD instantiation of USBDCD (see the "USB Device Charger Detection Module (USBDCD)" chapter for more information). If this field is 0, by default, the local bias powered from VBUS for 10 µA reference is used. If this field is 1, the bandgap bias powered from VREGIN0 or VREGIN1 for 10 µA reference is used. The V_{DP_SRC} and V_{DN_SRC} bias is generated from the USB1_VBUS voltage, independent of the value of this field. 0b - Local bias 1b - Bandgap bias
22-3 —	Reserved
2 PULLUP_DP	DP Pullup Resistor Enable Override Control Enables FS Device mode operation that is normally controlled using the UTMI bus signals that the USB controller sends to USBPHY. If this field is 1, the pullup resistor is enabled independently of the UTMI signals to the PHY. The USB software stack uses this capability during charger type detection, according to the obsolete *USB Battery Charging Specification Revision 1.1*. For USB battery charging detection and normal USB data communication (associated with the USB battery charging v1.2), retain 0 as the value of this field. 0b - Disables 1b - Enables
1 —	Reserved
0 —	This value of this bit must remain 0b to avoid interference with USB battery charging detection.

58.7.15 USBPHY Charger Detect Status (USB1_CHRG_DET_STAT)

Offset

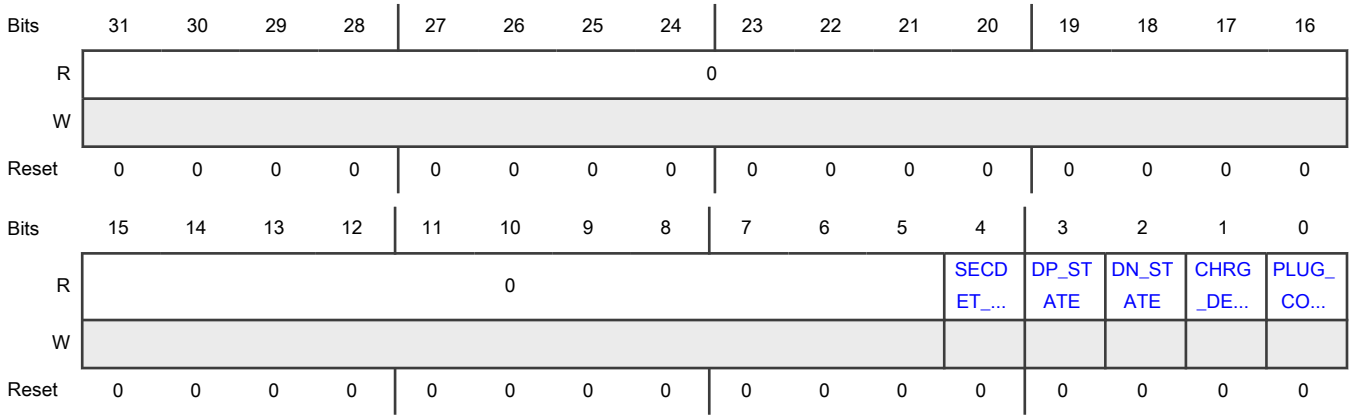
Register	Offset
USB1_CHRG_DET_STAT	F0h

Function

Indicates the USBPHY charger detect status.

Use the USBHSDCD registers for USB battery charging detection functions and standards-based charger detection purposes.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 SECDET_DCP	Battery Charging Secondary Detection Phase Output Indicates the type of charging port detected—charging downstream port (CDP) or downstream charging port (DCP)—during the USB battery charging secondary detection phase, using the USBHSDCD module. 0b - CDP detected 1b - DCP detected
3 DP_STATE	DP State Indicates the single-ended receiver output for the DP pin, from charger detection circuits. 0b - < 0.8 V 1b - > 2.0 V
2 DN_STATE	DN State Indicates the single-ended receiver output for the DN pin, from charger detection circuits. 0b - < 0.8 V 1b - > 2.0 V
1 CHRG_DETECTED	Battery Charging Primary Detection Phase Output Indicates whether a standard downstream port (SDP) or charging port is detected during the USB battery charging primary detection phase.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - SDP detected 1b - Charging port detected
0 PLUG_CONTACT	Battery Charging Data Contact Detection Phase Output Indicates whether an attached USB cable is detected between the remote host and the local device during the data contact detection phase, according to *USB Battery Charging Specification Revision 1.2* (using the USBHSDCD module). 0b - Not detected 1b - Detected

58.7.16 USBPHY Analog Control (ANACTRL)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
ANACTRL	100h	USBPHY Analog Control
ANACTRL_SET	104h	Writing 1 to a bit in this register ensures that the corresponding bit in ANACTRL is 1
ANACTRL_CLR	108h	Writing 1 to a bit in this register ensures that the corresponding bit in ANACTRL is 0
ANACTRL_TOG	10Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in ANACTRL

Function

Includes fields for adding pre-emphasis to the USB HS TX output drivers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	Reserved			Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved					DEV_PUL...	Reserved									Reserved
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31 —	Reserved
30-28 —	Reserved
27 —	This field must remain at value 0b.
26 —	This field must remain at value 0b.
25 —	This field must remain at value 1b.
24 —	This field must remain at value 0b.
23 —	This field must remain at value 0b.
22 —	This field must remain at value 0b.
21 —	This field must remain at value 0b.
20 —	This field must remain at value 0b.
19 —	This field must remain at value 0b.
18 —	This field must remain at value 0b.
17 —	This field must remain at value 0b.
16 —	This field must remain at value 0b.
15-11	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
10 DEV_PULLDOWN	Device Pulldown Enables the 15 kΩ pulldown resistors on both DP and DN pins in Device mode. You can use this feature in Device mode, while the USB cable is disconnected, to retain the data pins at known values, avoiding unnecessary interrupts from single-ended receivers. This field must be reset to 0 during normal USB data communication in Device mode, or during battery charger detection, using the USBHSDCD module. 0b - Disables 1b - Enables
9-1 —	Reserved
0 —	The value of this reserved bit must remain 0b for normal operation.

58.7.17 USBPHY Loopback Control and Status (USB1_LOOPBACK)

Offset

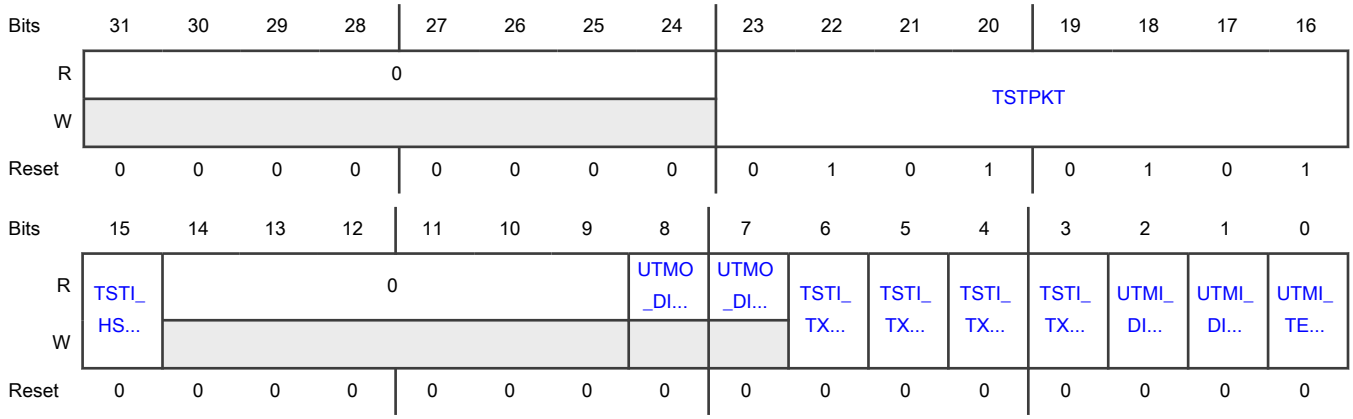
This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
USB1_LOOPBACK	110h	USBPHY Loopback Control and Status
USB1_LOOPBACK_SET	114h	Writing 1 to a bit in this register ensures that the corresponding bit in USB1_LOOPBACK is 1
USB1_LOOPBACK_CLR	118h	Writing 1 to a bit in this register ensures that the corresponding bit in USB1_LOOPBACK is 0
USB1_LOOPBACK_TOG	11Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in USB1_LOOPBACK

Function

Controls loopback testing for USBPHY. Loopback mode is for test purposes only; you cannot use it for normal USB data communication.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 TSTPKT	Testing Packet Selects the packet data byte used for USB loopback testing in Pulse mode. Pulse mode is selected if you write 01b to UTMI_DIG_TST0 and UTMI_DIG_TST1. The default value produces a data inversion at each unit interval.
15 TSTI_HSFS_MODE_EN	Loopback Test HS-FS Mode Enable Enables the loopback test to dynamically change the packet speed. It sends a number of HS packets determined by USB1_LOOPBACK_HSFSCNT[TSTI_HS_NUMBER], followed by a number of FS packets determined by USB1_LOOPBACK_HSFSCNT[TSTI_FS_NUMBER]. 0b - Disables 1b - Enables
14-9 —	Reserved
8 UTMO_DIG_TST1	UTMO Digital Test 1 Indicates USB loopback test results. Test results are only valid when UTMI_TESTSTART is 1. 0b - Not passing 1b - Passing
7 UTMO_DIG_TST0	UTMO Digital Test 0 Indicates USB loopback test results. Test results are only valid when UTMI_TESTSTART is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Passing 1b - Not passing
6 TSTI_TX_HIZ	Loopback Test Transmit Hi-Z Enables TX Hi-Z for USB loopback test. 0b - Disables 1b - Enables
5 TSTI_TX_EN	Loopback Test Transmit Enable Enables TX for USB loopback test. 0b - Disables 1b - Enables
4 TSTI_TX_LS_MODE	Loopback Test LS Mode Selects the TX mode. 0b - HS or FS (defined by TSTI1_TX_HS) 1b - LS
3 TSTI_TX_HS_MODE	Loopback Test HS Mode Selects HS or FS mode for USB loopback testing. 0b - FS 1b - HS
2 UTMI_DIG_TST1	UTMI Digital Test 1 Selects the mode for USB loopback test 1. If this field is 0, Pulse mode for the loopback test is selected. This field remains 0 when UTMI_DIG_TST0 is 1. If this field is 1, Pseudorandom mode for the loopback test is selected. This field becomes 1 when UTMI_DIG_TST0 is 0. 0b - Pulse mode 1b - Pseudorandom mode
1 UTMI_DIG_TST0	UTMI Digital Test 0 Selects the mode for USB loopback test 0. If this field is 0, Pseudorandom mode for the loopback test is selected. This field remains 0 when UTMI_DIG_TST1 is 1. If this field is 1, Pulse mode for the loopback test is selected. This field becomes 1 when UTMI_DIG_TST1 is 0. 0b - Pseudorandom mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Pulse mode
0 UTMI_TESTSTART	UTMI Test Start Enables the USB loopback test. 0b - Disables 1b - Enables

58.7.18 USBPHY Loopback Packet Number Selection (USB1_LOOPBACK_HSFSCNT)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

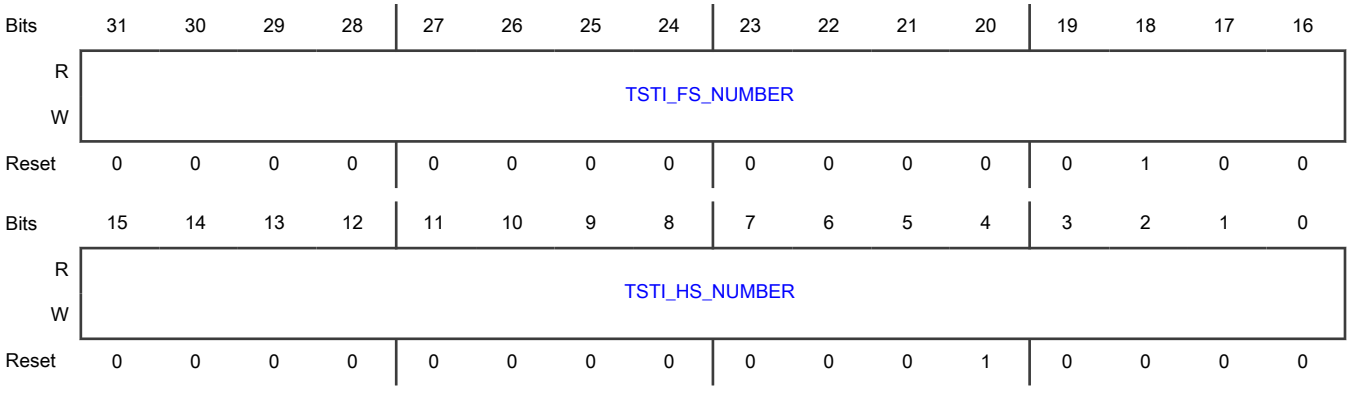
Register	Offset	Description
USB1_LOOPBACK_HSFSCNT	120h	USBPHY Loopback Packet Number Selection
USB1_LOOPBACK_HSFSCNT_SET	124h	Writing 1 to a bit in this register ensures that the corresponding bit in USB1_LOOPBACK_HSFSCNT is 1
USB1_LOOPBACK_HSFSCNT_CLR	128h	Writing 1 to a bit in this register ensures that the corresponding bit in USB1_LOOPBACK_HSFSCNT is 0
USB1_LOOPBACK_HSFSCNT_TOG	12Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in USB1_LOOPBACK_HSFSCNT

Function

Selects the packet counts used for the specific loopback test mode that dynamically alternates the transmit speed.

See USBPHY Loopback Control and Status (USB1_LOOPBACK) for other configuration of this mode.

Diagram



Fields

Field	Function
31-16 TSTI_FS_NUMBER	Loopback Test FS Packet Number Specifies the FS packet number, used when USB1_LOOPBACK[TSTI_HSFS_MODE_EN] is 1.
15-0 TSTI_HS_NUMBER	Loopback Test HS Packet Number Specifies the HS packet number, used when USB1_LOOPBACK[TSTI_HSFS_MODE_EN] is 1.

58.7.19 USBPHY Trim Override Enable (TRIM_OVERRIDE_EN)

Offset

This type of register has supplemental _SET, _CLR, and _TOG registers at adjacent offsets.

Register	Offset	Description
TRIM_OVERRIDE_EN	130h	USBPHY Trim Override Enable
TRIM_OVERRIDE_EN_SET	134h	Writing 1 to a bit in this register ensures that the corresponding bit in TRIM_OVERRIDE_EN is 1
TRIM_OVERRIDE_EN_CLR	138h	Writing 1 to a bit in this register ensures that the corresponding bit in TRIM_OVERRIDE_EN is 0
TRIM_OVERRIDE_EN_TOG	13Ch	Writing 1 to a bit in this register inverts the value of the corresponding bit in TRIM_OVERRIDE_EN

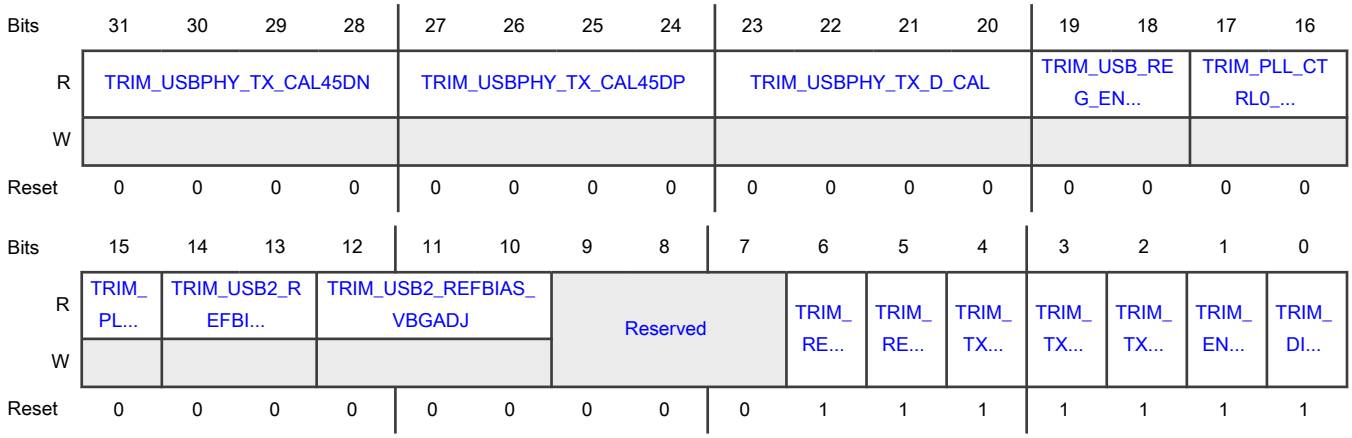
Function

Allows observation of the default IFR settings of the PHY parameters for the following fields:

- TRIM_USBPHY_TX_CAL45DN
- TRIM_USBPHY_TX_CAL45DP
- TRIM_USBPHY_TX_D_CAL
- TRIM_USB_REG_ENV_TAIL_ADJ_VD
- TRIM_PLL_CTRL0_DIV_SEL

Additional fields also determine whether you can overwrite those values by using the settings defined in USBPHY Transmitter Control (TX), UTMI Debug Status 1 (DEBUG1), and USBPHY PLL Control and Status (PLL_SIC).

Diagram



Fields

Field	Function
31-28 TRIM_USBPHY_TX_CAL45DN	DN Series Termination Resistance Trim Bits from Outside USBPHY Trims the termination resistance if TRIM_TX_CAL45DN_OVERRIDE is 0. If TRIM_TX_CAL45DN_OVERRIDE is 1, the resistance trim values in TXCAL45DN are used instead. Decode to trim the nominal 45 Ω series termination resistance to the DN output pin, with values loaded using signals outside USBPHY. The methods used to load the read-only values into this field during reset and boot are chip-specific. The maximum resistance is at value 0000b and minimum resistance is at value 1111b (resistance is centered at value 0111b). For USBPHY, each incrementally increasing trim value decreases the target resistance by approximately 3.5%, compared to the next lower value. Trimming this resistance impacts both the overshoot and undershoot of the FS TX output and the amplitude of the HS TX output. The methods used to load the read-only values into this field during reset and boot are chip-specific.
27-24 TRIM_USBPHY_TX_CAL45DP	DP Series Termination Resistance Trim Bits from Outside USBPHY Trims the termination resistance when TRIM_TX_CAL45DP_OVERRIDE is 0. If TRIM_TX_CAL45DP_OVERRIDE is 1, the resistance trim values in TX[TXCAL45DP] are used instead. Decode to trim the nominal 45 Ω series termination resistance to the DP output pin, with values loaded using signals outside USBPHY. The methods used to load the read-only values into this field during reset and boot are chip-specific. The maximum resistance is 0000b and minimum resistance is 1111b (the resistance is centered at value 0111b). For USBPHY, each incrementally increasing trim value decreases the target resistance by approximately 3.5%, compared to the next lower value. Trimming this resistance impacts both the overshoot and undershoot of the FS TX output and amplitude of the HS TX output. The methods used to load the read-only values into this field during reset and boot are chip-specific.
23-20	HS TX Output Current Trim Bits from Outside USBPHY

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRIM_USBPHY_TX_D_CAL	Trims the output current if TRIM_TX_D_CAL_OVERRIDE is 0. In this case, the current trim values in TX[D_CAL] are used instead. Decode to trim the nominal 17.78 mA current source for the HS TX drivers on DP and DN, with values loaded using signals from outside the PHY. This current is directly proportional to the amplitude of the HS TX eye diagram. The methods used to load the read-only values into this field during reset and boot are chip-specific. 0000b - Maximum current; approximately 19% above nominal 0111b - Nominal 1111b - Minimum current; approximately 19% below nominal
19-18 TRIM_USB_REG_ENV_TAIL_ADJ_VD	HS RX Squelch Rise Time Delay Trim Bits from Outside USBPHY Allows you to adjust the squelch rise time if TRIM_ENV_TAIL_ADJ_VD_OVERRIDE is 0. In this case, the squelch rise time trim values in DEBUG1[ENTAILADJVD] are used instead. This field defines the delay adjustment for the rise time of the squelch signal from the HS RX envelope detector, with values loaded using signals from outside the PHY. The methods used to load the read-only values into this field during reset and boot are chip-specific. See DEBUG1[ENTAILADJVD] to understand the available trims.
17-15 TRIM_PLL_CTRL0_DIV_SEL	PLL Divider Value Configuration Bits from Outside USBPHY Controls the PLL feedback loop divider if TRIM_DIV_SEL_OVERRIDE is 0. USBPHY's PLL produces a 480 MHz output clock. This field allows the use of different frequency signals for the PLL reference clock input connected to the OSCCLK signal from the system oscillator, with values loaded from outside USBPHY. If TRIM_DIV_SEL_OVERRIDE is 0, the PLL divider values in PLL_SIC[PLL_DIV_SEL] are used instead. See the field value descriptions for PLL_SIC[PLL_DIV_SEL] to understand the available input frequency settings. The methods used to load the read-only values into this field during reset and boot are chip-specific.
14-13 TRIM_USB2_REFBIAS_TST	Bias Current Control Adjustment Bits from Outside USBPHY Trims the PHY's current reference when TRIM_REFBIAS_TST_OVERRIDE is 0. If TRIM_REFBIAS_TST_OVERRIDE is 1, the current trim values in DEBUG1[USB2_REFBIAS_TST] are used instead. This field defines the main current trim for the USBPHY bias module current reference generator, with values loaded using signals from outside the PHY.
12-10 TRIM_USB2_REFBIAS_VBGADJ	Bandgap Voltage Adjustment Bits from Outside USBPHY Trims the PHY's bandgap voltage if TRIM_REFBIAS_VBGADJ_OVERRIDE is 0. In this case, the voltage trim values in DEBUG1[USB2_REFBIAS_VBGADJ] are used instead. This field defines voltage adjustment trim on the USBPHY bias module bandgap generator, with values loaded using signals from outside the PHY.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-7 —	The value of this reserved bit must remain 0b.
6 TRIM_REFBIAS_TST_OVERRIDE	Override Enable for Bias Current Control Enables override for bias current control. If this field is 1, the value of DEBUG1[USB2_REFBIAS_TST] is used. 0b - Disables 1b - Enables
5 TRIM_REFBIAS_VBGADJ_OVERRIDE	Override Enable for Bandgap Voltage Adjustment Enables override for bandgap voltage adjustment. If this field is 1, the value of DEBUG1[USB2_REFBIAS_VBGADJ] is used. 0b - Disables 1b - Enables
4 TRIM_TX_CAL45DN_OVERRIDE	Override Enable for DN Series Termination Trim Enables override for the DN series termination trim. If this field is 1, the value of TX[TXCAL45DN] is used. 0b - Disables 1b - Enables
3 TRIM_TX_CAL45DP_OVERRIDE	Override Enable for DP Series Termination Trim Enables override for the DP series termination trim. If this field is 1, the value of TX[TXCAL45DP] is used. 0b - Disables 1b - Enables
2 TRIM_TX_D_CAL_OVERRIDE	Override Enable for the HS TX Output Current Trim Enables override for HS TX output current trim. If this field is 1, the value of TX[D_CAL] is used. 0b - Disables 1b - Enables
1 TRIM_ENV_TAIL_ADJ_VD_OVERRIDE	Override Enable for HS RX Squelch Rise Time Delay Trim Enables override for the HS RX squelch rise time delay trim. If this field is 1, the value of DEBUG1[ENTAILADJVD] is used. 0b - Disables

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enables
0 TRIM_DIV_SEL_OVERRIDE	Override Enable for PLL Divider Value Enables override for the PLL divider value. If this field is 1, the value of PLL_SIC[PLL_DIV_SEL] is used. 0b - Disables 1b - Enables

Chapter 59
CAN (FlexCAN)

59.1 Chip-specific FlexCAN information

Table 927. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

In doze mode, If you want just CM33 to control the doze, mask CM7 by writing the corresponding bit in GPR_CTRL register in CCM module. Then upon detection of a recessive to dominant transition on the CAN bus, it generates a Wake Up interrupt to the CPU.

59.2 Overview

FlexCAN is a communication controller implementing the CAN protocol according to the ISO 11898-1:2015 standard and CAN 2.0 Part B protocol specifications.

The CAN protocol was primarily designed to be used as a vehicle serial data bus, meeting the specific real-time processing and reliable operation requirements in the electromagnetic interference (EMI) environment of a vehicle. FlexCAN is a full implementation of the CAN protocol specification, the CAN with flexible data rate (CAN FD) protocol, and the CAN version 2.0 Part B protocol. It supports both standard and extended message frames and long payloads.

NOTE

Legacy Receive (RX) FIFO cannot be used in Flexible Data (FD) mode.

NOTE

In CAN FD mode, use the Enhanced Receive FIFO feature instead of the Legacy Receive FIFO.

NOTE

FD mode is limited to Message Buffer (MB) mode only, the Enhanced Receive FIFO with DMA is supported.

59.2.1 Block diagram

Figure 437 shows the main submodules implemented in FlexCAN.

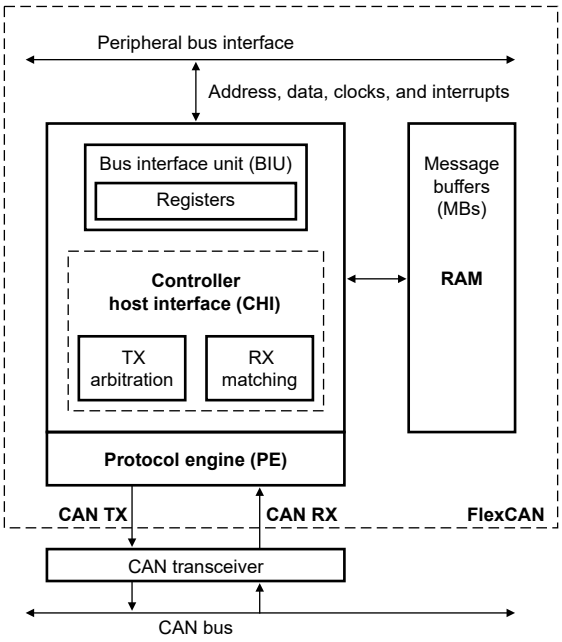


Figure 437. Block diagram

The Protocol Engine (PE) submodule manages the serial communication on the CAN bus:

- RAM access for receiving and transmitting message frames
- Receive message validation
- Error handling.
- CAN FD message detection

The Controller Host Interface (CHI) submodule manages:

- Message buffer (MB) selection for reception and transmission
- Arbitration and ID matching algorithms for both CAN FD and non-CAN FD message formats

The Bus Interface Unit (BIU) submodule controls access to and from the internal interface bus to establish connection to the CPU and to other blocks. The BIU manages access to:

- Clocks
- Address and data buses
- Interrupt outputs
- DMA
- Test signals

59.2.2 Features

- Full implementation of *CAN with Flexible data rate (CAN FD) protocol specification* and *CAN Specification Version 2.0, Part B*
  - Standard data frames
  - Extended data frames
  - Data length of 0–64 bytes
  - Content-related addressing

- Compliance with ISO 11898-1:2015 standard
- Flexible message buffers that can be configured to store a payload of 0, 8, 16, 32, or 64 bytes
  - Increasing the payload size decreases the number of supported message buffers (see FlexCAN memory partition for CAN FD).
  - Message buffers are configurable as receive or transmit, supporting standard and extended messages.
- Individual Receive Mask registers for each message buffer
- Full-featured Legacy RX FIFO with storage capacity for six frames and automatic internal pointer handling with DMA support
- Full-featured Enhanced RX FIFO with storage capacity of 20 CAN FD frames and automatic internal pointer handling with DMA support
- Transmission abort capability
- Programmable clock source to CAN Protocol Interface, either peripheral clock or oscillator clock
- Optional general purpose RAM space, using RAM not used by reception or transmission structures
- Listen-Only mode
- Programmable loopback mode supporting self-test operation
- Programmable transmission priority scheme: lowest ID, lowest buffer number, or highest priority
- Timestamp based on 32-bit free-running timer, with optional external time tick
- Global network time, synchronized by specific message
- Maskable interrupts
- Independence from transmission medium (external transceiver is assumed)
- Short latency time due to arbitration scheme for high-priority messages
- Low-power modes, with programmable wake-up on bus activity
- Transceiver delay compensation when transmitting CAN FD messages at faster data rates
- Management of remote request frames, automatically or by software
- Restriction only to write CAN bit time settings and configuration bits in Freeze mode
- Transmit message buffer status (lowest-priority buffer or empty buffer)
- Identifier Acceptance Filter Hit Indicator (IDHIT) register for received frames
- ESR1[SYNCH] to indicate module is synchronous with CAN bus
- CRC status for transmitted message
- Legacy RX FIFO Global Mask register
- Selectable priority between message buffers and receive FIFO during matching process
- Powerful Legacy RX FIFO ID filtering, capable of matching incoming IDs against either 128 extended IDs, 256 standard IDs, or 512 partial (8-bit) IDs, with 32 individual masking capability
- Powerful Enhanced RX FIFO ID filtering, capable of matching incoming IDs against either 64 extended or 128 standard ID filter elements with three filtering schemes: mask plus filter, range, and two filters without mask.
- Complete backward compatibility with previous FlexCAN version
- Detection and correction of errors in memory read accesses.
  - Each byte of FlexCAN memory is associated to five parity bits.
  - The error correction mechanism ensures that errors in one bit of this 13-bit word can be corrected (correctable errors).

— Errors in two bits can be detected but not corrected (noncorrectable errors).

59.3 Functional description

FlexCAN is a CAN protocol engine with a flexible message buffer system for transmitting and receiving CAN frames. The system is a set of message buffers (MBs) that stores configuration and control data, timestamp, message ID, and data (see Message buffer structure).

For classical CAN frames, FlexCAN supports simultaneous reception through Legacy FIFO and message buffer. For CAN FD frames, FlexCAN supports reception through message buffer and enhanced receive FIFO.

For message buffer reception, a matching algorithm makes it possible to store received frames only into MBs that have the same ID programmed in the ID field. A masking scheme makes it possible to match the ID programmed on the message buffer with a range of IDs on received CAN frames. For transmission, an arbitration algorithm decides the prioritization of message buffers to be transmitted based on the message ID (optionally augmented by three local priority bits) or the message buffer ordering.

A message buffer is active at a given time if it can participate in both the matching and arbitration processes. A receive message buffer with a 0b code is inactive (see Table 965). A transmit message buffer with a 1000b or 1001b code is also inactive (see Table 966).

FlexCAN can receive and transmit messages in CAN FD format. The message buffers are sized to store the quantity of data bytes selected by FDCTRL[MBDSRn]. The quantity of FD message buffers available for a given quantity of data bytes is described in the CAN FD Control (FDCTRL) register. See also FlexCAN memory partition for CAN FD.

59.3.1 Modes of operation

FlexCAN has these functional modes:

Table 928. Functional modes

Mode	Description
Normal (User or Supervisor)	In Normal mode, FlexCAN receives and transmits message frames, manages errors normally, and enables all CAN Protocol functions. User and Supervisor modes differ in the access to some restricted control registers.
Freeze	Freeze mode is enabled when MCR[FRZ] = 1. If enabled, FlexCAN enters Freeze mode when MCR[HALT] is 1 or when Debug mode is requested at chip level and FlexCAN writes 1 to MCR[FRZACK]. In this mode, no transmission or reception of frames is done, and synchronicity to the CAN bus is lost. See Freeze mode.
Loopback	FlexCAN enters this mode when CTRL1[LPB] becomes 1. In this mode, FlexCAN performs an internal loopback that can be used for self-test. The bit stream output of the transmitter is internally fed back to the receiver input. The receiving CAN input pin is ignored and the transmitting CAN output goes to the recessive state (logic 1). FlexCAN behaves as it normally does when transmitting and treats its own transmitted message as a message received from a remote node. To ensure proper reception of its own message, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field. Both transmit and receive interrupts are generated.
Listen-Only	FlexCAN enters this mode when CTRL1[LOM] becomes 1. In this mode, transmission is disabled, all error counters are frozen, and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station are received. If FlexCAN detects an unacknowledged message, it flags a BIT0 error (without changing the REC), as if it was trying to acknowledge the message.
CAN FD Active	In this mode, FlexCAN can transmit and receive all messages formatted according to the CAN FD Standard (2.0) and 2.0B Protocol in an interleaved fashion. The CPU can put FlexCAN into CAN FD Active mode by configuring MCR[FDEN] in Freeze mode.

Some features available in classical CAN are unavailable in CAN FD mode.

Table 929. Differences between classical CAN and CAN FD

Feature	Classical CAN	CAN FD
Legacy RX FIFO DMA	Yes	No
Legacy RX FIFO	Yes	No
Enhanced RX FIFO DMA	Yes	Yes
Enhanced RX FIFO	Yes	Yes

FlexCAN can operate in these low-power modes:

Table 930. Low-power modes

Mode	Description
Module Disable	FlexCAN enters this mode when the CPU writes 1 to MCR[MDIS] and FlexCAN writes 1 to MCR[LPMACK]. After FlexCAN is disabled, it issues a request to disable the clocks to the CAN Protocol Engine and Controller Host Interface submodules. Writing 0 to MCR[MDIS] exits this mode. See Module Disable mode.
Doze	FlexCAN enters this mode when MCR[DOZE] is 1, Doze mode is requested at chip level, and FlexCAN writes 1 to MCR[LPMACK]. When in Doze mode, FlexCAN issues a request to disable the clocks to the CAN Protocol Engine and the CAN Controller-Host Interface submodules. This mode is exited when: • MCR[DOZE] becomes 0 • The chip is removed from Doze mode • Or activity is detected on the CAN bus and the Self-Wake-up mechanism is enabled. See Doze mode.
Stop	FlexCAN enters this mode when Stop mode is requested at chip level and FlexCAN writes 1 to MCR[LPMACK]. When in Stop mode, FlexCAN puts itself in an inactive state and then informs the CPU that the clocks can be shut down globally. Exit from this mode occurs when the Stop mode request is removed, or when activity is detected on the CAN bus and the Self-Wake-up mechanism is enabled. See Stop mode.

59.3.1.1 Modes of operation details

FlexCAN has functional modes and low-power modes. See Modes of operation for an introductory description of all the modes of operation. The following subsections contain functional details about Freeze mode and low-power modes.

CAUTION

FlexCAN does not support "Permanent Dominant" failure on the CAN bus line. If a Low-Power request or Freeze mode request occurs during a "Permanent Dominant" condition, the corresponding acknowledgment field can never be 1.

59.3.1.1.1 Freeze mode

This mode is requested either by the CPU writing 1 to MCR[HALT] or when the chip is put into Debug mode. MCR[FRZ] must be 1 and the module must not be in a low-power mode.

When MECR[NCEFAFRZ] becomes 1 and a noncorrectable error is detected in a memory read access performed by FlexCAN internal processes (see Error response), FlexCAN also requests Freeze mode. This request occurs via both MCR[HALT] and MCR[FRZ] automatically becoming 1.

To obtain acknowledgment, FlexCAN writes 1 to MCR[FRZACK]. The CPU must only consider FlexCAN to be in Freeze mode when both request and acknowledgment conditions are satisfied.

When Freeze mode is requested, FlexCAN:

1. Waits to be in either Intermission, Error Passive, Bus Off, or Idle state.
2. Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. A pending move-in does not prevent entering Freeze mode.
3. Ignores the receive input pin and drives the transmit pin as recessive.
4. Stops the prescaler, halting all CAN protocol activities.
5. Grants write access to the Error Counters register, which is read-only in other modes.
6. Writes 1 to MCR[NOTRDY] and MCR[FRZACK].

After requesting Freeze mode, you must wait for MCR[FRZACK] to become 1 before executing any other action, otherwise FlexCAN may operate unpredictably. In Freeze mode, all memory-mapped registers are accessible, except for CTRL1[CLKSRC], which can be read but cannot be written.

Freeze mode is exited in one of these conditions:

- CPU writes 0 to MCR[FRZ].
- The chip is removed from Debug mode or the MCR[HALT] becomes 0.

MCR[FRZACK] becomes 0 after the protocol engine recognizes the negation of the freeze request. After leaving Freeze mode, FlexCAN tries to resynchronize to the CAN bus by waiting for 11 consecutive recessive bits.

59.3.1.1.2 Module Disable mode

This low-power mode is normally used to disable a complete FlexCAN block temporarily, with no power consumption. The CPU requests this mode by writing 1 to MCR[MDIS], and FlexCAN acknowledges the request by writing 1 to MCR[LPMACK]. The CPU must only consider FlexCAN to be in Disable mode when both the request and acknowledgment conditions are satisfied.

If FlexCAN is disabled during Freeze mode, the module requests to disable the clocks to the PE and CHI submodules, writes 1 to MCR[LPMACK] and writes 0 to MCR[FRZACK].

If the module is disabled during transmission or reception, FlexCAN:

1. Waits to be in either Idle or Bus Off state, or waits for the third bit of Intermission and then checks that it is recessive.
2. Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. FlexCAN does not take a pending move-in into account.
3. Ignores its receive input pin and drives its transmit pin as recessive.
4. Shuts down the clocks to the PE and CHI submodules.
5. Writes 1 to MCR[NOTRDY] and MCR[LPMACK].

In this mode, the Bus Interface Unit continues to operate, enabling the CPU to access memory-mapped registers, except for:

- The Receive Message Buffers Global Mask registers
- The Receive Buffer 14 Mask register
- The Receive Buffer 15 Mask register

When in Disable mode, these items may not be accessed:

- The message buffers
- The Receive Individual Mask registers
- The reserved words within RAM

To exit this mode, the CPU writes 0 to MCR[MDIS], causing FlexCAN to request to resume the clocks and write 0 to MCR[LPMACK]. This write occurs after the CAN protocol engine recognizes the negation of disable mode requested by the CPU.

59.3.1.1.3 Doze mode

This is a system low power mode in which the CPU bus is kept alive and a global Doze mode request is sent to all peripherals asking them to enter low-power mode. When Doze mode is globally requested, MCR[DOZE] needs to have been asserted previously for Doze mode to be triggered. The acknowledgement is obtained through the assertion by the FlexCAN of MCR[LPMACK]. The CPU must only consider the FlexCAN to be in Doze mode when both request and acknowledgement conditions are satisfied.

If Doze mode is triggered during Freeze mode, FlexCAN requests to shut down the clocks to the PE and CHI submodules, sets MCR[LPMACK] and negates MCR[FRZACK]. If Doze mode is triggered during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and checks it to be recessive.
- Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive.
- Shuts down the clocks to the PE and CHI submodules.
- Sets MCR[NOTRDY] and MCR[LPMACK].

The Bus Interface Unit continues to operate, enabling the CPU to access memory-mapped registers, except the Rx Mailboxes Global Mask registers, the Rx Buffer 14 Mask register, the Rx Buffer 15 Mask register. The the message buffers, the Rx Individual Mask registers, and the reserved words within RAM may not be accessed when the module is in Doze mode.

Exiting Doze mode is done in one of the following ways:

- CPU removing the Doze mode request
- CPU negating MCR [DOZE]
- Self Wake mechanism

In the Self Wake mechanism, if MCR[SLFWAK] was set at the time FlexCAN entered Doze mode, then upon detection of a recessive to dominant transition on the CAN bus, FlexCAN negates the DOZE bit, requests to resume its clocks and negates the LPMACK after the CAN protocol engine recognizes the negation of the Doze mode request. It also sets ESR [WAKINT] and, if enabled by MCR[WAKMSK], generates a Wake-Up interrupt to the CPU. FlexCAN will then wait for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, it will not receive the frame that woke it up. The following table details the effect of SLFWAK and WAKMSK upon wakeup from Doze mode.

Table 931. Wakeup from Doze mode

SLFWAK	WAKINT	WAKMSK	FlexCAN clocks enabled	Wakeup interrupt generated
0	-	-	No	No
0	-	-	No	No
1	0	0	No	No
1	0	1	No	No
1	1	0	Yes	No
1	1	1	Yes	Yes

The sensitivity to CAN bus activity can be modified by applying a low-pass filter function to the Rx CAN input line when in Doze mode. (See MCR[WAKSRC].) This feature can be used to protect FlexCAN from waking up due to short glitches on the CAN bus lines. Such glitches can result from electromagnetic interference within noisy environments.

59.3.1.1.4 Stop mode

In this low-power mode for the system, all chip clocks can be stopped for maximum power savings. Stop mode is globally requested by the CPU and FlexCAN writes 1 to a Stop Acknowledgment signal to indicate acknowledgment. The CPU must only consider FlexCAN to be in Stop mode when both request and acknowledgment conditions are satisfied.

If FlexCAN receives the global Stop mode request during Freeze mode, it shuts down the clocks globally by:

1. Writing 1 to MCR[LPMACK].
2. Writing 0 to MCR[FRZACK].
3. Sending the Stop Acknowledge signal to the CPU.

If Stop mode is requested during transmission or reception, FlexCAN:

1. Waits to be in either Idle or Bus Off state, or waits for the third bit of Intermission and verifies that it is recessive.
2. Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. FlexCAN does not take a pending move-in into account.
3. Ignores its receive input pin and drives its transmit pin as recessive.
4. Writes 1 to MCR[NOTRDY] and MCR[LPMACK].
5. Sends a Stop Acknowledge signal to the CPU, so that the CPU can shut down the clocks globally.

FlexCAN exits Stop mode when the CPU resumes the clocks and removes the Stop mode request. This exit can occur as a result of the Self-Wake mechanism.

In Self-Wake, if MCR[SLFWAK] = 1 when FlexCAN enters Stop mode, then upon detecting a recessive-to-dominant transition on the CAN bus, FlexCAN writes 1 to ESR1[WAKINT]. If enabled by MCR[WAKMSK], FlexCAN generates a wake-up interrupt to the CPU. Upon receiving the interrupt, the CPU should resume the clocks and remove the Stop mode request. FlexCAN waits for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, FlexCAN does not receive the frame that woke it up. Table 932 details the effect of MCR[SLFWAK] and MCR[WAKMSK] upon wakeup from Stop mode. Waking from Stop mode only works when both fields are 1.

After the CAN protocol engine recognizes the negation of the Stop mode request, FlexCAN writes 0 to MCR[LPMACK]. FlexCAN then waits for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, FlexCAN does not receive the frame that woke it up.

Table 932. Waking from Stop mode

SLFWAK	WAKINT	WAKMSK	Chip clocks enabled	Wake-up interrupt generated
0	—	—	No	No
0	—	—	No	No
1	0	0	No	No
1	0	1	No	No
1	1	0	No	No
1	1	1	Yes	Yes

The sensitivity to CAN bus activity can be modified by applying a low-pass filter function to the receive CAN input line when in Stop mode. (See the description in MCR[WAKSRC].) This feature can protect FlexCAN from waking up due to short glitches on the CAN bus lines. Such glitches can result from electromagnetic interference within noisy environments.

NOTE

When FlexCAN is in the Bus Off state and Low-Power mode, FlexCAN may take an extra 128 IDLE cycles to leave the Bus Off state after exiting Low-Power mode.

59.3.2 Transmission process

NOTE

Instances of MB_CS in this topic refer to items in message buffers. See Message buffer structure for details.

To transmit a CAN frame, the CPU must prepare a message buffer for transmission by executing the following procedure:

1. Check whether the respective interrupt flag is set, and clear it if necessary.
2. If the message buffer is active (transmission pending), request an abort of the transmission. Write the ABORT code (1001b) to the CODE field of the Control and Status word.
3. Poll the IFLAG register until the corresponding IFLAG flag is set, or wait for the interrupt request (if enabled by the respective IMASK field).
4. Read the CODE field to identify whether the transmission was aborted or transmitted (see Transmission abort mechanism).
5. Clear the corresponding interrupt flag.
6. Write the ID register (containing the local priority if enabled via MCR[LPRIOEN]).
7. Write payload data bytes.
8. Configure the Control and Status word as needed.
    a. Set ID type via MB_CS[IDE].
    b. Set Remote Transmission Request (if needed) via MB_CS[RTR].
    c. If MCR[FDEN] = 1, configure MB_CS[EDL] and MB_CS[BRS]. For details about the relationship between the written value and transmitted value of MB_CS[ESI], see Table 942.[7]
    d. Configure Data Length Code in bytes via MB_CS[DLC]. See Table 969 for detailed information.
    e. To transmit the CAN frame, activate the message buffer by writing Ch to MB_CS[CODE].

NOTE

To maximize software performance, configure all the fields in the MB_CS word in a single 32-bit write operation. If the fields are configured in separate writes, MB_CS[CODE] must be the last write in the Control and Status word.

When the MB is activated, it participates in the arbitration process and its contents are eventually transmitted according to its priority. When the DLC value stored in the MB selected for transmission exceeds the MB payload size, FlexCAN completes the expected DLC by adding a constant CCh pattern.

After a successful transmission:

1. The value of the free-running timer is written into the timestamp field.
2. The CODE field in the Control and Status word is updated.
3. Both Cyclic Redundancy Check (CRCR) and CAN FD CRC (FDCRC) are updated.
4. A status flag is set in the Interrupt Flag register.
5. If allowed by the corresponding Interrupt Mask Register field, an interrupt is generated.

After transmission, the new CODE field depends on the code used to activate the message buffer (see Table 965 and Table 966 in Message buffer structure).

When the Abort feature is enabled (MCR[AEN] is 1), after the Interrupt flag is set for an MB configured as transmit buffer, the message buffer is blocked. The CPU cannot update the message buffer until the Interrupt Flag is cleared by the CPU. The CPU must clear the corresponding IFLAG flag before preparing this MB for a new transmission or reception.

[7] There is no need to write the ESI field; it is automatically transmitted as dominant by error active nodes and as recessive by error passive nodes. If FlexCAN is operating as a network gateway, however, the CPU writes MB_CS[ESI] according to the error status of the node that sent the message.

NOTE

For backward compatibility (MCR[AEN] is 0), write the INACTIVE code (1000b) to the CODE field to deactivate the MB. In this case, the pending frame may be transmitted without notification (see Message buffer inactivation).

59.3.3 Arbitration process

The arbitration process scans the message buffers, searching for the transmission MB that holds the message to be sent at the next opportunity. This MB is called the arbitration winner. The scan starts from the lowest number MB and continues to the higher ones.

The arbitration process is triggered when at least one of the following occur:

- CRC field of the CAN frame arrives. The starting point depends on the value of CTRL2[TASD].
- Error Delimiter field of a CAN frame is in progress.
- Overload Delimiter field of a CAN frame is in progress.
- The winner of the arbitration is deactivated, and the CAN bus has not reached the first bit of the Intermission field.
- CPU writes to the Control and Status word of a winner MB, and the CAN bus has not reached the first bit of the Intermission field.
- CHI is in the Idle state and the CPU writes to the Control and Status word of any message buffer.
- FlexCAN exits Bus Off state.
- FlexCAN leaves Freeze mode or a low-power mode.

If the arbitration process does not evaluate all message buffers before the CAN bus reaches the first bit of the Intermission field, the temporary arbitration winner is invalidated. FlexCAN will not compete for the CAN bus at the next opportunity.

The arbitration process selects the winner among the active transmission message buffers at the end of the scan according to the values of CTRL1[LBUF] and MCR[LPRIOEN].

See Arbitration process (continued) for more information about this process.

59.3.3.1 Lowest-number message buffer first

If CTRL1[LBUF] is 1, the first (lowest number) active transmission message buffer found is the arbitration winner. MCR[LPRIOEN] has no effect when CTRL1[LBUF] is 1.

59.3.3.2 Highest-priority message buffer first

If CTRL1[LBUF] is 0, the arbitration process searches for the active transmission message buffer with the highest priority. The frame of this message buffer has a higher probability of winning the arbitration on the CAN bus when multiple external nodes compete for the bus simultaneously.

The sequence of bits considered for this arbitration is called the arbitration value of the message buffer. The transmission message buffer with the lowest arbitration value among all transmission message buffers has the highest priority.

If two or more message buffers have equivalent arbitration values, the message buffer with the lowest number is the arbitration winner.

The composition of the arbitration value depends on MCR[LPRIOEN].

59.3.3.2.1 Local priority disabled

If MCR[LPRIOEN] = 0, the arbitration value is built using the exact sequence of bits that would be transmitted in a CAN frame where local priority is disabled.

Table 933. Composition of the arbitration value when local priority is disabled

Format	Message buffer arbitration value (32 bits)				
Standard (IDE = 0)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	— (18 bits)	— (1 bit)
Extended (IDE = 1)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

59.3.3.2.2 Local priority enabled

To enable local priority, MCR[LPRIOEN] must be 1. In this case, the message buffer PRIO field (see Message buffer structure) is included at the very left of the arbitration value.

Table 934. Composition of the arbitration value when local priority is enabled

Format	Message buffer arbitration value (35 bits)					
Standard (IDE = 0)	PRIO (3 bits)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	— (18 bits)	— (1 bit)
Extended (IDE = 1)	PRIO (3 bits)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

Because the PRIO field is the most significant part of the arbitration value, message buffers with low PRIO values have higher priority than message buffers with high PRIO values. This priority is maintained regardless of the rest of their arbitration values.

The PRIO field is not part of the frame on the CAN bus. Its purpose is only to affect the internal arbitration process.

59.3.3.3 Arbitration process (continued)

After the arbitration winner is found (see Arbitration process), its content is copied to a hidden auxiliary message buffer called a transmit serial message buffer (TX SMB). The TX SMB has the same structure as a normal message buffer, but is not user-accessible. This copy operation is called move-out. After it is done, write access to the Control and Status word of the corresponding MB is blocked (if MCR[AEN] = 1). Write access is restored in one of the following events:

- The CPU clears the corresponding IFLAG flag after the message buffer is transmitted.
- FlexCAN enters Freeze mode or Bus Off state.
- FlexCAN loses the bus arbitration, or there is an error during the transmission.

At the first opportunity window on the CAN bus, the message on the TX SMB is transmitted according to the CAN protocol rules.

The arbitration process can be triggered under the following conditions:

- During RX and TX frames from CAN CRC field to end of frame. The value of CTRL2[TASD] may be changed to optimize the arbitration start point.
- During CAN Bus Off state from TX_ERR_CNT = 124 to 128. The value of CTRL2[TASD] may be changed to optimize the arbitration start point.
- During Control and Status write by CPU in Bus Idle. The first Control and Status write starts the arbitration process, and a second Control and Status write during this same arbitration restarts the process. If other Control and Status writes are performed, the transmission arbitration process is pending. If there is no arbitration winner after the arbitration process has finished, then the TX arbitration machine begins a new arbitration process. If there is a pending arbitration and Bus Idle state starts, then an arbitration process is triggered. In this case, the first and second Control and Status writes in Bus Idle do not restart the arbitration process. If there is not enough time to finish arbitration in the Wait For Bus Idle state and the next state is Idle, the scan is not interrupted. That scan is completed during Bus Idle state. During this arbitration, a Control and Status write does not cause an arbitration restart.

- Deactivation of an arbitration winner during a valid arbitration window.
- Upon exiting Freeze mode (first bit of the Wait For Bus Idle state). If a resynchronization occurs during Wait For Bus Idle, the arbitration process is restarted.

The arbitration process stops when:

- All message buffers are scanned.
- A transmission message buffer is found (if lowest buffer feature is enabled).
- An arbitration winner deactivates or aborts during any arbitration process.
- There is not enough time to finish the transmission arbitration process (for instance, when a deactivation is performed near the end of frame). In this case, the arbitration process is pending.
- An error or overload flag occurs in the bus.
- A low-power or Freeze mode request occurs in Idle state.

Arbitration is considered pending when:

- It is not possible to finish arbitration process in time.
- A Control and Status write occurs during arbitration, when that write is performed in an MB whose number is lower than the transmission arbitration pointer.
- Any Control and Status write occurs when there is no transmission arbitration process in progress.
- RX Match has updated an RX code to TX code.
- FlexCAN enters the Bus Off state.

If a Control and Status write is performed in the arbitration winner, a new process is restarted immediately.

If a Control and Status write is performed in an MB whose number is higher than the transmission arbitration pointer, the ongoing arbitration process scans this MB as normal.

59.3.4 Receive process

To be able to receive CAN frames into a message buffer, the CPU must prepare it for reception by executing the following steps:

1. If the message buffer is active (either TX or RX), deactivate it (see Message buffer inactivation), preferably with a safe deactivation (see Transmission abort mechanism).
2. Write the ID word into the message buffer.
3. To activate the message buffer, write the EMPTY code (0100b) to the CODE field of the Control and Status word. No setup is required for EDL, BRS, and ESI fields. The respective fields in the received message overwrite these fields.

After the MB is activated, it can receive frames that match the programmed filter. At the end of a successful reception, the move-in process updates the message buffer (see Move-in) as follows:

1. The received data field (up to eight bytes for Classical CAN message format and up to 64 bytes for CAN FD message format) is stored.
2. The received Identifier field is stored.
3. The value of the free-running timer when the second bit of the Identifier field of the frame is written into the Timestamp field of the MB.
4. The received SRR, IDE, RTR, EDL, BRS, ESI, and DLC fields are stored.
5. The CODE field in the Control and Status word is updated (see Table 965 and Table 966 in Section Message buffer structure).
6. If allowed by the corresponding Interrupt Mask field, a status flag is set in the Interrupt Flag register and an interrupt is generated.

The recommended way for the CPU to service (read) the frame received in a message buffer is:

1. Read the Control and Status word of that message buffer.
2. Verify that the BUSY bit is 0, indicating that the message buffer is locked. If it is not 0, repeat step 1 until it becomes 0. See Message buffer lock mechanism.
3. Read the contents of the message buffer. After the message buffer is locked, FlexCAN move-in processes do not modify its contents. See Move-in.
4. Acknowledge the proper flag in the IFLAG registers.
5. Unlock the message buffer by reading the free-running timer.

To verify frame reception, the CPU should poll the status flag bit for that specific message buffer in one of the IFLAG registers, not the CODE field of that message buffer. Polling the CODE field does not work in this case. After a frame is received and the CPU services the message buffer (by reading the Control and Status word and unlocking the message buffer), the CODE field does not return to EMPTY. It remains FULL, as explained in Table 965. If the CPU tries to work around this behavior by writing to the Control and Status word to force an EMPTY code after reading the message buffer without a prior safe deactivation, a newly received frame matching the filter of that message buffer may be lost.

CAUTION

In summary: never poll by reading the Control and Status word of the message buffers directly. Instead, read the IFLAG registers.

The Identifier field of the received frame is always stored in the matching message buffer. If the match was due to masking, the contents of the ID field in a message buffer may change. When MCR[SRXDIS] becomes 1, FlexCAN does not store frames transmitted by itself in any MB, even if it contains a matching receive MB. Also, no interrupt flag or interrupt signal is generated. Otherwise, when MCR[SRXDIS] becomes 0, if a matching receive MB exists, FlexCAN can receive frames transmitted by itself.

When MCR[DMA]) becomes 1, upon receiving a frame in the Legacy FIFO, IFLAG1[BUF5I] generates a DMA request and does not generate a CPU interrupt (see Legacy RX FIFO in DMA Operation). The IMASK1 fields in the Legacy RX FIFO region are not used.

The DMA controller must service the received frame using the following procedure:

1. Read the Control and Status word (read 80h address, optional).
2. Read the ID field (read 84h address, optional).
3. Read all data bytes (start read at 88h address, optional).

59.3.5 Matching process

The matching process scans the message buffer memory for RX MBs programmed with the same ID as the one received from the CAN bus. . The matching starts from the lowest number message buffer and continues toward the higher-numbered ones.

For enhanced RX FIFO, see Enhanced RX FIFO.

For legacy RX FIFO, see Legacy RX FIFO.

As the frame is received, it is stored in a hidden auxiliary MB called Receive Serial Message Buffer (RX SMB).

The starting point of the matching process depends on the following conditions:

- If the received frame is a remote frame, the starting point is the CRC field of the frame.
- If the received frame is a data frame with DLC field equal to zero, the starting point is the CRC field of the frame.
- If the received frame is a data frame and the DLC field has a nonzero value, the starting point is the DATA field of the frame.

If a matching ID is found in one of the message buffers, the move-in process transfers the contents of the RX SMB to the matched MB. If any CAN protocol error is detected, no match results are transferred to the matched MB at the end of reception.

The matching process scans all matching elements of the the active receive MBs (CODE is EMPTY, FULL, OVERRUN, or RANSWER). The process searches for a successful comparison to the matching elements of the RX SMB that is receiving the frame on the CAN bus. The RX SMB has the same structure as a message buffer. The reception structures (message buffers)

associated with the matching elements that had a successful comparison are the matched structures. The matching winner is selected at the end of the scan among those matched structures. The matching winner depends on conditions described in Table 935.

Table 935. Matching architecture

Structure	SMB[RTR]	CTRL2[RRS]	CTRL2[EACEN]	MB[IDE]	MB[RTR]	MB[ID[1]]	MB[CODE]
Message buffer	0	—	0	cmp[2]	no_cmp[3]	cmp_msk[4]	EMPTY, FULL, or OVERRUN
Message buffer	0	—	1	cmp_msk	cmp_msk	cmp_msk	EMPTY, FULL, or OVERRUN
Message buffer	1	0	—	cmp	no_cmp	cmp	RANSWER
Message buffer	1	1	0	cmp	no_cmp	cmp_msk	EMPTY, FULL, or OVERRUN
Message buffer	1	1	1	cmp_msk	cmp_msk	cmp_msk	EMPTY, FULL, or OVERRUN

1. For message buffer structure, if SMB[IDE] is 1, the ID is 29 bits (ID Standard plus ID Extended). If SMB[IDE] is 0, the ID is 11 bits (ID Standard). For Legacy RX FIFO structure, the ID depends on IDAM.
2. cmp: Compares the RX SMB contents to the MB contents regardless of masks.
3. no_cmp: The RX SMB contents are not compared to the MB contents.
4. cmp_msk: Compares the RX SMB contents to MB contents, accounting for masks.

NOTE

For Enhanced RX FIFO, see Enhanced RX FIFO matching process.

A reception structure is free-to-receive when any of the following conditions is satisfied:

- The CODE field of the message buffer is EMPTY.
- The CODE field of the message buffer is either FULL or OVERRUN, and it has already been serviced (the CPU has read the Control and Status word and unlocked it as described in Message buffer lock mechanism).
- The CODE field of the message buffer is either FULL or OVERRUN and an inactivation (see Message buffer inactivation) is performed.

The scan order for message buffers is from the matching element with the lowest number to the higher ones.

59.3.5.1 Matching priority

MCR[IRMQ] affects the matching winner search for MBs. If the field is 0, the matching winner is the first matched MB whether it is free-to-receive or not. If it is 1, the matching winner is selected according to this priority:

1. The first free-to-receive matched message buffer
2. The last non-free-to-receive matched message buffer

59.3.5.2 Special cases

If a non-safe MB inactivation (see Message buffer inactivation) occurs during the matching process and the inactivated MB is the temporary matching winner, the temporary matching winner is invalidated. The matching elements scan is not stopped and

not restarted; it continues normally. The consequence is that the current matching process works as if the matching elements compared before the inactivation did not exist. In this case, a message may be lost.

Consider an example where:

- MCR[IRMQ] is 1.
- There are two message buffers with the same ID: the second and fifth MBs in the array.
- FlexCAN starts receiving messages with that ID.

When the first message arrives, the matching algorithm finds the first match in message buffer number 2. The code of this message buffer is EMPTY, so the message is stored in that MB. When the second message arrives, the matching algorithm finds MB number 2 again, but it is not "free-to-receive." It continues looking, finds MB number 5, and stores the message in that MB. If yet another message with the same ID arrives, the matching algorithm finds no matching free-to-receive MBs, so it overwrites the last matched message buffer (MB number 5). In doing so, it updates the CODE field of the message buffer to OVERRUN.

The ability to match the same ID in more than one MB can be used to implement a reception queue to allow more time for the CPU to service the MBs. By programming more than one MB with the same ID, received messages are queued into the message buffers. The CPU can examine the Timestamp field of the message buffers to determine the order in which the messages arrived.

Matching a range of IDs is possible via ID acceptance masks. FlexCAN supports individual masking per message buffer (see Receive Individual Mask (RXIMR0 - RXIMR95)). During the matching algorithm, if a mask field is 1, the corresponding ID bit is compared. If the mask field is 0, the corresponding ID bit is a "don't care". Individual Mask Registers are implemented in RAM, so they are not initialized out of reset. Also, they can only be programmed when the module is in Freeze mode; otherwise, the module blocks them.

FlexCAN also supports an alternate masking scheme with only RX Message Buffers Global Mask (RXMGMASK), Receive 14 Mask (RX14MASK), and Receive 15 Mask (RX15MASK) for backward compatibility with legacy applications. This alternate masking scheme is enabled when the MCR[IRMQ] = 0.

59.3.6 Move process

There are two types of move process: move-in and move-out.

59.3.6.1 Move-in

The move-in process is the copying of a message received by an RX SMB to an RX message buffer that has matched it. Each RX SMB has its own move-in process, but only one is performed at a given time. The move-in starts only when the message held by the RX SMB has a corresponding match (see Matching process) and all of the following conditions are true:

- The CAN bus has reached or already gone past:
  - The second bit of Intermission field next to the frame that carried the message that is in the RX SMB.
  - The first bit of an overload frame next to the frame that carried the message in the RX SMB.
- There is no ongoing matching process.
- The CPU is not locking the destination message buffer.
- No move-in process from another RX SMB is ongoing. If more than one move-in process is to be started at the same time, both are performed and the newest process substitutes for the oldest.

The term "pending move-in" is used throughout the documentation and stands for a move-to-be that does not satisfy all of the above conditions.

If any of the following conditions is satisfied, the move-in is canceled and the RX SMB is able to receive another message:

- The destination message buffer is inactivated after the CAN bus has reached the first bit of the Intermission field next to the frame that carried the message. Also, its matching process has finished.
- There is a previous pending move-in to the same destination message buffer.
- The RX SMB is receiving a frame transmitted by FlexCAN itself and self-reception is disabled (MCR[SRXDIS] = 1).
- Any CAN protocol error is detected.

If the module enters Freeze or Low-Power mode, the pending move-in is not canceled. It remains on hold, waiting for Freeze and Low-Power mode to be exited and for the module to be unlocked. If a message buffer is unlocked during Freeze mode, the move-in occurs immediately.

The move-in process is FlexCAN executing the following steps:

1. Read all data words from the RX SMB in accordance with the selected payload size for the RX storage element.
2. Write all data words to the RX message buffer according to the selected payload size for the RX storage element. If the data size of the storage element is smaller than the original payload size described in the DLC field of the message, the payload is truncated. The high-order bytes that do not fit the destination size are lost.
3. Read the Control and Status and ID words from the RX SMB.
4. Write the Control and Status and ID words to the RX message buffer, and update the CODE field.

The move-in process is not atomic; the inactivation of the destination message buffer immediately cancels it (see Message buffer inactivation). In this case, the message buffer may remain partially updated, and therefore incoherent.

To alert the CPU that the message buffer content is temporarily incoherent, the BUSY Bit (least significant bit of the CODE field) of the destination message buffer becomes 1 during move-in.

59.3.6.2 Move-out

The move-out process is the copying of content from a TX message buffer to the TX SMB when a message for transmission is available (see Arbitration process). The move-out occurs in the following conditions:

- In the first bit of Intermission field
- During the Bus Off state, when TX Error Counter is in the 124 to 128 range
- During the Bus Idle state
- During the Wait For Bus Idle state

The move-out process is not atomic. Only the CPU has priority to access the memory concurrently outside the Bus Idle state. In Bus Idle, the move-out has the lowest priority of the concurrent memory accesses.

59.3.7 Data coherence

In order to maintain data coherency and proper FlexCAN operation, the CPU must obey the rules described in Transmission process and Receive process.

59.3.7.1 Transmission abort mechanism

The abort mechanism provides a safe way to request the abortion of a pending transmission. A feedback mechanism is provided to inform the CPU whether the transmission was aborted or the frame could not be aborted and was transmitted instead.

These primary conditions must be fulfilled in order to abort a transmission:

- MCR[AEN] must be 1.
- The first CPU action must be the writing of abort code (1001b) into the CODE field of the Control and Status word.

Active message buffers configured for transmission must be aborted before they can be updated. The write operation is blocked and the transmission is not disturbed when the abort code is written to:

- A message buffer currently being transmitted.
- A message buffer that was already loaded into the TX SMB for transmission.

In this case, the abort request is captured and kept pending until one of the following conditions is satisfied:

- The module loses the bus arbitration.
- There is an error during the transmission.
- The module is put into Freeze mode.

- The module enters the Bus Off state.
- There is an overload frame.

If none of the conditions above are reached:

1. The message buffer is transmitted correctly.
2. The interrupt flag is set in the proper IFLAG register.
3. If enabled, an interrupt to the CPU is generated.

The abort request is automatically cleared when the interrupt flag is set. If only one of the above conditions is reached, the frame is not transmitted. In this case:

1. The abort code is written into the CODE field.
2. The interrupt flag is set in the proper IFLAG register.
3. Optionally, an interrupt is generated to the CPU.

If the CPU writes the abort code before the transmission begins internally, the write operation is not blocked. The MB is updated and the interrupt flag is set. In this way, the CPU only needs to read the abort code to verify that the active MB was safely inactivated. In this case, although MCR[AEN] = 1 and the CPU wrote the abort code, the MB is inactivated and not aborted, because the transmission did not start yet. A message buffer is aborted only when the abort request is captured and kept pending until one of the previous conditions is satisfied.

59.3.7.2 Message buffer inactivation

Inactivation protects the message buffer against updates by FlexCAN internal processes. It allows the CPU to rely on message buffer data coherence after having updated it, even in Normal mode.

Inactivation of transmission message buffers must be performed only when MCR[AEN] = 0.

If a message buffer is inactivated, it does not participate in the arbitration process or the matching process until it is reactivated. See Transmission process and Receive process for detailed instructions on how to inactivate and reactivate a message buffer.

To inactivate a message buffer, the CPU must update its CODE field to INACTIVE (either 0b or 1000b).

Because you cannot synchronize the CODE field update with FlexCAN internal processes, an inactivation can have the following consequences:

- A frame in the bus that matches the filtering of an inactivated RX message buffer may be lost without notice. This loss can occur even if there are other message buffers with the same filter.
- A frame containing the message within an inactivated TX message buffer may be transmitted without setting the respective IFLAG flag.

To perform a safe inactivation and avoid the above consequences for TX message buffers, the CPU must use the transmission abort mechanism (see Transmission abort mechanism).

The inactivation automatically unlocks the message buffer (see Message buffer lock mechanism).

59.3.7.3 Message buffer lock mechanism

In addition to message buffer inactivation, FlexCAN uses a message buffer lock mechanism to maintain data coherence for the receive process. When the CPU reads the Control and Status word of an RX message buffer with codes FULL or OVERRUN, FlexCAN is configured to allow the CPU to read the whole message buffer in an atomic operation. FlexCAN sets an internal lock flag for that message buffer.

The lock is released in any of the following events:

- The CPU reads the free-running timer (global unlock operation).
- The CPU reads the Control and Status word of another message buffer, regardless of its code.

- The CPU writes into the Control and Status word. This procedure is not recommended for normal unlocking, because it cancels a pending move and may lose a received message.

The message buffer lock prevents a new frame from being written into the message buffer when the CPU is reading it.

NOTE

The locking mechanism applies only to RX message buffers that have a code other than INACTIVE (0b) or EMPTY[1] (0100b). TX message buffers cannot be locked.

Consider an example where:

- The second and the fifth message buffers of the array are programmed with the same ID.
- FlexCAN has already received and stored messages into these two message buffers.
- The CPU reads message buffer number 5 while another message with the same ID is arriving.

When the CPU reads the Control and Status word of message buffer number 5, this message buffer is locked. The new message arrives and the matching algorithm finds no free-to-receive message buffers, so it overrides message buffer number 5. This message buffer is locked, so the new message cannot be written to it. The message remains in the RX SMB until the message buffer is unlocked, and only then is it written to the message buffer.

If the message buffer remains locked and another new message with the same ID arrives, the new message overwrites the one in the RX SMB. There is no indication of lost messages in the CODE field of the message buffer or in Error and Status 1 (ESR1).

When the message is moved from the RX SMB to the message buffer, the BUSY bit on the CODE field becomes 1. If the CPU reads the Control and Status word and identifies that the BUSY bit is 1, it must wait until the BUSY bit becomes 0 to access the MB.

NOTE

If the BUSY bit is 1 or the message buffer is empty, reading the Control and Status word does not lock the message buffer.

Inactivation takes precedence over locking. If the CPU inactivates a locked receive message buffer, then its lock status is negated and the message buffer is marked as invalid for the current matching round. Any pending message on the RX SMB is not transferred to the message buffer. A message buffer is unlocked when the CPU reads Free-Running Timer (TIMER) or the Control and Status word of another message buffer.

The lock and unlock mechanisms have the same functionality in Normal and Freeze modes.

An unlock during Normal or Freeze mode results in the move-in of the pending message. If unlocking occurs during a low-power mode, however, the move-in is postponed (see Modes of operation). Move-in takes place only when the module returns to Normal or Freeze mode.

59.3.8 Enhanced RX FIFO

FlexCAN supports an enhanced RX FIFO engine which can store up to 20 CAN FD messages. The region 2000h–204Fh contains the output of the FIFO, which the CPU should read. To enable the enhanced RX FIFO, write 1 to ERFCR[ERFEN]. FlexCAN has two FIFO options, Legacy RX FIFO and Enhanced RX FIFO, but both options cannot be enabled at the same time. See Legacy RX FIFO for additional information.

To configure the enhanced RX FIFO watermark, write a value to ERFCR[ERFWM]. If ERFCR[ERFWM] is configured, the CPU is notified only if a minimum number of messages is stored in the FIFO. When the number of stored messages is greater than the value in ERFCR[ERFWM], the module sets ERFSR[ERFWMI]. Optionally, if MCR[DMA] or ERFIER[ERFWMIIE] are enabled, a DMA transfer or an interrupt can be triggered, respectively.

For the enhanced RX FIFO to receive, the CPU must execute the configuration procedure below. If the CPU must change any configurations of the Enhanced RX FIFO, the same procedure must be followed.

Prerequisites

[1] In previous FlexCAN versions, reading the Control and Status word locked the message buffer even when CODE indicates it is EMPTY. This behavior is maintained when the IRMQ bit is 0.

MCR[RFEN] must be 0.

Procedure

Step	Purpose	Programming	Notes
1	Enter Freeze mode.	See Freeze mode.	—
2	If enhanced RX FIFO is not already enabled, enable it.	Write 1 to ERFCR[ERFEN].	—
3	Reset enhanced RX FIFO engine.	Write 1 to ERFSR[ERFCLR].	—
4	If the enhanced RX FIFO error flags are set, clear them.	Write 1 to these flags: • ERFSR[ERFUFW] • ERFSR[ERFOVF] • ERFSR[ERFWMI] • ERFSR[ERFDA]	—
5	Specify the total number of enhanced RX FIFO filter elements to be used in Enhanced RX FIFO reception.	Write the number to ERFCR[NFE].	—
6	Specify the number of extended ID and standard ID filter elements to be used in Enhanced RX FIFO reception.	Write the number to ERFCR[NEXIF].	ERFCR[NEXIF] ≤ ERFCR[NFE] + 1.
7	If you are using DMA, enable DMA.	Write 1 to MCR[DMA].	—
8	If you are using DMA, specify the number of words to transfer for each Enhanced RX FIFO data element.	Write the number to ERFCR[DMALW].	—
9	Specify the Enhanced RX FIFO watermark.	Write the number to ERFCR[ERFWM].	If MCR[DMA] = 1, ERFCR[ERFWM] should be 0h.
10	If you are using interrupts, enable the interrupts.	Write 1 to the interrupt enables in Enhanced RX FIFO Interrupt Enable (ERFIER).	—
11	Configure the filter elements.	Write to the ERFFEL*n* registers.	ERFFEL*n* registers are implemented in RAM; you must explicitly initialize them before prior any reception.
12	Exit Freeze mode.	See Freeze mode.	—

There are two types of enhanced RX FIFO filter elements that can be stored in ERFFEL*n* registers: extended-ID filter elements and standard-ID filter elements. Each extended-ID filter element is stored in two ERFFEL*n* registers, and each standard-ID filter element is stored in one ERFFEL*n* register. ERFCR[NFE] defines the total number of Enhanced RX FIFO filter elements.

In addition, the filter memory space can be split into two regions: one for extended-ID filter elements and another for standard-ID filter elements, according to ERFCR[NEXIF]. Figure 438 shows how the enhanced RX filter elements are defined. See Enhanced RX FIFO matching process for information about the Enhanced RX FIFO matching process and filter element formats.

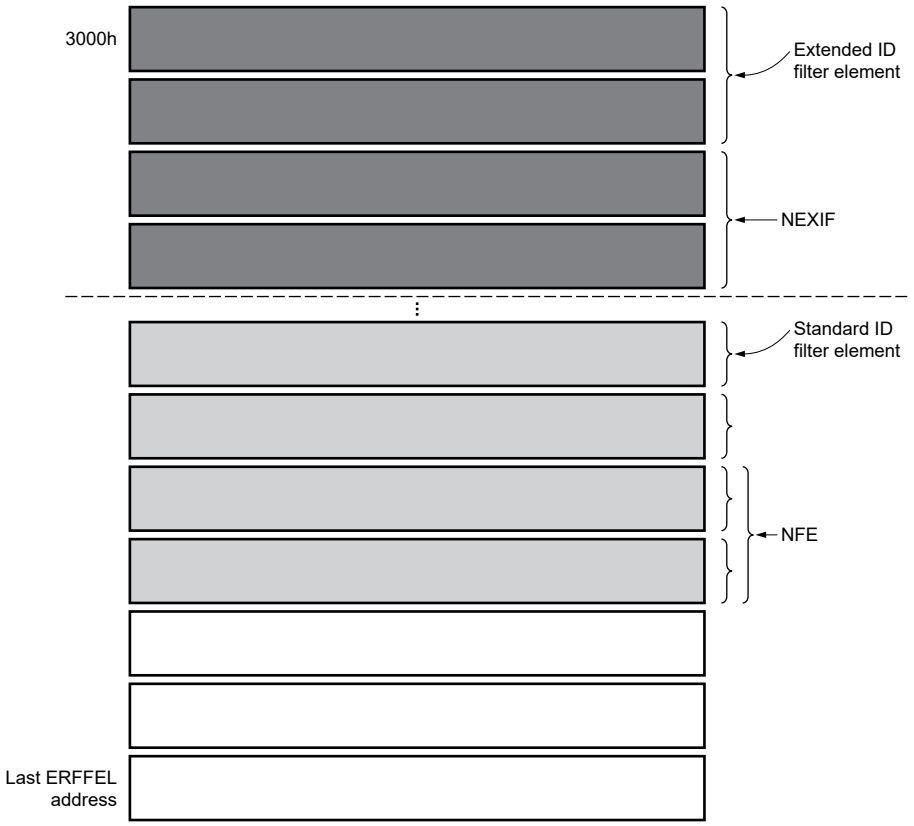


Figure 438. Number of enhanced RX FIFO filter elements

59.3.8.1 Enhanced RX FIFO matching process

When ERFCR[ERFEN] = 1, FlexCAN scans the ERFFEL*n* memory region. If at least one filter element satisfies the matching criteria, the CAN message content is transferred to the enhanced RX FIFO memory. If multiple filters match the incoming message ID, the first matching filter found by the matching process must be indicated in IDHIT.

Each ERFFEL*n* register can store one standard filter element. ERFFEL*n*[FEL][31:30], also called FSCH, determines the matching criteria in this way:

- If FSCH = b00, the filter scheme is based on mask and filter. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base-frame format (IDE = 0).
  2. (ID[*n*] = STD ID filter [*n*]) or (STD ID Mask[*n*] = 0) for each bit *n* from 0 to 10.
  3. (RTR = RTR Filter) or (RTR MASK = 0).

In this explanation, RTR and ID are the Remote Transmit Request field and the ID from a CAN message, respectively.

If FSCH = b00, the filters and masks are defined as shown in Table 936.

Table 936. Standard ID filter element with filter and mask scheme (FSCH = b00)

31	30	29	28	27	26										16	15			12	11	10										0
FSCH = b00		Reserved		RTR filter	STD ID filter											Reserved				RTR mask	STD ID mask										

- If FSCH = b01, the filter scheme is based on range. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0).
  2. ID ≥ STD ID Filter1.
  3. ID ≤ STD ID Filter2.
  4. (RTR = RTR filter) or (RTR MASK = 0).

RTR and ID are the Remote Transmit Request bit and ID from a CAN message, respectively. If FSCH = b01, the filters and mask are defined as shown in Table 937.

Table 937. Standard ID filter element with range scheme (FSCH = b01)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH = b01		Reserved		RTRfilter	STD ID Filter2		Reserved		RTR mask	STD ID Filter1	

- If FSCH = b10, the filter scheme is based on two filters without masks. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0).
  2. (ID[*n*] = STD ID Filter1[*n*]) or (ID[*n*] = STD ID Filter2[*n*]) for each bit *n* from 0 to 10.
  3. (RTR = RTR Filter1) or (RTR = RTR Filter2).

RTR and ID are the Remote Transmit Request bit and ID from a CAN message, respectively. If FSCH = b10, the filters are defined as shown in Table 938.

Table 938. Standard ID filter element with two-filter scheme (FSCH = b10)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH = b10		Reserved		RTR Filter 2	STD ID Filter2		Reserved		RTR Filter 1	STD ID Filter1	

Each pair of ERFFEL*n* registers can store one extended filter element. ERFFEL*n*[FSCH] determines the matching criteria in this way:

- If FSCH = b00, the filter scheme is based on mask and filter. A CAN message matches an extended ID filter element only if these criteria are reached:
  1. CAN message is extended frame format (IDE = 1).
  2. (ID[*n*] = EXT ID filter [*n*]) or (EXT ID Mask[*n*] = 0) for each bit *n* from 0 to 28.
  3. (RTR = RTR Filter) or (RTR MASK = 0).

If FSCH = b00, the filters and masks are defined as shown in Table 939.

Table 939. Extended ID filter element with filter + mask scheme (FSCH = b00)

31	30	29	28	0
FSCH		RTR filter	EXT ID filter	
Reserved		RTR mask	EXT ID mask	

- If FSCH = b01, the filter scheme is based on range. A CAN message matches an extended ID filter element only if the following criteria are reached:
  1. CAN message is extended frame format (IDE = 1).
  2. ID ≥ EXT ID Filter1.
  3. ID ≤ EXT ID Filter2.
  4. (RTR = RTR Filter) or (RTR MASK = 0).

If FSCH = b01, the filters and masks are defined as shown in Table 940.

Table 940. Extended ID filter element with range scheme (FSCH = b01)

31	30	29	28																										0
FSCH		RTR filter	EXT ID Filter2																										
Reserved		RTR mask	EXT ID filter 1																										

- If FSCH = b10, the filter scheme is based on two filters without masks. A CAN message matches an extended ID filter element only if these criteria are reached:
  1. CAN message is extended frame format (IDE = 1).
  2. (ID[*n*] = EXT ID Filter1[*n*]) or (ID[*n*] = EXT ID Filter2[*n*]) for each bit *n* from 0 to 28.
  3. (RTR = RTR Filter1) or (RTR = RTR Filter2).

If FSCH = b10, the filters are defined as shown in Table 941.

Table 941. Extended ID filter element with two-filter scheme (FSCH = b10)

31	30	29	28																										0
FSCH		RTR Filter2	EXT ID Filter2																										
Reserved		RTR Filter1	EXT ID filter 1																										

59.3.8.2 Enhanced RX FIFO under DMA operation

You can enable the DMA feature by writing 1 to both ERFCR[ERFEN] and MCR[DMA]. The DMA controller can read the received message by reading a message buffer structure at the enhanced FIFO output port at the address range defined in Enhanced RX FIFO structure.

NOTE

FlexCAN supports 32-bit access only for DMA transfers.

For proper FIFO engine operation, the CPU should not access the Enhanced FIFO output port address range during DMA operation. Before writing 1 to MCR[DMA], the CPU must service Enhanced RX FIFO status bits. Otherwise, these bits may show that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before writing 0 to MCR[DMA], the CPU must first clear the ERFSR[ERFUFW], ERFSR[ERFOVF], ERFSR[ERFWMI], and ERFSR[ERFDA] flags. It must then clear the enhanced RX FIFO engine by writing one to ERFSR[ERFCLR].

When there is one frame available to be read from the Enhanced RX FIFO, FlexCAN sets ERFSR[ERFDA]. Upon receiving the request, the DMA controller can read the message in the Enhanced RX FIFO output. Each message reading process must end by the address defined in ERFCR[DMALW].

Follow these rules for Enhanced RX FIFO DMA operation:

- Because a DMA transfer cannot be changed dynamically, program ERFCR[DMALW] so the enhanced RX FIFO element can store the largest CAN message present on the CAN bus.
- Data bytes are valid according to the DLC field. See Table 969.

Each time the DMA controller reads one message from the FIFO, FlexCAN clears ERFSR[ERFDA]. If there is at least one message stored in the FIFO, FlexCAN sets it again.

Consider an example where the maximum number of bytes in the data field of a CAN frame for a certain application is eight, and high-resolution timestamp is enabled. In that case, the last enhanced RX FIFO address offset can be found in Table 979 and Table 980. Using this address offset, ERFCR[DMALW] can be determined in this way:

- Maximum number of data bytes = 8
- HR_TIME_STAMP enabled
- Last address offset = TS_OFF = 2014h
- DMALW = 5

59.3.8.3 Enhanced RX FIFO clear operation

When ERFCR[ERFEN] is 1, the CPU can clear the Enhanced RX FIFO by writing 1 to ERFSR[ERFCLR]. The clear operation resets the internal FIFO pointers, but the FIFO content stored in RAM is not changed. This operation can only be performed in Freeze mode; the module blocks the operation in other modes. This operation does not clear ERFSR[ERFUFW], ERFSR[ERFOVF], ERFSR[ERFDA], or ERFSR[ERFWMI]. The CPU must service all these fields before executing the clear FIFO operation.

59.3.9 Legacy RX FIFO

The Legacy RX FIFO is receive-only. To enable it, write 1 to MCR[RFEN]. To maintain software backward compatibility with previous versions of FlexCAN that did not have the Legacy FIFO feature, the reset value of this field is zero.

CAUTION

Do not enable Legacy RX FIFO when the CAN FD feature is enabled.

The Legacy FIFO is six messages deep. The memory region the Legacy FIFO structure occupies (both message buffers and Legacy FIFO engine) is described in Legacy RX FIFO structure. The CPU can read the received messages sequentially, in the order they were received, by repeatedly reading a message buffer structure at the output of the Legacy FIFO.

IFLAG1[BUF5I] (Frames Available in Legacy RX FIFO) is set when at least one frame is available to be read from the Legacy FIFO. If the corresponding mask bit enables it, an interrupt is generated. Upon receiving the interrupt, the CPU can read the message (accessing the output of the Legacy FIFO as a message buffer) and Legacy RX FIFO Information (RXFIR), then clear the interrupt. If there are more messages in the Legacy FIFO, clearing the interrupt:

1. Updates the output of the Legacy FIFO with the next message.
2. Updates RXFIR with the attributes of that message.
3. Reissues the interrupt to the CPU.

Otherwise, the flag remains cleared. The output of the Legacy FIFO is valid only when IFLAG1[BUF5I] is set.

IFLAG1[BUF6I] (Legacy RX FIFO Warning) is set when the Legacy RX FIFO receives a new message that increases the number of unread messages from four to five. This change means that the Legacy RX FIFO is almost full. The flag remains set until the CPU clears it.

IFLAG1[BUF7I] (Legacy RX FIFO Overflow) is set when an incoming message is lost because the Legacy RX FIFO is full. The flag is not set when the Legacy RX FIFO is full and a message buffer captures the message. The flag remains set until the CPU clears it.

Clearing one of the three flags above does not affect the state of the other two.

If an IFLAG flag is set and the corresponding mask bit is 1, an interrupt is generated.

A powerful filtering scheme is provided to accept only frames intended for the target application, reducing the interrupt servicing workload. The filtering criteria are specified by programming a table of up to 128 32-bit registers, according to CTRL2[RFFN]. This table can be configured to one of the following formats (see also Legacy RX FIFO structure):

- Format A: 128 Identifier Acceptance Filters (IDAFs) — extended or standard IDs including IDE and RTR
- Format B: 256 IDAFs — standard IDs or extended 14-bit ID slices including IDE and RTR
- Format C: 512 IDAFs — standard or extended 8-bit ID slices

NOTE

A chosen format is applied to all entries of the filter table. It is not possible to mix formats within the table.

Every frame available in the Legacy RX FIFO has a corresponding Identifier Acceptance Filter Hit Indicator (IDHIT). The IDHIT can be read in the IDHIT field in the Control and Status word, as shown in the Legacy RX FIFO Structure description. The CPU can also obtain this information by accessing Legacy RX FIFO Information (RXFIR). RXFIR[IDHIT] refers to the message at the output of the Legacy FIFO, and is valid when IFLAG1[BUF5I] is set. RXFIR must be read only before clearing the flag, guaranteeing that the information refers to the correct frame within the Legacy FIFO.

The Individual Mask Registers (RXIMR*n*) individually affect up to 32 elements of the filter table, according to the value of CTRL2[RFFN]. This configuration allows very powerful filtering criteria to be defined. If MCR[IRMQ] is 0, the Legacy RX FIFO filter table is affected by Legacy RX FIFO Global Mask (RXFGMASK).

NOTE

See Table 929 for information about the difference between FD and non-FD regarding this feature.

59.3.9.1 Legacy RX FIFO in DMA Operation

The receive-only Legacy FIFO can support DMA. To enable this feature, write 1 to both MCR[RFEN] and MCR[DMA]. To maintain backward compatibility with previous versions of the module that did not have the DMA feature, the reset value of MCR[DMA] is zero.

The DMA controller can read the received message by reading a message buffer structure at the Legacy FIFO output port in the 80h–8Ch address range.

NOTE

FlexCAN supports 32-bit access only for DMA transfers.

When MCR[DMA] = 1, the CPU must not access the Legacy FIFO output port address range. Before writing 1 to MCR[DMA], the CPU must service the IFLAG flags set in the Legacy RX FIFO region. Otherwise, these flags may indicate that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before writing 0 to MCR[DMA], the CPU must perform a clear Legacy FIFO operation.

When at least one frame available to be read from the FIFO, IFLAG1[BUF5I] (Frames available in Legacy RX FIFO) is set. A DMA request is generated simultaneously. Upon receiving the request, the DMA controller can read the message (accessing the output of the Legacy FIFO as a message buffer). The DMA reading process must end by reading address 8Ch. This read operation:

- Clears IFLAG1[BUF5I].
- Updates the FIFO output with the next message (if the FIFO is not empty).
- Updates Legacy RX FIFO Information (RXFIR) with the attributes of the new message.

If there are more messages stored in the FIFO, IFLAG1[BUF5I] is reasserted and another DMA request is issued. Otherwise, the flag remains cleared.

IFLAG1[BUF6I] and IFLAG1[BUF7I] are not used when the DMA feature is enabled.

When FlexCAN is working with DMA, the CPU does not receive any Legacy RX FIFO interruption and must not clear the related IFLAG flags. The related IMASK bits are not used to mask the generation of DMA requests.

NOTE

See Table 929 for information about the difference between FD and non-FD regarding this feature.

59.3.9.2 Clear Legacy FIFO

When MCR[RFEN] = 1, you can use the clear Legacy FIFO operation to empty Legacy FIFO contents. When the CPU writes 1 to IFLAG1[BUF0I], the clear FIFO operation occurs. This operation can only be performed in Freeze mode; FlexCAN blocks it in other modes. This operation does not clear the FIFO IFLAG flags; the CPU must service all FIFO IFLAG flags before executing the clear FIFO operation.

When Legacy RX FIFO is working with DMA, the clear FIFO operation clears IFLAG1[BUF5I], and the DMA request is canceled.

CAUTION

The clear Legacy FIFO operation does not clear IFLAG flags, except when MCR[DMA] = 1; in this case, only IFLAG1[BUF5I] is cleared.

59.3.10 CAN protocol-related features

59.3.10.1 CAN FD ISO compliance

The CAN FD protocol has been improved to increase the failure-detection capability that was in the original CAN FD protocol. This original protocol is also called non-ISO CAN FD, by CAN in Automation (CiA). A three-bit stuff counter and a parity bit have been introduced in the improved CAN FD protocol, now called ISO CAN FD. The CRC calculation has also been modified. All these improvements make the ISO CAN FD protocol incompatible with the non-ISO CAN FD protocol. FlexCAN still supports non-ISO CAN FD, so it can be used during an intermediate phase, for evaluation and development purposes.

It is recommended that you configure FlexCAN with the ISO CAN FD protocol by writing 1 to CTRL2[ISOCANFDEN].

59.3.10.2 CAN FD frames

ISO 11898-1:2015 specifies the Classical Frame format compliant to ISO 11898-1:2003 (2003) and introduces the CAN Flexible Data Rate Frame format (CAN FD). The Classical Frame format allows bit rates up to one Mbit/s and payloads up to eight bytes per frame. The Flexible Data Rate Frame format allows bit rates faster than one Mbit/s and payloads longer than eight bytes per frame. FlexCAN can receive and transmit CAN FD messages interleaved with Classical CAN messages.

There are additional control bits in the CAN FD frame:

- The Extended Data Length (EDL) bit enables a longer data payload with different data length coding.
- The Bit Rate Switch (BRS) bit decides whether the bit rate is switched inside a CAN FD format frame.
- The Error State Indicator (ESI) flag is transmitted dominant by error active nodes, and recessive by error passive nodes.

There are no Remote Frames (see Remote frames) in the CAN FD format. A message configured to transmit a Remote Frame is always sent out in Classical CAN format. When an FD frame is received and matches a message buffer, the RTR bit in the receiving message buffer becomes 0. The RTR bit must be considered in classical frames only.

59.3.10.2.1 CAN FD messages

CAN FD messages may be formatted as long frames where the data field exceeds eight bytes, and may range from 12 up to 64 bytes. They can also be configured to support bit rate switching. In this case, the control field, data field, and CRC field of a CAN frame are transmitted with a higher bit rate than the beginning and end of the frame. Messages in Classical CAN format are limited to transport a maximum payload of eight bytes at nominal rate. Figure 439 illustrates the message formats for Classical and FD frames with either standard or extended ID.

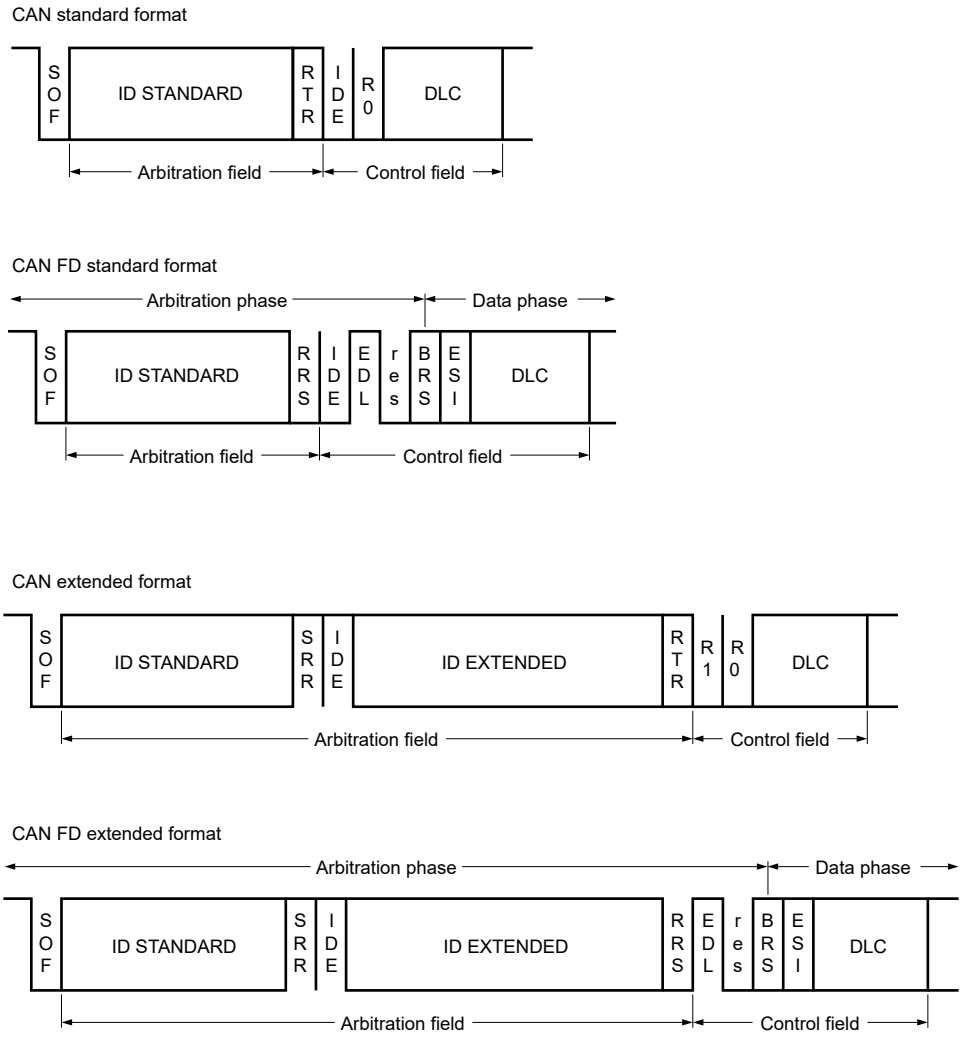


Figure 439. CAN message formats

MCR[FDEN] enables the ability to receive and transmit CAN FD messages. A recessive R0 bit in CAN frames with 11-bit identifiers, or a recessive R1 bit in CAN frames with 29-bit identifiers, is decoded as an EDL bit (not a reserved one). A recessive EDL bit identifies a CAN FD frame, and a dominant EDL bit identifies a Classical CAN frame. The BRS bit specifies whether this frame switches the bit rate in its data phase. A long frame is decoded according to the DLC field value (see DLC definition in Message buffer structure).

CAN FD messages can be transmitted with two different bit rates. The first part of a CAN FD frame, from the Start of Frame (SOF) bit until the Bit Rate Switch (BRS) bit is called the arbitration phase. This part is transmitted with the nominal bit rate based on a set of nominal CAN bit timing configuration values. The second part, from the BRS bit until the CRC Delimiter bit, is called the data phase. When this second part is transmitted, a second set of CAN data bit timing configuration values determines the data bit rate. Finally, from the CRC delimiter until the Intermission bits, the transmission returns to nominal bit rate.

59.3.10.2.2 BRS in CAN FD

In CAN FD frames with bit rate switching, the bit timing changes inside the frame at the sample point of the BRS bit if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the nominal CAN bit timing is used as defined by CAN Bit Timing (CBT). (Control 1 (CTRL1) also defines this timing for backward compatibility.) Upon detecting a recessive BRS bit, the CAN data bit timing is used as defined by CAN FD Bit Timing (FDCBT).

NOTE

If the time quantum length in nominal bit timing and in the data bit timing are not identical, a quantization error of up to one time quantum of the arbitration phase may be present as a phase error. This situation can occur after the switch from arbitration to data phase, and it lasts until the next synchronization event. The length of the time quantum should be the same in nominal and data bit timing. This configuration minimizes the chance of error frames on the CAN bus, and optimizes the clock tolerance in networks that use FD frames.

If BRS = 1 in the selected TX MB, FDCTRL[FDRATE] enables the transmission of all frames with bit rate switching. If FDCTRL[FDRATE] = 0, the transmission is performed at nominal rate regardless of the BRS bit value. FDCTRL[FDRATE] can be written at any time but takes effect only for the next message transmitted or received.

Nominal bit timing is resumed at the sample point of the CRC Delimiter bit or when an error is detected, whichever occurs first. Figure 440 describes the mechanism for entering and leaving the data phase when the BRS bit is recessive.

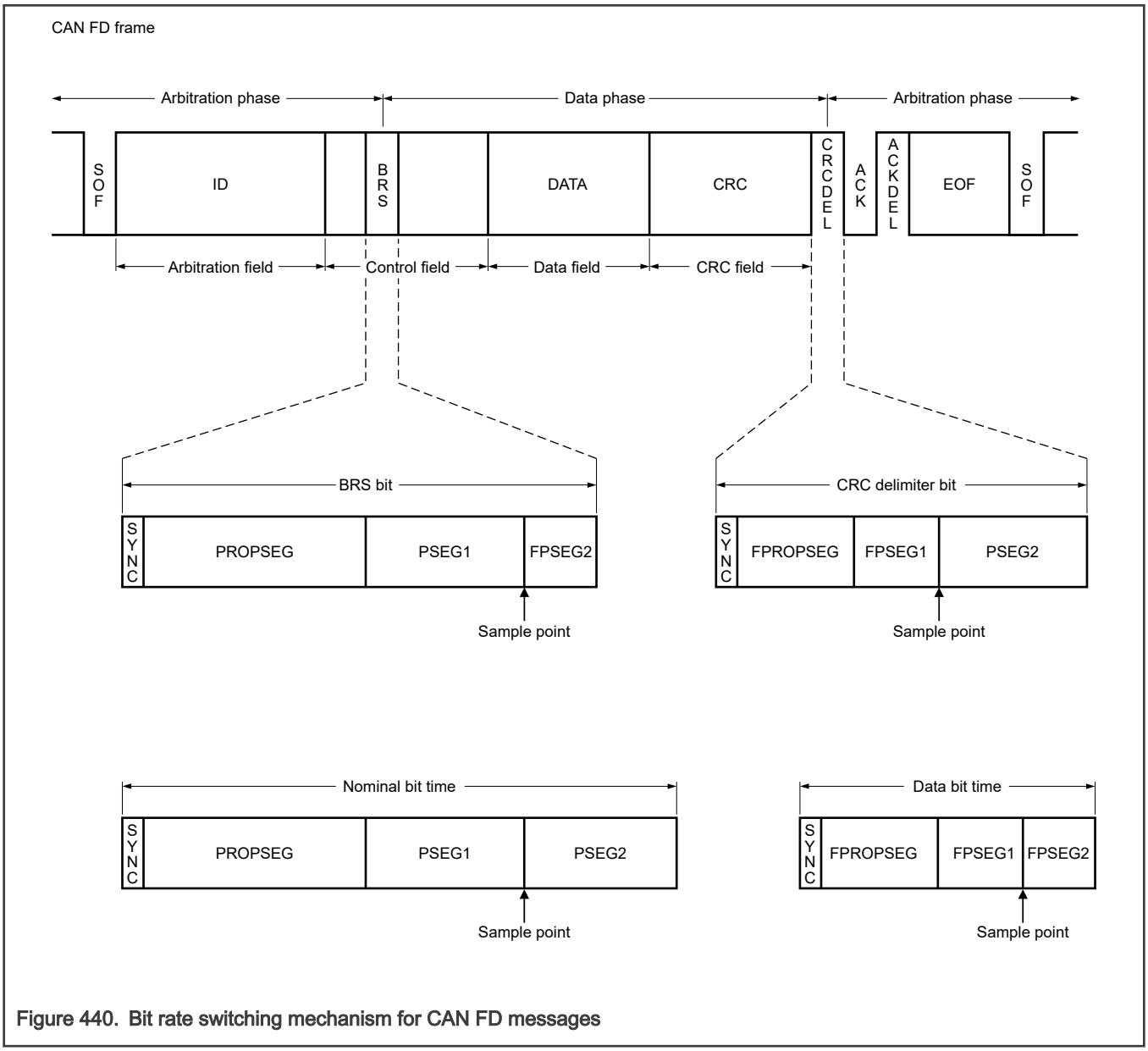


Figure 440. Bit rate switching mechanism for CAN FD messages

NOTE

In Classical CAN frames, the CRC delimiter is one recessive bit. In CAN FD frames, the CRC delimiter may consist of one or two recessive bits. FlexCAN sends only one recessive bit as the CRC delimiter. It accepts two recessive bits before the recessive-to-dominant edge that starts the acknowledge slot. As a receiver, FlexCAN sends its acknowledge bit after the first CRC delimiter bit. In CAN FD frames, FlexCAN accepts a two-bit dominant ACK slot as a valid ACK to compensate for phase shifts between the receivers.

The maximum configurable bit rate in the CAN FD data phase depends on the clock frequency of the CAN_PE subblock. For example, for a CAN_PE clock frequency of 40 MHz and the shortest configurable bit time of 5 time quanta, the bit rate in the data phase is 8 Mbit/s.

NOTE

The frequency used in this example may not be supported on this chip. It is shown only to demonstrate how the maximum configurable bit rate is calculated.

59.3.10.2.3 ESI in CAN FD

The value of the ESI bit is determined:

- By the error state of the transmitter at the start of the transmission, if the frame is originated in the FlexCAN node,
- Or by the original transmitting node when FlexCAN is acting as a gateway for the message.

If the transmitter is error-passive, ESI is transmitted recessive; otherwise, it is transmitted dominant. The permutations of the relationship between the written value and the transmitted value of the ESI are shown in Table 942.

Table 942. Written versus transmitted values of ESI field

FlexCAN fault confinement status at start of frame	ESI bit of TX MB	Transmitted ESI
Error active	0	0 (Error Active)
Error passive	0	1 (Error Passive)
Error active	1	1 (Error Passive)
Error passive	1	1 (Error Passive)

59.3.10.2.4 CRC calculations in CAN FD

Different CAN frame formats have different CRC polynomials. The first polynomial, CRC_15, is used for all frames in Classical CAN format. The second, CRC_17, is used for frames in CAN FD format with a data field up to 16 bytes long. The third, CRC_21, is used for frames in CAN FD format with a data field longer than 16 bytes. Each polynomial results in a Hamming distance of 6. At the start of the frame, all three CRC polynomials are calculated concurrently. The values of the EDL bit and the DLC field select the CRC sequence to be transmitted. When receiving a message, FlexCAN decodes EDL and DLC to select the adequate CRC polynomial to check for a CRC error.

In CAN FD format frames, stuff bits are included in the bit stream for CRC calculation. In Classical CAN format frames, stuff bits are not included. After the transmission of the last bit relevant to the CRC calculation, CAN FD CRC (FDCRC) stores the calculated CRC for the transmitted message. This storage is performed with adequate length for the type of message, for CAN FD and non-FD messages. Cyclic Redundancy Check (CRCR) reports a valid CRC for Classical CAN messages only.

In CAN FD format frames, the CAN bit stuffing method changes for the CRC sequence, so the stuff bits are inserted at fixed positions. When FlexCAN is transmitting a CAN FD frame, a fixed stuff bit is inserted just before the first bit of the CRC sequence. This insertion occurs even if the last bits of the preceding field do not fulfill the CAN stuff condition. Additional stuff bits are inserted after each fourth bit of the CRC sequence. The value of any fixed stuff bit is the inverse value of its preceding bit. When FlexCAN receives a CAN FD frame, it discards the fixed stuff bits from the bit stream for the CRC check. A stuff error is detected if the fixed stuff bit has the same value as its preceding bit.

59.3.10.2.5 CAN FD errors

FlexCAN detects errors in CAN FD frames the same way as in Classical CAN frames. The error counters ECR[RXERRCNT] and ECR[TXERRCNT] accumulate the counts of RX and TX errors, respectively, for both FD and non-FD frames indiscriminately. Two extra error counters, ECR[RXERRCNT_FAST] and ECR[TXERRCNT_FAST], accumulate RX and TX errors occurring in the data phase of CAN FD frames with BRS = 1 only. The rules for updating the error counters are the same for both CAN FD and non-FD frames (see Error Counter (ECR)).

These error flags report errors in both CAN FD and non-FD frames:

- ESR1[BIT1ERR]
- ESR1[BIT0ERR]
- ESR1[ACKERR]
- ESR1[CRCERR]
- ESR1[FRMERR]
- ESR1[STFERR]

If CTRL1[ERRMSK] = 1, they also generate the ERRINT interrupt.

These additional error flags indicate the occurrence of errors in the data phase of CAN FD frames with BRS = 1:

- ESR1[BIT1ERR_FAST]
- ESR1[BIT0ERR_FAST]
- ESR1[CRCERR_FAST]
- ESR1[FRMERR_FAST]
- ESR1[STFERR_FAST]

No ACKERR is detected in the data phase of a CAN FD frame. Fault confinement status reported in ESR1[FLTCONF] is the same for both CAN FD and Classical CAN frames, and is based on ECR[RXERRCNT] and ECR[TXERRCNT] only. Information in ECR[RXERRCNT_FAST] and ECR[TXERRCNT_FAST] may be considered as status to help detect the error nature related to the bit rate value.

When FlexCAN detects an error while transmitting or receiving a CAN FD message in the data phase, it immediately switches:

- Back to the arbitration phase, and
- Back to the nominal rate to start an error flag.

59.3.10.2.6 CAN FD synchronization

Resynchronization and hard synchronization occur in CAN FD frames in the same way as in Classical CAN ones. A hard synchronization is also performed at the recessive-to-dominant edge from EDL to R0 in CAN FD format frames. FlexCAN does not resynchronize when transmitting in the CAN FD data phase.

59.3.10.3 Transceiver delay compensation

The CAN FD protocol allows the transmission and reception of data at a higher bit rate than the nominal rate used in the arbitration phase, when BRS = 1 in the message. This feature enables the use of rates up to 8 Mbit/s.

During the data phase of a CAN FD frame, if the transmitter cannot receive its own latest transmitted bit at the sample point of that bit, it detects a bit error. When bit rate switching is enabled (BRS = 1), the CAN bit time in the data phase can become shorter than the loop delay of the transceiver. This condition impedes the correct comparison between the transmitted bit and the received bit within the current CAN bit time interval.

The transceiver delay compensation (TDC) process defines a secondary sample point where the transmitted bit is correctly compared to the received bit to check for bit errors.

You can enable the TDC mechanism via FDCTRL[TDCEN] or ETDC[ETDCEN]. The TDC mechanism is effective only during the data phase of FD frames with BRS = 1. It has no effect on either non-FD frames or FD frames transmitted at the normal bit rate. When the transmitted message has BRS = 1, TDC is active from the sample point of the BRS bit until the sample point of the CRC Delimiter bit. When TDC is active, the real received bit is compared to the delayed transmitted bit, where the delay is calculated based on the measured transceiver loop delay.

NOTE

The transmitters using TDC disregard the value of the CRC delimiter bit. A global error at the end of the CRC field causes the receivers to send error frames that the transmitter detects during Acknowledge or End of Frame.

For every transmitted FD frame with BRS = 1, the transition from the recessive EDL bit to the dominant R0 bit triggers the delay measurement (as shown in Figure 441). The loop delay is measured in Protocol Engine (PE) clock periods (CANCLK, see Protocol timing), from the transmitted EDL-R0 edge to the received EDL-R0 edge. The measured loop delay time added to an offset value specified in FDCTRL[TDCOFF] or ETDC[ETDCOFF] determines the position of the secondary sample point. FDCTRL[TDCVAL] or ETDC[ETDCVAL] stores the result of this calculation. The TDCVAL and ETDCVAL value saturates at its maximum value of 63 CANCLK and 255 CANCLK when the delay measurement is too long.

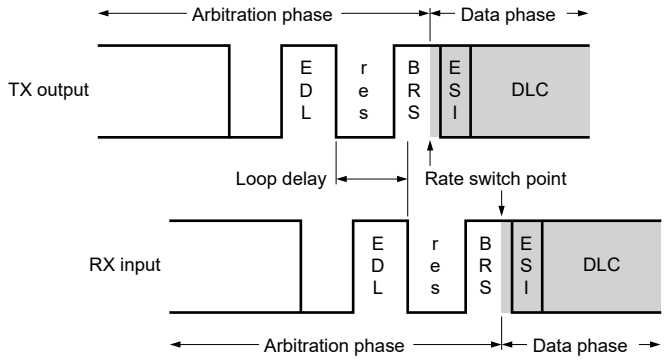


Figure 441. Transceiver loop delay measurement

The measured loop delay is not enough to define the secondary sample point, because it relates to the CAN bit edges. The transceiver delay compensation offset FDCTRL[TDCOFF] or ETDC[ETDCOFF] is used to shift the secondary sample point to an intermediate point inside the bit time, far away from its edges. The value of FDCTRL[TDCOFF] or ETDC[ETDCOFF] cannot be larger than the CAN bit duration in the data phase.

If the secondary sample point is set very near the CAN bit edge (SYNC field), problems may occur during the bit sampling in the data phase. For the TDC to work reliably, the offset must use optimal settings. To ensure that bit sampling is performed in the best region, configure the TDC offset as shown in this equation:

$$Offset = (FPSEG1 + FPROPSEG + 2) \times (FPRESDIV + 1)$$
or
$$Offset = (DTSEG1 + 2) \times (EDPRESDIV + 1), \textit{ if ETDCEN}$$

Equation 14. TDC offset calculation

Figure 442 shows the SSP position when these settings are used.

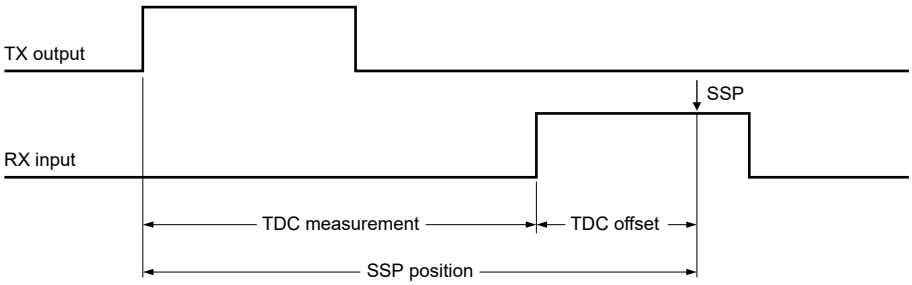


Figure 442. SSP position with optimal values

Alternatively, if CTRL2[BTE] and ETDC[ETDCEN] are 1, you can write 1 to ETDC[TDMDIS] to disable the transceiver delay measurement. In this case, only ETDC[ETDCOFF] defines the SSP position. Figure 443 shows the secondary sample point position when the transceiver delay measurement is disabled.

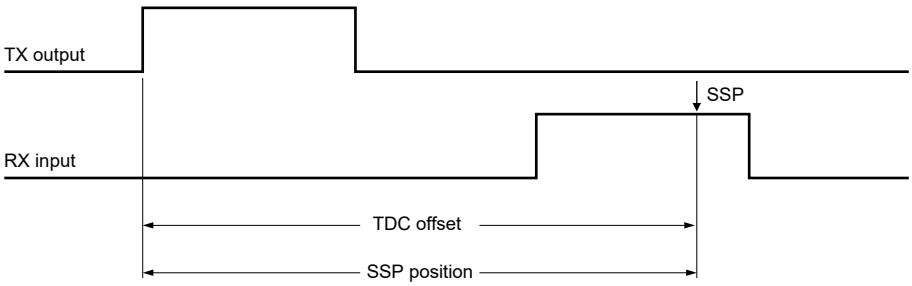


Figure 443. SSP position with TDC measurement disabled by ETDCEN and TDMDIS

During the data phase of CAN FD frames with bit rate switching enabled, at the onset of every TX CAN bit:

- The transmitted TX bit value is temporarily stored in a buffer.
- A time countdown based on FDCTRL[TDCVAL] or ETDC[ETDCVAL] is started. This countdown ends with the comparison of the received RX bit (delayed by the external loop delay plus the specified offset) to the stored TX bit.

If a bit error is detected at the secondary sample point, FlexCAN issues an error flag to the CAN bus at the next sample point.

During the arbitration phase, delay compensation is always disabled. During the data phase, the TDC mechanism of FlexCAN can compensate a maximum delay of 3 CAN bit times − 2 Tq. Beyond this limit, the FDCTRL[TDCFAIL] or ETDC[ETDCFAIL] flag is set. The flag indicates when the TDC mechanism is out of range and is unable to compensate the transceiver loop delay.

59.3.10.4 Remote frames

A remote frame is a special type of frame. You can program a message buffer to be a remote request frame by configuring the message buffer as Transmit with the RTR = 1. After the remote request frame is transmitted successfully, the message buffer becomes a receive message buffer, with the same ID as before.

When FlexCAN receives a remote request frame, the frame can be treated in different ways, depending on remote request storing (CTRL2[RRS]) and RX FIFO Enable (MCR[RFEN]):

- If RRS = 0, the ID of the frame is compared to the IDs of the transmit message buffers with the CODE field 1010b. If a matching ID exists, this message buffer frame is transmitted. If the matching message buffer has the RTR = 1, FlexCAN transmits a remote frame as a response. The received remote request frame is not stored in a receive buffer. It is only used to trigger a transmission of a frame in response.

  The mask registers are not used in remote frame matching, and all ID bits (except RTR) of the incoming received frame should match. If a remote request frame is received and matches a message buffer, this message buffer immediately enters the internal arbitration process. However, it is considered a normal TX message buffer, with no higher priority. The data length of this frame is independent of the DLC field in the remote frame that initiated its transmission.

- If CTRL2[RRS] = 1, the ID of the frame is compared to the IDs of the receive message buffers with the CODE field 0100b, 0010b, or 0110b. If a matching ID exists, this message buffer stores the remote frame in the same fashion as a data frame. No automatic remote response frame is generated. The mask registers are used in the matching process.
- If MCR[RFEN] = 1, FlexCAN does not generate an automatic response for remote request frames that match the Legacy FIFO filtering criteria. If the remote frame matches one of the target IDs, it is stored in the Legacy FIFO and presented to the CPU.

  For filtering formats A and B (see Legacy RX FIFO structure), it is possible to select whether remote frames are accepted or not. For format C, remote frames are always accepted if they match the ID. Remote request frames are considered as normal frames. They generate a Legacy FIFO overflow when a successful reception occurs and the Legacy FIFO is already full.
- If ERFCR[ERFEN] = 1, FlexCAN does not generate an automatic response for remote request frames that match the Enhanced RX FIFO filtering criteria. Remote Request Frames are considered normal frames. They generate an Enhanced RX FIFO overflow when a successful reception occurs and the enhanced RX FIFO is already full.

NOTE

There is no remote frame in the CAN FD format. A fixed dominant RRS bit replaces the RTR bit. FlexCAN receives and transmits remote frames in the Classical CAN format.

59.3.10.5 Overload frames

When a dominant bit is detected on the CAN bus in these locations, FlexCAN transmits overload frames:

- The first or second bit of Intermission.
- The seventh bit (last) of End of Frame field (RX frames).
- The eighth bit (last) of Error Frame Delimiter or Overload Frame Delimiter.

59.3.10.6 Message buffer timestamp

The value of the free-running timer is sampled at the beginning of the Identifier field on the CAN bus. This value is stored at the end of move-in in the TIME_STAMP field of a message buffer, providing network behavior regarding time.

When CTRL2[TIMER_SRC] = 1, an external time tick continuously clocks the free-running timer.

When CTRL2[TIMER_SRC] = 0, the FlexCAN bit clock clocks the free-running timer, which defines the baud rate on the CAN bus. During a message transmission or reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate.

The free-running timer is not incremented during Disable, Doze, Stop, and Freeze modes. It can be reset upon a specific frame reception, enabling network time synchronization. See CTRL1[TSYN].

Alternatively, by configuring CTRL2[MBTSBASE], the timestamp of the message buffer can capture the lower or higher 16 bits of the high-resolution dedicated counter.

59.3.10.7 High-resolution timestamp

The high-resolution timestamp (HR_TIME_STAMP) uses a dedicated timer with a 32-bit counter operating in free-running mode. CTRL2[TSTAMPCAP] enables the high-resolution timestamp. When this field is not zero, the dedicated 32-bit counter value is captured during a valid CAN frame and stored in an HR_TIME_STAMP*n* register.

Each HR_TIME_STAMP*n* corresponds to a specific message buffer. For example, HR_TIME_STAMP0 stores the 32-bit timestamp associated with message buffer 0. HR_TIME_STAMP1 stores the 32-bit timestamp associated with message buffer 1, and so on.

The counter value is captured according to CTRL2[TSTAMPCAP]. For classical CAN frames, the capture points can be the start of frame bit or the point in time a CAN frame is considered valid. This valid point is the seventh bit of end of frame for transmission and the sixth bit of end of frame for reception.

For CAN FD frames, the capture points can be:

- The start of frame

- The point in time a CAN FD frame is considered valid
- The res bit of a CAN FD frame

The 16-bit timestamp of the message buffer can be configured to capture the lower or higher 16 bits of the high-resolution timer. This configuration is made by CTRL2[MBTSBASE].

59.3.10.8 Protocol timing

Figure 444 shows the structure of the clock generation circuitry that feeds the CAN Protocol Engine (PE) submodule. The clock source field CTRL1[CLKSRC] defines whether the internal clock is connected to the output of a crystal oscillator (oscillator clock) or to the peripheral clock. To guarantee reliable operation, select the clock source when the module is in Disable mode (MCR[MDIS] = 1).

NOTE

To identify the proper clock source, see the clock distribution chapter (module clocks table).

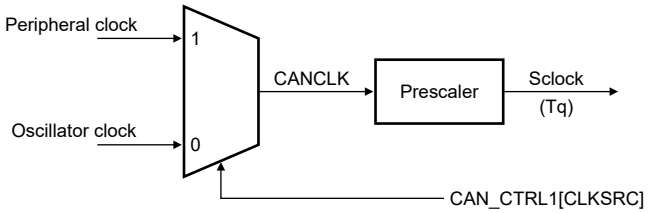


Figure 444. CAN engine clocking scheme

Select the oscillator clock whenever a tight tolerance (up to 0.1%) is required in the CAN bus timing. The crystal oscillator clock has better jitter performance than the peripheral clock.

59.3.10.8.1 Bit timing configuration

FlexCAN supports various means to configure bit timing parameters required by the CAN protocol. Control 1 (CTRL1) has various fields to control bit timing parameters:

- CTRL1[PRESDIV]
- CTRL1[PROPSEG]
- CTRL1[PSEG1]
- CTRL1[PSEG2]
- CTRL1[RJW]

CAN Bit Timing (CBT) extends the range of the CAN bit timing variables in CTRL1. Enhanced Data Phase CAN Bit Timing (EDCBT) provides a second set of CAN bit timing variables to be applied at the data phase of CAN FD frames with the Bit Rate Switch (BRS) = 1.

Enhanced Nominal CAN Bit Timing (ENCBT) extends the range of CAN bit timing variables in CBT. Enhanced Nominal CAN Bit Timing (ENCBT) extends the range of CAN bit timing variables in FDCBT. When using ENCBT and EDCBT, you must program the nominal bit timing and data phase serial clock (Sclock) dividers in Enhanced CAN Bit Timing Prescalers (EPRS).

NOTE

When the CAN FD feature is enabled, always write 1 to CBT[BTF] or CTRL2[BTE] and specify the CAN bit timing variables in CBT or ENCBT. See CAN Bit Timing (CBT) or Enhanced Nominal CAN Bit Timing (ENCBT).

CTRL1[PRESDIV], and its extended range CBT[EPRESDIV] (or EPRS[ENPRESDIV]) and FDCBT[FPRESDIV] (or EPRS[EDPRESDIV]) for the data phase bits of CAN FD messages, defines the prescaler value that generates the serial

clock (Sclock). (See Equation 15.) The period of Sclock defines the time quantum used to compose the CAN waveform. A time quantum (Tq) is the atomic unit of time managed by the CAN engine. It is the smallest time unit for all configuration values.

$$Tq = \frac{(PRESDIV + 1)}{f_{CANCLK}}$$

Equation 15. Time quantum

The bit rate, which defines the rate the CAN message is received or transmitted, is calculated with the formula:

$$CAN\ bit\ time = (Number\ of\ time\ quanta\ in\ 1\ bit\ time) \times Tq$$

$$Bit\ rate = \frac{1}{CAN\ bit\ time}$$

Equation 16. CAN bit time and baud rate

59.3.10.8.2 Bit time segments

A bit time is subdivided into three segments as shown in Figure 445. See also Figure 446 , Figure 447, and Table 943.

NOTE

For further explanation of the underlying concepts, see ISO 11898-1:2015. See also *CAN Specification Version 2.0, Part A and Part B* for bit timing.

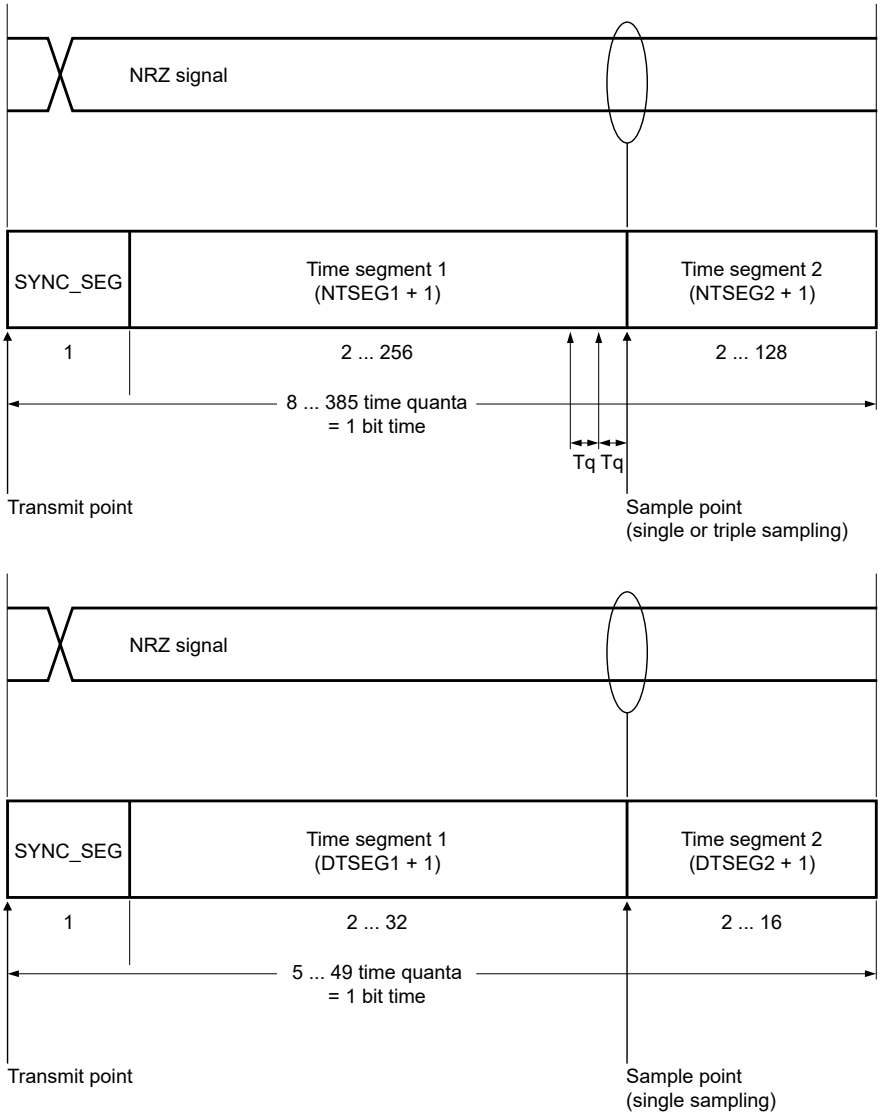


Figure 445. Segments within the bit timing (example using ENCBT and EDCBT bit timing variables)

The three bit time segments are:

- SYNC_SEG—this segment has a fixed length of one time quantum. Signal edges are expected to occur within this section.
- Time Segment 1—this segment includes the propagation segment and the phase segment 1 of the CAN standard.

  It can be programmed by configuring CTRL1[PROPSEG] and CTRL1[PSEG1] so that the sum (plus 2) is 2–16 time quanta. When CBT[BTF] = 1, FlexCAN uses CBT[EPROPSEG] and CBT[EPSEG1] so that the sum (plus 2) is 2–96 time quanta. For messages in CAN FD format with the BRS = 1, FlexCAN uses FDCBT[FPROPSEG] and FDCBT[FPSEG1] so that the sum (plus 1) is 2–39 time quanta.

  If CTRL2[BTE] = 1, FlexCAN uses ENCBT[NTSEG1] to configure time segment 1 to 2–256 time quanta. For the data phase in CAN FD messages with BRS = 1, EDCBT[DTSEG1] must be used for configuring time segment 1 to 2–32 time quanta.
- Time Segment 2—this segment represents the phase segment 2 of the CAN standard.

  It can be programmed by configuring CTRL1[PSEG2] (plus 1) to be 2–8 time quanta. When CBT[BTF] = 1, FlexCAN uses CBT[EPSEG2] so that its value (plus 1) is 2–32 time quanta. For messages in CAN FD format with the BRS = 1, FlexCAN uses

FDCBT[FPSEG2] instead, so that its value (plus 1) is 2–8 time quanta. Time segment 2 cannot be smaller than the Information Processing Time (IPT), which is 2 time quanta in FlexCAN.

If CTRL2[BTE] = 1, FlexCAN uses ENCBT[NTSEG2] to configure time segment 2 to 2–128 time quanta. For the data phase in CAN FD messages with BRS = 1, EDCBT[DTSEG2] must configure time segment 2 to 2–16 time quanta.

NOTE

The bit time defined by the above time segments must not be smaller than five time quanta. For bit time calculations, use an Information Processing Time (IPT) of two, which is the value implemented in the FlexCAN module.

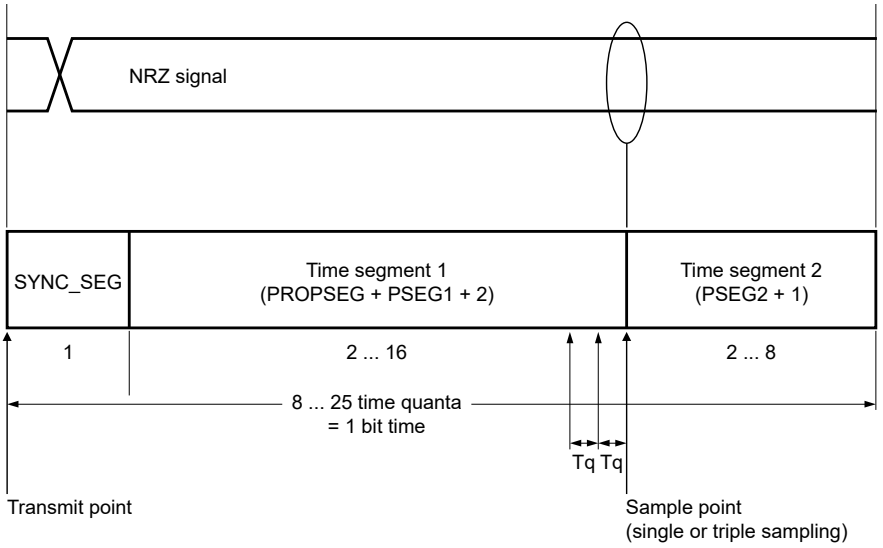


Figure 446. Segments within the bit time (example using CTRL1 bit timing variables for Classical CAN format)

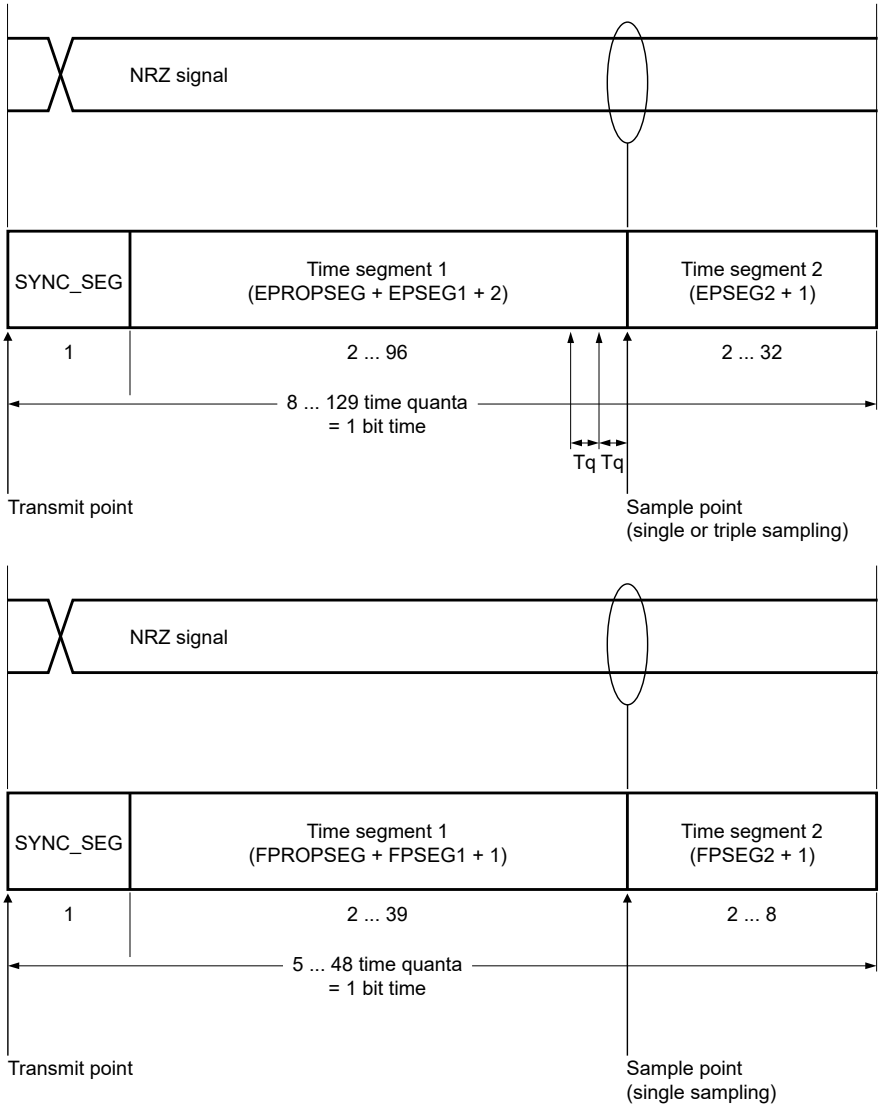


Figure 447. Segments within the bit time (example using CBT and FDCBT bit timing variables for CAN FD format)

Table 943. Time segment syntax

Syntax	Description
SYNC_SEG	Period during which the system expects transitions to occur on the bus
TSEG1	Period corresponding to the sum of PROPSEG and PSEG1
TSEG2	Period corresponding to the PSEG2 value
Transmit point	Point at which a node in Transmit mode transfers a new value to the CAN bus
Sample point	Point at which a node samples the bus. If the option of three samples per bit is selected, this point marks the position of the third sample.

Table 944 gives some examples of the CAN-compliant segment settings for Classical CAN format (Bosch CAN 2.0B) (non-FD) messages.

Table 944. Bosch CAN 2.0B standard compliant bit time segment settings

Time segment 1	Time segment 2	Resynchronization jump width
5 to 10	2	1 to 2
4 to 11	3	1 to 3
5 to 12	4	1 to 4
6 to 13	5	1 to 4
7 to 14	6	1 to 4
8 to 15	7	1 to 4
9 to 16	8	1 to 4

NOTE

You must ensure the bit time settings comply with the CAN Protocol standard (ISO 11898-1:2015).

59.3.10.8.3 Calculating peripheral clocks

A CAN bit can be used as a measure of duration (for example, estimating the occurrence of a CAN bit event in a message). When a CAN bit is used in this way, the number of peripheral clocks in one CAN bit (NumClkBit) can be calculated as:

$$NumClkBit = \frac{f_{SYS}}{f_{CANCLK}} \times (PRESDIV + 1) \times (PROPSEG + PSEG1 + PSEG2 + 4)$$

Equation 17. Number of peripheral clocks per CAN bit when CTRL2[BTE] = 0

Or, if CTRL2[BTE] = 1:

$$NumClkBit = \frac{f_{SYS}}{f_{CANCLK}} \times (ENPRESDIV + 1) \times (NTSEG1 + NTSEG2 + 3)$$

Equation 18. Number of peripheral clocks per CAN bit when CTRL2[BTE] = 1

Where:

- NumClkBit is the number of peripheral clocks in one CAN bit.
- f_{CANCLK} is the Protocol Engine (PE) Clock (see Figure 444), in Hz.
- f_{SYS} is the frequency of operation of the system (CHI) clock, in Hz.
- PSEG1 is the value of CTRL1[PSEG1].
- PSEG2 is the value of CTRL1[PSEG2].
- PROPSEG is the value of CTRL1[PROPSEG].
- PRESDIV is the value in CTRL1[PRESDIV].
- ENPRESDIV is the value of EPRS[ENPRESDIV].
- NTSEG1 is the value of ENCBT[NTSEG1].
- NTSEG2 is the value of ENCBT[NTSEG2].

The formula above is also applicable to the alternative CAN bit timing variables described in:

- CAN Bit Timing (CBT)

- Enhanced Nominal CAN Bit Timing (ENCBT)
- CAN FD Bit Timing (FDCBT)
- Enhanced Nominal CAN Bit Timing (ENCBT)

For example, 180 CAN bits = (180 × NumClkBit) peripheral clock periods.

59.3.10.9 Arbitration and matching timing

During normal reception and transmission, the matching, arbitration, move-in, and move-out processes are executed during certain time windows inside the CAN frame. These windows are shown in the following figures.

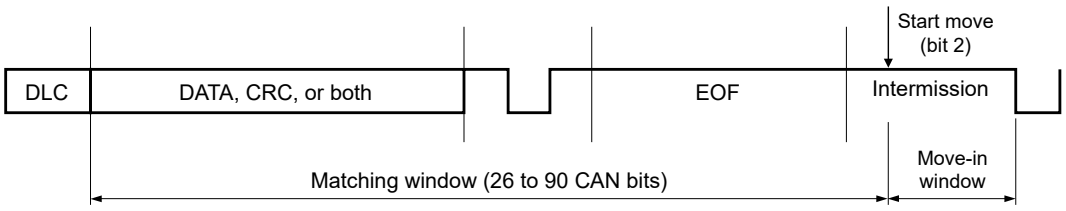


Figure 448. Matching and move-in time windows

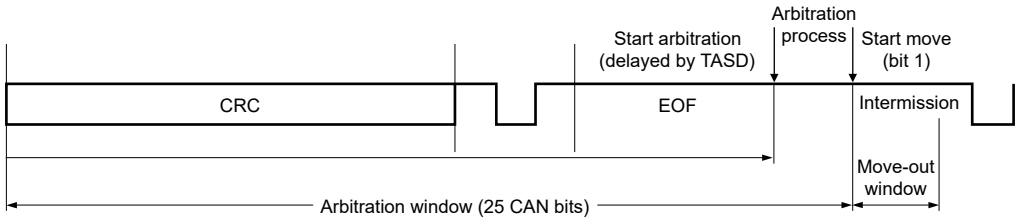


Figure 449. Arbitration and move-out time windows

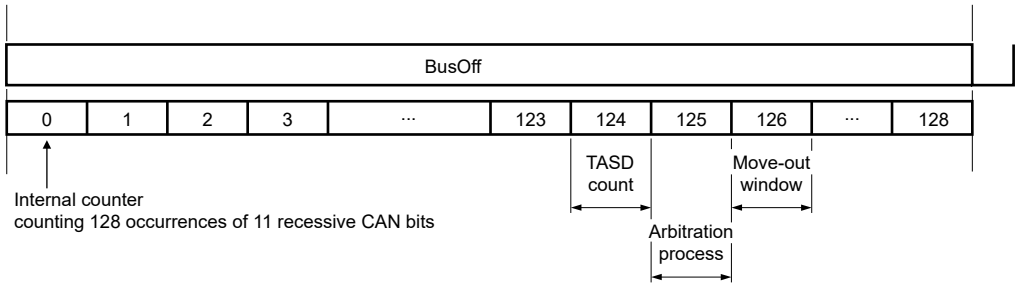


Figure 450. Arbitration at the end of bus off and move-out time windows

NOTE

In these figures, the matching and arbitration timing do not consider delays caused by concurrent memory access due to the CPU or other internal FlexCAN subblocks.

59.3.10.10 TX arbitration start delay

TX Arbitration Start Delay (CTRL2[TASD]) indicates the number of CAN bits that FlexCAN uses to delay the TX arbitration process starting point from the first bit of the CRC field of the current frame. This variable can be written only in Freeze mode; FlexCAN blocks it in other modes.

The ability of the CPU to reconfigure message buffers for transmission after the end of the internal arbitration process impacts transmission performance. In the arbitration process, FlexCAN finds the winner MB for transmission (see Arbitration process). If

the arbitration ends too early (before the first bit of the Intermission field) the CPU may reconfigure some TX message buffers. It is possible that the winning message buffer is no longer the best candidate to be transmitted.

TASD can optimize the transmission performance by defining the arbitration start point, as shown in Figure 451, based on factors such as:

- Peripheral-to-oscillator clock ratio
- CAN bit timing variables that determine the CAN bit rate
- Number of message buffers in use by the matching and arbitration processes

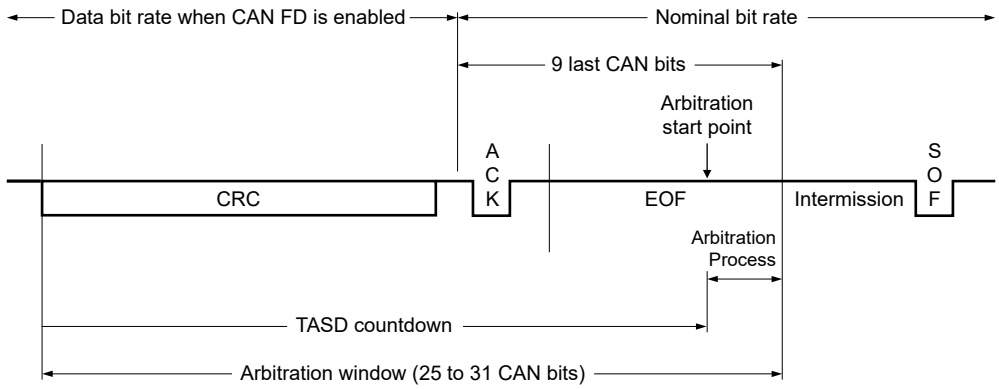


Figure 451. Optimal TX arbitration start point

The duration of an arbitration process, in terms of CAN bits, is:

- Directly proportional to the number of available message buffers
- Directly proportional to the CAN bit rate
- Inversely proportional to the peripheral clock frequency

The optimal arbitration timing occurs when the last message buffer is scanned immediately before the first bit of the Intermission field of a CAN frame. For instance, if the following are true:

- There are few message buffers.
- The peripheral-to-oscillator clock ratio is high.
- The CAN baud rate is low.

Then the arbitration can be placed closer to the end of the frame, adding more delay to its starting point, and vice versa.

If CTRL2[TASD] = 0, the arbitration start is not delayed, and more time is reserved for arbitration. Alternatively, if CTRL2[TASD] is close to 24, the CPU can configure a TX message buffer later, and less time is reserved for arbitration. If too little time is reserved for arbitration, FlexCAN may not be able to find a winner MB in time. The transmitted arbitration winner may not have the best chance to win the bus arbitration against external nodes on the CAN bus.

The optimal TASD value can be calculated as follows:

For CAN FD frames and (MAXMB + 1) ≤ NMB_{END}

$$TASD = 31 - \frac{2 * (MAXMB + 1) + 4}{CPCB_N}$$

For CAN FD frames and (MAXMB + 1) > NMB_{END}

$$TASD = 22 - \frac{2 * (MAXMB + 1) - NMB_{END}}{CPCB_F}$$

For non-FD frames

$$TASD = 25 - \frac{2 * (MAXMB + 1) + 4}{CPCB}$$

Equation 19. Optimal value for TASD

Where:

$$NMB_{END} = \frac{(9 \times CPCB_N) - 4}{2}$$

$$BITRATE_N = \left(\frac{f_{CANCLK}}{[1 + (EPSEG1 + 1) + (EPSEG2 + 1) + (EPROPSEG + 1)] \times (EPRESDIV + 1)} \right)$$

$$BITRATE_F = \left(\frac{f_{CANCLK}}{[1 + (FPSEG1 + 1) + (FPSEG2 + 1) + FPROPSEG] \times (FPRESDIV + 1)} \right)$$

$$CPCB_N = \frac{f_{SYS}}{BITRATE_N}$$

$$CPCB_F = \frac{f_{SYS}}{BITRATE_F}$$

$$CPCB = CPCB_N$$

Equation 20. Variables used in TASD calculation

- MAXMB is the value in MCR[MAXMB].
- NMB_{END} is the number of message buffers that the arbitration process can scan during the last nine CAN bits at the end of a frame. (See Equation 20.)
- $BITRATE_N$ is the CAN bit rate in bits per second calculated by the nominal CAN bit time variables.
- $BITRATE_F$ is the CAN bit rate in bits per second calculated by the data CAN bit time variables.
- $CPCB_N$ is the number of peripheral clocks per CAN bit in nominal bit rate for CAN FD frames.
- $CPCB_F$ is the number of peripheral clocks per CAN bit in data bit rate for CAN FD frames.
- CPCB is the number of peripheral clocks per CAN bit for non-FD frames.
- f_{CANCLK} is the oscillator clock, in Hz.
- f_{SYS} is the peripheral clock, in Hz.
- EPSEG1 is the value in CBT[EPSEG1] (CTRL1[PSEG1] can also be used).
- EPSEG2 is the value in CBT[EPSEG2] (CTRL1[PSEG2] can also be used).
- EPROPSEG is the value in CBT[EPROPSEG] (CTRL1[PROPSEG] can also be used).
- EPRESDIV is the value in CBT[EPRESDIV] (CTRL1[PRESDIV] can also be used).
- FPSEG1 is the value in FDCBT[FPSEG1].

- FPSEG2 is the value in FDCBT[FPSEG2].
- FPROPSEG is the value in FDCBT[FPROPSEG].
- FPRESDIV is the value in FDCBT[FPRESDIV].
- NTSEG1 is the value in ENCBT[NTSEG1].
- NTSEG2 is the value in ENCBT[NTSEG2].
- ENPRESDIV is the value in EPRS[ENPRESDIV].
- DTSEG1 is the value in EDCBT[DTSEG1].
- DTSEG2 is the value in EDCBT[DTSEG2].
- EDPRESDIV is the value in EPRS[EDPRESDIV].

If CTRL2[BTE] = 1, then:

$$BITRATE_N = \frac{f_{CANCLK}}{[1 + (NTSEG1 + 1) + (NTSEG2 + 1)] \times (ENPRESDIV + 1)}$$

Equation 21. Nominal baud rate when CTRL2[BTE] = 1

$$BITRATE_F = \frac{f_{CANCLK}}{[1 + (DTSEG1 + 1) + (DTSEG2 + 1)] \times (EDPRESDIV + 1)}$$

Equation 22. Fast baud rate when CTRL2[BTE] = 1

See also Protocol timing for more details.

59.3.10.10.1 TASD configuration examples

The following tables show the TASD value calculated for some configuration cases.

Case 1:

- Clock ratio = 2:1 (for example, peripheral clock 80 MHz and oscillator clock 40 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 945. TASD values in Case 1

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	24	8.0
64	23	8.0
96	22	8.0

Case 2:

- Clock ratio = 1:1 (for example, peripheral clock 40 MHz and oscillator clock 40 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 946. TASD values in Case 2

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid

Table continues on the next page...

Table 946. TASD values in Case 2 (continued)

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
32	23	6.67
54	22	5.0
64	21	3.33
96	20	1.6

Case 3:

- Clock ratio = 2:1 (for example, peripheral clock 40 MHz and oscillator clock 20 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 947. TASD values in Case 3

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	23	4.0
54	22	4.0
64	21	3.33
96	20	1.54

59.3.11 Clocks

The following table describes the clock sources for FlexCAN. See the Clock Controller Module (CCM) for clock setting, configuration, and gating information.

Table 948. FlexCAN clocks

Clock name	Description
MODULE_CLK (system_clk)	Peripheral clock
MODULE_CLK_CHI (host_clock)	Control Host Interface (CHI) clock
MODULE_CLK_PE (protocol_engine_clock)	Protocol Engine (PE) clock
MODULE_CLK_PE_NOGATE (protocol_engine_clock_nogate)	Protocol Engine clock (no gating)
MODULE_CLK_S (system_clock_nogate)	Peripheral access clock

59.3.11.1 Clock domains and restrictions

FlexCAN has two clock domains asynchronous to each other:

- The bus domain feeds the Control Host Interface (CHI) submodule.
- The oscillator domain feeds the CAN Protocol Engine (PE) submodule.

When CTRL1[CLKSRC] = 1, synchronous operation occurs because both domains are connected to the peripheral clock. This setting creates a 1:1 ratio between the peripheral and oscillator clocks.

When the two domains are connected to clocks with different frequencies or phases, the frequency relationship between the two clock domains is restricted. In asynchronous operation, the bus domain clock frequency must always be greater than the oscillator domain clock frequency.

NOTE

Asynchronous operation with a 1:1 ratio between peripheral and oscillator clocks is not allowed.

When performing matching and arbitration, FlexCAN must scan the whole message buffer memory during the time slot of one CAN frame, comprised of a number of CAN bits. To provide sufficient time for the scan, observe the following requirements:

- The peripheral clock frequency cannot be less than the oscillator clock frequency.
- There must be a minimum number of peripheral clocks per CAN bit, as specified in Table 949.

Table 949. Minimum number of peripheral clocks per CAN bit for Classical CAN format

Number of message buffers	Value of MCR[RFEN]	Value of ERFCR[ERFEN]	Minimum number of peripheral clocks per CAN bit
16	0	0	16
32	0	0	16
64	0	0	25
96	0	0	37
16	1	0	16
32	1	0	17
64	1	0	30
96	1	0	42
16	0	1	16
32	0	1	19
64	0	1	31
96	0	1	42

For classical frame format, the minimum number of peripheral clocks per CAN bit specified in Table 949 determines the minimum peripheral clock frequency for a given number of message buffers and for an expected CAN bit rate. The CAN bit rate depends on the number of time quanta in a CAN bit. This number can be defined by adjusting one or more of the bit timing values contained in:

- Control 1 (CTRL1)
- CAN Bit Timing (CBT)
- Enhanced Nominal CAN Bit Timing (ENCBT)

The time quantum (Tq) is defined in Protocol timing. The minimum number of time quanta per CAN bit must be eight, so the oscillator clock frequency should be at least eight times the CAN bit rate.

59.3.11.1.1 Clock restrictions for CAN FD

For CAN FD frame format, some constraints must be satisfied. The equation below calculates the number of peripheral clocks per CAN bit in nominal bit rate (NumClkNomBit).

$$NumClkNomBit = \frac{f_{SYS}}{f_{CANCLK}} \times (PRESDIV + 1) \times (PROPSEG + PSEG1 + PSEG2 + 4)$$

$$= \frac{f_{SYS}}{NomBitRate}$$

Equation 23. Number of peripheral clocks per nominal CAN bit

Where PRESDIV, PSEG1, and PSEG2 are CAN bit time values in Control 1 (CTRL1). Alternatively, EPRESDIV, EPSEG1, and EPSEG2 values in CAN Bit Timing (CBT) or the values of EPRS[ENPRESDIV], ENCBT[NTSEG1], and ENCBT[NTSEG2] can be used instead. NumClkNomBit can also be calculated as a function of the expected nominal bit rate used in the arbitration phase (NomBitRate), as shown in the equation above.

The number of CAN bits in the data phase of an FD frame with BRS = 1 (fast CAN bits) depends on the number of data bytes in the payload. The number of fast CAN bits (NumOfFastBits) can be determined in Table 950. Having fewer data bytes means having fewer fast CAN bits. It also means that less time is available for FlexCAN to scan the whole message buffer memory during the internal matching and arbitration processes.

Table 950. Number of fast CAN bits in a CAN FD frame

Minimum number of data bytes	DLC field	NumOfFastBits
0	0h	21
1	1h	29
2	2h	37
3	3h	45
4	4h	53
5	5h	61
6	6h	69
7	7h	77
8	8h	85
12	9h	117
16	Ah	149
20	Bh	186
24	Ch	218
32	Dh	282
48	Eh	410
64	Fh	538

The critical part of a CAN FD frame is during the data phase, where the CAN bit rate is faster than in the arbitration phase. The minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) can be calculated to guarantee that enough time is available for FlexCAN to scan the message buffer memory during reception and transmission. The equation below calculates this constraint.

$$MinNumClkFastBit_A = \frac{(8.5 \times MaxNumOfMb) + [ERFEN \times (2 \times NFE + 4)] + 64 - (9 \times NumClkNomBit)}{NumOfFastBits}$$

Equation 24. Minimum number of peripheral clocks per fast CAN bit for FlexCAN scan process

Where MaxNumOfMb is the maximum number of available message buffers defined in MCR[MAXMB]. NFE and ERFEN are the fields defined in Enhanced RX FIFO Control (ERFCR).

The clock-domain-crossing circuit between the CHI and PE subblocks also imposes a minimum number of peripheral clocks per fast CAN bit. This minimum is required for the handshake mechanism to work properly without losing status information through the interface, as shown in the equation below.

$$MinNumClkFastBit_B = 3 \times \left(1 + \frac{f_{SYS}}{f_{CANCLK}}\right)$$

Equation 25. Minimum number of peripheral clocks per fast CAN bit for FlexCAN clock domain interface

Therefore, the larger of the two values calculated above determines the minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit).

$$MinNumClkFastBit = Maximum\ (MinNumClkFastBitA, MinNumClkFastBitB)$$

Equation 26. Minimum number of peripheral clocks per fast CAN bit

Then, the maximum CAN bit rate in the data phase of CAN FD frames (DataBitRateMAX) can be calculated as below.

$$DataBitrate_{MAX} = \frac{f_{CANCLK}}{ROUNDUP\left(\frac{MinNumClkFastBit \times f_{CANCLK}}{f_{SYS}}\right)}$$

Equation 27. Maximum achievable baud rate for data phase

These factors affect the maximum data bit rate attainable by FlexCAN in CAN FD mode:

- The peripheral and oscillator clock frequencies
- The maximum number of message buffers
- The expected nominal bit rate

Also, the data bit rate depends on the minimum payload size of FD frames used in a given application.

To illustrate how the configuration of FlexCAN variables affects the CAN FD bit rate, consider this application example:

- The peripheral clock frequency is set to 50 MHz
- The oscillator clock frequency is set to 40 MHz

1. Considering the nominal bit rate as 1 Mbit/s, the number of peripheral clocks per CAN bit in nominal bit rate is calculated as below.

   $$NumClkNomBit = \frac{50 \times 10^6}{1 \times 10^6} = 50$$

   Equation 28. Calculation example for number of peripheral clocks per nominal CAN bit

2. The number of fast CAN bits (NumOfFastBits) is determined in Table 950. For example, if the minimum payload in FD frames is 8 bytes, there are 85 CAN bits in the data phase.
3. Assuming the maximum number of message buffers is 96, and Enhanced RX FIFO is disabled, the minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) can be calculated.

$$MinNumClkFastBit_A = \frac{(8.5 \times 96) + 64 - (9 \times 50)}{85} = 5.06$$

Equation 29. Calculation example for number of peripheral clocks per fast CAN bit for FlexCAN scan process

$$MinNumClkFastBit_B = 3 \times \left(1 + \frac{50}{40}\right) = 6.75$$

Equation 30. Calculation example for number of peripheral clocks per fast CAN bit for FlexCAN clock domain interface

$$MinNumClkFastBit = Maximum\ (\ 5.06, 6.75\ ) = 6.75$$

Equation 31. Calculation example for number of peripheral clocks per fast CAN bit

4. The maximum CAN bit rate in the data phase can finally be found.

$$DataBitRate_{MAX} = \frac{40 \times 10^6}{ROUNDUP\left(\frac{6.75 \times 40 \times 10^6}{50 \times 10^6}\right)} = 6.667\ Mbps$$

Equation 32. Calculation example for maximum achievable baud rate

Even though the oscillator clock frequency (40 MHz) is adequate to generate a data rate of 8 Mbit/s in CAN FD mode, the specific FlexCAN configuration limits this rate to 6.667 Mbit/s. This limitation is mainly due to the low peripheral clock frequency that imposes the MinNumClkFastBitB bound.

Table 951 shows the maximum data rate for CAN FD with Enhanced RX FIFO disabled according to clock frequencies, payload size, and number of available message buffers. For some cases, if the number of available message buffers is reduced, FlexCAN can then achieve a data rate up to 8 Mbit/s.

Table 951. Maximum CAN bit rate in data phase on CAN FD frames with Enhanced RX FIFO disabled

Peripheral clock frequency (MHz)	Payload size	Number of available message buffers	Maximum data rate (Mbit/s)
40	8	94	6.667
40	8	114	>5.0
40	12	>117	6.667
40	12	128	5.714
50	12–64	128	6.667
60	8	126	8.0
60	12	128	8.0
67	6	128	8.0
80	3	128	8.0
100	0	128	8.0

59.3.12 Reset

You can reset FlexCAN in the following ways:

- Chip-level hard reset, which resets all memory-mapped registers asynchronously.
- Soft reset, in one of two ways:
  - MCR[SOFTRST], which resets some of the memory-mapped registers synchronously. To see which registers soft reset affects, see Table 954.
  - Chip-level soft reset, which has the same effect as MCR[SOFTRST].

Soft reset is synchronous and must follow an internal request-and-acknowledge procedure across clock domains. Therefore, it may take some time to propagate its effects fully. MCR[SOFTRST] remains 1 when soft reset is pending, so software can poll this field to identify when the reset has completed. Soft reset cannot be applied when clocks are shut down in a low-power mode. The low-power mode should be exited and the clocks resumed before applying soft reset.

The clock source should be selected when the module is in Disable mode (see CTRL1[CLKSRC]). After the clock source is selected and the module is enabled (MCR[MDIS] becomes 0), FlexCAN automatically enters Freeze mode. In Freeze mode:

1. FlexCAN is unsynchronized to the CAN bus.
2. MCR[HALT] and MCR[FRZ] become 1.
3. The internal state machines are disabled.
4. MCR[FRZACK] and MCR[NOTRDY] become 1.

The TX pin is in the recessive state and FlexCAN does not initiate any transmission or reception of CAN frames. Reset does not affect the message buffers and the RX Individual Mask registers, so they are not automatically initialized.

59.3.13 Interrupts

FlexCAN has many interrupt sources:

- Interrupts due to message buffers
- Interrupts due to interrupts combined via an OR operator from:
  - Message buffers
  - Bus Off
  - Bus Off Done
  - Error
  - Error Fast (errors detected in the data phase of CAN FD format messages with BRS = 1)
  - Wake Up
  - TX Warning
  - RX Warning

If its corresponding IMASK bit is 1, each message buffer can be an interrupt source. There is no distinction between TX and RX interrupts for a particular buffer, under the assumption that the buffer is initialized for either transmission or reception. Each buffer has an assigned flag bit in the IFLAG registers. When the corresponding buffer completes a successful transfer, the flag is set. When the CPU writes 1 to it, the flag is cleared (unless another interrupt is generated at the same time).

NOTE

The CPU must clear only the bit causing the current interrupt. For this reason, do not use bit manipulation instructions (BSET) to clear interrupt flags. These instructions may cause accidental clearing of interrupt flags which are set after entering the current interrupt handler.

If the Legacy RX FIFO is enabled (MCR[RFEN] = 1) and DMA is disabled (MCR[DMA] = 0), the interrupts corresponding to message buffers 0–7 have different meanings.

- Bit 7 of Interrupt Flags 1 (IFLAG1) becomes the Legacy FIFO Overflow flag
- Bit 6 becomes the Legacy FIFO Warning flag

- Bit 5 becomes the Frames Available in Legacy FIFO flag
- Bits 4–0 are unused.

See Interrupt Flags 1 (IFLAG1) for more information.

If both Legacy RX FIFO and DMA are enabled (MCR[RFEN] = 1 and MCR[DMA] = 1), FlexCAN does not generate any Legacy FIFO interrupt. Bit 5 of IFLAG1 still indicates Frames Available in Legacy FIFO and generates a DMA request. Bits 7, 6, and 4–0 are unused.

CAUTION

Legacy FIFO cannot be enabled when CAN FD is enabled.

When multiple message buffer interrupt sources are combined via an OR operator into a single interrupt, the interrupt is generated when any associated message buffer (or FIFO, if applicable) generates an interrupt. In this case, the CPU must read the IFLAG registers to determine which message buffer or FIFO source caused the interrupt.

These interrupt sources generate interrupts like the message buffer interrupt sources, and can be read from Error and Status 1 (ESR1):

- Bus Off
- Bus Off Done
- Error
- Error Fast
- Wake Up
- TX Warning
- RX Warning

The Bus Off, Error, TX Warning, and RX Warning interrupt masks are located in Control 1 (CTRL1); the wake-up interrupt mask is located in Module Configuration (MCR).

59.3.14 Bus interface

CPU access to FlexCAN registers is subject to the following rules:

- Unrestricted read and write access to supervisor registers results in an access error. In Table 954, supervisor registers are registers identified with an S, and registers that are identified with S/U that are in Supervisor Mode
- Read and write access to implemented reserved address space results in an access error.
- Write access to positions whose bits are all currently read-only results in an access error. If at least one of the bits is not read-only, no access error is issued. Write permission to specific positions or some of their bits can change depending on the mode of operation or transitory state. See register and field descriptions for details.
- Read and write access to unimplemented address space results in an access error.
- Read and write access to RAM-located positions during Low-Power mode results in an access error.
- The RXIMR memory region can be considered as general-purpose memory and available for access via these methods:
  - If you write 0 to MCR[IRMQ], the individual masks (RXIMR) are disabled. In this case, the RXIMR memory region is considered general-purpose memory.
  - If MCR[MAXMB] is programmed with a value smaller than the available number of message buffers, the unused memory space can be used as general-purpose RAM space. Reserved words within RAM cannot be used. For example, suppose the RAM in FlexCAN can support up to 16 message buffers, CTRL2[RFFN] = 0h, and MCR[MAXMB] = 0.
    - In this case, the maximum number of message buffers becomes one.
    - The RAM starts at 0080h, and the space 0080h–008Fh is used by the one message buffer.

- The memory space 0090h–017Fh is available.
- The space 0180h–087Fh is reserved.
- The space 0880h–0883h is used by the one individual mask and the available memory in the mask register space is 0884h–08BFh.
- In the space from 08C0h–09DFh, there are reserved words for internal use which cannot be used as general-purpose RAM.

As a rule, free memory space for general purpose depends only on MCR[MAXMB].

— If MCR[FDEN] = 1, general-purpose memory can be used only outside Freeze mode.

Table 952. Access permissions

Supervisor access	0				1		
Supervisor mode MCR[SUPV] = 1	0			1	Any		
Modes of operation	Normal	Freeze	Low-power	Any	Normal	Freeze	Low-power
MCR	Bus error	Bus error	Bus error	Bus error	Read and write	Read and write	Read and write
CTRL1	Read and write	read write	Read and write	Bus error	Read and write	Read and write	Read and write
TIMER	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
TCR	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error
RXMGMASK[1]	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
RX14MASK[1]	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
RX15MASK[1]	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
ECR	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
ESR1	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
IMASK2	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
IMASK1	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
IFLAG2	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write

Table continues on the next page...

Table 952. Access permissions (continued)

Supervisor access	0				1		
Supervisor mode MCR[SUPV] = 1	0			1	Any		
Modes of operation	Normal	Freeze	Low-power	Any	Normal	Freeze	Low-power
IFLAG1	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
CTRL2	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ESR2	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
CRCR	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error	Bus error for write operation	Bus error for write operation	Bus error for write operation
RXFGMASK[1]	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
RXFIR[1]	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error	Bus error for write operation	Bus error for write operation	Bus error for write operation
CBT	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
DBG1	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation
DBG2	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error for write operation
IMASK3	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
IFLAG3	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
MB[1 2]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
Legacy FIFO header[1]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
Legacy FIFO reserved space[1]	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error
Legacy FIFO filters[1]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write

Table continues on the next page...

Table 952. Access permissions (continued)

Supervisor access	0			1			
Supervisor mode MCR[SUPV] = 1	0			1	Any		
Modes of operation	Normal	Freeze	Low-power	Any	Normal	Freeze	Low-power
RXIMR[1]	Bus error	Read and write	Bus error	Bus error	Bus error	Read and write	Bus error
MECR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ERRIAR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ERRIDPR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ERRIPPR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
RERRAR [3]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
RERRDR[3]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
RERRSYNR[3]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ERRSR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
EPRS	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
ENCBT	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
EDCBT	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
ETDC	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
FDCTRL	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
FDCBT	Read and write[3]	Read and write	Read and write[3]	Bus error	Read and write[3]	Read and write	Read and write[3]

Table continues on the next page...

Table 952. Access permissions (continued)

Supervisor access	0			1			
Supervisor mode MCR[SUPV] = 1	0			1	Any		
Modes of operation	Normal	Freeze	Low-power	Any	Normal	Freeze	Low-power
FDCRC	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error	Bus error for write operation	Bus error for write operation	Bus error for write operation
ERFCR	Read and write[3]	Read and write	Read and write[3]	Bus error	Read and write[3]	Read and write	Read and write[3]
ERFIER	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
ERFSR	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
HR_TIME_STAMP	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
Enhanced Rx FIFO header[4]	Bus error for write operation	Bus error for write operation	Bus error for write operation	Bus error	Bus error for write operation	Bus error for write operation	Bus error for write operation
Enhanced Rx FIFO reserved space[4]	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error
ERFFEL	Bus error for write operation	Read and write	Bus error for write operation	Bus error	Bus error for write operation	Read and write	Bus error for write operation
General purpose RAM[1]	Read and write	Read and write	Read and write	Bus error	Read and write	Read and write	Read and write
Reserved space (used)[1]	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error
Reserved space (empty)[1]	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error	Bus error

1. Access in low power is only possible if RAM_clk and MODULE_CLK are enabled.
2. If MCR[RFEN] = 1, see Legacy FIFO access rules in the rows below.
3. Write operation has no effect.
4. If MCR[RFEN] = 1, do not access Enhanced RX FIFO.

59.3.15 Detection and correction of memory errors

To update the parity bits in memory properly, all FlexCAN memory must be initialized before starting its operation. CTRL2[WRMFRZ] grants write access to all memory positions that require initialization, from 080h–ADFh and from C20h–31FFh. You must also initialize these registers:

- RX Message Buffers Global Mask (RXMGMASK)
- Receive 14 Mask (RX14MASK)
- Receive 15 Mask (RX15MASK)
- Legacy RX FIFO Global Mask (RXFGMASK)

MCR[RFEN] and ERFCR[ERFEN] must not be 1 during memory initialization.

FlexCAN supports detection and correction of errors in memory read accesses. Each byte of FlexCAN memory is associated with five parity bits that ensure a Hamming distance of 4. The error correction mechanism ensures that in this 13-bit word, errors in one bit can be corrected (correctable errors). Errors in two bits can be detected but not corrected (uncorrectable errors). Errors in more than two bits may not be detected. For uncorrectable errors, the error correction logic does not change the corrupted data. When a read access is performed, the parity bits are used to calculate a syndrome, which indicates the error in each byte.

When an all-zeroes or an all-ones read occurs, FlexCAN detects an uncorrectable error. See Error Report Syndrome (RERRSYNR).

Memory errors are indicated to the host via status register (Error Status (ERRSR)) and bus transfer errors. Memory errors are reported via these report registers:

- Error Report Address (RERRAR)
- Error Report Data (RERRDR)
- Error Report Syndrome (RERRSYNR)

MECR[ECCDIS] determines whether the error detection and correction mechanism can be activated. When disabled, updates on indications and reporting registers are stopped. To ensure that memory has consistent parity bits associated with the data, the parity bits are still calculated and written with data in memory write operations.

To avoid accidentally changing the critical error correction configuration, follow this protocol to enable the update of Memory Error Control (MECR):

1. Write 1 to CTRL2[ECRWRE]. (By default, CTRL2[ECRWRE] is 0 and MECR[ECRWRDIS] is 1.)
2. Write 0 to MECR[ECRWRDIS].
3. All writes to Memory Error Control (MECR) must keep MECR[ECRWRDIS] = 0.
4. After configuration is done, lock MECR by writing 1 to MECR[ECRWRDIS] or writing 0 to CTRL2[ECRWRE].

59.3.15.1 Sources of memory access

These major sources (or requestors) can access FlexCAN memory:

- Host (CPU). The largest word accessed is 32-bit.
- FlexCAN internal processes:
  - RX matching
  - TX arbitration
  - Move-in on reception
  - Move-out on transmission

  The largest word accessed is 64-bit.

The source of access determines the way that uncorrectable errors are indicated and reported.

59.3.15.2 Error indication

These flags indicate memory errors:

- ERRSR[HANCEIF]
- ERRSR[FANCEIF]
- ERRSR[CEIF]

Uncorrectable errors detected in memory reads requested by the host are indicated separately from the errors detected in requests by FlexCAN internal processes. When correctable errors are detected, FlexCAN makes no distinction of the source of the access. There are three independent flags for these three cases, and each flag raises an interrupt unless a mask in Memory Error Control (MECR) masks it. If both uncorrectable and correctable errors are found in different bytes in the same read operation, both flags are set.

An uncorrectable error detected in host access is also indicated as a bus transfer error. A bus wait request may be asserted to extend the memory transaction to the moment the report registers are updated. This indication cannot be masked. If ERRSR[HANCEIF] is not masked, the same uncorrectable error raises a bus transfer error and an interrupt request.

Each indication flag has one overrun flag in Error Status (ERRSR). The overrun flags do not request interrupts. Overrun flags for uncorrectable errors indicate that other errors of the same nature were detected after the current error being treated. Overrun flags for correctable errors indicate that other errors of the same nature were detected before the current error being treated.

The recommended handling sequence for error indication is:

1. Get error report information from report registers.
2. Use this information to take proper measures in the application.
3. Clear the ERRSR[HANCEIF], ERRSR[FANCEIF], and ERRSR[CEIF] flags.
4. If the overrun flag is active:
    a. Alert the application that at least one error could not be managed.
    b. Clear the overrun flag.

FlexCAN internal processes can access memory in transactions larger than 32 bits. For the indication, this kind of access is considered a consecutive sequence of 32-bit accesses. If errors are found in two or more 32-bit words, the interrupt and overrun flags are set simultaneously.

59.3.15.3 Error reporting

The error report registers provide detailed information about the address read, raw data, and syndrome read with error. The flags described in Error indication indicate these report registers:

- Error Report Address (RERRAR)
- Error Report Data (RERRDR)
- Error Report Syndrome (RERRSYNR)

The address, data, and syndrome registers are updated simultaneously with the error flags, according to these rules:

1. If either of the uncorrectable error flags is set, the report registers are not updated. The previous uncorrectable error reporting is preserved.
2. Otherwise, either no error flag is set or only the correctable error flag is set. The report registers are updated according to the new error, or according to the most severe new error, if uncorrectable and correctable errors are simultaneously detected.

Reporting of errors detected in accesses larger than 32 bits follows the rules described in Error indication and in the description of Error Report Address (RERRAR).

The addresses reported in RERRAR and defined in ERRIAR are not the same as the addresses listed in the module memory map. The relation between the reported addresses and the respective ones in the module memory map is shown in the description of Error Injection Address (ERRIAR).

Addresses reported when reading memory portions organized as FIFOs refer to the address of the specific entry accessed in the FIFO, not to the FIFO base address. Such memory portions include:

- Legacy RX FIFO Structure
- Enhanced RX FIFO Structure
- Legacy RX FIFO Information (RXFIR)

To ensure coherence of error report registers, disable the report update by writing 1 to MECR[RERRDIS] before reading the report registers.

59.3.15.4 Error response

Correctable errors have no consequence to FlexCAN operation because the host or FlexCAN internal processes corrects affected data before its use.

For host-initiated reads, an uncorrectable error may affect the host, but does not affect FlexCAN operation.

Uncorrectable errors detected on memory read operations requested by FlexCAN internal processes may result in incorrect operation depending on the state of MECR[NCEFAFRZ], as follows:

- During reception (either matching or move-in process), when an uncorrectable error occurs:
  - An incorrect destination may be selected to store the incoming frame.
  - A corrupted frame may be stored in the correct destination.
  - Both of these events may occur.

  If MECR[NCEFAFRZ] = 1, FlexCAN stops operation automatically and enters Freeze mode to prevent corrupted data from being treated as valid by FlexCAN internal processes. If MECR[NCEFAFRZ] = 0, FlexCAN continues working, and a corrupted frame is received.
- During arbitration, when an uncorrectable error occurs:
  - A non-highest-priority TX message buffer may be mistakenly selected for transmission.
  - Its data may be corrupted.

  If MECR[NCEFAFRZ] = 1, FlexCAN stops operation automatically and enters Freeze mode before starting the move-out. If MECR[NCEFAFRZ] = 0, FlexCAN proceeds to move-out with a corrupted frame that will be transmitted on the CAN bus.
- During move-out, when an uncorrectable error occurs, a corrupted frame is copied from the selected TX MB that won the arbitration to the TX SMB for transmission. If MECR[NCEFAFRZ] = 1, FlexCAN stops operation automatically and enters Freeze mode before starting the transmission. When MECR[NCEFAFRZ] = 0, the corrupted frame is transferred from the TX SMB to the Protocol Engine (PE) subblock and is transmitted on the CAN bus.
- An uncorrectable error can also be detected beyond move-out, when TX data is read from TX SMB (buffer located in RAM) to be transferred to the PE subblock for transmission. In this case, a frame with corrupted ID or data is transmitted on the CAN bus.

  To prevent the external nodes from successfully receiving the frame, FlexCAN inverts all bits in the CRC field (CRC sequence plus CRC delimiter). Also, it transmits an error flag just after CRC delimiter due to self-detecting a Bit1 error and a form error due to the CRC field inversion. When MECR[NCEFAFRZ] = 1, FlexCAN stops operation automatically and enters Freeze mode just after the error frame. When MECR[NCEFAFRZ] = 0, FlexCAN may attempt to retransmit the same frame, as long as no other higher-priority TX MB is configured for transmission after.

  If the uncorrectable error persists, FlexCAN eventually reaches the Bus Off state due to consecutive error detections. ECR[TXERRCNT] is updated every time FlexCAN inverts the CRC field, causing errors as described above.

When MECR[NCEFAFRZ] = 1 and FlexCAN enters Freeze mode, only the CPU can cause FlexCAN to exit Freeze mode and resume Normal mode. MECR[NCEFAFRZ] becoming 1 is the only way to prevent corrupted frames from being transmitted on the CAN bus to the move-out internal process.

The error report registers can provide information to the application for customized management of these situations.

59.3.15.5 Error injection

These error injection registers are used to inject errors in memory reads to force errors and update the indication and reporting registers:

- Error Injection Address (ERRIAR)
- Error Injection Data Pattern (ERRIDPR)
- Error Injection Parity Pattern (ERRIPPR)

The relation between the error injection addresses and the corresponding addresses in the module memory map is shown in Error Injection Address (ERRIAR).

The injection is done by flipping the data and parity bits corresponding to the bits set to 1 in ERRIDPR and ERRIPPR. You can select injection specifically for memory accesses requested by the host or by FlexCAN internal processes.

For accesses larger than 32 bits, MECR[EXTERRIE] extends the injection pattern, replicating it in 32-bit words to fill the width of the access.

NOTE

It is possible, but very unlikely, for error injection to correct a bit with error. This correction does not raise the error flags and reports as expected.

To ensure coherence among error injection registers and avoid spurious error injections, you must clear MECR[HAERRIE] and MECR[FAERRIE] when configuring the memory injection registers.

59.4 External signal descriptions

FlexCAN has two I/O signals connected to the external chip pins. These signals are summarized in Table 953 and described in the following subsections.

Table 953. FlexCAN signal descriptions

Signal	Description	I/O
CAN RX	CAN receive pin	Input
CAN TX	CAN transmit pin	Output

59.4.1 CAN RX

This pin is the receive pin from the CAN bus transceiver. Logic level 0 represents its dominant state. Logic level 1 represents its recessive state.

59.4.2 CAN TX

This pin is the transmit pin to the CAN bus transceiver. Logic level 0 represents its dominant state. Logic level 1 represents its recessive state.

59.5 Initialization and application information

59.5.1 FlexCAN initialization sequence

For any configuration change or initialization, you must put FlexCAN into Freeze mode (see Freeze mode). The module must be initialized after every reset. FlexCAN memory must be initialized before switching to functional mode.

The following is a generic initialization sequence applicable to FlexCAN:

1. Initialize Module Configuration (MCR).
    a. Enable the individual filtering per message buffer and reception queue features by writing 1 to MCR[IRMQ].
    b. Enable the warning interrupts by writing 1 to MCR[WRNEN].
    c. If required, disable frame self-reception by writing 1 to MCR[SRXDIS].
    d. Enable the Legacy RX FIFO by writing 1 to MCR[RFEN] or enable the Enhanced RX FIFO by writing 1 to ERFCR[ERFEN].
    e. If Legacy RX FIFO or Enhanced RX FIFO is enabled and DMA is required, write 1 to MCR[DMA].
    f. Enable the abort mechanism by writing 1 to MCR[AEN].

g. Enable the local priority feature by writing 1 to MCR[LPRIOEN].

2. Initialize Control 1 (CTRL1) and CAN FD Bit Timing (FDCBT). Optionally initialize CAN Bit Timing (CBT).
   a. Determine the bit timing parameters: CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW].
   b. Optionally determine the bit timing parameters: CBT[EPROPSEG], CBT[EPSEG1], CBT[EPSEG2], and CBT[ERJW].
   c. Determine the CAN FD bit timing parameters: FDCBT[FPROPSEG], FDCBT[FPSEG1], FDCBT[FPSEG2], and FDCBT[FRJW].
   d. Determine the bit rate by programming CTRL1[PRESDIV] and optionally programming CBT[EPRESDIV].
   e. Determine the CAN FD bit rate by programming FDCBT[FPRESDIV].
   f. Determine the internal arbitration mode by programming CTRL1[LBUF].
3. If Error Code Correction (ECC) is enabled, you must initialize all FlexCAN memory. See Detection and correction of memory errors.
4. Initialize the message buffers. (See Message buffer structure for message buffer details.)
   a. The control and status word of all message buffers must be initialized.
   b. If RX FIFO is enabled, the ID filter table must be initialized.
   c. Other entries in each message buffer should be initialized as required.
5. Initialize Receive Individual Mask (RXIMR0 - RXIMR95).
6. Write 1 to required interrupt mask bits in:
   - IMASK registers (for all message buffer interrupts)
   - Module Configuration (MCR) (for wake-up interrupt)
   - Control 1 (CTRL1) and Control 2 (CTRL2) (for Bus Off and Error interrupts)
7. Write 0 to MCR[HALT].

After the last step listed above, FlexCAN attempts to synchronize to the CAN bus.

59.6 Memory map and register definition

This section describes the registers and data structures in FlexCAN. The base address of the module depends on the particular memory map of the chip.

59.6.1 FlexCAN memory mapping

The address space occupied by FlexCAN has 128 bytes for registers starting at the module base address, followed by embedded RAM starting at address offset 0080h.

Each individual register is identified by its complete name and corresponding mnemonic. The access type can be Supervisor (S) or Unrestricted (U). Most registers can be configured to have either Supervisor or Unrestricted access by programming MCR[SUPV]. These registers are identified as S/U in the Access column of Table 954.

NOTE

An invalid register access results in a bus error. Invalid accesses include reading a write-only register, writing a read-only register, and accessing an invalid address.

NOTE

To update the parity bits in memory properly, all FlexCAN memory must be initialized before reading registers which are implemented in memory. CTRL2[WRMFRZ] grants write access to all memory positions that require initialization, from 080h–ADFh and from C20h–31FFh. You must also initialize RX Message Buffers Global Mask (RXMGMASK), Receive 14 Mask (RX14MASK), Receive 15 Mask (RX15MASK) and Legacy RX FIFO Global Mask (RXFGMASK). MCR[RFEN] and ERFCR[ERFEN] must not be 1 during memory initialization.

Table 954. Register access and reset information

Register	Access type	Affected by hard reset	Affected by soft reset
Module Configuration (MCR)	S	Yes	Yes
Control 1 (CTRL1)	S/U	Yes	No
Free-Running Timer (TIMER)	S/U	Yes	Yes
RX Message buffers Global Mask (RXMGMASK)	S/U	No	No
RX Buffer 14 Mask (RX14MASK)	S/U	No	No
RX Buffer 15 Mask (RX15MASK)	S/U	No	No
Error Counter (ECR)	S/U	Yes	Yes
Error and Status 1 (ESR1)	S/U	Yes	Yes
Interrupt Masks 2 (IMASK2)	S/U	Yes	Yes
Interrupt Masks 1 (IMASK1)	S/U	Yes	Yes
Interrupt Flags 2 (IFLAG2)	S/U	Yes	Yes
Interrupt Flags 1 (IFLAG1)	S/U	Yes	Yes
Control 2 (CTRL2)	S/U	Yes	No
Error and Status 2 (ESR2)	S/U	Yes	Yes
Cyclic Redundancy Check (CRCR)	S/U	Yes	Yes
RX FIFO Global Mask (RXFGMASK)	S/U	No	No
RX FIFO Information (RXFIR)	S/U	No	No
CAN Bit Timing (CBT)	S/U	Yes	No
Interrupt Masks 3 (IMASK3)	S/U	Yes	Yes
Interrupt Flags 3 (IFLAG3)	S/U	Yes	Yes
Message buffers	S/U	No	No

Table continues on the next page...

Table 954. Register access and reset information (continued)

Register	Access type	Affected by hard reset	Affected by soft reset
RX Individual Masks	S/U	No	No
Memory Error Control (MECR)	S/U	Yes	Yes
Error Injection Address (ERRIAR)	S/U	Yes	Yes
Error Injection Data Pattern (ERRIDPR)	S/U	Yes	Yes
Error Injection Parity Pattern (ERRIPPR)	S/U	Yes	Yes
Error Report Address (RERRAR)	S/U	Yes	Yes
Error Report Data (RERRDR)	S/U	Yes	Yes
Error Report Syndrome (RERRSYNR)	S/U	Yes	Yes
Error Status (ERRSR)	S/U	Yes	Yes
Enhanced CAN Bit Timing Prescalers (EPRS)	S/U	Yes	No
Enhanced Nominal CAN Bit Timing (ENCBT)	S/U	Yes	No
Enhanced Data Phase CAN bit Timing (EDCBT)	S/U	Yes	No
Enhanced Transceiver Delay Compensation (ETDC)	S/U	Yes	No
CAN FD Control (FDCTRL)	S/U	Yes	No
CAN FD Bit Timing (FDCBT)	S/U	Yes	No
CAN FD CRC (FDCRC)	S/U	Yes	Yes
Enhanced RX FIFO Control (ERFCR)	S/U	Yes	Yes
Enhanced RX FIFO Interrupt Enable (ERFIER)	S/U	Yes	Yes
Enhanced RX FIFO Status (ERFSR)	S/U	Yes	Yes
High-Resolution Timestamp (HR_TIME_STAMP)	S/U	No	No
Enhanced RX FIFO	S/U	No	No
Enhanced RX FIFO Filter Element (ERFFEL)	S/U	No	No

FlexCAN can store CAN messages for transmission and reception using message buffers and RX FIFO structures.

59.6.2 CAN register descriptions

The table below shows the FlexCAN memory map.

The address range from offset 80h–67Fh allocates the ninety-six 128-bit message buffers. The memory maps for the message buffers are in FlexCAN message buffer memory map.

The address range from offset 2000h–204Ch allocates the Enhanced RX FIFO output, and the address range from offset 2050h–263Ch allocates the rest of Enhanced RX FIFO 19 elements. The memory map for the Enhanced RX FIFO is in Enhanced RX FIFO structure.

59.6.2.1 CAN memory map

CAN1 base address: 443A_0000h

CAN2 base address: 425B_0000h

CAN3 base address: 445B_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration (MCR)	32	RW	5980_000Fh
4h	Control 1 (CTRL1)	32	RW	0000_0000h
8h	Free-Running Timer (TIMER)	32	RW	0000_0000h
10h	RX Message Buffers Global Mask (RXMGMASK)	32	RW	See section
14h	Receive 14 Mask (RX14MASK)	32	RW	See section
18h	Receive 15 Mask (RX15MASK)	32	RW	See section
1Ch	Error Counter (ECR)	32	RW	0000_0000h
20h	Error and Status 1 (ESR1)	32	RW	0000_0000h
24h	Interrupt Masks 2 (IMASK2)	32	RW	0000_0000h
28h	Interrupt Masks 1 (IMASK1)	32	RW	0000_0000h
2Ch	Interrupt Flags 2 (IFLAG2)	32	RW	0000_0000h
30h	Interrupt Flags 1 (IFLAG1)	32	RW	0000_0000h
34h	Control 2 (CTRL2)	32	RW	0060_0000h
38h	Error and Status 2 (ESR2)	32	R	0000_0000h
44h	Cyclic Redundancy Check (CRCR)	32	R	0000_0000h
48h	Legacy RX FIFO Global Mask (RXFGMASK)	32	RW	See section
4Ch	Legacy RX FIFO Information (RXFIR)	32	R	See section
50h	CAN Bit Timing (CBT)	32	RW	0000_0000h
6Ch	Interrupt Masks 3 (IMASK3)	32	RW	0000_0000h
74h	Interrupt Flags 3 (IFLAG3)	32	RW	0000_0000h
880h - 9FCh	Receive Individual Mask (RXIMR0 - RXIMR95)	32	RW	See section
AE0h	Memory Error Control (MECR)	32	RW	800C_0080h
AE4h	Error Injection Address (ERRIAR)	32	RW	0000_0000h
AE8h	Error Injection Data Pattern (ERRIDPR)	32	RW	0000_0000h
AECh	Error Injection Parity Pattern (ERRIPPR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
AF0h	Error Report Address (RERRAR)	32	R	0000_0000h
AF4h	Error Report Data (RERRDR)	32	R	0000_0000h
AF8h	Error Report Syndrome (RERRSYNR)	32	R	0000_0000h
AFCh	Error Status (ERRSR)	32	RW	0000_0000h
BF0h	Enhanced CAN Bit Timing Prescalers (EPRS)	32	RW	0000_0000h
BF4h	Enhanced Nominal CAN Bit Timing (ENCBT)	32	RW	0000_0000h
BF8h	Enhanced Data Phase CAN Bit Timing (EDCBT)	32	RW	0000_0000h
BFCh	Enhanced Transceiver Delay Compensation (ETDC)	32	RW	0000_0000h
C00h	CAN FD Control (FDCTRL)	32	RW	8000_0100h
C04h	CAN FD Bit Timing (FDCBT)	32	RW	0000_0000h
C08h	CAN FD CRC (FDCRC)	32	R	0000_0000h
C0Ch	Enhanced RX FIFO Control (ERFCR)	32	RW	0000_0000h
C10h	Enhanced RX FIFO Interrupt Enable (ERFIER)	32	RW	0000_0000h
C14h	Enhanced RX FIFO Status (ERFSR)	32	RW	0000_0000h
C30h - DACh	High-Resolution Timestamp (HR_TIME_STAMP0 - HR_TIME_STAMP95)	32	RW	See section
3000h - 31FCh	Enhanced RX FIFO Filter Element (ERFFEL0 - ERFFEL127)	32	RW	See section

59.6.2.2 Module Configuration (MCR)

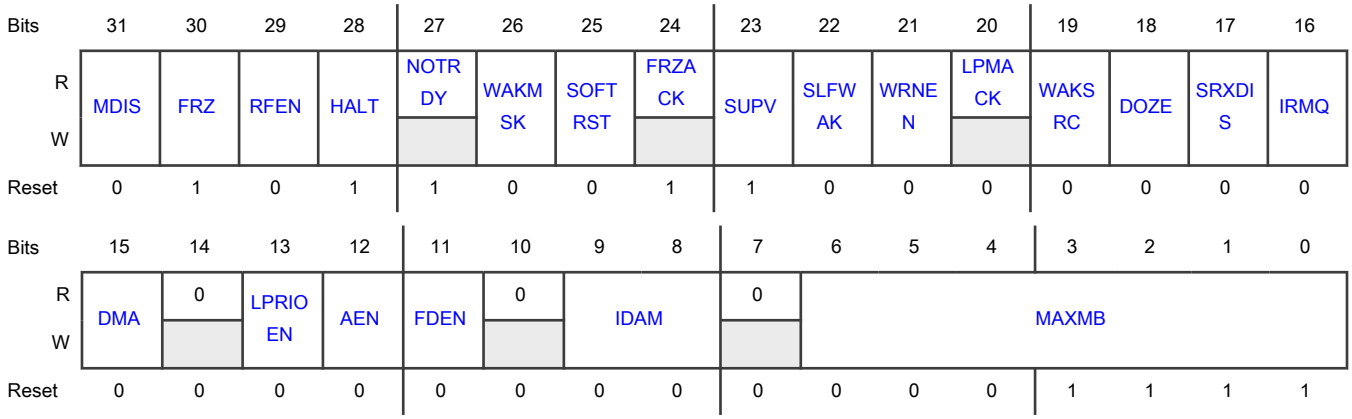
Offset

Register	Offset
MCR	0h

Function

Defines global system configurations, including the module operation modes and the maximum message buffer configuration.

Diagram



Fields

Field	Function
31 MDIS	Module Disable Disables FlexCAN. When disabled, FlexCAN disables the clocks to the CAN Protocol Engine and Controller Host Interface submodules. Soft reset does not affect this field. 0b - Enable 1b - Disable
30 FRZ	Freeze Enable Specifies FlexCAN behavior when MCR[HALT] = 1 or when Debug mode is requested at chip level. When this field becomes 1, FlexCAN can enter Freeze mode. Writing 0 to this field causes FlexCAN to exit from Freeze mode. The chip writes 1 to this field when a noncorrectable error is detected (MECR[NCEFAFRZ] becomes 1). 0b - Disable 1b - Enable
29 RFEN	Legacy RX FIFO Enable Enables the Legacy RX FIFO feature. When this field is 1, message buffers 0–5 cannot be used for normal reception and transmission. The corresponding memory region (80h–DCh) is used by the FIFO engine and additional message buffers (up to 32, depending on CTRL2[RFFN]). These message buffers are used as Legacy RX FIFO ID filter table elements. This field also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in Table 949. This field can be written in Freeze mode only; the module blocks it in other modes. **NOTE** When CAN FD operation is enabled (see MCR[FDEN]), you cannot write 1 to this field. **NOTE** This field must not be 1 if ERFCR[ERFEN] = 1. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 HALT	Halt FlexCAN Puts FlexCAN into Freeze mode. The CPU should write 0 to this field after initializing the message buffers and Control 1 (CTRL1) and Control 2 (CTRL2). FlexCAN performs no reception or transmission before this field becomes 0. Freeze mode cannot be entered when FlexCAN is in a low-power mode. The module writes 1 to this field when a noncorrectable error is detected and MECR[NCEFAFRZ] = 1. 0b - No request 1b - Enter Freeze mode, if MCR[FRZ] = 1.
27 NOTRDY	FlexCAN Not Ready Indicates whether FlexCAN is in Disable mode, Doze mode, Stop mode or Freeze mode. When FlexCAN has exited these modes, this field becomes 0. Soft reset does not affect this field. 0b - FlexCAN is in Normal mode, Listen-Only mode, or Loopback mode. 1b - FlexCAN is in Disable mode, Doze mode, Stop mode, or Freeze mode.
26 WAKMSK	Wake-up Interrupt Mask Enables the wake-up interrupt generation under the Self Wake-Up mechanism. 0b - Disabled 1b - Enabled
25 SOFTRST	Soft Reset Resets internal state machines of FlexCAN and some memory-mapped registers. The CPU can write 1 to this field directly. This field also becomes 1 when global soft reset is requested at the chip level. Because soft reset is synchronous and must follow a request-and-acknowledge procedure across clock domains, it may take some time to propagate its effect fully. When reset is pending, this field remains 1; it automatically becomes 0 when reset completes. You can poll this field to know when the soft reset has completed. Soft reset cannot be applied when clocks are shut down in a low-power mode. Transfer the module out of the low-power mode before applying soft reset. Soft reset does not affect this field. **NOTE** This field becomes 0 within 2 CAN bits after assertion of this bit. 0b - No reset 1b - Soft reset affects reset registers
24 FRZACK	Freeze Mode Acknowledge Indicates whether FlexCAN is in Freeze mode and its prescaler is stopped. The Freeze mode request cannot be granted until current transmission or reception processes have finished. Therefore you can poll this field to know when FlexCAN has entered Freeze mode. If the Freeze mode request is negated, this field becomes 0 after the FlexCAN prescaler is running again. If Freeze mode is requested when FlexCAN is in a low-power mode, this field becomes 1 only when the low-power mode is exited. See Freeze mode. Soft reset does not affect this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE FRZACK becomes 1 within 178 CAN bits following the Freeze mode request by the CPU. This field becomes 0 within 2 CAN bits after the Freeze mode request removal (see Protocol timing). 0b - Not in Freeze mode, prescaler running. 1b - In Freeze mode, prescaler stopped.
23 SUPV	Supervisor Mode Configures FlexCAN to be either in Supervisor or User mode. The registers that this field affects are marked as S/U in the Access type column of Table 954. The affected registers start with Supervisor access allowance only. In User mode, affected registers allow both Supervisor and Unrestricted accesses. In Supervisor mode, affected registers allow only Supervisor access. Unrestricted access behaves as though the access is performed on an unimplemented register location. This field can be written only in Freeze mode; the module blocks it in other modes. 0b - User mode 1b - Supervisor mode
22 SLFWAK	Self Wake-up Enables the Self Wake-up feature when FlexCAN is in a low-power mode other than Disable mode. When this feature is enabled, the FlexCAN module monitors the bus for a wake-up event (that is, a recessive-to-dominant transition). If a wake-up event is detected during Doze mode, FlexCAN requests to resume its clocks. If enabled to do so, it also generates a wake-up interrupt to the CPU. If a wake-up event is detected during Stop mode, FlexCAN generates, if enabled to do so, a wake-up interrupt to the CPU. The CPU can exit Stop mode globally and FlexCAN can request to resume the clocks. When FlexCAN is in a low-power mode other than Disable mode, this field cannot be written, as the module blocks it. 0b - Disable 1b - Enable
21 WRNEN	Warning Interrupt Enable Enables the generation of the flags ESR1[TWRNINT] and ESR1[RWRNINT]. When this field is 1, TWRNINT and RWRNINT flags are set when the respective error counter transitions from less than 96 to greater than or equal to 96. When this field is 0, the TWRNINT and RWRNINT flags are always zero, independent of the values of the error counters. No warning interrupt is generated. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Disable 1b - Enable
20	Low-Power Mode Acknowledge

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPMACK	Indicates whether FlexCAN is in a low-power mode (Disable mode, Doze mode, Stop mode). A low-power mode cannot be entered until all current transmission and reception processes have finished. The CPU can poll this field to know when FlexCAN has entered low-power mode. Soft reset does not affect this field. **NOTE** This field becomes 1 within 180 CAN bits after the low-power mode request by the CPU. This field becomes 0 within 2 CAN bits after the low-power mode request removal (see Protocol timing). 0b - Not in a low-power mode 1b - In a low-power mode
19 WAKSRC	Wake-Up Source Determines whether the integrated low-pass filter is applied to the RX input that FlexCAN uses to detect recessive-to-dominant edges on the CAN bus. This filter can protect the RX CAN input from spurious wake-up. This field can be written only in Freeze mode. The module blocks it in other modes. 0b - No filter applied 1b - Filter applied
18 DOZE	Doze Mode Enable Determines whether FlexCAN can enter low-power mode when Doze mode is requested at chip level. This field is automatically reset when FlexCAN wakes from Doze mode upon detecting activity on the CAN bus (self wake-up enabled). 0b - Disable 1b - Enable
17 SRXDIS	Self-Reception Disable Determines whether FlexCAN can receive frames transmitted by itself. If 1, frames transmitted by the module are not stored in any MB, regardless of whether the MB is programmed with an ID that matches the transmitted frame. No interrupt flag or interrupt signal is generated due to the frame reception. This field can be written only in Freeze mode; the module blocks it in other modes. 0b - Enable 1b - Disable
16 IRMQ	Individual RX Masking and Queue Enable Indicates whether RX matching process is based on individual masking and queue, or based on a masking scheme with RX Message Buffers Global Mask (RXMGMASK), Receive 14 Mask (RX14MASK), Receive 15 Mask (RX15MASK), and Legacy RX FIFO Global Mask (RXFGMASK). When this field is disabled, for backward compatibility with legacy applications, reading the Control and Status word locks the MB even if it is empty. This field can be written in Freeze mode only. The module blocks it in other modes. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
15 DMA	DMA Enable Enables DMA. The DMA feature can only be used in Legacy RX FIFO or Enhanced RX FIFO, so MCR[RFEN] or ERFCR[ERFEN] must be 1. When DMA and RFEN are 1, IFLAG1[BUF5I] generates the DMA request, and no RX FIFO interrupt is generated. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Disable 1b - Enable
14 —	Reserved
13 LPRIOEN	Local Priority Enable Enables the local priority feature. It is used to expand the ID used during the arbitration process. With this expanded ID concept, the arbitration process is done based on the full 32-bit word. However, the actual transmitted ID is 11 bits for standard frames and 29 bits for extended frames. This field can be written only in Freeze mode; the module blocks it in other modes. This bit is provided for backward compatibility with legacy applications. 0b - Disable 1b - Enable
12 AEN	Abort Enable Enables the TX abort mechanism. This mechanism guarantees a safe procedure for aborting a pending transmission, so that no frame is sent in the CAN bus without notification. This field can be written only in Freeze mode; the module blocks it in other modes. **NOTE** When this field is 1, only use the abort mechanism (see Transmission abort mechanism) to update message buffers configured for transmission. **CAUTION** Writing the abort code into RX message buffers can cause unpredictable results when this field is 1. 0b - Disabled 1b - Enabled
11 FDEN	CAN FD Operation Enable Enables the CAN with flexible data rate (CAN FD) operation. This field can be written in Freeze mode only. FlexCAN can receive and transmit messages in CAN 2.0 format. If this field is enabled, FlexCAN can also receive and transmit messages in CAN FD format. FlexCAN can transmit FD frame format according to ISO 11898-1:2015.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If the value of this field is 1, the Legacy RX FIFO Enable (MCR[RFEN]) field cannot be 1. 0b - Disable 1b - Enable
10 —	Reserved
9-8 IDAM	ID Acceptance Mode Identifies the format of the Legacy RX FIFO ID filter table elements. This field configures all elements of the table at the same time; they are all the same format. See Legacy RX FIFO structure. This field can be written only in Freeze mode; the module blocks it in other modes. 00b - Format A: One full ID (standard and extended) per ID filter table element. 01b - Format B: Two full standard IDs or two partial 14-bit (standard and extended) IDs per ID filter table element. 10b - Format C: Four partial 8-bit standard IDs per ID filter table element. 11b - Format D: All frames rejected.
7 —	Reserved
6-0 MAXMB	Number of the Last Message Buffer Defines the number of the last message buffer that takes part in the matching and arbitration processes. The reset value (0Fh) is equivalent to a 16-MB configuration. This field can be written only in Freeze mode; the module blocks it in other modes. **NOTE** You must write a value smaller than or equal to the number of available message buffers to this field, as described in FlexCAN memory partition for CAN FD. Additionally, the MAXMB value must consider the region of message buffers occupied by Legacy RX FIFO and its ID filters table space defined by CTRL2[RFFN]. MAXMB also impacts the definition of the minimum number of peripheral clocks per CAN bit, as described in Table 949.

59.6.2.3 Control 1 (CTRL1)

Offset

Register	Offset
CTRL1	4h

Function

Contains specific FlexCAN control features related to the CAN bus. These features include bit rate, programmable sampling point within an RX bit, Loopback mode, Listen-Only mode, Bus Off recovery behavior, and interrupt enabling (Bus-Off, Error, Warning). It also determines the division factor for the clock prescaler.

The CAN bit timing variables (CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW]) can also be configured in CAN Bit Timing (CBT), which extends the range of all these variables. If CBT[BTF] = 1, CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] become read-only.

If CTRL2[BTE] = 1, CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] are not used by the module. Instead, these fields are read as zero, and a write operation to them has no effect.

NOTE

When the CAN FD feature is enabled, do not use CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] for CAN bit timing. Instead use CBT[EPRESDIV], CBT[ERJW], CBT[EPSEG1], CBT[EPSEG2], and CBT[EPROPSEG].

The CAN bit variables in CTRL1 and in CBT are stored in the same internal register.

Soft reset does not affect the contents of this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PRESDIV								RJW		PSEG1		PSEG2			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BOFFMSK	ERRMSK	CLKSRC	LPB	TWRNMSK	RWRNMSK	0	0	SMP	BOFFREC	TSYN	LBUF	LOM	PROPSEG		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 PRESDIV	Prescaler Division Factor Determines the ratio between the PE clock frequency and the serial clock (Sclock) frequency. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency. The maximum value of this field is FFh, which gives a minimum Sclock frequency equal to the PE clock frequency divided by 256. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes. Sclock frequency = PE clock frequency ÷ (PRESDIV + 1).
23-22 RJW	Resync Jump Width Defines the maximum number of time quanta that one resynchronization can change a bit time. One time quantum is equal to one Sclock period. The valid programmable values are 0–3. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Resync Jump Width = RJW + 1.
21-19 PSEG1	Phase Segment 1 Defines the length of phase segment 1 in the bit time. The valid programmable values are 0–7. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes. PhaseBuffer Segment 1 = (PSEG1 + 1) × Time Quanta.
18-16 PSEG2	Phase Segment 2 Defines the length of phase segment 2 in the bit time. The valid programmable values are 1–7. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Buffer Segment 2 = (PSEG2 + 1) × Time Quanta.
15 BOFFMSK	Bus Off Interrupt Mask Provides a mask for the Bus Off interrupt ESR1[BOFFINT]. 0b - Interrupt disabled 1b - Interrupt enabled
14 ERRMSK	Error Interrupt Mask Provides a mask for the Error interrupt ESR1[ERRINT]. 0b - Interrupt disabled 1b - Interrupt enabled
13 CLKSRC	CAN Engine Clock Source Selects the clock source to the CAN Protocol Engine (PE) to be either the peripheral clock or the oscillator clock. When the oscillator clock is selected, the oscillator clock frequency must be lower than the bus clock. The selected clock is fed to the prescaler to generate the serial clock (Sclock). To guarantee reliable operation, this field can be written only in Disable mode. The module blocks it in other modes. See Protocol timing. Ensure the protocol engine clock tolerance according to the CAN Protocol standard (ISO 11898-1:2015). To identify the proper clock source, see the clock distribution chapter (module clocks table). If this field is 0, you can further select the peripheral clock source according to the Clock Root section of CCM chapter. 0b - Peripheral clock 1b - Bus clock
12 LPB	Loopback Mode Configures FlexCAN to operate in Loopback mode. In this mode, FlexCAN performs an internal loopback that can be used for self-test operation. The bit stream output of the transmitter is fed back internally to the receiver input. The RX CAN input pin is ignored and the TX CAN output goes to the recessive state (logic

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1). FlexCAN behaves as it normally does when transmitting, and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field. It generates an internal acknowledge bit to ensure proper reception of its own message. Both transmit and receive interrupts are generated. This field can be written only in Freeze mode; the module blocks it in other modes. **NOTE** In this mode, MCR[SRXDIS] cannot become 1, because it would impede the self-reception of a transmitted message. **NOTE** FDCTRL[TDCEN] and MCR[ETDCEN] must be 0 when this field is 1. 0b - Disabled 1b - Enabled
11 TWRNMSK	TX Warning Interrupt Mask Provides a mask for the TX Warning interrupt associated with the ESR1[TWRNINT] flag. When MCR[WRNEN] = 0, this field is read as 0. This field can be written only if MCR[WRNEN] = 1. 0b - Disabled 1b - Enabled
10 RWRNMSK	RX Warning Interrupt Mask Provides a mask for the RX Warning interrupt associated with the ESR1[RWRNINT] flag. When MCR[WRNEN] = 0, this field is read as 0. This field can be written only if MCR[WRNEN] = 1. 0b - Disabled 1b - Enabled
9 —	Reserved
8 —	Reserved
7 SMP	CAN Bit Sampling Determines the sampling mode of CAN bits at the RX input. This field can be written in Freeze mode only; the module blocks it in other modes. **NOTE** For proper operation, to write 1 to this field, you must guarantee a minimum value of two time quanta in CTRL1[PSEG1] (or CBT[EPSEG1]).This bit cannot become 1 when CAN FD is enabled (MCR[FDEN] = 1).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - One sample is used to determine the bit value. 1b - Three samples are used to determine the value of the received bit: the regular one (sample point) and two preceding samples. A majority rule is used.
6 BOFFREC	Bus Off Recovery Determines how FlexCAN recovers from Bus Off state. If 0, automatic recovering from Bus Off state occurs according to the CAN Specification 2.0B. If 1, automatic recovering from Bus Off is disabled. The module remains in Bus Off state until you write 1 to this field. If this field becomes 0 before 128 sequences of 11 recessive bits are detected on the CAN bus, Bus Off recovery happens as if this field had never become 1. If this field becomes 0 after 128 sequences of 11 recessive bits occurred, FlexCAN resynchronizes to the bus. It waits for 11 recessive bits before joining the bus. After this field becomes 0, it can become 1 again during Bus Off, but it will only be effective the next time the module enters Bus Off. If this field becomes 0 when the module is in Bus Off, writing 1 to this field is not effective for the current Bus Off recovery. **NOTE** See Bus Off in the CAN Protocol standard (ISO 11898-1:2015) for details. 0b - Enabled 1b - Disabled
5 TSYN	Timer Sync Enables a mechanism that resets the free-running timer each time a message is received in message buffer 0. This feature provides the means to synchronize multiple FlexCAN stations with a special "SYNC" message (that is, global network time). If MCR[RFEN] = 1 (Legacy RX FIFO enabled), the first available message buffer, according to CTRL2[RFFN], is used for timer synchronization instead of MB0. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Disable 1b - Enable
4 LBUF	Lowest Buffer Transmitted First Determines the ordering mechanism for message buffer transmission. When 1, MCR[LPRIOEN] does not affect the priority arbitration. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Buffer with highest priority is transmitted first. 1b - Lowest number buffer is transmitted first.
3 LOM	Listen-Only Mode Configures FlexCAN to operate in Listen-Only mode. In this mode, transmission is disabled, all error counters described in Error Counter (ECR) are frozen, and the module operates in CAN Error Passive mode. Only messages acknowledged by another CAN station are received. If FlexCAN detects an unacknowledged message, it flags a BIT0 error without changing the receive error counter (ECR[RXERRCNT]), as if it is trying to acknowledge the message.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	FlexCAN acknowledges Listen-Only mode when ESR1[FLTCONF] = 1, indicating the Error Passive state. There can be some delay between the Listen-Only mode request and its acknowledgment. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Listen-Only mode is deactivated. 1b - FlexCAN module operates in Listen-Only mode.
2-0 PROPSEG	Propagation Segment Defines the length of the propagation segment in the bit time. The valid programmable values are 0–7. This field can be written only in Freeze mode; the module blocks it in other modes. Propagation segment time = (PROPSEG + 1) × Time Quanta. One Time Quantum = one Sclock period.

59.6.2.4 Free-Running Timer (TIMER)

Offset

Register	Offset
TIMER	8h

Function

Represents a 16-bit free-running counter that the CPU can read and write. The timer starts from 0h after reset, counts linearly to FFFFh, and wraps around.

When CTRL2[TIMER_SRC] = 1, an external time tick continuously increments the timer. The time tick must be synchronous to the peripheral clock, with a minimum pulse width of one clock cycle.

When CTRL2[TIMER_SRC] = 0, the CAN bit clock increments the timer, which defines the baud rate on the CAN bus. During a message transmission or reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate. The timer is not incremented during Disable, Doze, Stop and Freeze modes.

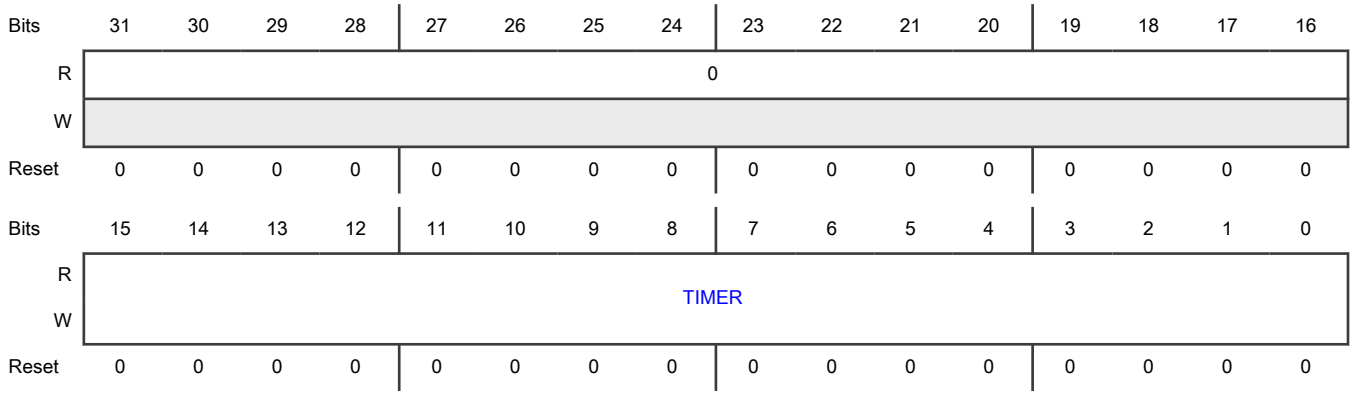
The timer value is captured when the second bit of the identifier field of any frame is on the CAN bus. This captured value is written into the timestamp entry in a message buffer after a successful reception or transmission of a message.

If CTRL1[TSYN] = 1, the timer is reset whenever a message is received in the first available message buffer, according to CTRL2[RFFN].

The CPU can write to this register anytime. However, if the write occurs simultaneously with the timer being reset by a reception in the first message buffer, the write value is discarded.

Reading this register affects the message buffer unlocking procedure (see Message buffer lock mechanism).

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TIMER	Timer Value Contains the free-running counter value.

59.6.2.5 RX Message Buffers Global Mask (RXMGMASK)

Offset

Register	Offset
RXMGMASK	10h

Function

Masks the filter bits of all RX message buffers, excluding MB14 and MB15, which have individual mask registers.

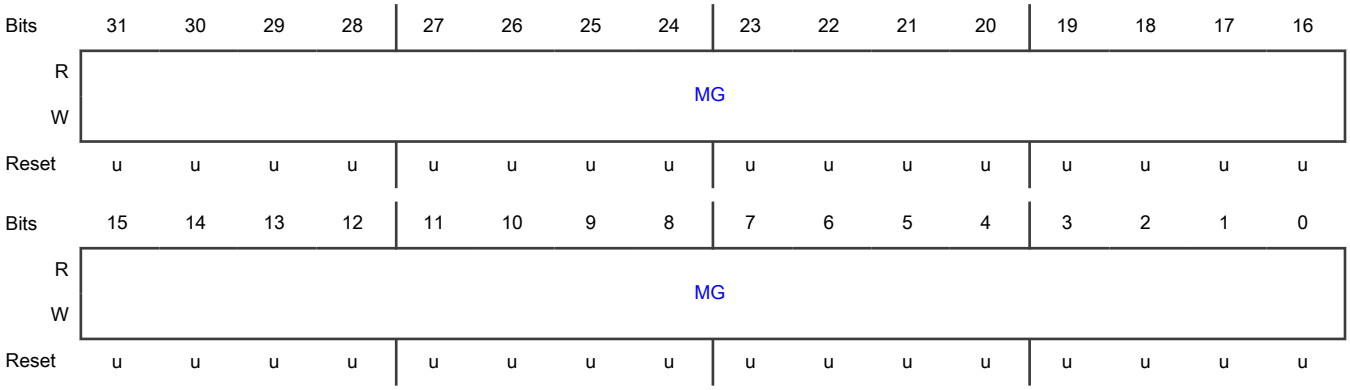
This register is located in RAM.

RXMGMASK is provided for legacy application support.

- When MCR[IRMQ] is 0, RXMGMASK is always in effect. The bits in RXMGMASK[MG] mask the MB filter bits.
- When MCR[IRMQ] is 1, RXMGMASK has no effect. The bits in RXMGMASK[MG] do not mask the MB filter bits.

This register can only be written in Freeze mode; the module blocks it in other modes.

Diagram



Fields

Field	Function
31-0 MG	Global Mask for RX Message Buffers Masks the message buffer filter bits. The alignment with the ID word of the message buffer is imperfect. The two most significant MG bits affect the fields RTR and IDE, which are located in the Control and Status word of the MB. The following table shows which MG bits mask each MB filter field.

SMB[RTR] [1]	CTRL2[RRS]	CTRL2[EACEN]	Message buffer filter fields			
			MB[RTR]	MB[IDE]	MB[ID]	Reserved
0	-	0	Note [2]	Note [3]	MG[28:0]	MG[31:29]
0	-	1	MG[31]	MG[30]	MG[28:0]	MG[29]
1	0	-	-	-	-	MG[31:0]
1	1	0	-	-	MG[28:0]	MG[31:29]
1	1	1	MG[31]	MG[30]	MG[28:0]	MG[29]

1. RTR bit of the incoming frame. It is saved into an auxiliary MB called RX serial message buffer (RX SMB).
2. If CTRL2[EACEN] is 0, the RTR bit of MB is never compared with the RTR bit of the incoming frame.
3. If CTRL2[EACEN] is 0, the IDE bit of MB is always compared with the IDE bit of the incoming frame.

0b - The corresponding bit in the filter is "don't care."

1b - The corresponding bit in the filter is checked.

59.6.2.6 Receive 14 Mask (RX14MASK)

Offset

Register	Offset
RX14MASK	14h

Function

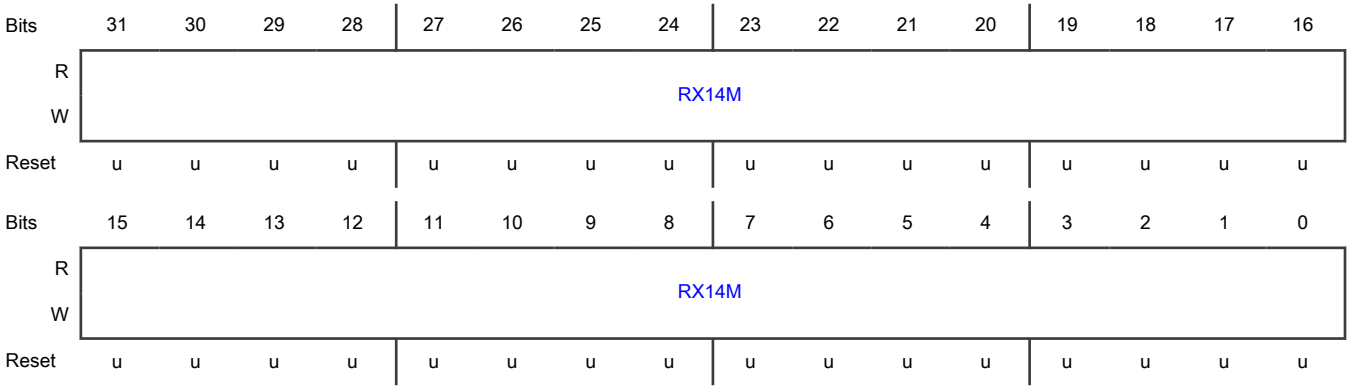
Masks the filter fields of MB14.

This register is located in RAM.

RX14MASK is provided for legacy application support. When MCR[IRMQ] = 1, RX14MASK has no effect.

This register can only be programmed when the module is in Freeze mode; the module blocks it in other modes.

Diagram



Fields

Field	Function
31-0 RX14M	RX Buffer 14 Mask Bits Masks the corresponding MB14 filter field in the same way that RX Message Buffers Global Mask (RXMGMASK) masks the filters of the other message buffers. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

59.6.2.7 Receive 15 Mask (RX15MASK)

Offset

Register	Offset
RX15MASK	18h

Function

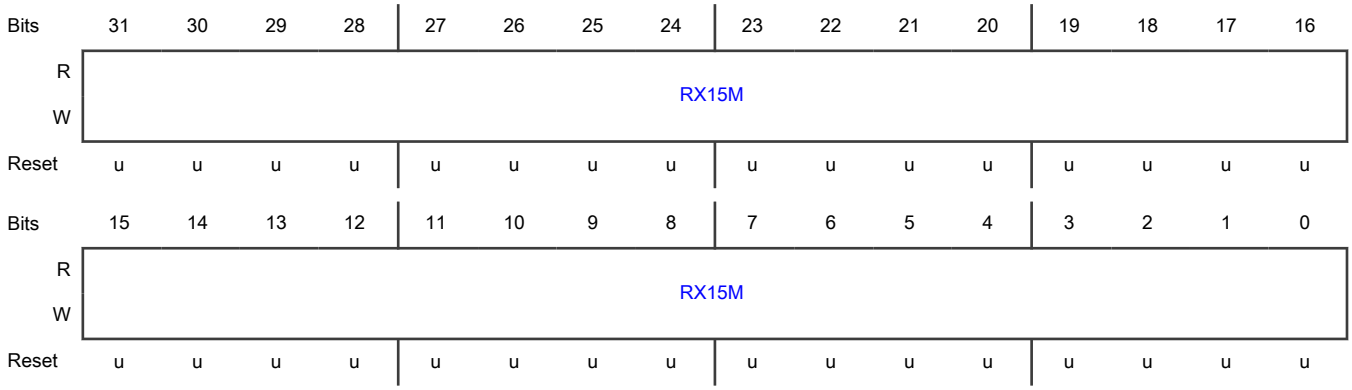
Masks the filter fields of MB15.

This register is located in RAM.

RX15MASK is provided for legacy application support. When MCR[IRMQ] = 1, RX15MASK has no effect.

This register can be programmed only when the module is in Freeze mode; the module blocks it in other modes.

Diagram



Fields

Field	Function
31-0 RX15M	RX Buffer 15 Mask Bits Masks the corresponding MB15 filter field in the same way that RX Message Buffers Global Mask (RXMGMASK) masks the filters of other message buffers. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

59.6.2.8 Error Counter (ECR)

Offset

Register	Offset
ECR	1Ch

Function

Contains error counters for received and transmitted messages.

TXERRCNT and RXERRCNT consider all errors in both CAN FD and non-FD message formats. TXERRCNT_FAST and RXERRCNT_FAST count only the errors that occur in the data phase of CAN FD frames that have BRS = 1.

The Fault Confinement state (ESR1[FLTCONF]) is updated based on TXERRCNT and RXERRCNT counters only. The rules for increasing and decreasing these counters are described in the CAN protocol and are entirely implemented in FlexCAN.

The basic rules for FlexCAN bus state transitions are:

- If the value of TXERRCNT or RXERRCNT becomes greater than or equal to 128, ESR1[FLTCONF] is updated to reflect Error Passive state.
- If the state of FlexCAN is Error Passive, and TXERRCNT or RXERRCNT decrements to a value less than 128 when the other already satisfies this condition, ESR1[FLTCONF] is updated to reflect Error Active state.
- If the value of TXERRCNT becomes greater than 255, ESR1[FLTCONF] is updated to reflect Bus Off state, and an interrupt may be issued. The value of TXERRCNT is then reset to zero.
- If FlexCAN is in Bus Off, TXERRCNT is cascaded with another internal counter to count the occurrences of 11 consecutive recessive bits on the bus. TXERRCNT is reset to zero. It counts in a manner where the internal counter

counts 11 such bits and then wraps around when incrementing the TXERRCNT. When TXERRCNT reaches the value of 128, ESR1[FLTCONF] is updated to Error Active, and both error counters are reset to zero. Upon any instance of a dominant bit following a stream of less than 11 consecutive recessive bits, the internal counter resets itself to zero without affecting the TXERRCNT value. The TXERRCNT_FAST counter is frozen during Bus Off.

- If only one node is operating during system startup, its TXERRCNT increases upon each attempted message transmission, as a result of acknowledge errors (indicated by ESR1[ACKERR]). After the transition to Error Passive state, TXERRCNT no longer increments upon acknowledge errors. The chip never goes into the Bus Off state.
- If RXERRCNT increases to a value greater than 127, it is not incremented further, even if more errors are detected when being a receiver. At the next successful message reception, the counter is set to a value between 119 and 127 to return to the Error Active state.
- TXERRCNT_FAST and RXERRCNT_FAST error counter values increment and decrement based on errors detected only in the data phase of CAN FD frames that have BRS = 1. These counters follow the same increment and decrement rules as TXERRCNT and RXERRCNT. These counters do not wrap around and get stuck at their maximum value (255). They stop counting and keep their values frozen when FlexCAN is in the Bus Off state. They are reset when FlexCAN leaves the Bus Off state and resume counting after FlexCAN returns to the Error Active state.

NOTE

See Fault confinement in the CAN Protocol standard (ISO 11898-1:2015) for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXERRCNT_FAST								TXERRCNT_FAST							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXERRCNT								TXERRCNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 RXERRCNT_FAST	Receive Error Counter for Fast Bits Counts errors detected in the data phase of received CAN FD messages that have BRS = 1. This field is read-only except in Freeze mode, when the CPU can write an 8-bit zero value only.
23-16 TXERRCNT_FAST	Transmit Error Counter for Fast Bits Counts errors detected in the data phase of transmitted CAN FD messages that have BRS = 1. This field is read-only except in Freeze mode, when the CPU can write an 8-bit zero value only.
15-8 RXERRCNT	Receive Error Counter Counts all errors detected in received messages. This field is read-only except in Freeze mode, when the CPU can write to it.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 TXERRCNT	Transmit Error Counter Counts all errors detected in transmitted messages. This field is read-only except in Freeze mode, when the CPU can write to it.

59.6.2.9 Error and Status 1 (ESR1)

Offset

Register	Offset
ESR1	20h

Function

Reports various error conditions detected in the reception and transmission of a CAN frame. This register provides status information about the chip, and is the source of some interrupts to the CPU. The reported error conditions are:

NOTE

Reading host can clear these fields.

- Errors detected in CAN frames of any format:
  - BIT1ERR
  - BIT0ERR
  - ACKERR
  - CRCERR
  - FRMERR
  - STFERR
- Errors detected in the data phase of CAN FD frames with the BRS bit set only:
  - BIT1ERR_FAST
  - BIT0ERR_FAST
  - CRCERR_FAST
  - FRMERR_FAST
  - STFERR_FAST

One or more error flags may report an error detected in a single CAN frame. To account for more error events occurring in subsequent frames when the CPU does not attempt to read this register, error reporting is cumulative.

Status flags:

- TXWRN
- RXWRN
- IDLE
- TX

- FLTCONF
- RX
- SYNCH

Interrupt flags:

- BOFFINT
- BOFFDONEINT
- ERRINT
- ERRINT_FAST
- WAKINT
- TWRNINT
- RWRNINT

The CPU should follow this procedure when servicing interrupt requests generated by these flags:

1. Read this register to capture all error condition and status flags. This action clears the respective flags that were set since the last read access.
2. Write 1 to clear the interrupt flag that triggered the interrupt request.
3. Write 1 to clear the ERROVR flag, if it is set.

Starting from all error flags cleared, a first error event sets either ERRINT or ERRINT_FAST (provided the corresponding mask bit is 1). If other error events in subsequent frames occur before the CPU serves the interrupt request, the ERROVR flag is set to indicate that errors from different frames have accumulated.

Table 955. CAN bus status

SYNCH	IDLE	TX	RX	FlexCAN state
0	0	0	0	Not synchronized to CAN bus
1	1	X	X	Idle
1	0	1	0	Transmitting
1	0	0	1	Receiving

NOTE

See Fault confinement in the CAN Protocol standard (ISO 11898-1:2015) for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BIT1ERR...	BIT0ERR...	0	CRCERR_...	FRMERR_...	STFERR_...	0		0	0	ERROVR	ERRINT_...	BOFFDON...	SYNCH	TWRNINT	RWRNINT
W											W1C	W1C	W1C		W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BIT1ERR	BIT0ERR	ACKERR	CRCERR	FRMERR	STFERR	TXWRN	RXWRN	IDLE	TX	FLTCONF		RX	BOFFINT	ERRINT	WAKINT
W														W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BIT1ERR_FAST	Fast Bit1 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in the data phase of CAN FD frames that have BRS = 1. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit transmitted as recessive is received as dominant.
30 BIT0ERR_FAST	Fast Bit0 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in the data phase of CAN FD frames that have BRS = 1. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit transmitted as dominant is received as recessive.
29 —	Reserved
28 CRCERR_FAST	Fast Cyclic Redundancy Check Error Flag Indicates that the receiver node has detected a CRC error in the CRC field of CAN FD frames that have BRS = 1. This error means that the calculated CRC is different from the received CRC. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A CRC error occurred since last read of this register.
27	Fast Form Error Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
FRMERR_FAST	Indicates whether the receiver node has detected a form error in the data phase of CAN FD frames that have BRS = 1. This error means that a fixed-form bit field contains at least one illegal bit. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A form error occurred since last read of this register.
26 STFERR_FAST	Fast Stuffing Error Flag Indicates that a stuffing error has been detected in the data phase of CAN FD frames that have BRS = 1. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A stuffing error occurred since last read of this register.
25-24 —	Reserved
23 —	Reserved
22 —	Reserved
21 ERROVR	Error Overrun Flag Indicates that an error condition occurred when any error flag is already set. 0b - No overrun 1b - Overrun
20 ERRINT_FAST	Fast Error Interrupt Flag Indicates that at least one error flag detected in the data phase of CAN FD frames that have BRS = 1 (BIT1ERR_FAST, BIT0ERR_FAST, CRCERR_FAST, FRMERR_FAST, or STFERR_FAST) is set. If CTRL2[ERRMSK_FAST] = 1, an interrupt is generated to the CPU. 0b - No such occurrence. 1b - Error flag set in the data phase of CAN FD frames that have BRS = 1.
19 BOFFDONEINT	Bus Off Done Interrupt Flag Indicates whether ECR[TXERRCNT] has finished counting 128 occurrences of 11 consecutive recessive bits on the CAN bus and is ready to leave Bus Off. If CTRL2[BOFFDONEMSK] = 1, an interrupt is generated to the CPU. 0b - No such occurrence 1b - FlexCAN module has completed Bus Off process.
18	CAN Synchronization Status Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYNCH	Indicates whether FlexCAN is synchronized to the CAN bus and able to participate in the communication process. FlexCAN sets and clears this flag. See the table in Error and Status 1 (ESR1). 0b - Not synchronized 1b - Synchronized
17 TWRNINT	TX Warning Interrupt Flag Indicates whether TX error counter changed from less than 96 to greater than or equal to 96. If MCR[WRNEN] = 1, this flag is set when the TXWRN flag transitions from 0 to 1, meaning that the TX error counters reached 96. If CTRL1[TWRNMSK] = 1, an interrupt is sent to the CPU. When MCR[WRNEN] = 0, this flag is masked. The CPU must clear this flag before writing 0 to MCR[WRNEN]. Otherwise, this flag is set when MCR[WRNEN] = 1 again. Writing 0 has no effect. This flag is not generated when in the Bus Off state. This flag is not updated during Freeze mode. 0b - No such occurrence 1b - TX error counter changed from less than 96 to greater than or equal to 96.
16 RWRNINT	RX Warning Interrupt Flag Indicates whether the RX error counter changed from less than 96 to greater than or equal to 96. If MCR[WRNEN] = 1, this flag is set when the RXWRN flag transitions from 0 to 1, meaning that the RX error counters reached 96. If CTRL1[RWRNMSK] = 1, an interrupt is sent to the CPU. When MCR[WRNEN] = 0, this flag is masked. The CPU must clear this flag before writing 0 to MCR[WRNEN]. Otherwise, this flag is set when MCR[WRNEN] = 1 again. Writing 0 has no effect. This flag is not updated during Freeze mode. 0b - No such occurrence 1b - RX error counter changed from less than 96 to greater than or equal to 96.
15 BIT1ERR	Bit1 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in a non-CAN FD message or in the arbitration or data phase of a CAN FD message. **NOTE** A transmitter does not set this flag for an arbitration field or ACK slot. It is not set for a node sending a error passive flag that detects dominant bits. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as recessive is received as dominant.
14 BIT0ERR	Bit0 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in a non-CAN FD message or in the arbitration or data phase of a CAN FD message. After a read operation, the field's value clears to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No such occurrence. 1b - At least one bit sent as dominant is received as recessive.
13 ACKERR	Acknowledge Error Flag Indicates whether the transmitter node has detected an acknowledge error. This error means that a dominant bit has not been detected during the ACK SLOT. After a read operation, the field's value clears to 0. 0b - No error 1b - Error occurred since last read of this register.
12 CRCERR	Cyclic Redundancy Check Error Flag Indicates whether the receiver node has detected a cyclic redundancy check (CRC) error either in a non-FD message or in the arbitration or data phase of a frame in CAN FD format. This error means that the calculated CRC is different from the received. After a read operation, the field's value clears to 0. 0b - No error 1b - Error occurred since last read of this register.
11 FRMERR	Form Error Flag Indicates whether a form error has been detected in a non-FD message or in the arbitration or data phase of an FD message by the receiver node. This error means that a fixed-form field contains at least one illegal bit. After a read operation, the field's value clears to 0. 0b - No error 1b - Error occurred since last read of this register.
10 STFERR	Stuffing Error Flag Indicates that a stuffing error has been detected in a non-FD message or in the arbitration or data phase of an FD message by the receiver node. After a read operation, the field's value clears to 0. 0b - No error 1b - Error occurred since last read of this register.
9 TXWRN	TX Error Warning Flag Indicates when repetitive errors occur during message transmission. Only the value of ECR[TXERRCNT] affects this flag. This flag is not updated during Freeze mode. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - TXERRCNT is 96 or greater.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 RXWRN	RX Error Warning Flag Indicates when repetitive errors occur during message reception. Only the value of ECR[RXERRCNT] affects this flag. This flag is not updated during Freeze mode. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - RXERRCNT is greater than or equal to 96.
7 IDLE	Idle Indicates whether CAN bus is in IDLE state. See Table 955. 0b - Not IDLE 1b - IDLE
6 TX	FlexCAN In Transmission Indicates whether FlexCAN is transmitting a message. See Table 955. 0b - Not transmitting 1b - Transmitting
5-4 FLTCONF	Fault Confinement State Indicates the confinement state of FlexCAN. If CTRL1[LOM] = 1, after a delay that depends on the CAN bit timing, this field indicates Error Passive. The same delay affects the way that this field reflects an update to ECR register by the CPU. It may be necessary to wait up to one CAN bit time for coherence to be restored. Soft reset affects this field, but if CTRL1[LOM] = 1, its reset value lasts for only one CAN bit. After that time, this field reports Error Passive. 00b - Error Active 01b - Error Passive 1xb - Bus Off
3 RX	FlexCAN in Reception Flag Indicates whether FlexCAN is receiving a message. See the table in Error and Status 1 (ESR1). 0b - Not receiving 1b - Receiving
2 BOFFINT	Bus Off Interrupt Flag Indicates whether FlexCAN has entered Bus Off state. If CTRL1[BOFFMSK]) = 1, an interrupt is generated to the CPU. Writing 0 to this field has no effect. 0b - No such occurrence. 1b - FlexCAN module entered Bus Off state.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 ERRINT	Error Interrupt Flag Indicates that at least one of the error flags (ESR1[BIT1ERR], ESR1[BIT0ERR], ESR1[ACKERR], ESR1[CRCERR], ESR1[FRMERR], or ESR1[STFERR]) is set. If the corresponding mask CTRL1[ERRMSK] = 1, an interrupt is generated to the CPU. Writing 0 to this field has no effect. 0b - No such occurrence. 1b - Indicates setting of any error flag in the Error and Status register.
0 WAKINT	Wake-up Interrupt Flag Generates an interrupt to the CPU when a recessive-to-dominant transition is detected on the CAN bus and MCR[WAKMSK] = 1. This field applies when FlexCAN is in low-power mode during Self Wake-up: • Doze mode • Stop mode When MCR[SLFWAK] = 0, this flag is masked. The CPU must clear this flag before disabling the field. Otherwise, it is set when MCR[SLFWAK] becomes 1 again. Writing 0 has no effect. 0b - No such occurrence. 1b - Indicates that a recessive-to-dominant transition was received on the CAN bus.

59.6.2.10 Interrupt Masks 2 (IMASK2)

Offset

Register	Offset
IMASK2	24h

Function

Masks interrupt flags. This register allows any of the 32 message buffer interrupts to be enabled or disabled for MB63–MB32. It contains one interrupt mask bit per buffer. This configuration allows the CPU to determine which buffer generates an interrupt after a successful transmission or reception when the corresponding Interrupt Flags 2 (IFLAG2) flag is set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BUF63TO32M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BUF63TO32M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF63TO32M	Buffer MBi Mask Masks the corresponding FlexCAN message buffer interrupt for MB63–MB32. **NOTE** If the corresponding Interrupt Flags 2 (IFLAG2) flag is set, writing 1 or 0 to a field in IMASK2 can set or clear an interrupt request. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

59.6.2.11 Interrupt Masks 1 (IMASK1)

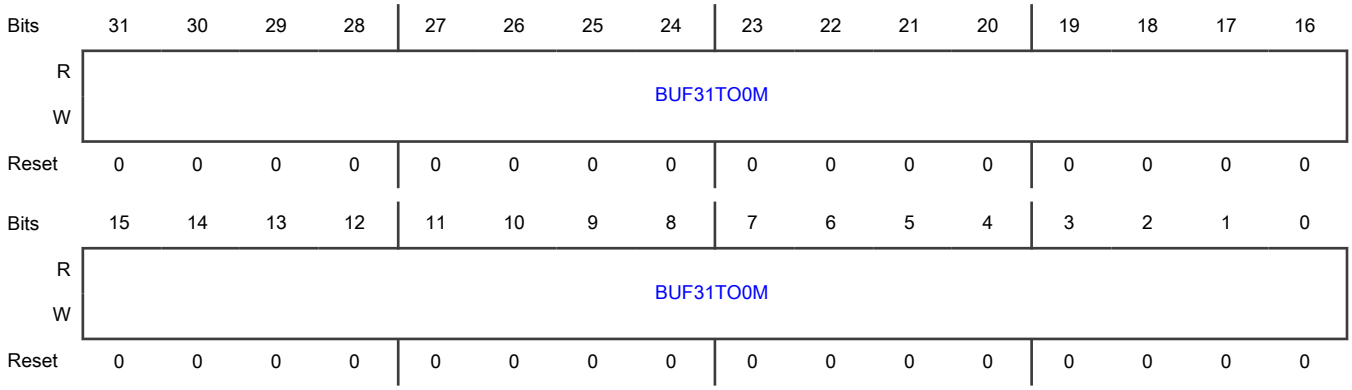
Offset

Register	Offset
IMASK1	28h

Function

Masks interrupt flags. This register allows any of the 32 message buffer interrupts to be enabled or disabled for MB31–MB0. It contains one interrupt mask bit per buffer. This configuration allows the CPU to determine which buffer generates an interrupt after a successful transmission or reception when the corresponding Interrupt Flags 1 (IFLAG1) flag is set.

Diagram



Fields

Field	Function
31-0 BUF31TO0M	Buffer MBi Mask Enables or disables the corresponding FlexCAN message buffer interrupt for MB31–MB0. **NOTE** If the corresponding Interrupt Flags 1 (IFLAG1) flag is set, writing 1 or 0 to a field in IMASK1 can set or clear an interrupt request. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

59.6.2.12 Interrupt Flags 2 (IFLAG2)

Offset

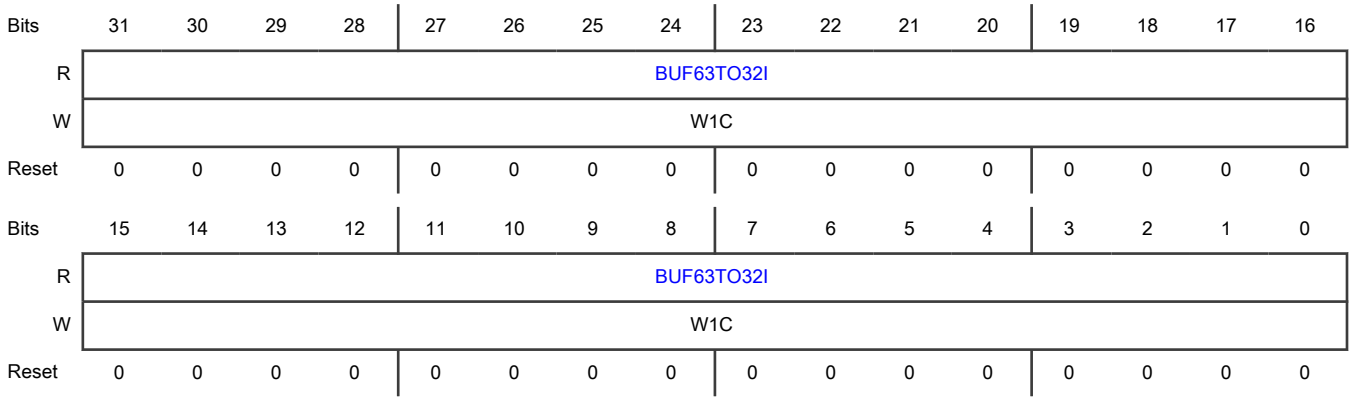
Register	Offset
IFLAG2	2Ch

Function

Contains the flags for the 32 message buffer interrupts for MB63–MB32. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the respective flag in this register. If the corresponding Interrupt Masks 2 (IMASK2) bit is set, an interrupt is generated.

Before updating MCR[MAXMB], the CPU must service the IFLAG2 flags whose MB value is greater than the MAXMB to be updated. Otherwise, those flags remain set and are inconsistent with the number of message buffers available.

Diagram



Fields

Field	Function
31-0 BUF63TO32I	Buffer MBi Interrupt Flags the corresponding FlexCAN message buffer interrupt for MB63–MB32. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.

59.6.2.13 Interrupt Flags 1 (IFLAG1)

Offset

Register	Offset
IFLAG1	30h

Function

Contains the flags for the 32 message buffer interrupts for MB31–MB0. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG1 bit. If the corresponding Interrupt Masks 1 (IMASK1) bit is set, an interrupt is generated. There is an exception when DMA for Legacy RX FIFO is enabled, as described below.

The BUF7I–BUF5I flags also represent Legacy FIFO interrupts when the Legacy RX FIFO is enabled. When MCR[RFEN] is 1 and MCR[DMA] is 0, the function of the eight least significant interrupt flags changes:

- BUF7I, BUF6I, and BUF5I indicate operating conditions of the Legacy FIFO.
- BUF4I–BUF1I fields are reserved.
- BUF0I empties the Legacy FIFO.

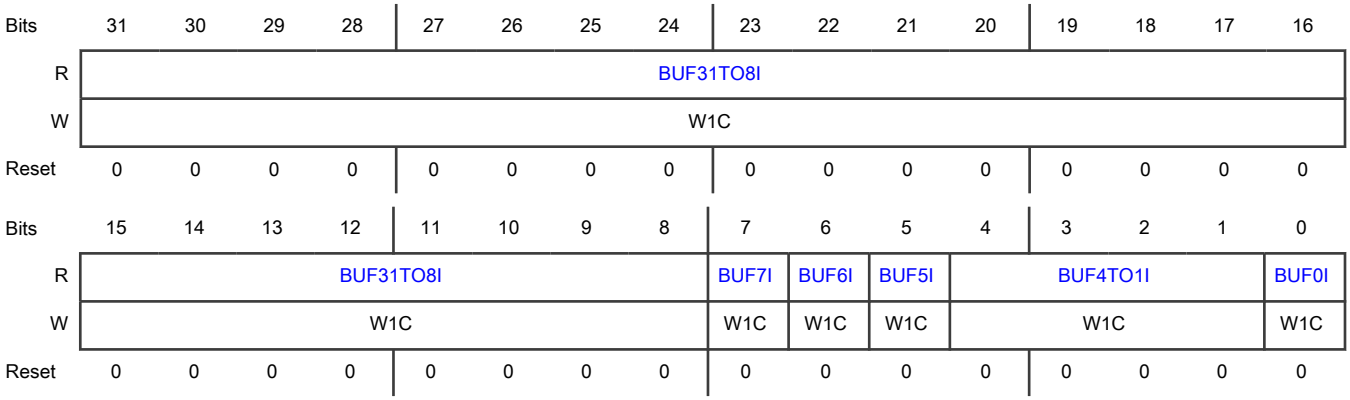
Before writing 1 to MCR[RFEN], the CPU must service the IFLAG flags set in the Legacy RX FIFO region; see Legacy RX FIFO. Otherwise, these IFLAG flags mistakenly show the related message buffers now belonging to Legacy FIFO as having contents to be serviced. When MCR[RFEN] is 0, the Legacy FIFO flags must be cleared. The same care must be taken when a CTRL2[RFFN] value is selected, extending Legacy RX FIFO filters beyond MB7. For example, when RFFN is 8h, Legacy RX FIFO filters occupy the MB23–MB0 range, and related IFLAG flags must be cleared.

When MCR[RFEN] and MCR[DMA] are 1 (DMA feature for Legacy RX FIFO enabled), the function of the eight least significant interrupt flags (BUF7I–BUF0I) changes to support DMA operation. BUF7I, BUF6I, and BUF4I–BUF1I are not used.

BUF5I indicates the operating condition of the Legacy FIFO, and BUF0I empties the Legacy FIFO. Moreover, BUF5I does not generate a CPU interrupt, but it does generate a DMA request. IMASK1 bits in the Legacy RX FIFO region are not considered when bit MCR[DMA] = 1. In addition, the CPU must not clear the BUF5I flag when DMA is enabled. Before writing 1 to MCR[DMA], the CPU must service the IFLAG flags set in the Legacy RX FIFO region. When MCR[DMA] is 0, the Legacy FIFO must be empty. Legacy FIFO must be disabled when MCR[FDEN] = 1.

Before updating MCR[MAXMB], the CPU must service the IFLAG1 flags whose MB value is greater than the MCR[MAXMB] to be updated. Otherwise, those flags remain set and are inconsistent with the number of message buffers available.

Diagram



Fields

Field	Function
31-8 BUF31TO8I	Buffer MBi Interrupt Flags the corresponding FlexCAN message buffer interrupt for MB31–MB8. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.
7 BUF7I	Buffer MB7 Interrupt or Legacy RX FIFO Overflow Flags the interrupt for MB7 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). **NOTE** When a CPU write changes the value of MCR[RFEN], FlexCAN clears this flag. When MCR[RFEN] = 1, this flag represents a Legacy RX FIFO overflow. In this case, the flag indicates that a message was lost because the Legacy RX FIFO is full. When the Legacy RX FIFO is full and a message buffer captures the message, this flag is not set. 0b - No occurrence of MB7 completing transmission or reception, or no FIFO overflow. 1b - MB7 completed transmission or reception, or FIFO overflow.
6 BUF6I	Buffer MB6 Interrupt or Legacy RX FIFO Warning Flags the interrupt for MB6 when MCR[RFEN] = 0 (Legacy RX FIFO disabled).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE When a CPU write changes the value of MCR[RFEN], FlexCAN clears this flag. When MCR[RFEN] = 1, this flag represents a Legacy RX FIFO warning. In this case, the flag indicates when the number of unread messages within the Legacy RX FIFO is increased to five from four due to the reception of a new message. In other words, the Legacy RX FIFO is almost full. If this flag is cleared when there are more than four unread messages, it does not set again until the number of unread messages in the Legacy RX FIFO decreases to four or fewer. 0b - No occurrence of MB6 completing transmission or reception, or FIFO not almost full. 1b - MB6 completed transmission or reception, or FIFO almost full.
5 BUF5I	Buffer MB5 Interrupt or Frames available in Legacy RX FIFO Flags the interrupt for MB5 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). **NOTE** When a CPU write changes the value of MCR[RFEN], FlexCAN clears this flag. When MCR[RFEN] = 1, the BUF5I flag represents frames available in Legacy RX FIFO. In this case, the flag indicates that at least one frame is available to be read from the Legacy RX FIFO. When MCR[DMA] = 1, this flag generates a DMA request. The CPU must not clear this field by writing 1 to BUF5I. 0b - No occurrence of completed transmission or reception, or no frames available 1b - MB5 completed transmission or reception, or frames available
4-1 BUF4TO1I	Buffer MBi Interrupt or Reserved Flags the interrupts for MB4–MB1 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). **NOTE** When a CPU write changes the value of MCR[RFEN], FlexCAN clears these flags. When MCR[RFEN] = 1, the BUF4TO1I flags are reserved. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.
0 BUF0I	Buffer MB0 Interrupt or Clear Legacy FIFO bit Flags the interrupt for MB0 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). If MCR[RFEN] = 1, this field is used to trigger the clear Legacy FIFO operation. This operation empties the Legacy FIFO contents. Before performing this operation, the CPU must service all Legacy FIFO-related IFLAG flags. When MCR[DMA] = 1, this operation also clears the BUF5I flag, aborting the DMA request. The clear Legacy FIFO operation occurs when the CPU writes 1 to BUF0I. This operation is only allowed in Freeze mode; the module blocks it in other conditions.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MB0 has no occurrence of successfully completed transmission or reception. 1b - MB0 has successfully completed transmission or reception.

59.6.2.14 Control 2 (CTRL2)

Offset

Register	Offset
CTRL2	34h

Function

Complements Control 1 (CTRL1), providing control bits for memory write-access in Freeze mode. This register extends Legacy FIFO filter quantity, and adjusts the operation of internal FlexCAN processes such as matching and arbitration.

Soft reset does not affect the contents of this register.

Table 956 shows how the value of CTRL2[RFFN] determines the Legacy RX FIFO filter structure.

Table 956. Possible Legacy RX FIFO filter structures

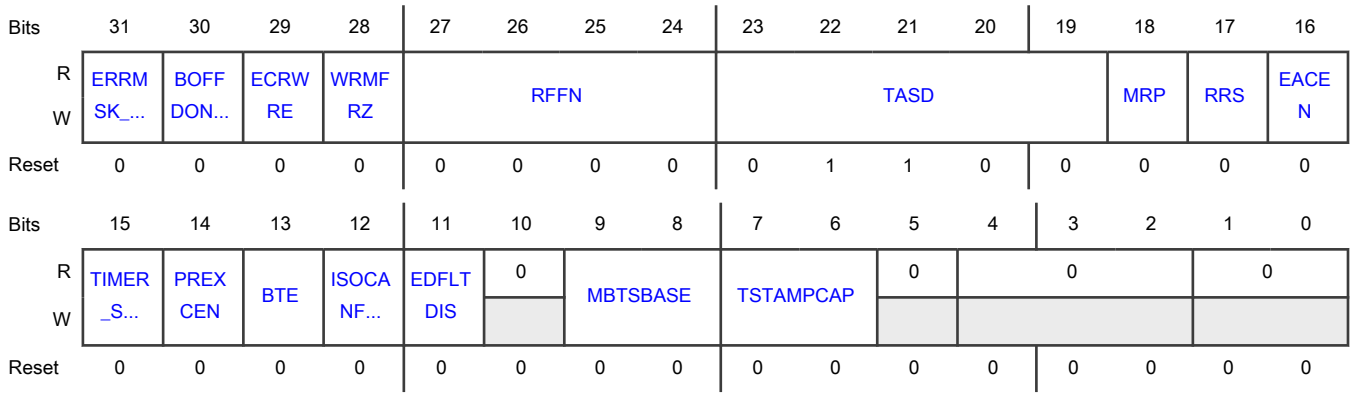
RFFN[3:0]	Number of Legacy RX FIFO filter elements	Message buffers occupied by Legacy RX FIFO and ID filter table	Remaining available message buffers	Legacy RX FIFO ID filter table elements affected by RX individual masks	Legacy RX FIFO ID filter table elements affected by Legacy RX FIFO global mask
0h	8	MB 0–7	MB 8–95	Elements 0–7	None
1h	16	MB 0–9	MB 10–95	Elements 0–9	Elements 10–15
2h	24	MB 0–11	MB 12–95	Elements 0–11	Elements 12–23
3h	32	MB 0–13	MB 14–95	Elements 0–13	Elements 14–31
4h	40	MB 0–15	MB 16–95	Elements 0–15	Elements 16–39
5h	48	MB 0–17	MB 18–95	Elements 0–17	Elements 18–47
6h	56	MB 0–19	MB 20–95	Elements 0–19	Elements 20–55
7h	64	MB 0–21	MB 22–95	Elements 0–21	Elements 22–63
8h	72	MB 0–23	MB 24–95	Elements 0–23	Elements 24–71
9h	80	MB 0–25	MB 26–95	Elements 0–25	Elements 26–79
Ah	88	MB 0–27	MB 28–95	Elements 0–27	Elements 28–87
Bh	96	MB 0–29	MB 30–95	Elements 0–29	Elements 30–95
Ch	104	MB 0–31	MB 32–95	Elements 0–31	Elements 32–103
Dh	112	MB 0–33	MB 34–95	Elements 0–31	Elements 32–111

Table continues on the next page...

Table 956. Possible Legacy RX FIFO filter structures (continued)

RFFN[3:0]	Number of Legacy RX FIFO filter elements	Message buffers occupied by Legacy RX FIFO and ID filter table	Remaining available message buffers	Legacy RX FIFO ID filter table elements affected by RX individual masks	Legacy RX FIFO ID filter table elements affected by Legacy RX FIFO global mask
Eh	120	MB 0–35	MB 36–95	Elements 0–31	Elements 32–119
Fh	128	MB 0–37	MB 38–95	Elements 0–31	Elements 32–127

Diagram



Fields

Field	Function
31 ERRMSK_FAST	Error Interrupt Mask for Errors Detected in the Data Phase of Fast CAN FD Frames Enables the ESR1[ERRINT_FAST] interrupt. 0b - Disable 1b - Enable
30 BOFFDONEMSK	Bus Off Done Interrupt Mask Enables the Bus Off Done interrupt, ESR1[BOFFDONEINT]. 0b - Disable 1b - Enable
29 ECRWRE	Error Correction Configuration Register Write Enable Enables updates for Memory Error Control (MECR). If the protocol described in Detection and correction of memory errors is not followed, this field is automatically 0. 0b - Disable 1b - Enable
28 WRMFRZ	Write Access to Memory in Freeze Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables unrestricted write access to FlexCAN memory in Freeze mode. When this field is 0, write access restrictions are maintained. This field can only be written in Freeze mode, and has no effect out of Freeze mode. **NOTE** Do not write 1 to MCR[RFEN] during FlexCAN memory initialization. 0b - Disable 1b - Enable
27-24 RFFN	Number of Legacy Receive FIFO Filters Defines the number of Receive Legacy FIFO filters, as shown in Table 956. The chip determines the maximum selectable number of filters. Do not program this field with values that cause the number of message buffers occupied by Legacy RX FIFO and Legacy RX FIFO ID Filter to exceed MCR[MAXMB]. MCR[MAXMB] defines the number of message buffers present. This field can only be written in Freeze mode; the module blocks it in other modes. Each group of eight filters occupies a memory space equivalent to two message buffers. The more filters are implemented, the fewer message buffers are available. The Legacy RX FIFO occupies the memory space originally reserved for MB5–MB0. This field should be programmed with a value corresponding to a number of filters less than the number of available memory words. The number of available words can be calculated as follows: (SETUP_MB - 6) × 4 Where SETUP_MB is the smaller of the parameter NUMBER_OF_MB and MCR[MAXMB]. The number of remaining message buffers available is: (SETUP_MB - 8) - (RFFN × 2) If the number of Legacy RX FIFO filters programmed through RFFN exceeds the SETUP_MB value (memory space available), the exceeding ones are not functional. **NOTE** • The number of the last remaining available message buffers is the smaller of (NUMBER_OF_MB - 1) and MCR[MAXMB]. • If RX Individual Mask registers are not enabled, the Legacy RX FIFO Global Mask affects all Legacy RX FIFO filters.
23-19 TASD	Transmission Arbitration Start Delay Indicates by how many CAN bits the transmission arbitration process start point can be delayed from the first bit of CRC field on CAN bus. See TX arbitration start delay for details. This field can be written only in Freeze mode; the module blocks it in other modes.
18 MRP	Message Buffers Reception Priority Sets the priority for the matching process. This field can be written only in Freeze mode; the module blocks it in other modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Matching starts from Legacy RX FIFO or Enhanced RX FIFO and continues on message buffers. 1b - Matching starts from message buffers and continues on Legacy RX FIFO or Enhanced RX FIFO.
17 RRS	Remote Request Storing Determines what the module does with a remote request. The remote request frame is submitted to a matching process. If this field is 1, the frame is stored in the corresponding message buffer in the same fashion as a data frame. No automatic remote response frame is generated. If this field is 0, an automatic remote response frame is generated if a message buffer with CODE = 1010b is found with the same ID. You can only write to this field in Freeze mode. The module blocks it in other modes. 0b - Generated 1b - Stored
16 EACEN	Entire Frame Arbitration Field Comparison Enable for RX Message Buffers Controls the comparison of IDE and RTR fields within RX message buffer filters with their corresponding bits in the incoming frame by the matching process. If enabled, the IDE and RTR fields of the RX message buffer are compared to their corresponding bits within the incoming frame (mask bits apply). If disabled, the IDE field of the RX message buffer filter is always compared and RTR is never compared despite mask bits. This field does not affect matching for Legacy RX FIFO or Enhanced RX FIFO. You can only write to this field in Freeze mode; the module blocks it in other modes. 0b - Disable 1b - Enable
15 TIMER_SRC	Timer Source Selects the time tick source used for incrementing the free-running timer counter. If CAN bit clock is selected, it defines the baud rate on the CAN bus. If External time tick is selected, the period can be adjusted to match the baud rate on the CAN bus. It can also be adjusted to a different value as required. See the chip-specific section for details about the external time tick. You can only write to this field in Freeze mode. 0b - CAN bit clock 1b - External time tick
14 PREXCEN	Protocol Exception Enable Enables the protocol exception feature. You can only write to this field in Freeze mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE See Protocol exception event in the CAN Protocol standard (ISO 11898-1:2015) for details. 0b - Disabled 1b - Enabled
13 BTE	Bit Timing Expansion Enable Enables the use of Enhanced CAN Bit Timing Prescalers (EPRS), Enhanced Data Phase CAN Bit Timing (EDCBT), and Enhanced Nominal CAN Bit Timing (ENCBT) to configure the CAN bit timing segments, instead of using the bit timing fields of CAN Bit Timing (CBT), CAN FD Bit Timing (FDCBT), and Control 1 (CTRL1). If this field is 1: • CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] are read as zero. A write operation to these fields has no effect. • CBT[EPRESDIV], CBT[ERJW], CBT[EPROPSEG], CBT[EPSEG1], and CBT[EPSEG2], and the corresponding fields in CAN FD Bit Timing (FDCBT), are read as zero. A write operation to these fields has no effect. • ETDC[ETDCOFF], ETDC[ETDCEN], ETDC[ETDCFAIL], and ETDC[ETDCVAL] are used by FlexCAN instead of FDCTRL[TDCOFF], FDCTRL[TDCEN], FDCTRL[TDCFAIL], and FDCTRL[TDCVAL]. These fields are read as zero, and a write operation to them has no effect. • ETDC[TDMDIS] can be used to disable transceiver delay measurement. 0b - Disable 1b - Enable
12 ISOCANFDEN	ISO CAN FD Enable Enables the CAN FD protocol according to ISO specification (ISO 11898-1:2015) (see CAN FD ISO compliance). When disabled, FlexCAN operates using the non-ISO CAN FD protocol. You can only write to this field in Freeze mode. **NOTE** FlexCAN is able to transmit FD frame format according to CAN Protocol standard (ISO 11898-1:2015). 0b - Disable 1b - Enable
11 EDFLTDIS	Edge Filter Disable Disables the edge filter used during the Bus Integration state. When the Edge Filter is enabled, two consecutive nominal time quanta with dominant bus states are required to detect an edge that causes synchronization. When synchronization occurs, the counting of the sequence of 11 consecutive recessive bits is restarted. The edge filter prevents dominant pulses that are shorter than a nominal bit time (present during the data phase of an FD frame) from being mistaken for an idle condition.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	You can only write to this field in Freeze mode. **NOTE** See Bus Integration state in the CAN Protocol standard (ISO 11898-1:2015) for details. 0b - Enabled 1b - Disabled
10 —	Reserved
9-8 MBTSBASE	Message Buffer Timestamp Base Selects the timebase used for capturing the 16-bit TIME_STAMP field of the message buffer register. This field can be written in Freeze Mode only. 00b - TIMER 01b - Lower 16 bits of high-resolution timer 10b - Upper 16 bits of high-resolution timer 11b - Reserved
7-6 TSTAMPCAP	Timestamp Capture Point Configures the point in time when a 32-bit timebase is captured during a CAN frame. This base is stored in the high-resolution timestamp register (HR_TIME_STAMP). For classical CAN frames, capture points can be the start-of-frame bit or the point when CAN frame is considered valid. This point is the seventh bit of the end-of-frame for transmission and the sixth bit of the end-of-frame for reception. For CAN FD frames, the high-resolution timestamp can be captured at the start of frame, when a CAN FD frame is considered valid, or the res bit. You can only write to this field in Freeze mode. 00b - Disabled 01b - End of the CAN frame 10b - Start of the CAN frame 11b - Start of frame for classical CAN frames; res bit for CAN FD frames
5 —	Reserved
4-2 —	Reserved
1-0 —	Reserved

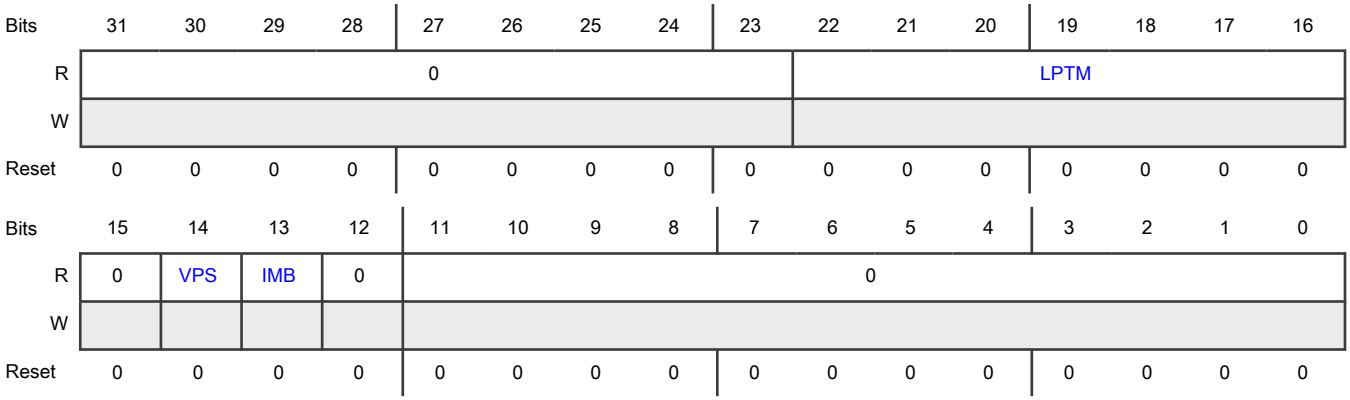
59.6.2.15 Error and Status 2 (ESR2)

Offset

Register	Offset
ESR2	38h

Function
Reports general status information.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 LPTM	Lowest Priority TX Message Buffer Indicates the lowest number inactive message buffer when ESR2[VPS] = 1 (see ESR2[IMB]). If no message buffer is inactive, the message buffer indicated depends on the value of CTRL1[LBUF]. If CTRL1[LBUF] = 0, the message buffer indicated is the one with the greatest arbitration value (see Highest-priority message buffer first). If CTRL1[LBUF] = 1, the message buffer indicated is the active TX message buffer with the highest number. If a TX message buffer is being transmitted, it is not considered in the LPTM calculation. If ESR2[IMB] is not 0 and a frame is transmitted successfully, the value of LPTM is updated with its message buffer number.
15 —	Reserved
14 VPS	Valid Priority Status Indicates whether the contents of ESR2[IMB] and ESR2[LPTM] are valid. It becomes 1 upon every complete TX arbitration process, unless the CPU writes to the Control and Status word of a message buffer already scanned. In other words, it is behind the TX Arbitration Pointer, during the TX arbitration process. If there is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	no inactive message buffer and only one TX message buffer that is being transmitted, this field remains 0. This field becomes 0 upon the start of every TX arbitration process or upon a write to the Control and Status word of any message buffer. **NOTE** No CPU write to the Control and Status of a message buffer that the abort mechanism blocks affects this field. When MCR[AEN] = 1, the abort code write to the Control and Status of an MB being transmitted (pending abort) is blocked. Any write attempt to a TX MB with its IFLAG flag set is also blocked. 0b - Invalid 1b - Valid
13 IMB	Inactive Message Buffer Indicates whether any message buffer is inactive (CODE field is either 1000b or 0b) when ESR2[VPS] = 1. This field becomes 1 when: • A lowest-priority TX message buffer (ESR2[LPTM]) is found and it is inactive during arbitration. • This field is not 1, and a frame is transmitted successfully. This field always becomes 0 at the start of arbitration (see Arbitration process). If a message buffer is successfully transmitted and this field is 0 (no inactive message buffer), ESR2[VPS] and this field both become 1. The index related to the MB transmitted is loaded into ESR2[LPTM]. In this case, the value of ESR2[LPTM] is the number of the first inactive message buffer. 0b - Message buffer indicated by ESR2[LPTM] is not inactive. 1b - At least one message buffer is inactive.
12 —	Reserved
11-0 —	Reserved

59.6.2.16 Cyclic Redundancy Check (CRCR)

Offset

Register	Offset
CRCR	44h

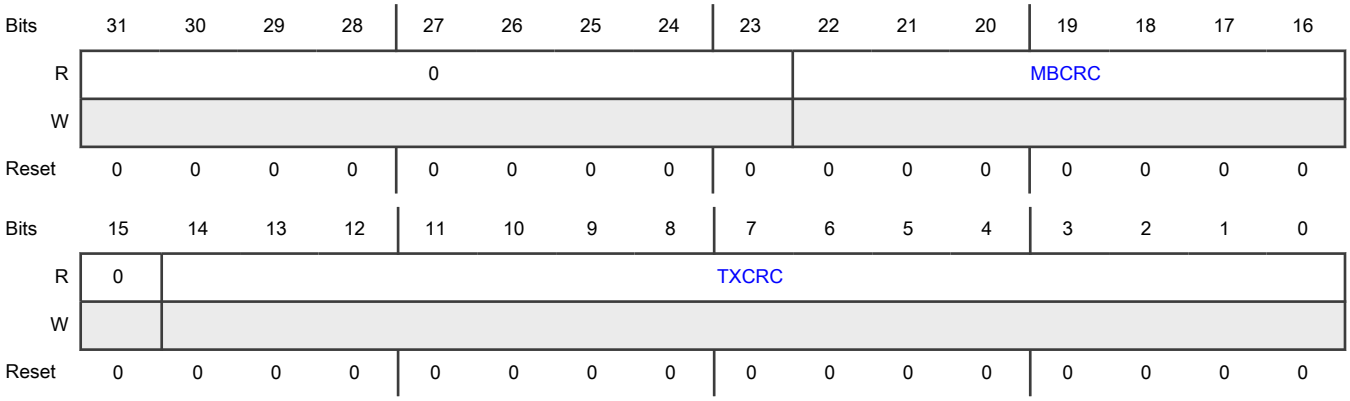
Function

Provides information about the CRC of transmitted messages for non-FD messages. This register only reports the 15 low-order bits of CRC calculations for messages in CAN FD format that require either 17 or 21 bits. For CAN FD format frames, you must use the CAN FD CRC (FDCRC). This register is updated at the same time that the TX interrupt flag is set.

NOTE

See CRC sequence calculation in the CAN Protocol standard (ISO 11898-1:2015) for details.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 MBCRC	CRC Message Buffer Indicates the number of the message buffer corresponding to the value in CRCR[TXCRC].
15 —	Reserved
14-0 TXCRC	Transmitted CRC value Indicates the CRC value of the last transmitted message for non-FD frames. For FD frames, CRC value is reported in CAN FD CRC (FDCRC).

59.6.2.17 Legacy RX FIFO Global Mask (RXFGMASK)

Offset

Register	Offset
RXFGMASK	48h

Function

Masks the Legacy RX FIFO ID filter table elements that do not have a corresponding RXIMR according to CTRL2[RFFN], when Legacy RX FIFO is enabled.

This register is located in RAM.

You can only write to this register in Freeze mode; the module blocks it in other modes.

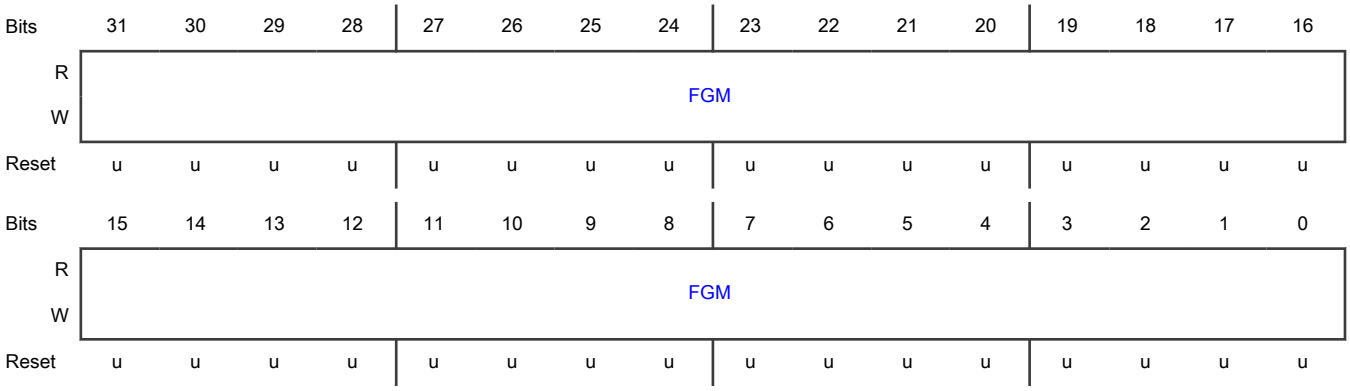
The following table shows how the FGM bits correspond to each IDAF field.

Table 957. Correspondence of Legacy RX FIFO global mask bits to IDF fields

Legacy RX FIFO ID filter table elements format (MCR[IDAM])	Identifier acceptance filter fields					
	RTR	IDE	RXIDA	RXIDB [1]	RXIDC [2]	Reserved
A	FGM[31]	FGM[30]	FGM[29:1]	—	—	FGM[0]
B	FGM[31], FGM[15]	FGM[30], FGM[14]	—	FGM[29:16], FGM[13:0]		—
C	—	—		—	FGM[31:24], FGM[23:16], FGM[15:8], FGM[7:0]	

1. If MCR[IDAM] is equivalent to format B, only the 14 most significant bits of the identifier of the incoming frame are compared with the Legacy RX FIFO filter.
2. If MCR[IDAM] is equivalent to format C, only the eight most significant bits of the identifier of the incoming frame are compared with the Legacy RX FIFO filter.

Diagram



Fields

Field	Function
31-0 FGM	Legacy RX FIFO Global Mask Bits Masks the ID filter table elements bits in a perfect alignment. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

59.6.2.18 Legacy RX FIFO Information (RXFIR)

Offset

Register	Offset
RXFIR	4Ch

Function

Provides information about Legacy RX FIFO.

This register is the port through which the CPU accesses the output of the Legacy RXFIR FIFO located in RAM. FlexCAN writes to the Legacy RXFIR FIFO when a new message is moved into the Legacy RX FIFO. Also, its output is updated whenever the output of the Legacy RX FIFO is updated with the next message. See Legacy RX FIFO for instructions on reading this register.

NOTE

RXFIR can be written only during memory initialization, due to the error code correction (ECC) feature. In every other case, this register is read-only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								IDHIT							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-9 —	Reserved
8-0 IDHIT	Identifier Acceptance Filter Hit Indicator Indicates which Identifier Acceptance filter that the received message hit in the output of the Legacy RX FIFO. If multiple filters match the incoming message ID, the first matching IDAF found (lowest number) by the matching process is indicated. This field is valid only when IFLAG1[BUF5I] is set.

59.6.2.19 CAN Bit Timing (CBT)

Offset

Register	Offset
CBT	50h

Function

Provides an alternative way to store the CAN bit timing variables described in Control 1 (CTRL1). EPRESDIV, EPROPSEG, EPSEG1, EPSEG2, and ERJW are extended versions of CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] respectively.

NOTE

The CAN bit variables in CTRL1 and in CBT are stored in the same register.

CBT[BTF] selects the use of the timing variables defined in this register.

When the CAN FD feature is enabled (MCR[FDEN] = 1), always write 1 to CBT[BTF].

Soft reset does not affect the contents of this register.

NOTE

Ensure that bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1:2015).

NOTE

If CTRL2[BTE] = 1, EPRESDIV, ERJW, EPROPSEG, EPSEG1, and EPSEG2 are read as zero. A write operation to them has no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BTF	EPRESDIV											ERJW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EPROPSEG				EPSEG1				EPSEG2							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BTF	Bit Timing Format Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables the use of extended CAN bit timing fields EPRESDIV, EPROPSEG, EPSEG1, EPSEG2, and ERJW. These fields replace the CAN bit timing variables defined in Control 1 (CTRL1). This field can be written in Freeze mode only. 0b - Disable 1b - Enable
30-21 EPRESDIV	Extended Prescaler Division Factor Defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PRESDIV] value range. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). This field can be written only in Freeze mode; the module blocks it in other modes. Sclock frequency = PE clock frequency ÷ (EPRESDIV + 1)
20-16 ERJW	Extended Resync Jump Width Defines the maximum number of time quanta that one resynchronization can change a bit time when CBT[BTF] = 1. Otherwise, it has no effect. It extends the CTRL1[RJW] value range. This field can be written only in Freeze mode; the module blocks it in other modes. Resync Jump Width = ERJW + 1. One Time Quantum = one Sclock period.
15-10 EPROPSEG	Extended Propagation Segment Defines the length of the propagation segment in the bit time when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PROPSEG] value range. This field can be written only in Freeze mode; the module blocks it in other modes. Propagation Segment Time = (EPROPSEG + 1) × Time Quanta. One Time Quantum = one Sclock period.
9-5 EPSEG1	Extended Phase Segment 1 Defines the length of phase segment 1 in the bit time when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PSEG1] value range. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Buffer Segment 1 = (EPSEG1 + 1) × Time Quanta. One Time Quantum = one Sclock period.
4-0 EPSEG2	Extended Phase Segment 2 Defines the length of phase segment 2 in the bit time when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PSEG2] value range. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Buffer Segment 1 = (EPSEG2 + 1) × Time Quanta. One Time Quantum = one Sclock period.

59.6.2.20 Interrupt Masks 3 (IMASK3)

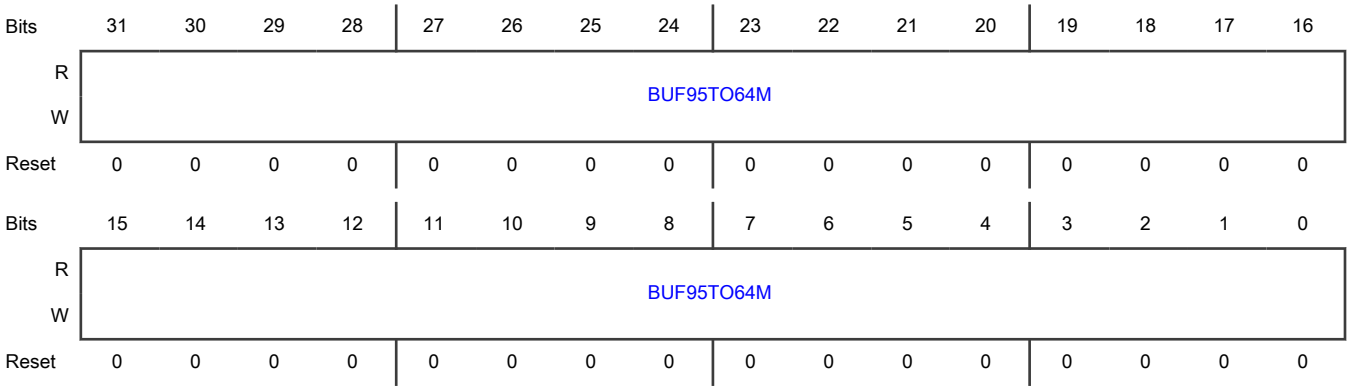
Offset

Register	Offset
IMASK3	6Ch

Function

Enables or disables any number of the 32 message buffer interrupts for MB95–MB64. It contains one interrupt mask bit per buffer. This configuration allows the CPU to determine which buffer generates an interrupt after a successful transmission or reception when the corresponding IFLAG3 flag is set.

Diagram



Fields

Field	Function
31-0 BUF95TO64M	Buffer MBi Mask Enables or disables the corresponding FlexCAN message buffer interrupt for MB95–MB64. When CAN FD is enabled, the MB range is defined in accordance with FDCTRL[MBDSRn]. **NOTE** If the corresponding IFLAG3 flag is set, writing 1 or 0 to a field in IMASK3 can set or clear an interrupt request. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

59.6.2.21 Interrupt Flags 3 (IFLAG3)

Offset

Register	Offset
IFLAG3	74h

Function

Defines the flags for the 32 message buffer interrupts for MB95–MB64. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG3 flag. If the corresponding IMASK3 bit is 1, an interrupt is generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect.

Before updating MCR[MAXMB], the CPU must service the IFLAG3 flags whose MB value is greater than the MAXMB to be updated. Otherwise, they remain set and are inconsistent with the number of message buffers available.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BUF95TO64															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BUF95TO64															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF95TO64	Buffer MBi Interrupt Flags the corresponding FlexCAN message buffer interrupt for MB95–MB64.When CAN FD is enabled, the MB range is defined in accordance with FDCTRL[MBDSRn]. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.

59.6.2.22 Receive Individual Mask (RXIMR0 - RXIMR95)

Offset

For n = 0 to 95:

Register	Offset
RXIMRn	880h + (n × 4h)

Function

Stores the acceptance masks for ID filtering in RX message buffers and the Legacy RX FIFO.

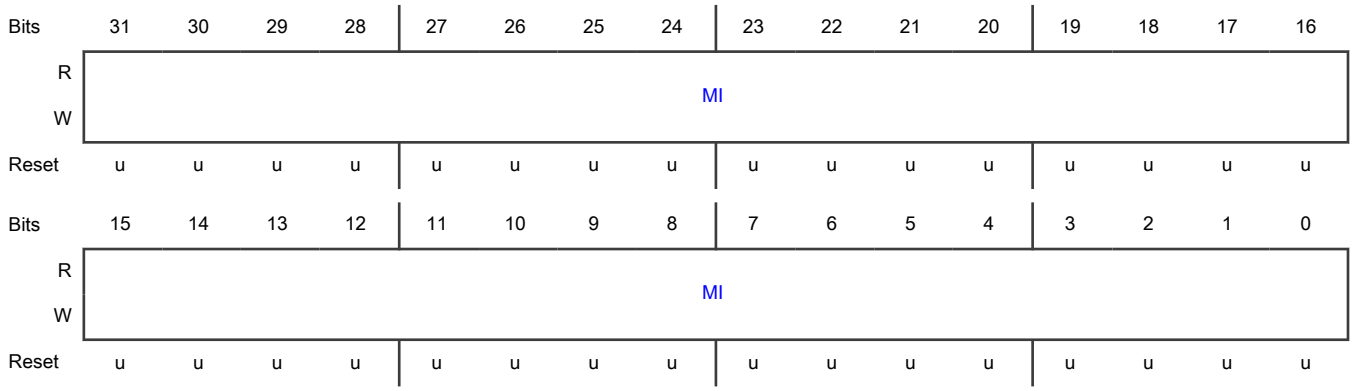
When the Legacy RX FIFO is disabled (MCR[RFEN] = 0), an individual mask is provided for each available RX message buffer on a one-to-one correspondence. When the Legacy RX FIFO is enabled (MCR[RFEN] = 1), an individual mask is provided for each Legacy RX FIFO ID filter table element on a one-to-one correspondence. This correspondence depends on the setting of CTRL2[RFFN] (see Legacy RX FIFO).

RXIMR0 stores the individual mask associated with either MB0 or ID filter table element 0. RXIMR1 stores the individual mask associated with either MB1 or ID filter table element 1, and so on.

The CPU can only access the RXIMR registers when the module is in Freeze mode; otherwise, the module blocks them. Reset does not affect these registers. They are located in RAM and must be explicitly initialized prior to any reception.

It is possible for the RXIMR memory region to be accessed as general-purpose memory. See Bus interface for more information.

Diagram



Fields

Field	Function
31-0 MI	Individual Mask Bits Masks the corresponding bit in both the message buffer filter and Legacy RX FIFO ID filter table element in distinct ways. For message buffer filters, see RX Message Buffers Global Mask (RXMGMASK). For Legacy RX FIFO ID filter table elements, see Legacy RX FIFO Global Mask (RXFGMASK). 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

59.6.2.23 Memory Error Control (MECR)

Offset

Register	Offset
MECR	AE0h

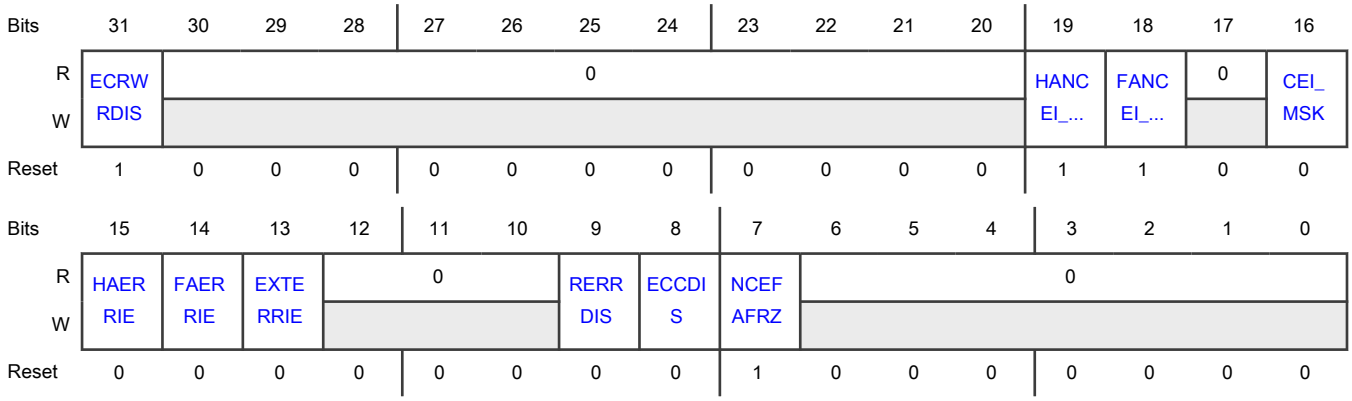
Function

Contains control bits for memory error detection and correction (ECC).

NOTE

When CTRL2[ECRWRE] = 0, writes to this register are blocked, except for MECR[ECRWRDIS].

Diagram



Fields

Field	Function
31 ECRWRDIS	Error Configuration Register Write Disable Disables writes on this register. This field automatically becomes 1 (disabled) when CTRL2[ECRWRE] is 1. The protocol described in Detection and correction of memory errors must be followed. 0b - Enable 1b - Disable
30-20 —	Reserved
19 HANCEI_MSK	Host Access with Noncorrectable Errors Interrupt Mask Enables the interrupt for noncorrectable errors detected in memory reads issued by the host (CPU). 0b - Disable 1b - Enable
18 FANCEI_MSK	FlexCAN Access with Noncorrectable Errors Interrupt Mask Enables the interrupt for noncorrectable errors detected in memory reads issued by FlexCAN internal processes. 0b - Disable 1b - Enable
17 —	Reserved
16 CEI_MSK	Correctable Errors Interrupt Mask Enables the interrupt for correctable errors detected in memory reads issued by the host or FlexCAN internal processes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
15 HAERRIE	Host Access Error Injection Enable Enables the injection of errors only in memory reads issued by the host (CPU). 0b - Disable 1b - Enable
14 FAERRIE	FlexCAN Access Error Injection Enable Enables the injection of errors only in memory reads issued by FlexCAN internal processes. 0b - Disable 1b - Enable
13 EXTERRIE	Extended Error Injection Enable Extends the error injection on 32-bit memory accesses to the complementary 32-bit word. This feature uses the same 32-bit error injection data and parity words used for 64-bit memory accesses performed by internal FlexCAN processes. See Error Injection Data Pattern (ERRIDPR) and Error Injection Parity Pattern (ERRIPPR). 0b - Disable. Apply error injection only to the 32-bit word. 1b - Enable. Apply error injection to the 64-bit word.
12-10 —	Reserved
9 RERRDIS	Error Report Disable Disables the update of the error report registers. The update of error-related flags and the generation of bus transfer errors are still active. **NOTE** When reading the report registers, this field must be 1 to ensure coherence on the consecutive register reads. 0b - Enable 1b - Disable
8 ECCDIS	Error Correction Disable Completely disables the memory detection and correction mechanism. Besides disabling the error report mechanism, it also stops the update of the error-related flags and the generation of bus transfer errors. The parity bits continue to be calculated and written into memory on write transactions. 0b - Enable 1b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 NCEFAFRZ	Noncorrectable Errors in FlexCAN Access Put Chip in Freeze Mode Determines whether to put FlexCAN into Freeze mode when a noncorrectable error is detected in a memory read performed by FlexCAN internal processes. In this case, entering Freeze mode prevents FlexCAN internal processes from treating corrupted data as valid. See Freeze mode for more information. 0b - Normal operation 1b - Freeze mode
6-0 —	Reserved

59.6.2.24 Error Injection Address (ERRIAR)

Offset

Register	Offset
ERRIAR	AE4h

Function

Contains the address where error is to be injected.

Use the following table to convert from the memory map address to the location in the physical FlexCAN RAM. Where pairs of values are provided, the first is the address for MCR[FDEN] = 0, the second is for MCR[FDEN] = 1.

Table 958. Error injection address for classical CAN format

RAM contents	Injection address	Memory map
FlexCAN registers	Not mapped	—
Message buffers	0000h	0080h
RXIMRs	600h	0880h
RXFIR_0	780h	0A80h
RXFIR_1	784h	0A84h
RXFIR_2	788h	0A88h
RXFIR_3	78Ch	0A8Ch
RXFIR_4	790h	0A90h
RXFIR_5	794h	0A94h
Reserved	—	0A98h
RXMGMASK	7A0h	0AA0h
RXFGMASK	7A4h	0AA4h

Table continues on the next page...

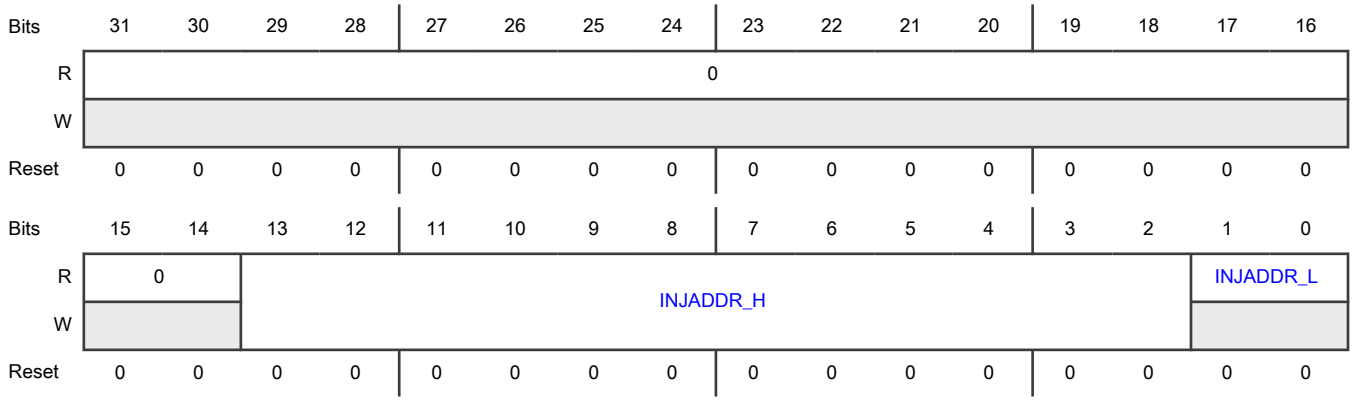
Table 958. Error injection address for classical CAN format (continued)

RAM contents	Injection address	Memory map
RX14MASK	7A8h	0AA8h
RX15MASK	7ACh	0AACh
Tx_SMB	7B0h	0AB0h / 0F28h
Rx_SMB0	7C0h / 7F8h	0AC0h / 0F70h
Rx_SMB1	7D0h / 840h	0AD0h / 0FB8h
Rx_SMB0_TIME_STAMP	888h	0C20h
Rx_SMB1_TIME_STAMP	88Ch	0C24h
HR_TIME_STAMP	890h	0C30h
Enhanced RX FIFO	A10h	2000h
ERFFEL	1050h	3000h

NOTE

For RXFIR, Enhanced RX FIFO, and HR_TIME_STAMP addresses, you must inject ECC errors in host access only.

Diagram



Fields

Field	Function
31-14 —	Reserved
13-2 INJADDR_H	Error Injection Address High Defines the 12 most significant bits of the physical RAM address where error is to be injected (see table above).
1-0	Error Injection Address Low

Table continues on the next page...

Table continued from the previous page...

Field	Function
INJADDR_L	Defines the two least significant bits of the physical RAM address where error is to be injected. These bits ensure that the address is on a 32-bit boundary.

59.6.2.25 Error Injection Data Pattern (ERRIDPR)

Offset

Register	Offset
ERRIDPR	AE8h

Function

Contains the error pattern to be injected in the data word read from memory.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DFLIP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DFLIP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DFLIP	Data Flip Pattern Contains the flip pattern. Bits set to 1 in the flip pattern cause the corresponding data bit in the word read from memory to invert.

59.6.2.26 Error Injection Parity Pattern (ERRIPPR)

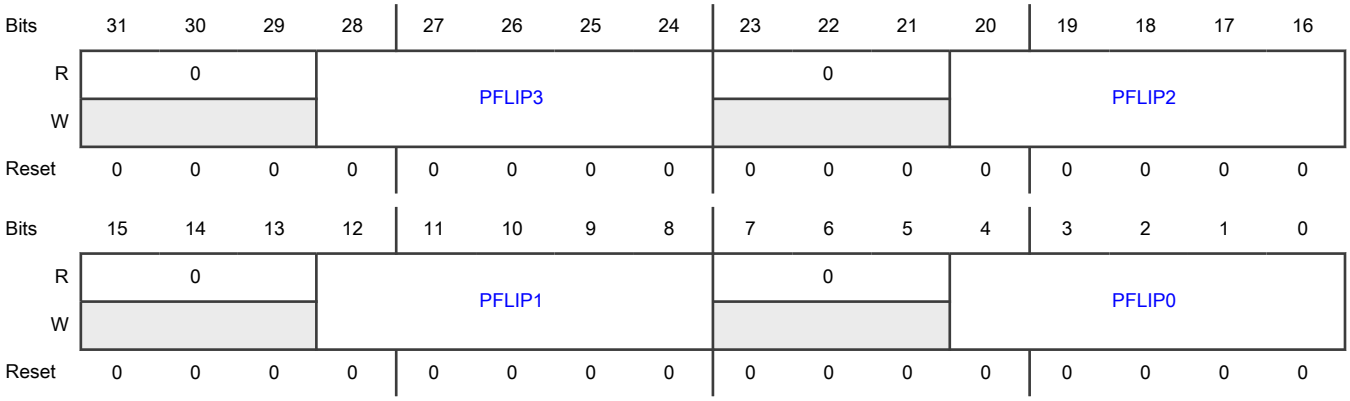
Offset

Register	Offset
ERRIPPR	AECh

Function

Contains the error pattern to be injected in parity bits read from memory with the data word. Bits set to 1 in the flip pattern cause the corresponding parity bit in the word read from memory to invert.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 PFLIP3	Parity Flip Pattern for Byte 3 (Most Significant) Contains the error injection pattern for Byte 3.
23-21 —	Reserved
20-16 PFLIP2	Parity Flip Pattern for Byte 2 Contains the error injection pattern for Byte 2.
15-13 —	Reserved
12-8 PFLIP1	Parity Flip Pattern for Byte 1 Contains the error injection pattern for Byte 1.
7-5 —	Reserved
4-0 PFLIP0	Parity Flip Pattern for Byte 0 (Least Significant) Contains the error injection pattern for Byte 0.

59.6.2.27 Error Report Address (RERRAR)

Offset

Register	Offset
RERRAR	AF0h

Function

Reports the address used for an access operation in which an error (correctable or noncorrectable) was detected. Also reports the identification of the source of that access.

This address is always reported using a 32-bit alignment. Non-aligned accesses (ERRADDR[1:0] nonzero) are reported with the address aligned, and data is reported in Error Report Data (RERRDR) shifted accordingly. When errors are detected in accesses larger than 32-bit (as performed by FlexCAN internal processes), the address of the 32-bit word where the error was detected is reported. For errors detected in more than one 32-bit word, only the least significant address is reported.

Table 959. Source of memory access

SAID[2:0]	Error during...
0	Move-out FlexCAN access
1	Move-in
2	TX arbitration
3	RX matching
4	Move-out host access
5–7	Reserved

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							NCE	0				SAID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		ERRADDR													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 NCE	Noncorrectable Error Indicates that the report is due to a noncorrectable error. 0b - Reporting a correctable error 1b - Reporting a noncorrectable error
23-19 —	Reserved
18-16 SAID	SAID SAID[2] — Identification of the requester of the memory read request: • 0 = Requested by FlexCAN internal processes • 1 = Requested by host (CPU) SAID[1] — Details of FlexCAN operation: • 0 = Move • 1 = Scanning SAID[0] — Operation that requested the memory read: • 0 = Transmission • 1 = Reception For more information, see Table 959.
15-14 —	Reserved
13-0 ERRADDR	Address Where Error Detected See description of Error Injection Address (ERRIAR).

59.6.2.28 Error Report Data (RERRDR)

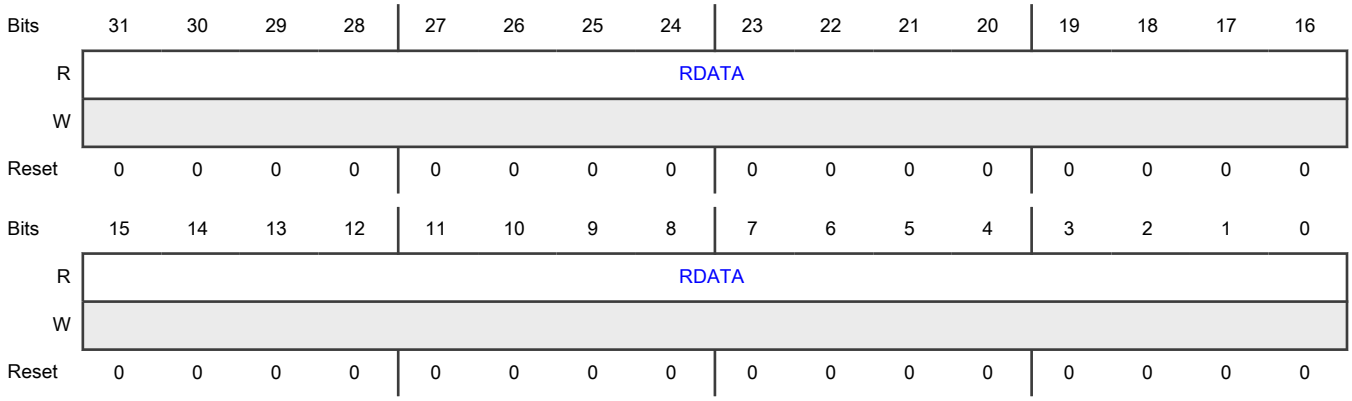
Offset

Register	Offset
RERRDR	AF4h

Function

Reports the raw data (unmodified by the correction performed by ECC logic) read from memory with error. The value reported does not represent the transient values of the BUSY bit (see Table 965) when reading a message buffer.

Diagram



Fields

Field	Function
31-0 RDATA	Raw Data Word Read from Memory with Error

59.6.2.29 Error Report Syndrome (RERRSYNR)

Offset

Register	Offset
RERRSYNR	AF8h

Function

Contains the syndrome detected in a memory read with error. It also reports the bytes which were read in this 32-bit read transaction.

Each SYND*n* field indicates the type of error and which bit in byte (*n*) the error affects. SYND3 corresponds to the most significant byte in the data word read from memory; SYND0 corresponds to the least significant.

Table 960. Syndrome definition

SYND*n* (hex)	Type	Bit affected
00	—	None (no error)
01	Code	0
02	Code	1
04	Code	2
07	Data	5
08	Code	3
0E	Data	7

Table continues on the next page...

Table 960. Syndrome definition (continued)

SYND*n* (hex)	Type	Bit affected
10	Code	4
13	Data	2
15	Data	6
16	Data	1
19	Data	3
1A	Data	4
1C	Data	0
06	—	All-zeroes noncorrectable error
1F	—	All-ones noncorrectable error
All others	—	Noncorrectable error

Each BEn field indicates which byte in the 32-bit word reported was effectively read. The syndrome bits are calculated for all bytes, including the non-read ones. Errors detected in non-read bytes are indicated (see Error indication) and reported (see Error reporting).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BE3	0			SYND3				BE2	0			SYND2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BE1	0			SYND1				BE0	0			SYND0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BE3	Byte Enabled for Byte 3 (Most Significant) 0b - Byte was not read. 1b - Byte was read.
30-29 —	Reserved
28-24	Error Syndrome for Byte 3 (Most Significant)

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYND3	See Table 960.
23 BE2	Byte Enabled for Byte 2 0b - Byte was not read. 1b - Byte was read.
22-21 —	Reserved
20-16 SYND2	Error Syndrome for Byte 2 See Table 960.
15 BE1	Byte Enabled for Byte 1 0b - Byte was not read. 1b - Byte was read.
14-13 —	Reserved
12-8 SYND1	Error Syndrome for Byte 1 See Table 960.
7 BE0	Byte Enabled for Byte 0 (Least Significant) 0b - Byte was not read. 1b - Byte was read.
6-5 —	Reserved
4-0 SYND0	Error Syndrome for Byte 0 (Least Significant) See Table 960.

59.6.2.30 Error Status (ERRSR)

Offset

Register	Offset
ERRSR	AFCh

Function

Contains the status bits of the error correction and detection operations. These flags can be cleared by writing 1 to them. Writing 0 has no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												HANCEIF	FANCEIF	0	CEIF
W													W1C	W1C		W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												HANCEIOF	FANCEIOF	0	CEIOF
W													W1C	W1C		W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 HANCEIF	Host Access with Noncorrectable Error Interrupt Flag Indicates that a noncorrectable error was detected in a memory read initiated by host. A bus transfer error is asserted for that access. If MECR[HANCEI_MSK] = 1, the interrupt is 1. 0b - No errors detected 1b - Error detected
18 FANCEIF	FlexCAN Access with Noncorrectable Error Interrupt Flag Indicates that a noncorrectable error was detected in a memory read initiated by FlexCAN internal processes. If MECR[FANCEI_MSK] = 1, the interrupt is 1. 0b - No errors detected 1b - Error detected
17 —	Reserved
16 CEIF	Correctable Error Interrupt Flag Indicates that a correctable error was detected in a memory read. If MECR[CEI_MSK] = 1, the interrupt is 1. 0b - No errors detected 1b - Error detected
15-4 —	Reserved
3	Host Access With Noncorrectable Error Interrupt Overrun Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
HANCEIOF	Indicates that a noncorrectable error was detected in a memory read initiated by host when ERRSR[HANCEIF] was set. No interrupt is associated with this flag. See Error indication. 0b - No errors detected 1b - Error detected
2 FANCEIOF	FlexCAN Access with Noncorrectable Error Interrupt Overrun Flag Indicates that a noncorrectable error was detected in a memory read initiated by FlexCAN internal processes when ERRSR[FANCEIF] was set. No interrupt is associated with this flag. See Error indication. 0b - No errors detected 1b - Error detected
1 —	Reserved
0 CEIOF	Correctable Error Interrupt Overrun Flag Indicates that a correctable error was detected in a memory read when ERRSR[CEIF] was set. No interrupt is associated with this flag. See Error indication. 0b - No errors detected 1b - Error detected

59.6.2.31 Enhanced CAN Bit Timing Prescalers (EPRS)

Offset

Register	Offset
EPRS	BF0h

Function

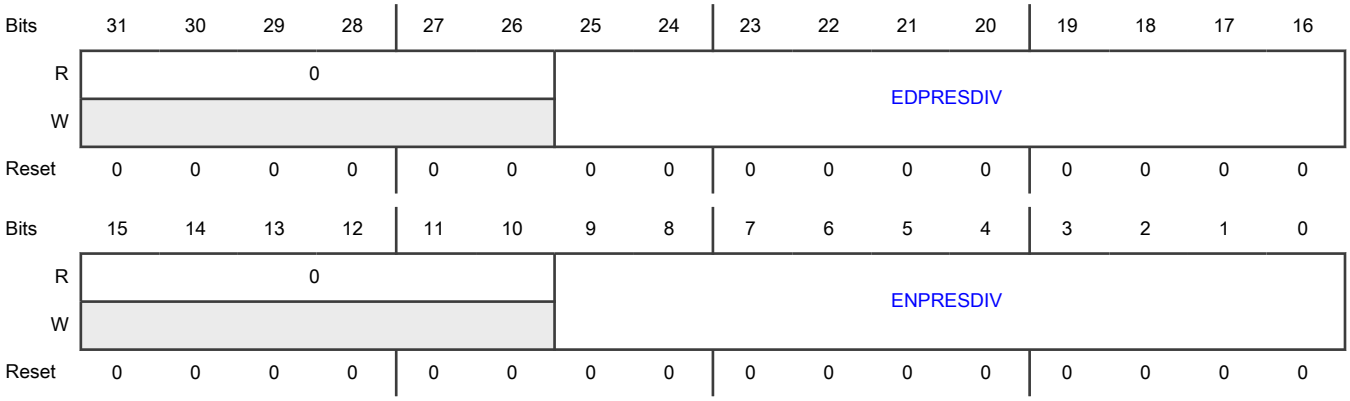
Defines the CAN bit timing prescalers for the nominal phase and data phase when CTRL2[BTE] = 1.

Used by the module only if CTRL2[BTE] = 1; otherwise a write operation has no effect and all fields are read as zero.

This register can be written only in Freeze mode; the module blocks it in other modes.

Soft reset does not affect the contents of this register.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 EDPRESDIV	Extended Data Phase Prescaler Division Factor Defines the ratio between the PE clock frequency and the Sclock frequency in the data phase of a CAN FD message when CTRL2[BTE] = 1. The Sclock period defines the time quantum of the CAN FD protocol for the data bit rate. Sclock frequency = PE clock frequency ÷ (EDPRESDIV + 1) **NOTE** To minimize errors when processing FD frames, use the same value for this field and for EPRS[ENPRESDIV]. See the first note in CAN FD frames for details.
15-10 —	Reserved
9-0 ENPRESDIV	Extended Nominal Prescaler Division Factor Defines the ratio between the PE clock frequency and the Sclock frequency when CTRL2[BTE] = 1. Otherwise, it reads as 0 and a write operation has no effect. The Sclock period defines the time quantum of the CAN protocol in the nominal phase. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). Sclock frequency = PE clock frequency ÷ (ENPRESDIV + 1)

59.6.2.32 Enhanced Nominal CAN Bit Timing (ENCBT)

Offset

Register	Offset
ENCBT	BF4h

Function

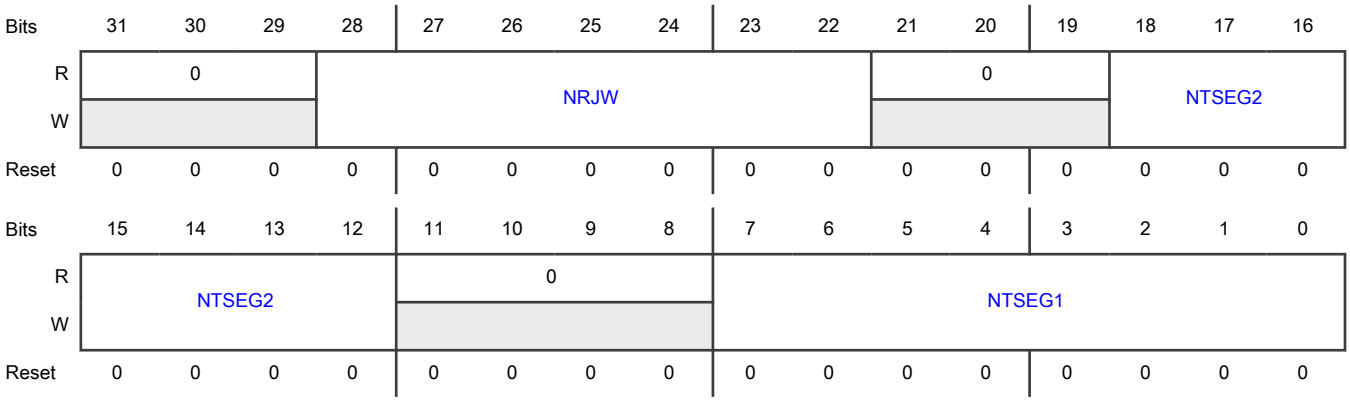
Provides an alternative way to store the CAN bit timing variables described in Control 1 (CTRL1) and CAN Bit Timing (CBT), to get higher CAN bit timing resolution.

This register is used by the module only if CTRL2[BTE] = 1. Otherwise, a write operation has no effect and all fields are read as zero.

Soft reset does not affect the contents of this register.

This register can be written only in Freeze mode; the module blocks it in other modes.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-22 NRJW	Nominal Resynchronization Jump Width Defines the maximum number of time quanta that one resynchronization can change a nominal bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. One time quantum = one Sclock period Nominal Resync Jump Width = NRJW + 1
21-19 —	Reserved
18-12 NTSEG2	Nominal Time Segment 2 Defines the length of Time Segment 2 in the nominal bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Nominal Time Segment 2 = (NTSEG2 + 1) × Time Quanta One time quantum = one Sclock period
11-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 NTSEG1	Nominal Time Segment 1 Defines the length of Time Segment 1 in the bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Nominal Time Segment 1 = (NTSEG1 + 1) × Time Quanta One time quantum = one Sclock period

59.6.2.33 Enhanced Data Phase CAN Bit Timing (EDCBT)

Offset

Register	Offset
EDCBT	BF8h

Function

Provides an alternative way to store the data phase CAN bit timing variables described in CAN FD Bit Timing (FDCBT) to achieve higher CAN bit timing resolution.

This register is used by the module only if CTRL2[BTE] = 1; otherwise a write operation has no effect and all fields are read as zero.

Soft reset does not affect the contents of this register.

This register can be written only in Freeze mode; the module blocks it in other modes.

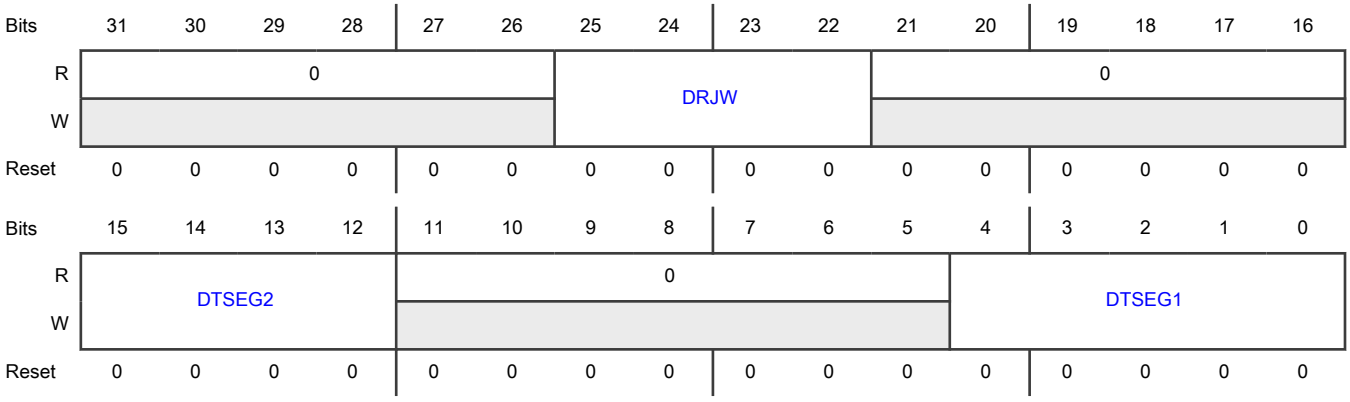
NOTE

Ensure that bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1:2015).

NOTE

DTSEG1 must be at least two time quanta.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-22 DRJW	Data Phase Resynchronization Jump Width Defines the maximum number of time quanta that one resynchronization can change a data phase bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Data Phase Resync Jump Width = DRJW + 1.
21-16 —	Reserved
15-12 DTSEG2	Data Phase Time Segment 2 Defines the length of time segment 2 in the data phase bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Data Phase Time Segment 2 = (DTSEG2 + 1) × Time Quanta. One Time Quantum = one Sclock period.
11-5 —	Reserved
4-0 DTSEG1	Data Phase Segment 1 Defines the length of time segment 1 in the data phase bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Data Phase Time Segment 1 = (NTSEG1 + 1) × Time Quanta. One Time Quantum = one Sclock period.

59.6.2.34 Enhanced Transceiver Delay Compensation (ETDC)

Offset

Register	Offset
ETDC	BFCh

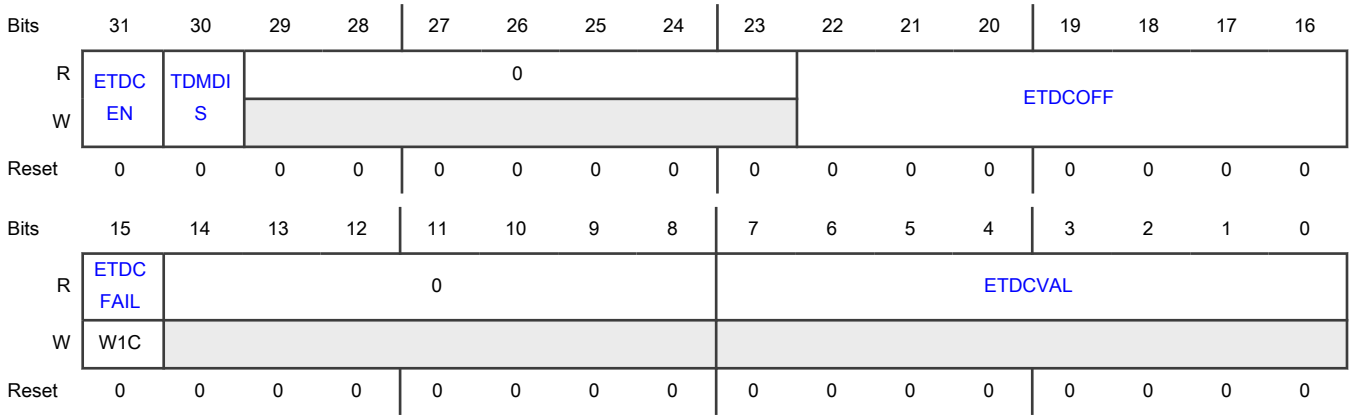
Function

Contains extended versions of FDCTRL[TDCOFF] and FDCTRL[TDCVAL]. This register is used by the module only if CTRL2[BTE] = 1. Otherwise, a write operation has no effect and all fields are read as zero.

NOTE

See Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1:2015) for details.

Diagram



Fields

Field	Function
31 ETDCEN	Transceiver Delay Compensation Enable Enables the TDC feature. It can be written in Freeze mode only. **NOTE** See Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1:2015) for details. **NOTE** TDC must be disabled when the Loop Back Mode is enabled. See CTRL1[LPB]. 0b - Disable 1b - Enable
30 TDMDIS	Transceiver Delay Measurement Disable Disables the transceiver delay measurement. When the TDC measurement is disabled, only ETDC[ETDCOFF] determines the secondary sample point position. If TCD measurement is enabled, the sum of the transceiver delay measurement plus the enhanced TDC offset determines the secondary sample point position. Soft reset does not affect this field. **NOTE** This bit can be enabled only if CTRL2[BTE] = 1. 0b - Enable 1b - Disable
29-23 —	Reserved
22-16	Enhanced Transceiver Delay Compensation Offset

Table continues on the next page...

Table continued from the previous page...

Field	Function
ETDCOFF	Contains the offset value to be added to the loop delay of the measured transceiver. This value defines the position of the delayed comparison point when bit rate switching is active. See Transceiver delay compensation for details on how the loop delay measurement is performed. This field can be written in Freeze mode only. Its value can be defined in protocol engine (PE) clock periods (CANCLK, see Protocol timing for more details). It must be smaller than the CAN bit duration in the data bit rate for proper operation. Do not write 0 to this field. **NOTE** If CTRL2[BTE] becomes 1 after a chip-level hard reset, this field is read as 1h.
15 ETDCFAIL	Transceiver Delay Compensation Fail Indicates whether the transceiver delay compensation (TDC) mechanism is out of range. In this case, it is unable to compensate the loop delay of the transceiver and successfully compare the delayed received bits to the transmitted ones. (See Transceiver delay compensation.) This field becomes 0 the first time FlexCAN detects the out of range condition. 0b - In range 1b - Out of range
14-8 —	Reserved
7-0 ETDCVAL	Enhanced Transceiver Delay Compensation Value Contains ETDC[ETDCOFF] added to the measured value of the transceiver loop delay in the latest transmitted CAN FD frame, with BRS = 1. The module only updates this field when ETDC[ETDCEN] = 1. Soft reset affects this field. **NOTE** If ETDC[TDMDIS] = 1, this field stores ETDC[ETDCOFF] only.

59.6.2.35 CAN FD Control (FDCTRL)

Offset

Register	Offset
FDCTRL	C00h

Function

Contains control bits for CAN FD operation. It also defines the data size of message buffers allocated in different partitions of RAM (memory blocks) as described in Table 961.

When an 8-byte payload is selected:

- Block R0 allocates MB0–MB31.
- Block R1 allocates MB32–MB63.
- Block R2 allocates MB64–MB95.

When a payload larger than eight bytes is selected, the maximum number of message buffers in a block is limited as described below.

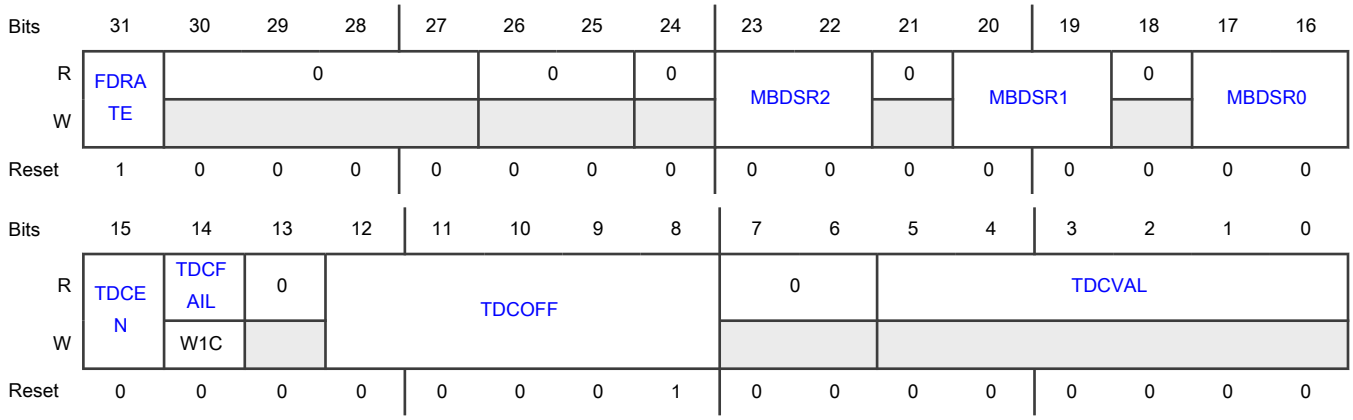
Table 961. Number of message buffers

Payload size	Maximum number of message buffers per RAM block
8 bytes	32
16 bytes	21
32 bytes	12
64 bytes	7

One memory block fits exactly 32 message buffers with an 8-byte payload. For other possible payload sizes, empty memory may exist between the last message buffer in a block and the beginning of the next block. This empty memory corresponds to less than one message buffer, and must not be used.

Soft reset does not affect the contents of this register.

Diagram



Fields

Field	Function
31 FDRATE	Bit Rate Switch Enable Enables the effect of the Bit Rate Switch (BRS bit) during the data phase of TX messages. When 1, if BRS = 1 in the TX message buffer, frames are transmitted with bit rate switching. When 0, frames are transmitted at a nominal rate, and the BRS bit in the TX MB has no effect. The CPU can write to this field at any time. However, its effect becomes active only under one of these conditions: • The CAN bus is in the Wait for Bus Idle state. • The CAN bus is in the Bus Idle state.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• The CAN bus is in the Bus Off state. • The current frame under reception or transmission reaches the interframe space. By writing 0 to FDCTRL[FDRATE], the CPU can force all bits in CAN FD messages to be transmitted at nominal bit rate. This transmission occurs regardless of the value in the BRS bit of the TX message buffers. 0b - Disable 1b - Enable
30-27 —	Reserved
26-25 —	Reserved
24 —	Reserved
23-22 MBDSR2	Message Buffer Data Size for Region 2 Selects the data size per message buffer for region R2 of message buffers allocated in RAM. This field can be written in Freeze mode only. 00b - 8 bytes 01b - 16 bytes 10b - 32 bytes 11b - 64 bytes
21 —	Reserved
20-19 MBDSR1	Message Buffer Data Size for Region 1 Selects the data size per message buffer for region R1 of message buffers allocated in RAM. This field can be written in Freeze mode only. 00b - 8 bytes 01b - 16 bytes 10b - 32 bytes 11b - 64 bytes
18 —	Reserved
17-16	Message Buffer Data Size for Region 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBDSR0	Selects the data size per message buffer for region R0 of message buffers allocated in RAM. This field can be written in Freeze mode only. 00b - 8 bytes 01b - 16 bytes 10b - 32 bytes 11b - 64 bytes
15 TDCEN	Transceiver Delay Compensation Enable Enables the TDC feature. It can be written in Freeze mode only. See Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1:2015) for details. TDC must be disabled when Loopback mode is enabled (see CTRL1[LPB]). **NOTE** If CTRL2[BTE] = 1, this field is read as 0 and a write operation has no effect. 0b - Disable 1b - Enable
14 TDCFAIL	Transceiver Delay Compensation Fail Indicates whether the Transceiver Delay Compensation (TDC) mechanism is out of range. In this case, the mechanism cannot compensate for the loop delay of the transceiver and successfully compare the delayed received bits to the transmitted ones (see Transceiver delay compensation). The first time that FlexCAN detects the out-of-range condition, this field becomes 1. **NOTE** If CTRL2[BTE] = 1, this field is read as 0 and a write operation has no effect. 0b - In range 1b - Out of range
13 —	Reserved
12-8 TDCOFF	Transceiver Delay Compensation Offset Contains the offset value to be added to the loop delay of the measured transceiver. This value defines the position of the delayed comparison point when bit rate switching is active. See Transceiver delay compensation for details about loop delay measurement. This field can be written in Freeze mode only. Its value can be defined in Protocol Engine Clock periods (CANCLK, see Protocol timing for more details). The value must be smaller than the CAN bit duration in the data bit rate for proper operation. **NOTE** If CTRL2[BTE] = 1, TDCOFF is read as 0 and a write operation has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Do not write 0 to this field.
7-6 —	Reserved
5-0 TDCVAL	Transceiver Delay Compensation Value Contains the value of the transceiver loop delay measured from the transmitted EDL-to-R0 transition edge to the respective received one added to FDCTRL[TDCOFF]. This value is an integer multiple of the Protocol Engine Clock period (CANCLK). If CTRL2[BTE] = 1, this field is read as 0. See Protocol timing for details on the loop delay measurement.

59.6.2.36 CAN FD Bit Timing (FDCBT)

Offset

Register	Offset
FDCBT	C04h

Function

Stores the CAN bit timing variables used in the data phase of CAN FD messages when the FDCTRL[FDRATE] = 1, compatible with the CAN FD specification. Fields in this register define:

- The time quantum duration
- The number of time quanta per CAN bit
- The sample point position for the data bit rate portion of a CAN FD message with BRS = 1

Soft reset does not affect the contents of this register.

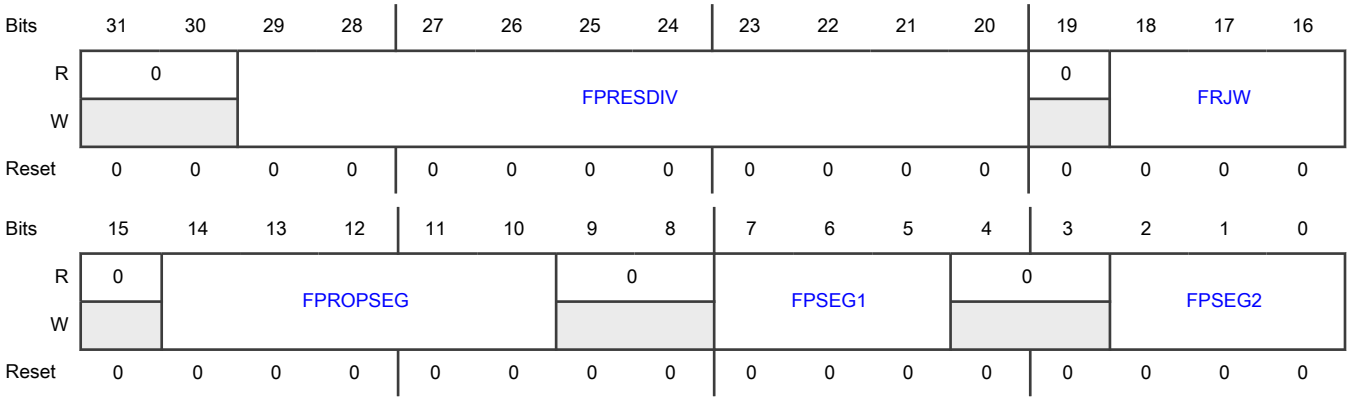
The sum of the Fast Propagation Segment (FPROPSEG) and Fast Phase Segment 1 (FPSEG1) must be at least two time quanta.

Ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1:2015).

NOTE

If CTRL2[BTE] = 1, this register is read as zero and a write operation has no effect.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-20 FPRESDIV	Fast Prescaler Division Factor Defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency in the data bit rate portion of a CAN FD message with BRS = 1. The Sclock period defines the time quantum of the CAN FD protocol for the data bit rate. This field can be written only in Freeze mode; the module blocks it in other modes. Sclock frequency = PE clock frequency ÷ (FPRESDIV + 1). **NOTE** To minimize errors when processing FD frames, use the same value for this field and for CTRL1[PRESDIV] or CBT[EPRESDIV]. See the first note in CAN FD frames for details.
19 —	Reserved
18-16 FRJW	Fast Resync Jump Width Defines the maximum number of time quanta that one resynchronization can change a bit time in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes. Resync Jump Width = FSJW + 1. One Time Quantum = one Sclock period.
15 —	Reserved
14-10 FPROPSEG	Fast Propagation Segment

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Defines the length of the propagation segment in the bit time in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes. Propagation Segment Time = FPROPSEG × Time Quanta. One Time Quantum = one Sclock period.
9-8 —	Reserved
7-5 FPSEG1	Fast Phase Segment 1 Defines the length of phase segment 1 in the bit time in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Segment 1 = (FPSEG1 + 1) × Time Quanta. One Time Quantum = one Sclock period.
4-3 —	Reserved
2-0 FPSEG2	Fast Phase Segment 2 Defines the length of phase segment 2 in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Segment 2 = (FPSEG2 + 1) × Time Quanta. One Time Quantum = one Sclock period.

59.6.2.37 CAN FD CRC (FDCRC)

Offset

Register	Offset
FDCRC	C08h

Function

Provides information about the cyclic redundancy check (CRC) of transmitted messages.

FlexCAN uses different CRC polynomials for different frame formats.

- The CRC_15 polynomial is used for all frames in CAN format.
- The CRC_17 polynomial is used for frames in CAN FD format with a DATA FIELD up to 16 bytes.
- The CRC_21 polynomial is used for frames in CAN FD format with a DATA FIELD longer than 16 bytes.

Each polynomial shown below results in a Hamming distance of 6. This register is updated at the same time that the TX Interrupt flag is set.

$$CRC\_15 = C599h: \ (x^{15} + x^{14} + x^{10} + x^8 + x^7 + x^4 + x^3 + 1)$$

$$CRC\_17 = 3685Bh: \ (x^{17} + x^{16} + x^{14} + x^{13} + x^{11} + x^6 + x^4 + x^3 + x^1 + 1)$$

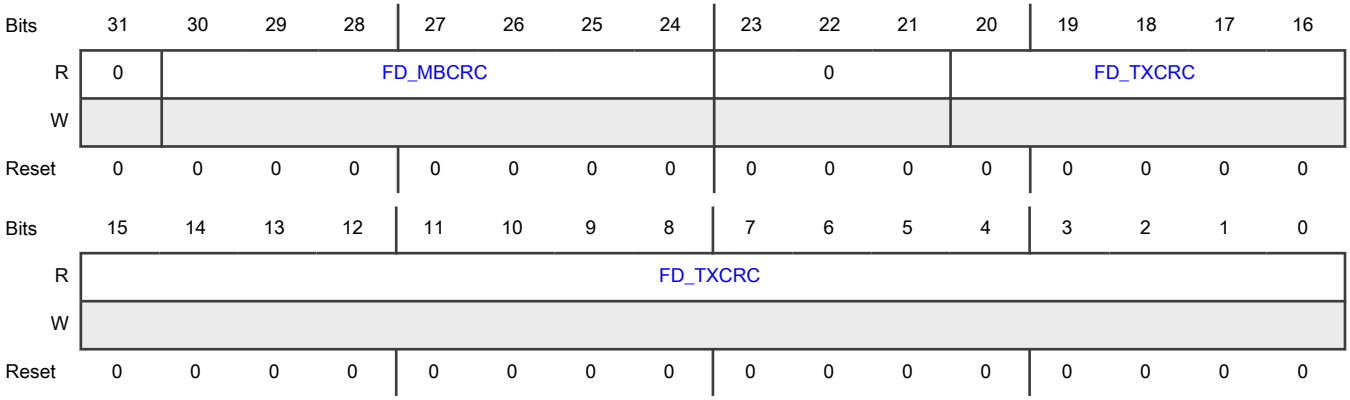
$$CRC\_21 = 302899h: \ (x^{21} + x^{20} + x^{13} + x^{11} + x^7 + x^4 + x^3 + 1)$$

Equation 33. CRC polynomial used on CAN frame

NOTE

See CRC sequence calculation in the CAN Protocol standard (ISO 11898-1:2015) for details.

Diagram



Fields

Field	Function
31 —	Reserved
30-24 FD_MBCRC	CRC Message Buffer Number for FD_TXCRC Indicates the number of the message buffer corresponding to the value in FDCRC[FD_TXCRC], for both FD and non-FD frames. It reports the same information as in CRCR[MBCRC].
23-21 —	Reserved
20-0 FD_TXCRC	Extended Transmitted CRC value Contains the CRC value calculated over the most recent transmitted message. Different CRC polynomials are used for different frame formats. For CRC_15 and CRC_17, the six most significant bits and the four most significant bits are reported as zeroes, respectively. For CRC_15, this field has the same content as Cyclic Redundancy Check (CRCR).

59.6.2.38 Enhanced RX FIFO Control (ERFCR)

Offset

Register	Offset
ERFCR	C0Ch

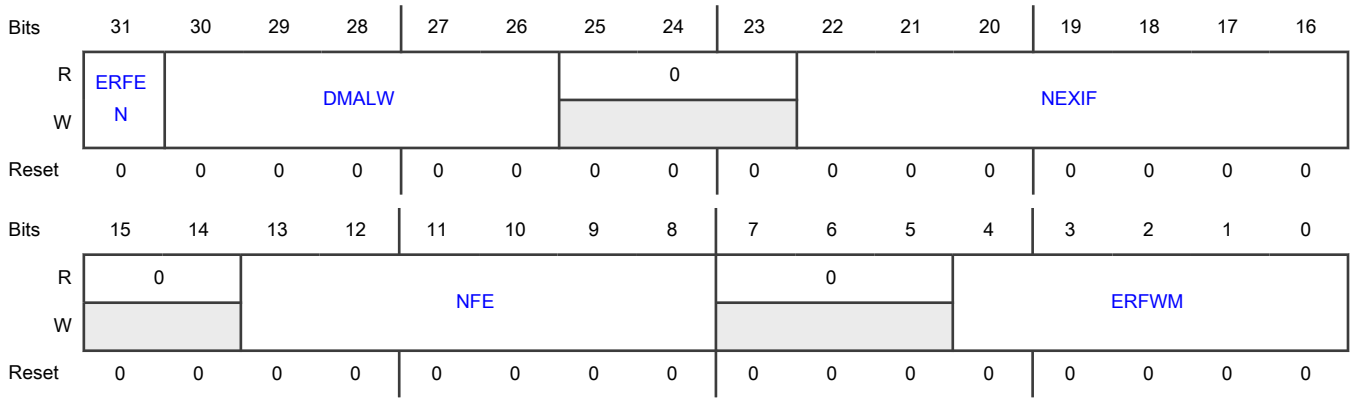
Function

Defines the Enhanced RX FIFO configuration.

This register can be written only in Freeze mode.

Soft reset does not affect any of the contents of this register.

Diagram



Fields

Field	Function
31 ERFEN	Enhanced RX FIFO enable Enables the Enhanced RX FIFO. **NOTE** If MCR[RFEN] = 1, do not write 1 to this field. 0b - Disable 1b - Enable
30-26 DMALW	DMA Last Word Defines the last DMA address for each Enhanced RX FIFO element. This table shows the number of elements and the last address for each Enhanced RX FIFO element according to the value of DMALW.

Table continued from the previous page...

Field	Function
	<table><tr><th>DMALW</th><th>Number of 32-bit words transferred</th><th>Last FIFO address</th></tr><tr><td>0</td><td>1</td><td>2000h</td></tr><tr><td>1</td><td>2</td><td>2004h</td></tr><tr><td>2</td><td>3</td><td>2008h</td></tr><tr><td>3</td><td>4</td><td>200Ch</td></tr><tr><td>4</td><td>5</td><td>2010h</td></tr><tr><td>5</td><td>6</td><td>2014h</td></tr><tr><td>6</td><td>7</td><td>2018h</td></tr><tr><td>7</td><td>8</td><td>201Ch</td></tr><tr><td>8</td><td>9</td><td>2020h</td></tr><tr><td>9</td><td>10</td><td>2024h</td></tr><tr><td>10</td><td>11</td><td>2028h</td></tr><tr><td>11</td><td>12</td><td>202Ch</td></tr><tr><td>12</td><td>13</td><td>2030h</td></tr><tr><td>13</td><td>14</td><td>2034h</td></tr><tr><td>14</td><td>15</td><td>2038h</td></tr><tr><td>15</td><td>16</td><td>203Ch</td></tr><tr><td>16</td><td>17</td><td>2040h</td></tr><tr><td>17</td><td>18</td><td>2044h</td></tr><tr><td>18</td><td>19</td><td>2048h</td></tr><tr><td>19</td><td>20</td><td>204Ch</td></tr></table> **NOTE** Undefined DMALW values in the table are reserved and must not be used.
25-23 —	Reserved
22-16 NEXIF	Number of Extended ID Filter Elements Defines the number of extended ID filter elements used during the Enhanced RX FIFO matching process. The value of this field must be less than or equal to NFE + 1. The number of standard ID filter elements is 2 × (NFE - NEXIF + 1).

Field	Function
	This table shows the number of extended ID filters and standard ID filters available for Enhanced RX FIFO if all filter elements are used.

NEXIF	NFE	Number of Extended ID filter elements	Number of Standard ID filter elements
0	63	0	128
1	63	1	126
2	63	2	124
3	63	3	122
4	63	4	120
5	63	5	118
6	63	6	116
7	63	7	114
8	63	8	112
9	63	9	110
10	63	10	108
11	63	11	106
12	63	12	104
13	63	13	102
14	63	14	100
15	63	15	98
16	63	16	96
17	63	17	94
18	63	18	92
19	63	19	90
20	63	20	88
21	63	21	86
22	63	22	84
23	63	23	82
24	63	24	80
25	63	25	78
26	63	26	76

Table continued from the previous page...

Field	Function			
	NEXIF	NFE	Number of Extended ID filter elements	Number of Standard ID filter elements
	27	63	27	74
	28	63	28	72
	29	63	29	70
	30	63	30	68
	31	63	31	66
	32	63	32	64
	33	63	33	62
	34	63	34	60
	35	63	35	58
	36	63	36	56
	37	63	37	54
	38	63	38	52
	39	63	39	50
	40	63	40	48
	41	63	41	46
	42	63	42	44
	43	63	43	42
	44	63	44	40
	45	63	45	38
	46	63	46	36
	47	63	47	34
	48	63	48	32
	49	63	49	30
	50	63	50	28
	51	63	51	26
	52	63	52	24
	53	63	53	22
	54	63	54	20
	55	63	55	18

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>NEXIF</th><th>NFE</th><th>Number of Extended ID filter elements</th><th>Number of Standard ID filter elements</th></tr>
<tr><td>56</td><td>63</td><td>56</td><td>16</td></tr>
<tr><td>57</td><td>63</td><td>57</td><td>14</td></tr>
<tr><td>58</td><td>63</td><td>58</td><td>12</td></tr>
<tr><td>59</td><td>63</td><td>59</td><td>10</td></tr>
<tr><td>60</td><td>63</td><td>60</td><td>8</td></tr>
<tr><td>61</td><td>63</td><td>61</td><td>6</td></tr>
<tr><td>62</td><td>63</td><td>62</td><td>4</td></tr>
<tr><td>63</td><td>63</td><td>63</td><td>2</td></tr>
<tr><td>64</td><td>63</td><td>64</td><td>0</td></tr>
</table>
</td></tr>
<tr><td>15-14
—</td><td>Reserved</td></tr>
<tr><td>13-8
NFE</td><td>Number of Enhanced RX FIFO Filter Elements
Defines the total number of filter elements used during the enhanced RX FIFO matching process according to the table.
<table>
<tr><th>NFE</th><th>Maximum number of extended ID filter elements (NEXIF = NFE + 1)</th><th>Maximum number of standard ID filter elements (NEXIF = 0)</th></tr>
<tr><td>0</td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>2</td><td>4</td></tr>
<tr><td>2</td><td>3</td><td>6</td></tr>
<tr><td>3</td><td>4</td><td>8</td></tr>
<tr><td>4</td><td>5</td><td>10</td></tr>
<tr><td>5</td><td>6</td><td>12</td></tr>
<tr><td>6</td><td>7</td><td>14</td></tr>
<tr><td>7</td><td>8</td><td>16</td></tr>
<tr><td>8</td><td>9</td><td>18</td></tr>
<tr><td>9</td><td>10</td><td>20</td></tr>
<tr><td>10</td><td>11</td><td>22</td></tr>
<tr><td>11</td><td>12</td><td>24</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function		
	NFE	Maximum number of extended ID filter elements (NEXIF = NFE + 1)	Maximum number of standard ID filter elements (NEXIF = 0)
	12	13	26
	13	14	28
	14	15	30
	15	16	32
	16	17	34
	17	18	36
	18	19	38
	19	20	40
	20	21	42
	21	22	44
	22	23	46
	23	24	48
	24	25	50
	25	26	52
	26	27	54
	27	28	56
	28	29	58
	29	30	60
	30	31	62
	31	32	64
	32	33	66
	33	34	68
	34	35	70
	35	36	72
	36	37	74
	37	38	76
	38	39	78
	39	40	80
	40	41	82

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>NFE</th><th>Maximum number of extended ID filter elements (NEXIF = NFE + 1)</th><th>Maximum number of standard ID filter elements (NEXIF = 0)</th></tr><tr><td>41</td><td>42</td><td>84</td></tr><tr><td>42</td><td>43</td><td>86</td></tr><tr><td>43</td><td>44</td><td>88</td></tr><tr><td>44</td><td>45</td><td>90</td></tr><tr><td>45</td><td>46</td><td>92</td></tr><tr><td>46</td><td>47</td><td>94</td></tr><tr><td>47</td><td>48</td><td>96</td></tr><tr><td>48</td><td>49</td><td>98</td></tr><tr><td>49</td><td>50</td><td>100</td></tr><tr><td>50</td><td>51</td><td>102</td></tr><tr><td>51</td><td>52</td><td>104</td></tr><tr><td>52</td><td>53</td><td>106</td></tr><tr><td>53</td><td>54</td><td>108</td></tr><tr><td>54</td><td>55</td><td>110</td></tr><tr><td>55</td><td>56</td><td>112</td></tr><tr><td>56</td><td>57</td><td>114</td></tr><tr><td>57</td><td>58</td><td>116</td></tr><tr><td>58</td><td>59</td><td>118</td></tr><tr><td>59</td><td>60</td><td>120</td></tr><tr><td>60</td><td>61</td><td>122</td></tr><tr><td>61</td><td>62</td><td>124</td></tr><tr><td>62</td><td>63</td><td>126</td></tr><tr><td>63</td><td>64</td><td>128</td></tr></table>
7-5 — 	Reserved
4-0 ERFWM	Enhanced RX FIFO Watermark Defines the minimum number of CAN messages stored in the Enhanced RX FIFO. When that number is reached, ERFSR[ERFWMI] becomes 1. Minimum number of CAN messages = ERFWM + 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If MCR[DMA] = 1, write 0h to this field.

59.6.2.39 Enhanced RX FIFO Interrupt Enable (ERFIER)

Offset

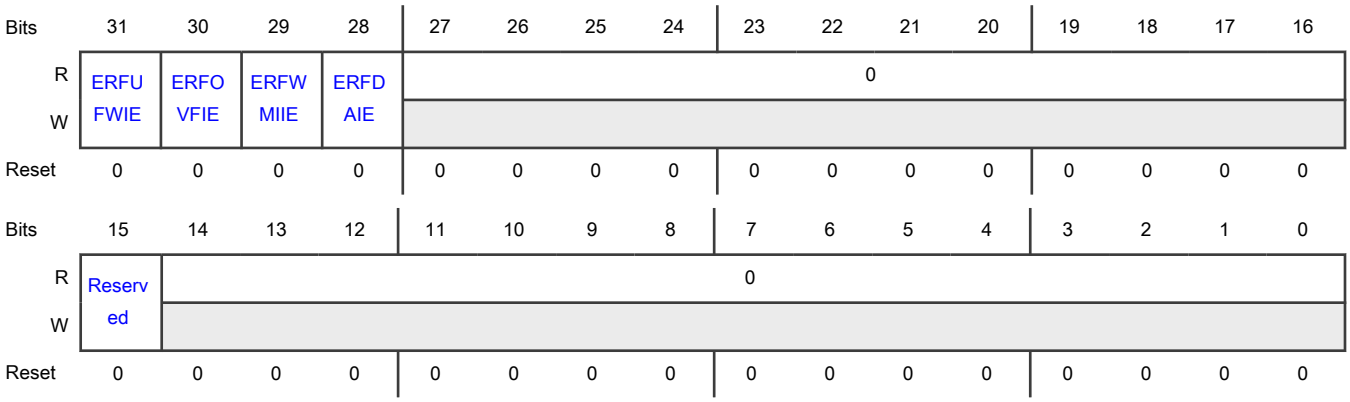
Register	Offset
ERFIER	C10h

Function

Contains the interrupt enables for the Enhanced RX FIFO.

Soft reset does not affect this register.

Diagram



Fields

Field	Function
31 ERFUFWIE	Enhanced RX FIFO Underflow Interrupt Enable Enables interrupt for ERFSR[ERFUFW]. 0b - Disable 1b - Enable
30 ERFOVFIE	Enhanced RX FIFO Overflow Interrupt Enable Enables interrupt for ERFSR[ERFOVF].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
29 ERFWMIIE	Enhanced RX FIFO Watermark Indication Interrupt Enable Enables interrupt for ERFSR[ERFWMI]. 0b - Disable 1b - Enable
28 ERFDAIE	Enhanced RX FIFO Data Available Interrupt Enable Enables interrupt for ERFSR[ERFDA]. 0b - Disable 1b - Enable
27-16 —	Reserved
15 —	Reserved
14-0 —	Reserved

59.6.2.40 Enhanced RX FIFO Status (ERFSR)

Offset

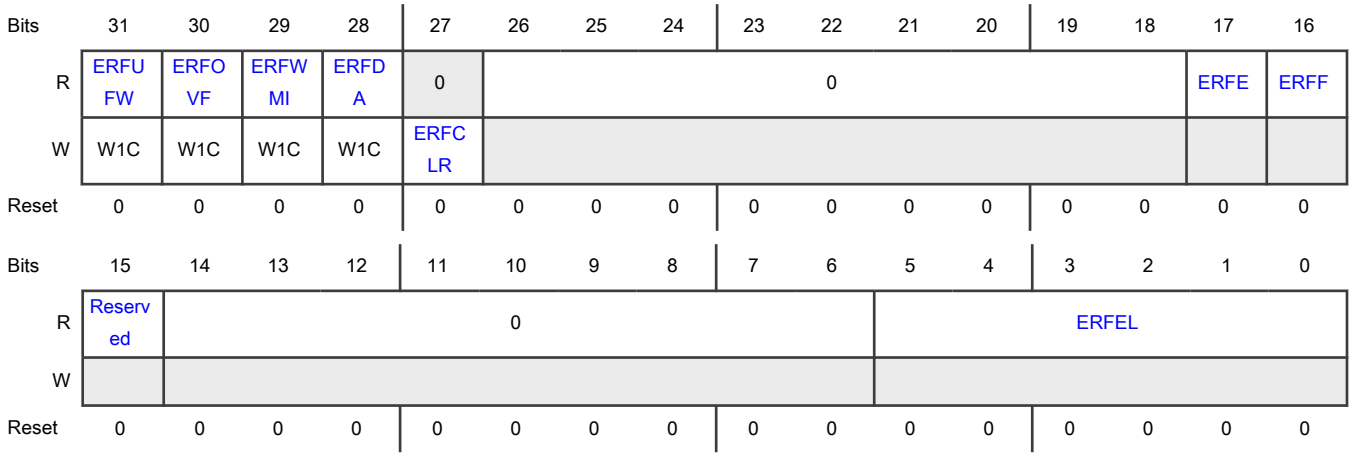
Register	Offset
ERFSR	C14h

Function

Contains the status fields of the Enhanced RX FIFO including error indications and a clear FIFO field.

Soft reset does not affect this register.

Diagram



Fields

Field	Function
31 ERFUFW	Enhanced RX FIFO Underflow Flag Indicates whether an underflow condition occurred in the enhanced RX FIFO. If ERFIER[ERFUFWIE] = 1, this field generates an interrupt. 0b - No such occurrence 1b - Underflow
30 ERFOVF	Enhanced RX FIFO Overflow Flag Indicates whether an overflow condition occurred in the Enhanced RX FIFO. If ERFIER[ERFOVFIE] = 1, this field generates an interrupt. 0b - No such occurrence 1b - Overflow
29 ERFWMI	Enhanced RX FIFO Watermark Indication Flag Indicates whether the number of messages available in the Enhanced RX FIFO is greater than the watermark defined in ERFCR[ERFWM]. If ERFIER[ERFWMIIE] = 1, this field generates an interrupt. 0b - No such occurrence 1b - Number of messages in FIFO is greater than the watermark
28 ERFDA	Enhanced RX FIFO Data Available Flag Indicates whether there is at least one message stored in the ERX FIFO. If ERFIER[ERFDAIE] = 1, this field generates an interrupt. 0b - No such occurrence 1b - At least one message stored in Enhanced RX FIFO

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 ERFCLR	Enhanced RX FIFO Clear Writing one to this field during Freeze mode clears Enhanced RX FIFO content. Writing to this field outside Freeze mode, or writing 0 to this field, has no effect. 0b - No effect 1b - Clear enhanced RX FIFO content
26-18 —	Reserved
17 ERFE	Enhanced RX FIFO Empty Flag Indicates whether Enhanced RX FIFO is empty. 0b - Not empty 1b - Empty
16 ERFF	Enhanced RX FIFO Full Flag Indicates whether enhanced RX FIFO is full. 0b - Not full 1b - Full
15 —	Reserved
14-6 —	Reserved
5-0 ERFEL	Enhanced RX FIFO Elements Indicates the number of CAN messages stored in the Enhanced RX FIFO.

59.6.2.41 High-Resolution Timestamp (HR_TIME_STAMP0 - HR_TIME_STAMP95)

Offset

For n = 0 to 95:

Register	Offset
HR_TIME_STAMPn	C30h + (n × 4h)

Function

Stores a copy of a 32-bit timer at the start or end of a CAN frame.

HR_TIME_STAMP0 stores the 32-bit timestamp associated with MB0, HR_TIME_STAMP1 stores the 32-bit timestamp associated with MB1, and so on.

Reset does not affect these registers.

NOTE

Do not write to these registers outside Freeze mode.

Table 962. High-Resolution Timestamp register operation

TSTAMPCAP	Captured timebase	Capture point
b00	None	None
b01	32 bits of high-resolution on-chip timer	Seventh bit of the end-of-frame field for transmission and sixth bit of the end-of-frame field for reception.
b10	32 bits of high-resolution on-chip timer	Start of frame
b11	32 bits of high-resolution on-chip timer	Start of frame for classical CAN frame format and res bit for CAN FD frame format

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TS															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TS															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 TS	High-Resolution Timestamp Captures the copy of the timestamp of corresponding message buffer. This field always captures a 32-bit timer value.

59.6.2.42 Enhanced RX FIFO Filter Element (ERFFEL0 - ERFFEL127)

Offset

For n = 0 to 127:

Register	Offset
ERFFELn	3000h + (n × 4h)

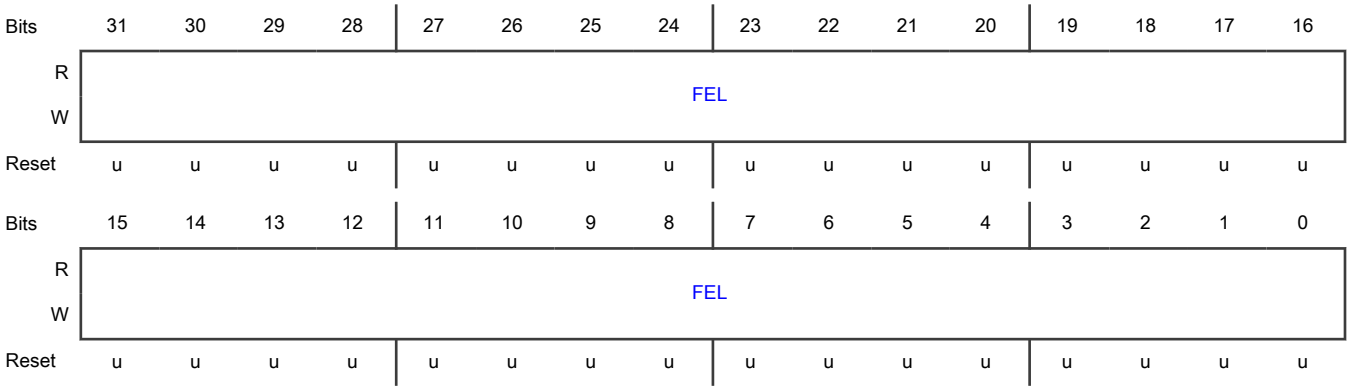
Function

Stores the filter elements of the Enhanced RX FIFO.

For standard ID filtering, each ERFFEL register stores one filter element. For extended ID filtering, each pair of ERFFEL registers stores one filter element.

ERFFEL registers can be written only in Freeze mode; otherwise, the module blocks them. Reset does not affect these registers. They are located in RAM and must be explicitly initialized prior to any reception.

Diagram



Fields

Field	Function
31-0 FEL	Filter Element Bits Stores filter elements. Each filter element is used during the match process. If the matching criteria are met, a message is stored in the Enhanced RX FIFO.

59.6.3 Message buffer structure

The message buffer structure used by FlexCAN is represented in the following figure. Both extended (29-bit identifier) and standard (11-bit identifier) frames used in the CAN specification (Version 2.0 Part B) are represented. Each individual message buffer is 16, 24, 40, or 72 bytes, depending on the quantity of data bytes allocated for the message payload: 8, 16, 32, or 64 data bytes, respectively.

The memory area 80h–67Fh is used by the message buffers. When CAN FD is enabled, the exact address for each message buffer depends on the size of its payload. See FlexCAN memory partition for CAN FD.

Table 963. Message buffer structure example with 64-byte payload

	31	30	29	28	27 24	23	22	21	20	19 18 17 16	15 8	7 0
0h	EDL	BRS	ESI		CODE		SRR	IDE	RTR	DLC	TIMESTAMP	
4h	PRIO			ID (standard/extended)							ID (extended)	
8h	Data byte 0				Data byte 1						Data byte 2	Data byte 3
Ch	Data byte 4				Data byte 5						Data byte 6	Data byte 7
10h	Data byte 8				Data byte 9						Data byte 10	Data byte 11

Table continues on the next page...

Table 963. Message buffer structure example with 64-byte payload (continued)

14h	Data byte 12	Data byte 13	Data byte 14	Data byte 15
18h	Data byte 16	Data byte 17	Data byte 18	Data byte 19
1Ch	Data byte 20	Data byte 21	Data byte 22	Data byte 23
20h	Data byte 24	Data byte 25	Data byte 26	Data byte 27
24h	Data byte 28	Data byte 29	Data byte 30	Data byte 31
28h	Data byte 32	Data byte 33	Data byte 34	Data byte 35
2Ch	Data byte 36	Data byte 37	Data byte 38	Data byte 39
30h	Data byte 40	Data byte 41	Data byte 42	Data byte 43
34h	Data byte 44	Data byte 45	Data byte 46	Data byte 47
38h	Data byte 48	Data byte 49	Data byte 50	Data byte 51
3Ch	Data byte 52	Data byte 53	Data byte 54	Data byte 55
40h	Data byte 56	Data byte 57	Data byte 58	Data byte 59
44h	Data byte 60	Data byte 61	Data byte 62	Data byte 63

		= Unimplemented or reserved

Table 964. Field descriptions

Mnemonic	Field	Description
EDL	Extended Data Length	Distinguishes between CAN format and CAN FD format frames. EDL must not be 1 for message buffers configured to RANSWER with code field 1010b (see Table 965).
BRS	Bit Rate Switch	Defines whether the bit rate is switched inside a CAN FD format frame.
ESI	Error State Indicator	Indicates whether the transmitting node is error-active or error-passive.
CODE	Message Buffer Code	Can be accessed (read or write) by the CPU and by the FlexCAN module itself, as part of the message buffer matching and arbitration process. The encoding is shown in Table 965 and Table 966. See Functional description.

Table 965. Message buffer code for RX buffers

CODE description	RX code BEFORE RX new frame	SRV[1]	RX code AFTER successful reception[2]	RRS[3]	Comment
0000b: INACTIVE. Message buffer is not active.	INACTIVE	—	—	—	Message buffer does not participate in the matching process.
0100b: EMPTY. Message buffer is active and empty.	EMPTY	—	FULL	—	When a frame is received successfully (after Move-in), CODE is automatically updated to FULL.

Table continues on the next page...

Table 965. Message buffer code for RX buffers (continued)

CODE description	RX code BEFORE RX new frame	SRV[1]	RX code AFTER successful reception[2]	RRS[3]	Comment
0010b: FULL. Message buffer is full.	FULL	Yes	FULL	—	The act of reading the Control and Status word followed by unlocking the message buffer (SRV) does not make CODE return to EMPTY. It remains FULL. If a new frame is moved to the message buffer after the message buffer is serviced, the code remains FULL. See Matching process for matching details related to FULL code.
		No	OVERRUN	—	If the message buffer is FULL and a new frame is moved to this message buffer before the CPU services it, CODE is automatically updated to OVERRUN. See Matching process for details about overrun behavior.
0110b: OVERRUN. Message buffer is being overwritten into a full buffer.	OVERRUN	Yes	FULL	—	If CODE indicates OVERRUN and the CPU has serviced the message buffer, when a new frame is moved to the message buffer, CODE returns to FULL.
		No	OVERRUN	—	If CODE already indicates OVERRUN, and another new frame must be moved, the message buffer is overwritten again, and CODE remains OVERRUN. See Matching process for details about overrun behavior.
1010b: RANSWER[4]. A frame was configured to recognize a Remote	RANSWER	—	TANSWER (1110b)	0	A Remote Answer was configured to recognize a Remote Request

Table continues on the next page...

Table 965. Message buffer code for RX buffers (continued)

CODE description	RX code BEFORE RX new frame	SRV[1]	RX code AFTER successful reception[2]	RRS[3]	Comment
Request frame and transmit a Response frame in return.[5]					frame received. After that, a message buffer is set to transmit a response frame. CODE is automatically changed to TANSWER (1110b). See Matching process for details. If CTRL2[RRS] = 0, transmit a response frame when a remote request frame with the same ID is received.
		—	—	1	This code is ignored during matching and arbitration process. See Matching process for details.
CODE[0] = 1: BUSY. FlexCAN is updating the contents of the message buffer. The CPU must not access the message buffer.	BUSY[6]	—	FULL	—	Indicates that the message buffer is being updated. It automatically becomes 0 and does not interfere with the next CODE.
		—	OVERRUN	—	

1. SRV: Serviced message buffer. Message buffer was read and unlocked by reading TIMER or other message buffer.
2. A frame is considered a successful reception after the frame is moved to a message buffer (move-in process). See Move-in.
3. Remote Request Stored field. See Control 2 (CTRL2).
4. Code 1010b is not considered TX and a message buffer with this code should not be aborted.
5. Code 1010b must be used in message buffers configured in CAN FD format, with EDL = 1.
6. For TX message buffers, the BUSY bit should be ignored upon read, except when MCR[AEN] = 1. If this field is 1, the corresponding message buffer does not participate in the matching process.

Table 966. Message buffer code for TX buffers

CODE Description	TX Code BEFORE TX frame	MB RTR	TX Code AFTER successful transmission	Comment
1000b: INACTIVE. Message buffer is not active.	INACTIVE	—	—	Message buffer does not participate in arbitration process.
1001b: ABORT. Message buffer is aborted.	ABORT	—	—	Message buffer does not participate in arbitration process.
1100b: DATA. Message buffer is a TX data frame (MB RTR must be 0).	DATA	0	INACTIVE	Transmit data frame unconditionally once. After transmission, the message

Table continues on the next page...

Table 966. Message buffer code for TX buffers (continued)

CODE Description	TX Code BEFORE TX frame	MB RTR	TX Code AFTER successful transmission	Comment
				buffer automatically returns to the INACTIVE state.
1100b: REMOTE. Message buffer is a Transmit Remote Request frame (MB RTR must be 1).	REMOTE	1	EMPTY	Transmit remote request frame unconditionally once. After transmission, the message buffer automatically becomes an RX Empty message buffer with the same ID.
1110b: TANSWER. Message buffer is a Transmit Response frame from an incoming Remote Request frame.	TANSWER	—	RANSWER	This intermediate code is automatically written to the message buffer by the CHI as a result of a match to a Remote Request frame. The Remote Response frame is transmitted unconditionally once, then the code automatically returns to RANSWER (1010b). The CPU can also write this code with the same effect. The Remote Response frame can be a data frame or another remote request frame, depending on the value of RTR. See Matching process and Arbitration process for details.

Table 967. RX and TX message buffer field descriptions

Mnemonic	Field	Description
SRR	Substitute Remote Request	Fixed recessive bit, used only in extended format. Write 1 to SRR for transmission (TX Buffers). SRR is stored with the value received on the CAN bus for RX receiving buffers. It can be received as either recessive or dominant. If FlexCAN receives this bit as dominant, it is interpreted as an arbitration loss. 1: Recessive value is compulsory for transmission in extended format frames. 0: Dominant is not a valid value for transmission in extended format frames.
IDE	ID Extended Bit	Identifies whether the frame format is standard or extended. 1: Frame format is extended 0: Frame format is standard

Table continues on the next page...

Table 967. RX and TX message buffer field descriptions (continued)

Mnemonic	Field	Description
RTR	Remote Transmission Request	Affects the behavior of remote frames and is part of the reception filter. See Table 965, Table 966, and CTRL2[RRS]. If FlexCAN transmits this field as 1 (recessive) and receives it as 0 (dominant), it is interpreted as an arbitration loss. If this field is transmitted as 0 (dominant) and it is received as 1 (recessive), FlexCAN treats it as a bit error. If the value received matches the value transmitted, it is considered a successful bit transmission. 1: If message buffer is TX, indicates that the current message buffer may have a Remote Request frame to be transmitted. If the message buffer is RX, incoming remote request frames may be stored. 0: Indicates that the current message buffer has a Data frame to be transmitted. In an RX message buffer, it may be considered in matching processes. **NOTE** When configuring CAN FD frames, this field must be 0.
DLC	Data Length Code	Indicates the length (in bytes) of the RX or TX data, which is located in offset 8h–Fh of the message buffer space (see Table 963). In reception, this field is written by FlexCAN, copied from the DLC field of the received frame. In transmission, this field is written by the CPU and corresponds to the DLC field value of the frame to be transmitted. When RTR = 1, the frame to be transmitted is a remote frame and does not include the data field, regardless of the DLC field (see Table 969).
TIMESTAMP	Free-Running Counter Timestamp	Provides a copy of the Free-Running Timer, captured for TX and RX frames when the beginning of the Identifier field appears on the CAN bus. See Table 968 for Timestamp operation.
PRIO	Local priority	Used only when MCR[LPRIOEN] = 1, and only makes sense for transmit message buffers. These bits are not transmitted. They are appended to the regular ID to define the transmission priority. See Arbitration process.
ID	Frame Identifier	In standard frame format, only the 11 most significant bits (28 to 18) are used for frame identification in both receive and transmit cases. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification in both receive and transmit cases.
DATA BYTE 0–63	Data Field	Up to 64 bytes can be used for a data frame, depending on the size of payload selected for the message buffers. For RX frames, the data is stored as it is received from the CAN bus. DATA BYTE (*n*) is valid only if *n* is less than DLC, as shown in Table 969.

Table 968. Timestamp operation

TSTAMPCAP	MBTSBASE	TIMER_SOURCE	Captured timebase	Capture point
b00	bxx	0	CAN_TIMER incremented by CAN bit clock	Second bit of identifier field
b00	bxx	1	CAN_TIMER incremented by on-chip timer clock	Second bit of identifier field
bxx	b00	0	CAN_TIMER incremented by CAN bit clock	Second bit of identifier field
bxx	b00	1	CAN_TIMER incremented by on-chip timer clock	Second bit of identifier field
b01	b01	x	Lower 16 bits of high-resolution on-chip timer	Seventh bit of the end of frame field for transmission and sixth bit of the end of frame field for reception
b01	b10	x	Upper 16 bits of high-resolution on-chip timer	Seventh bit of the end of frame field for transmission and sixth bit of the end of frame field for reception
b10	b01	x	Lower 16 bits of high-resolution on-chip timer	Start of frame
b10	b10	x	Upper 16 bits of high-resolution on-chip timer	Start of frame
b11	b01	x	Lower 16 bits of high-resolution on-chip timer	Start of frame for classical CAN frame format and res bit for CAN FD frame format
b11	b10	x	Upper 16 bits of high-resolution on-chip timer	Start of frame for classical CAN frame format and res bit for CAN FD frame format

Table 969. DATA BYTE validity

DLC	Valid data bytes
0	None
1	DATA BYTE 0
2	DATA BYTE 0–1
3	DATA BYTE 0–2
4	DATA BYTE 0–3
5	DATA BYTE 0–4
6	DATA BYTE 0–5
7	DATA BYTE 0–6
8	DATA BYTE 0–7

Table continues on the next page...

Table 969. DATA BYTE validity (continued)

DLC	Valid data bytes
9	DATA BYTE 0–11
10	DATA BYTE 0–15
11	DATA BYTE 0–19
12	DATA BYTE 0–23
13	DATA BYTE 0–31
14	DATA BYTE 0–47
15	DATA BYTE 0–63

59.6.4 FlexCAN memory partition for CAN FD

When CAN FD is enabled, FlexCAN RAM can be partitioned into blocks of 512 bytes each. Each block can accommodate a number of message buffers depending on the configuration provided by FDCTRL[MBDSR*n*] as shown in Table 970.

Table 970. RAM partition

RAM block	Number of MBs with 8 bytes (default range)	Size control field in FDCTRL	Number of MBs of different sizes, per block
0	0 to 31	MBDSR0	MBDSR0 = 00, 32 MBs with 8-byte payload MBDSR0 = 01, 21 MBs with 16-byte payload MBDSR0 = 10, 12 MBs with 32-byte payload MBDSR0 = 11, 7 MBs with 64-byte payload
1	32 to 63	MBDSR1	MBDSR1 = 00, 32 MBs with 8-byte payload MBDSR1 = 01, 21 MBs with 16-byte payload MBDSR1 = 10, 12 MBs with 32-byte payload MBDSR1 = 11, 7 MBs with 64-byte payload
2	64 to 95	MBDSR2	MBDSR2 = 00, 32 MBs with 8-byte payload MBDSR2 = 01, 21 MBs with 16-byte payload MBDSR2 = 10, 12 MBs with 32-byte payload MBDSR2 = 11, 7 MBs with 64-byte payload

Payload sizes of 16, 32, or 64 bytes may be configured in some or all of RAM blocks. In those cases, the total number of MBs and their respective number order may differ from the default configuration of 8 bytes. Consider an example where:

- Block0 is configured to an 8-byte payload
- Block1 is configured to a 16-byte payload
- Block2 is configured to 32-byte payload

In this case, Table 971 indicates how the message buffers are arranged in RAM.

Table 971. RAM partition example

RAM block	Payload size	Number of MBs in the RAM block	Message buffer range
0	FDCTRL[MBDSR0] = 00, 8-byte payload	32	0 to 31
1	FDCTRL[MBDSR1] = 01, 16-byte payload	21	32 to 52
2	FDCTRL[MBDSR2] = 10, 32-byte payload	12	53 to 64

59.6.5 FlexCAN message buffer memory map

The FlexCAN memory buffers are allocated in memory according to the tables below.

Table 972. 8-byte message buffers

Address offset (hex)	MBDSR = b00 8-byte payload
0080	MB0
0090	MB1
00A0	MB2
00B0	MB3
00C0	MB4
00D0	MB5
00E0	MB6
00F0	MB7
0100	MB8
0110	MB9
0120	MB10
0130	MB11
0140	MB12
0150	MB13
0160	MB14
0170	MB15
0180	MB16
0190	MB17
01A0	MB18
01B0	MB19
01C0	MB20

Table continues on the next page...

Table 972. 8-byte message buffers (continued)

Address offset (hex)	MBDSR = b00 8-byte payload
01D0	MB21
01E0	MB22
01F0	MB23
0200	MB24
0210	MB25
0220	MB26
0230	MB27
0240	MB28
0250	MB29
0260	MB30
0270	MB31
0280	MB32
0290	MB33
02A0	MB34
02B0	MB35
02C0	MB36
02D0	MB37
02E0	MB38
02F0	MB39
0300	MB40
0310	MB41
0320	MB42
0330	MB43
0340	MB44
0350	MB45
0360	MB46
0370	MB47
0380	MB48
0390	MB49
03A0	MB50
03B0	MB51

Table continues on the next page...

Table 972. 8-byte message buffers (continued)

Address offset (hex)	MBDSR = b00 8-byte payload
03C0	MB52
03D0	MB53
03E0	MB54
03F0	MB55
0400	MB56
0410	MB57
0420	MB58
0430	MB59
0440	MB60
0450	MB61
0460	MB62
0470	MB63
0480	MB64
0490	MB65
04A0	MB66
04B0	MB67
04C0	MB68
04D0	MB69
04E0	MB70
04F0	MB71
0500	MB72
0510	MB73
0520	MB74
0530	MB75
0540	MB76
0550	MB77
0560	MB78
0570	MB79
0580	MB80
0590	MB81
05A0	MB82

Table continues on the next page...

Table 972. 8-byte message buffers (continued)

Address offset (hex)	MBDSR = b00 8-byte payload
05B0	MB83
05C0	MB84
05D0	MB85
05E0	MB86
05F0	MB87
0600	MB88
0610	MB89
0620	MB90
0630	MB91
0640	MB92
0650	MB93
0660	MB94
0670	MB95

Table 973. 16-byte message buffers

Address offset (hex)	MBDSR = b01 16-byte payload
0080	MB0
0098	MB1
00B0	MB2
00C8	MB3
00E0	MB4
00F8	MB5
0110	MB6
0128	MB7
0140	MB8
0158	MB9
0170	MB10
0188	MB11
01A0	MB12
01B8	MB13

Table continues on the next page...

Table 973. 16-byte message buffers (continued)

Address offset (hex)	MBDSR = b01 16-byte payload
01D0	MB14
01E8	MB15
0200	MB16
0218	MB17
0230	MB18
0248	MB19
0260	MB20
0280	MB21
0298	MB22
02B0	MB23
02C8	MB24
02E0	MB25
02F8	MB26
0310	MB27
0328	MB28
0340	MB29
0358	MB30
0370	MB31
0388	MB32
03A0	MB33
03B8	MB34
03D0	MB35
03E8	MB36
0400	MB37
0418	MB38
0430	MB39
0448	MB40
0460	MB41
0480	MB42
0498	MB43
04B0	MB44

Table continues on the next page...

Table 973. 16-byte message buffers (continued)

Address offset (hex)	MBDSR = b01 16-byte payload
04C8	MB45
04E0	MB46
04F8	MB47
0510	MB48
0528	MB49
0540	MB50
0558	MB51
0570	MB52
0588	MB53
05A0	MB54
05B8	MB55
05D0	MB56
05E8	MB57
0600	MB58
0618	MB59
0630	MB60
0648	MB61
0660	MB62

Table 974. 32-byte message buffers

Address offset (hex)	MBDSR = b10 32-byte payload
0080	MB0
00A8	MB1
00D0	MB2
00F8	MB3
0120	MB4
0148	MB5
0170	MB6
0198	MB7
01C0	MB8

Table continues on the next page...

Table 974. 32-byte message buffers (continued)

Address offset (hex)	MBDSR = b10 32-byte payload
01E8	MB9
0210	MB10
0238	MB11
0280	MB12
02A8	MB13
02D0	MB14
02F8	MB15
0320	MB16
0348	MB17
0370	MB18
0398	MB19
03C0	MB20
03E8	MB21
0410	MB22
0438	MB23
0480	MB24
04A8	MB25
04D0	MB26
04F8	MB27
0520	MB28
0548	MB29
0570	MB30
0598	MB31
05C0	MB32
05E8	MB33
0610	MB34
0638	MB35

Table 975. 64-byte message buffers

Address offset (hex)	MBDSR = b11 64-byte payload
0080	MB0
00C8	MB1
0110	MB2
0158	MB3
01A0	MB4
01E8	MB5
0230	MB6
0280	MB7
02C8	MB8
0310	MB9
0358	MB10
03A0	MB11
03E8	MB12
0430	MB13
0480	MB14
04C8	MB15
0510	MB16
0558	MB17
05A0	MB18
05E8	MB19
0630	MB20

59.6.6 Legacy RX FIFO structure

When MCR[RFEN] = 1, the memory area 80h–DCh (which is normally occupied by MBs 0–5) is used by the reception Legacy FIFO engine.

The region 80h–8Ch contains the output of the Legacy RX FIFO, which the CPU must read as a message buffer. This output contains the oldest message that has been received but not yet read. The region 90h–DCh is reserved for internal use of the Legacy RX FIFO engine.

An additional memory area, which starts at E0h and may extend up to 2DCh (normally occupied by MBs 6–37) depending on the value of CTRL2[RFFN], contains the ID filter table (configurable from 8 to 128 table elements) that specifies filtering criteria for accepting frames into the Legacy RX FIFO.

Out of reset, the ID filter table flexible memory area defaults to E0h and extends only to FCh, which corresponds to MBs 6 to 7 for RFFN = 0, for backward compatibility with previous versions of FlexCAN.

The following shows the Legacy RX FIFO data structure.

Table 976. Legacy RX FIFO structure

	31 28	24 23	22	21	20	19 18 17 16	15 8	7 0
80h	IDHIT		SRR	IDE	RTR	DLC	TIMESTAMP	
84h		ID standard				ID extended		
88h	Data byte 0		Data byte 1				Data byte 2	Data byte 3
8Ch	Data byte 4		Data byte 5				Data byte 6	Data byte 7
90h–DCh	Reserved							
E0h	ID filter table element 0							
E4h	ID filter table element 1							
E8h–2D4h	ID filter table elements 2 to 125							
2D8h	ID filter table element 126							
2DCh	ID filter table element 127							
		= Unimplemented or reserved						

Each ID filter table element occupies an entire 32-bit word. One, two, or four Identifier Acceptance Filters (IDAF) can compound each element, depending on MCR[IDAM]. The following tables show the IDAF indexing.

Table 977 shows the three different formats of the ID table elements. All elements of the table must have the same format. See Legacy RX FIFO for more information.

Table 977. ID table structure

Format	31	30	29 24	23 16	15	14	13 8	7 1	0
A	RTR	IDE	RXIDA (standard = 29–19, extended = 29–1)						
B	RTR	IDE	RXIDB_0 (standard = 29–19, extended = 296–)		RTR	IDE	RXIDB_1 (standard = 13–3, extended = 13–0)		
C	RXIDC_0 (std and ext = 31–24)		RXIDC_1 (std and ext = 23–16)		RXIDC_2 (std and ext = 15–8)		RXIDC_3 (std and ext = 7–0)		
		= Unimplemented or Reserved							

Table 978. Field descriptions

Mnemonic	Field	Description
RTR	Remote Frame	Specifies whether remote frames are accepted into the Legacy FIFO if they match the target ID. 1: Remote frames can be accepted and data frames are rejected.

Table continues on the next page...

Table 978. Field descriptions (continued)

Mnemonic	Field	Description
		0: Remote frames are rejected and data frames can be accepted.
IDE	Extended Frame	Specifies whether extended or standard frames are accepted into the Legacy FIFO if they match the target ID. 1: Extended frames can be accepted and standard frames are rejected. 0: Extended frames are rejected and standard frames can be accepted.
RXIDA	RX Frame Identifier (Format A)	Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In the standard frame format, only the 11 most significant bits (29 to 19) are used for frame identification. In the extended frame format, all bits are used.
RXIDB_0, RXIDB_1	RX Frame Identifier (Format B)	Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In the standard frame format, the 11 most significant bits (a full standard ID) (29 to 19 and 13 to 3) are used for frame identification. In the extended frame format, all 14 bits of the field are compared to the 14 most significant bits of the received ID.
RXIDC_0, RXIDC_1, RXIDC_2, RXIDC_3	RX Frame Identifier (Format C)	Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In both standard and extended frame formats, all 8 bits of the field are compared to the 8 most significant bits of the received ID.
IDHIT	Identifier Acceptance Filter Hit Indicator	Indicates which identifier acceptance filter the received message in the output of the Legacy RX FIFO hit. See Legacy RX FIFO for more information.

59.6.7 Enhanced RX FIFO structure

When ERFCR[ERFEN] = 1, the Enhanced RX FIFO is enabled. The region 2000h–204Ch contains the output of the Enhanced RX FIFO, which the CPU must read as a message buffer. This output contains the oldest message that has been received but not yet read.

Table 979. Enhanced RX FIFO structure

	31	30	29	28 … 24	23	22	21	20	19	18	17	16	15 … 8	7	6 … 0
2000h	EDL	BRS	ESI	Reserved		SRR	IDE	RTR	DLC				TIMESTAMP LEGACY		
2004h	Reserved			ID (standard/extended)									ID (extended)		
2008h	Data byte 0					Data byte 1							Data byte 2		Data byte 3
200Ch	Data byte 4					Data byte 5							Data byte 6		Data byte 7
2010h	Data byte 8					Data byte 9							Data byte 10		Data byte 11
2014h	Data byte 12					Data byte 13							Data byte 14		Data byte 15
2018h	Data byte 16					Data byte 17							Data byte 18		Data byte 19
201Ch	Data byte 20					Data byte 21							Data byte 22		Data byte 23
2020h	Data byte 24					Data byte 25							Data byte 26		Data byte 27
2024h	Data byte 28					Data byte 29							Data byte 30		Data byte 31
2028h	Data byte 32					Data byte 33							Data byte 34		Data byte 35

Table continues on the next page...

Table 979. Enhanced RX FIFO structure (continued)

202Ch	Data byte 36	Data byte 37	Data byte 38	Data byte 39
2030h	Data byte 40	Data byte 41	Data byte 42	Data byte 43
2034h	Data byte 44	Data byte 45	Data byte 46	Data byte 47
2038h	Data byte 48	Data byte 49	Data byte 50	Data byte 51
203Ch	Data byte 52	Data byte 53	Data byte 54	Data byte 55
2040h	Data byte 56	Data byte 57	Data byte 58	Data byte 59
2044h	Data byte 60	Data byte 61	Data byte 62	Data byte 63
IH_OFF	Reserved			ID HIT
TS_OFF	HR TIMESTAMP			
2050h	19 Enhanced FIFO Elements (Reserved)			
...				
263Ch				

NOTE

ID HIT offset and high-resolution timestamp offset change dynamically according to data length code (DLC) as shown in Table 980.

Table 980. ID HIT offset and high-resolution timestamp offset

Data Length Code (DLC)	ID HIT offset (IH_OFF)	High-resolution timestamp offset (TS_OFF)
0	2008h	200Ch
1–4	200Ch	2010h
5–8	2010h	2014h
9	2014h	2018h
10	2018h	201Ch
11	201Ch	2020h
12	2020h	2024h
13	2028h	202Ch
14	2038h	203Ch
15	2048h	204Ch

Table 981. Field descriptions

Mnemonic	Field	Description
EDL	Extended Data Length	Distinguishes between classical CAN format and CAN FD format frames. 0: Classical CAN frame format 1: CAN FD frame format

Table continues on the next page...

Table 981. Field descriptions (continued)

Mnemonic	Field	Description
BRS	Bit Rate Switch	Defines whether the bit rate is switched inside a CAN FD format frame. 0: Bit rate is not switched in a CAN FD frame. 1: Bit rate is switched in a CAN FD frame.
ESI	Error State Indicator	Indicates whether the transmitting node is error-active or error-passive. This field is meaningful only if EDL = 1. 0: Error-active 1: Error-passive
SRR	Substitute Remote Request	Fixed recessive bit, used only in extended format. Transmitting nodes always send it as recessive and receiving nodes can receive it as either recessive or dominant. If FlexCAN receives this bit as dominant, it is interpreted as an arbitration loss.
IDE	ID Extended Bit	Identifies whether the frame format is standard or extended. 0: Standard 1: Extended
RTR	Remote Frame	Identifies whether the current frame is a data frame or a remote request. 0: Data frame 1: Remote request
DLC	Data Length Code	Defines the number of bytes in the data field of a CAN frame (Data byte 0 to Data byte 63). When RTR = 1, the frame is a remote request and does not include the data field, regardless of the DLC field. See Table 969 for more details.
LEGACY TIMESTAMP	16-bit Timestamp	Provides a copy of the Free-Running Timer, captured during the CAN frame. See Table 968 for details about legacy timestamp operation.
ID	Frame Identifier	In base frame format, only the 11 most significant bits are used for frame identification. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification.
DATA BYTE 0–63	Data Field	Up to 64 bytes can be stored in the data field.
IDHIT	Identifier Acceptance Filter Hit Indicator	Indicates which Enhanced RX FIFO Filter Element (ERFFEL0 - ERFFEL127) the received message in the output of the Enhanced RX FIFO hit. For each filter region, standard-ID filter space, and extended-ID filter space, there is an independent index starting from zero. Table 982 shows how FlexCAN writes IDHIT according to each filter element.
HR TIMESTAMP	High-resolution Timestamp	32-bit timebase captured during the CAN frame. When CTRL2[TSTAMPCAP] is not zero, a 32-bit timebase is captured from a dedicated on-chip timer which operates in free-running mode. See CTRL2[TSTAMPCAP] for details about capture point configuration of the high-resolution timestamp.

Table 982. IDHIT for Enhanced RX FIFO

Enhanced RX FIFO filter element - ERFFEL	IDHIT value	Filter element type
ERFFEL0	0	Extended-ID
ERFFEL1	1	Extended-ID
. . .	. . .	Extended-ID
ERFFEL(m-1)	m-1	Extended-ID
ERFFEL(m)	0	Standard-ID
ERFFEL(m+1)	1	Standard-ID
. . .	. . .	Standard-ID
ERFFEL(2n-m+1)	2x(n-m)+1	Standard-ID

NOTE

Where m = NEXIF and n = NFE. If NEXIF = 0, only standard-ID filter elements exist. If NEXIF > NFE, only extended-ID filter elements exist.

Chapter 60
Flexible I/O (FlexIO)

60.1 Chip-specific FlexIO Information

Table 983. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

NOTE

The FlexIO output triggers are not connected.

60.2 Overview

FLEXIO is a highly configurable module providing a wide range of functionality, including:

- Emulation of various serial or parallel communication protocols
- Flexible 16-bit timers with support for various trigger, reset, enable, and disable conditions
- Programmable logic blocks which allow the implementation of digital logic functions on-chip and configurable interaction of internal and external modules
- Programmable state machine for offloading basic system control functions from the CPU

60.2.1 Block diagram

The following diagram provides a high-level overview of the FLEXIO timers and shifters configuration.

FLEXIO uses shifters, timers, and external triggers to shift data into or out of the FLEXIO. As shown in the block diagram, timers control the timing of this data shift. You can configure the timers to use generic timer functions, external triggers, or various other conditions to determine the control.

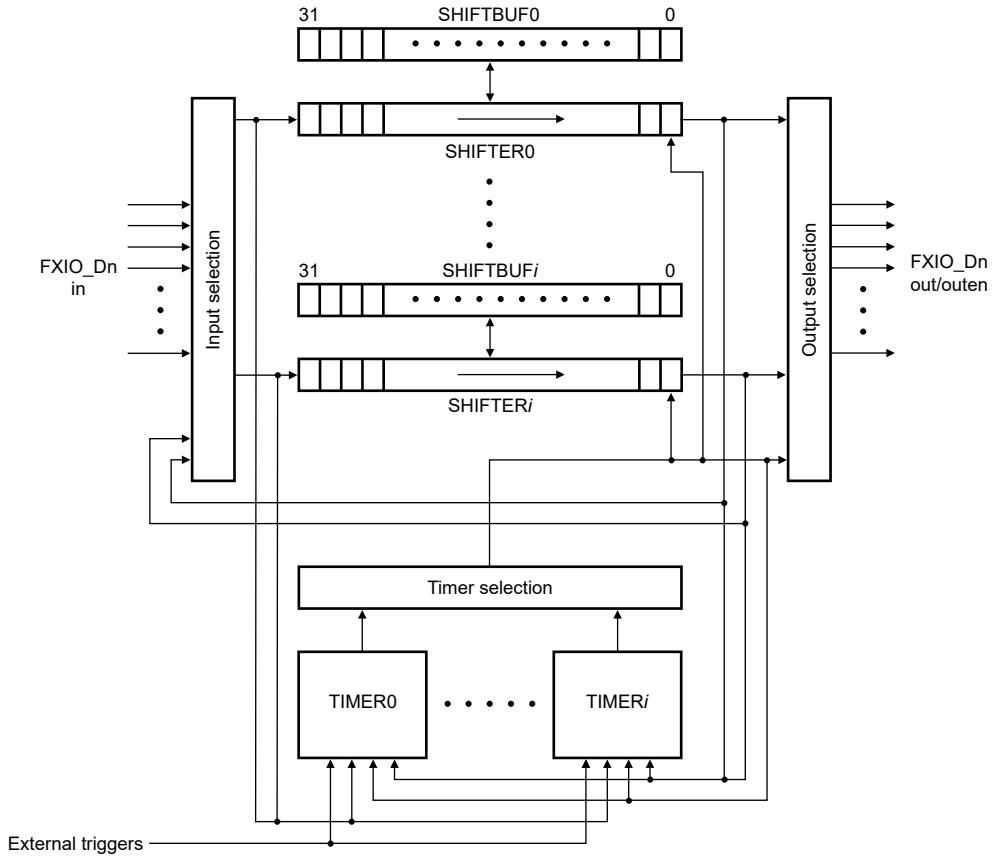


Figure 452. FLEXIO block diagram

60.2.2 Features

FLEXIO includes the following features:

- Array of 32-bit shift registers with transmit, receive, data match, logic, and state modes
  - Double-buffered shifter operation for continuous data transfer
  - Shifter concatenation to support large transfer sizes
  - Automatic start and stop bit generation
  - 1, 2, 4, 8, 16, or 32 multi-bit shift widths for parallel interface support
  - Interrupt, DMA, or polled transmit and receive operation
- Highly flexible 16-bit timers with support for various internal or external trigger, reset, enable, and disable conditions
  - Programmable baud rates independent of bus clock frequency, with support for asynchronous operation during Stop mode
  - Programmable logic mode for integrating external digital logic functions on-chip, or combining pin, shifter, or timer functions to generate complex outputs
  - Programmable state machine for offloading basic system control functions from CPU, with support for up to eight states, eight outputs, and three selectable inputs per state
- Integrated general-purpose input/output registers and pin rising or falling edge interrupts to simplify software support
- Support for a wide range of protocols, including but not limited to:

— UART

— I2C

— SPI

— I2S

— Camera IF

— Motorola 68K or Intel 8080 bus

— PWM or waveform generation

— Input-capture (pulse edge interval measurement), such as SENT

60.3 Functional description

60.3.1 Shifter operation

Shifters are responsible for buffering and shifting data into or out of the FLEXIO. The timing of shift, load, and store events is controlled by the timer assigned to the shifter via the SHIFTCTL[TIMSEL] register. Shifters are designed to support either DMA, interrupt, or polled operation. The following diagram provides a detailed view of the shifter microarchitecture.

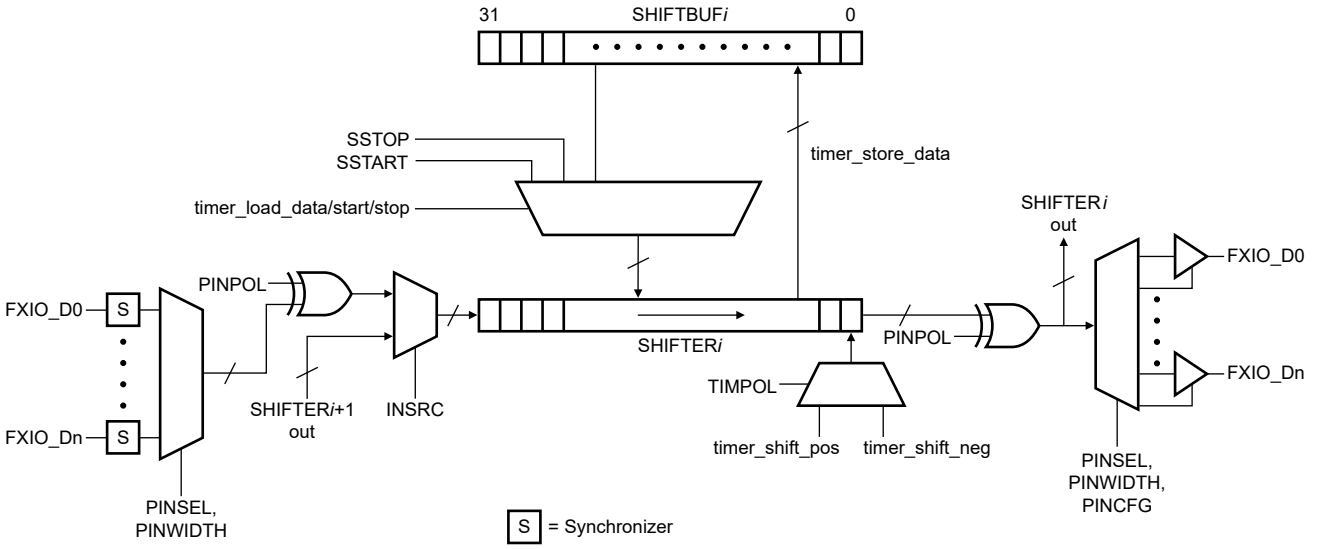


Figure 453. Shifter microarchitecture

60.3.1.1 Transmit mode

In Transmit mode (SHIFTCTL[SMOD]=Transmit), the shifter loads data from the SHIFTBUF register and shifts data out when a load event is signaled by the assigned timer. An optional start and stop bit can be automatically loaded before or after SHIFTBUF data by configuring either the SHIFTCFG[SSTART] and TIMCFG[TSTART], or SHIFTCFG[SSTOP] and TIMCFG[TSTOP] registers in the shifter and timer.

NOTE

If a stop bit is enabled, the shifter immediately loads a stop bit when it is initially configured for Transmit mode.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data has either been loaded from the SHIFTBUF register into the shifter or when the shifter is initially configured for Transmit mode. To clear the flag, write a logic 1 or write new data to the SHIFTBUF register. In Transmit mode, write any value to the SHIFTBUF register to clear the corresponding Shifter Status Flag. The flag is cleared regardless of what is writing or the state of the DMA or interrupt enables. See the SSF register description for information on how the flag is set and cleared for each mode.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to load data from an empty SHIFTBUF register occurs (buffer underrun). Clear the flag by writing logic 1.

60.3.1.2 Receive mode

When a store event is signaled by the assigned timer in Receive mode (SHIFTCTL[SMOD]=Receive), the shifter shifts data in and stores data in the SHIFTBUF register. You can check for a start and stop bit before or after the shifter data is sampled by configuring either SHIFTCFG[SSTART] and TIMCFG[TSTART] or SHIFTCFG[SSTOP] and TIMCFG[TSTOP] registers in the shifter and timer.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data has been stored in the SHIFTBUF register from the shifter. To clear the flag, write a logic 1 or read the data from the SHIFTBUF register. Any read of the SHIFTBUF register clears the corresponding shifter status flag when the shifter is in Receive mode. The flag is cleared regardless of what is reading or the state of the DMA or interrupt enables. See the SSF register description for information on how the flag is set or cleared for each mode.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set either when an attempt to store data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start or stop bit check. Write logic 1 to clear the flag.

60.3.1.3 Match Store mode

In Match Store mode (SHIFTCTL[SMOD]=Match Store), the shifter shifts data in, checks for a match result, and stores matched data into the SHIFTBUF register when a store event is signaled by the assigned timer. By configuring either the SHIFTCFG[SSTART], TIMCFG[TSTART], and SHIFTCFG[SSTOP], or TIMCFG[TSTOP] registers in the shifter and timer, you can check for a start and stop bit before or after the shifter data is sampled. You can compare up to 16 bits of data using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when a match occurs and the matched data is stored in the SHIFTBUF register from the shifter. To clear the flag, read the matched data from the SHIFTBUF register or write a logic 1 to the flag. Any read of the SHIFTBUF register clears the corresponding shifter status flag when the shifter is configured in Match Store mode. The flag is cleared regardless of what is reading or the state of the DMA or interrupt enables. See the SSF register description for information on how the flag is set or cleared for each mode.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to store matched data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start or stop bit check. Write logic 1 to clear the flag.

60.3.1.4 Match Continuous mode

In Match Continuous mode (SHIFTCTL[SMOD]=Match Continuous), the shifter shifts data in and continuously checks for a match result whenever a shift event is signaled by the assigned timer. You can compare up to 16 bits of data using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when a match occurs. The flag clears automatically as soon as there is no longer a match between the shifter data and the SHIFTBUF register. **The flag cannot be cleared by reading the SHIFTBUF register.**

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when a match occurs. To clear the flag, write logic 1 or perform a read from the SHIFTBUF register.

60.3.1.5 State mode

State mode enables you to implement any state machine with up to eight states, eight outputs, and three selectable inputs per state. This feature allows basic control functions to be offloaded from the CPU, which can remain in a low-power mode.

In State mode (SHIFTCTL[SMOD]=State) when the shifter has been selected by the current state pointer (SHIFTSTATE[STATE]), use the SHIFTBUF register to drive the output and compute the next state values. The following diagram provides a detailed view of shifter microarchitecture when configured for State mode.

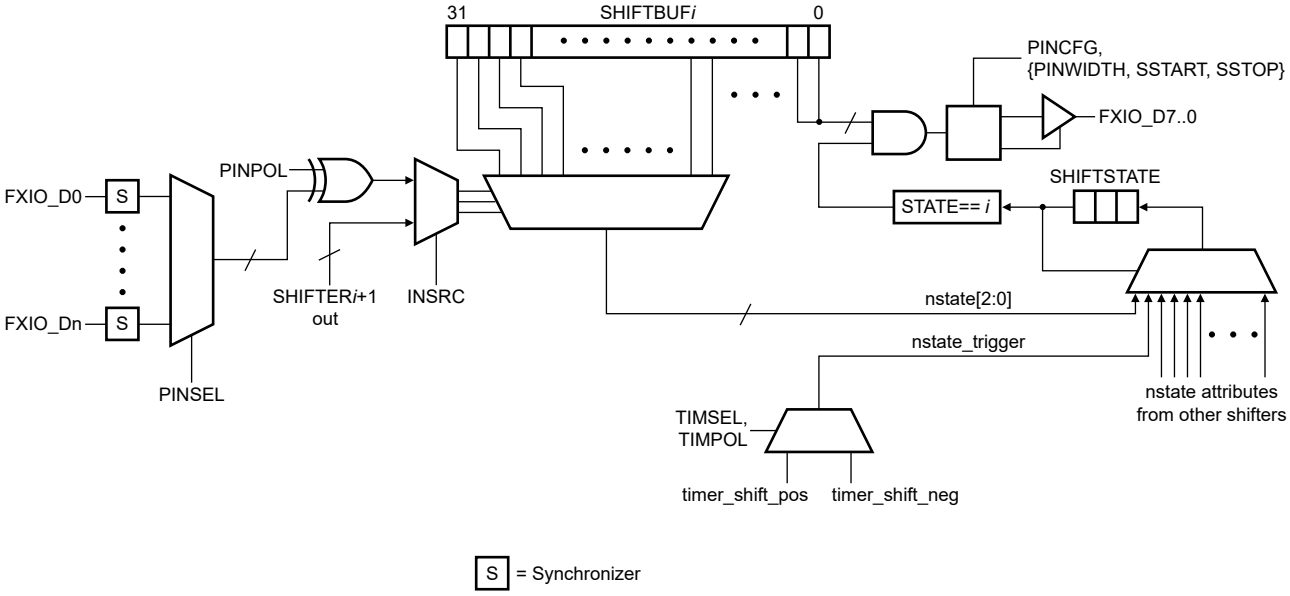


Figure 454. State microarchitecture

When a specific shifter (Shifter *n*) is selected by the current state pointer, output pins FXIO_D[7:0] are driven by SHIFTBUFn[31:24] the configuration set by SHIFTCTLi[PINCFG]. Set SHIFTCFGi{PWIDTH[3:0],SSTOP[1:0],SSTART[1:0]} to disable the output drive on pins FXIO_D[7:0] for state machine applications which require less than eight output pins.

Use the three input pins selected by SHIFTCTLi[PINSEL] and SHIFTBUFn[23:0] to compute the next state value.

NOTE

Each state can use a different set of three input pins.

The following table shows how the next state value is computed when the current state pointer is pointing to Shifter *n*.

Table 984. Next State computation for SHIFTSTATE[STATE]=*n*

FXIO_D[PINSEL+2]	FXIO_D[PINSEL+1]	FXIO_D[PINSEL]	Next State value
0	0	0	SHIFTBUFn[2:0]
0	0	1	SHIFTBUFn[5:3]
0	1	0	SHIFTBUFn[8:6]
0	1	1	SHIFTBUFn[11:9]
...	...	...	...
1	1	1	SHIFTBUFn[23:21]

NOTE

Other shifters and timers can be configured to drive the input pins of a given state, allowing you to create complex combinations of shifters and timers as needed. For example, the output of a shifter configured for Logic mode can be used to drive a state machine input.

The next state transition is triggered using the timer output selected by SHIFTCTLi[TIMSEL], with polarity controlled by SHIFTCTLi[TIMPOL].

NOTE

Each state can use a different timer to trigger each next state transition, allowing various internal or external trigger sources and clocking configurations to be used. See Timer section for more details.

The current state pointer defaults to Shifter 0 at reset; however, you can write to select a different shifter for the initial state. If the current state pointer selects a shifter which is not configured for State mode, then outputs are not driven and a next state transition is never triggered.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when the shifter is selected by the current state pointer. The flag is cleared when the current state pointer is updated to a different shifter.

60.3.1.6 Logic mode

Logic mode enables you to implement a small amount of programmable digital logic within a FLEXIO shifter.

In Logic mode (SHIFTCTL[SMOD]=Logic), use the SHIFTBUF register to implement a 5-input, 32-bit programmable logic lookup table. The following diagram provides a detailed view of shifter microarchitecture when configured for Logic mode.

Use SHIFTBUF*n* to configure the lookup table for the four pin inputs. You can also use SHIFTER*n* to configure a feedback or delayed pin source as the fifth input to the lookup table.

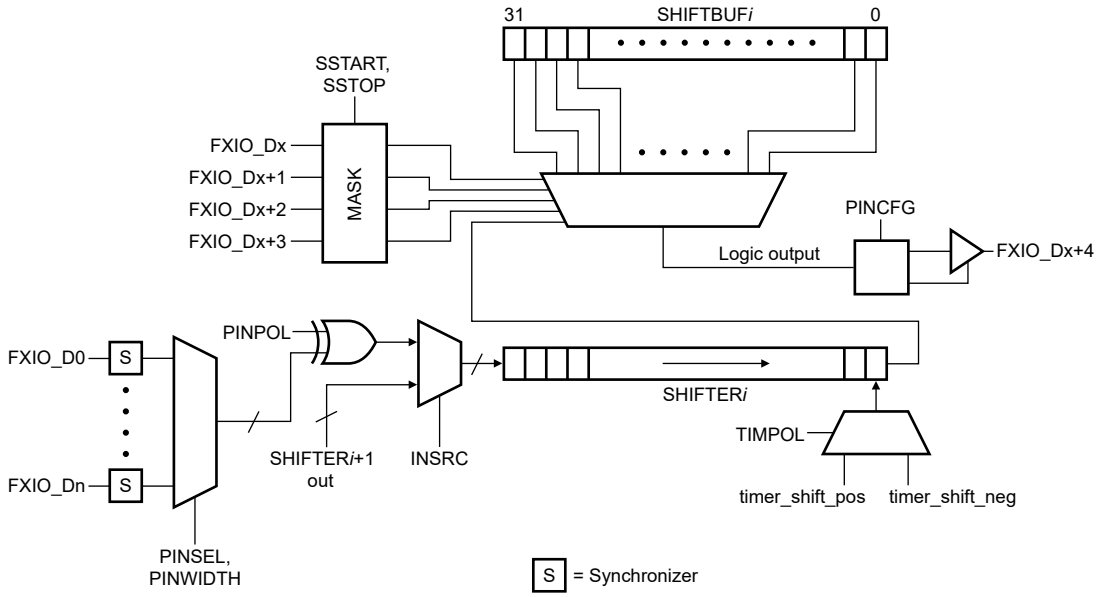


Figure 455. Logic microarchitecture

The lookup table is driven using four pin inputs (maskable using SHIFTCFG[SSTOP] and SHIFTCFG[SSTART]), plus one input from the internal shifter. It can be configured to drive an output pin using the SHIFTCTL[PINCFG] field. Pin inputs and outputs are fixed for each logic lookup table and are not selectable. The following table lists the logic output value selected by the lookup table for shifter *n*.

Table 985. Logic lookup table for Shifter *n*

SHIFTER*n*[0]	FXIO_D[x+3][1]	FXIO_D[x+2]	FXIO_D[x+1]	FXIO_D[x]	Logic Output to FXIO_D[x+4]
0	0	0	0	0	SHIFTBUF*n*[0]
0	0	0	0	1	SHIFTBUF*n*[1]

Table continues on the next page...

Table 985. Logic lookup table for Shifter *n* (continued)

0	0	0	1	0	SHIFTBUF*n*[2]
0	0	0	1	1	SHIFTBUF*n*[3]
...	...	...	...	...	...
1	1	1	1	1	SHIFTBUF*n*[31]

1. for Shifter *n*=0...3, x=*n*

   for Shifter *n*=4...7, x=*n*+4

To minimize output glitches, use SHIFTCFG[SSTOP] and SHIFTCFG[SSTART] to mask unused input pins. When set, {SSTOP[1:0] and SSTART[1:0]} mask FXIO_D[x+3]...FXIO_D[x] inputs respectively so that any transitions on these pins do not cause the logic output to glitch.

NOTE

Other shifters and timers can be configured to drive the input pins of a given lookup table, allowing you to concatenate lookup tables or create complex combinations of shifters and timers as needed.

SHIFTCFG[PWIDTH] controls the number of delay stages introduced by the internal shifter input (SHIFTERi[0]). For example, when configured for a 1-bit shift (PWIDTH=0), the internal shifter introduces a 32-shift clock delay before passing its input (selected by SHIFTCTL[PINSEL]) to the lookup table. When configured for a 32-bit shift (PWIDTH=16...31), the internal shifter introduces a 1-shift clock delay to its input.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set whenever the output pin allocated to the logic lookup table has a value of 1 after being synchronized to the FLEXIO clock. The flag clears when the output pin has a value of 0. This also allows the SSF flag to be used as a trigger to a timer if needed.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when the output pin allocated to the logic lookup table is asserted. Clear the flag by writing it with logic 1.

The Logic mode input pins, including pins driven by other shifters and timers, are synchronized to the FLEXIO functional clock before they are input to the programmable logic lookup table.

60.3.2 Timer operation

The FLEXIO 16-bit timers control the loading, shifting, and storing of the shift registers. The counters load the contents of the compare register and decrement down to zero on the FLEXIO clock. The counters can perform generic timer functions such as generating a clock or select output or a PWM waveform. Timers can be configured to:

- Enable in response to a trigger, pin, or shifter condition,
- Decrement always or only on a trigger or pin edge,
- Reset in response to a trigger or pin condition, or
- Disable on a trigger or pin condition or on a timer compare.

Timers can optionally include a start condition and a stop condition.

Although each timer operates independently, a timer can be configured to enable or disable at the same time as the previous timer (for example, Timer 1 can enable or disable at the same time as Timer 0) and a timer output can be used to trigger any other timer. The trigger used by each timer is configured independently and can be configured to be a timer output, shifter status flag, pin input, or an external trigger input. The trigger configuration is separate from the pin configuration which can be configured for input, output data, or output enable. See the chip-specific FLEXIO information for details on the external trigger connections.

The Timer Configuration Register (TIMCFGn) should be configured before setting the Timer Mode (TIMOD).

60.3.2.1 Timer 8-bit Baud Counter mode

In 8-bit Baud Counter mode, the 16-bit counter is divided into two 8-bit counters. The lower 8 bits are used to configure the baud rate of the shift clock and the upper 8 bits are used to configure the number of shift clock edges in the transfer. When the lower 8 bits decrement to zero, the timer output is toggled and the lower 8 bits reload from the compare register. The upper 8 bits decrement when the lower 8 bits equal zero and decrement.

NOTE

A timer reset event in 8-bit Baud Counter mode only resets the lower 8-bit counter. The upper 8-bit counter is not affected and can decrement if the timer reset is configured to update the state of the timer output. The timer output toggles as a result of the timer reset event.

A timer compare event occurs when the upper 8 bits equal zero and decrement. The timer status flag is set on a timer compare event.

60.3.2.2 Timer 8-bit High PWM mode

In 8-bit High PWM mode, the 16-bit counter is divided into two 8-bit counters. The lower 8 bits are used to configure the timer output high period and the upper 8 bits are used to configure the timer output low period. The lower 8 bits decrement when the output is high. When the lower 8 bits equal zero and decrement, the timer output is cleared and the lower 8 bits are reloaded from the compare register. The upper 8 bits decrement when the output is low. When the upper 8 bits equal zero and decrement, the timer output is set and the upper 8 bits are reloaded from the compare register.

A timer compare event occurs when the upper 8 bits equal zero and decrements. The timer status flag is set on a timer compare event.

60.3.2.3 Timer 16-bit Counter mode

In 16-bit Counter mode, the 16-bit counter can be used to configure either the baud rate of the shift clock (for example, TIMDEC[1:0] != 10 or 11) or the number of shift clock edges in the transfer (for example, TIMDEC[1:0] = 10 or 11). When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads from the compare register.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer compare event.

60.3.2.4 Timer 16-bit Counter Disable mode

In 16-bit Counter Disable mode, the 16-bit counter can be used to configure either the baud rate of the shift clock (for example, TIMDEC[1:0] != 10 or 11) or the number of shift clock edges in the transfer (for example, TIMDEC[1:0] = 10 or 11). When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads from the compare register.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer disable event.

60.3.2.5 Timer 8-bit Word Counter mode

In 8-bit Word Counter mode, the 16-bit counter is divided into two 8-bit counters. The lower 8 bits are used to configure the number of shift clock edges in each word and the upper 8 bits are used to configure the number of words in the transfer. When the lower 8 bits decrement to zero, the timer output is toggled and the lower 8 bits reload from the compare register. The upper 8 bits only decrement when the lower 8 bits equal zero and decrement.

A timer compare event occurs when the lower 8 bits equal zero and decrement. The timer status flag is set when the upper 8 bits equal zero and decrement.

60.3.2.6 Timer 8-bit Low PWM mode

In 8-bit Low PWM mode, the 16-bit counter is divided into two 8-bit counters. The lower 8 bits are used to configure the timer output low period and the upper 8 bits are used to configure the timer output high period. The lower 8 bits decrement when the output is low. When the lower 8 bits equal zero and decrement, the timer output is set and the lower 8 bits are reloaded from the compare

register. The upper 8 bits decrement when the output is high. When the upper 8 bits equal zero and decrement, the timer output is cleared and the upper 8 bits are reloaded from the compare register.

A timer compare event occurs when the upper 8 bits equal zero and decrements. The timer status flag is set on a timer compare event.

60.3.2.7 Timer Enable and Start bit

The following events occur when TIMOD is configured for the desired mode and the condition configured by the timer enable (TIMENA) is detected. When ONETIM is set, the timer status flag must be clear to generate a timer enable event, otherwise the timer enable event is blocked. You can use this to enforce software intervention after each timer iteration.

- Timer counter loads the current value of the compare register and starts decrementing as configured by TIMDEC.
- Timer output may update to its initial state depending on the TIMOUT configuration. Shifters that are controlled by this timer do not see this as a rising edge on the timer shift clock.
- Transmit shifters controlled by this timer either output their start bit value or load the shift register from the shift buffer and output the first bit, as configured by SSTART.

If the timer start bit is enabled, the timer counter reloads with the compare register on the first rising edge of the shift clock after the timer starts decrementing. If there is no falling edge on the shift clock before the first rising edge (for example, when TIMOUT=1), a shifter that is configured to shift on falling edge and load on the first shift will not load correctly.

60.3.2.8 Timer decrement and reset

The timer generates the timer output and timer shift clock depending on the TIMOD and TIMDEC fields. The shifter clock is either equal to the timer output (when TIMDEC != 10 or 11) or equal to the decrement clock (when TIMDEC = 10 or 11). When TIMDEC is configured to decrement from a pin or trigger, the timer decrements on both rising and falling edges.

If a timer is configured to decrement on the FLEXIO functional clock divided by 16 or 256 (when TIMDEC = 100 or 101), then a common prescaler that is shared by all timers is used to generate the two divide ratios. This prescaler is reset when all timers are either idle or configured not to use the prescaler (TIMDEC != 100 or 101).

When the timer is configured to reset as determined by the TIMRST field, then the timer counter loads the current value of the compare register again. The timer output and timer shift clock can be configured to update on timer reset, as configured by TIMOUT. If the time output toggles as a result of the timer reset, this can result in a timer shift clock edge. In 8-bit Baud Counter mode this will also decrement the upper 8-bits of the counter.

In general, when the timer counter decrements to zero, a timer compare event is triggered. The timer compare event causes the timer counter to load the contents of the timer compare register, the timer output to toggle, any configured transmit shift registers to load, and any configured receive shift registers to store. Depending on the timer mode, the timer status flag may also be set.

60.3.2.9 Timer Disable and Stop bit

When the timer is configured to add a stop bit on each compare, the following additional events occur:

- Transmit shifters controlled by this timer output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SSTOP.
- The timer counter reloads the current value of the Compare Register on the first rising edge of the shifter clock after the compare.

When the timer is configured to insert a stop bit on each compare, transmit shifters must be configured to load on the first shift.

When the condition configured by timer disable (TIMDIS) is detected, the following events occur:

- Timer counter reloads the current value of the Compare Register and starts decrementing as configured by TIMDEC.
- Timer output clears. Shifters that are controlled by this timer do not see this as a falling edge on the timer shift clock, but can generate a shift event if the timer shift clock would otherwise generate one.

- Transmit shifters controlled by this timer output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SSTOP.

If the timer stop bit is enabled, the timer counter continues decrementing until the next rising edge of the shift clock is detected, at which point it finishes. Although the timer output is forced low during the stop bit, the timer shift clock can toggle during the stop bit. The timer output does not generate shift events during the stop bit.

A timer enable condition can be detected in the same cycle as a timer disable condition (if timer stop bit is disabled), or on the first rising edge of the shift clock after the disable condition (if stop bit is enabled). When ONETIM is set, the timer status flag must be clear before the next timer enable condition is detected. When the timer is in the stop state condition, receive shift registers with stop bit enabled will store the contents of the shift register into the shift buffer and verify the state of the input data on the configured shift edge. If there is no configured edge between the timer disable and the next rising edge of the shift clock, then the final store and verify do not occur.

60.3.3 Pin operation

The pin configuration for each timer and shifter can be configured to use any FLEXIO pin with either polarity. Each timer and shifter can be configured as an input, output data, output enable, or bidirectional output. A pin configured for output enable can be used as an open drain (with inverted polarity, since the output enable assertion causes logic zero to be output on the pin) or to control the enable on the bidirectional output. Any timer or shifter can be configured to control the output enable for a pin where the bidirectional output data is driven by another timer or shifter.

60.3.3.1 Parallel interface

You can configure shifters to use multiple FLEXIO pins in parallel using the SHIFTCFG[PWIDTH] field. PWIDTH is used to configure the following settings of a shifter:

1. Number of bits shifted per shift clock.
2. Number of pins driven by the shifter per shift clock. (Only on shifters supporting parallel transmit. That is, SHIFTER0 and SHIFTER4.)
3. Number of pins sampled by the shifter per shift clock. (Only on shifters supporting parallel receive. That is, SHIFTER3 and SHIFTER7.)

When configured for parallel shift, either 4, 8, 16 or 32 bits can be shifted on every shift clock. If an adjacent shifter is selected as the input source (SHIFTCFG[INSRC]=1), the least significant 4, 8, 16 or 32 bits from the adjacent shifter are sampled on each shift clock.

For shifters supporting parallel receive (SHIFTER3, SHIFTER7), you can configure the shifter to sample multiple pins (SHIFTCFG[INSRC]=0), with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL].

NOTE

If PWIDTH is less than the number of bits being shifted on each shift clock, then the most significant bits are masked with 0. For example, if PINSEL=7 and PWIDTH=6, then SHIFTER[31:24] samples {0,0,FXIO_D[12:7]} on each shift clock.

For shifters supporting parallel transmit (SHIFTER0, SHIFTER4), you can configure the shifter to drive multiple pins using SHIFTCTL[PINCFG], with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL].

NOTE

If PWIDTH is less than the number of bits being shifted on each shift clock, then the most significant pins are not driven. For example, if PINSEL=7 and PWIDTH=6, then SHIFTER[5:0] drives only FXIO_D[12:7] on each shift clock.

60.3.3.2 Pin synchronization

When you configure a pin as an input (this includes a timer trigger configured as a pin input), the input signal is first synchronized to the FLEXIO clock before the signal is used by a timer or shifter. This introduces a small latency of between 0.5 to 1.5 FLEXIO clock cycles when using an external pin input to generate an output or control a shifter. This sets the maximum setup time at 1.5 FLEXIO clock cycles.

If an input is used by more than one timer or shifter then the synchronization occurs once to ensure any edge is seen on the same cycle by all timers and shifters using that input.

NOTE

FLEXIO pins are also connected internally. Configuring a FLEXIO shifter or timer to output data on an unused pin makes an internal connection that allows other shifters and timers to use this pin as an input. This allows a shifter output to be used to trigger a timer or a timer output to be shifted into a shifter. This path is also synchronized to the FLEXIO clock and therefore incurs a 1 cycle latency.

When using a pin input as a timer trigger, timer clock, or shifter data input, the following synchronization delays occur:

1. 0.5 to 1.5 FLEXIO clock cycles for external pin
2. 1 FLEXIO clock cycle for an internally driven pin

See Application information for timing considerations such as output valid time and input setup time for specific applications (SPI Controller, SPI Target, I2C Controller, I2S Controller, and I2S Target).

60.3.3.3 Pin override

You can change the state of any FLEXIO pin at any time with software. The pin output enable register configures any pin as an output and drives that pin with the value in the Pin Output Data register.

Alias registers for the Pin Output Enable and Data registers also exist. Writing a logic one to an alias register updates the corresponding register bits in both the Pin Output Enable Register and the Pin Output Data Register as follows.

- Pin Output Disable Register clears Output Enable Register and clear Output Data Register.
- Pin Output Clear Register sets Output Enable Register and clear Output Data Register.
- Pin Output Set Register sets Output Enable Register and set Output Data Register.
- Pin Output Toggle Register sets Output Enable Register and toggle Output Data Register.

60.3.3.4 Pin interrupt

You can read the state of any FLEXIO pin at any time with software. Software can also configure any pin to set a status flag when either a rising or falling edge is detected on that pin. The pin status flag can also be configured to generate an interrupt.

60.3.4 Low power modes

The FLEXIO remains functional during low power modes, provided the Doze Enable field (CTRL[DOZEN]) is clear and the FLEXIO functional clock remains enabled.

The exception to this is in LLS mode. In this case, the Doze Enable (CTRL[DOZEN]) field is ignored and the FLEXIO waits for all timers to complete any pending operation before acknowledging LLS mode entry.

60.3.5 Debug mode

FLEXIO remains functional in Debug mode, provided the Debug Enable field (CTRL[DBGE]) is set.

60.3.6 Clocking

Table 986. FLEXIO clocks

Clock	Description
Functional clock	The FLEXIO functional clock is asynchronous to the bus clock and can remain enabled in low power modes. The FLEXIO functional clock must be enabled before accessing any FLEXIO registers. Provided the FLEXIO functional clock is at least equal to the bus clock, the CTRL[FASTACC] field can be set to support fast register accesses.
Bus clock	The bus clock is only used for bus accesses to the control and configuration registers.

60.3.7 Reset

Table 987. FLEXIO resets

Reset	Description
Chip reset	The logic and registers for FLEXIO are reset to their default state on chip reset.
Software reset	The FLEXIO implements a software reset field in its Control Register. CTRL[SWRST] resets all logic and registers to their default state, except for the CTRL itself.

60.3.8 Interrupts and DMA requests

The following table shows the status flags that generate the FLEXIO interrupt and DMA requests.

Table 988. FLEXIO interrupts and DMA requests

Flag	Description	Interrupt	DMA request	Low power wakeup
SSF	Shifter Status Flag	Y	Y	Y
SEF	Shifter Error Flag	Y	N	Y
TSF	Timer Status Flag	Y	Y	Y
PSF	Pin Status Flag	Y	N	Y
ETSF	External Trigger Status Flag	Y	N	Y

60.3.9 Peripheral triggers

The connection between the FLEXIO peripheral triggers and other peripherals is device-specific.

Output triggers

Each FLEXIO timer generates an output trigger equal to the timer output. The output trigger is not affected by the timer pin polarity configuration.

Input trigger

FLEXIO supports multiple external trigger inputs that can be used to trigger one or more FLEXIO timers. If a rising edge is detected on an external trigger when FLEXIO is enabled, then the external trigger status flag is set. The external triggers are synchronized to the FLEXIO functional clock and must assert for at least two cycles of the FLEXIO functional clock to be sampled correctly.

60.4 External signals

Table 989. External signals

Signal	Description	Direction
FXIO_D*n* (*n*=0...31)	Bidirectional FLEXIO shifter and timer pin	Input or output

60.5 Initialization

To initialize FLEXIO registers:

1. Enable FLEXIO by writing CTRL[FLEXEN]=1
2. Configure shift registers for the given application; it is recommended to write Shifter Configuration N (SHIFTCFG0 - SHIFTCFG7) before the corresponding Shifter Control N (SHIFTCTL0 - SHIFTCTL7)
3. Configure timer registers for the given application; it is recommended to write Timer Compare N (TIMCMP0 - TIMCMP7) and Timer Configuration N (TIMCFG0 - TIMCFG7) before the corresponding Timer Control N (TIMCTL0 - TIMCTL7)
4. Enable interrupts and/or DMA requests as appropriate for the given application
5. Write transmit data to initiate a transfer (depending on the given application)

60.6 Application information

This section provides examples for a variety of FLEXIO module applications. See FLEXIO register descriptions for further details.

60.6.1 UART transmit

UART transmit can be supported using one timer, one shifter, and one pin (two pins, if supporting CTS). The start and stop bit insertion is handled automatically, and multiple transfers are supported using DMA controller. The timer status flag is used to indicate when the stop bit of each word is transmitted.

Break and idle characters require software intervention. Before transmitting a break or idle character, the SSTART and SSTOP fields must be altered to transmit the required state, and the data to transmit must equal FFh or 00h. Supporting a second stop bit requires the stop bit to be inserted into the data stream using software (and increasing the number of bits to transmit). When performing byte writes to SHIFTBUF*n* (or SHIFTBUFBIS for transmitting MSB first), the rest of the register remains unaltered. This allows an address mark bit or additional stop bit to remain undisturbed.

NOTE

FLEXIO does not support automatic insertion of parity bits.

Table 990. UART transmit configuration

Register	Value	Comments
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0003_0002h	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0. Can invert output data by setting PINPOL, or can support open-drain by setting PINPOL=1h and PINCFG=1h.
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1.

Table continues on the next page...

Table 990. UART transmit configuration (continued)

Register	Value	Comments
		Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG*n*	0000_2222h	Configure start bit, stop bit, enable on trigger asserted and disable on compare. Can support CTS by configuring TIMEN=3h.
TIMCTL*n*	01C0_0001h	Configure dual 8-bit counter using Shifter 0 status flag as inverted internal trigger source. Supports CTS by configuring PINSEL=1h (for Pin 1) and PINPOL=1h.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports MSB first transfer by writing to the SHIFTBUFBBS[7:0] register instead.

The following table shows an alternative configuration that supports slower baud rates. This configuration requires two timers.

Table 991. UART transmit configuration for slow baud rate

Register	Value	Comments
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0003_0002h	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0. Invert output data by setting PINPOL. Support open-drain by setting PINPOL=1h and PINCFG=1h.
TIMCMP*n*	0000_000Fh	Configure for 8-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG*n*	0030_2622h	Configure start bit, stop bit, enable on trigger rising edge, decrement on trigger and disable on compare.
TIMCTL*n*	0740_0003h	Configure 16-bit counter using Timer 1 output as internal trigger source.
TIMCMP(*n*+1)	0000_0001h	Configure baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:0] = (baud rate divider / 2) - 1.
TIMCFG(*n*+1)	0000_1200h	Configure enable on trigger asserted and disable on Timer 0 disable. Can support CTS by configuring TIMEN=3h.

Table continues on the next page...

Table 991. UART transmit configuration for slow baud rate (continued)

Register	Value	Comments
TIMCTL(*n*+1)	01C0_0003h	Configure 16-bit counter using Shifter 0 status flag as inverted internal trigger source. Support CTS by configuring PINSEL=1h (for Pin 1) and PINPOL=1h.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Support MSB first transfer by writing to SHIFTBUFBBS[7:0] instead.

60.6.2 UART receive

UART receive can be supported using one timer, one shifter, and one pin (two timers and two pins, if supporting RTS). The start and stop bit verification is handled automatically and multiple transfers are supported using the DMA controller. The timer status flag is used to indicate when the stop bit of each word is received.

Triple voting of the received data is not supported by FLEXIO; data is sampled only once in the middle of each bit. You can use a timer to implement a glitch filter on the incoming data and a different timer to detect an idle line of programmable length. Break characters cause the error flag to set and the shifter buffer register will return 00h.

NOTE

FLEXIO does not support automatic verification of parity bits.

Table 992. UART receiver configuration

Register	Value	Comments
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0080_0001h	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Can invert input data by setting PINPOL.
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG*n*	0204_2422h	Configure start bit, stop bit, enable on pin posedge and disable on compare. Enable resynchronization to received data with TIMOUT=2h and TIMRST=4h.
TIMCTL*n*	0000_0081h	Configure dual 8-bit counter using inverted Pin 0 input.
SHIFTBUF*n*	Data to receive	Received data can be read from SHIFTBUFBYS[7:0]. Use the Shifter Status Flag to indicate when data can

Table continues on the next page...

Table 992. UART receiver configuration (continued)

Register	Value	Comments
		be read using interrupt or DMA request. Supports MSB first transfer by reading from the SHIFTBUFBIS[7:0] register instead.

The UART receiver with RTS configuration uses a second timer to generate the RTS output. The RTS asserts when the start bit is detected and negates when the data is read from the shifter buffer register. If no start bit is detected while the RTS is asserted, the received data is ignored.

Table 993. UART receiver with RTS configuration

Register	Value	Comments
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0080_0001h	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Invert input data by setting PINPOL.
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG*n*	0204_2522h	Configure start bit, stop bit, enable on pin posedge with trigger asserted and disable on compare. Enable resynchronization to received data with TIMOUT=2h and TIMRST=4h.
TIMCTL*n*	02C0_0081h	Configure dual 8-bit counter using inverted Pin 0 input. Trigger is internal using inverted Pin 1 input.
TIMCMP(*n*+1)	0000_FFFFh	Never compare.
TIMCFG(*n*+1)	0030_6100h	Enable on Timer *n* enable and disable on trigger falling edge. Decrement on trigger to ensure no compare.
TIMCTL(*n*+1)	0143_0003h	Configure 16-bit counter and output on Pin 1. Trigger is internal using Shifter 0 flag.
SHIFTBUF*n*	Data to receive	Received data can be read from SHIFTBUFBYS[7:0]. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports MSB first transfer by reading from the SHIFTBUFBIS[7:0] register instead.

60.6.3 SPI Controller

SPI Controller mode can be supported using two timers, two shifters, and four pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The stop bit is used to guarantee a minimum of one clock cycle between the target select negating and before the next transfer. To initiate each transfer, write to the transmit buffer by either core or DMA.

NOTE

Due to synchronization delays, the setup time for the serial input data is 1.5 FLEXIO clock cycles. This means the maximum baud rate is divide by 4 of the FLEXIO clock frequency.

Table 994. SPI Controller (CPHA=0) configuration

Register	Value	Comments
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0083_0002h	Configure transmit using Timer 0 on negedge of clock with output data on Pin 0.
SHIFTCFG(*n*+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n*+1)	0000_0101h	Configure receive using Timer 0 on posedge of clock with input data on Pin 1.
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG*n*	0100_2222h	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0.
TIMCTL*n*	01C3_0201h	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as the inverted trigger. Set PINPOL to invert the output shift clock.
TIMCMP(*n*+1)	0000_FFFFh	Never compare.
TIMCFG(*n*+1)	0000_1100h	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.
TIMCTL(*n*+1)	0003_0383h	Configure 16-bit counter (never compare) using inverted Pin 3 output (as target select).
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUF. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports MSB first transfer by writing to the SHIFTBUFBIS register instead.

Table continues on the next page...

Table 994. SPI Controller (CPHA=0) configuration (continued)

Register	Value	Comments
SHIFTBUF(*n*+1)	Data to receive	Received data can be read from SHIFTBUF. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports MSB first transfer by reading from the SHIFTBUFBIS register instead.

Table 995. SPI Controller (CPHA=1) configuration

Register	Value	Comments
SHIFTCFG*n*	0000_0021h	Start bit loads data on first shift.
SHIFTCTL*n*	0003_0002h	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0.
SHIFTCFG(*n*+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n*+1)	0080_0101h	Configure receive using Timer 0 on negedge of clock with input data on Pin 1.
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG*n*	0100_2222h	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0.
TIMCTL*n*	01C3_0201h	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as the inverted trigger. Set PINPOL to invert the output shift clock. Set TIMDIS=3 to keep target select asserted for as long as there is data in the transmit buffer.
TIMCMP(*n*+1)	0000_FFFFh	Never compare.
TIMCFG(*n*+1)	0000_1100h	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.
TIMCTL(*n*+1)	0003_0383h	Configure 16-bit counter (never compare) using inverted Pin 3 output (as target select).
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUF. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.

Table continues on the next page...

Table 995. SPI Controller (CPHA=1) configuration (continued)

Register	Value	Comments
		Supports MSB first transfer by writing to the SHIFTBUFBIS register instead.
SHIFTBUF(*n*+1)	Data to receive	Received data can be read from SHIFTBUF. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports MSB first transfer by reading from SHIFTBUFBIS register instead.

60.6.4 SPI Target

SPI Target mode can be supported using one timer, two shifters, and four pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The transmit data must be written to the transmit buffer register before the external target select asserts, otherwise the shifter error flag is set.

NOTE

Due to synchronization delays, the output valid time for the serial output data is 2.5 FLEXIO clock cycles. This means the maximum baud rate is divide by 6 of the FLEXIO clock frequency.

Table 996. SPI Target (CPHA=0) configuration

Register	Value	Comments
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0083_0002h	Configure transmit using Timer 0 on falling edge of shift clock with output data on Pin 0.
SHIFTCFG(*n*+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n*+1)	0000_0101h	Configure receive using Timer 0 on rising edge of shift clock with input data on Pin 1.
TIMCMP*n*	0000_003Fh	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG*n*	0120_0600h	Configure enable on trigger rising edge. Initial clock state is logic 0 and decrements on pin input.
TIMCTL*n*	06C0_0203h	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (target select) as the inverted trigger.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUF. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.

Table continues on the next page...

Table 996. SPI Target (CPHA=0) configuration (continued)

Register	Value	Comments
		Supports MSB first transfer by writing to SHIFTBUFBIS register instead.
SHIFTBUF(*n*+1)	Data to receive	Received data can be read from SHIFTBUF. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports MSB first transfer by reading from the SHIFTBUFBIS register instead.

Table 997. SPI Target (CPHA=1) configuration

Register	Value	Comments
SHIFTCFG*n*	0000_0001h	Shifter configured to load on first shift and stop bit disabled.
SHIFTCTL*n*	0003_0002h	Configure transmit using Timer 0 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(*n*+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n*+1)	0080_0101h	Configure receive using Timer 0 on falling edge of shift clock with input data on Pin 1.
TIMCMP*n*	0000_003Fh	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG*n*	0120_6602h	Configure start bit, enable on trigger rising edge, disable on trigger falling edge. Initial clock state is logic 0 and decrements on pin input.
TIMCTL*n*	06C0_0203h	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (target select) as the inverted trigger.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUF. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports MSB first transfer by writing to the SHIFTBUFBIS register instead.
SHIFTBUF(*n*+1)	Data to receive	Received data can be read from SHIFTBUF. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports MSB first transfer by reading from SHIFTBUFBIS register instead.

60.6.5 I2C Controller

I2C Controller mode can be supported using two timers, two shifters, and two pins. One timer is used to generate the SCL output and one timer is used to control the shifters. The two shifters are used to transmit and receive for every word when receiving the transmitter must transmit FFh to 3-state the output. FLEXIO inserts a stop bit after every word to generate and verify the ACK or NACK. FLEXIO waits for the first write to the transmit data buffer before enabling SCL generation. Data transfers can be supported using the DMA controller and the shifter error flag sets on transmit underrun or receive overflow.

The first timer generates the bit clock for the entire packet (START to Repeated START or STOP), so the compare register must be programmed with the total number of clock edges in the packet (minus one). The timer supports clock stretching using the reset counter when pin is equal to output. However, this increases both the clock high and clock low periods by at least 1 FLEXIO clock cycle each. The second timer uses the SCL input pin to control the transmit and receive shift registers. This enforces an SDA data hold time by an extra 2 FLEXIO clock cycles.

Both the transmit and receive shifters must be serviced for each word in the transfer. The transmit shifter must transmit FFh when receiving and the receive shifter returns the data present on the SDA pin. The transmit shifter loads one additional word on the last falling edge of SCL pin. If generating a STOP condition or a repeated START condition, this word must be 00h or FFh, respectively. During the last word of a controller-receiver transfer, you must set the transmit SSTOP bit to generate a NACK.

The receive shift register asserts an error interrupt if a NACK is detected, but you are responsible for generating the STOP or repeated START condition. If a NACK is detected during controller-transmit, the interrupt routine must immediately write 00h (if generating STOP) or FFh (if generating repeated START) to the transmit shifter register. You must wait for the next rising edge on SCL before disabling both timers. The transmit shifter should be disabled after waiting the setup delay for a repeated START or STOP condition.

NOTE

Due to synchronization delays, the data valid time for the transmit output is two FLEXIO clock cycles. This means the maximum baud rate is divide by 6 of the FLEXIO clock frequency.

To guarantee SDA hold time, the I2C controller data valid is delayed two cycles because the clock output is passed through a synchronizer before clocking the transmit or receive shifter. Because the SCL output is synchronous with FLEXIO clock, the synchronization delay is one cycle, and then one cycle to generate the output.

Table 998. I2C Controller configuration

Register	Value	Comments
SHIFTCFG*n*	0000_0032h	Start bit enabled (logic 0) and stop bit enabled (logic 1).
SHIFTCTL*n*	0101_0082h	Configure transmit using Timer 1 on rising edge of clock with inverted output enable (open-drain output) on Pin 0.
SHIFTCFG(*n*+1)	0000_0020h	Start bit disabled and stop bit enabled (logic 0) for ACK or NACK detection.
SHIFTCTL(*n*+1)	0180_0001h	Configure receive using Timer 1 on falling edge of clock with input data on Pin 0.
TIMCMP*n*	0000_2501h	Configure 2 word transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] = (number of words x 18) + 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG*n*	0102_2222h	Configure start bit, stop bit, enable on trigger high, disable on compare, reset

Table continues on the next page...

Table 998. I2C Controller configuration (continued)

Register	Value	Comments
		if output equals pin. Initial clock state is logic 0 and is not affected by reset.
TIMCTL*n*	01C1_0101h	Configure dual 8-bit counter using Pin 1 output enable (SCL open-drain), with Shifter 0 flag as the inverted trigger.
TIMCMP(*n*+1)	0000_000Fh	Configure 8-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(*n*+1)	0020_1112h	Enable when Timer 0 is enabled, disable when Timer 0 is disabled. Enable start bit and stop bit at end of each word and decrement on pin input.
TIMCTL(*n*+1)	01C0_0183h	Configure 16-bit counter using inverted Pin 1 input (SCL).
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUFBBS[7:0]. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.
SHIFTBUF(*n*+1)	Data to receive	Received data can be read from SHIFTBUFBIS[7:0]. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

60.6.6 I2S Controller

I2S Controller mode can be supported using two timers, two shifters, and four pins. One timer is used to generate the bit clock and control the shifters and one timer is used to generate the frame sync. FLEXIO waits for the first write to the transmit data buffer before enabling bit clock and frame sync generation. Data transfers are supported using the DMA controller and the shifter error flag sets on transmit underrun or receive overflow.

The bit clock frequency is an even integer divide of the FLEXIO clock frequency. The initial frame sync assertion occurs at the same time as the first bit clock edge. The timer uses the start bit to ensure that the frame sync is generated one clock cycle before the first output data.

NOTE

Due to synchronization delays, the setup time for the receiver input is 1.5 FLEXIO clock cycles. This means the maximum baud rate is divide by 4 of the FLEXIO clock frequency.

Table 999. I2S controller configuration

Register	Value	Comments
SHIFTCFG*n*	0000_0001h	Load transmit data on first shift and stop bit disabled.

Table continues on the next page...

Table 999. I2S controller configuration (continued)

Register	Value	Comments
SHIFTCTL*n*	0003_0002h	Configure transmit using Timer 0 on rising edge of clock with output data on Pin 0.
SHIFTCFG(*n*+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n*+1)	0080_0101h	Configure receive using Timer 0 on falling edge of clock with input data on Pin 1.
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG*n*	0000_0202h	Configure start bit, enable on trigger high and never disable. Initial clock state is logic 1.
TIMCTL*n*	01C3_0281h	Configure dual 8-bit counter using inverted Pin 2 output (bit clock), with Shifter 0 flag as the inverted trigger. Clear PINPOL to invert the polarity of the output shift clock.
TIMCMP(*n*+1)	0000_007Fh	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:0] = (number of bits x baud rate divider) - 1.
TIMCFG(*n*+1)	0000_0100h	Enable when Timer 0 is enabled and never disable.
TIMCTL(*n*+1)	0003_0383h	Configure 16-bit counter using inverted Pin 3 output (as frame sync). Clear PINPOL to invert the polarity of the output frame sync
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUFBIS. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports LSB first transfer by writing to SHIFTBUF register instead.
SHIFTBUF(*n*+1)	Data to receive	Received data can be read from SHIFTBUFBIS. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports LSB first transfer by reading from SHIFTBUF register instead.

60.6.7 I2S Target

I2S Target mode can be supported using three timers, two shifters, and four pins. For single transmit and single receive, other combinations of transmit and receive are possible.

The transmit data must be written to the transmit buffer register before the external frame sync asserts, otherwise the shifter error flag is set.

NOTE

Due to synchronization delays, the output valid time for the serial output data is 2.5 FLEXIO clock cycles. This means the maximum baud rate is divide by 6 of the FLEXIO clock frequency.

The output valid time of I2S Target is max 2.5 cycles because there is a maximum 1.5 cycle delay on the clock synchronization, plus 1 cycle to output the data.

Timer 2 detects the falling edge of frame sync (start of new frame) and asserts output until the rising edge of bit clock (when the frame sync is normally sampled). Timer 0 detects rising edge of bit clock with Timer 2 output asserted and asserts output for length of frame. Timer 1 detects falling edge of bit clock with Timer 0 output asserted and controls shift registers for 32-bit transfers.

Table 1000. I2S Target configuration

Register	Value	Comments
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0103_0002h	Configure transmit using Timer 1 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(*n*+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n*+1)	0180_0101h	Configure receive using Timer 1 on falling edge of shift clock with input data on Pin 1.
TIMCMP*n*	0000_007Fh	Configure two 32-bit transfers per frame. Set TIMCMP[15:0] = (number of bits x 4) - 1.
TIMCFG*n*	0020_2500h	Configure enable on pin rising edge (inverted bit clock) with trigger high (Timer 2) and disable on compare. Initial clock state is logic 1 and decrements on pin input (bit clock).
TIMCTL*n*	0B40_0203h	Configure 16-bit counter using Pin 2 input (bit clock), with Timer 2 output as the trigger.
TIMCMP(*n*+1)	0000_003Fh	Configure 32-bit transfers. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(*n*+1)	0020_2500h	Configure enable on pin (bit clock) rising edge with trigger (Timer 0) high and disable on compare. Initial clock state is logic 1 and decrement on pin input (bit clock).

Table continues on the next page...

Table 1000. I2S Target configuration (continued)

Register	Value	Comments
TIMCTL(*n*+1)	0340_0283h	Configure 16-bit counter using inverted Pin 2 input (bit clock), with Timer 0 output as the trigger.
TIMCMP(*n*+2)	0000_0000h	Compare on zero (first edge).
TIMCFG(*n*+2)	0020_6400h	Configure enable on inverted pin (frame sync) rising edge and disable on trigger falling edge (bit clock). Initial clock state is logic 1 and decrement on inverted pin input (frame sync).
TIMCTL(*n*+2)	04C0_0383h	Configure 16-bit counter using inverted Pin3 input (frame sync), with Pin 2 inverted input (bit clock) as the trigger.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUFBIS. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports LSB first transfer by writing to SHIFTBUF register instead.
SHIFTBUF(*n*+1)	Data to receive	Received data can be read from SHIFTBUFBIS. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports LSB first transfer by reading from SHIFTBUF register instead.

60.6.8 SENT receiver

SENT receiver can be supported by using one timer and one pin. The timer is configured as Input-Capture mode, and captures the counter value at falling edge of the pin input. After the counter value is captured, the counter is automatically restarted from counter value 0. Therefore, the captured value always indicates the period between previous falling edge and current falling edge. The CPU interrupt or DMA trigger can be configured at each capture of the counter. The entire SENT frame decoding with the latest tick width adjustment is performed by CPU software.

Table 1001. SENT receiver configuration

Register	Value	Comments
TIMCMPn	-	Stores counter value at the falling edge of the pin.
TIMCFGn	0x0000_6000	Timer is always enabled. Timer is disabled on trigger falling edge, decrement counter on FLEXIO clock.
TIMCTLn	0xnn40_0007	Single 16-bit input capture mode. Timer pin input and output are selected by PINSEL. Timer pin output disabled, internal trigger selected, select pin nn as trigger.

60.6.9 Camera interface

Camera interface can be supported using one timer, one or more shifters, and multiple pins. Multiple transfers can be supported using DMA controller.

The example below describes the FLEXIO configuration for interfacing to an 8-bit CMOS sensor with PCLK, VSYNC, HREF, and D[7:0] outputs. The example uses a 128-bit buffer to capture 16-pixels of image data before interrupt or DMA transfer. You can use a bigger or smaller buffer depending on system DMA performance and FLEXIO resource usage by other applications.

NOTE

You can use additional timers to track the number of pixels per row and number of rows per frame, or HREF or VSYNC can be assigned as GPIO interrupts for software tracking.

Table 1002. Camera interface configuration for 8-bit CMOS sensor

Register	Value	Comments
SHIFTCFG*n*...*n*+2[1]	0007_0100h	Configure 8-bit parallel shift in from adjacent shifter.
SHIFTCFG*n*+3	0007_0000h	Configure 8-bit parallel shift in from pins FXIO_D[7:0] (D[7:0]).
SHIFTCTL*n*...*n*+3	0080_0001h	Configure receive using Timer 0 on negedge of clock.
TIMCMP*n*	0000_001Fh	Configure 16-pixel (8 bits/pixel x 16 pixels = 128-bits) transfer. Set TIMCMP[15:0] = (number of pixels x 2) - 1.
TIMCFG*n*	0120_6600h	Configure enable on trigger (HREF) rising edge and disable on trigger falling edge. Initial Shift clock state is logic 0 and decrement on PCLK input.
TIMCTL*n*	12C0_0803h	Configure 16-bit counter using FXIO_D[8] input (PCLK), with FXIO_D[9] input (HREF) as the inverted trigger.
SHIFTBUF*n*...*n*+3	Data to receive	Received data can be read from SHIFTBUF*n*...*n*+3. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

1. *n*=0 or 4

60.6.10 Motorola 68K and Intel 8080 bus interface

The Motorola 68K and Intel 8080 bus are two parallel interfaces commonly used by Smart/Asynchronous LCD controllers. With GPIO, FLEXIO can drive these interfaces using one Timer and one Shifter. Additional Shifters can be used to support large transfers via the DMA controller.

The table below provides an example of how to drive a 16-bit 68K or 8080 bus. For an 8080 bus, two GPIOs are used to drive the nCS and RS pins. For a 68K bus, an additional GPIO is required to drive the RDWR pin.

Table 1003. Motorola 68K and Intel 8080 write configuration

Register	Value	Comments
SHIFTCFG0...0	000F_0100h	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCTL0	0003_0002h	Configure transmit using Timer 0 on posedge of clock with data output to FXIO_D[15:0].
SHIFTCTL1...0	0000_0002h	Configure transmit using Timer 0 on posedge of clock.
TIMCMP0	0000_0101h (1-beat) 0000_1F01h (16-beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] = (number of beats x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0000_2200h	Configure enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	01C3_1001h (Motorola 68K, 1-beat) 1DC3_1001h (Motorola 68K, 16-beats) 01C3_1081h (Intel 8080, 1-beat) 1DC3_1081h (Intel 8080, 16-beats)	Configure dual 8-bit counter using FXIO_D[16] output (EN pin for 68K, nWR pin for 8080), with Shifter 0 (1-beat) or Shifter 0 (16-beats) flag as the inverted trigger.
SHIFTBUF0...0	Data to transmit	Transmit data can be written to SHIFTBUF0 (1-beat) or SHIFTBUF0...0(16-beats) to initiate a transfer, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.

Table 1004. Motorola 68K 8080 read configuration

Register	Value	Comments
SHIFTCFG0...3	000F_0100h	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCFG0	000F_0000h	Configure 16-bit parallel shift in from pin.
SHIFTCTL0...0	0080_0001h	Configure receive using Timer 0 on negedge of clock with data input from FXIO_D[15:0].
TIMCMP0	0000_0101h (1-beat) 0000_1F01h (16-beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] = (number of beats x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.

Table continues on the next page...

Table 1004. Motorola 68K 8080 read configuration (continued)

Register	Value	Comments
TIMCFG0	0000_2220h	Configure stop_bit. Enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	1DC3_1001h (Motorola 68K, 1 beat) 01C3_1001h (Motorola 68K, 16 beats) 1DC3_1181h (Intel 8080, 1 beat) 01C3_1181h (Intel 8080, 16 beats)	Configure dual 8-bit counter using either FXIO_D[16] output (EN pin for 68K) or FXIO_D[17] output (nRD pin for 8080), with Shifter 0 flag (1-beat) or Shifter 0 flag (16-beats) as the inverted trigger.
SHIFTBUF0...0	Data received	Received data can be read from SHIFTBUF0 (1-beat) or SHIFTBUF0...0 (16-beats). Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

In general, any operation to a 68K or 8080 bus target begins with a register write cycle followed by one or more data read or write cycles. To accomplish this, the following program flow must be used:

1. Configure FLEXIO with 1-beat write configuration
2. Configure GPIO to assert nCS, RS pins (and deassert RDWR pin for 68K)
3. Write register index data to SHIFTBUF0[15:0]
4. Configure GPIO to deassert RS pin (and assert RDWR pin for 68K data read)
5. Configure FLEXIO with desired read or write configuration (1 or 16-beats)
6. Use the Shifter Status Flag to trigger interrupt or DMA driven data transfers to and from SHIFTBUF registers
7. Configure GPIO to deassert nCS pin

60.6.11 Low-power state machine

The table below shows an example of a hypothetical state machine in order to illustrate the flexibility allowed in Shifter State mode.

In this example, FLEXIO waits for the FXIO_D[2] pin to assert and then drives a complementary square wave output at a frequency of FLEXIO_CLK/131072 on the FXIO_D[1:0] pins while the comparator output is asserted. This assumes that the comparator is connected to external trigger 15. See the chip-specific FLEXIO information for actual FLEXIO trigger mappings. Throughout this operation, the CPU can be kept in a STOP or VLPS mode by clearing the CTRL[DOZEN] field and ensuring the FLEXIO_CLK is enabled. The state diagram below shows the states and transitions implemented by this example.

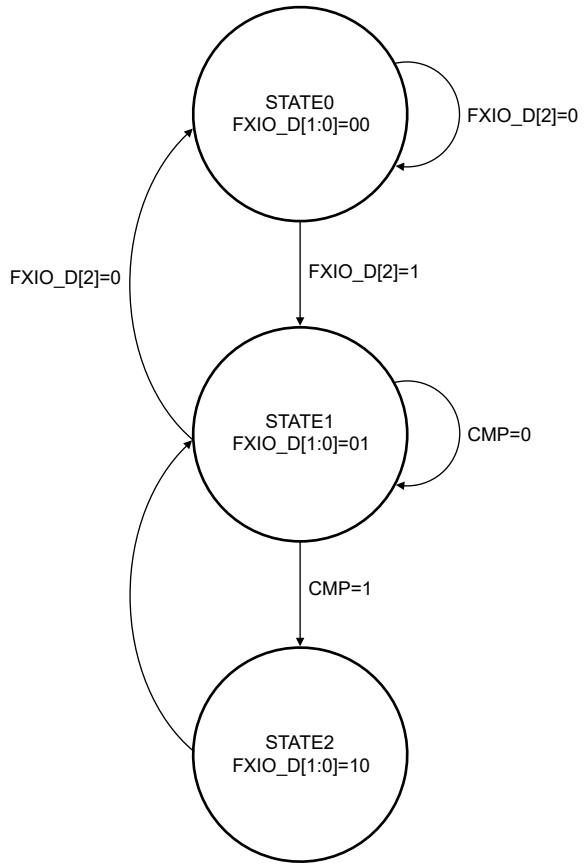


Figure 456. State diagram

Table 1005. State machine configuration

Register	Value	Comments
SHIFTCFG0...2	0000_0003h	Enable FXIO_D[1:0] as outputs.
SHIFTCTL0	0080_0206h	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer 0 output low to trigger next state.
SHIFTBUF0	0020_8208h	State0: Drive FXIO_D[1:0]=00, transition to State0 if FXIO_D[2]=0, State1 if FXIO_D[2]=1.
SHIFTCTL1	0000_0206h	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer 0 output high to trigger next state.
SHIFTBUF1	0140_8408h	State1: Drive FXIO_D[1:0]=01, transition to State0 if FXIO_D[2]=0, State1 if CMP=0, State2 if CMP=1 (FXIO_D[3]=1)

Table continues on the next page...

Table 1005. State machine configuration (continued)

Register	Value	Comments
SHIFTCTL2	0080_0206h	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer 0 output low to trigger next state.
SHIFTBUF2	0224_9249h	State2: Drive FXIO_D[1:0]=10, transition to State1 when Timer 0 output low
TIMCMP0	0000_FFFFh	Configure baud rate of divide by 131072 of the FLEXIO clock. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0000_0000h	Configure timer always enabled.
TIMCTL0	0000_0003h	Configure single 16-bit counter.
TIMCFG1	0010_7600h	Configure timer enabled on trigger rising edge, disabled on trigger falling edge, decrement on trigger edge.
TIMCTL1	0F03_0303h	Configure timer output enabled on FXIO_D[3], external trigger 15 (comparator output) selected.

60.6.12 Keypad interface

The keypad interface can support a 3 × 4 keypad matrix using three timers and three shifters, although a larger matrix can be supported using additional shifters. The configuration is designed for four columns configured as active low open-drain pins and three rows configured as input pins with pull-up resistors enabled.

One shifter is configured in logic mode to assert its output when any of the row inputs are low, indicating a key is pressed. Use a timer to filter the shifter output to ensure that the key is pressed for a minimum amount of time before performing the column scan.

A different shifter is configured for parallel transmit. Use this shifter to scan each column when a keypress is detected. When not scanning, the shifter output is configured to assert all active low open-drain column outputs to detect any keypress. Use a dedicated timer to control the transmit shifter.

The last shifter is configured for parallel receive. Use this shifter to capture the result of the column scanning for software to decode which key (or keys) was pressed. This configuration captures the state of both row and column pins for each scan, although the row state can also be deduced by the shift order. Use a dedicated timer to control the receive shifter, which shifts at half the frequency of the transmit shifter.

When the result of the key scan is available in the receive shifter register, FLEXIO continues to monitor the row inputs and can trigger multiple scans from the one keypress. To support debouncing, you can decide how many consecutive scans should be considered a single keypress.

NOTE

Because the pins used in Logic mode are fixed per shifter and the shifters that support parallel shifts are limited, this configuration is restricted in what pins and shifters it can use. Increasing the matrix beyond four-row inputs requires multiple shifters in logic mode. Increasing beyond four-column outputs requires concatenating transmit shifters to create a larger shift register.

Table 1006. Keypad interface configuration

Register	Value	Comments
SHIFTCFG0	0003_0032h	Start bit enabled (logic 0) and stop bit enabled (logic 1), configured for 4-bit shift.
SHIFTCTL0	0101_0402h	Configure transmit using Timer 1 on rising edge of clock generating open-drain output on Pins[7:4] (column outputs).
SHIFTBUF0	0804_0201h	Static data containing the column scan pattern, each column is scanned one-hot with dead time in between.
SHIFTCFG1	0000_0020h	In logic mode, mask input 3.
SHIFTCTL1	0000_0007h	Configure logic mode.
SHIFTBUF1	07FF_07FFh	Static data configuring logic mode LUT. Output asserts if pins [3:1] are logic 0.
SHIFTCFG3	0007_0000h	Configured for 8-bit shift.
SHIFTCTL3	0280_0001h	Configure receive using Timer 2 on falling edge of clock with input data on Pins [7:0] (both rows and columns, Pin 0 is don't care).
TIMCMP0	0000_00ffh	Configures pre-scanning glitch filter to 256 FLEXIO clock cycles. For different filter cycles, set TIMCMP[15:0] = (filter cycles) - 1.
TIMCFG0	0103_6600h	Configure enable on trigger rising edge and disable on trigger falling edge. Initial clock state is logic 0 and is not affected by reset.
TIMCTL0	0540_0003h	Configure 16-bit counter using Shifter 1 flag (logic state) as trigger.
TIMCMP1	0000_0F3Fh	Configure 8 shifts (twice for each column) at column scan rate of divide by 128. For different scan frequency, set TIMCMP[7:0] = (scan divider / 2) - 1.
TIMCFG1	0020_2622h	Enable on trigger rising edge, disable on compare. Start and stop bits are enabled.
TIMCTL1	0340_0001h	Configure dual 8-bit counter using Timer 0 output as the trigger.
TIMCMP2	0000_0801h	Configure 4 shifts at half the frequency of Timer 1 trigger.
TIMCFG2	0110_2100h	Enable on Timer 1 enable, disable on compare, decrement on trigger input

Table continues on the next page...

Table 1006. Keypad interface configuration (continued)

Register	Value	Comments
		with output initially negated, and not affected by reset
TIMCTL2	0740_0001h	Configure dual 8-bit counter using Timer 1 output as the trigger.
SHIFTBUF3	Keypad Scan Result	Keypad scan result can be read from SHIFTBUF3. Each byte is the result of one scan with both row and column pin state from the scan (pin 0 is not used). Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.

60.7 Memory map and registers

60.7.1 FLEXIO register descriptions

NOTE

An invalid register access results in a bus error. Invalid register accesses include reading a write-only register, writing a read-only register, or accessing an invalid address.

60.7.1.1 FLEXIO memory map

FLEXIO1 base address: 425C_0000h

FLEXIO2 base address: 425D_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0201_0003h
4h	Parameter (PARAM)	32	R	0420_0808h
8h	FLEXIO Control (CTRL)	32	RW	0000_0000h
Ch	Pin State (PIN)	32	R	0000_0000h
10h	Shifter Status (SHIFTSTAT)	32	RW	0000_0000h
14h	Shifter Error (SHIFTERR)	32	RW	0000_0000h
18h	Timer Status (TIMSTAT)	32	RW	0000_0000h
20h	Shifter Status Interrupt Enable (SHIFTSIEN)	32	RW	0000_0000h
24h	Shifter Error Interrupt Enable (SHIFTEIEN)	32	RW	0000_0000h
28h	Timer Interrupt Enable (TIMIEN)	32	RW	0000_0000h
30h	Shifter Status DMA Enable (SHIFTSDEN)	32	RW	0000_0000h
38h	Timer Status DMA Enable (TIMERSDEN)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
40h	Shifter State (SHIFTSTATE)	32	RW	0000_0000h
48h	Trigger Status (TRGSTAT)	32	RW	0000_0000h
4Ch	External Trigger Interrupt Enable (TRIGIEN)	32	RW	0000_0000h
50h	Pin Status (PINSTAT)	32	RW	0000_0000h
54h	Pin Interrupt Enable (PINIEN)	32	RW	0000_0000h
58h	Pin Rising Edge Enable (PINREN)	32	RW	0000_0000h
5Ch	Pin Falling Edge Enable (PINFEN)	32	RW	0000_0000h
60h	Pin Output Data (PINOUTD)	32	RW	0000_0000h
64h	Pin Output Enable (PINOUTE)	32	RW	0000_0000h
68h	Pin Output Disable (PINOUTDIS)	32	W	0000_0000h
6Ch	Pin Output Clear (PINOUTCLR)	32	W	0000_0000h
70h	Pin Output Set (PINOUTSET)	32	W	0000_0000h
74h	Pin Output Toggle (PINOUTTOG)	32	W	0000_0000h
80h - 9Ch	Shifter Control N (SHIFTCTL0 - SHIFTCTL7)	32	RW	0000_0000h
100h - 11Ch	Shifter Configuration N (SHIFTCFG0 - SHIFTCFG7)	32	RW	0000_0000h
200h - 21Ch	Shifter Buffer N (SHIFTBUF0 - SHIFTBUF7)	32	RW	0000_0000h
280h - 29Ch	Shifter Buffer N Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7)	32	RW	0000_0000h
300h - 31Ch	Shifter Buffer N Byte Swapped (SHIFTBUFBYS0 - SHIFTBUFBYS7)	32	RW	0000_0000h
380h - 39Ch	Shifter Buffer N Bit Byte Swapped (SHIFTBUFBBS0 - SHIFTBUFBBS7)	32	RW	0000_0000h
400h - 41Ch	Timer Control N (TIMCTL0 - TIMCTL7)	32	RW	0000_0000h
480h - 49Ch	Timer Configuration N (TIMCFG0 - TIMCFG7)	32	RW	0000_0000h
500h - 51Ch	Timer Compare N (TIMCMP0 - TIMCMP7)	32	RW	0000_0000h
680h - 69Ch	Shifter Buffer N Nibble Byte Swapped (SHIFTBUFNBS0 - SHIFTBUFNBS7)	32	RW	0000_0000h
700h - 71Ch	Shifter Buffer N Halfword Swapped (SHIFTBUFHWS0 - SHIFTBUFHWS7)	32	RW	0000_0000h
780h - 79Ch	Shifter Buffer N Nibble Swapped (SHIFTBUFNIS0 - SHIFTBUFNIS7)	32	RW	0000_0000h
800h - 81Ch	Shifter Buffer N Odd Even Swapped (SHIFTBUFOES0 - SHIFTBUFOES7)	32	RW	0000_0000h
880h - 89Ch	Shifter Buffer N Even Odd Swapped (SHIFTBUFEOS0 - SHIFTBUFEOS7)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
900h - 91Ch	Shifter Buffer N Halfword Byte Swapped (SHIFTBUFHBS0 - SHIFTBUFHBS7)	32	RW	0000_0000h

60.7.1.2 Version ID (VERID)

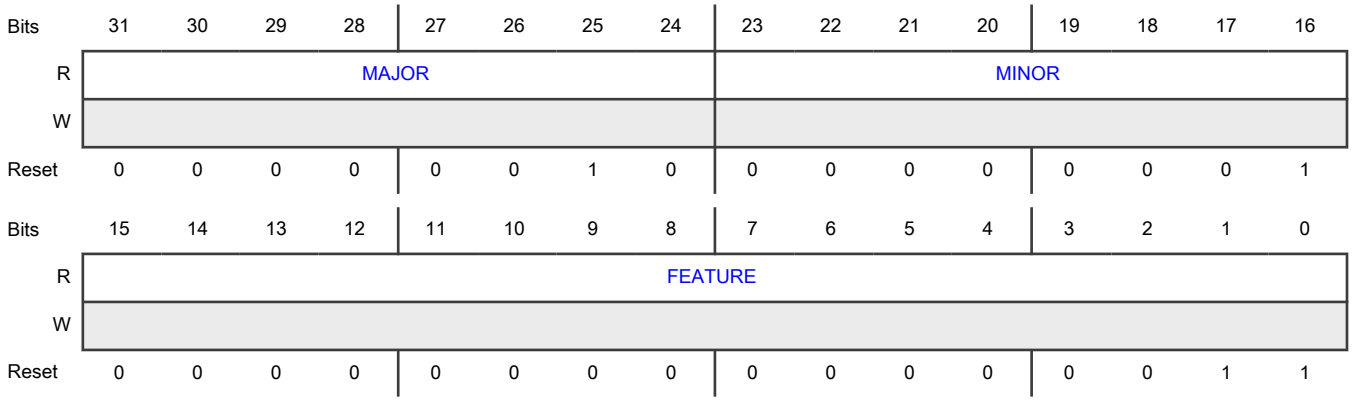
Offset

Register	Offset
VERID	0h

Function

Contains the version of the FLEXIO.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number This read-only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read-only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Specification Number This read-only field returns the feature set number. 0000_0000_0000_0000b - Standard features implemented.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0000_0000_0000_0001b - Supports state, logic, and parallel modes. 0000_0000_0000_0010b - Supports pin control registers. 0000_0000_0000_0011b - Supports state, logic, and parallel modes, plus pin control registers.

60.7.1.3 Parameter (PARAM)

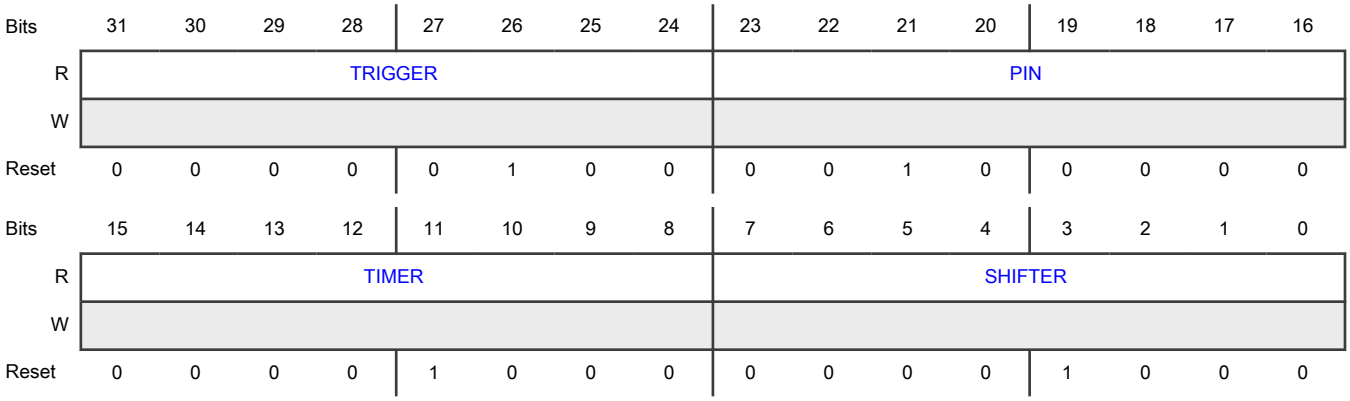
Offset

Register	Offset
PARAM	4h

Function

Contains the number of shifters, timers, pins, and triggers.

Diagram



Fields

Field	Function
31-24 TRIGGER	Trigger Number Number of external triggers implemented.
23-16 PIN	Pin Number Number of pins implemented.
15-8 TIMER	Timer Number Number of timers implemented.
7-0 SHIFTER	Shifter Number Number of shifters implemented.

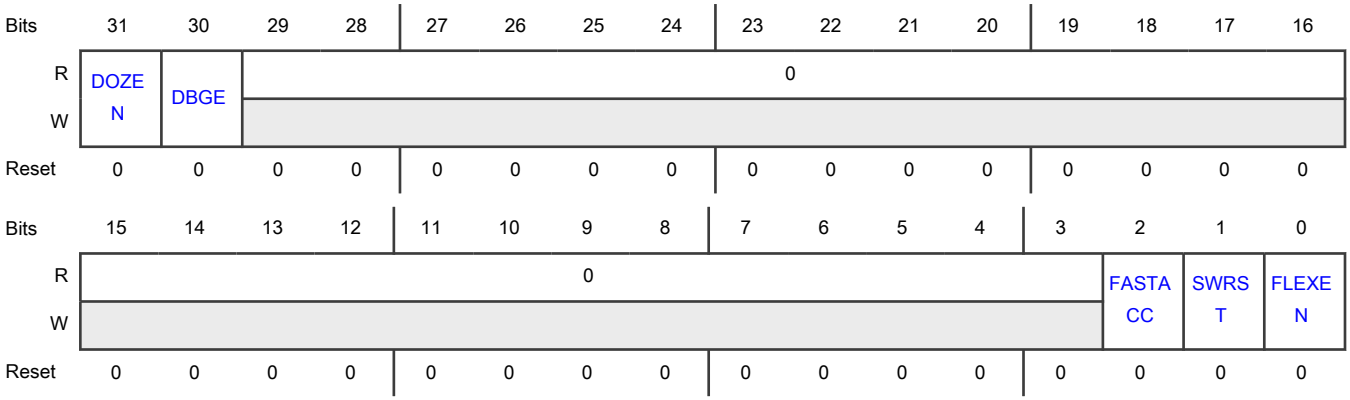
60.7.1.4 FLEXIO Control (CTRL)

Offset

Register	Offset
CTRL	8h

Function
Controls various aspects of FLEXIO operation.

Diagram



Fields

Field	Function
31 DOZEN	Doze Enable Disables FLEXIO operation in Doze modes. 0b - FLEXIO enabled in Doze modes. 1b - FLEXIO disabled in Doze modes.
30 DBGE	Debug Enable Enables FLEXIO operation in Debug mode. 0b - FLEXIO is disabled in Debug modes. 1b - FLEXIO is enabled in Debug modes.
29-3 —	Reserved
2 FASTACC	Fast Access Enables fast register accesses to FLEXIO registers, but requires the FLEXIO functional clock to be at least equal to the frequency of the bus clock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Configures for normal register accesses to FLEXIO 1b - Configures for fast register accesses to FLEXIO
1 SWRST	Software Reset The FLEXIO Control Register is not affected by the software reset. All other logic in the FLEXIO is affected by the software reset and all other register accesses are ignored until this field is cleared. This field remains set until cleared by software, and the reset has cleared in the FLEXIO clock domain. 0b - Software reset is disabled 1b - Software reset is enabled. All FLEXIO registers except the Control Register are reset.
0 FLEXEN	FLEXIO Enable Enables or disables the FLEXIO. 0b - FLEXIO module is disabled. 1b - FLEXIO module is enabled.

60.7.1.5 Pin State (PIN)

Offset

Register	Offset
PIN	Ch

Function

Reports status of the Pin Data Input.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PDI	Pin Data Input Returns the input data on each of the FLEXIO pins.

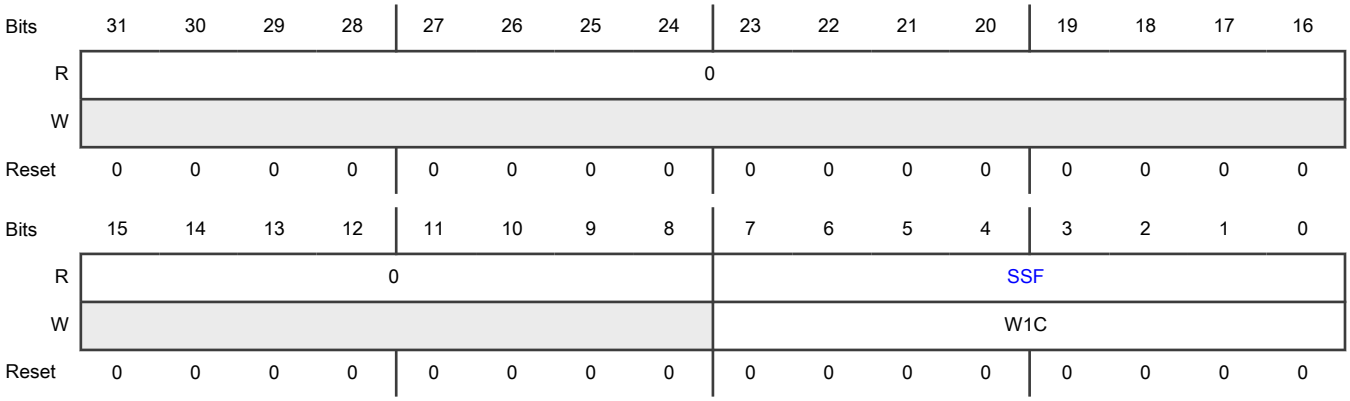
60.7.1.6 Shifter Status (SHIFTSTAT)

Offset

Register	Offset
SHIFTSTAT	10h

Function
Contains Shifter Status Flags.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SSF	Shifter Status Flag The shifter status flag is updated when one of the following events occurs: • For SMOD=Receive, the status flag is set when SHIFTBUF has been loaded with data from the shifter (SHIFTBUF is full). The status flag is cleared when SHIFTBUF register is read. • For SMOD=Transmit, the status flag is set when SHIFTBUF data has been transferred to the shifter (SHIFTBUF is empty) or when initially configured for SMOD=Transmit. The status flag is cleared when the SHIFTBUF register is written.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• For SMOD=Match Store, the status flag is set when a match has occurred between SHIFTBUF and the shifter. The status flag is cleared when the SHIFTBUF register is read. • For SMOD=Match Continuous, returns the current match result between the SHIFTBUF and shifter. The status flag cannot be cleared by reading SHIFTBUF. • For SMOD=State, the status flag for a shifter sets when it is selected by the current state pointer. • For SMOD=Logic, returns the current value of the programmable logic block output. The status flag can also be cleared by writing a logic one to the flag for all modes except Match Continuous/ State /Logic. 0b - Status flag is clear. 1b - Status flag is set.

60.7.1.7 Shifter Error (SHIFTERR)

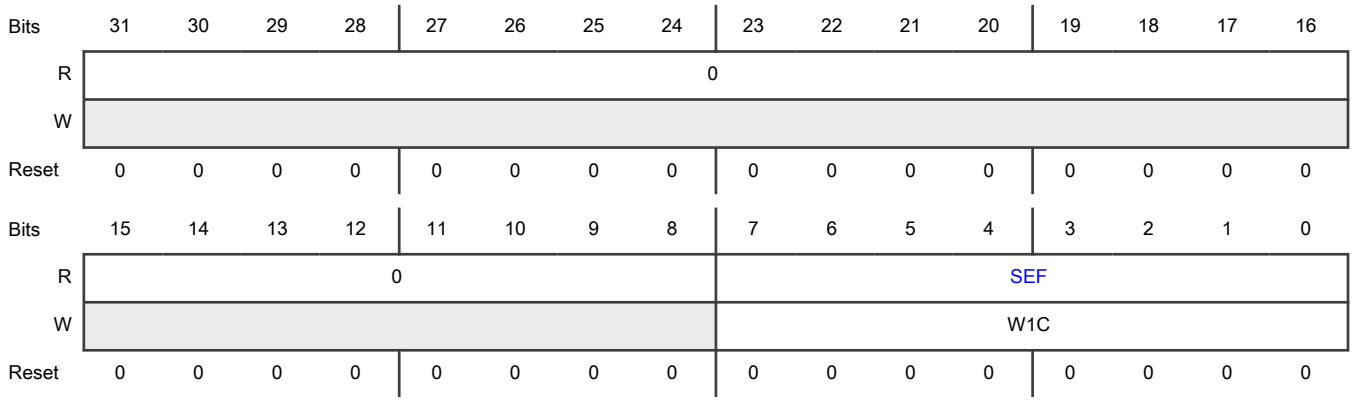
Offset

Register	Offset
SHIFTERR	14h

Function

Reports Shifter Error Flags.

Diagram



Fields

Field	Function
31-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 SEF	Shifter Error Flags The shifter error flag is set when one of the following events occurs: • For SMOD=Receive, this indicates that either the shifter was ready to store new data into SHIFTBUF before the previous data was read from SHIFTBUF (SHIFTBUF overrun), or it indicates that the received start or stop bit does not match the expected value. • For SMOD=Transmit, indicates that the shifter was ready to load new data from SHIFTBUF before new data had been written into SHIFTBUF (SHIFTBUF underrun). • For SMOD=Match Store, indicates that a match event occurred before the previous match data was read from SHIFTBUF (SHIFTBUF overrun). • For SMOD=Match Continuous, the error flag is set when a match has occurred between SHIFTBUF and the shifter. • For SMOD=Logic, the error flag is set when the output of the programmable logic block has asserted. Clear this field by writing logic one to the flag. For SMOD=Match Continuous, it can also be cleared when the SHIFTBUF register is read. 0b - Shifter Error Flag is clear. 1b - Shifter Error Flag is set.

60.7.1.8 Timer Status (TIMSTAT)

Offset

Register	Offset
TIMSTAT	18h

Function

Reports Timer Status Flags.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TSF							
W									W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 TSF	Timer Status Flags The timer status flag sets depending on the timer mode, and can be cleared by writing logic one to the flag. In 8-bit baud counter mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 8-bit high PWM mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 16-bit counter mode, the timer status flag is set when the 16-bit counter equals zero and decrements. In 16-bit counter disable mode, the timer status flag is set when a timer disable event is detected. In 8-bit word counter mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 8-bit low PWM mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 16-bit input capture mode, the timer status flag is set when a timer disable event is detected and the timer status flag is clear. In this mode, the timer compare register should only be read when the timer status flag is set. 0b - Timer Status Flag is clear. 1b - Timer Status Flag is set.

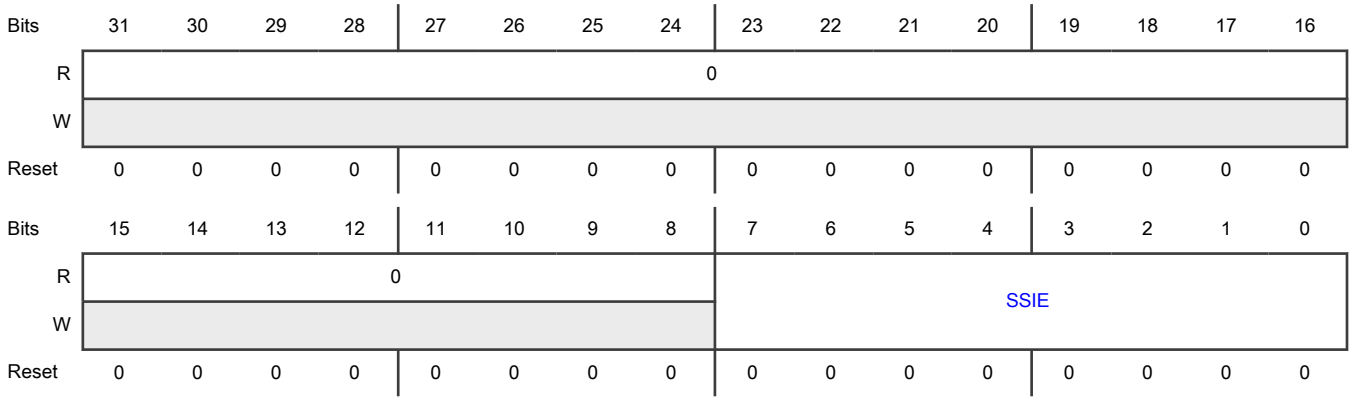
60.7.1.9 Shifter Status Interrupt Enable (SHIFTSIEN)

Offset

Register	Offset
SHIFTSIEN	20h

Function
Contains enables for Shifter Status Interrupts.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SSIE	Shifter Status Interrupt Enable Enables interrupt generation when corresponding SSF is set. 0b - Shifter Status Flag interrupt disabled. 1b - Shifter Status Flag interrupt enabled.

60.7.1.10 Shifter Error Interrupt Enable (SHIFTEIEN)

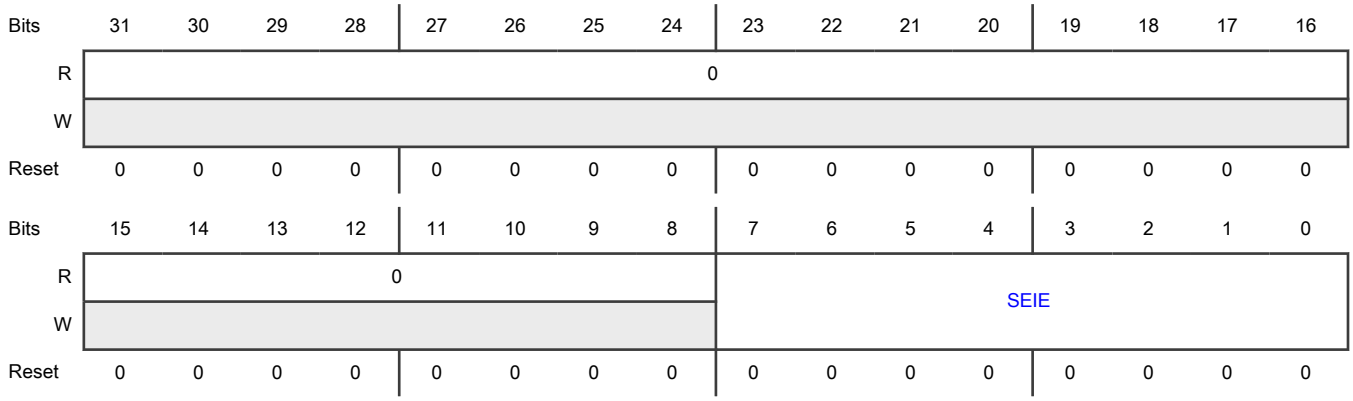
Offset

Register	Offset
SHIFTEIEN	24h

Function

Contains enables for Shifter Error Interrupts.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SEIE	Shifter Error Interrupt Enable Enables interrupt generation when corresponding SEF is set. 0b - Shifter Error Flag interrupt disabled. 1b - Shifter Error Flag interrupt enabled.

60.7.1.11 Timer Interrupt Enable (TIMIEN)

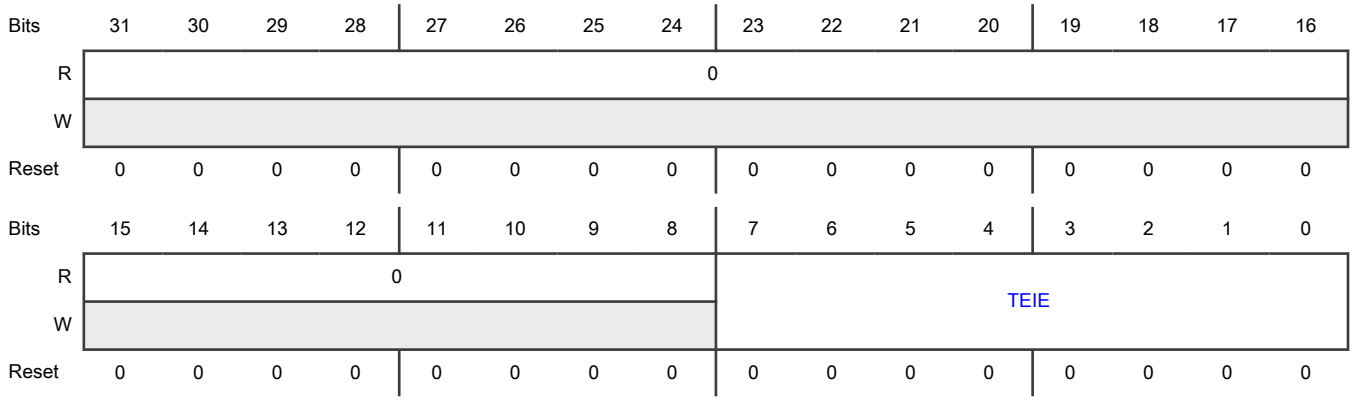
Offset

Register	Offset
TIMIEN	28h

Function

Contains enables for Timer Status Interrupts.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 TEIE	Timer Status Interrupt Enable Enables interrupt generation when corresponding TSF is set. 0b - Timer Status Flag interrupt is disabled. 1b - Timer Status Flag interrupt is enabled.

60.7.1.12 Shifter Status DMA Enable (SHIFTSDEN)

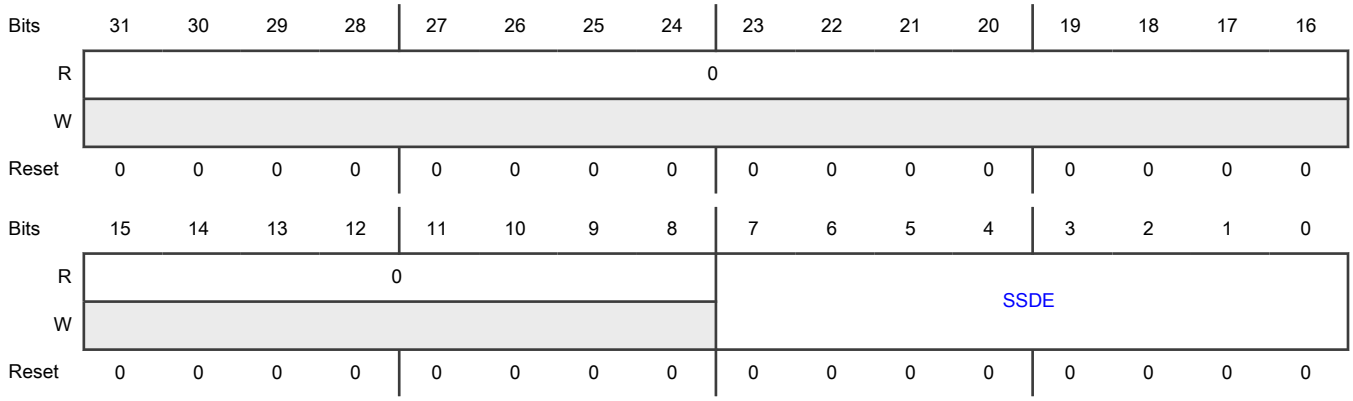
Offset

Register	Offset
SHIFTSDEN	30h

Function

Contains enables for Shifter DMA requests.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SSDE	Shifter Status DMA Enable Enables DMA request generation when corresponding SSF is set. 0b - Shifter Status Flag DMA request is disabled. 1b - Shifter Status Flag DMA request is enabled.

60.7.1.13 Timer Status DMA Enable (TIMERSDEN)

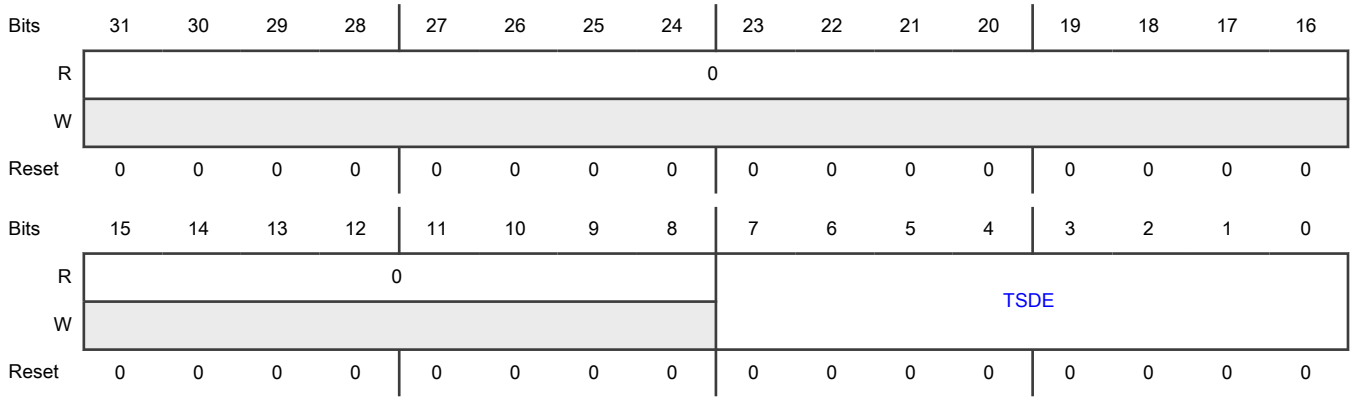
Offset

Register	Offset
TIMERSDEN	38h

Function

Contains enables for DMA requests when Timer Status Flag is set.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 TSDE	Timer Status DMA Enable Enables DMA request generation when corresponding TSF is set. When the Timer Status DMA request is enabled, reading or writing a Timer Compare Register clears the corresponding Timer Status Register. The DMA must therefore read or write the Timer Compare Register as part of the DMA transfer, otherwise the DMA request remains asserted. 0b - Timer Status Flag DMA request is disabled. 1b - Timer Status Flag DMA request is enabled.

60.7.1.14 Shifter State (SHIFTSTATE)

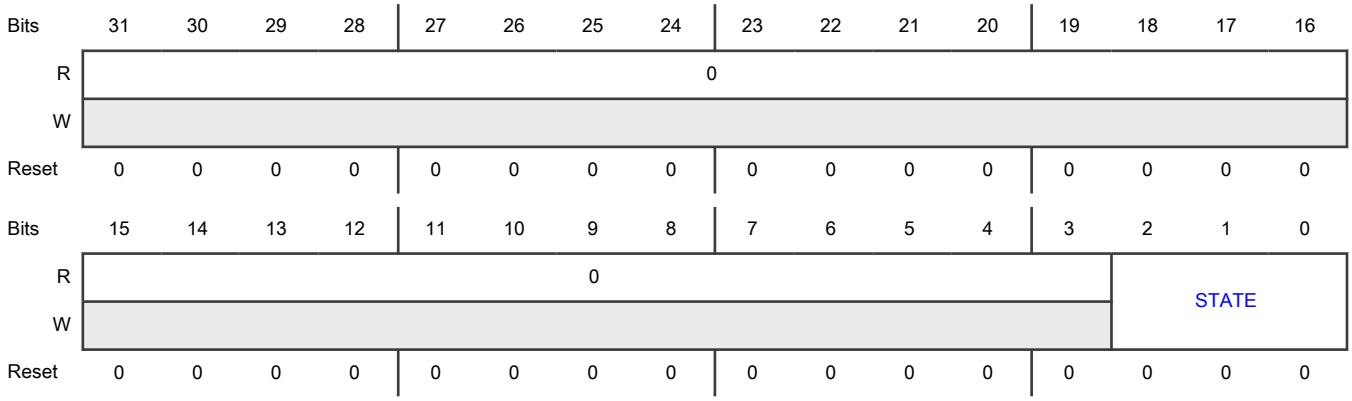
Offset

Register	Offset
SHIFTSTATE	40h

Function

Contains the pointer to track the current shifter.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 STATE	Current State Pointer The current state field maintains a pointer to track the current shifter (configured for State mode) enabled to drive outputs and compute the next state. Reading this register when the state pointer is updating can result in the incorrect state returned. This field is writable and the value written overrides the current state.

60.7.1.15 Trigger Status (TRGSTAT)

Offset

Register	Offset
TRGSTAT	48h

Function

Contains External Trigger Status Flags.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												ETSF			
W													W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 ETSF	External Trigger Status Flags The external trigger status flag is set when a rising edge is detected on the corresponding external trigger input. 0b - External Trigger Status Flag is clear 1b - External Trigger Status Flag is set

60.7.1.16 External Trigger Interrupt Enable (TRIGIEN)

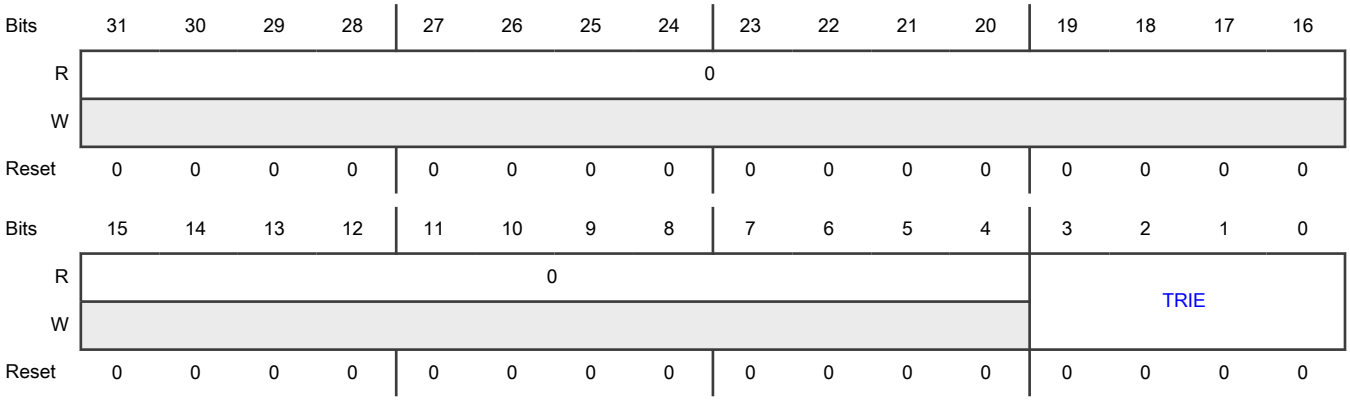
Offset

Register	Offset
TRIGIEN	4Ch

Function

Contains enables for External Trigger Interrupts.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TRIE	External Trigger Interrupt Enable Enables interrupt generation when corresponding ETSF is set. 0b - External Trigger Status Flag interrupt is disabled. 1b - External Trigger Status Flag interrupt is enabled.

60.7.1.17 Pin Status (PINSTAT)

Offset

Register	Offset
PINSTAT	50h

Function

Contains Pin Status Flags.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PSF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PSF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PSF	Pin Status Flags The pin status flag is set when a rising edge or falling edge (if configured) is detected on the corresponding pin, as configured by pin. 0b - Pin Status Flag is clear 1b - Pin Status Flag is set

60.7.1.18 Pin Interrupt Enable (PINIEN)

Offset

Register	Offset
PINIEN	54h

Function

Contains enables for Pin Status Interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PSIE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PSIE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PSIE	Pin Status Interrupt Enable Enables interrupt generation when corresponding PSF is set. 0b - Pin Status Flag interrupt is disabled. 1b - Pin Status Flag interrupt is enabled.

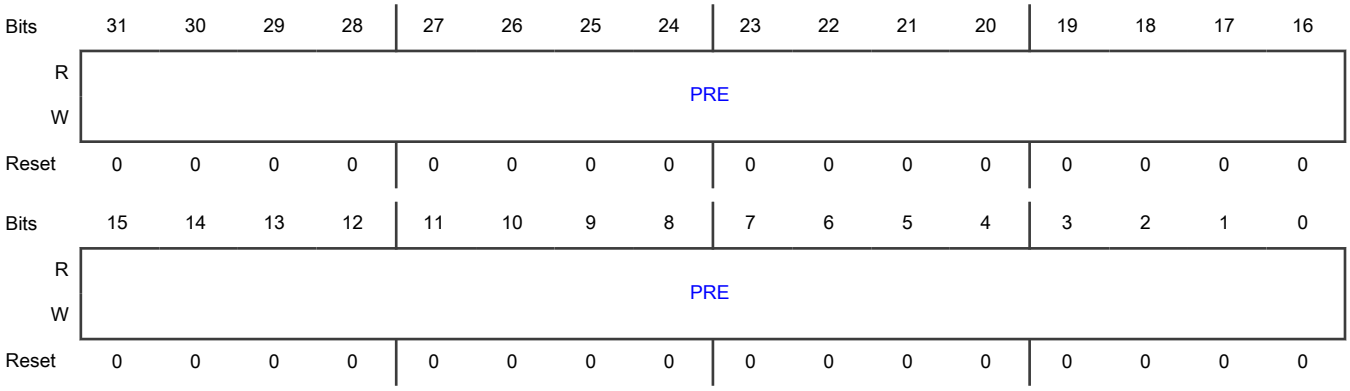
60.7.1.19 Pin Rising Edge Enable (PINREN)

Offset

Register	Offset
PINREN	58h

Function
Contains enables for Pin Status Flag on rising edge.

Diagram



Fields

Field	Function
31-0 PRE	Pin Rising Edge When set, the pin status flag is set whenever a rising edge is detected on the pin. 0b - Pin Status Flag does not set when a pin rising edge is detected. 1b - Pin Status Flag does set when a pin rising edge is detected.

60.7.1.20 Pin Falling Edge Enable (PINFEN)

Offset

Register	Offset
PINFEN	5Ch

Function
Contains enables for Pin Status Flag on falling edge.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PFE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PFE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PFE	Pin Falling Edge When set, the pin status flag is set whenever a falling edge is detected on the pin. 0b - Pin Status Flag does not set when a pin falling edge is detected. 1b - Pin Status Flag does set when a pin falling edge is detected.

60.7.1.21 Pin Output Data (PINOUTD)

Offset

Register	Offset
PINOUTD	60h

Function
Contains Data Output when Direct Pin Output enabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OUTD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OUTD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTD	Output Data Configures the value driven on the corresponding pin when direct pin output is enabled. 0b - Logic zero is driven on the pin when direct pin output is enabled. 1b - Logic one is driven on the pin when direct pin output is enabled.

60.7.1.22 Pin Output Enable (PINOUTE)

Offset

Register	Offset
PINOUTE	64h

Function
Contains enables for pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OUTE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OUTE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTE	Output Enable Enables direct output on the corresponding pin. 0b - Pin is controlled by timer/shifter configuration. 1b - Pin is an output and driven with value of PINOUTD register.

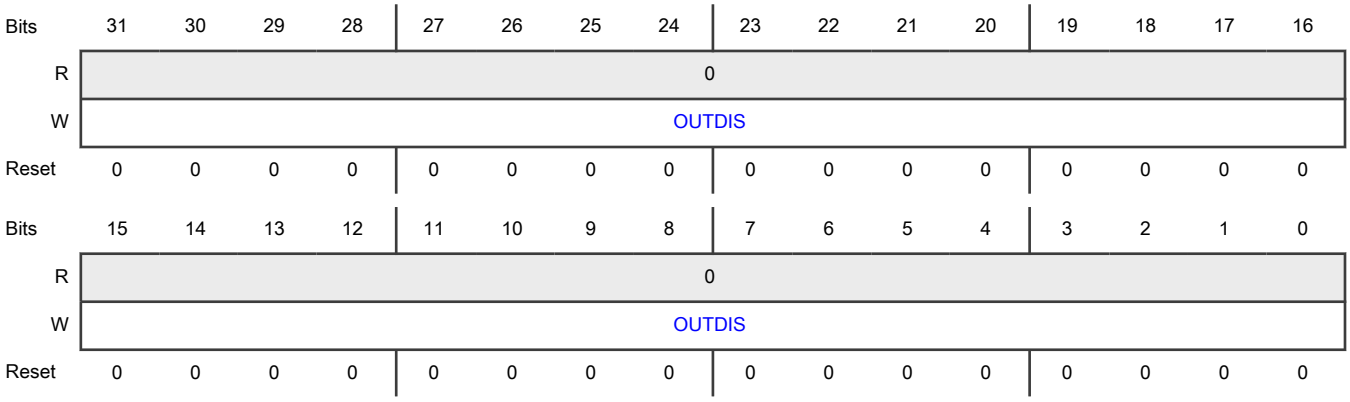
60.7.1.23 Pin Output Disable (PINOUTDIS)

Offset

Register	Offset
PINOUTDIS	68h

Function
Contains disables for pin output.

Diagram



Fields

Field	Function
31-0 OUTDIS	Output Disable Configures corresponding pins to disable direct output. 0b - No effect. 1b - Corresponding PINOUTD and PINOUTE bits are cleared.

60.7.1.24 Pin Output Clear (PINOUTCLR)

Offset

Register	Offset
PINOUTCLR	6Ch

Function
Clears pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	OUTCLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	OUTCLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTCLR	Output Clear Configures corresponding pins to output zero. 0b - No effect. 1b - Corresponding PINOUTD is cleared and PINOUTE is set.

60.7.1.25 Pin Output Set (PINOUTSET)

Offset

Register	Offset
PINOUTSET	70h

Function
Sets pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	OUTSET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	OUTSET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTSET	Output Set Configures corresponding pins to output logic one. 0b - No effect. 1b - Corresponding PINOUTD is set and PINOUTE is set.

60.7.1.26 Pin Output Toggle (PINOUTTOG)

Offset

Register	Offset
PINOUTTOG	74h

Function
Toggles pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	OUTTOG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	OUTTOG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTTOG	Output Toggle Configures corresponding pins to toggle. 0b - No effect. 1b - Corresponding PINOUTD is inverted and PINOUTE is set.

60.7.1.27 Shifter Control N (SHIFTCTL0 - SHIFTCTL7)

Offset

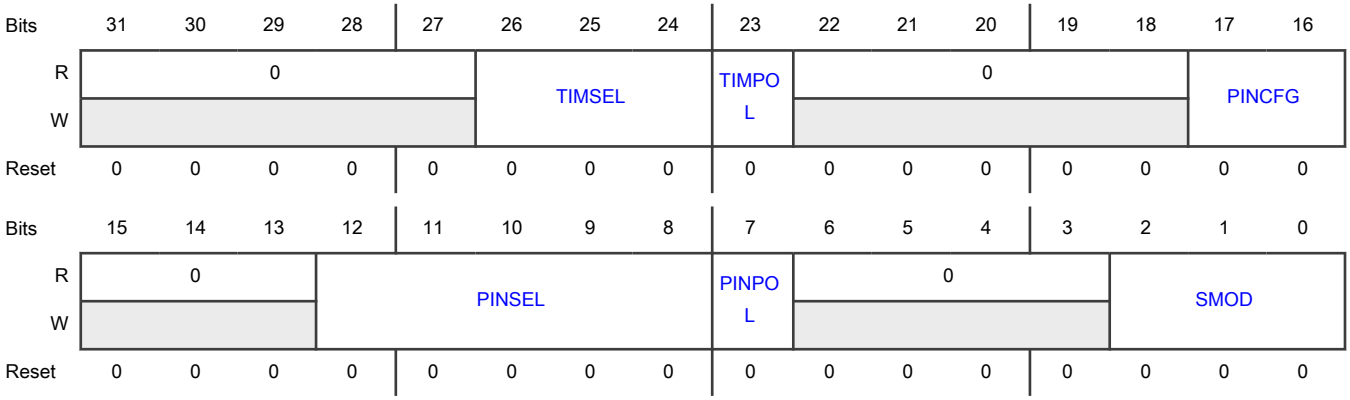
For n = 0 to 7:

Register	Offset
SHIFTCTLn	80h + (n × 4h)

Function

Controls Shift Register N.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 TIMSEL	Timer Select Selects which timer is used for controlling the logic or shift register and generating the shift clock. TIMSEL=i selects TIMERi.
23	Timer Polarity

Table continues on the next page...

Table continued from the previous page...

Field	Function
TIMPOL	Determines the shift occurs on the positive edge or negative edge of the shift clock. 0b - Shift on posedge of shift clock 1b - Shift on negedge of shift clock
22-18 —	Reserved
17-16 PINCFG	Shifter Pin Configuration For pins configured as an output (PINCFG=11b), this field takes effect when the register is written. **NOTE** When initially configuring PINCFG=11b, FLEXIO may briefly drive the pin low. To avoid this, you can configure PINCFG=10b along with the rest of the control register and then perform a subsequent write to set PINCFG=11b. Likewise, when changing PINCFG=11b to PINCFG=00b software should perform an initial write to set PINCFG=10b and then perform a subsequent write to update the rest of the control register with PINCFG=00b. 00b - Shifter pin output disabled 01b - Shifter pin open-drain or bidirectional output enable 10b - Shifter pin bidirectional output data 11b - Shifter pin output
15-13 —	Reserved
12-8 PINSEL	Shifter Pin Select Selects which pin is used by the Shifter input or output. PINSEL=i selects the FXIO_Di pin. For pins configured as an output (PINCFG=11b), this field takes effect when the register is written.
7 PINPOL	Shifter Pin Polarity For pins configured as an output (PINCFG=11b), this field takes effect when the register is written. 0b - Pin is active high 1b - Pin is active low
6-3 —	Reserved
2-0 SMOD	Shifter Mode Configures the mode of the shifter. 000b - Disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - Receive mode. Captures the current shifter content into the SHIFTBUF on expiration of the timer. 010b - Transmit mode. Load SHIFTBUF contents into the shifter on expiration of the timer. 011b - Reserved. 100b - Match Store mode. Shifter data is compared to SHIFTBUF content on expiration of the timer. 101b - Match Continuous mode. Shifter data is continuously compared to SHIFTBUF contents. 110b - State mode. SHIFTBUF contents are used for storing programmable state attributes. 111b - Logic mode. SHIFTBUF contents are used for implementing programmable logic lookup table.

60.7.1.28 Shifter Configuration N (SHIFTCFG0 - SHIFTCFG7)

Offset

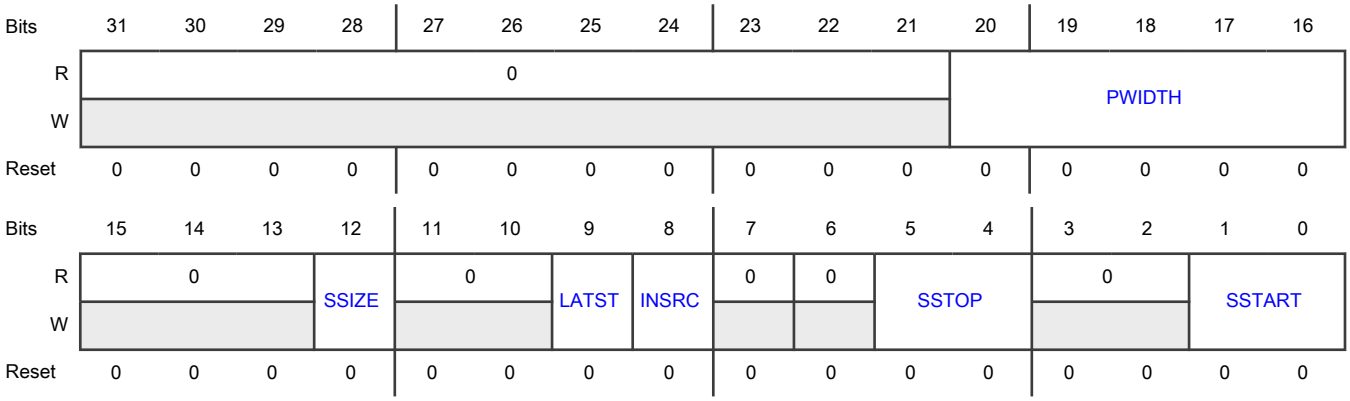
For n = 0 to 7:

Register	Offset
SHIFTCFGn	100h + (n × 4h)

Function

Contains configuration bit fields for Shifter Register N.

Diagram



Fields

Field	Function
31-21	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
20-16 PWIDTH	Parallel Width For all shifters, this field configures the number of bits to be shifted on each shift clock as follows: • 1-bit shift for PWIDTH=0 • 2-bit shift for PWIDTH=1 • 4-bit shift for PWIDTH=2...3 • 8-bit shift for PWIDTH=4...7 • 16-bit shift for PWIDTH=8...15 • 32-bit shift for PWIDTH=16...31 For shifters which support parallel transmit (SHIFTER0, SHIFTER4, ...) or parallel receive (SHIFTER3, SHIFTER7, ...), this field, together with SHIFTCTL[PINSEL], also selects the pins to be driven or sampled on each shift clock: • FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL] Shifters which do not support parallel transmit or parallel receive only support parallel shift when INSRC=1. For SMOD=State, use this field to disable state outputs. See State mode for more details.
15-13 —	Reserved
12 SSIZE	Shifter Size Configures the size of the Shift Registers. A 24-bit shift register shifts data only into bits [23:0] and does not update bits [31:24] during shift operations. When shift register is configured for a 24-bit shift, configuring PWIDTH=8..15 performs a 12-bit shift, and PWIDTH=16..31 performs a 24-bit shift. 0b - Shift register is 32-bit. 1b - Shift register is 24-bit.
11-10 —	Reserved
9 LATST	Late Store Configures what happens when a receive or match shift register is configured to both shift and store on the same cycle. 0b - Shift register stores the pre-shift register state. 1b - Shift register stores the post-shift register state.
8 INSRC	Input Source Selects the input source for the shifter. Configuring INSRC=1 is not supported for the last shifter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Pin 1b - Shifter N+1 Output
7 —	Reserved
6 —	Reserved
5-4 SSTOP	Shifter Stop bit For SMOD=Transmit, this field allows automatic stop bit insertion if the selected timer has also enabled a stop bit. For SMOD=Receive or Match Store, this field allows automatic stop bit checking if the selected timer has also enabled a stop bit. For SMOD=State, this field is used to disable state outputs. See 'State Mode' section for more detail. For SMOD=Logic, this field is used to mask logic pin inputs. See 'Logic Mode' section for more detail. 00b - Stop bit disabled for transmitter/receiver/match store 01b - Stop bit disabled for transmitter/receiver/match store. When timer is in stop condition, receiver/match store stores receive data on the configured shift edge. 10b - Transmitter outputs stop bit value 0 on store. If stop bit is not 0, receiver/match store sets error flag. When timer is in stop condition, receiver/match stores also store receive data on the configured shift edge. 11b - Transmitter outputs stop bit value 1 on store. If stop bit is not 1, receiver/match store sets error flag. When timer is in stop condition, receiver/match store also stores receive data on the configured shift edge.
3-2 —	Reserved
1-0 SSTART	Shifter Start Bit For SMOD=Transmit, this field allows automatic start bit insertion if the selected timer has also enabled a start bit. For SMOD=Receive or Match Store, this field allows automatic start bit checking if the selected timer has also enabled a start bit. For SMOD=State, this field is used to disable state outputs. See State mode for more detail. For SMOD=Logic, this field is used to mask logic pin inputs. See Logic mode for more detail. 00b - Start bit disabled for transmitter/receiver/match store. Transmitter loads data on enable. 01b - Start bit disabled for transmitter/receiver/match store. Transmitter loads data on first shift. 10b - Transmitter outputs start bit value 0 before loading data on first shift. If start bit is not 0, receiver/match store sets error flag.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Transmitter outputs start bit value 1 before loading data on first shift. If start bit is not 1, receiver/match store sets error flag.

60.7.1.29 Shifter Buffer N (SHIFTBUF0 - SHIFTBUF7)

Offset

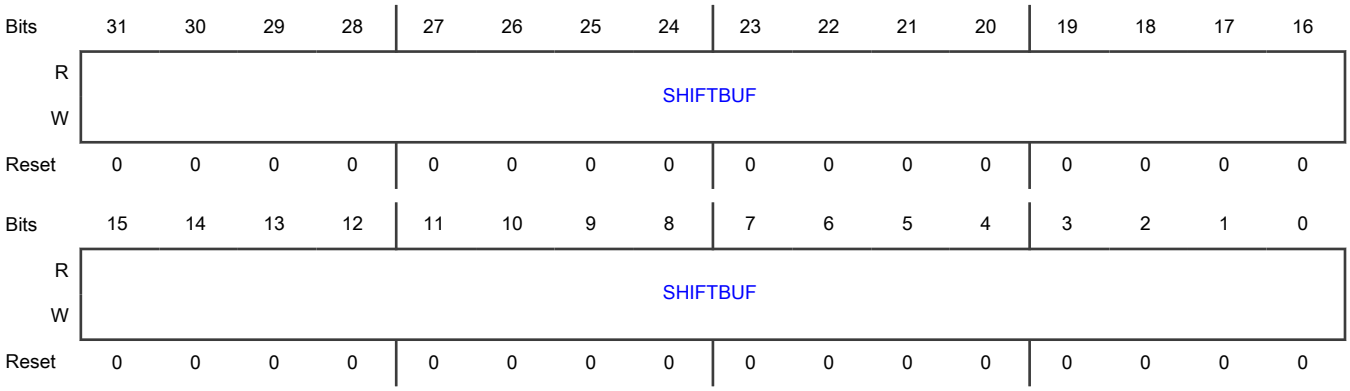
For n = 0 to 7:

Register	Offset
SHIFTBUFn	200h + (n × 4h)

Function

Contains Shift Buffer data.

Diagram



Fields

Field	Function
31-0 SHIFTBUF	Shift Buffer Shift buffer data is used for various functions depending on the SHIFTCTL0[SMOD] setting: For SMOD=Receive, shifter data is transferred into SHIFTBUF at the expiration of timer. Only read the SHIFTBUF register when the corresponding shifter status flag is set, indicating new shifter data is available. For SMOD=Transmit, SHIFTBUF data is transferred into the shifter before the timer begins. For SMOD=Match Store, SHIFTBUF[31:16] contains the data to be matched with the shifter contents and SHIFTBUF[15:0] can be used to mask the match result (1=mask, 0=no mask). The match is checked when the timer expires. Shifter data [31:16] is written to SHIFTBUF[31:16] whenever a match event occurs. Only read the SHIFTBUF register when the corresponding shifter status flag is set, indicating new shifter data is available.

Table continues on the next page...

Field	Function
	For SMOD=Match Continuous, SHIFTBUF[31:16] contains the data to be matched with the shifter contents and SHIFTBUF[15:0] can be used to mask the match result (1=mask, 0=no mask). For SMOD=Logic, SHIFTBUF[31:0] is used to implement a 5-input, 32-bit programmable logic look-up table. See Logic mode for more details. For SMOD=State, use SHIFTBUF[31:24] to drive the output value when this shifter is selected by the current state pointer and use SHIFTBUF[23:0] to configure the value of the next state transition. See State mode for more detail.

60.7.1.30 Shifter Buffer N Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTBUFBISn	280h + (n × 4h)

Function

Contains SHIFTBUF register content, but contents are bit-swapped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFBIS	Shift Buffer Alias to SHIFTBUF register, but reads or writes to this register are bit-swapped. Reads return SHIFTBUF[0:31].

60.7.1.31 Shifter Buffer N Byte Swapped (SHIFTBUFBYS0 - SHIFTBUFBYS7)

Offset

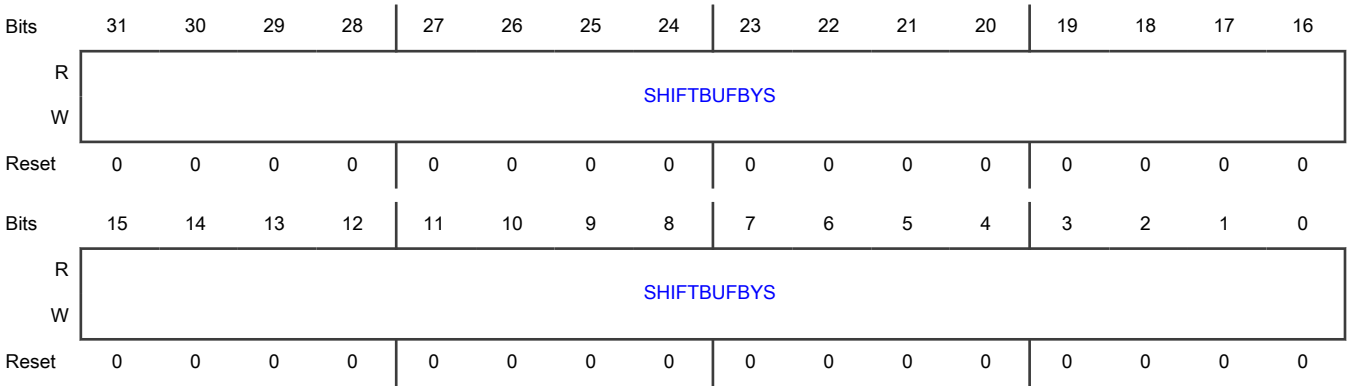
For n = 0 to 7:

Register	Offset
SHIFTBUFBYSn	300h + (n × 4h)

Function

Contains SHIFTBUF register content, but contents are byte-swapped.

Diagram



Fields

Field	Function
31-0 SHIFTBUFBYS	Shift Buffer Alias to SHIFTBUF register, but reads or writes to this register are byte-swapped. Reads return {SHIFTBUF[7:0], SHIFTBUF[15:8], SHIFTBUF[23:16], SHIFTBUF[31:24]}.

60.7.1.32 Shifter Buffer N Bit Byte Swapped (SHIFTBUFBBS0 - SHIFTBUFBBS7)

Offset

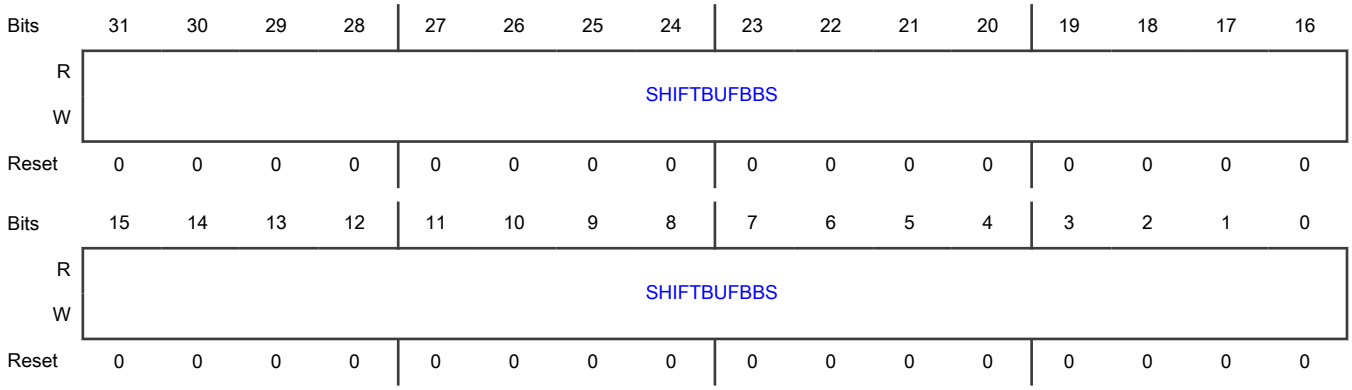
For n = 0 to 7:

Register	Offset
SHIFTBUFBBSn	380h + (n × 4h)

Function

Contains SHIFTBUF register data, but is bit-swapped within each byte.

Diagram



Fields

Field	Function
31-0 SHIFTBUFBBS	Shift Buffer Alias to SHIFTBUF register, except reads or writes to this register are bit-swapped within each byte. Reads return {SHIFTBUF[24:31], SHIFTBUF[16:23], SHIFTBUF[8:15], SHIFTBUF[0:7]}.

60.7.1.33 Timer Control N (TIMCTL0 - TIMCTL7)

Offset

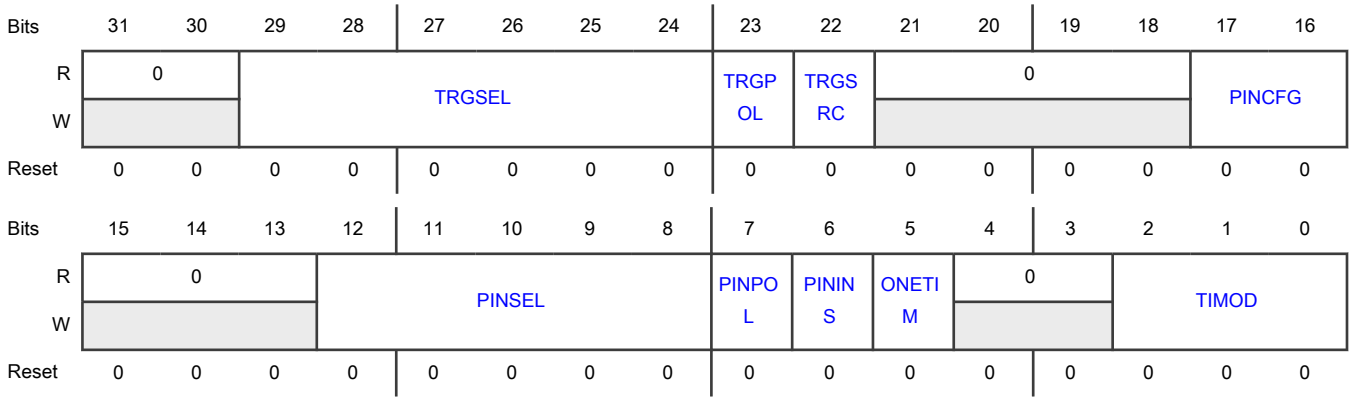
For n = 0 to 7:

Register	Offset
TIMCTLn	400h + (n × 4h)

Function

Controls various settings for Timer N.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 TRGSEL	Trigger Select The valid values for TRGSEL depend on the FLEXIO_PARAM register: • When TRGSRC = 1, the valid values for N depend on the PIN, TIMER, and SHIFTER fields in the FLEXIO_PARAM register. • When TRGSRC = 0, the valid values for N depend on the TRIGGER field in FLEXIO_PARAM register. See the chip-specific FLEXIO information for external trigger selection. **NOTE** For a pin, N=0 to 31. For a shifter, N=0 to 7. For a timer, N=0 to 7. When TRGSRC = 0 the trigger selection is configured as follows: • N - External trigger N input When TRGSRC = 1, the internal trigger can be configured to select an input pin as follows: • 2*N - Pin N input When TRGSRC = 1, the internal trigger can be configured to select a shifter or timer signal as follows: • 4*N+1 - Shifter N status flag • 4*N+3 - Timer N trigger output The expanded internal trigger selection (TRGSRC = 1) is as follows: • 0000 = Pin 0 • 0001 = Shifter 0 Flag • 0010 = Pin 1 • 0011 = Timer 0 Trigger • 0100 = Pin 2 • 0101 = Shifter 1 Flag • 0110 = Pin 3 • 0111 = Timer 1 Trigger • ... • This continues up to the Pin 31, Shifter 7, and Timer 7.
23 TRGPOL	Trigger Polarity Determines if the trigger is active high or active low. 0b - Trigger active high 1b - Trigger active low
22	Trigger Source

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRGSRC	Determines if the trigger source is external or internal. 0b - External trigger selected 1b - Internal trigger selected
21-18 —	Reserved
17-16 PINCFG	Timer Pin Configuration Configures the direction of the timer pin. For pins configured as an output (PINCFG=11b), this field takes effect when the register is written. **NOTE** When initially configuring PINCFG=11b, FLEXIO may briefly drive the pin low. To avoid this, configure PINCFG=10b along with the rest of the control register and then perform a subsequent write to set PINCFG=11b. Likewise, when changing PINCFG=11b to PINCFG=00b software should perform an initial write to set PINCFG=10b and then perform a subsequent write to update the rest of the control register with PINCFG=00b. 00b - Timer pin output disabled 01b - Timer pin open-drain or bidirectional output enable 10b - Timer pin bidirectional output data 11b - Timer pin output
15-13 —	Reserved
12-8 PINSEL	Timer Pin Select Selects which pin is used by the timer input or output. PINSEL=i selects the FXIO_Di pin. For pins configured as an output (PINCFG=11b), this field takes effect when the register is written.
7 PINPOL	Timer Pin Polarity For pins configured as an output (PINCFG=11b), this field takes effect when the register is written. 0b - Pin is active high 1b - Pin is active low
6 PININS	Timer Pin Input Select When set, the timer input pin is a different pin from the timer output pin. PINSEL must select an even-numbered pin when this field is set, meaning the output pin is even-numbered and input pin is odd-numbered. 0b - Timer pin input and output are selected by PINSEL. 1b - Timer pin input is selected by PINSEL+1. Timer pin output remains selected by PINSEL.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 ONETIM	Timer One Time Operation Use this field to configure the timer to perform a single enable/disable iteration. Clear the timer status flag for the timer to be enabled again. 0b - The timer enable event is generated as normal. 1b - The timer enable event is blocked unless timer status flag is clear.
4-3 —	Reserved
2-0 TIMOD	Timer Mode In 8-bit baud counter mode, the lower 8 bits of the counter and compare register are used to configure the baud rate of the timer shift clock. The upper 8 bits are used to configure the shifter bit count. In 8-bit PWM high mode, the lower 8 bits of the counter and compare register are used to configure the high period of the timer shift clock. The upper 8 bits are used to configure the low period of the timer shift clock. The shifter bit count is configured using another timer or external signal. In 16-bit counter mode, the full 16 bits of the counter and compare register are used to configure either the baud rate of the shift clock or the shifter bit count. In 16-bit counter disable mode, the full 16 bits of the counter and compare register are used to configure either the baud rate of the shift clock or the shifter bit count. In 8-bit word counter mode, the lower 8 bits of the counter and compare register are used to configure the shifter bit count. The upper 8 bits are used to configure the shifter word count. In 8-bit PWM low mode, the lower 8 bits of the counter and compare register are used to configure the low period of the timer shift clock. The upper 8 bits are used to configure the high period of the timer shift clock. Use another timer or external signal to configure the shifter bit count. In 16-bit input capture mode, the inverted value of the 16-bit counter is latched into the compare register when a timer counter disable condition is detected (as configured by TIMDIS). This also sets the timer status flag. The timer counter is immediately restarted from FFFFh 000b - Timer disabled. 001b - Dual 8-bit counters baud mode. 010b - Dual 8-bit counters PWM high mode. 011b - Single 16-bit counter mode. 100b - Single 16-bit counter disable mode. 101b - Dual 8-bit counters word mode. 110b - Dual 8-bit counters PWM low mode. 111b - Single 16-bit input capture mode.

60.7.1.34 Timer Configuration N (TIMCFG0 - TIMCFG7)

Offset

For n = 0 to 7:

Register	Offset
TIMCFGn	480h + (n × 4h)

Function

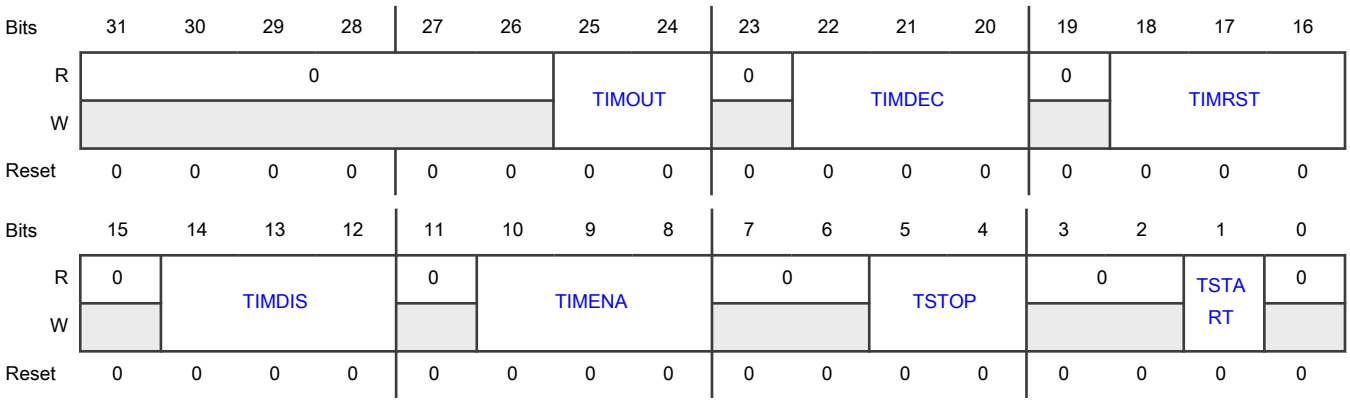
Controls various aspects of Timer Configuration.

The options to enable or disable the timer using the Timer N-1 enable or disable are reserved when N is evenly divisible by 4 (Timer 0, for example).

NOTE

The pin and trigger level/edges specified below refer to the signal state after being modified by the PINPOL or TRGPOL setting in the TIMCTL register. For example, "Trigger low" means that a trigger is actually at logic level 1 if the TRGPOL is set to 1 (active low).

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 TIMOUT	Timer Output Configures the initial state of the timer output and whether it is affected by the timer reset. 00b - Timer output is logic one when enabled and is not affected by timer reset 01b - Timer output is logic zero when enabled and is not affected by timer reset 10b - Timer output is logic one when enabled and on timer reset 11b - Timer output is logic zero when enabled and on timer reset
23 —	Reserved
22-20	Timer Decrement

Table continues on the next page...

Table continued from the previous page...

Field	Function
TIMDEC	Configures the source of the timer decrement and the source of the shift clock. 000b - Decrement counter on FLEXIO clock. Shift clock equals timer output. 001b - Decrement counter on trigger input (both edges). Shift clock equals timer output. 010b - Decrement counter on pin input (both edges). Shift clock equals pin input. 011b - Decrement counter on trigger input (both edges). Shift clock equals trigger input. 100b - Decrement counter on FLEXIO clock divided by 16. Shift clock equals timer output. 101b - Decrement counter on FLEXIO clock divided by 256. Shift clock equals timer output. 110b - Decrement counter on pin input (rising edge). Shift clock equals pin input. 111b - Decrement counter on trigger input (rising edge). Shift clock equals trigger input.
19 —	Reserved
18-16 TIMRST	Timer Reset Configures the condition that causes the timer counter (and optionally the timer output) to be reset. In 8-bit counter mode, the timer reset only resets the lower 8-bits that configure the baud rate. In all other modes, the timer reset resets the full 16-bits of the counter. 000b - Timer never reset 001b - Timer reset on timer output high. 010b - Timer reset on timer pin equal to timer output 011b - Timer reset on timer trigger equal to timer output 100b - Timer reset on timer pin rising edge 101b - Reserved 110b - Timer reset on trigger rising edge 111b - Timer reset on trigger rising or falling edge
15 —	Reserved
14-12 TIMDIS	Timer Disable Configures the condition that causes the timer to be disabled and stop decrementing. 000b - Timer never disabled 001b - Timer disabled on Timer N-1 disable 010b - Timer disabled on timer compare (upper 8-bits match and decrement) 011b - Timer disabled on timer compare (upper 8-bits match and decrement) and trigger low 100b - Timer disabled on pin rising or falling edge

Table continues on the next page...

Table continued from the previous page...

Field	Function
	101b - Timer disabled on pin rising or falling edge provided trigger is high 110b - Timer disabled on trigger falling edge 111b - Reserved
11 —	Reserved
10-8 TIMENA	Timer Enable Configures the condition that causes the timer to be enabled and start decrementing. 000b - Timer always enabled 001b - Timer enabled on Timer N-1 enable 010b - Timer enabled on Trigger high 011b - Timer enabled on Trigger high and Pin high 100b - Timer enabled on Pin rising edge 101b - Timer enabled on Pin rising edge and Trigger high 110b - Timer enabled on Trigger rising edge 111b - Timer enabled on Trigger rising or falling edge
7-6 —	Reserved
5-4 TSTOP	Timer Stop Bit The stop bit can be added on a timer compare (between each word) or on a timer disable. When stop bit is enabled, configured shifters output the contents of the stop bit when the timer is disabled. When stop bit is enabled on timer disable, the timer remains disabled until the next rising edge of the shift clock. If configured for both timer compare and timer disable, only one stop bit is inserted on timer disable. 00b - Stop bit disabled 01b - Stop bit is enabled on timer compare 10b - Stop bit is enabled on timer disable 11b - Stop bit is enabled on timer compare and timer disable
3-2 —	Reserved
1 TSTART	Timer Start Bit When start bit is enabled, configured shifters output the contents of the start bit when the timer is enabled. The timer counter reloads from the compare register on the first rising edge of the shift clock. 0b - Start bit disabled 1b - Start bit enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 —	Reserved

60.7.1.35 Timer Compare N (TIMCMP0 - TIMCMP7)

Offset

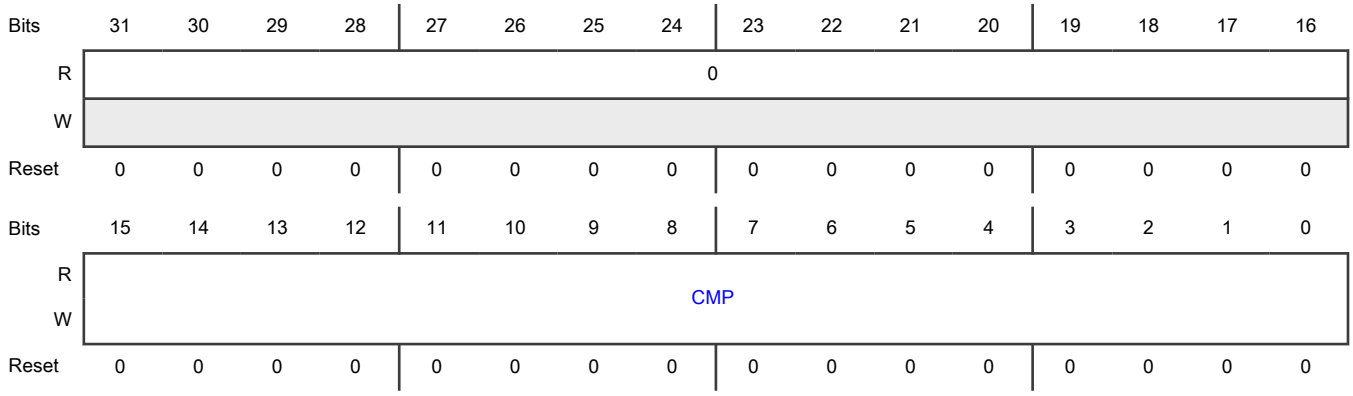
For n = 0 to 7:

Register	Offset
TIMCMPn	500h + (n × 4h)

Function

Contains the Timer Compare Value.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 CMP	Timer Compare Value The timer compare value is loaded into the timer counter when the timer is first enabled, when the timer is reset, and when the timer decrements down to zero. In 8-bit baud counter mode, the lower 8 bits configure the baud rate divider equal to (CMP[7:0] + 1) * 2. The upper 8 bits configure the number of bits in each word equal to (CMP[15:8] + 1) / 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In 8-bit PWM high mode, the lower 8 bits configure the high period of the output to (CMP[7:0] + 1) and the upper 8 bits configure the low period of the output to (CMP[15:8] + 1). In 16-bit counter mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) to equal (CMP[15:0] + 1) * 2. When the shift clock source is a pin or trigger input, the compare register is used to set the number of bits in each word equal to (CMP[15:0] + 1) / 2. In 16-bit counter disable mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) to equal (CMP[15:0] + 1) * 2. When the shift clock source is a pin or trigger input, the compare register is used to set the number of bits in each word equal to (CMP[15:0] + 1) / 2. In 8-bit word counter mode, the lower 8 bits configure the number of bits in each word equal to (CMP[7:0] + 1) / 2. The upper 8 bits configure the number of words to transfer equal to (CMP[15:8] + 1) / 2. In 8-bit PWM low mode, the lower 8 bits configure the low period of the output to (CMP[7:0] + 1) and the upper 8 bits configure the high period of the output to (CMP[15:8] + 1). In 16-bit input capture mode, the compare register is updated with the inverse of the timer counter value whenever the timer status flag is set. Only read the timer compare register when the timer status flag is set.

60.7.1.36 Shifter Buffer N Nibble Byte Swapped (SHIFTBUFNBS0 - SHIFTBUFNBS7)

Offset

For n = 0 to 7:

Register	Offset
SHIFTBUFNBSn	680h + (n × 4h)

Function

Contains SHIFTBUF register content, but contents are nibble-swapped within each byte.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFNBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFNBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFNBS	Shift Buffer Alias to SHIFTBUF register, but reads or writes to this register are nibble-swapped within each byte. Reads return {SHIFTBUF[27:24], SHIFTBUF[31:28], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[3:0], SHIFTBUF[7:4]}.

60.7.1.37 Shifter Buffer N Halfword Swapped (SHIFTBUFHWS0 - SHIFTBUFHWS7)

Offset

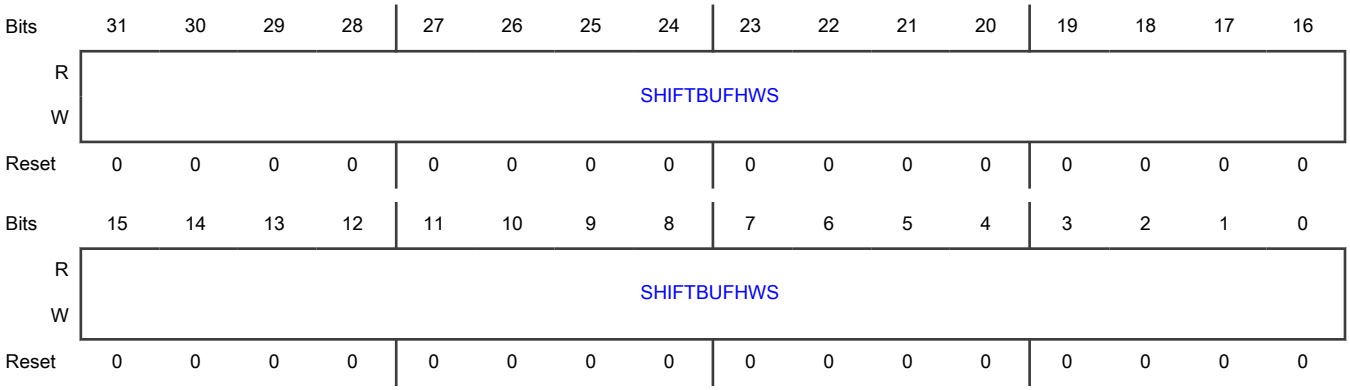
For n = 0 to 7:

Register	Offset
SHIFTBUFHWSn	700h + (n × 4h)

Function

Contains SHIFTBUF register content, but contents are halfword-swapped.

Diagram



Fields

Field	Function
31-0 SHIFTBUFHWS	Shift Buffer Alias to SHIFTBUF register, but reads or writes to this register are halfword-swapped. Reads return {SHIFTBUF[15:0], SHIFTBUF[31:16]}.

60.7.1.38 Shifter Buffer N Nibble Swapped (SHIFTBUFNIS0 - SHIFTBUFNIS7)

Offset

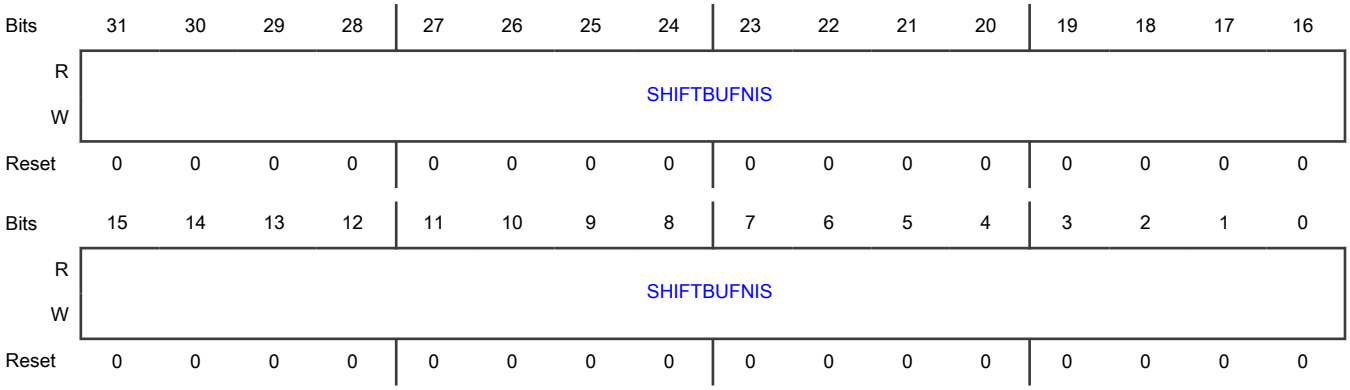
For n = 0 to 7:

Register	Offset
SHIFTBUFNISn	780h + (n × 4h)

Function

Contains SHIFTBUF register content, but contents are nibble-swapped.

Diagram



Fields

Field	Function
31-0 SHIFTBUFNIS	Shift Buffer Alias to SHIFTBUF register, but reads or writes to this register are nibble-swapped. Reads return {SHIFTBUF[3:0], SHIFTBUF[7:4], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[27:24], SHIFTBUF[31:28]}.

60.7.1.39 Shifter Buffer N Odd Even Swapped (SHIFTBUFOES0 - SHIFTBUFOES7)

Offset

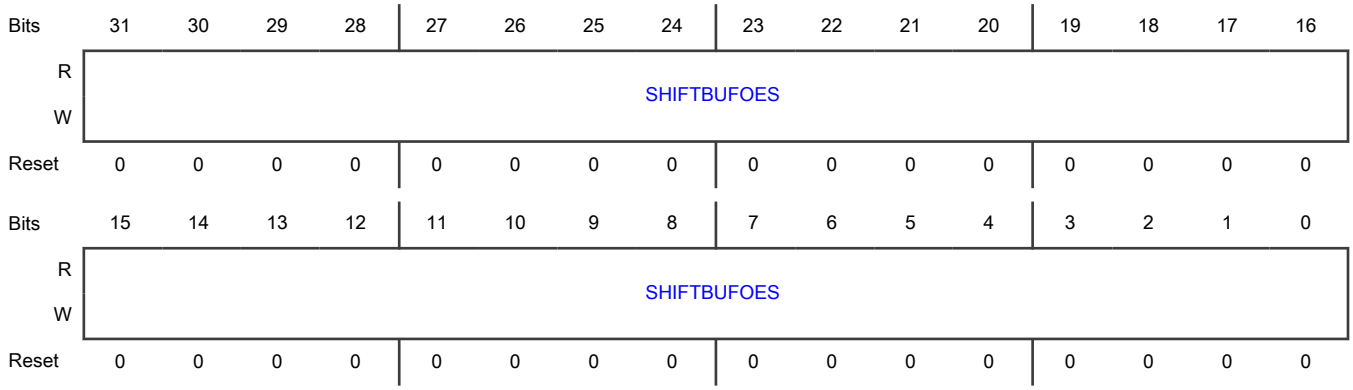
For n = 0 to 7:

Register	Offset
SHIFTBUFOESn	800h + (n × 4h)

Function

Contains SHIFTBUF register content, but contents have odd and even bits partitioned separately.

Diagram



Fields

Field	Function
31-0 SHIFTBUFOES	Shift Buffer Alias to SHIFTBUF register, but reads or writes to this register have the odd and even bits partitioned separately. Only 32-bit accesses are supported to this register. Reads return {SHIFTBUF[31], SHIFTBUF[29], SHIFTBUF[27],SHIFTBUF[25],SHIFTBUF[23],SHIFTBUF[21],SHIFTBUF[19],SHIFTBUF[17],SHIFTBUF[15],SHIFTBUF[13],SHIFTBUF[11],SHIFTBUF[9], SHIFTBUF[7], SHIFTBUF[5], SHIFTBUF[3], SHIFTBUF[1], SHIFTBUF[30], SHIFTBUF[28], SHIFTBUF[26], SHIFTBUF[24], SHIFTBUF[22], SHIFTBUF[20], SHIFTBUF[18], SHIFTBUF[16], SHIFTBUF[14], SHIFTBUF[12], SHIFTBUF[10], SHIFTBUF[8], SHIFTBUF[6], SHIFTBUF[4], SHIFTBUF[2], SHIFTBUF[0]}.

60.7.1.40 Shifter Buffer N Even Odd Swapped (SHIFTBUFEOS0 - SHIFTBUFEOS7)

Offset

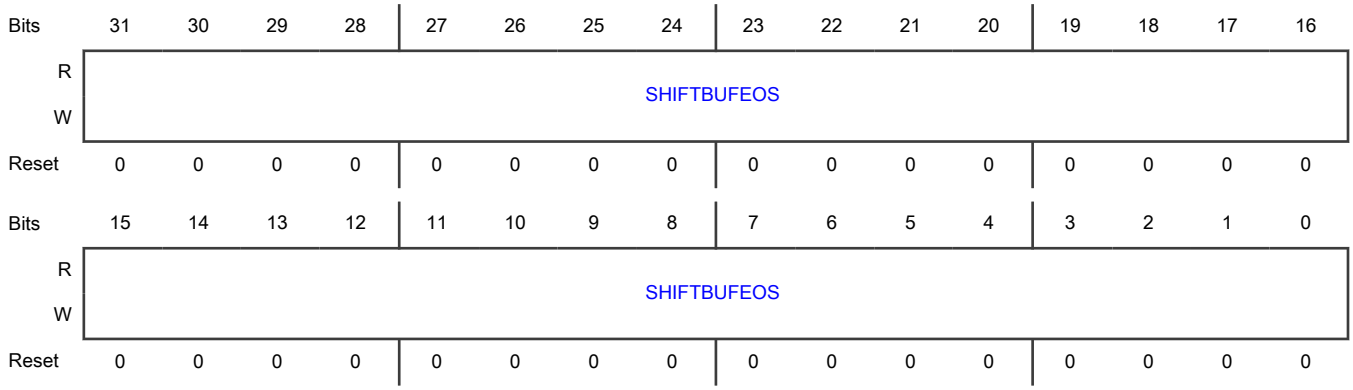
For n = 0 to 7:

Register	Offset
SHIFTBUFEOSn	880h + (n × 4h)

Function

Contains SHIFTBUF register content, but contents have even and odd bits partitioned separately.

Diagram



Fields

Field	Function
31-0 SHIFTBUFEOS	Shift Buffer Alias to SHIFTBUF register, but reads or writes to this register have the even and odd bits partitioned separately. Only 32-bit accesses are supported to this register. Reads return {SHIFTBUF[30], SHIFTBUF[28], SHIFTBUF[26], SHIFTBUF[24], SHIFTBUF[22], SHIFTBUF[20], SHIFTBUF[18], SHIFTBUF[16], SHIFTBUF[14], SHIFTBUF[12], SHIFTBUF[10], SHIFTBUF[8], SHIFTBUF[6], SHIFTBUF[4], SHIFTBUF[2], SHIFTBUF[0], SHIFTBUF[31], SHIFTBUF[29], SHIFTBUF[27],SHIFTBUF[25],SHIFTBUF[23],SHIFTBUF[21],SHIFTBUF[19],SHIFTBUF[17],SHIFTBUF[15],SHIFTBUF[13],SHIFTBUF[11],SHIFTBUF[9], SHIFTBUF[7], SHIFTBUF[5], SHIFTBUF[3], SHIFTBUF[1]}.

60.7.1.41 Shifter Buffer N Halfword Byte Swapped (SHIFTBUFHBS0 - SHIFTBUFHBS7)

Offset

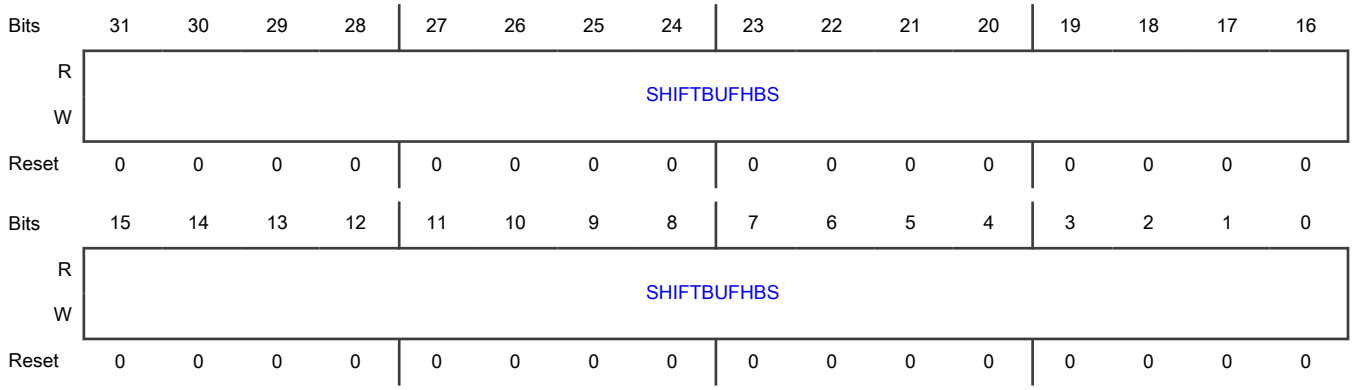
For n = 0 to 7:

Register	Offset
SHIFTBUFHBSn	900h + (n × 4h)

Function

Contains SHIFTBUF register content, but contents are halfword byte-swapped.

Diagram



Fields

Field	Function
31-0 SHIFTBUFHBS	Shift Buffer Alias to SHIFTBUF register, but reads or writes to this register are halfword byte-swapped. Reads return {SHIFTBUF[23:16], SHIFTBUF[31:24], SHIFTBUF[7:0], SHIFTBUF[15:8]}.

Chapter 61
Keypad Port (KPP)

61.1 Chip-specific KPP Information

Table 1007. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

61.2 Overview

The Keypad Port (KPP) is a 16-bit peripheral that can be used as a keypad matrix interface or as general purpose input/output (I/O). The figure below shows the KPP block diagram. The KPP provides interface for the keypad matrix with 2-point contact or 3-point contact keys. The KPP is designed to simplify the software task of scanning a keypad matrix. With appropriate software support, the KPP is capable of detecting, debouncing, and decoding one or multiple keys pressed simultaneously on the keypad.

61.2.1 Block diagram

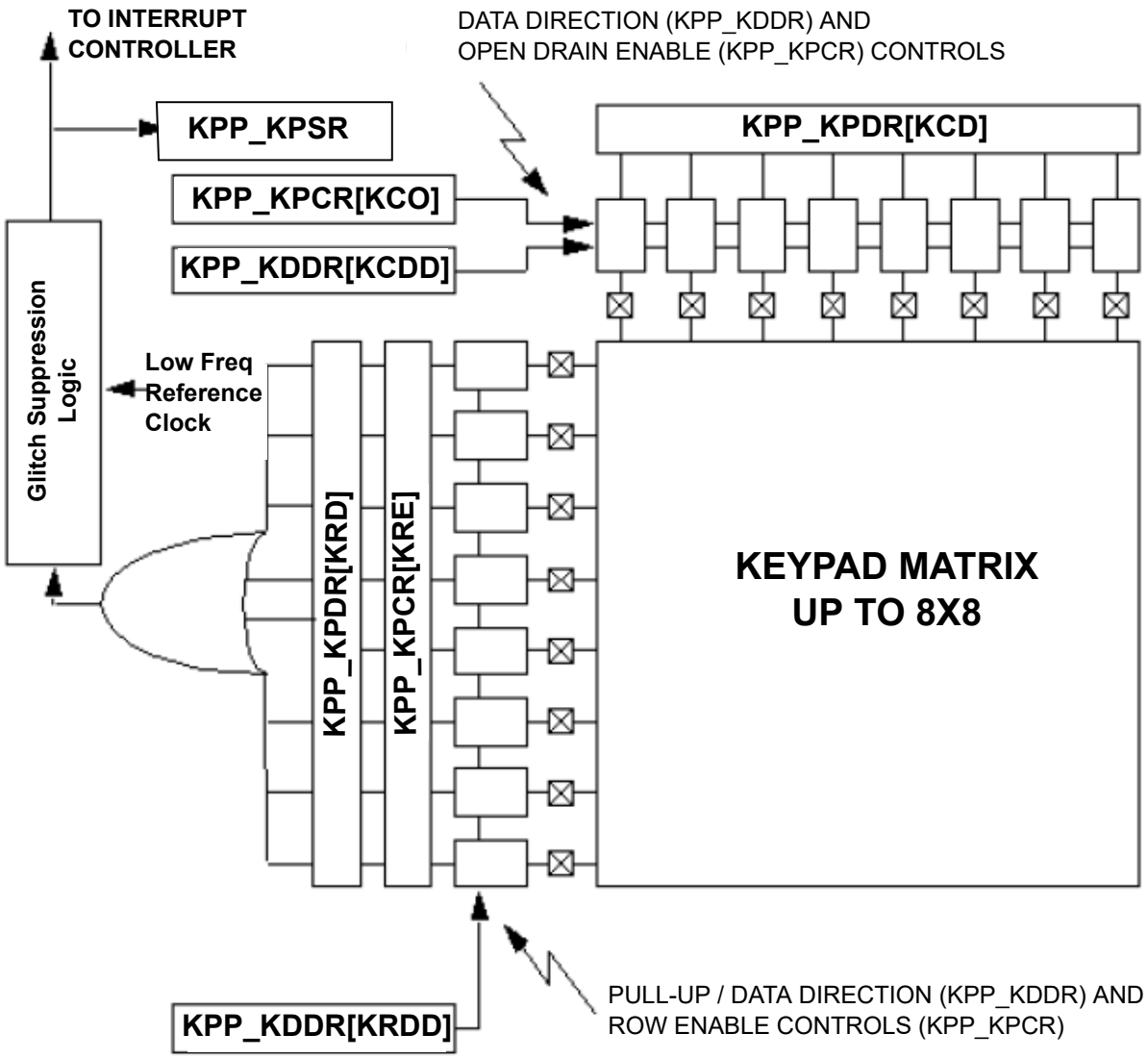


Figure 457. KPP Peripheral Block Diagram

61.2.2 Features

The KPP includes these distinctive features:

- Supports up to an 8 x 8 external key pad matrix
- Port pins can be used as general purpose I/O
- Open drain design
- Glitch suppression circuit design
- Multiple-key detection
- Long key-press detection
- Standby key-press detection

- Synchronizer chain clear
- Supports a 2-point and 3-point contact key matrix

61.3 Functional Description

The Keypad Port (KPP) is designed to simplify the software task of scanning a keypad matrix. With appropriate software support and matrix organization, the KPP is capable of detecting, debouncing, and decoding one or more keys pressed simultaneously on the keypad.

Logic in the KPP is capable of detecting a key press even while the processor is in one of the low power standby modes provided that a low frequency reference clock (ipg_clk_32k) is on. The KPP may generate an ARM platform interrupt any time a key press or key release is detected. This interrupt is capable of forcing the processor out of a low power mode.

61.3.1 Modes and Operations

This block supports the following modes:

- Run Mode-This is the normal functional mode in which the KPP can detect any key press event.
- Low Power Mode-The keypad can detect any key press even in low power modes (when there is no MCU clock).

61.3.2 Keypad Matrix Construction

The KPP is designed to interface to a keypad matrix, which shorts the intersecting row and column lines together whenever a key is depressed. The interface is not optimized for any other switch configuration.

61.3.3 Keypad Port Configuration

The software must initialize the KPP for the size of the keypad matrix. Pins connected to the keypad columns should be configured as open-drain outputs. Pins connected to the keypad rows should be configured as inputs. On-chip, pull-up resistors should be implemented for active keypad rows.

In addition to enabled row inputs in the Keypad Control register, corresponding interrupt (depress or/and release) must also be enabled to generate an interrupt.

Discrete switches that are not part of the matrix may be connected to any unused row inputs. The second terminal of the discrete switch is connected to ground. The hardware detects closures of these switches without the need for software polling.

61.3.4 Keypad Matrix Scanning

Keypad scanning is performed by a software loop that walks a zero across each of the keypad columns, reading the value on the rows at each step. The process is repeated several times in succession, with the results of each pass optionally compared to those from the previous pass. When several consecutive scans yield the same key closures, a valid key press has been detected. Software then can decode exactly which switch was depressed and pass the value up to the next software layer.

The KPP may be used to control the debounce period, but it must be defined in software by the user’s application. The basic period is the period between the scan of two consecutive columns, so the debouncing time between two consecutive scans of the whole matrix shall be the number of columns multiplied by the basic period.

61.3.5 Keypad Standby

There is no need for the ARM platform to continually scan the keypad. Between key presses, the keypad can be left in a state that requires no software intervention until the next key press is detected. To place the keypad in a standby state, software should write all column outputs low. Row inputs are left enabled. At this point, the ARM platform can attend to other tasks or revert to a low power standby mode. The KPP can interrupt the ARM platform if any key is pressed.

Upon receiving a keypad interrupt, the ARM platform should set all the column strobes high, and begin a normal keypad scanning routine to determine which key was pressed. It is important that open-drain drivers be used when scanning to prevent a possible DC path between power and ground through two or more switches.

61.3.6 Glitch Suppression on Keypad Inputs

A glitch suppression circuit qualifies the keypad inputs to prevent noise from inadvertently interrupting the ARM platform. The circuit is a 4-state synchronizer clocked from a low frequency reference clock (ipg_clk_32k) source. This clock must continue to run in any low power mode where the keypad is a wake-up source, as the ARM platform interrupt is generated from the synchronized input. An interrupt is not generated until all four synchronizer stages have latched a valid key assertion. This guarantees the filtering out of any noise less than three clock periods in duration of a low frequency reference clock. Noise filtering of the duration between three to four clock periods cannot be guaranteed. The interrupt output is latched in an S-R latch and remains asserted until cleared by the software. The Set input of the latch is rising-edge clocked. See the figure below.

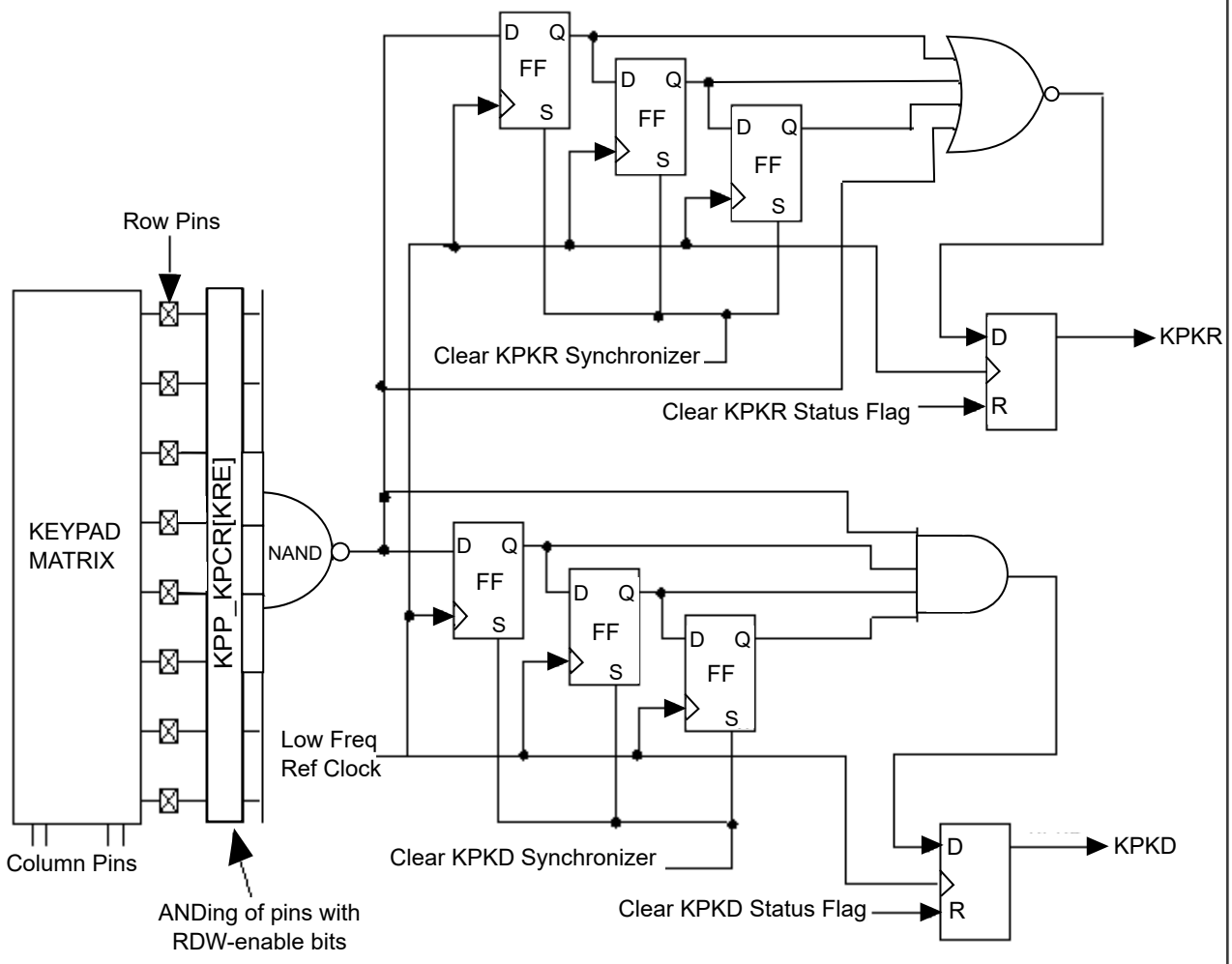


Figure 458. Keypad Synchronizer Functional Diagram

61.3.7 Multiple Key Closures

Using the key press and key release interrupts, the software can detect multiple keys or achieve n key rollover. The key scanning routine can be programmed accordingly (See Initialization/Application Information for more information).

The following figures illustrate the interface of a 2-contact keypad matrix with the KPP controller. With proper enabling of row lines and the performing scan-routine, multiple key presses can be detected. When keys present on the same row are pressed, corresponding row lines (multiple lines) become low when the column is driven low during a scan-routine. By reading the data-register, pressed keys can be detected. Similarly, when keys present on same column line are pressed, the corresponding row line (only one line) becomes low when logic "0" is driven on the column line during a scan-routine.

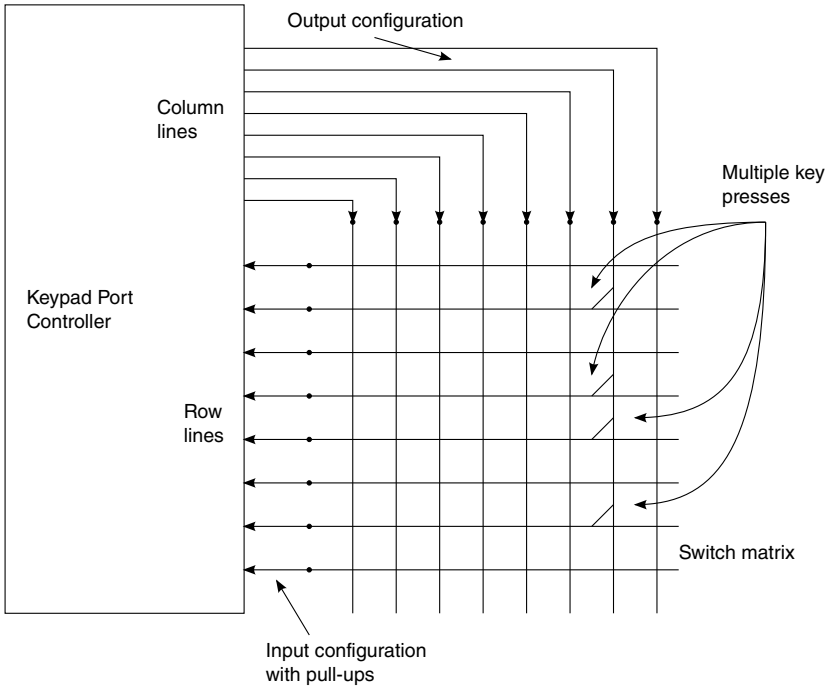


Figure 459. Multiple Key Presses on Same Column Line (Simplified View)

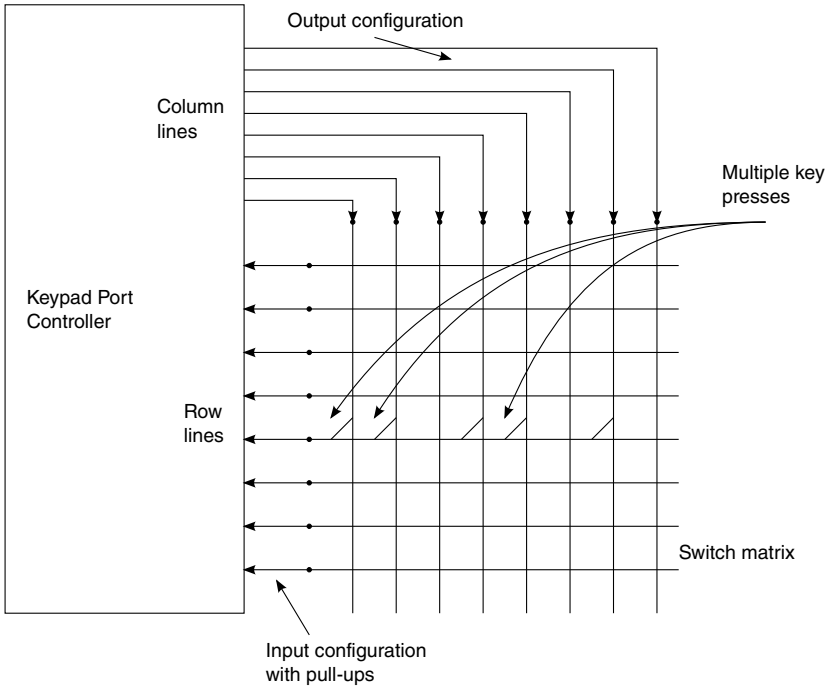


Figure 460. Multiple Key Presses on Same Row Line (Simplified View)

NOTE

An n key rollover is a technique with which the system can recognize the order in which keys are pressed.

61.3.7.1 Ghost Key Problem and Correction

The KPP detects if one or multiple keys are pressed or released. In the case where a simple keypad matrix with two-contact switches is used, there is a chance of "ghost" key detection when three or more keys are pressed. This is a limitation imposed by such a keypad matrix.

As seen in Figure 461, three keys pressed simultaneously can cause a short between the column currently "scanned" by the software and another column. Depending on the location of the third key pressed, a "ghost" key press may be detected.

However, this can be corrected by using a keypad matrix that provides "ghost" key protection. Such a matrix implements a one-way "diode" at all keypad points between rows and columns. This way, the multiple pressing of three keys will not cause a short at a fourth key (see Figure 462).

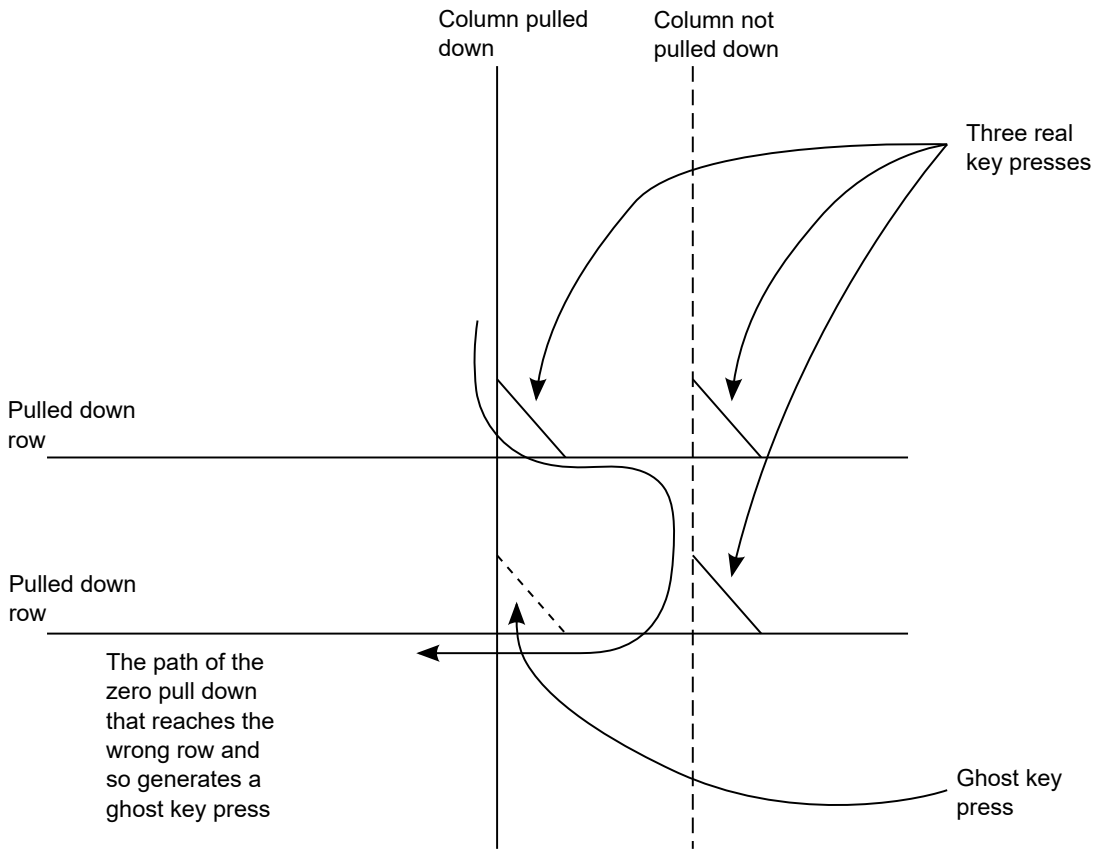


Figure 461. Decoding Wrong Three- Key-Presses

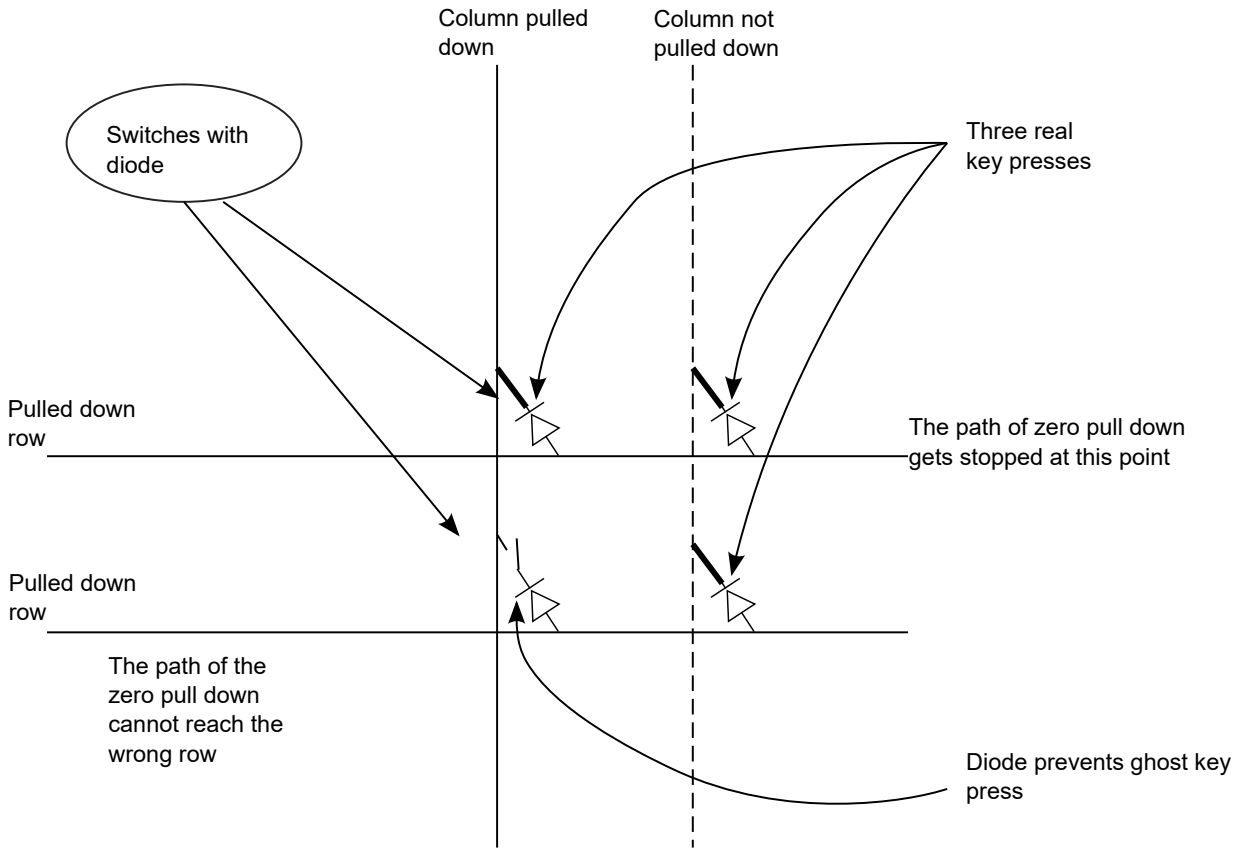


Figure 462. Matrix with "Ghost" Key Protections

61.3.8 3-Point Contact Keys Support

The KPP supports interfacing to a matrix consisting of 3-point contact keys. As shown in Figure 463, two points of such a key are connected to keypad lines, while a third point is connected to ground (low logic).

The keypad lines should be configured as input and a pull-up should to be present on these lines. When such a key is pressed, corresponding keypad lines go low and an interrupt is generated. There is no need to perform a scanning routine for identification of pressed key as it can be done by reading the keypad data-register. A limitation with such a matrix is that for every key at least one keypad row line should be used.

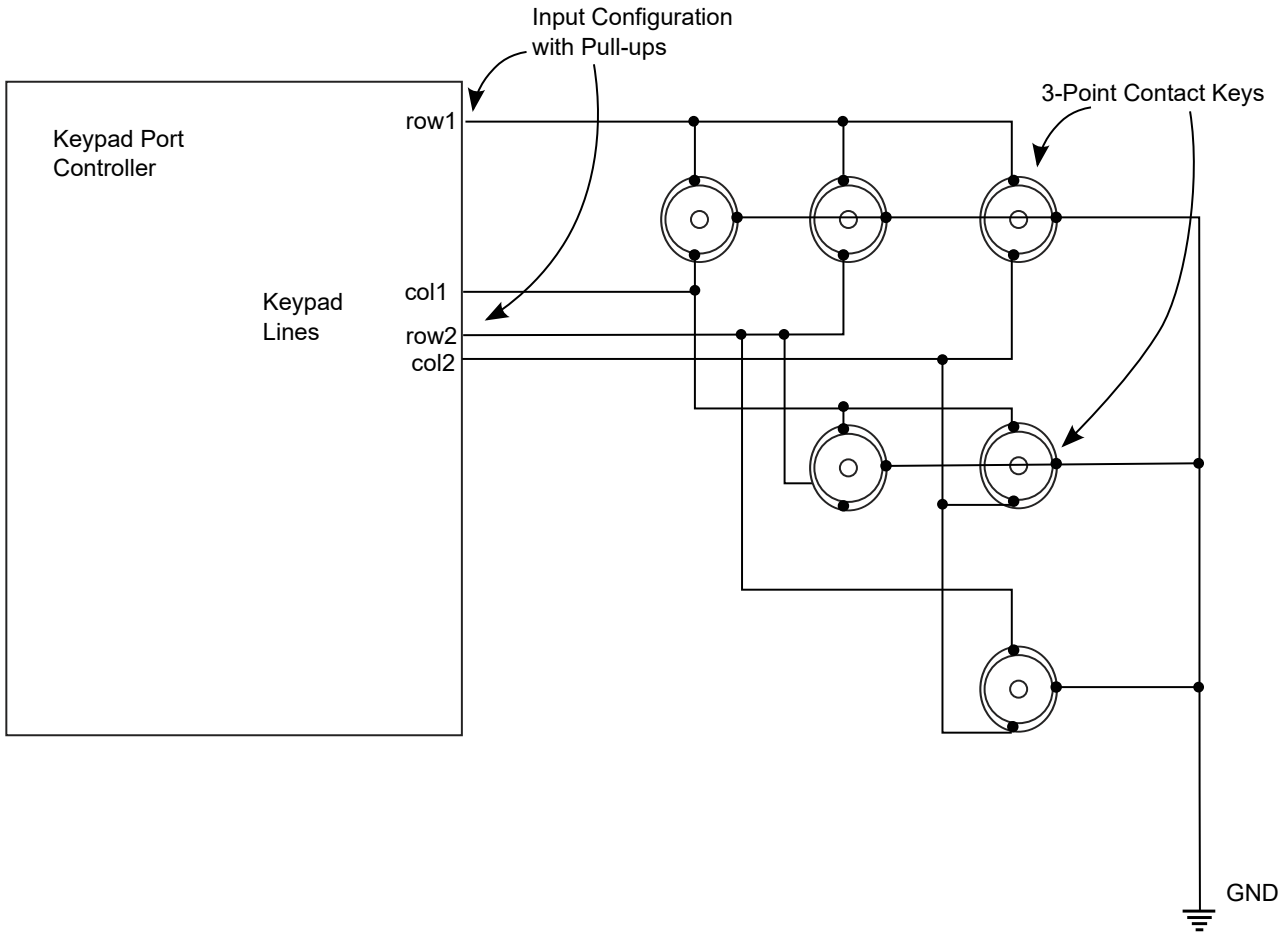


Figure 463. KPP Interface with 3-point Contact Key Matrix (Simplified View)

61.3.9 Clocks

The table found here describes the clock sources for KPP. Please see clock control block for clock setting, configuration and gating information.

Table 1008. KPP Clocks

Clock name	Description
ipg_clk_32k	Low-frequency reference clock (32 kHz)
ipg_clk_s	Peripheral access clock

61.4 External Signals

There are several pins dedicated to the KPP. Keypads of any configuration up to eight rows and eight columns are supported through the software configuration of the peripheral pins. Any pins not used for the keypad are available as general purpose I/O. The registers are configured such that the pins can be treated as an I/O port up to 16 bits wide. Please see IOMUX Controller for KPP pin mux assignments.

See the table below for the list of external signals.

Table 1009. KPP External Signals

Name	Direction	Function	Reset State	Pull-Up
KEY_COL[7:0]	IO	Column input or output pin, from chip	0	Active[1]
KEY_ROW[7:0]	IO	Row input or output pin, from chip	1	Active[1]

1. The corresponding pads are required to be pull-up enabled.

61.4.1 Input Pins

Any of the 16 pins associated with the KPP can be configured as inputs by writing a "0" to the appropriate bits in the KPP_KDDR. Additionally, the least significant 8 bits (ROW inputs) corresponding to KPP_KDDR[KRDD] have internal pull-ups, which are enabled when the pin is used as an input.

61.4.2 Output Pins

Any of the 16 pins associated with the KPP can be configured as outputs by writing the appropriate bits in the KPP_KDDR to a "1". Additionally, the 8 most significant bits (15-8) can be designated as open drain outputs by writing a "1" to the appropriate bits in the KPP_KPCR. The lower 8 bits (7-0) are always in "totem pole" style, driven when configured as outputs. See the table below.

Table 1010. Keypad Port Column Modes

KPP_KDDR (15:8)	KPP_KPCR (15:8)	Pin Function
0	x	Input
1	0	Totem-Pole Output
1	1	Open-Drain Output

NOTE

Totem pole configuration helps for a faster discharge of keypad capacitance when all columns need to be quickly brought to a "1" during the scan routine. With this configuration, delay between the scanning of two subsequent columns is reduced. Totem pole mode is to be used temporarily on the column pins to discharge the lines only, but should be switched to open-drain mode for normal operation.

61.4.3 Generation of Transfer Error Signal on Peripheral Bus

If there is an access to an address which is not implemented, then the KPP asserts a transfer error signal on Peripheral Bus.

61.5 Initialization/Application Information

61.5.1 Typical Keypad Configuration and Scanning Sequence

Perform the following steps to configure the keypad:

1. Enable the number of rows in the keypad (KPP_KPCR[KRE]).
2. Write 0s to KPP_KPDR[KCD].
3. Configure the keypad columns as open-drain (KPP_KPCR[KCO]).
4. Configure columns as output (KPP_KDDR[KCDD]) and rows as input (KPP_KDDR[KRDD]).

5. Clear the KPKD Status Flag and Synchronizer chain.
6. Set the KDIE control bit, and clear the KRIE control bit (avoid false release events).
7. (The system is now in standby mode, and awaiting a key press.)

61.5.2 Key Press Interrupt Scanning Sequence

Perform the following steps to perform a keypad scanning routine:

1. Disable both (depress and release) keypad interrupts.
2. Write 1s to KPP_KPDR[KCD], setting column data to 1s.
3. Configure columns as totem pole outputs (for quick discharging of keypad capacitance).
4. Configure columns as open-drain.
5. Write a single column to 0, and other columns to 1.
6. Sample row inputs and save data. Multiple key presses can be detected on a single column.
7. Repeat Steps 2-6 for remaining columns.
8. Return all columns to 0 in preparation for standby mode.
9. Clear KPKD and KPKR status bit(s) by writing to a "1"; set the KPKR synchronizer chain by writing a "1" to the KPP_KRSS register; and clear the KPKD synchronizer chain by writing a "1" to the KDSC register.
10. Re-enable the appropriate keypad interrupt(s) so that the KDIE detects a key hold condition, or the KRIE detects a key-release event.

61.5.3 Additional Comments

The order of key press detection can be done in software only. Therefore, the software may need to run the scan routines at very short intervals of time per the application's demands. The reason that such functionality cannot be put in the KPP is that the block is limited by the number of external pins.

For the keys that require a very precise order (such as game keys), individual GPIO pins may be more useful.

61.6 Memory Map and Register Definition

61.6.1 KPP register descriptions

The KPP contains four registers.

61.6.1.1 KPP memory map

KPP base address: 42A0_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Keypad Control Register (KPCR)	16	RW	0000h
2h	Keypad Status Register (KPSR)	16	RW	0002h
4h	Keypad Data Direction Register (KDDR)	16	RW	0000h
6h	Keypad Data Register (KPDR)	16	RW	0000h

61.6.1.2 Keypad Control Register (KPCR)

Offset

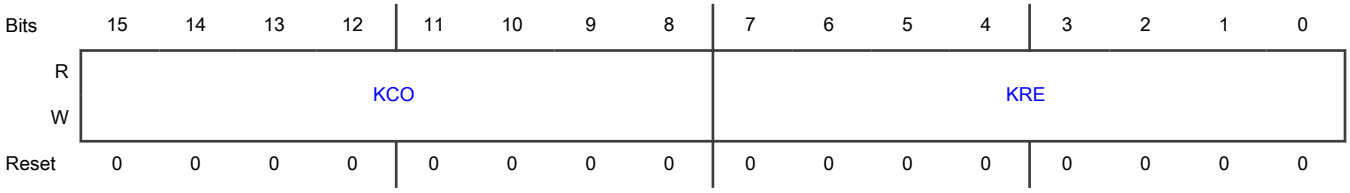
Register	Offset
KPCR	0h

Function

The Keypad Control Register determines which of the eight possible column strobes are to be open drain when configured as outputs, and which of the eight row sense lines are considered in generating an interrupt to the core.

It is up to the programmer to ensure that pins being used for functions other than the keypad are properly disabled. The KPP_KPCR register is byte- or half-word-addressable.

Diagram



Fields

Field	Function
15-8 KCO	KCO Keypad Column Strobe Open-Drain Enable. Setting a column open-drain enable bit (KCO7-KCO0) disables the pull-up driver on that pin. Clearing the bit allows the pin to drive to the high state. This bit has no effect when the pin is configured as an input. **NOTE** Configuration of external port control logic (for example, IOMUX) should be done properly so that the KPP controls an open-drain enable of the pin. 0000_0000b - Column strobe output is totem pole drive. 0000_0001b - Column strobe output is open drain.
7-0 KRE	KRE Keypad Row Enable. Setting a row enable control bit in this register enables the corresponding row line to participate in interrupt generation. Likewise, clearing a bit disables that row from being used to generate an interrupt. This register is cleared by a reset, disabling all rows. The row-enable logic is independent of the programmed direction of the pin. Writing a "0" to the data register of the pins configured as outputs will cause a keypad interrupt to be generated if the row enable associated with that bit is set. 0000_0000b - Row is not included in the keypad key press detect. 0000_0001b - Row is included in the keypad key press detect.

61.6.1.3 Keypad Status Register (KPSR)

Offset

Register	Offset
KPSR	2h

Function

The Keypad Status Register reflects the state of the key press detect circuit. The KPP_KPSR register is byte- or half-word-addressable.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						KRIE	KDIE	0				0	0	KPKR	KPKD
W													KRSS	KDSC	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
15-10 —	- Reserved
9 KRIE	KRIE Keypad Release Interrupt Enable. The software should ensure that the interrupt for a Key Release event is masked until it has entered the key pressed state, and vice versa, unless this activity is desired (as might be the case when a repeated interrupt is to be generated). The synchronizer chains are capable of being initialized to detect repeated key presses or releases. If they are not initialized when the corresponding event flag is cleared, false interrupts may be generated for depress (or release) events shorter than the length of the corresponding chain. 0b - No interrupt request is generated when KPKR is set. 1b - An interrupt request is generated when KPKR is set.
8 KDIE	KDIE Keypad Key Depress Interrupt Enable. Software should ensure that the interrupt for a Key Release event is masked until it has entered the key pressed state, and vice-versa, unless this activity is desired (as might be the case when a repeated interrupt is to be generated). The synchronizer chains are capable of being initialized to detect repeated key presses or releases. If they are not initialized when the corresponding event flag is cleared, false interrupts may be generated for depress (or release) events shorter than the length of the corresponding chain. 0b - No interrupt request is generated when KPKD is set. 1b - An interrupt request is generated when KPKD is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 —	- Reserved, should be cleared
3 KRSS	KRSS Key Release Synchronizer Set. Self-clear bit. The Key release synchronizer is set by writing a logic one into this bit. Reads return a value of "0". 0b - No effect 1b - Set bits which sets keypad release synchronizer chain
2 KDSC	KDSC Key Depress Synchronizer Clear. Self-clear bit. The Key depress synchronizer is cleared by writing a logic "1" into this bit. Reads return a value of "0". 0b - No effect 1b - Set bits that clear the keypad depress synchronizer chain
1 KPKR	KPKR Keypad Key Release. The keypad key release (KPKR) status bit is set when all enabled rows are detected high after synchronization (the KPKR status bit will be set when cleared by a reset). The KPKR bit may be used to generate a maskable key release interrupt. The key release synchronizer may be set high by software after scanning the keypad to ensure a known state. Due to the logic function of the release and depress synchronizer chains, it is possible to see the re-assertion of a status flag (KPKD or KPKR) if it is cleared by software prior to the system exiting the state it represents. Reset value of register is "0" as long as reset is asserted. However when reset is de-asserted, the value of the register depends upon the external row pins and can become "1". 0b - No key release detected 1b - All keys have been released
0 KPKD	KPKD Keypad Key Depress. The keypad key depress (KPKD) status bit is set when one or more enabled rows are detected low after synchronization. The KPKD status bit remains set until cleared by the software. The KPKD bit may be used to generate a maskable key depress interrupt. If desired, the software may clear the key press synchronizer chain to allow a repeated interrupt to be generated while a key remains pressed. In this case, a new interrupt will be generated after the synchronizer delay (4 cycles of the low frequency reference clock (ipg_clk_32k) elapses if a key remains pressed. This functionality can be used to detect a long key press. This allows detection of additional key presses of the same key or other keys. Due to the logic function of the release and depress synchronizer chains, it is possible to see the re-assertion of a status flag (KPKD or KPKR) if it is cleared by the software prior to the system exiting the state it represents. 0b - No key presses detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - A key has been depressed

61.6.1.4 Keypad Data Direction Register (KDDR)

Offset

Register	Offset
KDDR	4h

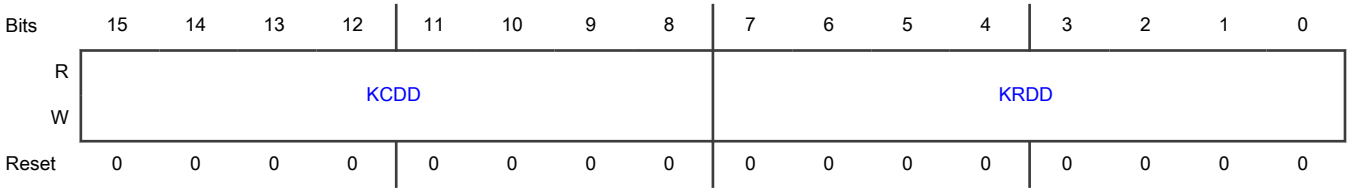
Function

The bits in the KPP_KDDR control the direction of the keypad port pins. The upper eight bits in the register affect the pins designated as column strobes, while the lower eight bits affect the row sense pins. Setting any bit in this register configures the corresponding pin as an output. Clearing any bit in this register configures the corresponding port pin as an input. For the Keypad Row DDR, an internal pull-up is enabled if the corresponding bit is clear. This register is cleared by a reset, configuring all pins as inputs. The KPP_KDDR register is byte- or half-word addressable.

NOTE

When a pin is used as row pin for keypad purposes, all corresponding pull-ups should be enabled at the upper level (for example, IOMUX) when the bit in KRDD is cleared.

Diagram



Fields

Field	Function
15-8 KCDD	KCDD Keypad Column Data Direction Register. Setting a bit configures the corresponding COL*n* pin as an output (where *n* = 7 through 0). 0000_0000b - COLn pin is configured as an input. 0000_0001b - COLn pin is configured as an output.
7-0 KRDD	KRDD Keypad Row Data Direction. Setting a bit configures the corresponding ROW*n* pin as an output (where *n* = 7 through 0).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0000_0000b - ROWn pin configured as an input. 0000_0001b - ROWn pin configured as an output.

61.6.1.5 Keypad Data Register (KPDR)

Offset

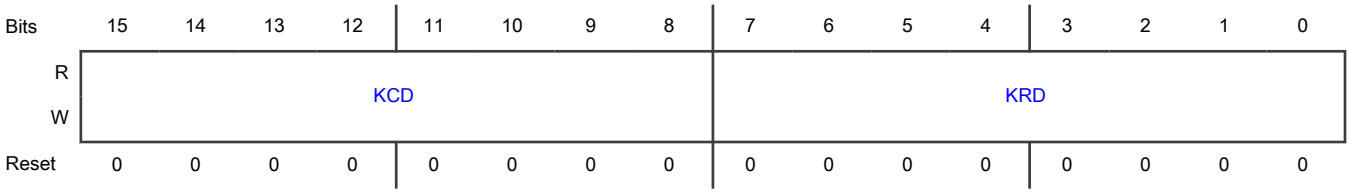
Register	Offset
KPDR	6h

Function

This 16-bit register is used to access the column and row data. Data written to this register is stored in an internal latch, and for each pin configured as an output, the stored data is driven onto the pin. A read of this register returns the value on the pin for those bits configured as inputs. Otherwise, the value read is the value stored in the register.

The KPP_KPDR register is byte- or half-word addressable. This register is not initialized by a reset. Valid data should be written to this register before any bits are configured as outputs.

Diagram



Fields

Field	Function
15-8 KCD	KCD Keypad Column Data. A read of these bits returns the value on the pin for those bits configured as inputs. Otherwise, the value read is the value stored in the register. 0 Read/Write "0" from/to column ports 1 Read/Write "1" from/to column ports
7-0 KRD	KRD Keypad Row Data. A read of these bits returns the value on the pin for those bits configured as inputs. Otherwise, the value read is the value stored in the register. 0 Read/Write "0" from/to row ports 1 Read/Write "1" from/to row ports

Chapter 62
Low Power Inter-Integrated Circuit (LPI2C)

62.1 Chip-specific LPI2C Information

Table 1011. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

NOTE

The LPI2C output triggers are not connected.

62.2 Overview

LPI2C supports an efficient interface to an I2C bus as a controller and target:

- Implements logic support for Standard, Fast, Fast+, HS-mode (target only) and Ultra-Fast modes of operation
- Uses little CPU overhead, with DMA offloading of FIFO register accesses
- Continues operating in Stop modes if an appropriate clock is available

LPI2C also complies with the System Management Bus (SMBus) Specification, version 3. The SMBus is a single-ended simple two-wire bus, which is typically used for low-bandwidth communications.

The Inter-Integrated Circuit (I^2C) serial bus is multi-controller, multi-target, packet-switched, and single-ended, and is often used to attach microcontroller ICs to lower-speed peripheral ICs.

NOTE

Terminology in this chapter has been updated to align with I²C-bus specification, Rev. 7.0, as shown in Table 1012.

Table 1012. Updated terms

Updated term	Deprecated term
Controller	Master
Target	Slave

62.2.1 Block diagram

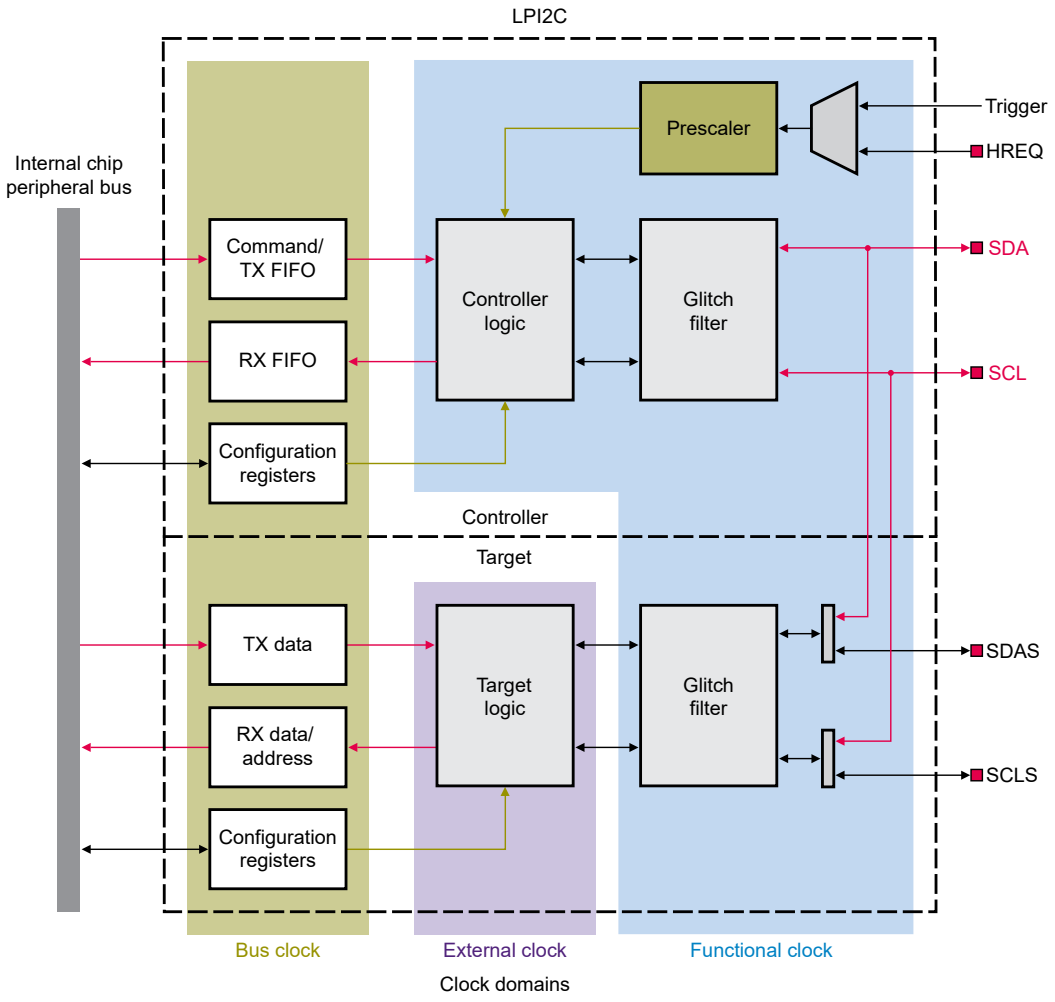


Figure 464. Block diagram

62.2.2 Features

LPI2C supports:

- Standard, Fast, Fast+ and Ultra Fast modes
- HS mode in target mode
- Multicontroller, including synchronization and arbitration, means that any number of controller nodes can be present. Also, controller and target roles can be changed between messages (after a Stop signal is sent).

- Clock stretching. Used on the SCL line, as an I2C flow control mechanism.
- Arbitration for when the system has more than one controller. When used on the SDA line, ensures that there is only one I2C transmitter at a time.
- General call, seven-bit addressing, and ten-bit addressing
- Software reset, Start byte, and device ID (also require software support)

The LPI2C controller supports:

- Command and transmit FIFO of 8 words (8-bit transmit data + 3-bit command)
- Receive FIFO of 8 words (8-bit receive data).
- Command FIFO waiting for an I2C idle bus before initiating a transfer
- Initiation of repeated Start and Stop conditions and one or more controller-receiver transfers by command FIFO
- Stop condition generation from command FIFO, or automatic generation of Stop condition when the transmit FIFO is empty
- Host request input to control the start time of an I2C bus transfer
- Interrupt generation on data match and unwanted data rejection, via flexible receive data match
- Flags and optional interrupt signals at repeated Start condition, Stop condition, loss of arbitration, unexpected NACK, and command word errors
- Configurable bus idle timeout and pin-stuck-low timeout

The LPI2C target supports:

- Separate I2C target registers to minimize software overhead because of controller or target switching
- 7-bit or 10-bit addressing, address range, SMBus alert, and general call address.
- Transmit data register that supports interrupt or DMA requests
- Receive data register that supports interrupt or DMA requests
- Software-controllable ACK or NACK, with optional clock stretching on ACK or NACK field
- Configurable clock stretching, to avoid transmit-FIFO-underrun and receive-FIFO-overrun errors
- Flags and optional interrupt at end of packet, Stop condition, or bit error detection

62.3 Functional description

62.3.1 Controller mode

The LPI2C controller logic operates independently from the target logic to perform all controller-mode transfers on the I2C bus.

62.3.1.1 Transmit and Command FIFO commands

The transmit FIFO stores command data to initiate various I2C operations. The following operations can be initiated through commands in the transmit FIFO:

- Start or repeated Start condition with address byte, expecting ACK or NACK.
- Transmit data. This operation is the default for zero-extended-byte writes to the transmit FIFO.
- Receive 1-256 bytes of data. You can configure this operation to discard received data and not to store it in the receive FIFO.
- Stop condition. You can configure this operation to send a Stop condition when the transmit FIFO is empty.

Multiple transmit and receive commands can be inserted between the Start and Stop conditions. To comply with the I2C specification, transmit and receive commands must not be interleaved. The receive data command and the receive data and

discard commands can be interleaved. This interleaving ensures that only the desired received data is stored in the receive FIFO (or compared with the data match logic).

The LPI2C controller automatically transmits a NACK on the last byte of a receive data command. It transmits the NACK unless the next command in the FIFO is also a receive data command. If the transmit FIFO is empty when a receive data command completes, a NACK is also automatically transmitted.

The LPI2C controller supports 10-bit addressing via a (repeated) Start condition, followed by a transmit data byte containing the second address byte, followed by any number of data bytes with the controller transmit data.

A Start or repeated Start condition expecting a NACK (for example, HS mode controller code) must be followed by a Stop or (repeated) Start condition.

62.3.1.2 Controller operations

When LPI2C is enabled, it monitors the I2C bus to detect when the I2C is idle (MSR[BBF]). If either SCL or SDA are low, the I2C bus is no longer considered idle. The bus becomes idle if a Stop condition is detected or if a bus idle timeout is detected (as configured by MCFGR2[BUSIDLE]). After the bus is idle, if the transmit FIFO is not empty and the host request is asserted or disabled, the LPI2C controller initiates a transfer on the bus. This transfer involves the following steps:

1. Wait the bus idle time equal to (MCCR0[CLKLO] + 1) multiplied by the prescaler (MCFGR1[PRESCALE]).
2. Transmit a Start condition and address byte using the timing configuration in Controller Clock Configuration 0 (MCCR0). If an HS mode transfer is configured, the timing configuration from Controller Clock Configuration 1 (MCCR1) is used instead.
3. Perform controller transmit or controller receive transfers, as configured by the transmit FIFO.
4. Transmit NACK on the last byte of a controller receive transfer. This action is performed unless the next command in the transmit FIFO is also a receive data command and the transmit FIFO is not empty.
5. Transmit a repeated Start or Stop condition as configured by the transmit FIFO or MCFGR1[AUTOSTOP]. A repeated Start can change which timing configuration register is used.

When MCFGR0[RELAX] is 1, the LPI2C controller does not wait for the I2C bus to be idle before starting a transfer. It also relaxes some restrictions on the order of FIFO commands. For example, it allows a Stop command when it is idle to generate a Start condition, followed by one SCL clock pulse, and then a Stop condition.

When the LPI2C controller is disabled, LPI2C continues emptying the transmit FIFO until a Stop condition is transmitted. (The controller could be disabled due to MCR[MEN] being 0, or automatically due to mode entry.) However, LPI2C no longer stalls the I2C bus by waiting for the transmit or receive FIFO. After the transmit FIFO is empty, LPI2C generates a Stop condition automatically.

The LPI2C controller can stall the I2C bus under certain conditions. This stalling results in SCL pulled low continuously on the first bit of a byte, until these conditions change:

- The LPI2C controller is enabled and busy, the transmit FIFO is empty, and MCFGR1[AUTOSTOP] is 0. The LPI2C controller continues to stall the bus until the transmit FIFO is loaded with more data.
- The LPI2C controller is enabled and receiving data, receive data is not being discarded (due to command or receive data match), and the receive FIFO is full. The LPI2C controller continues to stall the I2C bus until the receive FIFO is emptied.

The LPI2C controller aborts the existing transfer when MCFGR0[ABORT] becomes 1. This action causes the LPI2C controller to complete the existing word and then transmit a Stop condition. If the receive FIFO is full when the receive operation is aborted, the last received data is discarded. A new transfer does not start while MCFGR0[ABORT] is 1.

62.3.1.3 Receive FIFO and data matching

The receive FIFO stores receive data during controller-receiver transfers. You can configure the LPI2C controller to discard received data instead of storing it in the receive FIFO. This option is configured via the command word in the transmit FIFO.

Received data supports a receive data match function that can match received data against one of two bytes, or against a masked data byte. You can configure the data match function to compare only the first one or two data words received since the last

(repeated) Start condition. Received data that is already discarded due to the command word cannot cause a data match. It delays the match on the first data word received until after the discarded data is received.

You can configure the receiver match function to discard all received data until a data match is detected, using MCFGR0[RDMO]. Following a data match, write 0 to MCFGR0[RDMO] before writing 0 to MSR[DMF] to allow all subsequent data to be received.

62.3.1.4 Timing parameters

The LPI2C controller can configure the following timing parameters. Parameters are configured separately for HS mode (Controller Clock Configuration 1 (MCCR1)) and other modes (Controller Clock Configuration 0 (MCCR0)). This separation allows the HS mode controller code to be sent using regular timing parameters. Then it allows a switch to HS mode timing (following a repeated Start) until the next STOP condition.

Configure the LPI2C controller timing parameters, measured in LPI2C functional clock cycles, as shown in Table 1013. You must configure these parameters to meet the I2C timing specification for the required mode.

Table 1013. Timing parameters

I2C specification timing parameter	I2C specification timing symbol	LPI2C timing parameter (in LPI2C functional clock cycles)
SCL clock period	tSCL	(CLKHI + CLKLO + 2 + SCL_LATENCY) × (2 ^ PRESCALE)
Hold time (repeated) Start condition	tHD:STA	(SETHOLD +1) × (2 ^ PRESCALE)
Low period of the SCL clock	tLOW	(CLKLO + 1) × (2 ^ PRESCALE)
High period of the SCL clock	tHIGH	(CLKHI + 1 + SCL_LATENCY) × (2 ^ PRESCALE)
Setup time for a repeated Start condition or Stop condition	tSU:STA, tSU:STO	(SETHOLD + 1 + SCL_LATENCY) × (2 ^ PRESCALE)
Data hold time	tHD:DAT	(DATAVD + 1) × (2 ^ PRESCALE)
Data setup time	tSU:DAT	(SDA_LATENCY + 1) × (2 ^ PRESCALE)
Bus free time between a Stop and Start condition	tBUF	(CLKLO + 1 + SDA_LATENCY) × (2 ^ PRESCALE)
Data valid time, data valid acknowledge time	tVD:DAT, tVD:ACK	(DATAVD + 1) × (2 ^ PRESCALE)

Table 1014 defines the latency parameters. These parameters assume that the risetime is less than one LPI2C functional clock cycle. The risetime depends on a number of factors, including the I/O propagation delay, the I2C bus loading, and the external pullup resistor sizing. A larger risetime increases the number of cycles that the signal takes to propagate through the synchronizer (and glitch filter), which increases the latency.

Table 1014. Synchronization latency

Timing parameter	Timing definition
SCL_LATENCY	ROUNDDOWN ((2 + FILTSCL + SCL_RISETIME) ÷ (2 ^ PRESCALE))
SDA_LATENCY	ROUNDDOWN ((2 + FILTSDA + SDA_RISETIME) ÷ (2 ^ PRESCALE))

The following timing restrictions must be enforced to avoid unexpected Start or Stop conditions on the I2C bus. These restrictions also avoid unexpected Start or Stop conditions detected by the LPI2C controller. The timing restrictions can be summarized as "SDA cannot change when SCL is high outside a transmitted (repeated) Start or Stop condition."

Table 1015. LPI2C timing parameter restrictions

Timing parameter	Minimum	Maximum	Comment
CLKLO	03h	—	CLKLO x (2 ^ PRESCALE) > SCL_LATENCY
CLKHI	01h	—	Configure CLKHI to meet the duty cycle requirements in the I2C specification
SETHOLD	02h	—	SETHOLD × (2 ^ PRESCALE) > SDA_LATENCY
DATAVD	01h	CLKLO – SDA_LATENCY – 1	Configure DATAVD to meet the data hold requirement in the I2C specification
FILTSCL	00h	[CLKLO × (2 ^ PRESCALE)] – 3	FILTSCL and FILTSDA are the only parameters not multiplied by (2 ^ PRESCALE)
FILTSDA	FILTSCL	[CLKLO × (2 ^ PRESCALE)] – 3	Configuring FILTSDA greater than FILTSCL can delay the SDA input to compensate for board level skew
BUSIDLE	(CLKLO + SETHOLD + 2) × 2	—	Must also be greater than (CLKHI + 1)

See the UM10204, *I2C-bus specification and user manual*.

See Application information for example LPI2C timing configurations.

62.3.1.5 Error conditions

The LPI2C controller monitors errors while it is active. The following conditions generate an error flag and block a new Start condition from being sent, until the flag is cleared by software:

- A Start or Stop condition is detected and is not generated by the LPI2C controller (MSR[ALF] becomes 1).
- Transmitting data on SDA and different values are received (MSR[ALF] becomes 1). LPI2C controller stops driving SDA immediately, but continues to drive SCL for the remainder of the byte.
- NACK is detected when transmitting data, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- NACK is detected and is expecting ACK for the address byte, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- ACK is detected and is expecting NACK for the address byte, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- Transmit FIFO is requesting to transmit or receive data without a Start condition (MSR[FEF] becomes 1). This restriction is ignored when MCFGR0[RELAX] is 1.
- SCL (or SDA if MCFGR1[TIMECFG] is 1) is low for (MCFGR2[TIMELOW] × 256) prescaler cycles without a pin transition (MSR[PLTF] becomes 1).

You must respond to MSR[PLTF] to terminate the existing command. You can terminate the command cleanly by writing 0 to MCR[MEN], or you can terminate it abruptly by writing 1 to MCR[RST].

You can use MCFGR0[RELAX] to attempt to recover a target with SDA stuck low. If MCFGR0[RELAX] is 1, the LPI2C controller does not wait for the I2C bus to be idle before starting a transfer. Initiating a Start command with address FFh, then the Stop condition, should generate sufficient SCL clock edges to cause the target to release SDA.

You can use MCFGR2[BUSIDLE] to force the I2C bus to be considered idle when SCL and SDA remain high for (BUSIDLE + 1) prescaler cycles. The bus is considered idle when the LPI2C controller is first enabled. When BUSIDLE is configured greater than zero, then SCL or SDA must be high for (BUSIDLE + 1) prescaler cycles before the I2C bus is considered idle.

62.3.1.6 DMA support

For efficient DMA transfers to the transmit and command FIFO, two alias regions support incrementing 8-bit, 16-bit, or 32-bit write accesses to that FIFO.

- Controller Transmit Command Burst (MTCBR0 - MTCBR127) is a 512-byte region that supports pushing CMD and DATA into the transmit FIFO.
- Transmit Data Burst (MTDBR0 - MTDBR255) is a 1024-byte region that supports pushing zero extended DATA into the transmit FIFO.

The two regions are contiguous. A DMA transfer can start in the MTCBR region to initialize the transfer including START condition with address. The same transfer can complete in the MTDBR region with the data to transmit without changing the transfer size.

The transmit FIFO prevents writes that overflow the FIFO, but this prevention does not signal an error. Do not perform 32-bit writes to the MTDBR unless there are four empty slots in the transmit FIFO. Do not perform 16-bit writes to the MTDBR unless there are two empty slots in the transmit FIFO.

62.3.1.7 Pin configuration

Configuration	Description
Open-drain support	The LPI2C controller defaults to open-drain configuration of the SDA and SCL pins. Support for true open drain depends on the specific device, and requires the pins where LPI2C pins are muxed to support true open drain.
HS mode support	Support for HS mode depends on the specific device. This mode requires the SCL pin to support the current source pullup required in the I2C specification.
Ultra-Fast mode support	The LPI2C controller supports the output-only push-pull function required for I2C Ultra-Fast mode using the SDA and SCL pins. Support for Ultra-Fast mode also requires MCFGR1[IGNACK] to be 1.
Push-pull two-wire support	A push-pull two-wire configuration is available to the LPI2C controller. If LPI2C is the only controller and all I2C pins on the bus are at the same voltage, this configuration may support a partial HS mode. A partial HS mode, not a full HS mode, because this configuration actively drives high rather than enabling a current service pull-up. This configuration sets the SCL pin as push-pull for every clock except the ninth clock pulse, to allow HS-mode-compatible targets to perform clock stretching. In this mode, the SDA pin is tristated for controller-receive data bits and controller-transmit ACK/NACK bits, and is configured as push-pull at other times. To avoid the risk of contention when SDA is push-pull, configure the pin for open-drain operation, as part of the device-specific configuration.
Push-pull four-wire support	The push-pull four-wire configuration separates the SCL input data and output data into separate pins. It also separates the SDA input data and output data into separate pins. The SCL/SDA pins are used for input data; the SCLS/SDAS pins are used for output data, with configurable polarity. This configuration simplifies external connections when connecting the LPI2C to the I2C bus through external level shifters or discrete components. When using this four-wire configuration, the LPI2C controller logic and LPI2C target logic cannot connect to separate I2C buses.

62.3.2 Target mode

To perform all target mode transfers on the I2C bus, the LPI2C target logic operates independently from the LPI2C controller logic.

62.3.2.1 Address matching

You can configure the LPI2C target:

- To match one of two addresses, using either 7-bit or 10-bit addressing modes for each address.

- To match a range of addresses in either 7-bit or 10-bit addressing modes.
- To match the general call address and generate appropriate flags.
- To match the SMBus alert address and generate appropriate flags.
- To detect the HS mode controller code address, and to disable the digital filters and output valid delay time until the next Stop condition is detected.

After a valid address is matched, the LPI2C target automatically performs target-transmit or target-receive transfers until:

- A NACK is detected (unless SCFGR1[IGNACK] becomes 1).
- A bit error is detected (the LPI2C target is driving SDA, but a different value is sampled).
- A (repeated) Start or Stop condition is detected.

When SCFGR1[RXALL] is 1, the LPI2C target receives all addresses and data on the I2C bus. Unless the address is matched, it never drives the SDA pin. Configure the LPI2C target to match only on 7-bit address mode when this feature is enabled. Software can support 10-bit address mode. This option is intended for debugging the contents of an I2C transfer between another controller and target. Likewise, SCFGR1[SDCFG] and SCFGR1[RSCFG] configure the Stop or repeated Start flags to assert on all Stop or repeated Start conditions, even when there is no address match.

62.3.2.2 Transmit and receive data

Target Transmit Data (STDR) and Target Receive Data (SRDR) are double-buffered and only update during a target-transmit and target-receive transfer, respectively.

You can configure the target address that was received to be read from SRDR (for example, when using DMA to transfer data) or from Target Address Status (SASR).

You can configure STDR to request data only after a target-transmit transfer is detected. You can also configure it to request new data whenever STDR is empty.

Write to STDR only when SSR[TDF] is set.

Read SRDR only when SSR[RDF] is set, or when SSR[AVF] is set and SCFGR1[RXCFG] = 1.

Read SASR only when SSR[AVF] is set.

62.3.2.3 Read request

The LPI2C target generates a read request when the I2C bus is idle and you write 1 to SCFGR0[RDREQ]. This occurrence causes the LPI2C target to pull the SDA pin low until either the first SCL falling edge or SCFGR0[RDREQ] becomes 0. Following the next Stop condition or a software timeout, you can proceed as follows:

- If the LPI2C target is accessed at the correct address, the read request is acknowledged, and you must write 0 to SCFGR0[RDREQ].
- If SCFGR0[RDACK] is 1, the read request is acknowledged by a Start followed by one SCL pulse followed by a STOP condition. You must write 0 to SCFGR0[RDREQ].
- If neither of the first two cases occurred, the read request is not acknowledged. You must write 0 and then 1 to SCFGR0[RDREQ] after an appropriate delay, provided the I2C bus is idle.
- If a software timeout occurs, the read request is not acknowledged, and you should write 0 to SCFGR0[RDREQ]. Another request can be generated after an appropriate delay.

Writing to SCFGR0[RDREQ] when the I2C bus is busy results in a logic 0 being written. Write 1 to SCFGR0[RDREQ], then confirm that the register is written to correctly. If the register does not update correctly, the I2C bus is no longer idle and the read request must be attempted later.

62.3.2.4 Clock stretching

The LPI2C target supports many configurable options for clock stretching. You can configure these conditions to perform clock stretching:

- SSR[AVF] is set during the ninth clock pulse of the address byte.
- SSR[TDF] is set during the ninth clock pulse of a target-transmit transfer.
- SSR[RDF] is set during the ninth clock pulse of a target-receive transfer.
- SSR[TAF] is set during the eighth clock pulse of an address byte or a target-receive transfer. In HS mode, this option is disabled.
- Clock stretching can be extended for a number of cycles equal to the value of SCFGR2[CLKHOLD] cycles. This stretching allows additional setup time to sample the SDA pin externally. In HS mode, this option is disabled.

When clock stretching is enabled, clock stretching extends for one peripheral bus clock cycle after SDA updates, unless extended by the SCFGR2[CLKHOLD] configuration.

62.3.2.5 Timing parameters

The LPI2C target can configure the following timing parameters:

- SDA data valid time from SCL negation to SDA update
- SCL hold time when clock stretching is enabled to increase setup time when sampling SDA externally
- SCL glitch filter time
- SDA glitch filter time

These parameters are disabled when SCR[FILTEN] is 0, when SCR[FILTDZ] is 1 in Doze mode, and when LPI2C target detects HS mode. When disabled, the LPI2C target is clocked directly from the I2C bus. In this case, the target may not satisfy all timing requirements of the I2C specification (such as SDA minimum hold time in Standard/Fast mode).

The LPI2C target places the following restrictions on the timing parameters:

- You must configure SCFGR2[FILTSDA] to be greater than or equal to SCFGR2[FILTSCL] (unless compensating for board level skew between SDA and SCL).
- You must configure SCFGR2[DATAVD] to be less than the minimum SCL low period.

62.3.2.6 Error conditions

The LPI2C target can flag the following error conditions:

- SSR[BEF] is set when the LPI2C target is driving SDA but it samples a different value than what is expected.
- SSR[FEF] is set due to a transmit data underrun or a receive data overrun. To eliminate the possibility of underrun and overrun, enable clock stretching. When SCFGR1[RXNACK] is 1, the LPI2C target generates a NACK on a receive data overrun.
- SSR[FEF] is also set due to an address overrun, but only when SCFGR1[RXCFG] is 1. To eliminate the possibility of overrun, enable clock stretching. When SCFGR1[RXNACK] is 1, the LPI2C target generates a NACK on an address overrun regardless of the value of SCFGR1[RXCFG].

The LPI2C target does not implement a timeout due to SCL or SDA being stuck low. If this detection is required, use the LPI2C controller logic so you can reset the LPI2C target when this condition is detected.

62.3.3 Low-power modes

LPI2C remains functional during low-power modes, if MCR[DOZEN] = 0 and LPI2C uses an external or internal clock source that remains enabled. LPI2C can generate an interrupt or DMA request to cause a wake-up from low-power modes.

You can configure LPI2C to be disabled in low-power modes when MCR[DOZEN] = 1. In this case, LPI2C waits for the current transfer to complete any pending operation.

NOTE

See the chip-specific information for low-power modes available on your chip.

62.3.4 Debug mode

Table 1016. Debug mode

Mode	LPI2C operation
Debug	If MCR[DBGEN] = 1, can continue operating in Debug mode.

62.3.5 Peripheral triggers

The connection of the LPI2C peripheral triggers to other peripherals depends upon the specific device being used.

Table 1017. LPI2C triggers

Trigger	Description
Controller output trigger	Generates an output trigger that can be connected to other peripherals on the device. The controller output trigger asserts on either a repeated Start or a Stop condition. The trigger remains asserted for one cycle of the LPI2C functional clock divided by MCFGR1[PRESCALE].
Target output trigger	Generates an output trigger that can be connected to other peripherals on the device. The target output trigger asserts on either a repeated Start or a Stop condition that occurs after a target address match. The target output trigger remains asserted until the next target SCL pin negation.
Input trigger	To control the start of a LPI2C bus transfer, the LPI2C input trigger can be selected instead of the HREQ input. The input trigger is synchronized. To be detected, the input trigger must assert for at least two cycles of the LPI2C functional clock divided by the value of MCFGR1[PRESCALE]. When LPI2C is busy, the HREQ input (and therefore the input trigger) is ignored.

62.3.6 Clocking

Table 1018. LPI2C clocks

Clock	Description
LPI2C functional clock	The LPI2C functional clock is asynchronous to the bus clock. It can remain enabled in low-power modes to support I2C bus transfers by the LPI2C controller. The functional clock is also used by the LPI2C target to support digital filter and data hold time configurations. The LPI2C controller divides the functional clock by a prescaler (MCFGR1[PRESCALE]) and the resulting frequency must be at least eight times faster than the I2C bus bandwidth.
External clock	The LPI2C target logic is clocked directly from the external pins. These pins are SCL and SDA, or SCLS and SDAS if the controller and target are implemented on separate pins). This clocking allows the LPI2C target to remain operational, even when the LPI2C functional clock is disabled. **NOTE** If the LPI2C functional clock is disabled, the LPI2C target digital filter must be disabled. This condition can affect compliance with some timing parameters of the I2C specification, such as data hold time.

Table continues on the next page...

Table 1018. LPI2C clocks (continued)

Clock	Description
Bus clock	The bus clock is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPI2C controller and target registers.

For chip-specific clocking information, see the Clocking chapter.

62.3.7 Reset

Table 1019. LPI2C resets

Reset	Description
Chip reset	The logic and registers for the LPI2C controller and target are reset to their default states after a chip reset.
Software reset	The LPI2C controller implements a software reset field in its control register. MCR[RST] resets all controller logic and registers to their default states, except for Controller Control (MCR) itself. The LPI2C target implements a software reset field in its control register. SCR[RST] resets all target logic and registers to their default states, except for Target Control (SCR) itself.
FIFO reset	The LPI2C controller implements write-only control fields that reset the transmit FIFO (MCR[RTF]) and receive FIFO (MCR[RRF]). After a FIFO is reset, that FIFO is empty. The LPI2C target implements write-only control fields that reset the transmit data register (SCR[RTF]) and receive data register (SCR[RRF]). After a data register is reset, that data register is empty.

62.3.8 Interrupts and DMA requests

Depending on the configuration, interrupts and DMA requests can be combined:

- LPI2C controller and target interrupts
- LPI2C controller and target transmit DMA requests
- LPI2C controller and target receive DMA requests

62.3.8.1 Controller mode

Table 1020 lists the Controller mode sources that can generate LPI2C controller interrupts and LPI2C controller transmit and receive DMA requests.

Table 1020. Controller interrupts and DMA requests

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
Transmit Data Flag (MSR[TDF])	Data can be written to transmit FIFO, as configured by MFCR[TXWATER].	Y	TX	Y
Receive Data Flag (MSR[RDF])	Data can be read from the receive FIFO, as configured by MFCR[RXWATER].	Y	RX	Y

Table continues on the next page...

Table 1020. Controller interrupts and DMA requests (continued)

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
End Packet Flag (MSR[EPF])	Controller has transmitted a repeated Start or Stop condition.	Y	N	Y
Stop Detect Flag (MSR[SDF])	Controller has transmitted a Stop condition (and the LPI2C controller is idle, if MCFGR1[STOPCFG] = 1) .	Y	N	Y
NACK Detect Flag (MSR[NDF])	During an address byte, the controller expects an ACK but detects a NACK. During an address byte, the controller expects a NACK but detects an ACK. During a controller-transmitter data byte, the controller detects a NACK.	Y	N	Y
Arbitration Lost Flag (MSR[ALF])	The controller lost arbitration due to a Start or Stop condition detected at the wrong time, or the controller was transmitting data but received data different from the data that was transmitted.	Y	N	Y
FIFO Error Flag (MSR[FEF])	The controller expects a Start condition in the command FIFO, but the next entry in the command FIFO is not a Start condition.	Y	N	Y
Pin Low Timeout Flag (MSR[PLTF])	Pin low timeout is enabled and SCL (or SDA, if configured) is low for longer than the configured timeout.	Y	N	Y
Data Match Flag (MSR[DMF])	The received data matches the configured data match, but the received data is not discarded due to a command FIFO entry.	Y	N	Y
Start Detect Flag (MSR[STF])	A Start condition is detected on the I2C bus (and the LPI2C controller is idle, if MCFGR1[STARTCFG] = 1).	Y	N	Y
Controller Busy Flag (MSR[MBF])	LPI2C controller is busy transmitting or receiving data.	N	N	N
Bus Busy Flag (MSR[BBF])	LPI2C controller is enabled and activity is detected on the I2C bus, but no Stop condition is detected and no bus idle timeout (if enabled) occurred.	N	N	N

62.3.8.2 Target mode

Table 1021 lists the target mode sources that can generate LPI2C target interrupts and LPI2C target transmit and receive DMA requests.

Table 1021. Target interrupts and DMA requests

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
Transmit Data Flag (SSR[TDF])	Data can be written to Target Transmit Data (STDR).	Y	TX	Y
Receive Data Flag (SSR[RDF])	Data can be read from Target Receive Data (SRDR).	Y	RX	Y
Address Valid Flag (SSR[AVF])	Address can be read from Target Address Status (SASR).	Y	RX	Y
Transmit ACK Flag (SSR[TAF])	ACK or NACK can be written to Target Transmit ACK (STAR).	Y	N	Y
Repeated Start Flag (SSR[RSF])	Target has detected an address match followed by a repeated Start condition.	Y	RX	Y
Stop Detect Flag (SSR[SDF])	Target has detected an address match followed by a Stop condition.	Y	RX	Y
Bit Error Flag (SSR[BEF])	Target was transmitting data, but received data is different from what was transmitted.	Y	N	Y
FIFO Error Flag (SSR[FEF])	This flag is set by: • Transmit data underrun • Receive data overrun • Address status overrun when SCFGR1[RXCFG] = 1 This flag can only be set when clock stretching is disabled.	Y	N	Y
Address Match 0 Flag (SSR[AM0F])	Target detected an address match SAMR[ADDR0].	Y	N	N
Address Match 1 Flag (SSR[AM1F])	Target detected an address match with SAMR[ADDR1] or using an address range.	Y	N	N
General Call Flag (SSR[GCF])	Target detected an address match with the general call address.	Y	N	N
SMBus Alert Response Flag (SSR[SARF])	Target detected an address match with the SMBus alert address.	Y	N	N
Target Busy Flag (SSR[SBF])	LPI2C target is busy receiving an address byte or is transmitting or receiving data.	N	N	N
Bus Busy Flag (SSR[BBF])	LPI2C target is enabled and a Start condition is detected on I2C bus, but no Stop condition detected.	N	N	N

62.3.8.3 End-of-packet DMA transfer

End-of-packet functionality serves serial interfaces where the transfer size may not be known in advance and the data is pushed by an external device. Examples include UART receive, I2C Target mode, and SPI Target mode. End-of-packet processing

is intended to ensure that data does not become stranded in either the receive FIFO or DMA receive buffer. Support for end-of-packet processing must be implemented in both the serial interfaces and DMA controller.

The condition that signals the end of packet differs for each serial interface but is processed by the serial peripheral and DMA in the same way. For example:

- UART end of packet is signaled by an idle line condition.
- I2C end of packet is signaled by Stop or repeated Start condition.
- SPI end of packet is signaled by PCS negation.

When the serial peripheral is configured to signal an end-of-packet condition to the DMA and the serial interface detects an end-of-packet condition, it asserts the DMA request for the receive FIFO regardless of the watermark configuration. For larger watermark configurations, this behavior ensures that the last few words of the transfer are first flushed from the receive FIFO.

The DMA then reads the contents of the receive FIFO, depending on the first word in the FIFO:

- If the receive FIFO is empty, the serial interface signals an end-of-packet condition to the DMA controller.
- If the receive FIFO is not empty, but the first word in the FIFO starts a new packet, data is not pulled from the receive FIFO. Also, the serial interface signals an end-of-packet condition to the DMA controller.
- If the receive FIFO is not empty, and the first word in the FIFO does not start a new packet, the DMA transfers data as normal.

Because the DMA may transfer multiple words per request, the end-of-packet condition persists until the DMA minor loop finishes and no additional data is pulled from the FIFO. The status flag that triggered the end-of-packet condition is cleared when the minor loop completes after the end of packet is signaled to the DMA controller.

When the DMA detects the end-of-packet condition, it writes all received words up to the end of packet into system memory. It also saves the destination address for the word after the last valid data. The DMA then terminates the channel as if the major loop completed, including final offsets and optional interrupts, channel linking, and scatter-gather. The final destination address can be saved to system memory or is available in the destination address register.

Because the DMA terminates the major loop, the receive FIFO is not serviced until the DMA is reconfigured. You can perform this reconfiguration through software or hardware (for example, channel linking or scatter-gather). Minimize this delay to avoid receiver FIFO overrun. Automatic DMA end-of-packet processing is not recommended when there are only a few words transferred between end-of-packet conditions. In this case, the DMA spends more time processing the end of packet than transferring the data. For example, to avoid an excessive number of idle conditions, increase the UART idle line length as needed.

62.4 External signals

Table 1022. External signals

Signal	Name	Two-wire scheme	Four-wire scheme	Direction
SCL	LPI2C clock line	SCL	In Four-Wire mode, this pin is the SCL input pin.	Input or output
SDA	LPI2C data line	SDA	In Four-Wire mode, this pin is the SDA input pin.	Input or output
SCLS	Secondary I2C clock line	Not used	In Four-Wire mode, this pin is the SCLS output pin. If LPI2C controller/target are configured to use separate pins, then this pin is the LPI2C target SCL pin.	Input or output
SDAS	Secondary I2C data line	Not used	In Four-Wire mode, this pin is the SDAS output pin. If LPI2C controller/target are	Input

Table continues on the next page...

Table 1022. External signals (continued)

Signal	Name	Two-wire scheme	Four-wire scheme	Direction
			configured to use separate pins, then this pin is the LPI2C target SDA pin.	or output
HREQ	Host request	When configured for controller (input), if host request is asserted and the I2C bus is idle, it initiates a transfer. When configured for target (output), it asserts while valid data is present in Target Transmit Data (STDR). HREQ is an additional pin separate from the two-wire or four-wire scheme.		Input or output

Figure 465 shows the two-signal connection.

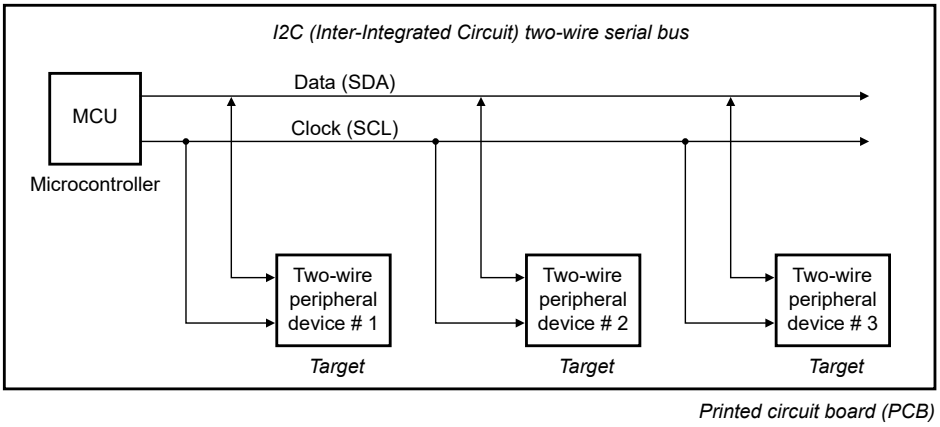


Figure 465. I²C two-wire serial bus

Figure 466 shows a possible four-signal connection.

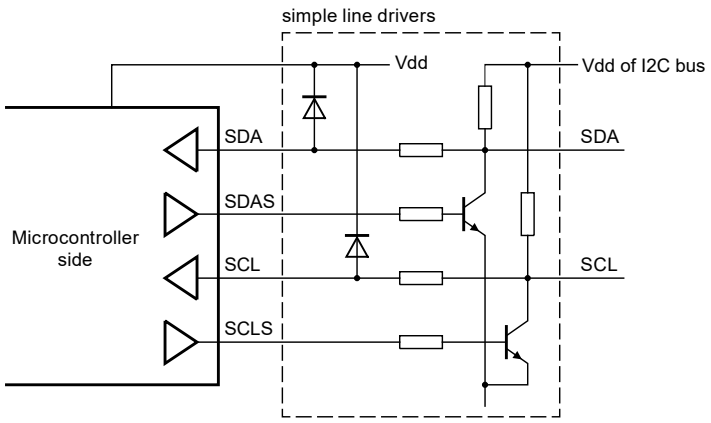


Figure 466. I²C four-wire serial bus

62.5 Initialization

To initialize the LPI2C controller:

1. Configure Controller Configuration 0 (MCFGR0)–Controller Configuration 3 (MCFGR3) as required by the application.

2. Configure Controller Clock Configuration 0 (MCCR0) and Controller Clock Configuration 1 (MCCR1) to satisfy the timing requirements of the I2C mode supported by the application.
3. Enable controller interrupts and DMA requests as required by the application.
4. Enable the LPI2C controller by writing 1 to MCR[MEN].

To initialize the LPI2C target:

1. Configure Target Address Match (SAMR) with the I2C address of the target location on the I2C bus.
2. Configure Target Configuration 0 (SCFGR0) and Target Configuration 1 (SCFGR1) as required by the application.
3. Configure Target Configuration 2 (SCFGR2) to satisfy the timing requirements of the I2C mode supported by the application.
4. Enable target interrupts and DMA requests as required by the application.
5. Enable the LPI2C target by writing 1 to SCR[SEN].

62.6 Application information

Configure the I2C timing parameters to meet the requirements of the I2C specification. This configuration depends on the supported mode and LPI2C functional clock frequency. When switching between two modes using different clock configuration registers (for example, Fast mode and HS mode), MCFGR1[PRESCALE] must remain constant between the modes.

Table 1023. Example timing configurations

I2C mode	Clock frequency	Baud rate	PRESCALE	FILTSCL / FILTSDA	SETHOLD	CLKLO	CLKHI	DATAVD
Standard	8 MHz	100 kbit/s	0h	0h/0h	24h	28h	24h	02h
Standard	48 MHz	100 kbit/s	2h	1h/1h	37h	3Fh	37h	03h
Standard	60 MHz	100 kbit/s	2h	1h/1h	45h	50h	44h	04h
Fast	8 MHz	400 kbit/s	0h	0h/0h	04h	0Bh	05h	02h
Fast+	8 MHz	1 Mbit/s	0h	0h/0h	02h	03h	01h	01h
Fast	48 MHz	400 kbit/s	0h	1h/1h	1Dh	3Eh	35h	0Fh
Fast	48 MHz	400 kbit/s	2h	1h/1h	07h	11h	0Bh	03h
Fast+	48 MHz	1 Mbit/s	2h	1h/1h	03h	06h	04h	04h
HS	48 MHz	3.2 Mbit/s	0h	0h/0h	07h	08h	03h	01h
Fast	60 MHz	400 kbit/s	1h	2h/2h	11h	28h	1Fh	08h
Fast+	60 MHz	1 Mbit/s	1h	2h/2h	07h	0Fh	0Bh	01h
HS	60 MHz	3.33 Mbit/s	1h	0h/0h	04h	04h	02h	01h

62.7 Memory map and registers

62.7.1 LPI2C register descriptions

Writing to a read-only register or reading from a write-only register can cause bus errors. This module does not check whether programmed values in the registers are correct; you must ensure that valid programmed values are written to the registers.

62.7.1.1 LPI2C memory map

LPI2C1 base address: 4434_0000h

LPI2C2 base address: 4435_0000h

LPI2C3 base address: 4253_0000h

LPI2C4 base address: 4254_0000h

LPI2C5 base address: 42D3_0000h

LPI2C6 base address: 42D4_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0103_0003h
4h	Parameter (PARAM)	32	R	0000_0303h
10h	Controller Control (MCR)	32	RW	0000_0000h
14h	Controller Status (MSR)	32	RW	0000_0001h
18h	Controller Interrupt Enable (MIER)	32	RW	0000_0000h
1Ch	Controller DMA Enable (MDER)	32	RW	0000_0000h
20h	Controller Configuration 0 (MCFGR0)	32	RW	0000_0000h
24h	Controller Configuration 1 (MCFGR1)	32	RW	0000_0000h
28h	Controller Configuration 2 (MCFGR2)	32	RW	0000_0000h
2Ch	Controller Configuration 3 (MCFGR3)	32	RW	0000_0000h
40h	Controller Data Match (MDMR)	32	RW	0000_0000h
48h	Controller Clock Configuration 0 (MCCR0)	32	RW	0000_0000h
50h	Controller Clock Configuration 1 (MCCR1)	32	RW	0000_0000h
58h	Controller FIFO Control (MFCR)	32	RW	0000_0000h
5Ch	Controller FIFO Status (MFSR)	32	R	0000_0000h
60h	Controller Transmit Data (MTDR)	32	W	0000_0000h
70h	Controller Receive Data (MRDR)	32	R	0000_4000h
78h	Controller Receive Data Read Only (MRDROR)	32	R	0000_4000h
110h	Target Control (SCR)	32	RW	0000_0000h
114h	Target Status (SSR)	32	RW	0000_0000h
118h	Target Interrupt Enable (SIER)	32	RW	0000_0000h
11Ch	Target DMA Enable (SDER)	32	RW	0000_0000h
120h	Target Configuration 0 (SCFGR0)	32	RW	0000_0000h
124h	Target Configuration 1 (SCFGR1)	32	RW	0000_0000h
128h	Target Configuration 2 (SCFGR2)	32	RW	0000_0000h
140h	Target Address Match (SAMR)	32	RW	0000_0000h
150h	Target Address Status (SASR)	32	R	0000_4000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
154h	Target Transmit ACK (STAR)	32	RW	0000_0000h
160h	Target Transmit Data (STDR)	32	W	0000_0000h
170h	Target Receive Data (SRDR)	32	R	0000_4000h
178h	Target Receive Data Read Only (SRDROR)	32	R	0000_4000h
200h - 3FCh	Controller Transmit Command Burst (MTCBR0 - MTCBR127)	32	W	0000_0000h
400h - 7FCh	Transmit Data Burst (MTDBR0 - MTDBR255)	32	W	0000_0000h

62.7.1.2 Version ID (VERID)

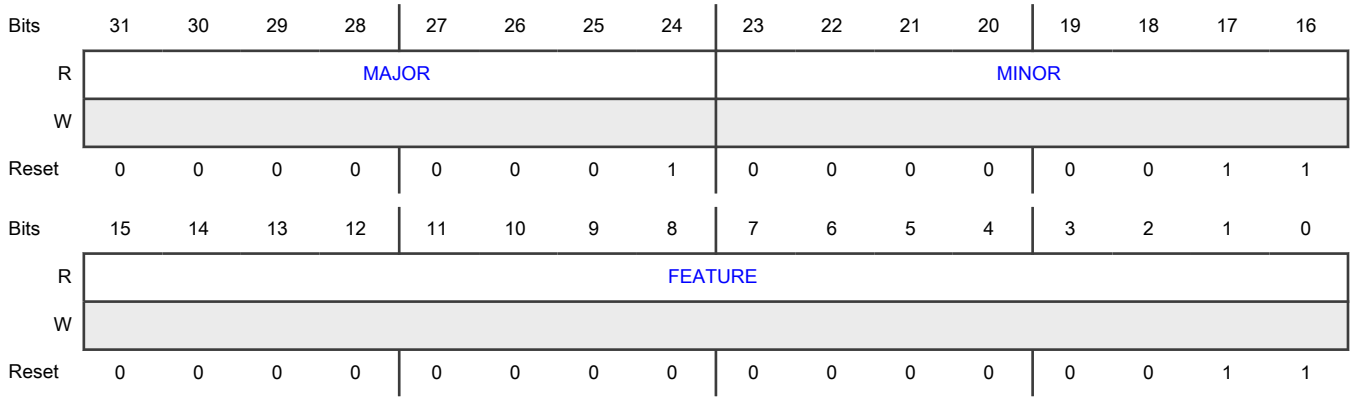
Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module design specification.
23-16	Minor Version Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
MINOR	Returns the minor version number for the module design specification.
15-0 FEATURE	Feature Specification Number Returns the feature set number. 0000_0000_0000_0010b - Controller only, with standard feature set 0000_0000_0000_0011b - Controller and target, with standard feature set

62.7.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains parameter values implemented in the module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				MRXFIFO				0				MTXFIFO			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

Fields

Field	Function
31-16 —	Reserved
15-12 —	Reserved
11-8	Controller Receive FIFO Size

Table continues on the next page...

Table continued from the previous page...

Field	Function
MRXFIFO	Configures the number of words in the controller receive FIFO to $2^{MRXFIFO}$.
7-4 —	Reserved
3-0 MTXFIFO	Controller Transmit FIFO Size Configures the number of words in the controller transmit FIFO to $2^{MTXFIFO}$.

62.7.1.4 Controller Control (MCR)

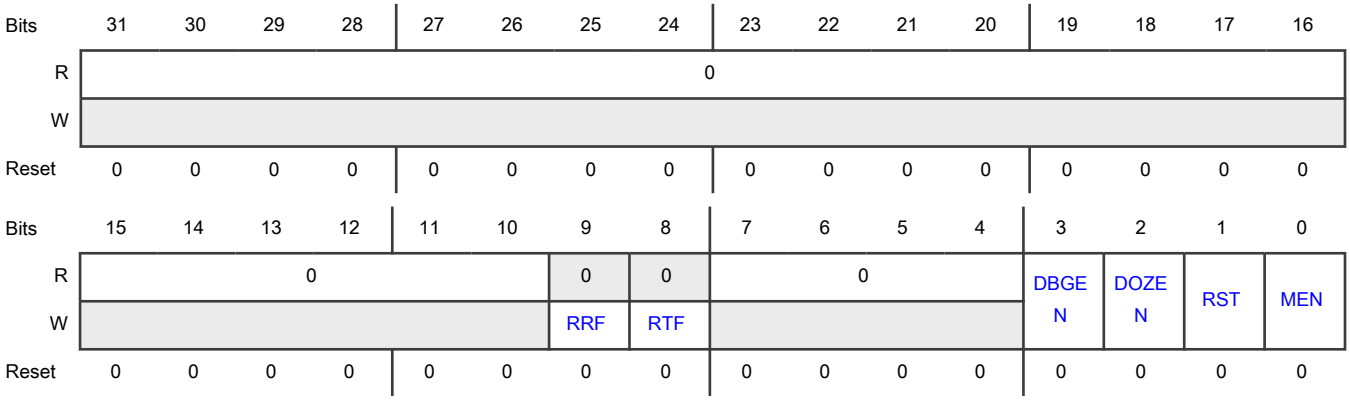
Offset

Register	Offset
MCR	10h

Function

Contains resets, debug enable, and other controller control settings.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Resets the receive FIFO.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Reset receive FIFO
8 RTF	Reset Transmit FIFO Resets the transmit FIFO. 0b - No effect 1b - Reset transmit FIFO
7-4 —	Reserved
3 DBGEN	Debug Enable Enables the controller in Debug mode. 0b - Disable 1b - Enable
2 DOZEN	Doze Mode Enable Enables the controller in Doze mode. 0b - Enable 1b - Disable
1 RST	Software Reset Resets all internal controller logic and registers except Controller Control (MCR). This field remains 1 (enabled) until you write 0 to it. The reset takes effect immediately and remains asserted until negated by software. There is no minimum delay required before clearing the software reset. 0b - No effect 1b - Reset
0 MEN	Controller Enable Enables the controller logic. 0b - Disable 1b - Enable

62.7.1.5 Controller Status (MSR)

Offset

Register	Offset
MSR	14h

Function

Contains status flags for transmit and receive data, for start and stop conditions, and for bus and controller busy or idle status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						BBF	MBF	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STF	DMF	PLTF	FEF	ALF	NDF	SDF	EPF	0						RDF	TDF
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-26 —	Reserved
25 BBF	Bus Busy Flag Specifies whether the I2C bus is busy. 0b - Idle 1b - Busy
24 MBF	Controller Busy Flag Specifies whether the I2C controller is busy. 0b - Idle 1b - Busy
23-16 —	Reserved
15 STF	Start Flag Specifies whether a Start condition is detected on the bus when the bus is idle. When MCFGR1[STARTCFG] is 0, this field becomes 1 only if the LPI2C controller is also idle. When MCFGR1[STARTCFG] is 1, STF becomes 1 for any Start condition when the I2C bus is idle. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Start condition not detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Start condition detected When writing 0b - No effect 1b - Clear the flag
14 DMF	Data Match Flag Indicates whether the received data matches MDMR[MATCH0] or MDMR[MATCH1] (as configured by MCFGR1[MATCFG]). Received data discarded due to MTDR[CMD] does not cause this flag to set. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Matching data not received 1b - Matching data received When writing 0b - No effect 1b - Clear the flag
13 PLTF	Pin Low Timeout Flag Indicates whether pin low timeout has occurred. Sets when the SCL or SDA input is low for more than the number of PINLOW cycles defined by MCFGR3[PINLOW], even when the LPI2C controller is idle. You must resolve the pin low condition via software. PLTF cannot be cleared as long as the pin low timeout continues. Before LPI2C can initiate a Start condition, you must clear this flag. See MCFGR1[TIMECFG] for the SCL and/or SDA timeout settings. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Pin low timeout did not occur 1b - Pin low timeout occurred When writing 0b - No effect 1b - Clear the flag
12 FEF	FIFO Error Flag Detects the LPI2C controller's attempt to send or receive data without first generating a (repeated) Start condition. This error can occur when the transmit FIFO underflows when MCFGR1[AUTOSTOP] = 1. When this flag is set, the LPI2C controller sends a Stop condition (if busy). The controller does not initiate a new Start condition until the flag is cleared.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - No FIFO error 1b - FIFO error When writing 0b - No effect 1b - Clear the flag
11 ALF	Arbitration Lost Flag Indicates whether arbitration is lost. Either of these conditions sets this flag: • The LPI2C controller transmits a logic 1 and detects a logic 0 on the I2C bus. • The LPI2C controller detects a Start or Stop condition when the LPI2C controller is transmitting data. When ALF is set, the LPI2C controller releases the I2C bus (goes idle), and the LPI2C controller does not initiate a new Start condition until the ALF is cleared. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Controller did not lose arbitration 1b - Controller lost arbitration When writing 0b - No effect 1b - Clear the flag
10 NDF	NACK Detect Flag Indicates whether an unexpected NACK has been detected. This flag is set when the LPI2C controller detects a NACK it was not expecting when transmitting an address or data. When set, the controller does not initiate a new Start condition until this flag is cleared. If a NACK is expected for a given address (as configured by the command word), this flag is set if a NACK is not generated. When this flag is set, the LPI2C controller automatically transmits a Stop condition if MCFGR1[AUTOSTOP] = 1, or if the transmit FIFO is not empty. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No unexpected NACK detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Unexpected NACK detected When writing 0b - No effect 1b - Clear the flag
9 SDF	Stop Detect Flag Indicates whether the LPI2C controller has generated a Stop condition. When MCFGR1[STOPCFG] = 0, this flag is set for any Stop condition. When MCFGR1[STOPCFG] = 1, this flag is set only when the LPI2C controller is also idle (transmit FIFO is empty). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No Stop condition generated 1b - Stop condition generated When writing 0b - No effect 1b - Clear the flag
8 EPF	End Packet Flag Indicates whether the LPI2C controller has generated a repeated Start condition or a Stop condition. When the controller first generates a Start condition, this flag is not set. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No Stop or repeated Start generated 1b - Stop or repeated Start generated When writing 0b - No effect 1b - Clear the flag
7-2 —	Reserved
1 RDF	Receive Data Flag Indicates whether the receive data is ready. This flag is set when the number of words in the receive FIFO is greater than MFCR[RXWATER]. 0b - Receive data not ready

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Receive data ready
0 TDF	Transmit Data Flag Indicates whether transmit data is requested. This flag is set when the number of words in the transmit FIFO is equal or less than MFCR[TXWATER]. 0b - Transmit data not requested 1b - Transmit data requested

62.7.1.6 Controller Interrupt Enable (MIER)

Offset

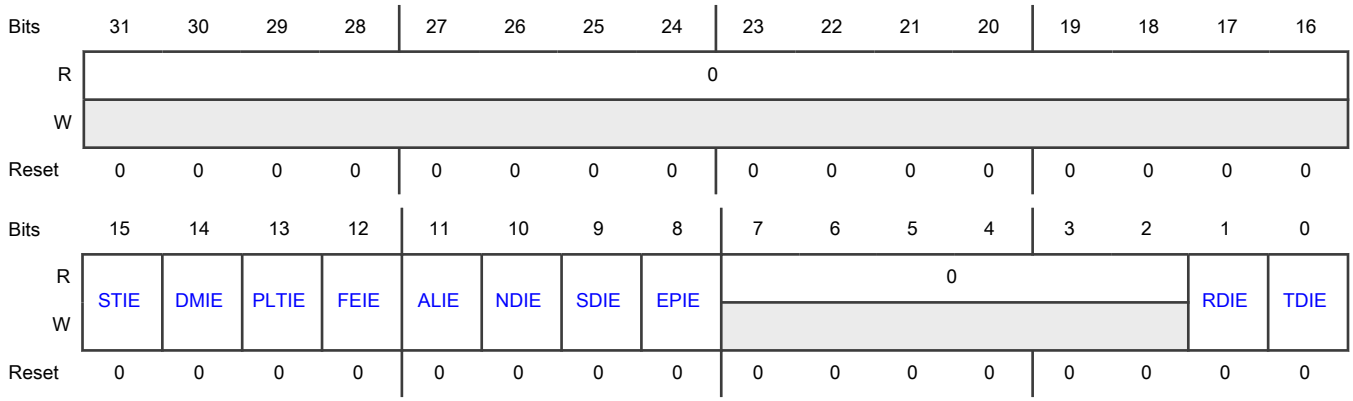
Register	Offset
MIER	18h

Function

Contains enables for:

- Transmit and receive data interrupts
- Start, Stop, and NACK detection interrupts
- DMA interrupts

Diagram



Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 STIE	Start Interrupt Enable Enables interrupt for Start condition. 0b - Disable 1b - Enable
14 DMIE	Data Match Interrupt Enable Enables interrupt for data match. 0b - Disable 1b - Enable
13 PLTIE	Pin Low Timeout Interrupt Enable Enables interrupt for pin-low timeout. 0b - Disable 1b - Enable
12 FEIE	FIFO Error Interrupt Enable Enables interrupt for FIFO error. 0b - Disable 1b - Enable
11 ALIE	Arbitration Lost Interrupt Enable Enables interrupt for arbitration lost. 0b - Disable 1b - Enable
10 NDIE	NACK Detect Interrupt Enable Enables interrupt for NACK detection. 0b - Disable 1b - Enable
9 SDIE	Stop Detect Interrupt Enable Enables interrupt for Stop detection. 0b - Disable 1b - Enable
8 EPIE	End Packet Interrupt Enable Enables interrupt for end packet. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable Enables interrupt for receive data. 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable Enables interrupt for transmit data. 0b - Disable 1b - Enable

62.7.1.7 Controller DMA Enable (MDER)

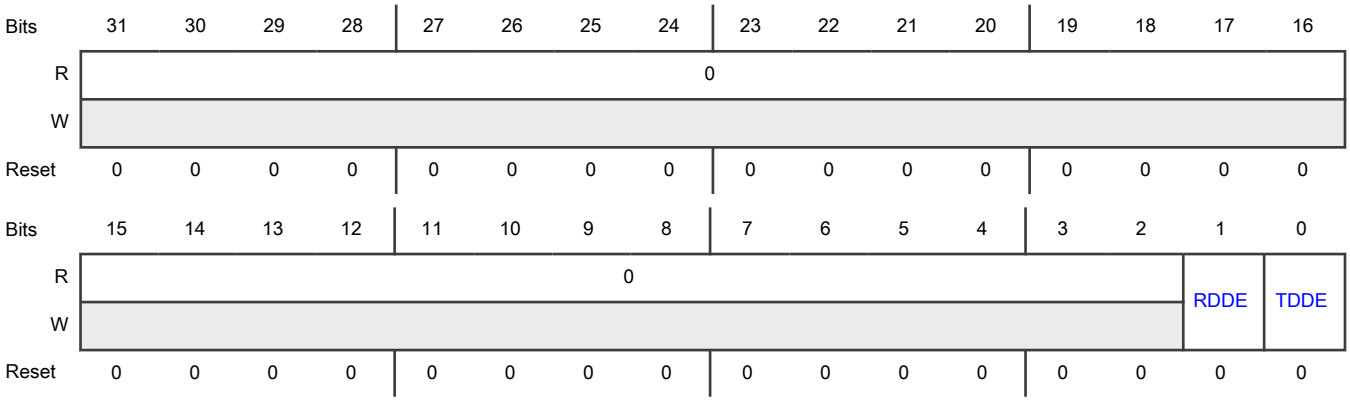
Offset

Register	Offset
MDER	1Ch

Function

Contains DMA transmit, request, and receive enables.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RDDE	Receive Data DMA Enable Enables DMA receive data. 0b - Disable 1b - Enable
0 TDDE	Transmit Data DMA Enable Enables DMA transmit data. 0b - Disable 1b - Enable

62.7.1.8 Controller Configuration 0 (MCFGR0)

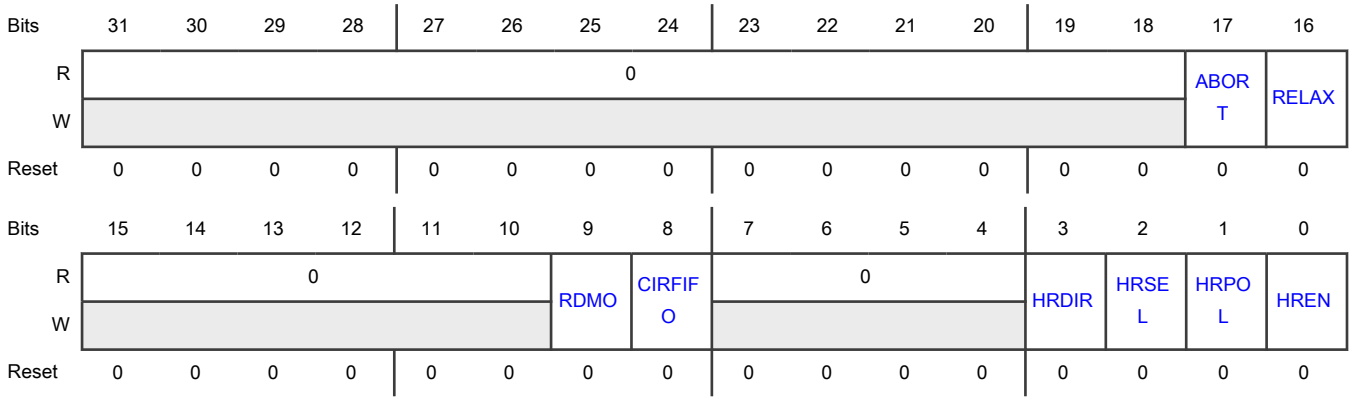
Offset

Register	Offset
MCFGR0	20h

Function

Contains host settings and other receive and transfer settings.

Diagram



Fields

Field	Function
31-18 —	Reserved
17 ABORT	Abort Transfer Determines whether the LPI2C controller completes the existing byte and then sends a Stop condition. A new transfer does not start while this field is 1. 0b - Normal transfer 1b - Abort existing transfer and do not start a new one
16 RELAX	Relaxed Mode Specifies the type of transfer. If this field is 1, the LPI2C controller relaxes several transfer restrictions. Transfers start when the bus is busy and FIFO commands are not checked for errors. You can use this mode to attempt recovery of a stuck I2C target by toggling the SCL pin. 0b - Normal transfer 1b - Relaxed transfer
15-10 —	Reserved
9 RDMO	Receive Data Match Only Determines whether all received data that does not set MSR[DMF] is discarded. After MSR[DMF] is set, the RDMO configuration is ignored. When disabling RDMO, write 0 to this field before writing 0 to MSR[DMF] to ensure that no receive data is lost. 0b - Received data is stored in the receive FIFO 1b - Received data is discarded unless MSR[DMF] is set
8 CIRFIFO	Circular FIFO Enable Enables the transmit FIFO read pointer to be saved to a temporary register. The transmit FIFO empties as normal. After the LPI2C controller is idle and the transmit FIFO is empty, the read pointer value is restored from the temporary register. This setting causes the contents of the transmit FIFO to be cycled through repeatedly. If MCFGR1[AUTOSTOP] is 1, then a Stop condition is sent whenever the transmit FIFO is empty and the read pointer is restored. 0b - Disable 1b - Enable
7-4 —	Reserved
3 HRDIR	Host Request Direction Configures the direction of the HREQ pin. When this field is 1, the LPI2C target drives the HREQ output pin and the pin becomes 1 when there is data in the target transmit data register. For proper operation when this field is 1, write 1 to SCFGR1[TXCFG].

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>LPI2C1</td><td>MCFGR0</td><td>—</td></tr>
<tr><td>LPI2C2</td><td>—</td><td>MCFGR0</td></tr>
<tr><td>LPI2C3</td><td>MCFGR0</td><td>—</td></tr>
<tr><td>LPI2C4</td><td>—</td><td>MCFGR0</td></tr>
<tr><td>LPI2C5</td><td>MCFGR0</td><td>—</td></tr>
<tr><td>LPI2C6</td><td>—</td><td>MCFGR0</td></tr>
</table>
0b - HREQ pin is input (for LPI2C controller)
1b - HREQ pin is output (for LPI2C target)
</td></tr>
<tr><td>2
HRSEL</td><td>
Host Request Select
Selects the source of the host request input. When host request is enabled, this field must not change.
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>LPI2C1</td><td>0b - Host request input is pin HREQ
1b - Host request input is input trigger</td></tr>
<tr><td>LPI2C3</td><td>0b - Host request input is pin HREQ
1b - Host request input is input trigger</td></tr>
<tr><td>LPI2C5</td><td>0b - Host request input is pin HREQ
1b - Host request input is input trigger</td></tr>
<tr><td>LPI2C2</td><td>0b - Reserved
1b - Host request input is input trigger</td></tr>
<tr><td>LPI2C4</td><td>0b - Reserved
1b - Host request input is input trigger</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>LPI2C6</td><td>0b - Reserved 1b - Host request input is input trigger</td></tr></table>
1 HRPOL	Host Request Polarity Configures the polarity of the host request input. When host request is enabled, this field must not change. HRPOL sets the polarity for both the HREQ pin and the input trigger. • When HRPOL=0, the polarity is configured for active low, so host request is asserted if the HREQ pin or input trigger are logic 0. • When HRPOL=1, the polarity is configured for active high, so host request is asserted if the HREQ pin or input trigger are logic 1. 0b - Active low 1b - Active high
0 HREN	Host Request Enable Enables host request. When enabled, the LPI2C controller only initiates a Start condition if the host request input is asserted and the bus is idle. A repeated Start condition is not affected by the host request. 0b - Disable 1b - Enable

62.7.1.9 Controller Configuration 1 (MCFGR1)

Offset

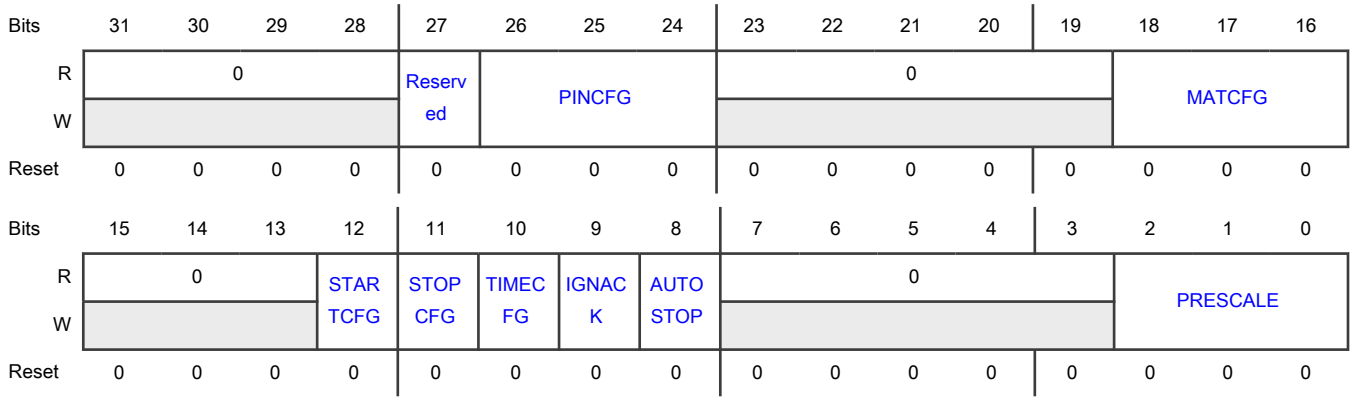
Register	Offset
MCFGR1	24h

Function

Contains controls for pin configuration, clock prescaler, and various other control settings.

Write to this register only when the I2C controller is disabled.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26-24 PINCFG	Pin Configuration Configures the pin mode for LPI2C. 000b - Two-pin open drain mode. SCL/SDA pins: Bidirectional open drain for controller and target. SCLS/SDAS pins: Not used. 001b - Two-pin output only mode (Ultra-Fast mode). SCL/SDA pins: Output-only (Ultra-Fast mode) open drain for controller and target. SCLS/SDAS pins: Not used. 010b - Two-pin push-pull mode. SCL/SDA pins: Bidirectional push-pull for controller and target. SCLS/SDAS pins: Not used. 011b - Four-pin push-pull mode. SCL/SDA pins: Input only for controller and target. SCLS/SDAS pins: Output-only push-pull for controller and target. 100b - Two-pin open-drain mode with separate LPI2C target. SCL/SDA pins: Bidirectional open drain for controller. SCLS/SDAS pins: Bidirectional open drain for target. 101b - Two-pin output only mode (Ultra-Fast mode) with separate LPI2C target. SCL/SDA pins: Output-only (Ultra-Fast mode) open drain for controller. SCLS/SDAS pins: Output-only open drain for target. 110b - Two-pin push-pull mode with separate LPI2C target. SCL/SDA pins: Bidirectional push-pull for controller. SCLS/SDAS pins: Bidirectional push-pull for target. 111b - Four-pin push-pull mode (inverted outputs). SCL/SDA pins: Input only for controller and target. SCLS/SDAS pins: Inverted output-only push-pull for controller and target.
23-19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 MATCFG	Match Configuration Configures the condition that sets MSR[DMF]. See Controller Data Match (MDMR). 000b - Match is disabled 001b - Reserved 010b - Match is enabled: first data word equals MDMR[MATCH0] OR MDMR[MATCH1] 011b - Match is enabled: any data word equals MDMR[MATCH0] OR MDMR[MATCH1] 100b - Match is enabled: (first data word equals MDMR[MATCH0]) AND (second data word equals MDMR[MATCH1) 101b - Match is enabled: (any data word equals MDMR[MATCH0]) AND (next data word equals MDMR[MATCH1) 110b - Match is enabled: (first data word AND MDMR[MATCH1]) equals (MDMR[MATCH0] AND MDMR[MATCH1]) 111b - Match is enabled: (any data word AND MDMR[MATCH1]) equals (MDMR[MATCH0] AND MDMR[MATCH1])
15-13 —	Reserved
12 STARTCFG	Start Configuration Configures the conditions that set MSR[STF] when a Start condition occurs. When this field is 0, MSR[STF] is set on a Start condition when both the I2C bus and LPI2C are idle. In other words, any nonrepeated Start condition is initiated by any controller on the bus except the LPI2C controller. When this field is 1, MSR[STF] is set on a Start condition when the I2C bus is idle. In other words, any nonrepeated Start condition is initiated by any controller on the bus including the LPI2C controller. 0b - Sets when both I2C bus and LPI2C controller are idle 1b - Sets when I2C bus is idle
11 STOPCFG	Stop Configuration Configures the conditions that set MSR[SDF]. When this field is 0, any Stop condition generated by the LPI2C controller sets SDF. When this field is 1, the last Stop condition before the LPI2C controller is idle sets SDF. In this case, the transmit FIFO is empty at the time of the Stop condition. 0b - Any Stop condition 1b - Last Stop condition
10 TIMECFG	Timeout Configuration Configures which signals must be low for longer than the configured timeout to set MSR[PLTF]. When this field is 0, MSR[PLTF] is set when SCL is low for longer than the configured timeout.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - SCL 1b - SCL or SDA
9 IGNACK	Ignore NACK Determines whether the LPI2C controller ignores a received NACK and treats it as an ACK. In this case, MSR[NDF] is never set. This field must be 1 in Ultra-Fast mode. 0b - No effect 1b - Treat a received NACK as an ACK
8 AUTOSTOP	Automatic Stop Generation Determines whether a Stop condition is generated when the LPI2C controller is busy and the transmit FIFO is empty. A Stop condition can also be generated using a transmit FIFO command. When this field is 1, a Stop condition is automatically generated when the transmit FIFO is empty and the LPI2C controller is busy. 0b - No effect 1b - Stop automatically generated
7-3 —	Reserved
2-0 PRESCALE	Prescaler Configures the clock prescaler used for all LPI2C controller logic except the digital glitch filters. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

62.7.1.10 Controller Configuration 2 (MCFGR2)

Offset

Register	Offset
MCFGR2	28h

Function

Contains the configuration for the bus idle timeout and glitch filters for SDA and SCL.

Write to this register only when the I2C controller is disabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				FILTSDA				0				FILTSCL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				BUSIDLE											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C controller digital glitch filters for the SDA input. The latency through the glitch filter is equal to the number of cycles defined by this field. The value of this field must be less than the minimum SCL low or high period. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. Writing 0 to this field disables the glitch filter. MCFGR1[PRESCALE] does not affect the glitch filter cycle count. It is automatically bypassed in HS mode.
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C controller digital glitch filters for SCL input. The latency through the glitch filter is equal to the number of cycles defined by this field. The value of this field must be less than the minimum SCL low or high period. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. These cycles are based on the functional clock. Writing 0 to this field disables the glitch filter. MCFGR1[PRESCALE] does not affect the glitch filter cycle count. It is automatically bypassed in HS mode.
15-12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-0 BUSIDLE	Bus Idle Timeout Configures the bus idle timeout period, in clock cycles. If both SCL and SDA are high for longer than the number of cycles defined by this field, the I2C bus is assumed to be idle and the controller can generate a Start condition. Writing 0 to this field disables the bus idle timeout.

62.7.1.11 Controller Configuration 3 (MCFGR3)

Offset

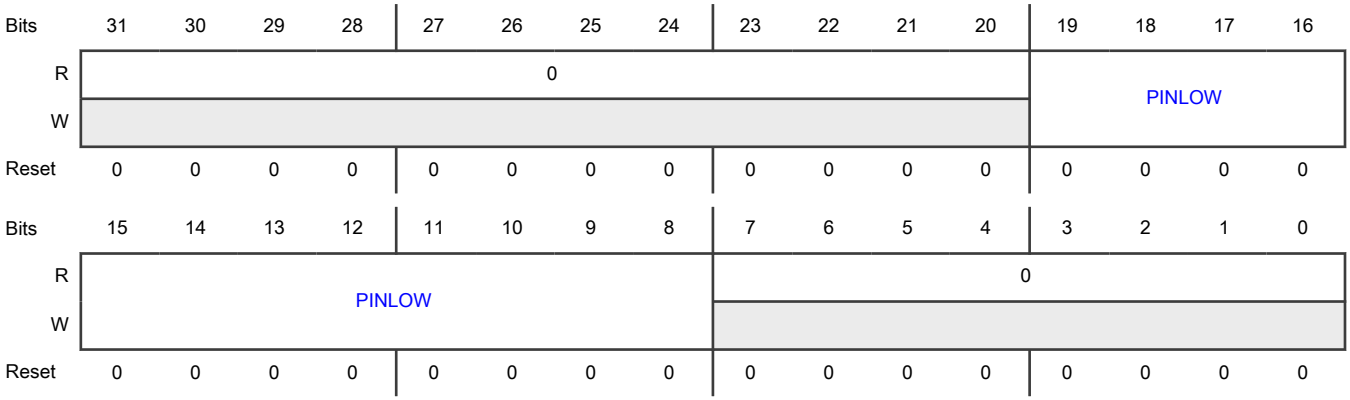
Register	Offset
MCFGR3	2Ch

Function

Configures the threshold value for the pin low timeout flag.

Write to this register only when the I2C controller is disabled.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-8 PINLOW	Pin Low Timeout Configures the threshold value, in clock cycles, that sets MSR[PLTF].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If SCL or SDA (selected by MCFGR1[TIMECFG]) is low for longer than (PINLOW × 256) cycles, MSR[PLTF] is set. When this field is 0, the pin low timeout feature is disabled.
7-0 —	Reserved

62.7.1.12 Controller Data Match (MDMR)

Offset

Register	Offset
MDMR	40h

Function

Contains data match values.

Write to this register only when the I2C controller is disabled or idle.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								MATCH1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								MATCH0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 MATCH1	Match 1 Value Specifies match 1 value that is compared to the received data when receive data match is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 —	Reserved
7-0 MATCH0	Match 0 Value Specifies match 0 value that is compared to the received data when receive data match is enabled.

62.7.1.13 Controller Clock Configuration 0 (MCCR0)

Offset

Register	Offset
MCCR0	48h

Function

Configures various clock controls.

You cannot make changes to this register when the I2C controller is enabled and is used for standard, fast, fast-mode plus, and ultra-fast transfers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DATAVD						0		SETHOLD					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		CLKHI						0		CLKLO					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 DATAVD	Data Valid Delay

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Specifies the minimum number of cycles (minus one) used as the data hold time for SDA. This value must be less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Specifies the minimum number of cycles (minus one) used by the controller for these conditions: • Hold time for a Start • Setup and hold time for a repeated Start • Setup time for a Stop The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock high. The SCL high time is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
7-6 —	Reserved
5-0 CLKLO	Clock Low Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock low. This value is also used for the minimum bus free time between a Stop and a Start condition. This period is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.

62.7.1.14 Controller Clock Configuration 1 (MCCR1)

Offset

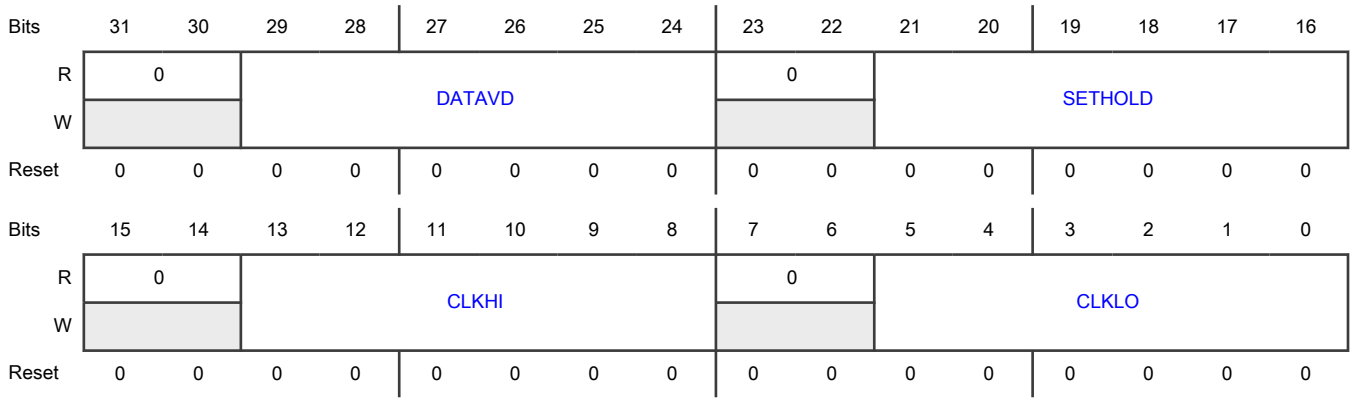
Register	Offset
MCCR1	50h

Function

Configures various clock controls.

You cannot makes changes to this register when the I2C controller is enabled and is used for HS mode transfers. The separate clock configuration for HS mode allows arbitration to take place in Fast mode (with timing configured by Controller Clock Configuration 0 (MCCR0)), before switching to HS mode (with timing configured by MCCR1).

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 DATAVD	Data Valid Delay Specifies the minimum number of cycles (minus one) used as the data hold time for SDA. This value must be less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Specifies the minimum number of cycles (minus one) used by the controller for these conditions: • Hold time for a Start condition • Setup and hold time for a repeated Start condition • Setup time for a Stop condition The setup time is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock high. The SCL high time is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring

Table continues on the next page...

Table continued from the previous page...

Field	Function
	any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
7-6 —	Reserved
5-0 CLKLO	Clock Low Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock low. This value is also used for the minimum bus free time between a Stop and a Start condition. This period is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.

62.7.1.15 Controller FIFO Control (MFCR)

Offset

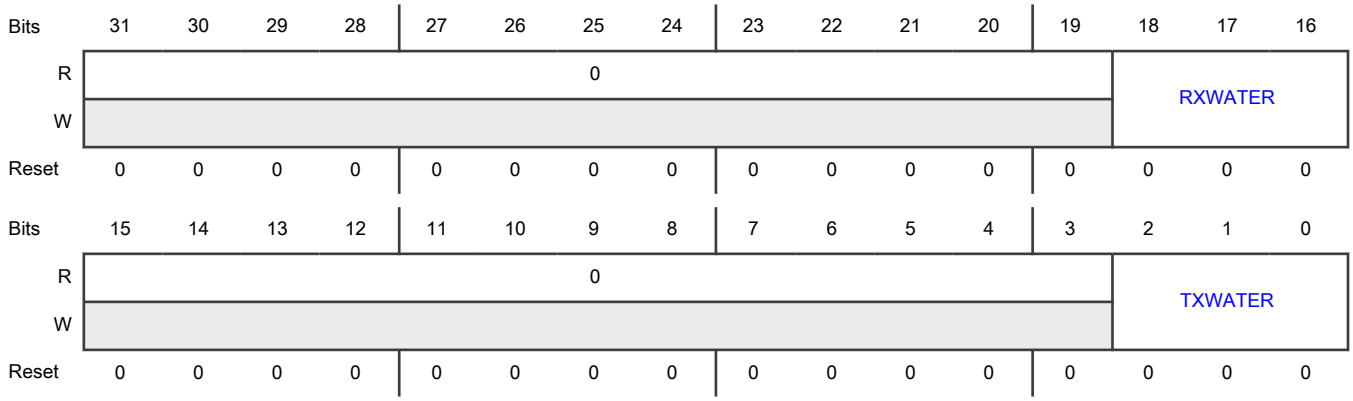
Register	Offset
MFCR	58h

Function

Controls the receive and transmit FIFO watermark values.

This register is used only in Stop mode, when this register is static (not changing).

Diagram



Fields

Field	Function
31-19	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
18-16 RXWATER	Receive FIFO Watermark Determines the watermark for setting SSR[RDF]. That flag is set when the number of words in the receive FIFO is greater than the value of this field. Writing a value equal to or greater than the FIFO size truncates the value.
15-3 —	Reserved
2-0 TXWATER	Transmit FIFO Watermark Determines the watermark for setting SSR[TDF]. That flag is set when the number of words in the transmit FIFO is equal or less than the value of this field. Writing a value equal to or greater than the FIFO size truncates the value.

62.7.1.16 Controller FIFO Status (MFSR)

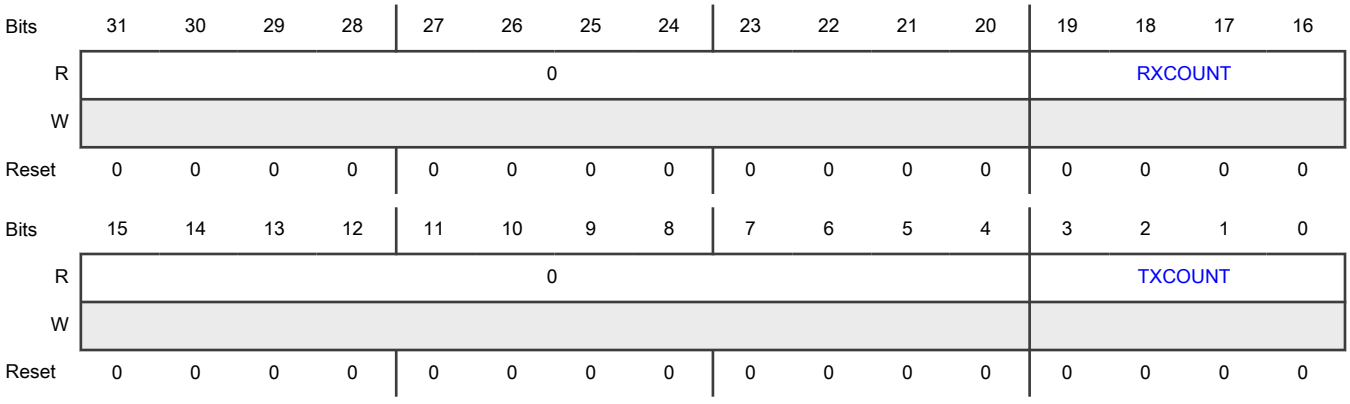
Offset

Register	Offset
MFSR	5Ch

Function

Specifies the number of words in the transmit and receive FIFOs.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 RXCOUNT	Receive FIFO Count Specifies the number of words in the receive FIFO.
15-4 —	Reserved
3-0 TXCOUNT	Transmit FIFO Count Specifies the number of words in the transmit FIFO.

62.7.1.17 Controller Transmit Data (MTDR)

Offset

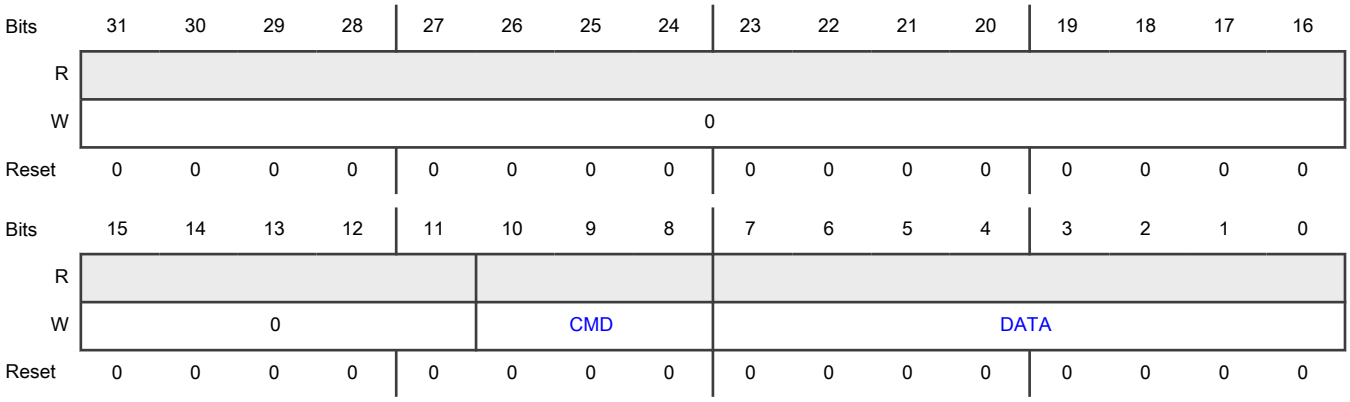
Register	Offset
MTDR	60h

Function

Configures transmit data:

- An 8-bit write to MTDR[CMD] is ignored and does not increment the FIFO write pointer.
- An 8-bit write to MTDR[DATA] zero-extends the value of MTDR[CMD] and increments the FIFO write pointer.
- A 16-bit or 32-bit write operation writes to both MTDR[CMD] and MTDR[DATA] and increments the FIFO write pointer.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 CMD	Command Data Selects command transmitted by controller. 000b - Transmit the value in DATA[7:0] 001b - Receive (DATA[7:0] + 1) bytes. DATA[7:0] is used as a byte counter. Receive that many bytes and check each for a data match (if configured) before storing the received data in the receive FIFO. 010b - Generate Stop condition on I2C bus 011b - Receive and discard (DATA[7:0] + 1) bytes. DATA[7:0] is used as a byte counter. Receive that many bytes but do not check for a data match or store those bytes in the receive FIFO. 100b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] 101b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] (this transfer expects a NACK to be returned) 110b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] using HS mode 111b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] using HS mode (this transfer expects a NACK to be returned)
7-0 DATA	Transmit Data Contains data used by the commands listed in MTDR[CMD]. Performing an 8-bit write to this field zero-extends the value of MTDR[CMD].

62.7.1.18 Controller Receive Data (MRDR)

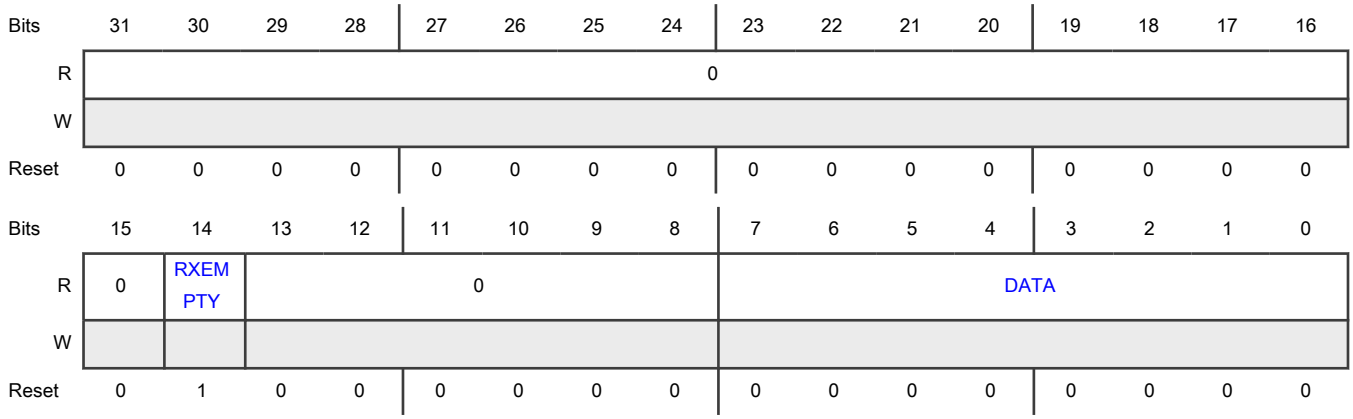
Offset

Register	Offset
MRDR	70h

Function

Contains the status of the receive FIFO and the data received by the I2C controller that has not been discarded.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 RXEMPTY	Receive Empty Indicates whether the controller receive data FIFO is empty. 0b - Not empty 1b - Empty
13-8 —	Reserved
7-0 DATA	Receive Data Contains data received by the I2C controller that has not been discarded. Received data can be discarded due to the command in MTDR[CMD], or the controller can be configured to discard nonmatching data.

62.7.1.19 Controller Receive Data Read Only (MRDROR)

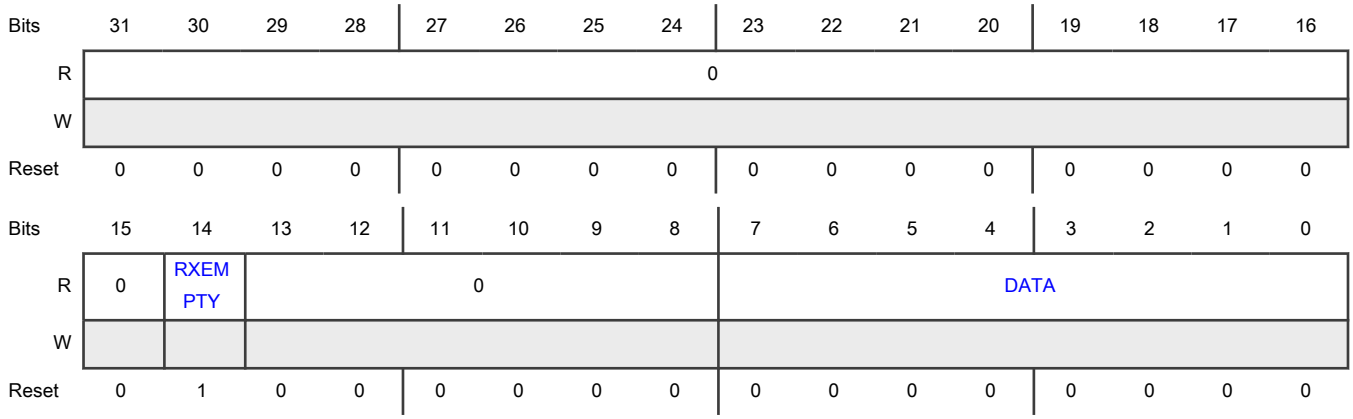
Offset

Register	Offset
MRDROR	78h

Function

Contains the status of the receive FIFO and returns the data received by the I2C controller, but does not pull the data from the FIFO.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 RXEMPTY	RX Empty Indicates whether the receive data FIFO is empty. 0b - Not empty 1b - Empty
13-8 —	Reserved
7-0 DATA	Receive Data

62.7.1.20 Target Control (SCR)

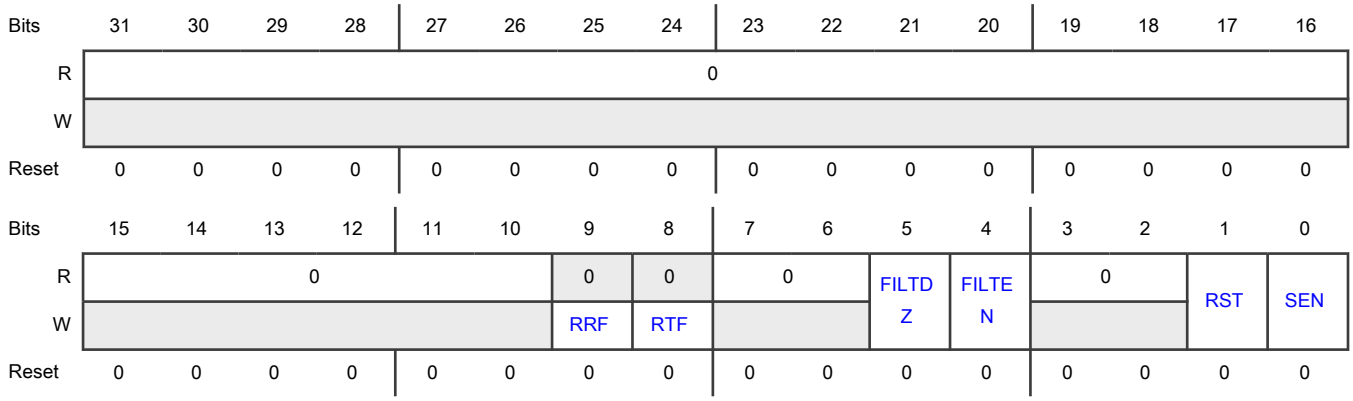
Offset

Register	Offset
SCR	110h

Function

Contains resets and other target control settings.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Empties the receive FIFO in Target Receive Data (SRDR). 0b - No effect 1b - SRDR is now empty
8 RTF	Reset Transmit FIFO Empties the transmit FIFO in Target Transmit Data (STDR). 0b - No effect 1b - STDR is now empty
7-6 —	Reserved
5 FILTDZ	Filter Doze Enable Enables filter in Doze mode. Update this field only when the I2C target is disabled. 0b - Enable 1b - Disable
4 FILTEN	Filter Enable Enables digital filter and output delay counter for target mode. Update this field only when the I2C target is disabled. 0b - Disable 1b - Enable
3-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 RST	Software Reset Resets target mode logic. The reset takes effect immediately. The value of this field remains 1 until you write 0 to it. There is no minimum delay required before clearing the software reset. 0b - Not reset 1b - Reset
0 SEN	Target Enable Enables I2C Target mode. 0b - Disable 1b - Enable

62.7.1.21 Target Status (SSR)

Offset

Register	Offset
SSR	114h

Function

Contains status flags for transmit and receive data, for error conditions, and for bus and target busy or idle status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						BBF	SBF	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SARF	GCF	AM1F	AM0F	FEF	BEF	SDF	RSF	0				TAF	AVF	RDF	TDF
W					W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25 BBF	Bus Busy Flag Indicates whether an I2C bus is idle or busy. 0b - Idle 1b - Busy
24 SBF	Target Busy Flag Indicates whether an I2C target is idle or busy. 0b - Idle 1b - Busy
23-16 —	Reserved
15 SARF	SMBus Alert Response Flag Indicates whether an SMBus alert response has been detected. You can clear this flag by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - Disabled or not detected 1b - Enabled and detected
14 GCF	General Call Flag Indicates whether a target has detected the general call address. You can clear this flag by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - General call address disabled or not detected 1b - General call address detected
13 AM1F	Address Match 1 Flag Indicates whether the received address matches the value in ADDR1, or it falls within the ADDR0 to ADDR1 range as configured by SCFGR1[ADDRCFG]. This flag is cleared by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - Matching address not received 1b - Matching address received
12	Address Match 0 Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
AM0F	Indicates whether the received address matches the ADDR0 field, as configured by SCFGR1[ADDRCFG]. This flag is cleared by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - ADDR0 matching address not received 1b - ADDR0 matching address received
11 FEF	FIFO Error Flag Indicates whether there is a FIFO error. This flag can only be set when clock stretching is disabled. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No FIFO error 1b - FIFO error When writing 0b - No effect 1b - Clear the flag
10 BEF	Bit Error Flag Indicates whether the LPI2C target has transmitted a logic 1 and detects a logic 0 on the I2C bus. The target ignores the rest of the transfer until the next (repeated) Start condition. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No bit error occurred 1b - Bit error occurred When writing 0b - No effect 1b - Clear the flag
9 SDF	Stop Detect Flag Indicates whether the LPI2C target detects a Stop condition, and if the LPI2C target matched the last address byte. When SCFGR1[SDCFG] = 1, this flag is set on any Stop condition. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No Stop detected 1b - Stop detected When writing 0b - No effect 1b - Clear the flag
8 RSF	Repeated Start Flag Indicates whether the LPI2C target detects a repeated Start condition and if the LPI2C target matched the last address byte. When SCFGR1[RSCFG] = 1, this flag is set on any repeated Start condition. This flag is not set when the target first detects a Start condition. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No repeated Start detected 1b - Repeated Start detected When writing 0b - No effect 1b - Clear the flag
7-4 —	Reserved
3 TAF	Transmit ACK Flag Indicates whether a transmit ACK or NACK is required. You can clear this flag by writing to Target Transmit ACK (STAR). 0b - Not required 1b - Required
2 AVF	Address Valid Flag Indicates whether the contents of Target Address Status (SASR) are valid. You can clear this flag by reading SASR. When SCFGR1[RXCFG] = 1, this flag is also cleared by reading Target Receive Data (SRDR). 0b - Not valid 1b - Valid
1 RDF	Receive Data Flag Indicates whether receive data is ready. You can clear this flag by reading Target Receive Data (SRDR). When SCFGR1[RXCFG] = 1, this flag is not cleared when reading Target Receive Data (SRDR) if SSR[AVF] = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not ready 1b - Ready
0 TDF	Transmit Data Flag Indicates whether transmit data has been requested. This flag is cleared by writing to Target Transmit Data (STDR). When SCFGR1[TXCFG] = 0, if a NACK, repeated Start, or Stop condition is detected, this flag is also cleared. 0b - Transmit data not requested 1b - Transmit data is requested

62.7.1.22 Target Interrupt Enable (SIER)

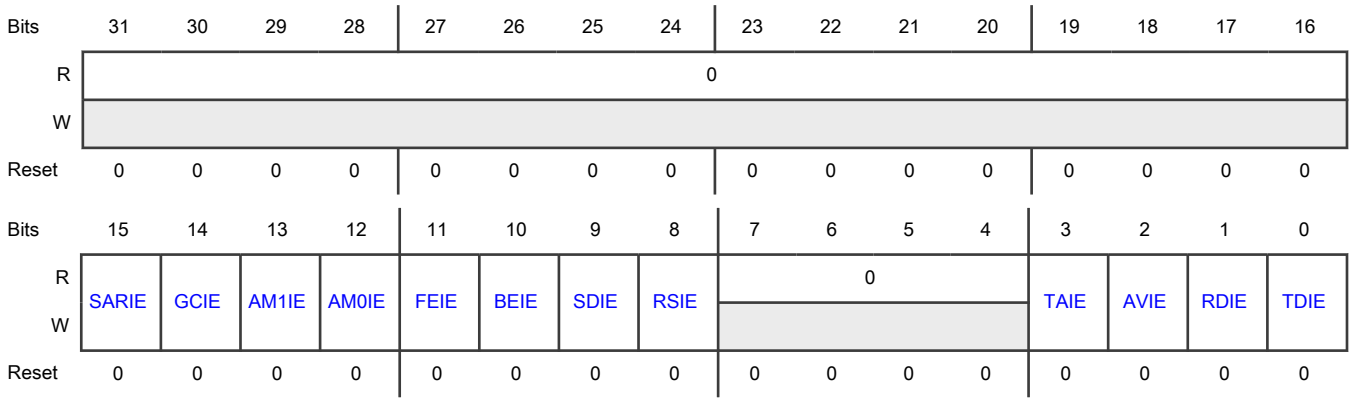
Offset

Register	Offset
SIER	118h

Function

Contains transmit and receive data interrupt enables, start and stop detect interrupt enables, and other target interrupt enables.

Diagram



Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 SARIE	SMBus Alert Response Interrupt Enable Enables interrupt for SMBus alert response. 0b - Disable 1b - Enable
14 GCIE	General Call Interrupt Enable Enables interrupt for general call. 0b - Disabled 1b - Enabled
13 AM1IE	Address Match 1 Interrupt Enable Enables interrupt for address match 1. 0b - Disable 1b - Enable
12 AM0IE	Address Match 0 Interrupt Enable Enables interrupt for address match 0. 0b - Disable 1b - Enable
11 FEIE	FIFO Error Interrupt Enable Enables interrupt for FIFO error. 0b - Disable 1b - Enable
10 BEIE	Bit Error Interrupt Enable Enables interrupt for bit error. 0b - Disable 1b - Enable
9 SDIE	Stop Detect Interrupt Enable Enables interrupt for Stop detection. 0b - Disable 1b - Enable
8 RSIE	Repeated Start Interrupt Enable Enables interrupt for repeated start. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
7-4 —	Reserved
3 TAIE	Transmit ACK Interrupt Enable Enables interrupt for transmit ACK. 0b - Disable 1b - Enable
2 AVIE	Address Valid Interrupt Enable Enables interrupt for valid address. 0b - Disable 1b - Enable
1 RDIE	Receive Data Interrupt Enable Enables interrupt for receive data. 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable Enables interrupt for transmit data. 0b - Disable 1b - Enable

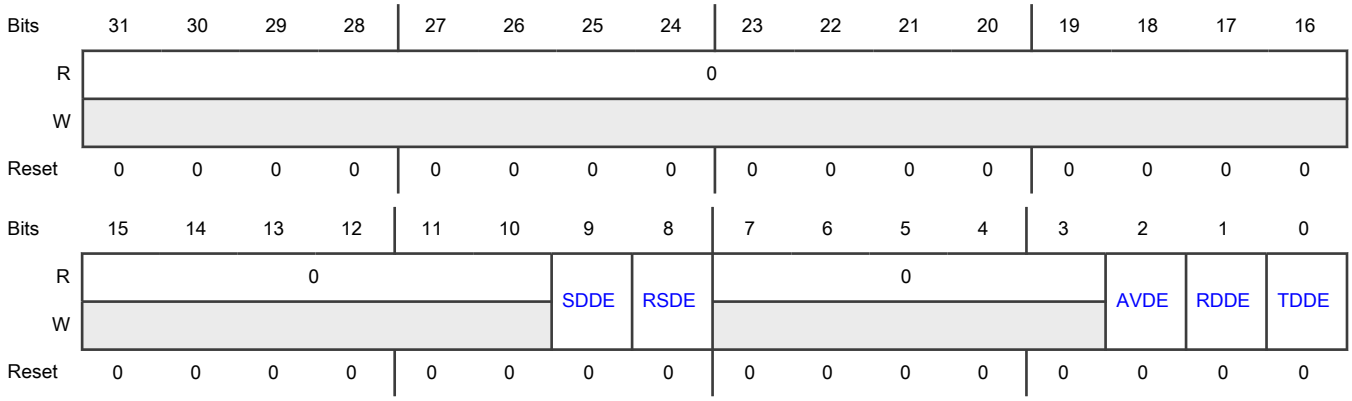
62.7.1.23 Target DMA Enable (SDER)

Offset

Register	Offset
SDER	11Ch

Function
Contains the transmit, request, and receive enables for DMA.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 SDDE	Stop Detect DMA Enable Enables DMA end-of-packet processing on stop detection. Reading Target Receive Data (SRDR) when it is empty: • Generates a DMA end-of-packet response. • Returns 0000_40FFh. • Writes 0 to MSR[SDF]. 0b - Disable 1b - Enable
8 RSDE	Repeated Start DMA Enable Enables DMA end-of-packet processing on repeated start. Reading Target Receive Data (SRDR) when it is empty: • Generates a DMA end-of-packet response. • Returns 0000_40FFh. • Writes 0 to MSR[RDF]. 0b - Disable 1b - Enable
7-3 —	Reserved
2 AVDE	Address Valid DMA Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables address valid DMA request. The address valid DMA request is shared with the receive data DMA request. If both are enabled, write 1 to SCFGR1[RXCFG] to allow the DMA to read the address from Target Receive Data (SRDR). 0b - Disable 1b - Enable
1 RDDE	Receive Data DMA Enable Enables receive data for DMA. 0b - Disable DMA request 1b - Enable DMA request
0 TDDE	Transmit Data DMA Enable Enables transmit data for DMA. 0b - Disable 1b - Enable

62.7.1.24 Target Configuration 0 (SCFGR0)

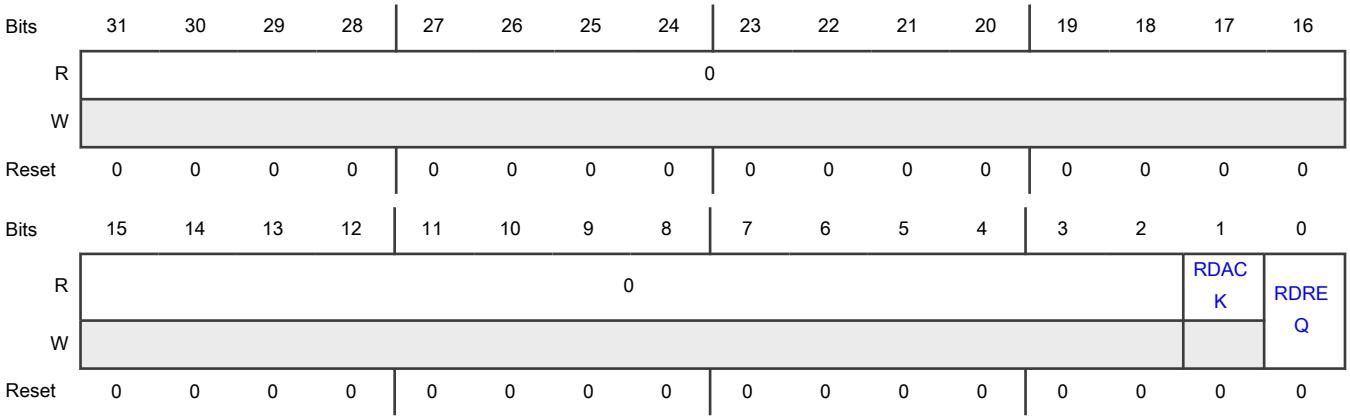
Offset

Register	Offset
SCFGR0	120h

Function

Configures the read request feature.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RDACK	Read Acknowledge Flag Indicates whether a Start then Stop sequence with one SCL pulse between them acknowledged the read request while SCFGR0[RDREQ] = 1. You can clear this flag by writing 0 to SCFGR0[RDREQ]. 0b - Read Request not acknowledged 1b - Read Request acknowledged
0 RDREQ	Read Request Enables read request. When the I2C bus is idle, writing 1 to this field causes the LPI2C target to drive SDA low, triggering a Start condition. The LPI2C target releases SDA on the next falling edge of SCL or when you write 0 to this field. To initiate a second read request (for example, following I2C bus activity that did not acknowledge the request), write 0 then 1 to this field. When the I2C bus is busy, writing to this register always writes 0 to this field. 0b - Disable 1b - Enable

62.7.1.25 Target Configuration 1 (SCFGR1)

Offset

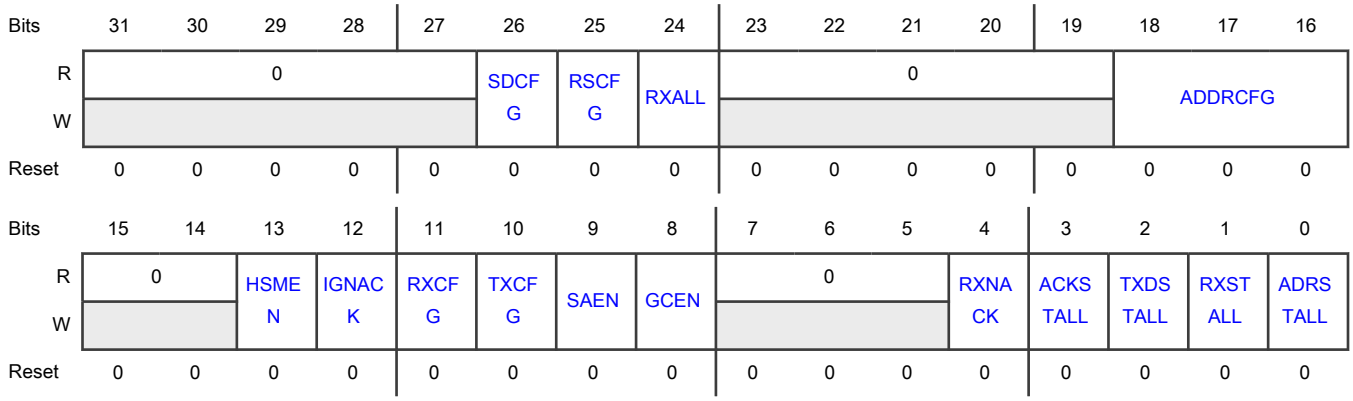
Register	Offset
SCFGR1	124h

Function

Configures various aspects of the target.

Write to this register only when the I2C target is disabled.

Diagram



Fields

Field	Function
31-27 —	Reserved
26 SDCFG	Stop Detect Configuration Configures the conditions that set MSR[SDF]. 0b - Any Stop condition following an address match 1b - Any Stop condition
25 RSCFG	Repeated Start Configuration Configures the conditions that set MSR[STF]. 0b - Any repeated Start condition following an address match 1b - Any repeated Start condition
24 RXALL	Receive All Enables receive-all functionality. When enabled, the LPI2C target stores all addresses on the I2C bus in Target Address Status (SASR) and all data on the I2C bus in Target Receive Data (SRDR). However, the LPI2C target does not drive SDA (for ACK or target-transmit transfer) unless there is an address match. The LPI2C target can drive SCL if clock stretching is enabled. When this field is 1, the LPI2C target only supports 7-bit addressing modes (you must configure SCFGR1[ADDRCFG] for 7-bit address match). Software can support 10-bit addresses, however. The first byte of a 10-bit address is saved to Target Address Match (SAMR). The second byte is saved as the first data byte in Target Receive Data (SRDR). Use this field to aid debugging of the I2C bus by saving all data on the bus without altering the state of the bus. When this field is 1, it is recommended to also write 1 to SCFGR1[RSCFG] and SCFGR1[SDCFG] so you can track the full state of the I2C bus. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 ADDRCFG	Address Configuration Configures the condition that causes an address to match. 000b - Address match 0 (7-bit) 001b - Address match 0 (10-bit) 010b - Address match 0 (7-bit) or address match 1 (7-bit) 011b - Address match 0 (10-bit) or address match 1 (10-bit) 100b - Address match 0 (7-bit) or address match 1 (10-bit) 101b - Address match 0 (10-bit) or address match 1 (7-bit) 110b - From address match 0 (7-bit) to address match 1 (7-bit) 111b - From address match 0 (10-bit) to address match 1 (10-bit)
15-14 —	Reserved
13 HSMEN	HS Mode Enable Enables detection of the HS mode controller code of target address 0000_1XX, but does not cause an address match on this code. When this field is 1 and any HS mode controller code is detected, SCR[FILTEN] and SCFGR1[ACKSTALL] are ignored until the next Stop condition is detected. 0b - Disable 1b - Enable
12 IGNACK	Ignore NACK Determines whether the target ends transfer when a NACK condition is detected. When this field is 1, the LPI2C target continues transfers after a NACK is detected. This field is required to be 1 in Ultra-Fast mode. 0b - End transfer on NACK 1b - Do not end transfer on NACK
11 RXCFG	Receive Data Configuration Configures which data is returned and which flags are cleared when reading Target Receive Data (SRDR). When this field is 0, reading SRDR returns received data and clears MSR[RDF]. When this field is 1, reading SRDR: • Returns the value of Target Address Status (SASR) and clears SSR[AVF] when SSR[AVF] is set. • Returns received data and clears MSR[RDF] when SSR[AVF] is not set. 0b - Return received data, clear MSR[RDF]

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Return SASR and clear SSR[AVF] when SSR[AVF] is set, return received data and clear MSR[RDF] when SSR[AFV] is not set
10 TXCFG	Transmit Flag Configuration Determines which conditions set MSR[TDF]. This field always becomes 1 before a NACK is detected at the end of a target-transmit transfer. This change can cause an extra word to be written to the transmit data FIFO. When this field is 0, Target Transmit Data (STDR) is automatically emptied when a target-transmit transfer is detected. MSR[TDF] is set when a target-transmit transfer is detected, and MSR[TDF] is cleared at the end of the target-transmit transfer. When this field is 1, MSR[TDF] is set when STDR is empty, and MSR[TDF] is cleared when STDR is full. This setting allows STDR to be filled before a target-transmit transfer is detected. However, it can cause STDR to be written before a NACK is detected on the last byte of a target-transmit transfer. 0b - MSR[TDF] is set only during a target-transmit transfer when STDR is empty 1b - MSR[TDF] is set whenever STDR is empty
9 SAEN	SMBus Alert Enable Enables a match on an SMBus alert. 0b - Disable 1b - Enable
8 GCEN	General Call Enable Enables a general call address. 0b - Disable 1b - Enable
7-5 —	Reserved
4 RXNACK	Receive NACK Determines whether to override the setting of STAR[TXNACK] when the LPI2C receives a matching address during an overrun. When this field is 1, the LPI2C target responds with a NACK under the following conditions: • SSR[AVF] would be set due to matching an address, but that flag is already 1 (address overrun). • SSR[RDF] would be set due to receiving data, but that flag is already 1 (receive data overrun). 0b - ACK or NACK always determined by STAR[TXNACK] 1b - NACK always generated on address overrun or receive data overrun, otherwise ACK or NACK is determined by STAR[TXNACK]
3	ACK SCL Stall

Table continues on the next page...

Table continued from the previous page...

Field	Function
ACKSTALL	Enables SCL clock stretching during target-transmit address bytes and target-receiver address and data bytes, so you can write to Target Transmit ACK (STAR) before the ACK or NACK is transmitted. Clock stretching occurs when transmitting the ninth bit, and is therefore not compatible with HS mode. If this field is 1: • You do not need to write 1 to SCFGR1[RXSTALL] or SCFGR1[ADRSTALL]. • When there is an address match on the first byte of a 10-bit address, SSR[AVF] is set, allowing you to read the received address before writing to Target Transmit ACK (STAR). 0b - Disable 1b - Enable
2 TXDSTALL	Transmit Data SCL Stall Enables SCL clock stretching when SSR[TDF] = 1 during a target-transmit transfer. Clock stretching occurs following the ninth bit, and is therefore compatible with HS mode. 0b - Disable 1b - Enable
1 RXSTALL	RX SCL Stall Enables SCL clock stretching when SSR[RDF] = 1 during a target-receive transfer. Clock stretching occurs following the ninth bit, and is therefore compatible with HS mode. 0b - Disable 1b - Enable
0 ADRSTALL	Address SCL Stall Enables SCL clock stretching when SSR[AVF] = 1. Clock stretching only occurs following the ninth bit, and is therefore compatible with HS mode. 0b - Disable 1b - Enable

62.7.1.26 Target Configuration 2 (SCFGR2)

Offset

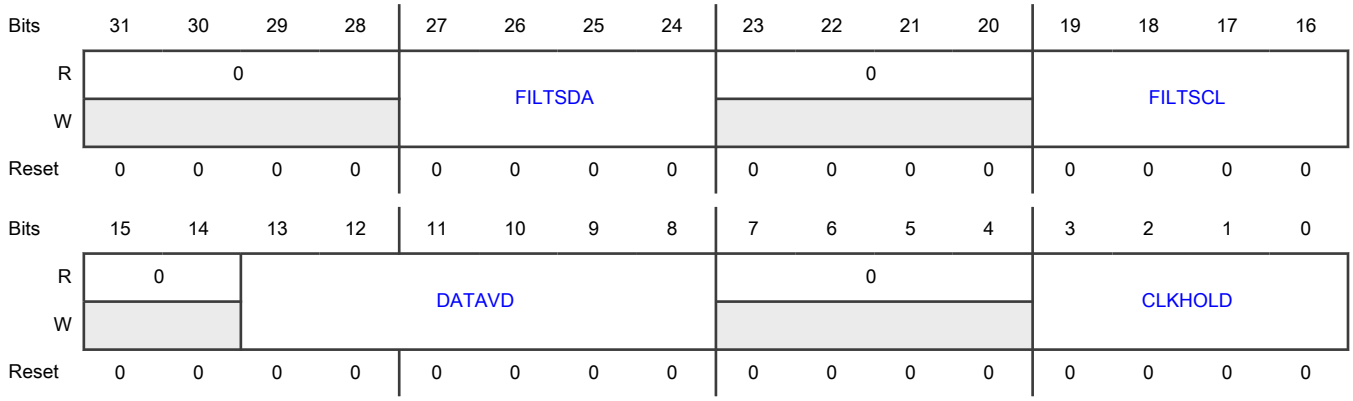
Register	Offset
SCFGR2	128h

Function

Configures data valid delay, clock hold time, and glitch filters for SDA and SCL.

Write to this register only when the I2C target is disabled.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C target digital glitch filters for SDA input. Writing 0 to this field disables the glitch filter. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. The latency through the glitch filter is equal to the number of cycles defined by this field + 3. The latency must be configured to be less than the minimum SCL low or high period. MCFGR1[PRESCALE] does not affect the glitch filter cycle count, and the glitch filter cycle count is disabled in HS mode.
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C target digital glitch filters for SCL input. Writing 0 to this field disables the glitch filter. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. The latency through the glitch filter is equal to the number of cycles defined by this field + 3. The latency must be configured to be less than the minimum SCL low or high period. MCFGR1[PRESCALE] does not affect the glitch filter cycle count, and the glitch filter cycle count is disabled in HS mode.
15-14 —	Reserved
13-8	Data Valid Delay

Table continues on the next page...

Table continued from the previous page...

Field	Function
DATAVD	Configures the SDA data valid delay time for the I2C target, which is equal to FILTSCL + DATAVD + 3 cycles. The data valid delay must be configured to be less than the minimum SCL low period. MCFGR1[PRESCALE] does not affect the I2C target data valid delay time, and the I2C target data valid delay time is disabled in HS mode.
7-4 —	Reserved
3-0 CLKHOLD	Clock Hold Time Configures the minimum clock hold time for the I2C target, when clock stretching is enabled. The minimum hold time is equal to the number of cycles defined by this field + 3. MCFGR1[PRESCALE] does not affect the I2C target clock hold time, and the I2C target clock hold time is disabled in HS mode.

62.7.1.27 Target Address Match (SAMR)

Offset

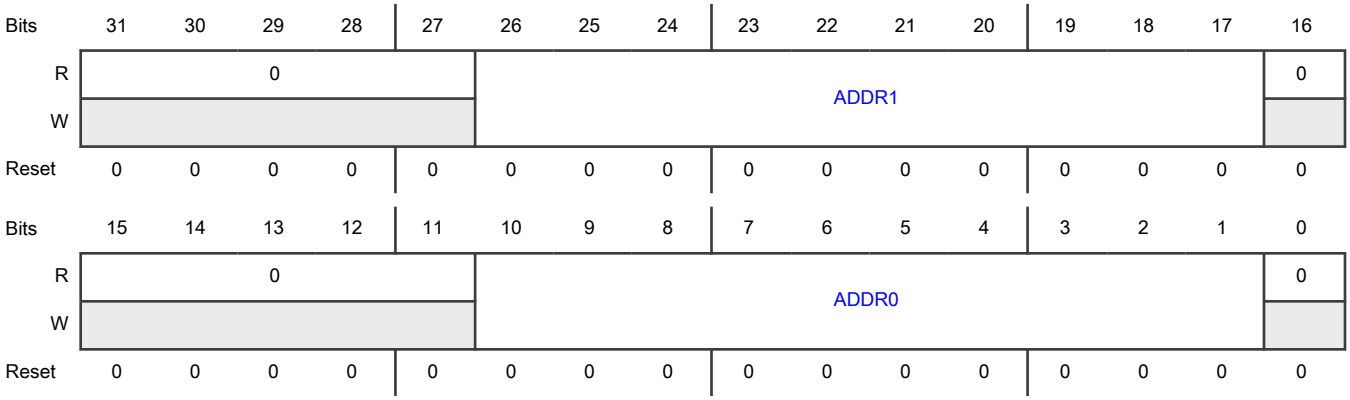
Register	Offset
SAMR	140h

Function

Contains address values for received target match comparison.

Write to this register only when the I2C target is disabled.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-17 ADDR1	Address 1 Value Contains the value of address 1, which is compared to the received address to detect the target address. In 10-bit mode, the first address byte is compared to {11110, ADDR1[26:25]} and the second address byte is compared to ADDR1[24:17]. In 7-bit mode, the address is compared to ADDR1[23:17].
16-11 —	Reserved
10-1 ADDR0	Address 0 Value Contains the value of address 0, which is compared to the received address to detect the target address. In 10-bit mode, the first address byte is compared to {11110, ADDR0[10:9]} and the second address byte is compared to ADDR0[8:1]. In 7-bit mode, the address is compared to ADDR0[7:1].
0 —	Reserved

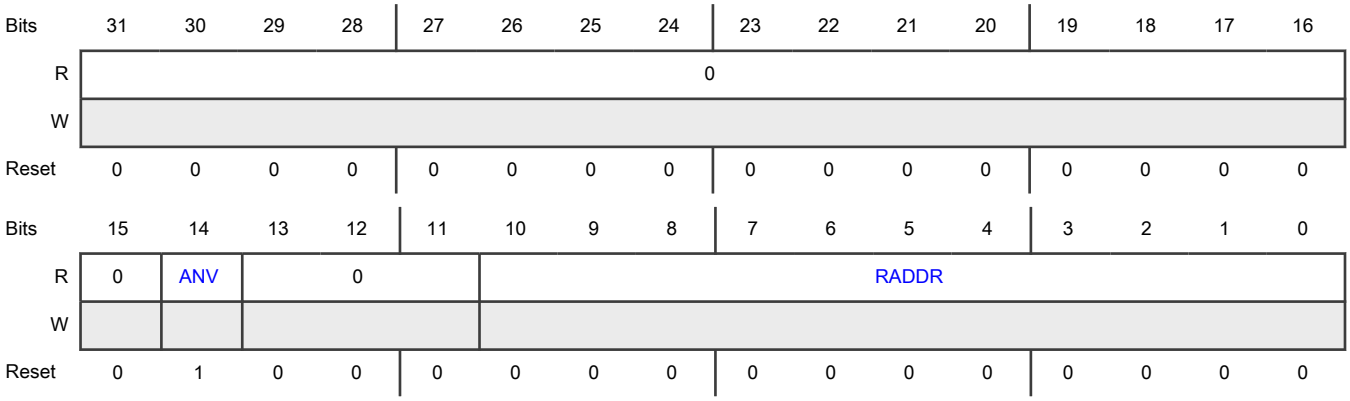
62.7.1.28 Target Address Status (SASR)

Offset

Register	Offset
SASR	150h

Function
Contains the received address and its validity.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 ANV	Address Not Valid Indicates whether SASR[RADDR] is valid. 0b - Valid 1b - Not valid
13-11 —	Reserved
10-0 RADDR	Received Address Contains the received address. Updates whenever SSR[AM0F] or SSR[AM1F] is set. Reading Target Address Status (SASR) clears SSR[AM0F] and SSR[AM1F]. In 7-bit mode, the address byte is stored in RADDR[7:0]. In 10-bit mode, the first address byte is {11110, RADDR[10:9], RADDR[0]} and the second address byte is RADDR[8:1]. The Read-or-Write bit is therefore always stored in RADDR[0]. When SCFGR1[ACKSTALL] = 1, if the first address byte matches in 10-bit mode, the first address byte is stored in RADDR[7:0] so you can read this field before writing the Transmit ACK. If the second address byte matches, this field is then updated with the full 10-bit address.

62.7.1.29 Target Transmit ACK (STAR)

Offset

Register	Offset
STAR	154h

Function

Configures choice of ACK or NACK on each received word.

You can write to this register only when SCFGR1[ACKSTALL] = 1.

SCFGR1[ACKSTALL] enables clock stretching during the ACK-or-NACK bit slot. During this time, you can write to this register.

The logic ensures that the clock stretching continues for at least one bus clock cycle after this register is updated.

This clock stretching time can be extended via SCFGR2[CLKHOLD].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															TXNACK
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 TXNACK	Transmit NACK Selects whether transmit ACK (logic 0) or NACK (logic 1) is returned on the bus by the I2C target after receiving each word. • When SCFGR1[ACKSTALL] = 1, a transmit NACK signal must be written once for each matching address byte and each received word. SCFGR1[ACKSTALL] must be 1, because that setting stalls the data transfer until software reads the received word (and determines whether to respond with an ACK or NACK). • To configure the default (ACK or NACK), you can write to this field when LPI2C target is disabled or idle. 0b - Transmit ACK 1b - Transmit NACK

62.7.1.30 Target Transmit Data (STDR)

Offset

Register	Offset
STDR	160h

Function

Contains the I2C target data to transmit.

Clock stretching (enabled or disabled) affects when the transmit data is transferred. SCFGR1[TXDSTALL] enables clock stretching during the first data bit of a target-transmit transfer.

If clock stretching is enabled (SCFGR1[TXDSTALL] = 1), the transmit data transfer is stalled until this register is updated. Clock stretching is extended by at least 1 bus clock cycle after this register is updated. Clock stretching can be delayed further by using SCFGR2[CLKHOLD].

If clock stretching is disabled (SCFGR1[TXDSTALL] = 0), the transmit data must be written before the start of the target-transmit transfer, otherwise SSR[FEF] is set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	0								DATA							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 DATA	Transmit Data Contains the I2C target data to transmit. Writing data to this register stores I2C target transmit data in this register.

62.7.1.31 Target Receive Data (SRDR)

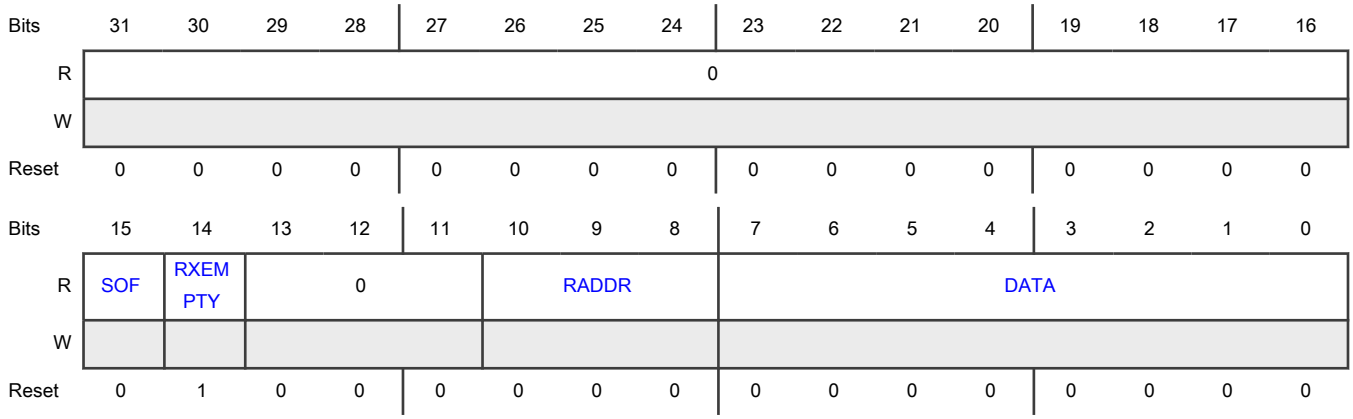
Offset

Register	Offset
SRDR	170h

Function

Contains status of target receive data transfer.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 SOF	Start of Frame Indicates whether this data word is the first data word since a (repeated) Start or Stop condition. 0b - Not first 1b - First
14 RXEMPTY	Receive Empty Indicates whether this register is empty. 0b - Not empty 1b - Empty
13-11 —	Reserved
10-8 RADDR	Received Address Contains the address received by the IC2 target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [10:8] of SASR[RADDR] are returned. Otherwise, this field returns zero.
7-0 DATA	Received Data Contains the data received by the I2C target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [7:0] of SASR[RADDR] are returned.

62.7.1.32 Target Receive Data Read Only (SRDROR)

Offset

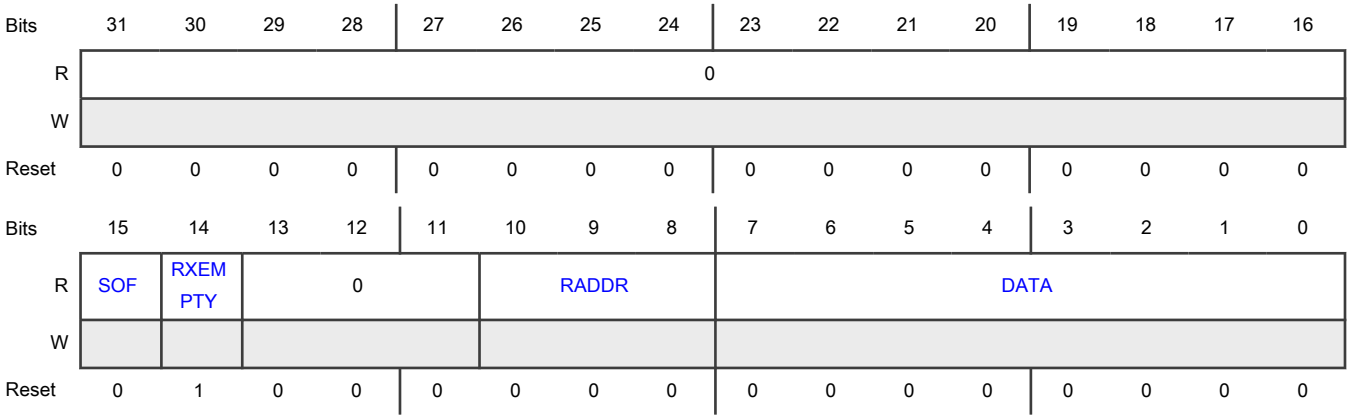
Register	Offset
SRDROR	178h

Function

Contains the data received by the I2C target.

Reading this register returns the data received by the I2C target, but does not pull the data from the register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 SOF	Start of Frame Indicates whether this data word is the first data word since a (repeated) Start or Stop condition. 0b - Not the first 1b - First
14 RXEMPTY	Receive Empty Indicates whether Target Receive Data (SRDR) is empty. 0b - Not empty 1b - Empty
13-11 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
10-8 RADDR	Received Address Contains address received by the LPI2C target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [10:8] of SASR[RADDR] are returned. Otherwise, this field returns zero.
7-0 DATA	Receive Data Contains data received by the LPI2C target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [7:0] of SASR[RADDR] are returned.

62.7.1.33 Controller Transmit Command Burst (MTCBR0 - MTCBR127)

Offset

For n = 0 to 127:

Register	Offset
MTCBRn	200h + (n × 4h)

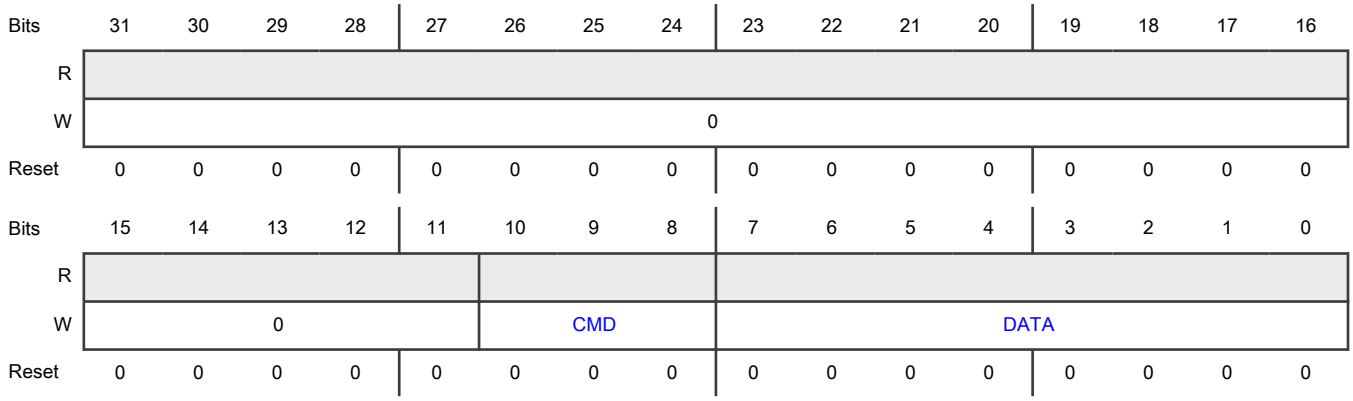
Function

Contains transmit Command and Data fields to support burst transfers.

Is an alias of the Controller Transmit Data Register designed to support incrementing burst transfers to the transmit FIFO by a DMA controller using aligned 8-bit, 16-bit, or 32-bit writes. The size of the Controller Transmit Data Burst Register is 512-bytes.

- An aligned 32-bit write in this region pushes one entry into the transmit FIFO.
- An aligned 16-bit write in this region to MTCBRn[15:0] pushes one entry into the transmit FIFO.
- An 8-bit write in this region to MTCBRn[7:0] updates DATA[7:0], but does not push the data into the transmit FIFO.
- An 8-bit write in this region to MTCBRn[15:8] pushes the data written to CMD[2:0] plus the previously written DATA[7:0] into the transmit FIFO.
- An 8-bit or 16-bit write in this region to MTXBRn[31:16] is ignored.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 CMD	Command Command is written to the Controller Transmit Data Register.
7-0 DATA	Data Data is written to the Controller Transmit Data Register.

62.7.1.34 Transmit Data Burst (MTDBR0 - MTDBR255)

Offset

For n = 0 to 255:

Register	Offset
MTDBRn	400h + (n × 4h)

Function

Contains transmit Data fields to support burst transfers.

Is an alias of the Transmit Data Register designed to support incrementing burst transfers to the transmit FIFO by a DMA controller using 8-bit, 16-bit, or 32-bit writes. The CMD field is always zero-extended for transmit data. The size of the Transmit Data Burst Register is 1024 bytes.

- An aligned 32-bit write in this region pushes four zero-extended DATA entries into the transmit FIFO (DATA0 byte first). Note that the register access is extended by 3 wait states.
- An aligned 16-bit write in this region to either half of a 32-bit word pushes two zero-extended DATA entries into the transmit FIFO (DATA0 or DATA2 first). Note that the register access is extended by 1 wait state.
- An 8-bit write in this region pushes one zero-extended DATA entry into the transmit FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	DATA3								DATA2							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	DATA1								DATA0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 DATA3	Data Data is written to the MTDR[DATA] with MTDR[CMD] zero-extended.
23-16 DATA2	Data Data is written to the MTDR[DATA] with MTDR[CMD] zero-extended.
15-8 DATA1	Data Data is written to the MTDR[DATA] with MTDR[CMD] zero-extended.
7-0 DATA0	Data Data is written to the MTDR[DATA] with MTDR[CMD] zero-extended.

Chapter 63
Low Power Serial Peripheral Interface (LPSPI)

63.1 Chip-specific LPSPI Information

Each instance of LPSPI has 4 peripheral chip selects (PCS).

Table 1024. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

NOTE

The LPSPI output triggers are not connected.

63.2 Overview

LPSPI provides an efficient interface to a SPI bus, either as a master or slave. A SPI bus is a synchronous serial communication interface used in embedded systems. It is typically used to perform short distance communications between microcontrollers and peripheral devices, on printed circuit boards. Typical applications include interfacing to Secure Digital cards and LCD displays.

63.2.1 Block diagram

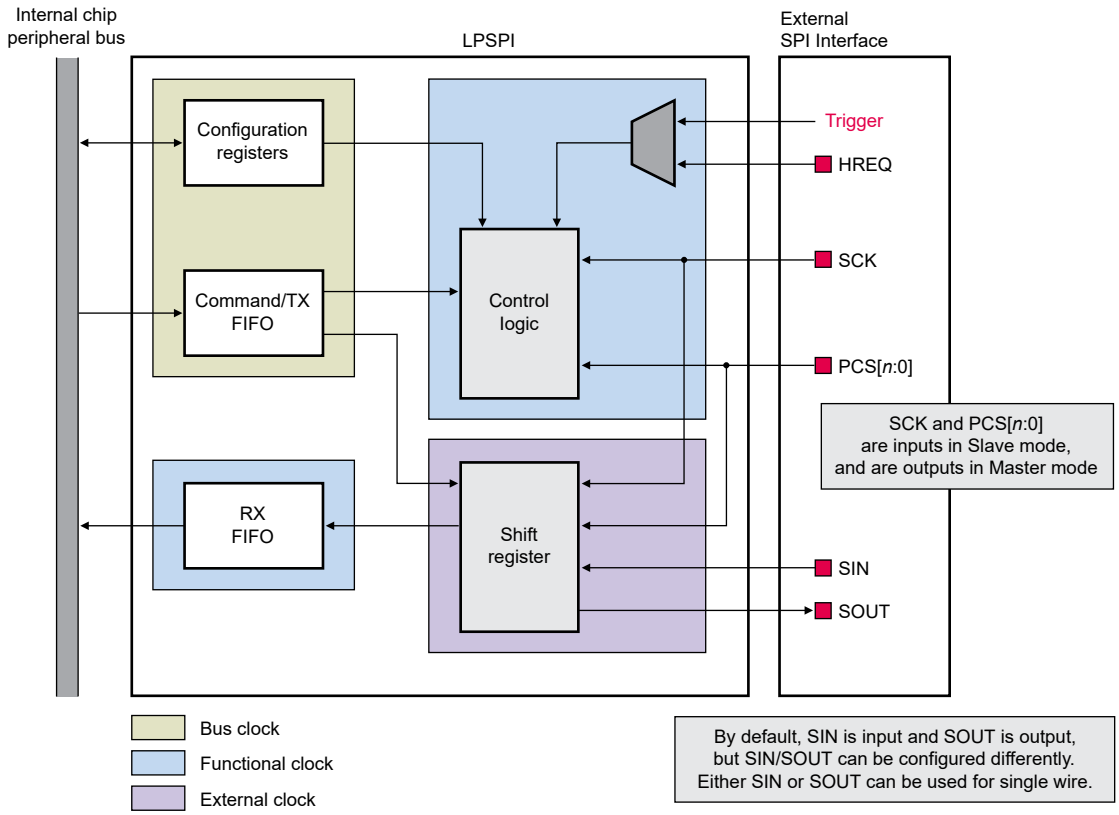


Figure 467. Block diagram

63.2.2 Features

- Requires minimal CPU overhead, with FIFO register access supported by DMA transmit and receive requests.
- Continues operating in Stop mode, if configured to do so and an appropriate clock is available
- 32-bit word size
- Configurable clock polarity and phase
- Master mode—supports 4 peripheral chip selects
- Slave mode
- 16-word transmit and command FIFO
- 16-word receive FIFO
- Flexible timing parameters in Master mode, including SCK frequency and duty cycle, and delays between PCS and SCK edges
- Continuous transfer option to keep PCS asserted across multiple frames
- Full-duplex transfers support 1-bit transmit and receive on each clock edge
- Half-duplex transfers support:
  - 1-bit transmit or receive on each clock edge
  - 2-bit transmit or receive on each clock edge
  - 4-bit transmit or receive on each clock edge

- Can use host request to control the start of a SPI bus transfer
- Receive data match logic supports discard of non-matching data and interrupt on data match

63.3 Functional description

63.3.1 Master mode

63.3.1.1 Transmit and command FIFO commands

The transmit and command FIFO is a combined FIFO that includes both transmit data words and command words.

- You store transmit data words to the transmit and command FIFO, by writing to Transmit Data (TDR).
- You store command words to the transmit and command FIFO, by writing to Transmit Command (TCR).

When a command word is at the top of the transmit and command FIFO, the actions that can occur depend upon whether the LPSPI module is busy or between frames. See TCR[CONT] and TCR[CONTC].

Table 1025. Possible actions when a command word is at the top of the transmit and command FIFO

Condition	Action
LPSPI is enabled and idle	The command word is pulled from the FIFO, and that command word controls all subsequent transfers.
LPSPI is busy and the Continuing Command field (TCR[CONTC]) is 0	The SPI frame completes at the end of the existing word, ignoring TCR[FRAMESZ]. The command word is then pulled from the FIFO and that command word controls all subsequent transfers (or until the next update to the command word). Note that a command word with TCR[CONTC] = 0 always terminates the existing transfer regardless of the previous TCR[CONT] value.
LPSPI is busy, the existing TCR[CONT] value is 1 and the new TCR[CONTC] value is 1	The command word must be updated at the frame boundary. The command word is pulled from the FIFO during the last SCK pulse of the existing frame (based on the FRAMESZ value), and the frame continues using the new command value for the rest of the frame (or until the next update to the command word). When TCR[CONTC] = 1, only the lower 24-bits of the command word are updated. If the command word is updated at a word boundary, then the transfer halts (stops) after that word. TCR[CONTC] is ignored when not at a frame boundary, so the frame ends prematurely.

About TCR[CONT] and TCR[CONTC]:

- TCR[CONT] = 1 keeps PCS asserted at end of frame, allowing the transfer to continue.
- TCR[CONTC] = 1 specifies that this command word should not terminate the existing frame, and the transfer can continue using the new command word.

TCR[CONTC] = 1 is restricted in the sense that the new command must load on a frame boundary, and the only way for a transfer to continue from a frame boundary is when the previous command has TCR[CONT] = 1.

You can read the current state of the existing command word from Transmit Command (TCR). It requires at least 3 LPSPI functional clock cycles for TCR to update after TCR is written (assuming an empty FIFO), and LPSPI must be enabled (CR[MEN] = 1).

Writing the TCR does not initiate a SPI bus transfer, unless the TCR[TXMSK] = 1. When the TCR[TXMSK] = 1, a new command word is not loaded until the end of the existing frame (based on the TCR[FRAMESZ] value); at the end of the transfer, TCR[TXMSK] transitions to 0.

In Master mode, the LPSPI command word in TCR controls SPI attributes based on selections in register fields.

Table 1026. Command word in Master mode

Transmit Command (TCR)		Description	Can this field be modified during a data transfer?
Field	Name		
CPOL	Clock polarity	Specifies the polarity of the SCK pin. Any change of CPOL value causes a transition on the SCK pin.	N
CPHA	Clock phase	Specifies the clock phase of the transfer.	N
PRESCALE	Prescaler value	Specifies a prescaler used to divide the LPSPI functional clock, to generate the timing parameters of the SPI bus transfer. Changing PRESCALE in conjunction with PCS enables the LPSPI module to connect to different slave devices at different frequencies.	N
PCS	Peripheral chip select	Specifies which PCS pin asserts for the transfer; the polarity of PCS is static and specified by CFGR1[PCSPOL]. If CFGR1[PCSCFG] = 1, do not select PCS[3:2].	N
LSBF	LSB first	Specifies whether LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.	Y
BYSW	Byte swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing to devices that organize data as big-endian.	Y
CONT	Continuous transfer	Configures LPSPI for a continuous transfer that keeps PCS asserted between frames (as specified by FRAMESZ). You must write a new command word to cause PCS to negate. Also supports changing the command word at the frame size boundaries.	Y
CONTC	Continuing command	Indicates that this is a new command word for the existing continuous transfer. When CONTC = 1, the command word must only be written to the transmit and command FIFO on a frame boundary.	Y
RXMSK	Receive data mask	Masks the receive data and does not store the masked receive data to the receive FIFO or perform receive data matching. This option is useful for half-duplex transfers or to specify which fields are compared during receive data matching.	Y
TXMSK	Transmit data mask	Masks the transmit data; masked transmit data is not pulled from the transmit FIFO, and the output data pin is 3-stated (unless otherwise configured by CFGR1[OUTCFG]). This option is useful for half-duplex transfers.	Y
WIDTH	Transfer width	Specifies the number of bits shifted on each SCK pulse. • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. • 2-bit and 4-bit half-duplex transfers are useful for interfacing to QuadSPI memory devices, and either TXMSK or RXMSK must also be 1.	Y
FRAMESZ	Frame size	Configures the frame size in number of bits equal to (FRAMESZ + 1).	Y

Table continues on the next page...

Table 1026. Command word in Master mode (continued)

Transmit Command (TCR)		Description	Can this field be modified during a data transfer?
Field	Name		
		• The minimum frame size is 8 bits. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contains the remaining bits. For example, a 72-bit transfer consists of 3 words: the first and second words are 32 bits, and the third word is 8 bits.	

63.3.1.1.1 SPI bus transfers

LPSPI initiates a SPI bus transfer when all of the following conditions are true:

- Data is written to the transmit FIFO.
- The HREQ pin is asserted (or the HREQ function is disabled).
- LPSPI is enabled.

To perform the SPI bus transfer, LPSPI uses the attributes configured in Transmit Command (TCR) and uses the timing parameters in Clock Configuration (CCR).

The SPI bus transfer ends after the number of bits indicated by the FRAMESZ value have been transferred (provided CONT = 0), or at the end of a word when a new transmit command word is at the top of the transmit and command FIFO. When LPSPI is disabled, the SPI bus transfers end after the transmit FIFO is empty and LPSPI is idle.

The HREQ input is only checked when PCS is negated.

63.3.1.1.2 Circular FIFO

The transmit and command FIFO supports a circular FIFO feature. This feature enables the LPSPI master to (periodically) repeat a short data transfer that fits within the transmit and command FIFO, without requiring additional FIFO accesses. When the circular FIFO is enabled (CFGR0[CIRFIFO] = 1), the current state of the FIFO read pointer is saved and the status flags do not update. After the FIFO is empty and LPSPI is idle, the FIFO read pointer is restored with the saved version, so the contents of the transmit and command FIFO are not permanently pulled from the FIFO while Circular FIFO mode is enabled.

63.3.1.2 Receive FIFO and data match

The receive FIFO stores received data during SPI bus transfers. When TCR[RXMSK] = 1, the received data is discarded instead of being stored in the receive FIFO.

- Received data is written to the receive FIFO when the last bit of the word is sampled.
- If the transmit FIFO is empty during a multiple-word or continuous transfer, then the receive data is written to the receive FIFO before the transfer stalls (assuming CFGR1[NOSTALL] = 0) while waiting for new transmit data or for a command word to be written.

LPSPI provides a receive data match function that can match received data against one of two words in DMR0 and DMR1, or against a masked data word. The received data match function can also be configured to compare only the first one or two received data words since the start of the frame.

- Received data that is already discarded due to the TCR[RXMSK] field cannot cause the data match flag to set, and delays the receive data match on the first received data word, until all discarded data is received.

- You can configure the receive data match function to discard all received data until a data match is detected, using CFGR0[RDMO].
- After a receive data match, to allow all subsequent data to be received, write 0 to the CFGR0[RDMO] field, then write 0 to SR[DMF].

63.3.1.3 Timing parameters

The timing parameters that are used for all SPI bus transfers are relative to the LPSPI functional clock divided by the TCR[PRESCALE] selection. Although you cannot change Clock Configuration (CCR) when the LPSPI module is busy, to support interfacing to different slave devices at different frequencies, you can change the TCR[PRESCALE] selection between SPI bus transfers using Transmit Command (TCR).

NOTE

The minimum value below is the minimum counter value, but the clock configuration register values must also satisfy the data sheet specs based on the LPSPI functional clock frequency and prescaler value.

Table 1027. Timing parameters

Clock Configuration (CCR) Clock Configuration 1 (CCR1)		Description	Min	Max
Field	Name			
SCKSET	SCK setup phase	Configures the SCK setup phase to (SCKSET + 1) cycles. The setup phase is the SCK high period when either CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. Otherwise, it is the SCK low period. The SCK period is defined as (SCKSET + SCKHLD + 2) and the duty cycle is the difference between SCKSET and SCKHLD.	0 (1 cycle)	255 (256 cycles)
SCKHLD	SCK hold phase	Configures the SCK hold phase to (SCKHLD + 1) cycles. The hold phase is the SCK low period when either CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. Otherwise, it is the SCK high period. The SCK period is defined as (SCKSET + SCKHLD + 2) and the duty cycle is the difference between SCKSET and SCKHLD.	0 (1 cycle)	255 (256 cycles)
PCSPCS	PCS-to-PCS delay	Configures the minimum delay between PCS negation and the next PCS assertion to (PCSPCS + PCSPCS + 2) cycles. When the command word is updated between transfers, there is a minimum of (PCSPCS + 1) cycles between the command word update and any change on PCS pins.	0 (2 cycles)	255 (512 cycles)
SCKSCK	SCK-to-SCK delay	Configures the delay during a continuous transfer between the last SCK edge of a frame and the first SCK edge of the continuing frame to (SCKSCK + 1) cycles. This is useful when the external slave requires a large delay between different words of an SPI bus transfer.	0 (1 cycle)	255 (256 cycles)

Table continues on the next page...

Table 1027. Timing parameters (continued)

Clock Configuration (CCR) Clock Configuration 1 (CCR1)		Description	Min	Max
Field	Name			
PCSSCK	PCS-to-SCK delay	Configures the minimum delay between PCS assertion and the first SCK edge to (PCSSCK + 1) cycles.	0 (1 cycle)	255 (256 cycles)
SCKPCS	SCK-to-PCS delay	Configures the minimum delay between the last SCK edge and the PCS negation to (SCKPCS + 1) cycles.	0 (1 cycle)	255 (256 cycles)

This figure shows the timing settings controlled by:

- TCR[CPHA]
- TCR[CPOL]
- CCR[SCKPCS]
- CCR[PCSSCK]
- CCR1[SCKSET]
- CCR1[SCKHLD]

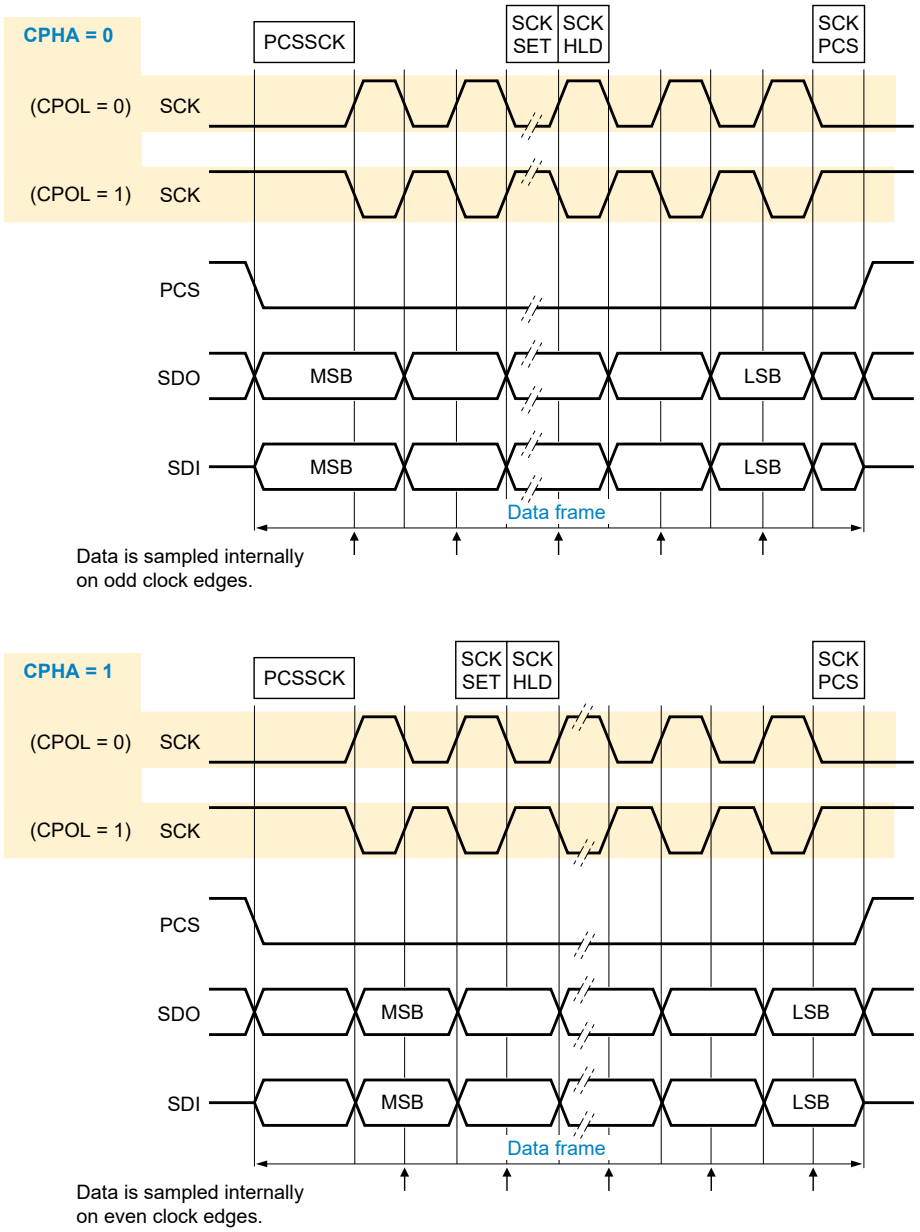


Figure 468. Clock phase (TCR[CPHA]) timing diagram example

To configure for a baud rate of 10 MHz with 50/50 duty cycle and with a functional clock frequency of 100 MHz, use the following settings:

- CCR1[SCKSET] = 0x4 (5 cycles)
- CCR1[SCKHLD] = 0x4 (5 cycles)
- CCR1[PCSPCS] = 0x8 (10 cycles)
- CCR1[SCKSCK] = 0x4 (5 cycles)
- CRR[PCSSCK] = 0x4 (5 cycles)
- CRR[SCKPCS] = 0x4 (5 cycles)
- TCR[PRESCALE] = 0x0 (divide by 1)

63.3.1.4 Pin configuration

- To swap directions or support half-duplex transfers on the same pin, you can configure the SIN and SOUT pins using CFGR1[PINCFG].
- To specify whether an output data pin (SOUT, for example) 3-states when PCS is negated, or if the output data pin retains the last value, use CFGR1[OUTCFG].
- When configuring for half-duplex transfers, you must configure the output data pins to 3-state when PCS is negated (CFGR1[OUTCFG] = 1).
- When performing half-duplex 2-bit transfers, you can set CFGR1[PCSCFG] to any value.
- When performing half-duplex 4-bit transfers, you must set CFGR1[PCSCFG] to 1h.

63.3.1.5 Clock loopback

Configure the LPSPI master to use one of the following clocks to sample the input data:

- The SCK output clock
- A delayed version of the SCK output clock

The delayed version of the SCK is chosen by the SCK pin output delay, plus the SCK pin input delay, and is selected by writing 1 to CFGR1[SAMPLE]. Enabling the loopback version of the SCK can improve the setup time of the input data from the slave.

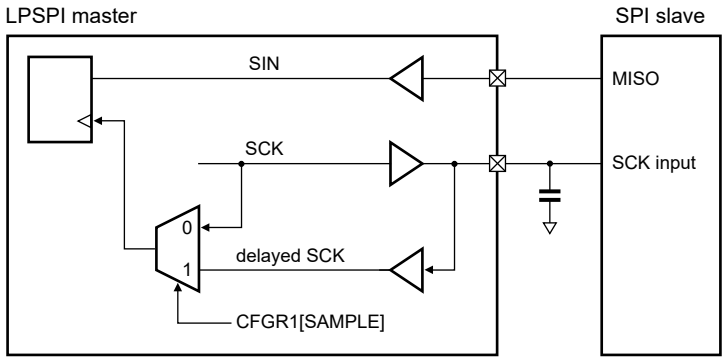


Figure 469. Clock loopback

See the chip data sheet for the specific input setup time in Master Loopback mode.

63.3.2 Slave mode

LPSPI slave mode:

- Uses the same shift register and logic that master mode uses.
- Does not use Clock Configuration Register (CCR).
- Requires that Transmit Command (TCR) remain static (unchanging) during SPI bus transfers.

63.3.2.1 Transmit and command FIFO commands

Before enabling LPSPI in Slave mode, initialize Transmit Command (TCR), although TCR does not update until after LPSPI is enabled. After LPSPI is enabled, only change TCR when LPSPI is idle. In Slave mode, the LPSPI command word in TCR controls SPI attributes. Before the PCS input asserts, the transmit FIFO must be filled with transmit data, or the transmit error flag sets.

Table 1028. Command word in Slave mode

Transmit Command (TCR)		Description
Field	Name	
CPOL	Clock polarity	Specifies the polarity of the external SCK input.
CPHA	Clock phase	Specifies the clock phase of transfer.
PRESCALE	Prescaler value	Specifies the LPSPI functional clock prescaler.
PCS	Peripheral chip select	Specifies which PCS is used. The polarity of PCS is static and configured by CFGR1[PCSPOL]. If CFGR1[PCSCFG] is not equal to zero, then do not select the PCS[3:2] pins.
LSBF	LSB first	Specifies whether LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.
BYSW	Byte swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing to devices that organize data as big-endian.
CONT	Continuous transfer	When continuous transfer is selected in Slave mode, after the number of bits indicated by FRAMESZ are transferred, the LPSPI will passthrough and transmit the received data until the next PCS negation. Whatever is shifted in on the receive data is shifted out as transmit data as if there were a 32-bit shift register.
CONTC	Continuing command	When continuing command is enabled in Slave mode, after the number of bits indicated by FRAMESZ are transferred, then RXMSK is considered equal to 1 and TXMSK is considered equal to 0 until the next PCS negation. CONTC can be used to change the direction of a transfer after the number of bits indicated by FRAMESZ.
RXMSK	Receive data mask	Masks the receive data; LPSPI does not store masked receive data to the receive FIFO or perform receive data matching. This option is useful for half-duplex transfers or to specify which fields are compared during receive data matching.
TXMSK	Transmit data mask	Masks the transmit data, so that the masked transmit data is not pulled from transmit FIFO, and the output data pin is 3-stated (unless otherwise specified by CFGR1[OUTCFG]). This option is useful for half-duplex transfers.
WIDTH	Transfer width	Specifies the number of bits shifted on each SCK pulse. • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. • 2-bit and 4-bit half-duplex transfers are useful for interfacing to QuadSPI memory devices, and at least one of TCR[TXMSK] or TCR[RXMSK] must be 1.
FRAMESZ	Frame size	Specifies the frame size in number of bits equal to (FRAMESZ + 1). • The minimum frame size is 8 bits. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contain the remainder bits. For example, a 72-bit transfer consists of 3 words: the first and second words are 32 bits, and the third word is 8 bits.

63.3.2.2 Receive FIFO and data match

The receive FIFO stores receive data during SPI bus transfers. When TCR[RXMSK] = 1, the received data is discarded instead of storing the received data in the receive FIFO.

Receive data supports a receive data match function that can match received data against one of two words in the DMR0 and DMR1 registers or against a masked data word. The data match function can also be configured to compare only the first one or two received data words since the start of the frame.

- Received data that is already discarded because TCR[RXMSK] = 1 cannot cause the data match to set, and delays the match on the first received data word, until all discarded data is received.
- The receiver match function can also be configured to discard all received data until a data match is detected, using the CFGR0[RDMO] bit.
- After a receive data match, to allow all subsequent data to be received, first clear the CFGR0[RDMO] bit, then clear the SR[DMF] bit.

63.3.2.3 Partial received word

When the PCS pin deasserts and the receive shift register has shifted in a partial word, the receive shift register can be configured to either discard the partial word or to store it in the receive FIFO. You specify this using CFGR1[PARTIAL].

A partial word is defined as less than FRAMESZ bits (when FRAMESZ is equal or less than 32 bits, or it is the last word in a multi-word frame) or less than 32 bits (when FRAMESZ is greater than 32 bits and not the last word in a multi-word frame).

A single-bit frame is not supported. A partial received word of 1 bit is fine, but a partial received frame of 1 bit is not supported.

63.3.2.4 Clocked interface

LPSPI supports interfacing to external masters that provide only clock and data pins (PCS is not required). This interface requires:

- Writing 1 to TCR[CPHA] (data is changed on the leading edge of SCK and captured on the following edge).
- Configuring the PCS input to be always asserted (CFGR1[PCSPOLn] = 1). For example, to configure PCS[0] to be always asserted, write 1 to PCSPOL[0], and do not configure PCS[0] in the pin muxing. The chip-level drives PCS to a certain value (ideally 1), CFGR1[PCSPOL*n*] could be used to invert that value.
- Writing 1 to CFGR1[AUTOPCS] to enable automatic PCS generation. When CFGR1[AUTOPCS] = 1, a minimum of four LPSPI functional clock cycles (divided by the TCR[PRESCALE] selection) is required between the last SCK edge of one word and the first SCK edge of the next word.

63.3.3 Low-power modes

Table 1029. Low-power modes

Chip mode	LPSPI operation
Run	Normal operation
Stop	Can continue operating in Stop mode if CR[DOZEN] = 0 and LPSPI is using an external or internal clock source that remains operating during Stop mode

63.3.4 Debug mode

Table 1030. Debug mode

Chip mode	LPSPI operation
Debug (the core is in Debug/ Halted mode)	Can continue operating in Debug mode, if the CR[DBGEN] = 1.

63.3.5 Interrupts and DMA requests

The following table lists the Slave mode sources (status flags) that can generate LPSPI interrupts and LPSPI slave transmit and receive DMA requests.

Table 1031. Interrupts and DMA requests

Status (SR)		Description	Can generate		
Status flag	Name		Interrupt?	DMA request?	Low-power wake-up?
TDF	Transmit data flag	Data can be written to transmit FIFO, as configured by the transmit FIFO watermark FCR[TXWATER]	Y	TX	Y
RDF	Receive data flag	Data can be read from the receive FIFO, as configured by the receive FIFO watermark FCR[RXWATER]	Y	RX	Y
WCF	Word complete flag	Word is complete, the last bit of the word has been sampled	Y	N	Y
FCF	Frame complete flag	Frame is complete, and PCS has deasserted	Y	RX	Y
TCF	Transfer complete flag	Transfer is complete, PCS has deasserted, and the transmit and command FIFO is empty	Y	N	Y
TEF	Transmit error flag	Indicates a transmit and command FIFO underrun. In Master mode when CFGR1[NOSTALL] = 0 (transfers stall when transmit FIFO is empty), the Transmit Error Flag cannot set.	Y	N	Y
REF	Receive error flag	Indicates a receive FIFO overflow. In Master mode when CFGR1[NOSTALL] = 0 (transfers stall when receive FIFO is full), the Receive Error Flag cannot set.	Y	N	Y
DMF	Data match flag	Indicates that the received data has matched the configured data match value	Y	N	Y
MBF	Module busy flag	LPSPI is busy performing a SPI bus transfer	N	N	N

63.3.5.1 End of packet DMA transfer

The end of packet functionality is designed for serial interfaces where the size of the transfer may not be known by software in advance and the data is being pushed by an external device. Examples include UART receive, I2C Slave mode and SPI Slave

mode. The end of packet processing is intended to ensure data does not become stranded in either the receive FIFO or the DMA receive buffer. Support for end of packet processing must be implemented in both the serial interfaces and the DMA controller.

The condition that signals the end of packet is different for each serial interface but is processed by the serial peripheral and DMA in the same way. For example, UART end of packet is signaled by idle line condition, I2C end of packet by STOP and/or Repeated START condition, and SPI end of packet by PCS negation.

When the serial peripheral is configured to signal end of packet condition to the DMA and an end of packet condition is detected by the serial interface, it asserts the DMA request for the receive FIFO irrespective of the watermark configuration. For larger watermark configurations, this ensures the last few words of the transfer are first flushed from the receive FIFO.

The DMA then reads the contents of the receive FIFO, depending on the first word in the FIFO:

- If the receive FIFO is empty, the serial interface signals an end of packet condition to the DMA controller.
- If the receive FIFO is not empty, but the first word in the FIFO is the start of a new packet, then data is not pulled from the receive FIFO and the serial interface signals an end of packet condition to the DMA controller.
- If the receive FIFO is not empty, and the first word in the FIFO is not the start of a new packet, the DMA will transfer the receive data as normal.

Since the DMA may be transferring multiple words on each request, the end of packet condition persists until the DMA minor loop has completed and no additional data is pulled from the receive FIFO. The status flag that triggered the end of packet condition is cleared when the minor loop completes following end of packet being signaled to the DMA controller.

When the DMA detects the end of packet condition, it writes all received words up to the end of packet into system memory and saves the destination address for the word after the last valid data. The DMA then terminates the channel as if the major loop completed, including final offsets and optional interrupts, channel linking and scatter/gather. The final destination address can optionally be saved to system memory or is available in the destination address register.

Since the DMA terminates the major loop, no servicing of the receive FIFO occurs until the DMA is reconfigured through either software or hardware (for example, channel linking or scatter-gather). This delay should be minimized to avoid receiver FIFO overrun. The automatic DMA end of packet processing is not recommended when there are only a few words transferred between end of packet conditions since the DMA will spend more time processing the end of packet than transferring the data. For example, the UART idle line length should be increased as needed to avoid an excessive number of idle conditions.

63.3.6 Clocks

Table 1032. LPSPI clocks

LPSPI functional clock	• Asynchronous to the bus clock. • If the LPSPI functional clock remains enabled in low-power modes, then LPSPI can perform SPI bus transfers and low-power wakeups, in both Master and Slave modes. • LPSPI divides the functional clock by a prescaler; the resulting frequency must be at least 2 times faster than the SPI external clock frequency (SCK).
External clock	• The LPSPI shift register is clocked directly by the SCK clock. • How the SCK clock is generated or supplied depends upon the mode (Master or Slave): — In Master mode: the SCK clock is generated internally. — In Slave mode: the SCK clock is supplied externally.
Bus clock	The bus clock is only used for bus accesses to the LPSPI control and configuration registers. The bus clock frequency must be high enough to support the data bandwidth requirements of the LPSPI registers, including the FIFOs.

See the chip-specific LPSPI information.

63.3.7 Resets

Table 1033. Resets

Chip reset	Resets the LPSPI logic and registers to their default state.
Software reset	• Resets the LPSPI logic and registers to their default state, except for the Control Register. • The LPSPI software reset is controlled using CR[RST].
FIFO resets	• Resets the transmit and command FIFO and the receive FIFO. • The Reset Transmit FIFO field, CR[RTF], and the Reset Receive FIFO field, CR[RRF], are write only. • After being reset, a FIFO is empty.

63.3.8 Peripheral triggers

The connection of the LPSPI peripheral triggers to other peripherals depend upon the specific device being used.

Table 1034. Peripheral triggers

Trigger	Description	Notes
Frame output trigger	The frame output trigger: • Asserts at the end of each frame (when PCS deasserts) • Remains asserted for one cycle of LPSPI functional clock divided by the PRESCALE configuration	LPSPI generates two output triggers that can be connected to other peripherals on the chip.
Word output trigger	The word output trigger: • Asserts at the end of each received word • Remains asserted for one cycle of LPSPI functional clock divided by the PRESCALE configuration	
Input trigger	To control the start of an LPSPI bus transfer, the LPSPI input trigger can be selected instead of the HREQ input. • The LPSPI input trigger is synchronized, and must assert for at least 2 cycles of the LPSPI functional clock divided by the PRESCALE configuration, so that the input trigger can be detected. • When the LPSPI module is busy, the HREQ input (and therefore the LPSPI input trigger) is ignored. • Both HREQ and the LPSPI input trigger are ignored when the module is busy. They are used to start a new transfer when the module is idle.	

63.4 Signals

Table 1035. Signals

Signal	Name	Description	I/O
SCK	Serial clock	• Input in Slave mode • Output in Master mode	I/O

Table continues on the next page...

Table 1035. Signals (continued)

Signal	Name	Description	I/O
PCS[0]	Peripheral chip select	• Input in Slave mode • Output in Master mode	I/O
PCS[1] / HREQ	Peripheral chip select or host request	Host request pin is selected when CFGR0[HREN] = 1 and CFGR0[HRSEL] = 0 • Input in either Slave mode or when used as master host request • Output in either Master mode or when used as slave host request	I/O
PCS[2] / DATA[2]	Peripheral chip select or data pin 2 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Slave mode • Output in Master mode When CFGR1[PCSCFG] = 1: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
PCS[3] / DATA[3]	Peripheral chip select or data pin 3 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Slave mode • Output in Master mode When CFGR1[PCSCFG] = 1: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
SOUT / DATA[0]	Serial data output	Can be configured as serial data input signal • Used as data pin 0 in half-duplex parallel data transfers	I/O
SIN / DATA[1]	Serial data input	Can be configured as serial data output signal • Used as data pin 1 in half-duplex parallel data transfers	I/O

63.5 Memory map and registers

NOTE

- Writing to a read-only register or reading a write-only register can cause bus errors.
- LPSPI does not check values programmed in registers for validity, so you must take care to write valid values only.

63.5.1 LPSPI register descriptions

LPSPI provides an efficient interface to a SPI bus, either as a master or slave. A SPI bus is a synchronous serial communication interface used in embedded systems. It is typically used to perform short distance communications between microcontrollers and peripheral devices, on printed circuit boards. Typical applications include interfacing to Secure Digital cards and LCD displays.

63.5.1.1 LPSPI memory map

LPSPI1 base address: 4436_0000h

LPSPI2 base address: 4437_0000h

LPSPI3 base address: 4255_0000h

LPSPI4 base address: 4256_0000h

LPSPI5 base address: 42D5_0000h

LPSPI6 base address: 42D6_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0200_0004h
4h	Parameter (PARAM)	32	R	0004_0404h
10h	Control (CR)	32	RW	0000_0000h
14h	Status (SR)	32	RW	0000_0001h
18h	Interrupt Enable (IER)	32	RW	0000_0000h
1Ch	DMA Enable (DER)	32	RW	0000_0000h
20h	Configuration 0 (CFGR0)	32	RW	0000_0000h
24h	Configuration 1 (CFGR1)	32	RW	0000_0000h
30h	Data Match 0 (DMR0)	32	RW	0000_0000h
34h	Data Match 1 (DMR1)	32	RW	0000_0000h
40h	Clock Configuration (CCR)	32	RW	0000_0000h
44h	Clock Configuration 1 (CCR1)	32	RW	0000_0000h
58h	FIFO Control (FCR)	32	RW	0000_0000h
5Ch	FIFO Status (FSR)	32	R	0000_0000h
60h	Transmit Command (TCR)	32	RW	0000_001Fh
64h	Transmit Data (TDR)	32	W	0000_0000h
70h	Receive Status (RSR)	32	R	0000_0002h
74h	Receive Data (RDR)	32	R	0000_0000h
78h	Receive Data Read Only (RDROR)	32	R	0000_0000h
3FCh	Transmit Command Burst (TCBR)	32	W	0000_0000h
400h - 5FCh	Transmit Data Burst (TDBR0 - TDBR127)	32	W	0000_0000h
600h - 7FCh	Receive Data Burst (RDBR0 - RDBR127)	32	R	0000_0000h

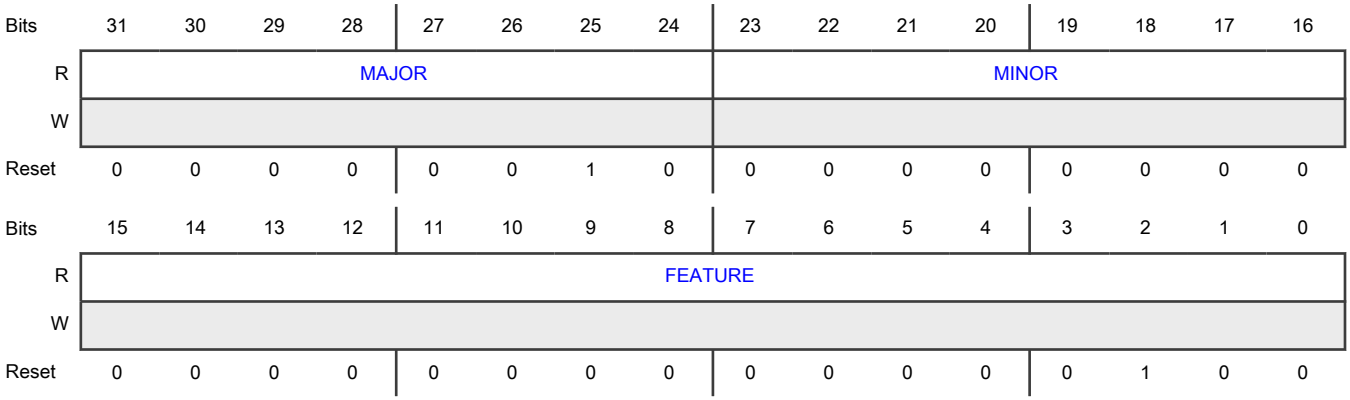
63.5.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function
Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the module specification.
23-16 MINOR	Minor Version Number Indicates the minor version number for the module specification.
15-0 FEATURE	Module Identification Number Indicates the feature set number. 0000_0000_0000_0100b - Standard feature set supporting a 32-bit shift register. All other values are reserved.

63.5.1.3 Parameter (PARAM)

Offset

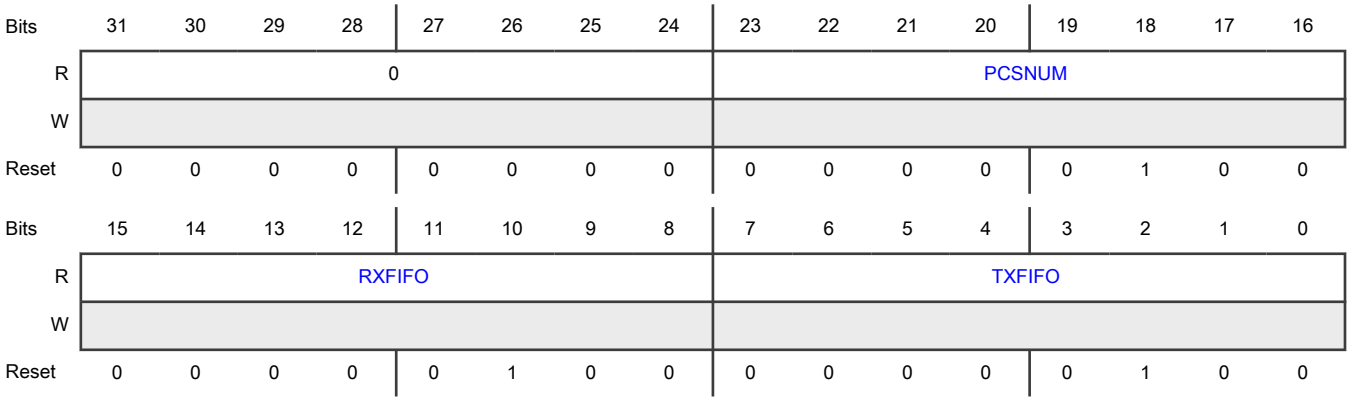
Register	Offset
PARAM	4h

Function

Contains:

- Number of PCS pins
- Receive FIFO size
- Transmit FIFO size

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 PCSNUM	PCS Number Indicates the number of PCS pins supported.
15-8 RXFIFO	Receive FIFO Size Indicates the maximum number of words in the receive FIFO. The maximum number of words is 2^{RXFIFO}.
7-0 TXFIFO	Transmit FIFO Size Indicates the maximum number of words in the transmit FIFO. The maximum number of words is 2^{TXFIFO}.

63.5.1.4 Control (CR)

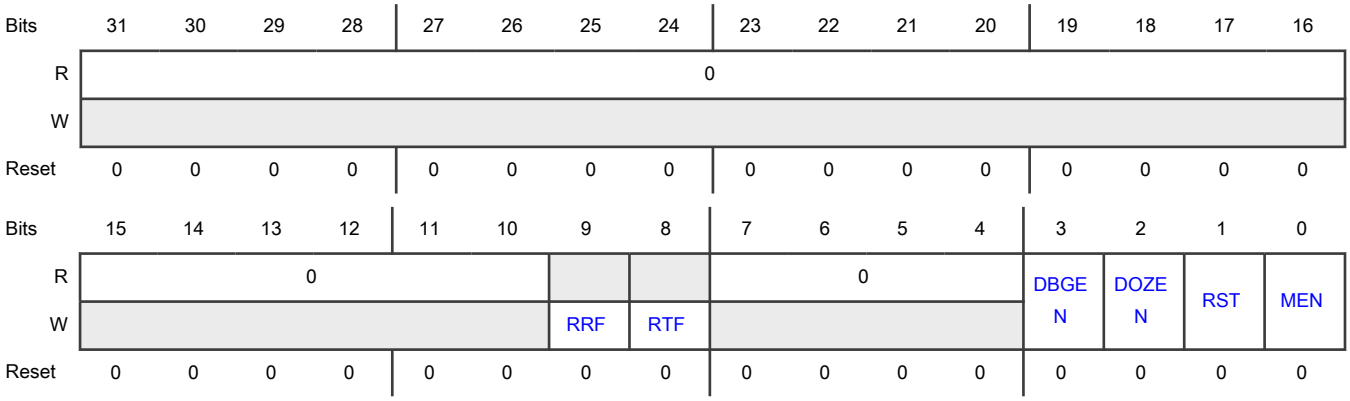
Offset

Register	Offset
CR	10h

Function

Contains fields that control module operation.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Deletes all entries in the receive FIFO. This field always reads 0. 0b - No effect 1b - Reset
8 RTF	Reset Transmit FIFO Deletes all entries in the transmit FIFO. This field always reads 0. 0b - No effect 1b - Reset
7-4 —	Reserved
3 DBGEN	Debug Enable Enables LPSPI in when the CPU is in Debug mode. When this field is 0, LPSPI is disabled when the CPU is halted; the PCS pin is deasserted after the transmit FIFO is empty regardless of the state of Transmit Command (TCR). Update this field only when the LPSPI module is disabled (MEN = 0). 0b - Disable 1b - Enable
2 DOZEN	Doze Mode Enable Enables LPSPI when the chip is in Doze mode. Update this field only when the LPSPI module is disabled (MEN = 0).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable 1b - Disable
1 RST	Software Reset Resets all internal logic and registers, except Control (CR). The reset takes effect immediately and remains asserted until you write 0 to it. There is no minimum delay required before clearing the software reset by writing 0. 0b - Not reset 1b - Reset
0 MEN	Module Enable After writing 0, MEN remains set until the LPSPI has completed the current transfer and is idle. 0b - Disable 1b - Enable

63.5.1.5 Status (SR)

Offset

Register	Offset
SR	14h

Function
Contains data flow status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							MBF	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		DMF	REF	TEF	TCF	FCF	WCF	0						RDF	TDF
W			W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-25 —	Reserved
24 MBF	Module Busy Flag In Master mode, indicates that there is data to transmit and LPSPI is able to transmit (for example, HREQ is asserted). It deasserts after the PCS pin deasserts and the LPSPI master has waited half the CCR[DBT] time with no new data to transmit. Slave mode asserts this flag when LPSPI is enabled and PCS is asserted. 0b - LPSPI is idle 1b - LPSPI is busy
23-14 —	Reserved
13 DMF	Data Match Flag Indicates that received data matches DMR0[MATCH0] and/or DMR1[MATCH1] (as configured by CFGR1[MATCFG]). 0b - No match 1b - Match
12 REF	Receive Error Flag Indicates a receive FIFO overflow error. When this flag is set: 1. End the transfer. 2. Empty the Receive FIFO. 3. Clear this flag. 4. Restart the transfer from the beginning. 0b - No overflow 1b - Overflow
11 TEF	Transmit Error Flag Indicates a Transmit FIFO underrun error. When this flag is set: 1. End the transfer. 2. Clear this flag. 3. Restart the transfer from the beginning. 0b - No underrun 1b - Underrun
10	Transfer Complete Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
TCF	In Master mode, indicates that all transfers have completed and that LPSPI has returned to Idle state and the transmit FIFO is empty. 0b - Not complete 1b - Complete
9 FCF	Frame Complete Flag Indicates that a frame transfer has completed, when the PCS deasserts. 0b - Not complete 1b - Complete
8 WCF	Word Complete Flag Indicates that the last bit of a received word is sampled. 0b - Not complete 1b - Complete
7-2 —	Reserved
1 RDF	Receive Data Flag Indicates that the number of words in the receive FIFO is greater than the value in FCR[RXWATER]. 0b - Receive data not ready 1b - Receive data is ready
0 TDF	Transmit Data Flag Indicates that the number of words in the transmit FIFO is equal to or less than the value in FCR[TXWATER]. 0b - Transmit data not requested 1b - Transmit data is requested

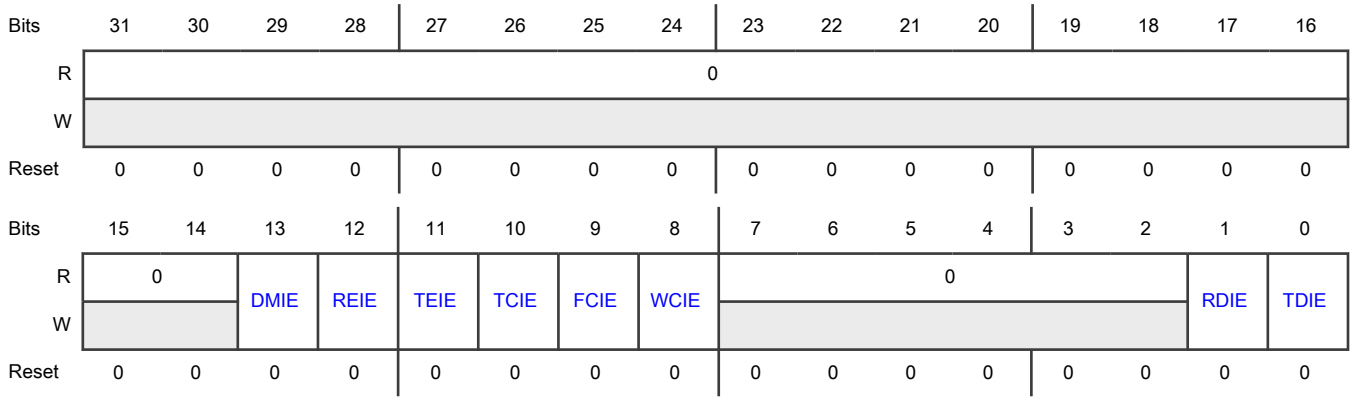
63.5.1.6 Interrupt Enable (IER)

Offset

Register	Offset
IER	18h

Function
Enables interrupts based on data flow and errors.

Diagram



Fields

Field	Function
31-14 —	Reserved
13 DMIE	Data Match Interrupt Enable 0b - Disable 1b - Enable
12 REIE	Receive Error Interrupt Enable 0b - Disable 1b - Enable
11 TEIE	Transmit Error Interrupt Enable 0b - Disable 1b - Enable
10 TCIE	Transfer Complete Interrupt Enable 0b - Disable 1b - Enable
9 FCIE	Frame Complete Interrupt Enable 0b - Disable 1b - Enable
8 WCIE	Word Complete Interrupt Enable 0b - Disable 1b - Enable
7-2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 RDIE	Receive Data Interrupt Enable 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable 0b - Disable 1b - Enable

63.5.1.7 DMA Enable (DER)

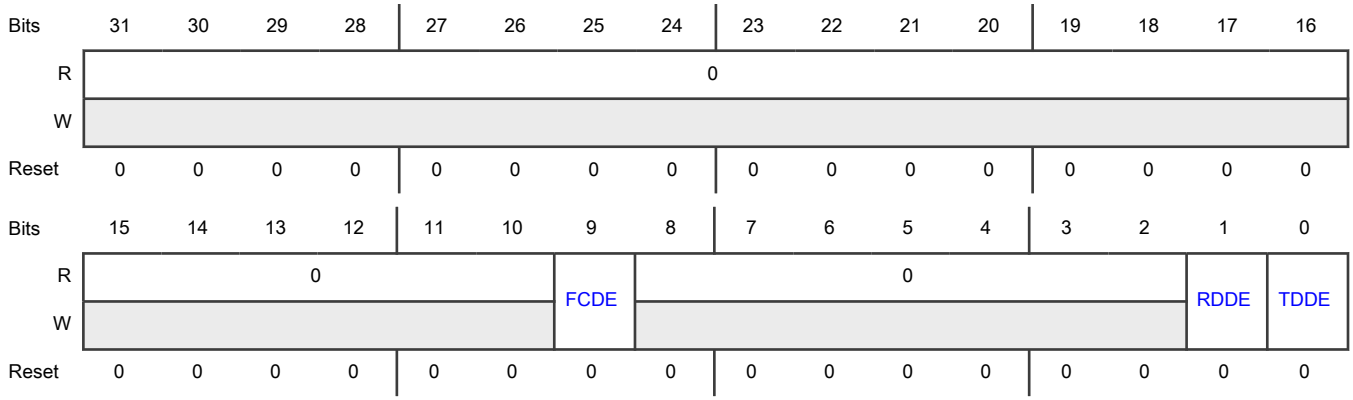
Offset

Register	Offset
DER	1Ch

Function

Enables DMA data flow.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 FCDE	Frame Complete DMA Enable Enables DMA end-of-packet processing. After the last word of a frame is read from the receive data FIFO, reading the receive data FIFO returns an end-of-packet signal with the receive data forced to

Table continues on the next page...

Table continued from the previous page...

Field	Function
	FFFF_FFFFh. This continues until the DMA minor loop completes, and then SR[FCF] deasserts if the receive FIFO is empty or if LPSPI is busy (SR[MBF] = 1). 0b - Disable 1b - Enable
8-2 —	Reserved
1 RDDE	Receive Data DMA Enable 0b - Disable 1b - Enable
0 TDDE	Transmit Data DMA Enable 0b - Disable 1b - Enable

63.5.1.8 Configuration 0 (CFGR0)

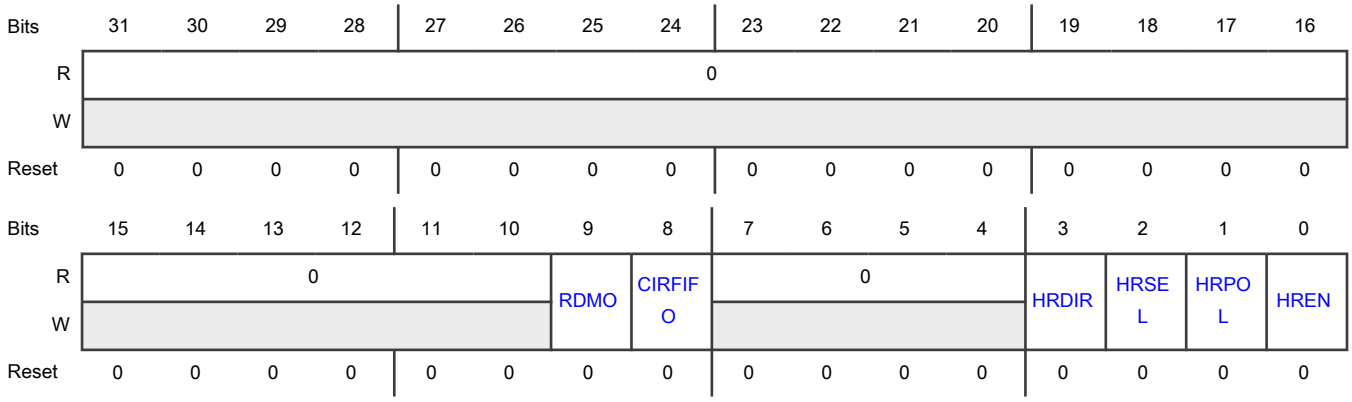
Offset

Register	Offset
CFGR0	20h

Function

Includes fields to configure LPSPI.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RDMO	Receive Data Match Only When enabled, all received data that does not cause the Data Match Flag (SR[DMF]) to assert is discarded. • Write 1 to this field when LPSPI is idle and SR[DMF] = 0. • After SR[DMF] = 1, this field is ignored. • To ensure that no receive data is lost when disabling RDMO, write 0 to this field before clearing SR[DMF]. See CFGR1[MATCFG] for the received data matching options. When disabled, all received data is stored in the receive FIFO. 0b - Disable 1b - Enable
8 CIRFIFO	Circular FIFO Enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO is emptied as in normal operation, but when LPSPI is idle and the transmit FIFO is empty, the read pointer value is restored from the temporary register. This restoring of the read pointer causes the contents of the transmit FIFO to be cycled through repeatedly. **NOTE** The read pointer is restored for as long as this field is 1. Writing additional words to the FIFO when this field is 1 adds them to the end of the FIFO, up to the size of the transmit FIFO. 0b - Disable 1b - Enable
7-4 —	Reserved
3 HRDIR	Host Request Direction Specifies the direction of the HREQ pin. Only configure the HREQ pin as an output when LPSPI is in Slave mode. The HREQ pin direction must be an input for Master mode. 0b - Input 1b - Output
2 HRSEL	Host Request Select Specifies the source of the host request input. When the host request function is enabled with the HREQ pin, the PCS[1] function is disabled. 0b - HREQ pin 1b - Input trigger

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 HRPOL	Host Request Polarity Specifies the polarity of the HREQ pin or input trigger. 0b - Active high 1b - Active low
0 HREN	Host Request Enable In Master mode, allows LPSPI to start a new SPI bus transfer only if the host request input is asserted. When LPSPI is busy, the host request input is ignored. In Slave mode, causes the HREQ output pin to assert when data is available to be transmitted. 0b - Disable 1b - Enable

63.5.1.9 Configuration 1 (CFGR1)

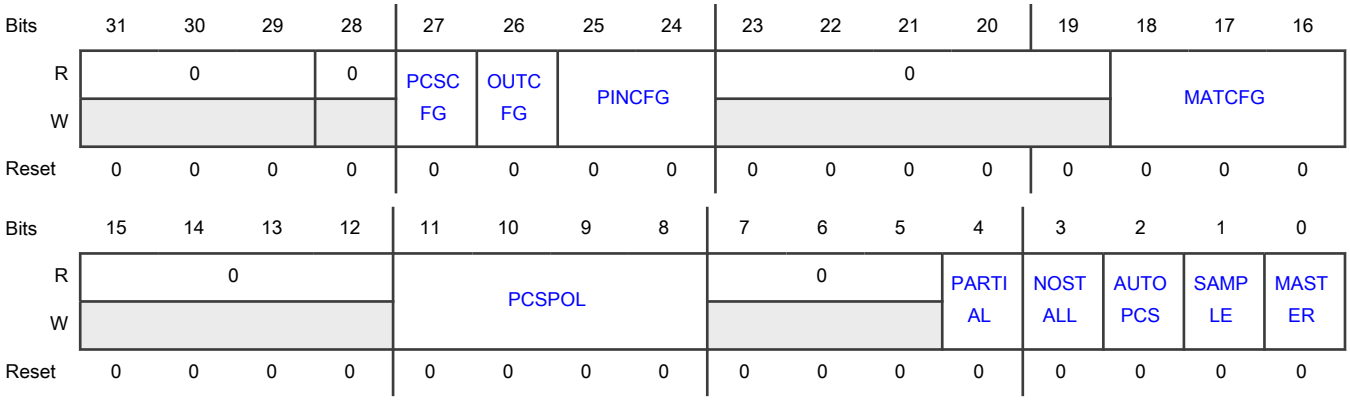
Offset

Register	Offset
CFGR1	24h

Function

Includes fields to configure LPSPI. Write to this register only when LPSPI is disabled.

Diagram



Fields

Field	Function
31-29	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
28 —	Reserved
27 PCSCFG	Peripheral Chip Select Configuration Specifies the PCS pin configuration. When performing parallel transfers, this field must be configured to enable the desired transfer. 0b - PCS[3:2] are configured for chip select function 1b - PCS[3:2] are configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])
26 OUTCFG	Output Configuration Specifies whether the output data is 3-stated between accesses (when PCS is deasserted). When performing half-duplex transfers, this field must be 1. 0b - Output data retains last value. 1b - Output data is 3-stated.
25-24 PINCFG	Pin Configuration Specifies the pins used for input and output data during serial transfers. This field is ignored when performing parallel transfers. 00b - SIN is used for input data; SOUT is used for output data. 01b - SIN is used for both input and output data. Only half-duplex serial transfers are supported. 10b - SOUT is used for both input and output data. Only half-duplex serial transfers are supported. 11b - SOUT is used for input data; SIN is used for output data.
23-19 —	Reserved
18-16 MATCFG	Match Configuration Specifies the condition that causes SR[DMF] to assert. **NOTE** When writing CFGR1[MATCFG], either the old value or new value should be the disabled state (0). You cannot go from a non-zero value to another non-zero value.

Condition	Description
Match first data word with compare word	Match if first data word equals MATCH0 logically ORed with MATCH1 *first_data_word* == (MATCH0 || MATCH1)

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Condition</th><th>Description</th></tr>
<tr><td>Match any data word with compare word</td><td>Match if any data word equals MATCH0 logically ORed with MATCH1
any_data_word == (MATCH0 || MATCH1)</td></tr>
<tr><td>Sequential match, first data word</td><td>Match if first data word equals MATCH0, and second data word equals MATCH1
(first_data_word == MATCH0) && (second_data_word == MATCH1)</td></tr>
<tr><td>Sequential match, any data word</td><td>Match if any data word equals MATCH0, and the next data word equals MATCH1
(any_data_word == MATCH0) && (next_data_word == MATCH1)</td></tr>
<tr><td>Match first data word (masked) with compare word (masked)</td><td>Match if first data word logically ANDed with MATCH1 equals MATCH0 logically ANDed with MATCH1
(first_data_word && MATCH1) == (MATCH0 && MATCH1)</td></tr>
<tr><td>Match any data word (masked) with compare word (masked)</td><td>Match if any data word logically ANDed with MATCH1 equals MATCH0 logically ANDed with MATCH1
(any_data_word && MATCH1) == (MATCH0 && MATCH1)</td></tr>
</table>
000b - Match is disabled
001b - Reserved
010b - Match first data word with compare word
011b - Match any data word with compare word
100b - Sequential match, first data word
101b - Sequential match, any data word
110b - Match first data word (masked) with compare word (masked)
111b - Match any data word (masked) with compare word (masked)
</td></tr>
<tr><td>15-12
—</td><td>Reserved</td></tr>
<tr><td>11-8
PCSPOL</td><td>Peripheral Chip Select Polarity
Specifies the polarity of each PCS pin. Bit n in this field (the least-significant bit is bit 0) corresponds to PCS[n].
For each PCSPOL bit:
0b – PCS[n] pin is active low
1b – PCS[n] pin is active high</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The entire PCSPOL field is not fully supported in every LPSPI module instance. See the LPSPI chip-specific information.
7-5 —	Reserved
4 PARTIAL	Partial Enable Specifies whether LPSPI, when in Slave mode, stores a partial received word in the receive FIFO, or discards it, when PCS deasserts. See Partial received word. 0b - Discard 1b - Store
3 NOSTALL	No Stall Disables a normal operating feature that causes LPSPI, when in Master mode, to stall transfers when the transmit FIFO is empty or when the receive FIFO is full. This feature prevents transmit FIFO underruns and receive FIFO overruns. Writing 1 to this field disables this functionality. 0b - Disable 1b - Enable
2 AUTOPCS	Automatic PCS Enables automatic PCS generation. For correct operation in Slave mode, LPSPI requires the PCS signal to deassert between frames. Writing 1 to this field generates an internal PCS signal at the end of each transfer word when TCR[CPHA] = 1. When this field is 1, SCK must remain idle for at least 4 LPSPI functional clock cycles, divided by the prescaler selected (see TCR[PRESCALE]) between each word, to ensure correct operation. This field is ignored in Master mode. 0b - Disable 1b - Enable
1 SAMPLE	Sample Point Specifies the SCK clock edge on which LPSPI, when in Master mode, samples input data. Writing 1 to this field causes LPSPI to sample input data on a delayed loopback SCK clock edge, which improves the setup time when sampling data (see Clock loopback). In this configuration. The input data setup time in Master mode is equal to the input data setup time in Slave mode. In Slave mode, this field is ignored. **NOTE** When SAMPLE = 1 both the input buffer and output buffer must be enabled for the SCK pin. 0b - SCK edge 1b - Delayed SCK edge

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 MASTER	Master Mode Specifies the LPSPI operating mode—either Master or Slave. This field directly controls the direction of the SCK and PCS pins. 0b - Slave mode 1b - Master mode

63.5.1.10 Data Match 0 (DMR0)

Offset

Register	Offset
DMR0	30h

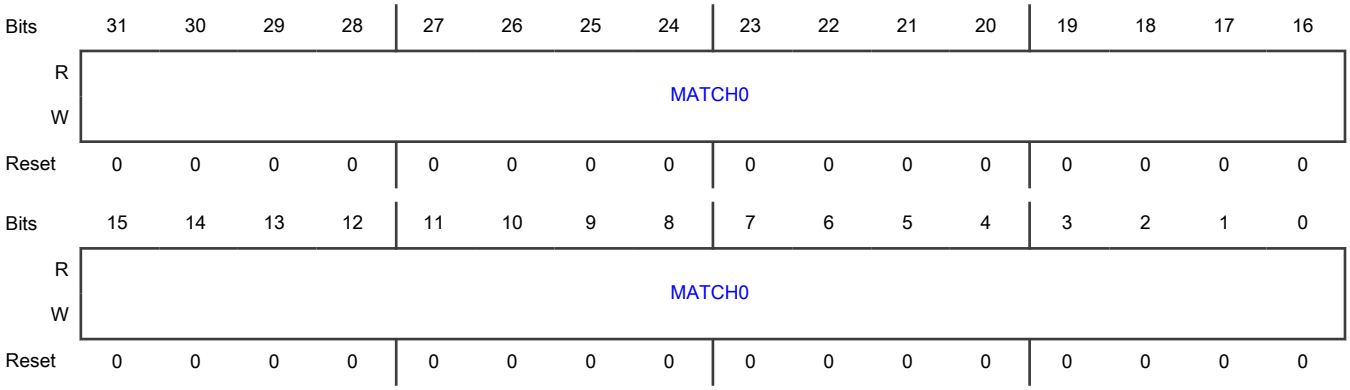
Function

Specifies match data to be used when data matching is enabled. See CFGR1[MATCFG] for the received data matching options.

NOTE

Do not change the value in this register while CFGR1[MATCFG] > 0.

Diagram



Fields

Field	Function
31-0 MATCH0	Match 0 Value MATCH0 value to be compared against received data.

63.5.1.11 Data Match 1 (DMR1)

Offset

Register	Offset
DMR1	34h

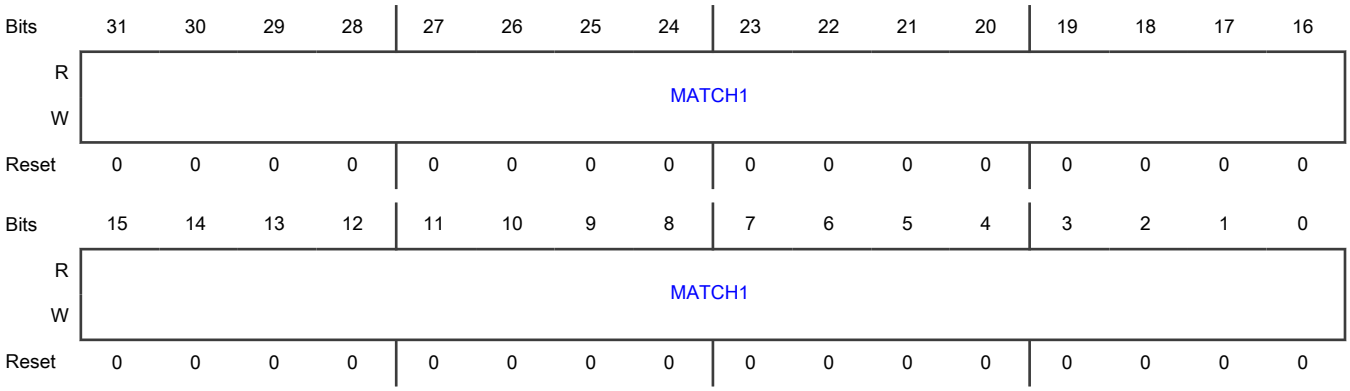
Function

Specifies match data to be used when data matching is enabled. See CFGR1[MATCFG] for the received data matching options.

NOTE

Do not change the value in this register while CFGR1[MATCFG] > 0.

Diagram



Fields

Field	Function
31-0 MATCH1	Match 1 Value MATCH1 value to be compared against received data.

63.5.1.12 Clock Configuration (CCR)

Offset

Register	Offset
CCR	40h

Function

Contains clock configuration fields. These fields are only used in Master mode and you can only change them when LPSPI is disabled (CR[MEN] = 0).

Warning

Writing a 32-bit value to this register overwrites Clock Configuration 1 (CCR1); DBT and SCKDIV always read 0.

To avoid overwriting CCR1:

- Either write all 4 CCR register fields simultaneously and only once in a 32-bit data.
- Or to modify SCKPCS and/or PCSSCK values, write only these 2 upper bytes in a 16-bit data or one of them in an 8-bit data.
- Or to modify CCR1[PCSPCS] and CCR1[SCKSCK] fields only or CCR1[SCKSET] and CCR1[SCKHLD] fields only, write respectively CCR[DBT] or CCR[SCKDIV] in 8-bit data

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R / W	SCKPCS								PCSSCK							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0							
W	DBT								SCKDIV							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SCKPCS	SCK-to-PCS Delay In Master mode: configures the delay from the last SCK edge to the PCS negation. • The delay is equal to (SCKPCS + 1) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 1 cycle. See Figure 468.
23-16 PCSSCK	PCS-to-SCK Delay In Master mode: configures the delay from the PCS assertion to the first SCK edge. • The delay is equal to (PCSSCK + 1) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 1 cycle. See Figure 468.
15-8 DBT	Delay Between Transfers Writing this field updates the contents of CCR1[PCSPCS] and CCR1[SCKSCK].
7-0	SCK Divider

Table continues on the next page...

Table continued from the previous page...

Field	Function
SCKDIV	Writing this field updates the contents of CCR1[SCKSET] and CCR1[SCKHLD]. Baud rate = Function clock / (2^PRESCALE * (SCKSET+SCKHLD+2))

63.5.1.13 Clock Configuration 1 (CCR1)

Offset

Register	Offset
CCR1	44h

Function

Contains clock configuration fields. These fields are only used in Master mode and you can only change them when LPSPI is disabled (CR[MEN] = 0).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SCKSCK								PCSPCS							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SCKHLD								SCKSET							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SCKSCK	SCK Inter-Frame Delay In Master mode: • Configures the delay from the last SCK pulse of a frame and the first SCK pulse of the following frame, in a continuous transfer. • The delay is equal to (SCKSCK + 1) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 1 cycle.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE For backward compatibility, writing CCR[DBT] updates CCR1[SCKSCK] with the value written.
23-16 PCSPCS	PCS to PCS delay In Master mode: • Configures the delay from the PCS negation to the next PCS assertion. • The delay is equal to (PCSPCS + PCSPCS + 2) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 2 cycles. • Half of the delay (PCSPCS + 1) occurs before PCS assertion and the other half of the delay (PCSPCS + 1) occurs after PCS negation. If the command word is updated between 2 transfers, then the command word is updated halfway between the PCS negation of the last transfer and PCS assertion of the next transfer. • The command word specifies which PCS signal is used, the polarity and phase of the SCK signal, and the prescaler selected. **NOTE** For backward compatibility, writing CCR[DBT] updates this field with (DBT÷2) rounded up.
15-8 SCKHLD	SCK Hold In Master mode, this field configures the hold phase of the SCK pin. • The hold phase is the delay between the SCK edge that samples the receive data and the SCK edge that drives the transmit data. • This is the SCK low period when CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. This is the SCK high period when CPHA = 0, CPOL = 0 and CPHA = 1, CPOL = 1. • The SCK hold phase delay is equal to (SCKHLD + 1) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 1 cycle. • The SCK period is equal to (SCKSET + SCKHLD + 2) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The SCK duty cycle is based on the difference between SCKSET and SCKHLD. Configure both fields to the same value for 50/50 duty cycle. See Figure 468. **NOTE** For backward compatibility, writing CCR[SCKDIV] updates this field with (SCKDIV ÷ 2) rounded down.
7-0 SCKSET	SCK Setup In Master mode, the SCK Setup configures the setup phase of the SCK pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• The setup phase is the delay between the SCK edge that drives the transmit data and the SCK edge that samples the receive data. • This is the SCK high period when CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. This is the SCK low period when CPHA = 0 and CPOL = 0, or CPHA = 1 and CPOL = 1. • The SCK setup phase delay is equal to (SCKSET + 1) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 1 cycle. • The SCK period is equal to (SCKSET + SCKHLD + 2) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The SCK duty cycle is based on the difference between SCKSET and SCKHLD, configure both fields to the same value for 50/50 duty cycle. See Figure 468. **NOTE** For backward compatibility, writing CCR[SCKDIV] updates this field with (SCKDIV ÷ 2) rounded up.

63.5.1.14 FIFO Control (FCR)

Offset

Register	Offset
FCR	58h

Function

Contains the receive FIFO and transmit FIFO watermark values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0				RXWATER			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0				TXWATER			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-20 —	Reserved
19-16 RXWATER	Receive FIFO Watermark Causes LPSPI to set the Receive Data Flag (SR[RDF]) when the number of words in the receive FIFO is greater than RXWATER. Writing a value equal or greater than the FIFO size truncates the written value.
15-8 —	Reserved
7-4 —	Reserved
3-0 TXWATER	Transmit FIFO Watermark Causes LPSPI to set the Transmit Data Flag (SR[TDF]) when the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal or greater than the FIFO size truncates the written value.

63.5.1.15 FIFO Status (FSR)

Offset

Register	Offset
FSR	5Ch

Function

Contains fields that indicate the number of words currently stored in the receive FIFO and transmit FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0				RXCOUNT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0				TXCOUNT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-21 —	Reserved
20-16 RXCOUNT	Receive FIFO Count Indicates the number of words currently stored in the receive FIFO.
15-8 —	Reserved
7-5 —	Reserved
4-0 TXCOUNT	Transmit FIFO Count Indicates the number of words currently stored in the transmit FIFO.

63.5.1.16 Transmit Command (TCR)

Offset

Register	Offset
TCR	60h

Function

Writes to either this register or Transmit Data (TDR) push the data into the transmit FIFO, in the order written. Only write to this register using 32-bit writes. Writes are tagged and cause the command register to update, after that entry reaches the top of the FIFO and the LPSPI is enabled. This allows changes to the command word and the transmit data itself to be interleaved. That is, the writes to the two registers can be interleaved (write command word, then data word, then command word, and so on). Changing the command word causes all subsequent SPI bus transfers to be performed using the new command word.

- **In Master mode,** writing a new command word does not initiate a new transfer, unless TXMSK is 1. Transfers are initiated by transmit data in the transmit FIFO, or by a new command word (with TXMSK = 1). Hardware writes 0 to TXMSK when the PCS deasserts.
- **In Master mode,** if the command word is changed before an existing frame has completed, then the existing frame terminates and the command word then updates. The command word can be changed during a continuous transfer, if CONTC of the new command word is 1 and the command word is written on a frame size boundary.
- **In Slave mode,** the command word should be changed only when LPSPI is idle and there is no SPI bus transfer.

Avoid resetting the Transmit FIFO after writing to the Transmit Command Register, wait for the command register to update from the FIFO first.

Avoid register reading problems: Reading the Transmit Command Register returns the current state of the command register. Reading the Transmit Command Register at the same time that the Transmit Command Register is loaded from the transmit FIFO, can return an incorrect Transmit Command Register value. It is recommended to:

- Read the Transmit Command Register when the transmit FIFO is empty,
- Read the Transmit Command Register more than once and then compare the returned values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CPOL	CPHA	PRESCALE			Reserved	PCS		LSBF	BYSW	CONT	CONTC	RXMSK	TXMSK	WIDTH	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				FRAMESZ											
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

Fields

Field	Function
31 CPOL	Clock Polarity Specifies the value of SCK when it is idle. This field can only be updated when PCS deasserted. See Figure 468. 0b - Inactive low 1b - Inactive high
30 CPHA	Clock Phase This field can only be updated when PCS deasserted. See Figure 468. 0b - Captured. Data is captured on the leading edge of SCK and changed on the following edge of SCK 1b - Changed. Data is changed on the leading edge of SCK and captured on the following edge of SCK
29-27 PRESCALE	Prescaler Value For all SPI bus transfers, this value is applied to Clock Configuration (CCR) clock. This field can only be updated when PCS deasserted. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128
26 —	Reserved
25-24 PCS	Peripheral Chip Select Configures the peripheral chip select used for the transfer. The Peripheral Chip Select field is only updated when PCS deasserted. **NOTE** The entire PCS field is not fully supported in every LPSPI module instance. See the chip-specific LPSPI information. 00b - Transfer using PCS[0] 01b - Transfer using PCS[1] 10b - Transfer using PCS[2] 11b - Transfer using PCS[3]
23 LSBF	LSB First 0b - Data is transferred MSB first 1b - Data is transferred LSB first
22 BYSW	Byte Swap Swaps the contents of [31:24] with [7:0] and [23:16] with [15:8] for each transmit data word read from the FIFO and for each received data word stored to the FIFO (or compared with match registers). 0b - Disabled 1b - Enabled
21 CONT	Continuous Transfer • In Master mode, CONT keeps the PCS asserted at the end of the frame size, until a command word is received that starts a new frame. • In Slave mode, when CONT is enabled, LPSPI only transmits the first FRAMESZ bits; after which LPSPI transmits received data (assuming a 32-bit shift register) until the next PCS negation. 0b - Continuous transfer is disabled 1b - Continuous transfer is enabled
20	Continuing Command

Table continues on the next page...

Table continued from the previous page...

Field	Function
CONTC	In Master mode, this field enables the command word to be changed within a continuous transfer. • The initial command word must enable continuous transfer (CONT = 1). • The continuing command must have CONTC = 1. • The continuing command word must be loaded on a frame size boundary. For example, if the continuous transfer has a frame size of 64-bits, then a continuing command word must be loaded on a 64-bit boundary. In Slave mode, the Continuing Command bit modifies the internal RXMSK/TXMSK configuration after the first FRAMESZ bits and until the PCS negation. • Receive data is discarded after the first FRAMESZ bits. If CONT is also 1 this does not block the transmission of received data. • Transmit data is not masked after the first FRAMESZ bits, this allows the first FRAMESZ bits to be received and a response transmitted. 0b - Command word for start of new transfer 1b - Command word for continuing transfer
19 RXMSK	Receive Data Mask Masks receive data (receive data is not stored in receive FIFO). 0b - Normal transfer 1b - Receive data is masked
18 TXMSK	Transmit Data Mask Masks transmit data (no data is loaded from transmit FIFO and the output pin is 3-stated). In Master mode, TXMSK initiates a new transfer which cannot be aborted by another command word. TXMSK automatically transitions to 0 at the end of the transfer. 0b - Normal transfer 1b - Mask transmit data
17-16 WIDTH	Transfer Width Configures serial (1-bit) or parallel transfers. For half-duplex parallel transfers, either Receive Data Mask (RXMSK) or Transmit Data Mask (TXMSK) must be 1. 00b - 1-bit transfer 01b - 2-bit transfer 10b - 4-bit transfer 11b - Reserved
15-12 —	Reserved
11-0	Frame Size

Table continues on the next page...

Table continued from the previous page...

Field	Function
FRAMESZ	Configures the frame size in number of bits equal to (FRAMESZ + 1). • The minimum frame size is 8 bits • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contains the remainder bits. For example, a 72-bit transfer consists of 3 words: the first and second words are 32 bits, and the third word is 8 bits.

63.5.1.17 Transmit Data (TDR)

Offset

Register	Offset
TDR	64h

Function

Writes to either Transmit Command (TCR) or this register push the data into the transmit FIFO, in the order that the data is written. You can write to TDR using 32-, 16-, or 8-bit writes, but each write pushes data into the FIFO with zero pushed in unwritten bytes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	Transmit Data Both 8- and 16-bit writes of transmit data zero-extend the data written and push the data into the transmit FIFO. To zero-extend 8- and 16-bit writes (to 32 bits), means that the higher order (most significant) empty parts of the 8-bit and 16-bit writes are filled with zeroes.

63.5.1.18 Receive Status (RSR)

Offset

Register	Offset
RSR	70h

Function

Contains data flow status fields for receive FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														RXEMPTY	SOF
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-2 —	Reserved
1 RXEMPTY	RX FIFO Empty 0b - Not empty 1b - Empty
0 SOF	Start Of Frame Indicates that this is the first data word received after PCS assertion. 0b - Subsequent data word 1b - First data word

63.5.1.19 Receive Data (RDR)

Offset

Register	Offset
RDR	74h

Function
Reading this register pulls the first entry from the receive FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	Receive Data

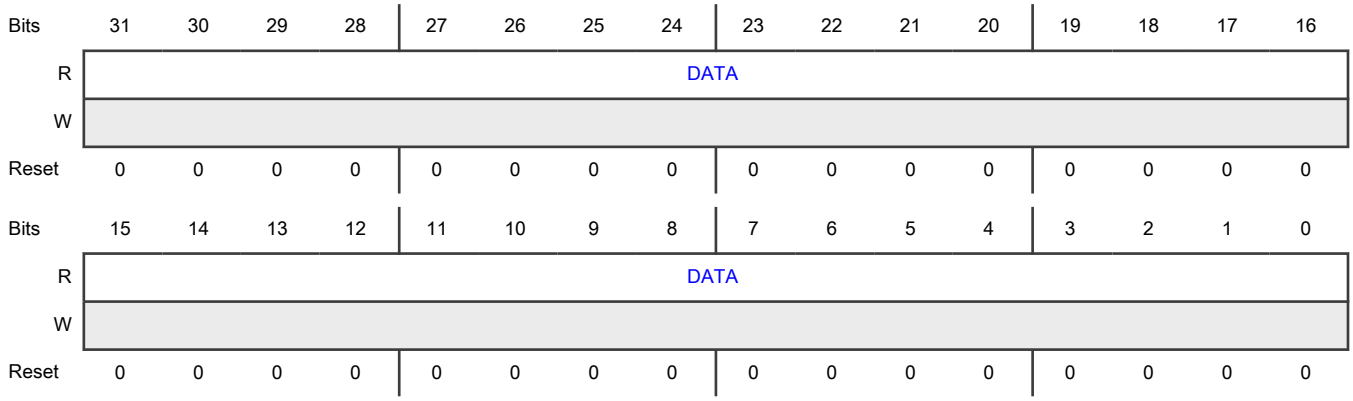
63.5.1.20 Receive Data Read Only (RDROR)

Offset

Register	Offset
RDROR	78h

Function
Returns the first entry in the receive FIFO, but does not remove the data from the FIFO.

Diagram



Fields

Field	Function
31-0 DATA	Receive Data

63.5.1.21 Transmit Command Burst (TCBR)

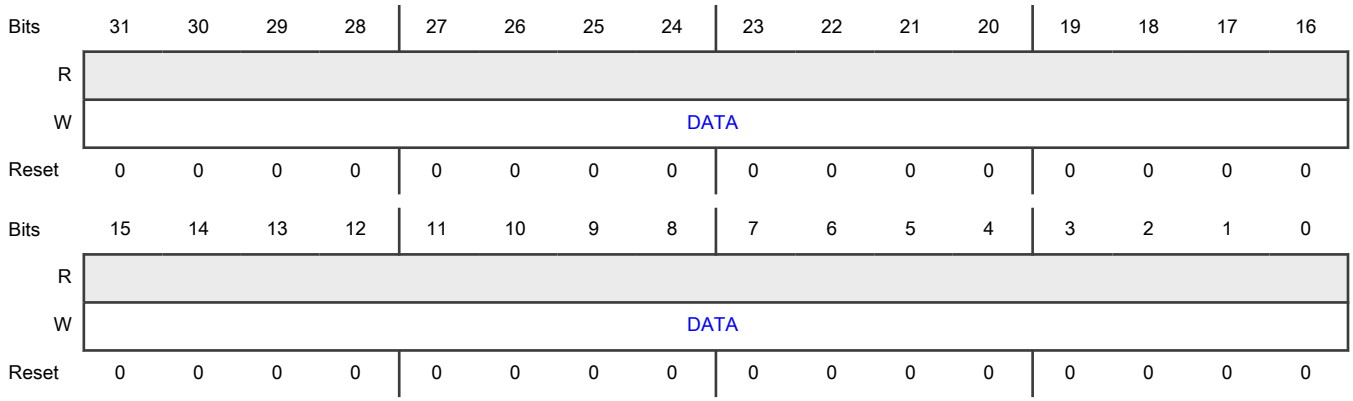
Offset

Register	Offset
TCBR	3FCh

Function

Supports burst transfers of command data to the transmit FIFO for use with the DMA controller.

Diagram



Fields

Field	Function
31-0 DATA	Command Data Data is written to the Transmit Command Register.

63.5.1.22 Transmit Data Burst (TDBR0 - TDBR127)

Offset

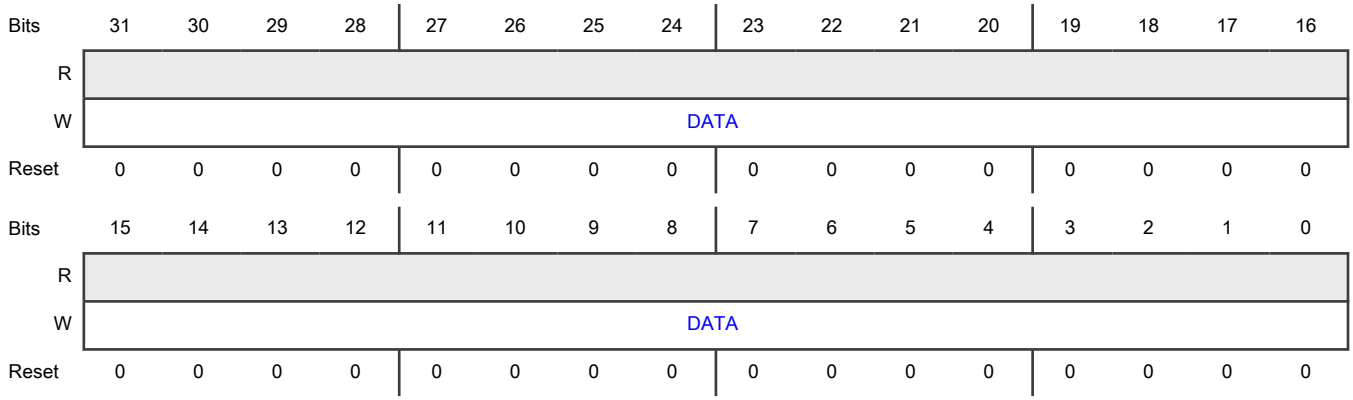
For n = 0 to 127:

Register	Offset
TDBRn	400h + (n × 4h)

Function

Supports burst transfers of data to the transmit FIFO for use with the DMA controller. The size of this register is 512 bytes.

Diagram



Fields

Field	Function
31-0 DATA	Data Data is written to Transmit Data (TDR).

63.5.1.23 Receive Data Burst (RDBR0 - RDBR127)

Offset

For n = 0 to 127:

Register	Offset
RDBRn	600h + (n × 4h)

Function

Supports burst transfers of data from the receive FIFO. The size of this register is 512 bytes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	Data Data is read from the Receive Data Register.

Chapter 64
Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)

64.1 Chip-specific LPUART Information

Table 1036. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

NOTE

The output triggers are not connected.

NOTE

See clock and Power chapter for details about low-power modes available in this chip.

64.2 Overview

LPUART provides asynchronous, serial communication capabilities with external devices. It supports the non-return-to-zero (NRZ) encoding format and infrared data association (IrDA)-compatible, low-speed serial infrared (SIR) protocol. LPUART can continue operating when the processor is in Low-Power mode, if an appropriate peripheral clock is available.

64.2.1 Block diagram

Figure 470 shows the transmitter portion of LPUART.

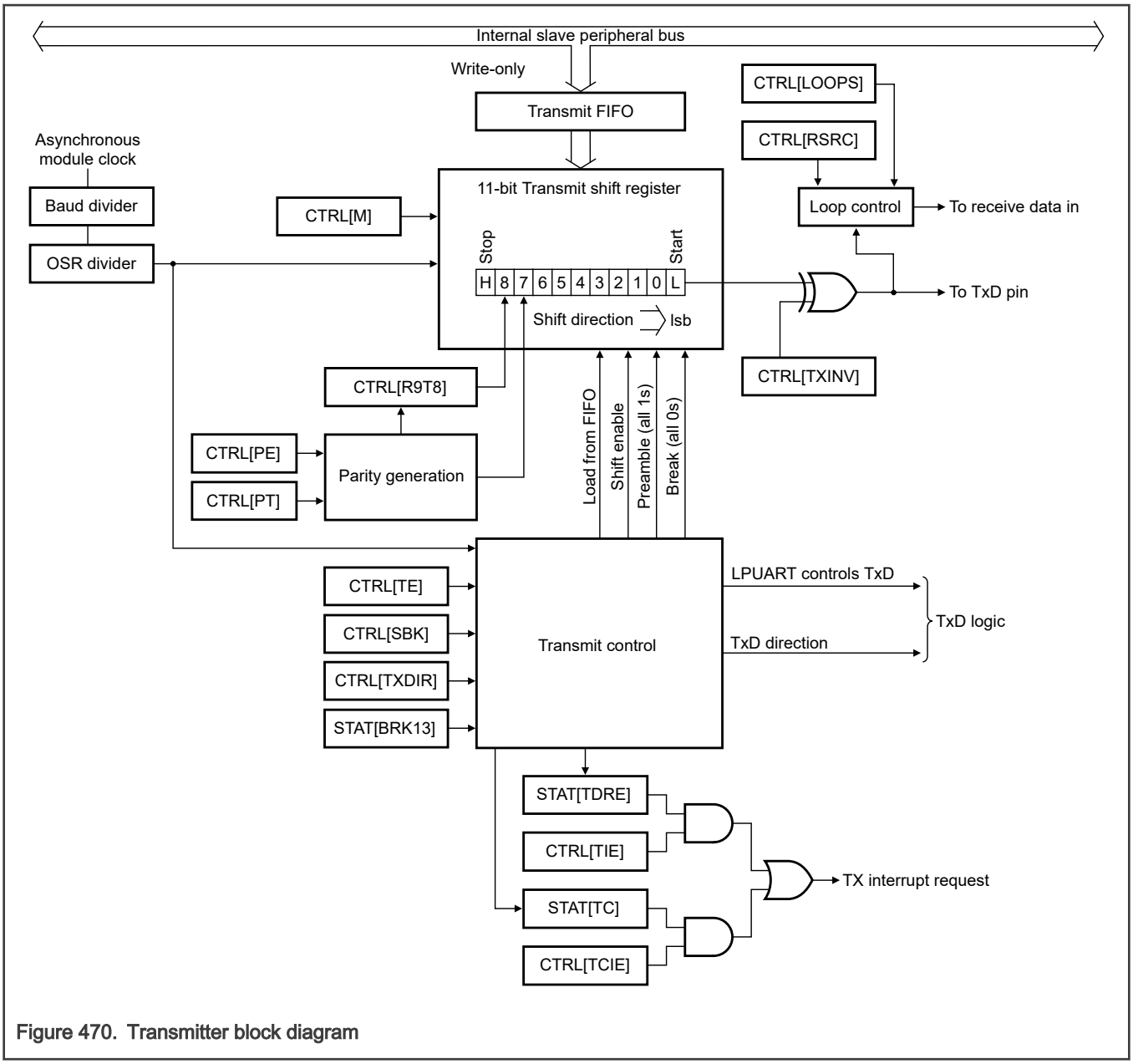


Figure 470. Transmitter block diagram

Figure 471 shows the receiver portion of LPUART.

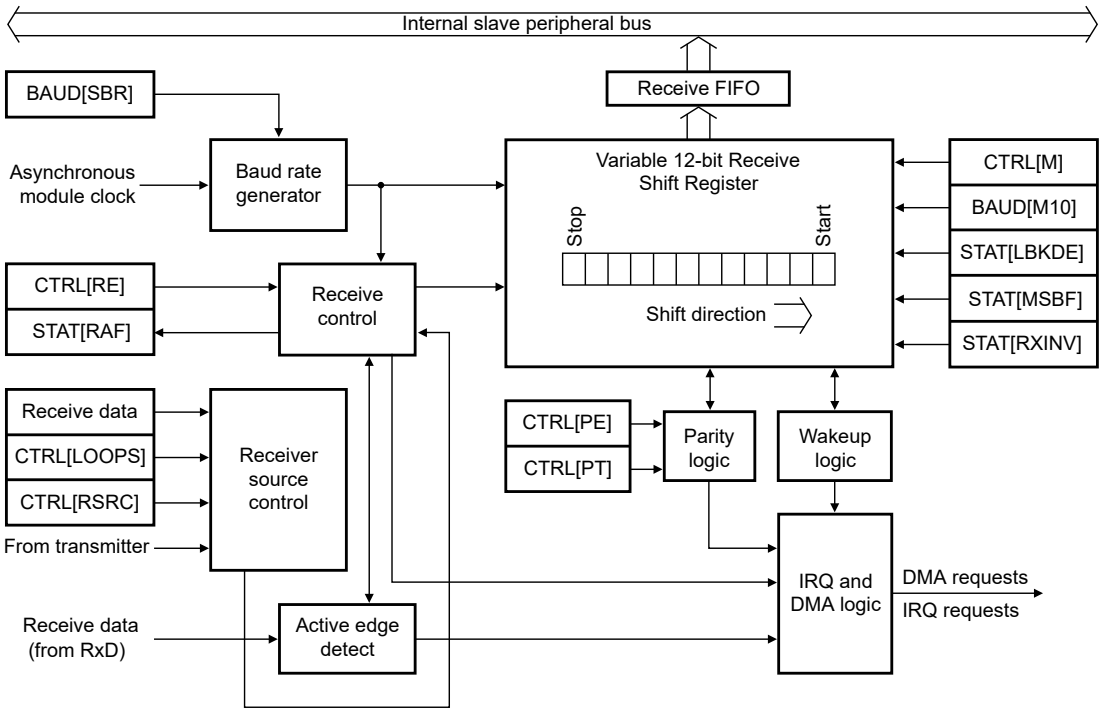


Figure 471. Receiver block diagram

64.2.2 Features

- Full-duplex, standard NRZ format
- Programmable baud rates (13-bit modulo divider) with a configurable oversampling ratio (OSR) from 4× to 32×
- Asynchronous operation of transmit and receive baud rates with respect to the bus clock:
  - Baud rate can be configured independently of the bus clock frequency.
  - Operation in Low-Power modes is supported.
- Interrupt, DMA, or polled operations:
  - Transmit data empty and transmission complete
  - Receive data full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
  - Receive data match
- Hardware parity generation and checking
- Programmable 7-bit, 8-bit, 9-bit, or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Support for three receiver wake-up methods:
  - Idle line wake-up

  - Address mark wake-up
  - Receive data match
- Automatic address matching to reduce ISR overhead:
  - Address mark matching
  - Idle line address matching
  - Address match start, address match end
- Optional 13-bit and 11-bit break character generation
- Configurable idle length detection supporting 1, 2, 4, 8, 16, 32, 64, or 128 idle characters
- Selectable transmitter output and receiver input polarity
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with a programmable pulse width
- Independent FIFO structure for transmit and receive functions:
  - Separate configurable watermarks for receive and transmit requests
  - Option for receiver to assert request after a configurable number of idle characters, if receive FIFO is not empty

64.3 Functional description

LPUART supports full-duplex, asynchronous, NRZ serial communication and comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. The following sections describe all LPUART blocks.

64.3.1 Baud rate generation

A 13-bit modulus counter in the baud rate generator derives the baud rate for both the receiver and transmitter. The value, ranging from 1 to 8191, written to BAUD[SBR] determines the baud clock divisor for the asynchronous LPUART baud clock. The baud rate clock drives the receiver, while a bit clock, generated from the baud rate clock divided by the OSR, drives the transmitter. Depending on the OSR, the receiver has an acquisition rate of 4 to 32 samples per bit time.

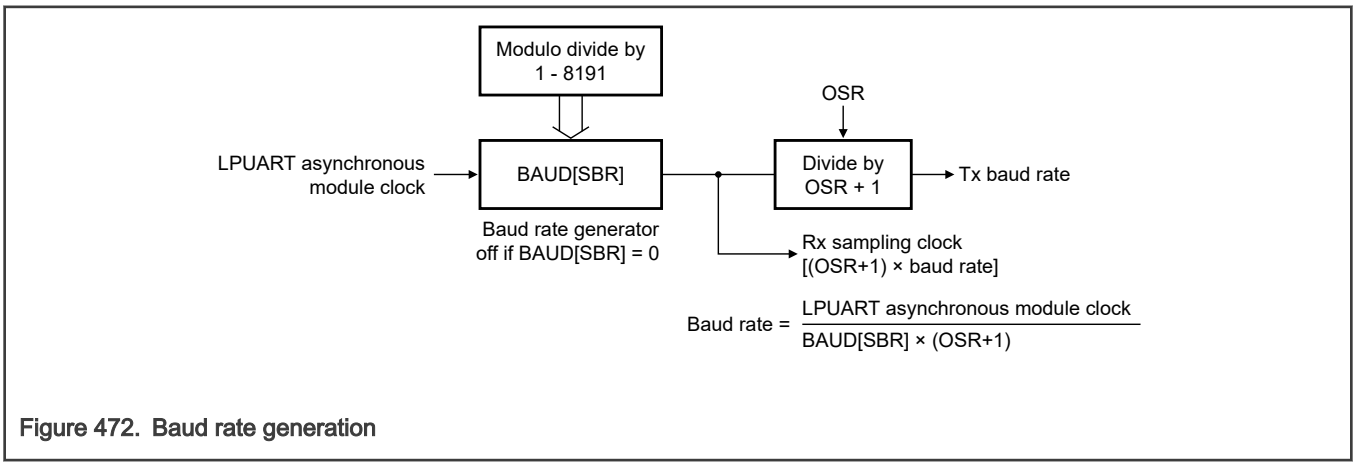


Figure 472. Baud rate generation

Baud rate generation is subject to these sources of error:

- Integer division of the asynchronous LPUART baud clock may not give the exact target frequency.
- Synchronization with the asynchronous LPUART baud clock can lead to a phase shift.

Baud rate generation is a free-running counter that continues whenever the transmitter or receiver is enabled. The transmitter bit clock continues whenever the transmitter is enabled; each transmitted character aligns to the next edge of the transmit bit clock.

In general, configuring OSR for a higher ratio and/or sampling on both edges of the clock slightly improves LPUART's tolerance to baud rate mismatch between the received data and LPUART configured baud rate. However, the three data samples in each bit (see Data sampling technique) are also closer together, which may impact noise sensitivity.

64.3.2 Baud rate tolerance

A transmitting device may operate at a baud rate below or above that of the receiver.

Accumulated bit time misalignment can cause one of the three stop bit data samples to fall outside the actual stop bit. A noise error will occur if the three samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the three stop bit samples are a logic zero.

As the receiver samples an incoming frame, it may re-synchronize the oversampling clock on any valid falling edge within the frame. Resynchronization within frames will correct a misalignment between transmitter bit times and receiver bit times.

In general, increasing the number of samples per bit will increase the baud rate tolerance and decreasing the number of samples per bit will reduce the baud rate tolerance. Note that since LPUART implements triple voting on consecutive receive data samples, increasing the number of samples per bit will move those samples closer together which would reduce the width of noise that can be filtered by the triple voting logic.

64.3.3 Calculating baud rate tolerance

Using the following definitions:

- SAM is the number of sample points per bit (valid range from 8 to 32; equal to (OSR + 1) × (BOTHEDGE + 1)).
- BIT is the number of bits in a character including start, data and stop bits (valid range from 9 to 13).

The ideal baud rate tolerance can be calculated as follows:

- Slow data rate tolerance = ((SAM ÷ 2) - 1) ÷ ((SAM × BIT) - (SAM ÷ 2) + 2)
- Fast data rate tolerance = ((SAM ÷ 2) - 2) ÷ (SAM × BIT)

As an example, if configured for 8-bit data, 1 stop bit (BIT = 10) and with OSR=0x7 and BOTHEDGE = 1 (SAM = 16):

- Slow data rate tolerance = (8 - 1) ÷ (160 - 8 + 2) = 7 ÷ 154 = 4.54%
- Fast data rate tolerance = (8 - 2) ÷ 160 = 6 ÷ 160 = 3.75%

If configured for 9-bit data with 1 stop bit (BIT = 11) with same oversampling configuration, then:

- Slow data rate tolerance = (8 - 1) ÷ (176 - 8 + 2) = 7 ÷ 170 = 4.12%
- Fast data rate tolerance = (8 - 2) ÷ 176 = 6 ÷ 176 = 3.41%

NOTE

Additional factors can contribute to a lower baud rate tolerance than the ideal. These include clock uncertainty or jitter on the LPUART functional clock source, differences in rise and fall times on the transmitter output and synchronization of the external receive pin to the local LPUART functional clock.

64.3.4 Transmitter functional description

This section describes the functioning of the LPUART transmitter, as shown in the transmitter portion of Block diagram, as well as specialized functions for sending break and idle characters.

The transmitter output (TXD) idle state defaults to logic high; the transmitter output is inverted when you write 1 to CTRL[TXINV], which becomes 0 following reset. You can enable the transmitter by writing 1 to CTRL[TE]. This queues a preamble character that is one full character frame of the Idle state. The transmitter then remains idle until data is available in the transmit FIFO and programs store data in the transmit FIFO by writing to Data (DATA).

The central element of the LPUART transmitter is the transmit shift register that is 9-bit to 13-bit long depending on the settings of CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS]. Going forward in this discussion, assume that CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS] are 0, selecting the normal 8-bit Data mode, in which the shift register holds a start bit, eight data

bits, and a stop bit. When the transmit shift register is available for a new character, the value waiting in transmit FIFO is transferred to the transmit shift register, synchronized with the baud rate clock, and STAT[TDRE] becomes 1 to indicate that another character may be written to the transmit FIFO at Data (DATA).

If no new character is waiting in the transmit FIFO after a stop bit is shifted out of the TXD pin, the transmitter sets the transmit complete flag and enters an idle mode, with TXD high, waiting for more characters to transmit.

Writing 0 to CTRL[TE] does not immediately disable the transmitter. The current transmit activity in progress must first be completed (that could include a data character, idle character, or break character), although the transmitter does not start transmitting another character.

64.3.4.1 Break character length

CTRL[SBK] sends break characters, originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 9-bit to 12-bit times, including the start and stop bits. You can enable a longer break of 13-bit times by writing 1 to STAT[BRK13]. Normally, a program waits for STAT[TDRE] to become 1 to indicate that the last character of a message has moved to the transmit shifter. Next, the program writes 1 and then writes 0 to CTRL[SBK]. This action queues a break character to be sent as soon as the shifter is available. If CTRL[SBK] remains 1 when the queued break moves into the shifter, synchronized with the baud rate clock, an additional break character is queued. When LPUART is the receiving module, it receives a break character as 0s in all data bits and a framing error (STAT[FE] = 1) is detected.

You can also transmit a break character by writing to Data (DATA) with DATA[FRETSC] = 1 and the data bits clear. This supports transmitting the break character as part of the normal data stream and also allows DMA to transmit a break character.

When idle line wake-up is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program waits for STAT[TDRE] to become 1 to indicate that the last character of a message has moved to the transmit shifter. Next, write 0 and then write 1 to CTRL[TE]. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while CTRL[TE] becomes 0, the LPUART transmitter does not release control of the TXD pin.

You can also write to Data (DATA) to transmit an idle character, with DATA[FRETSC] and DATA[R9T9] = 1 and the values of all the other fields = 0. This supports transmitting the idle character as part of the normal data stream and also allows DMA to transmit an idle character.

As shown in the following table, STAT[BRK13], CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS] affect the length of the break character.

Table 1037. Break character length

STAT[BRK13]	CTRL[M]	BAUD[M10]	CTRL[M7]	BAUD[SBNS]	Break character length (in bit times)
0	0	0	0	0	10
0	0	0	0	1	11
0	0	0	1	0	9
0	0	0	1	1	10
0	1	0	—	0	11
0	1	0	—	1	12
0	—	1	—	0	12
0	—	1	—	1	13
1	0	0	0	0	13
1	0	0	0	1	13
1	0	0	1	0	12

Table continues on the next page...

Table 1037. Break character length (continued)

STAT[BRK13]	CTRL[M]	BAUD[M10]	CTRL[M7]	BAUD[SBNS]	Break character length (in bit times)
1	0	0	1	1	12
1	1	0	—	0	14
1	1	0	—	1	14
1	—	1	—	0	15
1	—	1	—	1	15

64.3.4.2 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS_B. If the CTS operation is enabled, the character is transmitted when CTS_B is asserted. If CTS_B is deasserted in the middle of a transmission with characters remaining in the transmitter FIFO, the character in the transmit shift register is complete. Any characters in the FIFO wait for CTS_B to assert again, and TXD remains in the mark state (idle state) until CTS_B is reasserted. The CTS_B pin must assert for longer than one bit period to guarantee that a new transmission is started when the transmitter is idle with CTS.

If the CTS operation is disabled, the transmitter ignores the state of CTS_B.

The transmitter's CTS_B signal can be enabled even if the same LPUART receiver's RTS_B signal is disabled.

64.3.4.3 Transceiver driver enable

The transmitter can use RTS_B as an enable signal for the driver of an external transceiver. See Transceiver driver enable using RTS_B for details. If the RTS operation is enabled, when a character is placed into an empty transmit shift register, RTS_B asserts 1-bit time before the start bit is transmitted. RTS_B remains asserted for the whole time that the transmit shift register has any characters. RTS_B deasserts 1-bit time after all characters in the transmit FIFO and shift register are completely sent, including the last stop bit. In other words, when RTS_B is used as a transceiver enable, RTS_B asserts 1-bit time before the transmitter starts transmitting and negates 1-bit time after the transmitter goes idle.

Transmitting a break character also asserts RTS_B, with the same assertion and deassertion timing as having a character in the transmit shift register.

The transmitter's RTS_B signal asserts only when the transmitter is enabled. However, the transmitter's RTS_B signal is unaffected by its CTS_B signal. RTS_B remains asserted until the transfer is complete, even if the transmitter is disabled mid-way through a data transfer.

You can configure HDCR[RTSEXT] to the desired length by delaying the transmitter's RTS_B negation by up to 256-bit clock (baud rate) after the last stop bit.

64.3.4.4 Transceiver driver enable using RTS_B

RS-485 is a multiple drop communication protocol in which the LPUART transceiver's driver is three-stated unless LPUART is driving. The transmitter can use the RTS_B signal to enable the driver of a transceiver. The polarity of RTS_B can be matched to the polarity of the transceiver's driver enable signal.

The following figure shows the receiver enable signal asserted. This connection can also connect RTS_B to both DE and RE_B. The transceiver's receiver is disabled when driving. A pullup can pull RXD to a nonfloating value during this time. You can refine this option further by operating LPUART in Single-Wire mode, freeing the RXD pin for other uses.

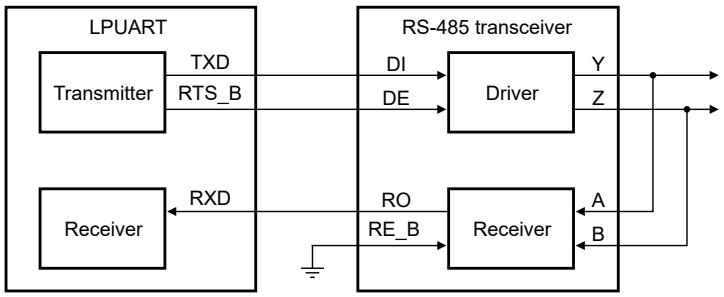


Figure 473. Transceiver driver enable using RTS_B

64.3.5 Receiver functional description

This section discusses the functioning of the LPUART receiver, as shown in the receiver portion of Block diagram. The section also discusses:

- The data sampling technique used to reconstruct receiver data.
- Different variations of the receiver wake-up function.

You can invert the receiver input by writing 1 to STAT[RXINV] and enable the receiver by writing 1 to CTRL[RE]. Character frames consist of a start bit of logic 0, along with N (7, 8, 9, 10) bits (MSB or LSB first), and one or two stop bits of logic 1. For information about 7-bit, 9-bit, or 10-bit Data mode, see Data modes. Going forward in this discussion, assume that LPUART is configured for a normal 8-bit Data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full (STAT[RDRF] = 0), the data character is transferred to the receive FIFO, resulting in STAT[RDRF] becoming 1. However, if STAT[RDRF] is already 1, indicating that the receive data buffer is already full, STAT[OR] becomes 1 and the new data is lost.

Because the LPUART receiver is separate from the receive FIFO, the receive shift register can receive the next word when the receive FIFO is full, and it is only at the end of the character that the next data is written into the receive FIFO, potentially triggering the overrun flag if the FIFO is full.

When a program detects that the receive data register is full (STAT[RDRF] = 1), it gets the data from the FIFO by reading Data (DATA). See Interrupts for details about flag clearing.

64.3.5.1 Data sampling technique

The LPUART receiver supports a configurable oversampling rate of between 4× and 32× of the baud rate clock for sampling. The receiver starts by considering logic level samples at the oversampling rate times the baud rate to search for a falling edge on the RXD serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The oversampling baud rate clock divides the bit time into 4 to 32 segments from 1 to OSR (where OSR is the configured oversampling ratio). When a falling edge is located, three more samples are taken at (OSR ÷ 2), (OSR ÷ 2) + 1, and (OSR ÷ 2) + 2 to ensure that this is a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes they are synchronized to a received character. If another falling edge is detected before the receiver is considered synchronized, the receiver restarts sampling from the first segment.

The receiver then samples each bit time, including the start and stop bits, at (OSR ÷ 2), (OSR ÷ 2) + 1, and (OSR ÷ 2) + 2, to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, noise flag (STAT[NF]) becomes 1 when the received character is transferred to the receive FIFO.

When the LPUART receiver is configured to sample on both edges of the baud rate clock (that is, when BAUD[BOTHEDGE] = 1), the number of segments in each received bit is effectively doubled (from 1 to OSR× 2). The start and data bits are then sampled at OSR, OSR + 1, and OSR + 2. You must enable sampling on both edges of the clock for oversampling rates of 4× to 7×. This sampling is optional for higher oversampling rates.

The synchronization feature of LPUART synchronizes the internal oversampling counter with a detected falling edge on the receive signal, and to adjust the data sampling window. The falling edge detection needs three consecutive 1s prior to the "1->0" (one to zero) transition. After the initial falling edge detection for the start bit, the circuit continuously monitors the next falling edge, and resets the counter after another falling edge is detected. This synchronization to the start bit is termed as resynchronizarion.

When BAUD[RESYNCDIS] is 0, you perform this falling edge detection and resynchronization not only for the start bit but also for the rest of the character reception after the start bit.

When BAUD[RESYNCDIS] is 1, you perform the falling edge detection and resynchronization only for the start bit. The use case for disabling the resynchronization is protocols that require this (for example, LIN 2.1 prohibits resynchronization within a byte).

The following table and figure explain LPUART resynchronization.

Table 1038. LPUART resynchronization settings

Resynchronization	BAUD[RESYNCDIS] = 0	BAUD[RESYNCDIS] = 1
For the starting bit falling edge	Yes	Yes
For all falling edges after the start bit	Yes	No

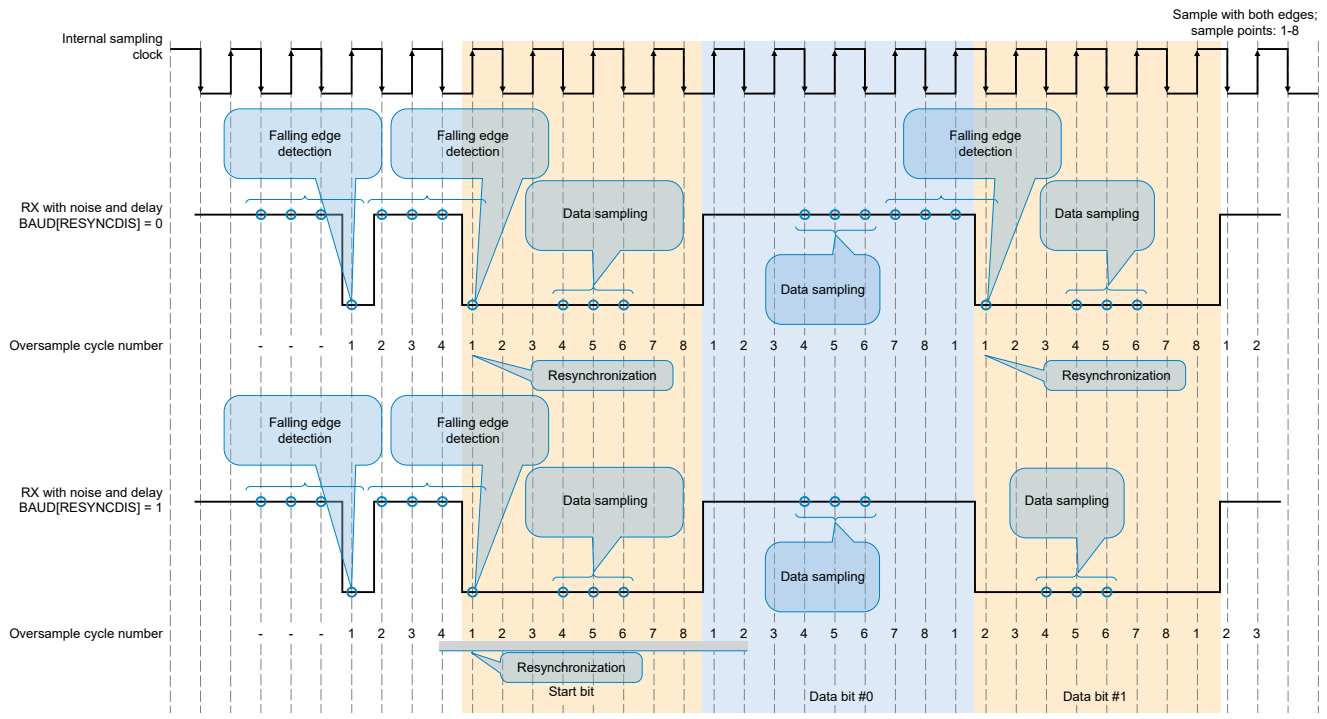


Figure 474. LPUART resynchronization diagram

64.3.5.2 Receiver wake-up operation

Receiver wake-up and receiver address matching are hardware mechanisms that allow an LPUART receiver to ignore the characters in a message intended for a different receiver.

During receiver wake-up, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write 1 to CTRL[RWU].

When CTRL[RWU] and STAT[RWUID] are 1, the status fields associated with the receiver, with the exception of STAT[IDLE], are inhibited from becoming 1, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, all receivers automatically force CTRL[RWU] to become 0. This results in all receivers waking up in time to look at the first character(s) of the next message.

During receiver address matching, the address matching is performed in hardware and the LPUART receiver ignores all characters that do not meet the address match requirements.

Table 1039. Receiver wake-up options

CTRL[RWU]	BAUD[MAEN1] | BAUD[MAEN2]	BAUD[MATCFG]	CTRL[WAKE]: STAT[RWUID]	Receiver wake-up
0	0	X	X	Normal operation
1	0	00	00	Receiver wake-up on idle line; STAT[IDLE] = 0
1	0	00	01	Receiver wake-up on idle line; STAT[IDLE] = 1
1	0	00	10	Receiver wake-up on address mark
1	1	11	10	Receiver wake-up on data match
0	1	00	X0	Address mark address match; STAT[IDLE] = 0 for discarded characters
0	1	00	X1	Address mark address match; STAT[IDLE] = 1 for discarded characters
0	1	01	X0	Idle line address match
0	1	10	X0	Match on and match off; STAT[IDLE] = 0 for discarded characters
0	1	10	X1	Match on and match off; STAT[IDLE] = 1 for discarded characters

64.3.5.2.1 Idle line wake-up

When CTRL[WAKE] is 0, you can configure the receiver for an idle line wake-up. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a full character time of the idle-line level.

CTRL[M], CTRL[M7], and BAUD[M10] select 7-bit to 10-bit Data mode and BAUD[SBNS] selects a 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 9 to 13 bit times because of the start and stop bits.

When CTRL[RWU] is 1 and STAT[RWUID] is 0, the idle condition that wakes up the receiver does not lead to STAT[IDLE] becoming 1. The receiver wakes up and waits for the first data character of the next message that leads to STAT[RDRF] becoming 1 and generates an interrupt if enabled. When STAT[RWUID] is 1, any idle condition leads to STAT[IDLE] becoming 1 and generates an interrupt if enabled, regardless of whether CTRL[RWU] is 0 or 1.

These are the ways to detect an idle line:

- When CTRL[ILT] is 0, the idle bit counter starts after the start bit so that the stop bit and any logic 1s at the end of a character count to calculate the full character time of idle.
- When CTRL[ILT] is 1, the idle bit counter does not start until after the stop bit time so that the data in the last character of the previous message does not impact the idle detection.

64.3.5.2.2 Address mark wake-up

When CTRL[WAKE] is 1, you can configure the receiver for an address mark wake-up. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a logic 1 in the most significant bit of the received character. When parity is enabled, the second most significant bit is used for address mark wake-up.

Address mark wake-up allows messages to contain idle characters, but requires one bit to be reserved for use in address frames. The logic 1 in the most significant bit (or second most significant bit when parity is enabled) of an address frame writes 0 to CTRL[RWU] and writes 1 to STAT[RDRF]. In this case, the character with the address mark bit is received even if the receiver is sleeping during most of this character time.

64.3.5.2.3 Data match wake-up

When CTRL[RWU] and CTRL[WAKE] are 1, and BAUD[MATCFG] equals 11, the receiver is configured for a data match wake-up. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a character that matches MATCH[MA1] when BAUD[MAEN1] is 1, or that matches MATCH[MA2] when BAUD[MAEN2] is 1.

64.3.5.2.4 Address match operation

You can enable the address match operation when either BAUD[MAEN1] or BAUD[MAEN2] is 1 and BAUD[MATCFG] is 0. In this function, a character that the RXD pin receives with a logic 1 in the most significant bit (or the second most significant bit when parity is enabled) is considered an address and is compared to the associated MATCH[MA1] or MATCH[MA2]. The character is only transferred to the receive buffer, and STAT[RDRF] becomes 1 if the comparison matches. All subsequent characters received with a logic 0 in the most significant bit (or the second most significant bit when parity is enabled) are considered to be data associated with the address and are transferred to the receive FIFO. If no marked address match occurs, no transfer is made to the receive FIFO, and all the characters that follow, with logic 0 in the most significant bit (or second most significant bit when parity is enabled), are also discarded. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

The address match operation functions in the same way for both MATCH[MA1] and MATCH[MA2]:

- If either BAUD[MAEN1] or BAUD[MAEN2] is 1, a marked address is compared only to the associated Match Address (MATCH) and data is transferred to the receive FIFO only on a match.
- If both BAUD[MAEN1] and BAUD[MAEN2] are 1, a marked address is compared to both MATCH[MA1] and MATCH[MA2] and data is transferred only on a match with either of these fields.

64.3.5.2.5 Idle match operation

You can enable the idle match operation when either BAUD[MAEN1] or BAUD[MAEN2] is 1 and BAUD[MATCFG] is 1. In this function, the first character that the RXD pin receives after an idle line condition is considered an address and is compared to the associated MATCH[MA1] or MATCH[MA2]. The character is transferred only to the receive buffer, and STAT[RDRF] becomes 1, if the comparison matches. All subsequent characters are considered to be data associated with the address and are transferred to the receive FIFO until the next idle line condition is detected. If no address match occurs, no transfer is made to the receive FIFO, and all the frames that follow, until the next idle condition, are also discarded. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

An idle match operation functions in the same way for both MATCH[MA1] and MATCH[MA2]:

- If either BAUD[MAEN1] or BAUD[MAEN2] is 1, the first character after an idle line is compared only to the associated Data (DATA) and data is transferred to the receive FIFO only on a match.

- If both BAUD[MAEN1] and BAUD[MAEN2] are 1, the first character after an idle line is compared to both MATCH[MA1] and MATCH[MA2] and data is transferred only on a match with either of these fields.

64.3.5.2.6 Match on, match off operation

The match on, match off operation is enabled when both BAUD[MAEN1] and BAUD[MAEN2] are 1 and BAUD[MATCFG] = 10. In this function, a character that the RXD pin receives matches MATCH[MA1] and is transferred to the receive buffer, and STAT[RDRF] becomes 1. All subsequent characters are considered to be data and are also transferred to the receive FIFO, until a character that matches MATCH[MA2] is received. The character that matches MATCH[MA2], along with all subsequent characters, is discarded; and this continues until another character that matches MATCH[MA1] is received. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

NOTE

The match on, match off operation requires both BAUD[MAEN1] and BAUD[MAEN2] to be 1.

64.3.5.3 Hardware flow control

To support hardware flow control, you can program the receiver to automatically assert and deassert RTS_B:

- RTS_B remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See Transceiver driver enable using RTS_B for more information.
- If the receiver RTS functionality is enabled, the receiver automatically deasserts RTS_B if STAT[RDRF] is 1 or a start bit is detected that causes STAT[RDRF] to become 1.
- The receiver asserts RTS_B when STAT[RDRF] is 0 and has not detected a start bit that causes STAT[RDRF] to become 1. There is no impact if STAT[RDRF] is 1 already.
- Even if RTS_B is deasserted, the receiver continues to receive characters until the receive FIFO is overrun.
- If the receiver RTS functionality is disabled, the receiver's RTS_B remains deasserted.

64.3.6 Additional LPUART functions

64.3.6.1 Data modes

You can configure the LPUART transmitter and receiver to operate in 7-bit Data mode by writing 1 to CTRL[M7], 9-bit Data mode by writing 1 to CTRL[M], or 10-bit Data mode by writing 1 to BAUD[M10]. In 9-bit Data mode, there exists a ninth data bit and in 10-bit mode, there exists a tenth data bit.

When performing 8-bit writes to the transmit FIFO, the ninth and tenth bits are pushed into the FIFO from CTRL[T8] and CTRL[T9]. For coherent 8-bit writes, you must write to CTRL[T8] and CTRL[T9] before writing to Data (DATA)[7:0]. However, if the values in CTRL[T8] or CTRL[T9] do not need to change, it is not necessary to update CTRL[T8] and CTRL[T9] before every 8-bit write to Data (DATA).

When performing 16-bit or 32-bit writes to the transmit FIFO, all 10 bits are pushed into the transmit FIFO from the write data.

When performing 8-bit reads of the receive FIFO, the ninth and tenth bits are held in CTRL[R8] and CTRL[R9] but you must read them before reading Data (DATA). A 16-bit or 32-bit read of the receive FIFO returns all 10 bits in Data (DATA).

The 9-bit Data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with the address mark wake-up so that the ninth data bit can serve as the wake-up bit. The 10-bit Data mode is typically used with parity and address mark wake-up so that the ninth data bit can serve as the wake-up bit and the tenth bit can serve as the parity bit. In custom protocols, the ninth and/or tenth bits can also serve as software-controlled markers.

64.3.6.2 Idle length

An idle character is one where the start bit, all data bits, and stop bits are in the mark position (idle state, generally logic 1). You can configure CTRL[ILT] to start detecting an idle character from the previous start bit (any data bits and stop bits count for idle character detection) or from the previous stop bit.

You can also use CTRL[IDLECFG] to configure the number of idle characters that must be received before an idle line condition is detected. This field configures the number of idle characters that must be received before STAT[IDLE] becomes 1, STAT[RAF] becomes 0, and DATA[IDLINE] becomes 1 with the next received character.

CTRL[IDLECFG] also affects the idle line wake-up and idle match operations. When either the address match or match on/off operation is enabled, writing 1 to STAT[RWUID] causes any discarded characters to be treated as idle characters.

After the extended idle time is enabled for the receiver, you can configure an idle line condition by using REIR[IDTIME], which specifies the number of bits (baud rate) since the last stop bit that is required for an idle condition to be detected. This replaces the configuration of CTRL[ILT] and CTRL[IDLECFG].

The transmitter can also enable the extended idle time. In this case, any idle character queued through the transmit FIFO forces the transmitter to be idled for the configured number of bit clocks before the idle character is read from the FIFO and the transmitter continues.

After you enable the transmitter extended idle time, the transmitter does not automatically queue an idle character whenever it is enabled.

64.3.6.3 Loop mode

Enable Loop mode by setting CTRL[LOOPS] = 1 and CTRL[RSRC] = 0. You, sometimes, use Loop mode to check software, independent of connections in the external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and LPUART does not use the RXD pin.

Loop mode also internally connects the RTS_B output to the CTS_B input and the DTR_B output to the DSR_B input.

64.3.6.4 Single-Wire mode

Enable Single-Wire mode by setting CTRL[LOOPS] = 1 and CTRL[RSRC] = 1. Single-Wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and TXD pin (the RXD pin is not used).

In Single-Wire mode, CTRL[TXDIR] controls the direction of serial data on the TXD pin. When CTRL[TXDIR] becomes 0, the TXD pin is an input to the receiver and the transmitter is temporarily disconnected from the TXD pin so that an external device can send serial data to the receiver. When CTRL[TXDIR] = 1, the TXD pin is an output that the transmitter drives. The internal loop back connection is disabled, and as a result, the receiver is unable to receive characters that the transmitter sends out.

Half Duplex Control (HDCR) replaces the implementation of CTRL[LOOPS] and CTRL[RSRC], and you can use this register to configure various options for both Single-Wire and Half-Duplex operations, using independent RXD and TXD pins:

- HDCR[TXSTALL] replaces the CTRL[TXDIR] functionality and prevents the transmitter from becoming busy or asserting the RTS_B transmitter if STAT[RAF] is 1.
- You can select the TXD pin, as the source for the receiver, to configure HDCR[RXSEL] for a single-wire operation. If HDCR[RXSEL] is 1, you must configure the TXD pin for an open-drain operation.
- HDCR[RXMSK] masks the receiver input when the RTS_B transmitter is asserted (this applies even if you have not configured RTS_B as an output).
- HDCR[RXWRMSK] blocks storage of the receive data in the receive FIFO when the RTS_B transmitter is asserted. This setting does not affect the receiver idle functionality.
- HDCR[RTSEXT] delays the negation of the RTS_B transmitter by the configured number of bit clocks (baud rate).

64.3.6.5 Timeout counter

LPUART implements four general-purpose timeout counters; counters 0 and 1 are used to monitor the receiver and counters 2 and 3 are used to monitor the transmitter. When enabled, you can configure each counter to monitor one of the following conditions:

- Idle time in number of bits, starting to increment when first enabled and an idle condition is detected. The counter restarts whenever a character is received or transmitted.
- Idle time in number of bits, starting to increment after the next character is received or transmitted. The counter restarts whenever a character is received or transmitted.

- Idle time is more than the timeout interval, in number of bits, but less than the extended idle timeout. The counter restarts whenever a character is received or transmitted. You can use this to detect a gap between characters that is greater than a threshold (timeout) but less than the configured extended idle time. This configuration requires the extended idle feature for the transmitter or receiver to be enabled for a proper operation.
- Number of characters received or transmitted is equal to the configured timeout. The counter asserts at the start of the character that equals the timeout value.

The timeout counters restart counting whenever the counter is disabled and then enabled. These timeout counters are disabled when the corresponding STAT[TSF] field is 1. If the timeout enable is still set after STAT[TSF] becomes 0, the timeout counter restarts as if the counter had been disabled and then enabled. You can measure the idle time in bit times from the end of the last stop bit and until a start bit is validated.

64.3.7 Peripheral triggers

The connection of the LPUART peripheral triggers with other peripherals is chip-specific.

64.3.7.1 Output triggers

LPUART generates the following output triggers that can be connected to other peripherals on the chip:

- The transmit word trigger asserts at the end of each transmitted word and negates after 1-bit period.
- The transmit data trigger is identical to the TXD pin output, but without support for input trigger modulation.
- The receive word trigger asserts at the end of each received word that is written to the receive FIFO, for one oversampling clock period.
- The receive idle trigger asserts when STAT[IDLE] becomes 1, and negates when the next valid start bit is detected.

64.3.7.2 Input trigger

LPUART supports a peripheral input trigger that you can configure in one of the following ways:

- By enabling the CTS function: You can connect the input trigger instead of the CTS_B pin input. The input trigger must assert for longer than 1-bit clock period when the transmitter is idle, with data to send, to guarantee a new transmission.
- By making the input trigger modulate the transmit data output (trigger is logically ANDed with the TXD output): The input trigger is expected to be a free-running clock (carrier signal) that generates from a timer or PWM source with a frequency that is greater than the bit-clock frequency. The carrier signal must not toggle faster than the maximum supported bit time.
- By connecting the input trigger instead of the RXD pin input: The input trigger is expected to be generated from a receive data source, such as an analog comparator or external pin.

64.3.8 Infrared (IR) interface

LPUART provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses, transforming them to serial bits, which are then sent to LPUART. The IrDA physical layer specification defines a half-duplex IR communication link for exchanging data. The full standard includes data rates up to 16 Mbit/s. The LPUART IrDA support is limited to SIR mode that supports data rates only between 2.4 kbit/s and 115.2 kbit/s.

LPUART has an infrared transmit encoder and a receive decoder. The infrared decoder converts the received character from the IrDA format to the NRZ format, which the receiver uses. It also has an OSR oversampling baud rate clock counter that filters noise and indicates when a 1 is received. LPUART transmits serial bits of data, which the infrared submodule encodes, to transmit a narrow pulse for every zero bit. No pulse is transmitted for every single bit. When receiving data, an IR photo diode (external to LPUART) detects the IR pulses. The IR receive decoder transforms them to CMOS levels. The infrared receive decoder then stretches the narrow pulses to get back to a serial bit stream that LPUART receives. You can invert the polarity of transmitted pulses and expected receive pulses so that a direct connection can be made to external IrDA transceiver modules that use active-high pulses.

The IR submodule receives its clock sources from LPUART. The submodule selects one of these clocks to generate either 1 ÷ OSR, 2 ÷ OSR, 3 ÷ OSR, or 4 ÷ OSR narrow pulses during transmission.

64.3.8.1 Infrared transmit encoder

The infrared transmit encoder converts serial bits of data from the transmit shift register to the TXD signal. A narrow pulse is transmitted for a 0 bit and no pulse is transmitted for a 1 bit. The narrow pulse is sent at the start of the bit with a duration of 1 ÷ OSR, 2 ÷ OSR, 3 ÷ OSR, or 4 ÷ OSR of a bit time. A narrow low pulse is transmitted for a 0 bit when CTRL[TXINV] is 0, while a narrow high pulse is transmitted for a 0 bit when CTRL[TXINV] is 1.

64.3.8.2 Infrared receive decoder

The infrared receive block converts data from the RXD signal to the receive shift register. A narrow pulse is expected for each 0 received and no pulse is expected for each 1 received. A narrow low pulse is expected for a 0 bit when STAT[RXINV] is 0, while a narrow high pulse is expected for a 0 bit when STAT[RXINV] is 1. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

64.3.8.3 Start-bit detection

When STAT[RXINV] is 0, the first falling edge of the received character corresponds to the start bit. The infrared decoder resets its counter. At this time, the receiver also begins its start bit detection process. After the start bit is detected, the receiver synchronizes its bit times to this start bit time. For the rest of the character reception, the infrared decoder's counter and the receiver's bit time counter count independently of each other.

64.3.8.4 Noise filtering

The decoder ignores any rising edges detected during the first half of the infrared decoder counter, and can leave any pulses less than one oversampling baud clock as undetected. This is regardless of whether the pulse is seen in the first or second half of the count.

64.3.8.5 Low-bit detection

During the second half of the decoder count, a rising edge is decoded as 0, which is sent to the receiver. The decoder counter is also reset.

64.3.8.6 High-bit detection

At OSR oversampling baud rate clocks after the previous rising edge, if a rising edge is not seen, the decoder sends a 1 to the receiver.

If the next bit is 0, which arrives late, a low bit is detected according to Low-bit detection. The value sent to the receiver is changed from 1 to 0. Then, if a noise pulse occurs outside the receiver's bit time sampling period, the delay of a 0 is not recorded as noise.

64.3.9 MODBUS protocol

64.3.9.1 MODBUS frame structure

MODBUS is an application layer messaging protocol, positioned at level 7 of the OSI model. It provides client and server communication between devices connected on different types of buses or networks.

The MODBUS protocol defines a simple protocol data unit (PDU) independent of the underlying communication layers. The mapping of the MODBUS protocol on specific buses or network can introduce some additional fields on the application data unit (ADU).

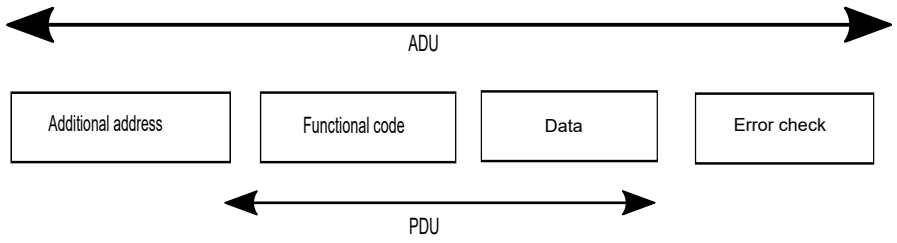


Figure 475. MODBUS frame structure

The function code field informs the server about which action to perform, as requested by the client (only the client can initiate the communication), where the data field contains additional information that the server uses to take the action defined by the function code. MODBUS PDU for serial line communication = 256 - server address (1 byte) - CRC (2 bytes) = 253 bytes.

You can set up a MODBUS controller to communicate on standard MODBUS networks using either of the two transmission modes: ASCII or RTU. RTU mode allows better character density and throughput for the same baud rate as the ASCII format allows 7 bits to represent a character where RTU keeps 8 bits for data representation in each packet.

64.3.9.2 MODBUS frame for LPUART

Though the protocol resides in the application layer, it can also be checked at the physical layer using LPUART by considering the frame structure shown in the following figure, following RTU mode of data transmission.

Start	Address	Function	Data	CRC	End
3.5 Char time	8 Bit	8 Bit	N * 8Bit	16 Bit	3.5 Char time

Figure 476. MODBUS frame for LPUART

An entire message frame of MODBUS must contain start character, address, function code, data, and end character. LPUART does not support CRC calculation for TX and RX, instead, each LPUART packet containing frame information is transmitted with its individual parity, and the same is checked in RX. Following the last transmitted character, a similar interval of at least 3.5 character times marks the end of the message and a new message can begin after this interval. The entire message frame must be transmitted as a continuous stream. If a silent interval of more than 1.5 character times occurs before the completion of the frame, the receiving device flushes the incomplete message, and assumes that the next byte is the address field of a new message. The correctness of timeout logic ensures the correctness of the MODBUS frame using LPUART.

Registers such as REIR, TEIR, TOCR, TOSR, TIMEOUT[0 - 3], TCBR, and TDBR are useful for realizing MODBUS using LPUART. The TX IDLE time is configured by using TEIR[IDTIME].

For higher baud rates, MODBUS recommends adopting fixed t1.5 and t3.5 times of 750us ms and 1.75 ms. The receiver idle line wake-up can be used by a MODBUS device to only wake-up on a matching address or broadcast address received after a valid idle time. The receiver end-of-packet DMA transfer can be used to offload the reception of a MODBUS packet onto the DMA controller.

64.3.9.3 MODBUS TX and RX programming sequence

The following programming sequences can be useful for sending and receiving MODBUS frames using LPUART.

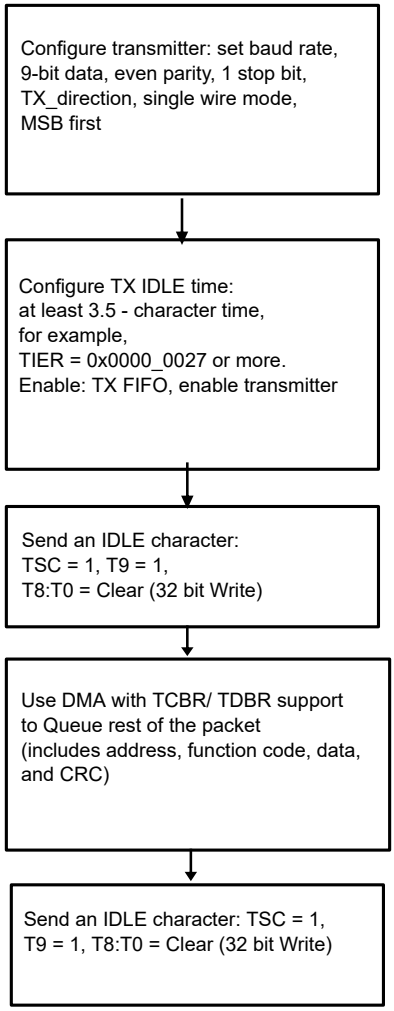


Figure 477. MODBUS TX programming

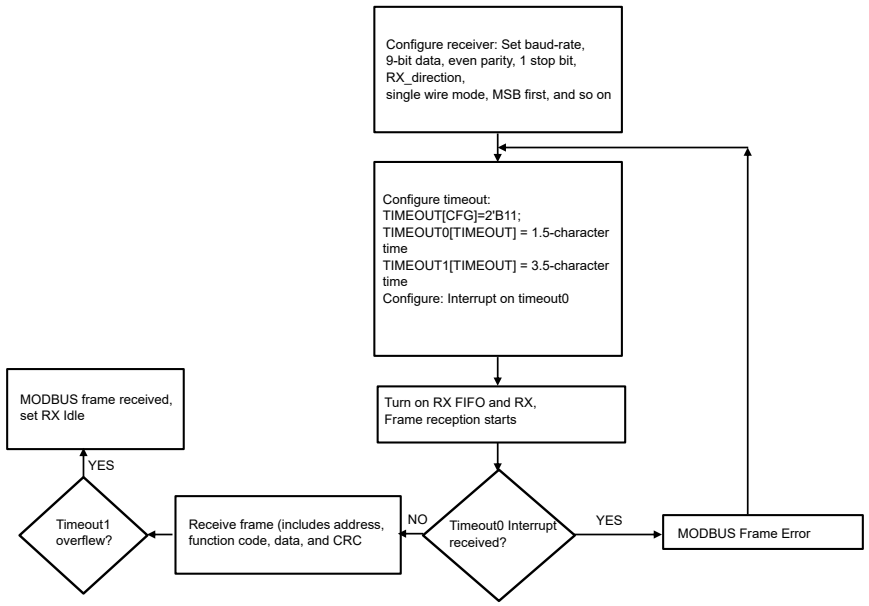


Figure 478. MODBUS RX programming

64.3.10 Modes of operation

64.3.10.1 Low-Power modes

LPUART remains functional during low-power modes, provided CTRL[DOZEEN] is 0 and the LPUART functional clock is enabled. LPUART can generate an interrupt or DMA request to cause a wake-up from low-power modes.

You can configure LPUART to be disabled in low-power modes, when CTRL[DOZEEN] is 1. In this case, the transmitter and receiver finish transmitting and receiving the current word.

If LPUART is disabled in low-power modes, it can generate a wake-up via STAT[RXEDGIF] if the receiver detects an active edge.

NOTE

See the chip-specific information for specific low-power modes available on your chip.

64.3.10.2 Debug mode

LPUART remains functional in Debug mode.

64.3.11 Clocking

Table 1040. Types of clocks

Clock	Description
Functional	Is asynchronous to the bus clock and can remain enabled in Low-Power modes to support transmit and/or receive functions, including low-power wake-up.
Bus	Is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPUART transmit and receive registers, including the FIFOs.

64.3.12 Reset

Table 1041. Types of resets

Reset	Description
Chip	Enables the logic and registers for the LPUART transmitter and receiver to reset to their default states.
Software	Resets the LPUART logic and registers to their default states, except for Global (GLOBAL). GLOBAL[RST] controls the LPUART software reset.
FIFO	Implements write-only control fields that reset the transmit FIFO (FIFO[TXFLUSH]) and receive FIFO (FIFO[RXFLUSH]). After a FIFO is reset, that FIFO becomes empty.

64.3.13 Interrupts

The LPUART transmitter has two status fields that can optionally generate hardware interrupt requests. If STAT[TDRE] is 1, it indicates that there is room in the transmit FIFO to write another transmit character to Data (DATA). If CTRL[TIE] is 1, a hardware interrupt is requested when STAT[TDRE] is 1.

STAT[TC] indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with TXD at the inactive level. This field is often used in systems with modems to determine when it is safe to turn off the modem. If CTRL[TCIE] is 1, a hardware interrupt is requested when STAT[TC] is 1. Instead of hardware interrupts, software polling may be used to monitor STAT[TDRE] and STAT[TC] if the corresponding CTRL[TIE] or CTRL[TCIE] field is 0.

When a program detects that STAT[RDRF] is 1, it gets the data from this field by reading Data (DATA). The field becomes 0 by reading Data (DATA).

STAT[IDLE] includes logic that prevents it from becoming 1 repeatedly when the RXD line remains idle for an extended period of time. STAT[IDLE] becomes 0 when you write 1 to it, and cannot become 1 again until the receiver has received at least one new character and has 1 as the value of STAT[RDRF].

If the associated error is detected in the received character that caused STAT[RDRF] to become 1, STAT[NF], STAT[FE], and STAT[PF] become 1 at the same time STAT[RDRF] becomes 1. These flags do not become 1 in overrun cases.

If STAT[RDRF] is already 1 when a new character is ready to be transferred from the receive shifter to the receive FIFO, STAT[OR] becomes 1, instead of the data along with any associated STAT[NF], STAT[FE], or STAT[PF] condition getting lost.

If the received character matches the contents of MATCH[MA1] and/or MATCH[MA2], then STAT[MA1F] and/or STAT[MA2F] become 1 at the same time that STAT[RDRF] becomes 1.

At any time, an active edge on the RXD serial data input pin causes STAT[RXEDGIF] to become 1. STAT[RXEDGIF] becomes 0 when you write 1 to it. This function depends on the receiver being enabled (the value of CTRL[RE] being 1).

MODEM Status (MSR) can generate an interrupt from a configured status field, which STAT[MSF] indicates.

Timeout Status (TOSR) can generate an interrupt from a configured status field, which STAT[TSF] indicates.

64.3.14 DMA

64.3.14.1 DMA burst support

To support efficient DMA transfers to the transmit FIFO, two alias regions are implemented to support incrementing 8-bit, 16-bit, or 32-bit write accesses to the transmit FIFO:

- Transmit Command Burst (TCBR0 - TCBR127) is a 512-byte region that supports pushing 16-bit data into the transmit FIFO.
- Transmit Data Burst (TDBR0 - TDBR255) is a 1024-byte region that supports pushing zero extended 8-bit data into the transmit FIFO.

The aforementioned regions are contiguous, so a DMA transfer can start in Transmit Command Burst (TCBR0 - TCBR127) to initialize the transfer including address mark, idle word, or break character, and then complete the transfer in Transmit Data Burst (TDBR0 - TDBR255) with the data to transmit, without changing the transfer size.

The transmit FIFO block writes overflow the FIFO, but that does not signal an error. Do not perform 32-bit writes to Transmit Data Burst (TDBR0 - TDBR255) unless there are four empty slots in the transmit FIFO, and do not perform 16-bit writes to this register unless there are two empty slots in the transmit FIFO.

64.3.14.2 End-of-packet DMA transfers

The end-of-packet functionality is designed for serial interfaces where you may not know the size of the transfer in advance and the data is being pushed by an external device. The end-of-packet processing ensures that data does not become stranded in either the receive FIFO or the DMA receive buffer. Support for end-of-packet processing must be implemented in both the serial interfaces and DMA controller.

The condition that signals the end of packet is different for each serial interface, but the serial peripheral and DMA process it in the same way. For example, an idle line condition signals the UART end of packet, the Stop and/or Repeated Start condition signals the I2C end of packet, and PCS negation signals the SPI end of packet.

When the serial peripheral is configured to signal the end-of-packet condition to the DMA and the serial interface detects an end-of-packet condition, it asserts the DMA request for the receive FIFO irrespective of the watermark configuration. For larger watermark configurations, this ensures that the last few words of the transfer are first flushed from the receive FIFO.

The DMA then reads the contents of the receive FIFO, depending on the first word in the FIFO:

- If the receive FIFO is empty, the serial interface signals an end of packet condition to the DMA controller.
- If the receive FIFO is not empty, but the first word in the FIFO is the start of a new packet, data is not pulled from the receive FIFO and the serial interface signals an end-of-packet condition to the DMA controller.
- If the receive FIFO is not empty, and the first word in the FIFO is not the start of a new packet, the DMA transfers the receive data as normal.

Because the DMA may be transferring multiple words on each request, the end-of-packet condition persists until the DMA minor loop has completed and no additional data is pulled from the receive FIFO. The field that triggered the end-of-packet condition becomes 0 when the minor loop completes following the end of packet being signaled to the DMA controller.

When the DMA detects the end-of-packet condition, it writes all received words up to the end of packet into the system memory and saves the destination address for the word after the last valid data. The DMA then terminates the channel as if the major loop completed, including final offsets and optional interrupts, channel linking, and scatter-gather. You can, optionally, save the final destination address to the system memory. The final destination address is also available in the destination address register.

Because the DMA terminates the major loop, no servicing of the receive FIFO occurs until you or the hardware reconfigures the DMA (for example, channel linking or scatter-gather). You must minimize this delay to avoid receiver FIFO overrun. The automatic DMA end-of-packet processing is not recommended when there are only a few words transferred between end-of-packet conditions. This is because the DMA spends more time processing the end of packet than transferring the data. For example, the UART idle line length must be increased as needed to avoid an excessive number of idle conditions.

64.4 External signals

Table 1042. External signals

Signal	Description	I/O
TXD	Transmit data: This pin is normally an output, but is an input (tristated) in Single-Wire mode whenever the transmitter is disabled or the transmit direction is configured for receive data.	I/O

Table continues on the next page...

Table 1042. External signals (continued)

Signal	Description	I/O
RXD	Receive data	I
CTS_B	Clear-to-send	I
RTS_B	Request-to-send	O
DTR_B	Data terminal ready	O
DSR_B	Data set ready	I
DCD_B	Data carrier detect	I
RIN_B	Ring indicator	I

64.5 Initialization

This module does not require initialization.

64.6 Register definition

LPUART includes registers to control baud rate, select options, report status, and store transmit and receive data. Access to an address outside the valid memory map generates a bus error.

NOTE

Writing to a read-only (RO) register or reading a write-only (WO) register can cause bus errors. LPUART does not verify whether programmed values in the registers are correct; you must write valid values to them.

64.6.1 LPUART register descriptions

64.6.1.1 LPUART memory map

LPUART1 base address: 4438_0000h

LPUART2 base address: 4439_0000h

LPUART3 base address: 4257_0000h

LPUART4 base address: 4258_0000h

LPUART5 base address: 4259_0000h

LPUART6 base address: 425A_0000h

LPUART7 base address: 4457_0000h

LPUART8 base address: 42DA_0000h

LPUART9 base address: 42D7_0000h

LPUART10 base address: 42D8_0000h

LPUART11 base address: 42D9_0000h

LPUART12 base address: 4458_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0404_0007h
4h	Parameter (PARAM)	32	R	0000_0404h
8h	Global (GLOBAL)	32	RW	0000_0000h
Ch	Pin Configuration (PINCFG)	32	RW	0000_0000h
10h	Baud Rate (BAUD)	32	RW	0F00_0004h
14h	Status (STAT)	32	RW	00C0_0000h
18h	Control (CTRL)	32	RW	0000_0000h
1Ch	Data (DATA)	32	RW	0000_1000h
20h	Match Address (MATCH)	32	RW	0000_0000h
24h	MODEM IrDA (MODIR)	32	RW	0000_0000h
28h	FIFO (FIFO)	32	RW	00C0_0033h
2Ch	Watermark (WATER)	32	RW	0000_0000h
30h	Data Read-Only (DATARO)	32	R	0000_1000h
40h	MODEM Control (MCR)	32	RW	0000_0000h
44h	MODEM Status (MSR)	32	RW	0000_0000h
48h	Receiver Extended Idle (REIR)	32	RW	0000_0000h
4Ch	Transmitter Extended Idle (TEIR)	32	RW	0000_0000h
50h	Half Duplex Control (HDCR)	32	RW	0000_0000h
58h	Timeout Control (TOCR)	32	RW	0000_0000h
5Ch	Timeout Status (TOSR)	32	RW	0000_000Fh
60h - 6Ch	Timeout N (TIMEOUT0 - TIMEOUT3)	32	RW	0000_0000h
200h - 3FCh	Transmit Command Burst (TCBR0 - TCBR127)	32	W	0000_0000h
400h - 7FCh	Transmit Data Burst (TDBR0 - TDBR255)	32	W	0000_0000h

64.6.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function
Indicates the version integrated for this instance on the chip and also specifies the inclusion and exclusion of several optional features.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the module specification.
23-16 MINOR	Minor Version Number Indicates the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number Indicates the feature set number. 0000_0000_0000_0001b - Standard feature set 0000_0000_0000_0011b - Standard feature set with MODEM and IrDA support 0000_0000_0000_0111b - Enhanced feature set with full MODEM, IrDA, and enhanced idle detection

64.6.1.3 Parameter (PARAM)

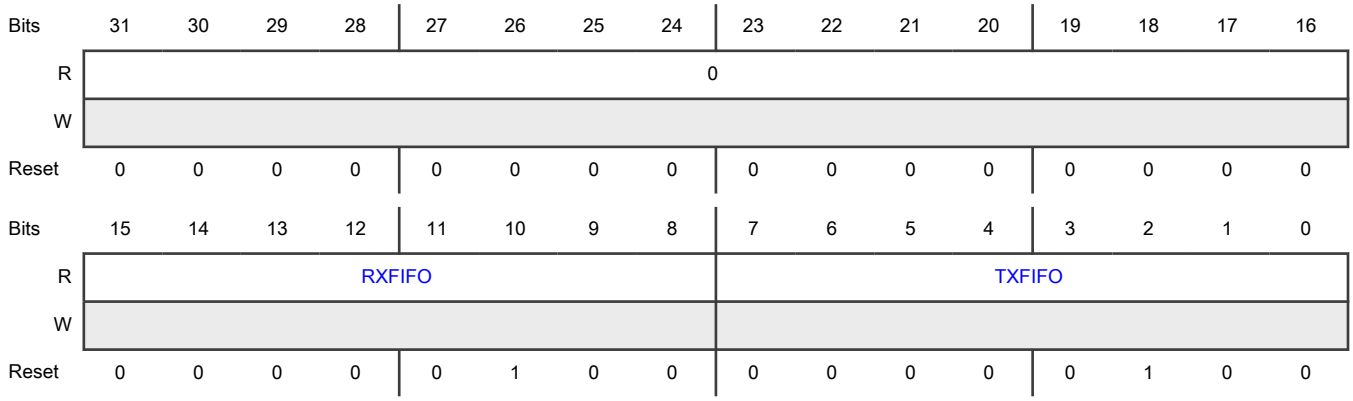
Offset

Register	Offset
PARAM	4h

Function

Indicates the parameter configuration for this instance on the chip.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 RXFIFO	Receive FIFO Size Indicates the number of characters in the receive FIFO, which is 2^RXFIFO.
7-0 TXFIFO	Transmit FIFO Size Indicates the number of characters in the transmit FIFO, which is 2^TXFIFO.

64.6.1.4 Global (GLOBAL)

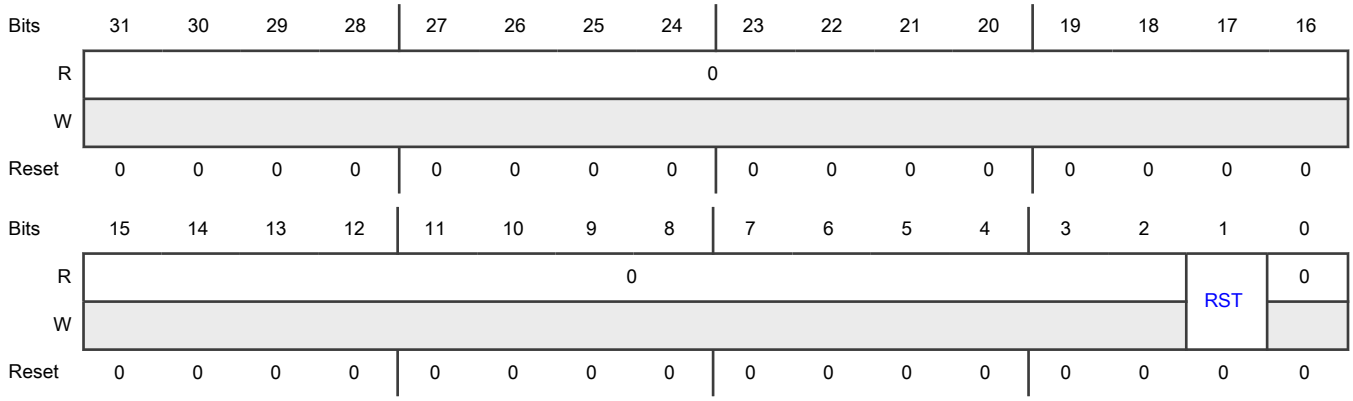
Offset

Register	Offset
GLOBAL	8h

Function

Performs global functions.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RST	Software Reset Specifies whether the module is reset. This field resets all internal logic and registers, except Global (GLOBAL). The reset takes effect immediately and remains asserted until you negate it. There is no minimum delay required before clearing the software reset. 0b - Not reset 1b - Reset
0 —	Reserved

64.6.1.5 Pin Configuration (PINCFG)

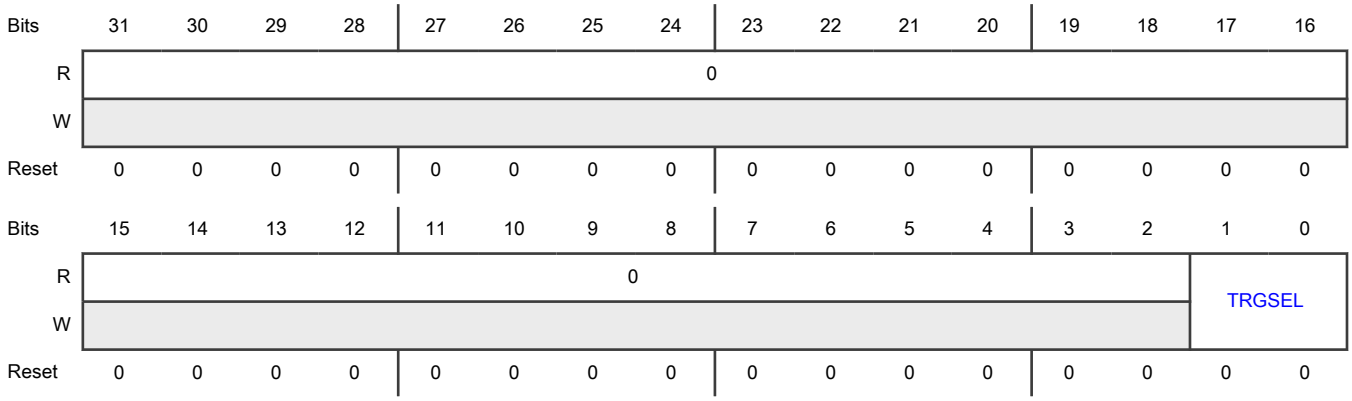
Offset

Register	Offset
PINCFG	Ch

Function

Enables the selection of input pins.

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 TRGSEL	Trigger Select Configures the input trigger usage. You must change the value of this field only when both the transmitter and receiver are disabled. 00b - Input trigger disabled 01b - Input trigger used instead of the RXD pin input 10b - Input trigger used instead of the CTS_B pin input 11b - Input trigger used to modulate the TXD pin output, which (after TXINV configuration) is internally ANDed with the input trigger

64.6.1.6 Baud Rate (BAUD)

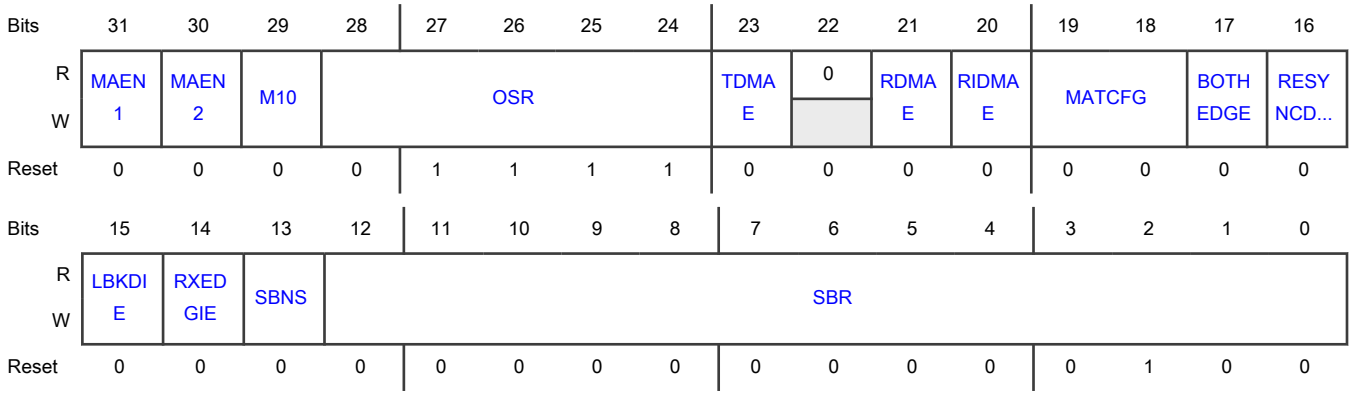
Offset

Register	Offset
BAUD	10h

Function

Configures the baud rate.

Diagram



Fields

Field	Function
31 MAEN1	Match Address Mode Enable 1 Enables automatic address matching or data matching mode for MATCH[MA1]. If this field is 0, normal operation takes place. 0b - Disable 1b - Enable
30 MAEN2	Match Address Mode Enable 2 Enables automatic address matching or data matching mode for MATCH[MA2]. If this field is 0, normal operation takes place. 0b - Disable 1b - Enable
29 M10	10-Bit Mode Select Causes the tenth bit to be a part of the serial transmission. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - Receiver and transmitter use 7-bit to 9-bit data characters 1b - Receiver and transmitter use 10-bit data characters
28-24 OSR	Oversampling Ratio Configures the OSR of the receiver. You must change the value of this field only when both the transmitter and receiver are disabled. **NOTE** BAUD[OSR] results in an OSR of BAUD[OSR] + 1, for example, BAUD[OSR] = 0_0101b results in a final division by 6. 0_0000b - Results in an OSR of 16 0_0001b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0_0010b - Reserved 0_0011b - Results in an OSR of 4 (requires BAUD[BOTHEDGE] to be 1) 0_0100b - Results in an OSR of 5 (requires BAUD[BOTHEDGE] to be 1) 0_0101b - Results in an OSR of 6 (requires BAUD[BOTHEDGE] to be 1) 0_0110b - Results in an OSR of 7 (requires BAUD[BOTHEDGE] to be 1) 0_0111b - Results in an OSR of 8 0_1000b - Results in an OSR of 9 0_1001b - Results in an OSR of 10 0_1010b - Results in an OSR of 11 0_1011b - Results in an OSR of 12 0_1100b - Results in an OSR of 13 0_1101b - Results in an OSR of 14 0_1110b - Results in an OSR of 15 0_1111b - Results in an OSR of 16 1_0000b - Results in an OSR of 17 1_0001b - Results in an OSR of 18 1_0010b - Results in an OSR of 19 1_0011b - Results in an OSR of 20 1_0100b - Results in an OSR of 21 1_0101b - Results in an OSR of 22 1_0110b - Results in an OSR of 23 1_0111b - Results in an OSR of 24 1_1000b - Results in an OSR of 25 1_1001b - Results in an OSR of 26 1_1010b - Results in an OSR of 27 1_1011b - Results in an OSR of 28 1_1100b - Results in an OSR of 29 1_1101b - Results in an OSR of 30 1_1110b - Results in an OSR of 31 1_1111b - Results in an OSR of 32
23 TDMAE	Transmitter DMA Enable Enables STAT[TDRE] to generate a DMA request. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
22 —	Reserved
21 RDMAE	Receiver Full DMA Enable Enables STAT[RDRF] to generate a DMA request. 0b - Disable 1b - Enable
20 RIDMAE	Receiver Idle DMA Enable Enables STAT[IDLE] to generate a DMA request. If this field is 1, reading Data (DATA) when either DATA[RXEMPT] or DATA[IDLINE] is 1 generates an end-of-packet response until the completion of the DMA minor loop. During an end-of-packet response, reading Data (DATA) returns 0000_33FFh and does not pull data from the receive FIFO. STAT[IDLE] becomes 0 on completion of the minor loop, provided an end-of-packet response is generated and either the receive FIFO is empty or the receiver is active. 0b - Disable 1b - Enable
19-18 MATCFG	Match Configuration Configures the match addressing mode used. You must change the value of this field only when both the transmitter and receiver are disabled. 00b - Address match wake-up 01b - Idle match wake-up 10b - Match on and match off 11b - Enables RWU on data match and match on or off for the transmitter CTS input
17 BOTHEDGE	Both Edge Sampling Enables sampling of the received data on both edges of the baud rate clock, effectively doubling the number of times the receiver samples the input data for a given OSR. This field must be 1 for OSRs between x4 and x7 and is optional for higher OSRs. You must change the value of this field only when the receiver is disabled. If this field is 0, the receiver samples input data using the rising edge of the baud rate clock. If this field is 1, the receiver samples input data using the rising and falling edges of the baud rate clock. 0b - Rising edge 1b - Both rising and falling edges
16 RESYNCDIS	Resynchronization Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Disables resynchronization of the received data word when a data one followed by data zero transition is detected. You must change the value of this field only when the receiver is disabled. 0b - Enable 1b - Disable
15 LBKDIE	LIN Break Detect Interrupt Enable Enables STAT[LBKDIF] to generate hardware interrupt requests. If this field is 0, hardware interrupts from STAT[LBKDIF] (uses polling) are disabled. If this field is 1, hardware interrupts are requested when STAT[LBKDIF] is 1. 0b - Disable 1b - Enable
14 RXEDGIE	RX Input Active Edge Interrupt Enable Enables STAT[RXEDGIF] to generate interrupt requests. If this field is 0, hardware interrupts from STAT[RXEDGIF] are disabled. If this field is 1, hardware interrupts are requested when STAT[RXEDGIF] is 1. Changing the value of CTRL[LOOPS] or CTRL[RSRC] when this field (RXEDGIE) is 1 can cause STAT[RXEDGIF] to become 1. 0b - Disable 1b - Enable
13 SBNS	Stop Bit Number Select Determines whether data characters include one or two stop bits. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - One stop bit 1b - Two stop bits
12-0 SBR	Baud Rate Modulo Divisor Sets the modulo divide rate for the baud rate generator. • If SBR is 0, baud rate generator is disabled. • If SBR is 1–8191, baud rate = baud clock ÷ ((OSR + 1) × SBR). You must update the 13-bit baud rate setting [SBR12:SBR0] only when both the transmitter and receiver are disabled (both CTRL[RE] and CTRL[TE] are 0).

64.6.1.7 Status (STAT)

Offset

Register	Offset
STAT	14h

Function

Provides the module status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LBKDIF	RXEDGIF	MSBF	RXINV	RWUID	BRK13	LBKDE	RAF	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W	W1C	W1C										W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MA1F	MA2F	0				TSF	MSF	0						AME	LBKFE
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LBKDIF	LIN Break Detect Interrupt Flag Indicates whether a LIN break character is detected. This field becomes 1 when the LIN break detect circuitry is enabled and a LIN break character is detected. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
30 RXEDGIF	RXD Pin Active Edge Interrupt Flag Indicates whether an active edge on the receive pin has occurred.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 whenever the receiver is enabled and an active edge (falling if STAT[RXINV] is 0; rising if STAT[RXINV] is 1) on the RXD pin occurs. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not occurred 1b - Occurred When writing 0b - No effect 1b - Clear the flag
29 MSBF	MSB First Specifies the first bit that is transmitted after the start bit. If this field is 0, LSB (bit 0) is the first bit transmitted after the start bit (which means, the first bit received after the start bit is identified as bit 0). If this field is 1, MSB (identified as bit 9, bit 8, bit 7, or bit 6) is the first bit that is transmitted, after the start bit, depending on the settings of CTRL[M], CTRL[PE], and BAUD[M10]. Writing 1 to this field reverses the order of the bits that are transmitted and received on the wire. This field does not affect the polarity of the bits, the location of the parity bit, or the location of the start or stop bits. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - LSB 1b - MSB
28 RXINV	Receive Data Inversion Specifies whether receive data is inverted. Writing 1 to this field reverses the polarity of the received data input. You must change the value of this field only when the receiver is disabled. **NOTE** Writing 1 to this field inverts the RXD input for all cases: data bits, start and stop bits, break, and idle. 0b - Inverted 1b - Not inverted
27 RWUID	Receive Wake Up Idle Detect Controls, for CTRL[RWU] on idle character detection, whether the idle character that wakes up the receiver writes 1 to STAT[IDLE]. For address match wake-up, this field controls whether STAT[IDLE] = 1 when the address does not match. You must change the value of this field only when the receiver is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 0, during the Receive Standby state (CTRL[RWU] = 1), STAT[IDLE] does not become 1 upon detection of an idle character. During address match wake-up, STAT[IDLE] does not become 1 when an address does not match. If this field is 1, during the Receive Standby state (CTRL[RWU] = 1), STAT[IDLE] becomes 1 upon detection of an idle character. During address match wake-up, STAT[IDLE] becomes 1 when an address does not match. 0b - STAT[IDLE] does not become 1 1b - STAT[IDLE] becomes 1
26 BRK13	Break Character Generation Length Selects the longer transmitted break character length. The state of this field does not affect the detection of a framing error. You must change the value of this field only when the transmitter is disabled. You can send a break character by writing 1 to CTRL[SBK], or by writing the transmit FIFO when DATA[FRETSC] is 1 and DATA[R9T9] is 0. 0b - 9 to 13 bit times 1b - 12 to 15 bit times
25 LBKDE	LIN Break Detection Enable Enables LIN break detection. If this field is 0, LIN break detect is disabled, and only a normal break character can be detected. If this field is 1, LIN break detect is enabled and the LIN break character is detected at a length of 11 bit times (if CTRL[M] is 0), 12 bit times (if CTRL[M] is 1), or 13 bit times (if BAUD[M10] is 1). This field selects a longer break character detection length. When the field is 1, receive data is not stored in the receive FIFO. **NOTE** This field enables the LIN break detect circuit and disables writing receive data to FIFO. Therefore, it ignores all characters except a LIN break. 0b - Disable 1b - Enable
24 RAF	Receiver Active Flag Indicates whether the LPUART receiver is idle or active. This field becomes 1 when the receiver detects the beginning of a valid start bit, and the field becomes 0 automatically when the receiver detects an idle line. 0b - Idle, waiting for a start bit 1b - Receiver active (RXD pin input not idle)
23 TDRE	Transmit Data Register Empty Flag Indicates whether the transmit FIFO level is greater than, equal to, or less than the watermark.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After the transmit FIFO is enabled, this field becomes 1 when the number of datawords in the transmit FIFO is equal to, or less than the number that WATER[TXWATER] indicates. To make the value of this field 0, write to it until the number of words in the transmit FIFO is greater than the number that WATER[TXWATER] indicates. After the transmit FIFO is disabled, this field becomes 1 to indicate that the FIFO level is less than the watermark. To make the value of this field 0 again, write to Data (DATA). This register is not affected by a character that is in the process of being transmitted; it is updated at the start of each transmitted character. 0b - Greater than watermark 1b - Equal to or less than watermark
22 TC	Transmission Complete Flag Indicates whether the transmitter is active. This field becomes 0 when a transmission is in progress or a preamble or break character is loaded; in other words, when the transmitter is active (sending data, a preamble, or a break). The field becomes 1 when the transmit buffer is empty and no data, preamble, or break character is being transmitted; in other words, when the transmission activity is complete. When this happens, the transmit data output signal becomes idle (logic 1). This field becomes 0 after you write to Data (DATA) to transmit new data, queuing a preamble by first writing 0 and then writing 1 to CTRL[TE], queuing a break character by writing 1 to CTRL[SBK]. 0b - Transmitter active 1b - Transmitter idle
21 RDRF	Receive Data Register Full Flag Indicates whether the receive FIFO level is less than, equal to, or greater than the watermark. This field becomes 1 when the number of datawords in the receive buffer is greater than the number that WATER[RXWATER] indicates and the receive FIFO is enabled. To write 0 to this field, read Data (DATA) until the number of datawords in the receive FIFO is equal to, or less than the number that WATER[RXWATER] indicates. When the receive FIFO is disabled, this field (RDRF) becomes 1 if the receive buffer (Data (DATA)) is full. To make this field 0, read Data (DATA). A character that is in the process of being received does not cause a change in this field until the entire character is received. Even if this field is 1, the character continues to be received until an overrun condition occurs after the entire character is received. 0b - Equal to or less than watermark 1b - Greater than watermark
20 IDLE	Idle Line Flag Indicates whether an idle line is detected. This field becomes 1 when the LPUART receive line becomes idle for a full character time after a period of activity. When CTRL[ILT] is 0, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bit time count towards the full character time of logic high, 10 to 13 bit times, needed for the receiver to detect an idle line. After CTRL[ILT] becomes 1, the receiver does not start counting idle bit times until after the stop bits. The stop bits and any logic high bit times at the end

Table continues on the next page...

Table continued from the previous page...

Field	Function
	of the previous character do not count towards the full character time of logic high needed for the receiver to detect an idle line. For this field to become 0, write 1 to it. After the field becomes 0, you cannot write 1 to it again until after a new character is stored in the receive buffer or a LIN break character writes 1 to STAT[LBKDIF]. This field becomes 1 only once, even if the receive line remains idle for an extended period. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Idle line detected 1b - Idle line not detected When writing 0b - No effect 1b - Clear the flag
19 OR	Receiver Overrun Flag Indicates whether there is receive overrun. This field becomes 1 when you cannot prevent STAT[RDRF] from overflowing with data. The field becomes 1 immediately after the stop bit is completely received for the dataword that overflows the buffer and all the other error fields (STAT[FE], STAT[NF], and STAT[PF]) are prevented from becoming 1. The data in the shift register is lost, but the data already in the LPUART data registers is not affected. If STAT[LBKDE] is enabled and a LIN break is detected, this field becomes 1 if STAT[LBKDIF] is not 0 before the next data character is received. When this field is 1, no additional data is stored in the receive FIFO even if sufficient room exists. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overrun 1b - Receive overrun (new LPUART data is lost) When writing 0b - No effect 1b - Clear the flag
18 NF	Noise Flag Indicates whether noise is detected in the received character of Data (DATA). The advanced sampling technique used in the receiver takes three samples in each of the received bits. If some of these samples disagree with the rest of the samples within any bit time in the frame, then noise is detected for that character. This field becomes 1 whenever the next character to be read from Data (DATA) is received with noise detected within the character.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - No noise detected 1b - Noise detected When writing 0b - No effect 1b - Clear the flag
17 FE	Framing Error Flag Indicates whether a framing error is detected. This field becomes 1 whenever the next character to be read from Data (DATA) is received with logic 0 detected where a stop bit was expected. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No framing error detected (this does not guarantee that the framing is correct) 1b - Framing error detected When writing 0b - No effect 1b - Clear the flag
16 PF	Parity Error Flag Indicates whether a parity error is detected. This field becomes 1 whenever the next character to be read from Data (DATA) is received when parity is enabled (CTRL[PE] is 1) and the parity bit in the received character does not agree with the expected parity value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No parity error detected 1b - Parity error detected When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 MA1F	Match 1 Flag Indicates whether the received data is equal to MATCH[MA1]. This field becomes 1 whenever the next character to be read from Data (DATA) matches the value of MATCH[MA1]. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not equal to MA1 1b - Equal to MA1 When writing 0b - No effect 1b - Clear the flag
14 MA2F	Match 2 Flag Indicates whether the received data is equal to MATCH[MA2]. This field becomes 1 whenever the next character to be read from Data (DATA) matches the value of MATCH[MA2]. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not equal to MA2 1b - Equal to MA2 When writing 0b - No effect 1b - Clear the flag
13-10 —	Reserved
9 TSF	Timeout Status Flag Indicates whether a field in Timeout Status (TOSR) is 1 and configured to generate an interrupt. 0b - Field is 0 1b - Field is 1
8 MSF	MODEM Status Flag Indicates whether a field in MODEM Status (MSR) is 1 and configured to generate an interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Field is 0 1b - Field is 1
7-2 —	Reserved
1 AME	Address Mark Enable Configures the location of the address mark when configured for MSB first transfers. This field has no effect when configured for LSB first and you must change the value of this field only when both the transmitter and receiver are disabled. If this field is 0, address mark in character is MSB. If this field is 1, the address mark is stored in Data (DATA) at MSB (or MSB-1 when the parity bit is enabled). In other words, the address mark in character is the last bit before the stop bit (or parity bit when enabled). 0b - Disable 1b - Enable
0 LBKFE	LIN Break Flag Enable Enables the LIN break flag to assert whenever a LIN break character is detected. Unlike STAT[LBKDE], this does not impact data being stored in the receive data buffer, but does cause STAT[LBKDIF] to become 1 whenever a LIN break is detected. Because a LIN break is longer than a normal character, the LIN break triggers a write to STAT[RDRF] with the data fields as 0 and STAT[FE] as 1. The character following the LIN break has DATA[LINBRK] as 1 to indicate that the previous character was a LIN break. You must change the value of this field only when both the transmitter and receiver are disabled. If this field is 1, the LIN break character is detected at a length of 11-bit times (if CTRL[M] is 0), 12 (if CTRL[M] is 1), or 13 (if BAUD[M10] is 1). 0b - Disable 1b - Enable

64.6.1.8 Control (CTRL)

Offset

Register	Offset
CTRL	18h

Function

Controls various optional features of the LPUART system.

You must write to the fields of this register only when both the transmitter and receiver are disabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R8T9	R9T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MA1IE	MA2IE	0	0	M7	IDLECFG			LOOPS	DOZEEN	RSRC	M	WAKE	ILT	PE	PT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 R8T9	Receive Bit 8 Transmit Bit 9 Contains R8 and T9 that correspond to different functions. R8 is the ninth data bit received after you configure LPUART for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading Data (DATA). T9 is the tenth data bit transmitted after you configure LPUART for 10-bit data formats. When writing 10-bit data, write T9 before writing to Data (DATA). If T9 does not need to change from its previous value, such as when it is used to generate address mark or parity, then you need not write to it each time you write to Data (DATA). **NOTE** R8 is a read-only bit and T9 is a write-only bit; the value read is different from the value written.
30 R9T8	Receive Bit 9 Transmit Bit 8 Contains R9 and T8 that correspond to different functions. R9 is the tenth data bit received after you configure LPUART for 10-bit data formats. When reading 10-bit data, read R9 before reading Data (DATA). T8 is the ninth data bit transmitted after you configure LPUART for 9-bit or 10-bit data formats. When writing 9-bit or 10-bit data, write T8 before writing to Data (DATA). If T8 does not need to change from its previous value, such as when it is used to generate address mark or parity, then you need not write to it each time you write to Data (DATA). **NOTE** R9 is a read-only field and T8 is a write-only field; the value read is different from the value written.
29 TXDIR	TXD Pin Direction in Single-Wire Mode Determines the direction of data at the TXD pin, in Single-Wire mode, when LPUART is configured for a single-wire half-duplex operation (CTRL[LOOPS] and CTRL[RSRC] are 1). When writing 0 to this field, the transmitter finishes transmitting the current character (if any) before the receiver starts receiving data from the TXD pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Input 1b - Output
28 TXINV	Transmit Data Inversion Specifies whether transmit data is inverted. Writing 1 to this field reverses the polarity of the transmitted data output. This action inverts the TXD output for all cases: data bits, start and stop bits, break, and idle. 0b - Not inverted 1b - Inverted
27 ORIE	Overrun Interrupt Enable Enables STAT[OR] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when OR interrupts are disabled. 0b - Disable 1b - Enable
26 NEIE	Noise Error Interrupt Enable Enables STAT[NF] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when NF interrupts are disabled. 0b - Disable 1b - Enable
25 FEIE	Framing Error Interrupt Enable Enables STAT[FE] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when FE interrupts are disabled. 0b - Disable 1b - Enable
24 PEIE	Parity Error Interrupt Enable Enables STAT[PF] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when PF interrupts are disabled. 0b - Disable 1b - Enable
23 TIE	Transmit Interrupt Enable Enables STAT[TDRE] to generate interrupt requests if STAT[TDRE] is 1. 0b - Disable 1b - Enable
22	Transmission Complete Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TCIE	Enables STAT[TC] to generate interrupt requests if STAT[TC] is 1. 0b - Disable 1b - Enable
21 RIE	Receiver Interrupt Enable Enables STAT[RDRF] to generate hardware interrupt requests if STAT[RDRF] is 1. 0b - Disable 1b - Enable
20 ILIE	Idle Line Interrupt Enable Enables hardware interrupts. This field enables STAT[IDLE] to generate interrupt requests. If this field is 0, hardware interrupts from STAT[IDLE] are disabled and polling is used, and if this field is 1, hardware interrupts are enabled when STAT[IDLE] is 1. 0b - Disable 1b - Enable
19 TE	Transmitter Enable Enables the LPUART transmitter. Using this field, you can also queue an idle preamble by first writing 0 and then writing 1 to this field. After this field becomes 0, the field reads 1 until the transmitter has completed the current character and the TXD pin is tristated. You can also queue a single idle character by writing to the transmit FIFO with DATA[FRETSC] and DATA[R9T9] = 1. 0b - Disable 1b - Enable
18 RE	Receiver Enable Enables the LPUART receiver. After you write 0 to this field, this field remains 1 until the receiver finishes receiving the current character (if any). 0b - Disable 1b - Enable
17 RWU	Receiver Wake-Up Control Specifies whether the LPUART receiver in standby is waiting for a wake-up condition. You can write 1 to this field to place the LPUART receiver in a Standby state. The field becomes 0 automatically when an RWU event occurs, that is, in case of an idle event when CTRL[WAKE] is 0 or an address match when CTRL[WAKE] is 1 and STAT[RWUID] is 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE You must write 1 to this field only when CTRL[WAKE] is 0 (wake-up on idle), if the channel is currently not idle. You can determine this by the value of STAT[RAF]. If the field is 1 to wake up an idle event and the channel is already idle, LPUART, possibly, discards the data. This is because the data must be received or a LIN break is detected after an Idle condition is detected before the IDLE flag is allowed to be reasserted. 0b - Normal receiver operation 1b - LPUART receiver in standby, waiting for a wake-up condition
16 SBK	Send Break Specifies whether queue break character(s) are to be sent. Writing 1 and then 0 to this field queues a break character in the transmit data stream. Additional break characters of 9 to 13 bits, or 12 to 15 bits if STAT[BRK13] is 1, and bit times of logic 0 are queued as long as this field is 1. Depending on the timing when this field is 1 and 0, relative to the character currently being transmitted, a second break character may be queued before you write 0 to this field. If the time taken to write 0 to this field is too long, for example, if the field does not become 0 by the end of the first break character, a second break character is sent. This is compared to queuing a break character through the transmit FIFO that guarantees only one break character is sent. You can also queue a single break character by writing to the transmit FIFO when DATA[FRETSC] is 1 and DATA[R9T9] is 0. 0b - Normal transmitter operation 1b - Queue break character(s) to be sent
15 MA1IE	Match 1 (MA1F) Interrupt Enable Enables the MA1F interrupt. 0b - Disable 1b - Enable
14 MA2IE	Match 2 (MA2F) Interrupt Enable Enables the MA2F interrupt. 0b - Disable 1b - Enable
13 —	Reserved
12 —	Reserved
11 M7	7-Bit Mode Select Specifies the data characters that the receiver and transmitter use.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	You must change the value of this field only after both the transmitter and receiver are disabled. 0b - 8-bit to 10-bit 1b - 7-bit
10-8 IDLECFG	Idle Configuration Configures the number of idle characters that must be received before you write 1 to STAT[IDLE]. 000b - 1 001b - 2 010b - 4 011b - 8 100b - 16 101b - 32 110b - 64 111b - 128
7 LOOPS	Loop Mode Select Selects Loop mode. After this field becomes 1, the RXD pin is disconnected from LPUART and the transmitter output is internally connected to the receiver input. The transmitter and receiver must be enabled to use the loop function. In Loop mode or Single-Wire mode, the transmitter outputs are internally connected to the receiver input (see CTRL[RSRC]). 0b - Normal operation: RXD and TXD use separate pins 1b - Loop mode or Single-Wire mode
6 DOZEEN	Doze Mode Enables LPUART in Doze mode. If this field is 1, LPUART remains active when not in Doze mode. 0b - Enable 1b - Disable
5 RSRC	Receiver Source Select Determines the source of the receiver shift register input if CTRL[LOOPS] is 1. This field has no effect unless CTRL[LOOPS] is 1. If this field is 0, internal Loopback mode is selected. LPUART does not use the RXD pin. Additionally, the CTS_B pin is not used and internally driven by the RTS_B output. If this field is 1, single-wire LPUART mode is selected where the TXD pin is connected to the transmitter output and receiver input.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Internal Loopback mode 1b - Single-wire mode
4 M	9-Bit Or 8-Bit Mode Select Specifies the data characters that the receiver and transmitter use. 0b - 8-bit 1b - 9-bit
3 WAKE	Receiver Wake-Up Method Select Determines which condition wakes up LPUART when CTRL[RWU] = 1 and BAUD[MATCFG] = 0 (this field must be 1 when BAUD[MATCFG] = 11): • Address mark in the bit preceding the stop bit (or bit preceding the parity bit when parity is enabled) of the received data character • An idle condition on the receive pin input signal If this field is 0, CTRL[RWU] is configured for idle line wake-up, and if this field is 1, CTRL[RWU] is configured with address mark wake-up. 0b - Idle 1b - Mark
2 ILT	Idle Line Type Select Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or the stop bit. If the count begins after the start bit, a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. **NOTE** In case you write 1 to this field, a logic 0 is automatically shifted after a received stop bit, therefore resetting the idle count. 0b - After the start bit 1b - After the stop bit
1 PE	Parity Enable Enables hardware parity generation and checking. If parity is enabled, the bit immediately before the stop bit is treated as the parity bit. 0b - Disable 1b - Enable
0 PT	Parity Type Selects the type of parity, even or odd, if parity is enabled (CTRL[PE] = 1):

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Odd parity means that the total number of logic 1 bits in the data character, including the parity bit, is odd. • Even parity means that the total number of 1s in the data character, including the parity bit, is even. 0b - Even parity 1b - Odd parity

64.6.1.9 Data (DATA)

Offset

Register	Offset
DATA	1Ch

Function

Supports 8-bit, 16-bit, or 32-bit writes, each type of write performing a separate function. An 8-bit write to DATA[7:0] pushes {CTRL[R8T9], CTRL[R9T8], DATA[7:0]} the transmit FIFO with TSC clear. A 16-bit or 32-bit write pushes the data written into the FIFO and does not update the value of CTRL[R8T9] or CTRL[R9T8].

Reads and writes of this register are also involved in the automatic flag clearing mechanisms for some of the LPUART status fields.

NOTE

Reads return the contents of the read-only receive FIFO and writes go to the write-only transmit FIFO, making this register work as a set of two separate registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NOISY	PARITYE	FRETSC	RXEMPT	IDLINE	LINBRK	R9T9	R8T8	R7T7	R6T6	R5T5	R4T4	R3T3	R2T2	R1T1	R0T0
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 NOISY	Noisy Data Received Indicates whether the current received dataword contained in DATA[R9:R0] is received with noise. 0b - Received without noise 1b - Received with noise
14 PARITYE	Parity Error Indicates whether the current received dataword contained in DATA[R9:R0] is received with a parity error. 0b - Received without a parity error 1b - Received with a parity error
13 FRETSC	Frame Error Transmit Special Character Specifies the way the dataword is received. For reads, this field indicates that the current received dataword contained in DATA[R9:R0] is received with a frame error. For writes, the field indicates that a break or idle character is to be transmitted instead of the contents in DATA[T9:T0]. T9 indicates a break character when it is 0 and indicates an idle character when it is 1. The contents of DATA[T8:T0] must be 0. 0b - Received without a frame error on reads or transmits a normal character on writes 1b - Received with a frame error on reads or transmits an idle or break character on writes
12 RXEMPT	Receive Buffer Empty Indicates whether the receive buffer contains valid data. This field becomes 1 when there is no data in the receive buffer. The field does not consider data in the receive shift register. 0b - Valid data 1b - Invalid data and empty
11 IDLINE	Idle Line Indicates whether the receiver line was idle before receiving the character in DATA[9:0]. It can be read as “1” on the first character when the receiver is first enabled. The difference between this field and STAT[IDLE] is that, STAT[IDLE] flag does not set on an idle line after the receiver is first enabled, it needs to receive a character before it can become set, whereas this field does not have this limitation and can be set on the first character received if an idle line is detected beforehand. 0b - Not idle 1b - Idle
10	LIN Break

Table continues on the next page...

Table continued from the previous page...

Field	Function
LINBRK	Indicates whether the receiver line detected a LIN break before receiving the character in DATA[9:0]. This field requires the value of STAT[LBKDIF] to be 1. If this field is 0, the LIN break detect circuitry is disabled. 0b - Not detected 1b - Detected
9 R9T9	Read receive FIFO bit 9 or write transmit FIFO bit 9
8 R8T8	Read receive FIFO bit 8 or write transmit FIFO bit 8
7 R7T7	Read receive FIFO bit 7 or write transmit FIFO bit 7
6 R6T6	Read receive FIFO bit 6 or write transmit FIFO bit 6
5 R5T5	Read receive FIFO bit 5 or write transmit FIFO bit 5
4 R4T4	Read receive FIFO bit 4 or write transmit FIFO bit 4
3 R3T3	Read receive FIFO bit 3 or write transmit FIFO bit 3
2 R2T2	Read receive FIFO bit 2 or write transmit FIFO bit 2
1 R1T1	Read receive FIFO bit 1 or write transmit FIFO bit 1
0 R0T0	Read receive FIFO bit 0 or write transmit FIFO bit 0

64.6.1.10 Match Address (MATCH)

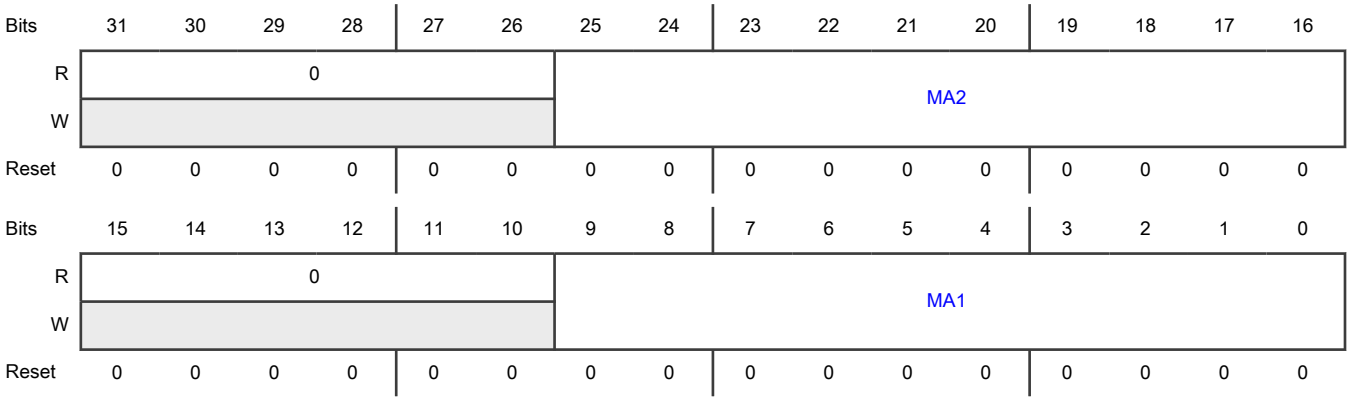
Offset

Register	Offset
MATCH	20h

Function

Provides addresses for address matching during the receiver operation.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 MA2	Match Address 2 Is compared to input data addresses when the most significant bit is 1 and the associated Baud Rate (BAUD) field is 1. If a match occurs, the data that follows is transferred to Data (DATA). If a match fails, the data that follows is discarded. You must write to MATCH[MA1] and MATCH[MA2] only when the associated Baud Rate (BAUD) field is 0.
15-10 —	Reserved
9-0 MA1	Match Address 1 Is compared to input data addresses when the most significant bit is 1 and the associated Baud Rate (BAUD) field is 1. If a match occurs, the data that follows is transferred to Data (DATA). If a match fails, the data that follows is discarded. You must write to MATCH[MA1] and MATCH[MA2] fields only when the associated field in Baud Rate (BAUD) is 0.

64.6.1.11 MODEM IrDA (MODIR)

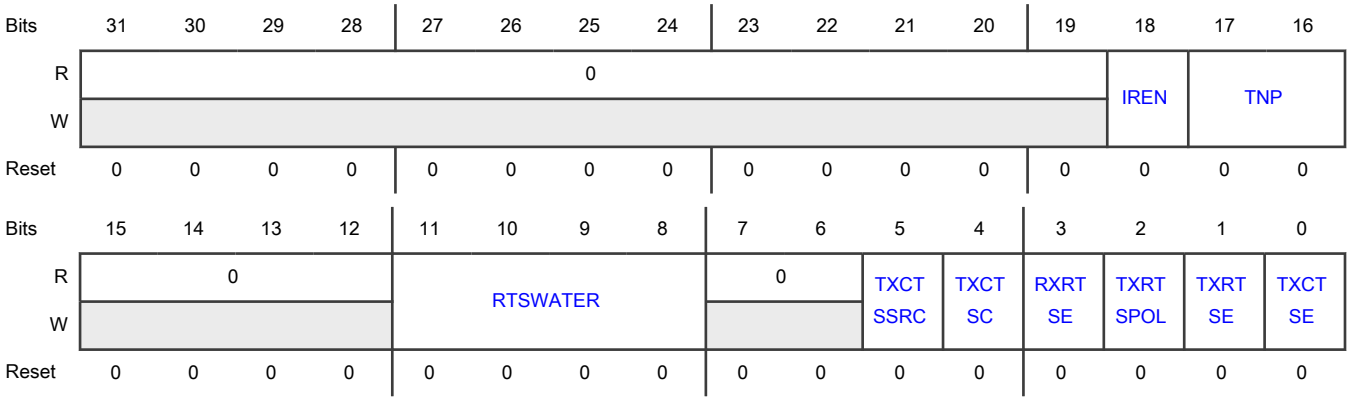
Offset

Register	Offset
MODIR	24h

Function

Controls options for setting the MODEM configuration.

Diagram



Fields

Field	Function
31-19 —	Reserved
18 IREN	IR Enable Enables IR modulation and demodulation. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - Disable 1b - Enable
17-16 TNP	Transmitter Narrow Pulse Specifies whether LPUART transmits a 1 ÷ OSR, 2 ÷ OSR, 3 ÷ OSR, or 4 ÷ OSR narrow pulse when the IR pulse is enabled. You must change the value of this field only when both the transmitter and receiver are disabled. The IR pulse width must be configured to less than half of the OSR. Common pulse widths are 3 ÷ 16, 1 ÷ 16, 1 ÷ 32, or 1 ÷ 4 of the bit length. You can configure these by selecting the appropriate OSR and pulse width. 00b - 1 ÷ OSR 01b - 2 ÷ OSR 10b - 3 ÷ OSR 11b - 4 ÷ OSR
15-12 —	Reserved
11-8	Receive RTS Configuration

Table continues on the next page...

Table continued from the previous page...

Field	Function
RTSWATER	Configures the assertion and negation of the receiver's RTS_B output. The receiver's RTS_B output negates when the number of empty words in the receive FIFO is greater or equal to the value of this field. If this field is 0, the RTS_B pin negates when the receive FIFO is full. For the purpose of receive RTS_B generation, the number of words in the receive FIFO updates when a start bit is detected. This supports additional latency between RTS_B negation and the external transmitter ceasing transmission. If both receive RTS_B and address or data matching is enabled, RTS_B could assert at the end of a character if there exists no match. You must change the value of this field only when the receiver is disabled.
7-6 —	Reserved
5 TXCTSSRC	Transmit CTS Source Configures the source of the CTS input. 0b - The CTS_B pin 1b - An internal connection to the receiver address match result
4 TXCTSC	Transmit CTS Configuration Configures whether the CTS state or input is checked or sampled at the start of each character or only when the transmitter is idle. 0b - Sampled at the start of each character 1b - Sampled when the transmitter is idle
3 RXRTSE	Receiver RTS Enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. You must change the value of this field only when the receiver is disabled. If this field is 0, the receiver has no effect on RTS. If this field is 1, RTS is deasserted if STAT[RDRF] is 1 or a start bit is detected that causes STAT[RDRF] to become 1. RTS is asserted if STAT[RDRF] is 0 and has not detected a start bit that causes STAT[RDRF] to become 1. **NOTE** Do not write 1 to both MODIR[RXRTSE] and MODIR[TXRTSE]. 0b - Disable 1b - Enable
2 TXRTSPOL	Transmitter RTS Polarity Controls the polarity of the transmitter RTS. This field does not affect the polarity of the receiver RTS that remains negated in the active-low state unless MODIR[TXRTSE] is 1. You must change the value of this field only when the transmitter is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Active low 1b - Active high
1 TXRTSE	Transmitter RTS Enable Controls the operation of RTS before and after a transmission. You must change the value of this field only when the transmitter is disabled. If this field is 0, the transmitter has no effect on RTS, and if this field is 1, a character is placed into an empty transmit shift register. RTS asserts 1-bit time before the start bit is transmitted and deasserts 1-bit time after all characters in the transmitter FIFO and shift register are completely sent, including the last stop bit. 0b - Disable 1b - Enable
0 TXCTSE	Transmitter CTS Enable Enables the operation of the transmitter. You can write 1 to this field irrespective of the states of MODIR[TXRTSE] and MODIR[RXRTSE]. If this field is 1, the transmitter checks the state of the CTS signal each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the TXD signal remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS, when a character is being sent, do not affect its transmission. 0b - Disable 1b - Enable

64.6.1.12 FIFO (FIFO)

Offset

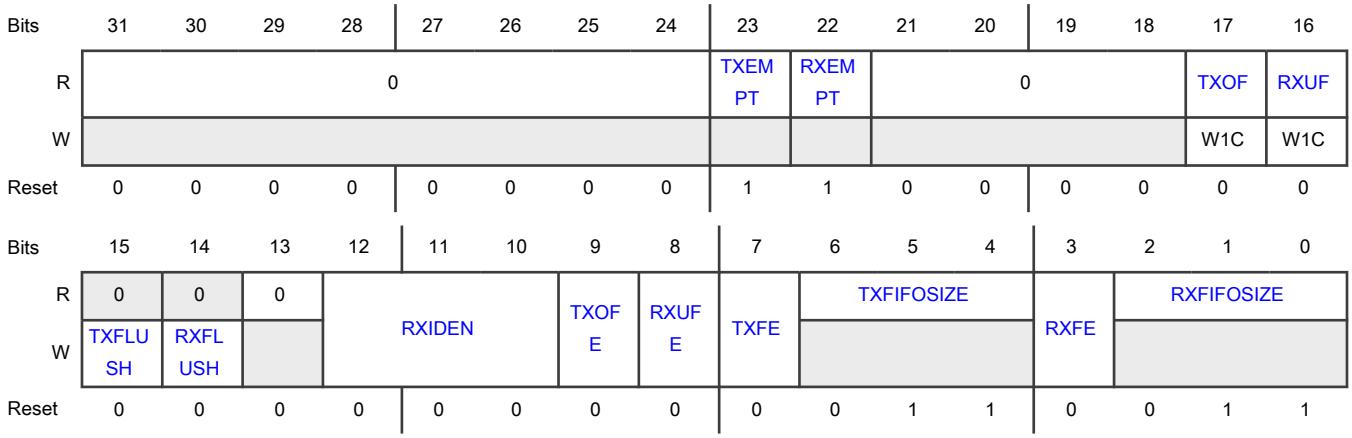
Register	Offset
FIFO	28h

Function

Provides you the ability to turn on and turn off the FIFO functionality.

This register also provides you the size of the FIFO that has been implemented. You can read this register at any time and must write to it only when CTRL[RE] and CTRL[TE] are 0 and the FIFO is empty.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 TXEMPT	Transmit FIFO Or Buffer Empty Indicates whether the transmit buffer is empty. This field becomes 1 when there is no data in the transmit FIFO or buffer. The field does not consider data in the transmit shift register. 0b - Not empty 1b - Empty
22 RXEMPT	Receive FIFO Or Buffer Empty Indicates whether the receive buffer is empty. This field becomes 1 when there is no data in the receive FIFO or buffer. The field does not consider data in the receive shift register. 0b - Not empty 1b - Empty
21-18 —	Reserved
17 TXOF	Transmitter FIFO Overflow Flag Indicates whether more data has been written to the transmit FIFO than it can hold. If this field is 0, no transmit FIFO overflow has occurred since the last time the field was cleared, and if this field is 1, at least one transmit FIFO overflow has occurred since the last time the field was cleared. This field becomes 1 regardless of the value of FIFO[TXOFE]. However, an interrupt is issued to the host only if FIFO[TXOFE] is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - No overflow 1b - Overflow When writing 0b - No effect 1b - Clear the flag
16 RXUF	Receiver FIFO Underflow Flag Indicates whether more data has been read from the receive FIFO than was present. If this field is 0, no receive FIFO underflow has occurred since the last time the field was cleared, and if this field is 1, at least one receive FIFO underflow has occurred since the last time the field was cleared. This field becomes 1 regardless of the value of FIFO[RXUFE]. However, an interrupt is issued to the host only if FIFO[RXUFE] is 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No underflow 1b - Underflow When writing 0b - No effect 1b - Clear the flag
15 TXFLUSH	Transmit FIFO Flush Causes all data that is stored in the transmit FIFO to be flushed. If you write 0 to this field, no flush operation occurs, and if you write 1 to this field, all data in the transmit FIFO or buffer clears out. This does not affect data in the transmit shift register. 0b - No effect 1b - All data flushed out
14 RXFLUSH	Receive FIFO Flush Causes all data that is stored in the receive FIFO to be flushed. If you write 0 to this field, no flush operation occurs, and if you write 1 to this field, all data in the receive FIFO or buffer clears out.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This does not affect data in the receive shift register. 0b - No effect 1b - All data flushed out
13 —	Reserved
12-10 RXIDEN	Receiver Idle Empty Enable Enables STAT[RDRF] to become 1 when the receiver is idle for a number of idle characters and the FIFO is not empty. This feature is not supported when the receiver extended idle time is enabled. 000b - Disable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle 001b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for one character 010b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for two characters 011b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for four characters 100b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for eight characters 101b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 16 characters 110b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 32 characters 111b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 64 characters
9 TXOFE	Transmit FIFO Overflow Interrupt Enable Enables FIFO[TXOF] to generate an interrupt to the host. 0b - Disable 1b - Enable
8 RXUFE	Receive FIFO Underflow Interrupt Enable Enables FIFO[RXUF] to generate an interrupt to the host. 0b - Disable 1b - Enable
7 TXFE	Transmit FIFO Enable Enables the transmit FIFO. If this field is 0, the transmit buffer operates as a FIFO of depth equal to 1 dataword, regardless of the value in FIFO[TXFIFOSIZE]. Both CTRL[TE] and CTRL[RE] must be 0 before you change the value of this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 1, the built-in FIFO structure for the transmit buffer is enabled. FIFO[TXFIFOSIZE] indicates the size of the FIFO structure. 0b - Disable 1b - Enable
6-4 TXFIFOSIZE	Transmit FIFO Buffer Depth Indicates the maximum number of transmit datawords (transmit FIFO buffer depth) that can be stored in the transmit buffer. 000b - 1 001b - 4 010b - 8 011b - 16 100b - 32 101b - 64 110b - 128 111b - 256
3 RXFE	Receive FIFO Enable Enables the receive FIFO. If this field is 0, the receive buffer operates as a FIFO of depth equal to 1 dataword, regardless of the value in FIFO[RXFIFOSIZE]. Both CTRL[RE] and CTRL[TE] must be 0 before you change the value of this field. If this field is 1, the built-in FIFO structure for the receive buffer is enabled. FIFO[RXFIFOSIZE] indicates the size of the FIFO structure. 0b - Disable 1b - Enable
2-0 RXFIFOSIZE	Receive FIFO Buffer Depth Indicates the maximum number of receive datawords (receive FIFO buffer depth) that can be stored in the receive buffer before an overrun occurs. 000b - 1 001b - 4 010b - 8 011b - 16 100b - 32 101b - 64 110b - 128 111b - 256

64.6.1.13 Watermark (WATER)

Offset

Register	Offset
WATER	2Ch

Function

Provides the ability to set a programmable threshold for notification, or sets the programmable thresholds to indicate that transmit data can be written or receive data can be read.

You may read this register at any time but must write to it only when CTRL[TE] is 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				RXCOUNT				0				RXWATER			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TXCOUNT				0				TXWATER			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 RXCOUNT	Receive Counter Indicates the number of datawords in the receive FIFO or buffer. If a dataword is being received in the receive shift register, it is not included in the count. This value may be used in conjunction with FIFO[RXFIFOSIZE] to calculate the room left in the receive FIFO or buffer.
23-20 —	Reserved
19-16 RXWATER	Receive Watermark Generates an interrupt or a DMA request if the number of datawords in the receive FIFO or buffer is greater than the value of this field. For proper operation, the value of this field must be less than the size of the receive FIFO or buffer, as indicated by FIFO[RXFIFOSIZE] and FIFO[RXFE].

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-13 —	Reserved
12-8 TXCOUNT	Transmit Counter Indicates the number of datawords in the transmit FIFO or buffer. If a dataword is being transmitted to the transmit shift register, it is not included in the count. This value may be used in conjunction with the value of FIFO[TXFIFOSIZE] to calculate the room left in the transmit FIFO or buffer.
7-4 —	Reserved
3-0 TXWATER	Transmit Watermark Generates an interrupt or a DMA request when the number of datawords in the transmit FIFO or buffer is equal to or less than the value of this field. For proper operation, the value of this field must be less than the size of the transmit buffer or FIFO, as indicated by FIFO[TXFIFOSIZE] and FIFO[TXFE].

64.6.1.14 Data Read-Only (DATARO)

Offset

Register	Offset
DATARO	30h

Function
Indicates the first entry in the receive FIFO, but does not pull data from the FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Receive Data Indicates the first entry from the receive FIFO. This register has the same functionality as that of Data (DATA).

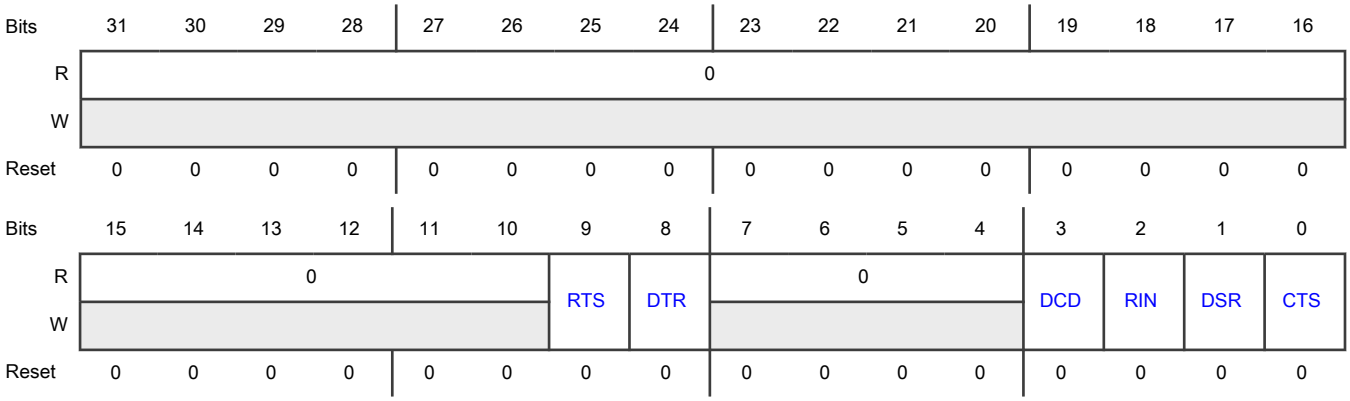
64.6.1.15 MODEM Control (MCR)

Offset

Register	Offset
MCR	40h

Function
Controls the operation of the MODEM pins.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RTS	Request To Send Configures the default state of the RTS_B pin when the function is disabled. 0b - Logic one

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Logic zero
8 DTR	Data Terminal Ready Configures the default state of the DTR_B pin. 0b - Logic one 1b - Logic zero
7-4 —	Reserved
3 DCD	Data Carrier Detect Configures the interrupt, DCD_B, for change of state of the DCD_B pin. 0b - Disable interrupt 1b - Enable interrupt
2 RIN	Ring Indicator Configures the interrupt, RIN_B, for change of state on the RIN_B pin. 0b - Disable interrupt 1b - Enable interrupt
1 DSR	Data Set Ready Configures the interrupt, DSR_B, for change of state on the DSR_B pin. 0b - Disable interrupt 1b - Enable interrupt
0 CTS	Clear To Send Configures the interrupt, CTS_B, for change of state on the CTS_B pin. 0b - Disable interrupt 1b - Enable interrupt

64.6.1.16 MODEM Status (MSR)

Offset

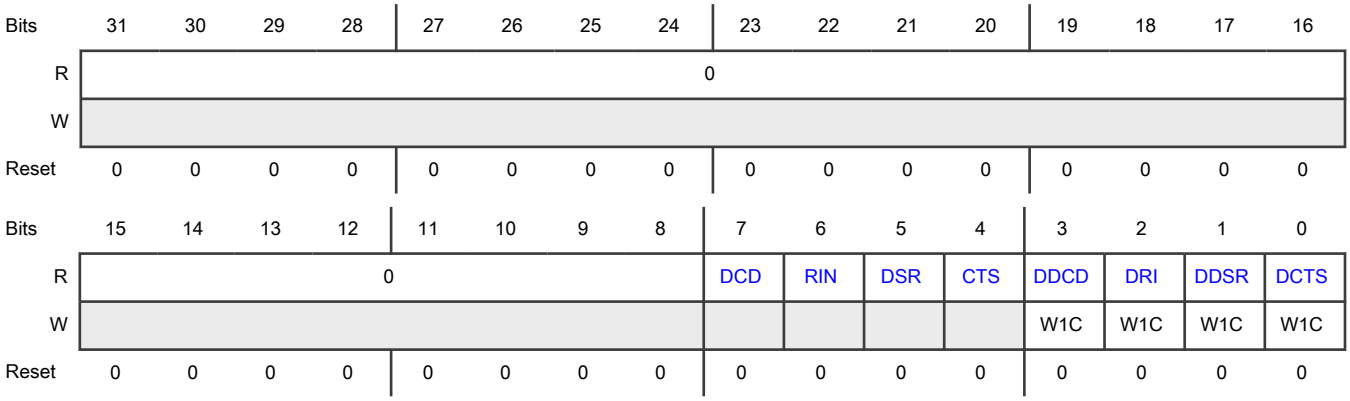
Register	Offset
MSR	44h

Function
Indicates the status of the MODEM pins.

NOTE

You must appropriately configure the PAD connecting to DCD_B, RIN_B, DSR_B, and CTS_B inputs to get appropriate reset values for the MSR register fields.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 DCD	Data Carrier Detect Indicates the state of the DCD_B pin. 0b - Logic one 1b - Logic zero
6 RIN	Ring Indicator Indicates the state of the RIN_B pin. 0b - Logic one 1b - Logic zero
5 DSR	Data Set Ready Indicates the state of the DSR_B pin. 0b - Logic one 1b - Logic zero
4 CTS	Clear To Send Indicates the state of the CTS_B pin. 0b - Logic one 1b - Logic zero

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 DDCD	Delta Data Carrier Detect Indicates whether the DCD_B pin changed state since the last time this field was 0. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change state 1b - Changed state When writing 0b - No effect 1b - Clear the flag
2 DRI	Delta Ring Indicator Indicates whether the RIN_B pin changed state since the last time this field was 0. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change state 1b - Changed state When writing 0b - No effect 1b - Clear the flag
1 DDSR	Delta Data Set Ready Indicates whether the DSR_B pin changed state since the last time this field was 0. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not change state 1b - Changed state When writing 0b - No effect 1b - Clear the flag
0 DCTS	Delta Clear To Send Indicates whether the CTS_B pin changed state since the last time this field was 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Did not change state 1b - Changed state When writing 0b - No effect 1b - Clear the flag

64.6.1.17 Receiver Extended Idle (REIR)

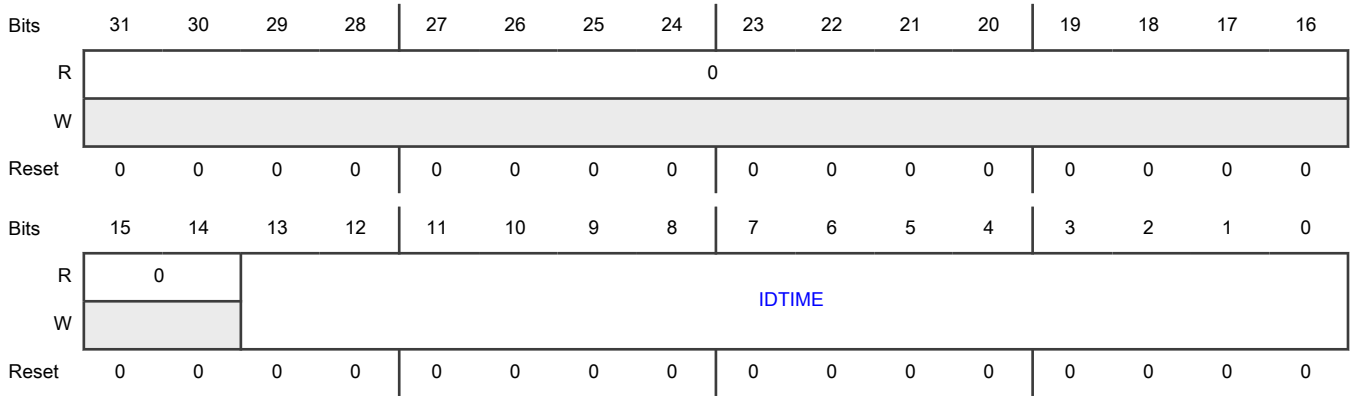
Offset

Register	Offset
REIR	48h

Function

Configures the receiver extended idle functionality. You must not change this value when the receiver is enabled.

Diagram



Fields

Field	Function
31-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-0 IDTIME	Idle Time Configures the idle length in number of bits (baud rate) since the end of the last stop bit. This affects the behavior of the idle wake-up, STAT[IDLE], DATA[IDLINE], and STAT[RAF]. The minimum supported extended idle time is equal to one idle character. The extended idle feature is disabled when this field is 0.

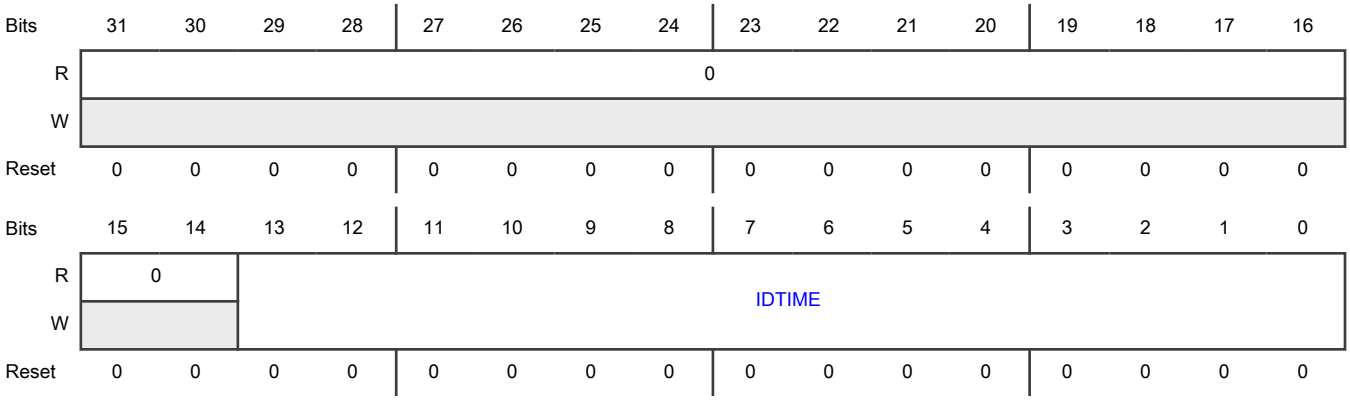
64.6.1.18 Transmitter Extended Idle (TEIR)

Offset

Register	Offset
TEIR	4Ch

Function
Configures the transmitter extended idle functionality. You must not change this value when the transmitter is enabled.

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 IDTIME	Idle Time Configures the transmitter idle time in number of bits (baud rate) whenever an idle character is queued through the transmit FIFO. An idle character is not automatically queued whenever the transmitter is enabled. The minimum supported extended idle time equals one idle character. The extended idle feature is disabled when this field is 0.

64.6.1.19 Half Duplex Control (HDCR)

Offset

Register	Offset
HDCR	50h

Function

Provides control for half-duplex-related operations.

You can use this register instead of CTRL[LOOPS], CTRL[RSRC], and CTRL[TXDIR] functions, although you can use CTRL[LOOPS] to loop-back the transmitter outputs to the receiver.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RTSEXT								0				RXMSK	RXWRMSK	RXSEL	TXSTALL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 RTSEXT	RTS Extended Specifies RTS extension. The transmit RTS_B remains asserted for (RTSEXT + 1) bit times after the end of the last stop bit. This applies even when the transmitter's RTS_B output is disabled and is only used internally to mask the receiver.
7-4 —	Reserved
3 RXMSK	Receive Mask Specifies whether the transmitter RTS_B masks the receive data pin. When enabled, the transmitter RTS_B masks the receive data pin. 0b - Do not mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Mask
2 RXWRMSK	Receive FIFO Write Mask Specifies whether the transmitter RTS_B masks writes to the receive FIFO. When enabled, the transmitter RTS_B masks receive FIFO writes, but the idle flag is not affected. 0b - Do not mask 1b - Mask
1 RXSEL	Receive Select Specifies the receive data pin. When enabled, the receive data is sourced from the TXD pin. 0b - RXD 1b - TXD
0 TXSTALL	Transmit Stall Specifies whether the transmitter becomes busy when the receiver is active. When enabled, the transmitter does not become busy or asserts transmitter RTS_B when the receiver is active (STAT[RAF] is 1). 0b - No effect 1b - Does not become busy

64.6.1.20 Timeout Control (TOCR)

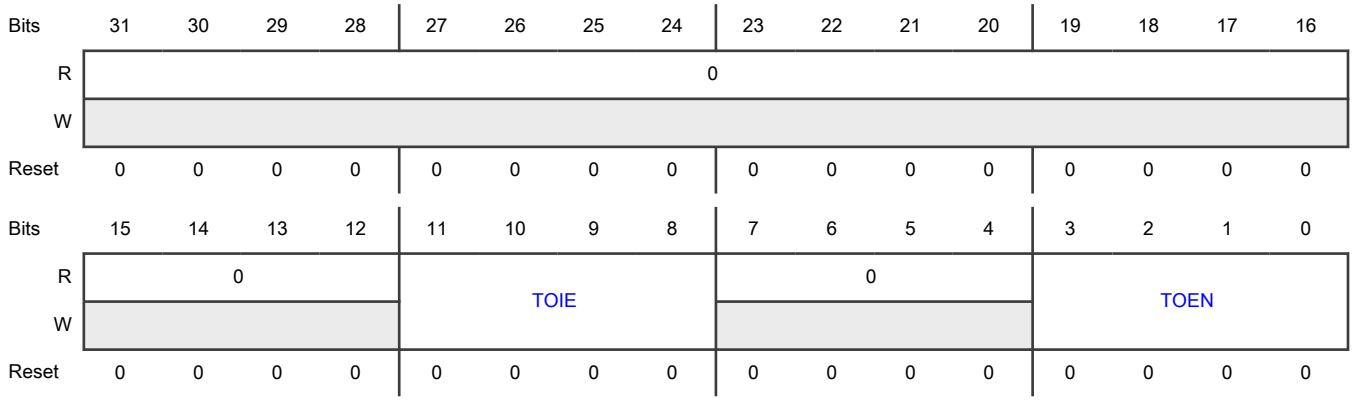
Offset

Register	Offset
TOCR	58h

Function

Configures the behavior of the timeout logic. Timeouts 0 and 1 are used to monitor the receiver and timeouts 2 and 3 are used to monitor the transmitter.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-8 TOIE	Timeout Interrupt Enable Enables the corresponding timeout flag to generate an interrupt.
7-4 —	Reserved
3-0 TOEN	Timeout Enable Enables the corresponding timeout counter.

64.6.1.21 Timeout Status (TOSR)

Offset

Register	Offset
TOSR	5Ch

Function

Indicates the status of the timeout logic. Timeouts 0 and 1 are used to monitor the receiver and timeouts 2 and 3 are used to monitor the transmitter.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TOF				0				TOZ			
W					W1C											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Fields

Field	Function
31-12 —	Reserved
11-8 TOF	Timeout Flag Indicates whether the corresponding timeout occurred. The timeout counter is disabled when this field is 1. **NOTE** This field behaves differently for register reads and writes. When reading 0000b - Not occurred 0001b - Occurred When writing 0000b - No effect 0001b - Clear the flag
7-4 —	Reserved
3-0 TOZ	Timeout Zero Indicates whether the corresponding timeout counter equals 0.

64.6.1.22 Timeout N (TIMEOUT0 - TIMEOUT3)

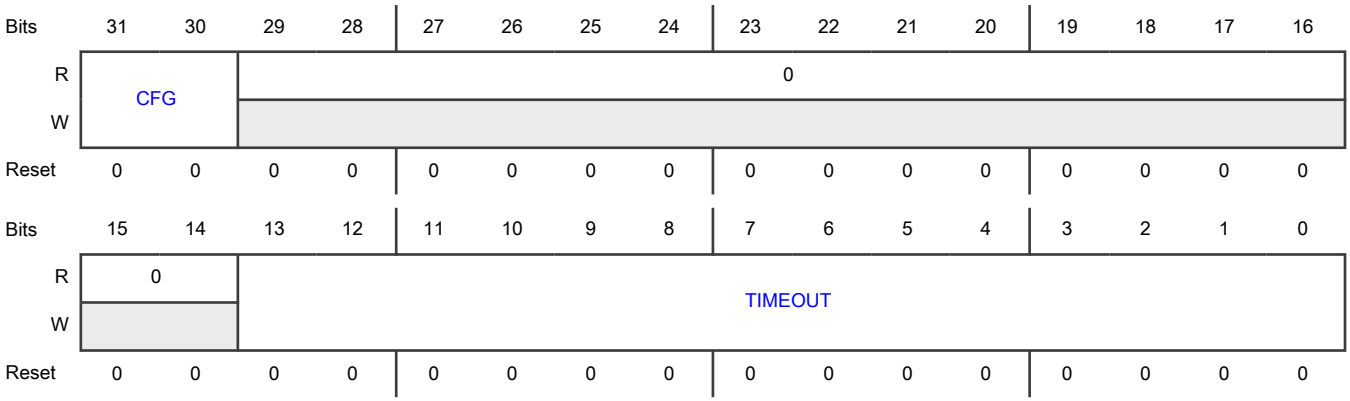
Offset

Register	Offset
TIMEOUT0	60h
TIMEOUT1	64h
TIMEOUT2	68h
TIMEOUT3	6Ch

Function

Configures the corresponding timeout counter and status field. Timeouts 0 and 1 are used to monitor the receiver and timeouts 2 and 3 are used to monitor the transmitter. You must write to this register only when the corresponding timeout is disabled or when the value of the corresponding timeout field is 1.

Diagram



Fields

Field	Function
31-30 CFG	Idle Configuration Configures the behavior of TIMEOUT[TIMEOUT]. 00b - Becomes 1 after timeout characters are received 01b - Becomes 1 when idle for timeout bit clocks 10b - Becomes 1 when idle for timeout bit clocks following the next character 11b - Becomes 1 when idle for at least timeout bit clocks, but a new character is detected before the extended idle timeout is reached
29-14 —	Reserved
13-0 TIMEOUT	Timeout Value Configures the timeout value.

64.6.1.23 Transmit Command Burst (TCBR0 - TCBR127)

Offset

For a = 0 to 127:

Register	Offset
TCBRa	200h + (a × 4h)

Function

Acts as an alias of Data (DATA), designed to support incrementing burst transfers to the transmit FIFO by a DMA controller, using aligned 8-bit, 16-bit, or 32-bit writes. The size of this register is 512 bytes:

- An aligned 32-bit write in this region pushes one entry into the transmit FIFO.
- An aligned 16-bit write in this region to TCBR*x*[15:0] pushes one entry into the transmit FIFO.
- An 8-bit write in this region to TCBR*x*[7:0] updates DATA[7:0], but does not push the data into the transmit FIFO.
- An 8-bit write in this region to TCBR*x*[15:8] pushes the data written to DATA[15:8] plus the previously written DATA[7:0] into the transmit FIFO.
- An 8-bit or 16-bit write in this region to TXBR*x*[31:16] is ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Data Enables writing data to Data (DATA).

64.6.1.24 Transmit Data Burst (TDBR0 - TDBR255)

Offset

For a = 0 to 255:

Register	Offset
TDBRa	400h + (a × 4h)

Function

Acts as an alias of Data (DATA), designed to support incrementing burst transfers to the transmit FIFO by a DMA controller using 8-bit, 16-bit, or 32-bit writes. The size of this register is 1024 bytes:

- An aligned 32-bit write in this region pushes four DATA byte entries into the transmit FIFO (DATA0 byte first). The register access is extended by three wait states.
- An aligned 16-bit write in this region to either half of a 32-bit word pushes two DATA byte entries into the transmit FIFO (DATA0 or DATA2 first). The register access is extended by one wait state.
- An 8-bit write in this region pushes one DATA byte entry into the transmit FIFO.

Byte writes to Data (DATA) use the contents of CTRL[R9T8] for the ninth data bit, CTRL[R8T9] for the tenth data bit, and zero extend DATA[FRETSC].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	DATA3								DATA2							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	DATA1								DATA0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 DATA3	Data3 Enables writing of data to Data (DATA).
23-16 DATA2	Data2 Enables writing of data to Data (DATA).
15-8 DATA1	Data1 Enables writing of data to Data (DATA).

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 DATA0	Data0 Enables writing of data to Data (DATA).

Chapter 65
Improved Inter-Integrated Circuit (I3C)

65.1 Chip-specific I3C Information

Table 1043. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

NOTE

HKEEP nets are connected to weak pullups in the SCL and SDA pads.

65.2 I3C Target Reset functionality

A Target Reset pattern detector is connected with of I3C module in this device. This functionality is active when related I3C module is configured as I3C target, that is, MCONFIG. MSTENA[0] = 0 . I3C Controller can set Reset action via defining byte of Broadcast or Direct write RSTACT CCC and read the same via Direct read RSTACT CCC. With default action set, on detection of any Target Reset Pattern a flag SSTATUS[SLVRST] is set and an interrupt is generated to system if enabled in SINTSET [SLVRST]. System can decide how to use this interrupt for any purpose, viz Reset to this module or any peripheral, or system etc. After taking action, the host clears the interrupt. Supported values of defining bytes in RSTACT CCC are 0x00 and 0x01.

65.3 Overview

I3C is a communications processor that improves upon the use and power of I2C, and provides an alternative to SPI for mid-speed applications.

The I3C bus protocol supports:

- In-band interrupts (IBI): these interrupts move from target to controller without extra wires, and the controller knows which target sent the interrupt
- Common command codes (CCC)
- Dynamic addressing
- Multi-controller and multi-drop

- Hot-Join (HJ)
- I2C compatibility

I3C supports all required and many optional features of the MIPI Alliance Specification for I3C, v1.0, except for ternary data rates (HDR-TSP and HDR-TSL). See Features for more information.

65.3.1 Block diagram

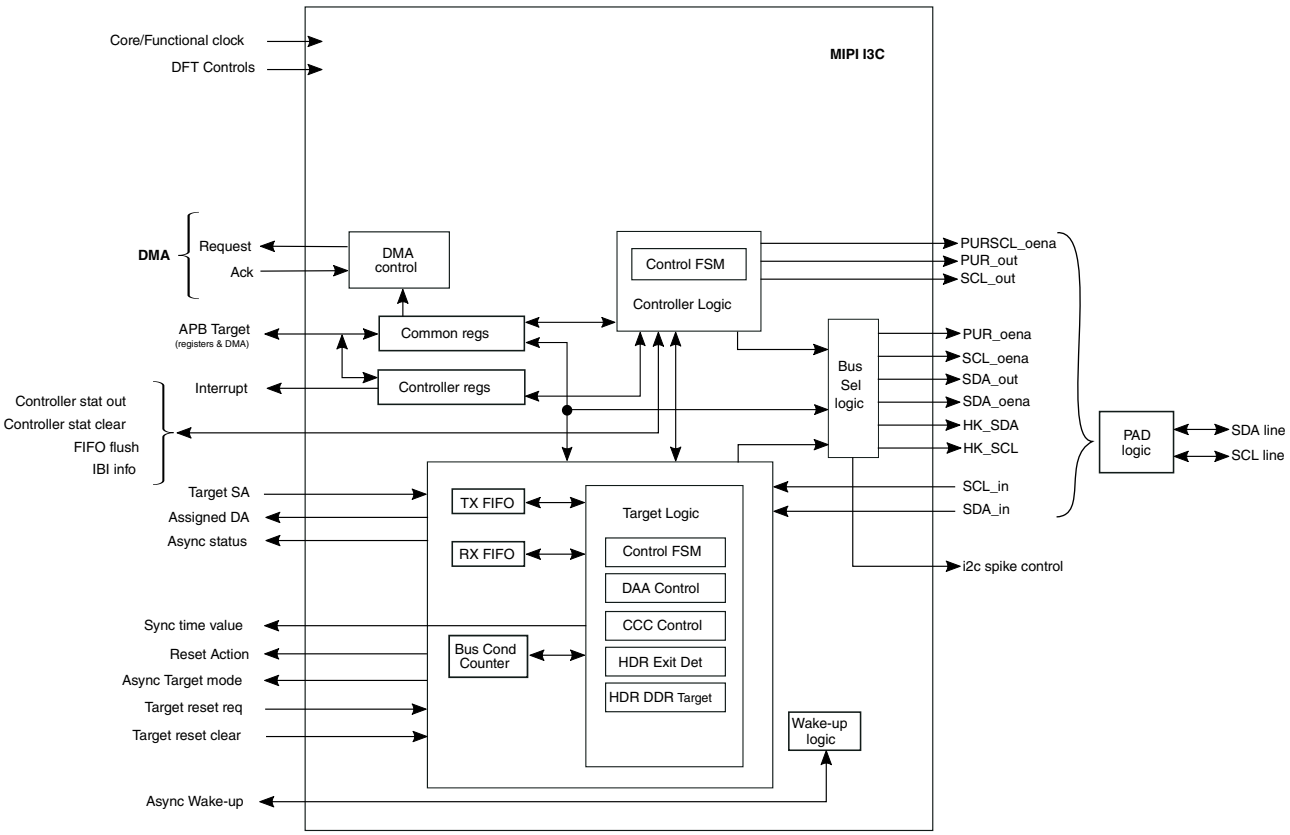


Figure 479. Block diagram

65.3.2 Features

- Two-wire multidrop bus that is capable of up to 12.5 MHz clock speeds with up to 11 devices:
  - Uses standard pads with 4 mA drive.
  - Dynamically assigns target addresses, and targets do not require static addresses (SAs). However, targets can have an I2C static address assigned at startup, so the target can operate on an I2C bus. By default, I3C supports 7-bit I2C-style addresses.
  - Supports extended I2C 10-bit addressing through Map Feature Control 1 (SMAPCTRL1).
  - Allows targets to use the inbound SCL clock as the peripheral clock (instead of the clock from the controller) so that devices can have slow or inaccurate clocks internally.
  - Allows simple targets, such as temperature sensors, to have no internal clock.
  - I3C controller supports handoff from Open-Drain to Push-Pull mode for ACK to data transfer.
  - Generally, the controller terminates the read, but for I3C, the target can also end the read.
- IBIs that allow targets to send notifications to a controller:

  - Can be equivalent to a separate GPIO, but can also be directly data-bearing.
  - Can be prioritized. When multiple targets send interrupts to a controller at the same time, the order is resolved. Dynamic addresses (DAs) establish the priority of the targets, so the controller controls the priority of the targets.

    Targets with lower-value DAs are higher-priority level IBIs.
  - Can start interrupts even when the controller is not active on the bus. A free-running clock is not required, but starting an interrupt requires a Bus Available condition.
  - Can resolve an initial event via a time-stamping option, not requiring an interrupt.
- Built-in commands for applications created in software or firmware by using the register interface maintained in a separate space. These commands do not collide with normal controller-to-target messages and:
  - Control bus behavior, modes and states, low-power state, inquiries, and more.
  - Have an extra room for new built-in commands for other groups.
- Organized forms of multicontroller modes that are secondary controllers using clean handoffs between different controllers.
- Hot-Join onto I3C bus allows devices to connect to the bus later than when the bus starts. This:
  - Enables a device or module to access the I3C bus after power up or when physically inserted onto the I3C bus.
  - Provides a clean method for notification when new devices or modules access the I3C bus.
- I3C can use both I2C and I3C buses:
  - I3C supports specific legacy I2C devices on the bus.
  - I3C target devices can operate on I2C buses.
  - A reset output is available for I2C mode SW reset for application usage. Asserted when the module in I2C Target and gets a I2C software General command from Controller.
- Higher data rate modes are available:
  - I3C has a high data rate: double data rate (HDR-DDR) mode, which is double the data rate of SDR .
  - Only the controller and the specific targets, which are capable and configured to work with HDR-DDR, support the higher data rate (the other targets can ignore it).

I3C supports most of the I3C features (see Target Capabilities (SCAPABILITIES) and Target Capabilities 2 (SCAPABILITIES2)), except for the ternary data rates (HDR-TSP and HDR-TSL), HDR-BT modes, and peer-to-peer messaging.

65.4 Functional description

65.4.1 Operating modes

This section describes all functional operation modes of the I3C module.

65.4.1.1 Target and controller roles for I3C

The I3C protocol defines these roles for devices on the I3C bus:

- Main controller: initially configures the I3C bus and serves as the first active controller
- Secondary controller: accepts the controller role from any active controller to become the new active controller (the secondary controller may then pass the controller role along: either back to the previous active controller, or on to any other controller-capable device)
- Target: responds to commands from any I3C controller
- I2C target: responds to compatible commands from any I3C controller

The I3C peripheral contains both controller and target components and can be configured to be either controller or target, or as target with secondary controller capability. However, if I3C is chosen to be a controller, then the I3C peripheral supports Target mode to facilitate handoffs to multiple controllers.

In general, any I3C controller can be a controller or a target because a controller becomes a target when giving over control to another controller. The only exceptions to this rule occur when using point-to-point communication or when using a controller that never shares controllership.

65.4.1.1.1 Controller requirements

Controller mode uses software that supports the requirements of the controller (including as a secondary controller):

- Managing the enter dynamic address assignment (ENTDAA) assigning dynamic addresses (DAs) to each target. The I3C peripheral supports this process but requires you to make choices.
- Driving the serial data (SDA) signal in both Open-Drain and Push-Pull modes. After START command, SDA is in Open-Drain mode and after Repeated START command, it is in Push-Pull mode:
  - SDA is subject to arbitration because both controller and target can drive the SDA line (arbitration is also useful for handoff from controller to target).
  - The ninth bit of SDA controller write data is an odd parity bit.
- Building a table of targets and their capabilities to control which actions and commands may be sent to the targets.
- Managing requests such as IBI and controller requests (handoff).
- Adjusting clock speed or write-to-read timing to match the limitations of the target. This adjustment can be done in hardware using dividers and uneven duty cycles, but you must decide how.
- Adjusting the maximum data length.
- Being a target after the controller role handoff to another controller-capable device.

The controller needs an accurate clock capable of running at a frequency that is the multiple of a frequency between 11 MHz and 12.5 MHz. Following are the possible clock frequencies:

- 24 MHz (multiple of 12 MHz)

If required to use a lower I3C SCL value, for example as small as 5 MHz (although the lower value does not support a mixed bus system):

- The controller clock can be a multiplier of that value.
- A lower SCL can also be achieved using a higher PPBAUD value.

65.4.1.1.2 I3C target acts like I2C target on I3C buses

If the target is assigned an I2C static address, the I3C target acts like an I2C device when it first powers on. If the I3C target is placed on an I2C bus with an I2C controller, then the I3C target stays in I2C mode and operates normally. The software is aware that the I3C target is in I2C mode because:

- There has not been an interrupt (see SSTATUS[DACHG]) indicating that a dynamic address was assigned.

For full I2C support of Fast mode (Fm) and Fast-mode Plus (Fm+), the pads must support a 50-ns spike filter. This filter must be turned off when the I3C 7Eh broadcast address is received (indicating an I3C controller). You can turn off the spike filters via hardware using the raw net indicating that the address was received, or via software. I3C does not need a 50-ns spike filter to operate on an I2C bus. Therefore, the spike filter is not a requirement for the I3C peripheral.

Depending on the configuration, the module supports most I2C features, except for clock stretching. Supported features include an extended 10-bit address, DeviceID, and software reset. See Target Capabilities (SCAPABILITIES) and Target Capabilities 2 (SCAPABILITIES2) for more information.

65.4.1.1.3 How a target rejoins the I3C bus

When a target tries to rejoin the I3C bus following a power-up or hard reset, the target needs a new dynamic address (DA). The target can rejoin the I3C bus in these ways:

- If the DA is lost, Hot-Join is used.
- If the DA is retained in the peripheral (for example, with state retention flip-flops), SCONFIG[OFFLINE] is used.

When SCONFIG[SLVENA] is 1, SCONFIG[OFFLINE] may become 1. This setting causes the peripheral to rejoin the bus safely. It does so by ensuring that the I3C bus is not in HDR mode, using the same approach as TE0 or TE1 exit. The peripheral waits for the HDR exit pattern from the controller, or for 60 µs of SCL and SDA lines not changing, whichever occurs first.

After using SCONFIG[OFFLINE], the I3C peripheral cannot safely use IBI until SSTATUS[STOP] becomes 1. This status ensures that the next START is safe to use for IBI.

If the application has to perform an IBI, it must wait for SSTATUS[STOP] to become 1, or for SCL and SDA lines to remain high for 200 µs. You can perform this process using the SSTATUS and SINTSET controls:

- If Target Status (SSTATUS) indicates that the bus is not busy and the peripheral interrupts on START or STOP, use a timer to measure 200 µs. If the timer finishes with no START or STOP, it is safe to use IBI.
- If a START causes an interrupt, then the timer must be turned off and the application must wait for a STOP.
- If a STOP causes an interrupt, it is safe to use IBI.

65.4.1.2 Using the I3C controller for I2C and I3C

The I3C controller operates with the following set of built-in capabilities with the application flow:

- Built-in enter dynamic address assignment (ENTDAA) mechanism to simplify the assignment of dynamic addresses to targets. This feature is used when set dynamic address from static address (SETDASA) is not available.

  NOTE

  When SETDASA CCC is available, use SETDASA CCC before using ENTDAA CCC; all targets without assigned dynamic addresses respond to ENTDAA CCC.

- Request for START + address with IBI support, including both I2C and I3C modes.
- SDR write flow via FIFO, with automatic parity in I3C and ACK or NACK detection in I2C.
- SDR read flow via the FIFO, with an automatic NACK generator for I2C, and optional use of a terminate generator for I3C.
- Request for Repeated START + address or STOP when finished with the previous request, including both I2C and I3C.
- Auto-IBI mode, which responds immediately to a target-initiated IBI and can be used when in Sleep or Deep Sleep mode.
- Special message mode for SDR and DDR to simplify DMA use.
- Automated SCL, SDA, pullup, and High-Keeper controls.
- I2C and I3C frequency and duty cycle configurations from functional clock.

  NOTE

  You must configure Controller Configuration (MCONFIG) and Controller In-band Interrupt Registry and Rules (MIBIRULES) before using the controller.

65.4.1.3 Protocol modes and states

The following modes are activated in I3C:

- I2C mode is the default mode on startup:
  - If no I2C static address is used, this mode has no effect other than to track frames looking for I3C transitional frames.

- If an I2C static address is used, the device can interact with the I2C bus controller using the address.

- When in I2C mode, I3C Transitory Frame mode occurs whenever the I3C broadcast address is received (7Eh). In particular:
  - This mode occurs when 7Eh is broadcast followed by SETDASA from the controller. If there is a static address match or 01h, the target is assigned a dynamic address and it enters I3C SDR mode.
  - This mode occurs when 7Eh is broadcast followed by ENTDAA from the controller. If the target can send a 48-bit provisioned ID (48b ID), a dynamic address is assigned, and the target enters I3C SDR mode.
  - If a Hot-Join arbitrated event occurs (sending 02h when the controller generates another address such as 7Eh), there is a conceptual Hot-Join mode. ENTDAA or SETDASA sets a dynamic address to resolve the event, and the target enters I3C SDR mode.

NOTE

An I3C target can operate as a normal I2C target only when it has a static address. Otherwise, it matches nothing in I2C and waits for the aforementioned events.

- I3C SDR mode is the standard mode after I3C activates, which is defined by a dynamic address being assigned. This mode is the resting mode of I3C, and all devices are normally in SDR mode:
  - RSTDAA CCC causes an exit from SDR mode and a return to I2C mode. This transition is not normally used.

    RSTDAA can be issued to ensure all I3C targets participate in an upcoming dynamic address assignment process, before the I3C controller starts assigning the target address.
  - SETNEWDA does not affect the SDR mode; it only changes the dynamic address of an I3C device that already had a dynamic address assigned.
  - When in SDR mode, the device ignores messages to or from its original I2C static address.
- The I3C CCC command submode of type Direct or Broadcast. Following are the options to exit the CCC:
  - Exit the Broadcast CCC submode by a Repeated START or by a STOP.
  - Exit the Direct CCC submode by a 7Eh broadcast address after a Repeated START or by a STOP.
  - I3C dynamic address assignment (DAA) CCC command enters DAA mode. This mode is a special mode for dynamic addressing. Use STOP to exit. If a target has a dynamic address, the command is ignored.
- I3C SETDASA CCC command enters Static Address Match submode, which allows matching the static address of the device (if any). The command is ignored when a target has a dynamic address, or when the target in I2C mode has no static address.

NOTE

The special point-to-point address is also matched.

- I3C HDR modes are activated by ENTHDR*n* CCC commands, where *n* represents the type of HDR (0 to 7). This mode is valid until an HDR exit pattern is detected, irrespective of whether HDR mode for a target is supported.

  Exit-pattern detection must always be on to prevent errors. Alternatively, you could set it to be activated only on ENTHDR.
- Internal states such as IBI or no-IBI and Low-Power mode are flagged internally (see SSTATUS[EVDET] for more information). These states are exported to the application and affect the engine to restrain it from performing a prohibited operation.

65.4.1.4 Address match

For an address match to occur, the target matches the I3C broadcast address and either the I2C-style static address or I3C dynamic address (see Table 1044).

The address match is inactive (just listening) for all Repeated START commands and also for the START command, unless an IBI, CR, or Hot-Join has been activated. If inactive, it simply tries to match. If not matched, it waits for a Repeated START or

STOP command. If an IBI, CR, or Hot-Join has been activated, the arbitration mechanism is used for START (but never for a Repeated START).

When matching the 7Eh broadcast address, it looks for a CCC until the next Repeated START or STOP command.

Table 1044. Address match conditions

For	Match condition occurs when	Notes
I3C broadcast address	The address is 111 1110b (written as 7Eh).	
I2C-style static address	• The device has a static address. • The device is not in I3C mode.	When in I3C mode, the address matches only until the ENTDAA, SETDASA, or SETAASA command has a dynamic address assigned. SETDASA matches the static address or the special one-controller-to-one-target point-to-point address.
I3C dynamic address	ENTDAA or SETDASA assigns it, or SETNEWDA modifies it.	

65.4.2 Operations

This section describes the operations of the module.

65.4.2.1 Reading and writing I2C messages using the normal method

1. Set up interrupts using the following fields of Controller Interrupt Set (MINTSET):
   - MCTRLDONE: Indicates when the module completes an MCTRL request.
   - COMPLETE: Indicates when data is finished sending or being received.
   - RXPEND: Used for read operations. Works with Controller Data Control (MDATACTRL) to set the FIFO trigger. You can also use this field to allow DMA to read out data.
   - TXNOTFULL: Used for write operations. Works with Controller Data Control (MDATACTRL) to set the FIFO trigger. You can also use this field to allow DMA to supply data.
   - IBIWON: Indicates that an IBI, CR, or HJ has won the arbitration on a header address.
   - Additionally, Controller Errors and Warnings (MERRWARN) indicates the causes of errors and warnings.
2. Configure the following fields in Controller Control (MCTRL) simultaneously:
   a. Write 1 to REQUEST (EmitStartAddr).
   b. Write 1 to TYPE (I2C).
   c. Configure IBIRESP to respond to IBIs in your chosen manner.
   d. Write 1 to DIR for read, or write 0 for write.
   e. Write the static address of the I2C target to ADDR.
   f. Configure RDTERM to the maximum length for read operations, to auto-terminate, or set it to STOP as the data is read out. For example, write 1 (with MCTRL[REQUEST] = 0) to stop after the next character.
3. Write or read the data.
   - For write operations, write to Controller Write Data Byte (MWDATAB) for each byte before the last byte, and then write to Controller Write Data Byte End (MWDATABE) for the last byte:

— You can perform or start this operation before setting up interrupts.

— If there is more data than the FIFO can hold, use the TXNOTFULL interrupt based on the trigger level. This interrupt allows the application to provide more data or to use DMA.

- For read operations, wait for RXPEND, and then read out data via Controller Read Data Byte (MRDATAB). DMA may also be used.

4. After message transfer is complete (MSTATUS[COMPLETE] = 1), the message may end with a STOP condition, or a new message may start with a Repeated START condition:
   a. Write 2 to MCTRL[REQUEST] (EmitStop to STOP). Then wait for the MCTRLDONE status to be asserted for its completion. (When sending STOP in I2C mode, MCONFIG[ODSTOP] and MCTRL[TYPE] must be 1.)
   b. Write 1 to MCTRL[REQUEST] (EmitStartAddr to restart). IBI is not possible in this case.

NOTE

I2C Fm and I2C Fm+ modes are supported for legacy I2C devices as per the I3C standard specifications. See I2C configuration to meet timing requirements for Fm and Fm+ modes for more information.

65.4.2.2 Reading and writing I3C messages using the normal methods (SDR and HDR-DDR)

The normal method for I3C is the same as the method for I2C with a few differences:

1. Set up interrupts using the following fields of Controller Interrupt Set (MINTSET):
   - MCTRLDONE: Indicates when the I3C module completes an MCTRL request.
   - COMPLETE: Indicates when data is finished sending or being received.
   - RXPEND: Used for read operations. Works with Controller Data Control (MDATACTRL) to set the FIFO trigger. You can also use it to allow DMA to read out bytes.
   - TXNOTFULL: Used for write operations. Works with Controller Data Control (MDATACTRL) to set the FIFO trigger. You can also use it to allow DMA to supply bytes.
   - IBIWON: Indicates that an IBI, CR, or HJ has won the arbitration on a header address.
   - Additionally, Controller Errors and Warnings (MERRWARN) indicates the causes of errors and warnings.
2. Set up Controller Control (MCTRL).
   - Option 1: Configure the following fields of Controller Control (MCTRL) simultaneously in this way:
     a. Write 1 to REQUEST (EmitStartAddr).
     b. Write 0 to TYPE for I3C Single Data Rate (SDR) mode or write 2 for Double Data Rate (DDR) mode.
     c. Configure IBIRESP to respond to IBIs in your chosen manner.
     d. Write 1 to DIR for read, or write 0 for write.
     e. Write the dynamic address of the I3C target to ADDR.
     f. For read operations, you can configure RDTERM to the maximum length to terminate automatically.
     g. For write operations, prewriting the data (MWDATAB or MWDATAH) is preferred to ensure that there are no time delays waiting for the data.

        For DMA with MCTRL, use Controller Write Byte Data 1 (to Bus) (MWDATAB1).

NOTE

HDR-DDR mode requires writing an 8-bit command value for read or write. This value must be written into the TX FIFO via Controller Write Data Byte (MWDATAB). The END field is not used for this byte.

- Option 2: This option is preferred when stopped (bus free condition) in SDR mode, and not in HDR-DDR mode. It allows any target to issue an IBI and avoids collisions with an IBI address. Also, it is faster (when the MSB of the dynamic address is always 0).

  Configure the following fields of Controller Control (MCTRL) in this way:

  a. Write 1 to REQUEST (EmitStartAddr). No prewritten transmit data can be in the FIFO.
  b. Write 0 to TYPE (I3C).
  c. Configure IBIRESP to respond to IBIs in your chosen manner.
  d. Write 0 to DIR.
  e. Write 7Eh to ADDR.
  f. Wait for MCTRLDONE (via interrupt, for example), then proceed as in option 1. The option 2 method has advantages for IBIs when 7Eh is sent on START (but not on Repeated START conditions).

3. Write or read the data.
   - For write operations, write to MWDATAB for each byte before the last byte, and then write to Controller Write Data Byte End (MWDATABE) for the last byte. For HDR-DDR, the byte with END must be even (second, fourth, sixth, and so on) because DDR uses byte pairs:
     — You can perform or start this operation before step 1 (REQUEST = 1) of the option 1 method, but not before the option 2 method.
     — If there is more data than the FIFO can hold, use the TXNOTFULL interrupt based on the trigger level. This interrupt allows the application to provide more data or to use DMA.
   - For read operations, wait for RXPEND, and then read out data via Controller Read Data Byte (MRDATAB) or Controller Read Data Halfword (MRDATAH). DMA may also be used. When using DMA, read using the same registers.
4. After message transfer is complete (MSTATUS[COMPLETE] = 1), the message may be ended with STOP (or EXIT in HDR mode). Alternatively, a new message may be started with a Repeated START (or HDR-Restart in HDR mode):
   - In SDR mode, write 2 to MCTRL[REQUEST] (EmitStop to STOP). Then wait for the MCTRLDONE status to be asserted for its completion.
   - For HDR mode, write 6 to MCTRL[REQUEST] (ForceExit) to end HDR mode. Then wait for the MCTRLDONE status to be asserted for its completion. When sending the HDR exit pattern, MCONFIG[ODSTOP] must be 0.
   - Write 1 to MCTRL[REQUEST] (EmitStartAddr to start another message). IBI is not possible in this case.

65.4.2.3 Determining bus types and modes

The settings of MCTRL[TYPE] and MCTRL[REQUEST] determine bus types and modes.

Table 1045. Determining bus types and modes

Value of MCTRL[TYPE]	Meaning when MCTRL[REQUEST] = 1 (EmitStartAddr)	Meaning when MCTRL[REQUEST] = 2 (EmitStop)	Meaning when MCTRL[REQUEST] = 6 (ForceExit)
0	SDR mode of I3C	SDR mode of I3C	Exit pattern
1	Standard I2C protocol	Standard I2C protocol	Reserved
2	The bus enters DDR mode (7E and then ENTHDR0), if the module is not in DDR mode already. The first byte written to the transmit FIFO must be a command and	Reserved	Target reset

Table continues on the next page...

Table 1045. Determining bus types and modes (continued)

Value of MCTRL[TYPE]	Meaning when MCTRL[REQUEST] = 1 (EmitStartAddr)	Meaning when MCTRL[REQUEST] = 2 (EmitStop)	Meaning when MCTRL[REQUEST] = 6 (ForceExit)
	already in the FIFO. To end DDR mode, use ForceExit.		
3	If not already in HDR-BT, the bus automatically enters BT mode (7E and then ENTHDR3). You must also configure the MHDRBTCFG register for multilane and CMD rules.	Reserved	Reserved

65.4.2.4 Sending a CCC to I3C targets

The normal common command code (CCC) method uses an I3C write operation with an address of 7Eh (the CCC is the first byte):

- For broadcast type, any remaining bytes are sent with the CCC. This operation ends with a STOP or a Repeated START and 7Eh.
- For direct type, only the CCC byte is sent (or by a CCC and a defining byte, if required by the CCC).

  These bytes are followed by a Repeated START and the address of the I3C target (for SETDASA, this address is its I2C static address). This sequence may be repeated with more Repeated START conditions and addresses until done. It may end in STOP or a Repeated START and 7Eh. After the Repeated START and target address, the values are read or written depending on the CCC.
- I3C provides interrupts (by hardware) for unhandled and handled CCC when received by the target. Its state is reflected in Target Status (SSTATUS).

The first broadcast 7Eh must meet an open-drain timing check:

- Use the configurations (MCONFIG[ODHPP] = 0 and the normal MCONFIG[ODBAUD]), ensuring at least 200 ns for the open-drain SCL half-clock period.
- Follow with a Repeated START or a STOP.

All subsequent messages must use MCONFIG[ODHPP] = 1.

I3C controllers require to emit a single START, 7E/W sequence with both SCL high and SCL low half periods at full open-drain timing (for example, 200 ns). This sequence allows I3C targets acting as I2C legacy devices to turn off their I2C 50-ns spike filters, if they have them.

NOTE

START, 7E/W is a notation indicating a START command, followed by the 7Eh broadcast address and then by a write command.

After that sequence, addresses following START may be sent with Open-Drain Low (for example, 200 ns) but high of Push-Pull timing (for example, 40 ns). The I3C controller must emit this START, 7E/W sequence with MCONFIG[ODHPP] = 0. ODHPP is Open-Drain High period at Push-Pull speeds, and MCONFIG[ODHPP] is usually 1.

NOTE

- MCONFIG[ODHPP] is ignored when sending a message to an I2C legacy device on an I3C bus.
- Each Repeated START is a new EmitStart request. This request can be chained by an interrupt or pushed by a message model using DMA.

65.4.2.5 IBI handling

An IBI occurs when a target sends its address after a START, and that address is numerically the lowest. That is, it is lower than the address sent by the controller and addresses sent by any other targets. When the controller sends 7Eh, the controller always loses the arbitration.

The IBI can occur unexpectedly when any new START (not a Repeated START) is sent. The IBI can also occur in response to a target pulling SDA low. This condition can occur in one of these ways:

- The controller has set the request to AutoIBI mode, so the IBI occurs automatically. The controller sends 7Eh to allow the target to win the arbitration.
- The application receives the SLVSTART (target START request) interrupt, so it sends 7Eh.

The IBI response is configured by MCTRL[IBIRESP], which can be set to:

- NACK (always reject the IBI).
- ACK (always accept the IBI):
  - You must configure Controller In-band Interrupt Registry and Rules (MIBIRULES) so that the engine knows whether IBI bytes follow.
  - If IBI bytes follow (also known as IBI mandatory byte), then the COMPLETE field does not become 1 when IBIWON becomes 1. RXPEND becomes 1 for one or more bytes in the receive FIFO. When the last byte is received, then the COMPLETE field becomes 1. The controller automatically stops IBI data after 9 bytes (including the mandatory data byte). MCTRL[RDTERM] can be used with MCTRL[REQUEST] = 0 to end the process of receiving data sooner. I3C supports address ACK to mandatory byte transition during IBI from Open_Drain (controller acknowledges the target address) to Push Pull (target sends SDR data).
  - I3C target supports up to 7 bytes (maximum limit is defined by IBIEXT1[MAX]) of extended IBI data following a mandatory data byte. Extended data to send, if any, is present in Extended IBI Data 1 (IBIEXT1) and Extended IBI Data 2 (IBIEXT2).
- Manual (allow the decision to be made by the application on a case-by-case basis):
  - The application rewrites it when stopped, pending an IBI.
  - The application can ACK or NACK the IBI based on the IBI address defined in MSTATUS.
  - This mode selects whether there is an IBI byte when accepting the IBI request from targets.

Accurate timestamping of I3C target data is supported in I3C though Async mode 0. In I3C Asynchronous Timing Control mode, a target device timestamp event occurs within that target. The target notifies the controller about the event by generating an IBI.

65.4.2.6 Assigning dynamic addresses to I3C devices

If dynamic addresses (DAs) are all assigned below 40h (7-bit values from 3Fh down to 03h, except where not allowed), then MIBIRULES[MSB0] can be 1. This setting optimizes the START timing. When dynamic addresses are assigned, you must program Controller In-band Interrupt Registry and Rules (MIBIRULES) based on which I3C target uses IBI bytes (present in SIDEXT[BCR]).

Any SETDASA assignments can be performed via the normal CCC model. These assignments cannot be performed when using the static address for the directed part.

There is a built-in mechanism to process DAA mode:

1. Set up interrupts using the following fields of Controller Interrupt Set (MINTSET):
   - MCTRLDONE: Indicates when a target has sent its ID and BCR or DCR, and a new DA is needed.
   - COMPLETE: Indicates when DAA is done (NACKed by all targets).
   - RXPEND: Indicates the reading of the IDs of the targets. You can also use this field to allow DMA to read out bytes.
   - IBIWON: Indicates when an IBI has occurred (normally there are no IBI conditions when assigning dynamic addresses to the I3C device).

- Additionally, Controller Errors and Warnings (MERRWARN) indicates the causes of errors and warnings.

2. Configure Controller Control (MCTRL):
   a. Write 4 to MCTRL[REQUEST] (ProcessDAA).
   b. Set MCTRL[IBIRESP] to IBI response, if possible. If not possible, set to the first target assignment.
3. Wait for the MCTRLDONE interrupt when reading ID using the RXPEND interrupt. If MSTATUS[STATE] = 5 (DAA mode) and MSTATUS[BETWEEN] = 1, it is waiting to write a dynamic address:
   a. Write the dynamic address into Controller Write Data Byte (MWDATAB) using bits 6:0, such as 14h for DA = 7'h14.
   b. Write 4 to MSTATUS[STATE] (ProcessDAA again). This option writes the DA, then moves to the next DA.
   c. If MSTATUS[COMPLETE] = 1 and MSTATUS[STATE] = 0, then all targets are assigned.
   d. If MSTATUS indicates NACK after writing a new DA, the DA is not accepted by the target. The next step is to write 2 to MCTRL[REQUEST] (EmitStop) and start over. It is also acceptable to write 4 to MCTRL[REQUEST] (ProcessDAA again).

65.4.2.7 Using Controller Message mode

Controller Message mode is intended for use with DMA although Message mode can also be used by the processor. In Message mode, all writes are to the same location, including control and data. Reads occur from an associated location.

NOTE

The data writes for Message mode work in the same way as the halfword access registers, Controller Write Data Halfword (MWDATAH) and Controller Read Data Halfword (MRDATAH).

To send a message via Single Data Rate (SDR), follow these steps (from DMA or processor):

1. Write the control request to Controller Write Message Control in SDR mode (MWMSG_SDR_CONTROL). This request includes the following items:
   - The address.
   - I2C or I3C.
   - Read or write.
   - The count of bytes to process.
   - How to end (on STOP or ready for Repeated START).
2. Process the data:
   - For write operations, write the rest of the data to Controller Write Message Data in SDR mode (MWMSG_SDR_DATA). Use the DMA trigger (or interrupt) to keep the transmit FIFO full, and the controller stops using the data when it has already consumed the number of bytes configured in the LEN field.
   - For read operations, a DMA trigger (or interrupt) indicates when the RX FIFO is ready to be read out via Controller Read Message in SDR mode (MRMSG_SDR).
3. Select the in-band interrupt (IBI) behavior in MCTRL[IBIRESP].
4. Use END type STOP (MCTRL[REQUEST] = 2) with a zero-length message or by using Controller Control (MCTRL). To exit Message mode with the original message, use END type STOP (MCTRL[REQUEST] = 2) with a zero-length message or by using the MCTRL register.

To send a message via Double Data Rate (DDR), follow these steps (from DMA or processor):

1. Write the control request to Controller Write Message in DDR mode: First Control Word (MWMSG_DDR_CONTROL). This request includes:
   - The count of byte pairs.

- How to end the message (through HDR exit or to wait for HDR restart).

2. Write the second control data to MWMSG_DDR_CONTROL. This second write includes:
   - The address.
   - Read or write.
   - The 7-bit command value.
3. Process the data:
   - For write operations, write the rest of the data to MWMSG_DDR_DATA. Use the DMA trigger (or interrupt) to keep the transmit FIFO full, and the controller stops using the data when the program count MWMSG_DDR_CONTROL register reaches 0.
   - For read operations, a DMA trigger (or interrupt) indicates when the receive FIFO is ready to be read out.
4. Select in-band interrupt (IBI) behavior in MCTRL[IBIRESP].
5. Use END type exit (MCTRL[REQUEST] = 6) with a zero-length message or by using the MCTRL register to exit DDR Message mode with the original message.

65.4.2.8 Handing off controllership to another target and getting it back

To hand off controllership, the controller can perform either of the following actions:

- Wait for a controller request (CR) (that is, MSTATUS[IBIWON] = 1 interrupt), indicating that the CR is using MSTATUS[IBITYPE].
- Push the request manually.

In either case, the controller sends a GETACCMST request, which is a directed GET informing the target that it is being assigned controllership. If the target accepts this request, it returns its dynamic address in bits 7:1 and the negative parity of its dynamic address in bit 0. A STOP must be issued, and MCONFIG[MSTENA] must be set to 2 (switching to Target mode).

To gain controllership, the target sends a CR using Target Control (SCTRL):

- If MCONFIG[MSTENA] is 2, the GETACCCR CCC is ACKed.
- If MCONFIG[MSTENA] is not 2, the GETACCCR CCC is NACKed.

After controllership is granted, MSTATUS[NOWMASTER] becomes 1. The application must enable MINTSET[NOWMASTER], so the application is interrupted when a controllership transfer occurs.

65.4.2.9 Controller engine flow diagram

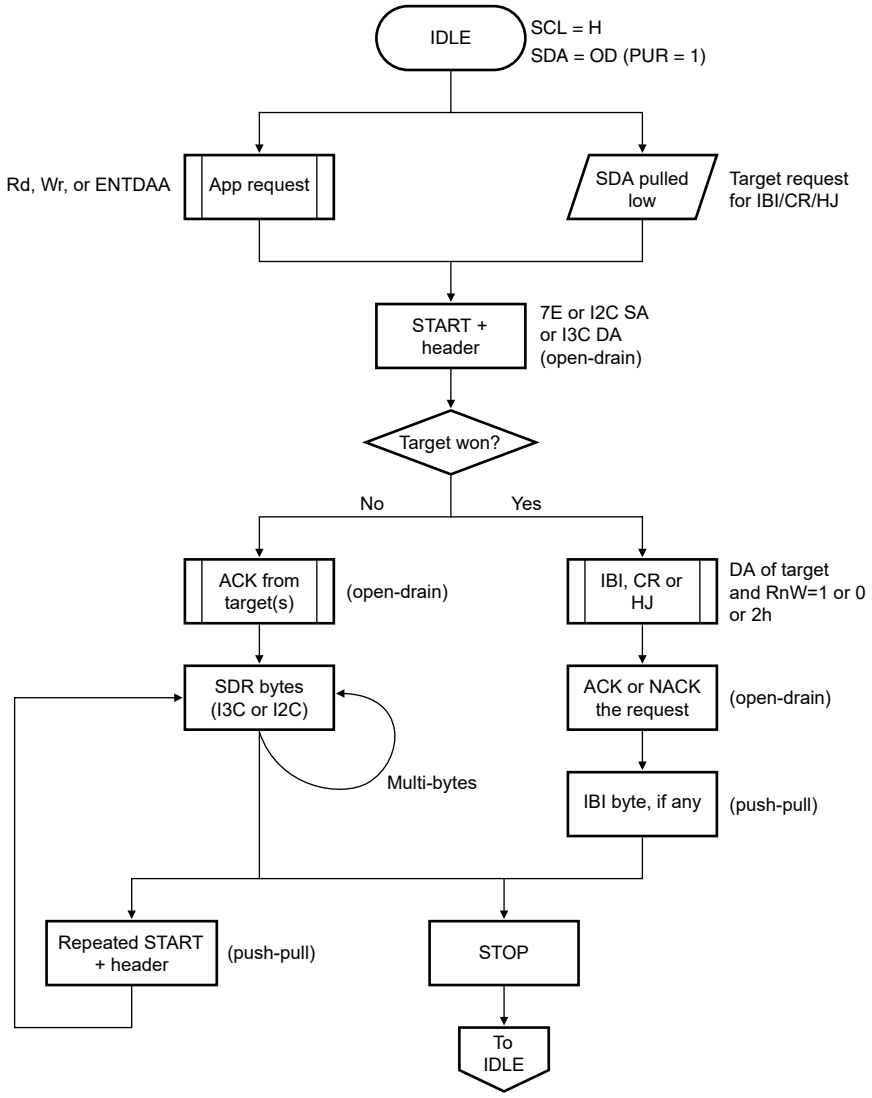


Figure 480. Controller engine flow diagram for SDR (HDR flow progresses from SDR bytes)

65.4.2.10 Target receives data from controller

When the address match is valid for the target dynamic address and the type is W (write from controller), the target ACKs the address. It does so unless some condition prevents it, such as a full buffer. On ACK, the Write state is entered. Once ACKed, it waits for each complete data byte. This process occurs in two steps:

1. Eight bits are clocked in and stored in the next buffer location when in the Write state.
2. On the ninth bit:
   - In I3C mode, the W9TH state (in I3C, the ninth data bit written by the controller is the parity of the preceding eight data bits) is used and the parity is checked. The buffer is marked as complete with or without a parity error.
   - In I2C mode, the W9TH state (in I2C, the ninth data bit written by the controller is an ACK by the target) is used and the data is ACKed and stored.

When the address match is valid for the 7Eh broadcast address, the target acknowledges it. After first data completion, it sets the in_ccc broadcast or direct flag (based on bit 7 of the command). If recognized, it parses the command, setting that bit as well. The

recognized commands are for dynamic address work and modes. The CCC and DAA blocks handle the workload afterward. If not supported, then the commands are passed up to the system.

65.4.2.11 Target sends data to controller

When the address match is valid for the target dynamic address, and the type is R (read by controller), the target ACKs the address. The target does so unless there is no data waiting for the read. On ACK, the Read state is entered.

After being ACKed, it emits the data byte at a time, with the ninth bit using the R9TH state. In I2C, the ninth data bit from target to controller is an ACK by the controller. In I3C, this bit allows the target to end a read, and allows the controller to abort a read:

- In I3C mode, the T bit is emitted to allow the device to indicate whether this byte is the last. If it is not the last byte, the controller may terminate the read via the T bit.
- In I2C mode, the ninth bit allows the controller to terminate via NACK, or else it allows the read to continue via ACK. On completion of each byte read, the "done" signal is pulsed to get the next byte.

If the read is terminated by the controller unexpectedly (abort in I3C, NACK before END marked), the application is notified.

65.4.3 Clocking

The controller works on the following primary clocks:

- The system clock, which controls access to the memory-mapped registers.
- A functional clock (FCLK), which is used to generate the SCL clock rate on the I2C or I3C bus.

I3C supports adjusting clock frequency and accuracy via Target Time Control Clock (STCCLOCK).

FCLK is used to check conditions such as bus availability, bus free for IBI, or HJ:

- To determine the Bus Available condition for IBI, the target needs a clock to generate the ~1 µs timing. To support this requirement, a slow clock (CLK_SLOW) is provided to save on power.

I3C supports a timeout when stalled for too long in a frame. This timeout occurs when:

- The transmit FIFO or receive FIFO is not handled and the bus is stuck in the middle of a message.
- No STOP is issued and the bus is between messages.
- IBI manual is used and no decision is made.

65.4.4 Reset

- Global or system reset fed into the module: Everything is reset by this global or system reset. Release is assumed to be synchronized with the system clock. It is not synchronized with SCL or SDA. These signals must not be active when the reset is released; otherwise, I3C enters Hot-Join mode and remains inactive.
- Software reset

65.4.5 Interrupts

This section describes all the interrupts (IRQs) that this module generates. All status interrupts are updated in Controller Status (MSTATUS) and Target Status (SSTATUS).

Supported interrupts occur:

- For pending IBI, CR, or Hot-Join that have been sent by a target.
- When a CCC is received and is handled by the module (SSTATUS[CHANDLED]).
- When an error or warning has occurred, such as data underrun, data overrun, parity error, HDR-DDR, or CRC error.
- When a CCC is received that is unhandled by I3C. CCC Mask for Unhandled CCCs (SCCCMASK) allows you to suppress any CCC which is intended to be unhandled.

65.5 External signals

Table 1046. External signals

Signal	Description	Direction
SCL	Serial clock	Input or output
SDA	Serial data	Input or output
PUR	Pull up resistance. There is an internal pull-up resistance on SDA, which is controlled by the I3C controller. If the internal pullup is not enough, PUR can be used to control an external pull-up resistance on SDA actively.	Output

65.6 Initialization

65.6.1 Controller configuration

Certain configuration is performed when initializing this module. Controller Configuration (MCONFIG) controls the frequencies, duty cycle, optimizations for performance, and other parameters.

Table 1047. Configuration for module initialization

Configuration parameter		Description
MSTENA	Controller enable	Determines whether the peripheral starts in Controller mode (main controller in I3C terms) or starts in Target mode (and switches to Controller mode later).
HKEEP	High-Keeper	Determines how the high-keeper (weak pullup) is implemented, depending on the device capabilities.
PPBAUD	Push-Pull baud	Sets the push-pull frequency as a divider from the FCLK (alternative clock fed to the peripheral). This frequency sets the SCL half-clock period baseline (used at least for the high time of SCL). The formula for SCL in Push-Pull mode using PPBAUD and PPLOW is: SCL frequency (in MHz) = FCLK ÷ ((2 + 2 × PPBAUD) + PPLOW) If PPLOW is 0, then SCL high = SCL low (in ns) = (1 + PPBAUD) × 1000 ÷ FCLK (in MHz) Examples: • If FCLK = 24 MHz, then PPBAUD = 0 yields 12 MHz (42.67 ns per half period) • If FCLK = 50 MHz, then PPBAUD = 1 yields 12.5 MHz (20 + 20 = 40 ns per half period)
PPLOW	Push-Pull low	Changes the duty cycle for Push-Pull. It indicates how many more FCLK cycles to use for low. The formula for SCL in Push-Pull mode using PPBAUD and PPLOW is: SCL frequency (in MHz) = FCLK ÷ ((2 + 2 × PPBAUD) + PPLOW) If PPLOW is a nonzero value, then: • SCL high (in ns) = (1 + PPBAUD) × 1000 ÷ FCLK (in MHz) • SCL low (in ns) = (1 + PPBAUD + PPLOW) × 1000 ÷ FCLK (in MHz) For example, when FCLK is 50 MHz, PPBAUD is 1, and PPLOW is 1, then the periods are: • (1 + 1) × 1000 ÷ 50 = 20 + 20 = 40 ns high

Table continues on the next page...

Table 1047. Configuration for module initialization (continued)

Configuration parameter		Description
		• (1 + 1 + 1) × 1000 ÷ 50 = 20 + 20 + 20 = 60 ns low This timing is equivalent to 10 MHz SCL, but this timing maintains the 40 ns high needed; therefore, I2C devices do not see the high periods. **NOTE** The PPLOW value does not have any impact on Open-Drain mode and I2C mode SCL rate calculations.
ODBAUD	Open-Drain baud	Determines the number of PPBAUD periods to make up one I3C Open-Drain half-clock baseline. If ODHPP is 0, the ODBAUD half-clock time period = (ODBAUD + 1) × PPBAUD high period. If ODHPP is 0, the formula for SCL in Open-Drain mode is: SCL (in MHz) = FCLK ÷ (2 + 2 × PPBAUD) × (ODBAUD + 1) Target to get open-drain half-clock period is 200 ns. For example: • If PPBAUD yields 12.5 MHz (40 ns per PPBAUD period), then ODBAUD = 4 can be used to get 200 ns. See ODHPP for details on short high and long low. • If PPBAUD = 1, FCLK = 50 MHz, and ODBAUD = 4, then open-drain SCL = 50 ÷ (2 + 2 × 1) × 5 = 2.5 MHz.
ODHPP	Open-Drain High Push-Pull	Optional field that allows the I3C open drain to be long low and short high. The high period of SCL is the PPBAUD period. This period leaves enough time for the pull-up resistor to pull the SDA high when SCL is low. It is also quick when SCL is high and there are no changes happening. ODBAUD low half-clock time period = (ODBAUD + 1) × PPBAUD high period. ODBAUD high half-clock time period = PPBAUD high period If ODHPP is 1, the formula for SCL in Open-Drain mode is: SCL (in MHz) = FCLK ÷ (1 + PPBAUD) × (ODBAUD + 1) + (1 + PBAUD) For example: • If PPBAUD produces 12.5 MHz (40 ns per PPBAUD period), then ODBAUD = 4 and ODHPP = 1. These settings provide a high period of 40 ns and a low period of 200 ns. • If PPBAUD = 1, FCLK = 50 MHz, and ODBAUD = 4, then open-drain SCL = 50 ÷ (1 + 1) × 5 + (1 + 1) = 4.16 MHz.
I2CBAUD	I2C baud	Indicates the number of ODBAUD periods that are required to communicate with I2C devices (MCTRL[TYPE] = 2 or MWMSG_SDR_CONTROL[I2C] = 1). For example, if ODBAUD gives 200 ns, and the goal is Fm+ (Fast Mode, 1 MHz), then the sum must be 1 µs. I2CBAUD acts differently for odd and even values. For example: • If I2CBAUD = 3, it gives three ODBAUD periods low and two ODBAUD periods high.

Table continues on the next page...

Table 1047. Configuration for module initialization (continued)

Configuration parameter		Description
		• If I2CBAUD = 4, it gives four ODBAUD periods for low and four ODBAUD periods for high. • Also, if I2CBAUD = 3, this yields 200 × 3 = 600 ns and 200 × 2 = 400 ns, with the sum 600 + 400 = 1000 ns = 1 µs.
SKEW	Skew	The normal SDA skew from SCL is handled by using the time for the SCL to reach its pad and return. This time is normally 2 ns to 5 ns (or sometimes more). If the skew is too fast, then add more delay. The skew allows specifying the number of FCLKs to insert.

Additional optimizations:

- Use MIBIRULES[MSB0] to obtain faster start header times. If the controller application assigns all I3C dynamic addresses to be less than 40h (it does not have MSB set), you can write 1 to MIBIRULES[MSB0]. When the controller emits 7Eh (broadcast) and a target does not drive the first bit low, the rest of the header can be at push-pull speeds. This speed is two times faster or more, depending on optimizations.
- Auto-emit 7Eh speeds up the frame when used with MIBIRULES[MSB0]. It allows the processor to sleep when the frame starts automatically (in response to a target).

65.6.2 Interrupt service flow

Set up interrupts for Controller Status (MSTATUS) in Controller Interrupt Set (MINTSET).

Set up interrupts for Target Status (SSTATUS) in Target Interrupt Set (SINTSET).

65.7 Application information

65.7.1 I2C configuration to meet timing requirements for Fm and Fm+ modes

I2C Fm and I2C Fm+ modes are supported according to the I3C standard specifications.

Table 1048 shows the configuration for Fm mode, where frequency is close to 400 kHz but timing requirement is not met (T_{SU_STA} is not met) because the actual value is 600 ns. Rest of the requirements are met per the I2C specifications in the chip data sheet.

Table 1049 shows the configuration for Fm mode, where all timing requirements are met.

Table 1048. Configuration for Fm mode with not all timing requirements met

FCLK	PPBAUD	ODBAUD	I2CBAUD	SCL frequency	Timing requirement status
24 MHz	0	4	11	370 kHz	T_{SU_STA} - 353 ns

Table 1049. Configuration for Fm mode with all timing requirements met

FCLK	PPBAUD	ODBAUD	I2CBAUD	SCL frequency	Timing requirement status
24 MHz	1	2	9	363 kHz	All requirements met per the I2C specifications in the chip data sheet

Table 1050 shows the configuration for Fm+ mode, where the frequency is close to 1 MHz. In this case, the T_{SU_STA} timing requirement is not met because the actual value is 260 ns. Rest of the requirements are met per the I2C specifications in the chip data sheet.

Table 1051 shows the configuration for Fm+ mode, where all timing requirements are met. To meet all the timing specifications, frequency may differ from 1 MHz. Use the configurations listed in Table 1051.

Table 1050. Configuration for Fm+ mode with not all timing requirements met

FCLK	PPBAUD	ODBAUD	I2CBAUD	SCL frequency	Timing requirement status
24 MHz	0	4	3	960 kHz	T_{SU_STA} - 228 ns

Table 1051. Configuration for Fm+ mode with all timing requirements met

FCLK	PPBAUD	ODBAUD	I2CBAUD	SCL frequency	Timing requirement status
24 MHz	0	3	6	749 kHz	All requirements met per the I2C specifications in the chip data sheet

NOTE

The controller module provides an option to slow down the clock for standard speed mode to support legacy I2C standard targets as well (although this option may not necessarily meet the exact timing requirement of the standard mode according to the standard I2C timing specification).

For the timing requirements under consideration, see the "I3C timing requirements when communicating with I2C legacy devices" table in the MIPI I3C v1.1.1 specification available on www.mipi.org.

65.8 I3C register descriptions

Writing to a read-only (RO) register, except to Controller Interrupt Mask (MINTMASKED), causes bus errors. This module does not verify whether programmed values in the registers are correct. You must ensure that valid programmed values are written.

However, the peripheral ignores writes to RO registers and returns 0 for reads from WO or nonexistent registers. It requires all 32-bit registers to be read and written as 32-bit and aligned to 32 bits. The peripheral uses the advanced peripheral bus (APB: AMBA 3 APB Protocol Specification v1.0); therefore, the peripheral does not know whether partial read or write operations (less than 32 bits) are used.

65.8.1 I3C memory map

I3C1 base address: 4433_0000h

I3C2 base address: 4252_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Controller Configuration (MCONFIG)	32	RW	0000_0000h
4h	Target Configuration (SCONFIG)	32	RW	0000_0000h
8h	Target Status (SSTATUS)	32	RW	0000_1400h
Ch	Target Control (SCTRL)	32	RW	0000_0000h
10h	Target Interrupt Set (SINTSET)	32	RW	0008_0000h
14h	Target Interrupt Clear (SINTCLR)	32	RW	See section
18h	Target Interrupt Mask (SINTMASKED)	32	R	0000_0000h
1Ch	Target Errors and Warnings (SERRWARN)	32	RW	0000_0000h
20h	Target DMA Control (SDMACTRL)	32	RW	0000_0010h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2Ch	Target Data Control (SDATACTRL)	32	RW	8000_0030h
30h	Target Write Data Byte (SWDATAB)	32	RW	See section
34h	Target Write Data Byte End (SWDATABE)	32	RW	See section
38h	Target Write Data Halfword (SWDATAH)	32	RW	See section
3Ch	Target Write Data Halfword End (SWDATAHE)	32	RW	See section
40h	Target Read Data Byte (SRDATAB)	32	R	0000_0000h
48h	Target Read Data Halfword (SRDATAH)	32	R	See section
54h	Target Write Data Byte (SWDATAB1)	32	RW	0000_0000h
5Ch	Target Capabilities 2 (SCAPABILITIES2)	32	R	0003_0331h
60h	Target Capabilities (SCAPABILITIES)	32	R	FC2F_FE70h
68h	Target Maximum Limits (SMAXLIMITS)	32	RW	0000_0000h
6Ch	Target ID Part Number (SIDPARTNO)	32	RW	5000_0000h
70h	Target ID Extension (SIDEXT)	32	RW	0066_0000h
74h	Target Vendor ID (SVENDORID)	32	RW	0000_011Bh
78h	Target Time Control Clock (STCCLOCK)	32	RW	0000_300Ah
7Ch	Target Message Map Address (SMSGMAPADDR)	32	R	0000_0000h
84h	Controller Control (MCTRL)	32	RW	0000_0000h
88h	Controller Status (MSTATUS)	32	RW	0000_1000h
8Ch	Controller In-band Interrupt Registry and Rules (MIBIRULES)	32	RW	0000_0000h
90h	Controller Interrupt Set (MINTSET)	32	RW	0000_0000h
94h	Controller Interrupt Clear (MINTCLR)	32	RW	See section
98h	Controller Interrupt Mask (MINTMASKED)	32	R	0000_0000h
9Ch	Controller Errors and Warnings (MERRWARN)	32	RW	0000_0000h
A0h	Controller DMA Control (MDMACTRL)	32	RW	0000_0010h
ACh	Controller Data Control (MDATACTRL)	32	RW	8000_0030h
B0h	Controller Write Data Byte (MWDATAB)	32	RW	See section
B4h	Controller Write Data Byte End (MWDATABE)	32	RW	See section
B8h	Controller Write Data Halfword (MWDATAH)	32	RW	See section
BCh	Controller Write Data Halfword End (MWDATAHE)	32	RW	See section
C0h	Controller Read Data Byte (MRDATAB)	32	R	0000_0000h
C8h	Controller Read Data Halfword (MRDATAH)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
CCh	Controller Write Byte Data 1 (to Bus) (MWDATAB1)	32	RW	0000_0000h
D0h	Controller Write Message Control in SDR mode (MWMSG_SDR_CONTROL)	32	RW	See section
D0h	Controller Write Message Data in SDR mode (MWMSG_SDR_DATA)	32	RW	0000_0000h
D4h	Controller Read Message in SDR mode (MRMSG_SDR)	32	R	0000_0000h
D8h	Controller Write Message in DDR mode: First Control Word (MWMSG_DDR_CONTROL)	32	RW	See section
D8h	Controller Write Message in DDR Mode Control 2 (MWMSG_DDR_CONTROL2)	32	RW	See section
D8h	Controller Write Message Data in DDR mode (MWMSG_DDR_DATA)	32	RW	0000_0000h
DCh	Controller Read Message in DDR mode (MRMSG_DDR)	32	R	0000_0000h
E4h	Controller Dynamic Address (MDYNADDR)	32	RW	0000_0000h
100h	Timing Rules for Target Reset Recovery (SRSTACTTIME)	32	RW	0000_0000h
10Ch	CCC Mask for Unhandled CCCs (SCCCMASK)	32	RW	0000_007Fh
110h	Target Errors and Warnings Mask (SERRWARNMASK)	32	RW	0000_0F1Fh
11Ch	Map Feature Control 0 (SMAPCTRL0)	32	R	0000_0000h
120h	Map Feature Control 1 (SMAPCTRL1)	32	RW	0000_0000h
140h	Extended IBI Data 1 (IBIEXT1)	32	RW	0000_0070h
144h	Extended IBI Data 2 (IBIEXT2)	32	RW	0000_0000h

65.8.2 Controller Configuration (MCONFIG)

Offset

Register	Offset
MCONFIG	0h

Function

Controls all controller states when the controller operation is enabled. You must not change this register during an active transaction.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	I2CBAUD				SKEW			ODHPP	ODBAUD							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPLOW				PPBAUD				Reserved	ODSTOP	HKEEP		DISTO	0	MSTENA	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 I2CBAUD	I2C Baud Rate Specifies the I2C low and high times in ODBAUD counts: • I2CBAUD >> 1 is the main count load, and it is count - 1. So, I2CBAUD >> 1: I2CBAUD = 0 for one ODBAUD beat and I2CBAUD = 1 for two ODBAUD beats. • If I2CBAUD[28] is 1, then the low time has one extra ODBAUD beat. The I2CBAUD field is normally 3, where ODBAUD gives 200 ns. For I2CBAUD >> 1, I2CBAUD = 1, which means two ODBAUD beats. To meet the requirements for Fast-mode Plus (Fm+), (2 + 1) × 200 = 600 ns low, with 2 × 200 = 400 ns high for 1 µs period. For Fast mode (Fm), I2CBAUD is normally 11 (giving 2.6 µs) or 6 (giving 2.4 µs).
27-25 SKEW	Skew Specifies the number of FCLK counts for an SDA change after SCL for I3C push-pull. This skew is in addition to the roundtrip of the SCL line from the pad back to the module (6 ns or more): • SKEW is normally not needed, so assign SKEW = 0. • SKEW is only used if SDA is not naturally skewing from an SCL change. • I2C automatically skews SDA (but not PUR) to match I2C rules. Use the following values: • In I3C Controller mode, use SKEW = 0.
24 ODHPP	Open Drain High Push-Pull Enables ODHPP. If you write 0 to this field, open-drain SCL high half-clock period is the same as the open-drain low SCL half period. If you write 1 to this field, open-drain high SCL half-lock period is one PPBAUD count for I3C messages. This setting is faster and works for I3C devices. Any legacy I2C devices on the bus do not see the SCL high level at all (less than the spike filter period). For open-drain timing, check upon the first 7Eh broadcast allowing I3C peripherals acting as I2C legacy devices to turn off their I2C 50-ns spike filters. See Sending a CCC to I3C targets for more information. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 ODBAUD	Open Drain Baud Rate Specifies the open drain baud rate. ODBAUD is configured in terms of number of PPBAUD counts - 1. This field must not be 0; this setting is not the same as push-pull. See Controller configuration for more information. The open-drain baud rate is usually 200 ns (see I2CBAUD for I2C counts). When used with MCONFIG[ODHPP], this setting produces 250 ns per clock in I3C. In this case, if MCONFIG[PPBAUD] provides 12 MHz, then one PPBAUD count is half of 12 MHz, or 41.67 ns. To obtain a rate around 200 ns, use a value of 5 - 1 = 4 for ODBAUD.
15-12 PPLOW	Push-Pull Low Acts as an adder for push-pull low to create a duty cycle with a longer low period, with up to 15 more FCLK cycles low than high. PPLOW = 0 produces a 50/50 duty cycle.
11-8 PPBAUD	Push-Pull Baud Rate Specifies the push-pull baud rate. The number of FCLK counts makes each push-pull low and normally high period. PPBAUD = 0 when run at 1/2 input FCLK speed. For example, a 24 MHz FCLK produces a 12 MHz SCL because each FCLK is SCL low or SCL high. MCONFIG[PPLOW] has an effect on the duty cycle. For example, 24 MHz with 50/50 duty cycle is 12 MHz. However, when PPLOW adds three more low beats, the push-pull baud rate becomes 4.8 MHz (from 24 MHz or 5 beats).
7 —	Reserved
6 ODSTOP	Open Drain Stop Enables open-drain stop. If you write 0 to this field, open-drain stop is disabled. ODSTOP must be disabled when sending an HDR exit pattern. If you write 1 to this field, open-drain stop is enabled. STOP condition is emitted at open-drain speeds even for I3C messages. In legacy devices, this feature can ensure that the legacy devices see the STOP condition. 0b - Disable 1b - Enable
5-4 HKEEP	High-Keeper Indicates how High-Keeper is supported. If this field is 0b, use pull-up resistor (PUR). No separate pin_HK_SDA or pin_HK_SCL is used. Only pin_PUR_oena is used for SDA. The pin_SCL_oena pin is held high in this mode, SCL clock is push-pull, and pin_SCL_out toggles, which is the actual clock. If this field is 1b, High-Keeper controls are in place; pin_HK_SDA or pin_HK_SCL (High-Keeper) controls are used. SCL may use an HK or that signal (pin_HK_SCL) may only OR into pin_SCL_oena. The pin_HK_SDA pin is used as well for SDA high keeper, along with pin_PUR_oena.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 10b, the functionality is passive on SDA, can Hi-Z (high-impedance) for bus free (Idle) and hold. The pins, pin_HK_SDA and pin_HK_SCL, are not used; only a combination of SDA_oena and pin_PUR_oena are used. If this field is 11b, the functionality is passive on SDA and SCL, can Hi-Z (high-impedance) both for bus free (Idle), and can Hi-Z SDA for hold. This is for I2C Clock Stretching mode, where both SCL and SDA are open drain. Use HKEEP = 2 for I2C modes. Use HKEEP = 0/1 for I3C modes. **NOTE** Your chip may not support any or all High-Keeper methods. You must check the availability of High-Keeper methods. 00b - None 01b - WIRED_IN 10b - PASSIVE_SDA 11b - PASSIVE_ON_SDA_SCL
3 DISTO	Disable Timeout Disables the timeout that produces application errors. If the controller is left in a state other than Stopped for more than 100 µs (because 10 kHz is the slowest allowed I3C speed), the timeout sends a MERRWARN interrupt. To prevent the MERRWARN interrupt during development or testing, write 1 to DISTO to disable the timeout. In systems that support timeouts, timeout is disabled automatically during debug. 0b - Enabled 1b - Disabled, if configured
2 —	Reserved
1-0 MSTENA	Controller Enable Indicates whether the controller is enabled and the states it can use. If this field is 0b, the controller is disabled. The module can only use Target mode. If this field is 1b, the controller is enabled. When used upon start-up, this module is the main controller by default. I3C controls the bus unless the controller is handed off. When this happens, MSTENA must become 2 so that it has the capability to become the controller again. Performing the handoff means emitting the GETACCMST CCC command. If the command is accepted, I3C emits a STOP condition and sets this field to 2 (or 0). If this field is 10b, I3C is controller-capable, but is operating as a target now. When used from the start, I3C starts as a target, but is prepared to switch to Controller mode. To switch to this mode, the target emits a controller request (CR) or receives a GETACCMST CCC command and accepts it (to switch on the STOP condition).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - CONTROLLER_OFF 01b - CONTROLLER_ON 10b - CONTROLLER_CAPABLE 11b - Reserved

65.8.3 Target Configuration (SCONFIG)

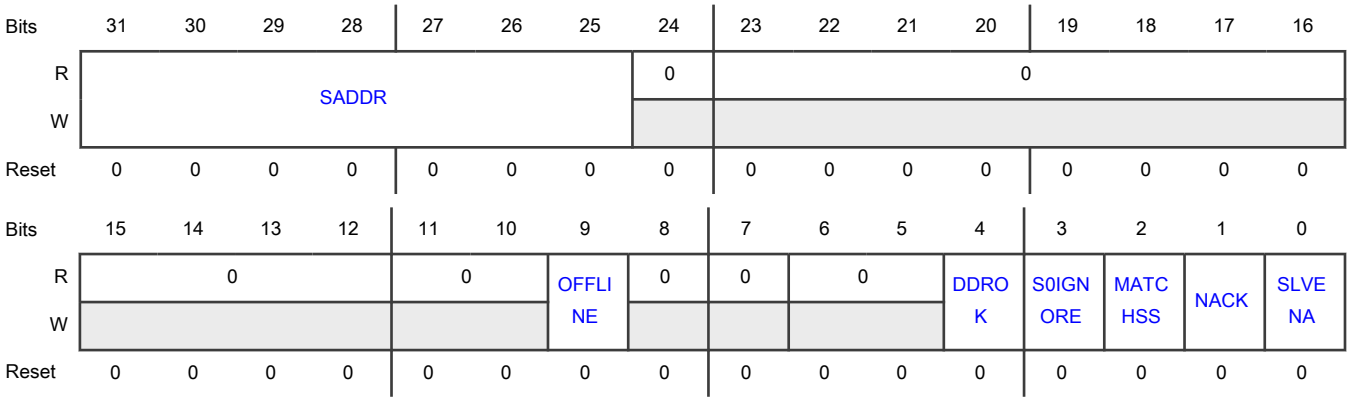
Offset

Register	Offset
SCONFIG	4h

Function

Contains fields that must be configured before the module is activated.

Diagram



Fields

Field	Function
31-25 SADDR	Static Address Sets the I2C 7-bit static address, which otherwise must be 0.
24 —	Reserved
23-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11-10 —	Reserved
9 OFFLINE	Offline Enables wait to ensure that the bus is not in HDR mode. If this field is 1 when SCONFIG[SLVENA] is 1, then I3C waits for either 60 µs of bus quiet or an HDR exit pattern. This waiting ensures that the bus is not in HDR mode, and so can safely monitor the next activity in Single Data Rate (SDR) mode. 0b - Disable 1b - Enable
8 —	Reserved
7 —	Reserved
6-5 —	Reserved
4 DDROK	Double Data Rate OK Allows HDR-DDR messaging. If DDROK = 1, write 1 to the corresponding SIDEXT[BCR] field to indicate that high data rate (HDR) is available, and configure the corresponding HDRCAP HDR-DDR field to allow using DDR mode. **NOTE** DDROK must be 1 before the target can connect to the I3C bus. The target peripheral indicates to the controller whether the target can support the feature during dynamic address assignment. 0b - Do not allow HDR-DDR messaging 1b - Allow HDR-DDR messaging
3 S0IGNORE	Ignore TE0 or TE1 Errors Ignores TE0 or TE1 errors. If you write 1 to this field, the target does not detect TE0 or TE1 errors, so it does not lock up waiting on an exit pattern. You must not use this setting when the bus does not use HDR mode. 0b - Do not ignore TE0 or TE1 errors 1b - Ignore TE0 or TE1 errors

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 MATCHSS	Match Start or Stop Enables match START or STOP condition. If you write 1 to this field, SINTSET[START] and SINTSET[STOP] become 1 only when SSTATUS[MATCHED] is 1. This setting allows the START and STOP fields to be used to detect the end of a message to or from this target. 0b - Disable 1b - Enable
1 NACK	Not Acknowledge Controls the ACK or NACK capability. If you write 1 to this field, the target rejects all requests, except for a CCC broadcast. NACK = 1 must be used with caution because the controller may decide that the target is missing, if NACK is overused. 0b - Always disable NACK mode 1b - Always enable NACK mode (works normally)
0 SLVENA	Target Enable Enables the target. If you write 0 to this field, the target ignores the I2C or I3C bus. If you write 1 to this field, the target can operate on the I2C or I3C bus. You must not write 1 to this field before registers such as Target Configuration (SCONFIG), Target ID Part Number (SIDPARTNO), Target ID Extension (SIDEXT), and others are configured because these registers affect the data to and from the controller. Target enable is configured just once before the bus comes up. If target enable is used at other times, SCAPABILITIES[IBI_MR_HJ] must be 1 before writing 1 to SLVENA so that the device does not see a START or STOP condition incorrectly. 0b - Disable 1b - Enable

65.8.4 Target Status (SSTATUS)

Offset

Register	Offset
SSTATUS	8h

Function

Indicates the sticky status for interrupts, states, and modes related to the I3C bus. Not all fields of the register are used if the module only acts as a target. The fields are divided into current activity, interrupt maskable actions, and then states and modes on the bus.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TIMECTRL		ACTSTATE		HJDIS	0	MRDIS	IBIDIS	0		EVDET		SLVRST	EVENT	CHANDLED	HDRMATCH
W													W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRWARN	CCC	DACHG	TXNOTFU...	RX_PEND	STOP	MATCHED	START	0	STHDR	STDAA	STREQWR	STREQRD	STCCCH	STMSG	STNOTST...
W		W1C	W1C			W1C	W1C	W1C								
Reset	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 TIMECTRL	Time Control Indicates whether time control is enabled, and in which mode. 00b - NO_TIME_CONTROL (no time control is enabled) 01b - SYNC_MODE (Synchronous mode is enabled) 10b - ASYNC_MODE (Asynchronous standard mode (0 or 1) is enabled) 11b - BOTHSYNCASYNC (both Synchronous and Asynchronous modes are enabled)
29-28 ACTSTATE	Activity State from Common Command Codes (CCC) Indicates the activity state from CCC. 00b - NO_LATENCY (normal bus operations) 01b - LATENCY_1MS (1 ms of latency) 10b - LATENCY_100MS (100 ms of latency) 11b - LATENCY_10S (10 seconds of latency)
27 HJDIS	Hot-Join Disabled Indicates whether hot-join is disabled. When hot-join is disabled, CTRL requests are not responded to. 0b - Enabled 1b - Disabled
26 —	Reserved
25 MRDIS	Controller Requests Disable Indicates whether controller requests are disabled. When controller requests are disabled, CTRL requests are not responded to.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enabled 1b - Disabled
24 IBIDIS	In-Band Interrupts Disable Indicates whether in-band interrupts are disabled. When in-band interrupts are disabled, CTRL requests are not responded to. 0b - Enabled 1b - Disabled
23-22 —	Reserved
21-20 EVDET	Event Details Indicates current details of the last (EVENT = 1) or pending event. 00b - NONE (no event or no pending event) 01b - NO_REQUEST (request is not sent yet; either there is no START condition yet, or is waiting for Bus-Available or Bus-Idle (HJ)) 10b - NACKed (not acknowledged, request sent and rejected); I3C tries again 11b - ACKed (acknowledged; request sent and accepted), so done (unless the time control data is still being sent)
19 SLVRST	Target Reset Flag Helps you perform follow-up tasks such as reinitialization (target reset to peripheral (not whole chip)).
18 EVENT	Event Flag Indicates, for a target, whether a pending in-band interrupt (IBI), controller request (CR), or hot-join (HJ) has been sent as requested. See the upper status register fields for details. This field becomes 1 only when acknowledged by a controller. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No event occurred 1b - IBI, CR, or HJ occurred When writing 0b - No effect 1b - Clear the flag
17 CHANDLED	Common Command Code Handled Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates whether an HDRMATCH CCC is being handled by I3C. This field is for notification only, but it may result in updates to this register. **NOTE** This field behaves differently for register reads and writes. When reading 0b - CCC handling not in progress 1b - CCC handling in progress When writing 0b - No effect 1b - Clear the flag
16 HDRMATCH	High Data Rate Command Match Flag Indicates whether the HDR command matched the I3C dynamic address of this device. The HDR command is available as the first byte, with RXPEND = 1. The MSB of the command byte indicates whether it is a read or a write command. If the HDR command is a read, and there are to-bus bytes waiting, then the command is ACKed and the data is sent back. Otherwise, the HDR command is NACKed. **NOTE** When this field is 1, you must check SSTATUS[ERRWARN] because the HPAR error may occur after signaling this HDR command. The parity occurs after the destination address and command. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not match 1b - Matched the I3C dynamic address When writing 0b - No effect 1b - Clear the flag
15 ERRWARN	Error Warning Indicates that an error or warning has occurred, such as data underrun, data overrun, parity error, HDR-DDR CRC error, or other error or warning condition (see Target Errors and Warnings (SERRWARN) for more information).
14 CCC	Common Command Code Flag Indicates whether a CCC has been received, and is not handled by I3C. There are two types of common command codes: • Broadcast CCC, which corresponds with RXPEND, and the first byte is the CCC (command).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Direct CCC, which may never be directed to this device. If the direct CCC is directed to this device, then TXSEND or RXPEND are triggered, and RXPEND contains the command. **NOTE** This field behaves differently for register reads and writes. When reading 0b - CCC not received 1b - CCC received When writing 0b - No effect 1b - Clear the flag
13 DACHG	Dynamic Address Change Flag Indicates an occurrence of dynamic address (DA) change. Actual DA can be seen in the DYNADDR register. If this field is 1, DA change is detected. The target DA has been assigned, reassigned, or reset (lost) and is now in the state of being valid or none. This field is also used when the MAP auto feature is configured, changing one or more MAP items. See DYNADDR and MAPCTRL*n* for more information. DYNAADDR for the main DA (0) indicates whether the last change was because of Auto-MAP. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No DA change detected 1b - DA change detected When writing 0b - No effect 1b - Clear the flag
12 TXNOTFULL	Transmit Buffer Not Full Indicates that the transmit buffer is not full. If DMA is enabled for transmitting, then it is also signaled to provide more data. To-bus buffer or FIFO can accept more data to be transmitted. For all but external FIFOs, this process uses SDATACTRL[TXTRIG], which defaults to "not full". 0b - Transmit buffer full 1b - Transmit buffer not full
11 RX_PEND	Received Message Pending Indicates whether a received message is pending. The field indicates when a message from the controller that is not being handled by I3C (not a CCC message) is received. I3C processes such messages internally. For all but external FIFOs, this process uses SDATACTRL[RXTRIG], which

Table continues on the next page...

Table continued from the previous page...

Field	Function
	defaults to "not empty". If DMA is enabled for receiving, then DMA is signaled as well. This field automatically becomes 0 if data is read (from FIFO and non-FIFO sources). 0b - No received message pending 1b - Received message pending
10 STOP	Stop Flag Indicates whether the Stopped state is detected. The STNOTSTOP state also indicates when the module is in Stop mode. A fast STOP and START combination may not trigger the Stop status. In that case, the Start status is always set. If this field is 1, it indicates that the Stop state was present on the bus since the bus was last cleared. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No Stopped state detected 1b - Stopped state detected When writing 0b - No effect 1b - Clear the flag
9 MATCHED	Matched Flag Indicates whether an incoming header matched the I3C dynamic or I2C static address (if any) of this device since the bus was last cleared. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Header not matched 1b - Header matched When writing 0b - No effect 1b - Clear the flag
8 START	Start Flag Indicates whether a START or Repeated START was detected after this flag was last cleared. This flag is not usually needed, but can be used for wake events.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
7 —	Reserved
6 STHDR	Status High Data Rate Indicates whether the I3C bus is in HDR-DDR mode, regardless of whether HDR mode is supported by this module, and regardless of whether the message is intended for this module or some other module. 0b - I3C bus not in HDR-DDR mode 1b - I3C bus in HDR-DDR mode
5 STDAA	Status Dynamic Address Assignment Indicates whether the I3C bus is in Enter Dynamic Address Assignment (ENTDAA) mode, regardless of whether this bus target has a dynamic address. 0b - Not in ENTDAA mode 1b - In ENTDAA mode
4 STREQWR	Status Request Write Indicates whether the REQ in process is SDR write data from the controller to this bus target (or all targets), but not in ENTDAA mode. See status high data rate (STHDR) for double data rate (DDR) handling. 0b - Not an SDR write 1b - SDR write data from the controller, but not in ENTDAA mode
3 STREQRD	Status Request Read Indicates whether the REQ in process is an SDR read from this target. See status high data rate (STHDR) for double data rate (DDR) handling. 0b - Not an SDR read 1b - SDR read from this target or an IBI is being pushed out
2	Status Common Command Code Handler

Table continues on the next page...

Table continued from the previous page...

Field	Function
STCCCH	Indicates whether a CCC message is being handled automatically. 0b - No CCC message handled 1b - Handled automatically
1 STMSG	Status Message Indicates whether the bus target is busy (listening to the bus traffic or responding). If STNOSTOP = 1, STMSG is 0 when a nonmatching address is seen, until the next Repeated START or STOP condition occurs. 0b - Idle 1b - Busy
0 STNOTSTOP	Status not Stop Indicates the status of the bus. If this field is 0, I3C is in a STOP condition. If this field is 1, the bus is busy (has activity). Other fields of this register may also be set when busy. STNOTSTOP can also become 1 after a TE0 or TE1 error, when I3C is waiting for an exit pattern. 0b - In STOP condition 1b - Busy

65.8.5 Target Control (SCTRL)

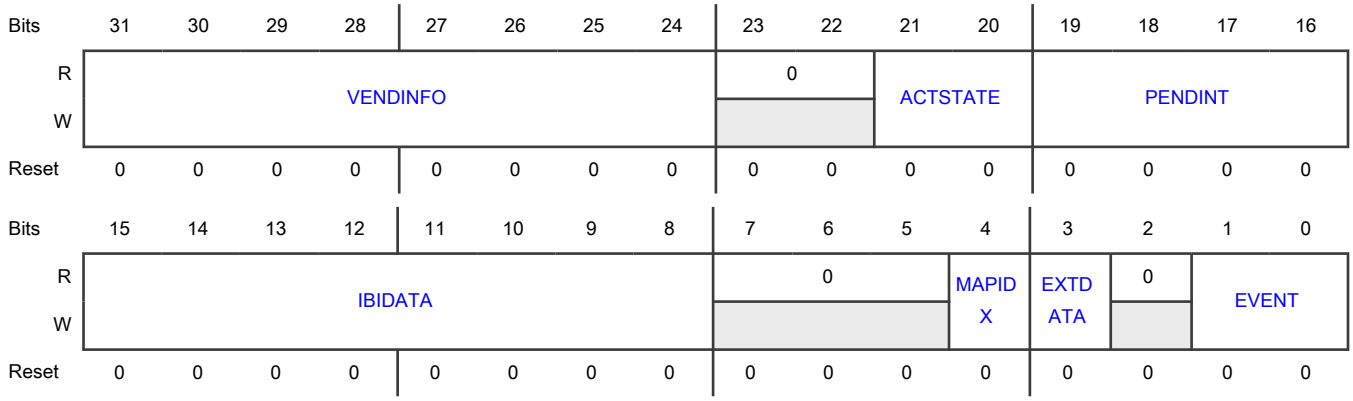
Offset

Register	Offset
SCTRL	Ch

Function

Contains controls for the active use of the I3C bus (for example, event generation such as interrupts to the controller). Only if I3C is configured to support various special operations for the target, this register is used to activate those operations. These events include IBI and GETSTATUS fields (except the protocol error, which is automatically set).

Diagram



Fields

Field	Function
31-24 VENDINFO	Vendor Information Controls vendor information that the GETSTATUS CCC returns. You must set this field to the Vendor Reserved field that the GETSTATUS CCC returns. The application must maintain the vendor information because the controller reads this field. If this field is not configured, then the GETSTATUS field always returns 0.
23-22 —	Reserved
21-20 ACTSTATE	Activity State of Target Controls the activity state of the target. You must set this field to the activity state of the target that the GETSTATUS CCC command returns as activity mode. The application must maintain the activity state because the controller reads this field. If you do not configure the activity state, then the GETSTATUS command always returns 0.
19-16 PENDINT	Pending Interrupt Specifies whether an IBI interrupt is pending. You must set this field to the pending interrupt that the GETSTATUS CCC command returns. The application must maintain the pending interrupt because the controller reads this field. If PENDINT = 0: • The GETSTATUS command returns 1 if an IBI interrupt is pending. • The GETSTATUS field returns 0 if an IBI interrupt is not pending.
15-8 IBIDATA	In-Band Interrupt Data Controls the data byte accompanying the IBI, if the module is enabled for IBI. If SCTRL[IBIDATA] is enabled, then IBI is required.
7-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 MAPIDX	Map Index Controls the index of the dynamic address of the IBI. This index is 0 for the main or base dynamic address. This field is used only if mapping is enabled. See Controller Dynamic Address (MDYNADDR) for more information.
3 EXTDATA	Extended Data Enables extended data. After IBIDATA is emitted, extended data is acquired from IBIEXT1 and IBIEXT2, if configured. If extended data is used with time control, the data follows the time information. See IBIEXT1[MAX] for more information. 0b - Disable 1b - Enable
2 —	Reserved
1-0 EVENT	Event Requests an event. If this field is 0b, it indicates Normal mode, in which if this field becomes 0 from a nonzero value and event processing has not yet started, the event processing is canceled. However, event processing is not canceled if it has already started. If this field is 1b, an IBI is pushed onto the I3C bus. If there is data associated with the IBI, the data is read from SCTRL[IBIDATA]. If time control is enabled, this data includes any time-control-related bytes. Additionally, SCTRL[IBIDATA][7] becomes 1 automatically (as is required for time control). The IBI interrupt occurs after the first (mandatory) IBIDATA, if any. If this field is 10b, a controller request is started; the meaning depends on SIDEXT[BCR] configured in I3C. If this field is 11b, a hot-join (HJ) request is started, which is used when the device is powered on after the I3C bus is already powered up. It is also used when the device is connected using hot-insertion methods (the device is powered up when it is physically inserted in the powered-up I3C bus). The HJ waits for Bus Idle, and SCTRL[EVENT] = HOT_JOIN_REQUEST must be set before the target enable (SCONFIG[SLVENA]). If this field is a nonzero value, it requests an event. After the request, SSTATUS[EVENT] and SSTATUS[EVDET] show the status as it progresses. After completion, this field automatically returns to 0. After this field becomes a nonzero value, you can write only 0 it (to cancel) until the event processing is finished. 00b - NORMAL_MODE 01b - IBI 10b - CONTROLLER_REQUEST 11b - HOT_JOIN_REQUEST

65.8.6 Target Interrupt Set (SINTSET)

Offset

Register	Offset
SINTSET	10h

Function

Sets interrupt enables for select Target Status (SSTATUS) fields. Reading this register (SINTSET) returns the status of the interrupt enable:

- To activate an interrupt enable, write 1 to its corresponding field in this register (SINTSET).
- To disable an interrupt, write 1 to its corresponding field in Target Interrupt Clear (SINTCLR). Writing 0 to the interrupt enable in this register (SINTSET) does not disable the interrupt.

The Interrupt registers allow the masking of interrupt sources. They also allow the checking of which interrupts are activated. The normal method is to enable an interrupt, and then after the interrupt occurs, clear the interrupt either by writing to Target Status (SSTATUS) or by performing an action on the corresponding data register. The interrupt is level-held, meaning the interrupt stays set until the cause is cleared by some method. The module prevents races; if a new event occurs, that new event is not lost:

- SINTSET sets interrupt enables for Target Status (SSTATUS) fields. Reading the SINTSET register returns the status of the interrupt enables.
- SINTCLR clears interrupt enables for the SSTATUS fields.
- SINTMASKED returns the value of the SSTATUS fields ANDed with their interrupt enables.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												SLVRST	EVENT	CHANDLED	DDRMATC...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRWARN	CCC	DACHG	TXSEND	RXPEND	STOP	MATCHED	START	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 SLVRST	Target Reset Enables target reset interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field specifies whether target reset pattern is detected and action is set to reset peripheral. The field helps you perform follow-up tasks such as reinitialization. You must reset peripheral or verify that it is correct. This interrupt must not be unmasked. Used only if SLVRST is configured. 0b - Disable 1b - Enable
18 EVENT	Event Interrupt Enable Enables event interrupts. This field indicates, for a target, whether a pending in-band interrupt (IBI), controller request (CR), or hot-join (HJ) has been sent as requested. The field is configured to support events. See SSTATUS[EVDET] for more information. 0b - Disable 1b - Enable
17 CHANDLED	Common Command Code (CCC) Interrupt Enable Enables CCC interrupts. This field specifies whether CCC is being handled by I3C. Target Status (SSTATUS) shows new results. You can use this field to track when activity states and masks on events (for example, IBIs) occur. The field is used for CCCs that are enabled. 0b - Disable 1b - Enable
16 DDRMATCHED	Double Data Rate Interrupt Enable Enables DDR match for read or write command. This field indicates when DDR matches for read or write command, and the field is used only if HDR is enabled. 0b - Disable 1b - Enable
15 ERRWARN	Error or Warning Interrupt Enable Enables error or warning interrupts. This field indicates whether an error or warning has occurred, such as data underrun, data overrun, parity error, or HDR-DDR CRC error, or any other error or warning condition. See Target Errors and Warnings (SERRWARN) for cause of error. Only available for errors related to configured features. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
14 CCC	Common Command Code (CCC) Interrupt Enable Enables CCC interrupts. This field indicates whether a CCC has been received, and is not handled by I3C. For CCCs that the block does not handle, RXPEND also interrupts, and SSTATUS[STREQRD] indicates that it is a CCC sending a read request. 0b - Disable 1b - Enable
13 DACHG	Dynamic Address Change Interrupt Enable Enables dynamic address change interrupts: interrupt on dynamic address defined (SETDASA or ENTDAA) or lost (RSTDAA). See Controller Dynamic Address (MDYNADDR) for more information. 0b - Disable 1b - Enable
12 TXSEND	Transmit Interrupt Enable Enables transmit interrupts. This field indicates whether to-bus buffer or FIFO can accept more data to be transmitted. The corresponding interrupt occurs when the controller requests data to read from this target. This interrupt occurs on the first request (header) as well as when it is ready for more data. The corresponding interrupt occurs when the controller requests data to be read from this target. If this interrupt is for a FIFO, it triggers on the transmit emptiness trigger. If this interrupt is for DMA, then it indicates message end (DMA end or termination). 0b - Disable 1b - Enable
11 RXPEND	Receive Interrupt Enable Enables receive interrupts. This field specifies when a message from the controller that is not being handled by the module is received. For example, where data is consumed by hardware directly when it goes into the FIFO (excludes CCCs being handled automatically). If this interrupt is for a FIFO, it indicates a receive fullness trigger. If this interrupt is for DMA, then it indicates message end. 0b - Disable 1b - Enable
10 STOP	Stop Interrupt Enable Enables stop interrupts. This field detects the Stopped state present on the bus since the bus was last cleared.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Use SINTSET[START] as the preferred interrupt when needed. This interrupt may not trigger for a quick STOP and START combination because it relates to the state of being stopped. 0b - Disable 1b - Enable
9 MATCHED	Match Interrupt Enable Enables match interrupts. Incoming header matched the I3C dynamic or I2C static address of this device since the bus was last cleared. If configured and if no dynamic address is set, this interrupt is also for matching a header on an I2C static address. See Controller Dynamic Address (MDYNADDR) for related information. 0b - Disable 1b - Enable
8 START	Start Interrupt Enable Enables start interrupts. A START or Repeated START (such as wakeup) is seen after this field last became 0. See SINTSET[STOP] for related information. 0b - Disable 1b - Enable
7-0 —	Reserved

65.8.7 Target Interrupt Clear (SINTCLR)

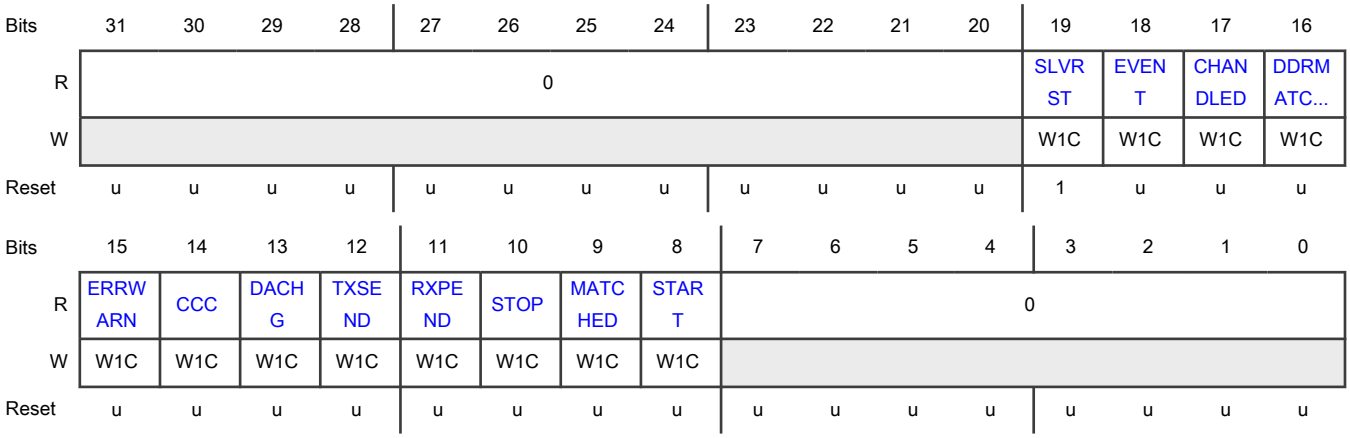
Offset

Register	Offset
SINTCLR	14h

Function

Clears interrupt enables for select Target Status (SSTATUS) fields. To clear an interrupt enable, write 1 to the corresponding field in this register (SINTCLR). Writing 0 has no effect.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 SLVRST	Target Reset Flag (SLVRST Interrupt Enable Clear)
18 EVENT	EVENT Interrupt Enable Clear Flag
17 CHANDLED	CHANDLED Interrupt Enable Clear Flag
16 DDRMATCHED	DDRMATCHED Interrupt Enable Clear Flag
15 ERRWARN	ERRWARN Interrupt Enable Clear Flag
14 CCC	CCC Interrupt Enable Clear Flag
13 DACHG	DACHG Interrupt Enable Clear Flag
12 TXSEND	TXSEND Interrupt Enable Clear Flag
11	RXPEND Interrupt Enable Clear Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXPEND	
10 STOP	STOP Interrupt Enable Clear Flag
9 MATCHED	Matched Interrupt Enable Clear Flag
8 START	START Interrupt Enable Clear Flag
7-0 —	Reserved

65.8.8 Target Interrupt Mask (SINTMASKED)

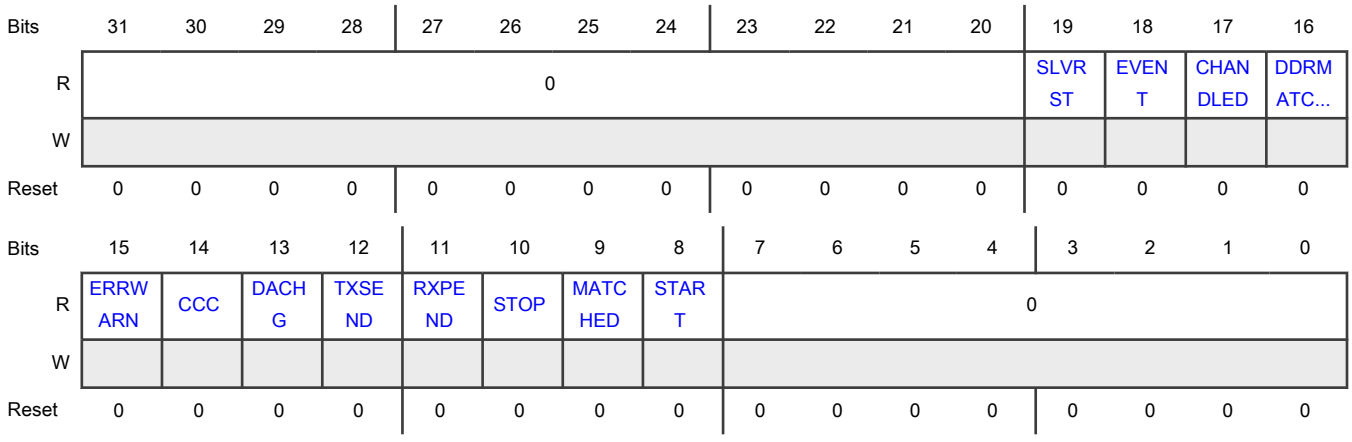
Offset

Register	Offset
SINTMASKED	18h

Function

Returns the status of enabled interrupts (the value of Target Status (SSTATUS) ANDed with the value of Target Interrupt Set (SINTSET)).

Diagram



Fields

Field	Function
31-20 —	Reserved
19 SLVRST	Target Reset Interrupt Mask
18 EVENT	EVENT Interrupt Mask
17 CHANDLED	CHANDLED Interrupt Mask
16 DDRMATCHED	DDRMATCHED Interrupt Mask
15 ERRWARN	ERRWARN Interrupt Mask
14 CCC	CCC Interrupt Mask
13 DACHG	DACHG Interrupt Mask
12 TXSEND	TXSEND Interrupt Mask
11 RXPEND	RXPEND Interrupt Mask
10 STOP	STOP Interrupt Mask
9 MATCHED	MATCHED Interrupt Mask
8 START	START Interrupt Mask
7-0 —	Reserved

65.8.9 Target Errors and Warnings (SERRWARN)

Offset

Register	Offset
SERRWARN	1Ch

Function

Contains errors and warnings from I3C and I2C protocols. This includes internal issues such as overrun and underrun, detected errors and conditions like parity errors, CRC errors, and read terminations by the controller. See SSTATUS[ERRWARN] and SINTSET[ERRWARN] for related information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0										0		0		OWRITE	OREAD
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		0	0	S0S1	HCRC	HPAR	SPAR	0			INVSTART	TERM	URUNNACK	URUN	ORUN
W					W1C	W1C	W1C	W1C				W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-22 —	Reserved
21 —	Reserved
20-18 —	Reserved
17 OWRITE	Over-Write Error Flag Indicates that Target Write Data Byte (SWDATAB) or Target Write Data Byte End (SWDATABE) was written to when full. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No overwrite error 1b - Overwrite error When writing 0b - No effect 1b - Clear the flag
16 OREAD	Over-Read Error Flag Indicates that Target Read Data Byte (SRDATAB) was read for more bytes than were available, by the application. This field also indicates over-read errors for Target Read Data Halfword (SRDATAH), if it is enabled. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No over-read error 1b - Over-read error When writing 0b - No effect 1b - Clear the flag
15-14 —	Reserved
13 —	Reserved
12 —	Reserved
11 S0S1	TE0 or TE1 Error Flag Indicates whether a TE0 or TE1 error occurred and the target is locked and waiting for an HDR exit pattern. Writing 1 to S0S1 causes I3C to release the lock, but this method must be used with great care. S0S1 becomes 0 automatically when an exit pattern is detected, so writing 1 to S0S1 must be used under controlled circumstances to avoid problems. Before starting to operate normally after this error, I3C waits for a START (or Repeated START) or STOP state. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No TE0 or TE1 error occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - TE0 or TE1 error occurred When writing 0b - No effect 1b - Clear the flag
10 HCRC	HDR-DDR CRC Error Flag Indicates whether an HDR-DDR CRC error occurred on a message from the controller. Error reasons include an HDR restart and an exit being issued before an HDR-DDR message from the controller has finished. This error calls into question the data from the entire DDR command frame. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No HDR-DDR CRC error occurred 1b - HDR-DDR CRC error occurred When writing 0b - No effect 1b - Clear the flag
9 HPAR	HDR Parity Error Flag Indicates when an HDR parity error or framing error on a message from the controller occurs. The corresponding command or data that has the error is usually in the RX buffer, which can be read using Target Read Data Byte (SRDATAB). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No HDR parity error 1b - HDR parity error When writing 0b - No effect 1b - Clear the flag
8 SPAR	SDR Parity Error Flag Indicates when an SDR parity error on a message from the controller occurs. This error also sets the GETSTATUS protocol error field (which becomes 0 after a GETSTATUS read). For read operations, this field becomes 1 when a read abort (timeout) occurs because of the controller not driving clock for more than 100 µs during an SDR read.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - No SDR parity error 1b - SDR parity error When writing 0b - No effect 1b - Clear the flag
7-5 —	Reserved
4 INVSTART	Invalid Start Error Flag Indicates an invalid condition with SCL falling before SDA falls, so there is no start. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No invalid start error 1b - Invalid start error When writing 0b - No effect 1b - Clear the flag
3 TERM	Terminated Error Flag Indicates when the controller terminates a read from a target, if an END is not set by the target (on the same read or the previous read). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No terminated error 1b - Terminated error When writing 0b - No effect 1b - Clear the flag
2	Underrun and Not Acknowledged (NACKed) Error Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
URUNNACK	Indicates when the internal to-bus buffer or FIFO is underrun in the read header and so the module NACKed the header. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No underrun; not acknowledged error 1b - Underrun; not acknowledged error When writing 0b - No effect 1b - Clear the flag
1 URUN	Underrun Error Flag Indicates when the internal to-bus buffer or FIFO is underrun during data read (the application is not providing the data fast enough). The END field or register must be used if the read was the last one. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No underrun error 1b - Underrun error When writing 0b - No effect 1b - Clear the flag
0 ORUN	Overrun Error Flag Indicates when the internal from-bus buffer or FIFO has overrun (arrival of too many characters that you cannot process fast enough). Cutting off the RX_FIFO too close to the boundary when DUT is a target for SDR write or HDR-DDR write can cause an ORUN error when the FIFO is full. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overrun error 1b - Overrun error When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Field	Function

65.8.10 Target DMA Control (SDMACTRL)

Offset

Register	Offset
SDMACTRL	20h

Function

Allows DMA to be used for inbound and outbound messages. This register is limited in value for target use because the target must be reactive. These are the two common use case models:

- To avoid an overrun in from-bus collection: if SCONFIG[MATCHSS] becomes 1, then the processor enables the interrupts for START and STOP and enables the DMA to collect the data. The START or STOP interrupt only occurs after a message is directed to the target (MATCHED is 1). The DMA copied data can then be examined.

  NOTE

  Do not enable DMA after a transaction starts. To avoid an RX FIFO overrun, DMA must be enabled only when enabling the target and interrupts.

- To perform larger reads from the to-bus: I3C and I2C reads are preceded by a write that indicates what will be read (or in response to an IBI from the target). Because of this process, the DMA can be used to push through the data:
  - For I3C, the processor must manage the last value, unless the DMA moves wider words and is able to write 1 to the END field. These values are 16-bit values when in Byte mode or 32-bit values when in Halfword mode.
  - For I2C, the controller determines the last value, so the DMA may end early or run out when the controller expects more values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0		DMAWIDTH		DMATB		DMAFB	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7-6 —	Reserved
5-4 DMAWIDTH	Width of DMA Operations Indicates the width of DMA operations, if configured to allow halfword data accesses. 00b,01b - Byte 10b - Halfword (16 bits) (this value ensures that two bytes are available in the FIFO) 11b - Reserved
3-2 DMATB	DMA Write (To-Bus) Trigger Enables DMA writes. If this field is enabled, it starts a request DMA on a transmit trigger (see Target Data Control (SDATACTRL)). DMATB requests until full, unless the DMA is set up as a trigger. DMATB becomes 0 when MSTATUS[ERRWARN] becomes 1. If you write 1b to this field, DMA is enabled for one frame (ended by DMA or terminated). DMATB automatically becomes 0 after a STOP or START. See SCONFIG[MATCHSS] for more information. If you write 10b to this field, DMA is enabled until turned off. The value must only be used with Controller Message mode. 00b - DMA not used 01b - DMA enabled for one frame 10b - DMA enabled until turned off 11b - Reserved
1-0 DMAFB	DMA Read (From-Bus) Trigger Enables DMA reads. If this field is enabled, DMAFB requests DMA on receive trigger (see SDATACTRL). It requests until empty, unless the DMA is set up as a trigger. This field becomes 0 when SSTATUS[ERRWARN] becomes 1. If this field is 1b (DMA is enabled for one frame), it automatically becomes 0 on STOP or Repeated START. See SCONFIG[MATCHSS] for more information. **NOTE** Do not enable DMA after a transaction starts. It must be enabled only when enabling the target and interrupts to avoid any kind of RX FIFO overrun. 00b - DMA not used 01b - DMA enabled for one frame

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - DMA enabled until turned off 11b - Reserved

65.8.11 Target Data Control (SDATACTRL)

Offset

Register	Offset
SDATACTRL	2Ch

Function

Assists in data control when no FIFO is used. Also assists in data control of FIFO when the FIFO is available (regardless of size) allowing some control over the FIFO behavior. This register controls when to interrupt based on fullness or emptiness of a buffer or FIFO. It also controls behavior related to width, when the buffer or FIFO is not 1 byte wide.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXEMPTY	TXFULL	0	RXCOUNT					0		0		TXCOUNT			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RXTRIG		TXTRIG			0		
W													UNLOCK		FLUSHFB	FLUSHTB
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

Fields

Field	Function
31 RXEMPTY	Receive is Empty 0b - Not empty 1b - Empty
30 TXFULL	Transmit is Full 0b - Not full 1b - Full

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 —	Reserved
28-24 RXCOUNT	Count of Bytes in Receive
23-22 —	Reserved
21 —	Reserved
20-16 TXCOUNT	Count of Bytes in Transmit
15-8 —	Reserved
7-6 RXTRIG	Receive Trigger Level Indicates the trigger level for receive fullness when using a FIFO. The field affects the RXPEND interrupt. 00b - Trigger when not empty 01b - Trigger when 1/4 or more full 10b - Trigger when 1/2 or more full 11b - Trigger when 3/4 or more full
5-4 TXTRIG	Transmit Trigger Level Indicates the trigger level for transmit emptiness when using a FIFO. The field affects the TXNOTFULL interrupt. 00b - Trigger when empty 01b - Trigger when 1/4 full or less 10b - Trigger when 1/2 full or less 11b - Default (trigger when 1 less than full or less)
3 UNLOCK	Unlock Indicates whether the RXTRIG and TXTRIG fields can be changed on a write. This field must be 1 in the same cycle while writing to TXTRIG or RXTRIG. 0b - Cannot be changed 1b - Can be changed
2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 FLUSHFB	Flush From-Bus Buffer or FIFO Not normally used.
0 FLUSHTB	Flush To-Bus Buffer or FIFO Indicates when the controller terminates a to-bus (read) message prematurely.

65.8.12 Target Write Data Byte (SWDATAB)

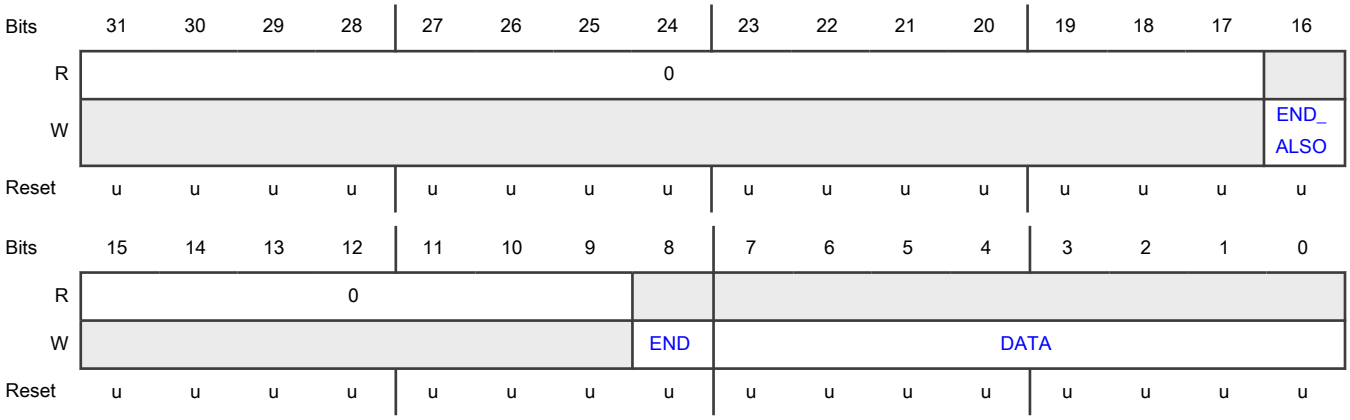
Offset

Register	Offset
SWDATAB	30h

Function

Allows writing a byte to the bus (to the controller) unless an external FIFO is used. Writing a byte requires a byte plus an end-of-data (last) marker bit. A byte must not be written unless there is room, indicated by SSTATUS[TXNOTFULL] = 1.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END_ALSO	End Also Marks the last byte of the message. This field is required for I3C, but is optional for I2C.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 0, it indicates that there are more bytes in the message. For HDR-DDR, the byte with END_ALSO must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. 0b - Not the end 1b - End
15-9 —	Reserved
8 END	End Marks the last byte of the message. If this field is 0, it indicates that there are more bytes in the message. This field is required for I3C, but is optional for I2C. For HDR-DDR, the byte with END must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. 0b - Not the end 1b - End
7-0 DATA	Data Indicates the data byte to be sent to the controller.

65.8.13 Target Write Data Byte End (SWDATABE)

Offset

Register	Offset
SWDATABE	34h

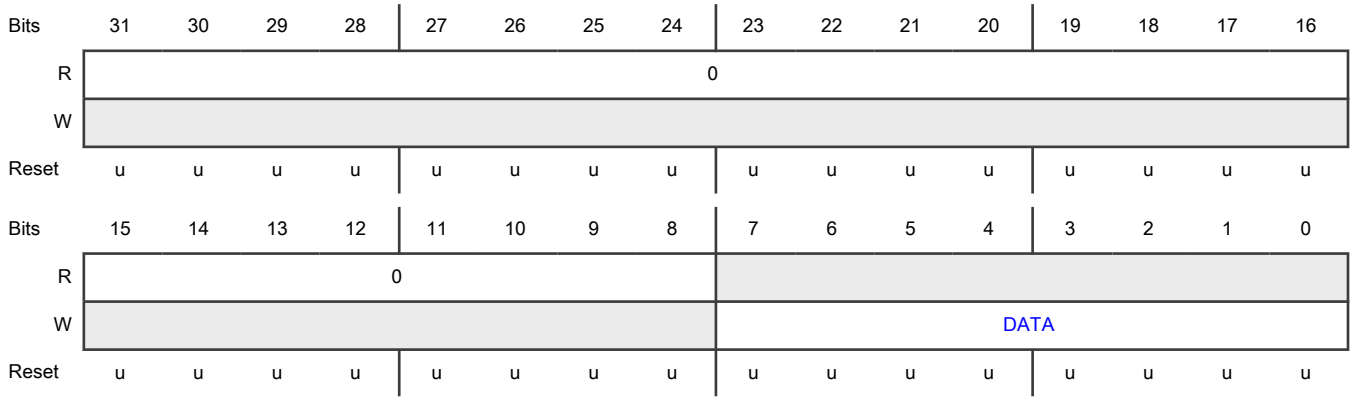
Function

Allows writing a byte to the bus (to the controller) unless an external FIFO is used.

Unlike Target Write Data Byte (SWDATAB), writing a byte only requires the byte itself, and is marked as end-of-data (last byte). A byte must not be written unless there is room, indicated by SSTATUS[TXNOTFULL] = 1.

For HDR-DDR, the byte with the END must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 DATA	Data Indicates the data byte to be sent to the controller.

65.8.14 Target Write Data Halfword (SWDATAH)

Offset

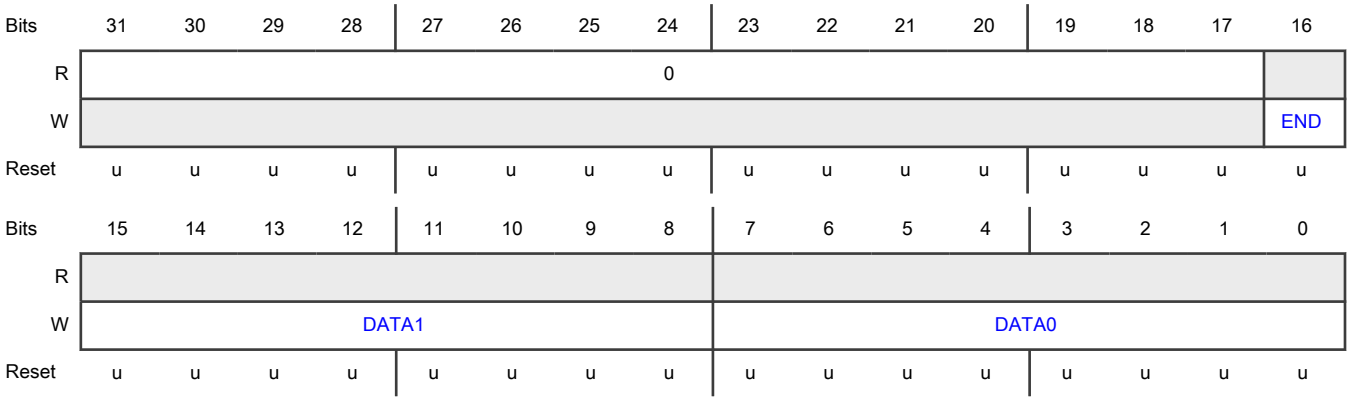
Register	Offset
SWDATAH	38h

Function

Allows writing a halfword (pair of bytes) to the bus unless an external FIFO is used. The low byte is followed by the high byte. The 16th bit marks the end; that is, the last byte of the halfword is the end.

An end-of-data (last) marker bit is allowed (or must be 0). A halfword must not be written unless there is room for both, as indicated by the use of transmit FIFO level trigger or SDATACTRL[TXCOUNT].

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END	End of Message Marks the last byte of the message. This field always marks DATA1 as the end. The field is required for I3C, but is optional for I2C. If this field is 0, it indicates that there are more bytes in the message. For HDR-DDR, the byte with the END must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. 0b - Not the end 1b - End
15-8 DATA1	Data 1 Indicates the second byte to be sent to the controller.
7-0 DATA0	Data 0 Indicates the first byte to be sent to the controller.

65.8.15 Target Write Data Halfword End (SWDATAHE)

Offset

Register	Offset
SWDATAHE	3Ch

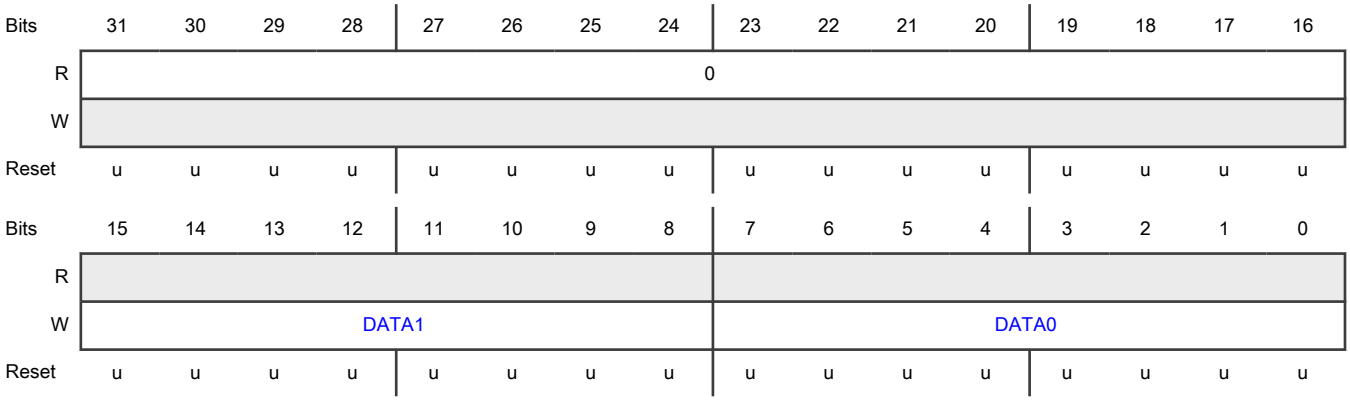
Function

Allows writing a halfword of data, which is the end (the last byte of the halfword is the end). The target writes the halfword (byte pair) in the same way it writes to Target Write Data Halfword (SWDATAH), but marks the second byte as end-of-data (last byte).

For HDR-DDR, the byte with the END must be even (second, fourth, sixth, and so on) because DDR uses byte pairs.

A halfword must not be written unless there is room for both, as indicated by the use of transmit FIFO level trigger or SDATACTRL[TXCOUNT].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 DATA1	Data 1 Indicates the second byte to be sent to the controller.
7-0 DATA0	Data 0 Indicates the first byte to be sent to the controller.

65.8.16 Target Read Data Byte (SRDATAB)

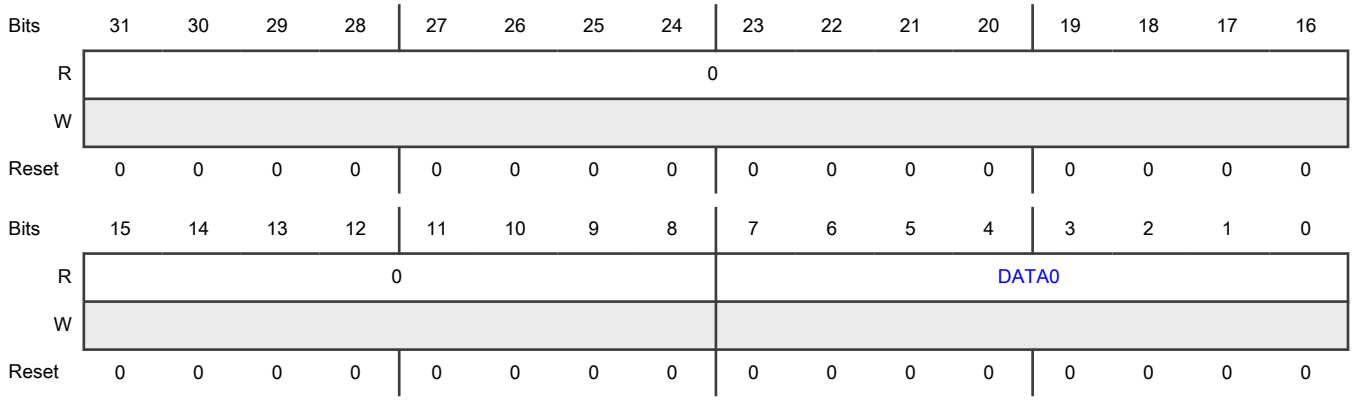
Offset

Register	Offset
SRDATAB	40h

Function

Allows reading a byte from the bus (controller). A byte must not be read unless there is data waiting, as indicated by SSTATUS[RX_PEND] = 1.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 DATA0	Data 0 Indicates the byte read from the controller.

65.8.17 Target Read Data Halfword (SRDATAH)

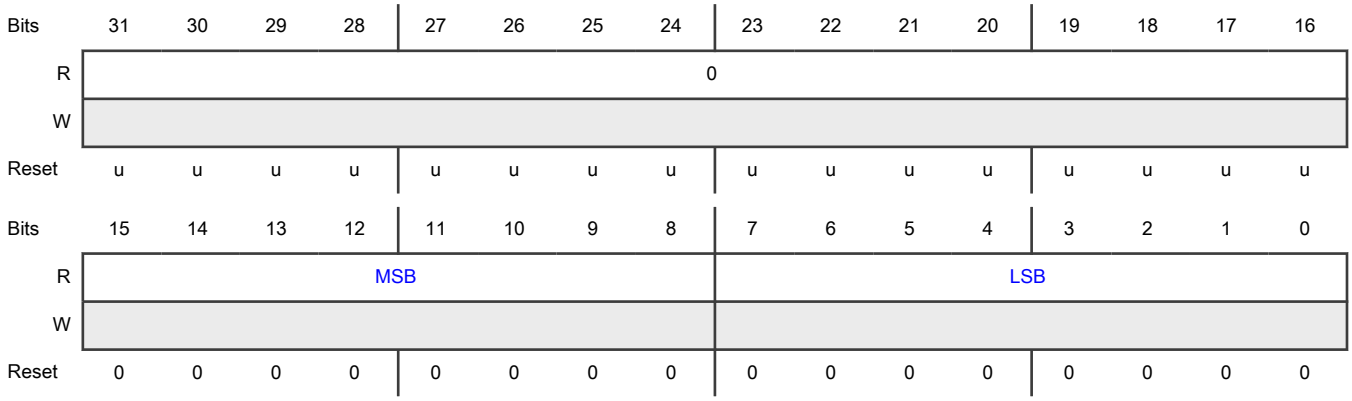
Offset

Register	Offset
SRDATAH	48h

Function

Allows reading a halfword (byte pair) written by the target after an SDR read or DAA or DDR. This register is used only when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively. A halfword must not be read unless there are at least two bytes of data waiting, as indicated by the use of receive FIFO level trigger or SDATACTRL[RXCOUNT].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 MSB	High Byte Indicates the second byte read from the target.
7-0 LSB	Low Byte Indicates the first byte read from the target.

65.8.18 Target Write Data Byte (SWDATAB1)

Offset

Register	Offset
SWDATAB1	54h

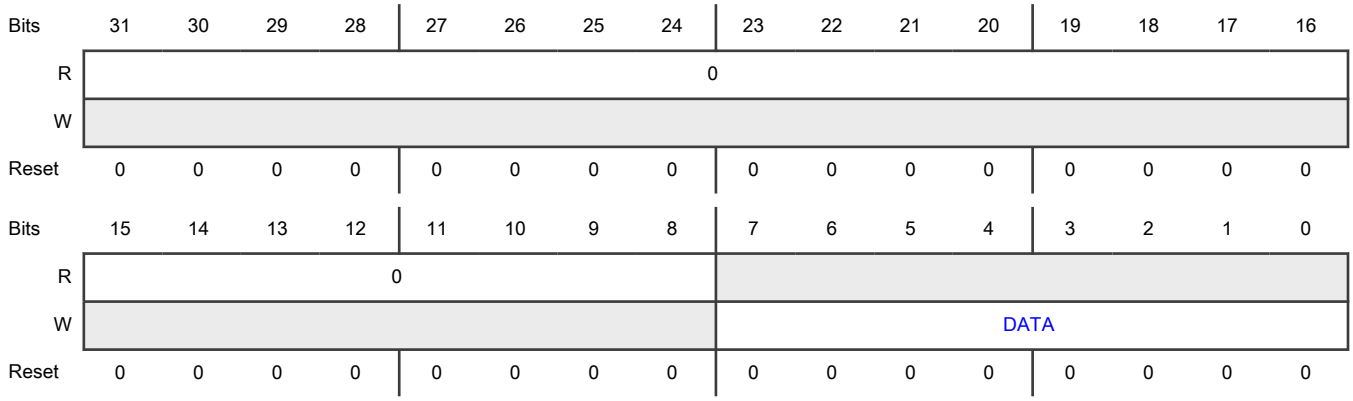
Function

Allows writing a single byte to the bus (to the controller) in a way that only bits 7:0 are used.

This register is intended to be used by DMAs (the upper bytes of the APB word are ignored).

A byte must not be written unless there is room, as indicated by SSTATUS[TXNOTFULL] = 1.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 DATA	Data Indicates the byte to be sent to the controller.

65.8.19 Target Capabilities 2 (SCAPABILITIES2)

Offset

Register	Offset
SCAPABILITIES2	5Ch

Function

Indicates which features are available and supported in this module, including controller and target capabilities, HDR modes, and others.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								SSTWR	SSTSUB	AASA	0	GROUP		SLVRST	V1_1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						IBIXREG	IBIEXT	0	I2CDEVID	Reserved	I2C10B	MAPCNT			
W																
Reset	0	0	0	0	0	0	1	1	0	0	1	1	0	0	0	1

Fields

Field	Function
31-24 —	Reserved
23 SSTWR	Target-Target(s)-Tunnel Write Capable Indicates whether target-target(s)-tunnel is write capable. 0b - Not write capable 1b - Write capable
22 SSTSUB	Target-Target(s)-Tunnel Subscriber Capable Indicates whether target-target(s)-tunnel is subscriber capable. 0b - Not subscriber capable 1b - Subscriber capable
21 AASA	SETAASA Supports the set static address as dynamic address CCC (SETAASA) feature. 0b - SETAASA not supported 1b - SETAASA supported
20 —	Reserved
19-18 GROUP	Group Indicates whether v1.1 group addressing is supported. Groups use mapping, so MAPCNT must be the same or greater than GROUP. 00b - v1.1 group addressing not supported 01b - One group supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Two groups supported 11b - Three groups supported
17 SLVRST	Target Reset Indicates whether v1.1 target reset is supported. 0b - Not supported 1b - Supported
16 V1_1	Version 1.1 Indicates whether I3C v1.1 GETCAPS is supported. 0b - Not supported 1b - Supported
15-10 —	Reserved
9 IBIXREG	In-Band Interrupt Extended Register Supports extended registers for IBIs. This field indicates whether Extended IBI Data 1 (IBIEXT1) is available. Extended IBI Data 2 (IBIEXT2) is available if IBIEXT1[MAX] > 3. 0b - Not supported 1b - Supported
8 IBIEXT	In-Band Interrupt EXTDATA Supports SCTRL[EXTDATA] to allow data beyond the mandatory data byte (MDB) for IBIs. 0b - Not supported 1b - Supported
7 —	Reserved
6 I2CDEVID	I2C Device ID Indicates whether I2C device ID is supported. 0b - Not supported 1b - Supported
5 —	Reserved
4	I2C 10-bit Address

Table continues on the next page...

Table continued from the previous page...

Field	Function
I2C10B	Supports 10-bit I2C address in MAP 1. If this field is 1, MAPCNT must be 1 or greater. 0b - Not supported 1b - Supported
3-0 MAPCNT	Map Count Indicates the number of maps that are allowed. If there is no mapping, this field is 0.

65.8.20 Target Capabilities (SCAPABILITIES)

Offset

Register	Offset
SCAPABILITIES	60h

Function

Indicates which features are available and supported in this module, including controller and/or target capabilities, HDR modes, and others.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DMA	INT	FIFORX		FIFOTX		EXTFIFO		0		TIMECTRL		IBI_MR_HJ			
W																
Reset	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CCCHANDLE				SADDR		MASTER	0	HDRSUPP		IDREG				IDENA	
W																
Reset	1	1	1	1	1	1	1	0	0	1	1	1	0	0	0	0

Fields

Field	Function
31 DMA	Direct Memory Access Indicates whether DMA is supported. 0b - Not supported 1b - Supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 INT	Interrupts Indicates whether interrupts are supported. 0b - Not supported 1b - Supported
29-28 FIFORX	FIFO Receive Indicates the size of receive (from-bus) FIFO. **NOTE** Controller and target use the same receive FIFO, so size is the same for the controller receive FIFO as well. FIFO size of SDR and HDR-DDR is in bytes. 00b - Two or three 01b - Four 10b - Eight 11b - 16 or larger
27-26 FIFOTX	FIFO Transmit Indicates the size of transmit (to-bus) FIFO. **NOTE** Controller and target use the same transmit FIFO, so size is the same for the controller transmit FIFO as well. FIFO size of SDR and HDR-DDR is in bytes. 00b - Two 01b - Four 10b - Eight 11b - 16 or larger
25-23 EXTFIFO	External FIFO Indicates whether external FIFOs are enabled. If they are not enabled, then check FIFOTX and FIFORX for the internal FIFO. 000b - No external FIFO available 001b - Standard available or free external FIFO 010b - Request track external FIFO All other values are reserved.
22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 TIMECTRL	Time Control Specifies whether any time-control type is supported. 0b - No time control supported 1b - At least one time-control type supported
20-16 IBI_MR_HJ	In-Band Interrupts, Controller Requests, Hot-Join Events Indicates which events (IBI, CR, and HJ) are allowed. For example, if this field is 00011b, IBI (bit 0) and IBI_HAS_DATA (bit 1) functionality are both enabled. 0_0000b - Application cannot generate IBI, CR, or HJ 1_xxxxb - Application can use SCONFIG[BAMATCH] for bus-available timing x_1xxxb - Application can generate a Hot-Join event x_x1xxb - Application can generate a controller request for a secondary controller x_xx1xb - When bit 0 = 1, the IBI has data from the SCTRL register x_xxx1b - Application can generate an IBI
15-12 CCCHANDLE	Common Command Codes Handling Indicates what manages CCC between I3C and the application. 0000b - All handling features disabled 1xxxb - GETSTATUS CCC returns the value of SCTRL[VENDINFO] x1xxb - GETSTATUS CCC returns the values of SCTRL[PENDINT] and SCTRL[ACTSTATE] xx1xb - The I3C module manages maximum read and write lengths, and max data speed xxx1b - The I3C module manages events, activities, status, HDR, and if enabled for it, ID and static-address-related items
11-10 SADDR	Static Address Indicates how the static address is managed. 00b - No static address 01b - Static address is fixed in hardware 10b - Hardware controls the static address dynamically (for example, from the pin strap) 11b - SCONFIG register supplies the static address
9 MASTER	Controller Specifies whether controller capability is supported. 0b - Not supported 1b - Supported
8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7-6 HDRSUPP	High Data Rate Support Indicates which HDR modes are supported. 00b - No HDR modes supported 01b - DDR mode supported All other values are reserved.
5-2 IDREG	ID Register Indicates which ID features are available in the configurable registers. 0000b - All ID register features disabled 1xxxb - A Bus Characteristics Register (BCR) is available x1xxb - A Device Characteristic Register (DCR) is available xx1xb - An ID Random field is available xxx1b - ID Instance is a register; used if there is no PARTNO register
1-0 IDENA	ID 48b Handler Indicates what handles the 48-bit ID value. 00b - Application 01b - Hardware 10b - Hardware, but the I3C module instance handles ID 48b 11b - A part number register (PARTNO)

65.8.21 Target Maximum Limits (SMAXLIMITS)

Offset

Register	Offset
SMAXLIMITS	68h

Function

Indicates the limits set by the controller (or the originally requested limits). The maximum limits are not enabled in the hardware design, including maximum read and write lengths. If the maximum read and write lengths are enabled, then the current setting (including the default request) shows up in this register (SMAXLIMITS).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				MAXWR											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				MAXRD											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-16 MAXWR	Maximum Write Length Indicates the maximum write length, which must be between 8 to 4095 (saturation). The application must not set the maximum write length to a higher value than the maximum write length set by the controller.
15-12 —	Reserved
11-0 MAXRD	Maximum Read Length Indicates the maximum read length, which must be between 16 to 4095 (saturation). The application must not set the maximum read length to a higher value than the maximum read length set by the controller.

65.8.22 Target ID Part Number (SIDPARTNO)

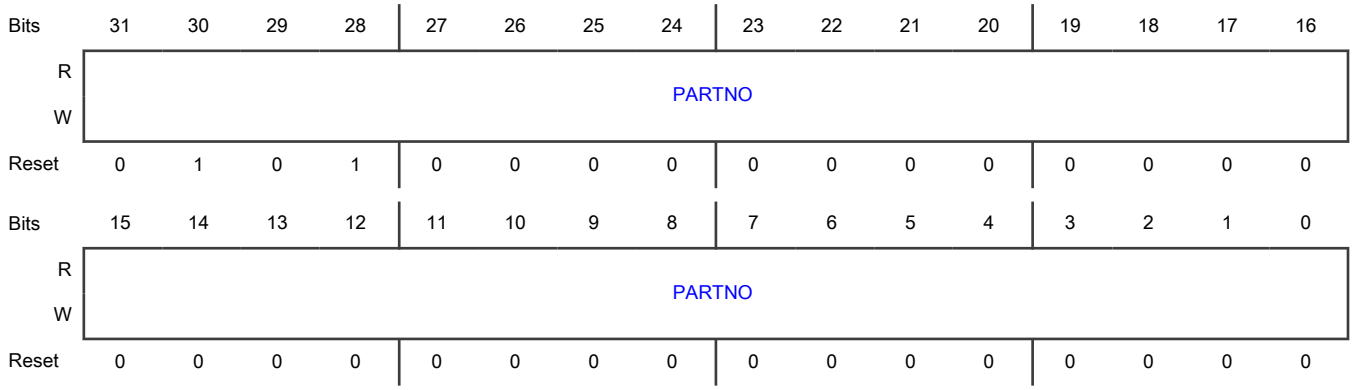
Offset

Register	Offset
SIDPARTNO	6Ch

Function

Allows you to write the ID part number. You must write a nonzero value into the PARTNO field because 0 is not valid.

Diagram



Fields

Field	Function
31-0 PARTNO	Part Number

65.8.23 Target ID Extension (SIDEXT)

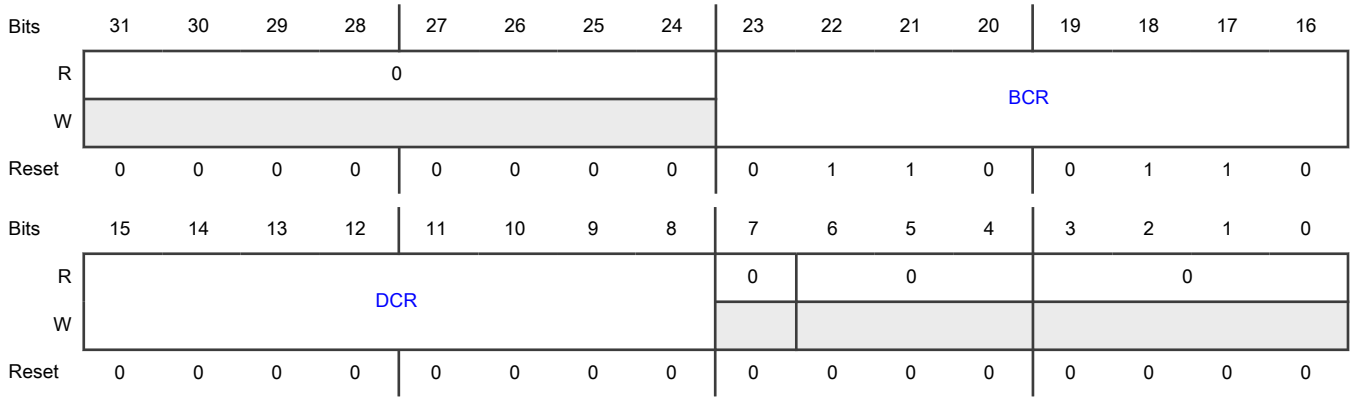
Offset

Register	Offset
SIDEXT	70h

Function

Allows you to write the ID extension of DCR and/or BCR.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 BCR	Bus Characteristics Register Sets the value for BCR, if this field is configured. Otherwise, the default value is considered. This field controls features such as secondary controller and slow-speed requirements.
15-8 DCR	Device Characteristic Register Sets the value for DCR, if this field is configured. Otherwise, the default value is considered.
7 —	Reserved
6-4 —	Reserved
3-0 —	Reserved

65.8.24 Target Vendor ID (SVENDORID)

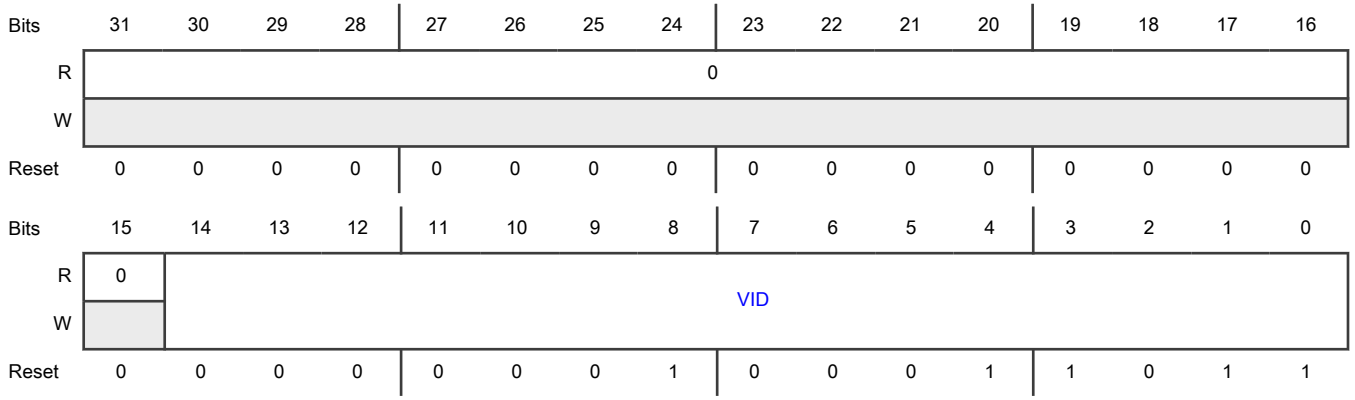
Offset

Register	Offset
SVENDORID	74h

Function

Allows you to write the vendor ID. The default value is the chip vendor ID, and is set from the constant field. When using the chip vendor ID, the part number (PARTNO) does not collide with other uses. The MIPI vendor ID is available to all companies (MIPI membership is not required). To get a vendor ID, make a request at the mipi.org website.

Diagram



Fields

Field	Function
31-15 —	Reserved
14-0 VID	Vendor ID Configures the 15-bit MIPI vendor ID. If not configured, the default value is considered.

65.8.25 Target Time Control Clock (STCCLOCK)

Offset

Register	Offset
STCCLOCK	78h

Function

Allows you to dynamically set the time control clock and accuracy information. The clock frequency and accuracy are constants set by the hardware. If the clock can be adjusted (that is, divided) or if the accuracy could vary with knowable information, then the clock may be set via this register, which must be updated whenever the clock source is changed.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FREQ								ACCURACY							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	1	0	1	0

Fields

Field	Function
31-16 —	Reserved
15-8 FREQ	Clock Frequency Indicates the clock frequency in 0.5 MHz steps. For example, a value of 20 in this field indicates a frequency of 10 MHz. Default set by parameters if configured.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 ACCURACY	Clock Accuracy Indicates the clock accuracy in 1/10ths of %. For example, a value of 15 indicates an accuracy of 1.5%. Default set by parameters if configured.

65.8.26 Target Message Map Address (SMSGMAPADDR)

Offset

Register	Offset
SMSGMAPADDR	7Ch

Function

Allows you to determine which address is matched when STATUS[MATCHED] = 1. This register is used when the DYNADDR register builds a list of extra DAs or SAs to match. It holds the last three matches.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												MAPLASTM2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				MAPLASTM1				0			LASTSTA...	MAPLAST			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19-16 MAPLASTM2	Matched Previous Index 2 Indicates index 2 of the previous matched address (0 for the base address).
15-12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-8 MAPLASTM1	Matched Previous Address Index 1 Indicates index 1 of the previous matched address (0 for the base address).
7-5 —	Reserved
4 LASTSTATIC	Last Static Address Matched Indicates whether the last matched address was an I2C static address or an I3C dynamic address. 0b - I3C dynamic address 1b - I2C static address
3-0 MAPLAST	Matched Address Index Indicates the matched address index for current or last matched message (0 for the base address).

65.8.27 Controller Control (MCTRL)

Offset

Register	Offset
MCTRL	84h

Function

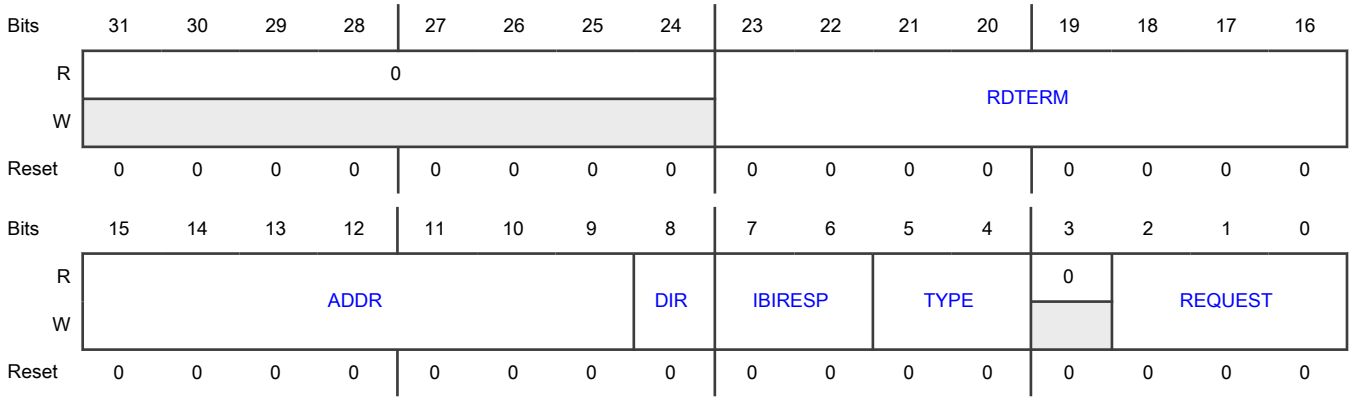
Starts activities on the I3C or I2C bus (see the MWMSG registers). A request cannot be changed when a message is in progress; the REQUEST field becomes 0 automatically.

You must write to MCTRL fields as per use case or as mentioned in Operating modes. Bit read and write checking may not work independently.

NOTE

If MCONFIG[MSTENA] is configured for I2C Controller mode (legacy I2C), only REQUEST = 1 and REQUEST = 2 are accepted. Also, fields are constrained to I2C-supported fields.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 RDTERM	Read Terminate Counter Determines when to terminate a read operation: • For I2C, this field controls when to NACK a read. • For I3C, this field can be used to terminate (end) a read: — If RDTERM is 0, it has no effect. — If RDTERM is 1, the read terminates after the next character. — If RDTERM is 2, the read terminates after the next two characters. The field supports up to 255 characters. In DDR mode, RDTERM terminates the read based on word counts (for DDR) instead of byte counts (for SDR). The field self-clears when MSTATUS[COMPLETE] becomes 1. **NOTE** You may write 1 to this field anytime, but a value greater than 1 must be written when starting EmitStartAddress.
15-9 ADDR	Address Indicates the type of address: I3C dynamic address or I2C static address. Some values are not allowed based on the bus that is used (I3C or I2C).
8 DIR	Direction Indicates the direction, write or read. 0b - Write 1b - Read
7-6	In-Band Interrupt Response

Table continues on the next page...

Table continued from the previous page...

Field	Function
IBIRESP	Indicates the response to use when you get an IBI from START, and when to force using an IBI ACK NACK request when completing a manual IBI. Completion of a manual IBI means that the target DA is known, and so the mandatory byte (or not) is specified by the application when acknowledging. This field is also used when a message is emitted in Message mode using the MWMSG_SDR or MWMSG_DDR registers. If an IBI with MDB (mandatory byte) is ACKed, the controller limits it to a maximum of eight more bytes after the MDB. RDTERM = 1 can be used with REQUEST = NONE to terminate data from a target sooner. If this field is 0b, it indicates ACK (acknowledge). When REQUEST = 1 (EmitStartAddr) or REQUEST = 7 (AutoIBI), Controller In-band Interrupt Registry and Rules (MIBIRULES) decides whether to ACK with mandatory byte. When REQUEST = 3 (IBIAckNack), ACK with no mandatory byte. If this field is 1b, it indicates NACK (reject) or no acknowledgement. If this field is 10b, it indicates acknowledge with mandatory byte. When REQUEST = 1 or REQUEST = 7, ignore Controller In-band Interrupt Registry and Rules (MIBIRULES). Do not use this setting unless only targets with a mandatory byte can cause an IBI. When REQUEST = 3, ACK with mandatory byte. If this field is 11b, it indicates manual. When REQUEST = 1 or REQUEST = 7, stop and wait for a decision using the IBI ACK NACK request. When REQUEST = 3, this field is reserved. **NOTE** A CR and HJ always cause IBIRESP to become 3 (manual) so the application must decide. 00b - ACK (acknowledge) 01b - NACK (reject) 10b - Acknowledge with mandatory byte 11b - Manual
5-4 TYPE	Bus Type with EmitStartAddr Works with REQUEST to determine the bus type. See Determining bus types and modes for more information. The following values are valid only if REQUEST is 1. 00b - I3C 01b - I2C 10b - DDR 11b - Reserved
3 —	Reserved
2-0 REQUEST	Request Emits the requested operation when performing in pieces, instead of performing by message. You must check Controller Status (MSTATUS) because some requests can only be made in some states. For

Table continues on the next page...

Table continued from the previous page...

Field	Function
	example, the system cannot enter SDR mode from DDR mode, and it cannot use an incorrect request in DAA mode. If this field is 0b, it indicates that no request is present (NONE). The REQUEST field returns to NONE after finishing a request. Controller Status (MSTATUS) indicates the state of the controller. See AutoIBI mode. NONE is written as 0 only in these cases: when writing 1 to MCTRL[RDTERM] (to stop a read in progress) or when configuring MCTRL[IBIRESP] for MSG use. If this field is 1b, it emits START with address and direction (EMITSTARTADDR), either from Stopped state or in the middle of a single data rate (SDR) message. If from Stopped state (Idle), then EmitStart may be prevented by an event (such as IBI, CR, HJ). In this case, an appropriate interrupt is signaled. EmitStart can be resubmitted. If this field is 10b, it emits a STOP on bus (EMITSTOP). Must be in SDR mode. In DAA mode, emitting STOP exits DAA mode. If this field is 11b, it indicates manual IBI ACK or NACK (IBIACKNACK). When MCTRL[IBIRESP] indicates a hold on an IBI to allow a manual decision, this request completes it. The field uses MCTRL[IBIRESP] to provide the information. If this field is 100b, and if currently not in DAA mode, it emits START, 7E, ENTDAA sequence, and then emits 7E/R to process the first target (PROCESSDAA). Stops just before the new dynamic address (DA) is to be emitted. The DA is written using Controller Write Data Byte (MWDATAB), then process DAA is requested again to write the new address, and then it starts the next unless marked to STOP. An MSTATUS indicating NACK means DA was not accepted (for example, parity error). If PROCESSDAA is NACKed on the 7E/R request, meaning no more targets need a DA, then a COMPLETE is signaled (along with DONE) and a STOP issued. If this field is 110b, it emits an exit pattern from any state (force exit and target reset). End DDR mode (including MSGDDR), including a STOP afterward. If this field is 111b, the target is held in a Stopped state, but autoemits a START, 7E sequence when the target holds SDA low for an IBI (AUTOIBI). Actual IBI handling is defined by MCTRL[IBIRESP]. 000b - NONE 001b - EMITSTARTADDR 010b - EMITSTOP 011b - IBIACKNACK 100b - PROCESSDAA 101b - Reserved 110b - Force Exit and Target Reset 111b - AUTOIBI

65.8.28 Controller Status (MSTATUS)

Offset

Register	Offset
MSTATUS	88h

Function

Indicates the controller status, including which events cause interrupts. The peripherals share the IRQ (called parallel-to-target status).

Because a peripheral can either be in Controller or Target mode, but not both at the same time, only one (target or controller peripheral) can cause the IRQ. If there is an IRQ and the peripheral is a controller, then this register (MSTATUS) indicates the status.

If there is an IRQ and the peripheral is a target, then Target Status (SSTATUS) indicates the status.

This register self-clears if the COMPLETE field becomes 1.

If the MCONFIG register is configured only for I2C, some states and fields never remain active (for example, DAA or IBI).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	IBIADDR							0				NOWMAST...	0		
W													W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRWARN	0	IBIWON	TXNOTFU...	RXPEND	COMPLETE	MCTRLDO...	SLVSTART	IBITYPE		NACKED	BETWEEN	0	STATE		
W			W1C			W1C	W1C	W1C								
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-24 IBIADDR	IBI Address Indicates the address of: • The IBI when MSTATUS[IBITYPE] = 1. • The CR, when MSTATUS[IBITYPE] = 2. • 7'h2 when HJ, when MSTATUS[IBITYPE] = 3.
23-20 —	Reserved
19 NOWMASTER	Module is now Controller Flag Indicates whether the module is now a controller. That is, it was previously a target, and controllership acceptance was requested from the previous controller and controllership was accepted. The reverse operation (controller becomes a target) does not need an interrupt because the application grants it through the GETACCMST CCC.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Not a controller 1b - Controller When writing 0b - No effect 1b - Clear the flag
18-16 —	Reserved
15 ERRWARN	Error or Warning Indicates whether an error occurred, such as improper register use, overrun, or underrun of FIFO or buffer, or invalid parity or CRC in a DDR read. See Controller Errors and Warnings (MERRWARN) for more information. 0b - No error or warning 1b - Error or warning
14 —	Reserved
13 IBIWON	In-Band Interrupt (IBI) Won Flag Indicates whether an IBI, CR, or HJ has won the arbitration on a header address, regardless of whether it was NACKed or ACKed. Arbitration requires manual intervention for CR and HJ, and optionally requires it for IBI if MCTRL[IBIRESP] = 3 (manual). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No IBI arbitration won 1b - IBI arbitration won When writing 0b - No effect 1b - Clear the flag
12 TXNOTFULL	TX Buffer or FIFO Not Full

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates whether the buffer, FIFOm, or message register can accept another byte or halfword. FIFO uses trigger level. If DMA is enabled for transmitting, it transfers data as long as it is not full. 0b - Receive buffer or FIFO full 1b - Receive buffer or FIFO not full
11 RXPEND	RXPEND Indicates whether a message is being received from a target and bytes are in the input buffer or FIFO: • When using a FIFO, this message is at least one FIFO trigger's worth (a minimum of one byte in the FIFO). • When DMA is enabled for receiving, the DMA is signaled. RXPEND becomes 0 when the data is read. 0b - No receive message pending 1b - Receive message pending
10 COMPLETE	Complete Flag Indicates whether a message is complete: • With MWMSG_SDR or MWMSG_DDR, the COMPLETE condition occurs when MWMSG_SDR_CONTROL[LEN] or MWMSG_DDR_CONTROL[LEN] reaches 0. • When MCTRL[REQUEST] = 1 (EmitStartAddr), this COMPLETE condition occurs after the end of a write operation, or after a read operation has terminated or ended. • With an IBI (AutoIBI or EmitStartAddr), the COMPLETE condition occurs at the end of IBI data (if any). **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clear the flag
9 MCTRLDONE	Controller Control Done Flag Indicates whether the module has completed an MCTRL request. MCTRLDONE automatically becomes 0 when writing a new control. If MCTRL[REQUEST] = 1 (EmitStartAddr), MCTRLDONE becomes 1 when the address goes out (and is ACKed, NACKed, or ended in an IBI). If ACKed, MSTATUS[COMPLETE] becomes 1 after the write or read data is complete.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If MCTRL[REQUEST] = 4 (ProcessDAA), MCTRLDONE becomes 1 when the module is ready to emit the dynamic address (DA) for the target, or when no more targets are ACKing. This condition can be determined by using the MSTATUS[BETWEEN] and MSTATUS[STATE] fields. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not done 1b - Done When writing 0b - No effect 1b - Clear the flag
8 SLVSTART	Target Start Flag Indicates whether a target is or was requesting a START by holding SDA low. Handling starts automatically when MCTRL[REQUEST] = 7 (AutoIBI). **NOTE** This field behaves differently for register reads and writes. When reading 0b - Target not requesting START 1b - Target requesting START When writing 0b - No effect 1b - Clear the flag
7-6 IBITYPE	In-Band Interrupt (IBI) Type Indicates the type of IBI of the last event that won the arbitration, whether the interrupt was ACKed, NACKed, or pending. 00b - NONE (no IBI: this status occurs when MSTATUS[IBIWON] becomes 0) 01b - IBI 10b - CR 11b - HJ
5 NACKED	Not Acknowledged Indicates whether the last start and address sequence was NACKed (was not ACKed by the addressed target). 0b - Not NACKed 1b - NACKed (not acknowledged)

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 BETWEEN	Between Indicates whether the controller is active between messages or dynamic address assignments (DAA). It is active when: • MSTATUS[STATE] is MSGSDR, DDR, or DAA, or NORMACT, and the state is between messages or DAAs. It is expecting a new message or DAA to start (or STOP or exit). • MSTATUS[STATE] is NORMACT. The module is waiting for the transmit FIFO to be not empty or the receive FIFO to be not full. 0b - Inactive (for other cases) 1b - Active
3 —	Reserved
2-0 STATE	State of the Controller Indicates the current controller state. If this field is 0b, it indicates that the controller is idle (the bus has stopped.) If this field is 1b, it indicates target request. The bus has stopped but a target is holding SDA low. When using auto-emit IBI (MCTRL[REQUEST] = 7), the controller does not remain in this state. If this field is 10b, it indicates entry into Single Data Rate Message mode, using MWMSG_SDR. If this field is 11b, it indicates entry into normal active SDR mode, using MCTRL, MWDATA*n*, and MRDATA*n* registers. The controller remains in this state until a STOP is issued. If this field is 100b, it indicates entry into Double Data Rate Message mode, using MWMSG_DDR or the normal method with DDR. The controller remains in the DDR state until the controller exits using EXIT (emitting the Exit pattern). If this field is 101b, it indicates entry into Dynamic Address Assignment (ENTDAA) mode. If this field is 110b, it indicates wait for an IBI ACK/NACK decision. If this field is 111b, it indicates receiving an IBI. This state is used after IBI, CR, or HJ has won an arbitration. The IBIRCV state is also used for IBI mandatory byte (if any) and any bytes that follow. 000b - IDLE (bus has stopped) 001b - SLVREQ (target request) 010b - MSGSDR 011b - NORMACT 100b - MSGDDR 101b - DAA 110b - IBIACK 111b - IBIRCV

65.8.29 Controller In-band Interrupt Registry and Rules (MIBIRULES)

Offset

Register	Offset
MIBIRULES	8Ch

Function

Contains the rules for using IBI, and keeps a registry of the targets that use the IBI byte.

This register defines the IBI mandatory byte rules for the targets and determines which targets do (or do not) have a mandatory byte.

Concerning ADDR*n* fields: The address has 6 bits. Assuming that MIBIRULES[MSB0] = 1, the most significant bit of each address is 0. In this case, each address is 7 bits (usually written as A7 to A1) and A7 must be 0. If the application does not use that optimal convention, the "manual" method of IBI ACK handling must be used (see MCTRL[IBIRESP] for more information).

By default, the ADDR*n* values indicate the targets with a mandatory byte. If MIBIRULES[NOBYTE] = 1, the ADDR*n* values indicate targets that do not have a mandatory IBI byte.

NOTE

A7 = 0 is only needed for targets that use IBI. For legacy I2C devices, A7 can use any valid value because I2C devices cannot cause an IBI.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NOBYTE	MSB0	ADDR4						ADDR3				ADDR2			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR2				ADDR1				ADDR0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 NOBYTE	No IBI byte Specifies whether the ADDR*n* fields refer to targets with or without a mandatory IBI byte. 0b - With mandatory IBI byte 1b - Without mandatory IBI byte
30 MSB0	Most Significant Address Bit is 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Specifies whether MSB is 0 for all I3C dynamic addresses. Assigning 1 to this field allows the START header to be optimized. 0b - MSB is not 0 1b - MSB is 0
29-24 ADDR4	ADDR4 Specifies the address of the target with or without mandatory IBI byte. If 0, then the address does not apply.
23-18 ADDR3	ADDR3 Specifies the address of the target with or without mandatory IBI byte. If 0, then the address does not apply.
17-12 ADDR2	ADDR2 Specifies the address of the target with or without mandatory IBI byte. If 0, then the address does not apply.
11-6 ADDR1	ADDR1 Specifies the address of the target with or without mandatory IBI byte. If 0, then the address does not apply.
5-0 ADDR0	ADDR0 Specifies the address of the target with or without mandatory IBI byte. If 0, then the address does not apply.

65.8.30 Controller Interrupt Set (MINTSET)

Offset

Register	Offset
MINTSET	90h

Function

Returns the status of the interrupt enables when read. This register sets interrupt enables for select fields in Controller Status (MSTATUS):

- To activate an interrupt enable, write 1 to its corresponding field.
- To disable an interrupt enable, write 1 to the appropriate field in Controller Interrupt Clear (MINTCLR). Writing 0 to the interrupt enable in MINTSET does not disable the interrupt.

The interrupt registers allow masking of interrupt sources as well as checking which interrupts are activated in Controller Status (MSTATUS). The normal method is to enable an interrupt and then after that interrupt fires, clear the interrupt either by writing to Controller Status (MSTATUS) or via an action on the corresponding data register. The interrupt is level-held, meaning that the interrupt remains 1 until the cause is cleared by some method. The module prevents races; if a new event comes in, that new event is not lost.

These interrupts are parallel to Target Interrupt Set (SINTSET) and Target Interrupt Clear (SINTCLR), and only one interrupt set is active depending on the state (controller or target operation):

- MINTSET: Sets interrupt enables for MSTATUS fields. Reading the MINTSET register returns the status of the interrupt enables.
- MINTCLR: Clears interrupt enables for MSTATUS fields.
- MINTMASKED: Returns the value of the MSTATUS fields ANDed with their interrupt enables.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												NOWMAST...	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRWARN	0	IBIWON	TXNOTFU...	RXPEND	COMPLETE	MCTRLDO...	SLVSTART	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 NOWMASTER	Now Controller Interrupt Enable Enables the corresponding interrupt to indicate whether the module is now a controller (now this I3C module is a controller). That is, it was previously a target; controllership acceptance was requested from the previous controller and it was accepted. 0b - Disable 1b - Enable
18-16 —	Reserved
15 ERRWARN	Error or Warning (ERRWARN) Interrupt Enable Enables the corresponding interrupt to indicate whether an error occurred, such as improper register use, overrun or underrun of FIFO or buffer, or invalid parity or CRC in a DDR read. 0b - Disable 1b - Enable
14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 IBIWON	IBI Won Interrupt Enable Enables the corresponding interrupt to indicate whether an IBI, CR, or HJ has won the arbitration on a header address, regardless of whether it was NACKed or ACKed. 0b - Disable 1b - Enable
12 TXNOTFULL	Transmit Buffer/FIFO Not Full Interrupt Enable Enables the corresponding interrupt to indicate whether the buffer, FIFOm, or message register can accept another byte or halfword. 0b - Disable 1b - Enable
11 RXPEND	Receive Pending Interrupt Enable Enables the corresponding interrupt to indicate whether a message is being received from a target and bytes are in the input buffer or FIFO.
10 COMPLETE	Completed Message Interrupt Enable Enables the corresponding interrupt to indicate whether a message is complete. 0b - Disable 1b - Enable
9 MCTRLDONE	Controller Control Done Interrupt Enable Enables the corresponding interrupt to indicate whether the module has completed an MCTRL request. 0b - Disable 1b - Enable
8 SLVSTART	Target Start Interrupt Enable Enables the corresponding interrupt to indicate whether a target is or was requesting a START by holding SDA low. 0b - Disable 1b - Enable
7-0 —	Reserved

65.8.31 Controller Interrupt Clear (MINTCLR)

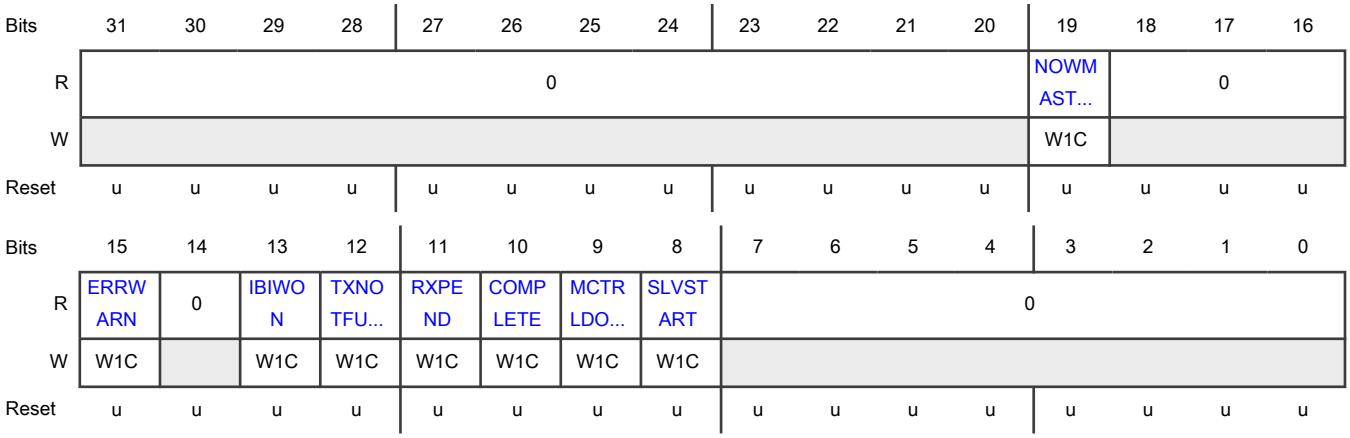
Offset

Register	Offset
MINTCLR	94h

Function

Clears interrupt enables for select fields in Controller Status (MSTATUS). Writing 1 clears the corresponding interrupt enable. Writing 0 has no effect.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 NOWMASTER	NOWCONTROLLER Interrupt Enable Clear Flag Clears the corresponding NOWCONTROLLER interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 —	Reserved
15 ERRWARN	ERRWARN Interrupt Enable Clear Flag Clears the corresponding ERRWARN interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag
14 —	Reserved
13 IBIWON	IBIWON Interrupt Enable Clear Flag Clears the corresponding IBIWON interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag
12 TXNOTFULL	TXNOTFULL Interrupt Enable Clear Flag Clears the corresponding TXNOTFULL interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
11 RXPEND	RXPEND Interrupt Enable Clear Flag Clears the corresponding RXPEND interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag
10 COMPLETE	COMPLETE Interrupt Enable Clear Flag Clears the corresponding COMPLETE interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag
9 MCTRLDONE	MCTRLDONE Interrupt Enable Clear Flag Clears the corresponding MCTRLDONE interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 SLVSTART	SLVSTART Interrupt Enable Clear Flag Clears the corresponding SLVSTART interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag
7-0 —	Reserved

65.8.32 Controller Interrupt Mask (MINTMASKED)

Offset

Register	Offset
MINTMASKED	98h

Function

Returns the status of enabled interrupts (the value of Controller Status (MSTATUS) ANDed with the value of Controller Interrupt Set (MINTSET)).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												NOWMAST...	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRWARN	0	IBIWON	TXNOTFU...	RXPEND	COMPLETE	MCTRLDO...	SLVSTART	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 NOWMASTER	NOWCONTROLLER Interrupt Mask Indicates whether the NOWCONTROLLER interrupt is enabled and active. 0b - Disabled 1b - Enabled
18-16 —	Reserved
15 ERRWARN	ERRWARN Interrupt Mask Indicates whether the ERRWARN interrupt is enabled and active. 0b - Disabled 1b - Enabled
14 —	Reserved
13 IBIWON	IBIWON Interrupt Mask Indicates whether the IBIWON interrupt is enabled and active. 0b - Disabled 1b - Enabled
12 TXNOTFULL	TXNOTFULL Interrupt Mask Indicates whether the TXNOTFULL interrupt is enabled and active. 0b - Disabled 1b - Enabled
11 RXPEND	RXPEND Interrupt Mask Indicates whether the RXPEND interrupt is enabled and active.
10 COMPLETE	COMPLETE Interrupt Mask Indicates whether the COMPLETE interrupt is enabled and active. 0b - Disabled 1b - Enabled
9 MCTRLDONE	MCTRLDONE Interrupt Mask Indicates whether the MCTRLDONE interrupt is enabled and active.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
8 SLVSTART	SLVSTART Interrupt Mask Indicates whether the SLVSTART interrupt is enabled and active. 0b - Disabled 1b - Enabled
7-0 —	Reserved

65.8.33 Controller Errors and Warnings (MERRWARN)

Offset

Register	Offset
MERRWARN	9Ch

Function

Contains errors and warnings. When any errors or warnings are not 0, then MSTATUS[ERRWARN] is 1.

Parallel-to-target ERRWARN:

- In Controller mode, use this register (MERRWARN).
- In Target mode, use Target Errors and Warnings (SERRWARN).

The error fields in both registers (MERRWARN and SERRWARN) are similar in meaning.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0										0	TIMEOUT	INVREQ	MSGERR	OWRITE	OREAD
W												W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			0	0	HCRC	HPAR	0				Reserved	WRABT	NACK	0	0
W						W1C	W1C					W1C	W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-22 —	Reserved
21 —	Reserved
20 TIMEOUT	Timeout Error Flag Indicates an error caused by the module stalling for too long in a frame. This stalling occurs when: • The transmit FIFO or receive FIFO is not handled, and the bus is stuck in the middle of a message. • No STOP is issued after data transfer (there is a gap between messages). • IBI manual is used and no decision has been made. The maximum stall period is 10 kHz or 100 µs. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
19 INVREQ	Invalid Request Error Flag Indicates an error caused by an invalid use of a request: • Not using IBI ACK NACK when stopped in manual hold for IBI acknowledgment. • Using a request other than ForceStop or ForceExit when in a message. Other requests are valid when the message is done. • Other mismatched uses (for example, IBI ACK NACK in normal states). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 MSGERR	Message Error Flag Indicates an error caused by: • Trying to write to or read from the MWMSG_SDR register when in a DDR message. • Trying to write to or read from the MWMSG_DDR register when in an SDR message. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
17 OWRITE	Overwrite Error Flag Indicates an error caused by trying to write to Controller Write Data Byte (MWDATAB) when the FIFO is full. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
16 OREAD	Overread Error Flag Indicates an error caused by: • Trying to read from Controller Read Data Byte (MRDATAB) when the FIFO is empty. • Trying to read from the MRMSG_SDR register or the MRMSG_DDR register when no message has yet started. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Error When writing 0b - No effect 1b - Clear the flag
15-13 —	Reserved
12 —	Reserved
11 —	Reserved
10 HCRC	High Data Rate CRC Error Flag Indicates that a cyclic redundancy check (CRC) error occurred from a DDR read. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
9 HPAR	High Data Rate Parity Flag Indicates a parity error from a DDR read, including a bad preamble on a read. This does not stop the read because it is not safe to terminate; the read data may become misframed. Ends on a run of 1 second. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
8-5 —	Reserved
4 —	Reserved
3 WRABT	Write Abort Error Flag Indicates an error caused by the I2C target NACKing the write data, terminating the message. For example, the controller is writing in I2C and the target NACKed the write. If you write to the MCTRL register, this field automatically becomes 0. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
2 NACK	Not Acknowledge Error Flag Indicates an error caused by the target or targets NACKing (not acknowledging) the last address. If 7Eh is the address, then it indicates all targets NACKed the last address. If you write to the MCTRL register, this field automatically becomes 0. **NOTE** In HDR mode, this error occurs when an address is not accepted (as opposed to NACKed). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 —	Reserved

65.8.34 Controller DMA Control (MDMACTRL)

Offset

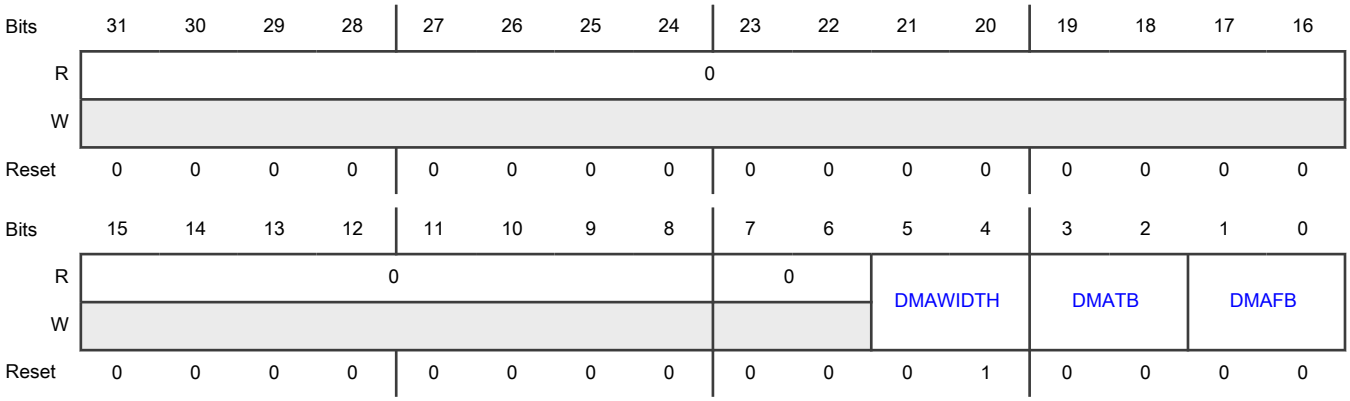
Register	Offset
MDMACTRL	A0h

Function

DMA can be used with a controller in these ways:

- Push or pull data for MCTRL[REQUEST] = 1 (EmitStartAddr) request written by the processor.
- Implementing message mode, completely controlled by DMA.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-6 —	Reserved
5-4 DMAWIDTH	DMA Width Specifies the data width of DMA operations.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The halfword (16 bits) setting ensures that two bytes are free or available in FIFO. 00b,01b - Byte 10b - Halfword (16 bits) 11b - Reserved
3-2 DMATB	DMA to Bus Represents the DMA write (to-bus) trigger. When DMAFB = 1 or DMAFB = 2, I3C requests DMA when a transmit trigger occurs (see Controller Data Control (MDATACTRL)). I3C requests until full, unless the DMA is set up as a trigger. DMAFB becomes 0 when MSTATUS[ERRWARN] = 1. If this field is 1b, STOP or START causes DMATB to become 0 (see SCONFIG[MATCHSS]). Normally, DMA enable must be used only in Controller Message mode. 00b - DMA not used 01b - Enable DMA for one frame (ended by DMA or terminated) 10b - Enable DMA until DMA is turned off 11b - Reserved
1-0 DMAFB	DMA from Bus Represents the DMA read (from-bus) trigger. When DMAFB = 1 or DMAFB = 2, I3C requests DMA when a receive trigger occurs (see Controller Data Control (MDATACTRL)). I3C requests until empty unless the DMA is set up as a trigger. DMAFB becomes 0 when MSTATUS[ERRWARN] = 1. If this field is 1b, STOP or Repeated START causes DMAFB to automatically become 0 (see SCONFIG[MATCHSS]). **NOTE** Do not enable DMA after a transaction starts. It must be enabled only when enabling the controller and interrupts to avoid any kind of RX FIFO overrun. 00b - DMA not used 01b - Enable DMA for one frame 10b - Enable DMA until DMA is turned off 11b - Reserved

65.8.35 Controller Data Control (MDATACTRL)

Offset

Register	Offset
MDATACTRL	ACh

Function

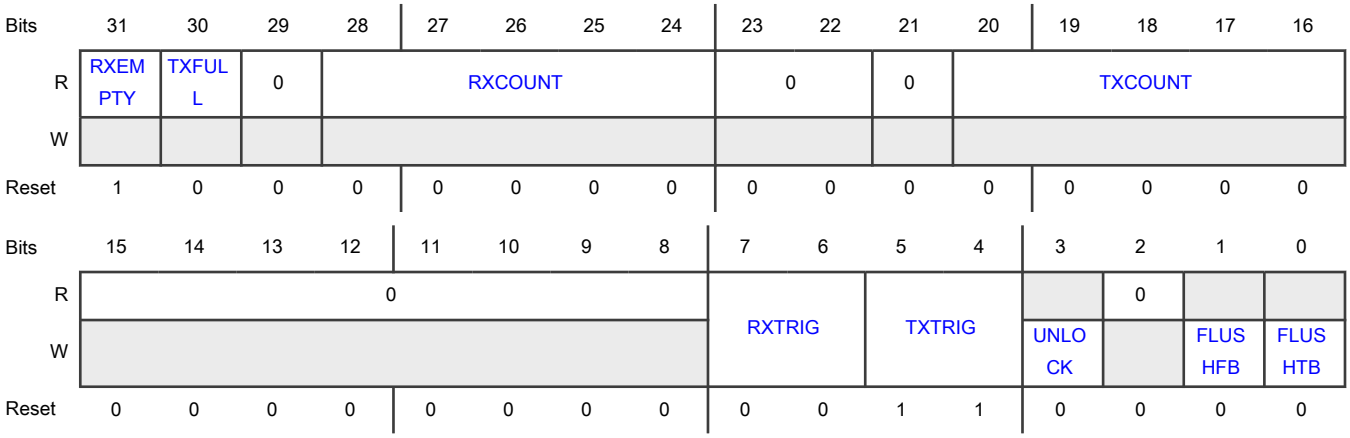
Assists in data control when there is no FIFO. This register also assists the FIFO when FIFO is available (regardless of size), and this assistance allows some control over the FIFO behavior. In particular, the register provides control over when to interrupt when a particular state of fullness or emptiness is reached. It also controls behavior related to width, when the width is not 1 byte wide.

This register acts as an alias of Target Data Control (SDATACTRL).

NOTE

When flushing a FIFO if DMA is in use, disable the DMA channel first. You must not use FIFO flush when a message (in that direction) is in flight.

Diagram



Fields

Field	Function
31 RXEMPTY	Receive is Empty Indicates whether the receive FIFO or buffer is empty. 0b - Not empty 1b - Empty
30 TXFULL	Transmit is Full Indicates whether the transmit FIFO or buffer is full. 0b - Not full 1b - Full
29 —	Reserved
28-24 RXCOUNT	Receive Byte Count Contains the count of bytes in the receive FIFO or buffer.
23-22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21 —	Reserved
20-16 TXCOUNT	Transmit Byte Count Contains the count of bytes waiting in the TXFIFO. This count is the number of bytes that the application has written to the transmit FIFO that have not yet gone onto the bus.
15-8 —	Reserved
7-6 RXTRIG	Receive Trigger Level Indicates the trigger level for receive fullness when using a FIFO. This field affects the RXPEND interrupt. 00b - Trigger when not empty 01b - Trigger when 1/4 full or more 10b - Trigger when 1/2 full or more 11b - Trigger when 3/4 full or more
5-4 TXTRIG	Transmit Trigger Level Indicates the trigger level for transmit emptiness when using a FIFO. This field affects the TXNOTFULL interrupt. 00b - Trigger when empty 01b - Trigger when 1/4 full or less 10b - Trigger when 1/2 full or less 11b - Trigger when 1 less than full or less (default)
3 UNLOCK	Unlock Unlocks the functionality so that the RXTRIG and TXTRIG fields can be changed on a write. This field must be 1 (unlocked) in the same cycle when writing to TXTRIG or RXTRIG. If this field is 0 (locked), RXTRIG and TXTRIG fields cannot be changed on a write. 0b - Locked 1b - Unlocked
2 —	Reserved
1 FLUSHFB	Flush From-Bus Buffer or FIFO Flushes from-bus buffer or FIFO. You normally do not use this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No action 1b - Flush the buffer
0 FLUSHTB	Flush To-Bus Buffer or FIFO Flushes to-bus buffer or FIFO. This field is used when the controller terminates a to-bus message (read) prematurely. 0b - No action 1b - Flush the buffer

65.8.36 Controller Write Data Byte (MWDATAB)

Offset

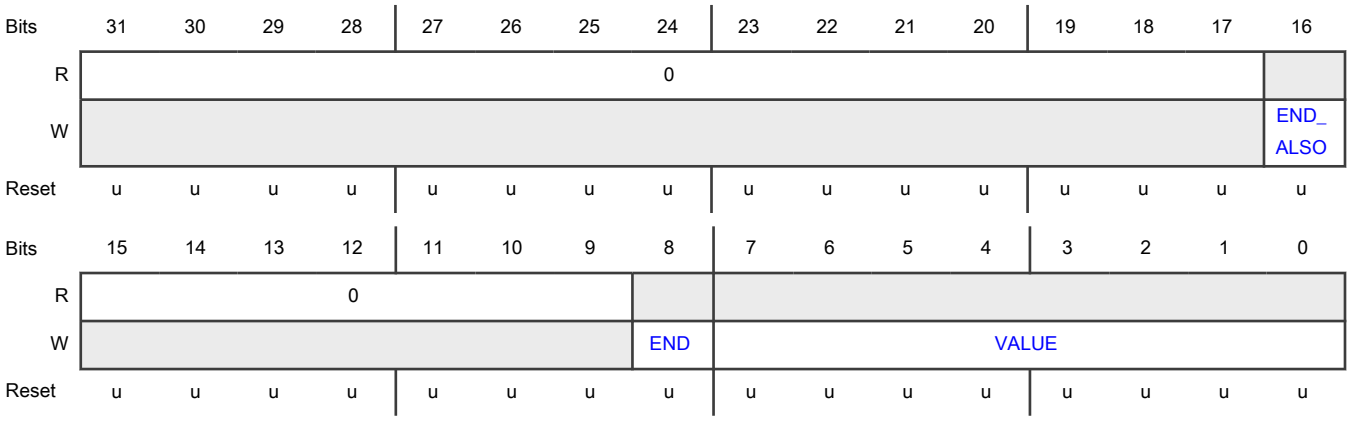
Register	Offset
MWDATAB	B0h

Function

Allows writing bytes to send onto the bus. This register is used only when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

This register acts as an alias of Target Write Data Byte (SWDATAB).

Diagram



Fields

Field	Function
31-17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 END_ALSO	End of Message ALSO Indicates end of message, used to end an outbound message normally. Every message must indicate when it is the last message to be sent. This method can be used with the MDATABE register. This field is required for I3C and is optional for I2C. For HDR-DDR, the byte with END_ALSO must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. If this field is 0, more bytes are assumed to be in the message. If this field is 1, the END bit marks the last byte of the message. 0b - Not the end 1b - End
15-9 —	Reserved
8 END	End of Message Indicates the end of a message. This field is required for I3C and is optional for I2C. For HDR-DDR, the byte with the END must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. If this field is 0, more bytes are assumed to be in the message. If this field is 1, the END bit marks the last byte of the message. 0b - Not the end 1b - End
7-0 VALUE	Data Byte Represents the byte written to the target (stored in transmit FIFO): • I3C computes the parity. • I2C manages the ACK or NACK.

65.8.37 Controller Write Data Byte End (MWDATABE)

Offset

Register	Offset
MWDATABE	B4h

Function

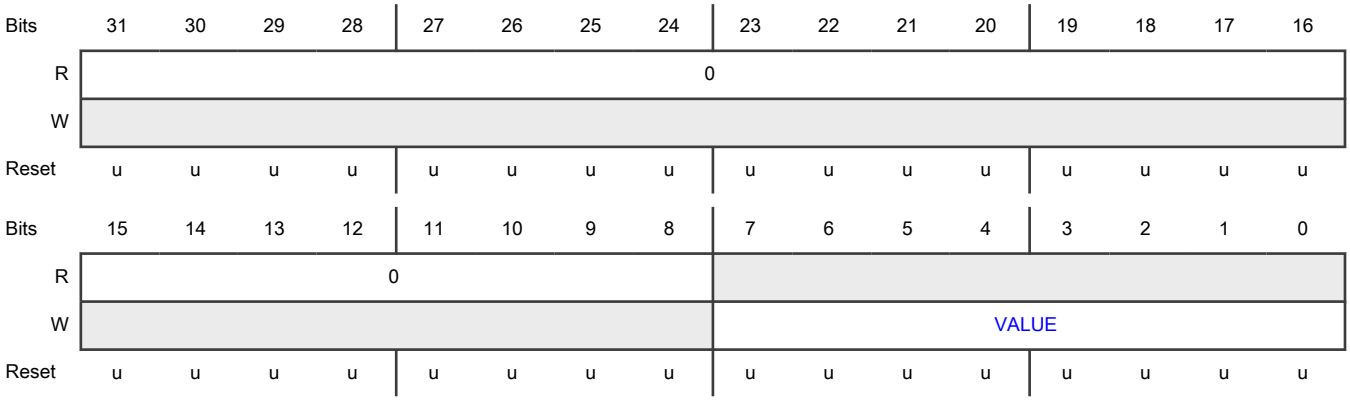
Allows writing the last byte to send onto the bus. This register is used only when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

This register acts as an alias of Target Write Data Byte End (SWDATABE).

NOTE

MWDATABE can also indicate END using bit 8.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 VALUE	Data Represents the last byte written to the target (stored in transmit FIFO): • I3C computes the parity. • I2C manages the ACK or NACK.

65.8.38 Controller Write Data Halfword (MWDATAH)

Offset

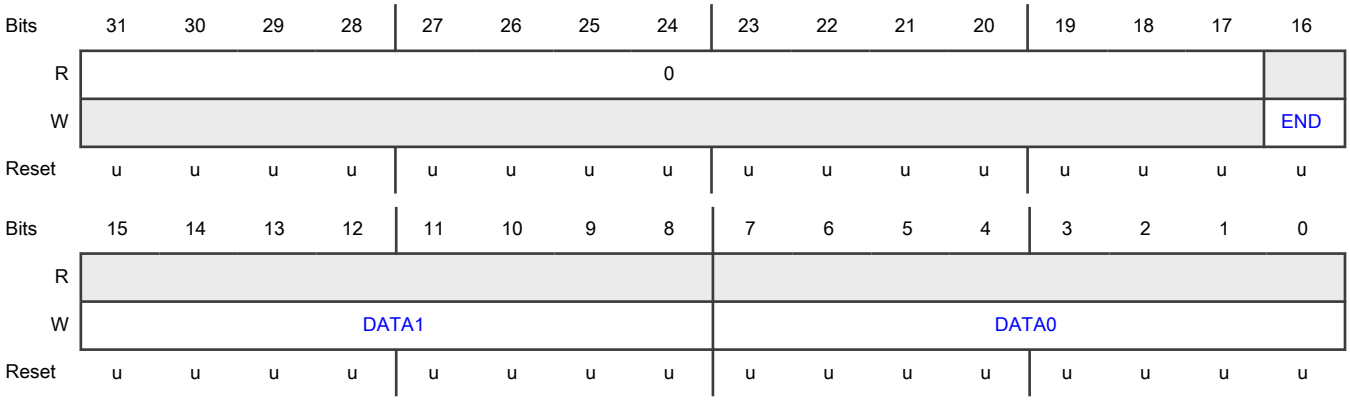
Register	Offset
MWDATAH	B8h

Function

Allows writing a halfword (pair of bytes) to the bus unless an external FIFO is used, sending the low byte followed by the high byte. An end-of-data (last) marker bit is allowed (or must be 0). A halfword must not be written unless there is room for both, as indicated by the use of transmit FIFO level trigger or MDATACTRL[TXCOUNT].

This register acts as an alias of Target Write Data Halfword (SWDATAH).

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END	End of Message Indicates the end of a message. For this register, this field always marks DATA1 as the end. This field is required for I3C and is optional for I2C. For HDR-DDR, the byte with the END must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. If this field is 0, more bytes are assumed to be in the message. If this field is 1, the END bit marks the last byte of the message. 0b - Not the end 1b - End
15-8 DATA1	Data Byte 1 Represents the second byte that is sent to the target (written to transmit FIFO).
7-0 DATA0	Data Byte 0 Represents the first byte that is sent to the target (written to transmit FIFO).

65.8.39 Controller Write Data Halfword End (MWDATAHE)

Offset

Register	Offset
MWDATAHE	BCh

Function

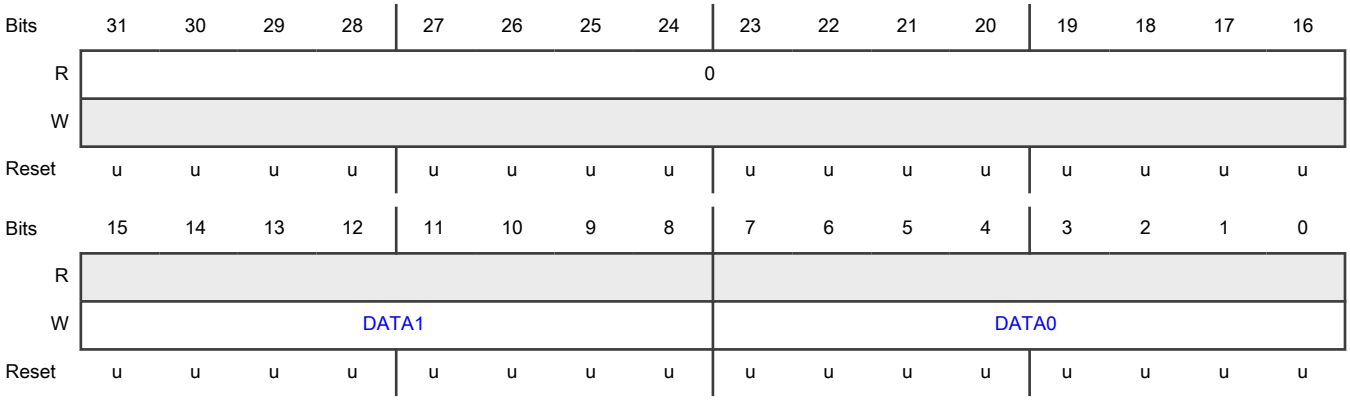
Allows writing a halfword of data, which is the end (the last byte of the halfword is the end). The target writes the halfword (byte pair) in the same way as it does to Controller Write Data Halfword (MWDATAH), but marks the second byte as end-of-data (last byte).

For HDR-DDR, the byte with the END must be even (second, fourth, sixth, and so on) because DDR uses byte pairs.

You must not write a halfword unless there is room for both, as indicated by the use of transmit FIFO level trigger or MDATACTRL[TXCOUNT].

This register acts as an alias of Target Write Data Halfword End (SWDATAHE).

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 DATA1	Data Byte 1 Represents the second byte that is sent to the target (written to transmit FIFO).
7-0 DATA0	Data Byte 0 Represents the first byte that is sent to the target (written to transmit FIFO).

65.8.40 Controller Read Data Byte (MRDATAB)

Offset

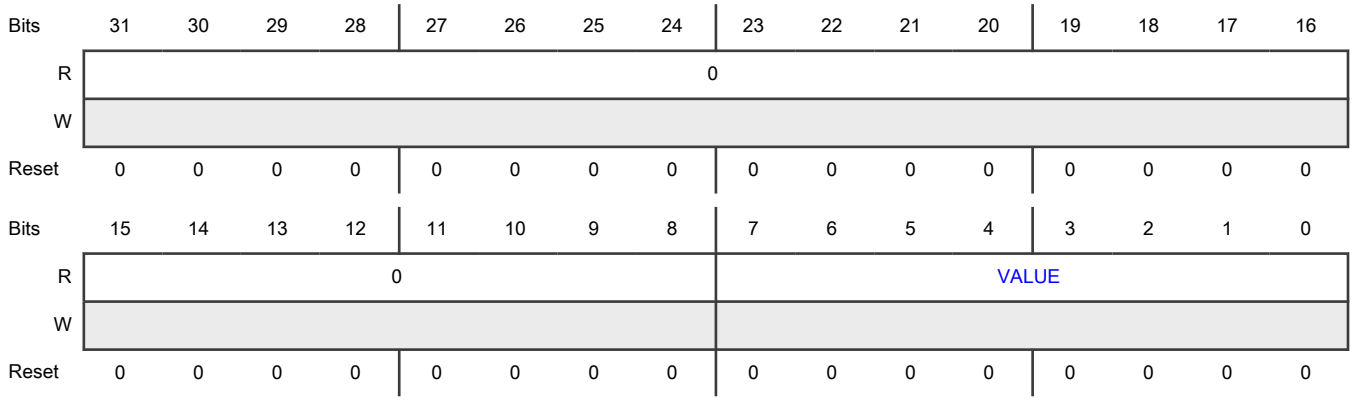
Register	Offset
MRDATAB	C0h

Function

Allows reading bytes written by the target after an SDR read, or DAA or DDR. This register is used only when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

This register acts as an alias of Target Read Data Byte (SRDATAB).

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 VALUE	Value Represents the byte read from the controller (and written by the target).

65.8.41 Controller Read Data Halfword (MRDATAH)

Offset

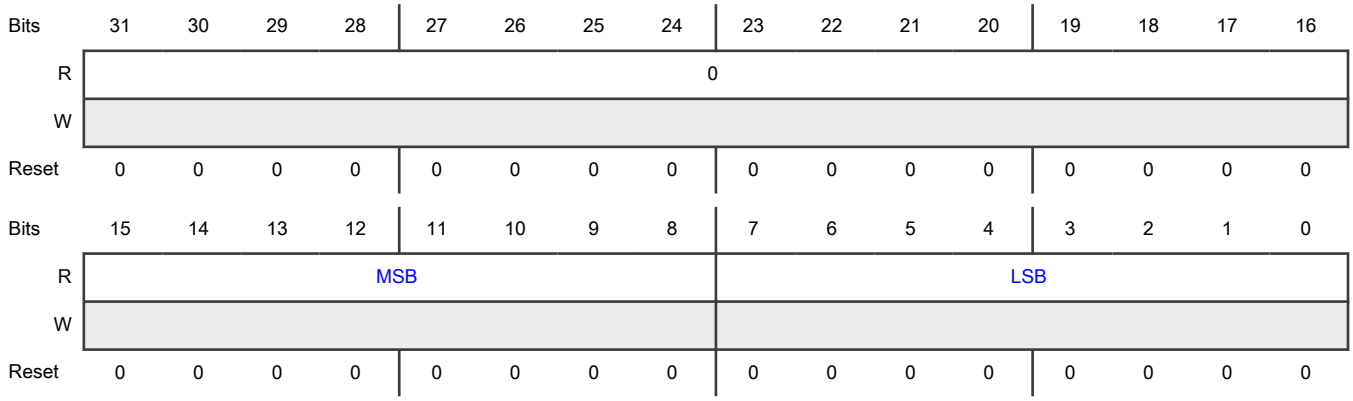
Register	Offset
MRDATAH	C8h

Function

Allows reading a halfword (byte pair) written by the target after an SDR read or DAA or DDR. This register is used only when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

A halfword must not be read unless there are at least two bytes of data waiting, as indicated by the receive FIFO level trigger or MDATACTRL[RXCOUNT].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 MSB	High Byte Represents the second byte read from the controller (and written by the target).
7-0 LSB	Low Byte Represents the first byte read from the controller (and written by the target).

65.8.42 Controller Write Byte Data 1 (to Bus) (MWDATAB1)

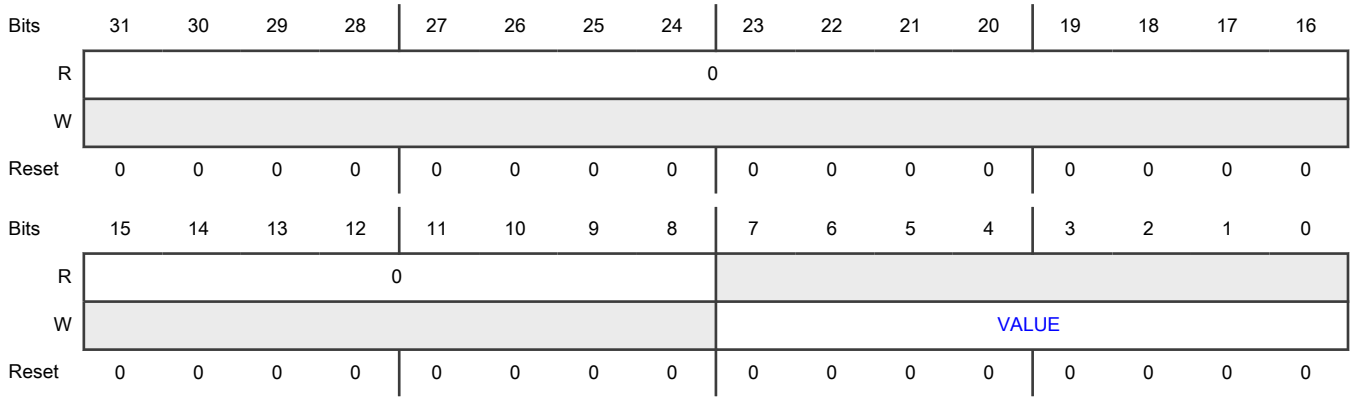
Offset

Register	Offset
MWDATAB1	CCh

Function

Allows writing bytes to send onto the bus, and is intended for DMAs that do not clear the upper bits of the word. This register does not have the END bits and is only used when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 VALUE	Value Represents the byte to write out. The I3C module computes the parity for I3C, or manages the ACK or NACK for I2C.

65.8.43 Controller Write Message Control in SDR mode (MWMSG_SDR_CONTROL)

Offset

Register	Offset
MWMSG_SDR_CONTROL	D0h

Function

Allows setting up and writing 16-bit words in Single Data Rate (SDR) mode. The MWMSG_SDR_ register is modal and has two modes—control and data:

- For the first write to set up a new message, this register functions as the MWSMSG_SDR_CONTROL register.
- For subsequent writes, this register functions as MWMSG_SDR_DATA.
- The control information is not pipelined. You must write to control information registers for a start and do not write to them until data is sent.

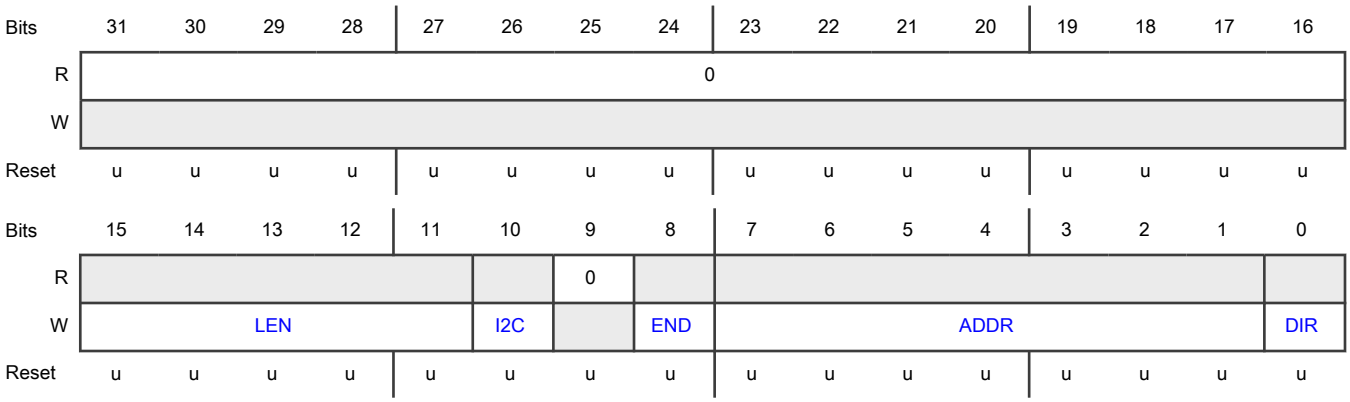
When not in the middle of a message, the MWMSG_SDR_CONTROL register is used to start a new message (or emit a STOP). After starting a message, the MWMSG_SDR_DATA register is used until the length (see the LEN field) counts down, or until data with END = 1 is used.

The MWMSG_SDR_CONTROL register is written with the 16-bit control word if currently stopped or after the end of the previous message. If MSTATUS[STATE] = 2 (MSGSDR) and MSTATUS[BETWEEN] = 0, the register (at this offset address) functions as the MWMSG_SDR_DATA register. Otherwise, this register (at this offset address) functions as the MWMSG_SDR_CONTRL register, as long as MSTATUS[STATE] is not in another mode.

The control word contains the byte length (6-bit), address, direction, and how it ends (stop, ready for next, continuation with more length). If the command is START and an event (IBI, CR, HJ) occurs, MCTRL[IBIRESP] is used to determine action, and the corresponding interrupt occurs. In that case, the message is restarted.

The MWMSG_SDR_CONTROL and MWMSG_SDR_DATA registers are only targeted for DMA operations, but the processor can also write to the MWMSG_SDR_CONTROL register (instead of using MCTRL and MxDATAB).

Diagram



Fields

Field	Function
31-16 —	Reserved
15-11 LEN	Length Indicates the byte length of the message. If LEN = 0, then only END is used (and it does not use the address if stopped). LEN = 1 must not be used. The minimal LEN size is 2 bytes (LEN = 2).
10 I2C	I2C Specifies whether the message is I2C or I3C. 0b - I3C message 1b - I2C message
9 —	Reserved
8 END	End of SDR Message Indicates the end of SDR message. MSTATUS[COMPLETE] = 1 when done. The end can happen: • Before LEN bytes are read in I3C mode, if a target ends sooner. • Before LEN bytes are written in I2C mode, if a target NACKs. If this field is 0, the SDR message ends waiting for a new SDR message (issues a Repeated START for a new message). If this field is 1, the SDR message ends at the STOP.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not the end 1b - End
7-1 ADDR	Address Contains the address to be written.
0 DIR	Direction 0b - Write 1b - Read

65.8.44 Controller Write Message Data in SDR mode (MWMSG_SDR_DATA)

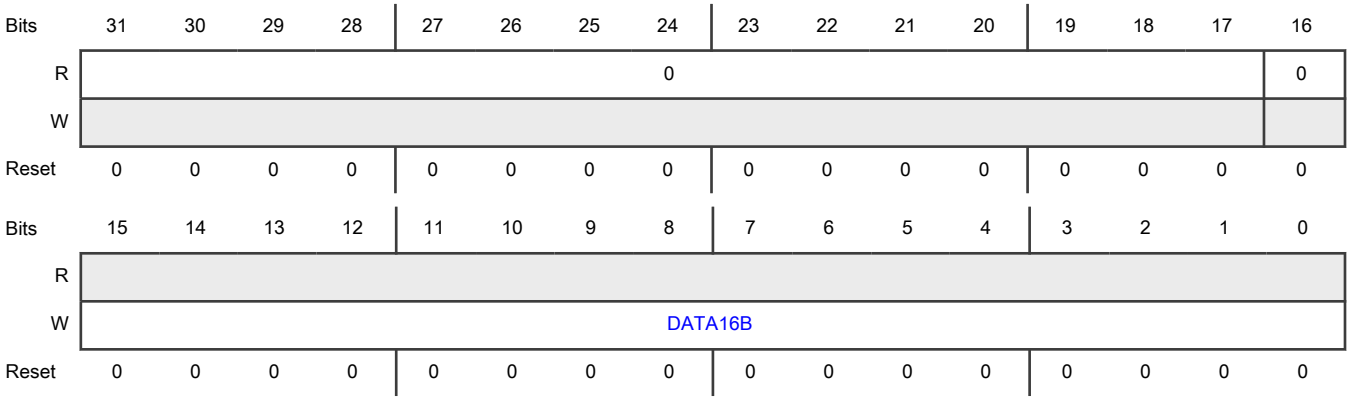
Offset

Register	Offset
MWMSG_SDR_DATA	D0h

Function

Contains the 16-bit word to be written in Single Data Rate (SDR) mode. This register functions in a way similar to MWMSG_SDR_CONTROL.

Diagram



Fields

Field	Function
31-17 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 —	Reserved
15-0 DATA16B	Data

65.8.45 Controller Read Message in SDR mode (MRMSG_SDR)

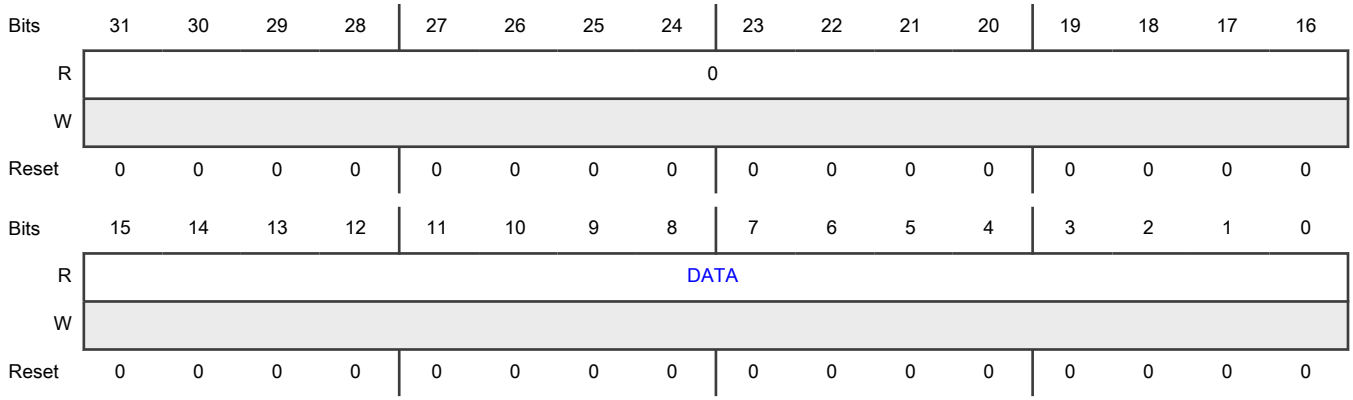
Offset

Register	Offset
MRMSG_SDR	D4h

Function

Allows reading 16-bit words from a target in SDR Message mode. The MRMSG_SDR register is used to read 16-bit words from an active message started with MWMSG_SDR. These words are intended to be read by DMA.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Data Contains the 16-bit word read from the target: • If the length (LEN) is an odd number, the upper byte is 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If the target ends before LEN is finished, the module treats the read as completed. • If the target is not done before LEN is finished and END is not a continuation, the read is terminated.

65.8.46 Controller Write Message in DDR mode: First Control Word (MWMSG_DDR_CONTROL)

Offset

Register	Offset
MWMSG_DDR_CONTROL	D8h

Function

Allows setting up and writing 16-bit words in DDR mode. The register has three modes:

- The first write to set up a new message functions as the MWSMSG_DDR_CONTROL register.
- The second write contains the functions as shown in the MWMSG_DDR_CONTROL2 register.
- For subsequent writes, this register functions in a way similar to Controller Write Message Data in DDR mode (MWMSG_DDR_DATA).
- The control information is not pipelined. You must write to control information registers for a start and do not write to them until data is sent.

When not in the middle of a message, the MWMSG_DDR_CONTROL register is used to start a new message (or emit a STOP). After starting a message, the MWMSG_DDR_DATA register is used until the length (see LEN field) counts down or until data with END = 1 is used.

The MWMSG_DDR_CONTROL register is written with the 16-bit control word if currently stopped or after the end of the previous message. If MSTATUS[STATE] = 4 (MSGDDR) and MSTATUS[BETWEEN] = 0, then the register (at this offset address) functions as the MWMSG_DDR_DATA register. Otherwise, this register (at this offset address) functions as the MWMSG_DDR_CONTROL register, as long as MSTATUS[STATE] is not in another mode.

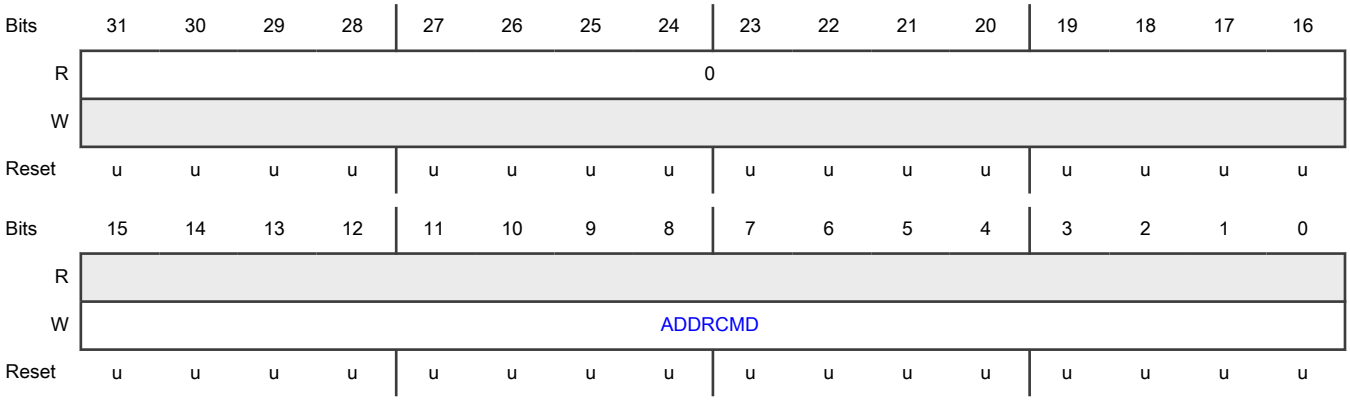
The main control word contains the 16-bit word length and how it ends (stop, ready for next, continuation with more length). Then the command word contains the command and address for read or write. If the command is START and an event (IBI, CR, HJ) occurs, MCTRL[IBIRESP] is used to determine action, and the corresponding interrupt occurs. In that case, the message is restarted.

The MWMSG_DDR_CONTROL, MWMSG_DDR_CONTROL2, and MWMSG_DDR_DATA registers are only targeted for DMA operations, but the processor can also write to the MWMSG_DDR_CONTROL register (instead of using MCTRL and MxDATAB).

NOTE

The module handles preamble, parity, and CRC.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 ADDRCMD	Address Command Indicates the first data write after control write, with LEN ≠ 0. This is formatted as: • Bits 15:9: target address to read or write • Bit 8: reserved, must be 0 • Bit 7: 1 if read, 0 if write • Bits 6:0: CMD as 7-bit value, always for controller

65.8.47 Controller Write Message in DDR Mode Control 2 (MWMSG_DDR_CONTROL2)

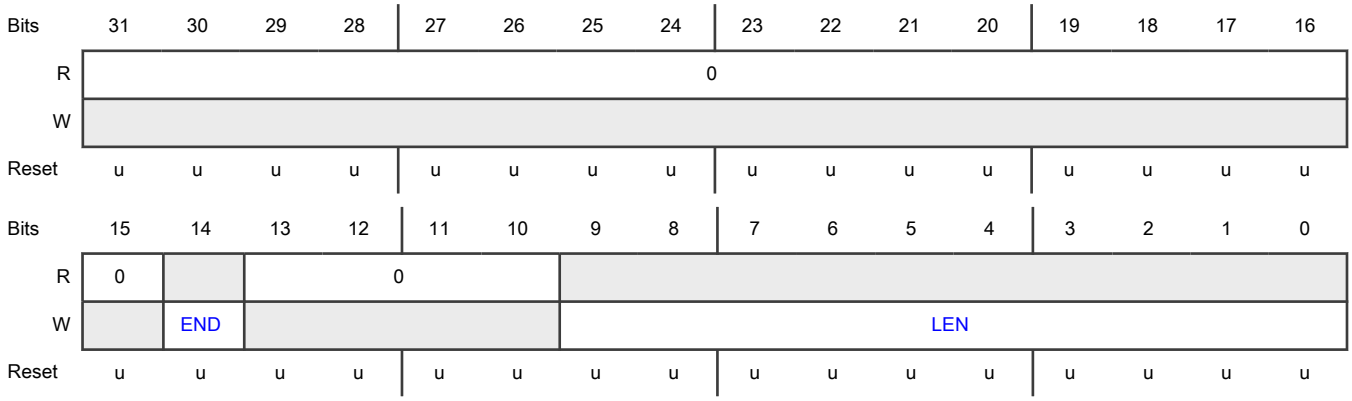
Offset

Register	Offset
MWMSG_DDR_CONTROL2	D8h

Function

Contains the second control word instructions with the length of message and end.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 —	Reserved
14 END	End of Message Indicates the end of DDR message. MSTATUS[COMPLETE] becomes 1 when done. The end can happen before LEN bytes are read if the target ends sooner. If this field is 0, DDR message ends waiting for a new DDR message (issues an HDR restart for the new message). If this field is 1, DDR message ends on HDR exit. 0b - Not the end 1b - End
13-10 —	Reserved
9-0 LEN	Length of Message Contains the length of the message (including the command) in halfwords, up to 2046 bytes. If LEN = 0, then only END is applied: • For reads, + 1 for the CRC. For example, to read 4 bytes (2 halfwords), use 1 + 2 + 1 for CMD + 2 halfwords + CRC. • For writes, LEN is the number of halves of data bytes + 1 (for command).

65.8.48 Controller Write Message Data in DDR mode (MWMSG_DDR_DATA)

Offset

Register	Offset
MWMSG_DDR_DATA	D8h

Function

Contains the 16-bit word to be written in DDR mode. This register functions in a way similar to Controller Write Message in DDR mode: First Control Word (MWMSG_DDR_CONTROL).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	DATA16B															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 —	Reserved
15-0 DATA16B	Data

65.8.49 Controller Read Message in DDR mode (MRMSG_DDR)

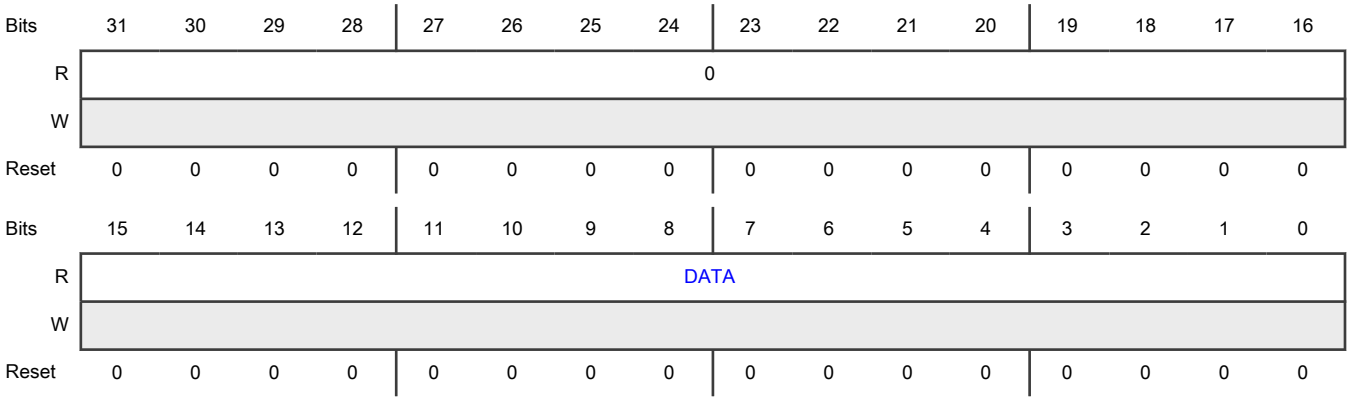
Offset

Register	Offset
MRMSG_DDR	DCh

Function

Allows reading 16-bit words from a target in DDR Message mode from an active message started with MWMSG_DDR. These words are intended to be read by DMA.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Data Contains the 16-bit word read from a target. The first byte is the LSB, and is in DATA[7:0]. The second byte is the MSB, and is in DATA[15:8]: • If the target ends before the entire length of the message (MWMSG_DDR[LEN]) is read, the module considers the DATA read as completed. In I3C mode, the target can indicate the end of message (the last byte). Otherwise, the controller terminates the message if the message is more than the controller can accept. • If the target has not yet finished sending DATA before the entire length of the message (MWMSG_DDR[LEN]) is read and END is not a continuation, the DATA read is terminated.

65.8.50 Controller Dynamic Address (MDYNADDR)

Offset

Register	Offset
MDYNADDR	E4h

Function

Allows the I3C module to write its own dynamic address (DA) when the module changes from Controller mode to Target mode.

If this device is the main controller (the controller during bus initialization), then the device may use this mechanism to assign itself its DA. When the device hands off control to a secondary controller, it becomes a target itself. This DA must be written before switching to Target mode and must not be changed once in Target mode (it is not clock-safe to do so). It must be written with a valid address value in DADDR if DAVALID = 1.

NOTE

The main controller also uses DEFSLVS CCC to define the target addresses, including itself. This mechanism is how secondary controllers know this address. If the controller is not the main controller, then this mechanism must not be used.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								DADDR							DAVALID
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-1 DADDR	Dynamic Address Contains the assigned dynamic address when DAVALID = 1.
0 DAVALID	Dynamic Address Valid 0b - No valid DA assigned 1b - Valid DA assigned

65.8.51 Timing Rules for Target Reset Recovery (SRSTACTTIME)

Offset

Register	Offset
SRSTACTTIME	100h

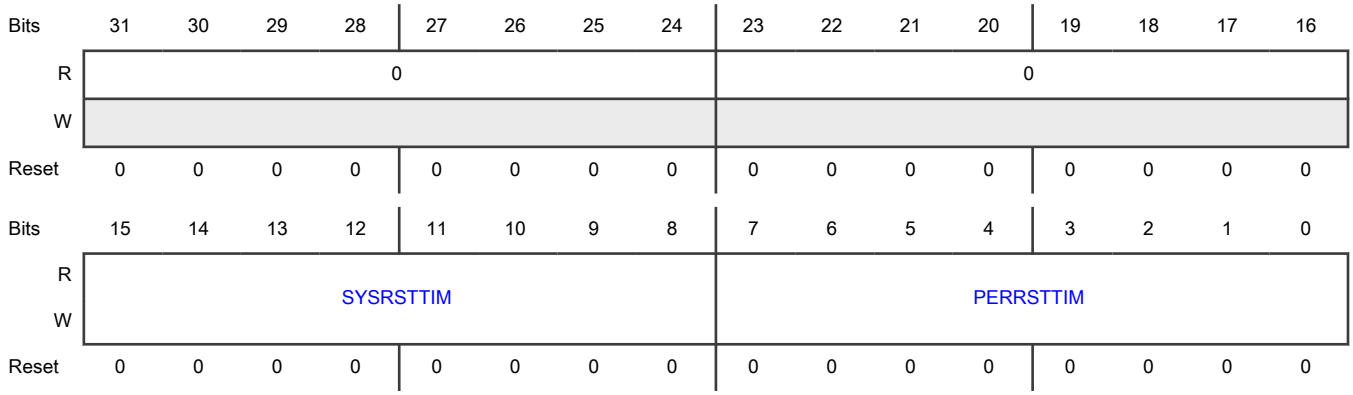
Function

Indicates the timing rules for a recovery following a TargetReset.

When the controller issues a TargetReset, some part of the device may be reset, such as a peripheral, the whole device, or some subset of the device. The controller must know how long the device takes to recover and be ready for new I3C messages. This behavior may be hard-coded or configured using this register.

This register is used if the device is enabled for MMR.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 SYSRSTTIM	Time to Recover from Chip Reset Specifies the time to recover, in seconds, from a full system or chip reset.
7-0 PERRSTTIM	Time to Recover from the I3C Peripheral Specifies the time to recover, in milliseconds, from the I3C peripheral reset.

65.8.52 CCC Mask for Unhandled CCCs (SCCCMASK)

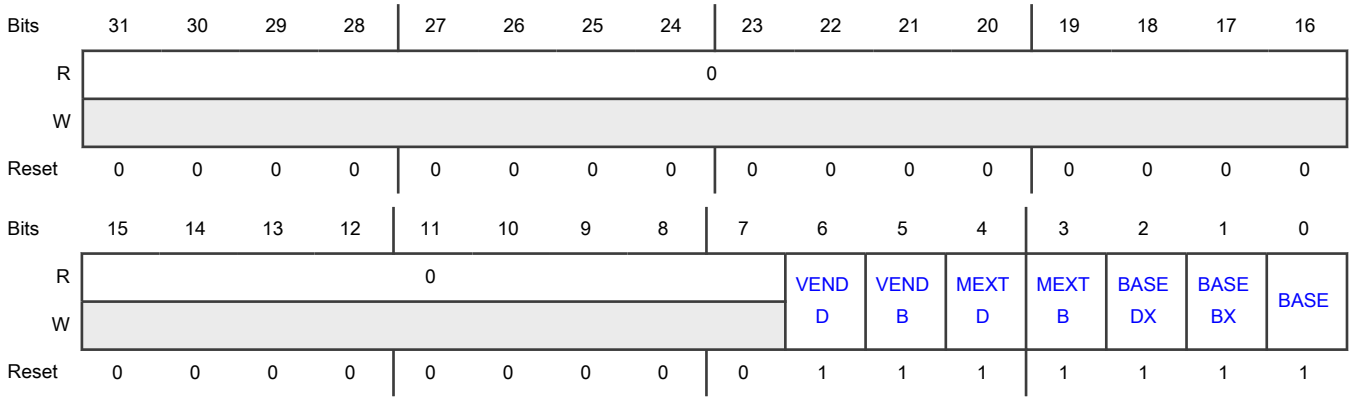
Offset

Register	Offset
SCCCMASK	10Ch

Function

Allows masking of unhandled (not handled by hardware) CCC commands, when enabled.

Diagram



Fields

Field	Function
31-7 —	Reserved
6 VENDD	VENDD Determines whether any unhandled CCC commands in the range of E4h to FFh are passed to the application or suppressed. 0b - Suppressed 1b - Passed to application
5 VENDB	VENDB Determines whether any unhandled CCC commands in the range of 65h to 7Fh are passed to the application or suppressed. 0b - Suppressed 1b - Passed to application
4 MEXTD	MEXTD Determines whether any unhandled CCC commands in the range of C0h to E3h are passed to the application or suppressed. 0b - Suppressed 1b - Passed to application
3 MEXTB	MEXTB Determines whether any unhandled CCC commands in the range of 49h to 64h are passed to the application or suppressed. 0b - Suppressed 1b - Passed to application
2	BASEDX

Table continues on the next page...

Table continued from the previous page...

Field	Function
BASEDX	Determines whether any unhandled CCC commands in the range of A0h to BFh are passed to the application or suppressed. 0b - Suppressed 1b - Passed to application
1 BASEBX	BASEBX Determines whether any unhandled CCC commands in the range of 20h to 48h are passed to the application or suppressed. 0b - Suppressed 1b - Passed to application
0 BASE	Base Determines whether any unhandled CCC commands in the range of 0h to 1Fh and 80h to 8Fh are passed to the application or suppressed. 0b - Suppressed 1b - Passed to application

65.8.53 Target Errors and Warnings Mask (SERRWARNMASK)

Offset

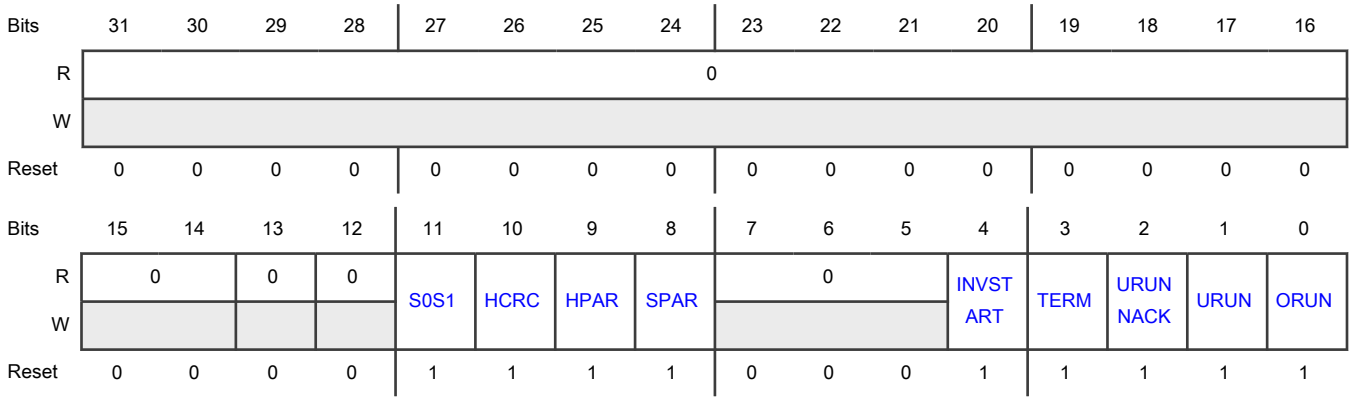
Register	Offset
SERRWARNMASK	110h

Function

Allows masking of target errors and warnings.

Target Errors and Warnings (SERRWARN) contains I3C protocol errors and issues detected on the line. If any are 1, SSTATUS[ERRWARN] becomes 1. The corresponding interrupt field also becomes 1, if enabled. SERRWARNMASK allows masking them, so only some form the status bit.

Diagram



Fields

Field	Function
31-14 —	Reserved
13 —	Reserved
12 —	Reserved
11 S0S1	S0S1 Mask Determines whether to allow SERRWARN[S0S1] to contribute to the status of SSTATUS[ERRWARN]. 0b - Deny 1b - Allow
10 HCRC	HCRC Mask Determines whether to allow SERRWARN[HCRC] to contribute to the status of SSTATUS[ERRWARN]. 0b - Deny 1b - Allow
9 HPAR	HPAR Mask Determines whether to allow SERRWARN[HPAR] to contribute to the status of SSTATUS[ERRWARN]. 0b - Deny 1b - Allow
8 SPAR	SPAR Mask Determines whether to allow SERRWARN[SPAR] to contribute to the status of SSTATUS[ERRWARN]. 0b - Deny

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Allow
7-5 —	Reserved
4 INVSTART	INVSTART Mask Determines whether to allow SERRWARN[INVSTART] to contribute to the status of SSTATUS[ERRWARN]. 0b - Deny 1b - Allow
3 TERM	TERM Mask Determines whether to allow SERRWARN[TERM] to contribute to the status of SSTATUS[ERRWARN]. 0b - Deny 1b - Allow
2 URUNNACK	URUNNACK Mask Determines whether to allow SERRWARN[URUNNACK] to contribute to the status of SSTATUS[ERRWARN]. 0b - Deny 1b - Allow
1 URUN	URUN Mask Determines whether to allow SERRWARN[URUN] to contribute to the status of SSTATUS[ERRWARN]. 0b - Deny 1b - Allow
0 ORUN	ORUN Mask Determines whether to allow SERRWARN[ORUN] to contribute to the status of SSTATUS[ERRWARN]. 0b - Deny 1b - Allow

65.8.54 Map Feature Control 0 (SMAPCTRL0)

Offset

Register	Offset
SMAPCTRL0	11Ch

Function

Provides map feature control.

The SMAPCTRL*n* registers are named SMAPCTRL0, SMAPCTRL1, and so on based on the number of mapped addresses. MAPCTRL0 represents the primary DA or SA with SMAPCTRL1 onwards being the mapped addresses.

The features of the SMAPCTRL*n* registers depend on configuration. SMAPCTRL0 acts differently, as described in this chapter.

In general, this mechanism is intended to replace the DYNADDR register for all MAP-related uses.

When using the Auto-MAP and DASA or AASA, the slot is changed from SA to DA. If the controller then issues RSTDAA, the application must rewrite the static addresses and enable them because they are marked disabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				CAUSE				DA							ENA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 —	Reserved
15-11 —	Reserved
10-8 CAUSE	Cause Indicates the cause of the most recent DA assignment, which lead to SSTATUS[DACHG] interrupt. This field has MAP enabled. If this field is 100b (auto MAP change happened last), the change may have changed this DA as well (for example, ENTDAA and SETAASA), but at least one MAP entry automatically changed after that. 000b - No information (this value occurs when not configured to write DA) 001b - Set using ENTDAA 010b - Set using SETDASA, SETAASA, or SETNEWDA

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - Cleared using RSTDAA 100b - Auto MAP change happened last All other values are reserved.
7-1 DA	Dynamic Address Contains primary DA when ENA = 1. When ENA = 0, static address is used (but not described here). This field has MAP enabled.
0 ENA	Enable Primary Dynamic Address Indicates whether the MAP is enabled. 0b - Disabled 1b - Enabled

65.8.55 Map Feature Control 1 (SMAPCTRL1)

Offset

Register	Offset
SMAPCTRL1	120h

Function

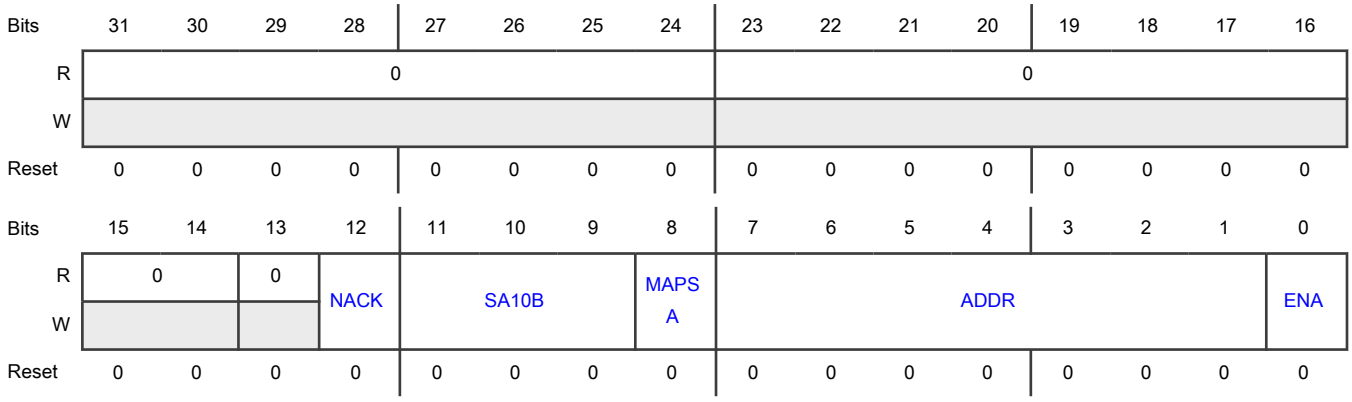
The SMAPCTRL*n* registers are named SMAPCTRL0, SMAPCTRL1, and so on based on the number of mapped addresses. SMAPCTRL0 represents the primary DA or SA, with SMAPCTRL1 onwards being mapped addresses.

The features of the SMAPCTRL*n* registers depend on configuration. SMAPCTRL0 acts somewhat differently, as described in this chapter.

In general, this mechanism is intended to replace the DYNADDR register for all MAP-related uses.

When using the AutoMAP and DASA or AASA, the slot is changed from static address to dynamic address. If the controller then issues RSTDAA, the application must rewrite the SAs and enable them because as they are marked disabled.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-14 —	Reserved
13 —	Reserved
12 NACK	Not Acknowledged Indicates whether messages to this address are always NACKed, if enabled. 0b - Do not always NACK messages 1b - Always NACK messages
11-9 SA10B	Static Address 10-Bit Extension Contains a 10-bit extension for a static address, if this field is in MAPCTRL1. In the case of a normal address, this field is 0; otherwise, it contains the 10-bit extension. This field is used if enabled for 10-bit.
8 MAPSA	MAP Static Address Indicates whether MAP entry is a static address or a dynamic address, if ENA = 1. This field has MAP enabled. 0b - I3C dynamic address 1b - Static address (I2C style)
7-1 ADDR	Address Contains static or dynamic address. If ENA = 1, contains static address or dynamic address (as indicated by the MAPSA field) with MAP enabled.
0 ENA	Enable Enables MAP entry. This entry could be a static or dynamic address with MAP enabled. 0b - Disable 1b - Enable

65.8.56 Extended IBI Data 1 (IBIEXT1)

Offset

Register	Offset
IBIEXT1	140h

Function

Contains extended IBI data.

The CTRL register is used to submit IBI, CR, and Hot-Join when enabled to do so. If allowed, an IBI may have additional bytes following the mandatory data byte (MDB). Extended IBI data is allowed when SCTRL[EXTDATA] = 1. If allowed, the extra bytes are indicated using these registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EXT3								EXT2							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EXT1								0	MAX			0	CNT		
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0

Fields

Field	Function
31-24 EXT3	Extra Byte 3 Contains the third extra byte.
23-16 EXT2	Extra Byte 2 Contains the second extra byte.
15-8 EXT1	Extra Byte 1 Contains the first extra byte.
7 —	Reserved
6-4 MAX	Maximum Indicates the maximum number of extra bytes allowed by configuration. This field is 0 if there are no extra bytes.
3 —	Reserved
2-0 CNT	Count Contains the number of extra bytes beyond the MDB to be used. This field is 0 if there are no extra bytes.

65.8.57 Extended IBI Data 2 (IBIEXT2)

Offset

Register	Offset
IBIEXT2	144h

Function

Contains extended IBI data.

The CTRL register is used to submit IBI, CR, and Hot-Join when enabled to do so. If allowed, an IBI may have additional bytes following the mandatory data byte (MDB). Extended IBI data is allowed when SCTRL[EXTDATA] = 1. If allowed, the extra bytes are indicated using these registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EXT7								EXT6							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EXT5								EXT4							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 EXT7	Extra Byte 7 Contains the seventh extra byte.
23-16 EXT6	Extra Byte 6 Contains the sixth extra byte.
15-8 EXT5	Extra Byte 5 Contains the fifth extra byte.
7-0 EXT4	Extra Byte 4 Contains the forth extra byte.

Chapter 66
Timers Overview

66.1 Overview

The following timers are supported in this chip:

- General Purpose Timer (GPT): A 32-bit up-counter with 12-bit pre-scaler
- Low Power Periodic Interrupt Timer (LPIT): A 32-bit counter timer that features programmable count modulus, clock division features etc.
- Quad Timer (TMR): It provides four timer channels with variety of controls for individual and multi-channel features
- Quadrature Decoder (eQDC): It provides interfacing capability to position/speed sensors
- Enhanced FlexPWM: It contains PWM submodules each of which is set up to control a single half bridge power stage
- Watchdog timers (WDOG1-WDOG5): It is a high reliability independent timer that is available for system use
- External Watchdog Monitor (EWM): It is designed to monitor external circuits, as well as the MCU software flow

66.1.1 General Purpose Timer (GPT)

The GPT has a 32-bit up-counter. The timer counter value can be captured in a register using an event on an external pin. The capture trigger can be programmed to be a rising or/and falling edge. The GPT also features output compare event generation, when the timer matches a preset value, GPT can generate an interrupt for software and a compare event signal on output pins for external hardware.

The GPT has a 12-bit pre-scaler, which provides a programmable clock frequency derived from multiple clock sources.

66.1.2 Low Power Periodic Interrupt Timer (LPIT)

The LPIT is a basic 32-bit counter timer. It features 32-bit counter timer, programmable count modulus, clock division features, interrupt generation, and ability to chain adjacent timers to achieve longer interval.

66.1.3 Quad Timer (TMR)

The quad-timer provides four timer channels. Specific features include up/down count, cascading of counters, programmable modulo, count once/repeated, counter preload, compare registers with preload, shared use of input signals, prescaler controls, independent capture/compare, fault input control, programmable input filters, and multi-channel synchronization.

As shown in diagram below, input of QTimer can be sourced from PADs or XBAR1:

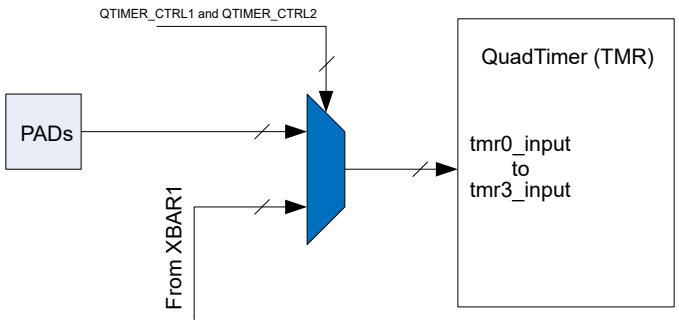


Figure 481. QTimer inputs

66.1.4 Enhanced Flex Pulse Width Modulator (eFlexPWM)

This module can generate various switching patterns, including highly sophisticated waveforms. Each module shall be configured with 4 channels supporting A, B, and X PWM output. All channels are configured for standard edge placement and have their capture function enabled.

66.1.5 Quadrature Decoder (eQDC)

The enhanced Quadrature Decoder module provides interfacing capability to position speed sensors used in industrial motor control applications, and capability of compare output as well as input capture. The Enhanced Quadrature Decoder module interfaces to position/speed sensors that are used in industrial motor control applications and decodes shaft position, revolution count, and speed. The quadrature decoder gets 8 input signals: PHASEA, PHASEB, INDEX/PRESET, TRIGGER, HOME/ENABLE and ICAP[3:1] from those position/speed sensors, and outputs 11 signals: POS_MATCH[3:0], COMP_FLG[3:0], DIR, CNT_DN, and CNT_UP for system use.

66.1.6 Watchdog Timer (WDOG)

The WDOG module is an high reliability independent timer that is available for system use. It provides a safety feature to ensure that software is executing as planned and that the CPU is not stuck in an infinite loop or executing unintended code. If the WDOG module is not serviced (refreshed) within a certain period, it resets the CPU.

There are 5 watch dog timers. One for the Cortex-M33 non-secure world, one for the Cortex-M33 secure world, and one for the Cortex-M7. Two additional for customer usage

Main features of the WDOG module are:

- Configurable independent clock source input from bus clock
- Programmable time-out period
- Robust write sequence for counter refresh
- Windowed refresh option
- Optional timeout interrupt to allow post-processing diagnostics
- Configuration bits are write-once-after-reset to ensure watchdog configuration cannot be mistakenly altered
- Robust write sequence for unlocking write-once control/configuration bits

The WDOG clocks can be sourced from:

- Fixed 24MHz (ext_clk)
- Fixed 32KHz (lpo_clk)

66.1.7 External Watchdog Monitor (EWM)

The External Watchdog Monitor provides a back-up mechanism to the internal WDOG that can reset the system. The EWM differs from the internal WDOG in that it does not reset the system.

The EWM, if allowed to time-out, provides an independent trigger pin that when asserted resets or places an external circuit into a safe mode.

Chapter 67
Enhanced Flex Pulse Width Modulator (eFlexPWM)

67.1 Chip-specific eFlexPWM Information

Table 1052. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

Fault functionality

eFlexPWM fault may work abnormally when the fault signal is very narrow. If the fault signal pulse width is narrower than a certain threshold, the protected PWM channels may generate a glitch, which occurs after the PWM channel outputs become inactive.

The following workaround can be used to mitigate this abnormal functioning.

- When FCTRL2[NOCOMB] = 0, FFILT [GSTR]= 0, and FFILT[FILT_PER]=0, pulse width of fault signals must be larger than 6 PWM clock periods, otherwise a glitch may be generated on the protected PWM channels.
- When FCTRL2[NOCOMB] = 0, FFILT [GSTR]= 1, and FFILT[FILT_PER]=0, pulse width of fault signals must be larger than 3 PWM clock periods, otherwise a glitch may be generated on the protected PWM channels.
- When FCTRL2[NOCOMB] = 0, FFILT [GSTR]= 1, and FFILT[FILT_PER] has non-zero values, pulse width of fault signals must be larger than FILT_PER*(FILT_CNT+3)+6 PWM clock periods, otherwise a glitch may be generated on the protected PWM channels.
- When FCTRL2[NOCOMB] = 0, FFILT [GSTR]= 0, and FFILT[FILT_PER] has non-zero values, pulse width of fault signals must be larger than FILT_PER*(FILT_CNT+3)+9 PWM clock periods, otherwise a glitch may be generated on the protected PWM channels.

67.2 Overview

The Pulse Width Modulator (PWM) module contains PWM submodules, each of which can be used to control a single half-bridge power stage. Fault channel support is provided.

This module generates various switching patterns, including highly sophisticated waveforms. It can be used to control all known motor types and is ideal for controlling different Switched Mode Power Supplies (SMPS) topologies.

67.2.1 Block diagram

The following figure shows the PWM block diagram.

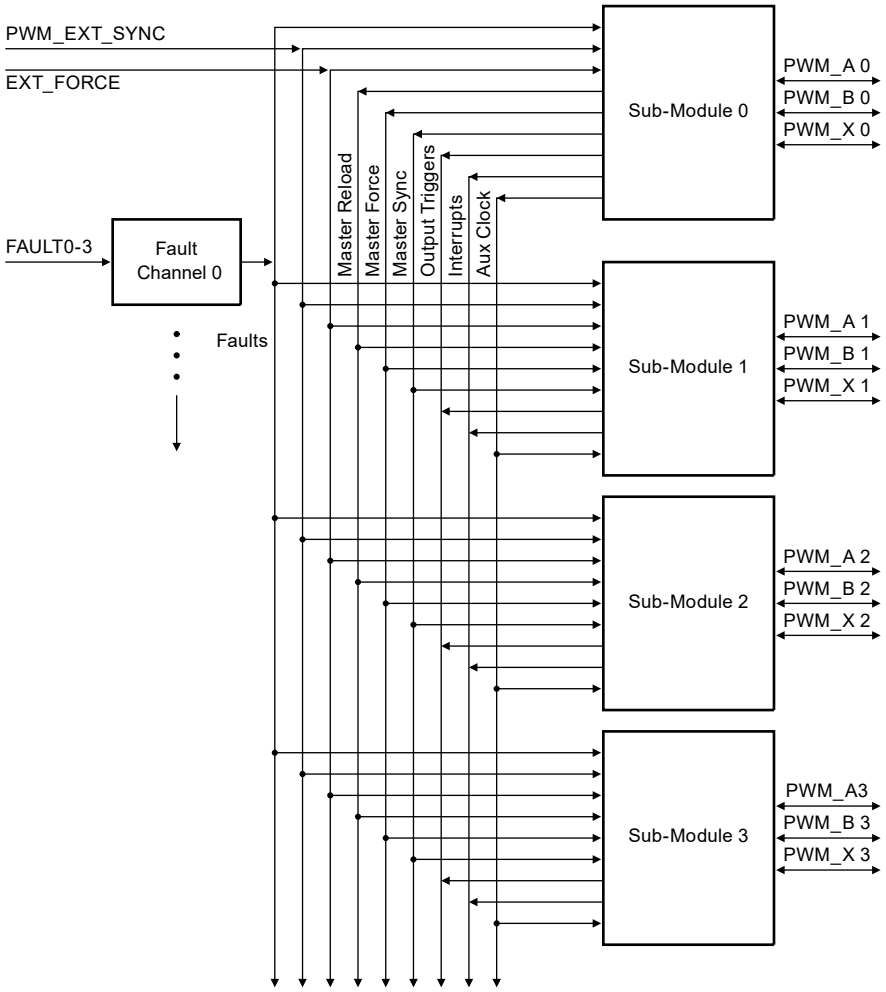


Figure 482. PWM block diagram

67.2.2 Features

Following are the features of PWM:

- 16-bit resolution for center, edge-aligned, and asymmetrical PWMs
- Dithering to simulate enhanced resolution when fine edge placement is not available
- PWM outputs that can operate as complementary pairs or independent channels
- Ability to accept signed numbers for PWM generation
- Independent control of both edges of each PWM output
- Support for synchronization to external hardware or other PWM
- Double buffered PWM registers
  - Integral reload rates from 1 to 16
  - Half cycle reload capability
- Multiple output trigger events can be generated per PWM cycle via hardware

- Support for double switching PWM outputs
- Fault inputs can be assigned to control multiple PWM outputs
- Programmable filters for fault inputs
- Independently programmable PWM output polarity
- Independent top and bottom deadtime insertion
- Each complementary pair can operate with its own PWM frequency and deadtime values
- Individual software control for each PWM output
- All outputs can be programmed to change simultaneously via a FORCE_OUT event
- PWM_X pin can optionally output a third PWM signal from each submodule
- Channels not used for PWM generation can be used for buffered output compare functions and for input capture functions
- Enhanced dual edge capture functionality

67.3 Functional description

67.3.1 PWM submodule

The following figure shows the PWM Submodule Block Diagram.

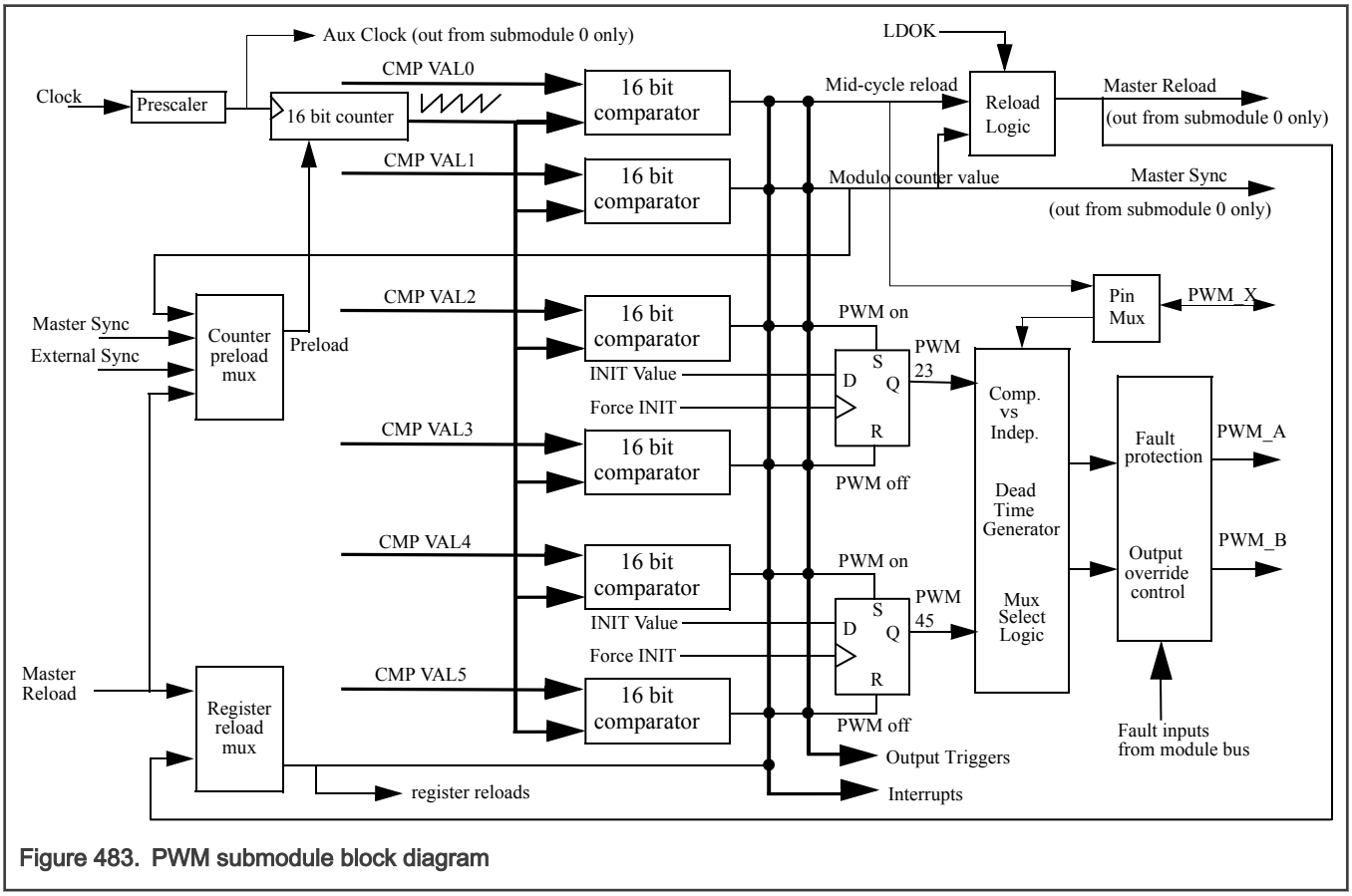


Figure 483. PWM submodule block diagram

67.3.2 PWM capabilities

This section describes some capabilities of the PWM module.

67.3.2.1 Center aligned PWMs

Each submodule has its own timer that is capable of generating PWM signals on two output pins. The edges of each of these signals are controlled independently as shown in Figure 484.

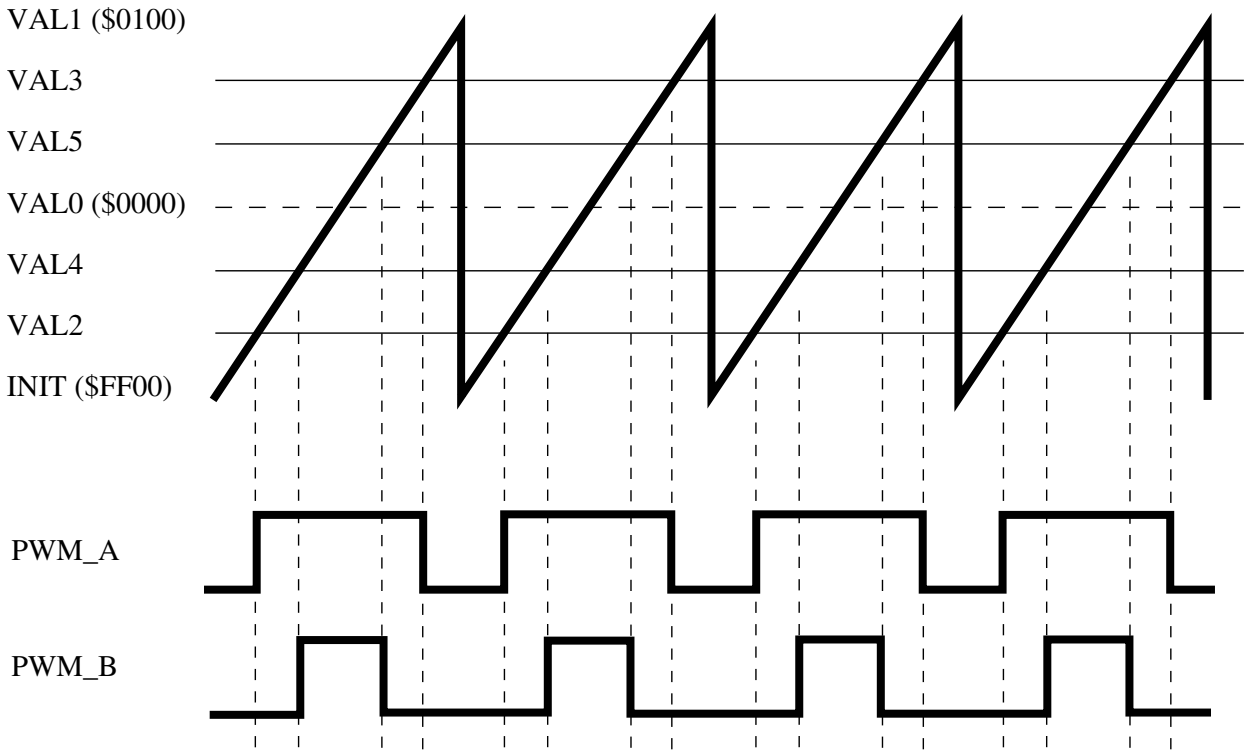


Figure 484. Center aligned PWM example

The submodule timers only count in the up direction and then reset to the INIT value. Instead of having a single value that determines pulse width, there are two values that must be specified: the turn-on edge and the turn-off edge. This double-action edge generation provides the user control over the pulse width and also the relative alignment of the signal. As a result, there is no need to support separate PWM alignment modes since the PWM alignment mode is inherently a function of the turn-on and turn-off edge values.

Figure 484 also illustrates an additional enhancement to the PWM generation process. When the counter resets, it is reloaded with a user-specified value, which may or may not be zero. If the value chosen happens to be the 2's complement of the modulus value, then the PWM generator operates in "signed" mode. This means if each PWM's turn-on and turn-off edge values are same in numbers but different in their sign, the "on" portion of the output signal is centered around a count value of zero. Therefore, only one PWM value is calculated in software and then this value and its negative are provided to the submodule as the turn-off and turn-on edges respectively. This technique results in a pulse width consists of an odd number of timer counts. If all PWM signal edge calculations follow this convention, then the signals will be center aligned with each other, which is the goal. The center alignment between the signals is not restricted to symmetry around the zero count value, as any other number would also work. However, centering on zero provides the greatest range in signed mode and also simplifies the calculations.

67.3.2.2 Edge aligned PWMs

Figure 485 shows the results of edge aligned operation when the turn-on edge for each pulse is specified to be the INIT value. Therefore, only the turn-off edge value needs to be periodically updated to change the pulse width.

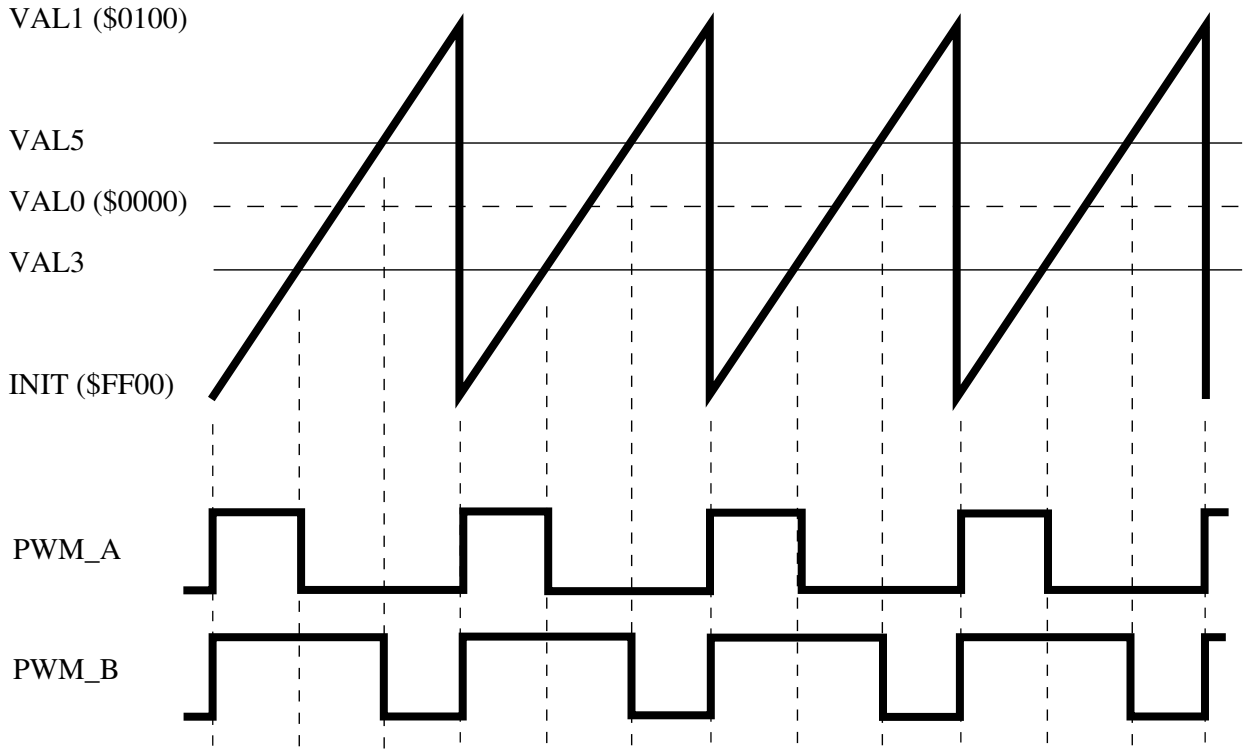


Figure 485. Edge aligned example (INIT=VAL2=VAL4)

With edge aligned PWMs, another example of the benefits of signed mode can be seen. Use "bipolar" PWMs to drive an H-bridge, where a 50% duty cycle results in zero volts on the load. Duty cycles less than 50% generate negative load voltages and duty cycles greater than 50% generate positive load voltages. If the module is set to signed mode operation (the INIT and VAL1 values are the same number with opposite signs), then there is a direct proportionality between the PWM turn-off edge value and the motor inverter voltage, including the sign. Therefore, signed mode of operation simplifies the software interface to the PWM module since no offset calculations are required to translate the output variable control algorithm to the voltage on an H-Bridge load.

67.3.2.3 Phase shifted PWMs

In the previous sections, the benefits of the signed mode of operation were discussed in the context of simplifying the required software calculations by eliminating the requirement to bias up signed variables before applying them to the module. However, if numerical biases are applied to the turn-on and turn-off edges of different PWM signals, the signals will be phase shifted to each other, as shown in Figure 486. This results in certain advantages when applied to a power stage. For example, when operating a multi-phase inverter at a low modulation index, all of the PWM switching edges from the different phases occur at the same time. This can be troublesome from a noise standpoint, especially if ADC readings of the inverter must be scheduled near those times. Phase shifting the PWM signals can open up timing windows between the switching edges to allow a signal to be sampled by the ADC. However, phase shifting does not affect the duty cycle so average load voltage is not affected.

If the outputs of submodules 1 - 3 need to be delayed from the output of submodule 0 (and from each other), instead of just creating a phase delay by adding an offset to the turn on and turn off times of the different submodules, another method is to use the PHASEDLY registers for submodules 1 - 3 to indicate their delay from the submodule 0 timing. This method can be used when the master sync signal from submodule 0 is selected as the initialization source (CTRL2[INIT_SEL]==b10). This method allows all of the submodules to be programmed with the same turn on and turn off time but submodules 1 - 3 can still be delayed the time from submodule 0.

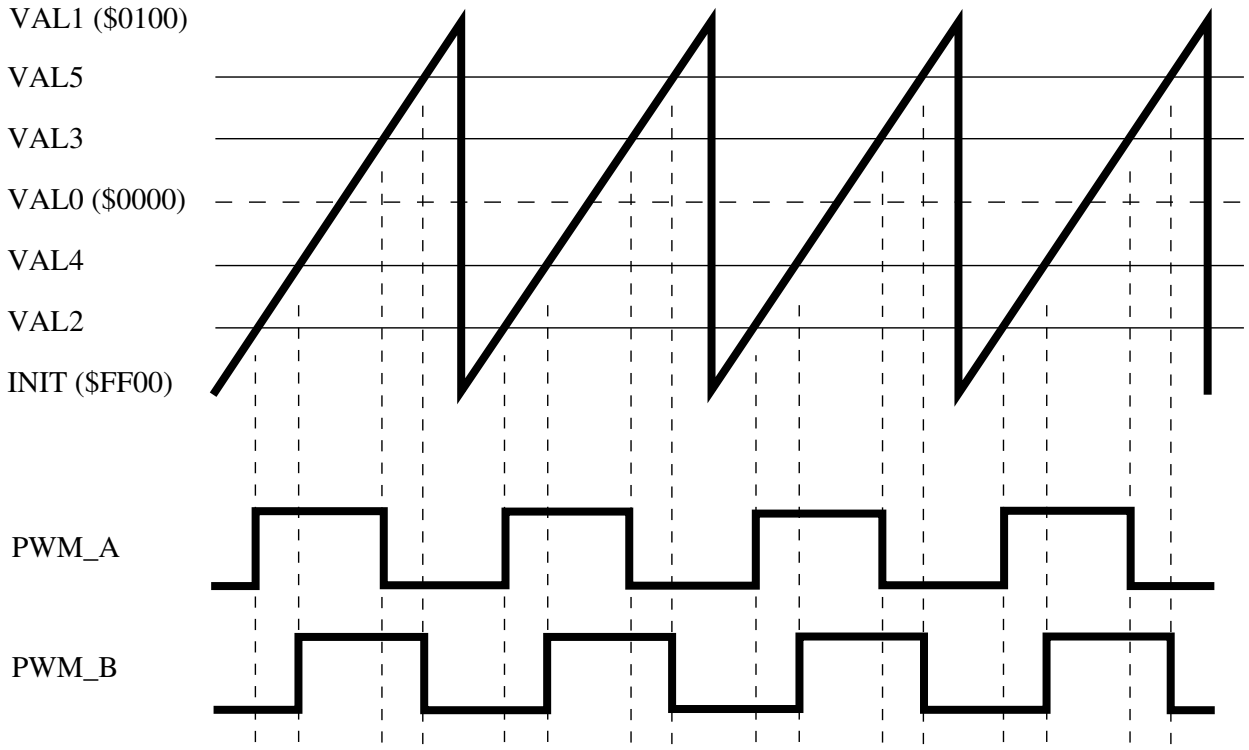


Figure 486. Phase shifted outputs - example for 1 submodule

An additional benefit of phase shifted PWMs is shown in Figure 487. In this case, an H-Bridge circuit is driven by 4 PWM signals to control the voltage waveform on the primary of a transformer. Both left and right side PWMs are configured to generate a square wave with 50% duty cycle. This works for the H-Bridge since no narrow pulse widths are generated reducing the high frequency switching requirements of the transistors. The RMS value of this waveform is directly controlled by the amount of phase shift of the square waves. Regardless of the phase shift, no DC component appears in the load voltage as long as the duty cycle of each square wave remains at 50% suitable for transformer loads. As a result, this topology is frequently used in industrial welders to adjust the amount of energy delivered to the weld arc.

NOTE

The square wave on the right side of the H-Bridge is phase shifted compared to the left side of the H-Bridge. As a result, the transformer primary sees the bottom waveform across its terminals.

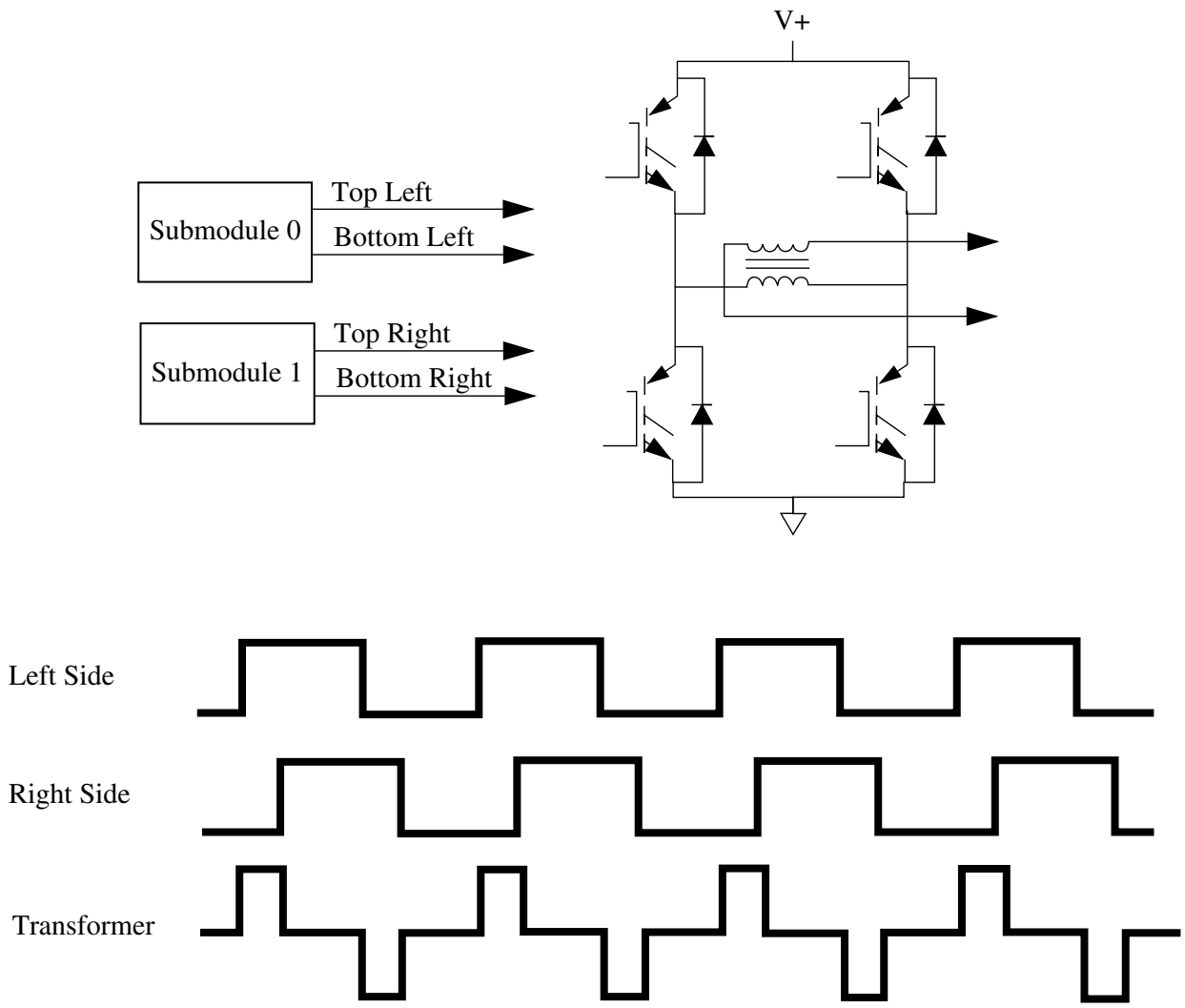


Figure 487. Phase Shifted PWMs Applied to a Transformer Primary

67.3.2.4 Double switching PWMs

Double switching PWM output is supported to aid in single shunt current measurement and three-phase reconstruction. This method supports two independent rising edges and two independent falling edges per PWM cycle. The VAL2 and VAL3 registers are used to generate the even channel (labeled as PWM_A in Figure 488) while VAL4 and VAL5 are used to generate the odd channel. The two channels (PWM23 or PWM_A and PWM45 or PWM_B from force out logic) are combined using XOR logic (force out logic) as the following figure shows. The DBLPWM signal can be run through the deadtime insertion logic.

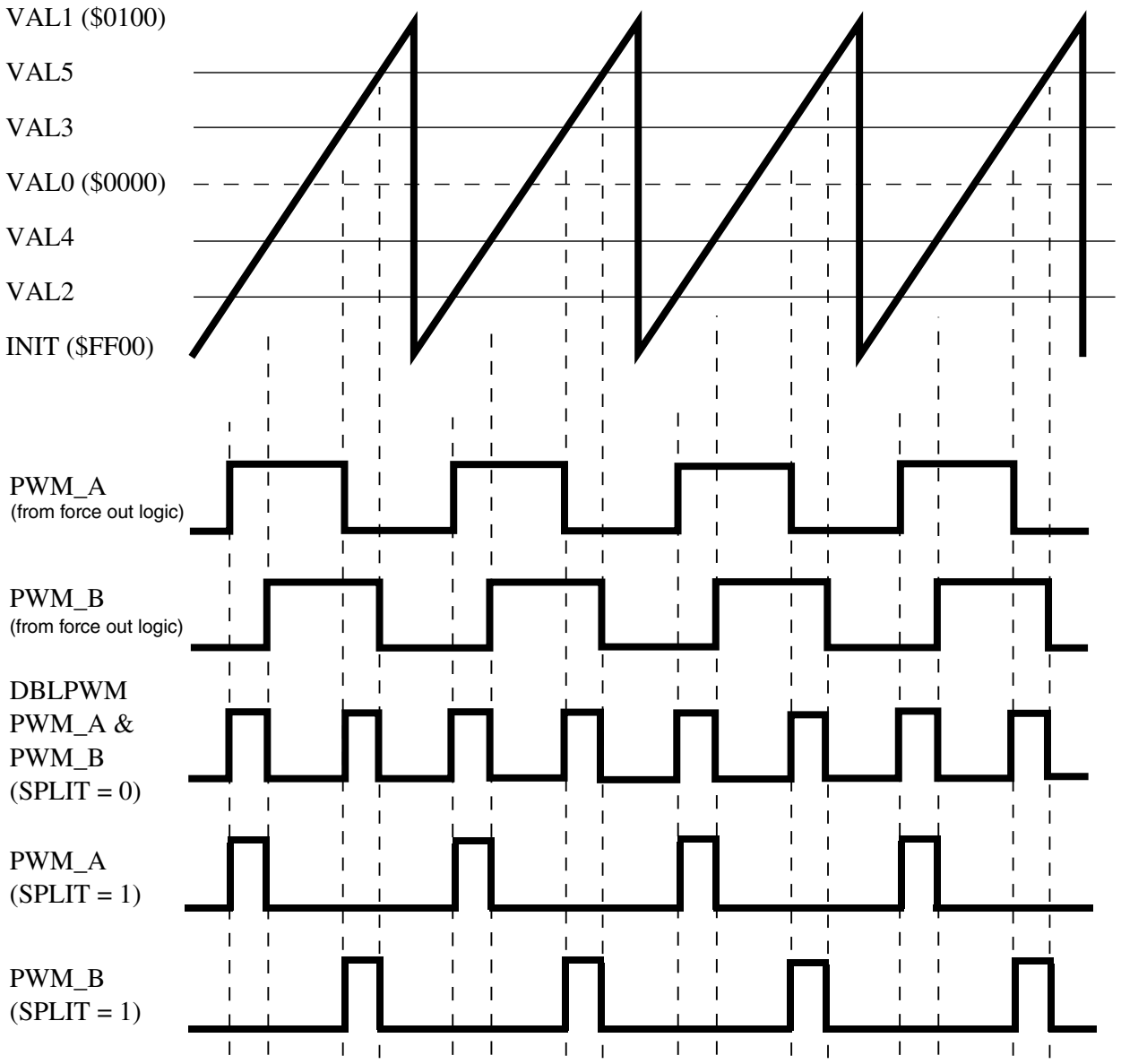


Figure 488. Double switching output example

67.3.2.5 ADC triggering

In cases where the timing of the ADC triggering is critical, it must be scheduled as a hardware event instead of software activated. With this PWM module, multiple ADC triggers can be generated in hardware per PWM cycle without the requirement of another timer module. Figure 489 shows how this is accomplished. When specifying a complementary mode of operation, only two edge comparators are required to generate the output PWM signals for a given submodule. This means the other comparators are free to perform other functions. In this example, the software does not need to quickly respond after the first conversion to set up other conversions that must occur in the same PWM cycle.

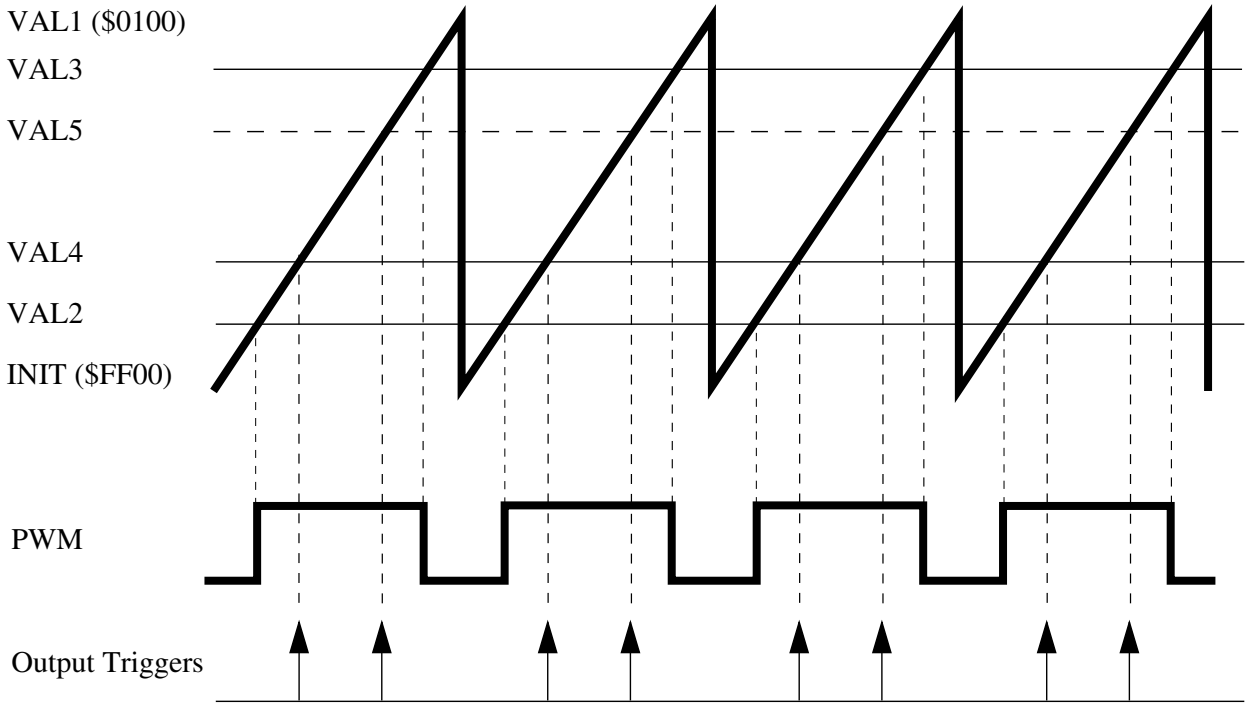


Figure 489. Multiple output trigger generation in hardware

Because each submodule has its own timer, it is possible for each submodule to run at a different frequency. One of the options possible with this PWM module is to have one or more submodules running at a lower frequency, but still synchronized to the timer in submodule0. Figure 490 shows how this feature can be used to schedule ADC triggers over multiple PWM cycles. You can use the lower-frequency submodule to control the sampling frequency of the software control algorithm where multiple ADC triggers can now be scheduled over the entire sampling period. In Figure 490, *all* submodule comparators are shown being used for ADC trigger generation.

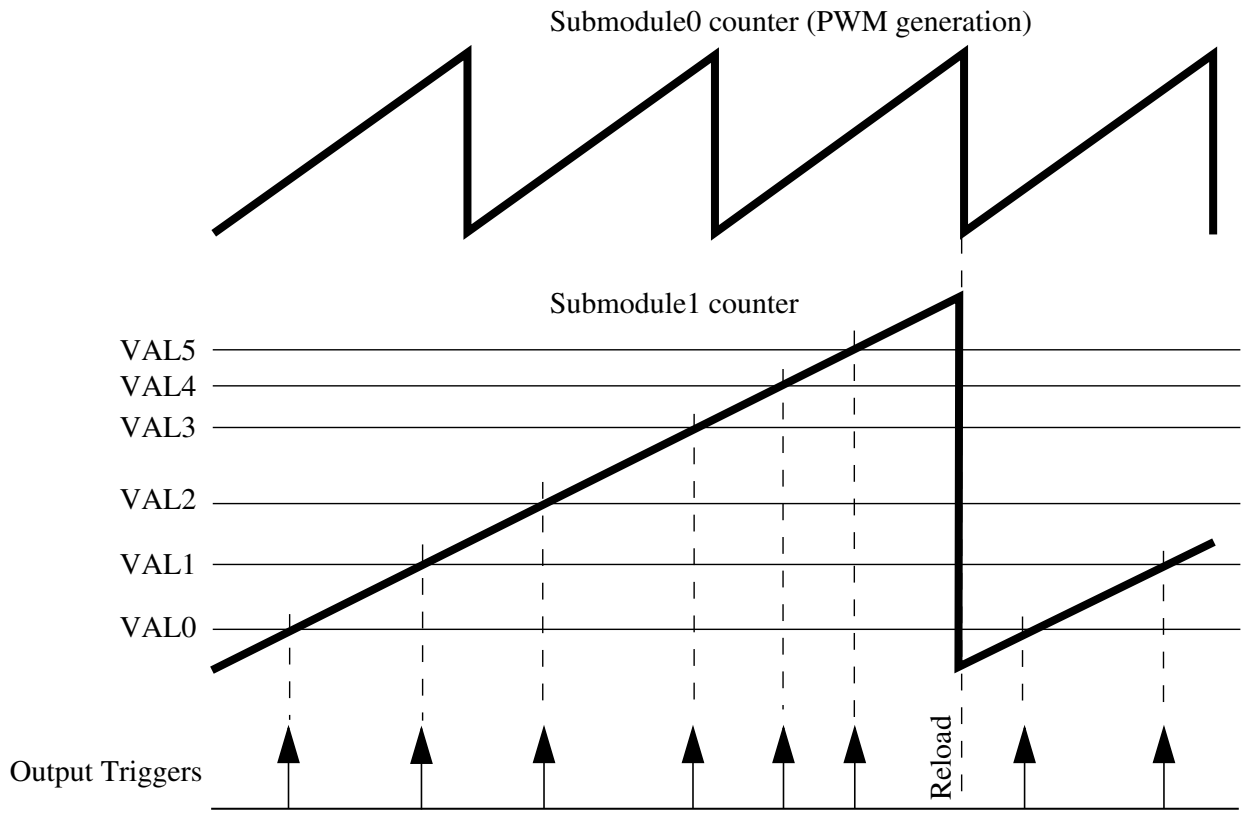


Figure 490. Multiple output triggers over several PWM cycles

67.3.2.6 Enhanced capture capabilities (E-Capture)

When a PWM pin is not used for PWM generation, it can be used to perform input captures. For PWM generation, both edges of the PWM signals are specified via separate compare register values. When programmed for input capture, both of these registers work on the same pin to capture multiple edges, toggling from one to the other in either a free running or one-shot fashion. Programming the desired edge of each capture circuit, period, and pulse width of an input signal can easily be measured without the requirement to re-arm the circuit. In addition, each edge of the input signal can clock an 8-bit counter where the counter output is compared to a user specified value (EDGCMP). When the counter output equals EDGCMP, the value of the submodule timer is captured and the counter is automatically reset. This feature counts a specified number of edge events and then performs a capture and interrupt. The following figure illustrates some of the functionality of the E-Capture circuit.

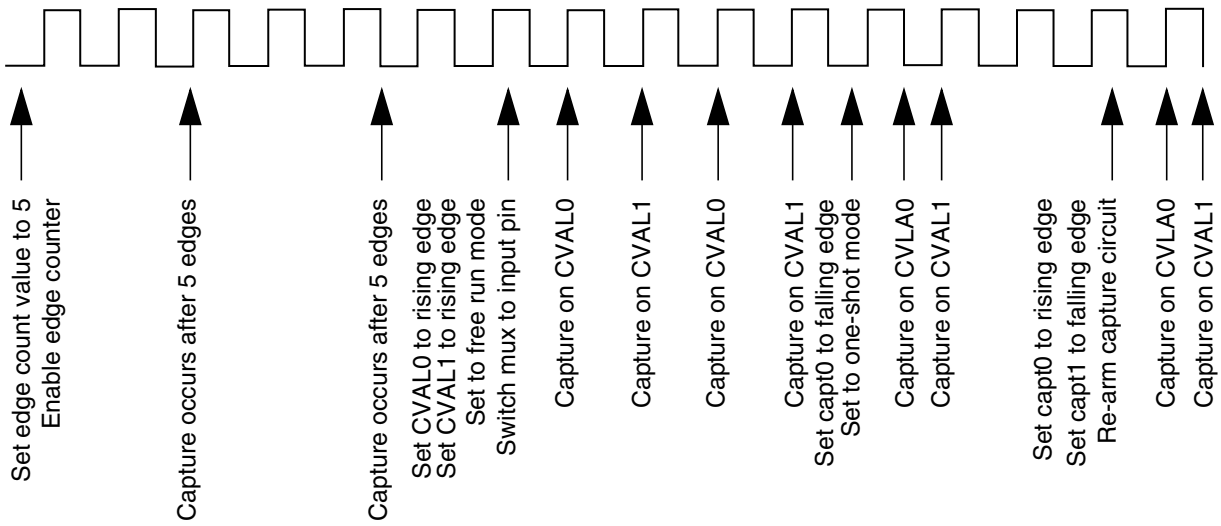


Figure 491. Capture capabilities of the E-Capture circuit

When a submodule is used for PWM generation, its timer counts up to the modulus value used to specify the PWM frequency and then is re-initialized. Therefore, using this timer for input captures on one of the other pins (for example, PWM_X) has limited utility since it does not count through all of the numbers and the timer reset represents a discontinuity in the 16-bit number range. However, when measuring a signal that is synchronous to the PWM frequency, the timer modulus range is suited for the application. As shown in Figure 492, the output of a PWM power stage is connected to the PWM_X pin that is configured for free running input captures. Specifically, the CVAL0 capture circuitry is programmed for rising edges and the CVAL1 capture circuitry is set for falling edges. This results in new load pulse width data acquired every PWM cycle. To calculate the pulse width, subtract the CVAL0 register value from the CVAL1 register value. This measurement is beneficial when performing deadtime distortion correction on a half bridge circuit driving an inductive load. Also, these values can be directly compared to the VALx registers responsible for generating the PWM outputs to obtain a measurement of system propagation delays. For details, refer to the separate discussion of deadtime distortion correction.

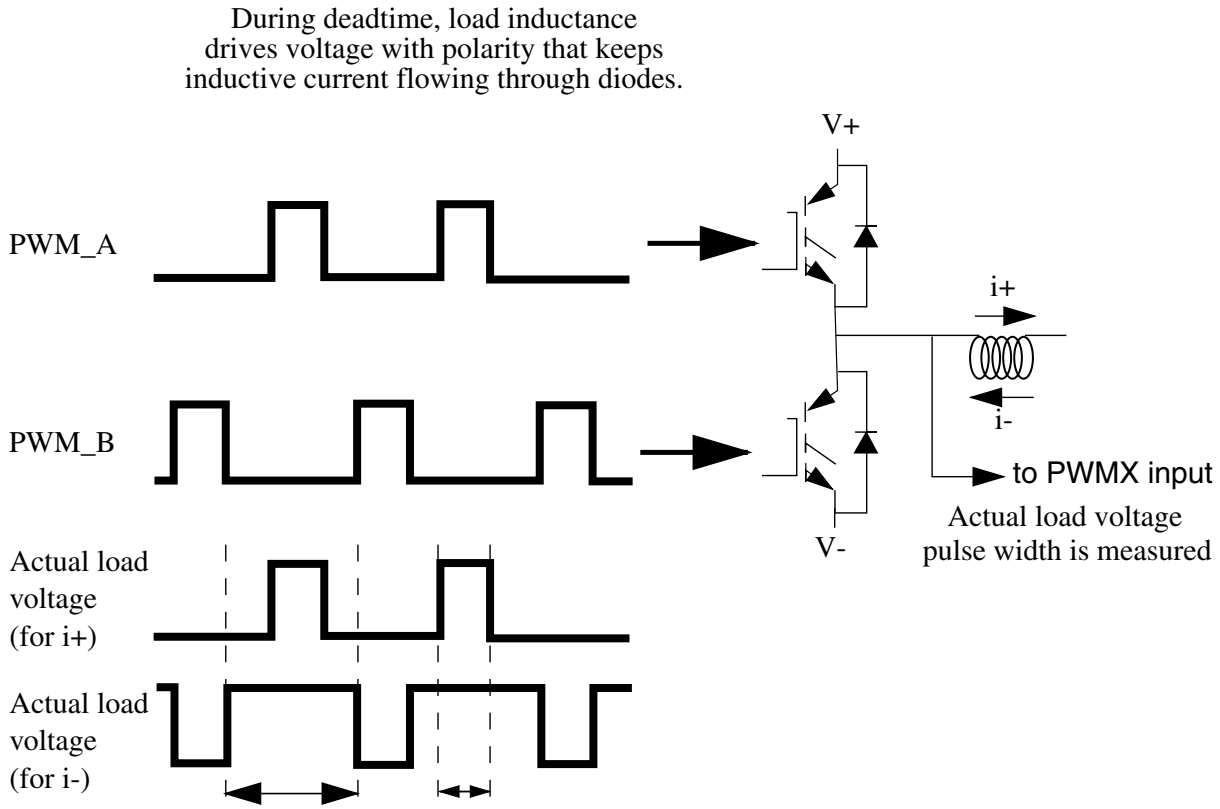


Figure 492. Output pulse width measurement possible with the E-capture circuit

67.3.2.7 Synchronous switching of multiple outputs

Before the PWM signals are routed to the output pins, they are processed by a hardware block that permits all submodule outputs to be switched synchronously. This feature is useful in commutated motor applications where the next commutation state can be laid in ahead of time and then immediately switched to the outputs when the appropriate condition or time is reached. All the changes occur immediately after the trigger event occurs eliminating any interrupt latency and also the changes occurs synchronously on all submodule outputs.

The synchronous output switching is accomplished via a signal called FORCE_OUT. This signal originates from the local FORCE bit within the submodule, from submodule0, or from external to the PWM module, and in most cases, is supplied from an external timer channel configured for output compare. In a typical application, software sets up the desired states of the output pins in preparation for the next FORCE_OUT event. This selection lays dormant until the FORCE_OUT signal transitions and then all outputs are switched simultaneously. The signal switching is performed upstream from the deadtime generator so that any abrupt changes that occur do not violate deadtime on the power stage when in complementary mode.

Figure 493 shows an application that can benefit from this feature. On a brushless DC motor in many cases, it is required to spin the motor without need of hall-effect sensor feedback. Instead, the back EMF of the motor phases is monitored and this information is used to schedule the next commutation event. The top waveforms of Figure 493 represent these back EMF signals. Timer compare events (represented by the long vertical lines in the diagram) are scheduled based on the zero crossings of the back-EMF waveforms. The PWM module is configured via software ahead of time with the next state of the PWM pins in anticipation of the compare event. When it happens, the output compare of the timer drives the FORCE_OUT signal which immediately changes the state of the PWM pins to the next commutation state with no software latency.

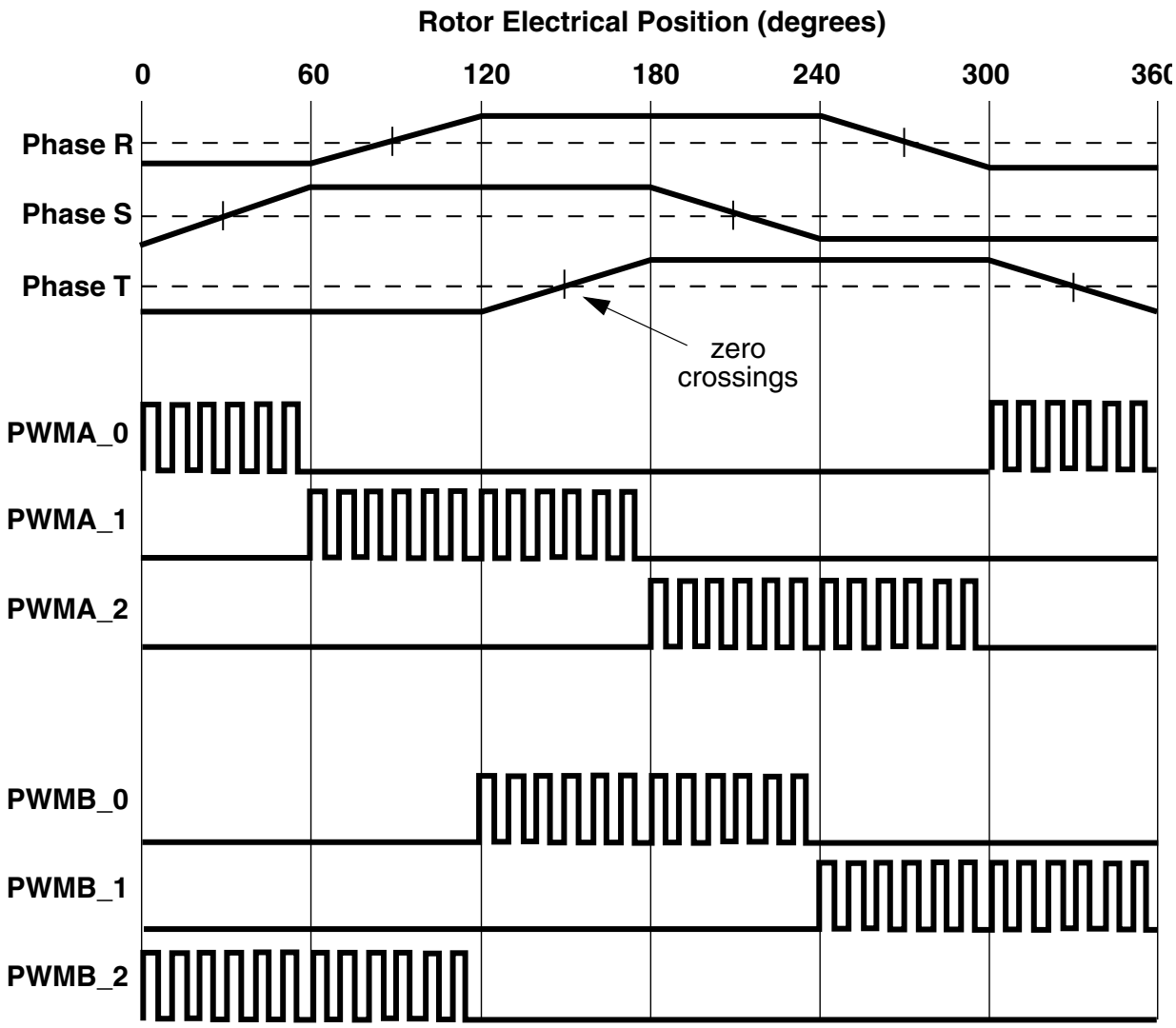


Figure 493. Sensorless BLDC commutation using the force out function

67.3.3 Operation

This section describes the implementation of various sections of the PWM in detail.

The following figure is a high-level block diagram of output PWM generation.

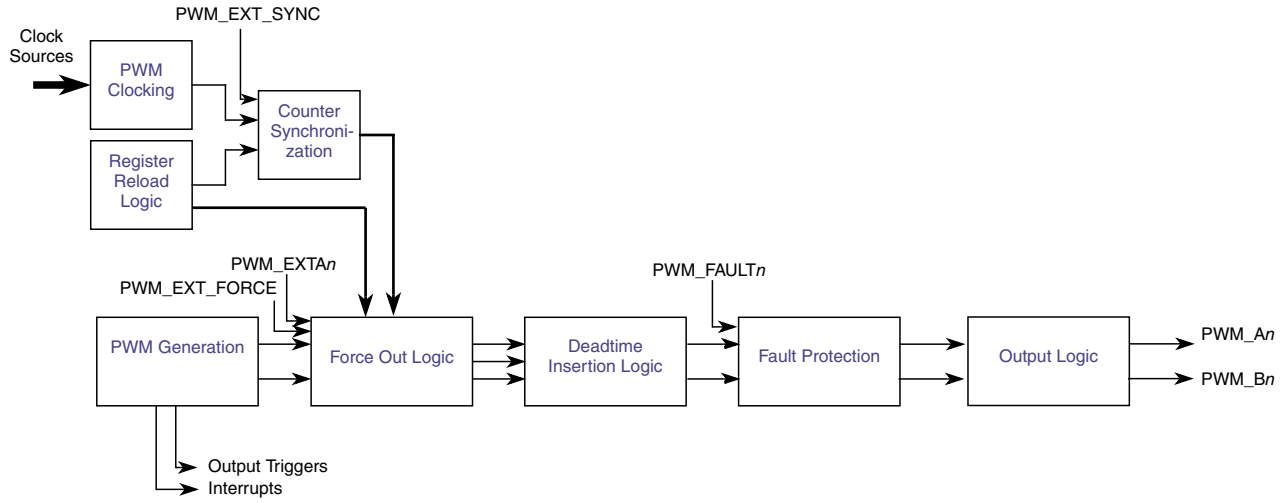


Figure 494. High-level output PWM generation block diagram

67.3.3.1 Register reload logic

The register reload logic is used to determine when the outer set of registers for all double buffered register pairs will be transferred to the inner set of registers. The register reload event can be scheduled to occur every "n" PWM cycles using CTRL[LDFQ] and CTRL[FULL], which is defined by VAL1 register. A half cycle reload option is also supported (CTRL[HALF]) where the reload can take place in the middle of a PWM cycle. The half cycle point is defined by the VAL0 register and does not have to be exactly in the middle of the PWM cycle.

As shown in Figure 495 the reload signal from submodule0 can be broadcast as the Master Reload signal allowing the reload logic from submodule0 to control the reload of registers in other submodules.

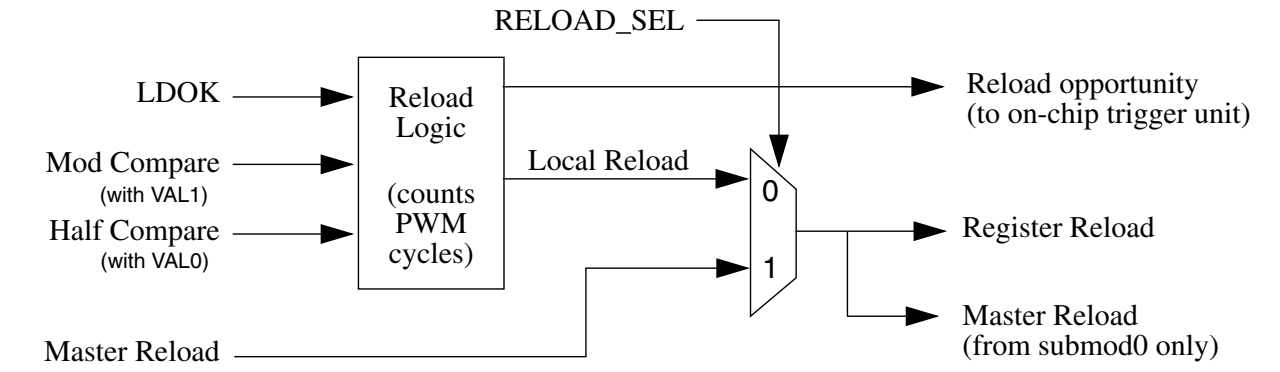


Figure 495. Register reload logic of each submodule

67.3.3.2 Counter synchronization

As shown in Figure 496, the 16-bit counter counts up until its output equals VAL1 which is used to specify the counter modulus value. The resulting compare causes a rising edge to occur on the Local Synchronization signal which is one of four possible sources used to cause the 16-bit counter to be initialized with INIT. If Local Synchronization is selected as the counter initialization signal, VAL1 within the submodule effectively controls the timer period (and then the PWM frequency generated by that submodule), and everything operates or functions at a local level.

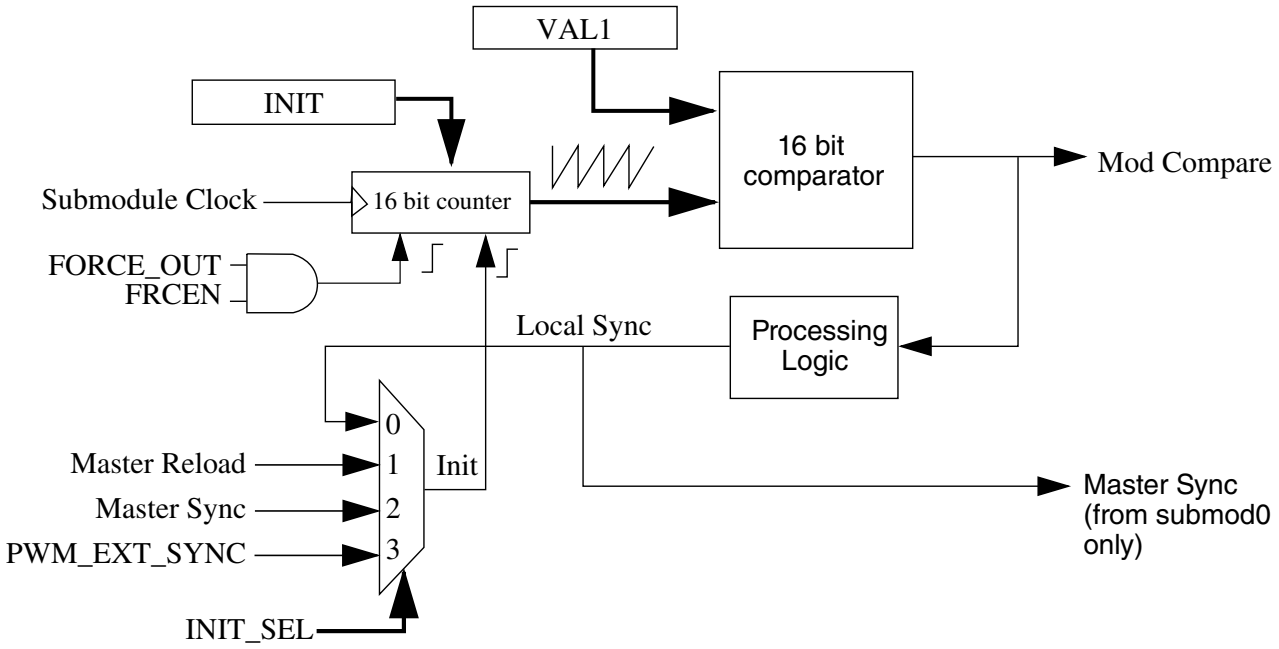


Figure 496. Counter synchronization for each submodule

The Master Synchronization signal originates as the Local Sync from submodule0. If configured to do so, the timer period of any submodule can be locked to the period of the timer in submodule0.

The PWM_EXT_SYNC signal originates on chip or off chip depending on the system architecture. This signal may be selected as the source for counter initialization so that an external source can control the period of all submodules.

If the Master Reload signal is selected as the source for counter initialization, then the period of the counter will be locked to the register reload frequency of submodule0. Since the reload frequency is commensurate to the sampling frequency of the software control algorithm, the submodule counter period is equal the sampling period. As a result, this timer can be used to generate output compares or output triggers over the entire sampling period which may consist of several PWM cycles. The Master Reload signal can only originate from submodule0.

The counter can optionally initialize upon the assertion of the FORCE_OUT signal assuming that CTRL2[FRCEN] is set. As shown in Figure 496, this constitutes a second initialization input into the counter, which causes the counter to initialize regardless of which signal is selected as the counter initialization signal. A forced initialization causes a register reload if MCTRL[LDOK] is set.

The counter can be initialized by FORCE_OUT signal only when MCTRL[RUN] = 1 or CTLR2[CLK_SEL] = 2 which chooses auxiliary clock from SM0.

The FORCE_OUT signal is provided mainly for commutated applications. When PWM signals are commutated on an inverter controlling a brushless DC motor, it is necessary to restart the PWM cycle at the beginning of the commutation interval. This action effectively resynchronizes the PWM waveform to the commutation timing. Otherwise, the average voltage applied to a motor winding integrated over the entire commutation interval will be a function of the timing between the asynchronous commutation event for the PWM cycle. The effect is more critical at higher motor speeds where each commutation interval may consist of only a few PWM cycles. If the counter is not initialized at the start of each commutation interval, the result is an oscillation caused by the beating between the PWM frequency and the commutation frequency.

67.3.3.3 PWM generation

Figure 497 illustrates how PWM generation is accomplished in each submodule. In each case, two comparators and associated VALx registers are utilized for each PWM output signal. One comparator and VALx register are used to control the turn-on edge, while a second comparator and VALx register control the turn-off edge.

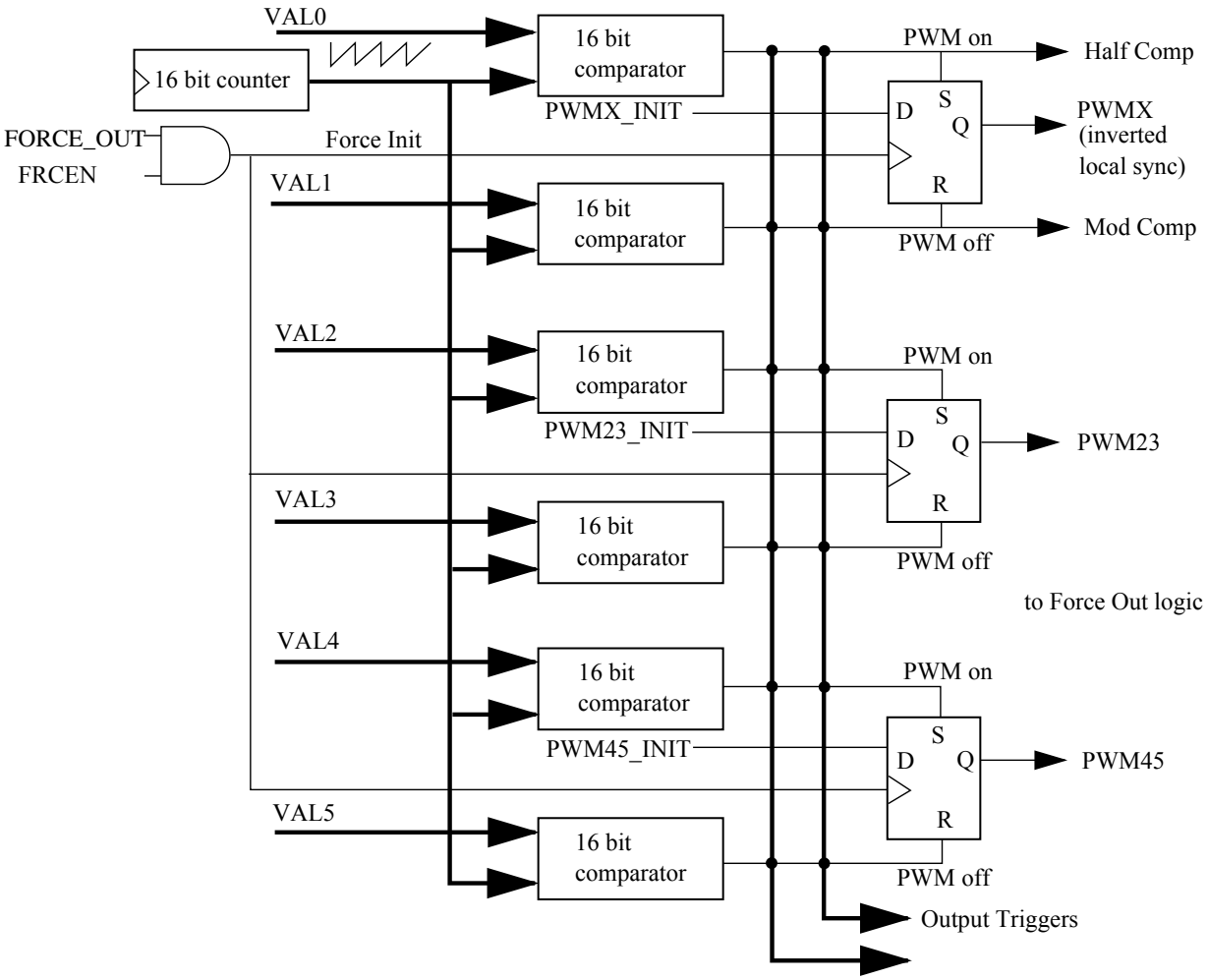


Figure 497. PWM Generation Hardware of each submodule

The generation of the Local Synchronization signal is performed exactly the same way as the other PWM signals in the submodule. While comparator 0 causes a falling edge of the Local Synchronization signal, comparator 1 generates a rising edge. Comparator 1 is also hardwired to the reload logic to generate the full cycle reload indicator.

If VAL1 is controlling the modulus of the counter and VAL0 is half of the VAL1 register minus the INIT value, then the half cycle reload pulse occurs exactly half way through the timer count period and the Local Synchronization will have a 50% duty cycle. On the other hand, if the VAL1 and VAL0 registers are not required for register reloading or counter initialization, they can be used to modulate the duty cycle of the Local Synchronization signal, effectively turning it into an auxiliary PWM signal (PWM_X) assuming that the PWM_X pin is not being used for another function such as input capture or deadtime distortion correction. Including the Local Synchronization signal, each submodule is capable of generating three PWM signals where software has complete control over each edge of each of the signals.

If the comparators and edge value registers are not required for PWM generation, they can be used for other functions such as output compares, generating output triggers, or generating interrupts at timed intervals.

The 16-bit comparators shown in Figure 497 are "equal to" comparators. In addition, if both the set and reset of the flip-flop are asserted, then the flop output goes to 0.

67.3.3.4 Output compare capabilities

By using the VALx registers in conjunction with the submodule timer and 16-bit comparators, buffered output compare functionality can be achieved with no additional hardware required. Specifically, the following output compare functions are possible:

- An output compare sets the output high
- An output compare sets the output low
- An output compare generates an interrupt
- An output compare generates an output trigger

In PWM generation, an output compare is initiated by programming a VALx register for a timer compare, which in turn causes the output of the D flip-flop to either set or reset. For example, if an output compare is desired on the PWM_A signal that sets it high, VAL2 would be programmed with the counter value where the output compare should take place. However, to prevent the D flip-flop from being reset again after the compare has occurred, the VAL3 register must be programmed to a value out of the modulus range of the counter. Therefore, a comparison that would result in resetting the D flip-flop output would never occur. Conversely, if an output compare is desired on the PWM_A signal that sets it low, the VAL3 register is programmed with the appropriate count value and the VAL2 register is programmed with a value out of the counter modulus range. Regardless of whether a high compare or low compare is programmed, an interrupt, or output trigger can be generated when the compare event occurs.

67.3.3.5 Force out logic

For each submodule, the software can select between eight signal sources for the FORCE_OUT signal depending on the chip architecture:

1. Local CTRL2[FORCE]
2. Master Force signal from submodule0
3. Local Reload signal
4. Master Reload signal from submodule0
5. Local Synchronization signal
6. Master Synchronization signal from submodule0
7. EXT_SYNC signal from on or off chip
8. EXT_FORCE signal from on or off chip

The local signals are used to change the signals on the output pins of the submodule without regard for synchronization with other submodules. However, if it is required that all signals on all submodule outputs change at the same time, the Master, EXT_SYNC, or EXT_FORCE signals must be selected.

Figure 498 illustrates the Force logic. The SEL23 and SEL45 fields each choose from one of four signals that can be supplied to the submodule outputs: the PWM signal, the inverted PWM signal, a binary level specified by software via the OUT23 and OUT45 bits, or the PWM_EXTA alternate external control signals. The selection can be determined ahead of time, and when a FORCE_OUT event occurs, these values are presented to the signal selection mux that immediately switches the requested signal to the output of the mux for further processing downstream.

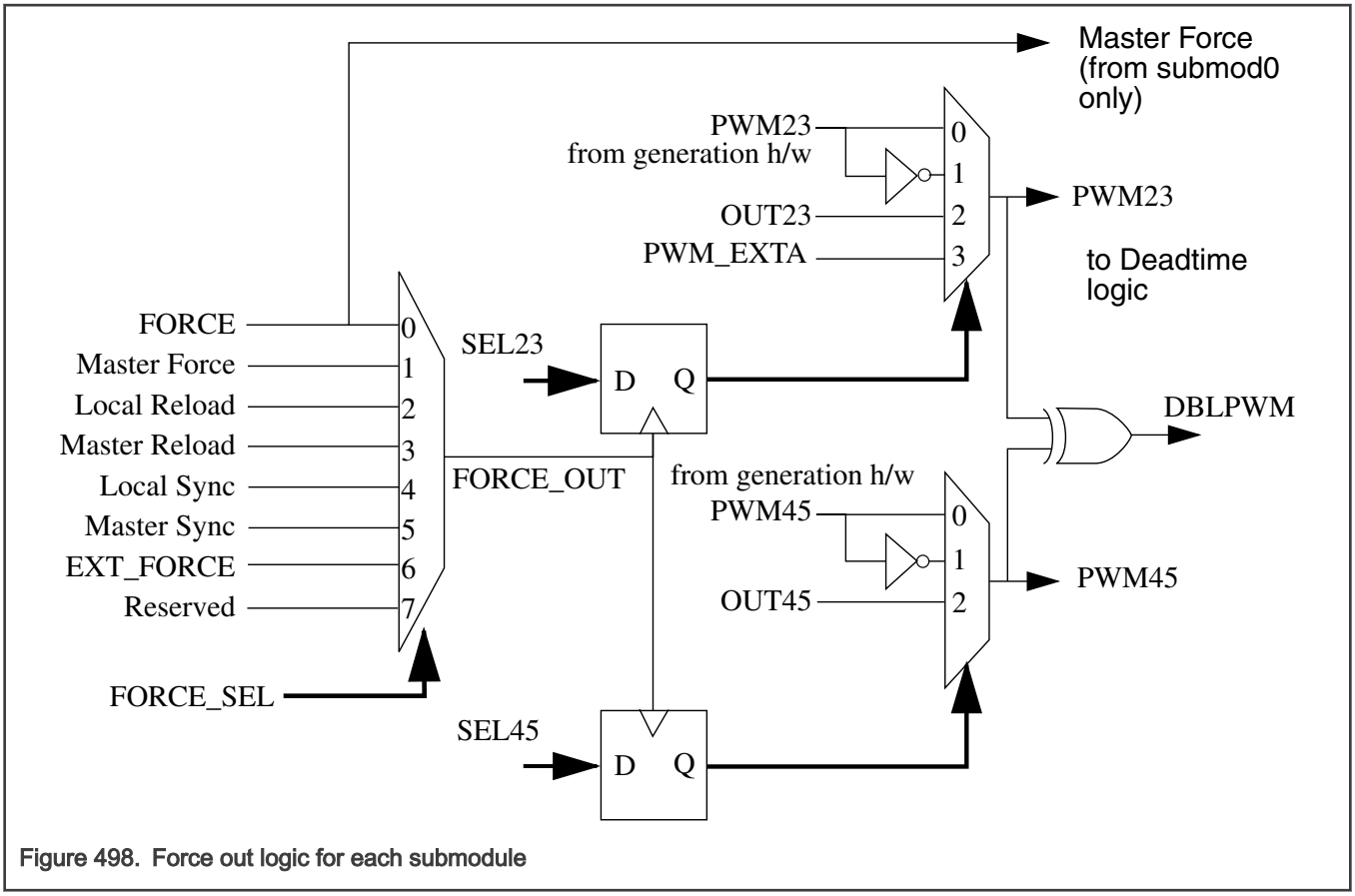


Figure 498. Force out logic for each submodule

The local CTRL2[FORCE] signal of submodule0 can be broadcast as the Master Force signal to other submodules. This feature allows the CTRL2[FORCE] of submodule0 to synchronously update all of the submodule outputs at the same time. The EXT_FORCE signal originates from outside the PWM module from a source such as a timer or digital comparators in the Analog-to-Digital Converter.

67.3.3.6 Independent or complementary channel operation

Writing a logic one to CTRL2[INDEP] configures the pair of PWM outputs as two independent PWM channels. Each PWM output is controlled by its own VALx pair operating independently of the other output.

Writing a logic zero to CTRL2[INDEP] configures the PWM output as a pair of complementary channels. The PWM pins are paired as shown in Figure 3-16 in complementary channel operation. The signal that is connected to the output pin (PWM23 or PWM45) is determined by MCTRL[IPOL].

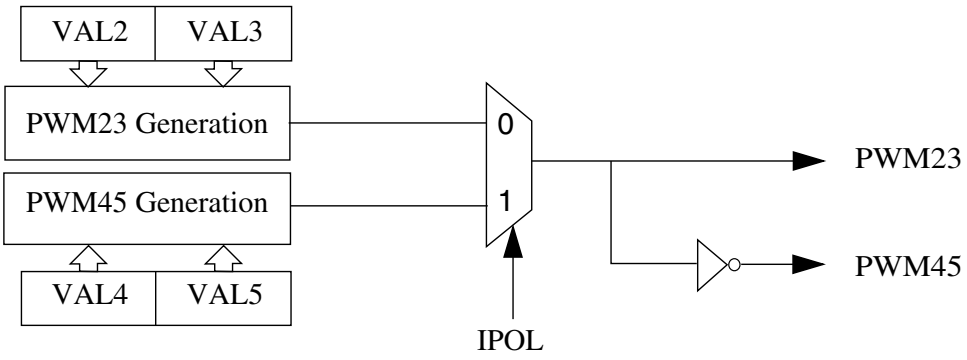


Figure 499. Complementary channel pair for each submodule

The complementary channel operation is for driving top and bottom transistors in a motor drive circuit, as shown in Figure 500.

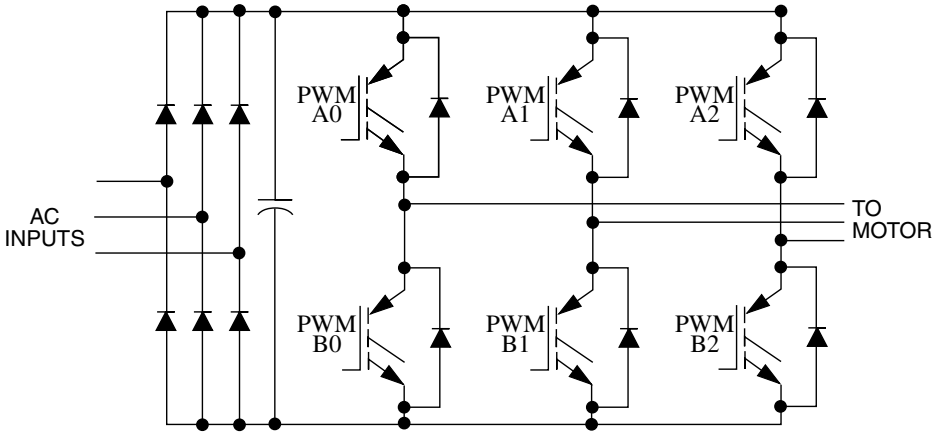


Figure 500. Typical 3 phase AC motor drive

The complementary operation allows the use of the deadtime insertion feature.

67.3.3.7 Deadtime insertion logic

The following figure shows the deadtime insertion logic of each submodule which is used to create non-overlapping complementary signals when not in independent mode.

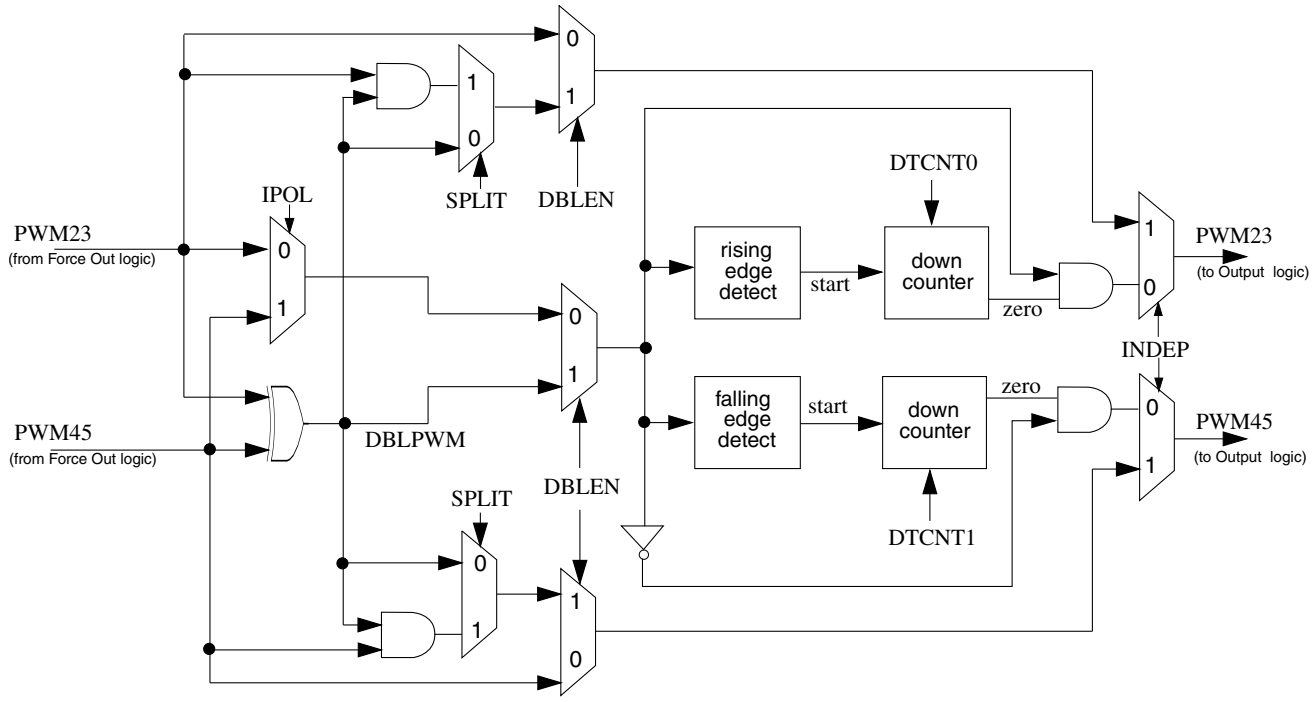


Figure 501. Deadtime Insertion Logic for each submodule

While in the complementary mode, a PWM pair can be used to drive top/bottom transistors, as shown in the figure. When the top PWM channel is active, the bottom PWM channel is inactive, and vice versa.

NOTE

To avoid short-circuiting the DC bus and endangering the transistor, there must be no overlap of conducting intervals between the top and bottom transistors. But the transistor's characteristics make its switching-off time longer than switching-on time. To avoid the conducting overlap of top and bottom transistors, deadtime needs to be inserted in the switching period, as shown in Figure 502.

The deadtime generators automatically insert software-selectable activation delays into the pair of PWM outputs. The deadtime registers (DTCNT0 and DTCNT1) specify the number of IPBus clock cycles to use for deadtime delay. Every time the deadtime generator inputs change state, deadtime is inserted. Deadtime forces both PWM outputs in the pair to the inactive state.

When deadtime is inserted in complementary PWM signals connected to an inverter driving an inductive load, the PWM waveform on the inverter output will have a different duty cycle than what appears on the output pins of the PWM module. This results in a distortion in the voltage applied to the load. A method of correcting this, adding to or subtracting from the PWM value used, is discussed next.

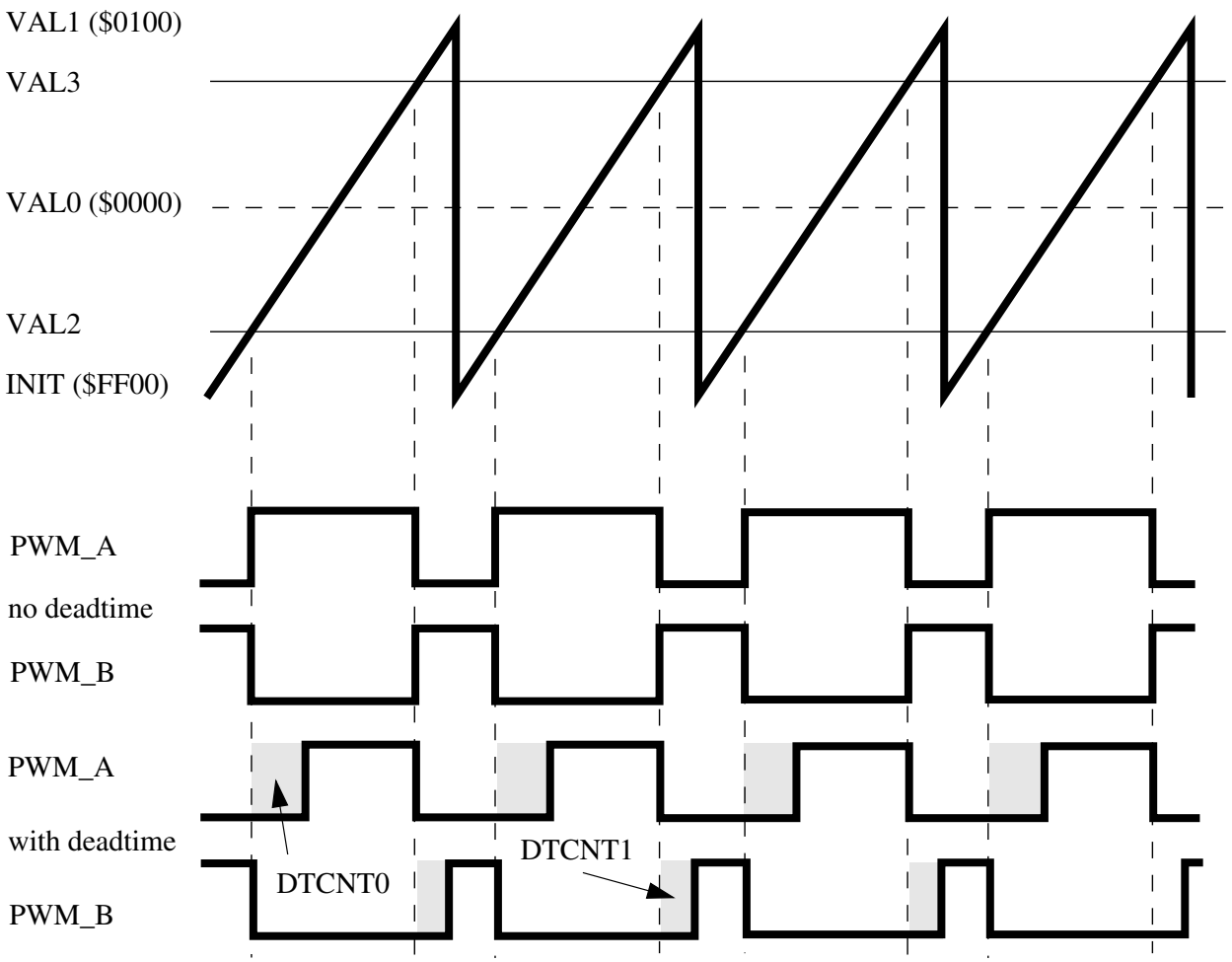


Figure 502. Deadtime insertion

67.3.3.7.1 Top/Bottom correction

In complementary mode, either the top or the bottom transistor controls the output voltage. However, deadtime has to be inserted to avoid overlap of conducting interval between the top and bottom transistor. Both transistors in complementary mode are off during deadtime, allowing the output voltage to be determined by the current status of load and introducing distortion in the output voltage, as shown in Figure 503. On AC induction motors running open-loop, the distortion typically manifests itself as poor low-speed performance, such as torque ripple and rough operation.

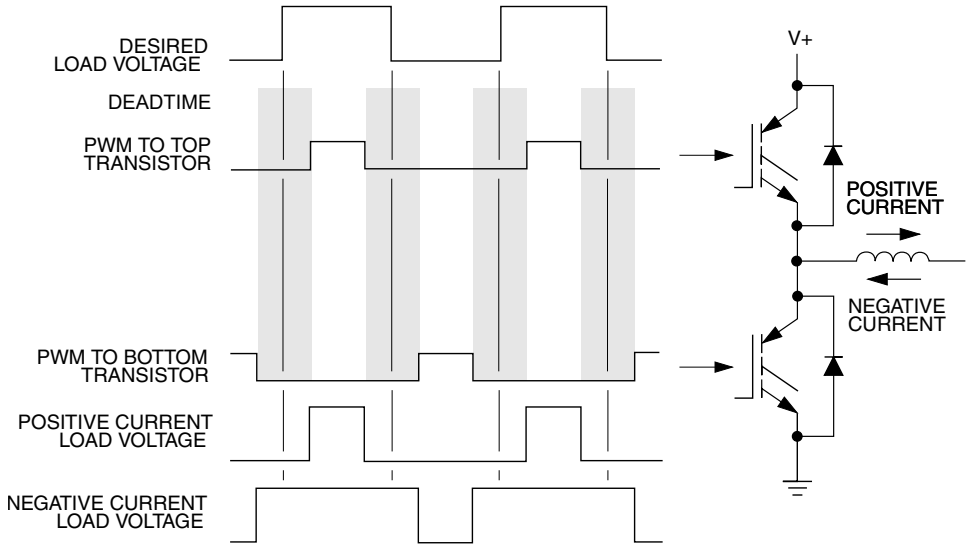


Figure 503. Deadtime distortion

During deadtime, load inductance distorts output voltage by keeping current flowing through the diodes. This deadtime current flow creates a load voltage that varies with the current direction. With a positive current flow, the load voltage during deadtime is equal to the bottom supply, putting the top transistor in control. With a negative current flow, the load voltage during deadtime is equal to the top supply putting the bottom transistor in control.

Remembering that the original PWM pulse widths were shortened by deadtime insertion, the averaged sinusoidal output is less than the desired value. However, when deadtime is inserted, it distorts in the motor current waveform inverter outputs. This distortion is aggravated by different turn-on and turn-off delays of each of the transistors. By giving the PWM module information on which transistor is controlling at a given time this distortion can be corrected.

For a typical circuit in complementary channel operation, only one of the transistors is effective in controlling the output voltage at any given time. This depends on the direction of the motor inverter current for that pair, as shown in Figure 503. To correct distortion one of two different factors must be added to the desired PWM value, depending on whether the top or bottom transistor is controlling the output voltage. Therefore, the software is responsible for calculating both compensated PWM values prior to placing them in the VALx registers. Either the VAL2/VAL3 or the VAL4/VAL5 register pair controls the pulse width at any given time. For a given PWM pair, whether the VAL2/VAL3 or VAL4/VAL5 pair is active depends on either:

- The state of the current status pin, PWMX, for that driver
- The state of the odd/even correction bit, MCTRL[IPOL], for that driver

To correct deadtime distortion, the software can decrease or increase the value in the appropriate VALx register.

- In edge-aligned operation, decreasing or increasing the PWM value by a correction value equal to the deadtime typically compensates for deadtime distortion.
- In center-aligned operation, decreasing or increasing the PWM value by a correction value equal to one-half the deadtime typically compensates for deadtime distortion.

67.3.3.7.2 Manual correction

To detect the current status, the voltage on each PWMX pin is sampled twice in a PWM period, at the end of each deadtime. The value is stored in CTRL[DT]. CTRL[DT] is a timing marker indicating when to toggle between PWM value registers. The software can then set MCTRL[IPOL] to switch between VAL2/VAL3 and VAL4/VAL5 register pairs according to CTRL[DT] values.

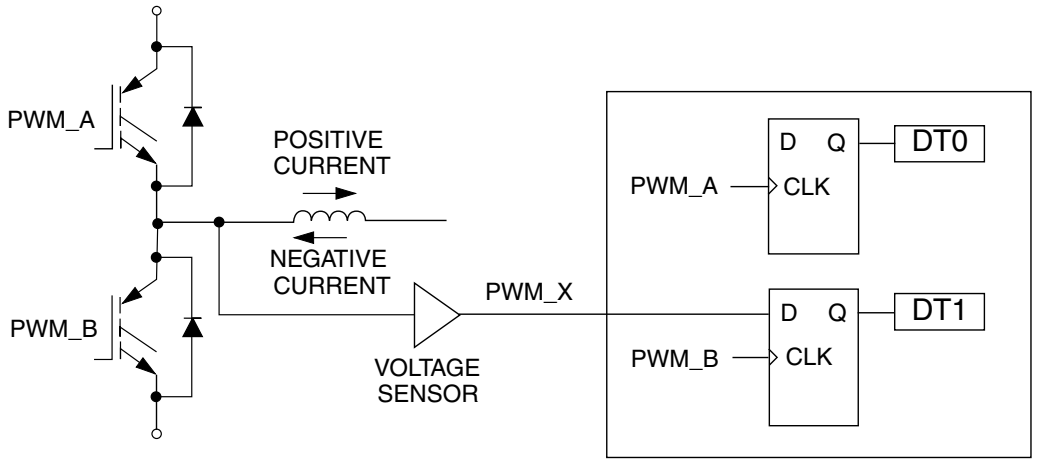


Figure 504. Current-status sense scheme for deadtime correction

Both D flip-flops latch low, CTRL[DT] = 00, during deadtime periods if the current is large and flowing out of the complementary circuit. See the preceding figure. Both D flip-flops latch the high, CTRL[DT] = 11, during deadtime periods if the current is also large and flowing into the complementary circuit.

However, under low-current, the output voltage of the complementary circuit during deadtime is between the high and low levels. The current cannot free-wheel through the opposition anti-body diode, regardless of polarity, giving additional distortion when the current crosses zero. Sampled results are CTRL[DT] = b10. Thus, the best time to change one PWM value register to another is just before the current zero crossing.

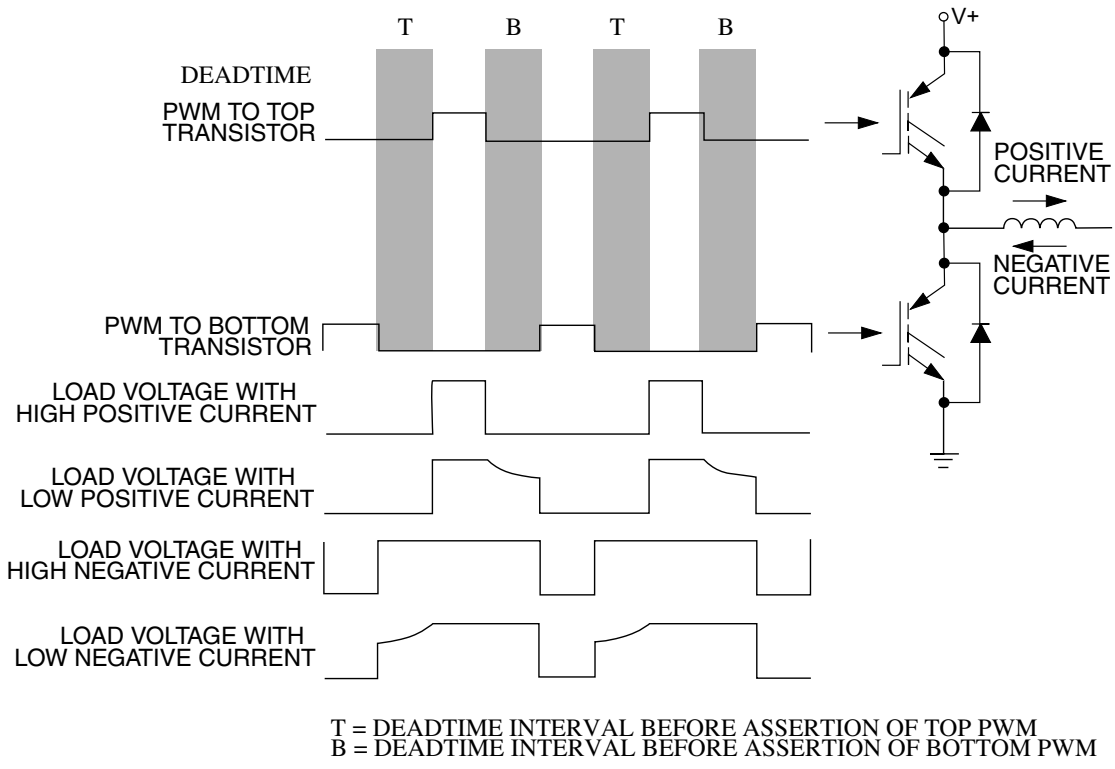


Figure 505. Output voltage waveforms

67.3.3.8 Fractional delay logic

For applications where more resolution than a single IPBus clock period is needed, the fractional delay logic can be used to achieve fine resolution on the rising and falling edges of the PWM_A and PWM_B outputs. Enable the use of the fractional delay logic by setting FRCTRL[FRACx_EN]. The FRACVALx registers act as a fractional clock cycle addition to the turn-on and turn-off count specified by the VAL2, VAL3, VAL4, or VAL5 registers. The FRACVAL1 register acts as a fractional increase in the PWM period as defined by VAL1. If FRACVAL1 is programmed to a non-zero value, then the largest value for the VAL1 register is 0xFFFE for unsigned usage or 0x7FFE for signed usage. This limit is required to avoid counter rollovers when accumulating the fractional additional period.

Both the fractional enables (1, 23, 45) and the fractional values (1-5) are double buffered and reloaded at the same time as the value registers.

Each PWM cycle, the value compare point is increased by 1 when there is overflow on the double buffered value of the fractional register plus the 5-bit accumulated fractional value. The accumulated fractional value starts at zero, so it is impossible to overflow the first PWM cycle.

At the end of each PWM cycle, if the corresponding fractional enable is set then the accumulated fractional value increments by the fractional value (double buffered value). The accumulated fractional value is 5-bits, so in the case of overflow only the remainder is kept. If the corresponding fractional enable is clear, then the accumulated fractional value is reset. This is the only way to reset the accumulated fractional value.

The accumulated fractional values are not accessible to software.

To fine-tune the PWM period using the FRACVAL1 register, if you want a period of 100.25 clock cycles, program VAL1 with 0x0064 and FRACVAL1 with 0x4000. The fractional value will accumulate so that every 4 PWM cycles will be 1 clock cycle longer (101 instead of 100). The rising and falling edges of the PWM outputs will also use the accumulated fraction to delay their edges and maintain a consistent 100.25 cycles spacing between corresponding edges from one cycle to the next.

The results of the fractional delay logic depend on whether or not the PWM submodule has an analog NanoEdge placer block available. With fractional delay enabled, PWM works in digital dithering mode: the value of FRACVALx is accumulated in each PWM period; when the accumulated value of FRACVALx overflows, VALx increments one in the next PWM period.

67.3.3.8.1 Fractional delay logic without NanoEdge placement block

For submodules that are not supported by the NanoEdge placer, the PWM can use dithering to simulate fine edge control. Enable this feature by setting the FRCTRL[FRAC1_EN], FRCTRL[FRAC23_EN], and FRCTRL[FRAC45_EN] bits. The PWM period or the PWM edges will dither from the nearest whole number values to achieve an average value that is equivalent to the programmed fractional value. The added cycles are based on the accumulation of the fractional component. For example, if you want the PWM period to be 50.25 clock cycles, then program VAL1 with 0x0032 and FRACVAL1 with 0x4000. The PWM period will be 50 cycles long most of the time, but will occasionally be 51 cycles long to achieve a long-term average of 50.25 cycles.

In submodules that are not supported by a NanoEdge placer, the clock frequency is not required to be any specific value to achieve proper operation.

67.3.3.9 Output logic

The following figure shows the output logic of each submodule including how each PWM output has individual fault disabling, polarity control, and output enable. This allows for maximum flexibility when interfacing with the external circuitry.

The PWM23 and PWM45 signals which are output from the deadtime logic (as shown in Figure 506) are positive true signals. In other words, a high level on these signals should result in the corresponding transistor in the PWM inverter being turned ON. The voltage level required at the PWM output pin to turn the transistor ON or OFF is a function of the logic between the pin and the transistor. Therefore, it is imperative that the user program OCTRL[POLA] and OCTRL[POLB] before enabling the output pins. A fault condition can result in the PWM output being put in a high-impedance state, forced to a logic 1 state, or forced to a logic 0 state, depending on the values programmed into the OCTRL[PWMxFS] fields.

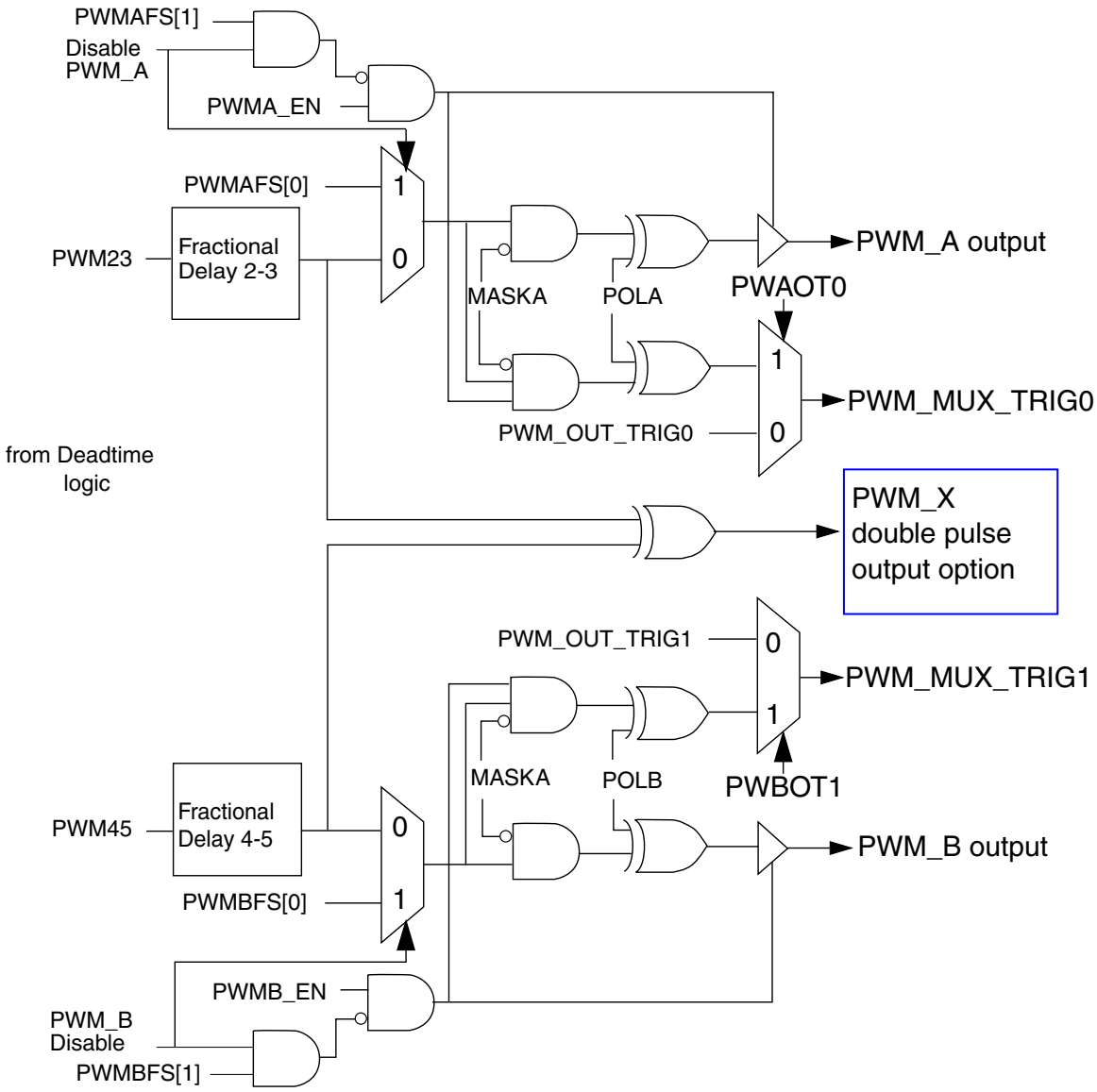


Figure 506. Output logic for each submodule

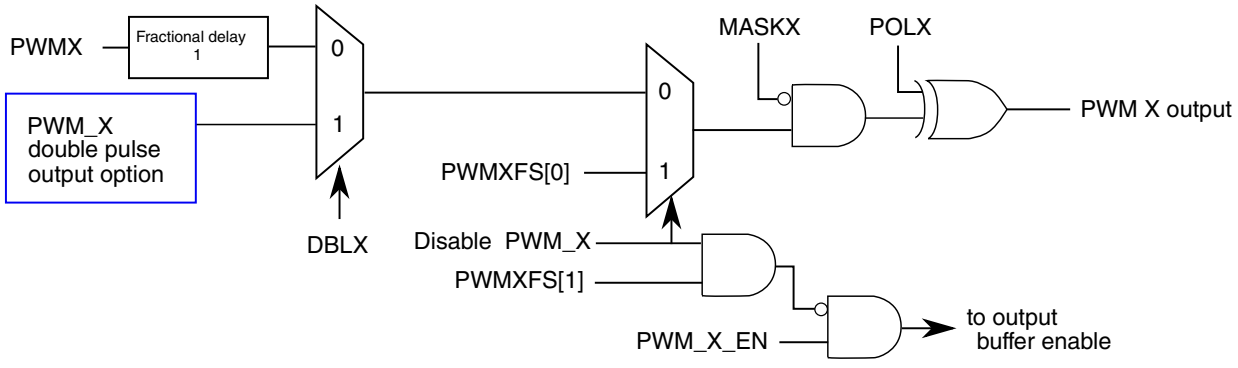


Figure 507. Output logic continued

67.3.3.10 E-Capture

The Enhanced Capture (E-Capture) logic is designed to measure both edges of an input signal. As a result, when a submodule pin is configured for input capture, the CVALx registers associated with that pin are used to record the edge values.

The following figure shows block diagram of the E-Capture circuit. Upon entering the pin input, the signal is split into two paths. One goes straight to a mux input where software can select to pass the signal directly to the capture logic for processing. The other path connects the signal to an 8-bit counter which counts both the rising and falling edges of the signal. The output of this counter is compared to an 8-bit value that is specified by the user (EDGCMPx) and when the two values are equal, the comparator generates a pulse that resets the counter. This pulse is also supplied to the mux input where software can select it to be processed by the capture logic. This feature permits the E-Capture circuit to count up to 256 edge events before initiating a capture event. This feature is useful for dividing down high frequency signals for capture processing so that capture interrupts do not overwhelm the CPU. Also, this feature can be used to generate an interrupt after "n" events have been counted.

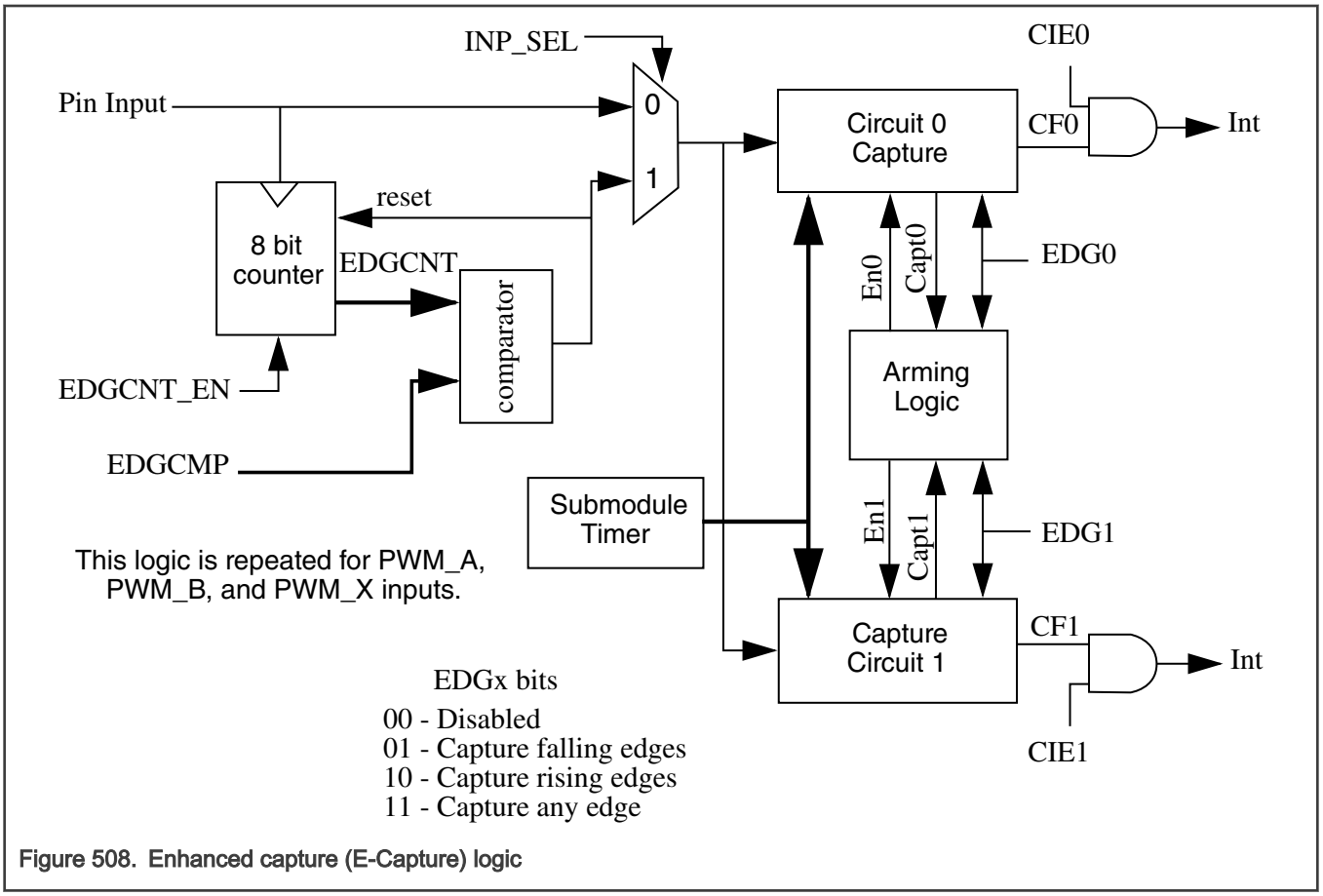


Figure 508. Enhanced capture (E-Capture) logic

Based on the mode selection, the mux selects either the pin input or the compare output from the count/compare circuit to be processed by the capture logic. The selected signal is routed to two separate capture circuits which work in tandem to capture sequential edges of the signal. The type of edge to be captured by each circuit is determined by CAPTCTRLx[EDGx1] and CAPTCTRLx[EDGx0], whose functionality is listed in the above figure. Also, controlling the operation of the capture circuits is the arming logic which allows captures to be performed in a free running (continuous) or one shot mode. In free running mode, the capture sequences will be performed indefinitely. If both capture circuits are enabled, they will work together in a ping-pong style where a capture event from one circuit leads to the arming of the other and vice versa. In one shot mode, only one capture sequence will be performed. If both capture circuits are enabled, capture circuit 0 is first armed and when a capture event occurs, capture circuit 1 is armed. Once the second capture occurs, further captures are disabled until another capture sequence is initiated. Both capture circuits are also capable of generating an interrupt to the CPU.

67.3.3.11 Fault protection

Fault protection can control any combination of PWM output pins. Faults are generated by a logic one on any of the FAULTx pins. This polarity can be changed via FCTRL[FLVL]. Each FAULTx pin can be mapped arbitrarily to any of the PWM outputs. When fault protection hardware disables PWM outputs, the PWM generator continues to run, only the output pins are forced to logic 0, logic 1, or high impedance depending on the values of OCTRL[PWMxFS].

The fault decoder disables PWM pins selected by the fault logic and the disable mapping (DISMAPn) registers. The following figure shows an example of the fault disable logic. Each bank of bits in DISMAPn control the mapping for a single PWM pin. See the following table.

The fault protection is enabled even when the PWM module is not enabled. Therefore, a fault is latched in and must be cleared to prevent an interrupt when the PWM is enabled.

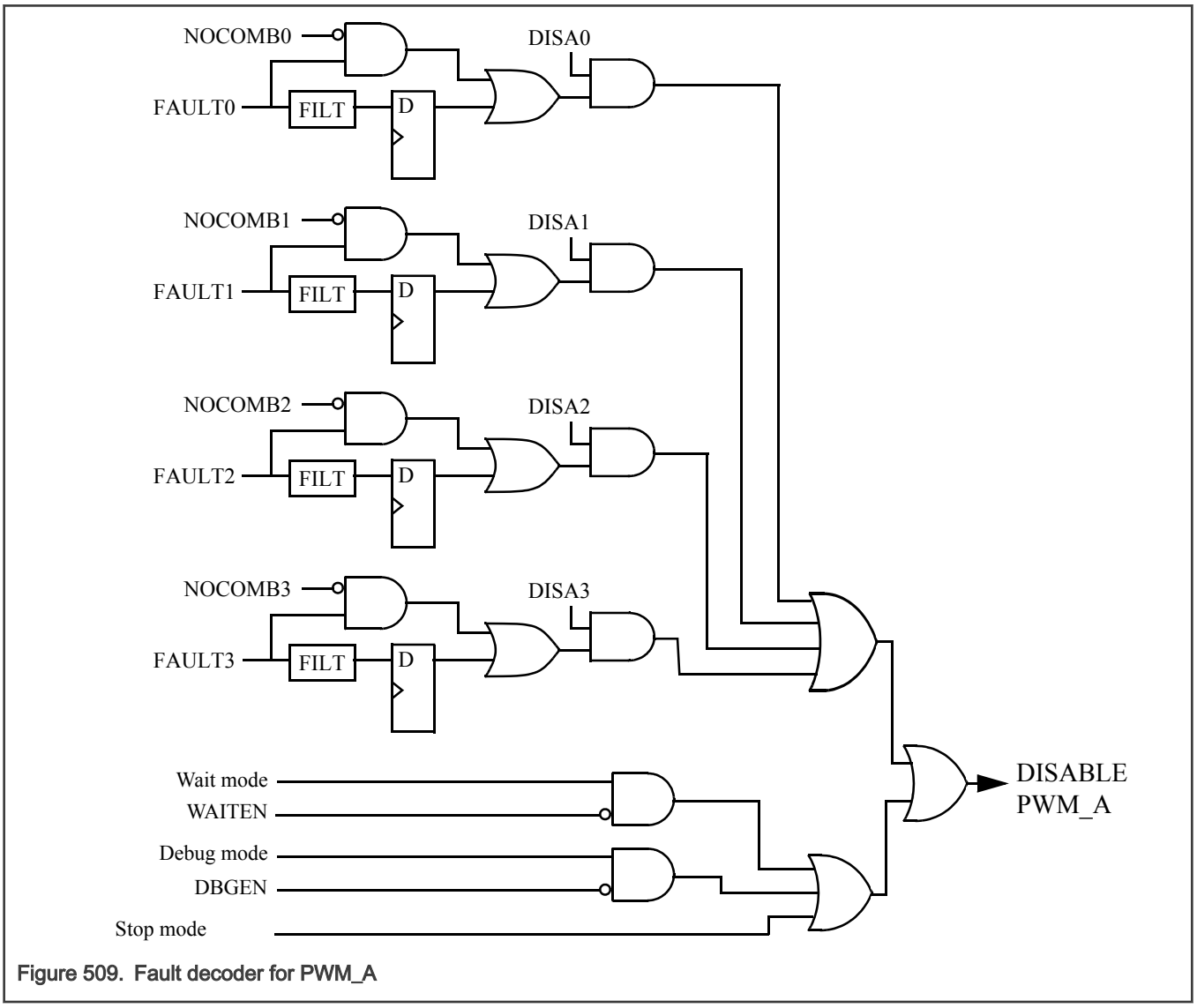


Figure 509. Fault decoder for PWM_A

Table 1053. Fault Mapping

PWM Pin for 1 submodule	Controlling Register Bits
PWM_A	DISMAP0[DIS0A]

Table continues on the next page...

Table 1053. Fault Mapping (continued)

PWM Pin for 1 submodule	Controlling Register Bits
PWM_B	DISMAP0[DIS0B]
PWM_X	DISMAP0[DIS0X]

67.3.3.11.1 Fault pin filter

Each fault pin has a programmable filter that can be bypassed. The sampling period of the filter can be adjusted with FFILT[FILT_PER]. The number of consecutive samples that must agree before an input transition is recognized can be adjusted using FFILT[FILT_CNT]. Setting FFILT[FILT_PER] to all 0 disables the input filter for a given FAULTx pin.

Upon detecting a logic 0 on the filtered FAULTx pin (or a logic 1 if FCTRL[FLVLx] is set), the corresponding FSTS[FFPINx] and FSTS[FFLAGx] bits are set. FSTS[FFPINx] remains set as long as the filtered FAULTx pin is zero. Clear FSTS[FFLAGx] by writing a logic 1 to FSTS[FFLAGx].

If the FIEx, FAULTx pin interrupt enable bit is set, FSTS[FFLAGx] generates a CPU interrupt request. The interrupt request latch remains set until:

- Software clears FSTS[FFLAGx] by writing a logic one to the bit
- Software clears the FIEx bit by writing a logic zero to it
- A reset occurs

Even with the filter enabled, there is a combinational path from the FAULTx inputs to the PWM pins, which in turn bypasses the filter when FCTRL20[NOCOMBx] = 0. This logic is also capable of holding a fault condition in the event of loss of clock to the PWM module.

67.3.3.11.2 Automatic fault clearing

Setting an automatic clearing mode bit, FCTRL[FAUTOx] configures faults from the FAULTx pin for automatic clearing.

When FCTRL[FAUTOx] is set, disabled PWM pins are enabled when the FAULTx pin returns to logic one and a new PWM full or half cycle begins. See the following figure. If FSTS[FFULLx] is set and the fault condition on FAULTx disappears, then the disabled PWM pins are enabled at the start of next full cycle. If FSTS[FHALFx] is set, then the disabled PWM pins are enabled at the start of a half cycle. Clearing FSTS[FFLAGx] does not affect disabled PWM pins when FCTRL[FAUTOx] is set.

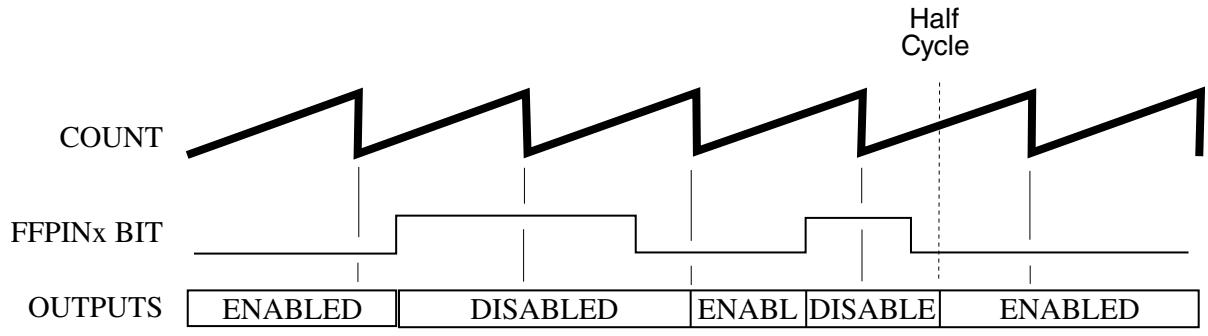


Figure 510. Automatic fault clearing

67.3.3.11.3 Manual fault clearing

Clearing the automatic clearing mode bit, FCTRL[FAUTOx], configures faults from the FAULTx pin for manual clearing:

- If the fault safety mode bits FCTRL[FSAFEx] are clear, then PWM pins disabled by the FAULTx pins are enabled when:
  - Software clears the corresponding FSTS[FFLAGx] flag

— The pins are enabled when the next PWM full or half cycle begins regardless of the logic level detected by the filter at the FAULTx pin, as shown in Figure 511. If FSTS[FFULLx] is set, then the disabled PWM pins are enabled at the start of a full cycle. If FSTS[FHALFx] is set, then the disabled PWM pins are enabled at the start of a half cycle.

- If the fault safety mode bits FCTRL[FSAFEx] are set, then PWM pins disabled by the FAULTx pins are enabled when:
  — Software clears the corresponding FSTS[FFLAGx] flag
  — The filter detects a logic zero on the FAULTx pin at the start of the next PWM full or half cycle boundary, as shown in Figure 512. If FSTS[FFULLx] is set, then the disabled PWM pins are enabled at the start of a full cycle. If FSTS[FHALFx] is set, then the disabled PWM pins are enabled at the start of a half cycle.

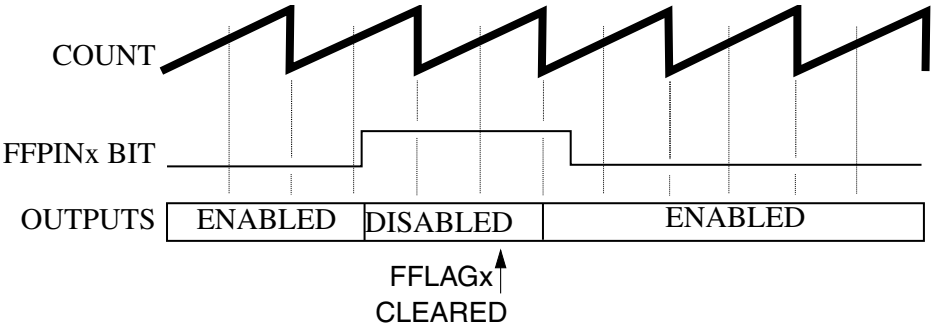


Figure 511. Manual fault clearing (FCTRL[FSAFEx] = 0, FSTS[FFULLx] = 1)

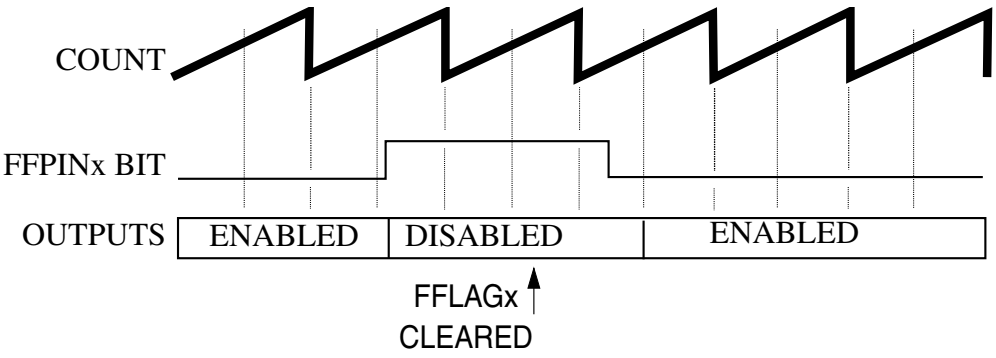


Figure 512. Manual fault clearing (FCTRL[FSAFEx] = 1, FSTS[FHALFx] = 1)

NOTE

Fault protection also applies during software output control when the SEL23 and SEL45 fields are set to select OUT23 and OUT45 bits or PWM_EXTA. Fault clearing still occurs at half or full PWM cycle boundaries while the PWM generator is engaged, MCTRL[RUN] equals one. But the OUTx bits can control the PWM pins while the PWM generator is off, MCTRL[RUN] equals zero. Thus, fault clearing occurs at IPBus cycles while the PWM generator is off and at the start of PWM cycles when the generator is engaged.

67.3.3.11.4 Fault testing

FTST[FTEST] is used to simulate a fault condition on each of the fault inputs within that fault channel.

67.3.3.12 PWM generator loading

67.3.3.12.1 Load enable

MCTRL[LDOK] enables loading of the following PWM generator parameters.

- The prescaler divisor: from CTRL[PRSC]

- The PWM period and pulse width: from the INIT, FRACVALx, and VALx registers

MCTRL[LDOK] allows the software to finish calculating all of these PWM parameters so they can be synchronously updated. The CTRL[PRSC], INIT, and VALx registers are loaded by software into a set of outer buffers. When MCTRL[LDOK] is set, these values are transferred to an inner set of registers at the beginning of the next PWM reload cycle to be used by the PWM generator when CTRL[LDMOD] is cleared. These values can be transferred to the inner set of registers immediately upon setting MCTRL[LDOK] if CTRL[LDMOD] is set. Set MCTRL[LDOK] by reading it when it is a logic zero and then writing a logic one to it. After loading, MCTRL[LDOK] is automatically cleared.

67.3.3.12.2 Load frequency

CTRL[LDFQ] selects an integral loading frequency of one to 16 PWM reload opportunities. CTRL[LDFQ] takes effect at every PWM reload opportunity, regardless of the state of MCTRL[LDOK]. CTRL[HALF] and CTRL[FULL] control reload timing. If CTRL[FULL] is set, a reload opportunity occurs at the end of every PWM cycle when the count equals VAL1. If CTRL[HALF] is set, a reload opportunity occurs at the half cycle when the count equals VAL0. If both CTRL[HALF] and CTRL[FULL] are set, a reload opportunity occurs twice per PWM cycle when the count equals VAL1 and when it equals VAL0.

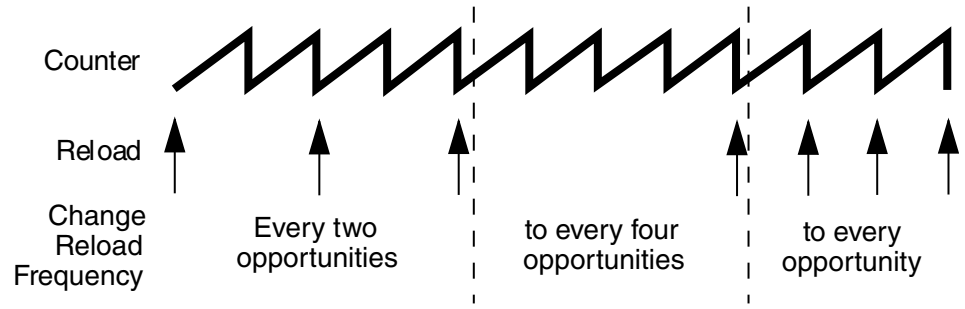


Figure 513. Full cycle reload frequency change

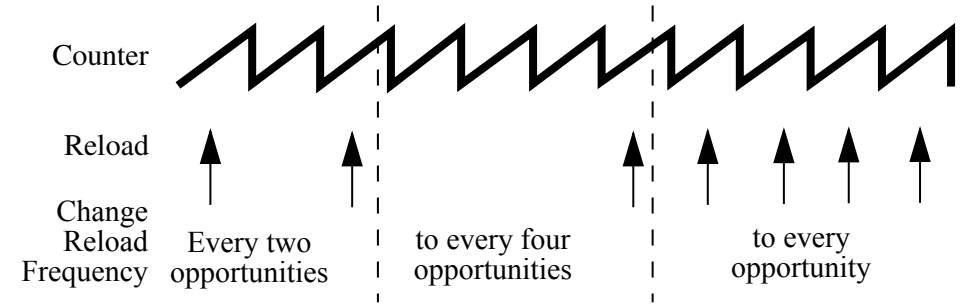


Figure 514. Half cycle reload frequency change

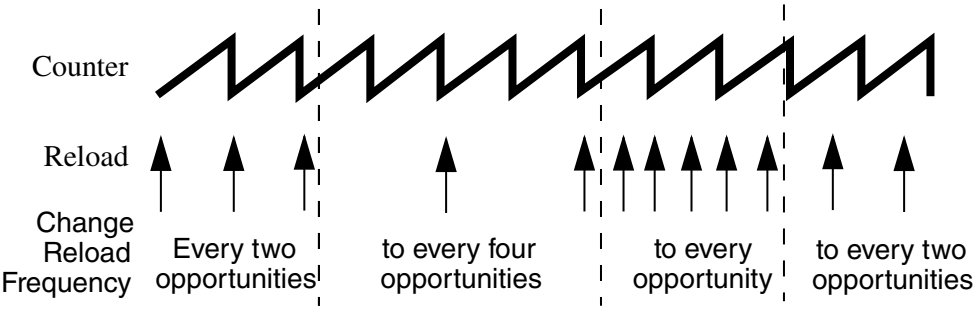


Figure 515. Full and half cycle reload frequency change

67.3.3.12.3 Reload flag

At every reload opportunity the PWM Reload Flag (STS[RF]) is set. Setting STS[RF] happens even if an actual reload is prevented by MCTRL[LDOK]. If the PWM reload interrupt enable bit INTEN[RIE] is set, the STS[RF] flag generates CPU interrupt requests allowing software to calculate new PWM parameters in real time. When INTEN[RIE] is not set, reloads still occur at the selected reload rate without generating CPU interrupt requests.

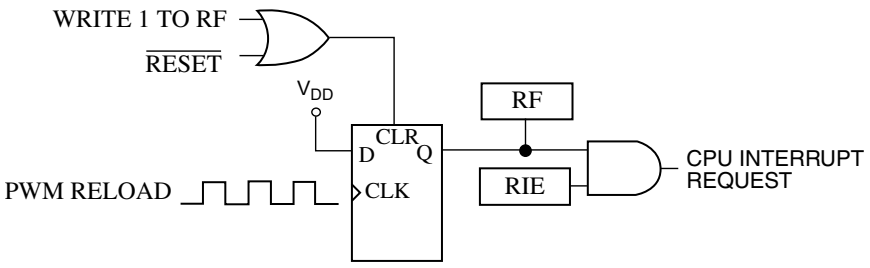


Figure 516. PWMF reload interrupt request

67.3.3.12.4 Reload errors

When one of the INIT, VALx, FRACVALx, or CTRL[PRSC] registers is updated (written by software), the STS[RUF] flag is set to indicate the data in the set of double buffered registers is not coherent. STS[RUF] is cleared by a successful reload which consists of the reload signal while MCTRL[LDOK] is set. If STS[RUF] is set and MCTRL[LDOK] is clear when the reload signal occurs, a reload error takes place and STS[REF] is set. If STS[RUF] is clear when a reload signal asserts, then the data is coherent and no error is flagged.

67.3.3.12.5 Initialization

Initialize all registers and then set MCTRL[LDOK] before setting MCTRL[RUN].

NOTE

If MCTRL[LDOK] is not set, setting MCTRL[RUN] also sets the STS[RF] flag. To prevent a CPU interrupt request, clear INTEN[RIE] before setting MCTRL[RUN].

The PWM generator uses the last values loaded if MCTRL[RUN] is cleared and then set while MCTRL[LDOK] equals zero.

When MCTRL[RUN] is cleared:

- The STS[RF] flag and pending CPU interrupt requests are not cleared
- All fault circuitry remains active
- Software/external output control remains active
- Deadtime insertion continues during software/external output control

67.3.4 Power modes

Be careful when using this module in Stop, Wait, and Debug operating modes.

CAUTION

Some applications (such as three-phase AC motors) require regular software updates for proper operation. Failure to provide regular software updates could result in destroying the hardware setup.

To accommodate this situation, PWM outputs are placed in their inactive states in Stop mode, and they can optionally be placed in inactive states in Wait and Debug modes. PWM outputs are reactivated (assuming they were active beforehand) when these modes are exited.

Table 1054. Modes when PWM operation is restricted

Mode	Description
Stop	PWM outputs are inactive.
Wait	PWM outputs are driven or inactive as a function of CTRL2[WAITEN].
Debug	PWM outputs are driven or inactive as a function of CTRL2[DBGEN].

67.3.5 Clocking

Figure 517 shows the logic used to generate the main counter clock. Each submodule can select between three clock signals: the IPBus clock, EXT_CLK, and AUX_CLK. The EXT_CLK goes to all of the submodules. The AUX_CLK signal is broadcast from submodule0 and can be selected as the clock source by other submodules so that the 8-bit prescaler and MCTRL[RUN] from submodule0 can control all of the submodules.

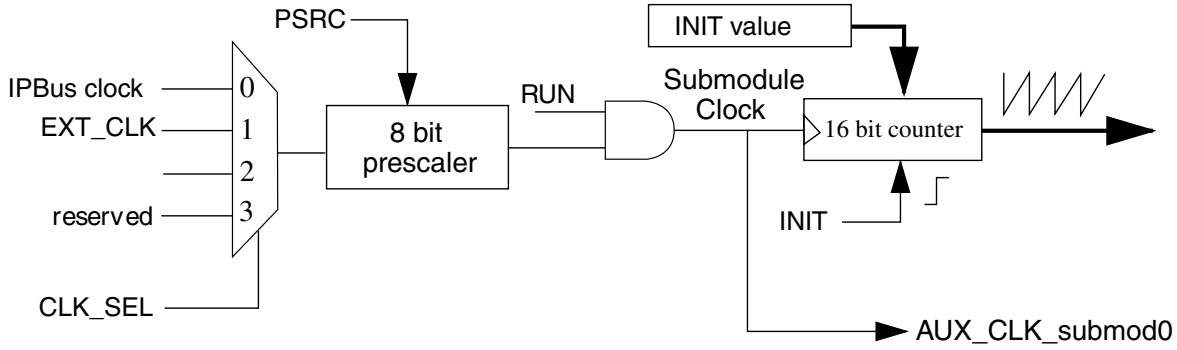


Figure 517. Clocking Block Diagram for PWM Submodule 0

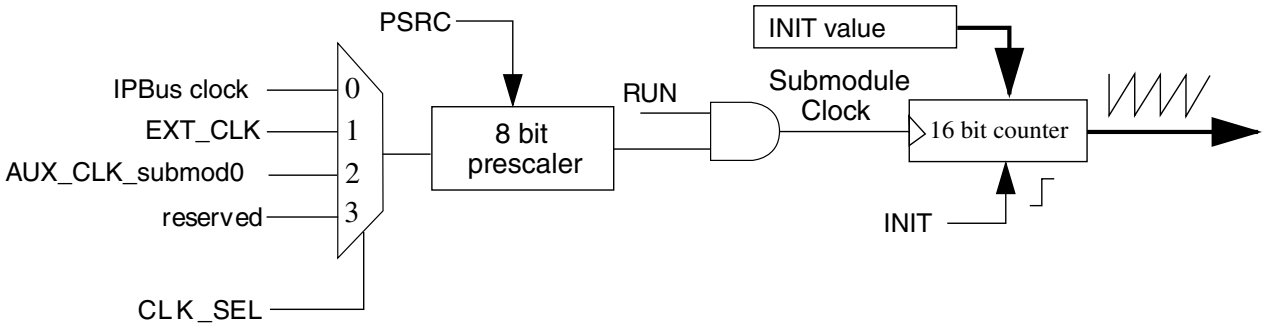


Figure 518. Clocking Block Diagram for all other PWM Submodules

To permit lower PWM frequencies, the prescaler produces the PWM clock frequency by dividing the IPBus clock frequency by 1-128. The prescaler bits, CTRL[PRSC], select the prescaler divisor. This prescaler is buffered and will not be used by the PWM generator until MCTRL[LDOK] is set and a new PWM reload cycle begins or CTRL[LDMOD] is set.

67.3.6 Resets

All PWM registers are reset to their default values upon any system reset.

The reset forces all registers to their reset states and tristates the PWM outputs.

67.3.7 Interrupts

Each of the submodules within the eFlexPWM module can generate an interrupt from several sources. The fault logic can also generate interrupts. The interrupt service routine (ISR) must check the related interrupt enables and interrupt flags to determine the actual cause of the interrupt.

Table 1055. Interrupt Summary

Core Interrupt	Interrupt Flag	Interrupt Enable	Name	Description
PWM_CMP0	SM0STS[CMPF]	SM0INTEN[CMPIE]	Submodule 0 compare interrupt	Compare event has occurred
PWM_CAP0	SM0STS[CFA1], SM0STS[CFA0], SM0STS[CFB1], SM0STS[CFB0], SM0STS[CFX1], SM0STS[CFX0]	SM0INTEN[CFA1IE], SM0INTEN[CFA0IE], SM0INTEN[CFB1IE], SM0INTEN[CFB0IE], SM0INTEN[CFX1IE], SM0INTEN[CFX0IE]	Submodule 0 input capture interrupt	Input capture event has occurred
PWM_RELOAD0	SM0STS[RF]	SM0INTEN[RIE]	Submodule 0 reload interrupt	Reload event has occurred
PWM_CMP1	SM1STS[CMPF]	SM1INTEN[CMPIE]	Submodule 1 compare interrupt	Compare event has occurred
PWM_CAP1	SM1STS[CFA1], SM1STS[CFA0], SM1STS[CFB1], SM1STS[CFB0], SM1STS[CFX1], SM1STS[CFX0]	SM1INTEN[CFA1IE], SM1INTEN[CFA0IE], SM1INTEN[CFB1IE], SM1INTEN[CFB0IE], SM1INTEN[CFX1IE], SM1INTEN[CFX0IE]	Submodule 1 input capture interrupt	Input capture event has occurred
PWM_RELOAD1	SM1STS[RF]	SM1INTEN[RIE]	Submodule 1 reload interrupt	Reload event has occurred
PWM_CMP2	SM2STS[CMPF]	SM2INTEN[CMPIE]	Submodule 2 compare interrupt	Compare event has occurred
PWM_CAP2	SM2STS[CFA1], SM2STS[CFA0], SM2STS[CFB1], SM2STS[CFB0], SM2STS[CFX1], SM2STS[CFX0]	SM2INTEN[CFA1IE], SM2INTEN[CFA0IE], SM2INTEN[CFB1IE], SM2INTEN[CFB0IE], SM2INTEN[CFX1IE], SM2INTEN[CFX0IE	Submodule 2 input capture interrupt	Input capture event has occurred
PWM_RELOAD2	SM2STS[RF]	SM2INTEN[RIE]	Submodule 2 reload interrupt	Reload event has occurred

Table continues on the next page...

Table 1055. Interrupt Summary (continued)

Core Interrupt	Interrupt Flag	Interrupt Enable	Name	Description
PWM_CMP3	SM3STS[CMPF]	SM3INTEN[CMPIE]	Submodule 3 compare interrupt	Compare event has occurred
PWM_CAP3	SM3STS[CFA1], SM3STS[CFA0], SM3STS[CFB1], SM3STS[CFB0], SM3STS[CFX1], SM3STS[CFX0]	SM3INTEN[CFA1IE], SM3INTEN[CFA0IE], SM3INTEN[CFB1IE], SM3INTEN[CFB0IE], SM3INTEN[CFX1IE], SM3INTEN[CFX0IE	Submodule 3 input capture interrupt	Input capture event has occurred
PWM_RELOAD3	SM3STS[RF]	SM3INTEN[RIE]	Submodule 3 reload interrupt	Reload event has occurred
PWM_RERR	SM0STS[REF]	SM0INTEN[REIE]	Submodule 0 reload error interrupt	Reload error has occurred
	SM1STS[REF]	SM1INTEN[REIE]	Submodule 1 reload error interrupt	
	SM2STS[REF]	SM2INTEN[REIE]	Submodule 2 reload error interrupt	
	SM3STS[REF]	SM3INTEN[REIE]	Submodule 3 reload error interrupt	
PWM_FAULT	FSTS[FFLAG]	FCTRL[FIE]	Fault input interrupt	Fault condition has been detected

67.3.8 DMA

Each submodule can request a DMA read access for its capture FIFOs and a DMA write request for its double buffered registers.

Table 1056. DMA summary

DMA request	DMA enable	Name	Description
Submodule 0 read request	SM0DMAEN[CX0DE]	SM0 Capture FIFO X0 read request	SM0CVAL0 contains a value to be read
Submodule 0 read request	SM0DMAEN[CX1DE]	SM0 Capture FIFO X1 read request	SM0CVAL1 contains a value to be read
Submodule 0 read request	SM0DMAEN[CA0DE]	SM0 Capture FIFO A0 read request	SM0CVAL2 contains a value to be read
Submodule 0 read request	SM0DMAEN[CA1DE]	SM0 Capture FIFO A1 read request	SM0CVAL3 contains a value to be read

Table continues on the next page...

Table 1056. DMA summary (continued)

DMA request	DMA enable	Name	Description
Submodule 0 read request	SM0DMAEN[CB0DE]	SM0 Capture FIFO B0 read request	SM0CVAL4 contains a value to be read
Submodule 0 read request	SM0DMAEN[CB1DE]	SM0 Capture FIFO B1 read request	SM0CVAL5 contains a value to be read
Submodule 0 read request	SM0DMAEN[CAPTDE]	SM0 Capture FIFO read request source select	Selects source of submodule0 read DMA request
Submodule 0 write request	SM0DMAEN[VALDE]	SM0VALx write request	SM0VALx and SM0FRACVALx registers need to be updated
Submodule 1 read request	SM1DMAEN[CX0DE]	SM1 Capture FIFO X0 read request	SM1CVAL0 contains a value to be read
Submodule 1 read request	SM1DMAEN[CX1DE]	SM1 Capture FIFO X1 read request	SM1CVAL1 contains a value to be read
Submodule 1 read request	SM1DMAEN[CA0DE]	SM1Capture FIFO A0 read request	SM1CVAL2 contains a value to be read
Submodule 1 read request	SM1DMAEN[CA1DE]	SM1 Capture FIFO A1 read request	SM1CVAL3 contains a value to be read
Submodule 1 read request	SM1DMAEN[CB0DE]	SM1 Capture FIFO B0 read request	SM1CVAL4 contains a value to be read
Submodule 1 read request	SM1DMAEN[CB1DE]	SM1 Capture FIFO B1 read request	SM1CVAL5 contains a value to be read
Submodule 1 read request	SM1DMAEN[CAPTDE]	SM1 Capture FIFO read request source select	Selects source of submodule1 read DMA request
Submodule 1 write request	SM1DMAEN[VALDE]	SM1VALx write request	SM1VALx and SM1FRACVALx registers need to be updated
Submodule 2 read request	SM2DMAEN[CX0DE]	SM2 Capture FIFO X0 read request	SM2CVAL0 contains a value to be read
Submodule 2 read request	SM2DMAEN[CX1DE]	SM2 Capture FIFO X1 read request	SM2CVAL1 contains a value to be read
Submodule 2 read request	SM2DMAEN[CA0DE]	SM2 Capture FIFO A0 read request	SM2CVAL2 contains a value to be read
Submodule 2 read request	SM2DMAEN[CA1DE]	SM2 Capture FIFO A1 read request	SM2CVAL3 contains a value to be read
Submodule 2 read request	SM2DMAEN[CB0DE]	SM2 Capture FIFO B0 read request	SM2CVAL4 contains a value to be read

Table continues on the next page...

Table 1056. DMA summary (continued)

DMA request	DMA enable	Name	Description
Submodule 2 read request	SM2DMAEN[CB1DE]	SM2 Capture FIFO B1 read request	SM2CVAL5 contains a value to be read
Submodule 2 read request	SM2DMAEN[CAPTDE]	SM2 Capture FIFO read request source select	Selects source of submodule2 read DMA request
Submodule 2 write request	SM2DMAEN[VALDE]	SM2VALx write request	SM2VALx and SM2FRACVALx registers need to be updated
Submodule 3 read request	SM3DMAEN[CX0DE]	SM3 Capture FIFO X0 read request	SM3CVAL0 contains a value to be read
Submodule 3 read request	SM3DMAEN[CX1DE]	SM3 Capture FIFO X1 read request	SM3CVAL1 contains a value to be read
Submodule 3 read request	SM3DMAEN[CA0DE]	SM3 Capture FIFO A0 read request	SM3CVAL2 contains a value to be read
Submodule 3 read request	SM3DMAEN[CA1DE]	SM3 Capture FIFO A1 read request	SM3CVAL3 contains a value to be read
Submodule 3 read request	SM3DMAEN[CB0DE]	SM3 Capture FIFO B0 read request	SM3CVAL4 contains a value to be read
Submodule 3 read request	SM3DMAEN[CB1DE]	SM3 Capture FIFO B1 read request	SM3CVAL5 contains a value to be read
Submodule 3 read request	SM3DMAEN[CAPTDE]	SM3 Capture FIFO read request source select	Selects source of submodule3 read DMA request
Submodule 3 write request	SM3DMAEN[VALDE]	SM3VALx write request	SM3VALx and SM3FRACVALx registers need to be updated

67.4 External signals

The PWM has pins named PWM_An, PWM_Bn, PWM_Xn, FAULTn, EXT_SYNC, EXT_FORCE, and PWMn_EXTA. The PWM also has an on-chip input called EXT_CLK and output signals called PWMn_OUT_TRIGx and PWMn_MUX_TRIGx.

67.4.1 PWM_An and PWM_Bn - External PWM output pair

These pins are the output pins of the PWM channels. These pins can be independent PWM signals or a complementary pair. When not needed as an output, they can be used as inputs to the input capture circuitry.

67.4.2 PWM_Xn - Auxiliary PWM output signal

These pins are the auxiliary output pins of the PWM channels. They can be independent PWM signals. When not needed as an output, they can be used as inputs to the input capture circuitry or used to detect the polarity of the current flowing through the complementary circuit at deadtime correction.

67.4.3 FAULTn - Fault Inputs

These are input pins for disabling selected PWM outputs.

67.4.4 EXT_SYNC - External synchronization signal

These input signals allow a source external to the PWM to initialize the PWM counter. Therefore, the PWM can be synchronized to external circuitry.

67.4.5 EXT_FORCE - External output force signal

This input signal allows a source external to the PWM to force an update of the PWM outputs. Therefore, the PWM can be synchronized to external circuitry.

67.4.6 PWMn_EXTA - Alternate PWM control signals

These pins allow an alternate source to control the PWM_An and PWM_Bn outputs. Typically, the PWMn_EXTA input (depending on the state of MCTRL[IPOL]) is used for the generation of a complementary pair. Typical control signals include ADC conversion high/low limits, TMR outputs, GPIO inputs, and comparator outputs.

For pin input details, see chip-specific eFlexPWM information.

67.4.7 PWMn_OUT_TRIG0 and PWMn_OUT_TRIG1 - Output triggers

These outputs allow the PWM submodules to control timing of ADC conversions. See the description of the SMnTCTRL[OUT_TRIG_EN] for information about how to enable these outputs and how the compare registers match up to the output triggers.

67.4.8 PWM[n]_MUX_TRIG0 and PWM[n]_MUX_TRIG1 - Output triggers

These outputs can be either PWMn_OUT_TRIG0/PWMn_OUT_TRIG1 signals or PWM_A/PWM_B signals from output logic. See the description of the PWAOT0 and PWBOT1 bits in the Output Trigger Control Register for information about how to enable these outputs.

67.4.9 EXT_CLK - External clock signal

This signal allows a source external to the PWM (typically a timer or an off-chip source) to control the PWM clocking. Therefore, the PWM can be synchronized to the timer, or multiple chips can be synchronized to each other.

67.5 PWM register descriptions

The address of a register is the sum of a base address and an address offset. The base address is defined at the core level, and the address offset is defined at the module level. The PWM module has a set of registers for each PWM submodule, for the configuration logic, and for each fault channel.

NOTE

While the registers are 16-bit wide, they can be accessed in pairs as 32-bit registers, if the pairs are word-aligned.

Submodule registers are repeated for each PWM submodule. To designate which submodule that they are in, register names are prefixed with SMx(x=0,1...). The base address of submodule 0 is the same as the base address for the PWM module as a whole. The base address of submodule 1 is offset 0x60 from the base address for the PWM module as a whole. This 0x60 offset is based on the number of registers in a submodule. The base address of submodule 2 is equal to the base address of submodule 1 plus this same 0x60 offset. The pattern repeats for the base address of submodule 3.

The base address of the configuration registers is equal to the base address of the PWM module as a whole plus an offset of 0x180.

67.5.1 PWM memory map

PWM1 base address: 4265_0000h

PWM2 base address: 4266_0000h

PWM3 base address: 4267_0000h

PWM4 base address: 4268_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Counter Register (SM0CNT)	16	R	0000h
2h	Initial Count Register (SM0INIT)	16	RW	0000h
4h	Control 2 Register (SM0CTRL2)	16	RW	0000h
6h	Control Register (SM0CTRL)	16	RW	0400h
Ah	Value Register 0 (SM0VAL0)	16	RW	0000h
Ch	Fractional Value Register 1 (SM0FRACVAL1)	16	RW	0000h
Eh	Value Register 1 (SM0VAL1)	16	RW	0000h
10h	Fractional Value Register 2 (SM0FRACVAL2)	16	RW	0000h
12h	Value Register 2 (SM0VAL2)	16	RW	0000h
14h	Fractional Value Register 3 (SM0FRACVAL3)	16	RW	0000h
16h	Value Register 3 (SM0VAL3)	16	RW	0000h
18h	Fractional Value Register 4 (SM0FRACVAL4)	16	RW	0000h
1Ah	Value Register 4 (SM0VAL4)	16	RW	0000h
1Ch	Fractional Value Register 5 (SM0FRACVAL5)	16	RW	0000h
1Eh	Value Register 5 (SM0VAL5)	16	RW	0000h
20h	Fractional Control Register (SM0FRCTRL)	16	RW	0000h
22h	Output Control Register (SM0OCTRL)	16	RW	0000h
24h	Status Register (SM0STS)	16	RW	0000h
26h	Interrupt Enable Register (SM0INTEN)	16	RW	0000h
28h	DMA Enable Register (SM0DMAEN)	16	RW	0000h
2Ah	Output Trigger Control Register (SM0TCTRL)	16	RW	0000h
2Ch	Fault Disable Mapping Register 0 (SM0DISMAP0)	16	RW	FFFFh
30h	Deadtime Count Register 0 (SM0DTCNT0)	16	RW	07FFh
32h	Deadtime Count Register 1 (SM0DTCNT1)	16	RW	07FFh
34h	Capture Control A Register (SM0CAPTCTRLA)	16	RW	0000h
36h	Capture Compare A Register (SM0CAPTCOMPA)	16	RW	0000h
38h	Capture Control B Register (SM0CAPTCTRLB)	16	RW	0000h
3Ah	Capture Compare B Register (SM0CAPTCOMPB)	16	RW	0000h
3Ch	Capture Control X Register (SM0CAPTCTRLX)	16	RW	0000h
3Eh	Capture Compare X Register (SM0CAPTCOMPX)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
40h	Capture Value 0 Register (SM0CVAL0)	16	R	0000h
42h	Capture Value 0 Cycle Register (SM0CVAL0CYC)	16	R	0000h
44h	Capture Value 1 Register (SM0CVAL1)	16	R	0000h
46h	Capture Value 1 Cycle Register (SM0CVAL1CYC)	16	R	0000h
48h	Capture Value 2 Register (SM0CVAL2)	16	R	0000h
4Ah	Capture Value 2 Cycle Register (SM0CVAL2CYC)	16	R	0000h
4Ch	Capture Value 3 Register (SM0CVAL3)	16	R	0000h
4Eh	Capture Value 3 Cycle Register (SM0CVAL3CYC)	16	R	0000h
50h	Capture Value 4 Register (SM0CVAL4)	16	R	0000h
52h	Capture Value 4 Cycle Register (SM0CVAL4CYC)	16	R	0000h
54h	Capture Value 5 Register (SM0CVAL5)	16	R	0000h
56h	Capture Value 5 Cycle Register (SM0CVAL5CYC)	16	R	0000h
5Ah	Capture PWM_A Input Filter Register (SM0CAPTFILTA)	16	RW	0000h
5Ch	Capture PWM_B Input Filter Register (SM0CAPTFILTB)	16	RW	0000h
5Eh	Capture PWM_X Input Filter Register (SM0CAPTFILTX)	16	RW	0000h
60h	Counter Register (SM1CNT)	16	R	0000h
62h	Initial Count Register (SM1INIT)	16	RW	0000h
64h	Control 2 Register (SM1CTRL2)	16	RW	0000h
66h	Control Register (SM1CTRL)	16	RW	0400h
6Ah	Value Register 0 (SM1VAL0)	16	RW	0000h
6Ch	Fractional Value Register 1 (SM1FRACVAL1)	16	RW	0000h
6Eh	Value Register 1 (SM1VAL1)	16	RW	0000h
70h	Fractional Value Register 2 (SM1FRACVAL2)	16	RW	0000h
72h	Value Register 2 (SM1VAL2)	16	RW	0000h
74h	Fractional Value Register 3 (SM1FRACVAL3)	16	RW	0000h
76h	Value Register 3 (SM1VAL3)	16	RW	0000h
78h	Fractional Value Register 4 (SM1FRACVAL4)	16	RW	0000h
7Ah	Value Register 4 (SM1VAL4)	16	RW	0000h
7Ch	Fractional Value Register 5 (SM1FRACVAL5)	16	RW	0000h
7Eh	Value Register 5 (SM1VAL5)	16	RW	0000h
80h	Fractional Control Register (SM1FRCTRL)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
82h	Output Control Register (SM1OCTRL)	16	RW	0000h
84h	Status Register (SM1STS)	16	RW	0000h
86h	Interrupt Enable Register (SM1INTEN)	16	RW	0000h
88h	DMA Enable Register (SM1DMAEN)	16	RW	0000h
8Ah	Output Trigger Control Register (SM1TCTRL)	16	RW	0000h
8Ch	Fault Disable Mapping Register 0 (SM1DISMAP0)	16	RW	FFFFh
90h	Deadtime Count Register 0 (SM1DTCNT0)	16	RW	07FFh
92h	Deadtime Count Register 1 (SM1DTCNT1)	16	RW	07FFh
94h	Capture Control A Register (SM1CAPTCTRLA)	16	RW	0000h
96h	Capture Compare A Register (SM1CAPTCOMPA)	16	RW	0000h
98h	Capture Control B Register (SM1CAPTCTRLB)	16	RW	0000h
9Ah	Capture Compare B Register (SM1CAPTCOMPB)	16	RW	0000h
9Ch	Capture Control X Register (SM1CAPTCTRLX)	16	RW	0000h
9Eh	Capture Compare X Register (SM1CAPTCOMPX)	16	RW	0000h
A0h	Capture Value 0 Register (SM1CVAL0)	16	R	0000h
A2h	Capture Value 0 Cycle Register (SM1CVAL0CYC)	16	R	0000h
A4h	Capture Value 1 Register (SM1CVAL1)	16	R	0000h
A6h	Capture Value 1 Cycle Register (SM1CVAL1CYC)	16	R	0000h
A8h	Capture Value 2 Register (SM1CVAL2)	16	R	0000h
AAh	Capture Value 2 Cycle Register (SM1CVAL2CYC)	16	R	0000h
ACh	Capture Value 3 Register (SM1CVAL3)	16	R	0000h
AEh	Capture Value 3 Cycle Register (SM1CVAL3CYC)	16	R	0000h
B0h	Capture Value 4 Register (SM1CVAL4)	16	R	0000h
B2h	Capture Value 4 Cycle Register (SM1CVAL4CYC)	16	R	0000h
B4h	Capture Value 5 Register (SM1CVAL5)	16	R	0000h
B6h	Capture Value 5 Cycle Register (SM1CVAL5CYC)	16	R	0000h
B8h	Phase Delay Register (SM1PHASEDLY)	16	RW	0000h
BAh	Capture PWM_A Input Filter Register (SM1CAPTFILTA)	16	RW	0000h
BCh	Capture PWM_B Input Filter Register (SM1CAPTFILTB)	16	RW	0000h
BEh	Capture PWM_X Input Filter Register (SM1CAPTFILTX)	16	RW	0000h
C0h	Counter Register (SM2CNT)	16	R	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C2h	Initial Count Register (SM2INIT)	16	RW	0000h
C4h	Control 2 Register (SM2CTRL2)	16	RW	0000h
C6h	Control Register (SM2CTRL)	16	RW	0400h
CAh	Value Register 0 (SM2VAL0)	16	RW	0000h
CCh	Fractional Value Register 1 (SM2FRACVAL1)	16	RW	0000h
CEh	Value Register 1 (SM2VAL1)	16	RW	0000h
D0h	Fractional Value Register 2 (SM2FRACVAL2)	16	RW	0000h
D2h	Value Register 2 (SM2VAL2)	16	RW	0000h
D4h	Fractional Value Register 3 (SM2FRACVAL3)	16	RW	0000h
D6h	Value Register 3 (SM2VAL3)	16	RW	0000h
D8h	Fractional Value Register 4 (SM2FRACVAL4)	16	RW	0000h
DAh	Value Register 4 (SM2VAL4)	16	RW	0000h
DCh	Fractional Value Register 5 (SM2FRACVAL5)	16	RW	0000h
DEh	Value Register 5 (SM2VAL5)	16	RW	0000h
E0h	Fractional Control Register (SM2FRCTRL)	16	RW	0000h
E2h	Output Control Register (SM2OCTRL)	16	RW	0000h
E4h	Status Register (SM2STS)	16	RW	0000h
E6h	Interrupt Enable Register (SM2INTEN)	16	RW	0000h
E8h	DMA Enable Register (SM2DMAEN)	16	RW	0000h
EAh	Output Trigger Control Register (SM2TCTRL)	16	RW	0000h
ECh	Fault Disable Mapping Register 0 (SM2DISMAP0)	16	RW	FFFFh
F0h	Deadtime Count Register 0 (SM2DTCNT0)	16	RW	07FFh
F2h	Deadtime Count Register 1 (SM2DTCNT1)	16	RW	07FFh
F4h	Capture Control A Register (SM2CAPTCTRLA)	16	RW	0000h
F6h	Capture Compare A Register (SM2CAPTCOMPA)	16	RW	0000h
F8h	Capture Control B Register (SM2CAPTCTRLB)	16	RW	0000h
FAh	Capture Compare B Register (SM2CAPTCOMPB)	16	RW	0000h
FCh	Capture Control X Register (SM2CAPTCTRLX)	16	RW	0000h
FEh	Capture Compare X Register (SM2CAPTCOMPX)	16	RW	0000h
100h	Capture Value 0 Register (SM2CVAL0)	16	R	0000h
102h	Capture Value 0 Cycle Register (SM2CVAL0CYC)	16	R	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
104h	Capture Value 1 Register (SM2CVAL1)	16	R	0000h
106h	Capture Value 1 Cycle Register (SM2CVAL1CYC)	16	R	0000h
108h	Capture Value 2 Register (SM2CVAL2)	16	R	0000h
10Ah	Capture Value 2 Cycle Register (SM2CVAL2CYC)	16	R	0000h
10Ch	Capture Value 3 Register (SM2CVAL3)	16	R	0000h
10Eh	Capture Value 3 Cycle Register (SM2CVAL3CYC)	16	R	0000h
110h	Capture Value 4 Register (SM2CVAL4)	16	R	0000h
112h	Capture Value 4 Cycle Register (SM2CVAL4CYC)	16	R	0000h
114h	Capture Value 5 Register (SM2CVAL5)	16	R	0000h
116h	Capture Value 5 Cycle Register (SM2CVAL5CYC)	16	R	0000h
118h	Phase Delay Register (SM2PHASEDLY)	16	RW	0000h
11Ah	Capture PWM_A Input Filter Register (SM2CAPTFILTA)	16	RW	0000h
11Ch	Capture PWM_B Input Filter Register (SM2CAPTFILTB)	16	RW	0000h
11Eh	Capture PWM_X Input Filter Register (SM2CAPTFILTX)	16	RW	0000h
120h	Counter Register (SM3CNT)	16	R	0000h
122h	Initial Count Register (SM3INIT)	16	RW	0000h
124h	Control 2 Register (SM3CTRL2)	16	RW	0000h
126h	Control Register (SM3CTRL)	16	RW	0400h
12Ah	Value Register 0 (SM3VAL0)	16	RW	0000h
12Ch	Fractional Value Register 1 (SM3FRACVAL1)	16	RW	0000h
12Eh	Value Register 1 (SM3VAL1)	16	RW	0000h
130h	Fractional Value Register 2 (SM3FRACVAL2)	16	RW	0000h
132h	Value Register 2 (SM3VAL2)	16	RW	0000h
134h	Fractional Value Register 3 (SM3FRACVAL3)	16	RW	0000h
136h	Value Register 3 (SM3VAL3)	16	RW	0000h
138h	Fractional Value Register 4 (SM3FRACVAL4)	16	RW	0000h
13Ah	Value Register 4 (SM3VAL4)	16	RW	0000h
13Ch	Fractional Value Register 5 (SM3FRACVAL5)	16	RW	0000h
13Eh	Value Register 5 (SM3VAL5)	16	RW	0000h
140h	Fractional Control Register (SM3FRCTRL)	16	RW	0000h
142h	Output Control Register (SM3OCTRL)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
144h	Status Register (SM3STS)	16	RW	0000h
146h	Interrupt Enable Register (SM3INTEN)	16	RW	0000h
148h	DMA Enable Register (SM3DMAEN)	16	RW	0000h
14Ah	Output Trigger Control Register (SM3TCTRL)	16	RW	0000h
14Ch	Fault Disable Mapping Register 0 (SM3DISMAP0)	16	RW	FFFFh
150h	Deadtime Count Register 0 (SM3DTCNT0)	16	RW	07FFh
152h	Deadtime Count Register 1 (SM3DTCNT1)	16	RW	07FFh
154h	Capture Control A Register (SM3CAPTCTRLA)	16	RW	0000h
156h	Capture Compare A Register (SM3CAPTCOMPA)	16	RW	0000h
158h	Capture Control B Register (SM3CAPTCTRLB)	16	RW	0000h
15Ah	Capture Compare B Register (SM3CAPTCOMPB)	16	RW	0000h
15Ch	Capture Control X Register (SM3CAPTCTRLX)	16	RW	0000h
15Eh	Capture Compare X Register (SM3CAPTCOMPX)	16	RW	0000h
160h	Capture Value 0 Register (SM3CVAL0)	16	R	0000h
162h	Capture Value 0 Cycle Register (SM3CVAL0CYC)	16	R	0000h
164h	Capture Value 1 Register (SM3CVAL1)	16	R	0000h
166h	Capture Value 1 Cycle Register (SM3CVAL1CYC)	16	R	0000h
168h	Capture Value 2 Register (SM3CVAL2)	16	R	0000h
16Ah	Capture Value 2 Cycle Register (SM3CVAL2CYC)	16	R	0000h
16Ch	Capture Value 3 Register (SM3CVAL3)	16	R	0000h
16Eh	Capture Value 3 Cycle Register (SM3CVAL3CYC)	16	R	0000h
170h	Capture Value 4 Register (SM3CVAL4)	16	R	0000h
172h	Capture Value 4 Cycle Register (SM3CVAL4CYC)	16	R	0000h
174h	Capture Value 5 Register (SM3CVAL5)	16	R	0000h
176h	Capture Value 5 Cycle Register (SM3CVAL5CYC)	16	R	0000h
178h	Phase Delay Register (SM3PHASEDLY)	16	RW	0000h
17Ah	Capture PWM_A Input Filter Register (SM3CAPTFILTA)	16	RW	0000h
17Ch	Capture PWM_B Input Filter Register (SM3CAPTFILTB)	16	RW	0000h
17Eh	Capture PWM_X Input Filter Register (SM3CAPTFILTX)	16	RW	0000h
180h	Output Enable Register (OUTEN)	16	RW	0000h
182h	Mask Register (MASK)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
184h	Software Controlled Output Register (SWCOUT)	16	RW	0000h
186h	PWM Source Select Register (DTSRCSEL)	16	RW	0000h
188h	Master Control Register (MCTRL)	16	RW	0000h
18Ah	Master Control 2 Register (MCTRL2)	16	RW	0000h
18Ch	Fault Control Register (FCTRL0)	16	RW	0000h
18Eh	Fault Status Register (FSTS0)	16	RW	See section
190h	Fault Filter Register (FFILT0)	16	RW	0000h
192h	Fault Test Register (FTST0)	16	RW	0000h
194h	Fault Control 2 Register (FCTRL20)	16	RW	0000h

67.5.2 Counter Register (SM0CNT - SM3CNT)

Offset

Register	Offset
SM0CNT	0h
SM1CNT	60h
SM2CNT	C0h
SM3CNT	120h

Function

This field displays the state of the signed 16-bit submodule counter. This register is not byte accessible.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 CNT	Counter Register Bits

67.5.3 Initial Count Register (SM0INIT - SM3INIT)

Offset

Register	Offset
SM0INIT	2h
SM1INIT	62h
SM2INIT	C2h
SM3INIT	122h

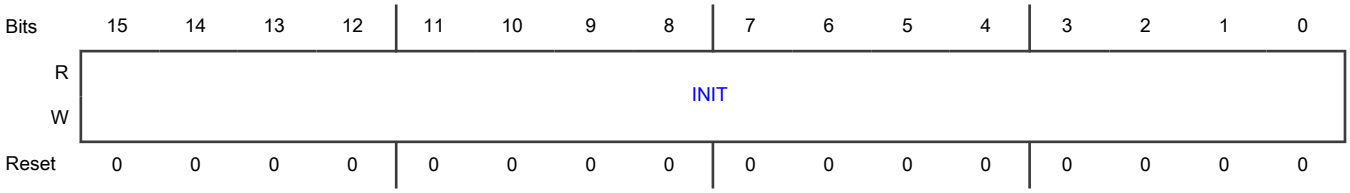
Function

The 16-bit signed value in this buffered register defines the initial count value for the PWM in PWM clock periods. This is the value loaded into the submodule counter when local sync, master sync, or master reload is asserted (based on the value of CTRL2[INIT_SEL]) or when CTRL2[FORCE] is asserted and force init is enabled. For PWM operation, the buffered contents of this register are loaded into the counter at the start of every PWM cycle. This register is not byte accessible.

NOTE

The INIT register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. This register cannot be written when MCTRL[LDOK] is set. Reading INIT reads the value in a buffer and not necessarily the value the PWM generator is currently using.

Diagram



Fields

Field	Function
15-0 INIT	Initial Count Register Bits

67.5.4 Control 2 Register (SM0CTRL2 - SM3CTRL2)

Offset

Register	Offset
SM0CTRL2	4h
SM1CTRL2	64h
SM2CTRL2	C4h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
SM3CTRL2	124h

Function

Contains control fields for clock select and forcing output and initialization.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DBGEN	WAITEN	INDEP	PWM23_I...	PWM45_I...	PWMX_IN...	INIT_SEL		FRCEN	0	FORCE_SEL			RELOAD_...	CLK_SEL	
W										FORCE						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 DBGEN	Debug Enable When set to one, the PWM continues to run while the chip is in Debug mode. If the device enters Debug mode and this bit is zero, then the PWM outputs are disabled until Debug mode is exited. At that point, the PWM pins resume operation as programmed in the PWM registers. For certain types of motors (such as 3-phase AC), it is imperative that this bit be left in its default state (in which the PWM is disabled in Debug mode). Failure to do so could result in damage to the motor or inverter. For other types of motors (example: DC motors), this bit might safely be set to one, enabling the PWM in Debug mode. The key point is PWM parameter updates will not occur in Debug mode. Any motors requiring such updates should be disabled during Debug mode. If in doubt, leave this bit set to zero.
14 WAITEN	Wait Enable When set to one, the PWM continues to run while the chip is in Wait mode. In this mode, the peripheral clock continues to run but the CPU clock does not. If the device enters Wait mode and this bit is zero, then the PWM outputs are disabled until Wait mode is exited. At that point the PWM pins resume operation as programmed in the PWM registers. For certain types of motors (such as 3-phase AC), it is imperative that this bit be left in its default state (in which the PWM is disabled in Wait mode). Failure to do so could result in damage to the motor or inverter. For other types of motors (example: DC motors), this bit might safely be set to one, enabling the PWM in Wait mode. The key point is PWM parameter updates will not occur in this mode. Any motors requiring such updates should be disabled during Wait mode. If in doubt, leave this bit set to zero.
13 INDEP	Independent or Complementary Pair Operation This bit determines if the PWM_A and PWM_B channels are independent PWMs or a complementary PWM pair. 0b - PWM_A and PWM_B form a complementary PWM pair. 1b - PWM_A and PWM_B outputs are independent PWMs.

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 PWM23_INIT	PWM23 Initial Value This bit determines the initial value for PWM23 and the value to which it is forced when FORCE_INIT is asserted.
11 PWM45_INIT	PWM45 Initial Value This bit determines the initial value for PWM45 and the value to which it is forced when FORCE_INIT is asserted.
10 PWMX_INIT	PWM_X Initial Value This bit determines the initial value for PWM_X and the value to which it is forced when FORCE_INIT is asserted.
9-8 INIT_SEL	Initialization Control Select These bits control the source of the INIT signal which goes to the counter. 00b - Local sync (PWM_X) causes initialization. 01b - Master reload from submodule 0 causes initialization. This setting should not be used in submodule 0 as it forces the INIT signal to logic 0. The submodule counter will only re-initialize when a master reload occurs. 10b - Master sync from submodule 0 causes initialization. This setting should not be used in submodule 0 as it forces the INIT signal to logic 0. 11b - EXT_SYNC causes initialization.
7 FRCEN	Force Enable This bit allows the CTRL2[FORCE] signal to initialize the counter without regard to the signal selected by CTRL2[INIT_SEL]. This is a software controlled initialization. A forced initialization will also assert the register reload if MCTRL[LDOK] is set. 0b - Initialization from a FORCE_OUT is disabled. 1b - Initialization from a FORCE_OUT is enabled.
6 FORCE	Force Initialization If CTRL2[FORCE_SEL] is set to 000, writing a 1 to this bit results in a FORCE_OUT event. This causes the following actions to be taken: • The PWM_A and PWM_B output pins assume values based on DTSRCSEL[SMxSEL23] and DTSRCSEL[SMxSEL45]. • If CTRL2[FRCEN] is set, the counter value is initialized with the INIT register value only when the submodule MCTRL[RUN] = 1 or CTRL2[CLK_SEL] = 2.
5-3 FORCE_SEL	Force Select This bit determines the source of the FORCE OUTPUT signal for this submodule. 000b - The local force signal, CTRL2[FORCE], from this submodule is used to force updates.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - The master force signal from submodule 0 is used to force updates. This setting should not be used in submodule 0 as it holds the FORCE OUTPUT signal to logic 0. 010b - The local reload signal from this submodule is used to force updates without regard to the state of LDOK. 011b - The master reload signal from submodule0 is used to force updates if LDOK is set. This setting should not be used in submodule0 as it holds the FORCE OUTPUT signal to logic 0. 100b - The local sync signal from this submodule is used to force updates. 101b - The master sync signal from submodule0 is used to force updates. This setting should not be used in submodule0 as it holds the FORCE OUTPUT signal to logic 0. 110b - The external force signal, EXT_FORCE, from outside the PWM module causes updates. 111b - The external sync signal, EXT_SYNC, from outside the PWM module causes updates.
2 RELOAD_SEL	Reload Source Select This bit determines the source of the RELOAD signal for this submodule. When this bit is set, MCTRL[LDOK[0]] for submodule 0 should be used since the local MCTRL[LDOK] will be ignored. 0b - The local RELOAD signal is used to reload registers. 1b - The master RELOAD signal (from submodule 0) is used to reload registers. This setting should not be used in submodule 0 as it forces the RELOAD signal to logic 0.
1-0 CLK_SEL	Clock Source Select These bits determine the source of the clock signal for this submodule. 00b - The IPBus clock is used as the clock for the local prescaler and counter. 01b - EXT_CLK is used as the clock for the local prescaler and counter. 10b - Submodule 0’s clock (AUX_CLK) is used as the source clock for the local prescaler and counter. This setting should not be used in submodule 0 as it forces the clock to logic 0. 11b - Reserved

67.5.5 Control Register (SM0CTRL - SM3CTRL)

Offset

Register	Offset
SM0CTRL	6h
SM1CTRL	66h
SM2CTRL	C6h
SM3CTRL	126h

Function

Includes control settings for timing, loading, and buffering.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LDFQ				HALF	FULL	DT		COMP MODE	PRSC			SPLIT	LDMOD	DBLX	DBLEN
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-12 LDFQ	Load Frequency These buffered bits select the PWM load frequency. Reset clears LDFQ, selecting loading every PWM opportunity. A PWM opportunity is determined by HALF and FULL. **NOTE** LDFQ takes effect when the current load cycle is complete, regardless of the state of MCTRL[LDOK]. Reading LDFQ reads the buffered values and not necessarily the values currently in effect. 0000b - Every PWM opportunity 0001b - Every 2 PWM opportunities 0010b - Every 3 PWM opportunities 0011b - Every 4 PWM opportunities 0100b - Every 5 PWM opportunities 0101b - Every 6 PWM opportunities 0110b - Every 7 PWM opportunities 0111b - Every 8 PWM opportunities 1000b - Every 9 PWM opportunities 1001b - Every 10 PWM opportunities 1010b - Every 11 PWM opportunities 1011b - Every 12 PWM opportunities 1100b - Every 13 PWM opportunities 1101b - Every 14 PWM opportunities 1110b - Every 15 PWM opportunities 1111b - Every 16 PWM opportunities
11 HALF	Half Cycle Reload This bit enables half-cycle reloads. A half cycle is defined by when the submodule counter matches the VAL0 register and does not have to be halfway through the PWM cycle. 0b - Half-cycle reloads disabled. 1b - Half-cycle reloads enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 FULL	Full Cycle Reload This bit enables full-cycle reloads. A full cycle is defined by when the submodule counter matches the VAL1 register. Either CTRL[HALF] or CTRL[FULL] must be set to move the buffered data into the registers used by the PWM generators or CTRL[LDMOD] must be set. If both CTRL[HALF] and CTRL[FULL] are set, then reloads can occur twice per cycle. 0b - Full-cycle reloads disabled. 1b - Full-cycle reloads enabled.
9-8 DT	Deadtime These bits reflect the sampled values of the PWM_X input at the end of each deadtime. Sampling occurs at the end of deadtime 0 for DT[0] and the end of deadtime 1 for DT[1]. Reset clears these bits.
7 COMPMODE	Compare Mode This bit controls how comparisons are made between the VAL* registers and the PWM submodule counter. This bit can be written one time after which it requires a reset to release the bit for writing again. 0b - The VAL* registers and the PWM counter are compared using an "equal to" method. This means that PWM edges are only produced when the counter is equal to one of the VAL* register values. This implies that a PWM_A output that is high at the end of a period maintains this state until a match with VAL3 clears the output in the following period. 1b - The VAL* registers and the PWM counter are compared using an "equal to or greater than" method. This means that PWM edges are produced when the counter is equal to or greater than one of the VAL* register values. This implies that a PWM_A output that is high at the end of a period could go low at the start of the next period if the starting counter value is greater than (but not necessarily equal to) the new VAL3 value.
6-4 PRSC	Prescaler These buffered bits select the divide ratio of the PWM clock frequency selected by CTRL2[CLK_SEL]. **NOTE** Reading CTRL[PRSC] reads the buffered values and not necessarily the values currently in effect. CTRL[PRSC] takes effect at the beginning of the next PWM cycle and only when the load okay bit MCTRL[LDOK] is set, or CTRL[LDMOD] is set. This field cannot be written when MCTRL[LDOK] is set. 000b - Prescaler 1. PWM clock frequency = f_{clk} 001b - Prescaler 2. PWM clock frequency = f_{clk} /2 010b - Prescaler 4. PWM clock frequency = f_{clk} /4 011b - Prescaler 8. PWM clock frequency = f_{clk} /8 100b - Prescaler 16. PWM clock frequency = f_{clk} /16 101b - Prescaler 32. PWM clock frequency = f_{clk} /32 110b - Prescaler 64. PWM clock frequency = f_{clk} /64 111b - Prescaler 128. PWM clock frequency = f_{clk} /128

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 SPLIT	Split the DBLPWM signal to PWM_A and PWM_B This bit is used in independent mode when DBLEN is set. This bit allows the two PWM pulses generated by DBLEN to be split with one pulse on PWM_A and one on PWM_B. The two pulses within the same PWM period are created by an XOR function of the PWM_A and PWM_B sources. The splitting function causes PWM_A to output the pulse that occurs when the PWM_A source is 1 and the PWM_B source is 0. The PWM_B output occurs when the PWM_B source is 1 and the PWM_A source is 0. (See Double switching PWMs .) 0b - DBLPWM is not split. PWM_A and PWM_B each have double pulses. 1b - DBLPWM is split to PWM_A and PWM_B.
2 LDMOD	Load Mode Select This bit selects the timing of loading the buffered registers for this submodule. 0b - Buffered registers of this submodule are loaded and take effect at the next PWM reload if MCTRL[LDOK] is set. 1b - Buffered registers of this submodule are loaded and take effect immediately upon MCTRL[LDOK] being set. In this case, it is not necessary to set CTRL[FULL] or CTRL[HALF].
1 DBLX	PWM_X Double Switching Enable This bit enables the double switching behavior on PWM_X. When this bit is set, the PWM_X output shall be the exclusive OR combination of PWM_A and PWM_B prior to polarity and masking considerations. 0b - PWM_X double pulse disabled. 1b - PWM_X double pulse enabled.
0 DBLEN	Double Switching Enable This bit enables the double switching PWM behavior (See Double switching PWMs). 0b - Double switching disabled. 1b - Double switching enabled.

67.5.6 Value Register 0 (SM0VAL0 - SM3VAL0)

Offset

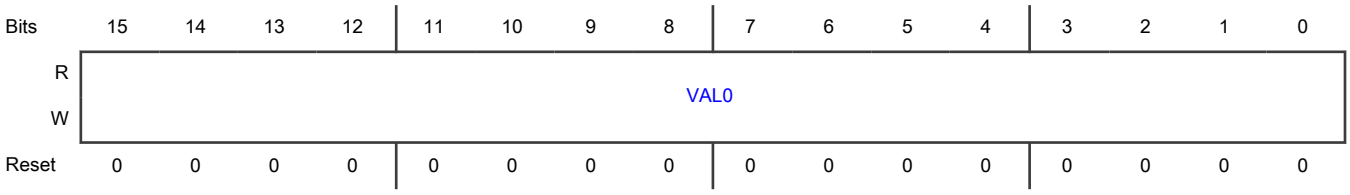
Register	Offset
SM0VAL0	Ah
SM1VAL0	6Ah
SM2VAL0	CAh
SM3VAL0	12Ah

Function

NOTE

The VAL0 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL0 cannot be written when MCTRL[LDOK] is set. Reading VAL0 reads the value in a buffer. It is not necessarily the value that the PWM generator is currently using.

Diagram



Fields

Field	Function
15-0 VAL0	Value 0 The 16-bit signed value in this buffered register defines the mid-cycle reload point for the PWM in PWM clock periods. This value also defines when the PWM_X signal is set and the local sync signal is reset. This register is not byte accessible. **NOTE** The actual behavior takes effect when counter equal VAL0+1.

67.5.7 Fractional Value Register 1 (SM0FRACVAL1 - SM3FRACVAL1)

Offset

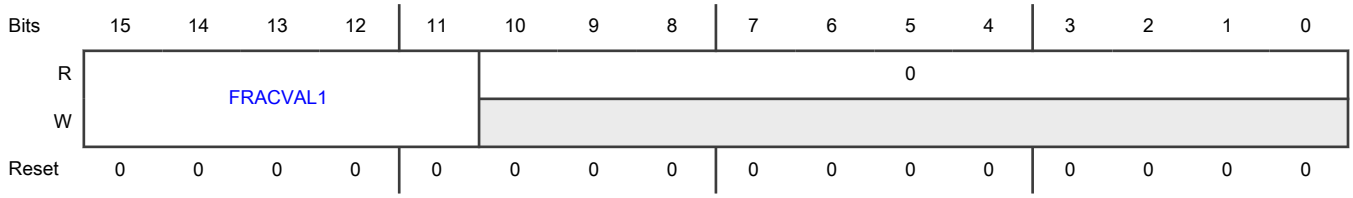
Register	Offset
SM0FRACVAL1	Ch
SM1FRACVAL1	6Ch
SM2FRACVAL1	CCh
SM3FRACVAL1	12Ch

Function

NOTE

The FRACVAL1 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRACVAL1 cannot be written when MCTRL[LDOK] is set. Reading FRACVAL1 reads the value in a buffer and not necessarily the value the PWM generator is currently using.

Diagram



Fields

Field	Function
15-11 FRACVAL1	Fractional Value 1 These bits act as a fractional addition to the value in the VAL1 register which controls the PWM period. With fractional delay enabled, PWM works in digital dithering mode. The PWM period is computed in terms of IPBus clock cycles. This fractional portion is accumulated at the end of every cycle until an additional whole IPBus cycle is reached. At this time the value being used for VAL1 is temporarily incremented, and the PWM period is extended by one clock to compensate for the accumulated fractional values.
10-0 —	Reserved

67.5.8 Value Register 1 (SM0VAL1 - SM3VAL1)

Offset

Register	Offset
SM0VAL1	Eh
SM1VAL1	6Eh
SM2VAL1	CEh
SM3VAL1	12Eh

Function

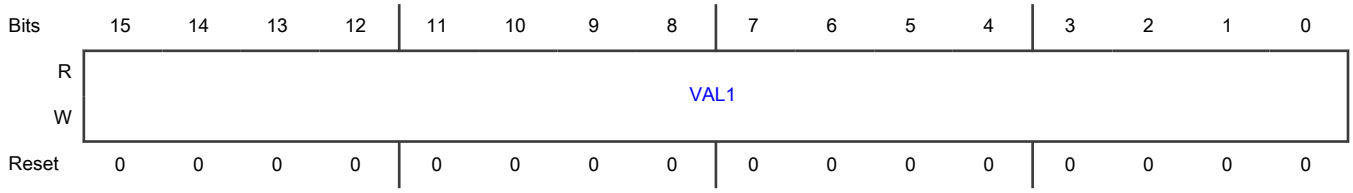
NOTE

The VAL1 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL1 cannot be written when MCTRL[LDOK] is set. Reading VAL1 reads the value in a buffer. It is not necessarily the value that the PWM generator is currently using.

When using FRACVAL1, limit the maximum value of VAL1 to 0xFFFE for unsigned applications or to 0x7FFE for signed applications, to avoid counter rollovers caused by accumulating the fractional period defined by FRACVAL1.

If the VAL1 register defines the timer period (Local Sync is selected as the counter initialization signal), a 100% duty cycle cannot be achieved on the PWM_X output. After the count reaches VAL1, the PWM_X output is low for a minimum of one count every cycle. When the Master Sync signal (only originated by the Local Sync from submodule 0) is used to control the timer period, the VAL1 register can be free for other functions such as PWM generation without the duty cycle limitation.

Diagram



Fields

Field	Function
15-0 VAL1	Value 1 The 16-bit signed value written to this buffered register defines the modulo count value (maximum count) for the submodule counter. When reaching this count value, the counter reloads itself with the contents of the INIT register and asserts the local sync signal while resetting PWM_X. This register is not byte accessible. **NOTE** The actual behavior takes effect when the counter equals VAL1+1.

67.5.9 Fractional Value Register 2 (SM0FRACVAL2 - SM3FRACVAL2)

Offset

Register	Offset
SM0FRACVAL2	10h
SM1FRACVAL2	70h
SM2FRACVAL2	D0h
SM3FRACVAL2	130h

Function

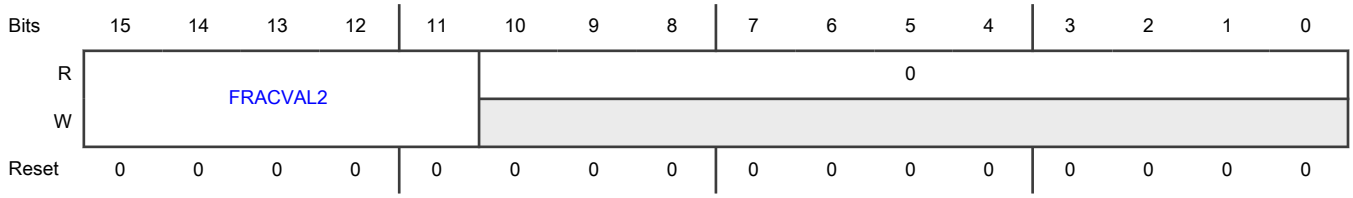
NOTE

The FRACVAL2 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRACVAL2 cannot be written when MCTRL[LDOK] is set. Reading FRACVAL2 reads the value in a buffer and not necessarily the value the PWM generator is currently using.

NOTE

FRCTRL[FRAC23_EN] should be set to 0 when the values of VAL2 and VAL3 cause the high or low time of the PWM output to be 3 cycles or less.

Diagram



Fields

Field	Function
15-11 FRACVAL2	Fractional Value 2 These bits act as a fractional addition to the value in the VAL2 register which controls the PWM_A turn on timing. It is also used to control the fractional addition to the turn off delay of PWM_B when MCTRL[IPOLx]=0 in complementary mode, CTRL2[INDEP]=0. With fractional delay enabled, PWM works in digital dithering mode. The PWM_A turn on delay is computed in terms of IPBus clock cycles. This fractional portion is accumulated at the end of every cycle until an additional whole IPBus cycle is reached. At this time the value being used for VAL2 is temporarily incremented, and the PWM_A turn on delay is extended by one clock to compensate for the accumulated fractional values.
10-0 —	Reserved

67.5.10 Value Register 2 (SM0VAL2 - SM3VAL2)

Offset

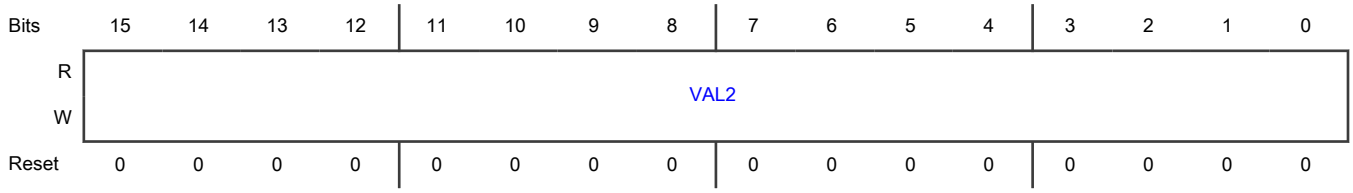
Register	Offset
SM0VAL2	12h
SM1VAL2	72h
SM2VAL2	D2h
SM3VAL2	132h

Function

NOTE

The VAL2 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL2 cannot be written when MCTRL[LDOK] is set. Reading VAL2 reads the value in a buffer and not necessarily the value that the PWM generator is currently using.

Diagram



Fields

Field	Function
15-0 VAL2	Value 2 The 16-bit signed value in this buffered register defines the count value to set PWM23 high. This register is not byte accessible. **NOTE** The actual behavior takes effect when the counter equals VAL2+1.

67.5.11 Fractional Value Register 3 (SM0FRACVAL3 - SM3FRACVAL3)

Offset

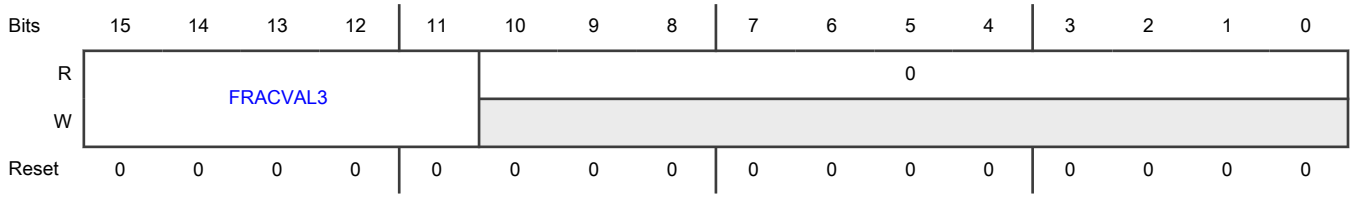
Register	Offset
SM0FRACVAL3	14h
SM1FRACVAL3	74h
SM2FRACVAL3	D4h
SM3FRACVAL3	134h

Function

NOTE

The FRACVAL3 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRACVAL3 cannot be written when MCTRL[LDOK] is set. Reading FRACVAL3 reads the value in a buffer and not necessarily the value the PWM generator is currently using.

Diagram



Fields

Field	Function
15-11 FRACVAL3	Fractional Value 3 These bits act as a fractional addition to the value in the VAL3 register which controls the PWM_A turn off timing. It is also used to control the fractional addition to the turn on delay of PWM_B when MCTRL[IPOLx]=0 in complementary mode, CTRL2[INDEP]=0. **NOTE** FRCTRL[FRAC23_EN] should be set to 0 when the values of VAL2 and VAL3 cause the high or low time of the PWM output to be 3 cycles or less. With fractional delay enabled, PWM works in digital dithering mode. The PWM_A turn off delay is computed in terms of IPBus clock cycles. This fractional portion is accumulated at the end of every cycle until an additional whole IPBus cycle is reached. At this time the value being used for VAL3 is temporarily incremented, and the PWM_A turn off delay is extended by one clock to compensate for the accumulated fractional values.
10-0 —	Reserved

67.5.12 Value Register 3 (SM0VAL3 - SM3VAL3)

Offset

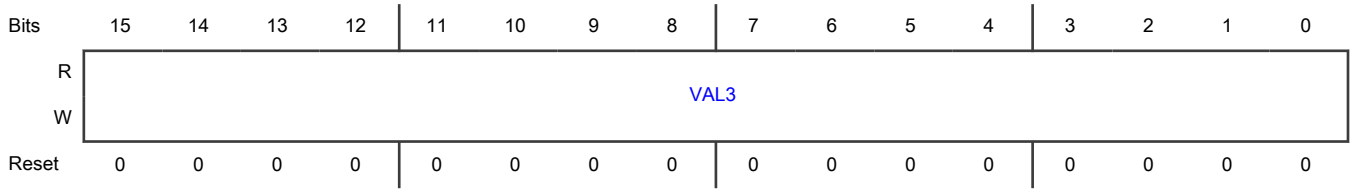
Register	Offset
SM0VAL3	16h
SM1VAL3	76h
SM2VAL3	D6h
SM3VAL3	136h

Function

NOTE

The VAL3 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL3 cannot be written when MCTRL[LDOK] is set. Reading VAL3 reads the value in a buffer and not necessarily the value that the PWM generator is currently using.

Diagram



Fields

Field	Function
15-0 VAL3	Value 3 The 16-bit signed value in this buffered register defines the count value to set PWM23 low. This register is not byte accessible. **NOTE** The actual behavior takes effect when the counter equals VAL3+1.

67.5.13 Fractional Value Register 4 (SM0FRACVAL4 - SM3FRACVAL4)

Offset

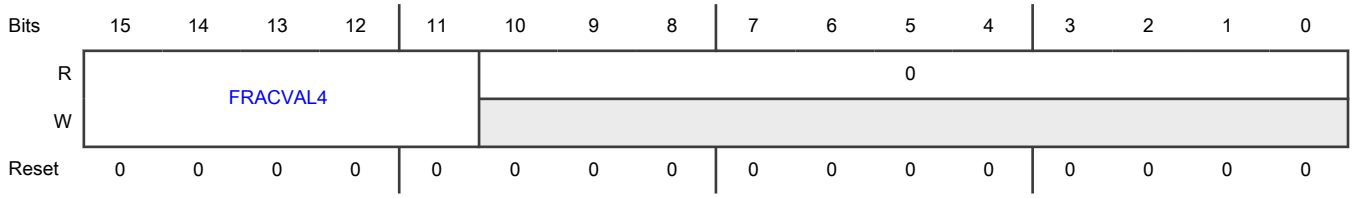
Register	Offset
SM0FRACVAL4	18h
SM1FRACVAL4	78h
SM2FRACVAL4	D8h
SM3FRACVAL4	138h

Function

NOTE

The FRACVAL4 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRACVAL4 cannot be written when MCTRL[LDOK] is set. Reading FRACVAL4 reads the value in a buffer and not necessarily the value the PWM generator is currently using.

Diagram



Fields

Field	Function
15-11 FRACVAL4	Fractional Value 4 These bits act as a fractional addition to the value in the VAL4 register which controls the PWM_B turn on timing. It is also used to control the fractional addition to the turn off delay of PWM_A when MCTRL[IPOLx]=1 in complementary mode, CTRL2[INDEP]=0. **NOTE** FRCTRL[FRAC45_EN] should be set to 0 when the values of VAL4 and VAL5 cause the high or low time of the PWM output to be 3 cycles or less. With fractional delay enabled, PWM works in digital dithering mode. The PWM_B turn on delay is computed in terms of IPBus clock cycles. This fractional portion is accumulated at the end of every cycle until an additional whole IPBus cycle is reached. At this time the value being used for VAL4 is temporarily incremented, and the PWM_B turn on delay is extended by one clock to compensate for the accumulated fractional values.
10-0 —	Reserved

67.5.14 Value Register 4 (SM0VAL4 - SM3VAL4)

Offset

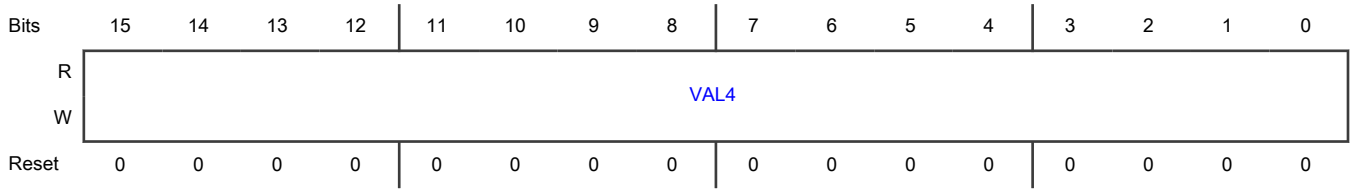
Register	Offset
SM0VAL4	1Ah
SM1VAL4	7Ah
SM2VAL4	DAh
SM3VAL4	13Ah

Function

NOTE

The VAL4 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL4 cannot be written when MCTRL[LDOK] is set. Reading VAL4 reads the value in a buffer and not necessarily the value that the PWM generator is currently using.

Diagram



Fields

Field	Function
15-0 VAL4	Value 4 The 16-bit signed value in this buffered register defines the count value to set PWM45 high. This register is not byte accessible. **NOTE** The actual behavior takes effect when the counter equals VAL4+1.

67.5.15 Fractional Value Register 5 (SM0FRACVAL5 - SM3FRACVAL5)

Offset

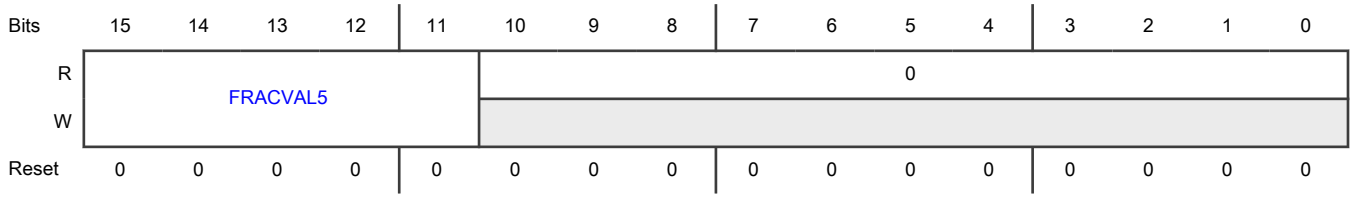
Register	Offset
SM0FRACVAL5	1Ch
SM1FRACVAL5	7Ch
SM2FRACVAL5	DCh
SM3FRACVAL5	13Ch

Function

NOTE

The FRACVAL5 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRACVAL5 cannot be written when MCTRL[LDOK] is set. Reading FRACVAL5 reads the value in a buffer and not necessarily the value the PWM generator is currently using.

Diagram



Fields

Field	Function
15-11 FRACVAL5	Fractional Value 5 These bits act as a fractional addition to the value in the VAL5 register which controls the PWM_B turn off timing. It is also used to control the fractional addition to the turn on delay of PWM_A when MCTRL[IPOLx]=1 in complementary mode, CTRL2[INDEP]=0. **NOTE** FRCTRL[FRAC45_EN] should be set to 0 when the values of VAL4 and VAL5 cause the high or low time of the PWM output to be 3 cycles or less. With fractional delay enabled, PWM works in digital dithering mode. The PWM_B turn off delay is computed in terms of IPBus clock cycles. This fractional portion is accumulated at the end of every cycle until an additional whole IPBus cycle is reached. At this time the value being used for VAL5 is temporarily incremented, and the PWM_B turn off delay is extended by one clock to compensate for the accumulated fractional values.
10-0 —	Reserved

67.5.16 Value Register 5 (SM0VAL5 - SM3VAL5)

Offset

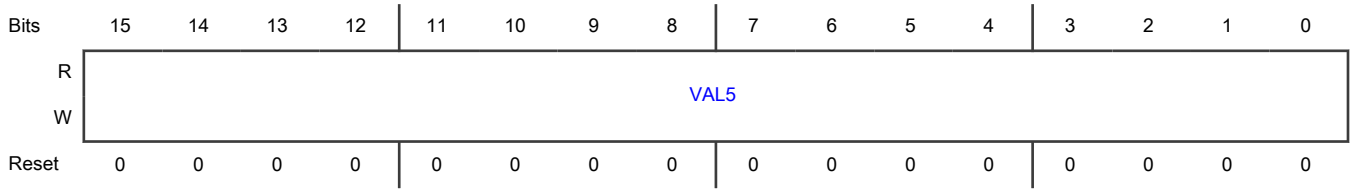
Register	Offset
SM0VAL5	1Eh
SM1VAL5	7Eh
SM2VAL5	DEh
SM3VAL5	13Eh

Function

NOTE

The VAL5 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL5 cannot be written when MCTRL[LDOK] is set. Reading VAL5 reads the value in a buffer and not necessarily the value that the PWM generator is currently using.

Diagram



Fields

Field	Function
15-0 VAL5	Value 5 The 16-bit signed value in this buffered register defines the count value to set PWM45 low. This register is not byte accessible. **NOTE** The actual behavior takes effect when the counter equals VAL5+1.

67.5.17 Fractional Control Register (SM0FRCTRL - SM3FRCTRL)

Offset

Register	Offset
SM0FRCTRL	20h
SM1FRCTRL	80h
SM2FRCTRL	E0h
SM3FRCTRL	140h

Function

NOTE

The FRAC1_EN bit is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRAC1_EN cannot be written when MCTRL[LDOK] is set. Reading FRAC1_EN reads the value in a buffer and not necessarily the value that the PWM generator is currently using.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TEST	0							0			FRAC45_...	0	FRAC23_...	FRAC1_EN	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 TEST	Test Status Bit This is a test bit for factory use. This bit resets to 0 but may be either 0 or 1 during PWM operation.
14-8 —	Reserved
7-5 —	Reserved
4 FRAC45_EN	Fractional Cycle Placement Enable for PWM_B This bit is used to enable the fractional cycle edge placement of PWM_B using the FRACVAL4 and FRACVAL5 registers. When disabled, bypass the fractional cycle edge placement of PWM_B. **NOTE** The FRAC45_EN bit is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRAC45_EN cannot be written when MCTRL[LDOK] is set. Reading FRAC45_EN reads the value in a buffer and not necessarily the value that the PWM generator is currently using. 0b - Disable fractional cycle placement for PWM_B. 1b - Enable fractional cycle placement for PWM_B.
3 —	Reserved
2 FRAC23_EN	Fractional Cycle Placement Enable for PWM_A This bit is used to enable the fractional cycle edge placement of PWM_A using the FRACVAL2 and FRACVAL3 registers. When disabled, bypass the fractional cycle edge placement of PWM_A. **NOTE** The FRAC23_EN bit is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRAC23_EN cannot be written when MCTRL[LDOK] is set. Reading FRAC23_EN reads the value in a buffer and not necessarily the value that the PWM generator is currently using. 0b - Disable fractional cycle placement for PWM_A. 1b - Enable fractional cycle placement for PWM_A.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 FRAC1_EN	Fractional Cycle PWM Period Enable This bit is used to enable the fractional cycle length of the PWM period using the FRACVAL1 register. When disabled, bypass the fractional cycle length of the PWM period. **NOTE** The FRAC1_EN bit is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRAC1_EN cannot be written when MCTRL[LDOK] is set. Reading FRAC1_EN reads the value in a buffer and not necessarily the value that the PWM generator is currently using. 0b - Disable fractional cycle length for the PWM period. 1b - Enable fractional cycle length for the PWM period.
0 —	Reserved

67.5.18 Output Control Register (SM0OCTRL - SM3OCTRL)

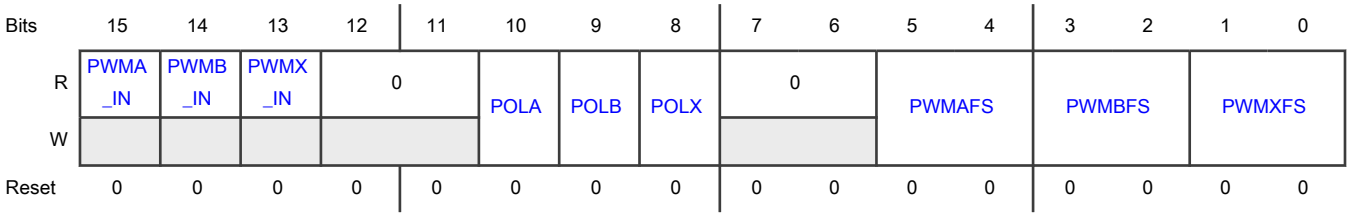
Offset

Register	Offset
SM0OCTRL	22h
SM1OCTRL	82h
SM2OCTRL	E2h
SM3OCTRL	142h

Function

Contains output controls for fault states.

Diagram



Fields

Field	Function
15 PWMA_IN	PWM_A Input This bit shows the logic value currently being driven into the PWM_A input. The reset state is undefined.
14 PWMB_IN	PWM_B Input This bit shows the logic value currently being driven into the PWM_B input. The reset state is undefined.
13 PWMX_IN	PWM_X Input This bit shows the logic value currently being driven into the PWM_X input. The reset state is undefined.
12-11 —	Reserved
10 POLA	PWM_A Output Polarity This bit inverts the PWM_A output polarity. 0b - PWM_A output not inverted. A high level on the PWM_A pin represents the "on" or "active" state. 1b - PWM_A output inverted. A low level on the PWM_A pin represents the "on" or "active" state.
9 POLB	PWM_B Output Polarity This bit inverts the PWM_B output polarity. 0b - PWM_B output not inverted. A high level on the PWM_B pin represents the "on" or "active" state. 1b - PWM_B output inverted. A low level on the PWM_B pin represents the "on" or "active" state.
8 POLX	PWM_X Output Polarity This bit inverts the PWM_X output polarity. 0b - PWM_X output not inverted. A high level on the PWM_X pin represents the "on" or "active" state. 1b - PWM_X output inverted. A low level on the PWM_X pin represents the "on" or "active" state.
7-6 —	Reserved
5-4	PWM_A Fault State

Table continues on the next page...

Table continued from the previous page...

Field	Function
PWMAFS	These bits determine the fault state for the PWM_A output during fault conditions and Stop mode. It may also define the output state during Wait and Debug modes depending on the settings of CTRL2[WAITEN] and CTRL2[DBGEN]. 00b - Output is forced to logic 0 state prior to consideration of output polarity control. 01b - Output is forced to logic 1 state prior to consideration of output polarity control. 10b,11b - Output is put in a high-impedance state.
3-2 PWMBFS	PWM_B Fault State These bits determine the fault state for the PWM_B output during fault conditions and Stop mode. It may also define the output state during Wait and Debug modes depending on the settings of CTRL2[WAITEN] and CTRL2[DBGEN]. 00b - Output is forced to logic 0 state prior to consideration of output polarity control. 01b - Output is forced to logic 1 state prior to consideration of output polarity control. 10b,11b - Output is put in a high-impedance state.
1-0 PWMXFS	PWM_X Fault State These bits determine the fault state for the PWM_X output during fault conditions and Stop mode. It may also define the output state during Wait and Debug modes depending on the settings of CTRL2[WAITEN] and CTRL2[DBGEN]. 00b - Output is forced to logic 0 state prior to consideration of output polarity control. 01b - Output is forced to logic 1 state prior to consideration of output polarity control. 10b,11b - Output is put in a high-impedance state.

67.5.19 Status Register (SM0STS - SM3STS)

Offset

Register	Offset
SM0STS	24h
SM1STS	84h
SM2STS	E4h
SM3STS	144h

Function

Contains Compare and Capture flag status.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	RUF	REF	RF	CFA1	CFA0	CFB1	CFB0	CFX1	CFX0	CMPF					
W			W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 —	Reserved
14 RUF	Registers Updated Flag This bit is set when one of the INIT, VALx, FRACVALx, or CTRL[PRSC] registers has been written, which indicates potentially non-coherent data in the set of double buffered registers. Clear this bit by a proper reload sequence consisting of a reload signal while MCTRL[LDOK] = 1. Reset clears this bit. 0b - No register update has occurred since last reload. 1b - At least one of the double buffered registers has been updated since the last reload.
13 REF	Reload Error Flag This bit is set when a reload cycle occurs while MCTRL[LDOK] is 0 and the double buffered registers are in a non-coherent state (STS[RUF] = 1). Clear this bit by writing a logic one to this location. Reset clears this bit. 0b - No reload error occurred. 1b - Reload signal occurred with non-coherent data and MCTRL[LDOK] = 0.
12 RF	Reload Flag This bit is set at the beginning of every reload cycle regardless of the state of MCTRL[LDOK]. Clear this bit by writing a logic one to this location when DMAEN[VALDE] is clear (non-DMA mode). This bit can also be cleared by the DMA done signal when DMAEN[VALDE] is set (DMA mode) . Reset clears this bit. 0b - No new reload cycle since last STS[RF] clearing 1b - New reload cycle since last STS[RF] clearing
11 CFA1	Capture Flag A1 This bit is set when a capture event occurs on the Capture A1 circuit . This bit is cleared by writing a one to this bit position if DMAEN[CA1DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CA1DE] is set (DMA mode) . Reset clears this bit.
10 CFA0	Capture Flag A0 This bit is set when a capture event occurs on the Capture A0 circuit . This bit is cleared by writing a one to this bit position if DMAEN[CA0DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CA0DE] is set (DMA mode) . Reset clears this bit.
9	Capture Flag B1

Table continues on the next page...

Table continued from the previous page...

Field	Function
CFB1	This bit is set when a capture event occurs on the Capture B1 circuit . This bit is cleared by writing a one to this bit position if DMAEN[CB1DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CB1DE] is set (DMA mode) . Reset clears this bit.
8 CFB0	Capture Flag B0 This bit is set when a capture event occurs on the Capture B0 circuit . This bit is cleared by writing a one to this bit position if DMAEN[CB0DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CB0DE] is set (DMA mode) . Reset clears this bit.
7 CFX1	Capture Flag X1 This bit is set when a capture event occurs on the Capture X1 circuit . This bit is cleared by writing a one to this bit position if DMAEN[CX1DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CX1DE] is set (DMA mode) . Reset clears this bit.
6 CFX0	Capture Flag X0 This bit is set when a capture event occurs on the Capture X0 circuit . This bit is cleared by writing a one to this bit position if DMAEN[CX0DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CX0DE] is set (DMA mode) . Reset clears this bit.
5-0 CMPF	Compare Flags These bits are set when the submodule counter value matches the value of one of the VALx registers. Clear these bits by writing a 1 to a bit position. 00_0000b - No compare event has occurred for a particular VALx value. 00_0001b - A compare event has occurred for a particular VALx value.

67.5.20 Interrupt Enable Register (SM0INTEN - SM3INTEN)

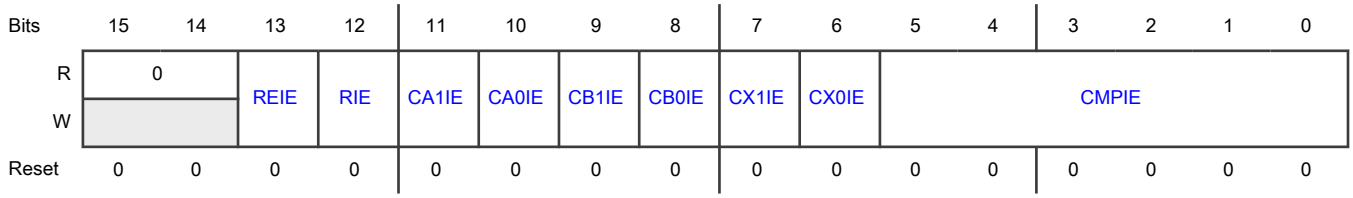
Offset

Register	Offset
SM0INTEN	26h
SM1INTEN	86h
SM2INTEN	E6h
SM3INTEN	146h

Function

Contains Compare and Capture interrupt enables.

Diagram



Fields

Field	Function
15-14 —	Reserved
13 REIE	Reload Error Interrupt Enable This bit enables the reload error flag, STS[REF], to generate CPU interrupt requests. Reset clears this bit. 0b - STS[REF] CPU interrupt requests disabled 1b - STS[REF] CPU interrupt requests enabled
12 RIE	Reload Interrupt Enable This bit enables the reload flag, STS[RF], to generate CPU interrupt requests. Reset clears this bit. 0b - STS[RF] CPU interrupt requests disabled 1b - STS[RF] CPU interrupt requests enabled
11 CA1IE	Capture A 1 Interrupt Enable This bit allows the STS[CFA1] flag to create an interrupt request to the CPU. Do not set this bit and DMAEN[CA1DE]. 0b - Interrupt request disabled for STS[CFA1] 1b - Interrupt request enabled for STS[CFA1]
10 CA0IE	Capture A 0 Interrupt Enable This bit allows the STS[CFA0] flag to create an interrupt request to the CPU. Do not set this bit and DMAEN[CA0DE]. 0b - Interrupt request disabled for STS[CFA0]. 1b - Interrupt request enabled for STS[CFA0].
9 CB1IE	Capture B 1 Interrupt Enable This bit allows the STS[CFB1] flag to create an interrupt request to the CPU. Do not set this bit and DMAEN[CB1DE]. 0b - Interrupt request disabled for STS[CFB1]. 1b - Interrupt request enabled for STS[CFB1].
8	Capture B 0 Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
CB0IE	This bit allows the STS[CFB0] flag to create an interrupt request to the CPU. Do not set this bit and DMAEN[CB0DE]. 0b - Interrupt request disabled for STS[CFB0]. 1b - Interrupt request enabled for STS[CFB0].
7 CX1IE	Capture X 1 Interrupt Enable This bit allows the STS[CFX1] flag to create an interrupt request to the CPU. Do not set this bit and DMAEN[CX1DE]. 0b - Interrupt request disabled for STS[CFX1]. 1b - Interrupt request enabled for STS[CFX1].
6 CX0IE	Capture X 0 Interrupt Enable This bit allows the STS[CFX0] flag to create an interrupt request to the CPU. Do not set this bit and DMAEN[CX0DE]. 0b - Interrupt request disabled for STS[CFX0]. 1b - Interrupt request enabled for STS[CFX0].
5-0 CMPIE	Compare Interrupt Enables These bits enable the STS[CMPF] flags to cause a compare interrupt request to the CPU. 00_0000b - The corresponding STS[CMPF] bit will not cause an interrupt request. 00_0001b - The corresponding STS[CMPF] bit will cause an interrupt request.

67.5.21 DMA Enable Register (SM0DMAEN - SM3DMAEN)

Offset

Register	Offset
SM0DMAEN	28h
SM1DMAEN	88h
SM2DMAEN	E8h
SM3DMAEN	148h

Function

Contains controls for DMA.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						VALDE	FAND	CAPTDE		CA1DE	CA0DE	CB1DE	CB0DE	CX1DE	CX0DE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-10 —	Reserved
9 VALDE	Value Registers DMA Enable This bit enables DMA write requests for the VALx and FRACVALx registers when STS[RF] is set. Reset clears this bit. 0b - DMA write requests disabled 1b - Enabled. DMA write requests for the VALx and FRACVALx registers enabled
8 FAND	FIFO Watermark AND Control This bit works in conjunction with the DMAEN[CAPTDE] field when it is set to watermark mode (DMAEN[CAPTDE] = 01). While DMAEN[CAxDE], DMAEN[CBxDE], and DMAEN[CXxDE] determine which FIFO watermarks the DMA read request is sensitive to. This bit determines if the selected watermarks are AND'ed together or OR'ed together to create the request. 0b - Selected FIFO watermarks are OR'ed together. 1b - Selected FIFO watermarks are AND'ed together.
7-6 CAPTDE	Capture DMA Enable Source Select These bits select the source of enabling the DMA read requests for the capture FIFOs. Reset clears these bits. 00b - Read DMA requests disabled. 01b - Exceeding a FIFO watermark sets the DMA read request. This requires at least one of DMAEN[CA1DE], DMAEN[CA0DE], DMAEN[CB1DE], DMAEN[CB0DE], DMAEN[CX1DE], or DMAEN[CX0DE] to be set to determine which watermark(s) the DMA request is sensitive. 10b - A local synchronization (VAL1 matches counter) sets the read DMA request. 11b - A local reload (STS[RF] being set) sets the read DMA request.
5 CA1DE	Capture A1 FIFO DMA Enable This bit enables DMA read requests for the Capture A1 FIFO data when STS[CFA1] is set. Reset clears this bit. Do not set this bit and INTEN[CA1IE].
4 CA0DE	Capture A0 FIFO DMA Enable This bit enables DMA read requests for the Capture A0 FIFO data when STS[CFA0] is set. Reset clears this bit. Do not set both this bit and INTEN[CA0IE].

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 CB1DE	Capture B1 FIFO DMA Enable This bit enables DMA read requests for the Capture B1 FIFO data when STS[CFB1] is set. Reset clears this bit. Do not set both this bit and INTEN[CB1IE].
2 CB0DE	Capture B0 FIFO DMA Enable This bit enables DMA read requests for the Capture B0 FIFO data when STS[CFB0] is set. Reset clears this bit. Do not set both this bit and INTEN[CB0IE].
1 CX1DE	Capture X1 FIFO DMA Enable This bit enables DMA read requests for the Capture X1 FIFO data when STS[CFX1] is set. Reset clears this bit. Do not set both this bit and INTEN[CX1IE].
0 CX0DE	Capture X0 FIFO DMA Enable This bit enables DMA read requests for the Capture X0 FIFO data when STS[CFX0] is set. Reset clears this bit. Do not set both this bit and INTEN[CX0IE].

67.5.22 Output Trigger Control Register (SM0TCTRL - SM3TCTRL)

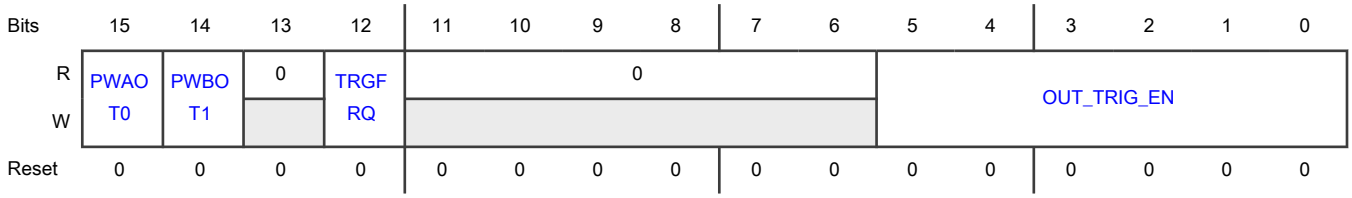
Offset

Register	Offset
SM0TCTRL	2Ah
SM1TCTRL	8Ah
SM2TCTRL	EAh
SM3TCTRL	14Ah

Function

Contains trigger controls.

Diagram



Fields

Field	Function
15 PWAOT0	Mux Output Trigger 0 Source Select This bit selects which signal to bring out on the PWM's PWM_MUX_TRIG0 port. 0b - Route the PWM_OUT_TRIG0 signal to PWM_MUX_TRIG0 port. 1b - Route the PWM_A output to the PWM_MUX_TRIG0 port.
14 PWBOT1	Mux Output Trigger 1 Source Select This bit selects which signal to bring out on the PWM's PWM_MUX_TRIG1 port. 0b - Route the PWM_OUT_TRIG1 signal to PWM_MUX_TRIG1 port. 1b - Route the PWM_B output to the PWM_MUX_TRIG1 port.
13 —	Reserved
12 TRGFRQ	Trigger Frequency This bit allows control over the frequency of the trigger outputs when using non-zero values of CTRL[LDFQ]. 0b - Trigger outputs are generated during every PWM period even if the PWM is not reloaded every period due to CTRL[LDFQ] being non-zero. 1b - Trigger outputs are generated only during the final PWM period prior to a reload opportunity when the PWM is not reloaded every period due to CTRL[LDFQ] being non-zero.
11-6 —	Reserved
5-0 OUT_TRIG_EN	Output Trigger Enables These bits enable the generation of PWM_OUT_TRIG0 and PWM_OUT_TRIG1 outputs based on the counter value matching the value in one or more of the VAL0-5 registers. **NOTE** Due to delays in creating the PWM outputs, the output trigger signals lead the PWM output edges by 2-3 clock cycles depending on the fractional cycle value being used. 1x_xxxxb - PWM_OUT_TRIG1 will set when the counter value matches the VAL5 value. x1_xxxxb - PWM_OUT_TRIG0 will set when the counter value matches the VAL4 value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	xx_1xxxb - PWM_OUT_TRIG1 will set when the counter value matches the VAL3 value. xx_x1xxb - PWM_OUT_TRIG0 will set when the counter value matches the VAL2 value. xx_xx1xb - PWM_OUT_TRIG1 will set when the counter value matches the VAL1 value. xx_xxx1b - PWM_OUT_TRIG0 will set when the counter value matches the VAL0 value.

67.5.23 Fault Disable Mapping Register 0 (SM0DISMAP0 - SM3DISMAP0)

Offset

Register	Offset
SM0DISMAP0	2Ch
SM1DISMAP0	8Ch
SM2DISMAP0	ECh
SM3DISMAP0	14Ch

Function

This register determines which PWM pins are disabled by the fault protection inputs. Reset sets all of the bits in the fault disable mapping register.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	1				DIS0X				DIS0B				DIS0A			
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
15-12 —	Reserved
11-8 DIS0X	PWM_X Fault Disable Mask 0 Each of the four bits of this field is one-to-one associated with the four FAULTx inputs of fault channel 0. The PWM_X output is turned off if there is a logic 1 on a FAULTx input and a 1 in the corresponding bit of this field. DIS0X[0] is associated with FAULT0. DIS0X[1] is associated with FAULT1. DIS0X[2] is associated with FAULT2. DIS0X[3] is associated with FAULT3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	A reset sets all bits in this field.
7-4 DIS0B	PWM_B Fault Disable Mask 0 Each of the four bits of this field is one-to-one associated with the four FAULTx inputs of fault channel 0. The PWM_B output is turned off if there is a logic 1 on a FAULTx input and a 1 in the corresponding bit of this field. DIS0B[0] is associated with FAULT0. DIS0B[1] is associated with FAULT1. DIS0B[2] is associated with FAULT2. DIS0B[3] is associated with FAULT3. A reset sets all bits in this field.
3-0 DIS0A	PWM_A Fault Disable Mask 0 Each of the four bits of this field is one-to-one associated with the four FAULTx inputs of fault channel 0. The PWM_A output is turned off if there is a logic 1 on a FAULTx input and a 1 in the corresponding bit of this field. DIS0A[0] is associated with FAULT0. DIS0A[1] is associated with FAULT1. DIS0A[2] is associated with FAULT2. DIS0A[3] is associated with FAULT3. A reset sets all bits in this field.

67.5.24 Deadtime Count Register 0 (SM0DTCNT0 - SM3DTCNT0)

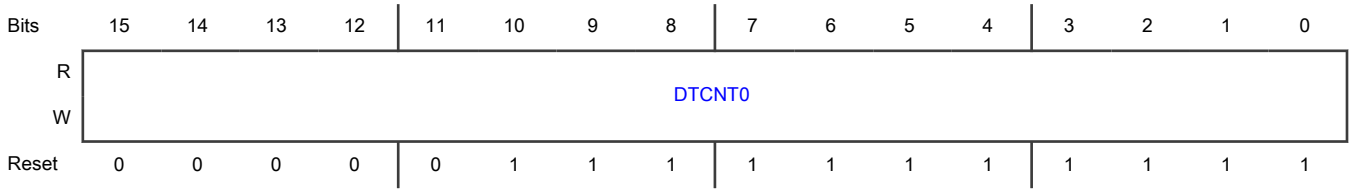
Offset

Register	Offset
SM0DTCNT0	30h
SM1DTCNT0	90h
SM2DTCNT0	F0h
SM3DTCNT0	150h

Function

Deadtime operation applies only to complementary channel operation. The values written to the DTCNTx registers are in terms of IPBus clock cycles regardless of the setting of CTRL[PRSC] and/or CTRL2[CLK_SEL]. Reset sets the deadtime count registers to a default value of 0x07FF, selecting a deadtime of 2047 IPBus clock cycles, when fractional delay is not enabled. The DTCNTx registers are not byte accessible.

Diagram



Fields

Field	Function
15-0 DTCNT0	DTCNT0 The DTCNT0 field is used to control the deadtime during 0 to 1 transitions of the PWM_A output (assuming normal polarity). The DTCNT0 field is interpreted differently depending on whether the fractional delays are enabled or not. • If the fractional delays are disabled (FRCTRL[FRAC23_EN] cleared to 0 when MCTRL[IPOL]=0, or FRCTRL[FRAC45_EN] cleared to 0 when MCTRL[IPOL]=1), then the upper 5 bits of DTCNT0 are ignored and the remaining 11 bits are used to specify the number of PWM clock cycles of deadtime, i.e. the number cycles of deadtime = DTCNT0[10:0] . In this case, the maximum value is 0x07FF which indicates 2047 cycles of deadtime. • If the fractional delays are enabled (FRCTRL[FRAC23_EN] set to 1 when MCTRL[IPOL]=0, or FRCTRL[FRAC45_EN] set to 1 when MCTRL[IPOL]=1), then the upper 11 bits of DTCNT0 represent the number of PWM clock cycles of deadtime, while the lower 5 bits of each register represent the fractional number of cycle. With fractional delay enabled, PWM works in digital dithering mode. When the accumulation of fractional cycles exceeds 1, the whole number specified in DTCNT0[15:5] increments one additional cycle, and the reminder is used for the next fractional cycle accumulation. **NOTE** It is highly recommended to clear DTCNT0[4:0], to prevent unexpected deadtime increase, when the fractional digital dithering delays are adopted. Then the number of cycles of deadtime = DTCNT0[15:5].

67.5.25 Deadtime Count Register 1 (SM0DTCNT1 - SM3DTCNT1)

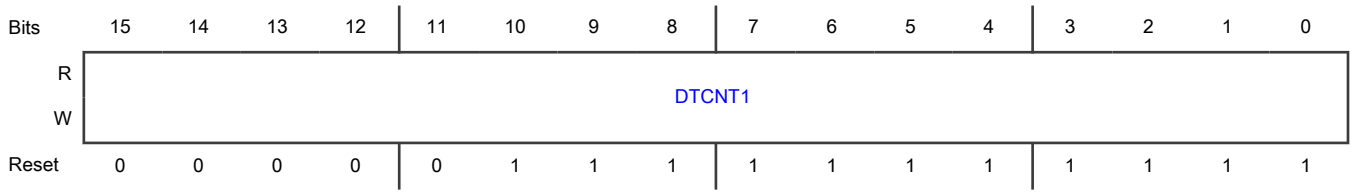
Offset

Register	Offset
SM0DTCNT1	32h
SM1DTCNT1	92h
SM2DTCNT1	F2h
SM3DTCNT1	152h

Function

Deadtime operation applies only to complementary channel operation. The values written to the DTCNTx registers are in terms of IPBus clock cycles regardless of the setting of CTRL[PRSC] and/or CTRL2[CLK_SEL]. Reset sets the deadtime count registers to a default value of 0x07FF, selecting a deadtime of 2047 IPBus clock cycles, when fractional delay is not enabled. The DTCNTx registers are not byte accessible.

Diagram



Fields

Field	Function
15-0 DTCNT1	DTCNT1 The DTCNT1 field is used to control the deadtime during 0 to 1 transitions of the PWM_B output (assuming normal polarity). The DTCNT1 field is interpreted differently depending on whether the fractional delays are enabled or not. • If the fractional delays are disabled (FRCTRL[FRAC23_EN] cleared to 0 when MCTRL[IPOL]=0, or FRCTRL[FRAC45_EN] cleared to 0 when MCTRL[IPOL]=1), then the upper 5 bits of DTCNT1 are ignored and the remaining 11 bits are used to specify the number of PWM clock cycles of deadtime, i.e. the number cycles of deadtime = DTCNT1[10:0] . In this case, the maximum value is 0x07FF which indicates 2047 cycles of deadtime. • If the fractional delays are enabled (FRCTRL[FRAC23_EN] set to 1 when MCTRL[IPOL]=0, or FRCTRL[FRAC45_EN] set to 1 when MCTRL[IPOL]=1), then the upper 11 bits of DTCNT1 represent the number of PWM clock cycles of deadtime, while the lower 5 bits of each register represent the fractional number of cycle. In this case, the maximum value is 0xFFFF which represents 2047 + 31/32 cycles of deadtime. With fractional delay enabled, PWM works in digital dithering mode. When the accumulation of fractional cycles exceeds 1, the whole number specified in DTCNT1[15:5] increments one additional cycle, and the reminder is used for the next fractional cycle accumulation. **NOTE** It is highly recommended to clear DTCNT1[4:0], to prevent unexpected deadtime increase, when the fractional digital dithering delays are adopted. Then the number of cycles of deadtime = DTCNT1[15:5].

67.5.26 Capture Control A Register (SM0CAPTCTRLA)

Offset

Register	Offset
SM0CAPTCTRLA	34h

Function

Contains capture controls for mode A.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA1CNT			CA0CNT			CFAWM		EDGCNTA...	INP_SELA	EDGA1		EDGA0		ONESHOTA	ARMA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 CA1CNT	Capture A1 FIFO Word Count This field reflects the number of words in the Capture A1 FIFO. (FIFO depth is 1.)
12-10 CA0CNT	Capture A0 FIFO Word Count This field reflects the number of words in the Capture A0 FIFO. (FIFO depth is 1.)
9-8 CFAWM	Capture A FIFOs Water Mark This field represents the water mark level for capture A FIFOs. The capture flags, STS[CFA1] and STS[CFA0], are not set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTA_EN	Edge Counter A Enable This field enables the edge counter which counts rising and falling edges on the PWM_A input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELA	Input Select A This field selects between the raw PWM_A input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_A input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLA[EDGA0] and CAPTCTRLA[EDGA1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRA[EDGA0] and/or CAPTCTRLA[EDGA1] fields to enable one or both of the capture registers.
5-4 EDGA1	Edge A 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Capture rising edges 11b - Capture any edge
3-2 EDGA0	Edge A 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTA	One Shot Mode A This bit selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLA[ARMA] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLA[ARMA] is set. Once a capture occurs, capture circuit 0 is disarmed and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and CAPTCTRLA[ARMA] is cleared. No further captures are performed until CAPTCTRLA[ARMA] is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and CAPTCTRLA[ARMA] is then cleared.
0 ARMA	Arm A Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLA[EDGAx] is enabled.

67.5.27 Capture Compare A Register (SM0CAPTCOMPA)

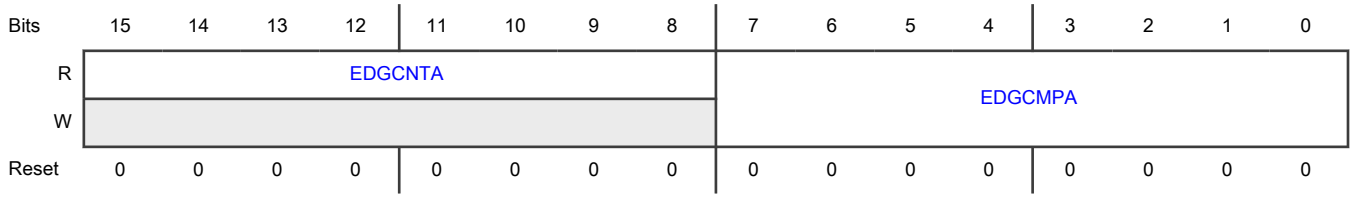
Offset

Register	Offset
SM0CAPTCOMPA	36h

Function

Contains capture and compare values for mode A.

Diagram



Fields

Field	Function
15-8 EDGCNTA	Edge Counter A This field contains the edge counter value for the PWM_A input capture circuitry.
7-0 EDGCMPA	Edge Compare A This field is the compare value associated with the edge counter for the PWM_A input capture circuitry.

67.5.28 Capture Control B Register (SM0CAPTCTRLB)

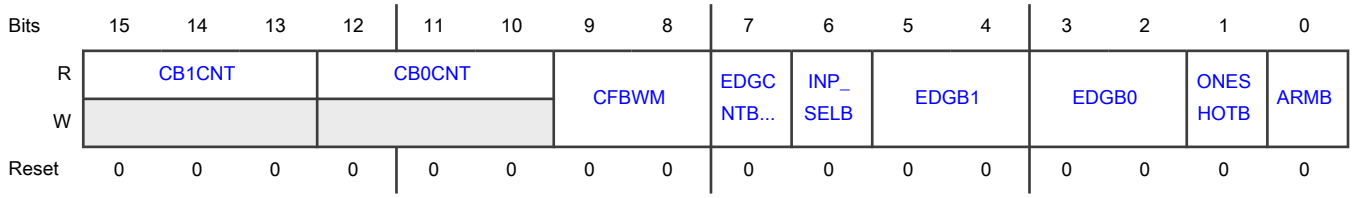
Offset

Register	Offset
SM0CAPTCTRLB	38h

Function

Contains capture controls for mode B.

Diagram



Fields

Field	Function
15-13 CB1CNT	Capture B1 FIFO Word Count This field reflects the number of words in the Capture B1 FIFO. (FIFO depth is 1.)
12-10 CB0CNT	Capture B0 FIFO Word Count This field reflects the number of words in the Capture B0 FIFO. (FIFO depth is 1.)

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 CFBWM	Capture B FIFOs Water Mark This field represents the water mark level for capture B FIFOs. The capture flags, STS[CFB1] and STS[CFB0], will not be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTB_EN	Edge Counter B Enable This field enables the edge counter which counts rising and falling edges on the PWM_B input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELB	Input Select B This field selects between the raw PWM_B input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_B input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLB[EDGB0] and CAPTCTRLB[EDGB1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRB[EDGB0] and/or CAPTCTRLB[EDGB1] fields to enable one or both of the capture registers.
5-4 EDGB1	Edge B 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2 EDGB0	Edge B 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTB	One Shot Mode B This bit selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLB[ARMB] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLB[ARMB] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and CAPTCTRLB[ARMB] is cleared. No further captures are performed until CAPTCTRLB[ARMB] is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and CAPTCTRLB[ARMB] is then cleared.
0 ARMB	Arm B Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLB[EDGBx] is enabled.

67.5.29 Capture Compare B Register (SM0CAPTCOMPB)

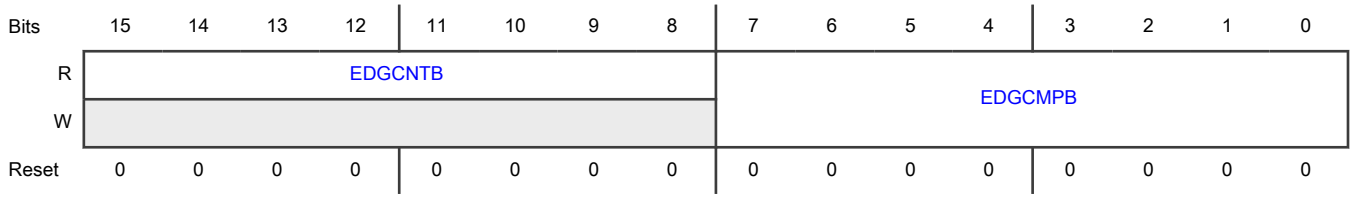
Offset

Register	Offset
SM0CAPTCOMPB	3Ah

Function

Contains capture and compare values for mode B.

Diagram



Fields

Field	Function
15-8 EDGCNTB	Edge Counter B This field contains the edge counter value for the PWM_B input capture circuitry.
7-0 EDGCMPB	Edge Compare B This field is the compare value associated with the edge counter for the PWM_B input capture circuitry.

67.5.30 Capture Control X Register (SM0CAPTCTRLX)

Offset

Register	Offset
SM0CAPTCTRLX	3Ch

Function

Contains capture controls for mode X.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CX1CNT				CX0CNT				CFXWM				EDGCNTX...	INP_SELX	EDGX1		EDGX0		ONESHOTX	ARMX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 CX1CNT	Capture X1 FIFO Word Count This field reflects the number of words in the Capture X1 FIFO. (FIFO depth is 1.)
12-10 CX0CNT	Capture X0 FIFO Word Count This field reflects the number of words in the Capture X0 FIFO. (FIFO depth is 1.)
9-8 CFXWM	Capture X FIFOs Water Mark This field represents the water mark level for capture X FIFOs. The capture flags, STS[CFX1] and STS[CFX0], will not be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTX_EN	Edge Counter X Enable This bit enables the edge counter which counts rising and falling edges on the PWM_X input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELX	Input Select X This bit selects between the raw PWM_X input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_X input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the

Table continues on the next page...

Table continued from the previous page...

Field	Function
	CAPTCTRLX[EDGX0] and CAPTCTRLX[EDGX1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRX[EDGX0] and/or CAPTCTRLX[EDGX1] fields to enable one or both of the capture registers.
5-4 EDGX1	Edge X 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2 EDGX0	Edge X 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTX	One Shot Mode Aux This field selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and the ARMX bit is cleared. No further captures are performed until the ARMX bit is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and the ARMX bit is then cleared.
0 ARMX	Arm X Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLX[EDGXx] is enabled.

67.5.31 Capture Compare X Register (SM0CAPTCOMPX)

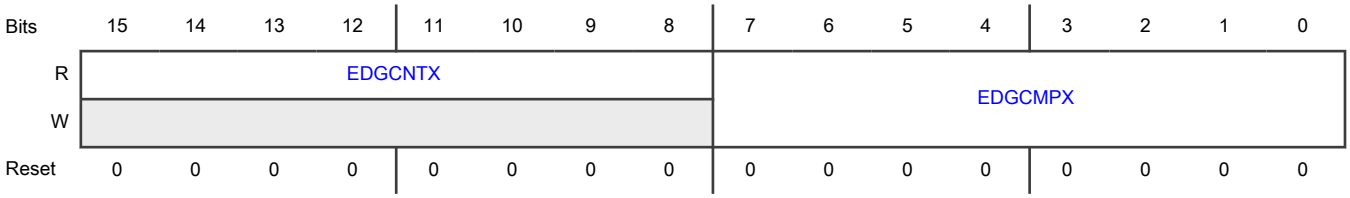
Offset

Register	Offset
SM0CAPTCOMPX	3Eh

Function

Contains capture and control values for mode X.

Diagram



Fields

Field	Function
15-8 EDGCNTX	Edge Counter X This field contains the edge counter value for the PWM_X input capture circuitry.
7-0 EDGCMPX	Edge Compare X This field is the compare value associated with the edge counter for the PWM_X input capture circuitry.

67.5.32 Capture Value 0 Register (SM0CVAL0)

Offset

Register	Offset
SM0CVAL0	40h

Function

Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CAPTVAL0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 CAPTVAL0	Capture Value 0 This field stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLX[EDGX0]. Each capture increases the value of CAPTCTRLX[CX0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX0CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.33 Capture Value 0 Cycle Register (SM0CVAL0CYC)

Offset

Register	Offset
SM0CVAL0CYC	42h

Function

Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												CVAL0CYC			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-4 —	Reserved
3-0 CVAL0CYC	Capture Value 0 Cycle This field stores the cycle number corresponding to the value captured in CVAL0. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

67.5.34 Capture Value 1 Register (SM0CVAL1)

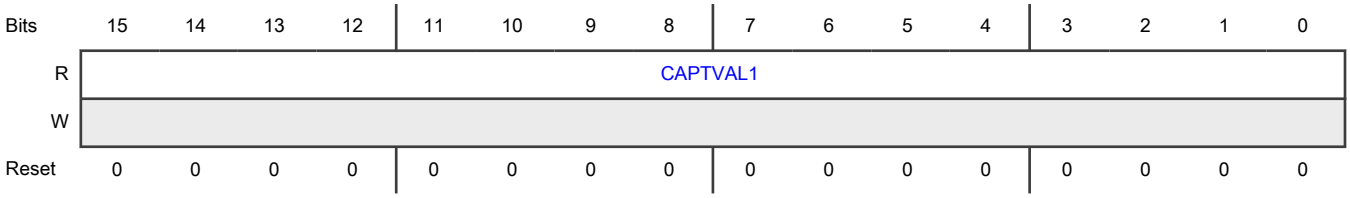
Offset

Register	Offset
SM0CVAL1	44h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL1	Capture Value 1 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLX[EDGX1]. Each capture increases the value of CAPTCTRLX[CX1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX1CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.35 Capture Value 1 Cycle Register (SM0CVAL1CYC)

Offset

Register	Offset
SM0CVAL1CYC	46h

Function

Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												CVAL1CYC			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-4 —	Reserved
3-0 CVAL1CYC	Capture Value 1 Cycle This field stores the cycle number corresponding to the value captured in CVAL1. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle. Resets to 0 at POR or hard reset.

67.5.36 Capture Value 2 Register (SM0CVAL2)

Offset

Register	Offset
SM0CVAL2	48h

Function

Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CAPTVAL2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 CAPTVAL2	Capture Value 2 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLA[EDGA0]. Each capture increases the value of CAPTCTRLA[CA0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLA[CA0CNT] by 1 until 0 is reached. This field is not byte accessible.

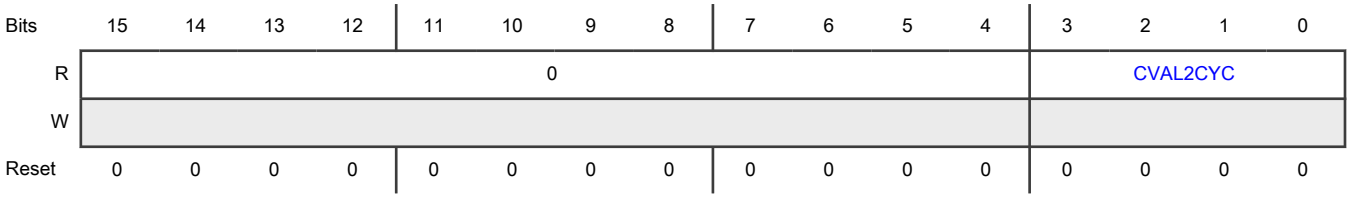
67.5.37 Capture Value 2 Cycle Register (SM0CVAL2CYC)

Offset

Register	Offset
SM0CVAL2CYC	4Ah

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL2CYC	Capture Value 2 Cycle This field stores the cycle number corresponding to the value captured in CVAL2. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

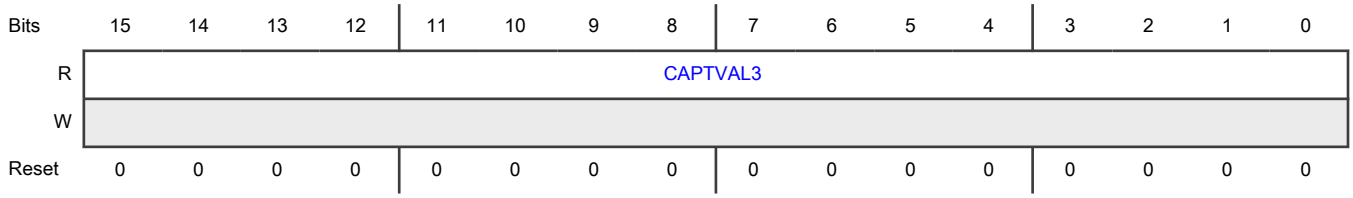
67.5.38 Capture Value 3 Register (SM0CVAL3)

Offset

Register	Offset
SM0CVAL3	4Ch

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL3	Capture Value 3 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLA[EDGA1]. Each capture increases the value of CAPTCTRLA[CA1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLA[CA1CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.39 Capture Value 3 Cycle Register (SM0CVAL3CYC)

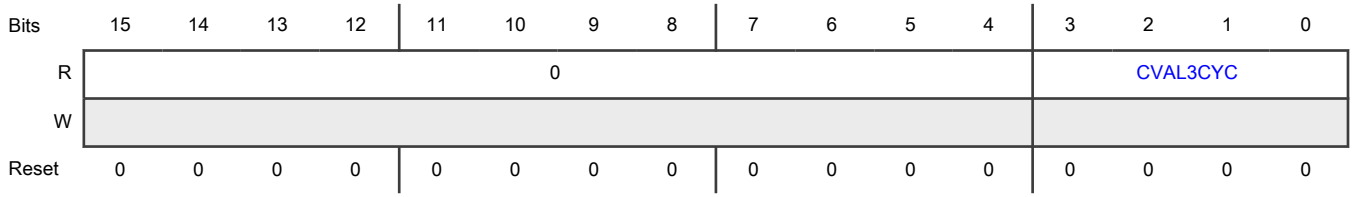
Offset

Register	Offset
SM0CVAL3CYC	4Eh

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL3CYC	Capture Value 3 Cycle This field stores the cycle number corresponding to the value captured in CVAL3. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

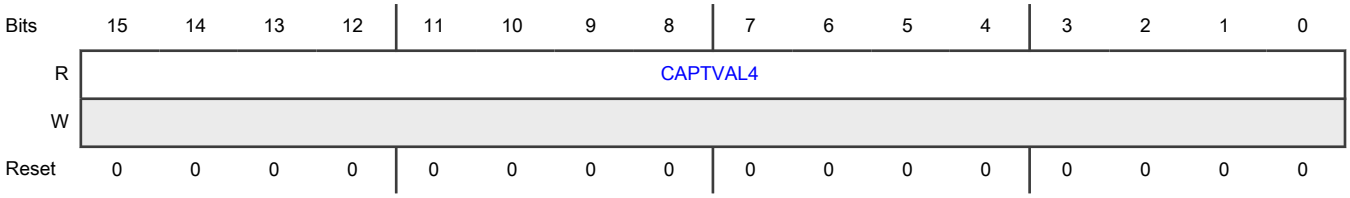
67.5.40 Capture Value 4 Register (SM0CVAL4)

Offset

Register	Offset
SM0CVAL4	50h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL4	Capture Value 4 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLB[EDGB0]. Each capture increases the value of CAPTCTRLB[CB0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLB[CB0CNT] by 1 until 0 is reached. This field is not byte accessible.

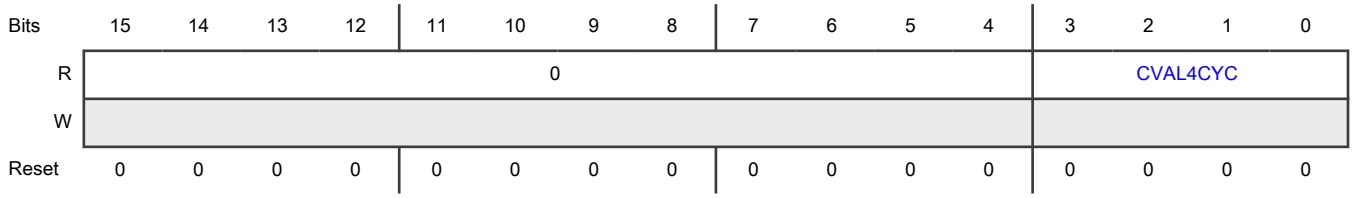
67.5.41 Capture Value 4 Cycle Register (SM0CVAL4CYC)

Offset

Register	Offset
SM0CVAL4CYC	52h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL4CYC	Capture Value 4 Cycle This field stores the cycle number corresponding to the value captured in CVAL4. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

67.5.42 Capture Value 5 Register (SM0CVAL5)

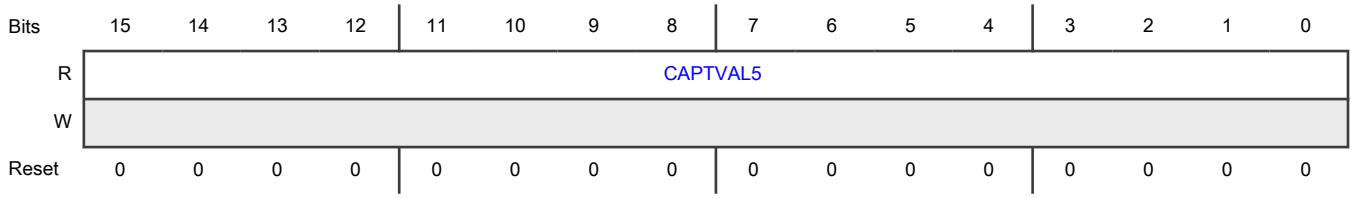
Offset

Register	Offset
SM0CVAL5	54h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL5	Capture Value 5 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLB[EDGB1]. Each capture increases the value of CAPTCTRLB[CB1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLB[CB1CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.43 Capture Value 5 Cycle Register (SM0CVAL5CYC)

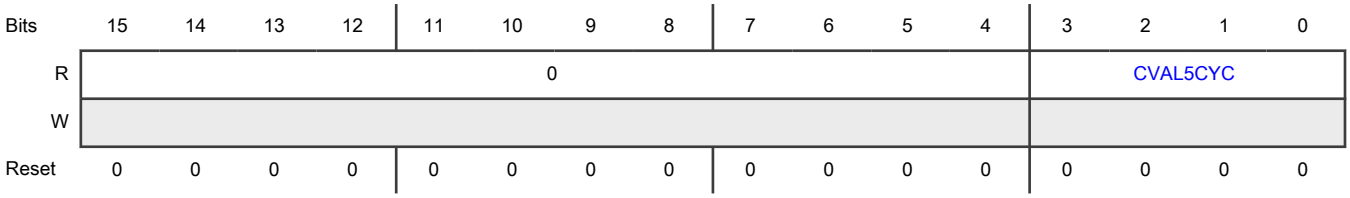
Offset

Register	Offset
SM0CVAL5CYC	56h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL5CYC	Capture Value 5 Cycle This field stores the cycle number corresponding to the value captured in CVAL5. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

67.5.44 Capture PWM_A Input Filter Register (SM0CAPTFILTA - SM3CAPTFILTA)

Offset

Register	Offset
SM0CAPTFILTA	5Ah
SM1CAPTFILTA	BAh
SM2CAPTFILTA	11Ah
SM3CAPTFILTA	17Ah

Function

Input filter considerations include:

- The CAPTA_FILT_PER value should be set such that the sampling period is larger than the period of the expected noise. This way a noise spike will only corrupt one sample. The CAPTA_FILT_CNT value should be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the CAPTA_FILT_CNT+3 power.

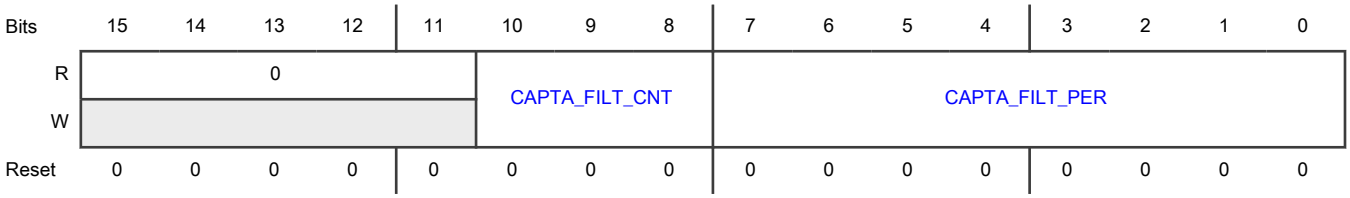
- The values of CAPTA_FILT_PER and CAPTA_FILT_CNT must be traded off against the desire for minimal latency in recognizing input transitions. Turning on the input filter (setting CAPTA_FILT_PER to a non-zero value) introduces a latency of ((CAPTA_FILT_CNT+4) x CAPTA_FILT_PER x IPBus clock period).

NOTE

When the filter is enabled, there is a combinational path to disable the PWM outputs to ensure rapid response to input capture conditions if the PWM module loses its clock. The latency induced by the filter will be seen in the time to set FSTS[FFLAG] and FSTS[FFPIN].

This register is not byte accessible

Diagram



Fields

Field	Function
15-11 —	Reserved
10-8 CAPTA_FILT_CNT	Input Capture Filter Count This field represents the number of consecutive samples that must agree prior to the input filter accepting an input transition. The number of samples is the decimal value of this field plus three: the bit field value of 0-7 represents 3-10 samples, respectively. The value of CAPTA_FILT_CNT affects the input latency.
7-0 CAPTA_FILT_PER	Input Capture Filter Period This field applies universally to all capture inputs. These bits represent the sampling period (in IPBus clock cycles) of the input capture pin input filter. Each input is sampled multiple times at the rate specified by this field. If CAPTA_FILT_PER is 0x00 (default), then the input filter is bypassed. The value of CAPTA_FILT_PER affects the input latency. **NOTE** When changing values for CAPTA_FILT_PER from one non-zero value to another non-zero value, first write a value of zero to clear the filter.

67.5.45 Capture PWM_B Input Filter Register (SM0CAPTFILTB - SM3CAPTFILTB)

Offset

Register	Offset
SM0CAPTFILTB	5Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
SM1CAPTFILTB	BCh
SM2CAPTFILTB	11Ch
SM3CAPTFILTB	17Ch

Function

Input filter considerations include:

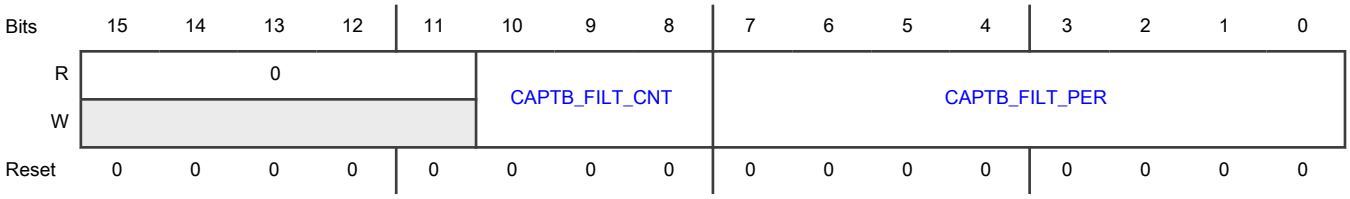
- The CAPTB_FILT_PER value should be set such that the sampling period is larger than the period of the expected noise. This way a noise spike will only corrupt one sample. The CAPTB_FILT_CNT value should be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the CAPTB_FILT_CNT+3 power.
- The values of CAPTB_FILT_PER and CAPTB_FILT_CNT must be traded off against the desire for minimal latency in recognizing input transitions. Turning on the input filter (setting CAPTB_FILT_PER to a non-zero value) introduces a latency of ((CAPTB_FILT_CNT+4) x CAPTB_FILT_PER x IPBus clock period).

NOTE

When the filter is enabled, there is a combinational path to disable the PWM outputs to ensure rapid response to input capture conditions if the PWM module loses its clock. The latency induced by the filter will be seen in the time to set FSTS[FFLAG] and FSTS[FFPIN].

This register is not byte accessible

Diagram



Fields

Field	Function
15-11 —	Reserved
10-8 CAPTB_FILT_CNT	Input Capture Filter Count These bits represent the number of consecutive samples that must agree prior to the input filter accepting an input transition. The number of samples is the decimal value of this field plus three: the bit field value of 0-7 represents 3-10 samples, respectively. The value of CAPTB_FILT_CNT affects the input latency.
7-0	Input Capture Filter Period This field applies universally to all capture inputs.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CAPTB_FILT_PER	These bits represent the sampling period (in IPBus clock cycles) of the input capture pin input filter. Each input is sampled multiple times at the rate specified by this field. If CAPTB_FILT_PER is 0x00 (default), then the input filter is bypassed. The value of CAPTB_FILT_PER affects the input latency. **NOTE** When changing values for CAPTB_FILT_PER from one non-zero value to another non-zero value, first write a value of zero to clear the filter.

67.5.46 Capture PWM_X Input Filter Register (SM0CAPTFILTX - SM3CAPTFILTX)

Offset

Register	Offset
SM0CAPTFILTX	5Eh
SM1CAPTFILTX	BEh
SM2CAPTFILTX	11Eh
SM3CAPTFILTX	17Eh

Function

Input filter considerations include:

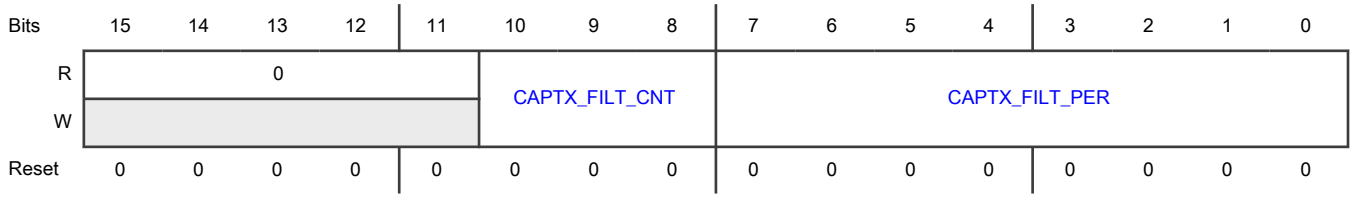
- The CAPTX_FILT_PER value should be set such that the sampling period is larger than the period of the expected noise. This way a noise spike will only corrupt one sample. The CAPTX_FILT_CNT value should be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the CAPTX_FILT_CNT+3 power.
- The values of FILT_PER and CAPTX_FILT_CNT must also be traded off against the desire for minimal latency in recognizing input transitions. Turning on the input filter (setting CAPTX_FILT_PER to a non-zero value) introduces a latency of ((CAPTX_FILT_CNT+4) x CAPTX_FILT_PER x IPBus clock period).

NOTE

When the filter is enabled, there is a combinational path to disable the PWM outputs to ensure rapid response to input capture conditions if the PWM module loses its clock. The latency induced by the filter will be seen in the time to set FSTS[FFLAG] and FSTS[FFPIN].

This register is not byte accessible

Diagram



Fields

Field	Function
15-11 —	Reserved
10-8 CAPTX_FILT_CNT	Input Capture Filter Count These bits represent the number of consecutive samples that must agree prior to the input filter accepting an input transition. The number of samples is the decimal value of this field plus three: the bit field value of 0-7 represents 3-10 samples, respectively. The value of CAPTX_FILT_CNT affects the input latency.
7-0 CAPTX_FILT_PER	Input Capture Filter Period This field applies universally to all capture inputs. These bits represent the sampling period (in IPBus clock cycles) of the input capture pin input filter. Each input is sampled multiple times at the rate specified by this field. If CAPTX_FILT_PER is 0x00 (default), then the input filter is bypassed. The value of CAPTX_FILT_PER affects the input latency. **NOTE** When changing values for CAPTX_FILT_PER from one non-zero value to another non-zero value, first write a value of zero to clear the filter.

67.5.47 Capture Control A Register (SM1CAPTCTRLA)

Offset

Register	Offset
SM1CAPTCTRLA	94h

Function

Contains capture controls for mode A.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA1CNT				CA0CNT				EDGCNTA...	INP_SELA	EDGA1		EDGA0		ONESHOTA	ARMA
W							CFAWM									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 CA1CNT	Capture A1 FIFO Word Count This field reflects the number of words in the Capture A1 FIFO. (FIFO depth is 1.)
12-10 CA0CNT	Capture A0 FIFO Word Count This field reflects the number of words in the Capture A0 FIFO. (FIFO depth is 1.)
9-8 CFAWM	Capture A FIFOs Water Mark This field represents the water mark level for capture A FIFOs. The capture flags, STS[CFA1] and STS[CFA0], are not set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTA_EN	Edge Counter A Enable This field enables the edge counter which counts rising and falling edges on the PWM_A input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELA	Input Select A This field selects between the raw PWM_A input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_A input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLA[EDGA0] and CAPTCTRLA[EDGA1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRA[EDGA0] and/or CAPTCTRLA[EDGA1] fields to enable one or both of the capture registers.
5-4 EDGA1	Edge A 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2	Edge A 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
EDGA0	These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTA	One Shot Mode A This bit selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLA[ARMA] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLA[ARMA] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and CAPTCTRLA[ARMA] is cleared. No further captures are performed until CAPTCTRLA[ARMA] is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and CAPTCTRLA[ARMA] is then cleared.
0 ARMA	Arm A Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLA[EDGAx] is enabled.

67.5.48 Capture Compare A Register (SM1CAPTCOMPA)

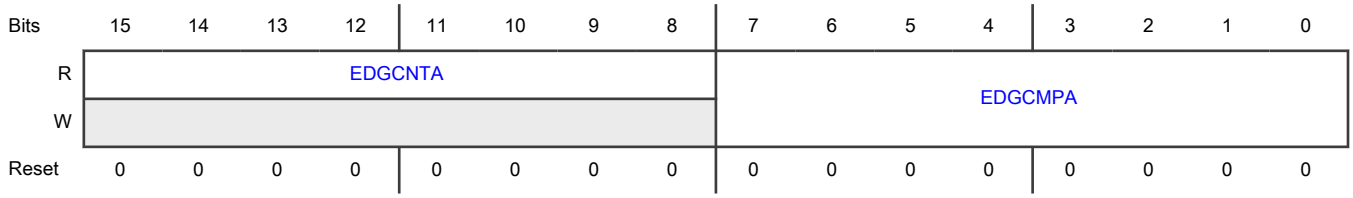
Offset

Register	Offset
SM1CAPTCOMPA	96h

Function

Contains capture and compare values for mode A.

Diagram



Fields

Field	Function
15-8 EDGCNTA	Edge Counter A This field contains the edge counter value for the PWM_A input capture circuitry.
7-0 EDGCMPA	Edge Compare A This field is the compare value associated with the edge counter for the PWM_A input capture circuitry.

67.5.49 Capture Control B Register (SM1CAPTCTRLB)

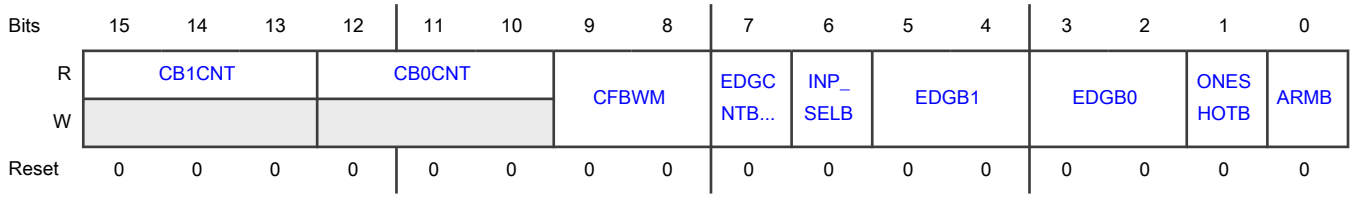
Offset

Register	Offset
SM1CAPTCTRLB	98h

Function

Contains capture controls for mode B.

Diagram



Fields

Field	Function
15-13 CB1CNT	Capture B1 FIFO Word Count This field reflects the number of words in the Capture B1 FIFO. (FIFO depth is 1.)
12-10 CB0CNT	Capture B0 FIFO Word Count This field reflects the number of words in the Capture B0 FIFO. (FIFO depth is 1.)

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 CFBWM	Capture B FIFOs Water Mark This field represents the water mark level for capture B FIFOs. The capture flags, STS[CFB1] and STS[CFB0], will not be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTB_EN	Edge Counter B Enable This field enables the edge counter which counts rising and falling edges on the PWM_B input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELB	Input Select B This field selects between the raw PWM_B input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_B input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLB[EDGB0] and CAPTCTRLB[EDGB1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRB[EDGB0] and/or CAPTCTRLB[EDGB1] fields to enable one or both of the capture registers.
5-4 EDGB1	Edge B 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2 EDGB0	Edge B 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTB	One Shot Mode B This bit selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLB[ARMB] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLB[ARMB] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and CAPTCTRLB[ARMB] is cleared. No further captures are performed until CAPTCTRLB[ARMB] is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and CAPTCTRLB[ARMB] is then cleared.
0 ARMB	Arm B Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLB[EDGBx] is enabled.

67.5.50 Capture Compare B Register (SM1CAPTCOMPB)

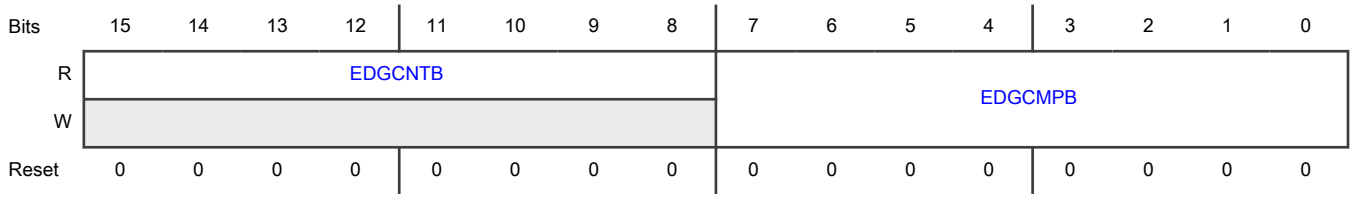
Offset

Register	Offset
SM1CAPTCOMPB	9Ah

Function

Contains capture and compare values for mode B.

Diagram



Fields

Field	Function
15-8 EDGCNTB	Edge Counter B This field contains the edge counter value for the PWM_B input capture circuitry.
7-0 EDGCMPB	Edge Compare B This field is the compare value associated with the edge counter for the PWM_B input capture circuitry.

67.5.51 Capture Control X Register (SM1CAPTCTRLX)

Offset

Register	Offset
SM1CAPTCTRLX	9Ch

Function

Contains capture controls for mode X.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CX1CNT			CX0CNT			CFXWM		EDGCNTX...	INP_SELX	EDGX1		EDGX0		ONESHOTX	ARMX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 CX1CNT	Capture X1 FIFO Word Count This field reflects the number of words in the Capture X1 FIFO. (FIFO depth is 1.)
12-10 CX0CNT	Capture X0 FIFO Word Count This field reflects the number of words in the Capture X0 FIFO. (FIFO depth is 1.)
9-8 CFXWM	Capture X FIFOs Water Mark This field represents the water mark level for capture X FIFOs. The capture flags, STS[CFX1] and STS[CFX0], will not be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTX_EN	Edge Counter X Enable This bit enables the edge counter which counts rising and falling edges on the PWM_X input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELX	Input Select X This bit selects between the raw PWM_X input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_X input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the

Table continues on the next page...

Table continued from the previous page...

Field	Function
	CAPTCTRLX[EDGX0] and CAPTCTRLX[EDGX1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRX[EDGX0] and/or CAPTCTRLX[EDGX1] fields to enable one or both of the capture registers.
5-4 EDGX1	Edge X 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2 EDGX0	Edge X 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTX	One Shot Mode Aux This field selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and the ARMX bit is cleared. No further captures are performed until the ARMX bit is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and the ARMX bit is then cleared.
0 ARMX	Arm X Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLX[EDGXx] is enabled.

67.5.52 Capture Compare X Register (SM1CAPTCOMPX)

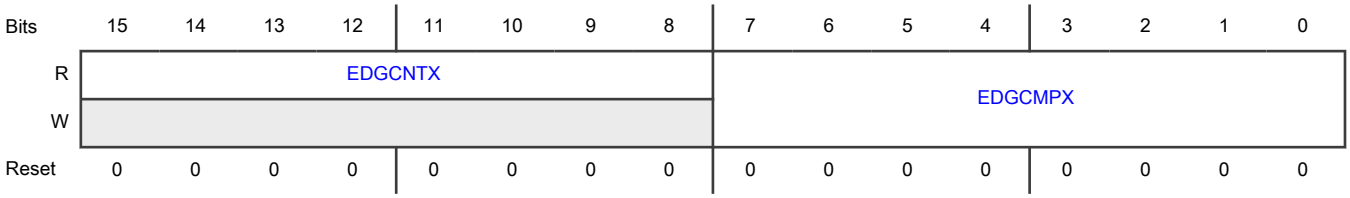
Offset

Register	Offset
SM1CAPTCOMPX	9Eh

Function

Contains capture and control values for mode X.

Diagram



Fields

Field	Function
15-8 EDGCNTX	Edge Counter X This field contains the edge counter value for the PWM_X input capture circuitry.
7-0 EDGCMPX	Edge Compare X This field is the compare value associated with the edge counter for the PWM_X input capture circuitry.

67.5.53 Capture Value 0 Register (SM1CVAL0)

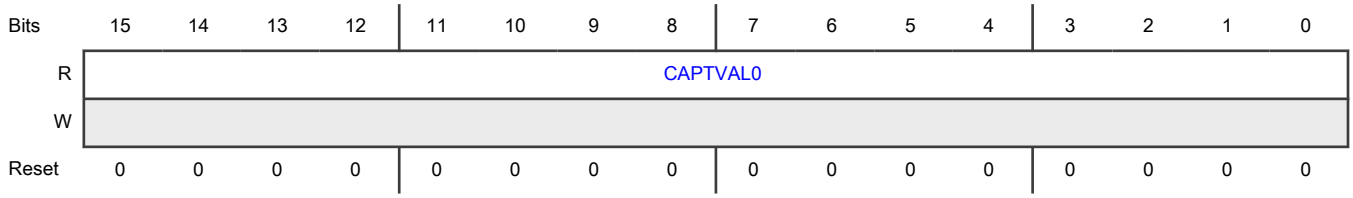
Offset

Register	Offset
SM1CVAL0	A0h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL0	Capture Value 0 This field stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLX[EDGX0]. Each capture increases the value of CAPTCTRLX[CX0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX0CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.54 Capture Value 0 Cycle Register (SM1CVAL0CYC)

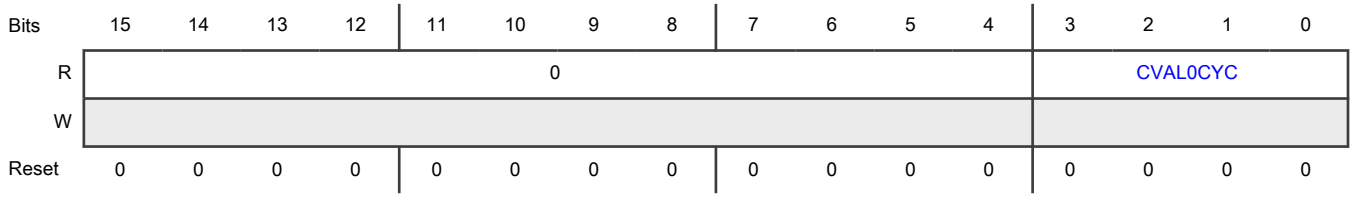
Offset

Register	Offset
SM1CVAL0CYC	A2h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL0CYC	Capture Value 0 Cycle This field stores the cycle number corresponding to the value captured in CVAL0. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

67.5.55 Capture Value 1 Register (SM1CVAL1)

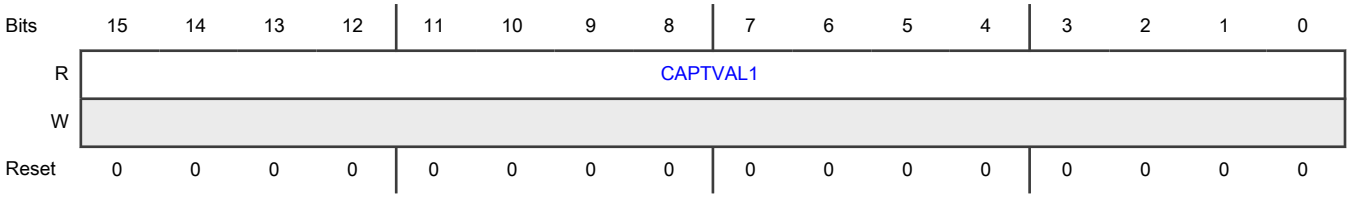
Offset

Register	Offset
SM1CVAL1	A4h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL1	Capture Value 1 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLX[EDGX1]. Each capture increases the value of CAPTCTRLX[CX1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX1CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.56 Capture Value 1 Cycle Register (SM1CVAL1CYC)

Offset

Register	Offset
SM1CVAL1CYC	A6h

Function

Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												CVAL1CYC			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-4 —	Reserved
3-0 CVAL1CYC	Capture Value 1 Cycle This field stores the cycle number corresponding to the value captured in CVAL1. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle. Resets to 0 at POR or hard reset.

67.5.57 Capture Value 2 Register (SM1CVAL2)

Offset

Register	Offset
SM1CVAL2	A8h

Function

Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CAPTVAL2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 CAPTVAL2	Capture Value 2 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLA[EDGA0]. Each capture increases the value of CAPTCTRLA[CA0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLA[CA0CNT] by 1 until 0 is reached. This field is not byte accessible.

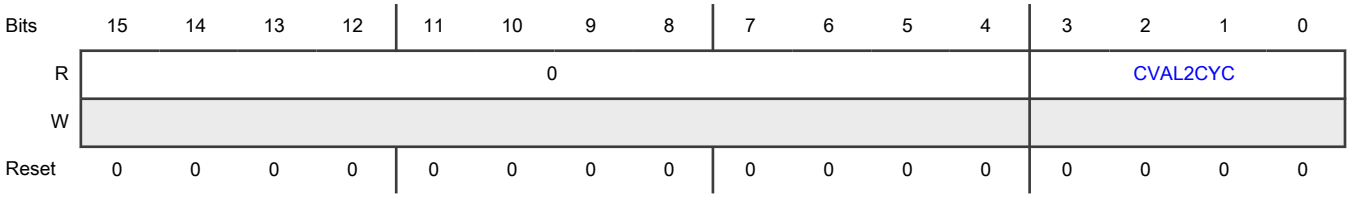
67.5.58 Capture Value 2 Cycle Register (SM1CVAL2CYC)

Offset

Register	Offset
SM1CVAL2CYC	AAh

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL2CYC	Capture Value 2 Cycle This field stores the cycle number corresponding to the value captured in CVAL2. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

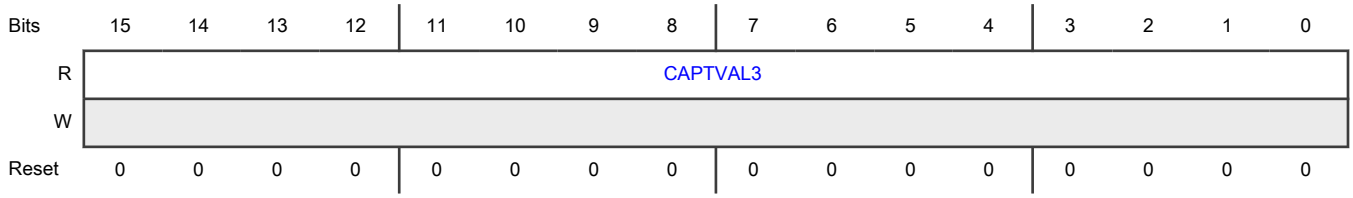
67.5.59 Capture Value 3 Register (SM1CVAL3)

Offset

Register	Offset
SM1CVAL3	ACh

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL3	Capture Value 3 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLA[EDGA1]. Each capture increases the value of CAPTCTRLA[CA1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLA[CA1CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.60 Capture Value 3 Cycle Register (SM1CVAL3CYC)

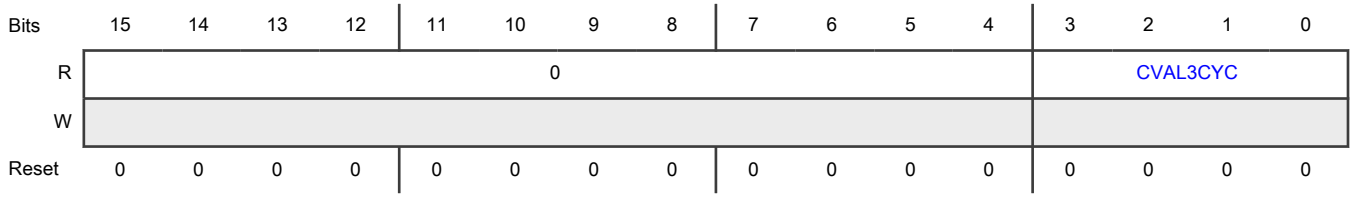
Offset

Register	Offset
SM1CVAL3CYC	AEh

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL3CYC	Capture Value 3 Cycle This field stores the cycle number corresponding to the value captured in CVAL3. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

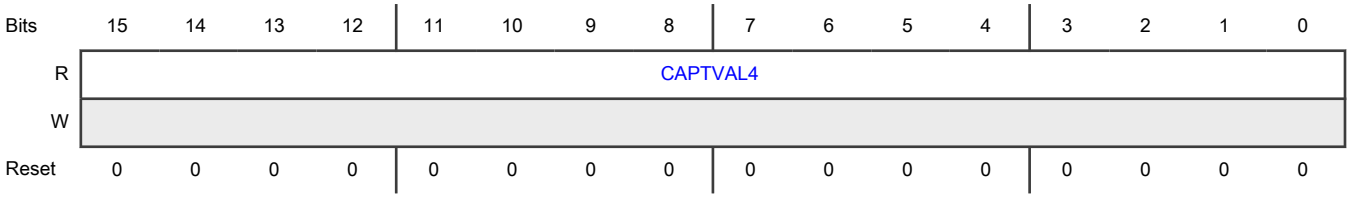
67.5.61 Capture Value 4 Register (SM1CVAL4)

Offset

Register	Offset
SM1CVAL4	B0h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL4	Capture Value 4 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLB[EDGB0]. Each capture increases the value of CAPTCTRLB[CB0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLB[CB0CNT] by 1 until 0 is reached. This field is not byte accessible.

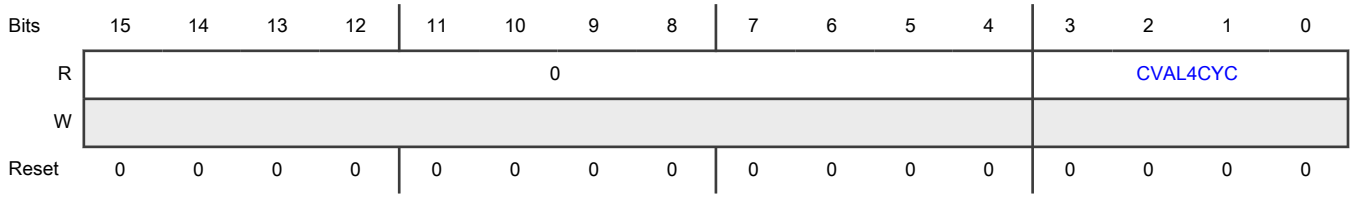
67.5.62 Capture Value 4 Cycle Register (SM1CVAL4CYC)

Offset

Register	Offset
SM1CVAL4CYC	B2h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL4CYC	Capture Value 4 Cycle This field stores the cycle number corresponding to the value captured in CVAL4. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

67.5.63 Capture Value 5 Register (SM1CVAL5)

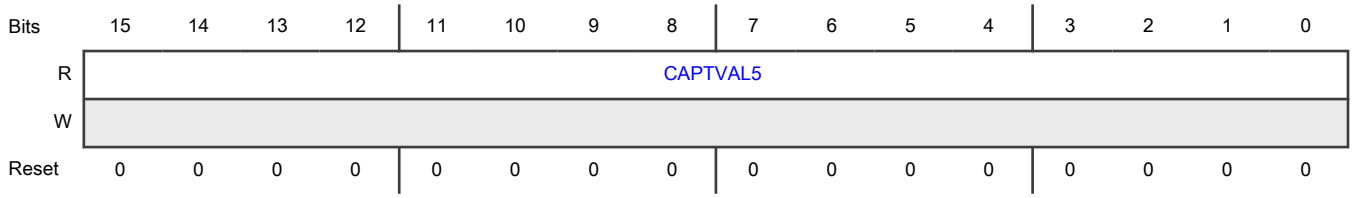
Offset

Register	Offset
SM1CVAL5	B4h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL5	Capture Value 5 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLB[EDGB1]. Each capture increases the value of CAPTCTRLB[CB1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLB[CB1CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.64 Capture Value 5 Cycle Register (SM1CVAL5CYC)

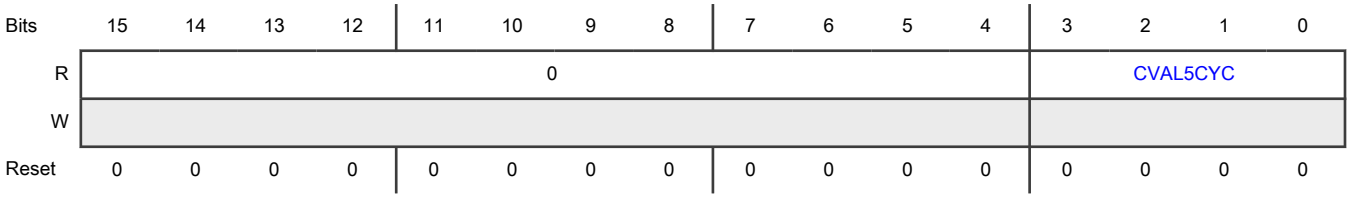
Offset

Register	Offset
SM1CVAL5CYC	B6h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL5CYC	Capture Value 5 Cycle This field stores the cycle number corresponding to the value captured in CVAL5. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

67.5.65 Phase Delay Register (SM1PHASEDLY - SM3PHASEDLY)

Offset

Register	Offset
SM1PHASEDLY	B8h
SM2PHASEDLY	118h
SM3PHASEDLY	178h

Function

The 16-bit unsigned value in this buffered register defines the delay from the master sync signal of submodule 0 to the time that this submodule recognizes the master sync in PWM clock periods. CTRL2[INIT_SEL] must be set to 10b to select the master sync signal as the source for initialization when using this register. Setting this register with a non-zero value and using the master sync signal as the initialization source, allows the output of this submodule to be a fixed number of cycles delayed from submodule 0. For PWM operation, the buffered contents of this register are updated at the start of every PWM cycle. This field is not byte accessible.

NOTE

The PHASEDLY register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. This register cannot be written when MCTRL[LDOK] is set. Reading PHASEDLY reads the value in a buffer and not necessarily the value that the PWM generator is currently using. Also, the value of this register should not be set to a value larger than the period defined in submodule 0.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PHASEDLY															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 PHASEDLY	Initial Count Register Bits

67.5.66 Capture Control A Register (SM2CAPTCTRLA)

Offset

Register	Offset
SM2CAPTCTRLA	F4h

Function

Contains capture controls for mode A.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA1CNT			CA0CNT			CFAWM		EDGCNTA...	INP_SELA	EDGA1		EDGA0		ONESHOTA	ARMA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 CA1CNT	Capture A1 FIFO Word Count This field reflects the number of words in the Capture A1 FIFO. (FIFO depth is 1.)

Table continues on the next page...

Table continued from the previous page...

Field	Function
12-10 CA0CNT	Capture A0 FIFO Word Count This field reflects the number of words in the Capture A0 FIFO. (FIFO depth is 1.)
9-8 CFAWM	Capture A FIFOs Water Mark This field represents the water mark level for capture A FIFOs. The capture flags, STS[CFA1] and STS[CFA0], are not set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTA_EN	Edge Counter A Enable This field enables the edge counter which counts rising and falling edges on the PWM_A input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELA	Input Select A This field selects between the raw PWM_A input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_A input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLA[EDGA0] and CAPTCTRLA[EDGA1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRA[EDGA0] and/or CAPTCTRLA[EDGA1] fields to enable one or both of the capture registers.
5-4 EDGA1	Edge A 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2 EDGA0	Edge A 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTA	One Shot Mode A This bit selects between free running and one shot mode for the input capture circuitry.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLA[ARMA] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLA[ARMA] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and CAPTCTRLA[ARMA] is cleared. No further captures are performed until CAPTCTRLA[ARMA] is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and CAPTCTRLA[ARMA] is then cleared.
0 ARMA	Arm A Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLA[EDGAx] is enabled.

67.5.67 Capture Compare A Register (SM2CAPTCOMPA)

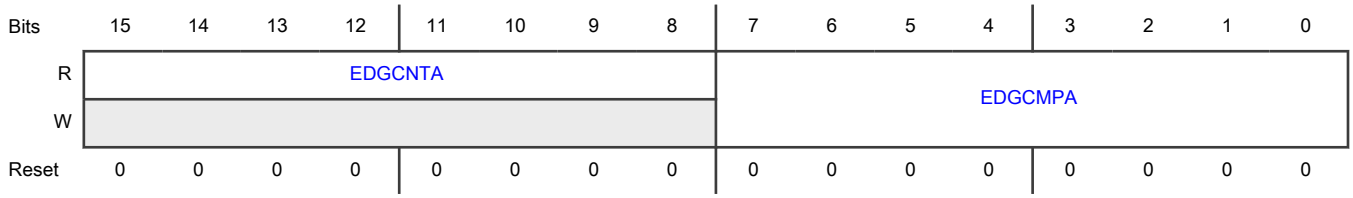
Offset

Register	Offset
SM2CAPTCOMPA	F6h

Function

Contains capture and compare values for mode A.

Diagram



Fields

Field	Function
15-8	Edge Counter A

Table continues on the next page...

Table continued from the previous page...

Field	Function
EDGCNTA	This field contains the edge counter value for the PWM_A input capture circuitry.
7-0 EDGCMPA	Edge Compare A This field is the compare value associated with the edge counter for the PWM_A input capture circuitry.

67.5.68 Capture Control B Register (SM2CAPTCTRLB)

Offset

Register	Offset
SM2CAPTCTRLB	F8h

Function

Contains capture controls for mode B.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CB1CNT			CB0CNT			CFBWM		EDGCNTB...	INP_SELB	EDGB1		EDGB0		ONESHOTB	ARMB
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 CB1CNT	Capture B1 FIFO Word Count This field reflects the number of words in the Capture B1 FIFO. (FIFO depth is 1.)
12-10 CB0CNT	Capture B0 FIFO Word Count This field reflects the number of words in the Capture B0 FIFO. (FIFO depth is 1.)
9-8 CFBWM	Capture B FIFOs Water Mark This field represents the water mark level for capture B FIFOs. The capture flags, STS[CFB1] and STS[CFB0], will not be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTB_EN	Edge Counter B Enable This field enables the edge counter which counts rising and falling edges on the PWM_B input signal. 0b - Edge counter disabled and held in reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Edge counter enabled
6 INP_SELB	Input Select B This field selects between the raw PWM_B input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_B input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLB[EDGB0] and CAPTCTRLB[EDGB1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRB[EDGB0] and/or CAPTCTRLB[EDGB1] fields to enable one or both of the capture registers.
5-4 EDGB1	Edge B 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2 EDGB0	Edge B 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTB	One Shot Mode B This bit selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLB[ARMB] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLB[ARMB] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and CAPTCTRLB[ARMB] is cleared. No further captures are performed until CAPTCTRLB[ARMB] is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and CAPTCTRLB[ARMB] is then cleared.
0	Arm B

Table continues on the next page...

Table continued from the previous page...

Field	Function
ARMB	Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLB[EDGBx] is enabled.

67.5.69 Capture Compare B Register (SM2CAPTCOMPB)

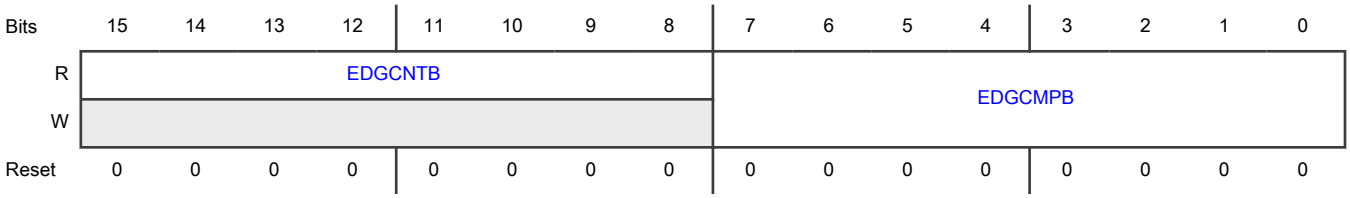
Offset

Register	Offset
SM2CAPTCOMPB	FAh

Function

Contains capture and compare values for mode B.

Diagram



Fields

Field	Function
15-8 EDGCNTB	Edge Counter B This field contains the edge counter value for the PWM_B input capture circuitry.
7-0 EDGCMPB	Edge Compare B This field is the compare value associated with the edge counter for the PWM_B input capture circuitry.

67.5.70 Capture Control X Register (SM2CAPTCTRLX)

Offset

Register	Offset
SM2CAPTCTRLX	FCh

Function

Contains capture controls for mode X.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CX1CNT			CX0CNT			CFXWM		EDGCNTX...	INP_SELX	EDGX1		EDGX0		ONESHOTX	ARMX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 CX1CNT	Capture X1 FIFO Word Count This field reflects the number of words in the Capture X1 FIFO. (FIFO depth is 1.)
12-10 CX0CNT	Capture X0 FIFO Word Count This field reflects the number of words in the Capture X0 FIFO. (FIFO depth is 1.)
9-8 CFXWM	Capture X FIFOs Water Mark This field represents the water mark level for capture X FIFOs. The capture flags, STS[CFX1] and STS[CFX0], will not be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTX_EN	Edge Counter X Enable This bit enables the edge counter which counts rising and falling edges on the PWM_X input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELX	Input Select X This bit selects between the raw PWM_X input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_X input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLX[EDGX0] and CAPTCTRLX[EDGX1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRX[EDGX0] and/or CAPTCTRLX[EDGX1] fields to enable one or both of the capture registers.
5-4 EDGX1	Edge X 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Capture rising edges 11b - Capture any edge
3-2 EDGX0	Edge X 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTX	One Shot Mode Aux This field selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and the ARMX bit is cleared. No further captures are performed until the ARMX bit is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and the ARMX bit is then cleared.
0 ARMX	Arm X Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLX[EDGXx] is enabled.

67.5.71 Capture Compare X Register (SM2CAPTCOMPX)

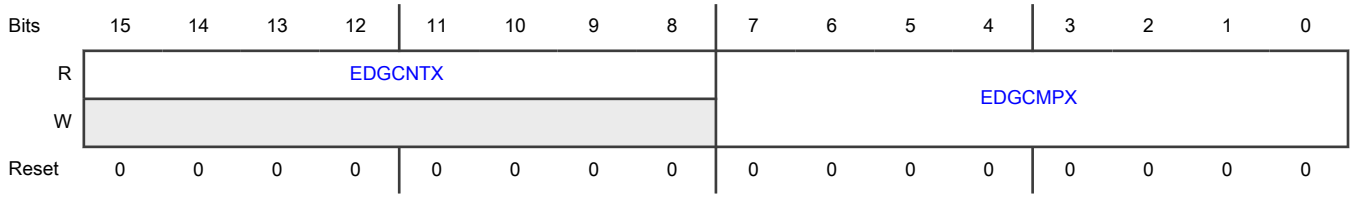
Offset

Register	Offset
SM2CAPTCOMPX	FEh

Function

Contains capture and control values for mode X.

Diagram



Fields

Field	Function
15-8 EDGCNTX	Edge Counter X This field contains the edge counter value for the PWM_X input capture circuitry.
7-0 EDGCMPX	Edge Compare X This field is the compare value associated with the edge counter for the PWM_X input capture circuitry.

67.5.72 Capture Value 0 Register (SM2CVAL0)

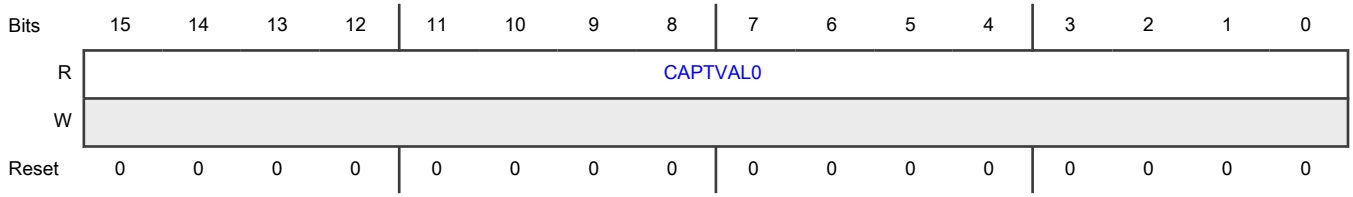
Offset

Register	Offset
SM2CVAL0	100h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL0	Capture Value 0 This field stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLX[EDGX0]. Each capture increases the value of CAPTCTRLX[CX0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX0CNT] by 1 until 0 is reached. This field is not byte accessible.

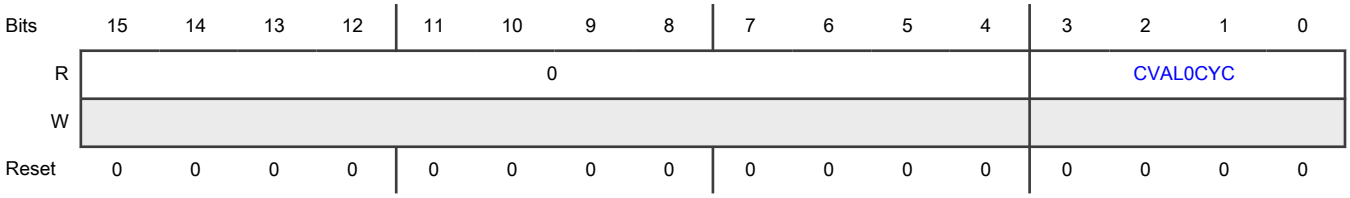
67.5.73 Capture Value 0 Cycle Register (SM2CVAL0CYC)

Offset

Register	Offset
SM2CVAL0CYC	102h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL0CYC	Capture Value 0 Cycle This field stores the cycle number corresponding to the value captured in CVAL0. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

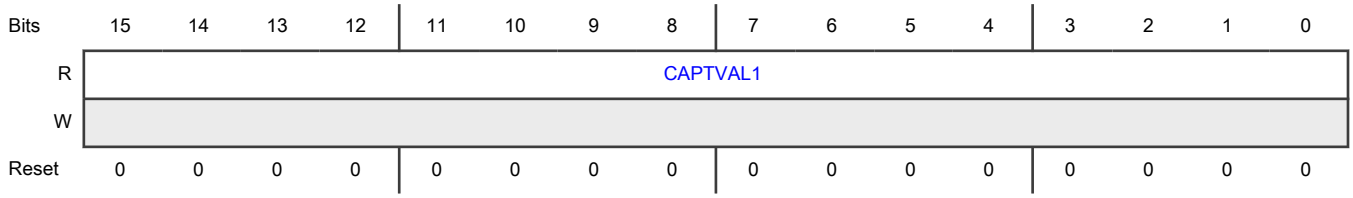
67.5.74 Capture Value 1 Register (SM2CVAL1)

Offset

Register	Offset
SM2CVAL1	104h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL1	Capture Value 1 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLX[EDGX1]. Each capture increases the value of CAPTCTRLX[CX1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX1CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.75 Capture Value 1 Cycle Register (SM2CVAL1CYC)

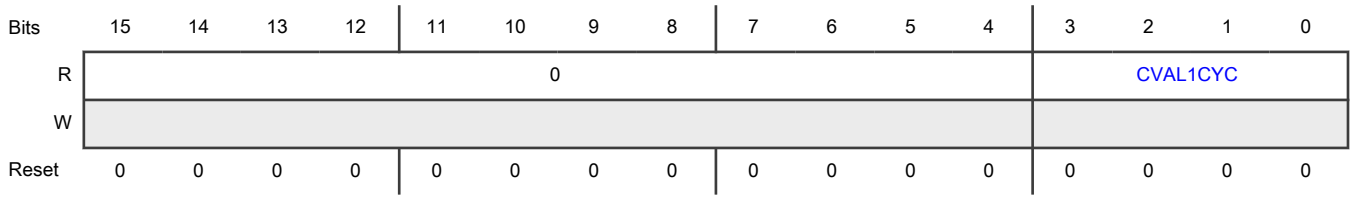
Offset

Register	Offset
SM2CVAL1CYC	106h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL1CYC	Capture Value 1 Cycle This field stores the cycle number corresponding to the value captured in CVAL1. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle. Resets to 0 at POR or hard reset.

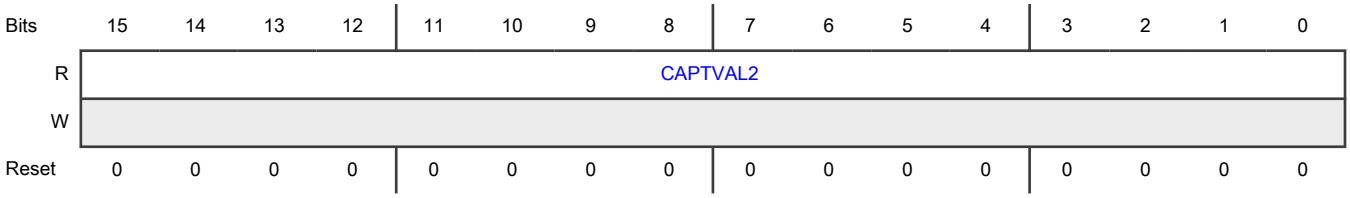
67.5.76 Capture Value 2 Register (SM2CVAL2)

Offset

Register	Offset
SM2CVAL2	108h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL2	Capture Value 2 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLA[EDGA0]. Each capture increases the value of CAPTCTRLA[CA0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLA[CA0CNT] by 1 until 0 is reached. This field is not byte accessible.

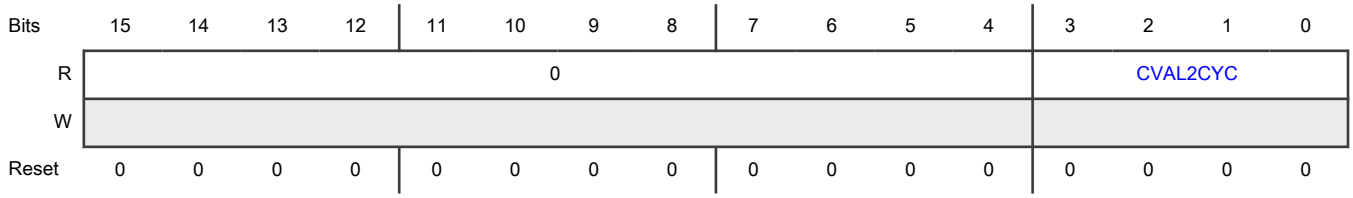
67.5.77 Capture Value 2 Cycle Register (SM2CVAL2CYC)

Offset

Register	Offset
SM2CVAL2CYC	10Ah

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL2CYC	Capture Value 2 Cycle This field stores the cycle number corresponding to the value captured in CVAL2. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

67.5.78 Capture Value 3 Register (SM2CVAL3)

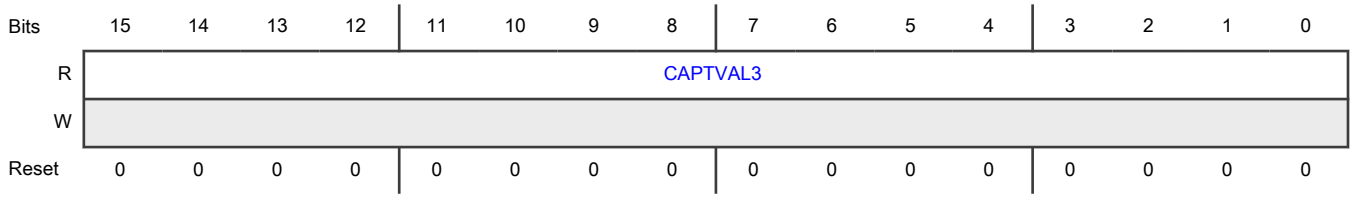
Offset

Register	Offset
SM2CVAL3	10Ch

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL3	Capture Value 3 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLA[EDGA1]. Each capture increases the value of CAPTCTRLA[CA1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLA[CA1CNT] by 1 until 0 is reached. This field is not byte accessible.

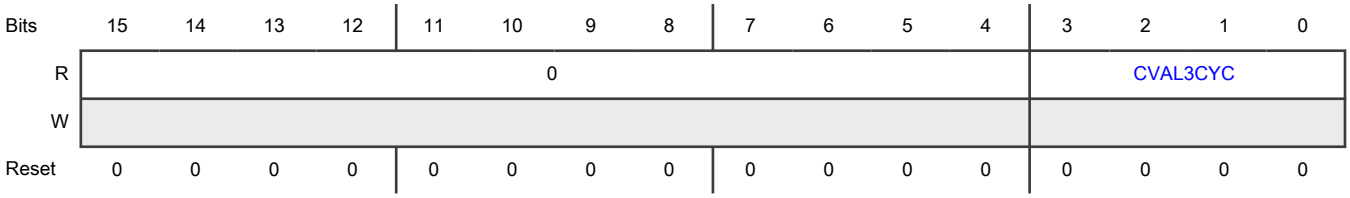
67.5.79 Capture Value 3 Cycle Register (SM2CVAL3CYC)

Offset

Register	Offset
SM2CVAL3CYC	10Eh

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL3CYC	Capture Value 3 Cycle This field stores the cycle number corresponding to the value captured in CVAL3. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

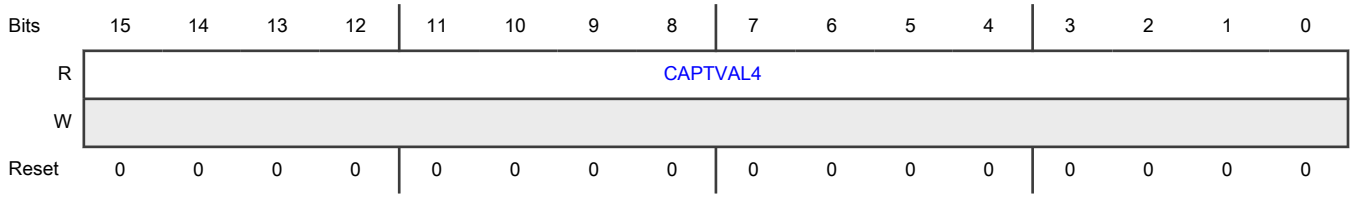
67.5.80 Capture Value 4 Register (SM2CVAL4)

Offset

Register	Offset
SM2CVAL4	110h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL4	Capture Value 4 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLB[EDGB0]. Each capture increases the value of CAPTCTRLB[CB0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLB[CB0CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.81 Capture Value 4 Cycle Register (SM2CVAL4CYC)

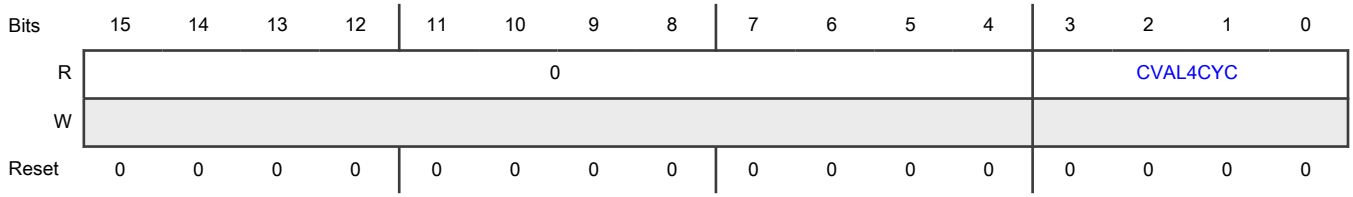
Offset

Register	Offset
SM2CVAL4CYC	112h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL4CYC	Capture Value 4 Cycle This field stores the cycle number corresponding to the value captured in CVAL4. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

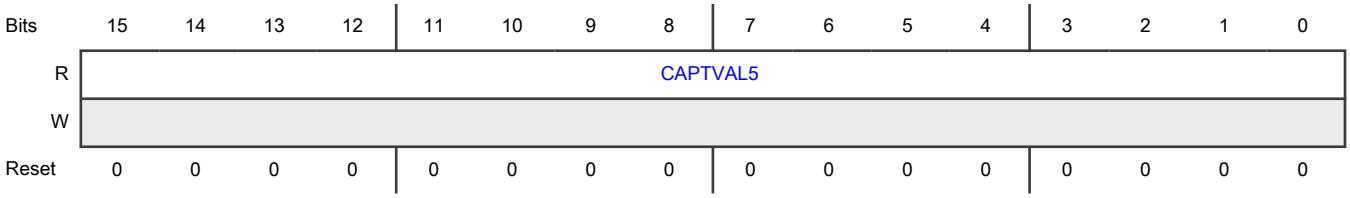
67.5.82 Capture Value 5 Register (SM2CVAL5)

Offset

Register	Offset
SM2CVAL5	114h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL5	Capture Value 5 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLB[EDGB1]. Each capture increases the value of CAPTCTRLB[CB1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLB[CB1CNT] by 1 until 0 is reached. This field is not byte accessible.

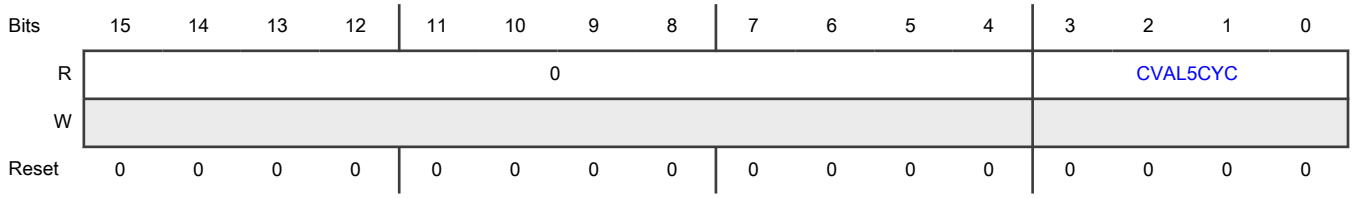
67.5.83 Capture Value 5 Cycle Register (SM2CVAL5CYC)

Offset

Register	Offset
SM2CVAL5CYC	116h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL5CYC	Capture Value 5 Cycle This field stores the cycle number corresponding to the value captured in CVAL5. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

67.5.84 Capture Control A Register (SM3CAPTCTRLA)

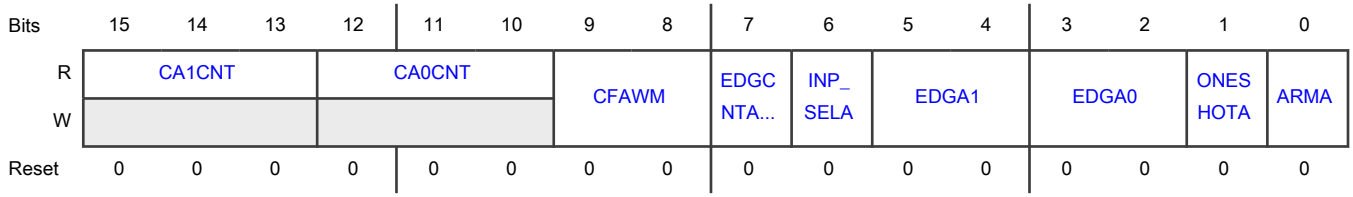
Offset

Register	Offset
SM3CAPTCTRLA	154h

Function

Contains capture controls for mode A.

Diagram



Fields

Field	Function
15-13 CA1CNT	Capture A1 FIFO Word Count This field reflects the number of words in the Capture A1 FIFO. (FIFO depth is 1.)
12-10 CA0CNT	Capture A0 FIFO Word Count This field reflects the number of words in the Capture A0 FIFO. (FIFO depth is 1.)

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 CFAWM	Capture A FIFOs Water Mark This field represents the water mark level for capture A FIFOs. The capture flags, STS[CFA1] and STS[CFA0], are not set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTA_EN	Edge Counter A Enable This field enables the edge counter which counts rising and falling edges on the PWM_A input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELA	Input Select A This field selects between the raw PWM_A input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_A input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLA[EDGA0] and CAPTCTRLA[EDGA1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRA[EDGA0] and/or CAPTCTRLA[EDGA1] fields to enable one or both of the capture registers.
5-4 EDGA1	Edge A 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2 EDGA0	Edge A 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTA	One Shot Mode A This bit selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLA[ARMA] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLA[ARMA] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and CAPTCTRLA[ARMA] is cleared. No further captures are performed until CAPTCTRLA[ARMA] is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and CAPTCTRLA[ARMA] is then cleared.
0 ARMA	Arm A Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLA[EDGAx] is enabled.

67.5.85 Capture Compare A Register (SM3CAPTCOMPA)

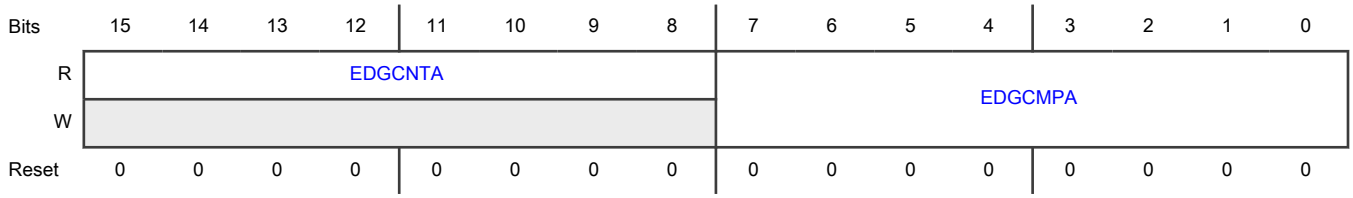
Offset

Register	Offset
SM3CAPTCOMPA	156h

Function

Contains capture and compare values for mode A.

Diagram



Fields

Field	Function
15-8 EDGCNTA	Edge Counter A This field contains the edge counter value for the PWM_A input capture circuitry.
7-0 EDGCMPA	Edge Compare A This field is the compare value associated with the edge counter for the PWM_A input capture circuitry.

67.5.86 Capture Control B Register (SM3CAPTCTRLB)

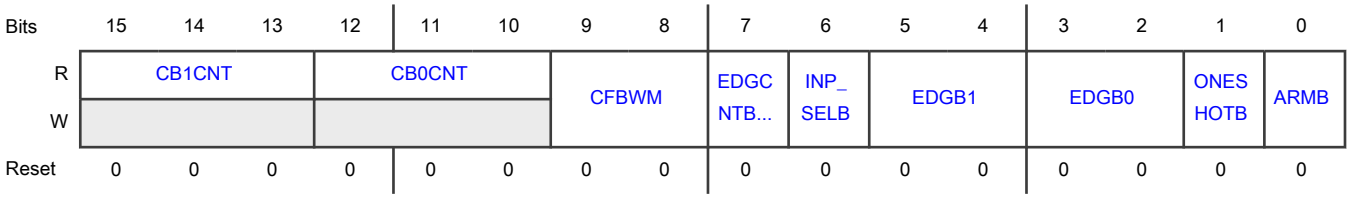
Offset

Register	Offset
SM3CAPTCTRLB	158h

Function

Contains capture controls for mode B.

Diagram



Fields

Field	Function
15-13 CB1CNT	Capture B1 FIFO Word Count This field reflects the number of words in the Capture B1 FIFO. (FIFO depth is 1.)
12-10 CB0CNT	Capture B0 FIFO Word Count This field reflects the number of words in the Capture B0 FIFO. (FIFO depth is 1.)
9-8 CFBWM	Capture B FIFOs Water Mark This field represents the water mark level for capture B FIFOs. The capture flags, STS[CFB1] and STS[CFB0], will not be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTB_EN	Edge Counter B Enable This field enables the edge counter which counts rising and falling edges on the PWM_B input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELB	Input Select B This field selects between the raw PWM_B input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_B input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the

Table continues on the next page...

Table continued from the previous page...

Field	Function
	CAPTCTRLB[EDGB0] and CAPTCTRLB[EDGB1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRB[EDGB0] and/or CAPTCTRLB[EDGB1] fields to enable one or both of the capture registers.
5-4 EDGB1	Edge B 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2 EDGB0	Edge B 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTB	One Shot Mode B This bit selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLB[ARMB] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLB[ARMB] is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and CAPTCTRLB[ARMB] is cleared. No further captures are performed until CAPTCTRLB[ARMB] is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and CAPTCTRLB[ARMB] is then cleared.
0 ARMB	Arm B Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLB[EDGBx] is enabled.

67.5.87 Capture Compare B Register (SM3CAPTCOMPB)

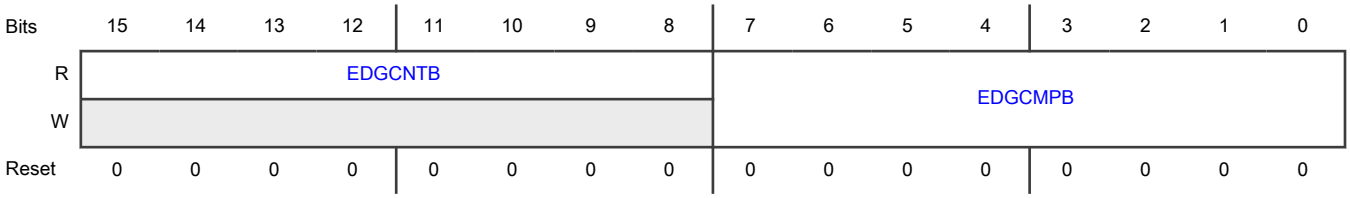
Offset

Register	Offset
SM3CAPTCOMPB	15Ah

Function

Contains capture and compare values for mode B.

Diagram



Fields

Field	Function
15-8 EDGCNTB	Edge Counter B This field contains the edge counter value for the PWM_B input capture circuitry.
7-0 EDGCMPB	Edge Compare B This field is the compare value associated with the edge counter for the PWM_B input capture circuitry.

67.5.88 Capture Control X Register (SM3CAPTCTRLX)

Offset

Register	Offset
SM3CAPTCTRLX	15Ch

Function

Contains capture controls for mode X.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CX1CNT				CX0CNT				EDGCNTX...	INP_SELX	EDGX1		EDGX0		ONESHOTX	ARMX
W							CFXWM									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 CX1CNT	Capture X1 FIFO Word Count This field reflects the number of words in the Capture X1 FIFO. (FIFO depth is 1.)
12-10 CX0CNT	Capture X0 FIFO Word Count This field reflects the number of words in the Capture X0 FIFO. (FIFO depth is 1.)
9-8 CFXWM	Capture X FIFOs Water Mark This field represents the water mark level for capture X FIFOs. The capture flags, STS[CFX1] and STS[CFX0], will not be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTX_EN	Edge Counter X Enable This bit enables the edge counter which counts rising and falling edges on the PWM_X input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELX	Input Select X This bit selects between the raw PWM_X input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_X input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLX[EDGX0] and CAPTCTRLX[EDGX1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRX[EDGX0] and/or CAPTCTRLX[EDGX1] fields to enable one or both of the capture registers.
5-4 EDGX1	Edge X 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2	Edge X 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
EDGX0	These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTX	One Shot Mode Aux This field selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and the ARMX bit is cleared. No further captures are performed until the ARMX bit is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and the ARMX bit is then cleared.
0 ARMX	Arm X Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLX[EDGXx] is enabled.

67.5.89 Capture Compare X Register (SM3CAPTCOMPX)

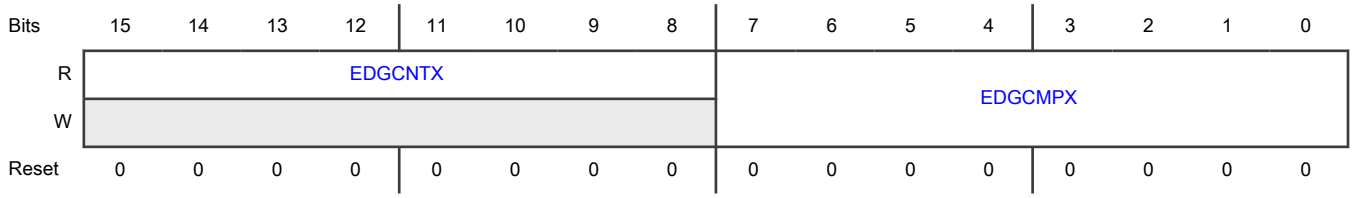
Offset

Register	Offset
SM3CAPTCOMPX	15Eh

Function

Contains capture and control values for mode X.

Diagram



Fields

Field	Function
15-8 EDGCNTX	Edge Counter X This field contains the edge counter value for the PWM_X input capture circuitry.
7-0 EDGCMPX	Edge Compare X This field is the compare value associated with the edge counter for the PWM_X input capture circuitry.

67.5.90 Capture Value 0 Register (SM3CVAL0)

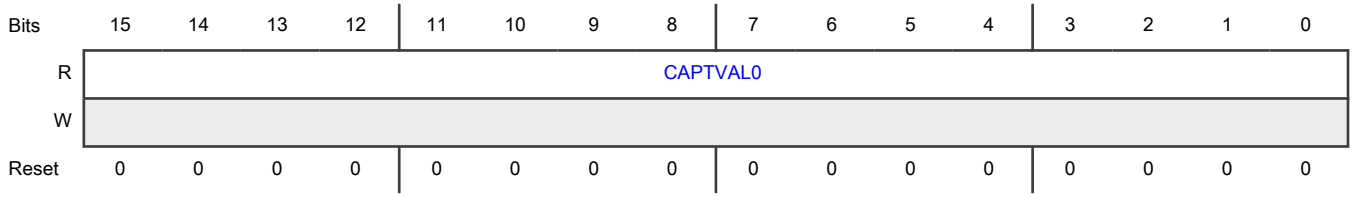
Offset

Register	Offset
SM3CVAL0	160h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL0	Capture Value 0 This field stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLX[EDGX0]. Each capture increases the value of CAPTCTRLX[CX0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX0CNT] by 1 until 0 is reached. This field is not byte accessible.

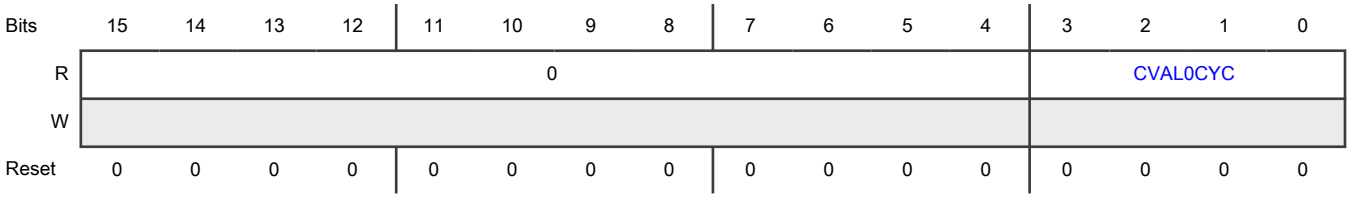
67.5.91 Capture Value 0 Cycle Register (SM3CVAL0CYC)

Offset

Register	Offset
SM3CVAL0CYC	162h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL0CYC	Capture Value 0 Cycle This field stores the cycle number corresponding to the value captured in CVAL0. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

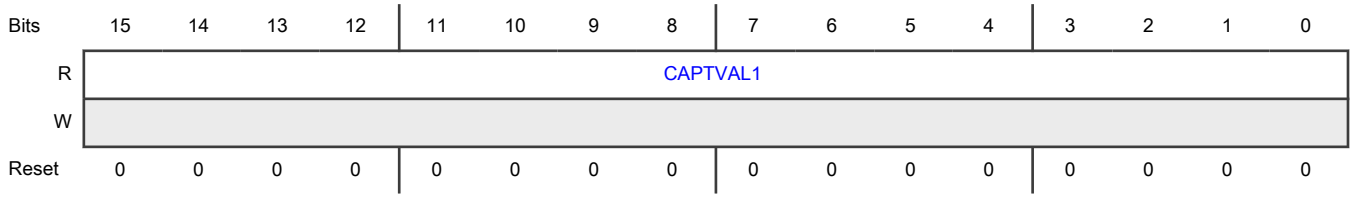
67.5.92 Capture Value 1 Register (SM3CVAL1)

Offset

Register	Offset
SM3CVAL1	164h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL1	Capture Value 1 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLX[EDGX1]. Each capture increases the value of CAPTCTRLX[CX1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX1CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.93 Capture Value 1 Cycle Register (SM3CVAL1CYC)

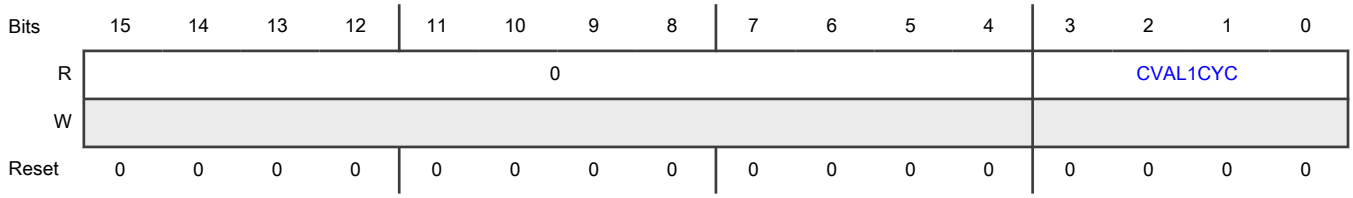
Offset

Register	Offset
SM3CVAL1CYC	166h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL1CYC	Capture Value 1 Cycle This field stores the cycle number corresponding to the value captured in CVAL1. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle. Resets to 0 at POR or hard reset.

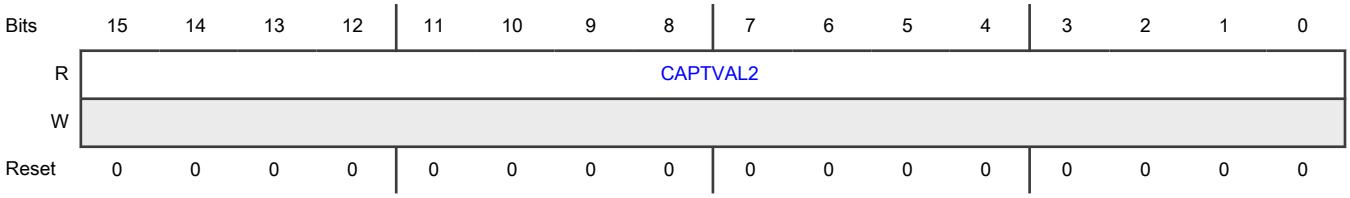
67.5.94 Capture Value 2 Register (SM3CVAL2)

Offset

Register	Offset
SM3CVAL2	168h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL2	Capture Value 2 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLA[EDGA0]. Each capture increases the value of CAPTCTRLA[CA0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLA[CA0CNT] by 1 until 0 is reached. This field is not byte accessible.

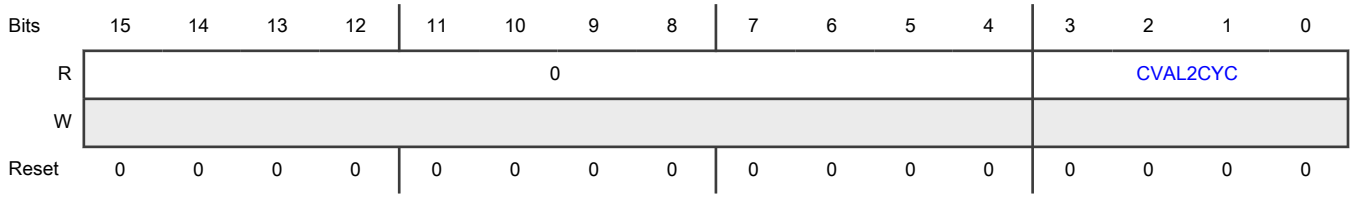
67.5.95 Capture Value 2 Cycle Register (SM3CVAL2CYC)

Offset

Register	Offset
SM3CVAL2CYC	16Ah

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL2CYC	Capture Value 2 Cycle This field stores the cycle number corresponding to the value captured in CVAL2. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

67.5.96 Capture Value 3 Register (SM3CVAL3)

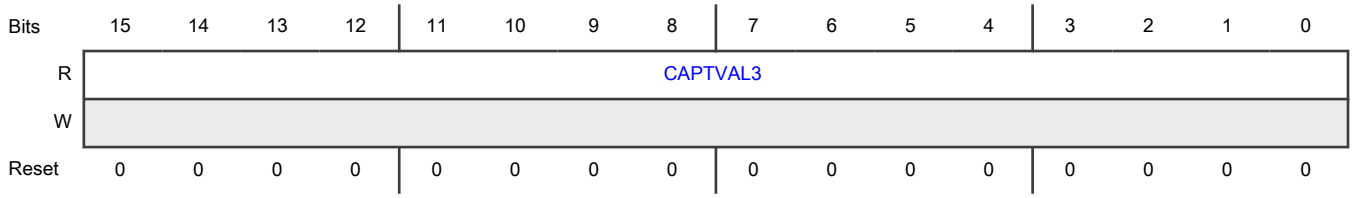
Offset

Register	Offset
SM3CVAL3	16Ch

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL3	Capture Value 3 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLA[EDGA1]. Each capture increases the value of CAPTCTRLA[CA1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLA[CA1CNT] by 1 until 0 is reached. This field is not byte accessible.

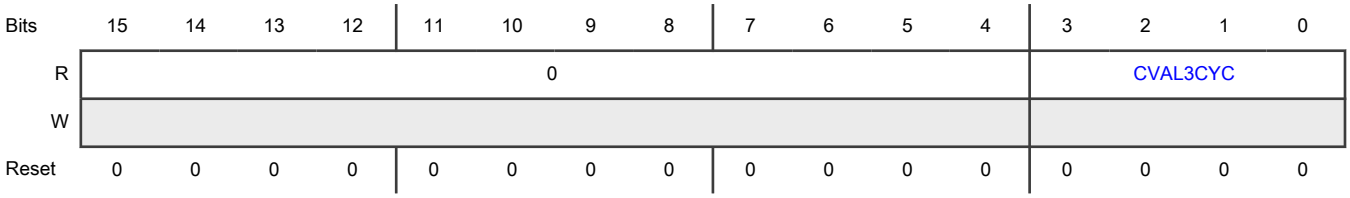
67.5.97 Capture Value 3 Cycle Register (SM3CVAL3CYC)

Offset

Register	Offset
SM3CVAL3CYC	16Eh

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL3CYC	Capture Value 3 Cycle This field stores the cycle number corresponding to the value captured in CVAL3. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

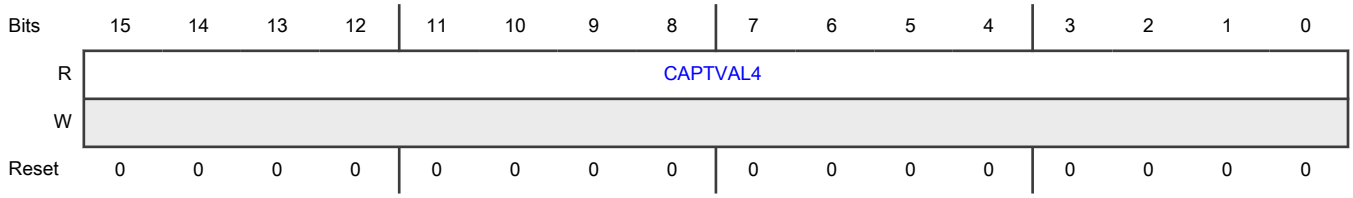
67.5.98 Capture Value 4 Register (SM3CVAL4)

Offset

Register	Offset
SM3CVAL4	170h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL4	Capture Value 4 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLB[EDGB0]. Each capture increases the value of CAPTCTRLB[CB0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLB[CB0CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.99 Capture Value 4 Cycle Register (SM3CVAL4CYC)

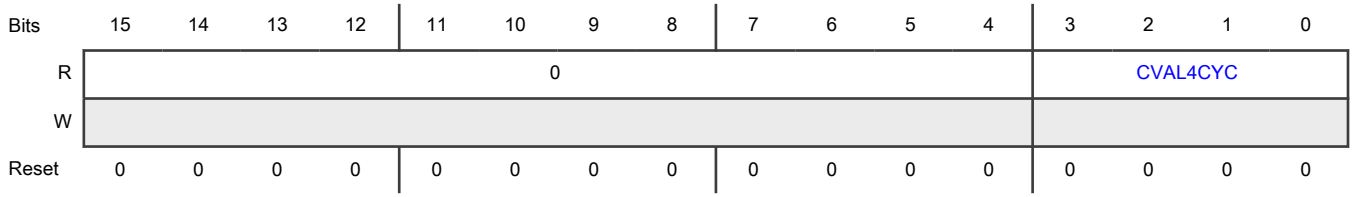
Offset

Register	Offset
SM3CVAL4CYC	172h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL4CYC	Capture Value 4 Cycle This field stores the cycle number corresponding to the value captured in CVAL4. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

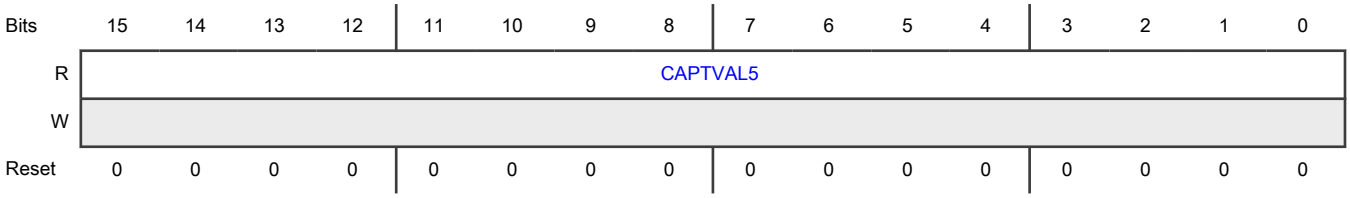
67.5.100 Capture Value 5 Register (SM3CVAL5)

Offset

Register	Offset
SM3CVAL5	174h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL5	Capture Value 5 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLB[EDGB1]. Each capture increases the value of CAPTCTRLB[CB1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLB[CB1CNT] by 1 until 0 is reached. This field is not byte accessible.

67.5.101 Capture Value 5 Cycle Register (SM3CVAL5CYC)

Offset

Register	Offset
SM3CVAL5CYC	176h

Function
Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												CVAL5CYC			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-4 —	Reserved
3-0 CVAL5CYC	Capture Value 5 Cycle This field stores the cycle number corresponding to the value captured in CVAL5. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

67.5.102 Output Enable Register (OUTEN)

Offset

Register	Offset
OUTEN	180h

Function

Contains PWM output enables.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				PWMA_EN				PWMB_EN				PWMX_EN			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-12 —	Reserved
11-8 PWMA_EN	PWM_A Output Enables

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field enables the PWM_A outputs of submodules 3 - 0, respectively. Set these bits to 0 (output disabled) when a PWM_A pin is being used for input capture. • 0b0 PWM_A output disabled. • 0b1 PWM_A output enabled.
7-4 PWMB_EN	PWM_B Output Enables This field enables the PWM_B outputs of submodules 3 - 0, respectively. Set these bits to 0 (output disabled) when a PWM_B pin is being used for input capture. • 0b0 PWM_B output disabled. • 0b1 PWM_B output enabled.
3-0 PWMX_EN	PWM_X Output Enables This field enables the PWM_X outputs of submodules 3 - 0, respectively. Set these bits to 0 (output disabled) when a PWM_X pin is being used for input capture or deadtime correction. • 0b0 PWM_X output disabled. • 0b1 PWM_X output enabled.

67.5.103 Mask Register (MASK)

Offset

Register	Offset
MASK	182h

Function

MASK is double buffered and does not take effect until a FORCE_OUT event occurs within the appropriate submodule. Reading MASK reads the buffered values and not necessarily the values currently in effect.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				MASKA				MASKB				MASKX			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-12 —	Reserved
11-8 MASKA	PWM_A Masks This field masks the PWM_A outputs of submodules 3 - 0, respectively by forcing the output to logic 0 prior to consideration of the output polarity. • 0b0 PWM_A output normal. • 0b1 PWM_A output masked. MASKA[3:0] masks PWM_A_3 through PWM_A_0
7-4 MASKB	PWM_B Masks This field masks the PWM_B outputs of submodules 3 - 0, respectively by forcing the output to logic 0 prior to consideration of the output polarity. • 0b0 PWM_B output normal. • 0b1 PWM_B output masked. MASKB[3:0] masks PWM_B_3 through PWM_B_0
3-0 MASKX	PWM_X Masks This field masks the PWM_X outputs of submodules 3 - 0, respectively by forcing the output to logic 0 prior to consideration of the output polarity. • 0b0 PWM_X output normal. • 0b1 PWM_X output masked. MASKX[3:0] masks PWM_X_3 through PWM_X_0

67.5.104 Software Controlled Output Register (SWCOUT)

Offset

Register	Offset
SWCOUT	184h

Function

These bits are double buffered and do not take effect until a FORCE_OUT event occurs within the appropriate submodule. Reading these bits reads the buffered value and not necessarily the value currently in effect.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								SM3OUT23	SM3OUT45	SM2OUT23	SM2OUT45	SM1OUT23	SM1OUT45	SM0OUT23	SM0OUT45
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8 —	Reserved
7 SM3OUT23	Submodule 3 Software Controlled Output 23 This field is used when DTSRCSEL[SM3SEL23] is set to 0b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 3 instead of PWM23. 1b - A logic 1 is supplied to the deadtime generator of submodule 3 instead of PWM23.
6 SM3OUT45	Submodule 3 Software Controlled Output 45 This field is used when DTSRCSEL[SM3SEL45] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 3 instead of PWM45. 1b - A logic 1 is supplied to the deadtime generator of submodule 3 instead of PWM45.
5 SM2OUT23	Submodule 2 Software Controlled Output 23 This field is used when DTSRCSEL[SM2SEL23] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 2 instead of PWM23. 1b - A logic 1 is supplied to the deadtime generator of submodule 2 instead of PWM23.
4 SM2OUT45	Submodule 2 Software Controlled Output 45 This field is used when DTSRCSEL[SM2SEL45] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 2 instead of PWM45. 1b - A logic 1 is supplied to the deadtime generator of submodule 2 instead of PWM45.
3 SM1OUT23	Submodule 1 Software Controlled Output 23

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is used when DTSRCSEL[SM1SEL23] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 1 instead of PWM23. 1b - A logic 1 is supplied to the deadtime generator of submodule 1 instead of PWM23.
2 SM1OUT45	Submodule 1 Software Controlled Output 45 This field is used when DTSRCSEL[SM1SEL45] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 1 instead of PWM45. 1b - A logic 1 is supplied to the deadtime generator of submodule 1 instead of PWM45.
1 SM0OUT23	Submodule 0 Software Controlled Output 23 This field is used when DTSRCSEL[SM0SEL23] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 0 instead of PWM23. 1b - A logic 1 is supplied to the deadtime generator of submodule 0 instead of PWM23.
0 SM0OUT45	Submodule 0 Software Controlled Output 45 This field is used when DTSRCSEL[SM0SEL45] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 0 instead of PWM45. 1b - A logic 1 is supplied to the deadtime generator of submodule 0 instead of PWM45.

67.5.105 PWM Source Select Register (DTSRCSEL)

Offset

Register	Offset
DTSRCSEL	186h

Function

The PWM source select bits are double buffered and do not take effect until a FORCE_OUT event occurs within the appropriate submodule. Reading these bits reads the buffered value and not necessarily the value currently in effect.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SM3SEL23		SM3SEL45		SM2SEL23		SM2SEL45		SM1SEL23		SM1SEL45		SM0SEL23		SM0SEL45	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 SM3SEL23	Submodule 3 PWM23 Control Select This field selects possible overrides to the generated SM3PWM23 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM3PWM23 signal used by the deadtime logic. 01b - Inverted generated SM3PWM23 signal used by the deadtime logic. 10b - SWCOUT[SM3OUT23] used by the deadtime logic. 11b - PWM3_EXTA signal used by the deadtime logic.
13-12 SM3SEL45	Submodule 3 PWM45 Control Select This field selects possible overrides to the generated SM3PWM45 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM3PWM45 signal used by the deadtime logic. 01b - Inverted generated SM3PWM45 signal used by the deadtime logic. 10b - SWCOUT[SM3OUT45] used by the deadtime logic. 11b - Reserved
11-10 SM2SEL23	Submodule 2 PWM23 Control Select This field selects possible overrides to the generated SM2PWM23 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM2PWM23 signal used by the deadtime logic. 01b - Inverted generated SM2PWM23 signal used by the deadtime logic. 10b - SWCOUT[SM2OUT23] used by the deadtime logic. 11b - PWM2_EXTA signal used by the deadtime logic.
9-8 SM2SEL45	Submodule 2 PWM45 Control Select This field selects possible overrides to the generated SM2PWM45 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM2PWM45 signal used by the deadtime logic. 01b - Inverted generated SM2PWM45 signal used by the deadtime logic. 10b - SWCOUT[SM2OUT45] used by the deadtime logic.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Reserved
7-6 SM1SEL23	Submodule 1 PWM23 Control Select This field selects possible overrides to the generated SM1PWM23 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM1PWM23 signal used by the deadtime logic. 01b - Inverted generated SM1PWM23 signal used by the deadtime logic. 10b - SWCOUT[SM1OUT23] used by the deadtime logic. 11b - PWM1_EXTA signal used by the deadtime logic.
5-4 SM1SEL45	Submodule 1 PWM45 Control Select This field selects possible overrides to the generated SM1PWM45 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM1PWM45 signal used by the deadtime logic. 01b - Inverted generated SM1PWM45 signal used by the deadtime logic. 10b - SWCOUT[SM1OUT45] used by the deadtime logic. 11b - Reserved
3-2 SM0SEL23	Submodule 0 PWM23 Control Select This field selects possible overrides to the generated SM0PWM23 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM0PWM23 signal used by the deadtime logic. 01b - Inverted generated SM0PWM23 signal used by the deadtime logic. 10b - SWCOUT[SM0OUT23] used by the deadtime logic. 11b - PWM0_EXTA signal used by the deadtime logic.
1-0 SM0SEL45	Submodule 0 PWM45 Control Select This field selects possible overrides to the generated SM0PWM45 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM0PWM45 signal used by the deadtime logic. 01b - Inverted generated SM0PWM45 signal used by the deadtime logic. 10b - SWCOUT[SM0OUT45] used by the deadtime logic. 11b - Reserved

67.5.106 Master Control Register (MCTRL)

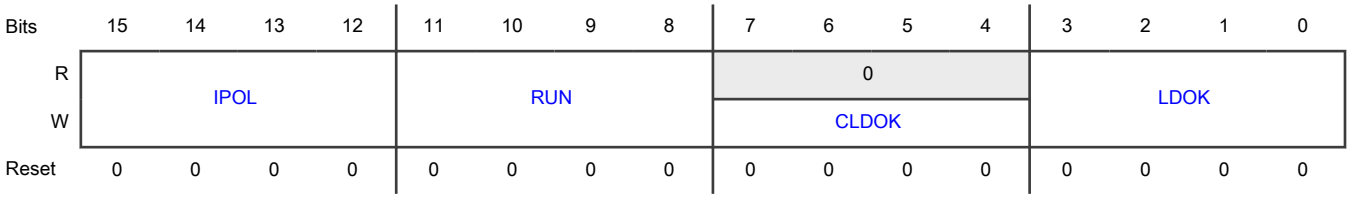
Offset

Register	Offset
MCTRL	188h

Function

In every 4-bit field in this register, each bit acts on a separate submodule. Accordingly, the description of every bit field refers to the effect of an individual bit.

Diagram



Fields

Field	Function
15-12 IPOL	Current Polarity This field corresponds to submodules 3 - 0, respectively. Each bit selects between PWM23 and PWM45 as the source for the generation of the complementary PWM pair output for the corresponding submodule. MCTRL[IPOL] is ignored in independent mode. MCTRL[IPOL] does not take effect until a FORCE_OUT event takes place in the appropriate submodule. Reading MCTRL[IPOL] reads the buffered value and not necessarily the value currently in effect. 0000b - PWM23 is used to generate complementary PWM pair in the corresponding submodule. 0001b - PWM45 is used to generate complementary PWM pair in the corresponding submodule.
11-8 RUN	Run This field enables the clocks to the PWM generator of submodules 3 - 0, respectively. The corresponding MCTRL[RUN] bit must be set for each submodule that is using its input capture functions or is using the local reload as its reload source. When this bit equals zero, the submodule counter is reset, and PWM outputs are held. A reset clears this field. 0000b - PWM counter is stopped, but PWM outputs hold the current state. 0001b - PWM counter is started in the corresponding submodule.
7-4 CLDOK	Clear Load Okay This field corresponds to submodules 3 - 0, respectively. Each write-only bit is used to clear the corresponding bit of MCTRL[LDOK]. Write a 1 to CLDOK to clear the corresponding MCTRL[LDOK] bit. If a reload occurs within a submodule with the corresponding MCTRL[LDOK] bit set at the same time that

Table continues on the next page...

Table continued from the previous page...

Field	Function
	MCTRL[CLDOK] is written, then the reload in that submodule will not be performed and MCTRL[LDOK] will be cleared. CLDOK bit is self-clearing and always reads as a 0.
3-0 LDOK	Load Okay This field corresponds to submodules 3 - 0, respectively. Each read/set bit loads CTRL[PRSC] and the INIT, FRACVALx, and VALx registers of the corresponding submodule into a set of buffers. The buffered prescaler divisor, submodule counter modulus value, and PWM pulse width take effect at the next PWM reload if CTRL[LDMOD] is clear or if CTRL[LDMOD] is set. Set the corresponding MCTRL[LDOK] bit by reading it when it is logic zero and then writing a logic one to it. The VALx, FRACVALx, INIT, and CTRL[PRSC] registers of the corresponding submodule cannot be written while the corresponding MCTRL[LDOK] bit is set. In Master Reload Mode (CTRL2[RELOAD_SEL]=1), it is necessary to set the LDOK bit corresponding to submodule0; however, it is recommended to also set the LDOK bit of the slave submodules, to prevent unwanted writes to the registers in the slave submodules. The MCTRL[LDOK] bit is automatically cleared after the new values are loaded, or it can be manually cleared before a reload by writing a logic 1 to the appropriate MCTRL[CLDOK] bit. LDOK bits cannot be written with a zero. MCTRL[LDOK] can be set in DMA mode when the DMA indicates that it has completed the update of all CTRL[PRSC], INIT, FRACVALx, and VALx registers in the corresponding submodule. Reset clears LDOK field. 0000b - Do not load new values. 0001b - Load prescaler, modulus, and PWM values of the corresponding submodule.

67.5.107 Master Control 2 Register (MCTRL2)

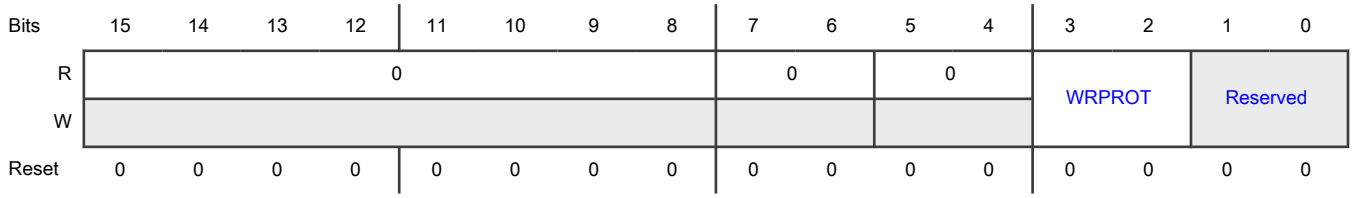
Offset

Register	Offset
MCTRL2	18Ah

Function

Includes control for monitoring the PLL state and write protection of some configuration registers.

Diagram



Fields

Field	Function
15-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 WRPROT	Write protect Enable write protection of some configuration registers of eFlexPWM. 00b - Write protection off (default). 01b - Write protection on. 10b - Write protection off and locked until chip reset. 11b - Write protection on and locked until chip reset.
1-0 —	Reserved

67.5.108 Fault Control Register (FCTRL0)

Offset

Register	Offset
FCTRL0	18Ch

Function

For every 4-bit field in this register, the bits act on the fault inputs in order. For example, FLVL bits 15-12 act on faults 3-0, respectively.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FLVL				FAUTO				FSAFE				FIE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-12 FLVL	Fault Level This field selects the active logic level of the individual fault inputs 3-0, respectively. A reset clears this field. 0000b - A logic 0 on the fault input indicates a fault condition. 0001b - A logic 1 on the fault input indicates a fault condition.
11-8 FAUTO	Automatic Fault Clearing This field selects automatic or manual clearing of faults 3-0, respectively. A reset clears this field. 0000b - Manual fault clearing. PWM outputs disabled by this fault are not enabled until FSTS[FFLAGx] is clear at the start of a half cycle or full cycle depending on the states of FSTS[FHALF] and FSTS[FFULL]. If neither FFULL nor FHALF is set, then the fault condition cannot be cleared. This is further controlled by FCTRL[FSAFE]. 0001b - Automatic fault clearing. PWM outputs disabled by this fault are enabled when FSTS[FFPINx] is clear at the start of a half cycle or full cycle depending on the states of FSTS[FHALF] and FSTS[FFULL] without regard to the state of FSTS[FFLAGx]. If neither FFULL nor FHALF is set, then the fault condition cannot be cleared.
7-4 FSAFE	Fault Safety Mode This field selects the safety mode during manual fault clearing. A reset clears this field. FSTS[FFPINx] may indicate that a fault condition still exists even though the actual fault signal at the FAULTx pin is clear due to the fault filter latency. 0000b - Normal mode. PWM outputs disabled by this fault are not enabled until FSTS[FFLAGx] is clear at the start of a half cycle or full cycle depending on the states of FSTS[FHALF] and FSTS[FFULL] without regard to the state of FSTS[FFPINx]. If neither FHALF nor FFULL is set, then the fault condition cannot be cleared. The PWM outputs disabled by this fault input will not be re-enabled until the actual FAULTx input signal de-asserts since the fault input will combinationally disable the PWM outputs (as programmed in DISMAPn). 0001b - Safe mode. PWM outputs disabled by this fault are not enabled until FSTS[FFLAGx] is clear and FSTS[FFPINx] is clear at the start of a half cycle or full cycle depending on the states of FSTS[FHALF] and FSTS[FFULL]. If neither FHLAF nor FFULL is set, then the fault condition cannot be cleared.
3-0 FIE	Fault Interrupt Enables This field enables CPU interrupt requests generated by the FAULTx pins. A reset clears this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The fault protection circuit is independent of the FIEx bit and is always active. If a fault is detected, the PWM outputs are disabled according to the disable mapping register. 0000b - FAULTx CPU interrupt requests disabled. 0001b - FAULTx CPU interrupt requests enabled.

67.5.109 Fault Status Register (FSTS0)

Offset

Register	Offset
FSTS0	18Eh

Function

Includes controls related to fault conditions.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FHALF				FFPIN				FFULL				FFLAG			
W																
Reset	0	0	0	0	u	u	u	u	0	0	0	0	u	u	u	u

Fields

Field	Function
15-12 FHALF	Half Cycle Fault Recovery This field is used to control the timing for re-enabling the PWM outputs after a fault condition. These bits apply to both automatic and manual clearing of a fault condition. **NOTE** Both FHALF and FFULL can be set so that the fault recovery occurs at the start of a full cycle and at the start of a half cycle (as defined by VAL0). If neither FHALF nor FFULL is set, then no fault recovery is possible. 0000b - PWM outputs are not re-enabled at the start of a half cycle. 0001b - PWM outputs are re-enabled at the start of a half cycle (as defined by VAL0).
11-8 FFPIN	Filtered Fault Pins

Table continues on the next page...

Table continued from the previous page...

Field	Function
	These read-only bits reflect the current state of the filtered FAULTx pins converted to high polarity. A logic 1 indicates that a fault condition exists on the filtered FAULTx pin. A reset has no effect on this field. After the system reset de-asserts, these are the possible values of FFPIN: If FCTRL[FLVL] = 0 and FAULTx = 0, then FFPIN is set to 1. If FCTRL[FLVL] = 0 and FAULTx = 1, then FFPIN is kept as 0. If FCTRL[FLVL] = 1 and FAULTx = 0, then FFPIN is kept as 0. If FCTRL[FLVL] = 1 and FAULTx = 1, then FFPIN is set to 1.
7-4 FFULL	Full Cycle This field is used to control the timing for re-enabling the PWM outputs after a fault condition. These bits apply to both automatic and manual clearing of a fault condition. **NOTE** Both FHALF and FFULL can be set so that the fault recovery occurs at the start of a full cycle and at the start of a half cycle (as defined by VAL0). If neither FHALF nor FFULL is set, then no fault recovery is possible. 0000b - PWM outputs are not re-enabled at the start of a full cycle 0001b - PWM outputs are re-enabled at the start of a full cycle
3-0 FFLAG	Fault Flags These read-only flags are set within two CPU cycles after a transition to active on the FAULTx pin. Clear this bit by writing a logic one to it. A reset clears this field. While the reset value is 0, these bits may be set to 1 by the time they can be read depending on the state of the fault input signals. After the system reset de-asserts, these are the possible values of FFLAG: If FCTRL[FLVL] = 0 and FAULTx = 0, then FFLAG is set to 1. If FCTRL[FLVL] = 0 and FAULTx = 1, then FFLAG is kept as 0. If FCTRL[FLVL] = 1 and FAULTx = 0, then FFLAG is kept as 0. If FCTRL[FLVL] = 1 and FAULTx = 1, then FFLAG is set to 1. 0000b - No fault on the FAULTx pin. 0001b - Fault on the FAULTx pin.

67.5.110 Fault Filter Register (FFILT0)

Offset

Register	Offset
FFILT0	190h

Function

The settings in this register are shared among each of the fault input filters within the fault channel.

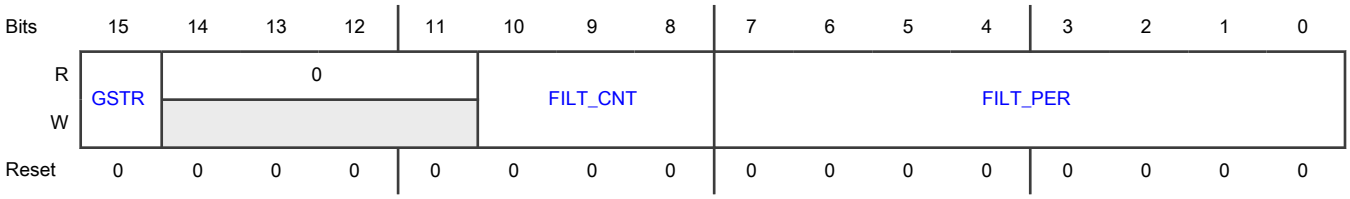
Input filter considerations include:

- Set the FILT_PER value such that the sampling period is larger than the period of the expected noise. This way a noise spike corrupts one sample. Select the FILT_CNT value to reduce and recognize the probability of noisy samples causing an incorrect transition. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the FILT_CNT+3 power.
- The values of FILT_PER and FILT_CNT must be traded off against the desire for minimal latency in recognizing input transitions. Turning on the input filter (setting FILT_PER to a non-zero value) introduces a latency of ((FILT_CNT+4) x FILT_PER x IPBus clock period).

NOTE

When the filter is enabled, there is a combinational path to disable the PWM outputs to ensure rapid response to fault conditions, and fault response if the PWM module loses its clock. The latency induced by the filter will be seen in the time to set FSTS[FFLAG] and FSTS[FFPIN].

Diagram



Fields

Field	Function
15 GSTR	Fault Glitch Stretch Enable This field is used to enable the fault glitch-stretching logic. This logic ensures that narrow fault glitches are stretched to be at least 2 IPBus clock cycles wide. In some cases, a narrow fault input can cause problems due to the short PWM output shutdown/re-activation time. The stretching logic ensures that a glitch on the fault input, when the fault filter is disabled, will be registered in the fault flags. 0b - Fault input glitch stretching is disabled. 1b - Input fault signals are stretched to at least 2 IPBus clock cycles.
14-11 —	Reserved
10-8 FILT_CNT	Fault Filter Count These bits represent the number of consecutive samples that must agree prior to the input filter accepting an input transition. The number of samples is the decimal value of this field plus three: the bit field value of 0-7 represents 3-10 samples, respectively. The value of FILT_CNT affects the input latency.
7-0 FILT_PER	Fault Filter Period This 8-bit field applies universally to all fault inputs. These bits represent the sampling period (in IPBus clock cycles) of the fault pin input filter. Each input is sampled multiple times at the rate specified by this field. If FILT_PER is 0x00 (default), then the input filter is bypassed. The value of FILT_PER affects the input latency.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE When changing values for FILT_PER from one non-zero value to another non-zero value, first write a value of zero to clear the filter.

67.5.111 Fault Test Register (FTST0)

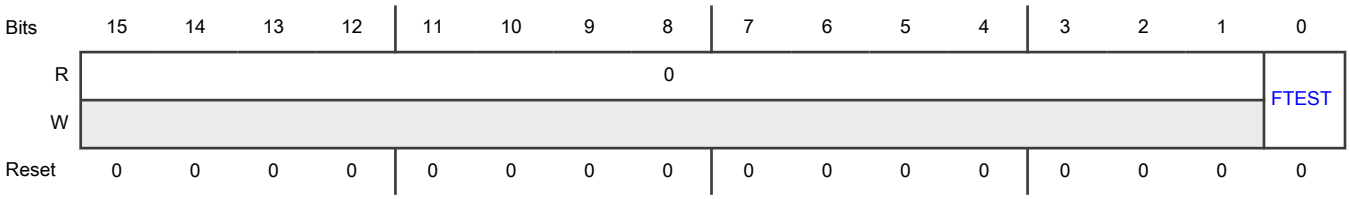
Offset

Register	Offset
FTST0	192h

Function

Contains FTEST field for fault simulation.

Diagram



Fields

Field	Function
15-1 —	Reserved
0 FTEST	Fault Test This field is used to simulate a fault condition. Setting this bit causes a simulated fault to be sent into all of the fault filters. The condition propagates to the fault flags and possibly the PWM outputs depending on the DISMAPn settings. Clearing this bit removes the simulated fault condition. 0b - No fault 1b - Cause a simulated fault

67.5.112 Fault Control 2 Register (FCTRL20)

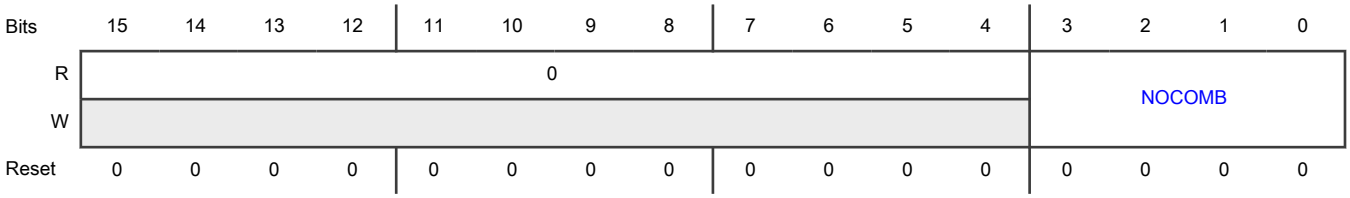
Offset

Register	Offset
FCTRL20	194h

Function

Controls combinational link from fault inputs to PWM outputs.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 NOCOMB	No Combinational Path From Fault Input To PWM Output This field is used to control the combinational path from the fault inputs to the PWM outputs. When these bits are low (default), the corresponding fault inputs have a combinational path to the PWM outputs that are sensitive to these fault inputs (as defined by DISMAP0). This combinational path is a safety feature that ensures the output is disabled even if the SOC has a failure of its clocking system. The combinational path also means that a pulse on the fault input can cause a brief disable of the PWM output even if the fault pulse is not wide enough to get through the input filter and be latched in the fault logic. Setting these bits removes the combinational path and uses the filtered and latched fault signals as the fault source to disable the PWM outputs. This eliminates fault glitches from creating PWM output glitches but also increases the latency to respond to a real fault. 0000b - There is a combinational link from the fault inputs to the PWM outputs. The fault inputs are combined with the filtered and latched fault signals to disable the PWM outputs. 0001b - The direct combinational path from the fault inputs to the PWM outputs is disabled and the filtered and latched fault signals are used to disable the PWM outputs.

Chapter 68
General Purpose Timer (GPT)

68.1 Chip-specific GPT Information

Table 1057. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

68.2 Overview

GPT is a 32-bit, up-counter with clock source and frequency scaling options that provide a wide range of timing rates for various system applications. External hardware can send GPT a Capture Event signal on an input pin to load the current timer value into a register for software to read. This Capture Event signal can be configured to trigger GPT on a rising edge, a falling edge, or both. When the timer matches a preset value, GPT can generate an interrupt for software and a Compare Event signal on output pins for external hardware.

68.2.1 Block diagram

Figure 519 is a functional block diagram of GPT.

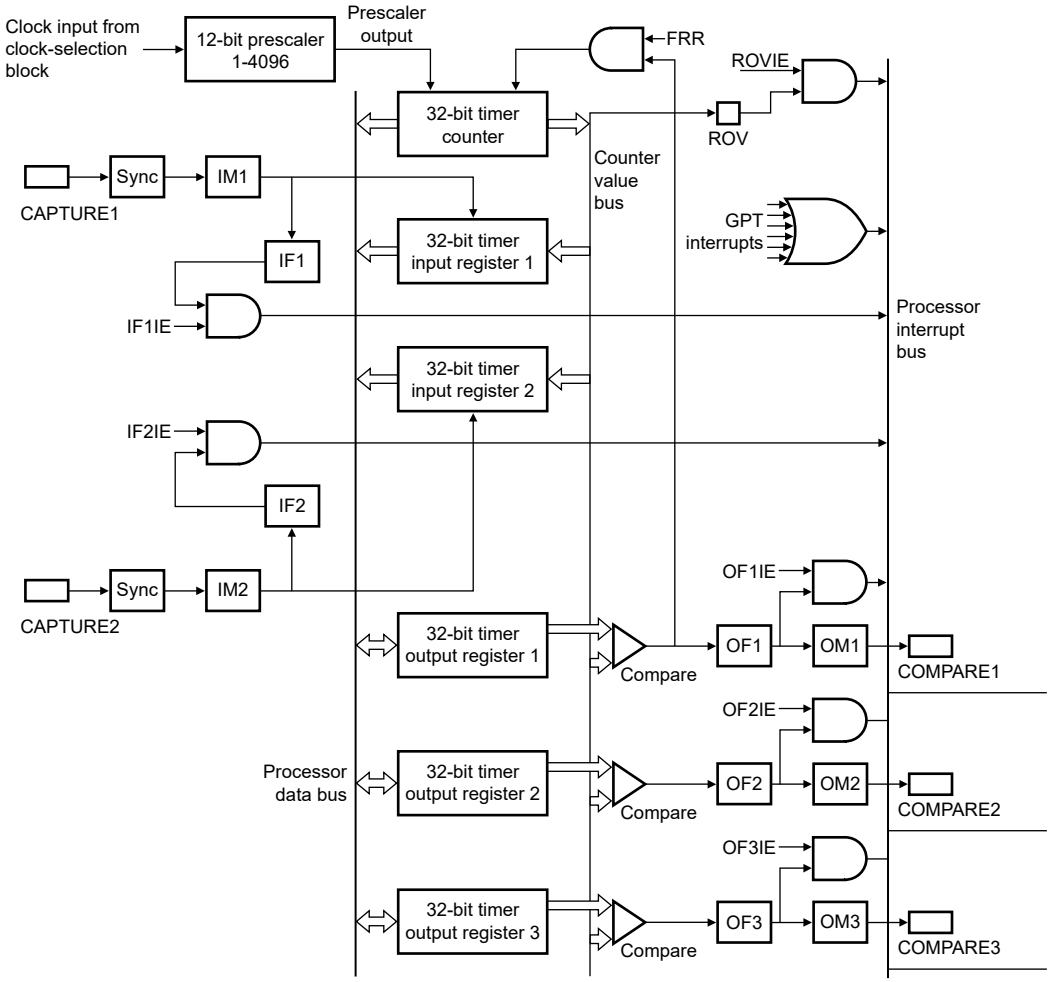


Figure 519. GPT block diagram

68.2.2 Features

- One 32-bit up-counter with clock source selection, including external clock.
- Two input capture channels with a programmable trigger edge.
- Three output compare channels with a programmable output mode. A forced-compare feature is also available.
- Programmable active option in Low Power and Debug modes.
- Interrupt generation at capture, compare, and rollover events.
- Restart or Free-Run modes for counter operations.

68.3 Functional description

68.3.1 Operating modes

You can program the GPT counter to work in one of two modes: Restart or Free-Run.

68.3.1.1 Restart mode

In Restart mode (selectable through the Control (CR)), the counter resets and starts again from 0000_0000h when the counter reaches the compared value. The restart feature is only associated with Compare Channel 1.

Any write access to the Channel 1 compare register resets the GPT counter. This is done to avoid missing a compare event when the compare value is changed from a higher value to a lower value while counting is already in process.

For the other two compare channels, the counter is not reset when the compare event occurs.

68.3.1.2 Free-Run mode

In Free-Run mode, the counter is mot reset when compare events occur for all three channels. Instead, the counter continues to count until FFFF_FFFFh, and then rolls over to 0000_0000h.

68.3.2 Operation

GPT has a single counter (GPT_CNT) which is a 32-bit free-running up-counter. It starts counting after it is enabled by writing 1 to CR[EN]. The clock source of the counter is the output of the prescaler labeled "Prescaler output" in Figure 519.

If the GPT timer is disabled (EN=0), then both the Main Counter and Prescaler Counter freeze their current count values. Use the CR[ENMOD] field to determine the value of the GPT counter when the CR[EN] field is programmed and the Counter is enabled again.

- If you write 1 to CR[ENMOD], then the Main Counter and Prescaler Counter values are reset to 0 when GPT is enabled (EN=1).
- If you write 0 to CR[ENMOD], then the Main Counter and Prescaler Counter restart counting from their frozen values when GPT is enabled again (EN=1).

If you program GPT to be disabled in a low-power mode (that is, Stop or Wait mode), then the Main Counter and Prescaler Counter freeze at their current count values when GPT enters Low Power mode. When GPT exits a low-power mode, the Main Counter and Prescaler Counter start counting from their frozen values regardless of the CR[ENMOD] field value.

NOTE

Both Input Capture Channels use the same counter (GPT_CNT). The processor can read GPT_CNT at any time.

A hardware reset resets all the GPT registers to their respective reset values. All registers, except the Output Compare (OCR1 - OCR3) registers (OCR1, OCR2, and OCR3), obtain a value of 0h. The Compare registers are reset to FFFF_FFFFh.

Writing to CR[SWR] resets all of the register fields except for the CR[EN], CR[ENMOD], CR[STOPEN], CR[WAITEN], and CR[DBGEN] fields. The state of these fields is not affected by a software reset.

NOTE

You can perform a software reset while GPT is disabled.

68.3.2.1 Input capture

There are two input capture channels and each channel has a dedicated capture signal, capture register, and input edge detection and selection logic. Each input capture function has an associated status flag and can cause the processor to make an interrupt service request.

When a selected edge transition occurs on an input capture signal, the contents of Counter (CNT) are captured on the corresponding capture register and the appropriate interrupt status flag is set. You can generate an interrupt request when the transition is detected by writing to its corresponding enable field in Interrupt (IR). You can program the capture to occur on the input signal's rising edge, falling edge, on both the rising and falling edges, or you can disable the capture. The events are synchronized with the clock you selected to run the counter. Only those transitions that occur at least one clock cycle after the previous recorded transition will be guaranteed to trigger a capture event. There can be up to one clock cycle of uncertainty in the latching of the input transition.

NOTE

The Input Capture registers can be read at any time without affecting their values.

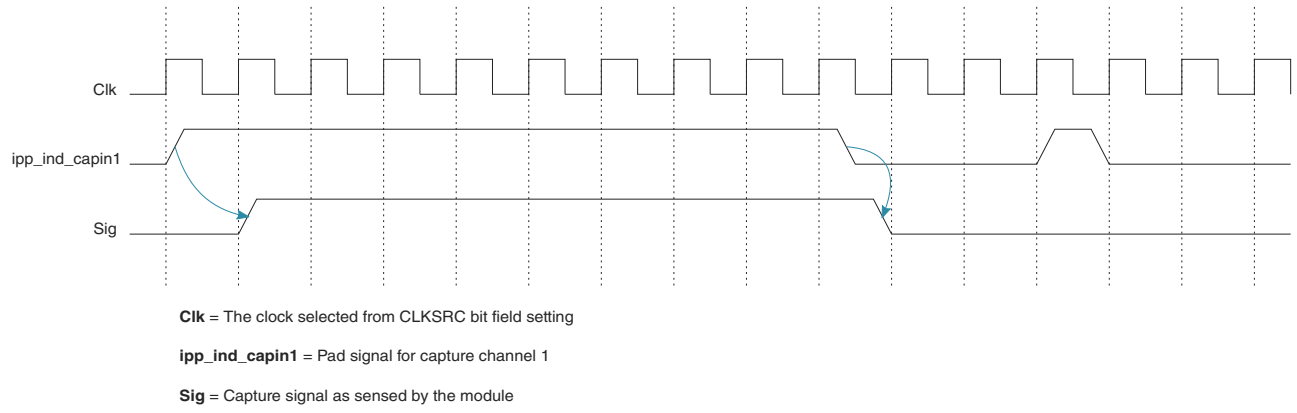


Figure 520. Input capture event timing

68.3.2.2 Output compare

The three Output Compare Channels use the same counter (GPT_CNT) as the Input Capture Channels. When the programmed content of an Output Compare (OCR1 - OCR3) register matches the value in GPT_CNT, an output compare status flag is set. An interrupt is also generated, if the corresponding field is 1 in Interrupt (IR). Consequently, the Output Compare signal either sets, clears, toggles, is not affected at all, or provides an active-low pulse for one input clock period, according to the CR[OM*n*] field setting.

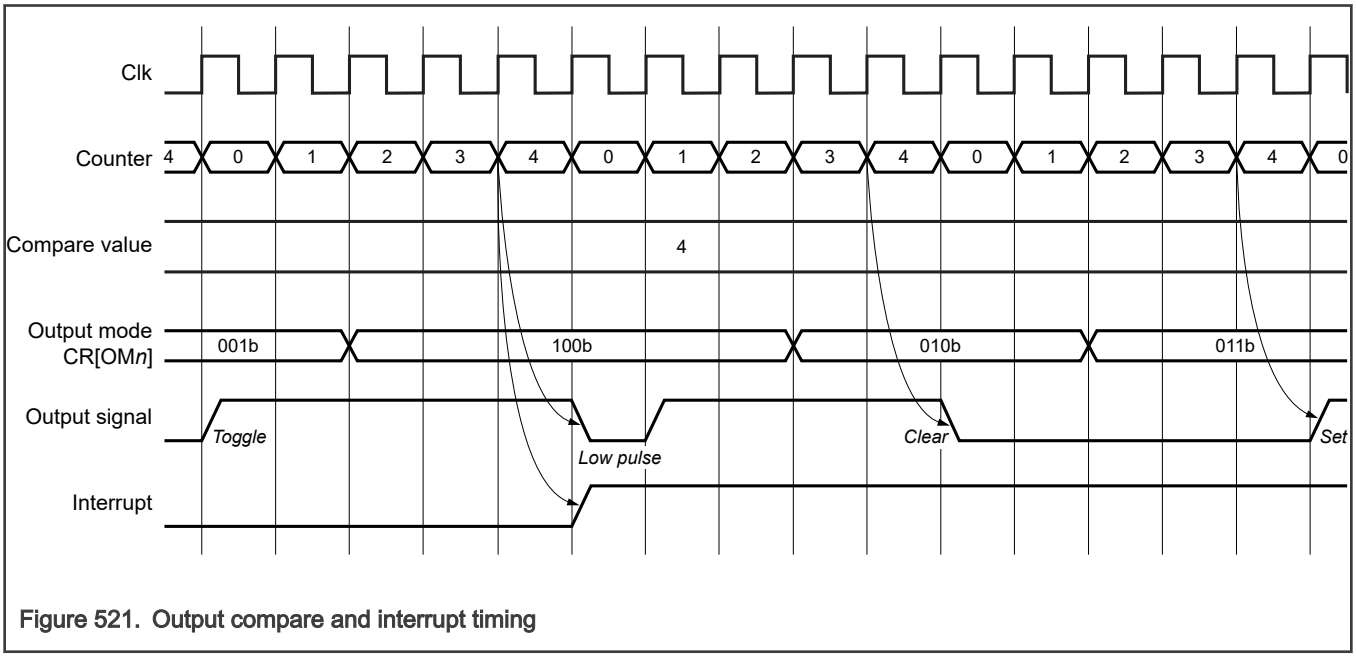


Figure 521. Output compare and interrupt timing

There is also a forced-compare feature that allows you to generate a compare event when needed, without requiring the counter value to be equal to the compare value. The action taken as a result of a forced compare is the same as when an output compare match occurs, except that the status flags are not set and no interrupt is generated. Forced channels take programmed action immediately after you write to the force-compare fields. These fields are self-negating and always read as zeroes.

68.3.2.3 Low Power mode behavior

If the clock from the selected clock source is available in low-power modes, the counter continues to run depending on how you program the control field for that mode. You can only use the external clock (GPT_CLK) as the clock source if the peripheral clock (MODULE_CLK) is available. If the clock is not present or if the corresponding low-power field in Control (CR) is 0, the Main Counter and the Prescaler Counter freeze at their current values and resume counting (from their frozen values) when the chip exits Low Power mode.

68.3.2.4 Debug mode behavior

In Debug mode, the modules in the chip can either continue to run or halt. Use CR[DBGEN] to enable or disable the GPT timer in Debug mode.

68.3.3 Clocking

You can select the clock input to the prescaler from multiple clock sources, as shown in Figure 522. See the Clock Controller Module (CCM) for the chip-specific GPT clock source connection, configuration, and gating information.

NOTE

All clocking options may not be supported.

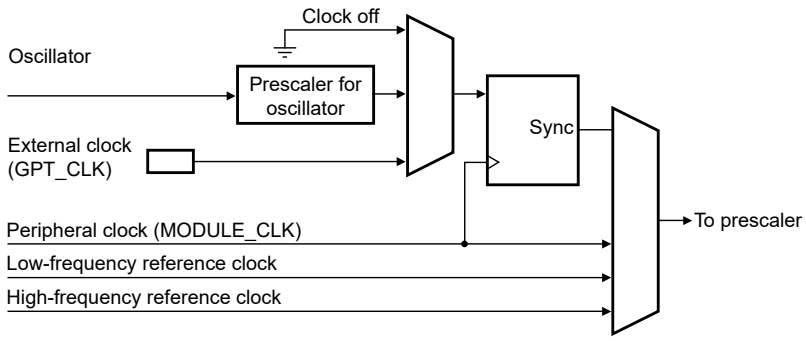


Figure 522. GPT counter clocks diagram

Table 1058 describes the clock sources for GPT.

Table 1058. GPT clocks

Clock name	Description
Peripheral clock (MODULE_CLK)	This is the internal clock used to access the module registers. If you select the peripheral clock or external clock (GPT_CLK) as the clock source (by writing 001 or 011 to CR[CLKSRC]), then the peripheral clock will be on in normal GPT operations. If you program GPT to be disabled in low-power modes (when STOPEN, WAITEN, or DOZEN equals zero), then the peripheral clock can be switched off.
Low-frequency reference clock (ipg_clk_32k)	This low-frequency, 32 kHz reference clock (provided by CCM) is intended to be on in Low Power mode when the peripheral clock is turned off. This enables GPT to operate in Low Power mode using the low-frequency reference clock.
High-frequency reference clock (ipg_clk_highfreq)	This high-frequency reference clock (provided by CCM) is intended to be on in Normal Power mode when the peripheral clock is turned off. This enables GPT to operate in Normal Power mode using the high-frequency reference clock.

Table continues on the next page...

Table 1058. GPT clocks (continued)

Clock name	Description
Oscillator clock (ipg_clk_24M)	This 24-MHz oscillator reference clock is intended to be used against frequency changes of the peripheral clock and to provide a more accurate timer clock for system operation. The 24 MHz oscillator clock is provided without synchronization to the system bus clock (ahb_clk) in Normal functional mode. Synchronization is done in GPT. Before synchronization, the 24 MHz oscillator clock is divided by a dedicated prescaler to ensure the clock frequency is less than half of the system bus clock.
External clock (GPT_CLK)	The external clock comes from outside the chip and can be selected to run the GPT counter. The external clock frequency is limited to less than 1/4 frequency of the peripheral clock (MODULE_CLK) for proper GPT operations. **NOTE** If the peripheral clock is not available in low-power modes, then you cannot use the external clock to run the counter.

Use CR[CLKSRC] to configure the clock input source. You can disable the clock input to the prescaler by writing 000 to the CLKSRC field. Only change the CLKSRC field value after disabling GPT (using the CR[EN] field).

Use the PR[PRESCALER] field to select the divide ratio of the input clock that drives the main counter. The prescaler can divide the input clock by a value from 1 to 4096 and can be changed at any time. Changing the value of the PR[PRESCALER] field immediately affects the output clock frequency.

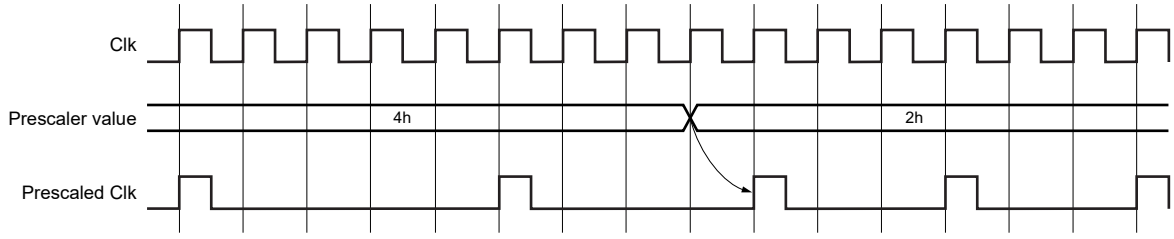


Figure 523. Prescaler value change timing diagram

68.3.4 Interrupts

GPT generates six different interrupts. If the selected clock for running the counter is available, then GPT can generate all interrupts in Low Power and Debug modes.

Table 1059. Interrupts

Interrupt	Description
Rollover Interrupt	GPT generates the Rollover Interrupt when the GPT counter reaches FFFF_FFFFh, then resets to 0000_0000h and continues counting. Use IR[ROVIE] to enable the Rollover Interrupt. SR[ROV] indicates the rollover status.
Input Capture Interrupt 1, 2	After a capture event occurs, the associated Input Capture Channel generates an interrupt. Use the IF2IE and IF1IE fields of the Interrupt (IR) to enable the capture-event interrupts. The IF2 and IF1 fields of the Status (SR) indicate their statuses.

Table continues on the next page...

Table 1059. Interrupts (continued)

Interrupt	Description
	A pending capture interrupt does not affect the counter value captured because of a capture event. The Control (CR) is updated with a new counter value when a capture event occurs, regardless of whether the interrupt of that capture channel has been serviced or not.
Output Compare Interrupt 1, 2, 3	After a compare event occurs, the associated Output Compare Channel generates an interrupt. Use OF3IE, OF2IE, and OF1IE fields of the Interrupt (IR) to enable compare-event interrupts. The OF3, OF2, and OF1 fields of the Status (SR) indicate their statuses. A forced compare does not generate an interrupt.

NOTE

A cumulative interrupt line is also present. It asserts whenever any of the above interrupts post. The cumulative interrupt line has no associated enable or status bits.

68.4 External signals

GPT follows the IP Bus protocol for interfacing with the processor core. The only interface signals GPT has with other modules inside the chip are the clock and reset inputs (from the clock and reset controller module) and the interrupt signals to the processor interrupt handler. There are also functional and clock inputs and functional output signals going outside the chip boundary.

Table 1060 describes all block signals that connect off-chip. See the IOMUX Controller for GPT pin mux assignments.

Table 1060. Off-chip module signals

Name	Direction	Function	Reset state	Pullup
GPT_CLK	I	Input for an external clock that the counter can be operated at.	–	Passive hysteresis
CAPTURE1	I	Input for a capture event for Input Capture Channel 1.	–	Passive
CAPTURE2	I	Input for a capture event for Input Capture Channel 2.	–	Passive
COMPARE1	O	Output that indicates a compare event occurrence in Output Compare Channel 1.	0	Passive
COMPARE2	O	Output that indicates a compare event occurrence in Output Compare Channel 2.	0	Passive
COMPARE3	O	Output that indicates a compare event occurrence in Output Compare Channel 3.	0	Passive

68.4.1 External clock input

You can operate the GPT counter using an external clock from outside the chip. The external clock input (GPT_CLK) is the input used for that purpose.

The external clock input is treated as asynchronous to the peripheral clock (MODULE_CLK). To ensure proper GPT operations, the external clock input frequency must be less than 1/4 of the frequency of the peripheral clock. Because GPT_CLK is a clock input, hysteresis characteristics on this pad are required.

68.4.2 Input capture trigger signals

The GPT counter value can be stored in a register, triggered by an event from outside the chip. A positive or negative edge on the CAPTURE*n* signals can trigger this capture event. These signals are treated as asynchronous to the peripheral clock (MODULE_CLK). Only those transitions which occur at least a single clock cycle (for the clock you select to run the counter) after the previous recorded transition are guaranteed to trigger a capture event.

68.4.3 Output compare signals

The COMPARE*n* signals indicate that an output compare event has occurred on the corresponding Output Compare Channel *n*. See Output compare for more information.

68.5 Initialization and application information

68.5.1 Selecting the clock source

To select the clock source, use the clock source field in the Control Register (CR[CLKSRC]). Only change the clock source field after disabling GPT, using CR[EN].

Use this sequence to change the clock source:

1. Write 0 to CR[EN] to disable GPT.
2. Clear Interrupt (IR) to disable GPT interrupts.
3. Write 0 to CR[OM*n*] to disable the Output mode for all channels.
4. Write 0 to CR[IM*n*] to disable the input capture modes.
5. Change the clock source (CR[CLKSRC]) to the desired value.
6. Write 1 to CR[SWR] to assert a GPT software reset.
7. Clear the flags in the Status (SR) register.
8. Write 1 to CR[ENMOD] to reset the GPT counter to zero.
9. Write 1 to CR[EN] to enable GPT.
10. Write 1 to the desired interrupt enable fields in Interrupt (IR) to enable GPT interrupts.

68.6 Memory map and register definitions

68.6.1 GPT register descriptions

GPT has 10 user-accessible, 32-bit registers used to configure, operate, and monitor the GPT module.

Writing to the GPT Status Registers (read-only registers ICR1, ICR2, and CNT) generates a bus exception.

68.6.1.1 GPT memory map

GPT1 base address: 446C_0000h

GPT2 base address: 42EC_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	0000_0000h
4h	Prescaler (PR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8h	Status (SR)	32	RW	0000_0000h
Ch	Interrupt (IR)	32	RW	0000_0000h
10h - 18h	Output Compare (OCR1 - OCR3)	32	RW	FFFF_FFFFh
1Ch - 20h	Input Capture (ICR1 - ICR2)	32	R	0000_0000h
24h	Counter (CNT)	32	R	0000_0000h

68.6.1.2 Control (CR)

Offset

Register	Offset
CR	0h

Function

Contains configuration options for GPT operations, such as selecting the clock source for the counter and selecting the output response for a channel when a compare event occurs. The GPT Control Register also includes the control bit used to trigger a software reset (CR[SWR]).

NOTE

Writing 1 to SWR resets all of the GPT registers to their default reset values, except for the EN, ENMOD, STOPEN, DOZEEN, WAITEN, and DBGEN fields in this register.

For applications where precise timing is critical, note that writes to CR take effect after one cycle of the GPT register bus clock (1 wait state of latency). Reads to the register occur immediately (0 wait states).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	OM3			OM2			OM1			IM2		IM1	
W	FO3	FO2	FO1													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SWR	0			EN_24M	FRR	CLKSRC				STOPEN	DOZEEN	WAITEN	DBGEN	ENMOD	EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 FOn	Force Output Compare for Channel n Forces the Output Compare *n* signal to respond according to the action programmed in the OM*n* field (in this register). This field is self-clearing and always reads as zero. **NOTE** The OF*n* flag in the Status register (SR) is not set. 0b - No effect 1b - Trigger the programmed response on the output signal
28-26: OM3 25-23: OM2 22-20: OM1	Output Compare Operating Mode for Channel n Selects the response on the output signal when a compare event occurs. 000b - Output disabled. No response on the signal. 001b - Toggle output 010b - Clear output 011b - Set output 1xxb - Generate a low pulse that is one input clock cycle wide on the output signal. When OMn is first programmed as 1xx, the output is set to one immediately on the next input clock (if it was not already). The input clock here refers to the clock selected by the CLKSRC field of this register.
19-18: IM2 17-16: IM1	Input Capture Operating Mode for Channel n Selects which input transition triggers a capture event. 00b - Capture disabled 01b - Capture on rising edge only 10b - Capture on falling edge only 11b - Capture on both edges
15 SWR	Software Reset Resets all of the registers to their default reset values, except for the DOZEEN, EN, ENMOD, STOPEN, WAITEN, and DBGEN fields in the GPT Control register. The SWR field remains set while the module is resetting. It automatically clears when the reset procedure finishes. 0b - GPT is not in software reset state 1b - GPT is in software reset state
14-11 —	Reserved Write zeros to these reserved bits. These bits always read as zero.
10 EN_24M	Enable Oscillator Clock Input Enables the 24-MHz clock input from an oscillator.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
9 FRR	Free-Run or Restart Mode Selects the behavior of GPT when a compare event in channel 1 occurs. 0b - Restart mode. After a compare event, the counter resets to 0000_0000h and resumes counting. 1b - Free-Run mode. After a compare event, the counter continues counting until FFFF_FFFFh and then rolls over to 0.
8-6 CLKSRC	Clock Source Select Selects which clock goes to the prescaler and then runs the GPT counter. **NOTE** Only change the CLKSRC field value after disabling GPT (EN=0). For other programming requirements when changing the clock source, see Selecting the clock source. 000b - No clock 001b - Peripheral clock (MODULE_CLK) 010b - High-frequency reference clock (ipg_clk_highfreq) 011b - External clock 100b - Low-frequency reference clock (ipg_clk_32k)
5 STOPEN	GPT Stop Mode Enable Enables GPT operation during Stop mode. 0b - Disable in Stop mode 1b - Enable in Stop mode
4 DOZEEN	GPT Doze Mode Enable Enables GPT operation during Doze mode. 0b - Disable in Doze mode 1b - Enable in Doze mode
3 WAITEN	GPT Wait Mode Enable Enables GPT operation during Wait mode. 0b - Disable in Wait mode 1b - Enable in Wait mode
2 DBGEN	GPT Debug Mode Enable Enables GPT operation during Debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable in Debug mode 1b - Enable in Debug mode
1 ENMOD	GPT Enable Mode When GPT is disabled (EN=0), both the Main and Prescaler counters freeze their current count values. The ENMOD field selects the value of the Main Counter and Prescaler Counter when GPT is enabled again (EN=1). If you program GPT to be disabled in low-power modes (STOP or WAIT), the Main Counter and Prescaler Counter freeze at their current count values when the GPT enters Low Power mode. When GPT exits Low Power mode, the Main Counter and Prescaler Counter start counting from their frozen values, regardless of the ENMOD value. A software reset (SWR=1) clears the Main Counter and Prescaler Counter values, regardless of the value of the EN or ENMOD fields. 0b - Restart counting from frozen values after GPT is enabled (EN=1). 1b - Reset counting from 0 after GPT is enabled (EN=1).
0 EN	GPT Enable Enables the GPT module. **NOTE** Program all registers before writing 1 to the EN field. 0b - Disable 1b - Enable

68.6.1.3 Prescaler (PR)

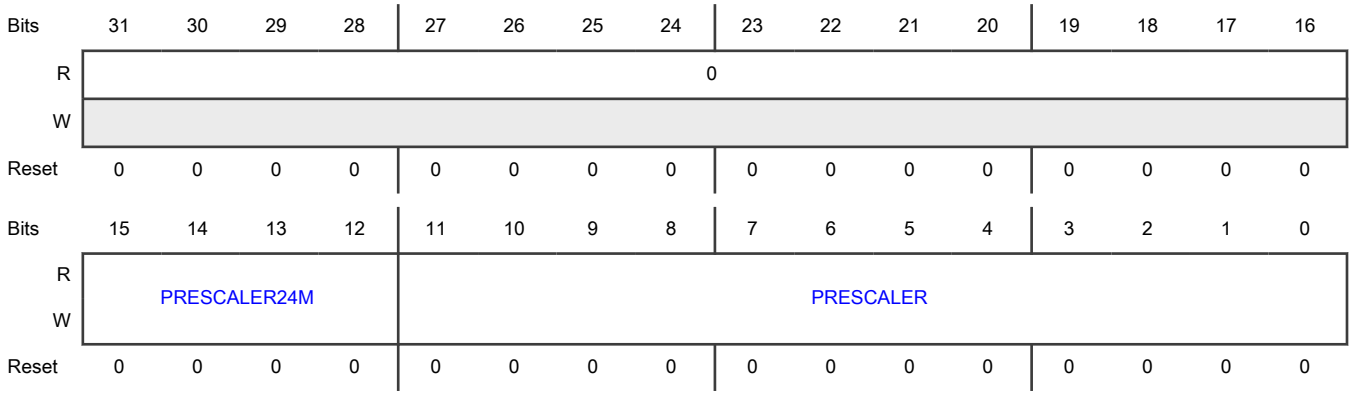
Offset

Register	Offset
PR	4h

Function

Contains the divide value of the clock that runs the counter.

Diagram



Fields

Field	Function
31-16 —	Reserved Write zero to this field. These bits always read as zero.
15-12 PRESCALER24M	Prescaler Divide Value for the Oscillator Clock The 24-MHz oscillator clock is divided by [PRESCALER24M + 1] before it is selected by the CLKSRC field. If the 24-MHz oscillator clock is not selected, this field does not take effect. 0000b - Divide by 1 0001b - Divide by 2 1111b - Divide by 16
11-0 PRESCALER	Prescaler Divide Value The clock selected by the CLKSRC field is divided by [PRESCALER + 1], and then used to run the counter. **NOTE** Changing the value of PRESCALER causes the Prescaler counter to reset and a new count period to start immediately. See Clocking for an example timing diagram. 0000_0000_0000b - Divide by 1 0000_0000_0001b - Divide by 2 1111_1111_1111b - Divide by 4096

68.6.1.4 Status (SR)

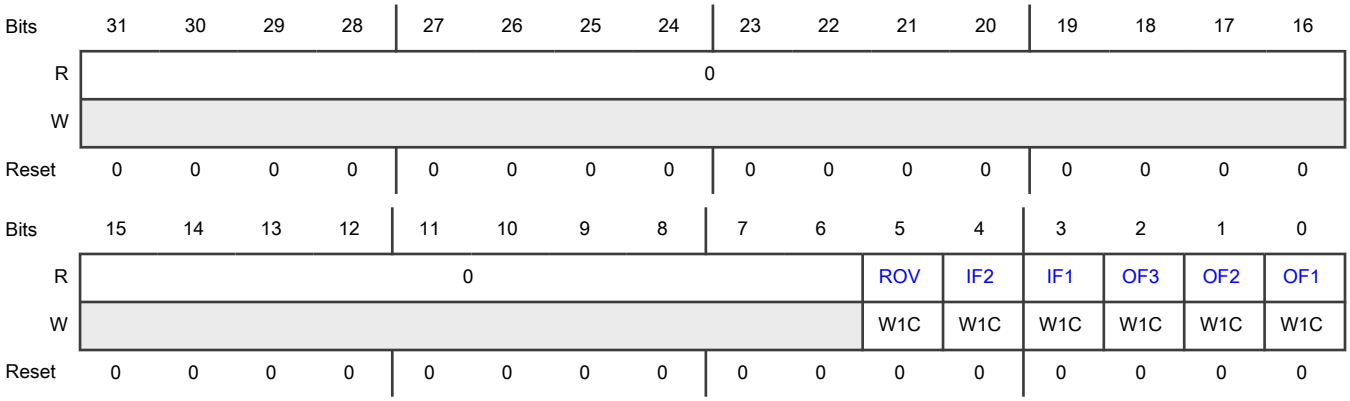
Offset

Register	Offset
SR	8h

Function

Contains flags to indicate that a counter has rolled over and if any event has occurred on the Input Capture and Output Compare channels. Write 1 to clear the flags.

Diagram



Fields

Field	Function
31-6 —	Reserved Write zero to this field. These bits always read as zero.
5 ROV	Rollover Flag Indicates that the counter has reached its maximum possible value and rolled over to 0 (from which the counter continues counting). ROV is set only when the counter has reached FFFF_FFFFh in both Restart and Free-Run modes. 0b - Rollover has not occurred. 1b - Rollover has occurred.
4-3 IFn	Input Capture Flag for Channel n Indicates that a capture event has occurred on Input Capture channel *n*. 0b - Capture event has not occurred. 1b - Capture event has occurred.
2-0 OFn	Output Compare Flag for Channel n Indicates that a compare event has occurred on Output Compare channel *n*. 0b - Compare event has not occurred. 1b - Compare event has occurred.

68.6.1.5 Interrupt (IR)

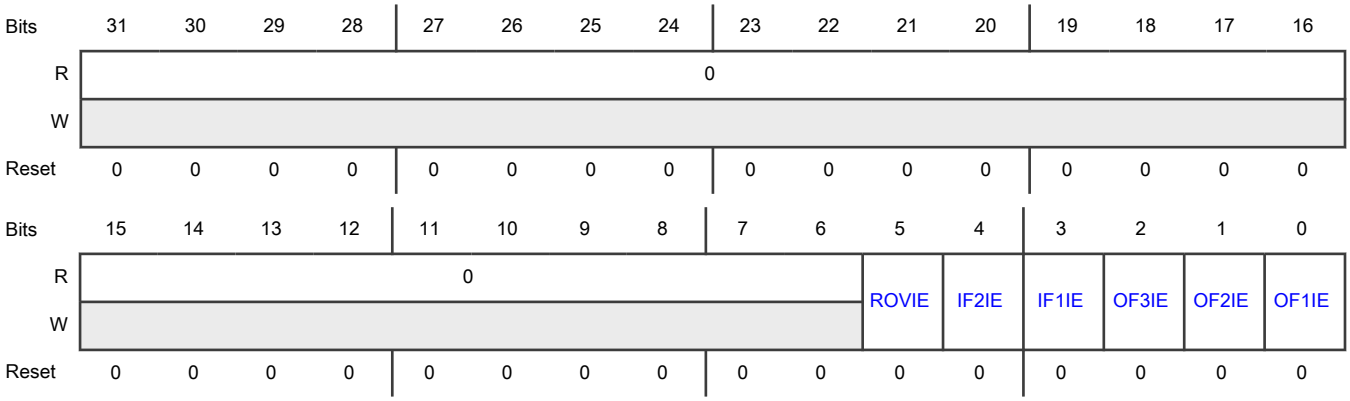
Offset

Register	Offset
IR	Ch

Function

Contains enable bits that control whether interrupts are generated after rollover, input capture, and output compare events.

Diagram



Fields

Field	Function
31-6 —	Reserved Writing a value to this field does not affect GPT operations. These reserved bits always read as zero.
5 ROVIE	Rollover Interrupt Enable Enables the Rollover interrupt. 0b - Disable 1b - Enable
4-3 IFnIE	Input Capture Flag for Channel n Interrupt Enable Enables the Input Capture Flag for Channel *n* interrupt. 0b - Disable 1b - Enable
2-0 OFnIE	Output Compare Flag for Channel n Interrupt Enable Enables the Output Compare Flag for Channel *n* interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable

68.6.1.6 Output Compare (OCR1 - OCR3)

Offset

Register	Offset
OCR1	10h
OCR2	14h
OCR3	18h

Function

Holds the value that determines when a compare event is generated on the corresponding Output Compare Channel *n*.

NOTE

The behavior of OCR1 is different from OCR2 and OCR3:

- A write access to OCR1 while in Restart mode (CR[FRR]=0) resets the GPT counter.
- Writes to OCR1 take effect after one cycle of the module's register bus clock (1 wait state of latency). Reads to the register occur immediately (0 wait states).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COMP															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMP															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 COMP	Compare Value When the counter value equals COMP, a compare event is generated.

68.6.1.7 Input Capture (ICR1 - ICR2)

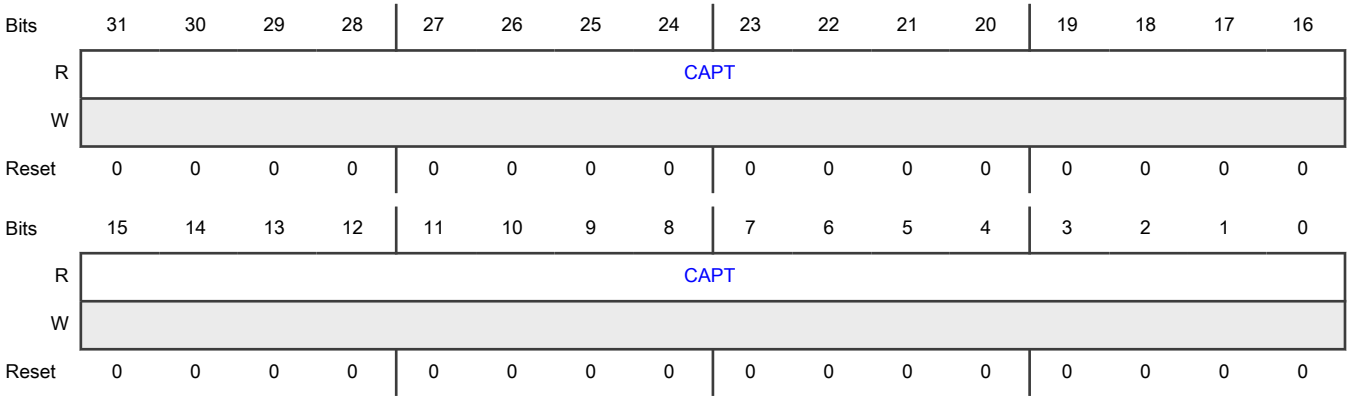
Offset

Register	Offset
ICR1	1Ch
ICR2	20h

Function

Holds the value that was in the counter during the last capture event on the corresponding Input Capture Channel *n*.

Diagram



Fields

Field	Function
31-0 CAPT	Capture Value After a capture event on Input Capture Channel *n* occurs, the current value of the counter is loaded into the corresponding ICR*n*[CAPT].

68.6.1.8 Counter (CNT)

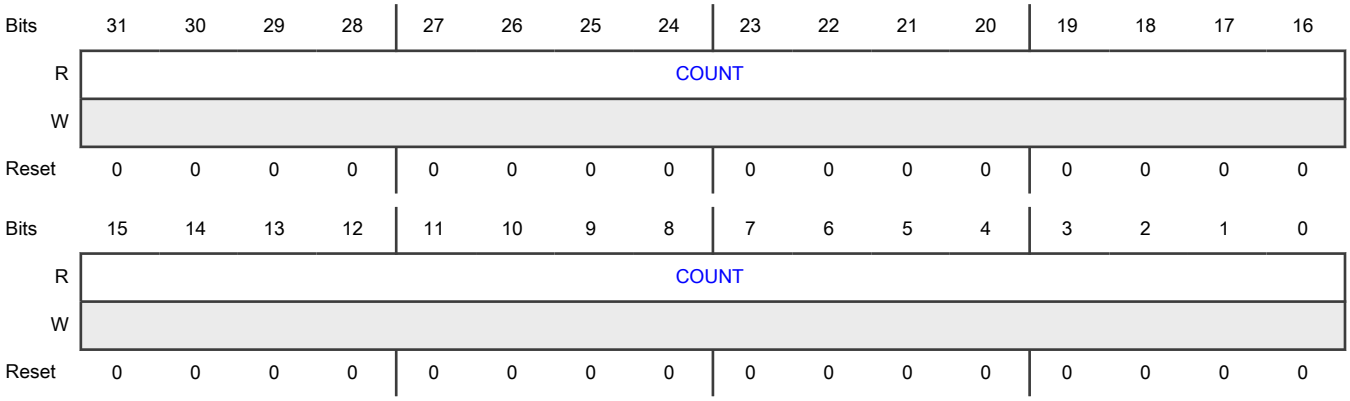
Offset

Register	Offset
CNT	24h

Function

Contains the current value of the main counter. CNT can be read at any time without affecting the GPT counting process.

Diagram



Fields

Field	Function
31-0 COUNT	Counter Value Provides the current value of the GPT counter.

Chapter 69
Low-Power Timer (LPTMR)

69.1 Chip-specific LPTMR Information

Table 1061. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

69.2 LPTMR clocks

The prescaler and glitch filter of the LPTMR module (the module functional clock) can be clocked from one of four sources determined by LPTMRn_PSR[PCS]. The chosen clock must remain enabled if the LPTMR is to continue operating in all required low-power modes. The following table shows the clock assignments for this field.

Table 1062. LPTMR1*n* prescaler/glitch filter clocking options

LPTMR*n*_PSR[PCS]	Prescaler/glitch filter clock number	Chip clock
00	0	ipg_clk_irclk: The frequency depends on configuration of lptmr1_clk_root.
01	1	ipg_clk_1kHz: Tied to the fixed 32 kHz clock
10	2	ipg_clk_32kHz: It is fixed at 32 kHz.
11	3	ipg_clk_erclk: Tied to the fixed 32 kHz clock.

69.3 Overview

You can configure LPTMR to operate as a time counter with an optional prescaler, or as a pulse counter with an optional glitch filter, across all power modes, including low-power modes. It is reset only on Power on Reset (POR) or Low Voltage Detect (LVD), allowing it to be used as a time-of-day counter.

69.3.1 Block diagram

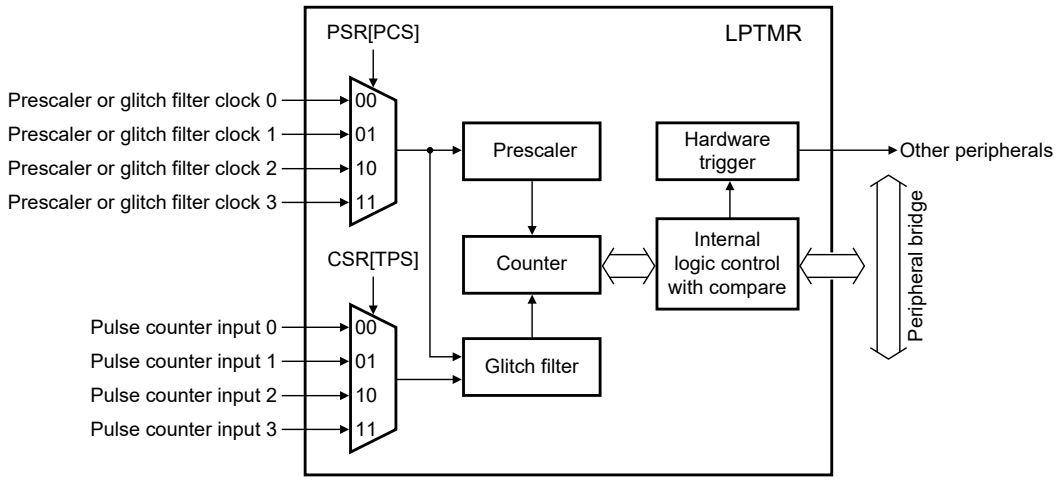


Figure 524. Block diagram

69.3.2 Features

- 32-bit time counter or pulse counter with compare:
  - Optional interrupt that can generate an asynchronous wake-up from any low-power mode
  - Hardware trigger output
  - Counter that supports a free-running mode or reset on compare
- Configurable clock source for prescaler and glitch filter
- Configurable input source for pulse counter (rising-edge or falling-edge)

69.4 Functional description

69.4.1 Low-power modes

In low-power modes, LPTMR continues to operate normally. You can configure LPTMR to exit a low-power mode by generating either an interrupt or a DMA request.

69.4.2 Clocks

The LPTMR prescaler and glitch filter can be clocked by one of the clocks that you configure by using PSR[PCS]. You must enable the clock source before you enable LPTMR.

In Pulse Counter mode, with the glitch filter bypassed, the selected input source directly clocks Counter (CNR), and no other clock source is required. To minimize power in this case, configure the glitch filter clock source for a clock that is disabled.

NOTE

- You may need to configure the clock source that you select in PSR[PCS] for it to remain enabled in low-power modes. Otherwise, LPTMR does not operate in low-power modes.
- The clock source or pulse input source selected for LPTMR must not exceed the maximum frequency of f_{LPTMR} defined in the chip data sheet.

69.4.3 Reset

LPTMR is reset only on Power on Reset (POR) or Low Voltage Detect (LVD). When configuring LPTMR registers, you must initially write to Control Status (CSR) with LPTMR disabled, before configuring Prescaler and Glitch Filter (PSR) and Compare (CMR). Then, you must write 1 to CSR[TEN] as the last step in the initialization. Doing so ensures that LPTMR is configured correctly and the LPTMR counter is reset to zero following a Power on Reset (POR) or Low Voltage Detect (LVD).

69.4.4 Prescaler and glitch filter

The LPTMR prescaler and glitch filter share the same logic, which operates as a prescaler in Time Counter mode and as a glitch filter in Pulse Counter mode.

NOTE

You must not alter the prescaler and glitch filter configuration when LPTMR is enabled.

69.4.4.1 Prescaler enabled

In Time Counter mode, when the prescaler is enabled, the output of the prescaler directly clocks Counter (CNR). When LPTMR is enabled, CNR increments every 2^1 to 2^{16} prescaler clock cycles. After LPTMR is enabled, the first increment of CNR takes an additional one or two prescaler clock cycles because of the synchronization logic.

69.4.4.2 Prescaler bypassed

In Time Counter mode, when the prescaler is bypassed, the selected prescaler clock increments Counter (CNR) on every clock cycle. When LPTMR is enabled, the first increment takes an additional one or two prescaler clock cycles because of the synchronization logic.

69.4.4.3 Glitch filter enabled

In Pulse Counter mode, when the glitch filter is enabled, the output of the glitch filter directly clocks Counter (CNR). When LPTMR is first enabled, the output of the glitch filter is asserted, that is, logic 1 for active-high and logic 0 for active-low. The following table shows the change in glitch filter output with the selected input source.

Table 1063. Glitch filter output with the selected input source

If	Then
The selected input source remains deasserted for at least 2^1 to 2^{15} consecutive prescaler clock rising edges	The glitch filter output also deasserts.
The selected input source remains asserted for at least 2^1 to 2^{15} consecutive prescaler clock rising edges	The glitch filter output also asserts.

NOTE

The input is sampled only on the rising clock edge.

The value of CNR increments each time the glitch filter output asserts. In Pulse Counter mode, the maximum rate at which CNR can increment is once every 2^2 to 2^{16} glitch filter clock edges. When first enabled, the glitch filter waits for an additional one or two glitch filter clock edges because of the synchronization logic.

69.4.4.4 Glitch filter bypassed

In Pulse Counter mode, when the glitch filter is bypassed, the selected input source increments the value of Counter (CNR) every time it asserts. Before LPTMR is first enabled, the selected input source is forced to be asserted. This prevents CNR from incrementing if the selected input source is already asserted when LPTMR is first enabled.

69.4.5 Counter

The value of Counter (CNR) increments by 1 on every:

- Prescaler clock in Time Counter mode, with prescaler bypassed.
- Prescaler output in Time Counter mode, with prescaler enabled.
- Input source assertion in Pulse Counter mode, with glitch filter bypassed.
- Glitch filter output in Pulse Counter mode, with glitch filter enabled.

CNR is reset when LPTMR is disabled or if the counter register overflows. If CSR[TFC] = 0, then CNR is also reset whenever CSR[TCF] = 1.

When the core is halted in Debug mode:

- CNR continues incrementing if configured for Pulse Counter mode.
- CNR stops incrementing if configured for Time Counter mode.

You cannot initialize CNR but can read it at any time. On each read of CNR, you must first write a value to it. This synchronizes and registers the current value of CNR into a temporary register. The contents of the temporary register are returned on each read of CNR.

When reading CNR, the bus clock must be at least two times faster than the rate at which the LPTMR counter is incrementing; otherwise, incorrect data may be returned.

69.4.6 Compare

After the next Counter (CNR) increment (when its value is equal to that of Compare (CMR)), the following events occur:

- CSR[TCF] is read as 1b.
- LPTMR generates an interrupt if CSR[TIE] is 1 as well.
- LPTMR generates a hardware trigger.
- LPTMR writes 0 to CNR if CSR[TFC] is 0.

When LPTMR is enabled, you can modify the value of CMR only when CSR[TCF] is 1. When updating CMR, you must write to it and clear CSR[TCF] before the LPTMR counter increments past the new LPTMR compare value.

NOTE

When LPTMR is enabled in Time Counter mode, the first increment takes an additional one or two clock cycles because of the synchronization logic. This results in the first compare (and therefore interrupt and hardware trigger) occurring slightly later. A faster prescaler clock or larger prescaler value minimizes this impact.

69.4.7 Interrupt

LPTMR generates an interrupt whenever CSR[TIE] and CSR[TCF] are 1. CSR[TCF] is cleared by disabling LPTMR or writing a logic 1 to it.

You can modify the value of CSR[TIE] and write 1 to CSR[TCF] when LPTMR is enabled.

LPTMR generates an interrupt asynchronously to the system clock. The interrupt can be used to generate a wake-up from any low-power mode, provided LPTMR is enabled as a wake-up source.

69.4.8 Hardware trigger

The LPTMR hardware trigger asserts at the same time CSR[TCF] is set and can be used to trigger hardware events in other peripherals without your intervention. The hardware trigger is always enabled.

Table 1064. Hardware trigger

When	Then
CMR[COMPARE] and CSR[TFC] are 0	The LPTMR hardware trigger asserts on the first compare and does not deassert.
CMR[COMPARE] is set to a nonzero value, or if CSR[TFC] = 1	The LPTMR hardware trigger asserts on each compare and deasserts on the following increment of Counter (CNR).

69.5 External signals

Table 1065. External signals

Signal	Description		Direction
LPTMR_ALT*n*	Pulse Counter Input LPTMR can select one of the input pins to be used in Pulse Counter mode.		Input
	State meaning	Assertion—If configured for Pulse Counter mode with an active-high input, assertion causes Counter (CNR) to increment. Deassertion—If configured for Pulse Counter mode with an active-low input, deassertion causes CNR to increment.	
	Timing	Assertion or deassertion may occur at any time; input may assert asynchronously to the bus clock.	

69.6 Initialization

Perform the following procedure to initialize LPTMR:

1. Configure Control Status (CSR) for the selected mode and pin configuration, when CSR[TEN] is 0. This resets the counter and clears the flag.
2. Configure Prescaler and Glitch Filter (PSR) with the selected clock source and prescaler or glitch filter configuration.
3. Configure Compare (CMR) with the selected compare point.
4. Write 1 to CSR[TEN] to enable LPTMR.

69.7 Application information

69.7.1 Application 1: Generate an interrupt every 100 ms using 32.768 kHz clock source

1. Disable LPTMR by writing 0 to CSR[TEN].
2. Select a 32.768 kHz clock source by configuring PSR[PCS].
3. Bypass the prescaler and glitch filter by writing 1 to PSR[PBYP].
4. Assert an interrupt every 3277 cycles by configuring CMR[COMPARE] = 0CCCh.
5. Enable LPTMR by writing 1 to CSR[TEN].
6. Enable the LPTMR interrupt by writing 1 to CSR[TIE].

NOTE

This is just an example. See the chip-specific SPC information for the clocks supported on a given chip.

69.7.2 Application 2: Generate an interrupt once a minute using 32.768 kHz clock source

1. Disable LPTMR by writing 0 to CSR[TEN].
2. Select a 32.768 kHz clock source by configuring PSR[PCS].
3. Select the prescaler to divide the prescaler clock by 32768 to increment the counter once a second by configuring PSR[PRESCALE] = 00h.
4. Assert an interrupt every 60 seconds by configuring CMR[COMPARE] = 003Bh.
5. Enable LPTMR by writing 1 to CSR[TEN].
6. Enable the LPTMR interrupt by writing 1 to CSR[TIE].

NOTE

This is just an example. See the chip-specific SPC information for the clocks supported on a given chip.

69.8 Memory map and register definition

NOTE

The LPTMR registers are reset only on Power on Reset (POR) or Low Voltage Detect (LVD). See Reset for more information.

69.8.1 LPTMR register descriptions

69.8.1.1 LPTMR memory map

LPTMR1 base address: 4430_0000h

LPTMR2 base address: 424D_0000h

LPTMR3 base address: 42CD_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Status (CSR)	32	RW	0000_0000h
4h	Prescaler and Glitch Filter (PSR)	32	RW	0000_0000h
8h	Compare (CMR)	32	RW	0000_0000h
Ch	Counter (CNR)	32	RW	0000_0000h

69.8.1.2 Control Status (CSR)

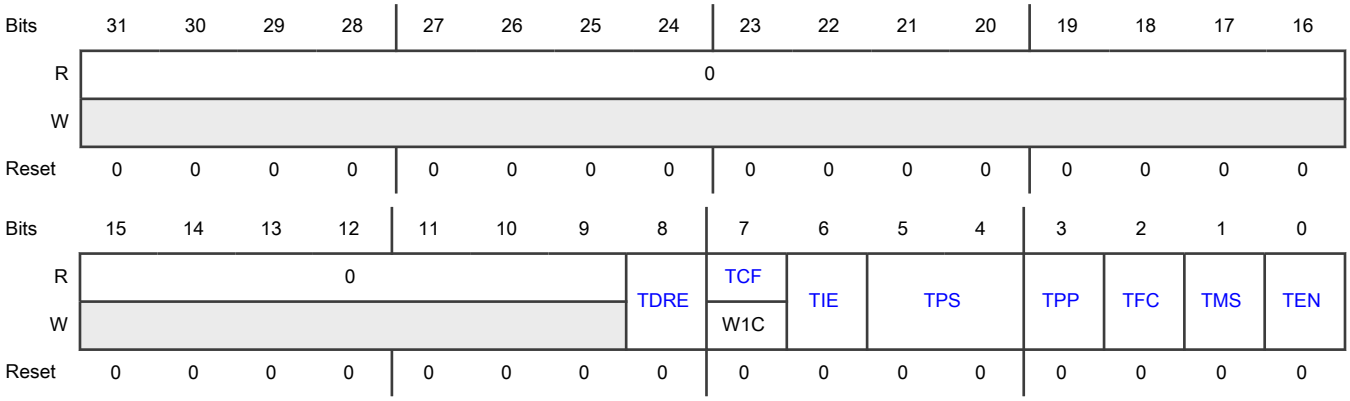
Offset

Register	Offset
CSR	0h

Function

Controls various features of LPTMR.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 TDRE	Timer DMA Request Enable Enables the timer DMA request. When TDRE is 1, the LPTMR DMA request is generated whenever CSR[TCF] is also set. Then, CSR[TCF] is cleared after the DMA controller completes execution. 0b - Disable 1b - Enable
7 TCF	Timer Compare Flag Compares the timer. TCF sets on the next Counter (CNR) increment when LPTMR is enabled and Counter (CNR) equals Compare (CMR). TCF is cleared when LPTMR is disabled or a logic 1 is written to it. **NOTE** You must clear this flag before enabling the timer interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes. When reading 0b - CNR ≠ (CMR + 1) 1b - CNR = (CMR + 1) When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 TIE	Timer Interrupt Enable Enables the timer interrupt. If TIE is 1, then LPTMR generates an interrupt if CSR[TCF] is 1 as well. 0b - Disable 1b - Enable
5-4 TPS	Timer Pin Select Configures the input source to be used in Pulse Counter mode. The input connections vary by chip. For details, see the chip configuration information about connections to these inputs. You must modify this field only when LPTMR is disabled. 00b - Input 0 01b - Input 1 10b - Input 2 11b - Input 3
3 TPP	Timer Pin Polarity Configures the polarity of the input source in Pulse Counter mode. If TPP is 0, then the pulse counter input source is active-high, and Counter (CNR) increments on the rising edge. If TPP is 1, then the pulse counter input source is active-low, and CNR increments on the falling edge. You must modify this field only when LPTMR is disabled. 0b - Active-high 1b - Active-low
2 TFC	Timer Free-Running Counter Specifies when the counter resets. If TFC is 0, Counter (CNR) resets on the count cycle following Counter (CNR) becoming equal to Compare (CMR). If TFC is 1, CNR resets on overflow. In both cases, CSR[TCF] sets to 1 on the cycle after CNR and CMR match. You must modify this field only when LPTMR is disabled. 0b - Reset when TCF asserts 1b - Reset on overflow
1 TMS	Timer Mode Select Configures the mode of LPTMR. You must modify this field only when LPTMR is disabled. 0b - Time Counter 1b - Pulse Counter
0 TEN	Timer Enable Enables the LPTMR timer. If TEN is 0, it resets the LPTMR internal logic, including CNR[COUNTER] and CSR[TCF]. If TEN is 1, LPTMR is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Do not alter CSR[5:1] when writing 1 to this field. 0b - Disable 1b - Enable

69.8.1.3 Prescaler and Glitch Filter (PSR)

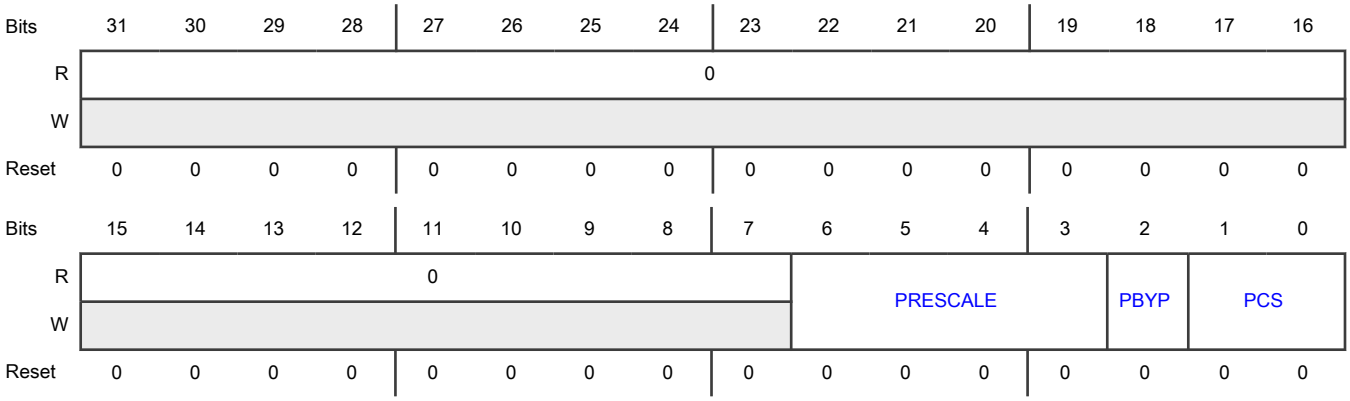
Offset

Register	Offset
PSR	4h

Function

Configures features related to the prescaler and glitch filter.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-3 PRESCALE	Prescaler and Glitch Filter Value Configures the size of the prescaler in Time Counter mode and the width of the glitch filter in Pulse Counter mode. The width of the glitch filter can vary by one cycle because of the pulse counter input synchronization. You must modify this field only when LPTMR is disabled. 0000b - Prescaler divides the prescaler clock by 2; glitch filter does not support this configuration

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0001b - Prescaler divides the prescaler clock by 4; glitch filter recognizes change on input pin after two rising clock edges 0010b - Prescaler divides the prescaler clock by 8; glitch filter recognizes change on input pin after four rising clock edges 0011b - Prescaler divides the prescaler clock by 16; glitch filter recognizes change on input pin after eight rising clock edges 0100b - Prescaler divides the prescaler clock by 32; glitch filter recognizes change on input pin after 16 rising clock edges 0101b - Prescaler divides the prescaler clock by 64; glitch filter recognizes change on input pin after 32 rising clock edges 0110b - Prescaler divides the prescaler clock by 128; glitch filter recognizes change on input pin after 64 rising clock edges 0111b - Prescaler divides the prescaler clock by 256; glitch filter recognizes change on input pin after 128 rising clock edges 1000b - Prescaler divides the prescaler clock by 512; glitch filter recognizes change on input pin after 256 rising clock edges 1001b - Prescaler divides the prescaler clock by 1024; glitch filter recognizes change on input pin after 512 rising clock edges 1010b - Prescaler divides the prescaler clock by 2048; glitch filter recognizes change on input pin after 1024 rising clock edges 1011b - Prescaler divides the prescaler clock by 4096; glitch filter recognizes change on input pin after 2048 rising clock edges 1100b - Prescaler divides the prescaler clock by 8192; glitch filter recognizes change on input pin after 4096 rising clock edges 1101b - Prescaler divides the prescaler clock by 16,384; glitch filter recognizes change on input pin after 8192 rising clock edges 1110b - Prescaler divides the prescaler clock by 32,768; glitch filter recognizes change on input pin after 16,384 rising clock edges 1111b - Prescaler divides the prescaler clock by 65,536; glitch filter recognizes change on input pin after 32,768 rising clock edges
2 PBYP	Prescaler and Glitch Filter Bypass Controls the clocking of Counter (CNR). If PBYP is 0, the output of the prescaler or glitch filter clocks CNR. If PBYP is 1, the selected prescaler clock in Time Counter mode, or else the selected input source in Pulse Counter mode, directly clocks CNR. You must modify this field only when LPTMR is disabled. 0b - Prescaler and glitch filter enable 1b - Prescaler and glitch filter bypass
1-0	Prescaler and Glitch Filter Clock Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
PCS	Selects the clock to be used by the LPTMR prescaler and glitch filter. In Time Counter mode, PCS selects the input clock to the prescaler. In Pulse Counter mode, PCS selects the input clock to the glitch filter. See the chip configuration details for information on connections to these inputs. You must modify this field only when LPTMR is disabled. 00b - Clock 0 01b - Clock 1 10b - Clock 2 11b - Clock 3

69.8.1.4 Compare (CMR)

Offset

Register	Offset
CMR	8h

Function

Configures the compare values to Counter (CNR) (see Compare for more information).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COMPARE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMPARE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 COMPARE	Compare Value Configures the compare values to Counter (CNR).

Table continues on the next page...

Field	Function
	On the next CNR increment, if LPTMR is enabled and Counter (CNR) equals Compare (CMR), then: 1. LPTMR writes 1 to CSR[TCF]. 2. The hardware trigger asserts until the next time CNR increments. If CMR = 0, the hardware trigger remains asserted until LPTMR is disabled. If LPTMR is enabled, you must modify the value of CMR only if CSR[TCF] is 1.

69.8.1.5 Counter (CNR)

Offset

Register	Offset
CNR	Ch

Function

Specifies counter values (see Counter for more information).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COUNTER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COUNTER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 COUNTER	Counter Value Contains the current value of the LPTMR counter at the time you last wrote to this register.

Chapter 70
Timer/PWM Module (TPM)

70.1 Chip-specific TPM Information

Table 1066. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

70.2 Overview

TPM is a 4-channel timer that controls electric motor and power management applications by supporting:

- Input capture
- Output comparison
- Generation of PWM signals

An asynchronous clock that can remain enabled in low-power modes clocks the counter, compare, and capture registers.

70.2.1 Block diagram

TPM uses one input/output (I/O) pin per channel: CH*n* (TPM channel *n*) where *n* is the channel number.

The central component of TPM is a 32-bit counter with a programmable final value. Its counting can be up or up-down.

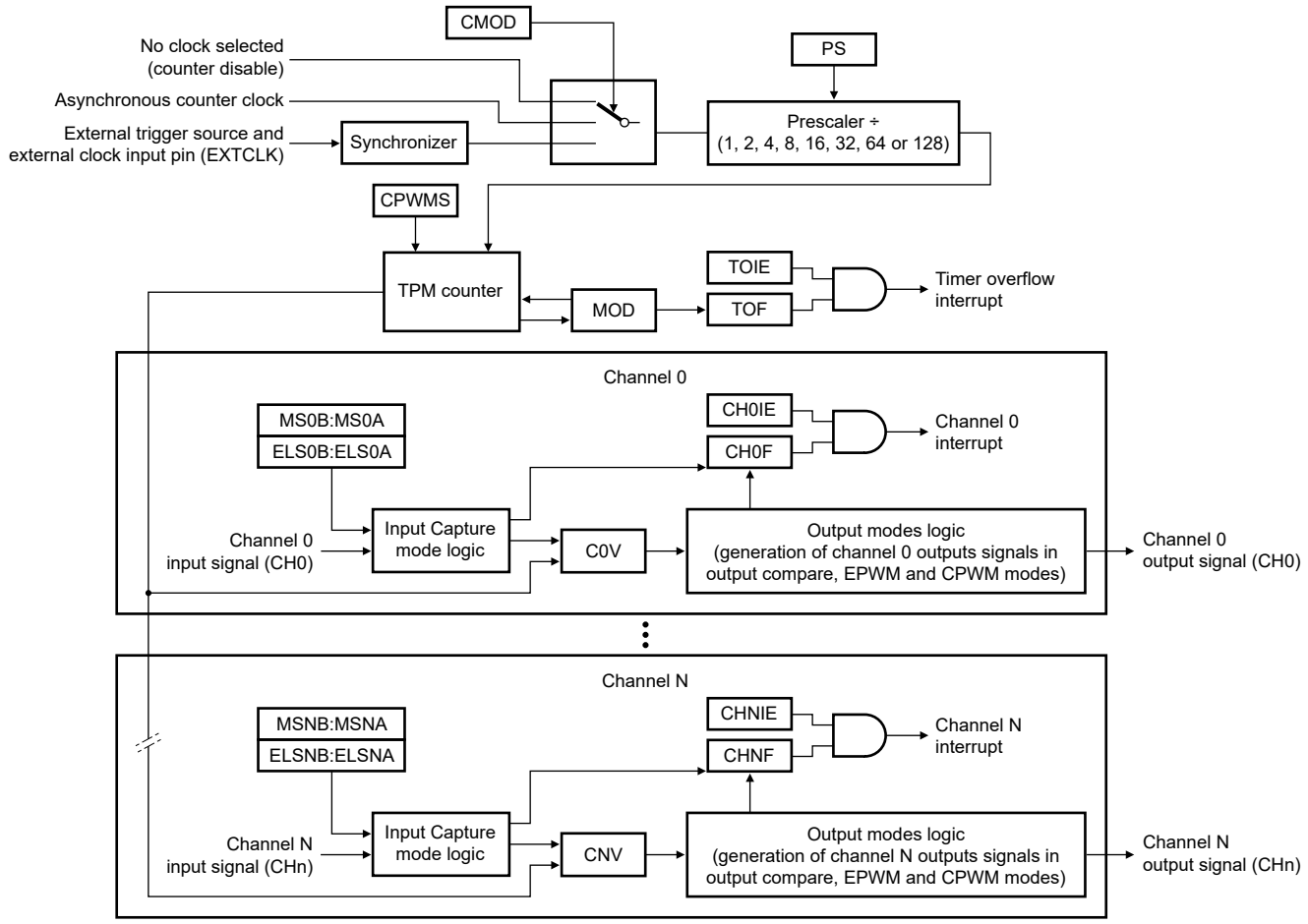


Figure 525. Block diagram

70.2.2 Features

- Supports selectable clock modes:
  - Can increment on every edge of the asynchronous counter clock
  - Can increment on the rising edge of an external clock input that is synchronized to the asynchronous counter clock
- Includes a prescaler divided by 1, 2, 4, 8, 16, 32, 64, or 128
- Includes a 32-bit TPM counter:
  - Can be a free-running counter or a modulo counter
  - The counting can be up or up-down
- Includes 4 channels that can be configured as follows:
  - Input Capture mode: the capture can occur on rising edges, falling edges, or both edges.
  - Output Compare mode: the output signal can be set, cleared, pulsed, or toggled on the match.
  - Edge-Aligned or Center-Aligned PWM mode for all channels.
- Supports the generation of an interrupt and DMA request per channel
- Supports the generation of an interrupt and DMA request when the counter overflows

- Supports selectable trigger input to either reset the counter to 0 or else cause the counter to start incrementing
  - The counter can also optionally stop incrementing on counter overflow.
- Supports the generation of hardware triggers when the counter overflows and per channel

70.3 Functional description

70.3.1 Counter

TPM has a 32-bit counter that the channels use for either input or output modes. The counter updates from the selected clock after it has been divided by the prescaler. The TPM counter has these modes of operation:

- Up counting
- Up-down counting

70.3.1.1 Counter Clock mode

SC[CMOD] either disables the TPM counter or enables one of the three possible clock sources for the TPM counter. After any reset, SC[CMOD] becomes 0, which disables the TPM counter. You can configure SC[CMOD] for one of the following counter clock sources:

- Asynchronous counter clock
- External clock input pin
- External trigger source

You can read or write to SC[CMOD] at any time. Disabling the TPM counter by writing 0 to SC[CMOD] does not affect the TPM counter value or other registers, but the TPM counter clock domain must acknowledge this disabling action before SC[CMOD] becomes 0.

The external clock input and external trigger source pass through a synchronizer. The TPM counter clock clocks this synchronizer to ensure that counter transitions are properly aligned to counter clock transitions. Therefore, to meet Nyquist criteria, and also considering jitter, the frequency of the external clock source must be less than half of the counter clock frequency.

70.3.1.2 Prescaler

The counter clock source that you select passes through a prescaler that is a 7-bit counter. SC[PS] defines the value of the prescaler. The following figure shows an example of the prescaler counter and TPM counter.

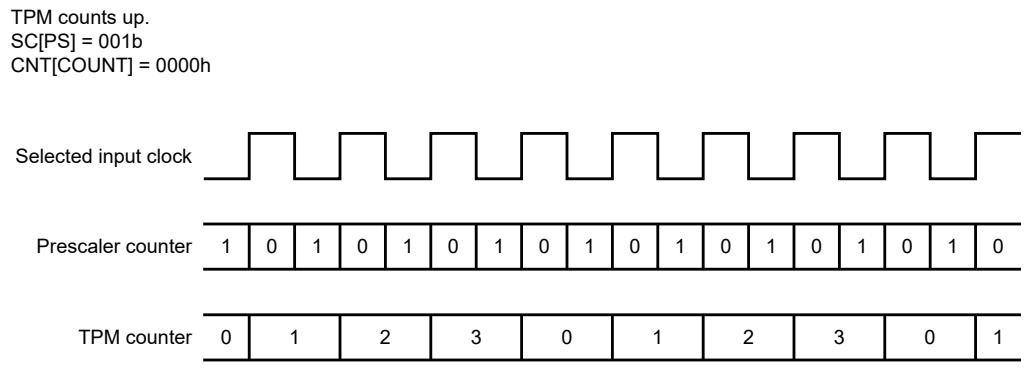


Figure 526. Example of the prescaler counter

70.3.1.3 Up counting

Up counting is enabled when you write 0 to SC[CPWMS].

In this mode, the TPM counter loads with value 0 and increments until it reaches the value defined in MOD[MOD], then the counter reloads with 0.

The TPM period when using up counting is (MOD[MOD] + 0001h) × period of the TPM counter clock.

SC[TOF] becomes 1 when the TPM counter changes from the value defined in MOD[MOD] to 0.

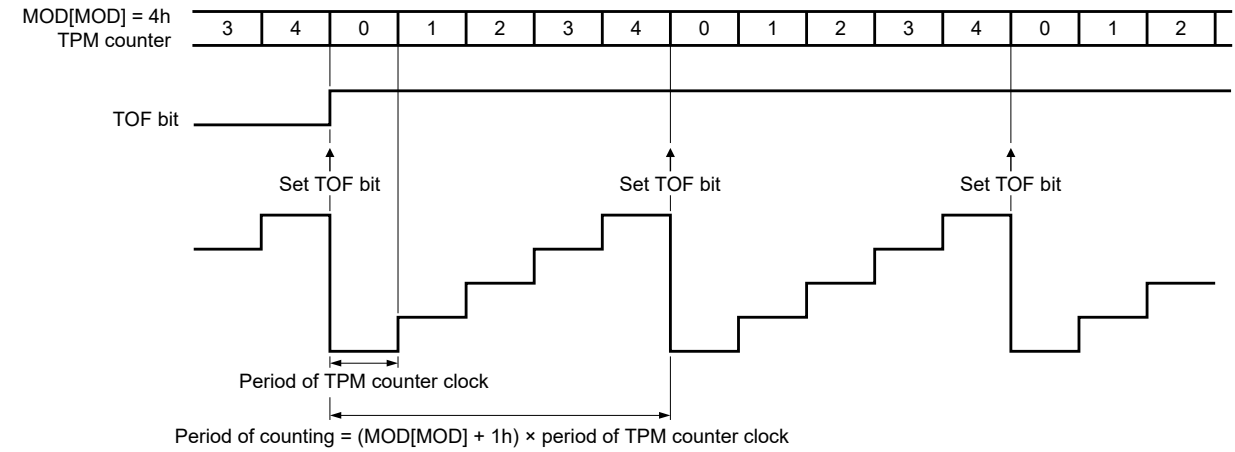


Figure 527. Example of TPM up counting

NOTE

- Configuring MOD[MOD] = 0 is a redundant condition. In this case, the TPM counter is always equal to the value of MOD[MOD], and SC[TOF] remains 1 for each rising edge of the TPM counter clock.
- Configuring MOD[MOD] = 1 and SC[PS] = 0 causes TPM to attempt to set SC[TOF] every second TPM counter clock. Because of synchronization delays between the TPM counter clock and the bus clock, you must write 1 to SC[TOF] to clear it two counter clock cycles before SC[TOF] can become 1 again.

70.3.1.4 Up-down counting

You enable up-down counting when you write 1 to SC[CPWMS]. This mode does not support configuring the value of Modulo (MOD) to be less than 2.

In this mode, the TPM counter loads with value 0 and increments until it reaches the value defined in MOD[MOD]. Then the counter starts decrementing until it returns to 0 and the up-down counting restarts.

The TPM period when using up-down counting is 2 × MOD[MOD] × period of the TPM counter clock.

TPM writes 1 to SC[TOF] when the TPM counter changes from the value defined in MOD[MOD] to (MOD[MOD] – 1).

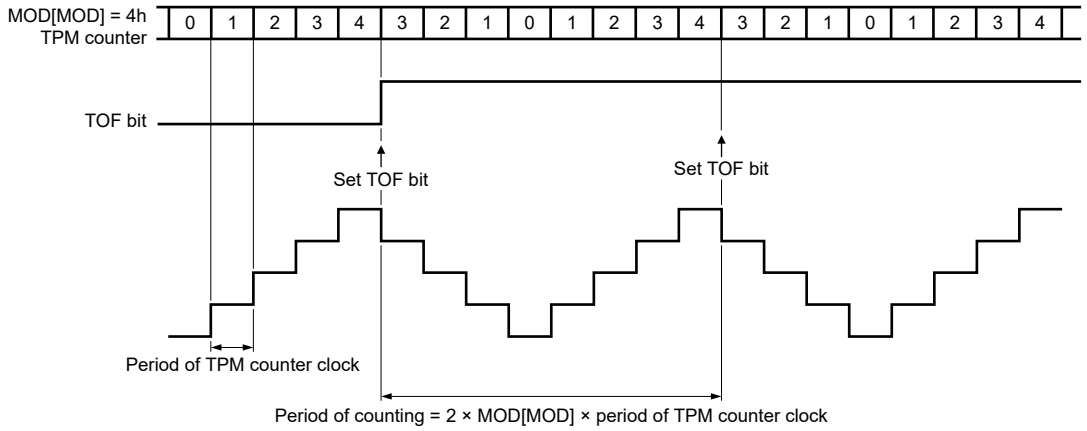


Figure 528. Example of up-down counting

70.3.1.5 Counter reset

Any write to CNT[COUNT] resets the TPM counter and forces the channel outputs to their initial values (except for channels in Output Compare mode).

70.3.1.6 Counter trigger

You can configure the TPM counter to start, stop, or reset in response to a hardware trigger input. The trigger input synchronizes with the asynchronous counter clock, so there is a three-counter-clock delay between the trigger assertion and counter response.

- When CONF[CSOT] = 1, the counter does not start incrementing until the trigger input detects a rising edge.
- When CONF[CSOO] = 1, the counter stops incrementing whenever SC[TOF] = 1. The counter does not increment again unless:
  - It is disabled.
  - CONF[CSOT] = 1 and the trigger input detects a rising edge.
- When CONF[CROT] = 1, the counter resets to 0, as if an overflow occurred, whenever the trigger input detects a rising edge. Any PWM channels are updated to their reload state.
- When CONF[CPOT] = 1, the counter pauses incrementing whenever you assert the trigger input. The counter continues incrementing when the trigger input negates. PWM output channels are forced to their default state.

You can configure the polarity of the external input trigger with CONF[TRGPOL].

When you enable an internal trigger source, the trigger input is selected from one or more channel input capture events. The input capture filters are used with the internal trigger sources. You can use POL[POL*n*] fields to invert the polarity of the input channels.

The following restrictions apply with input capture channel sources:

- When CONF[CSOT] = 1, the counter starts incrementing only on a rising edge on the channel input, if C*n*SC[ELS*n*A] = 1.
- When CONF[CROT] = 1, the counter resets to 0 on either edge of the channel input, as you configure it using C*n*SC[ELS*n*B] and C*n*SC[ELS*n*A].
- When CONF[CPOT] = 1, the counter pauses incrementing whenever you assert the channel input. PWM output channels are forced to their default state.

70.3.2 Registers updated from write buffers

70.3.2.1 MOD register update

If SC[CMOD] = 0, then Modulo (MOD) updates whenever you write to MOD[MOD].

If SC[CMOD] ≠ 0, then the MOD register updates according to SC[CPWMS]:

- If CPWM mode is disabled (SC[CPWMS] = 0), then the MOD register updates whenever you write to MOD[MOD], and the TPM counter value changes from the value defined in MOD[MOD] to 0.
- If CPWM mode is enabled (SC[CPWMS] = 1), then the MOD register updates whenever you write to MOD[MOD], and the TPM counter value changes from the value defined in MOD[MOD] to (MOD[MOD] – 1).

70.3.2.2 C*n*V register update

If SC[CMOD] = 0, then the C*n*V register updates whenever you write to C*n*V[VAL].

If SC[CMOD] ≠ 0, then the C*n*V register updates according to the selected mode:

- If you select Output Compare mode, then the C*n*V register updates on the next TPM counter increment (end of the prescaler counting) after you write to C*n*V[VAL].
- If you select EPWM mode, then the C*n*V register updates after you write to C*n*V[VAL], and the TPM counter value changes from the value defined in MOD[MOD] to 0.
- If you select CPWM mode, then the C*n*V register updates after you write to C*n*V[VAL], and the TPM counter value changes from the value defined in MOD[MOD] to (MOD[MOD] – 1).

70.3.3 Peripheral triggers

See the chip-specific TPM information for more details.

70.3.3.1 Output triggers

TPM generates output triggers for the counter and for each channel that you can use to trigger events in other peripherals. The counter trigger asserts whenever SC[TOF] is 1 and remains asserted until the next increment.

Each TPM channel generates both a pre-trigger output and a trigger output.

1. The pre-trigger output asserts whenever C*n*SC[CH*n*F] sets.
2. The trigger output asserts on the first counter increment after the pre-trigger asserts.
3. Both the trigger and pre-trigger negate on the first counter increment after the trigger asserts.

When (COMBINE[COMBINE*n*] = 1) in Output Compare mode, the pre-trigger output for both channel *n* and channel *n* + 1 asserts when the CH*n*F flag is set and negates when the CH(*n* + 1)F flag is set. The trigger continues to assert on the first counter increment, after the pre-trigger asserts and negates at the same time as the pre-trigger negation.

70.3.3.2 Input trigger

You can configure the TPM input trigger to perform the following operations with configurable polarity:

- Increment the counter on trigger rising edge
- Start incrementing the counter on trigger rising edge
- Reset or reload the counter on trigger rising edge
- Pause the counter while trigger high

The TPM input trigger is synchronized and must remain asserted for at least two counter clock cycles, so that the input trigger can be detected.

70.3.4 Modes of operations

70.3.4.1 Input Capture mode

Input Capture mode is enabled when:

- (SC[CPWMS] = 0)
- (MS*n*B:MS*n*A = 0:0)
- (ELS*n*B:ELS*n*A ≠ 0:0)

When a selected edge occurs on the channel input:

- The C*n*V register captures the current value of the TPM counter.
- It also writes 1 to C*n*SC[CH*n*F] and generates a channel interrupt if C*n*SC[CH*n*IE] = 1 (see Figure 529).

When you configure a channel for input capture, the CH*n* pin is an edge-sensitive input. ELS*n*B:ELS*n*A control fields determine which edge, falling or rising, triggers the input-capture event. To detect the input signal correctly and to meet Nyquist criteria for signal sampling, the maximum frequency for the channel input signal is the counter clock divided by 4.

Input Capture mode ignores any writes to the C*n*V register.

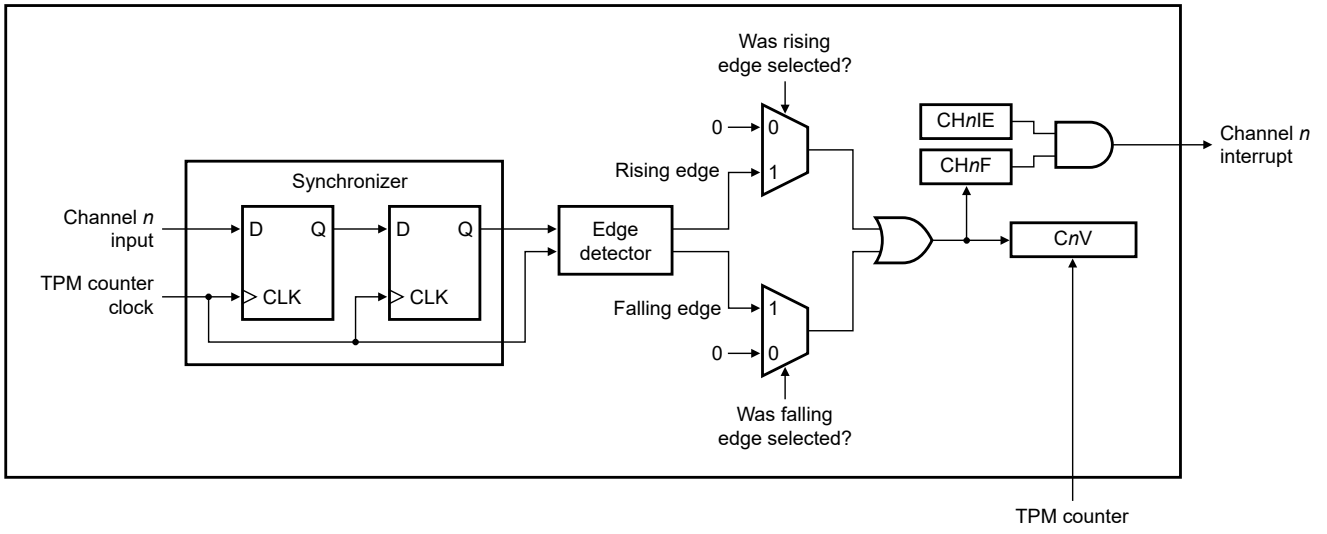


Figure 529. Input Capture mode

The CH*n*F flag is set on the third rising edge of the TPM counter clock after a valid edge occurs on the channel input.

70.3.4.1.1 Input capture filter

The input capture filter function is only available in Input Capture or Software Compare mode when Quadrature Decoder mode is enabled.

First, the TPM counter clock synchronizes the input signal. After synchronization, the input signal enters the filter block (see Figure 530).

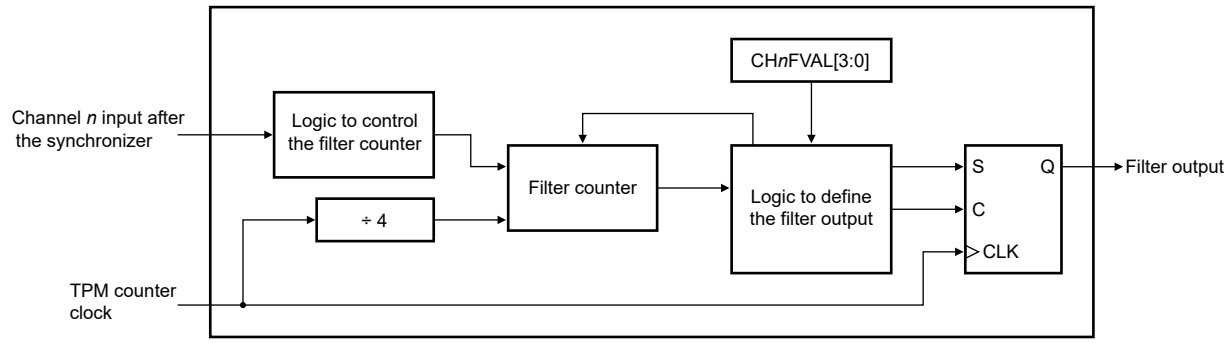


Figure 530. Channel input filter

When there is a state change in the input signal, TPM resets the counter, and it starts counting up. As long as the new state is stable on the input, the counter continues to increment. When the counter is equal to (CH*n*FVAL[3:0] × 4), the state change of the input signal is validated.

TPM resets the counter if the opposite edge appears on the input signal before it is validated. At the next input transition, the counter starts counting again. Any pulse that is less than the minimum value selected by (CH*n*FVAL[3:0] × 4 counter clocks) is considered as a glitch and does not pass through the filter. Figure 531 shows a timing diagram of the input filter.

The filter function is disabled when CH*n*FVAL[3:0] = 0. In this case, the input signal is delayed by 2 rising edges of the counter clock. If CH*n*FVAL[3:0] ≠ 0000, then the input signal is delayed by the minimum pulse width (CH*n*FVAL[3:0] × 4 system clocks) plus a further 3 rising edges of the system clock: two rising edges to the synchronizer, plus one more to the edge detector. In other words, the CH*n*F flag is set for (3 + (4 × CH*n*FVAL[3:0])) counter clock periods after a valid edge occurs on the channel input (see Figure 531).

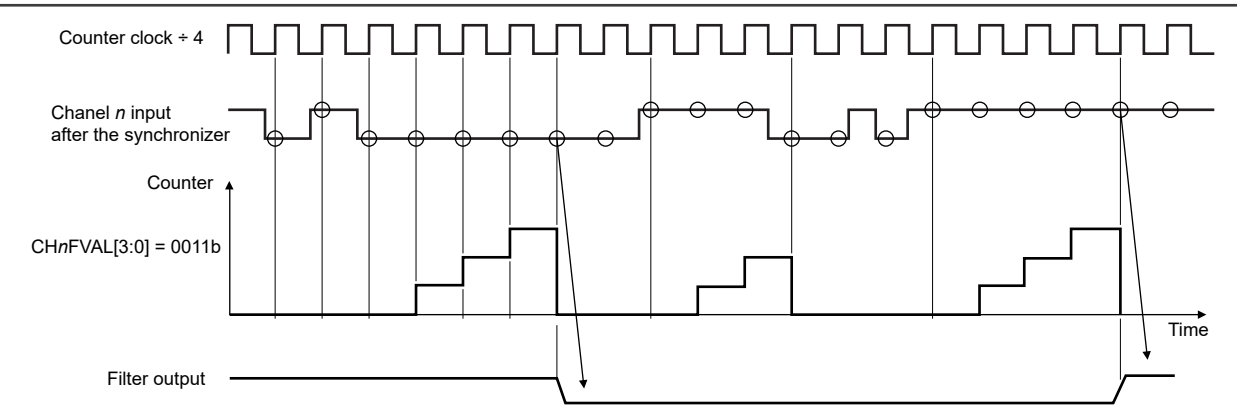


Figure 531. Channel input filter example

70.3.4.2 Output Compare mode

Output Compare mode is enabled when (SC[CPWMS] = 0) and (MS*n*B:MS*n*A = X:1).

In this mode, TPM can generate timed pulses with programmable:

- Position
- Polarity
- Duration
- Frequency

When the TPM counter matches the value defined in C*n*V[VAL] of an output compare channel, then channel *n* output can be set, cleared, or toggled if MS*n*B = 0. If MS*n*B = 1, then channel *n* output is pulsed high or low (as long as the TPM counter matches the value defined in C*n*V[VAL]).

When you initially configure a channel to Output Compare mode, the channel output is updated with its negated value (logic 0 for set/toggle/pulse high and logic 1 for clear/pulse low).

The channel *n* match (TPM counter = C*n*V[VAL]) writes 1 to C*n*SC[CH*n*F] and generates a channel *n* interrupt (if C*n*SC[CH*n*IE] = 1).

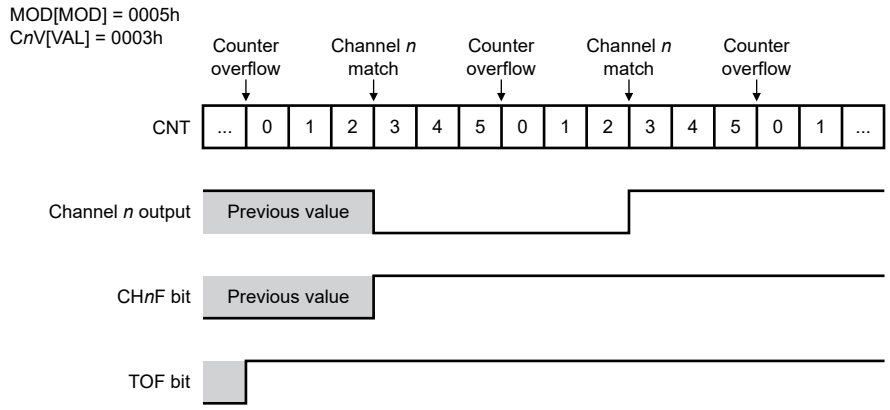


Figure 532. Example of Output Compare mode when a match toggles the channel output

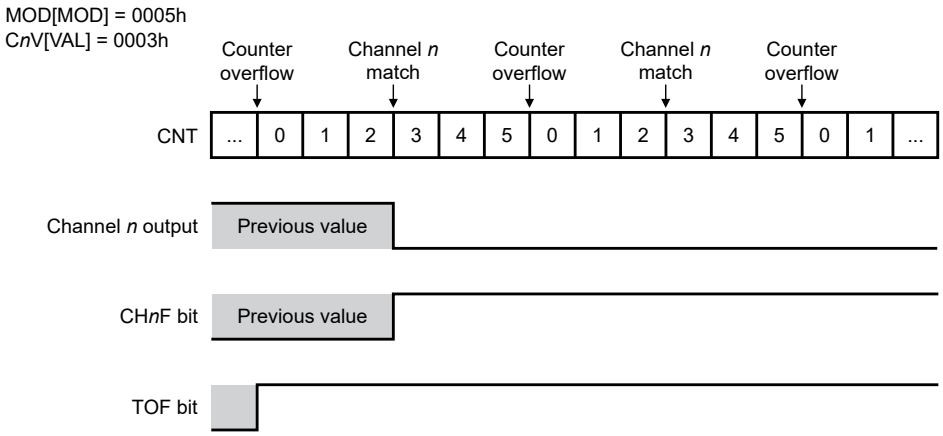


Figure 533. Example of Output Compare mode when a match clears the channel output

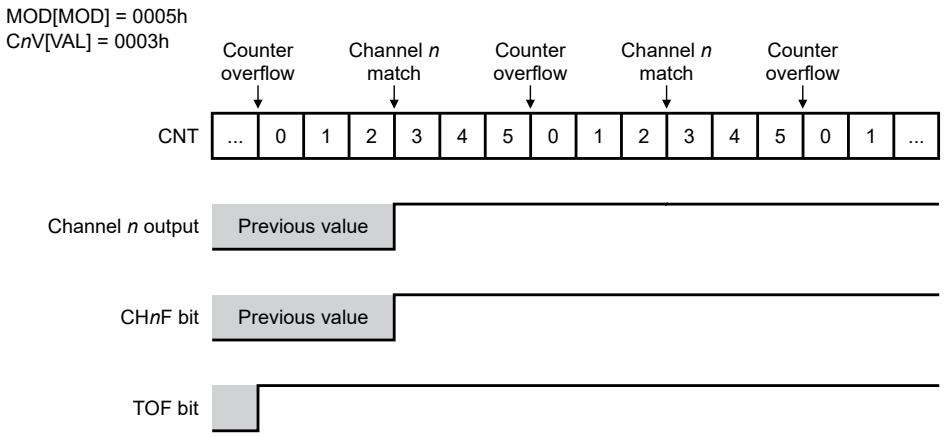


Figure 534. Example of Output Compare mode when a match sets the channel output

You can also use Output Compare mode with:

- (SC[CPWMS] = 0) and (MS*n*B:MS*n*A = 0:1)
- (SC[CPWMS] = 0) and (ELS*n*B:ELS*n*A = 0:0)

This condition is known as Software Compare mode. This mode sets the CH*n*F flag and generates a channel *n* interrupt (if CH*n*IE = 1) when the TPM counter reaches the value defined in the C*n*V register; however, TPM does not modify or control the channel *n* output.

70.3.4.3 Edge-Aligned PWM (EPWM) mode

EPWM mode is enabled when (SC[CPWMS] = 0) and (MS*n*B:MS*n*A = 1:0). The EPWM period is determined by (MOD[MOD] + 0001h) and the pulse width (duty cycle) is determined by C*n*V[VAL].

The channel *n* match (TPM counter = C*n*V[VAL]) writes 1 to C*n*SC[CH*n*F] and generates a channel *n* interrupt (if C*n*SC[CH*n*IE] = 1) at the end of the pulse width (see Figure 535).

This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within TPM.

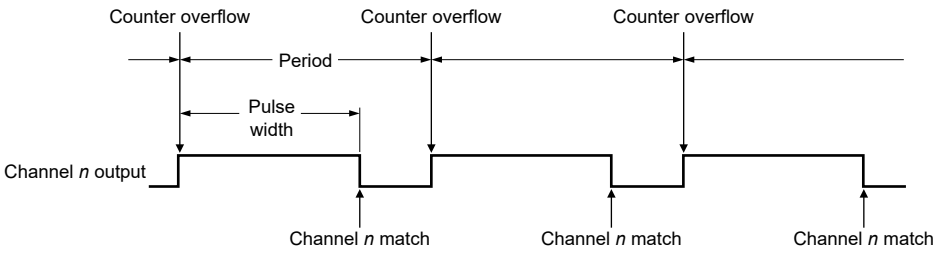


Figure 535. EPWM period and pulse width with ELS*n*B:ELS*n*A = 1:0

If (ELS*n*B:ELS*n*A = 1:0) then the channel *n* output is forced high at the counter overflow or reload (when the TPM counter loads with 0), and it is forced low at the channel *n* match (TPM counter = C*n*V[VAL]). The channel *n* output is forced high when you first enable or whenever you pause the TPM counter (see Figure 536).

If (ELS*n*B:ELS*n*A = 0:0) then the channel *n* output is forced high at the counter overflow or reload (when the TPM counter loads with 0), and it is forced low at the channel *n* match (TPM counter = C*n*V[VAL]). The channel *n* output is forced low when you first enable or whenever you pause the TPM counter.

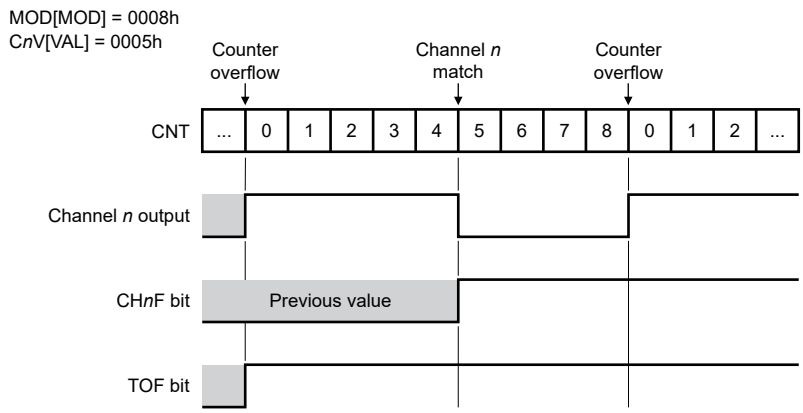


Figure 536. EPWM signal with ELS*n*B:ELS*n*A = 1:0

If (ELS*n*B:ELS*n*A = 0:1) then the channel *n* output is forced low at the counter overflow or reload (when the TPM counter loads with 0), and it is forced high at channel *n* match (TPM counter = C*n*V[VAL]). Channel *n* output is forced low when you first enable or whenever you pause the TPM counter (see Figure 537).

If (ELS*n*B:ELS*n*A = 1:1) then the channel *n* output is forced low at the counter overflow or reload (when the TPM counter loads with 0), and it is forced high at the channel *n* match (TPM counter = C*n*V[VAL]). Channel *n* output is forced high when you first enable or whenever you pause the TPM counter.

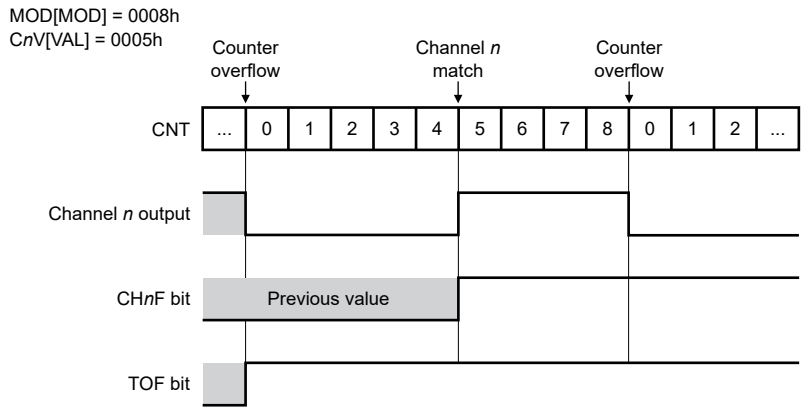


Figure 537. EPWM signal with ELS*n*B:ELS*n*A = 0:1

If (C*n*V[VAL] = 0000h) then the channel *n* output is a 0% duty cycle EPWM signal.

If (C*n*V[VAL] > MOD[MOD]), then the channel *n* output is a 100% duty cycle EPWM signal and the CH*n*F flag is not set because there is never a channel *n* match. Therefore, MOD[MOD] must be less than FFFFFFFFh to get a 100% duty cycle EPWM signal.

70.3.4.4 Center-Aligned PWM (CPWM) mode

CPWM mode is enabled when (SC[CPWMS] = 1) and (MS*n*B:MS*n*A = 1:0).

The CPWM pulse width (duty cycle) is determined by 2 × C*n*V[VAL] and the period is determined by 2 × MOD[MOD] (see Figure 538). You must keep MOD[MOD] in the range of 0001h to 7FFFFFFF because values outside this range can produce ambiguous results.

In CPWM mode, the TPM counter counts up until it reaches the value defined in MOD[MOD], and then counts down until it reaches 0.

The channel *n* match (TPM counter = C*n*V[VAL]) sets C*n*SC[CH*n*F] and generates a channel *n* interrupt (if C*n*SC[CH*n*IE] = 1) when TPM is counting down (at the begin of the pulse width), and when TPM is counting up (at the end of the pulse width).

This type of PWM signal is called center-aligned because the pulse width centers for all channels are at the same place when the TPM counter is 0.

The other channel modes are not designed to be used with the up-down counter when SC[CPWMS] = 1. Therefore, you must use all TPM channels in CPWM mode when SC[CPWMS] = 1.

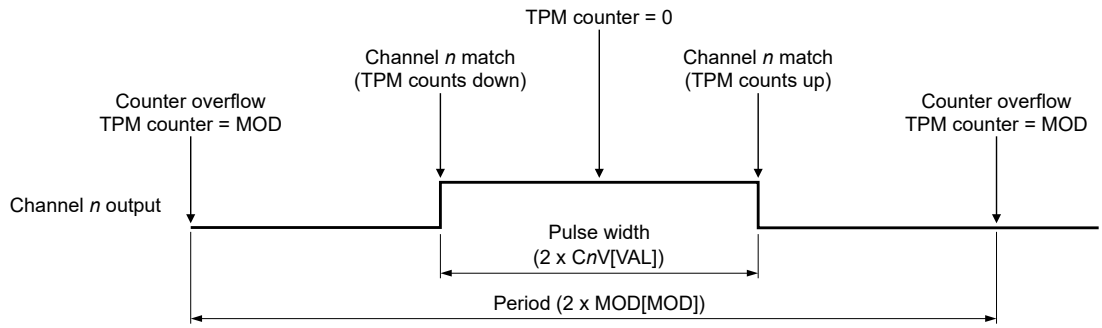


Figure 538. CPWM period and pulse width with ELS*n*B:ELS*n*A = 1:0

If (ELS*n*B:ELS*n*A = 1:0) then the channel *n* output is forced high at the channel *n* match (TPM counter = C*n*V) when counting down, and it is forced low at the channel *n* match when counting up. The channel *n* output is forced high when you first enable or reload the TPM counter, or whenever you pause it (see Figure 539).

If (ELS*n*B:ELS*n*A = 0:0) then the channel *n* output is forced high at the channel *n* match (TPM counter = C*n*V) when counting down, and it is forced low at the channel *n* match when counting up. The channel *n* output is forced low when you first enable or reload the TPM counter, or whenever you pause it.

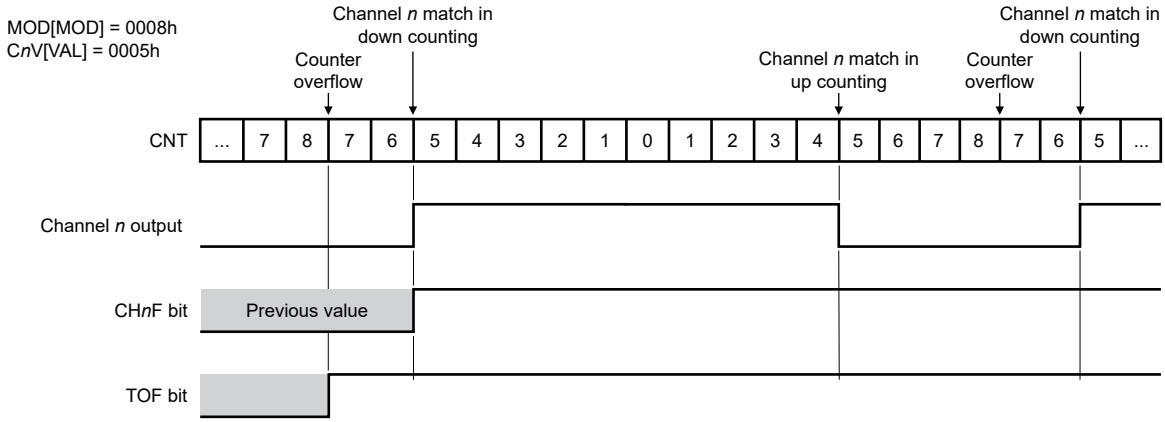


Figure 539. CPWM signal with ELS*n*B:ELS*n*A = 1:0

If (ELS*n*B:ELS*n*A = 0:1) then the channel *n* output is forced low at the channel *n* match (TPM counter = C*n*V) when counting down, and it is forced high at the channel *n* match when counting up. The channel *n* output is forced low when you first enable or reload the TPM counter, or whenever you pause it (see Figure 540).

If (ELS*n*B:ELS*n*A = 1:1) then the channel *n* output is forced low at the channel *n* match (TPM counter = C*n*V) when counting down, and it is forced high at the channel *n* match when counting up. The channel *n* output is forced high when you first enable or reload the TPM counter, or whenever you pause it.

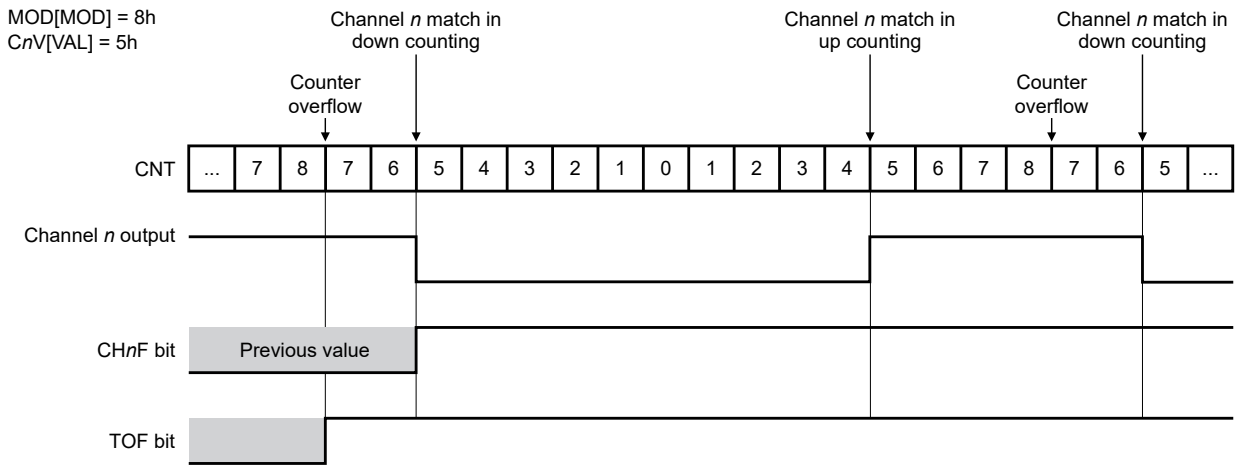


Figure 540. CPWM signal with ELS*n*B:ELS*n*A = 0:1

If (C*n*V[VAL] = 0000h) then the channel *n* output is a 0% duty cycle CPWM signal.

If (C*n*V[VAL] > MOD[MOD]) then the channel *n* output is a 100% duty cycle CPWM signal, although the CH*n*F flag is set when the counter changes from incrementing to decrementing. Therefore, MOD[MOD] must be less than 7FFFFFFF in order to get a 100% duty cycle CPWM signal.

70.3.4.5 Combine PWM mode

Combine PWM mode is enabled when:

- MS*n*B:MS*n*A = 1:0
- COMBINE[COMBINE*n*] = 1
- QDCTRL[QUADEN] = 0
- SC[CPWMS] = 0

In Combine PWM mode, an even channel *n* and adjacent odd channel *n* + 1 are combined to generate a PWM signal in the channel *n* output.

Determine the PWM period and pulse width according to Equation 34.

```
PWM period = MOD[MOD] + 0001h
PWM pulse width (duty cycle) = |C(n + 1)V[VAL] - CnV[VAL]|
```

Equation 34. PWM period and pulse width

- If (C*n*V[VAL] = C(*n* + 1)V[VAL] = 0000h), then the channel *n* output has a 0% duty cycle.
- If (C*n*V[VAL] = C(*n* + 1)V[VAL] > MOD[MOD]), then the channel *n* output has a 100% duty cycle. CH*n*F flag is not set because there is never a channel *n* match.

Therefore, MOD[MOD] must be less than FFFFh in order to get a 100% duty cycle signal.

The channel *n* match (TPM counter = C*n*V[VAL]) writes 1 to C*n*SC[CH*n*F] and generates a channel *n* interrupt (if C*n*SC[CH*n*IE] = 1). The channel *n* + 1 match (TPM counter = C(*n* + 1)V[VAL]), sets the CH(*n* + 1)F flag and generates a channel *n* + 1 interrupt, if CH(*n* + 1)IE = 1.

If channel *n* (ELS*n*B:ELS*n*A = 0:1), then the channel *n* output is forced low at the beginning of the period (TPM counter = 0) and at the channel *n* + 1 match (TPM counter = C(*n* + 1)V[VAL]). It is forced high at the channel *n* match (TPM counter = C*n*V[VAL]).

If channel *n* (ELS*n*B:ELS*n*A = 1:0), then the channel *n* output is forced high at the beginning of the period (TPM counter = 0) and at the channel *n* + 1 match (TPM counter = C(*n* + 1)V[VAL]). It is forced low at the channel *n* match (TPM counter = C*n*V). See Figure 541 for pictorial representation.

When COMBINE[COMSWAP*n*] = 1, then the channel *n* output is forced low or high at the beginning of the period (TPM counter = 0) and at the channel *n* match (TPM counter = C*n*V[VAL]). It is forced high or low at the channel *n* + 1 match (TPM counter = C(*n* + 1)V[VAL]).

Channel *n* + 1 generates the same output as the channel *n* output, but the channel *n* + 1 (ELS*n*B:ELS*n*A) configuration controls the output polarity.

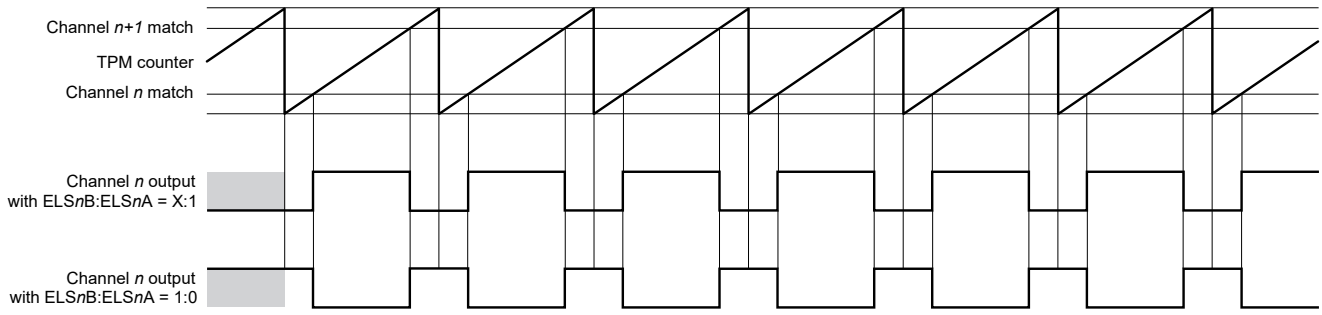


Figure 541. Combine mode

The following figures illustrate generation of PWM signals using the Combine mode.

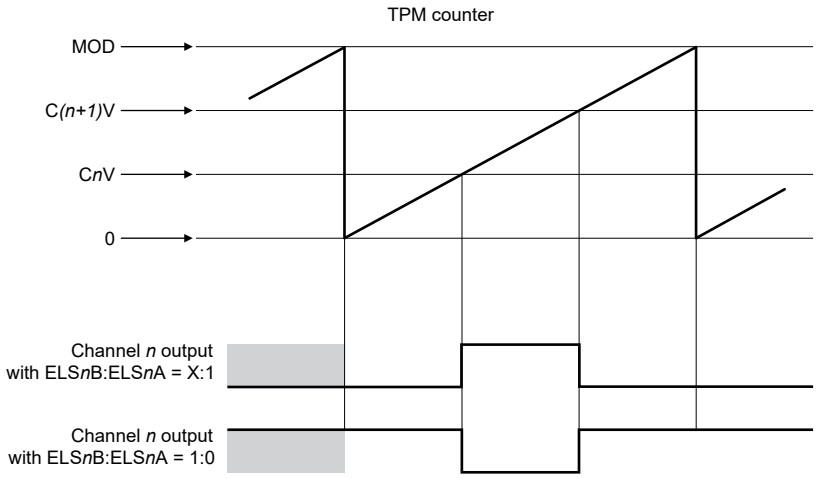


Figure 542. Channel *n* output if (C*n*V < MOD) and (C(*n* + 1)V < MOD) and (C*n*V < C(*n* + 1)V)

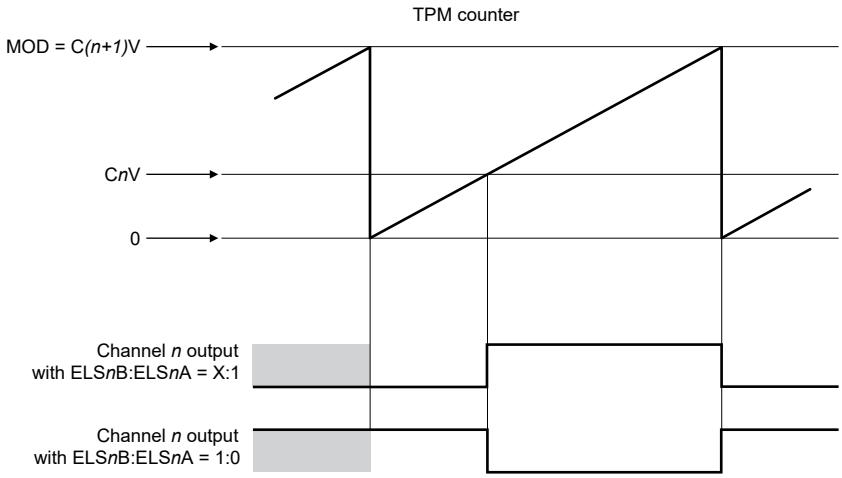


Figure 543. Channel *n* output if (C*n*V < MOD) and (C(*n* + 1)V = MOD)

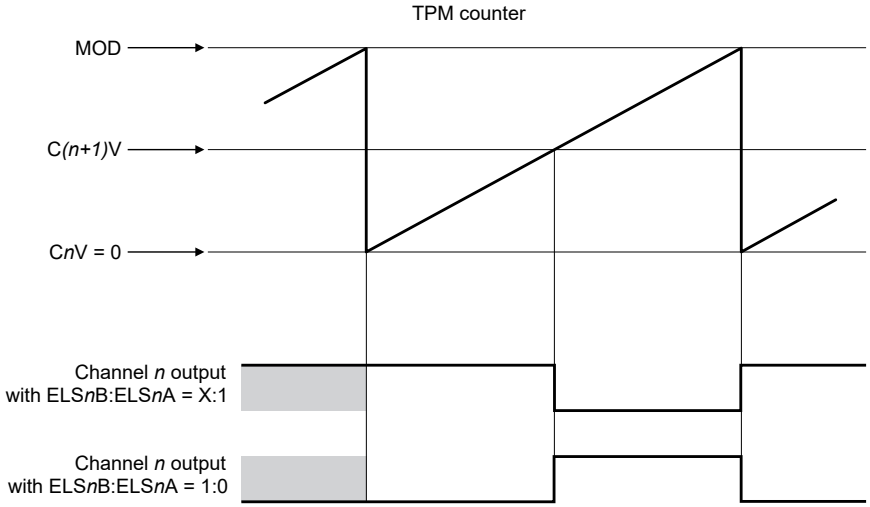


Figure 544. Channel *n* output if (C*n*V = 0) and (C(*n* + 1)V < MOD)

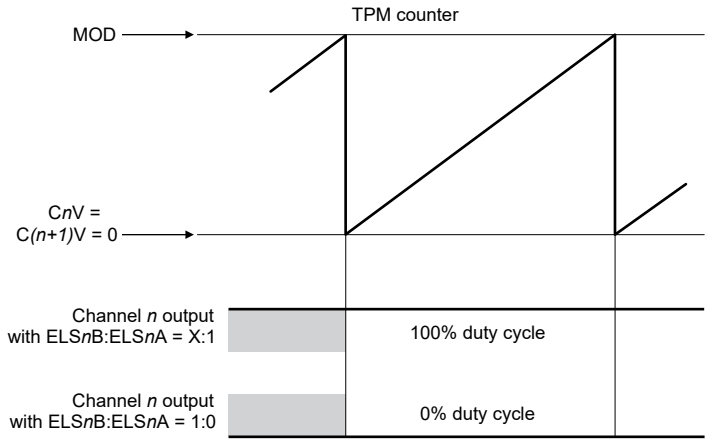


Figure 545. Channel (n) output if (C*n*V = C(*n* + 1)V = 0)

70.3.4.5.1 Dead time insertion

Dead time insertion is enabled in PWM Combine mode for each pair of channels when CH*n*FVAL and CH(*n* + 1)FVAL are non-zero. The dead time delay that is used for each TPM channel is defined as (CH(*n* + 1)FVAL[3:0] × 64) + (CH*n*FVAL[3:0] × 4).

The dead time delay insertion ensures that two complementary signals (even channel *n* and odd channel *n* + 1) do not drive the active state at the same time.

If ELS*n*B:ELS*n*A = 0:1, ELS(*n* + 1)B:ELS(*n* + 1)A = 1:0, COMBINE[COMSWAP*n*] = 0 and dead time is enabled, then:

- When a channel *n* match (TPM counter = C*n*V) occurs, the channel *n* output remains at the low value until the end of the dead time delay, when the channel *n* output is set.
- When a channel *n* + 1 match (TPM counter = C(*n* + 1)V) occurs, the channel *n* + 1 output remains at the low value until the end of the dead time delay, when the channel *n* + 1 output is set (see Figure 546).

If ELS*n*B:ELS*n*A = 1:0, ELS(*n*+1)B:ELS(*n*+1)A = 0:1, COMBINE[COMSWAP*n*] = 1, and dead time is enabled, then:

- When a channel *n* + 1 match (TPM counter = C(*n* + 1)V) occurs, the channel (*n* + 1) output remains at the low value until the end of the dead time delay, when channel *n* + 1 output is set.
- When a channel *n* match (TPM counter = C*n*V) occurs, the channel *n* output remains at the low value until the end of the dead time delay, when channel *n* output is set (see Figure 547).

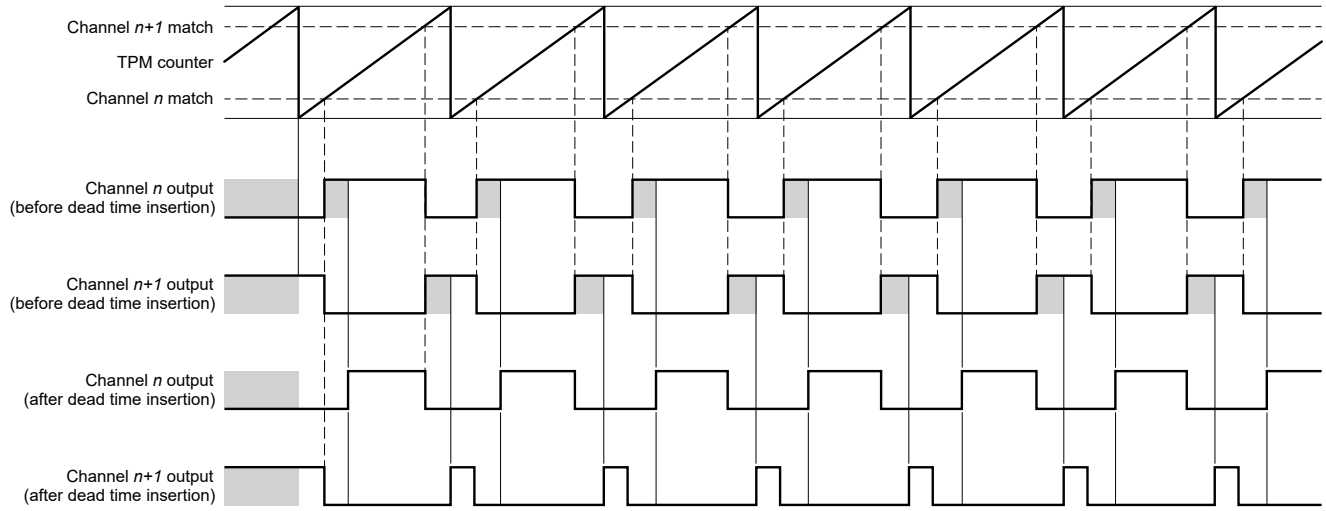


Figure 546. Dead time insertion with ELS*n*B:ELS*n*A = 0:1, ELS(*n* + 1)B:ELS(*n* + 1)A = 1:0, COMBINE[COMSWAP*n*] = 0

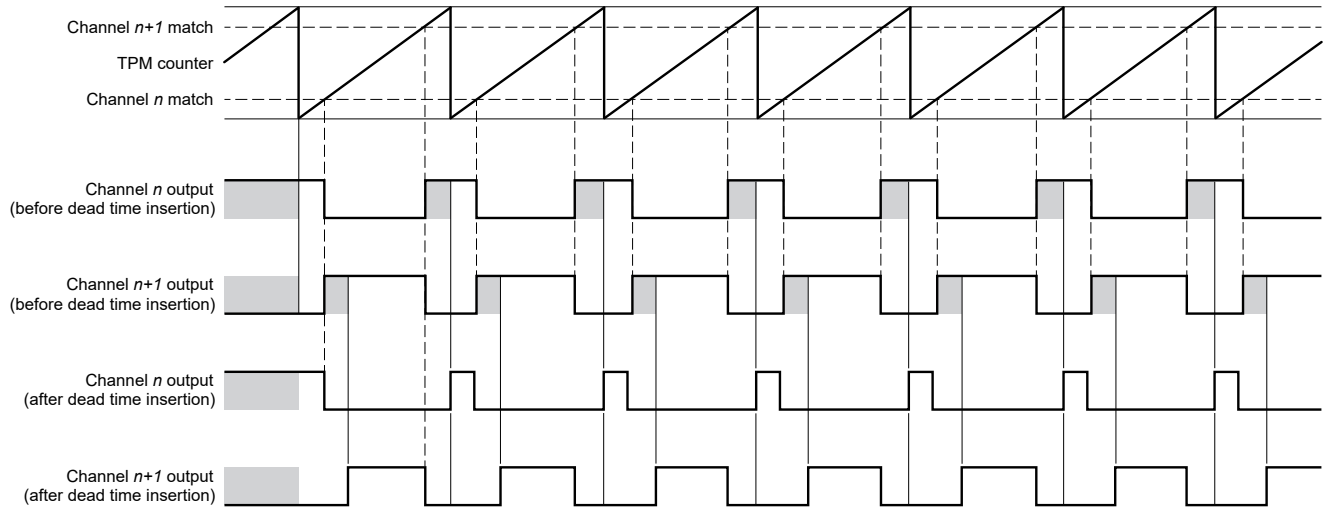


Figure 547. Dead time insertion with ELS*n*B:ELS*n*A = 1:0, ELS(*n* + 1)B:ELS(*n* + 1)A = 0:1, COMBINE[COMSWAP*n*] = 1

70.3.4.6 Combine Input Capture mode

Combine Input Capture mode is enabled if:

- (COMBINE[COMBINE*n*] = 1)
- (MS*n*B:MS*n*A = 0:0)
- (ELS*n*B:ELS*n*A ≠ 0:0)

This mode allows you to measure the pulse width of the signal at the input of channel *n* of a channel pair. The channel *n* filter can be active in this mode.

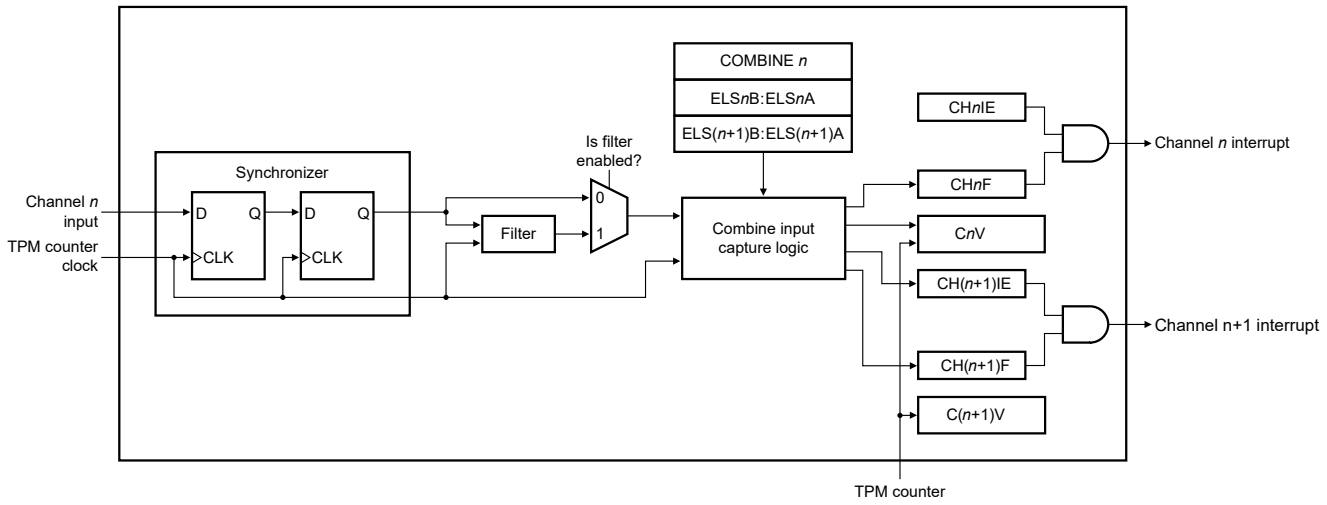


Figure 548. Block diagram

ELS*n*B:ELS*n*A fields select the edge that channel *n* captures. ELS(*n* + 1)B:ELS(*n* + 1)A fields select the edge that channel *n* + 1 captures.

This mode uses only the channel *n* input and ignores the channel *n* + 1 input. If COMBINE[COMSWAP*n*] = 1, then it uses only the channel *n* + 1 input and ignores the channel *n* input.

Detection of the edge selected by channel *n* fields at channel *n* input writes 1 to C*n*SC[CH*n*F] and generates a channel *n* interrupt (if C*n*SC[CH*n*IE] = 1). Detection of the edge selected by channel *n* + 1 fields at channel *n* input sets the CH(*n* + 1)F flag and generates the channel *n* + 1 interrupt (if CH(*n* + 1)IE = 1).

The C*n*V register stores the value of the TPM counter when the channel *n* input detects the edge selected by channel *n*. The C(*n* + 1)V register stores the value of the TPM counter when a channel *n* input detects the edge selected by channel *n* + 1.

NOTE

- Channel *n* fields are:
  - CH*n*F
  - CH*n*IE
  - MS*n*A
  - ELS*n*B
  - ELS*n*A
- Channel *n* + 1 fields are:
  - CH(*n* + 1)F
  - CH(*n* + 1)IE
  - MS(*n* + 1)A
  - ELS(*n* + 1)B
  - ELS(*n* + 1)A
- Use the Combine Input Capture mode with (ELS*n*B:ELS*n*A = 0:1 or 1:0) and (ELS(*n* + 1)B:ELS(*n* + 1)A = 0:1 or 1:0).

70.3.4.7 Quadrature Decoder mode

Quadrature Decoder mode is enabled if (QDCTRL[QUADEN] = 1). This mode uses channel 0 (phase A) and channel 1 (phase B) input signals to control the TPM counter increment and decrement (see Figure 549).

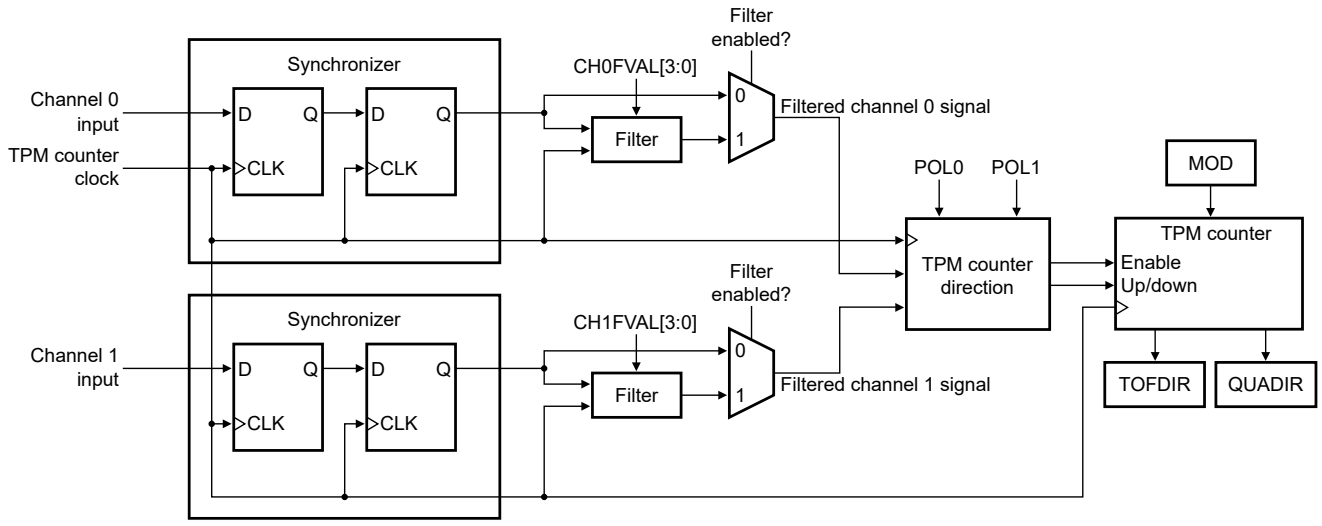


Figure 549. Block diagram

You can configure the input filter and polarity for channel 0 and channel 1 inputs in Quadrature Decode mode using Filter Control (FILTER) and Channel Polarity (POL).

NOTE

When Quadrature Decoder mode is enabled, then channel 0 and channel 1 input signals clock the TPM counter. Therefore, in this mode, you can only use Software Compare mode for channel 0 and channel 1, and you can only use Input Capture or Output Compare mode for other TPM channels.

QDCTRL[QUADMODE] defines the encoding mode that you use in Quadrature Decoder mode.

- If QDCTRL[QUADMODE] = 1, then count and direction encoding mode is enabled.
- If QDCTRL[QUADMODE] = 0, then phase encoding mode is enabled.

In count and direction encoding mode, the channel 1 input value indicates the counting direction, and the channel 0 input value defines the counting rate. A rising edge at the channel 0 input signal updates the TPM counter (see Figure 550).

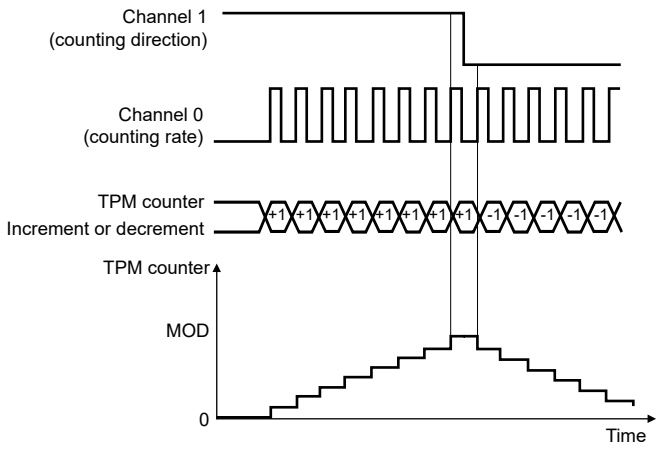


Figure 550. Quadrature Decoder—count and direction encoding mode

In phase encoding mode, the relationship between channel 0 and channel 1 signals indicates the counting direction, and channel 0 and channel 1 signals define the counting rate. An edge at either channel 0 or channel 1 signals updates the TPM counter (see Figure 551).

If POL[POL0] = 0 and POL[POL1] = 0, then:

- The TPM counter increment happens at:
  - A rising edge at channel 0 signal, and channel 1 signal is at logic 0
  - A rising edge at channel 1 signal, and channel 0 signal is at logic 1
  - A falling edge at channel 1 signal, and channel 0 signal is at logic 0
  - A falling edge at channel 0 signal, and channel 1 signal is at logic 1
- The TPM counter decrement happens at:
  - A falling edge at channel 0 signal, and channel 1 signal is at logic 0
  - A falling edge at channel 1 signal, and channel 0 signal is at logic 1
  - A rising edge at channel 1 signal, and channel 0 signal is at logic 0
  - A rising edge at channel 0 signal, and channel 1 signal is at logic 1

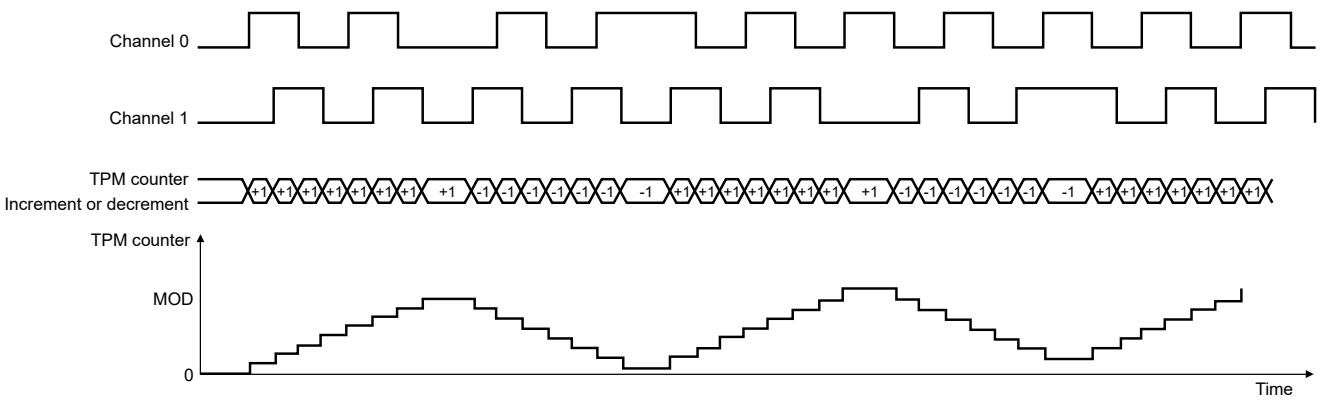


Figure 551. Quadrature Decoder—phase encoding mode

In Figure 552, when the TPM counter changes from the value defined in MOD[MOD] to 0, then SC[TOF] is 1 and QDCTRL[TOFDIR] becomes 1. SC[TOF] indicates that a TPM counter overflow has occurred. QDCTRL[TOFDIR] indicates that TPM was counting up when the TPM counter overflow occurred.

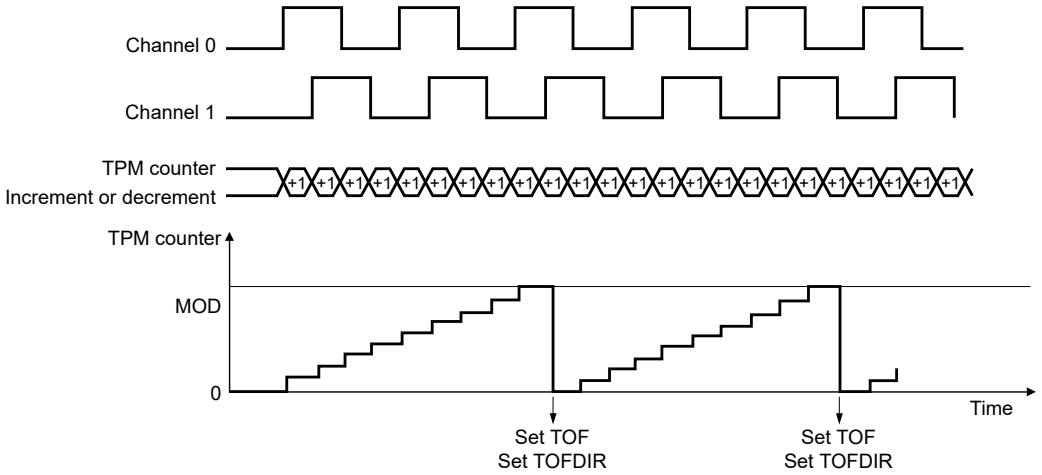


Figure 552. TPM counter overflow in up counting for Quadrature Decoder mode

In Figure 553, when the TPM counter changes from 0 to MOD[MOD], then SC[TOF] is 1 and QDCTRL[TOFDIR] becomes 0. SC[TOF] indicates that a TPM counter overflow has occurred. QDCTRL[TOFDIR] indicates that TPM was counting down when the TPM counter overflow occurred.

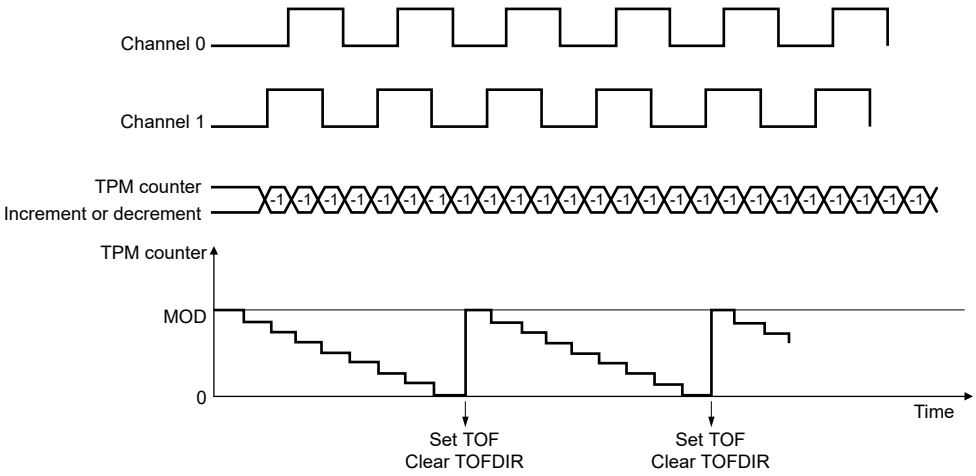


Figure 553. TPM counter overflow in down counting for Quadrature Decoder mode

70.3.5 Low-Power mode

During chip Low-Power mode, you can configure TPM to operate normally or to pause all counting.

Chip mode	TPM operation
Low-Power mode	• When CONF[DOZEEN] = 0, and TPM uses an external or internal clock source that remains functional, then counting continues. TPM can generate an asynchronous interrupt to exit the chip from Low-Power mode. • When CONF[DOZEEN] = 1, then counting pauses during Low-Power mode. Trigger inputs and input capture events are ignored, and PWM outputs are forced to their initial state.

70.3.6 Debug mode

During chip Debug mode, you can configure TPM to operate normally or to pause all counting temporarily.

Chip mode	TPM operation
Debug (core is in Debug or Halted mode)	• When CONF[DBGMODE] = 00b, then counting pauses until the core returns to normal user operating mode. Trigger inputs and input capture events are ignored, and PWM outputs are forced to their initial state. • When CONF[DBGMODE] = 11b, then counting continues.

70.3.7 Clocking

TPM supports two clock domains.

The bus clock domain is used by the register interface and for synchronizing interrupts and DMA requests.

The TPM counter clock domain clocks the counter and prescaler along with the output compare and input capture logic. The TPM counter clock is considered asynchronous to the bus clock. It can be a higher or lower frequency than the bus clock. It can optionally remain operational in low-power modes. When the global timebase feature is enabled, you must clock any TPM instance that is sharing the global timebase by the same TPM counter clock.

70.3.8 Reset

TPM resets whenever any chip reset occurs.

When TPM exits from reset:

- The TPM counter and the prescaler counter are 0 and are stopped (SC[CMOD] = 0).
- The timer overflow interrupt is 0.
- The channel interrupts are 0.
- The channels are in Input Capture mode.
- The channel outputs are 0.
- TPM does not control the channel pins (ELS*n*B:ELS*n*A = 0:0).

70.3.9 Interrupts

70.3.9.1 Timer overflow interrupt

TPM generates the timer overflow interrupt when (SC[TOIE] = 1) and (SC[TOF] = 1).

70.3.9.2 Channel *n* interrupt

TPM generates the channel *n* interrupt when (C*n*SC[CH*n*IE] = 1) and (C*n*SC[CH*n*F] = 1).

70.3.10 DMA

The channel and overflow flags generate a DMA transfer request according to SC[DMA] and C*n*SC[CH*n*IE]/SC[TOIE] fields. See the following table for more information.

Table 1067. DMA transfer request

DMA	CH*n*IE/ TOIE	Channel and overflow DMA transfer request	Channel and overflow interrupt
0	0	Not generated	Not generated
0	1	Not generated	Generated if (CH*n*F/TOF = 1)
1	0	Generated if (CH*n*F/TOF = 1)	Not generated
1	1	Generated if (CH*n*F/TOF = 1)	Generated if (CH*n*F/TOF = 1)

If DMA = 1, the CH*n*F/TOF field can be cleared either by completion of the DMA transfer or by writing 1 to CH*n*F/TOF (see Table 1068).

Table 1068. Clear CH*n*F/TOF field

DMA	How to clear CH*n*F/TOF
0	CH*n*F/TOF field is cleared by writing 1 to it.
1	CH*n*F/TOF field is cleared either when the DMA transfer is completed or by writing 1 to it.

70.4 External signals

Table 1069 shows the user-accessible signals for TPM.

Table 1069. TPM signal descriptions

Signal	Description	Direction
EXTCLK	External clock If selected in SC[CMOD], EXTCLK increments the TPM counter on every rising edge synchronized to the TPM counter clock. EXTCLK must be less than half of the TPM counter clock frequency. The TPM counter prescaler selection and settings are also used when EXTCLK is selected.	I
CH*n*	Channel *n* I/O pin (where *n* = 3 to 0) You can configure each CH*n* to operate either as input or output. CH*n* is an output when configured in Output Compare or PWM mode and the TPM counter is enabled; otherwise, CH*n* is an input.	I/O

70.5 Initialization

To initialize TPM:

1. Write to the following registers, in any order, to configure the counter mode of operation:
   - Configuration (CONF)
   - Quadrature Decoder Control and Status (QDCTRL)
   - Modulo (MOD)
2. Write to the following registers, in any order, to configure the mode of operation for each channel:
   - Channel n Value (C0V - C3V)
   - Combine Channel (COMBINE)
   - Channel Trigger (TRIG)
   - Channel Polarity (POL)
   - Filter Control (FILTER)
3. Use Channel n Status and Control (C0SC - C3SC) to configure the mode of operation for each channel and to enable the channel.
4. Use Status and Control (SC) to configure the counter mode of operation and to enable the TPM (TPM is enabled and counting when SC[CMOD] ≠ 00).

70.6 Application information

To configure TPM counter to generate an overflow interrupt every 64,000 cycles:

1. Write any data to CNT to reset the counter.
2. Write MOD[MOD] = F9FFh to configure counter overflow.
3. Write SC = 48h to enable the counter and overflow interrupt.
4. Write STATUS = 100h to clear the flag during the interrupt routine.

To configure TPM channel 0 for input capture on either edge generating an interrupt:

1. Write C0SC = 4Ch to configure channel 0.
2. Optionally, write CONF = 840000h to configure counter to reset on each channel 0 input capture. Write CONF = 850000h to delay starting the counter until the first edge.
3. Write any data to CNT to reset the counter.
4. Write MOD[MOD] = FFFFFFFFh to configure the counter overflow to maximum.
5. Write SC = 48h to enable the counter and overflow interrupt.
6. During the interrupt routine, first check the setting for the STATUS flag:
   - If STATUS[TOF] is set, clear flag by writing STATUS = 100h. Note that software is responsible for tracking number of overflows between consecutive input capture events.
   - If STATUS[CH0F] is set, first read C0V to obtain the input capture value and then clear flag by writing STATUS = 0001h.

To configure TPM channel 0 for output compare to toggle the output pin using an array of counter values:

1. Write C0SC = 54h for interrupt or write C0SC = 15h for DMA.
2. Write C0V = array[0] to configure initial compare value.
3. Write any data to CNT to reset the counter.
4. Write MOD[MOD] = FFFFFFFFh to configure the counter overflow to maximum.
5. Write SC = 08h to enable the counter.
6. Write the next compare value C0V = array[N] during the interrupt routine and then write STATUS = 0001h to clear the flag.

7. Write the next compare value C0V = array[N] during the DMA transfer. DMA transfer will automatically clear the channel flag.

The array is an array of counter values where the first output compare occurs at “array[0]” cycles after the counter is enabled and subsequent output compares occur at “array[N] – array[N-1]” cycles after the last compare. If the output waveform repeats, then MOD can be set to an output compare value where the waveform starts to repeat.

An example of using TPM with the asynchronous DMA is described in AN4631:Using the Asynchronous DMA features of the Kinetis L Series.

70.7 Memory map and register definition

This section provides a detailed description of all TPM registers.

An attempt to access a reserved register location in the TPM memory map generates a bus error.

70.7.1 TPM register descriptions

70.7.1.1 TPM memory map

TPM1 base address: 4431_0000h

TPM2 base address: 4432_0000h

TPM3 base address: 424E_0000h

TPM4 base address: 424F_0000h

TPM5 base address: 4250_0000h

TPM6 base address: 4251_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0600_0007h
4h	Parameter (PARAM)	32	R	0020_0404h
8h	TPM Global (GLOBAL)	32	RW	0000_0000h
10h	Status and Control (SC)	32	RW	0000_0000h
14h	Counter (CNT)	32	RW	0000_0000h
18h	Modulo (MOD)	32	RW	0000_FFFFh
1Ch	Capture and Compare Status (STATUS)	32	RW	0000_0000h
20h	Channel n Status and Control (C0SC)	32	RW	0000_0000h
24h	Channel n Value (C0V)	32	RW	0000_0000h
28h	Channel n Status and Control (C1SC)	32	RW	0000_0000h
2Ch	Channel n Value (C1V)	32	RW	0000_0000h
30h	Channel n Status and Control (C2SC)	32	RW	0000_0000h
34h	Channel n Value (C2V)	32	RW	0000_0000h
38h	Channel n Status and Control (C3SC)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3Ch	Channel n Value (C3V)	32	RW	0000_0000h
64h	Combine Channel (COMBINE)	32	RW	0000_0000h
6Ch	Channel Trigger (TRIG)	32	RW	0000_0000h
70h	Channel Polarity (POL)	32	RW	0000_0000h
78h	Filter Control (FILTER)	32	RW	0000_0000h
80h	Quadrature Decoder Control and Status (QDCTRL)	32	RW	0000_0000h
84h	Configuration (CONF)	32	RW	0000_0000h

70.7.1.2 Version ID (VERID)

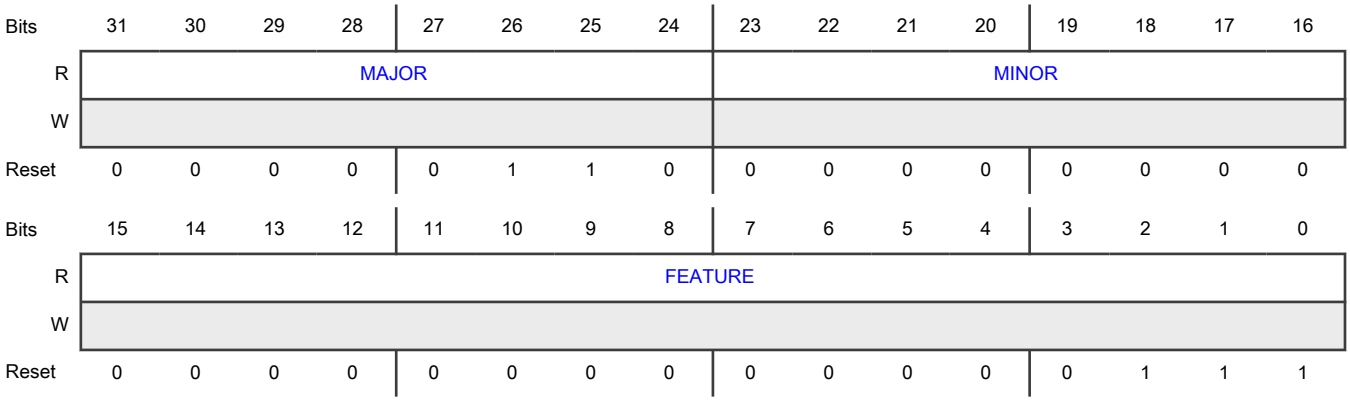
Offset

Register	Offset
VERID	0h

Function

Contains the module version ID and feature information.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Specifies the major version number for the module specification.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 MINOR	Minor Version Number Specifies the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number Configures different feature sets for TPM that implements a standard feature set with Filter Control (FILTER), Combine Channel (COMBINE), and Quadrature Decoder Control and Status (QDCTRL). 0000_0000_0000_0001b - Standard feature set 0000_0000_0000_0011b - Standard feature set with the filter and combine registers implemented 0000_0000_0000_0101b - Standard feature set with the quadrature register implemented 0000_0000_0000_0111b - Standard feature set with the filter, combine, and quadrature registers implemented

70.7.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function
Contains values of the module parameters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								WIDTH							
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TRIG								CHAN							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23-16 WIDTH	Counter Width Specifies the width of the counter and timer channels.
15-8 TRIG	Trigger Count Specifies the number of trigger inputs that TPM implements.
7-0 CHAN	Channel Count Specifies the number of timer channels that TPM implements.

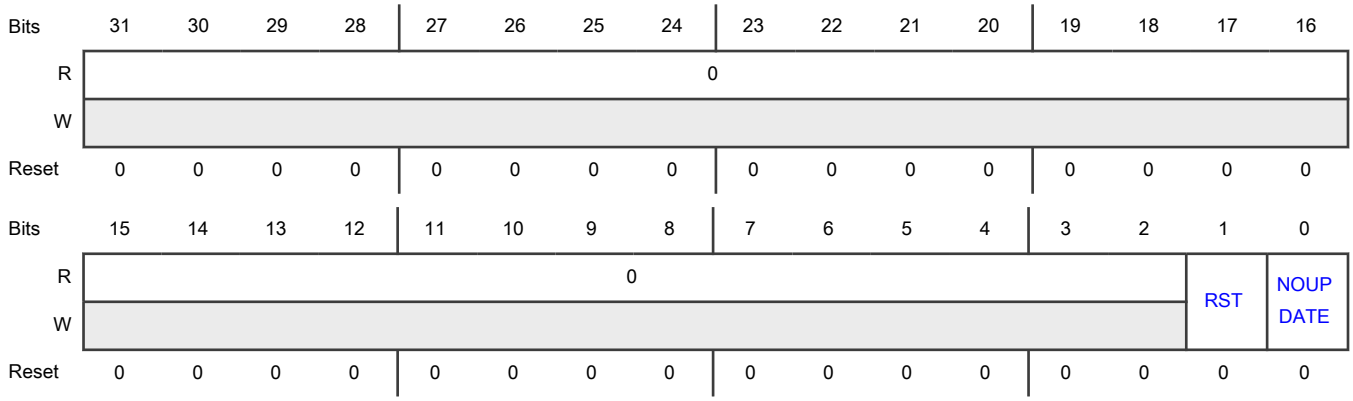
70.7.1.4 TPM Global (GLOBAL)

Offset

Register	Offset
GLOBAL	8h

Function
Includes the software reset control field.

Diagram



Fields

Field	Function
31-2 —	Reserved
1	Software Reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
RST	Resets all internal logic and registers, except TPM Global (GLOBAL). This field remains 1 until you write 0 to it. 0b - Module is not reset 1b - Module is reset
0 NOUPDATE	No Update Blocks updates to all internal double-buffered registers while this field remains 1. You can still update the software visible registers, but changes cannot take effect until you write 0 to this field. At that point they update as normal. 0b - Internal double-buffered registers update as normal 1b - Internal double-buffered registers do not update

70.7.1.5 Status and Control (SC)

Offset

Register	Offset
SC	10h

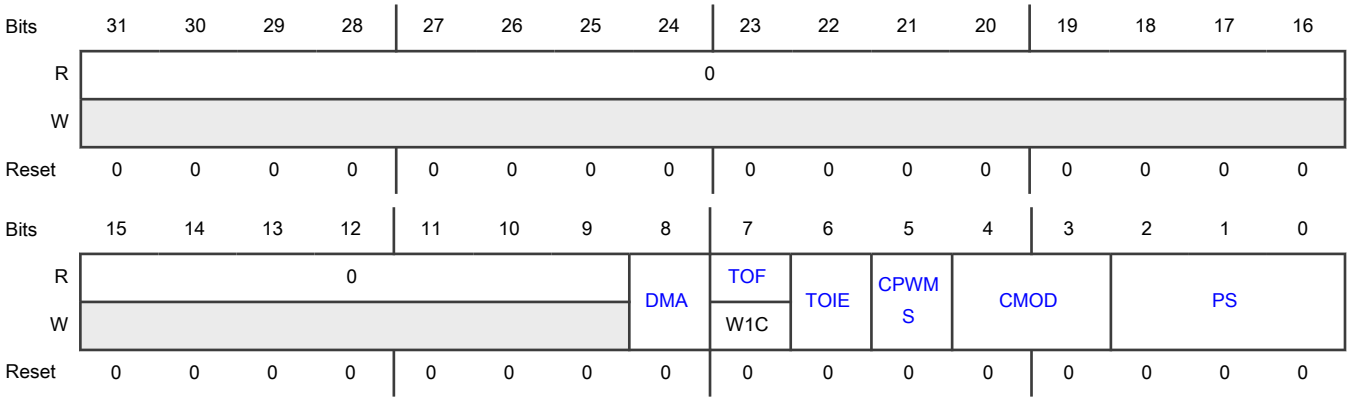
Function

Contains the overflow status flag and control fields that you can use to configure:

- Interrupt enable
- Module configuration
- Prescaler factor

These controls relate to all channels within this module.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 DMA	DMA Enable Enables DMA transfers for the overflow flag. 0b - Disable 1b - Enable
7 TOF	Timer Overflow Flag Indicates whether a TPM overflow has occurred. Hardware sets this flag when the TPM counter value is equal to the value defined in MOD[MOD], and increments after this value. If another TPM overflow occurs between the flag setting and the flag clearing, then the write operation has no effect. Therefore, TOF remains set, which indicates that another overflow has occurred. In this case, a TOF interrupt request is not lost due to a delay in clearing the previous TOF. 0b - No overflow 1b - Overflow
6 TOIE	Timer Overflow Interrupt Enable Enables TPM overflow interrupts. A TOF interrupt is generated when SC[TOF] = 1. If TOIE = 0, use software polling or DMA request. 0b - Disable 1b - Enable
5 CPWMS	Center-Aligned PWM Select Enables CPWM mode. This mode configures TPM to operate in up-down counting mode. This field is write-protected. Disable the TPM counter before you write to this field. 0b - Up counting mode 1b - Up-down counting mode
4-3 CMOD	Clock Mode Selection Selects the TPM counter clock modes. When disabling the counter, this field remains 1 until acknowledged in the TPM clock domain. 00b - TPM counter is disabled 01b - TPM counter increments on every TPM counter clock 10b - TPM counter increments on the rising edge of EXTCLK synchronized to the TPM counter clock 11b - TPM counter increments on the rising edge of the selected external input trigger
2-0	Prescale Factor Selection

Table continues on the next page...

Table continued from the previous page...

Field	Function
PS	Selects one of 8 division factors for the clock mode selected by SC[CMOD]. This field is write-protected. Disable the TPM counter before you write to this field. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

70.7.1.6 Counter (CNT)

Offset

Register	Offset
CNT	14h

Function

Contains the TPM counter value. You can write 0 to this register with reset. Writing any value to CNT[COUNT] triggers a reload of the counter and forces PWM outputs to their reload state.

When debug is active, the TPM counter does not increment unless you configure it otherwise. Reading this register adds two wait states to the register access, because of synchronization delays.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COUNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COUNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 COUNT	Counter Value Specifies the TPM counter value.

70.7.1.7 Modulo (MOD)

Offset

Register	Offset
MOD	18h

Function

Contains the modulo value for the TPM counter. When the TPM counter reaches the modulo value and increments, SC[TOF] sets and the next value of the TPM counter depends on the selected counting method (see Counter).

Writing to the MOD register latches the value into a buffer. This register updates with the value of its write buffer according to MOD register update. Additional writes to the MOD write buffer are ignored until the register has been updated.

To avoid confusion about occurrence of the first counter overflow, initialize the TPM counter (write to CNT[COUNT]) before writing to the MOD register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MOD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MOD															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 MOD	Modulo Value Specifies the modulo value for the TPM counter. Write a single 16-bit or 32-bit access value to this field.

70.7.1.8 Capture and Compare Status (STATUS)

Offset

Register	Offset
STATUS	1Ch

Function

Contains a copy of the status flag (C*n*SC[CHF]) for each TPM channel, as well as SC[TOF], for software convenience. Each CH*n*F field in STATUS is a mirror of the corresponding C*n*SC[CHF]. You can check all C*n*SC[CHF] fields using only one read of STATUS. Writing all ones to STATUS writes 0 to all C*n*SC[CHF] fields.

Hardware sets the individual channel flags when an event occurs on the channel. If another event occurs between the flag setting and the write operation, the write operation has no effect; therefore, CHF remains set, which indicates that another event has occurred. In this case, a CHF interrupt request is not lost because of the clearing sequence for a previous CHF.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							TOF	0				CH3F	CH2F	CH1F	CH0F
W								W1C					W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8 TOF	Timer Overflow Flag Contains a copy of SC[TOF]. 0b - No overflow 1b - Overflow
7-4 —	Reserved
3 CH3F	Channel 3 Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Contains a copy of C3SC[CHF]. Sets the flag when an event occurs on the channel (see the register description for more). 0b - Event not occurred 1b - Event occurred
2 CH2F	Channel 2 Flag Contains a copy of C2SC[CHF]. Sets the flag when an event occurs on the channel (see the register description for more). 0b - Event not occurred 1b - Event occurred
1 CH1F	Channel 1 Flag Contains a copy of C1SC[CHF]. Sets the flag when an event occurs on the channel (see the register description for more). 0b - Event not occurred 1b - Event occurred
0 CH0F	Channel 0 Flag Contains a copy of C0SC[CHF]. Sets the flag when an event occurs on the channel (see the register description for more). 0b - Event not occurred 1b - Event occurred

70.7.1.9 Channel n Status and Control (C0SC - C3SC)

Offset

Register	Offset
C0SC	20h
C1SC	28h
C2SC	30h
C3SC	38h

Function

Contains the channel-interrupt-status flag and control fields used to configure:

- Interrupt enable
- Channel configuration
- Pin function

Table 1070. Mode, edge, and level selection

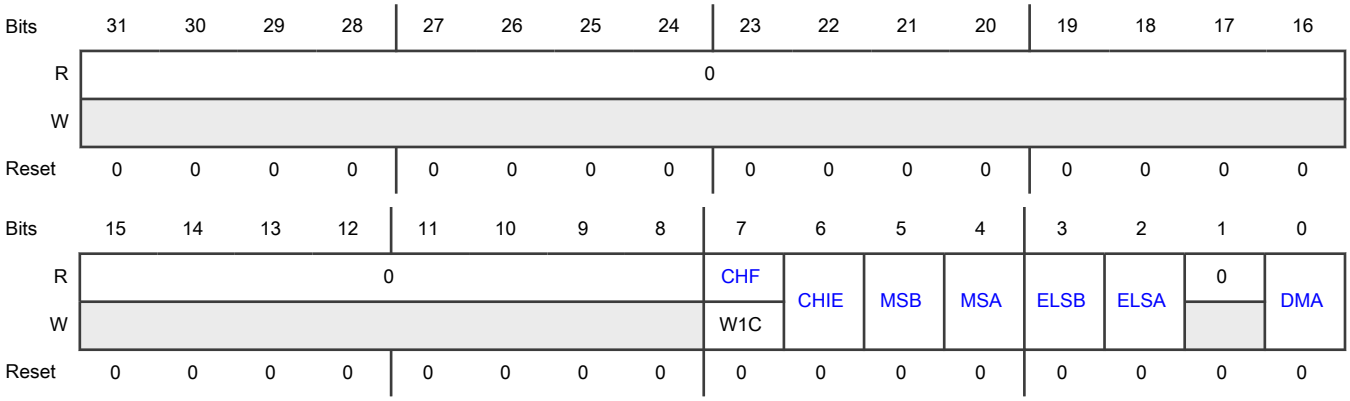
SC[CPWMS]	MS*n*B:MS*n*A	ELS*n*B:ELS*n*A	Mode	Configuration
0	00	00	None	Channel disabled
0	01	00	Software Compare	Pin not used for TPM
0	00	01	Input Capture	Capture on rising edge only
		10		Capture on falling edge only
		11		Capture on rising or falling edge
	01	01	Output Compare	Toggle output on match
		10		Clear output on match
		11		Set output on match
	10	10	Edge-Aligned PWM	High-true pulses (clear output on counter match, set output on counter reload, set output when counter first enabled or paused)
		00		High-true pulses (clear output on counter match, set output on counter reload, clear output when counter first enabled or paused)
		01		Low-true pulses (set output on counter match, clear output on counter reload, clear output when counter first enabled or paused)
		11		Low-true pulses (set output on counter match, clear output on counter reload, set output when counter first enabled or paused)
	11	10	Output Compare	Pulse output low on match
		01		Pulse output high on match
1	10	10	Center-Aligned PWM	High-true pulses (clear output on counter match-up, set output on counter match-down, set output on counter reload, or when counter first enabled or paused)
		00		High-true pulses (clear output on counter match-up, set output on counter match-down, clear output on counter reload, or when counter first enabled or paused)

Table continues on the next page...

Table 1070. Mode, edge, and level selection (continued)

SC[CPWMS]	MS*n*B:MS*n*A	ELS*n*B:ELS*n*A	Mode	Configuration
		01		Low-true pulses (set output on match-up, clear output on match-down, clear output on counter reload, or when counter first enabled or paused)
		11		Low-true pulses (set output on match-up, clear output on match-down, set output on counter reload, or when counter first enabled or paused)

Diagram



Fields

Field	Function
31-8 —	Reserved
7 CHF	Channel Flag Indicates whether an event has occurred on the channel. TPM writes 1 to this field when an event occurs on the channel. If another event occurs between the CHF being set and the write operation, the write operation has no effect; therefore, CHF remains set, which indicates that another event has occurred. In this case, a CHF interrupt request is not lost due to the delay in clearing the previous CHF. 0b - Event not occurred 1b - Event occurred
6 CHIE	Channel Interrupt Enable Enables channel interrupts.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
5 MSB	Channel Mode Select B Configures further selections in the channel logic. Its functionality depends on the channel mode (see Table 1070).
4 MSA	Channel Mode Select A Configures further selections in the channel logic. Its functionality depends on the channel mode (see Table 1070).
3 ELSB	Edge or Level Select B Performs the function that depends on the channel mode (see Table 1070).
2 ELSA	Edge or Level Select A Performs the function that depends on the channel mode (see Table 1070).
1 —	Reserved
0 DMA	DMA Enable Enables DMA transfers for the channel. 0b - Disable 1b - Enable

70.7.1.10 Channel n Value (C0V - C3V)

Offset

Register	Offset
C0V	24h
C1V	2Ch
C2V	34h
C3V	3Ch

Function

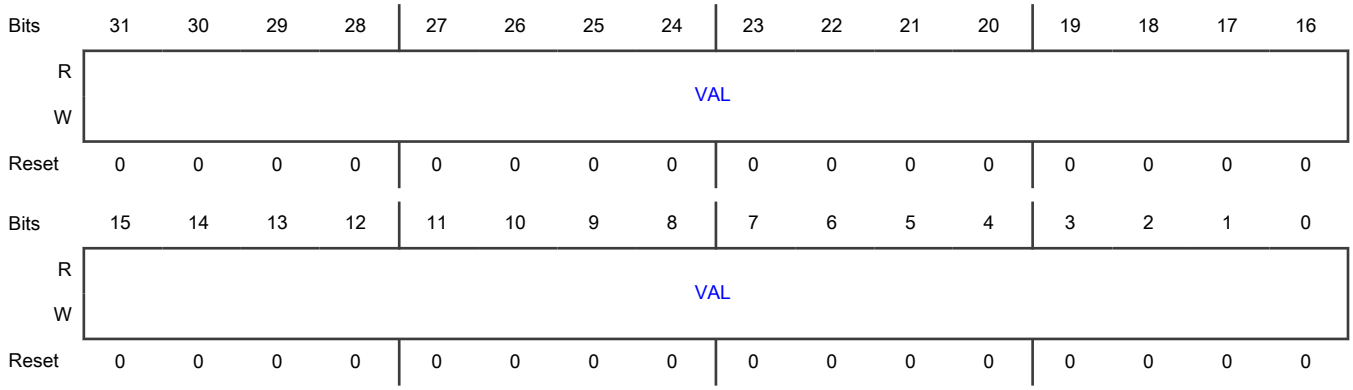
Contains the captured TPM counter value for the input modes or the match value for the output modes.

Input Capture mode ignores any write to a C*n*V register. In compare modes, writing to a C*n*V register latches the value into a buffer. A C*n*V register updates with the value of its write buffer according to C*n*V register update.

NOTE

Additional writes to the C*n*V write buffer are ignored until the register has been updated.

Diagram



Fields

Field	Function
31-0 VAL	Channel Value Contains the captured TPM counter value for the input modes, or the match value for the output modes. Write a single 16-bit or 32-bit access value to this field.

70.7.1.11 Combine Channel (COMBINE)

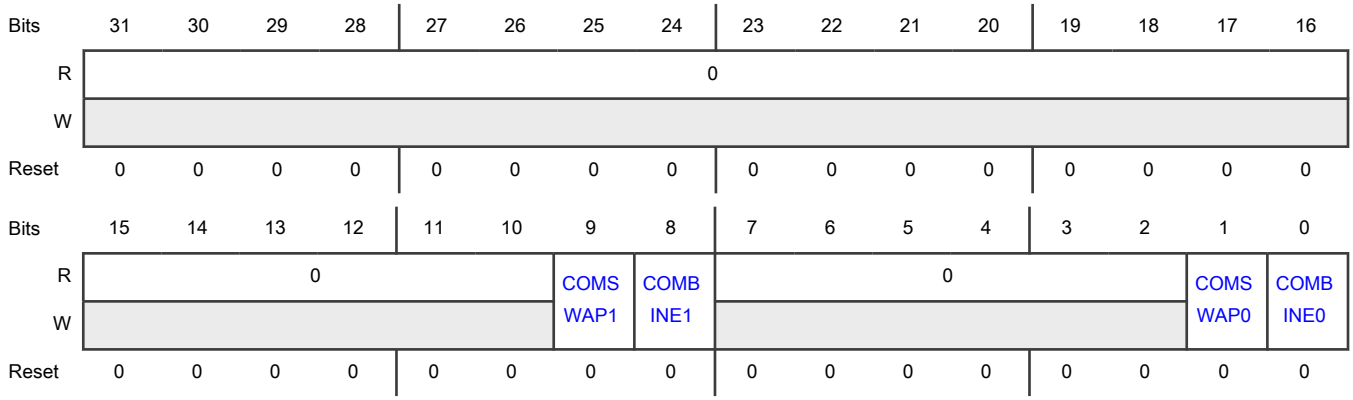
Offset

Register	Offset
COMBINE	64h

Function

Contains the control fields used to configure the combine channel modes for each pair of channels *n* and *n*+1, where n is all the even-numbered channels.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-10 —	Reserved
9 COMSWAP1	Combine Channels 2 and 3 Swap When set in combine modes, the odd channel is used for the input capture and first compare, while the even channel is used for the second compare. 0b - Even channel 1b - Odd channel
8 COMBINE1	Combine Channels 2 and 3 Enables the combine feature for channels 2 and 3. In Input Capture mode, the combined channels use the even channel input. In Software Compare mode, the even channel match asserts the output trigger, and the odd channel match negates the output trigger. In PWM mode, the even channel match is used for the first compare, and odd channel match for the second compare. 0b - Independent 1b - Combined
7-2 —	Reserved
1 COMSWAP0	Combine Channel 0 and 1 Swap When set in combine modes, the odd channel is used for the input capture and first compare, while the even channel is used for the second compare. 0b - Even channel 1b - Odd channel
0 COMBINE0	Combine Channels 0 and 1 Enables the combine feature for channels 0 and 1. In Input Capture mode, the combined channels use the even channel input. In Software Compare mode, the even channel match asserts the output trigger, and the odd channel match negates the output trigger. In PWM mode, the even channel match is used for the first compare and odd channel match for the second compare. 0b - Independent 1b - Combined

70.7.1.12 Channel Trigger (TRIG)

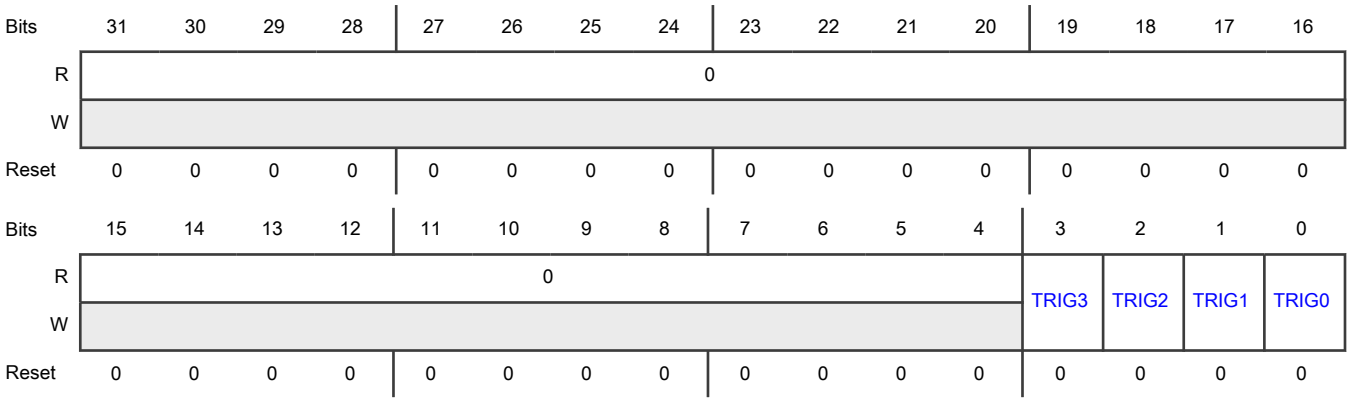
Offset

Register	Offset
TRIG	6Ch

Function

Configures the trigger input that is used by the channel to capture the counter value in Input Capture mode. In Output Compare or PWM mode, configures the trigger input used to modulate the channel output. When modulating the output, the output is forced to the channel initial value whenever the trigger is not asserted. The even-numbered channels share the first input trigger source, and the odd-numbered channels share the second input trigger source.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 TRIG3	Channel 3 Trigger 0b - No effect 1b - Configures trigger input 1 to be used by channel 3
2 TRIG2	Channel 2 Trigger 0b - No effect 1b - Configures trigger input 0 to be used by channel 2
1 TRIG1	Channel 1 Trigger 0b - No effect 1b - Configures trigger input 1 to be used by channel 1
0	Channel 0 Trigger

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRIG0	0b - No effect 1b - Configures trigger input 0 to be used by channel 0

70.7.1.13 Channel Polarity (POL)

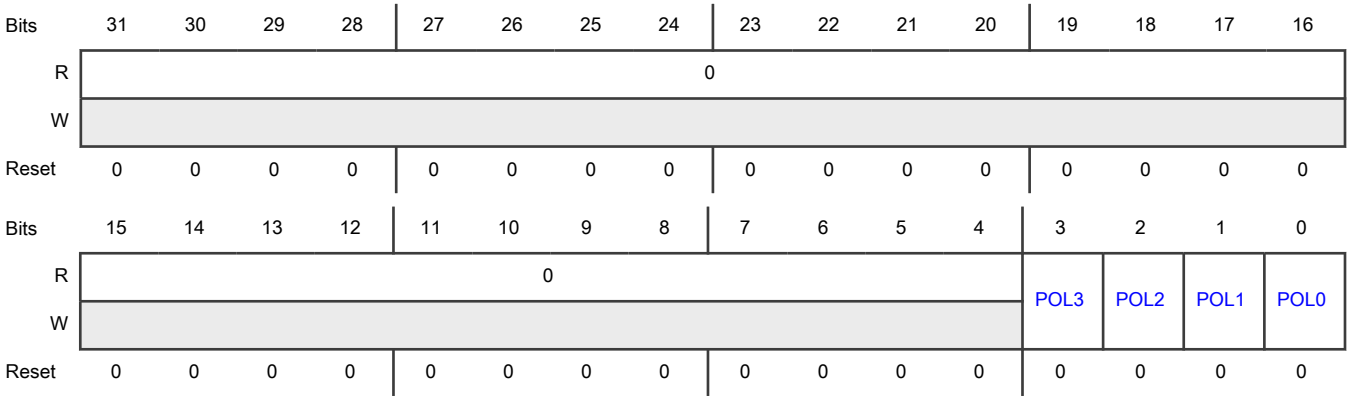
Offset

Register	Offset
POL	70h

Function

Defines the input and output polarity of each of the channels.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 POL3	Channel 3 Polarity 0b - Active high 1b - Active low
2 POL2	Channel 2 Polarity 0b - Active high 1b - Active low

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 POL1	Channel 1 Polarity 0b - Active high 1b - Active low
0 POL0	Channel 0 Polarity 0b - Active high 1b - Active low

70.7.1.14 Filter Control (FILTER)

Offset

Register	Offset
FILTER	78h

Function

Selects the filter value of the channel inputs, and an additional output delay value for the channel outputs. In PWM Combine mode, you can use this field to implement dead-time insertion.

NOTE

To write values to CH*n*FVAL, ensure that the corresponding channel-interrupt status flag and control (C*n*SC[5:2]) fields are set to 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CH3FVAL				CH2FVAL				CH1FVAL				CH0FVAL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-12 CH3FVAL	Channel 3 Filter Value When configured for Input Capture or Software Compare mode, configures the filter value for the channel 3 input. The filter is disabled when the value is zero. Otherwise, the filter is configured as (CH3FVAL×4)clock cycles. Sets the dead-time insertion time for channel 3 in PWM modes. Dead time insertion is disabled when CH3FVAL is zero. Otherwise, dead time insertion for channel 3 is configured as (CH3FVAL×4)clock cycles.
11-8 CH2FVAL	Channel 2 Filter Value When configured for Input Capture or Software Compare mode, configures the filter value for the channel 2 input. The filter is disabled when the value is zero. Otherwise, the filter is configured as (CH2FVAL×4)clock cycles. Sets the dead-time insertion time for channel 2 in PWM modes. Dead time insertion is disabled when CH2FVAL is zero. Otherwise, dead time insertion for channel 2 is configured as (CH2FVAL×4)clock cycles.
7-4 CH1FVAL	Channel 1 Filter Value When configured for Input Capture or Software Compare mode, configures the filter value for the channel 1 input. The filter is disabled when the value is zero. Otherwise, the filter is configured as (CH1FVAL×4)clock cycles. Sets the dead-time insertion time for channel 1 in PWM modes. Dead time insertion is disabled when CH1FVAL is zero. Otherwise, dead time insertion for channel 1 is configured as (CH1FVAL×4)clock cycles.
3-0 CH0FVAL	Channel 0 Filter Value When configured for Input Capture or Software Compare mode, configures the filter value for the channel 0 input. The filter is disabled when the value is zero. Otherwise, the filter is configured as (CH0FVAL×4)clock cycles. Sets the dead time insertion time for channel 0 in PWM modes. Dead time insertion is disabled when CH0FVAL is zero. Otherwise, dead time insertion for channel 0 is configured as (CH0FVAL×4)clock cycles.

70.7.1.15 Quadrature Decoder Control and Status (QDCTRL)

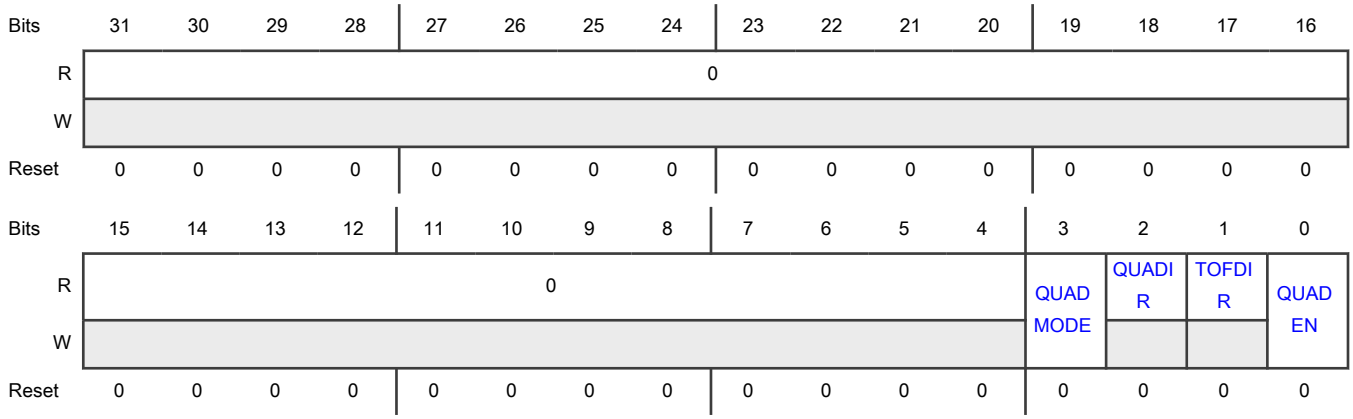
Offset

Register	Offset
QDCTRL	80h

Function

Contains the control and status fields for Quadrature Decoder mode.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 QUADMODE	Quadrature Decoder Mode Selects the encoding mode used in Quadrature Decoder mode. 0b - Phase encoding mode 1b - Count and direction encoding mode
2 QUADIR	Counter Direction in Quadrature Decode Mode Indicates whether the counting direction is decreasing or increasing. 0b - Decreasing (counter decrement) 1b - Increasing (counter increment)
1 TOFDIR	Timer Overflow Direction Indicates if SC[TOF] was set on the top or bottom of counting. • Field value 0 indicates that SC[TOF] was set on the bottom of counting. There was a TPM counter decrement and TPM counter changes from its minimum value (0) to its maximum value (MOD[MOD]). • Field value 1 indicates that SC[TOF] was set on the top of counting. There was a TPM counter increment and TPM counter changes from its maximum value (MOD[MOD]) to its minimum value (0). 0b - Bottom of counting 1b - Top of counting
0 QUADEN	Quadrature Decoder Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables Quadrature Decoder mode. In this mode, the channel 0 and channel 1 inputs control the TPM counter direction and can only be used for Software Compare mode. Quadrature Decoder mode has precedence over the other modes. 0b - Disable 1b - Enable

70.7.1.16 Configuration (CONF)

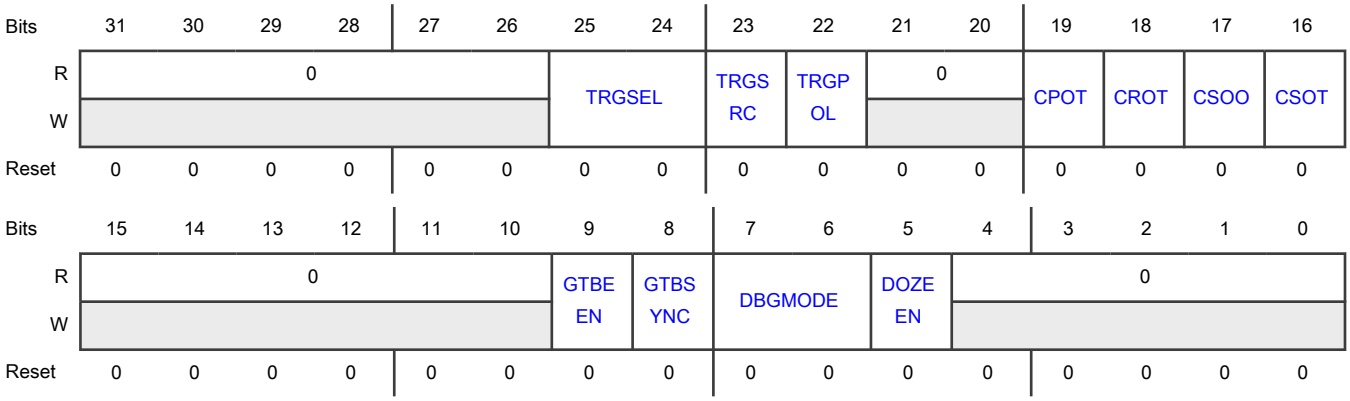
Offset

Register	Offset
CONF	84h

Function

Selects the TPM behavior in Debug mode and Low-Power mode, and the use of an external GTB.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 TRGSEL	Trigger Select Selects the input trigger that you use to start, reload, and pause the counter. You can configure the source of the trigger (external or internal to TPM) by CONF[TRGSRC]. Disable the TPM counter before you change this field. See the chip-specific TPM information for available external trigger options.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The below list shows available internal trigger sources. 01b - Channel 0 pin input capture 10b - Channel 1 pin input capture 11b - Channel 0 or channel 1 pin input capture
23 TRGSRC	Trigger Source Selects between internal (channel pin input capture) or external trigger sources. CONF[TRGSEL] selects this trigger source. When selecting an internal trigger, configure the selected channel for input captures as follows: • CONF[CSOT] enables you to start the counter initially using only a rising edge input capture. • CONF[CROT] enables you to reload the counter using either a rising edge or falling edge input capture. • CONF[CPOT] enables you to pause the counter using the state of the channel input pin. You can use Channel Polarity (POL) to invert the polarity of the channel input pins. Disable the TPM counter before you change this field. 0b - External 1b - Internal (channel pin input capture)
22 TRGPOL	Trigger Polarity Selects the polarity of the external trigger source. Disable the TPM counter before you change this field. 0b - Active high 1b - Active low
21-20 —	Reserved
19 CPOT	Counter Pause on Trigger Specifies that while the trigger remains asserted (level sensitive), and this field is 1, the counter pauses incrementing. While being paused, the counter ignores the input capture events and forces PWM outputs to their default state. If the TPM counter pauses during Debug mode or Low-Power mode, then the counter ignores the trigger input. Disable the TPM counter before you change this field. 0b - TPM counter continues 1b - TPM counter pauses
18 CROT	Counter Reload on Trigger Specifies that when you detect a rising edge on the selected trigger input and this field is 1, then the TPM counter reloads with 0 (and sets PWM outputs to their reload state). If the TPM counter pauses during Debug mode or Low-Power mode, then the counter ignores the trigger input. Disable the TPM counter before you change this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No reload 1b - Reload
17 CSOO	Counter Stop on Overflow Specifies that the TPM counter pauses when SC[TOF] = 1 and this field is 1. Reloading the counter with 0 by writing to Counter (CNT), or by a trigger input, does not cause the counter to stop incrementing. When stopped, the counter does not start incrementing unless you first disable and then enable it, or if you detect a rising edge on the selected trigger input when CONF[CSOT] is 1. While the counter is paused, it ignores the input capture events and forces PWM outputs to their default state. Disable the TPM counter before you change this field. 0b - TPM counter continues 1b - TPM counter stops
16 CSOT	Counter Start on Trigger Specifies that until you detect a rising edge on the selected trigger input, and this field is 1, the TPM counter pauses incrementing when it is enabled. If the TPM counter pauses due to an overflow, a rising edge on the selected trigger input also causes the TPM counter to start incrementing again. While being paused, the counter ignores the input capture events, and forces PWM outputs to their default state. If the TPM counter pauses during Debug mode or Low-Power mode, then the counter ignores the trigger input. Disable the TPM counter before you change this field. 0b - Counter starts immediately 1b - Counter starts after detection of a rising edge on the selected input trigger
15-10 —	Reserved
9 GTBEEN	GTB Enable Configures TPM to use an externally generated GTB counter. When you use an externally generated timebase, the channels do not use the internal TPM counter, but you can use the internal TPM counter to generate a periodic interrupt or DMA request using Modulo (MOD) and SC[TOF]. 0b - Internally generated TPM counter 1b - Externally generated GTB counter
8 GTBSYNC	GTB Synchronization Enables the TPM counter to synchronize with GTB. It uses the GTB enable, trigger, paused state and overflow to ensure that the TPM counter starts incrementing, stops incrementing, and resets at the same time as GTB. Disable the TPM counter before you change this field. 0b - Disable 1b - Enable
7-6	Debug Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
DBGMODE	Configures the TPM behavior in Debug mode (see Debug mode). All other configurations are reserved. 00b - TPM counter pauses 11b - TPM counter continues
5 DOZEEN	Doze Enable Configures the TPM behavior in Low-Power mode (see Low-Power mode). 0b - TPM counter continues 1b - TPM counter pauses
4-0 —	Reserved

Chapter 71
Low Power Periodic Interrupt Timer (LPIT)

71.1 Chip-specific LPIT Information

Table 1071. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

71.2 Overview

LPIT is a low-power periodic interrupt timer with multiple timer channels. After a timer reaches a programmed count, the respective timer channels generate pre-trigger and trigger output signals, and these outputs can be used to trigger other modules on the chip.

71.2.1 Block diagram

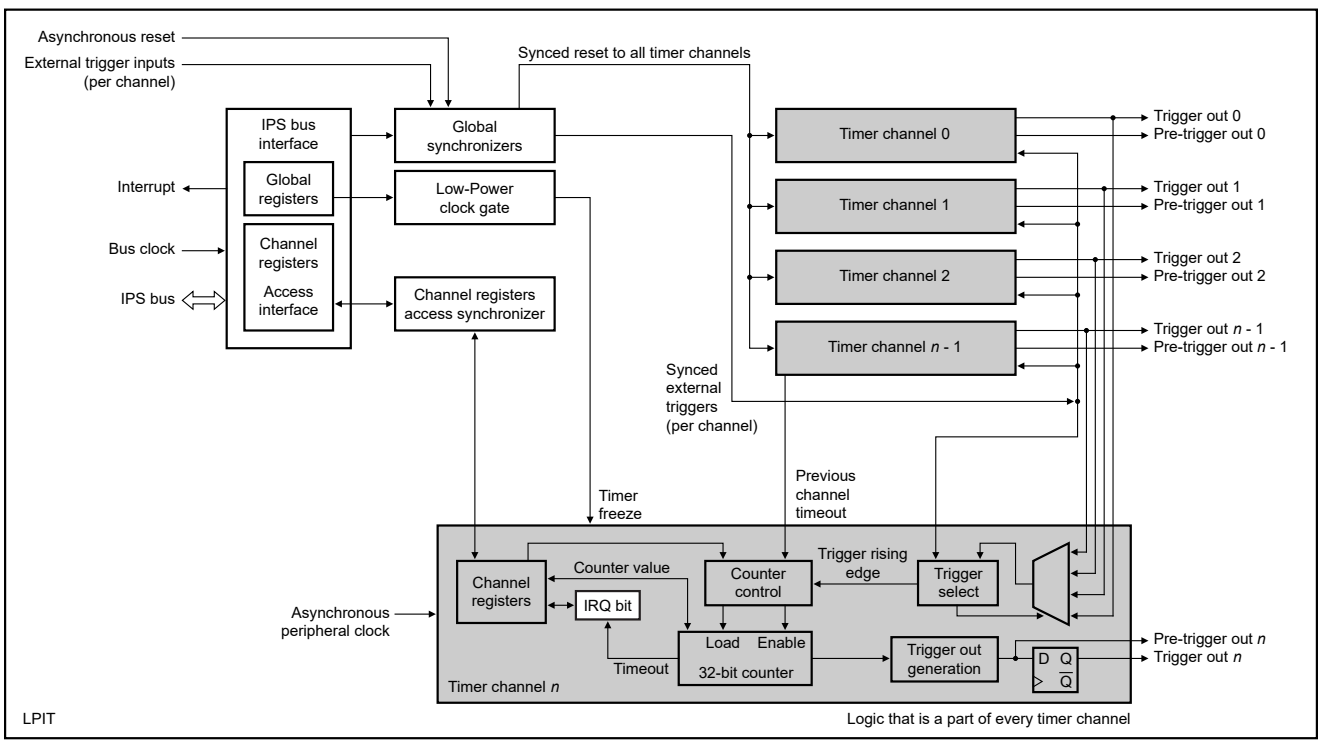


Figure 554. Block diagram

71.2.2 Features

LPIT allows you to configure timer channels in a way that they could be:

- Controlled using external triggers (triggers from outside LPIT).
- Controlled using internal triggers (triggers from other timer channels inside LPIT).
- Chained together, to form a larger width timer.
- Reloaded and counted again, or stopped after reaching the programmed count, depending on the timer modes used.

71.3 Functional description

71.3.1 Programming model

The LPIT programming model (see Figure 555) comprises:

- A global register set (common to all timer channels).
- Registers for each timer channel (that control their respective timer channels).

Access to these registers is synchronized with the asynchronous peripheral clock and then affects the timer channel registers:

- Each timer channel contains a 32-bit counter that loads the starting value and down counts after every peripheral clock's positive edge.
- After reaching a zero value (a channel timer timeout), a trigger output is generated.
- A timer enable register control field, external or internal triggers, or a previous channel timeout (when using timer chaining) control the counter enable.

- After a channel timer timeout, an interrupt flag is also set to tell the CPU about the timer timeout.

You must configure the following global registers only once:

- Module Control (MCR)
- Module Status (MSR)
- Module Interrupt Enable (MIER)
- Set Timer Enable (SETTEN)
- Clear Timer Enable (CLRTEN)

You must configure the following channel registers for each channel:

- Timer Value (TVAL0 - TVAL3)
- Timer Control (TCTRL0 - TCTRL3)

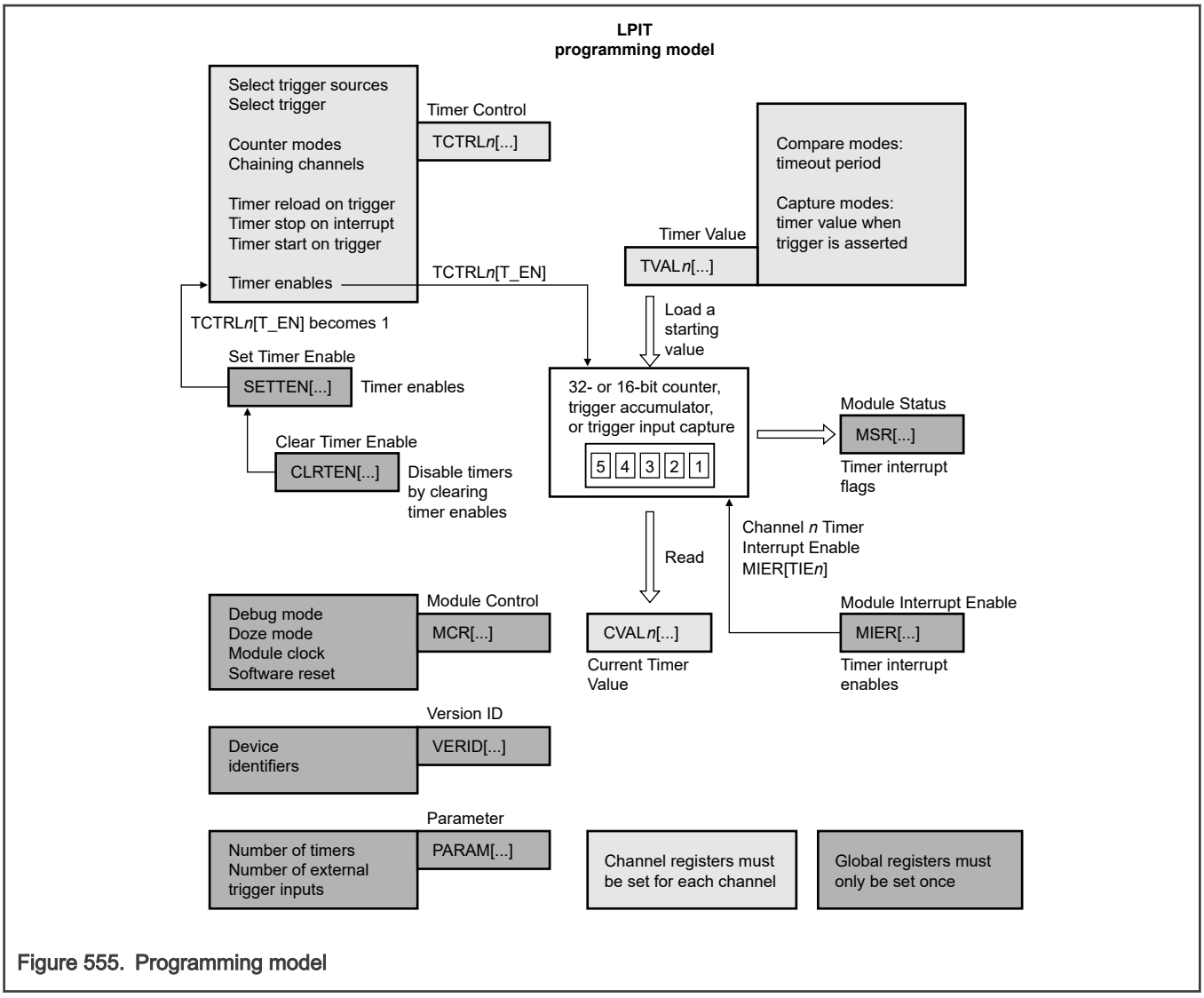


Figure 555. Programming model

71.3.2 Interfacing with other modules

The following figure shows the interface of an LPIT module with other modules on the chip:

- The CPU interface provides the clock, reset, and register read and write bus interface. It handles LPIT interrupts.

- The LPIT trigger output signals may trigger other modules on the chip, such as DMA and ADC.
- Similarly, other timer modules may provide trigger inputs to LPIT to control when an LPIT timer channel must start.

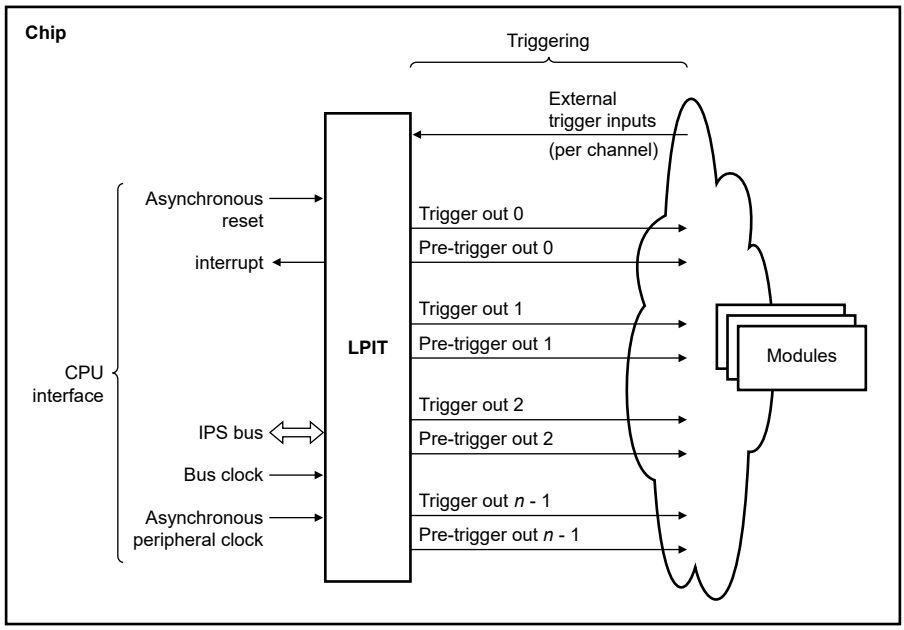


Figure 556. Interfacing with other modules

71.3.3 Chip power modes

LPIT supports the following chip power modes.

Table 1072. Chip power modes

Chip mode	LPIT operation
Run	Normal operation
Stop and Wait	Can continue operating in this mode if MCR[DOZE_EN] = 1 and LPIT is using an external or internal clock source that remains operational during Stop and Wait modes.
Debug	Can continue operating in this mode if MCR[DBG_EN] = 1

71.3.4 Supported timer modes

To configure a timer mode, you must select an appropriate value using TCTRL*n*[MODE].

Table 1073. Supported timer modes

Timer mode	Operation
32-bit periodic counter	The counter loads and decrements down to 0, and this sets the timer interrupt flag and asserts the output pretrigger.

Table continues on the next page...

Table 1073. Supported timer modes (continued)

Timer mode	Operation
Dual 16-bit periodic counter	• The counter loads and then the lower 16 bits decrement down to 0. This asserts the output pretrigger. • The upper 16 bits then decrement down to 0. This sets the timer interrupt flag and negates the output pretrigger.
32-bit trigger accumulator	The counter loads after the first trigger rising edge and then decrements down to 0 after each trigger rising edge. This sets the timer interrupt flag and asserts the output pretrigger.
32-bit trigger input capture	• The counter loads with a value of FFFF_FFFFh and then decrements down to 0. • If a trigger rising edge is detected, it stores the inverse of the current counter value in Timer Value (TVAL0 - TVAL3). This sets the timer interrupt flag and asserts the output pretrigger.

TCTRL*n*[TSOT], TCTRL*n*[TSOI], and TCTRL*n*[TROT] control the timer operation. These fields control the timer load, reload, start, and restart of the timers.

NOTE

- The trigger output is asserted one peripheral timer clock cycle after the pre-trigger output. The trigger and pre-trigger outputs deassert at the same time.
- The pre-trigger output is asserted for two clock cycles and the trigger output is asserted for one clock cycle (except in 16-bit Periodic Counter mode, where both pre-trigger and trigger outputs are asserted for many cycles depending on the value of TVAL*n*[TMR_VAL][31:16]).
- Timer changes (that are based on external triggers) take effect after four peripheral clocks, after the actual external trigger assertion. This is because of clock synchronization, rise edge detection, and timer update.

71.3.5 Timer channel modes

You can configure each timer channel in LPIT to work in either compare modes or capture modes.

The timer channels operate on an asynchronous clock, which is independent of the register read and write access clock. Clock synchronization between clock domains ensures normal operations.

Table 1074. Timer channel modes

Mode	Function
Compare	The timers decrement when enabled and generate an output pretrigger and trigger output. The trigger output is one clock cycle delayed of the pre-trigger pulse. You can configure certain control fields to control each timer channel's start, reload, and restart (see Trigger control for timers for more information). You can also configure the timer to always decrement from a programmed start value, on selected trigger inputs, or previous channel timeout (when channels are chained). By chaining timer channels, applications can achieve larger timeout durations.
Capture	You can use the timer to perform measurements by using Timer Value (TVAL0 - TVAL3), as the timer value is captured when a selected trigger input is asserted. The timer can support once-off or multiple measurements, such as frequency measurements.

71.3.6 Trigger control for timers

You can configure various LPIT register fields to control how trigger inputs and the timer operate:

- TCTRL*n*[TRG_SEL] helps you select the input trigger for the channel from all other channels' trigger outputs.
- TCTRL*n*[TRG_SRC] helps you select between the internal and external trigger inputs to the channel.

The selected trigger affects how the timer operates, using the configuration of TCTRL*n*[TSOT], TCTRL*n*[TSOI], and TCTRL*n*[TROT] (see Table 1075).

Table 1075. Fields that control timer operations

If	=	Then
Timer stop on interrupt (TCTRL*n*[TSOI])	1	The counter stops after MSR[TIF*n*] assertion. To reload and decrement, it requires: • A trigger (if TCTRL*n*[TSOT] = 1). • A T_EN rising edge (if TCTRL*n*[TSOT] = 0).
	0	The counter does not stop after timeout.
Timer reload on trigger (TCTRL*n*[TROT])	1	The counter is loaded after each trigger.
	0	The counter is loaded after every T_EN rising edge or timeout rising edge (timeout is not used in Capture modes).
Timer start on trigger (TCTRL*n*[TSOT])	1	The counter starts decrementing after a trigger. Subsequent triggers are ignored until the counter times out.
	0	The counter decrements immediately after the next clock edge. When the channel is chained or is in Capture mode, TCTRL*n*[TSOT] has no effect.

In different timer modes, the programmable fields listed in Table 1076 affect the timer operation differently.

Table 1076. Timer modes and programmable fields

Mode (TCTRL*n*[MODE])	Fields affecting timer operations
32-bit periodic counter	TCTRL*n*[TSOT], TCTRL*n*[TSOI], and TCTRL*n*[TROT] affect the timer operation as described in Table 1075.
Dual 16-bit periodic counter	
32-bit trigger accumulator	• Only TCTRL*n*[TSOI] controls the timer function. • TCTRL*n*[TROT] and TCTRL*n*[TSOT] have no effect on timer operations.
32-bit input trigger capture	• Only TCTRL*n*[TSOI] and TCTRL*n*[TROT] control the timer function. • TCTRL*n*[TSOT] has no effect on timer operations.

71.3.7 Channel chaining

You can chain individual timer channels together to achieve a larger value of timeout. Chaining enables these channels to work in a "nested loop" manner, thereby leading to an effective timeout value of $TVAL_{CHn} \times (TVAL_{CHn-1} + 1)$.

To chain the channels together, write 1 to TCTRL*n*[CHAIN] for the corresponding channel. When a channel is chained, that channel's timer decrements after the previous channel's timeout pulse, regardless of the timer mode that TCTRL*n*[MODE] defines.

TCTRL*n*[TSOT] does not have any effect if you chain the channel timer (channel *n*) to the previous channel's timer (channel *n* - 1).

71.3.8 Detailed timing

The following table represents various timing diagrams. The diagrams are not "cycle-accurate," which means, they may not show some of the cycles, but they do show the timer channel behavior across several clock cycles.

Table 1077. Timing diagrams

Mode (TCTRL*n*[MODE]) setting		Timing diagram	TCTRL*n*[TSOT]	TCTRL*n*[TROT]	TCTRL*n*[TSOI]	TCTRL*n*[CHAIN]
32-bit periodic counter (Compare mode)	00	Case 1: TCTRL*n*[MODE] = 0 • For a use case requiring repeated interrupts with reload • Trigger outputs have equal periods	0	0	0	0
		Case 2: TCTRL*n*[MODE] = 0 • Useful for One-Shot Trigger mode • Trigger starts again after TCTRL*n*[T_EN] becomes 1	0	0	1	0
		Case 3: TCTRL*n*[MODE] = 0 • For a use case requiring repeated interrupts with reload • Trigger outputs have unequal periods	0	1	0	0
		Case 4: TCTRL*n*[MODE] = 0 • For a one-shot timer with reload before timeout of Timer mode • Timer starts again after you write 1 to TCTRL*n*[T_EN]	0	1	1	0
		Case 5: TCTRL*n*[MODE] = 0 • Useful for generating periodic interrupts after a predefined event (input trigger) • Output triggers have equal periods	1	0	0	0
		Case 6: TCTRL*n*[MODE] = 0 • Triggers One-Shot Timer mode • Output trigger period depends on the input trigger	1	0	1	0
		Case 7: TCTRL*n*[MODE] = 0 • Repeated interrupts with Reload Timer mode • Outputs triggers have unequal periods	1	1	0	0
		Case 8: TCTRL*n*[MODE] = 0 • Case shown for a nonperiodic input trigger (might not be a use case)	1	1	1	0

Table continues on the next page...

Table 1077. Timing diagrams (continued)

Mode (TCTRL*n*[MODE]) setting		Timing diagram	TCTRL*n*[TSOT]	TCTRL*n*[TROT]	TCTRL*n*[TSOI]	TCTRL*n*[CHAIN]
		• If input trigger is periodic and greater than timer timeout, it is the same as TCTRL*n*[TROT] = 0 • If input trigger is periodic and less than timer timeout, the timer never times out and always reloads on the input trigger (not a valid use case)				
16-bit dual periodic counter (Compare mode)	01	Case 1: TCTRL*n*[MODE] = 1 • The effect of TCTRL*n*[TSOT], TCTRL*n*[TROT], and TCTRL*n*[TSOI] is the same as that of TCTRL*n*[MODE] = 0 (32-bit Counter Compare mode) • Both halves of the counter are affected in the same way	0	0	0	0
32-bit trigger accumulator mode	10	Case 1: TCTRL*n*[MODE] = 10 • Useful for a continuous pulse counting mode • Trigger and timeout generated after programmed number of pulses are accumulated	X	X	0	0
		Case 2: TCTRL*n*[MODE] = 10 • Useful for One-Shot Pulse Counting mode • Trigger and timeout generated after programmed number of pulses are accumulated	X	X	1	0
32-bit trigger capture mode	11	Case 1: TCTRL*n*[MODE] = 11 • Useful for determining the duration between pulses • Proper clock selection can ensure that the timer does not rollover more than once between two pulses	X	0	0	0
		Case 2: TCTRL*n*[MODE] = 11 • Useful for determining the duration between pulses • Selecting a fast timer clock provides accurate measurements but it can also cause timer rollover between pulses	X	1	0	0

Table continues on the next page...

Table 1077. Timing diagrams (continued)

Mode (TCTRL*n*[MODE]) setting		Timing diagram	TCTRL*n*[TSOT]	TCTRL*n*[TROT]	TCTRL*n*[TSOI]	TCTRL*n*[CHAIN]
		Case 3: TCTRL*n*[MODE] = 11 • One-Shot Timer Count mode • You can enable it again by writing 1 to TCTRL*n*[T_EN]	X	0	1	0
Timer chaining: effects on timing operations	XX	Timer chaining	X	X	X	1

71.3.8.1 Case 1: TCTRL*n*[MODE] = 0

The following figure represents case 1 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 0
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0

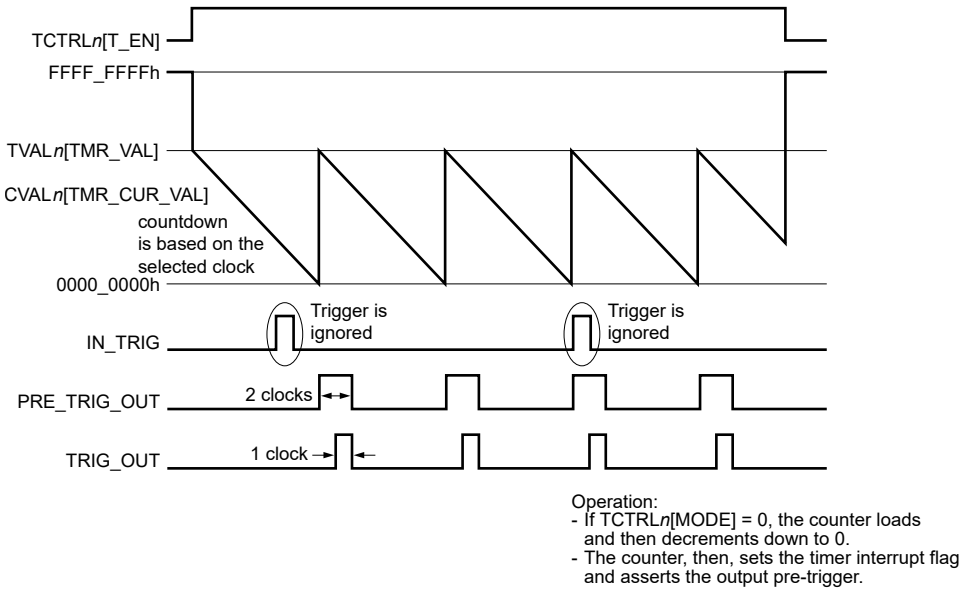


Figure 557. Case 1: TSOT = 0, TROT = 0, TSOI = 0, CHAIN = 0

71.3.8.2 Case 2: TCTRL*n*[MODE] = 0

The following figure represents case 2 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 0
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0

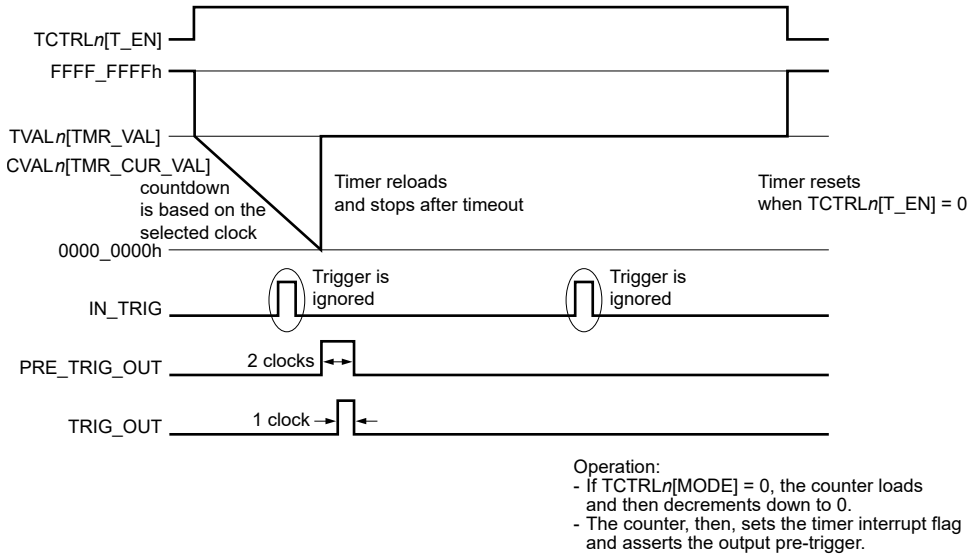


Figure 558. Case 2: TSOT = 0, TROT = 0, TSOI = 1, CHAIN = 0

71.3.8.3 Case 3: TCTRL*n*[MODE] = 0

The following figure represents case 3 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 0
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0

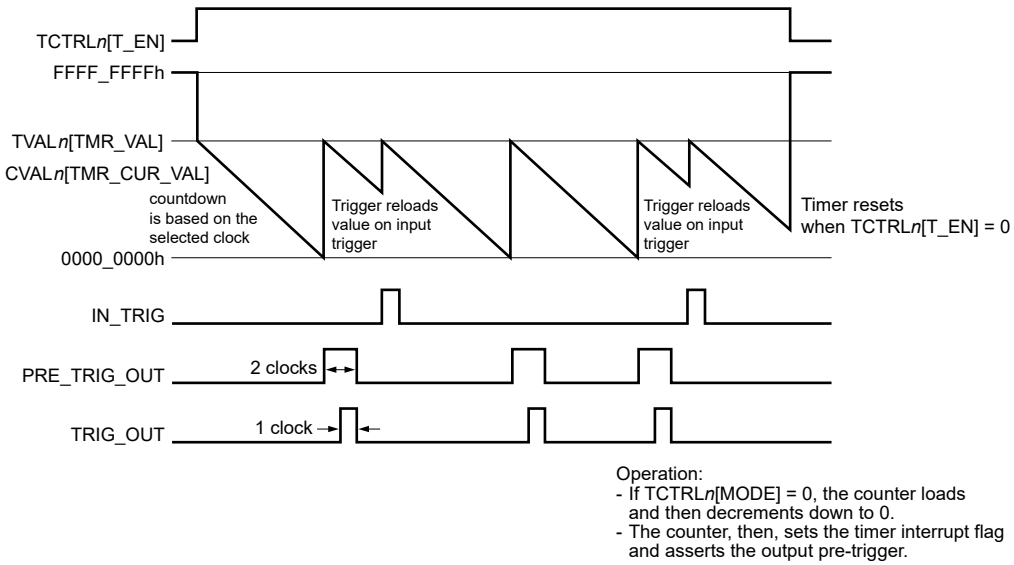


Figure 559. Case 3: TSOT = 0, TROT = 1, TSOI = 0, CHAIN = 0

71.3.8.4 Case 4: TCTRL*n*[MODE] = 0

The following figure represents case 4 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 0
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0

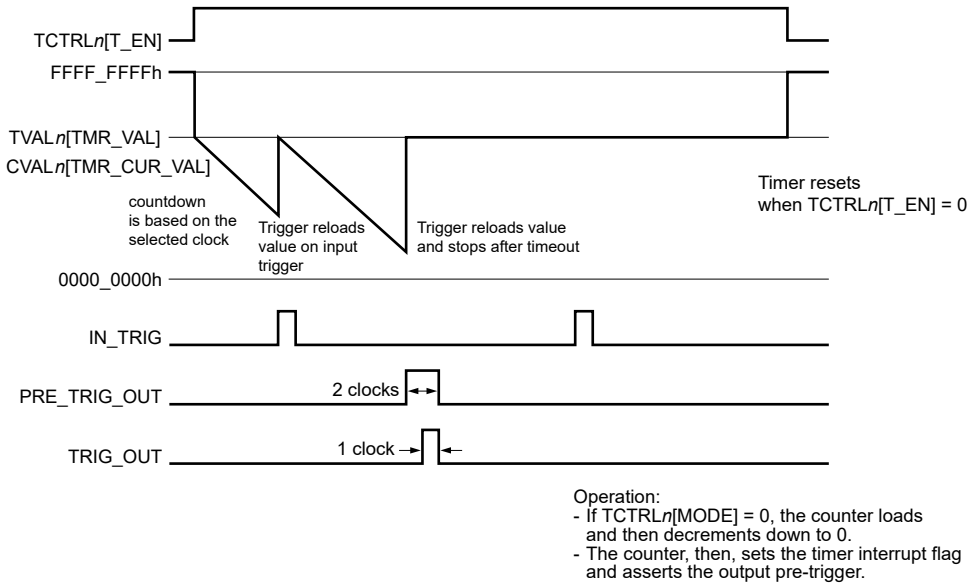


Figure 560. Case 4: TSOT = 0, TROT = 1, TSOI = 1, CHAIN = 0

71.3.8.5 Case 5: TCTRL*n*[MODE] = 0

The following figure represents case 5 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 1
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0

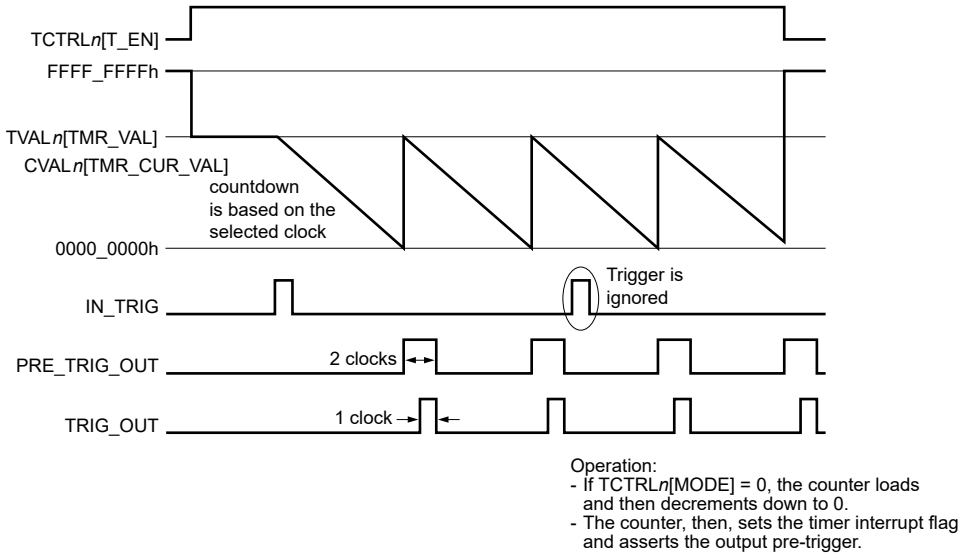


Figure 561. Case 5: TSOT = 1, TROT = 0, TSOI = 0, CHAIN = 0

71.3.8.6 Case 6: TCTRL*n*[MODE] = 0

The following figure represents case 6 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 1
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0

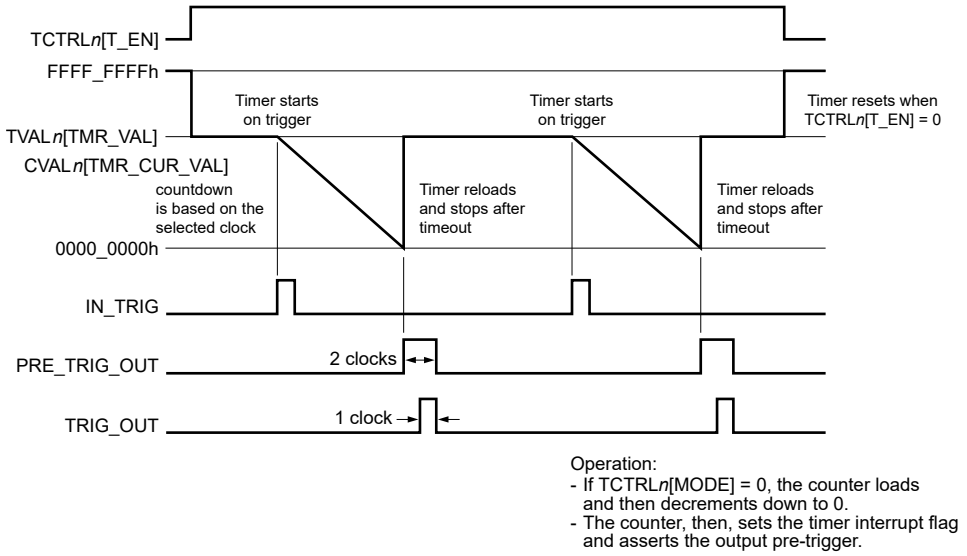


Figure 562. Case 6: TSOT = 1, TROT = 0, TSOI = 1, CHAIN = 0

71.3.8.7 Case 7: TCTRL*n*[MODE] = 0

The following figure represents case 7 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 1
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0

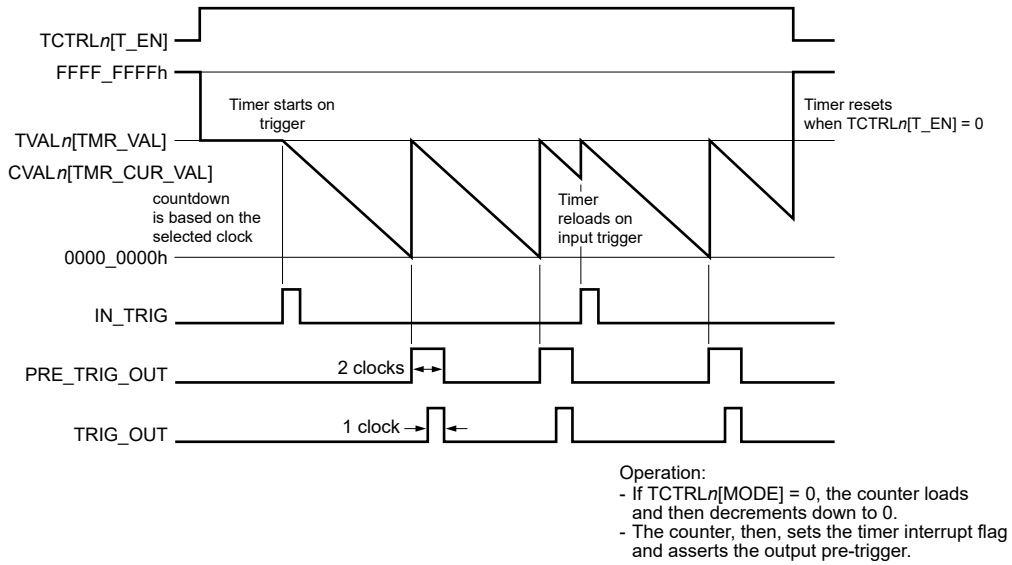


Figure 563. Case 7: TSOT = 1, TROT = 1, TSOI = 0, CHAIN = 0

71.3.8.8 Case 8: TCTRL*n*[MODE] = 0

The following figure represents case 8 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter).

LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 1
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0

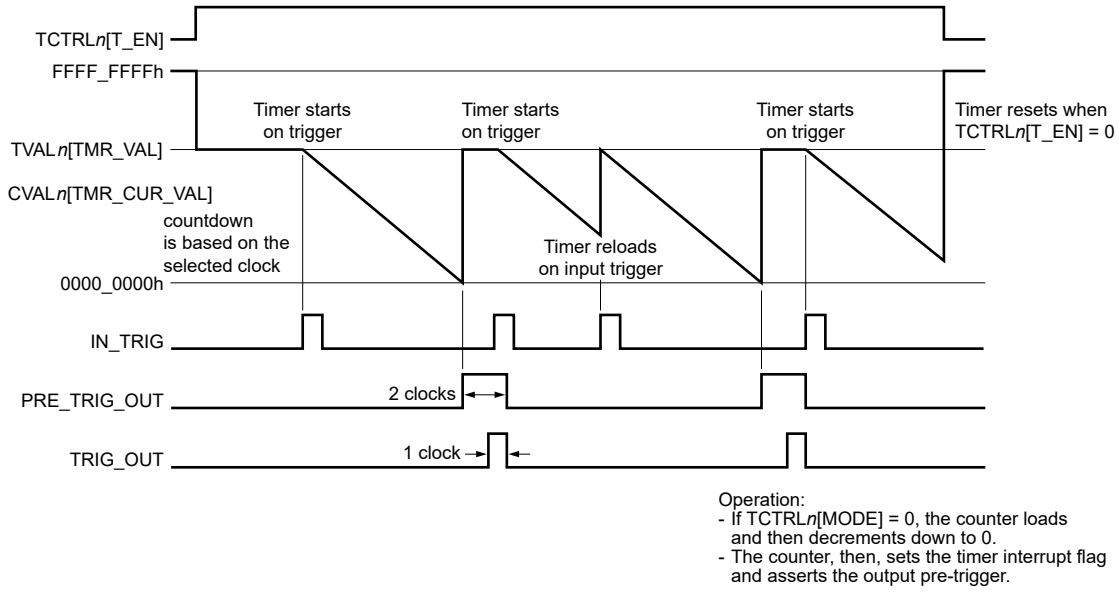


Figure 564. Case 8: TSOT = 1, TROT = 1, TSOI = 1, CHAIN = 0

71.3.8.9 Case 1: TCTRL*n*[MODE] = 1

The following figure represents case 1 in which TCTRL*n*[MODE] = 1 (16-bit dual periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 0
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0

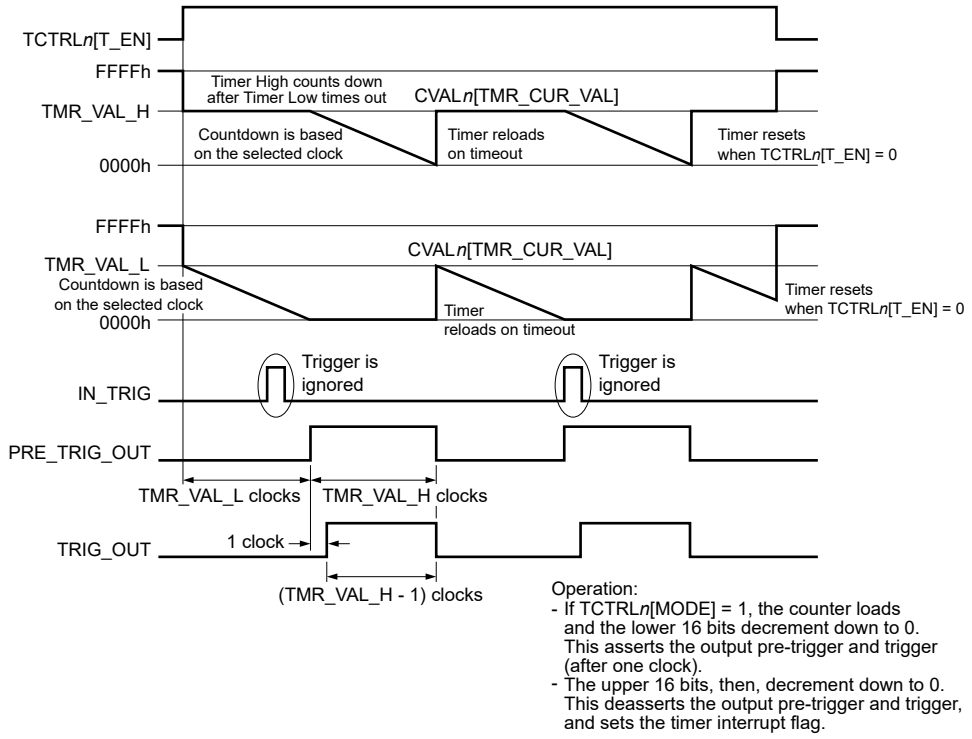


Figure 565. Case 1: TSOT = 0, TROT = 0, TSOI = 0, CHAIN = 0

If TCTRL*n*[MODE] = 1:

- The effect of timing control fields is similar to the effect of timer control fields when TCTRL*n*[MODE] = 0. See the individual timing diagrams for cases with TCTRL*n*[MODE] = 0 for more information.
- The timer interrupt (timeout) asserts when {TMR_H,TMR_L} = 0000_0000h.

See the following table for the behavior of timer control fields when TCTRL*n*[MODE] = 1.

Table 1078. Timer control fields when TCTRL*n*[MODE] = 1

TCTRL*n*[TSOT]	TCTRL*n*[TROT]	TCTRL*n*[TSOI]	Function	Effect on timer
0	0	0	• For repeated interrupts with reload • Trigger outputs have equal periods	Similar to Case 1: TCTRL*n*[MODE] = 1.
0	0	1	One-shot mode	• Similar to Case 2: TCTRL*n*[MODE] = 0. • Both timers stop after first count down and then time out. • Timers do not count again until TCTRL*n*[T_EN] becomes 1.
0	1	0	• For repeated interrupts with reload • Trigger outputs have unequal periods	• Similar to Case 3: TCTRL*n*[MODE] = 0 . • Both timers reload the value of TVAL*n*[TMR_VAL] after the trigger rising edge.

Table continues on the next page...

Table 1078. Timer control fields when TCTRL*n*[MODE] = 1 (continued)

TCTRL*n*[TSOT]	TCTRL*n*[TROT]	TCTRL*n*[TSOI]	Function	Effect on timer
				• Output triggers clear after reload, if asserted.
0	1	1	Reloadable one-shot mode	• Similar to Case 4: TCTRL*n*[MODE] = 0. • If a trigger occurs before timeout, then both timers reload and count down (as shown); the timers stop after timeout. • A trigger assertion after timeout reloads the value of TVAL*n*[TMR_VAL] into the timers. • The timers do not count again until TCTRL*n*[T_EN] becomes 1 again.
1	0	0	• For generating periodic interrupts after a predefined event (input trigger) • Output triggers have equal periods	• Similar to Case 5: TCTRL*n*[MODE] = 0. • After TCTRL*n*[T_EN] rises, the timers do not start until after the first trigger's rising edge. • Subsequent triggers have no effect.
1	0	1	• Triggered one-shot timer mode • Output trigger period depends on the input trigger	• Similar to Case 6: TCTRL*n*[MODE] = 0. • After TCTRL*n*[T_EN] becomes 1, the timers do not start until after the first trigger's rising edge. • The timer stops counting after a timeout assertion. • The timer does not start counting again until a new trigger's rising edge is detected.
1	1	0	• For repeated interrupts with reload timer mode • Output triggers have unequal periods	• Similar to Case 7: TCTRL*n*[MODE] = 0. • After TCTRL*n*[T_EN] becomes 1, the timers do not start until the first trigger's rising edge. • Subsequent triggers cause the timer to reload the value of TVAL*n*[TMR_VAL] into both counters. • The output triggers clear after a reload, if asserted.
1	1	1	• For a nonperiodic input trigger • If input trigger is periodic and greater than timer timeout, then it is the	• Similar to Case 8: TCTRL*n*[MODE] = 0. • After TCTRL*n*[T_EN] becomes 1, the timers do not start until the first trigger's rising edge.

Table continues on the next page...

Table 1078. Timer control fields when TCTRL*n*[MODE] = 1 (continued)

TCTRL*n*[TSOT]	TCTRL*n*[TROT]	TCTRL*n*[TSOI]	Function	Effect on timer
			same as TCTRL*n*[TROT] = 0 (which is triggered one-shot timer mode; output trigger period depends on the input trigger)	• The timers stop counting after a timeout assertion. • A trigger's rising edge causes the timers to reload and then count down.

71.3.8.10 Case 1: TCTRL*n*[MODE] = 10

The following figure represents case 1 in which TCTRL*n*[MODE] = 10. LPIT works in 32-bit trigger accumulator mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = X
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0

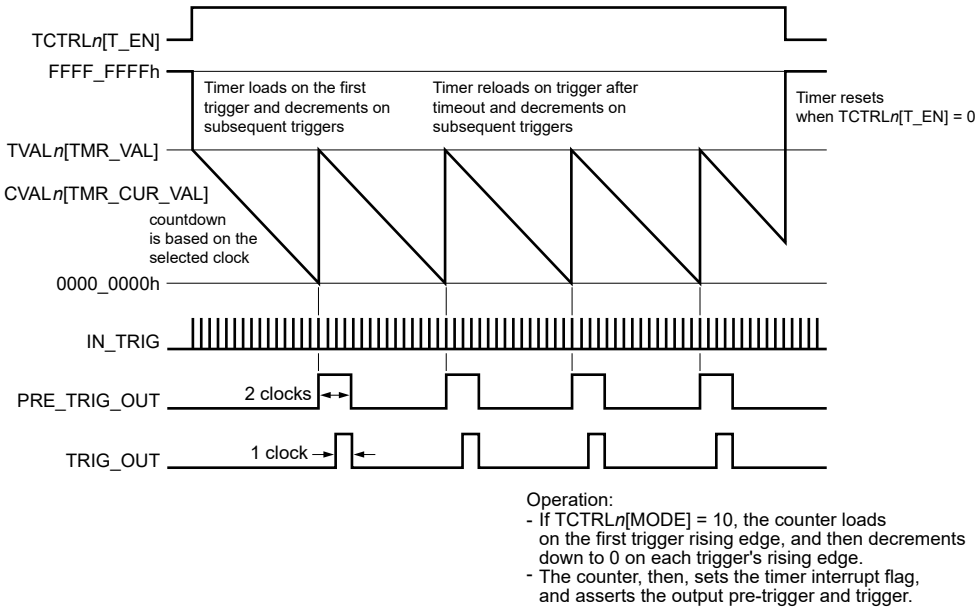


Figure 566. Case 1: TSOT = X, TROT = X, TSOI = 0, CHAIN = 0

71.3.8.11 Case 2: TCTRL*n*[MODE] = 10

The following figure represents case 2 in which TCTRL*n*[MODE] = 10. LPIT works in 32-bit trigger accumulator mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = X
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0

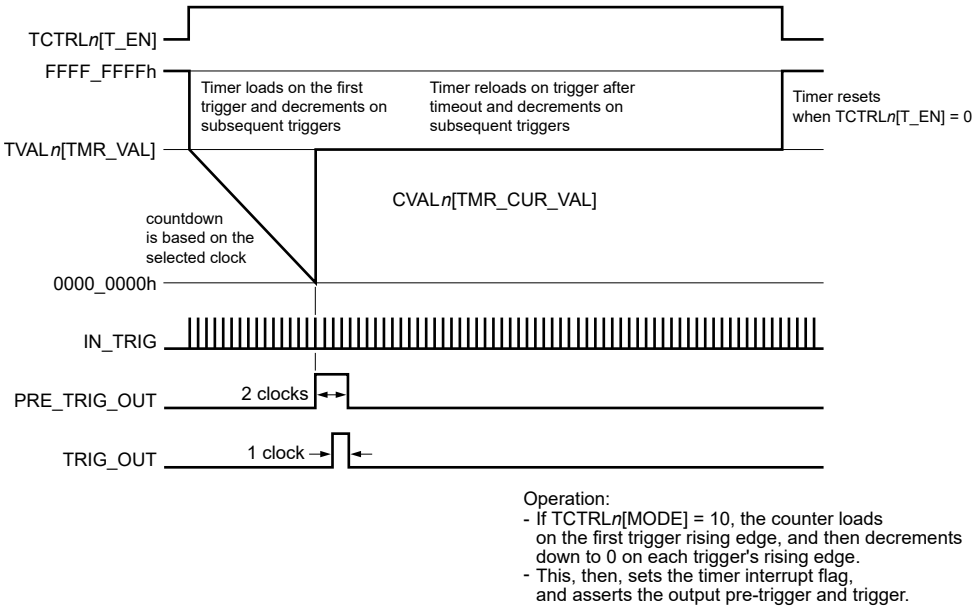


Figure 567. Case 2: TSOT = X, TROT = X, TSOI = 1, CHAIN = 0

71.3.8.12 Case 1: TCTRL*n*[MODE] = 11

The following figure represents case 1 in which TCTRL*n*[MODE] = 11. LPIT works in 32-bit trigger capture mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0

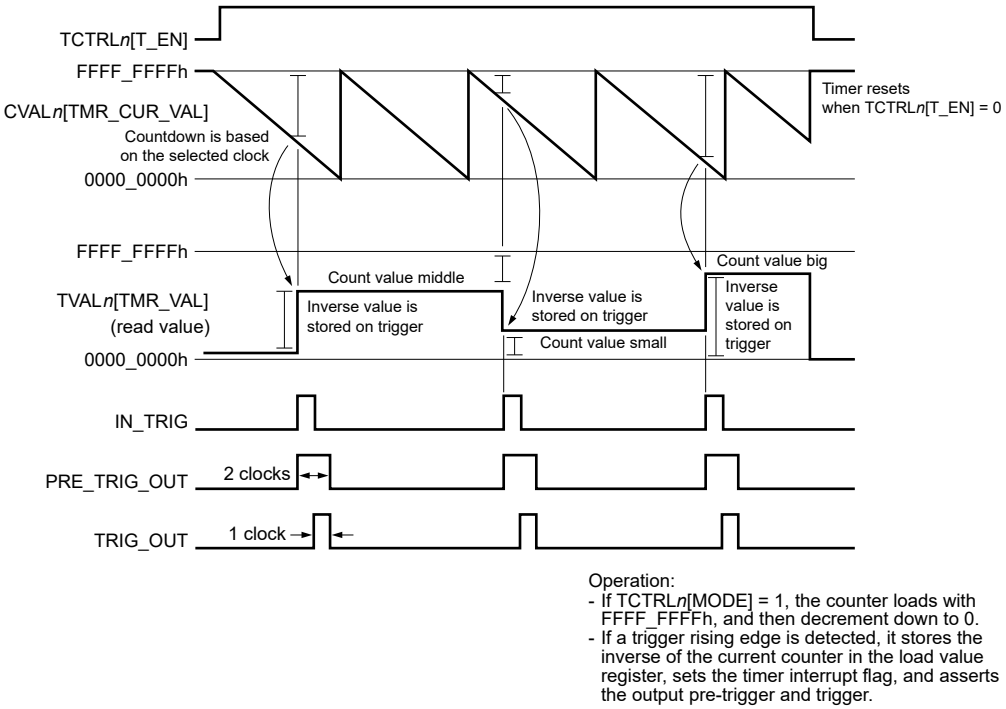


Figure 568. Case 1: TSOT = X, TROT = 0, TSOI = 0, CHAIN = 0

71.3.8.13 Case 2: TCTRL*n*[MODE] = 11

The following figure represents case 2 in which TCTRL*n*[MODE] = 11. LPIT works in 32-bit trigger capture mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0

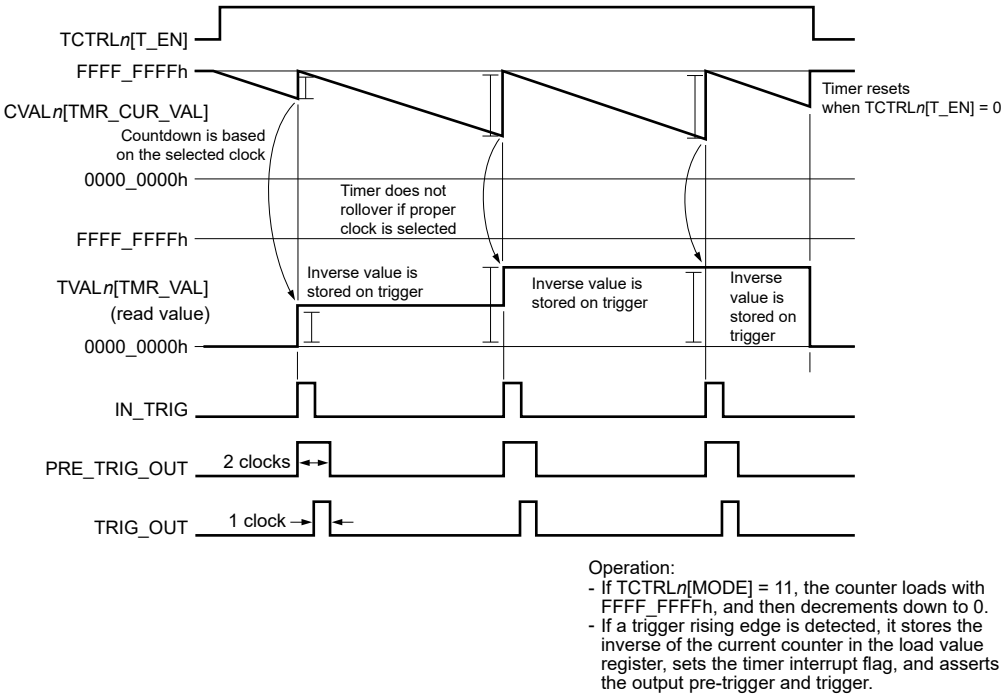


Figure 569. Case 2: TSOT = X, TROT = 1, TSOI = 0, CHAIN = 0

71.3.8.14 Case 3: TCTRL*n*[MODE] = 11

The following figure represents case 3 in which TCTRL*n*[MODE] = 11. LPIT works in 32-bit trigger capture mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0

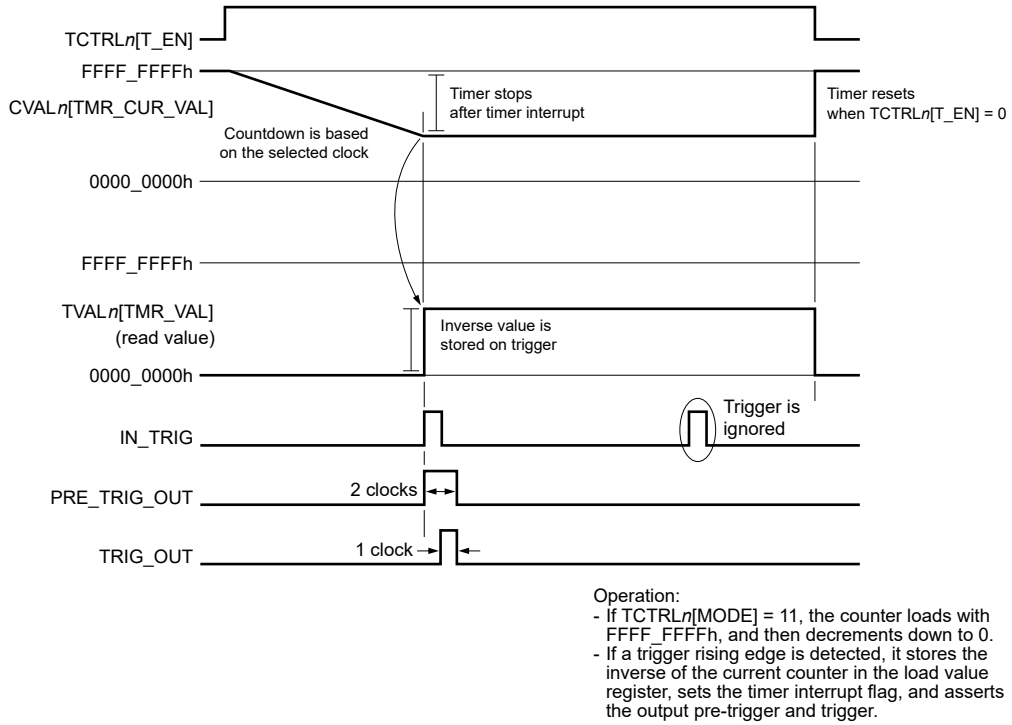


Figure 570. Case 3: TSOT = X, TROT = 0, TSOI = 1, CHAIN = 0

71.3.8.15 Case 4: TCTRL*n*[MODE] = 11

Case 4, in which TCTRL*n*[MODE] = 11, is the same as case 3, except that the timer reloads to FFFF_FFFFh and then stops. The timer does not start until TCTRL*n*[T_EN] becomes 1 again. In this case, LPIT works in 32-bit trigger capture mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0

71.3.9 Timer chaining

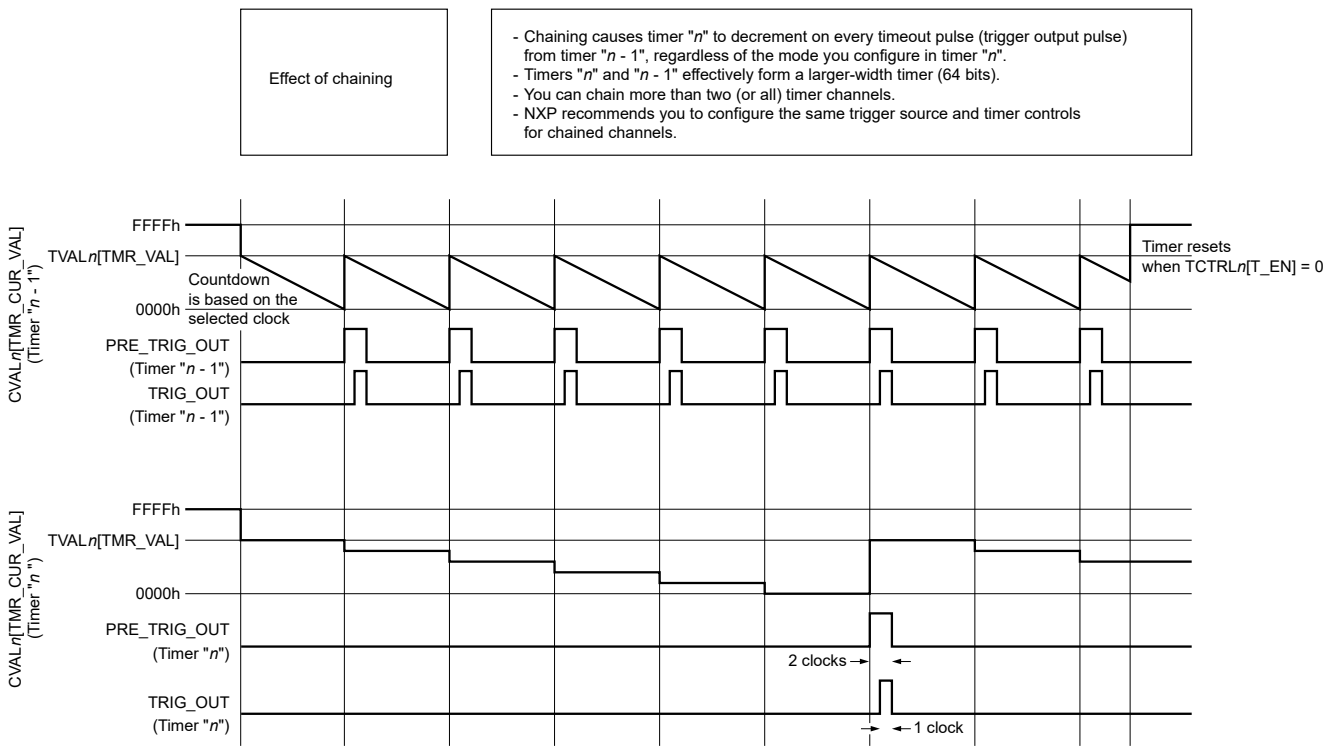


Figure 571. Chaining effects

71.4 Initialization

Table 1079. Initializing LPIT

Step	Action	How or why to perform the step
1	Enable the peripheral clock	By writing 1 to MCR[M_CEN]. **NOTE** • Accessing certain registers (Module Status (MSR), Set Timer Enable (SETTEN), Clear Timer Enable (CLRTEN), Timer Value (TVAL0 - TVAL3), Current Timer Value (CVAL0 - CVAL3), and Timer Control (TCTRL0 - TCTRL3)) while MCR[M_CEN] = 0 leads to the assertion of a transfer error for that bus access. However, writing to CVAL*n* and reserved registers generates a transfer error. • There might be additional clock gating fields in the chip that gate the peripheral clock to this module. When enabling the clock to this module, you must configure those additional clock gating fields (in addition to configuring MCR[M_CEN]).
2	Wait for four peripheral clock cycles	To allow time for clock synchronization and reset deassertion.

Table continues on the next page...

Table 1079. Initializing LPIT (continued)

Step	Action	How or why to perform the step
3	Configure timer control fields	For each timer channel that is to be enabled: • Timer mode of operation fields, TCTRL*n*[MODE] • Trigger source selection fields, TCTRL*n*[TRG_SEL] and TCTRL*n*[TRG_SRC] • Trigger control fields, TCTRL*n*[TROT], TCTRL*n*[TSOT], and TCTRL*n*[TSOI] **NOTE** You must not update timer control fields when the timer is disabled. You can disable a timer by using any one of the following methods: • Write 1 to the specific timer's CLRTEN[CLR_T_EN_*n*] field. • Write 0 to TCTRL*n*[T_EN] for that channel.
4	Configure the channels that are to be chained	By writing 1 to TCTRL*n*[CHAIN] in the corresponding channel's Timer Control (TCTRL0 - TCTRL3).
5	Set the timer timeout value	By programming an appropriate value in TVAL*n*[TMR_VAL] for the channels that you configure in Compare mode.
6	Configure MIER[TIE*n*]	For those channels that are required to generate interrupts after timer timeouts.
7	Configure the low-power modes of the module	By writing 1 to MCR[DBG_EN] and MCR[DOZE_EN]. This is common to all timer channels.
8	Enable the channel timers	By writing 1 to the corresponding TCTRL*n*[T_EN].

NOTE

When you enable a timer channel in Compare mode, the first decrement takes an additional one or two clock cycles because of synchronization logic. This results in the first compare (and therefore interrupt and hardware trigger) occurring slightly later. A faster counter clock minimizes this impact.

Additionally,

- For channels that you configure in Capture mode, you can read the timer value from Timer Value (TVAL0 - TVAL3) when a channel timeout occurs.
- At any time, you can read the current value of the timer for any channel by reading the corresponding channel's Current Timer Value (CVAL0 - CVAL3). Ensure that MCR[M_CEN] = 1 when doing so.
- MSR[TIF*n*] are asserted after timer timeout. To clear these timer interrupt flags, write 1 to them.

71.5 Memory map and registers

71.5.1 LPIT register descriptions

The LPIT memory map comprises 32-bit aligned registers, which you can access via 8-bit, 16-bit, or 32-bit accesses. Read and write accesses to reserved locations generate a transfer error, and the read bus shows all 0s.

NOTE

- The memory map and complete module are in big-endian (BE) format.
- LPIT does not check whether programmed values in these registers are correct—you must write the correct values.

71.5.1.1 LPIT memory map

LPIT1 base address: 442F_0000h

LPIT2 base address: 424C_0000h

LPIT3 base address: 42CC_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0000h
4h	Parameter (PARAM)	32	R	0000_0404h
8h	Module Control (MCR)	32	RW	0000_0000h
Ch	Module Status (MSR)	32	RW	0000_0000h
10h	Module Interrupt Enable (MIER)	32	RW	0000_0000h
14h	Set Timer Enable (SETTEN)	32	RW	0000_0000h
18h	Clear Timer Enable (CLRTEN)	32	RW	0000_0000h
20h	Timer Value (TVAL0)	32	RW	0000_0000h
24h	Current Timer Value (CVAL0)	32	R	FFFF_FFFFh
28h	Timer Control (TCTRL0)	32	RW	0000_0000h
30h	Timer Value (TVAL1)	32	RW	0000_0000h
34h	Current Timer Value (CVAL1)	32	R	FFFF_FFFFh
38h	Timer Control (TCTRL1)	32	RW	0000_0000h
40h	Timer Value (TVAL2)	32	RW	0000_0000h
44h	Current Timer Value (CVAL2)	32	R	FFFF_FFFFh
48h	Timer Control (TCTRL2)	32	RW	0000_0000h
50h	Timer Value (TVAL3)	32	RW	0000_0000h
54h	Current Timer Value (CVAL3)	32	R	FFFF_FFFFh
58h	Timer Control (TCTRL3)	32	RW	0000_0000h

71.5.1.2 Version ID (VERID)

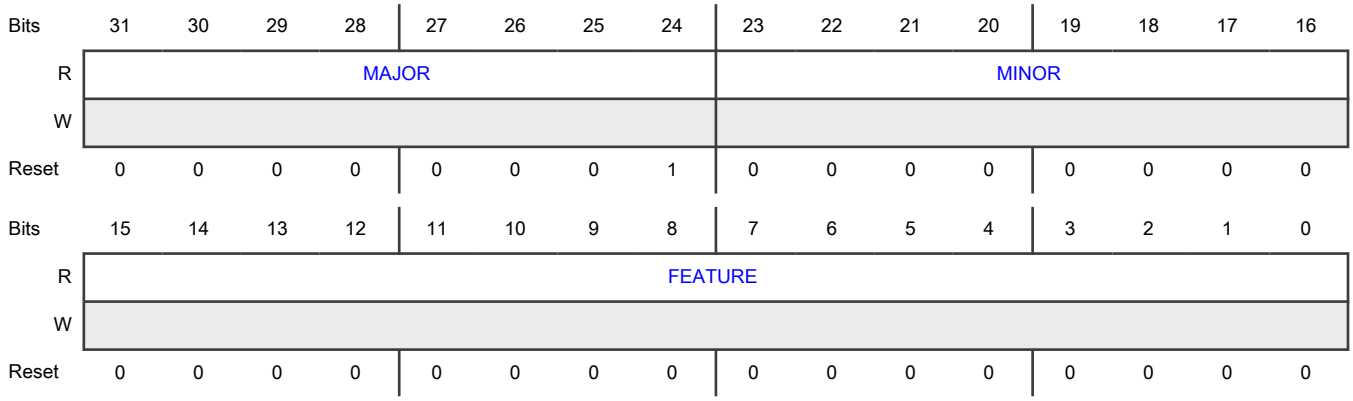
Offset

Register	Offset
VERID	0h

Function

Contains design version specification numbers.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the module design specification.
23-16 MINOR	Minor Version Number Indicates the minor version number for the module design specification.
15-0 FEATURE	Feature Number Indicates the feature set number.

71.5.1.3 Parameter (PARAM)

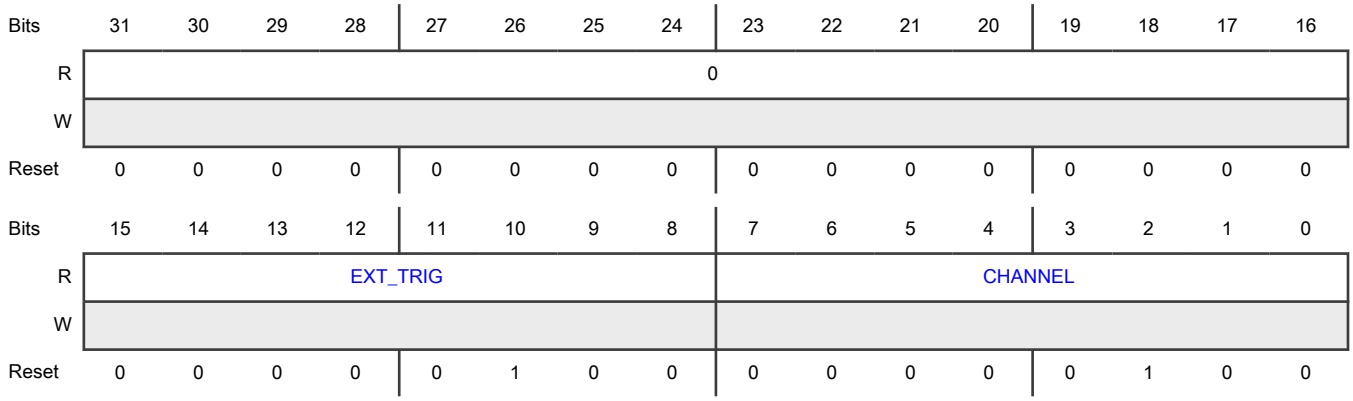
Offset

Register	Offset
PARAM	4h

Function

Provides parameter settings that are used when incorporating this module into the chip.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 EXT_TRIG	Number of External Trigger Inputs Specifies the number of external triggers implemented in this chip.
7-0 CHANNEL	Number of Timer Channels Specifies the number of timer channels implemented in this chip.

71.5.1.4 Module Control (MCR)

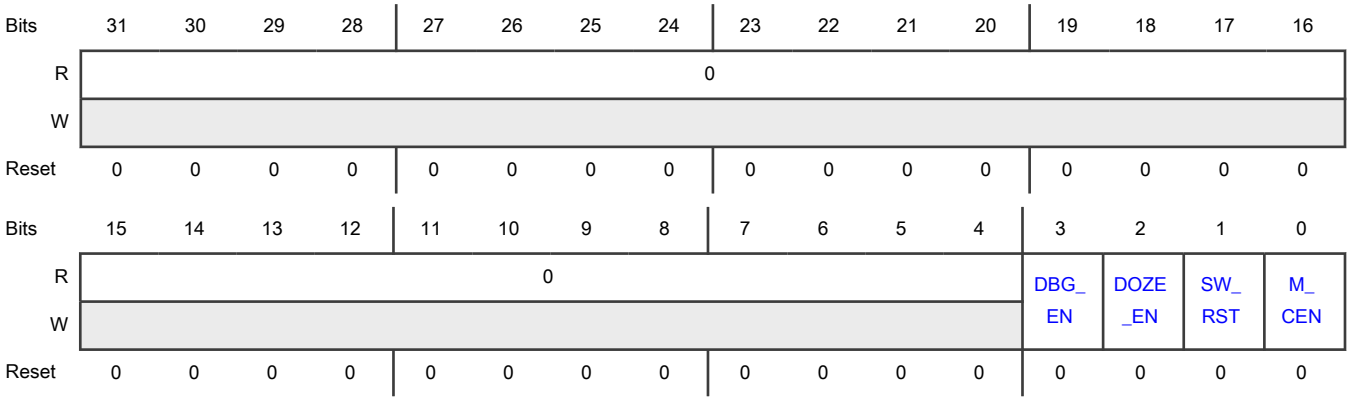
Offset

Register	Offset
MCR	8h

Function

Contains software reset, clock enable, and mode enable fields.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 DBG_EN	Debug Mode Enable Stops the timer channels when the chip enters Debug mode. 0b - Stops timer channels 1b - Allows timer channels to continue running
2 DOZE_EN	DOZE Mode Enable Stops the timer channels when the chip enters Doze mode. 0b - Stops timer channels 1b - Allows timer channels to continue running
1 SW_RST	Software Reset Resets all timer channels and registers, except Module Status (MSR). This field remains 1 until software clears it. Before clearing this field, software must wait for four peripheral clocks (for clock synchronization and reset propagation). 0b - Does not reset 1b - Resets
0 M_CEN	Module Clock Enable Enables the peripheral clock to LPIT module timers. This field must become 1 when accessing the following registers: • Module Status (MSR) • Set Timer Enable (SETTEN) • Clear Timer Enable (CLRTEN)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Timer Value (TVAL0 - TVAL3) • Current Timer Value (CVAL0 - CVAL3) • Timer Control (TCTRL0 - TCTRL3) The following considerations apply when using this field: • You must enable both the bus and peripheral clocks to allow clock synchronization and update of the aforementioned registers. Accessing these registers when MCR[M_CEN] = 0 asserts a transfer error for that bus cycle. • Writing to Current Timer Value (CVAL0 - CVAL3) and reserved registers always generates a transfer error. **NOTE** There may be additional clock gating fields available in this chip that gate the peripheral clock to LPIT. You must configure those additional clock gating fields appropriately to enable the peripheral clock to LPIT. 0b - Disable 1b - Enable

71.5.1.5 Module Status (MSR)

Offset

Register	Offset
MSR	Ch

Function

Contains channel timer interrupt flags.

NOTE

Unless the peripheral clock to the timers is enabled (MCR[M_CEN] = 1), reading or writing to this register generates a transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												TIF3	TIF2	TIF1	TIF0
W													W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3 TIF3	Channel 3 Timer Interrupt Flag Specifies whether the channel 3 timer has timed out. In compare modes, at the end of the timer period, this flag becomes 1. In capture modes, when the trigger asserts, this flag becomes 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not timed out 1b - Timed out When writing 0b - No effect 1b - Clear the flag
2 TIF2	Channel 2 Timer Interrupt Flag Specifies whether the channel 2 timer has timed out. In compare modes, at the end of the timer period, this flag becomes 1. In capture modes, when the trigger asserts, this flag becomes 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not timed out

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Timed out When writing 0b - No effect 1b - Clear the flag
1 TIF1	Channel 1 Timer Interrupt Flag Specifies whether the channel 1 timer has timed out. In compare modes, this flag becomes 1 at the end of the timer period. In capture modes, this flag becomes 1 when the trigger asserts. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not timed out 1b - Timed out When writing 0b - No effect 1b - Clear the flag
0 TIF0	Channel 0 Timer Interrupt Flag Specifies whether the channel 0 timer has timed out. In compare modes, this flag becomes 1 at the end of the timer period. In capture modes, this flag becomes 1 when the trigger asserts. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not timed out 1b - Timed out When writing 0b - No effect 1b - Clear the flag

71.5.1.6 Module Interrupt Enable (MIER)

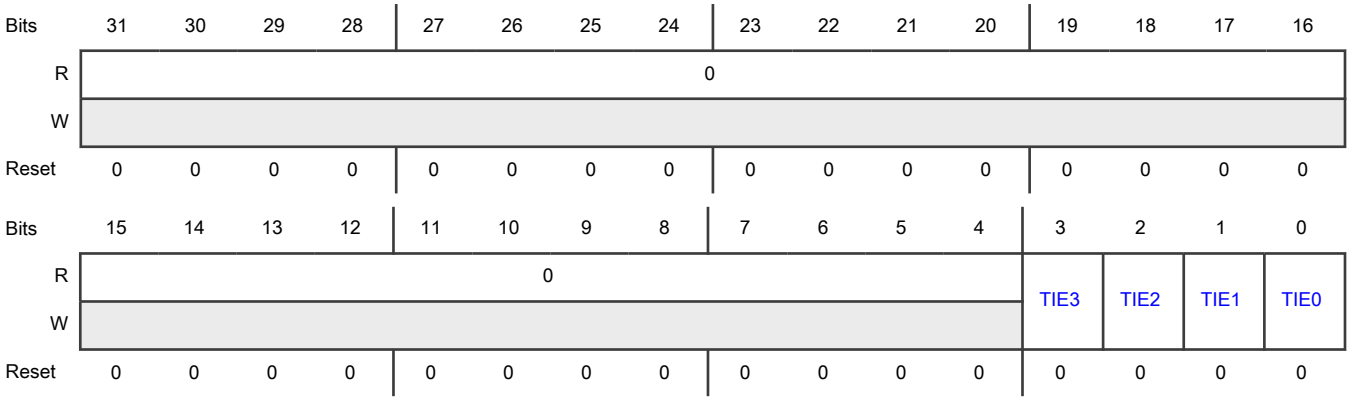
Offset

Register	Offset
MIER	10h

Function

Contains channel timer interrupt enable fields.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 TIE3	Channel 3 Timer Interrupt Enable Enables interrupt generation when: • This field = 1. • The corresponding timer interrupt flag, MSR[TIF3] = 1. 0b - Disable 1b - Enable
2 TIE2	Channel 2 Timer Interrupt Enable Enables interrupt generation when: • This field = 1. • The corresponding timer interrupt flag, MSR[TIF2] = 1. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 TIE1	Channel 1 Timer Interrupt Enable Enables interrupt generation when: • This field = 1. • The corresponding timer interrupt flag, MSR[TIF1] = 1. 0b - Disable 1b - Enable
0 TIE0	Channel 0 Timer Interrupt Enable Enables interrupt generation when: • This field = 1. • The corresponding timer interrupt flag, MSR[TIF0] = 1. 0b - Disable 1b - Enable

71.5.1.7 Set Timer Enable (SETTEN)

Offset

Register	Offset
SETTEN	14h

Function

Allows the simultaneous enabling of timer channels.

You can enable timer channels by using either of the following ways:

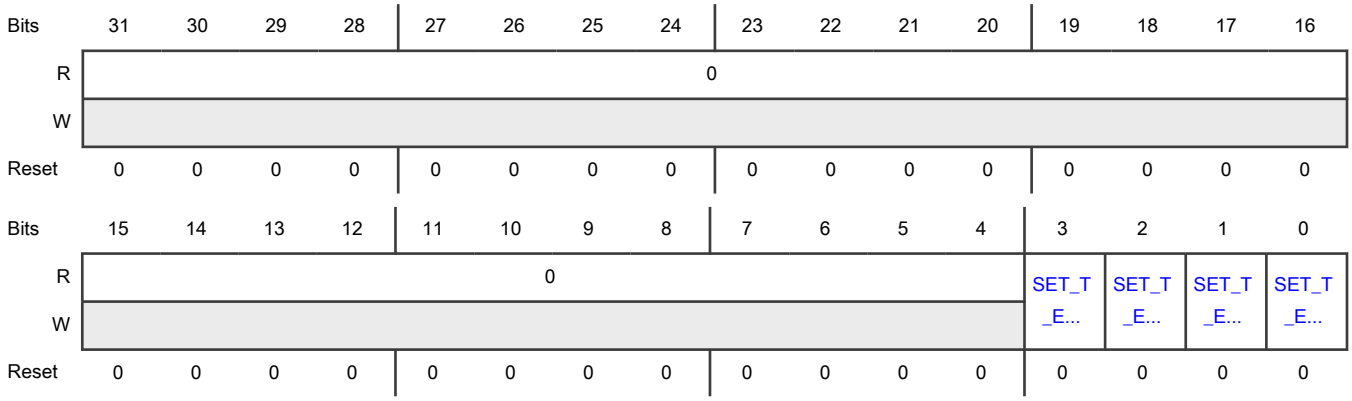
- Writing 1 to TCTRL*n*[T_EN] in the respective TCTRL*n* register.
- Writing 1 to the corresponding SETTEN[SET_T_EN_*n*] field.

To disable timer channels simultaneously, use Clear Timer Enable (CLRTEN). Writing 0 to the fields of this register has no effect.

NOTE

Unless the peripheral clock to the timers is enabled (MCR[M_CEN] = 1), reading or writing to this register generates a transfer error.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 SET_T_EN_3	Set Timer 3 Enable Works with TCTRL3[T_EN] and enables timer channel 3. Writing 0 to this field does not disable the counter; rather it has no effect. This field becomes 0 if any of the following conditions is true: • TCTRL3[T_EN] = 0 • You write 1 to CLRTEN[CLR_T_EN_3] 0b - No effect 1b - Enables timer channel 3
2 SET_T_EN_2	Set Timer 2 Enable Works with TCTRL2[T_EN] and enables timer channel 2. Writing 0 to this field does not disable the counter; rather it has no effect. This field becomes 0 if any of the following conditions is true: • TCTRL2[T_EN] = 0 • You write 1 to CLRTEN[CLR_T_EN_2] 0b - No Effect 1b - Enables timer channel 2
1 SET_T_EN_1	Set Timer 1 Enable Works with TCTRL1[T_EN] and enables timer channel 1. Writing 0 to this field does not disable the counter; rather it has no effect. This field becomes 0 if any of the following conditions is true:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• TCTRL1[T_EN] = 0 • You write 1 to CLRTEN[CLR_T_EN_1] 0b - No Effect 1b - Enables timer channel 1
0 SET_T_EN_0	Set Timer 0 Enable Works with TCTRL0[T_EN] and enables timer channel 0. Writing 0 to this field does not disable the counter; rather it has no effect. This field becomes 0 if any of the following conditions is true: • TCTRL0[T_EN] = 0 • You write 1 to CLRTEN[CLR_T_EN_0] 0b - No effect 1b - Enables timer channel 0

71.5.1.8 Clear Timer Enable (CLRTEN)

Offset

Register	Offset
CLRTEN	18h

Function

Allows the simultaneous disabling of timer channels.

NOTE

Unless the peripheral clock to the timers is enabled (MCR[M_CEN] = 1), reading or writing to this register generates a transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	0	0	0
W													CLR_T_E...	CLR_T_E...	CLR_T_E...	CLR_T_E...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3 CLR_T_EN_3	Clear Timer 3 Enable Works with TCTRL3[T_EN] and disables timer channel 3. Writing 1 to this self-clearing field does not enable the counter. It disables timer channel 3. It also turns TCTRL3[T_EN] = 0 for timer channel 3. 0b - No action 1b - Turns TCTRL3[T_EN] = 0 for timer channel 3
2 CLR_T_EN_2	Clear Timer 2 Enable Works with TCTRL2[T_EN] and disables timer channel 2. Writing 1 to this self-clearing field does not enable the counter. It disables timer channel 2. It also turns TCTRL2[T_EN] = 0 for timer channel 2. 0b - No action 1b - Turns TCTRL2[T_EN] = 0 for timer channel 2
1 CLR_T_EN_1	Clear Timer 1 Enable Works with TCTRL1[T_EN] and disables timer channel 1. Writing 1 to this self-clearing field does not enable the counter. It disables timer channel 1. It also turns TCTRL1[T_EN] = 0 for timer channel 1. 0b - No action 1b - Turns TCTRL1[T_EN] = 0 for timer channel 1
0 CLR_T_EN_0	Clear Timer 0 Enable Works with TCTRL0[T_EN] and disables timer channel 0. Writing 1 to this self-clearing field does not enable the counter. It disables timer channel 0. It also turns TCTRL0[T_EN] = 0 for timer channel 0. 0b - No action 1b - Turns TCTRL0[T_EN] = 0 for timer channel 0

71.5.1.9 Timer Value (TVAL0 - TVAL3)

Offset

Register	Offset
TVAL0	20h
TVAL1	30h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TVAL2	40h
TVAL3	50h

Function

Contains timer values:

- In compare modes, this register selects the timeout period for the timer channels.
- In capture modes, this register is loaded with the value of the counter when the trigger asserts.

NOTE

Unless the peripheral clock to the timers is enabled (MCR[M_CEN] = 1), reading or writing to this register generates a transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TMR_VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TMR_VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TMR_VAL	Timer Value Specifies the timer value and whether it is valid. • In Compare mode, this field specifies the timer channel start value: — The timer counts down for (TVAL*n* + 1) cycles until the timer reaches 0, then the timer generates an interrupt and loads the TVAL*n* value again. — Writing a new value to TVAL*n* does not restart the timer channel; instead, the new value is loaded "after the timer expires." — To abort the current timer cycle and start a timer period with a new value, you must disable the timer channel and then enable it again. • In Capture mode, whenever the trigger asserts, this register stores the inverse of the counter value.

Table continues on the next page...

Field	Function
	0000_0000_0000_0000_0000_0000_0000_0000b,0000_0000_0000_0000_0000_0000_0000_0001b - Invalid load value in Compare mode 0000_0000_0000_0000_0000_0000_0000_0010b-1111_1111_1111_1111_1111_1111_1111_1111b - In Compare mode: the value to be loaded; in Capture mode, the value of the timer

71.5.1.10 Current Timer Value (CVAL0 - CVAL3)

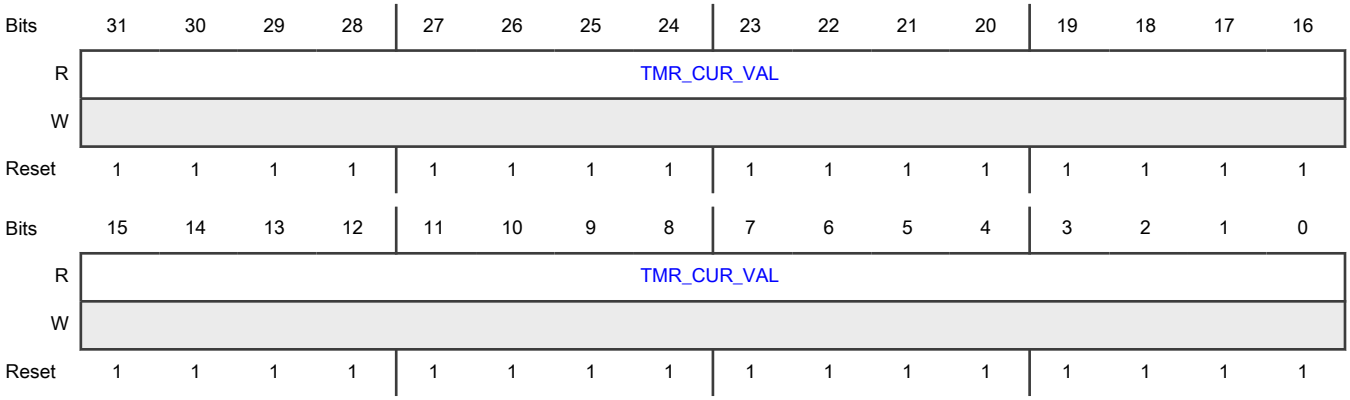
Offset

Register	Offset
CVAL0	24h
CVAL1	34h
CVAL2	44h
CVAL3	54h

Function

Indicates the current timer counter value.

Diagram



Fields

Field	Function
31-0 TMR_CUR_VAL	Current Timer Value Represents the current timer value, if the timer is enabled.

71.5.1.11 Timer Control (TCTRL0 - TCTRL3)

Offset

Register	Offset
TCTRL0	28h
TCTRL1	38h
TCTRL2	48h
TCTRL3	58h

Function

Contains control fields for each timer channel:

- TRG_SEL for trigger selection
- TRG_SRC for trigger source selection
- TROT for timer reload
- TSOI for timer stoppage
- TSOT for timer decrementing
- MODE for timer operation mode selection
- CHAIN for channel chaining

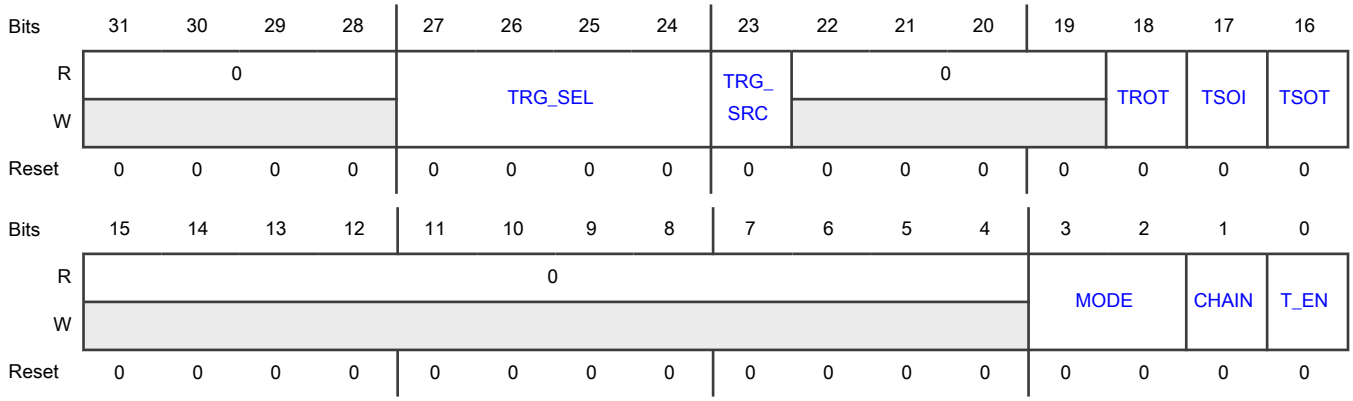
You must update timer controls when the timer is disabled, and use either of the following ways to disable a timer:

- By writing 1 to the specific timer's CLRTEN[CLR_T_EN_*n*] field.
- By writing 0 to T_EN for that channel.

NOTE

Unless the peripheral clock to the timers is enabled (MCR[M_CEN] = 1), reading or writing to this register generates a transfer error.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 TRG_SEL	Trigger Select Selects the trigger to use for starting and/or reloading the LPIT timer. This field selects one trigger from the set of internal or external triggers that TRG_SRC provides. TRG_SRC helps you make a choice between internal and external trigger signals for each channel. **NOTE** You must change the value of this field when the LPIT timer channel is disabled. 0000b-0011b - Timer channel 0–3 trigger source 0100b-1111b - Reserved
23 TRG_SRC	Trigger Source Selects whether to use internal or external trigger sources. You can select the trigger to be used by using this field or the TRG_SEL field. If a channel does not have an associated external trigger, write 1 to TRG_SRC. See LPIT chip-specific information for the available external trigger options. 0b - External 1b - Internal
22-19 —	Reserved
18 TROT	Timer Reload on Trigger Specifies whether the timer reloads after the selected trigger. If this field = 1, the LPIT timer reloads when a rising edge is detected on the selected trigger input. The trigger input is ignored if LPIT is disabled during Debug mode (MCR[DBG_EN] = 0) or Doze mode (MCR[DOZE_EN] = 0). 0b - Does not reload 1b - Reloads
17 TSOI	Timer Stop on Interrupt Controls whether the channel timer stops after being timed out. If this field = 0, the channel timer does not stop after timeout. If this field = 1, the channel timer stops after a timeout and then restarts based on the configuration of TSOT. If TSOT = 0, the channel timer restarts after a rising edge on T_EN is detected, which means that the timer channel is disabled and then enabled. If TSOT = 1, the channel timer restarts after a rising edge on the selected trigger is detected. 0b - Does not stop 1b - Stops

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 TSOT	Timer Start on Trigger Controls when the timer starts decrementing. If this field = 0, the timer starts decrementing immediately based on the restart condition (controlled by TSOI). If this field = 1, the timer starts decrementing when a rising edge on a selected trigger is detected. 0b - Immediately 1b - When a rising edge is detected
15-4 —	Reserved
3-2 MODE	Timer Operation Mode Configures the channel timer's mode of operation. This field controls how the timer decrements. 00b - 32-bit periodic counter 01b - Dual 16-bit periodic counter 10b - 32-bit trigger accumulator 11b - 32-bit trigger input capture
1 CHAIN	Chain Channel Specifies whether channel chaining is enabled. If channel chaining is enabled, the timer channel decrements after the previous channel's timeout (when timer channel N-1 trigger asserts). Timer channel 0 cannot be chained. If channel chaining is disabled, the channel timer runs independently. 0b - Disabled 1b - Enabled
0 T_EN	Timer Enable Enables the timer channel. 0b - Disable 1b - Enable

Chapter 72
Quad Timer (TMR)

72.1 Chip-specific TMR Information

Table 1080. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

72.2 Overview

Each TMR contains four identical counter/timer groups. Each 16-bit counter/timer group contains a prescaler, a counter, a load register, a hold register, a capture register, two compare registers, two status and control registers, and one control register. All of the registers except the prescaler are read/writable.

NOTE

This document uses the terms "Timer" and "Counter" interchangeably because the counter/timers may perform either or both tasks.

The load register provides the initialization value to the counter when the counter's terminal value has been reached.

The hold register captures the counter's value when other counters are being read. This feature supports the reading of cascaded counters.

The capture register enables an external signal to take a "snap shot" of the counter's current value.

The COMP1 and COMP2 registers provide the values to which the counter is compared. If a match occurs, the OFLAG (TMR Output signal) can be set, cleared, or toggled. At match time, an interrupt is generated if enabled, and the new compare value is loaded into the COMP1 or COMP2 registers from CMPLD1 and CMPLD2 if enabled.

The prescaler provides different time bases useful for clocking the counter/timer.

The counter provides the ability to count internal or external events.

Within a TMR (set of four timer/counters), the four inputs are shared by the four counters, but the four outputs are dedicated to each counter.

72.2.1 Block diagram

Each of the timer/counter groups within TMR are shown in this figure.

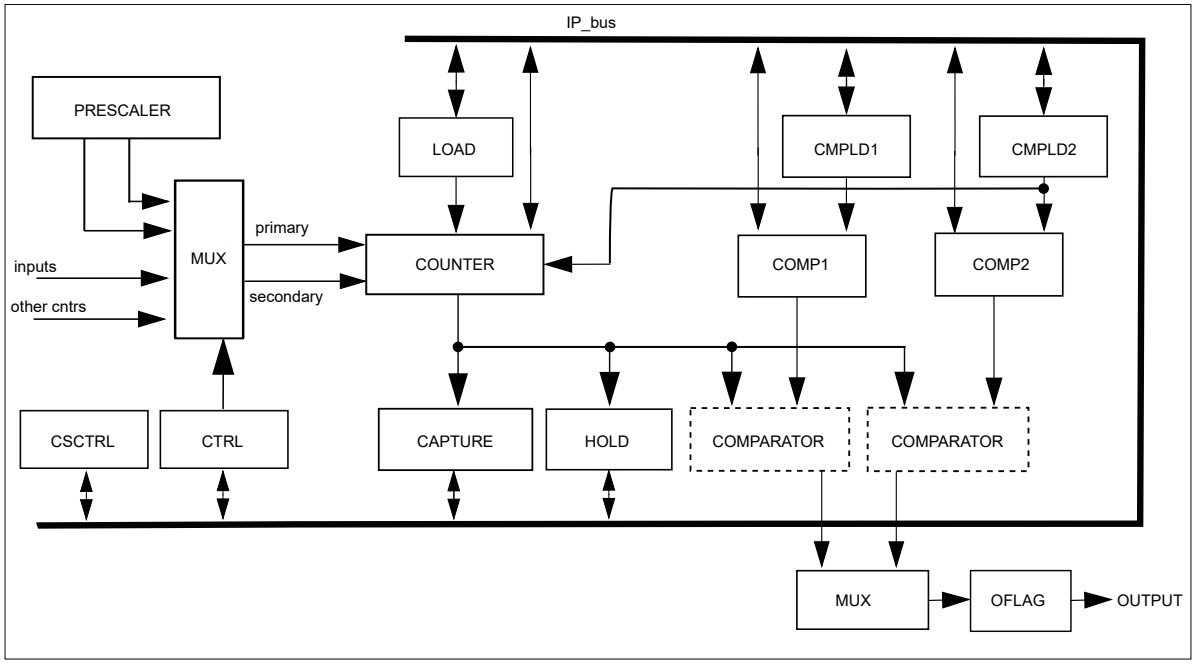


Figure 572. Quad timer block diagram

72.2.2 Features

TMR design includes these features:

- Four 16-bit counters/timers
- Count up/down
- Counters are cascadable
- Programmable count modulo
- Max count rate equals IP Bus Clock 2 for external clocks
- Max count rate equals IP Bus Clock for internal clocks
- Count once or repeatedly
- Counters are preloadable
- Compare registers are preloadable (available with compare load feature)
- Counters can share available input pins
- Separate prescaler for each counter
- Each counter has capture and compare capability
- Programmable operation during debug mode
- Inputs may act as fault inputs
- Programmable input filter
- Counting start can be synchronized across counters

72.3 Functional description

72.3.1 General

The counter/timer has two basic modes of operation: it can count internal or external events, or it can count an internal clock source while an external input signal is asserted, thus timing the width of the external input signal.

- The counter can count the rising, falling, or both edges of the selected input pin.
- The counter can decode and count quadrature encoded input signals.
- The counter can count up and down using dual inputs in a "count with direction" format.
- The counter's terminal count value (modulo) is programmable.
  - The value that is loaded into the counter after reaching its terminal count is programmable.
- The counter can count repeatedly, or it can stop after completing one count cycle.
- The counter can be programmed to count to a programmed value and then immediately reinitialize, or it can count through the compare value until the count "rolls over" to zero.

The external inputs to each counter/timer are shareable among each of the four counter/timers within the module. The external inputs can be used as:

- Count commands
- Timer commands
- They can trigger the current counter value to be "captured"
- They can be used to generate interrupt requests

The polarity of the external inputs are selectable.

The primary output of each timer/counter is the output signal OFLAG. The OFLAG output signal can be:

- Set, cleared, or toggled when the counter reaches the programmed value.
- The OFLAG output signal may be output to an external pin instead of having that pin serve as a timer input.
- The OFLAG output signal enables each counter to generate square waves, PWM, or pulse stream outputs.
- The polarity of the OFLAG output signal is selectable.

Any counter/timer can be assigned as a master. A master's compare signal can be broadcast to the other counter/timers within the module. The other counters can be configured to reinitialize their counters and/or force their OFLAG output signals to predetermined values when a master's counter/timer compare event occurs.

72.3.2 Usage of compare registers

The dual compare registers (COMP1 and COMP2) provide a bidirectional modulo count capability. The COMP1 register is used when the counter is *counting up*, and the COMP2 register is used when the counter is *counting down*. Alternating compare mode is the only exception.

The COMP1 register should be set to the desired maximum count value or FFFFh to indicate the maximum unsigned value prior to roll-over, and the COMP2 register should be set to the minimum count value or 0000h to indicate the minimum unsigned value prior to roll-under.

If CTRL[OUTMODE] is set to 100, the OFLAG will toggle while using alternating compare registers. In this variable frequency PWM mode, the COMP2 value defines the desired pulse width of the on time, and the COMP1 register defines the off time.

Use caution when changing COMP1 and COMP2 while the counter is active. If the counter has already passed the new value, it will count to FFFFh or 0000h, roll over, then begin counting toward the new value. The check is: Count=CMPx, *not* Count > COMP1 or Count < COMP2.

The use of the CMPLD1 and CMPLD2 registers to compare values will help to minimize this problem.

72.3.3 Usage of compare load registers

The CMPLD1, CMPLD2, and CSCTRL registers offer a high degree of flexibility for loading compare registers with user-defined values on different compare events. To ensure correct functionality while using these registers we strongly suggest using the following method described in this section.

The purpose of the compare load feature is to allow quicker updating of the compare registers. In the past, a compare register could be updated using interrupts. However, because of the latency between an interrupt event occurring and the service of that interrupt, there was the possibility that the counter may have already counted past the new compare value by the time the compare register was updated by the interrupt service routine. The counter would then continue counting until it rolled over and reached the new compare value.

To address this, the compare registers are now updated in hardware in the same way the counter register is re-initialized to the value stored in the load register. The compare load feature allows the user to calculate new compare values and store them in to the comparator load registers. When a compare event occurs, the new compare values in the comparator load registers are written to the compare registers eliminating the use of software to do this.

The compare load feature is intended to be used in variable frequency PWM mode. The COMP1 register determines the pulse width for the logic low part of OFLAG and COMP2 determines the pulse width for the logic high part of OFLAG. The period of the waveform is determined by the COMP1 and COMP2 values and the frequency of the primary clock source. See the following figure.

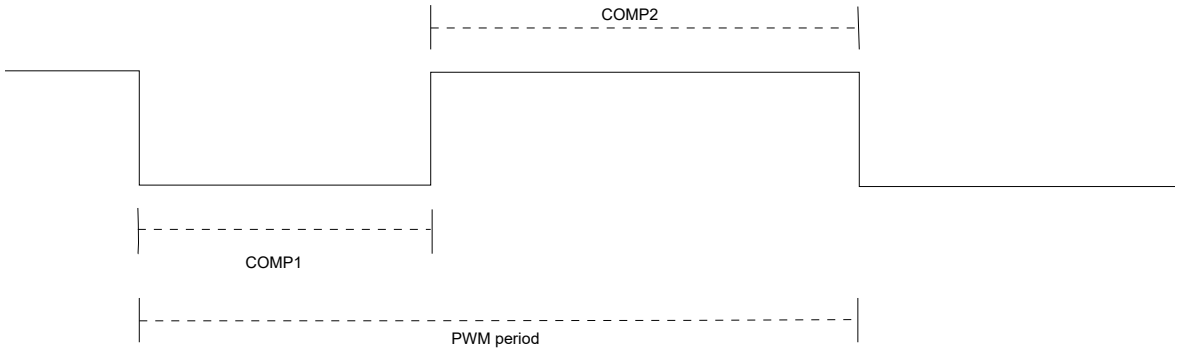


Figure 573. Variable PWM waveform

Should we desire to update the duty cycle or period of the above waveform, we would need to update the COMP1 and COMP2 values using the compare load feature.

72.3.4 Usage of the capture register

The capture register stores a copy of the counter's value when an input edge (positive, negative, or both) is detected. After a capture event occurs, no further updating of the capture register will occur until the SCTRL[IEF] (input edge flag) is cleared by writing a zero to the SCTRL[IEF].

72.3.5 Modes of operation

TMR design operates only in one mode of operation: Functional mode. See Functional mode for the various counting modes in Functional mode.

72.3.5.1 Functional mode

The selected external count signals are sampled at the TMR's base clock rate and then run through a transition detector. The maximum count rate is one-half of the TMR's base clock rate. Internal clock sources can be used to clock the counters at the TMR's base clock rate.

If a counter is programmed to count to a specific value and then stop, the CTRL[CM] field is cleared when the count terminates.

72.3.5.1.1 Stop mode

If CTRL[CM] is set to '000', the counter is inert. No counting will occur. Stop mode will also disable the interrupts caused by input transitions on a selected input pin.

72.3.5.1.2 Count mode

If CTRL[CM] is set to '001', the counter will count the rising edges of the selected clock source. This mode is useful for generating periodic interrupts for timing purposes, or counting external events such as "widgets" on a conveyor belt passing a sensor. If the selected input is inverted by setting SCTRL[IPS] (input polarity select), then the negative edge of the selected external input signal is counted.

Count pulses from external source

```
//    This example uses TMR1 to count pulse (actually counts rising edges of
the pulse)
//      from an external source (QT3).
//
void Pulse_Init(void)
{

  /* TMR1_CTRL: CM=0,PCS=3,SCS=0,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0x0600);           /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x00);
  setReg(TMR1_CNTR,0x00);             /* Reset counter register */
  setReg(TMR1_LOAD,0x00);             /* Reset load register */
  setRegBitGroup(TMR1_CTRL,CM,0x01);  /* Run counter */
}
```

Generate periodic interrupt by counting internal clocks

```
//   This example generates an interrupt every 100ms,
//     assuming the chip is operating at 60 MHz.
//
// It does this by using the IP_bus_clk divided by 128 as the counter clock
source.
// The counter then counts to 46874 where it matches the COMP1 value.
// At that time an interrupt is generated, the counter is reloaded and
// the next COMP1 value is loaded from CMPLD1.
//
void TimerInt_Init(void)
{
  /* TMR0_CTRL: CM=0,PCS=0,SCS=0,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR0_CTRL,0x20);             /* Stop all functions of the timer */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR0_SCTRL,0x00);
  setReg(TMR0_LOAD,0x00);            /* Reset load register */
  setReg(TMR0_COMP1,46874);          /* Set up compare 1 register */
  setReg(TMR0_CMPLD1,46874);         /* Also set the compare preload register */
  /* TMR0_CSCTRL:
DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,TCF2EN=0,TCF1EN=1,
                   TCF2=0,TCF1=0,CL2=0,CL1=1 */
  setReg(TMR0_CSCTRL,0x41);          /* Enable compare 1 interrupt and */
                                      /*   compare 1 preload */
  setRegBitGroup(TMR0_CTRL,PCS,0xF); /* Primary Count Source to IP_bus_clk /
128 */
  setReg(TMR0_CNTR,0x00);            /* Reset counter register */
  setRegBitGroup(TMR0_CTRL,CM,0x01); /* Run counter */
}
```

72.3.5.1.3 Edge-Count mode

If CTRL[CM] is set to '010', the counter will count both edges of the selected external clock source. This mode is useful for counting the changes in the external environment, such as a simple encoder wheel.

Count both edges of external source signal

```
//  This example uses TMR1 to count pulse (actually counts both edges of the
pulse)
//    from an external source (QT3).
//
void Pulse_Init(void)
{
  /* TMR1_CTRL: CM=0,PCS=3,SCS=0,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0x0600);           /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x00);
  setReg(TMR1_CNTR,0x00);             /* Reset counter register */
  setReg(TMR1_LOAD,0x00);             /* Reset load register */
  setRegBitGroup(TMR1_CTRL,CM,0x02);  /* Run counter */
}
```

72.3.5.1.4 Gated-Count mode

If CTRL[CM] is set to '011', the counter will count while the selected secondary input signal is high. This mode is used to time the duration of external events. If the selected input is inverted by setting SCTRL[IPS] (input polarity select), then the counter will count while the selected secondary input is low.

Capture duration of external pulse

```
//   This example uses TMR1 to determine the duration of an external pulse.
//
//   The IP_bus clock is used as the primary counter. If the duration of the
//     external pulse is longer than 0.001 seconds one of the other IP_bus clock
//     dividers can be used. If the pulse duration is longer than 0.128 seconds
//     an external clock source will have to be used as the primary clock
source.
//
void Pulse1_Init(void)
{
  /* TMR1_CTRL: CM=0,PCS=8,SCS=1,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0x1080);            /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x00);
  setReg(TMR1_CNTR,0x00);             /* Reset counter register */
  setReg(TMR1_LOAD,0x00);             /* Reset load register */
  setRegBitGroup(TMR1_CTRL,CM,0x03);  /* Run counter */
}
```

72.3.5.1.5 Quadrature-Count mode

If CTRL[CM] is set to '100', the counter will decode the primary and secondary external inputs as quadrature encoded signals. Quadrature signals are usually generated by rotary or linear sensors used to monitor movement of motor shafts or mechanical equipment. The quadrature signals are square waves that are 90 degrees out of phase. The decoding of quadrature signal provides both count and direction information.

This figure shows a timing diagram illustrating the basic operation of a quadrature incremental position encoder.

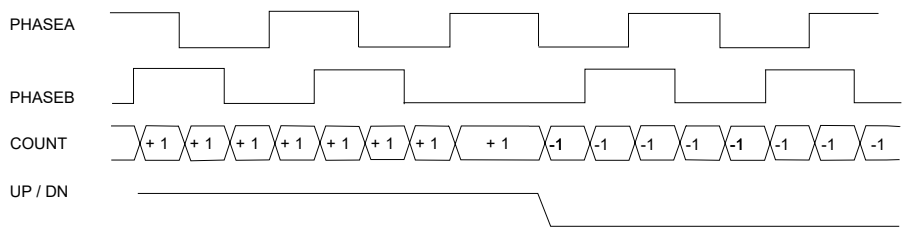


Figure 574. Quadrature incremental position encoder

Quadrature-Count mode example

```
//  This example uses TMR0 for counting states of a quadrature position encoder.
//
//  Timer input 0 is used as the primary count source (PHASEA).
//  Timer input 1 is used as the secondary count source (PHASEB).
//
void Pulse_Init(void)
{
  /* TMR0_CTRL: CM=0,PCS=0,SCS=1,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMRC0_CTRL,0x80);              /* Set up mode */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=1 */
  setReg(TMR0_SCTRL,0x00);
  setReg(TMR0_CNTR,0x00);             /* Reset counter register */
  setReg(TMR0_LOAD,0x00);             /* Reset load register */
  setReg(TMR0_COMP1,0xFFFF);           /* Set up compare 1 register */
  setReg(TMR0_COMP2,0x00);             /* Set up compare 2 register */
  /* TMR0_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR0_CSCTRL,0x00);
  setRegBitGroup(TMR0_CTRL,CM,0x04);  /* Run counter */
}
```

72.3.5.1.6 Quadrature-Count mode with index input

As an extension to the quadrature count mode discussed in the previous paragraph, some rotary shafts have a HOME or INDEX indicator. This would be a third input to the timer that is used to reset the timer's counter.

In this example, channel 0 is used to decode the quadrature inputs, but it doesn't actually count. Because its upper and lower limits are both set to 0, its output is cascaded count up and count down signals each time the quadrature inputs indicate a change in count. Channel 1 works in cascaded count mode receiving its counting instructions from channel 0. When an input capture event occurs, channel 1 is programmed to reset its counter value. The channel 1 counter contains the position value for the shaft.

Quadrature-Count mode with index input example

```
//  This example uses TMR0and TMR1 for counting states of a quadrature position
encoder.
//
// Timer input 0 is used as the primary count source (PHASEA).
// Timer input 1 is used as the secondary count source (PHASEB).
// Timer input 2 is used as the index input source (INDEX).
//
void Pulse_Init(void)
{
  /* TMR0_CTRL: CM=0,PCS=0,SCS=1,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR0_CTRL,0xA0);              /* Set up mode */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR0_SCTRL,0x00);
  setReg(TMR0_CNTR,0x00);             /* Reset counter register */
```

```
  setReg(TMR0_LOAD,0x00);            /* Reset load register */
  setReg(TMR0_COMP1,0x00);          /* Set up compare 1 register */
  setReg(TMR0_COMP2,0x00);           /* Set up compare 2 register */
  /* TMR0_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR0_CSCTRL,0x00);
/* TMR1_CTRL: CM=7,PCS=100,SCS=2,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0xE900);           /* Set up capture edge */
/* TMR1_SCTRL: Capture_Mode=10 */
  setReg(TMR1_SCTRL,0x0080);
/* TMR1_CSCTRL: ROC=1 */
  setReg(TMR1_CSCTRL,0x0800);          /* Set up reload on capture */
setRegBitGroup(TMR0_CTRL,CM,0x04);  /* Run counter */
}
```

72.3.5.1.7 Signed-Count mode

If CTRL[CM] is set to '101', the counter counts the primary clock source while the selected secondary source provides the selected count direction (up/down). If the value of the selected secondary source equals to that of the CTRL[DIR] bit, the count direction is up. Otherwise the count direction is down.

Signed-Count mode example

```
//  This example uses TMR0 for signed mode counting.
//
//  Timer input 2 is used as the primary count source.
//  Timer input 1 is used to determine the count direction.
//
void Pulse_Init(void)
{
  /* TMR0_CTRL: CM=0,PCS=2,SCS=1,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR0_CTRL,0x0480);             /* Set up mode */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=1,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR0_SCTRL,0x1000);
  setReg(TMR0_CNTR,0x00);             /* Reset counter register */
  setReg(TMR0_LOAD,0x00);             /* Reset load register */
  setRegBitGroup(TMR0_CTRL,CM,0x05);  /* Run counter */
}
```

72.3.5.1.8 Triggered-Count mode 1

If CSCTRL[TCI] is clear and CTRL[CM] is set to '110', the counter will begin counting the primary clock source after a positive transition (negative edge if SCTRL[IPS]=1) of the secondary input occurs. The counting will continue until a compare event occurs or another positive input transition is detected. If a second input transition occurs before a terminal count is reached, counting will stop and SCTRL[TCF] (timer compare flag) will be set. Subsequent secondary input transitions will continue to restart and stop the counting until a compare event occurs.

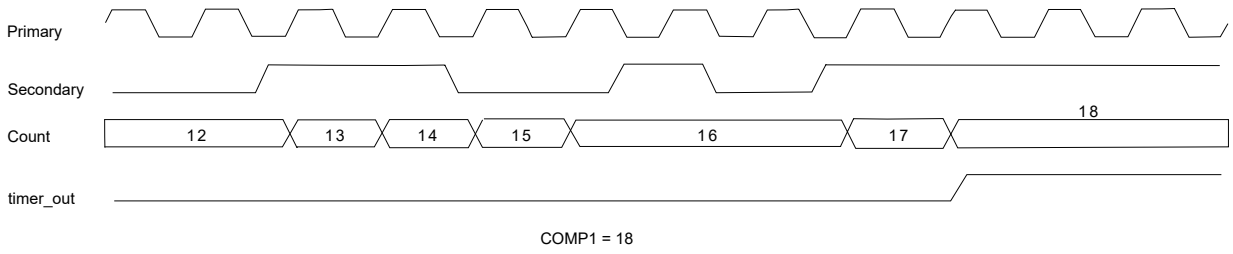


Figure 575. Triggered-Count mode 1 (CTRL[LENGTH]=0)

Triggered-Count mode 1 example

```
//   This example uses TMR1 for triggered mode counting.
//
//   Timer input 3 is used as the primary count source.
//   Timer input 2 is used for the trigger input.
//
void Pulse_Init(void)
{
  /* TMR1_CTRL: CM=0,PCS=3,SCS=2,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0x0700);            /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=1,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x1000);
  setReg(TMR1_CNTR,0x00);              /* Reset counter register */
  setReg(TMR1_LOAD,0x00);              /* Reset load register */
  setReg(TMR1_COMP1,0x0012);            /* Set up compare 1 register */
  /* TMR1_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR1_CSCTRL,0x00);
  setRegBitGroup(TMR1_CTRL,CM,0x06);    /* Run counter */
}
```

72.3.5.1.9 Triggered-Count mode 2

If CSCTRL[TCI] is set and CTRL[CM] is set to '110', the counter will begin counting the primary clock source after a positive transition (negative edge if SCTRL[IPS]=1) of the secondary input occurs. The counting will continue until a compare event occurs or another positive input transition is detected. If a second input transition occurs before a terminal count was reached, the counter will reload and continue counting. When CSCTRL[TCI] is set, the OFLAG output mode, CTRL[OUTMODE], should probably be set to '101' (cleared on init, set on compare) to ensure the output will be in a known state after the second input transition and subsequent reload takes place.

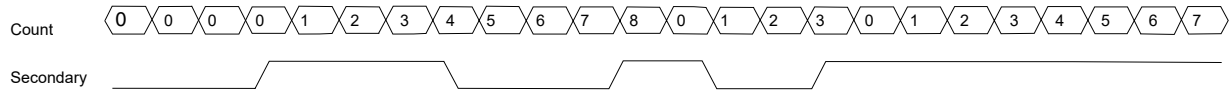


Figure 576. Triggered-Count mode 2 (CTRL[LENGTH]=0)

Triggered-Count mode 2 example

```
//   This example uses TMR1 for triggered mode counting.
//
//   Timer input 3 is used as the primary count source.
//   Timer input 2 is used for the trigger input.
//
void Pulse_Init(void)
{
  /* TMR1_CTRL: CM=0,PCS=3,SCS=2,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR1_CTRL,0x0700);            /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=1,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x1000);

  setReg(TMR1_CNTR,0x00);              /* Reset counter register */
  setReg(TMR1_LOAD,0x00);              /* Reset load register */
  setReg(TMR1_COMP1,0x0012);            /* Set up compare 1 register */
  /* TMR1_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=1,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR1_CSCTRL,0x400);
```

```
  setRegBitGroup(TMR1_CTRL,CM,0x06); /* Run counter */
}
```

72.3.5.1.10 One-Shot mode

If CTRL[CM] is set to '110', and the counter is set to reinitialize at a compare event (CTRL[LENGTH]=1),and CTRL[OUTMODE] is set to '101' (cleared on init, set on compare), the counter works in a one-shot mode. An external event causes the counter to count, and when the terminal count is reached, the output is asserted. This delayed output can be used to provide timing delays.

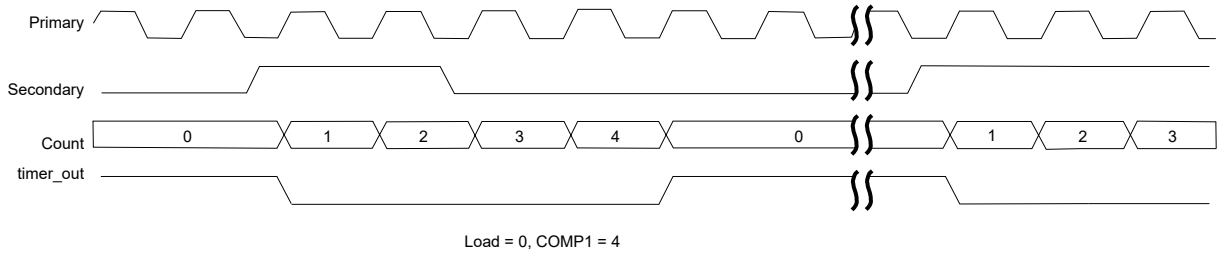


Figure 577. One-Shot mode (CTRL[LENGTH]=1)

One-Shot mode example

```
//  This example uses TMR1 for one-shot mode counting.
//
//  Timer input 3 is used as the primary count source.
//  Timer input 2 is used for the trigger input.
//
void Pulse_Init(void)
{
  /* TMR1_CTRL: CM=0,PCS=3,SCS=2,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=5 */
  setReg(TMR1_CTRL,0x0725);           /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=1,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR1_SCTRL,0x1000);
  setReg(TMR1_CNTR,0x00);             /* Reset counter register */
  setReg(TMR1_LOAD,0x00);             /* Reset load register */
  setReg(TMR1_COMP1,0x0004);           /* Set up compare 1 register */
  /* TMR1_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR1_CSCTRL,0x00);
  setRegBitGroup(TMR1_CTRL,CM,0x06);  /* Run counter */
}
```

72.3.5.1.11 Cascade-Count mode

If CTRL[CM] is set to '111', the counter's input is connected to the output of another selected counter. The counter will count up and down as compare events occur in the selected source counter. This cascade or daisy-chained mode enables multiple counters to be cascaded to yield longer counter lengths. When operating in cascade mode, a special high-speed signal path is used between modules rather than the OFLAG output signal. If the selected source counter is counting up and it experiences a compare event, the counter will be incremented. If the selected source counter is counting down and it experiences a compare event, the counter will be decremented.

Up to four counters may be cascaded to create a 64-bit wide synchronous counter. Check the data sheet to see if there are any frequency limits for cascaded counting mode.

Whenever any counter is read within a counter module, all of the counters' values within the module are captured in their respective hold registers. This action supports the reading of a cascaded counter chain. First read any counter of a cascaded counter chain, then read the hold registers of the other counters in the chain. The cascaded counter mode is synchronous.

NOTE

It is possible to connect counters together by using the other (non-cascade) counter modes and selecting the outputs of other counters as a clock source. In this case, the counters are operating in a ripple mode, where higher order counters will transition a clock later than a purely synchronous design.

Generate periodic interrupt cascading two counters

```
// This example generates an interrupt every 30 seconds,
//    assuming the chip is operating at 60 MHz.
//
// To do this, counter 2 is used to count 60,000 IP_bus clocks, which means it
//    will compare and reload every 0.001 seconds.
// Counter 3 is cascaded and used to count the 0.001 second ticks and
//    generate the desired interrupt interval.
//
void TimerInt_Init(void)
{
  // Set counter 2 to count IP_bus clocks
  /* TMR2_CTRL: CM=0,PCS=8,SCS=0,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR2_CTRL,0x1020);           /* Stop all functions of the timer */
  // Set counter 3 as cascaded and to count counter 2 outputs
  /* TMR3_CTRL: CM=7,PCS=6,SCS=0,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=0 */
  setReg(TMR3_CTRL,0xEC20);            /* Set up cascade counter mode */
  /* TMR3_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR3_SCTRL,0x00);
  /* TMR2_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
  setReg(TMR2_SCTRL,0x00);
  setReg(TMR3_CNTR,0x00);             /* Reset counter register */
  setReg(TMR2_CNTR,0x00);
  setReg(TMR3_LOAD,0x00);             /* Reset load register */
  setReg(TMR2_LOAD,0x00);
  setReg(TMR3_COMP1, 30000-1);        /* milliseconds in 30 seconds */
  setReg(TMR3_CMPLD1,30000-1);
  setReg(TMR2_COMP1, 60000-1);        /* Set to cycle every milisecond */
  setReg(TMR2_CMPLD1,60000-1);
  /* TMR3_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=1,TCF2=0,TCF1=0,CL2=0,CL1=1 */
  setReg(TMR3_CSCTRL,0x41);           /* Enable compare 1 interrupt and */
                                      /* compare 1 preload              */
  /* TMR2_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                  TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=1 */
  setReg(TMR2_CSCTRL,0x01);           /* Enable Compare 1 preload */
  setRegBitGroup(TMR2_CTRL,CM,0x01);  /* Run counter */
}
```

72.3.5.1.12 Pulse-Output mode

If CTRL[CM]=001, and CTRL[OUTMODE] is set to 111' (gated clock output), and CTRL[ONCE] is set, then the counter will output a pulse stream of pulses that has the same frequency of the selected clock source, and the number of output pulses is equal to the compare value minus the init value. This mode is useful for driving step motor systems.

NOTE

This does not work if CTRL[PCS] is set to 1000 (IP_bus/1).

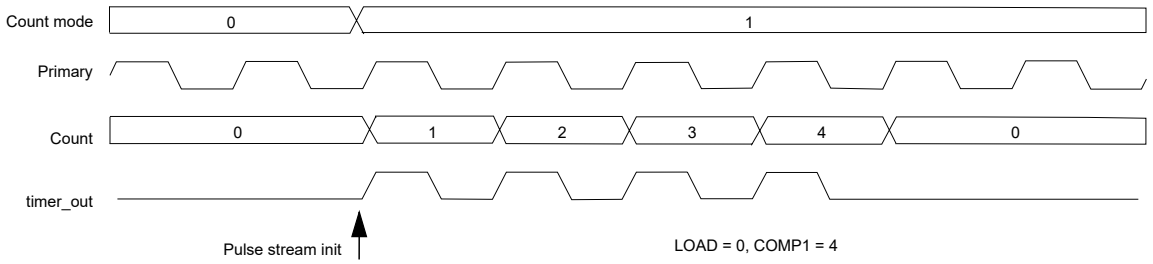


Figure 578. Pulse-Output mode

Pulse outputs using two counters

```
// This example generates six 10ms pulses, from QT1 output.
// Assuming the chip is operating at 60 MHz.
//
// To do this, timer 3 is used to generate a clock with a period of 10ms.
//
// Timer 1 is used to gate these clocks and count the number of pulses that have
//  been generated.
//
void PulseStream_Init(void)
{
// Select IP_bus_clk/16 as the clock source for Timer 3
  /* TMR3_CTRL: CM=0,PCS=0x0C,SCS=0,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=3 */
   setReg(TMR3_CTRL,0x1823);             /* Set up mode */
  /* TMR3_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=0 */
   setReg(TMR3_SCTRL,0x00);
   setReg(TMR3_LOAD,0x00);              /* Reset load register */
   setReg(TMR3_COMP1,37500-1);          /* (16 * 37500 ) / 60e6 = 0.01 sec */
 /* TMR3_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                   TCF2EN=0,TCF1EN=0,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR3_CSCTRL,0x00);             /* Set up comparator control register */



// Timer 3 output is the clock source for this timer.
  /* TMR1_CTRL: CM=0,PCS=7,SCS=0,ONCE=1,LENGTH=1,DIR=0,COINIT=0,OUTMODE=7 */
  setReg(TMR1_CTRL,0x0E67);             /* Set up mode */
  /* TMR1_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=0,OPS=0,OEN=1 */
   setReg(TMR1_SCTRL,0x01);
   setReg(TMR1_CNTR,0x00);              /* Reset counter register */
   setReg(TMR1_LOAD,0x00);              /* Reset load register */
   setReg(TMR1_COMP1,0x04);             /* Set up compare 1 register */


// set to interrupt after the last pulse
  /* TMR1_CSCTRL: DBG_EN=0,FAULT=0,ALT_LOAD=0,ROC=0,TCI=0,UP=0,OFLAG=0,
                   TCF2EN=0,TCF1EN=1,TCF2=0,TCF1=0,CL2=0,CL1=0 */
  setReg(TMR1_CSCTRL,0x40);             /* Set up comparator control register */
// Finally, start the counters running
  setReg(TMR3_CNTR,0);                  /* Reset counter */
  setRegBitGroup(TMR3_CTRL,CM,0x01);    /* Run source clock counter */
  setRegBitGroup(TMR1_CTRL,CM,0x01);    /* Run counter */
}
```

72.3.5.1.13 Fixed-Frequency PWM mode

If CTRL[CM]=001, count through roll-over (CTRL[LENGTH]=0), continuous count (CTRL[ONCE]=0) and CTRL[OUTMODE] is '110' (set on compare, cleared on counter roll-over), then the counter output yields a pulse-width modulated (PWM) signal with a frequency equal to the count clock frequency divided by 65,536 and a pulse-width duty cycle equal to the compare value divided by 65,536. This mode of operation is often used to drive PWM amplifiers used to power motors and inverters.

Fixed-Frequency PWM mode example

```
// This example uses TMR0 for Fixed-Frequency PWM mode timing.
//
// The timer will count IP_bus clocks continuously until it rolls over.
// This results in a PWM period of 65536 / 60e6 = 1092.267 usec
//
// Initially, an output pulse width of 25 usec is generated ( 1500 / 60e6 )
//   giving a PWM ratio of 1500 / 65536 = 2.289%
// This pulse width can be changed by changing the COMP1 register value (using
CMPLD1).
//
void PWM1_Init(void)
{
  setReg(TMR0_CNTR,0);           /* Reset counter */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=1,OPS=0,OEN=1 */
  setReg(TMR0_SCTRL,0x05);       /* Enable output */
  setReg(TMR0_COMP1,1500-1);     /* Store initial value to the duty-compare
register */
  /* TMR0_CTRL: CM=1,PCS=8,SCS=0,ONCE=0,LENGTH=0,DIR=0,COINIT=0,OUTMODE=6 */
  setReg(TMR0_CTRL,0x3006);      /* Run counter */
}
```

72.3.5.1.14 Variable-Frequency PWM mode

If CTRL[CM]=001, count until compare (CTRL[LENGTH]=1), continuous count (CTRL[ONCE] = 0) and CTRL[OUTMODE] is '100' (toggle OFLAG and alternate compare registers), then the counter output yields a pulse-width modulated (PWM) signal whose frequency and pulse width is determined by the values programmed into the COMP1 and COMP2 registers, and the input clock frequency. This method of PWM generation has the advantage of allowing almost any desired PWM frequency and/or constant on or off periods. This mode of operation is often used to drive PWM amplifiers used to power motors and inverters. The CMPLD1 and CMPLD2 registers are especially useful for this mode, as they allow the programmer time to calculate values for the next PWM cycle while the PWM current cycle is underway.

To set up the TMR to run in variable frequency PWM mode with compare preload please use the following setup for the specific counter you would like to use. When performing the setup, update the TMR_CTRL register last because the counter will start counting if the count mode is changed to any value other than 000 (assuming the primary count source is already active).

Timer Control Register (CTRL)

- CM=001 (count rising edges of primary source)
- PCS=1000 (IP bus clock for best granularity for waveform timing)
- SCS=Any (ignored in this mode)
- ONCE=0 (want to count repeatedly)
- LENGTH=1 (want to count until compare value is reached and re-initialize counter register)
- DIR=Any (user's choice. The compare register values must be chosen carefully to account for things like roll-under, etc.)
- COINIT=0 (user can set this if they need this function)
- OUTMODE=100 (toggle OFLAG output using alternating compare registers)

Timer Status and Control Register (SCTRL)

- OEN = 1 (output enable to allow OFLAG output to be put on an external pin. Set this bit as needed.)
- OPS = Any (user's choice)
- Make sure the rest of the bits are cleared for this register. We will enable interrupts in the comparator status and control register instead of in this register.

Comparator Status and Control Register (CSCTRL)

- TCF2EN=1 (allow interrupt to be issued when CSCTRL[TCF2] is set)
- TCF1EN=0 (do not allow interrupt to be issued when CSCTRL[TCF1] is set)
- TCF1=0 (clear timer compare 1 interrupt source flag. This is set when counter register equals compare register 1 value and OFLAG is low)
- TCF2=0 (clear timer compare 2 interrupt source flag. This is set when counter register equals compare register 2 value and OFLAG is high)
- CL1=10 (load compare register when CSCTRL[TCF2] is asserted)
- CL2=01 (load compare register when CSCTRL[TCF1] is asserted)

Interrupt Service Routines

To service the CSCTRL[TCF2] interrupts generated by the timer, the interrupt controller must be configured to enable the interrupts for the particular timer being used. Additionally the user will need to write an interrupt service routine to do at a minimum the following:

- Clear CSCTRL[TCF2] and CSCTRL[TCF1] flags.
- Calculate and write new values for both CMPLD1 and CMPLD2.

Timing

This figure contains the timing for using the compare preload feature. The compare preload cycle begins with a compare event on COMP2 causing CSCTRL[TCF2] to be asserted. COMP1 is loaded with the value in the CMPLD1 (c3) one IP bus clock later. In addition an interrupt is asserted by the timer and the interrupt service routine is executed during which both comparator load registers are updated with new values (c4 and c5). When CSCTRL[TCF1] is asserted, COMP2 is loaded with the value in CMPLD2 (c4). And on the subsequent CSCTRL[TCF2] event, COMP1 is loaded with the value in CMPLD1 (c5). The cycle starts over again as an interrupt is asserted and the interrupt service routine clears CSCTRL[TCF1] and CSCTRL[TCF2] and calculates new values for CMPLD1 and CMPLD2.

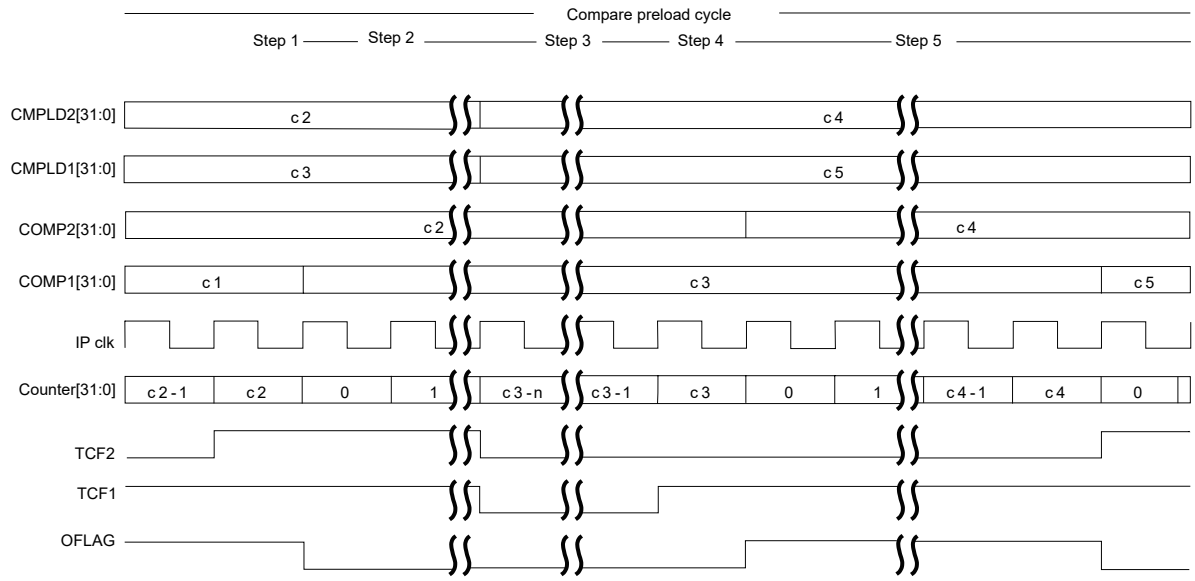


Figure 579. Compare load timing

Variable-Frequency PWM mode

```
// This example starts with an 11 msec with a 31 msec cycle.
// Assuming the chip is operating at 60 MHz, the timer use IP_bus_clk/32 as its
//   clock source.
//
// Initial pulse period: 60e6/32 clocks/sec * 31 ms = 58125 total clocks in
period
// Initial pulse width: 60e6/32 clocks/sec * 11 ms = 20625 clocks in pulse
//
//
// Once the initial values of COMP1/CMPLD1 and COMP2/CMPLD2 are set the pulse
width
//   can be varied by load new values of CMPLD1 and CMPLD2 on each compare
interrupt.
//    (See Usage of compare load registers.)
//
void PPG1_Init(void)
{
  setReg(TMR0_LOAD,0);                 /* Clear load register */
  setReg(TMR0_CNTR,0);                 /* Clear counter */
  /* TMR0_SCTRL: TCF=0,TCFIE=0,TOF=0,TOFIE=0,IEF=0,IEFIE=0,IPS=0,INPUT=0,
                 Capture_Mode=0,MSTR=0,EEOF=0,VAL=0,FORCE=1,OPS=0,OEN=1 */
  setReg(TMR0_SCTRL,5);                /* Set Status and Control Register */
  // Set compare preload operation and enable an interrupt on compare2 events.
  /* TMR0_CSCTRL: TCF2EN=1,TCF1EN=0,TCF2=0,TCF1=0,CL21=0,CL20=1,CL11=1,CL10=0 */
  setReg(TMR0_CSCTRL,0x86);            /* Set Comparator Status and Control
Register */

  setReg(TMR0_COMP1, 20625-1);         /* Set the pulse width of the off time */
  setReg(TMR0_CMPLD1,20625-1);         /* Set the pulse width of the off time */
  setReg(TMR0_COMP2, 58125-20625-1);   /* Set the pulse width of the on time */
  setReg(TMR0_CMPLD2,58125-20625-1);   /* Set the pulse width of the on time */
  /* TMR0_CTRL: CM=1,PCS=0xD,SCS=0,ONCE=0,LENGTH=1,DIR=0,COINIT=0,OUTMODE=4 */
  setRegBits(TMR0_CTRL,0x3A24);        /* Set variable PWM mode and run counter
*/
}
```

72.3.6 Clocking

72.3.6.1 General

The timer only receives BUS_CLK_ROOT .

72.3.7 Resets

72.3.7.1 General

TMR can be reset by the RST_B signal. This forces all registers to their reset state and clears the OFLAG signal if it is asserted. The counter will be turned off until the settings in the control register are changed.

Table 1081. Reset Summary

Reset	Priority	Source	Characteristics
RST_B	n/a	hardware reset and system software reset	full system reset

72.3.8 Interrupts

72.3.8.1 General

TMR can generate 20 interrupts, Five for each of the four counters/channels.

Table 1082. Interrupt Summary

Core Interrupt	Interrupt	Description
TMR Channel 0	TMR0_COMP_IRQ_B	Compare Interrupt Request for Timer Channel 0
	TMR0_COMP1_IRQ_B	Compare 1 Interrupt Request for Timer Channel 0
	TMR0_COMP2_IRQ_B	Compare 2 Interrupt Request for Timer Channel 0
	TMR0_OVF_IRQ_B	Overflow Interrupt Request for Timer Channel 0
	TMR0_EDGE_IRQ_B	Input Edge Interrupt Request for Timer Channel 0
TMR Channel 1	TMR1_COMP_IRQ_B	Compare Interrupt Request for Timer Channel 1
	TMR1_COMP1_IRQ_B	Compare 1 Interrupt Request for Timer Channel 1
	TMR1_COMP2_IRQ_B	Compare 2 Interrupt Request for Timer Channel 1
	TMR1_OVF_IRQ_B	Overflow Interrupt Request for Timer Channel 1
	TMR1_EDGE_IRQ_B	Input Edge Interrupt Request for Timer Channel 1
TMR Channel 2	TMR2_COMP_IRQ_B	Compare Interrupt Request for Timer Channel 2
	TMR2_COMP1_IRQ_B	Compare 1 Interrupt Request for Timer Channel 2

Table continues on the next page...

Table 1082. Interrupt Summary (continued)

Core Interrupt	Interrupt	Description
	TMR2_COMP2_IRQ_B	Compare 2 Interrupt Request for Timer Channel 2
	TMR2_OVF_IRQ_B	Overflow Interrupt Request for Timer Channel 2
	TMR2_EDGE_IRQ_B	Input Edge Interrupt Request for Timer Channel 2
TMR Channel 3	TMR3_COMP_IRQ_B	Compare Interrupt Request for Timer Channel 3
	TMR3_COMP1_IRQ_B	Compare 1 Interrupt Request for Timer Channel 3
	TMR3_COMP2_IRQ_B	Compare 2 Interrupt Request for Timer Channel 3
	TMR3_OVF_IRQ_B	Overflow Interrupt Request for Timer Channel 3
	TMR3_EDGE_IRQ_B	Input Edge Interrupt Request for Timer Channel 3

72.3.8.2 Description of Interrupt Operation

72.3.8.2.1 Timer Compare Interrupts

These interrupts are generated when a successful compare occurs between a counter and it's compare registers while SCTRL[TCFIE] is set. These interrupts are cleared by writing a zero to the appropriate SCTRL[TCF].

When a timer compare interrupt is set in TMR_SCTRL and the Compare Load registers are available, one of the following two interrupts will also be asserted.

72.3.8.2.1.1 Timer Compare 1 Interrupts (Available with Compare Load Feature)

These interrupts are generated when a successful compare occurs between a counter and its COMP1 register while CSCTRL[TCF1EN] is set. These interrupts are cleared by writing a zero to the appropriate CSCTRL[TCF1].

72.3.8.2.1.2 Timer Compare 2 Interrupts (Available with Compare Load Feature)

These interrupts are generated when a successful compare occurs between a counter and its COMP2 register while CSCTRL[TCF2EN] is set. These interrupts are cleared by writing a zero to the appropriate CSCTRL[TCF2].

72.3.8.2.2 Timer Overflow Interrupts

These interrupts are generated when a counter rolls over its maximum value while SCTRL[TOFIE] is set. These interrupts are cleared by writing zero to the appropriate SCTRL[TOF].

72.3.8.2.3 Timer Input Edge Interrupts

These interrupts are generated by a transition of the input signal (either positive or negative depending on IPS setting) while SCTRL[IEFIE] is set. These interrupts are cleared by writing a zero to the appropriate SCTRL[IEF].

72.3.9 DMA

TMR can generate twelve DMA requests: three for each of the four counters/channels. Refer to the following table.

Table 1083. DMA Summary

DMA Request	DMA Enable	Description
Channels 0-3	DMA[IEFDE]	CAPT contains a value
	DMA[CMPLD1DE]	CMPLD1 needs an update
	DMA[CMPLD2DE]	CMPLD2 needs an update

72.4 External signals

Table 1084. TMR external signals description

Signal	Description	I/O
tmr0_output	Channel 0 output data	O
tmr1_output	Channel 1 output data	
tmr2_output	Channel 2 output data	
tmr3_output	Channel 3 output data	
tmr0_output_en_b	Channel 0 output data enable	O
tmr1_output_en_b	Channel 1 output data enable	
tmr2_output_en_b	Channel 2 output data enable	
tmr3_output_en_b	Channel 3 output data enable	
tmr0_input	Channel 0 input data[1]	I
tmr1_input	Channel 1 input data[1]	
tmr2_input	Channel 2 input data[1]	
tmr3_input	Channel 3 input data[1]	

1. This input signal may be tied low. See chip-specific chapter for more information.

72.5 Memory map/register definition

72.5.1 TMR register descriptions

The address of a register is the sum of a base address and an address offset. The base address is defined at the chip level and the address offset is defined at the module level. Make certain to check which quad timer is available on the chip being used, and which timer channels have external I/O.

72.5.1.1 TMR memory map

TMR1 base address: 4269_0000h

TMR2 base address: 426A_0000h

TMR3 base address: 426B_0000h

TMR4 base address: 426C_0000h

TMR5 base address: 426D_0000h

TMR6 base address: 426E_0000h

TMR7 base address: 426F_0000h

TMR8 base address: 4270_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Timer Channel Compare Register 1 (COMP10)	16	RW	0000h
2h	Timer Channel Compare Register 2 (COMP20)	16	RW	0000h
4h	Timer Channel Capture Register (CAPT0)	16	RW	0000h
6h	Timer Channel Load Register (LOAD0)	16	RW	0000h
8h	Timer Channel Hold Register (HOLD0)	16	RW	0000h
Ah	Timer Channel Counter Register (CNTR0)	16	RW	0000h
Ch	Timer Channel Control Register (CTRL0)	16	RW	0000h
Eh	Timer Channel Status and Control Register (SCTRL0)	16	RW	0000h
10h	Timer Channel Comparator Load Register 1 (CMPLD10)	16	RW	0000h
12h	Timer Channel Comparator Load Register 2 (CMPLD20)	16	RW	0000h
14h	Timer Channel Comparator Status and Control Register (CSCTRL0)	16	RW	0000h
16h	Timer Channel Input Filter Register (FILT0)	16	RW	0000h
18h	Timer Channel DMA Enable Register (DMA0)	16	RW	0000h
1Eh	Timer Channel Enable Register (ENBL)	16	RW	000Fh
20h	Timer Channel Compare Register 1 (COMP11)	16	RW	0000h
22h	Timer Channel Compare Register 2 (COMP21)	16	RW	0000h
24h	Timer Channel Capture Register (CAPT1)	16	RW	0000h
26h	Timer Channel Load Register (LOAD1)	16	RW	0000h
28h	Timer Channel Hold Register (HOLD1)	16	RW	0000h
2Ah	Timer Channel Counter Register (CNTR1)	16	RW	0000h
2Ch	Timer Channel Control Register (CTRL1)	16	RW	0000h
2Eh	Timer Channel Status and Control Register (SCTRL1)	16	RW	0000h
30h	Timer Channel Comparator Load Register 1 (CMPLD11)	16	RW	0000h
32h	Timer Channel Comparator Load Register 2 (CMPLD21)	16	RW	0000h
34h	Timer Channel Comparator Status and Control Register (CSCTRL1)	16	RW	0000h
36h	Timer Channel Input Filter Register (FILT1)	16	RW	0000h
38h	Timer Channel DMA Enable Register (DMA1)	16	RW	0000h
40h	Timer Channel Compare Register 1 (COMP12)	16	RW	0000h
42h	Timer Channel Compare Register 2 (COMP22)	16	RW	0000h
44h	Timer Channel Capture Register (CAPT2)	16	RW	0000h
46h	Timer Channel Load Register (LOAD2)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
48h	Timer Channel Hold Register (HOLD2)	16	RW	0000h
4Ah	Timer Channel Counter Register (CNTR2)	16	RW	0000h
4Ch	Timer Channel Control Register (CTRL2)	16	RW	0000h
4Eh	Timer Channel Status and Control Register (SCTRL2)	16	RW	0000h
50h	Timer Channel Comparator Load Register 1 (CMPLD12)	16	RW	0000h
52h	Timer Channel Comparator Load Register 2 (CMPLD22)	16	RW	0000h
54h	Timer Channel Comparator Status and Control Register (CSCTRL2)	16	RW	0000h
56h	Timer Channel Input Filter Register (FILT2)	16	RW	0000h
58h	Timer Channel DMA Enable Register (DMA2)	16	RW	0000h
60h	Timer Channel Compare Register 1 (COMP13)	16	RW	0000h
62h	Timer Channel Compare Register 2 (COMP23)	16	RW	0000h
64h	Timer Channel Capture Register (CAPT3)	16	RW	0000h
66h	Timer Channel Load Register (LOAD3)	16	RW	0000h
68h	Timer Channel Hold Register (HOLD3)	16	RW	0000h
6Ah	Timer Channel Counter Register (CNTR3)	16	RW	0000h
6Ch	Timer Channel Control Register (CTRL3)	16	RW	0000h
6Eh	Timer Channel Status and Control Register (SCTRL3)	16	RW	0000h
70h	Timer Channel Comparator Load Register 1 (CMPLD13)	16	RW	0000h
72h	Timer Channel Comparator Load Register 2 (CMPLD23)	16	RW	0000h
74h	Timer Channel Comparator Status and Control Register (CSCTRL3)	16	RW	0000h
76h	Timer Channel Input Filter Register (FILT3)	16	RW	0000h
78h	Timer Channel DMA Enable Register (DMA3)	16	RW	0000h

72.5.1.2 Timer Channel Compare Register 1 (COMP10 - COMP13)

Offset

Register	Offset
COMP10	0h
COMP11	20h
COMP12	40h
COMP13	60h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMPARISON_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 COMPARISON_1	Comparison Value 1 Stores the value used for comparison with the counter value in count up mode.

72.5.1.3 Timer Channel Compare Register 2 (COMP20 - COMP23)

Offset

Register	Offset
COMP20	2h
COMP21	22h
COMP22	42h
COMP23	62h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMPARISON_2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

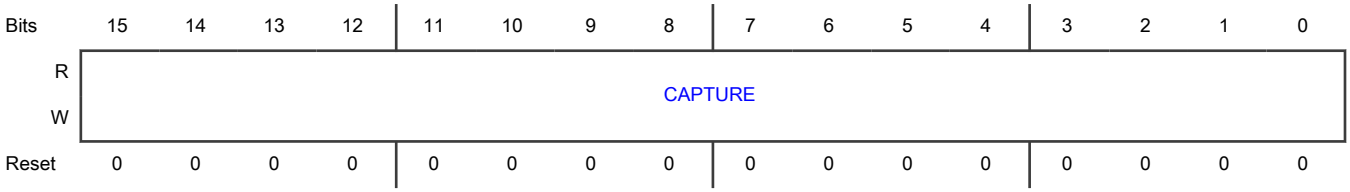
Field	Function
15-0 COMPARISON_2	Comparison Value 2 Stores the value used for comparison with the counter value in count down mode or alternating compare mode.

72.5.1.4 Timer Channel Capture Register (CAPT0 - CAPT3)

Offset

Register	Offset
CAPT0	4h
CAPT1	24h
CAPT2	44h
CAPT3	64h

Diagram



Fields

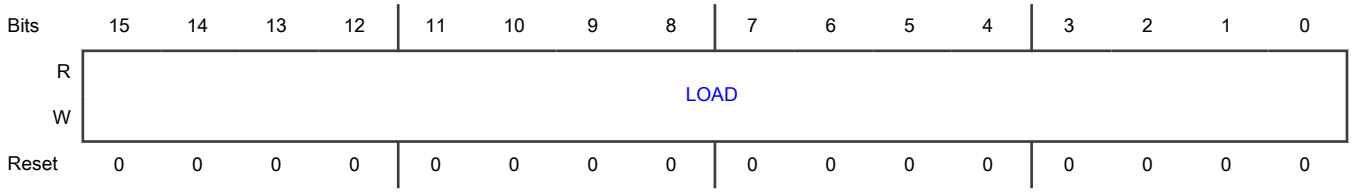
Field	Function
15-0 CAPTURE	Capture Value Stores the value captured from the counter.

72.5.1.5 Timer Channel Load Register (LOAD0 - LOAD3)

Offset

Register	Offset
LOAD0	6h
LOAD1	26h
LOAD2	46h
LOAD3	66h

Diagram



Fields

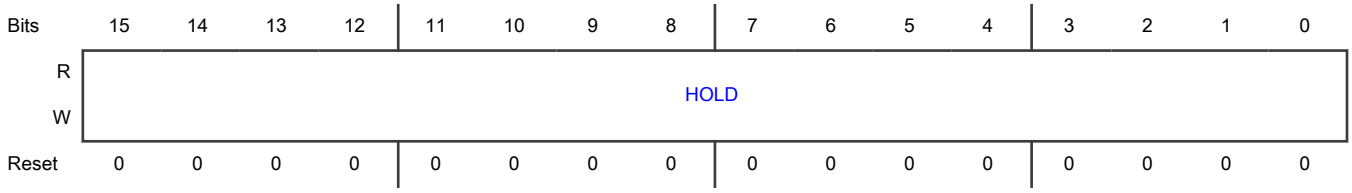
Field	Function
15-0 LOAD	Timer Load Register Stores the value used to initialize the counter after counter compare or re-initialization.

72.5.1.6 Timer Channel Hold Register (HOLD0 - HOLD3)

Offset

Register	Offset
HOLD0	8h
HOLD1	28h
HOLD2	48h
HOLD3	68h

Diagram



Fields

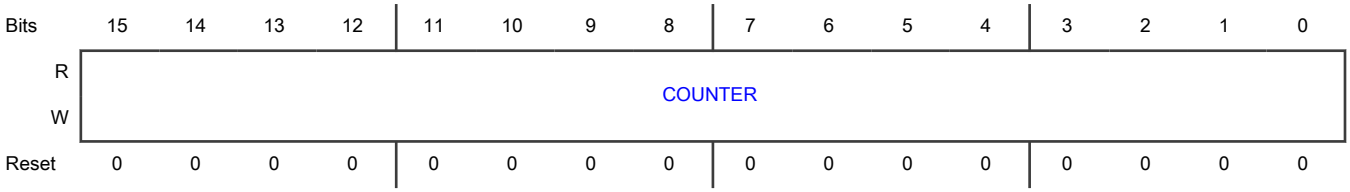
Field	Function
15-0 HOLD	HOLD Stores the counter's values of specific channels whenever any of the four counters within a module is read.

72.5.1.7 Timer Channel Counter Register (CNTR0 - CNTR3)

Offset

Register	Offset
CNTR0	Ah
CNTR1	2Ah
CNTR2	4Ah
CNTR3	6Ah

Diagram



Fields

Field	Function
15-0 COUNTER	COUNTER Is the counter for the corresponding channel in a timer module.

72.5.1.8 Timer Channel Control Register (CTRL0 - CTRL3)

Offset

Register	Offset
CTRL0	Ch
CTRL1	2Ch
CTRL2	4Ch
CTRL3	6Ch

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CM				PCS				SCS		ONCE	LENGTH	DIR	COINIT	OUTMODE	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 CM	Count Mode Controls the basic counting and behavior of the counter. 000b - No operation 001b - Count rising edges of primary source[1] 010b - Count rising and falling edges of primary source[2] 011b - Count rising edges of primary source while secondary input high active 100b - Quadrature count mode, uses primary and secondary sources 101b - Count rising edges of primary source; secondary source specifies direction[3] 110b - Edge of secondary source triggers primary count until compare 111b - Cascaded counter mode (up/down)[4]
12-9 PCS	Primary Count Source Selects the primary count source. **NOTE** A timer selecting its own output for input is not a legal choice. The result is no counting. 0000b - Counter 0 input pin 0001b - Counter 1 input pin 0010b - Counter 2 input pin 0011b - Counter 3 input pin 0100b - Counter 0 output 0101b - Counter 1 output 0110b - Counter 2 output 0111b - Counter 3 output 1000b - IP bus clock divide by 1 prescaler 1001b - IP bus clock divide by 2 prescaler 1010b - IP bus clock divide by 4 prescaler 1011b - IP bus clock divide by 8 prescaler 1100b - IP bus clock divide by 16 prescaler

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1101b - IP bus clock divide by 32 prescaler 1110b - IP bus clock divide by 64 prescaler 1111b - IP bus clock divide by 128 prescaler
8-7 SCS	Secondary Count Source Identifies the external input pin to be used as a count command or timer command. The selected input can trigger the timer to capture the current value of CNTR . The selected input can also be used to specify the count direction. The selected signal can also be used as a fault input when CSCTRL[FAULT] is set. The polarity of the signal can be inverted by SCTRL[IPS]. 00b - Counter 0 input pin 01b - Counter 1 input pin 10b - Counter 2 input pin 11b - Counter 3 input pin
6 ONCE	Count Once Selects between continuous count mode and one-shot count mode. 0b - Count repeatedly. 1b - Count until compare and then stop. If counting up, a successful compare occurs when the counter reaches a COMP1 value. If counting down, a successful compare occurs when the counter reaches a COMP2 value. When output mode $4 is used, the counter re-initializes after reaching the COMP1 value, continues to count to the COMP2 value, and then stops.
5 LENGTH	Count Length Determines whether the counter: • counts to the compare value and then re-initializes itself to the value specified in the LOAD (or CMPLD2) register, or • continues counting past the compare value to the binary roll over. 0b - Count until roll over at $FFFF and then continue by re-initializing the counter from the LOAD register. 1b - Count until compare, then re-initialize using the LOAD register. If counting up, a successful compare occurs when the counter reaches a COMP1 value. If counting down, a successful compare occurs when the counter reaches a COMP2 value. When output mode $4 is used, alternating values of COMP1 and COMP2 are used to generate successful comparisons. For example, the counter counts until a COMP1 value is reached, re-initializes, counts until COMP2 value is reached, re-initializes, counts until COMP1 value is reached, and so on.
4 DIR	Count Direction Selects between the normal count direction (up) and the reverse direction (down). 0b - Count up. 1b - Count down.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 COINIT	Co-Channel Initialization Enables another counter/timer within the module to force the re-initialization of this counter/timer when it has an active compare event. 0b - Co-channel counter/timers cannot force a re-initialization of this counter/timer 1b - Co-channel counter/timers may force a re-initialization of this counter/timer
2-0 OUTMODE	Output Mode Determines the mode of operation for the OFLAG output signal. 000b - Asserted while counter is active 001b - Clear OFLAG output on successful compare 010b - Set OFLAG output on successful compare 011b - Toggle OFLAG output on successful compare 100b - Toggle OFLAG output using alternating compare registers 101b - Set on compare, cleared on secondary source input edge 110b - Set on compare, cleared on counter rollover 111b - Enable gated clock output while counter is active

1. Rising edges are counted only when SCTRL[IPS] = 0. Falling edges are counted when SCTRL[IPS] = 1. If the primary count source is IP bus clock divide by 1, only rising edges are counted regardless of the value of SCTRL[IPS].
2. IP bus clock divide by 1 cannot be used as a primary count source in edge count mode.
3. Rising edges are counted only when SCTRL[IPS] = 0. Falling edges are counted when SCTRL[IPS] = 1.
4. The primary count source must be set to one of the counter outputs.

72.5.1.9 Timer Channel Status and Control Register (SCTRL0 - SCTRL3)

Offset

Register	Offset
SCTRL0	Eh
SCTRL1	2Eh
SCTRL2	4Eh
SCTRL3	6Eh

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TCF	TCFIE	TOF	TOFIE	IEF	IEFIE	IPS	INPUT	CAPTURE_MODE		MSTR	EEOF	VAL	0	OPS	OEN
W														FORCE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 TCF	Timer Compare Flag Is set when a successful compare occurs, and is cleared by writing a zero to this bit.
14 TCFIE	Timer Compare Flag Interrupt Enable When set, enables interrupts when TCF is set.
13 TOF	Timer Overflow Flag Is set when the counter rolls over its maximum value $FFFF or $0000 (depending on count direction), and is cleared by writing a zero to this bit.
12 TOFIE	Timer Overflow Flag Interrupt Enable When set, enables interrupts when TOF is set.
11 IEF	Input Edge Flag Is set when CAPTMODE is enabled and a proper input transition occurs (on an input selected as a secondary count source) while the count mode does not equal 000, and is cleared by writing a zero to this bit. This bit can also be cleared automatically by a read of CAPT when DMA[IEFDE] is set. **NOTE** Setting the input polarity select bit (IPS) changes the edge to be detected. Also, the control register's secondary count source determines which external input pin is monitored by the detection circuitry.
10 IEFIE	Input Edge Flag Interrupt Enable When set, enables interrupts when IEF is set. **RESTRICTION** Do not set both this bit and DMA[IEFDE].
9 IPS	Input Polarity Select When set, inverts the input signal polarity.
8 INPUT	External Input Signal Indicates the current state of the external input pin selected via the secondary count source after application of IPS and filtering.

Table continues on the next page...

Table continued from the previous page...

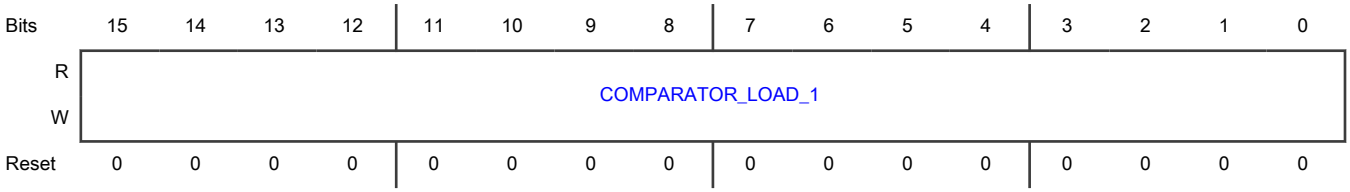
Field	Function
7-6 CAPTURE_MODE	Input Capture Mode Specifies the operation of the capture register as well as the operation of the input edge flag. The input source is the secondary count source. 00b - Capture function is disabled 01b - Load capture register on rising edge (when IPS=0) or falling edge (when IPS=1) of input 10b - Load capture register on falling edge (when IPS=0) or rising edge (when IPS=1) of input 11b - Load capture register on both edges of input
5 MSTR	Master Mode When set, enables the compare function's output to be broadcasted to the other counters/timers in the module. This signal then can be used to re-initialize the other counters and/or force their OFLAG signal outputs.
4 EEOF	Enable External OFLAG Force When set, enables the compare from another counter/timer within the same module to force the state of this counter's OFLAG output signal.
3 VAL	Forced OFLAG Value Determines the value of the OFLAG output signal when software triggers a FORCE command.
2 FORCE	Force OFLAG Output Forces the current value of VAL to be written to the OFLAG output. VAL and FORCE can be written simultaneously in a single write operation. Write to FORCE only if the counter is disabled. Setting this bit while the counter is enabled may yield unpredictable results.
1 OPS	Output Polarity Select Determines the polarity of the OFLAG output signal. 0b - True polarity. 1b - Inverted polarity.
0 OEN	Output Enable Determines the direction of the external pin. 0b - The external pin is configured as an input. 1b - The OFLAG output signal is driven on the external pin. Other timer groups using this external pin as their input see the driven value. The polarity of the signal is determined by OPS.

72.5.1.10 Timer Channel Comparator Load Register 1 (CMPLD10 - CMPLD13)

Offset

Register	Offset
CMPLD10	10h
CMPLD11	30h
CMPLD12	50h
CMPLD13	70h

Diagram



Fields

Field	Function
15-0 COMPARATOR_LOAD_1	COMPARATOR_LOAD_1 Is the comparator 1 preload value for the COMP1 register for the corresponding channel in a timer module.

72.5.1.11 Timer Channel Comparator Load Register 2 (CMPLD20 - CMPLD23)

Offset

Register	Offset
CMPLD20	12h
CMPLD21	32h
CMPLD22	52h
CMPLD23	72h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMPARATOR_LOAD_2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 COMPARATOR_LOAD_2	COMPARATOR_LOAD_2 Is the comparator 2 preload value for the COMP2 register for the corresponding channel in a timer module.

72.5.1.12 Timer Channel Comparator Status and Control Register (CSCTRL0 - CSCTRL3)

Offset

Register	Offset
CSCTRL0	14h
CSCTRL1	34h
CSCTRL2	54h
CSCTRL3	74h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DBG_EN		FAULT	ALT_LOAD	ROC	TCI	UP	0	TCF2EN	TCF1EN	TCF2	TCF1	CL2		CL1	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 DBG_EN	Debug Actions Enable Enables the TMR module to perform certain actions in response to the chip entering debug mode. 00b - Continue with normal operation during debug mode. (default) 01b - Halt TMR counter during debug mode. 10b - Force TMR output to logic 0 (prior to consideration of SCTRL[OPS]). 11b - Both halt counter and force output to 0 during debug mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 FAULT	Fault Enable The selected secondary input acts as a fault signal so that the timer OFLAG is cleared when the secondary input is set. When the secondary input is used in this mode, there is no resynchronization of the input so that there is a combinational path to clear the OFLAG. Fault inputs less than two clock periods wide will not be latched. Latched faults will be cleared the next time that the counter logic sets the OFLAG. This bit specifies whether to enable or disable the fault function. 0b - Disables 1b - Enables
12 ALT_LOAD	Alternative Load Enable Allows for an alternative method for loading the counter during modulo counting. Normally, the counter can be loaded only with the value from the LOAD register. When this bit is set, the counter is loaded from the LOAD register when counting up and a match with COMP1 occurs, or the counter is loaded from the CMPLD2 register when counting down and a match with COMP2 occurs. 0b - Counter can be re-initialized only with the LOAD register. 1b - Counter can be re-initialized with the LOAD or CMPLD2 registers depending on count direction.
11 ROC	Reload on Capture Specifies whether to enable or disable the capture function to cause the counter to be reloaded from the LOAD register. 0b - Disables 1b - Enables
10 TCI	Triggered Count Initialization Control During triggered count mode (CTRL[CM] = 110), enables the counter to be re-initialized when a second trigger occurs while the counter is still counting. Normally, the second trigger causes the counting to stop/pause until a third trigger occurs. With this bit set, a second trigger event causes the counter to re-initialize and continue counting. 0b - Stop the counter upon receiving a second trigger event while still counting from the first trigger event. 1b - Reload the counter upon receiving a second trigger event while still counting from the first trigger event.
9 UP	Counting Direction Indicator Indicates the direction of the last count during quadrature count mode (CTRL[CM] = 100). CTRL[DIR] reverses the sense of this bit. 0b - The last count was in the DOWN direction. 1b - The last count was in the UP direction.
8	RESERVED

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7 TCF2EN	Timer Compare 2 Interrupt Enable When set, enables an interrupt if TCF2 is set.
6 TCF1EN	Timer Compare 1 Interrupt Enable When set, enables an interrupt if TCF1 is set.
5 TCF2	Timer Compare 2 Interrupt Flag When set, indicates a successful comparison of the timer and the COMP2 register has occurred. This bit is sticky, and remains set until explicitly cleared by writing a zero to it.
4 TCF1	Timer Compare 1 Interrupt Flag When set, indicates a successful comparison of the timer and the COMP1 register has occurred. This bit is sticky, and remains set until explicitly cleared by writing a zero to it.
3-2 CL2	Compare Load Control 2 Controls when COMP2 is preloaded with the value from CMPLD2. 00b - Never preload 01b - Load upon successful compare with the value in COMP1 10b - Load upon successful compare with the value in COMP2 11b - Reserved
1-0 CL1	Compare Load Control 1 Controls when COMP1 is preloaded with the value from CMPLD1. 00b - Never preload 01b - Load upon successful compare with the value in COMP1 10b - Load upon successful compare with the value in COMP2 11b - Reserved

72.5.1.13 Timer Channel Input Filter Register (FILT0 - FILT3)

Offset

Register	Offset
FILT0	16h
FILT1	36h
FILT2	56h
FILT3	76h

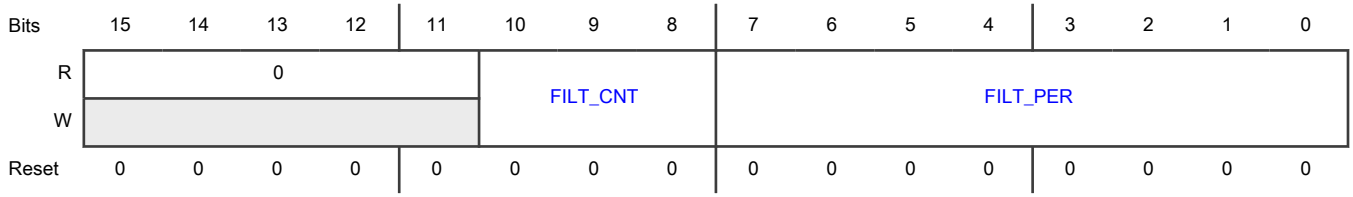
Function

The FILT register programs the values for the filtering of the corresponding input without regard for the fact that any timer channel can use the input as a count source.

Input filter considerations:

- Set the FILT_PER value such that the sampling period is larger than the period of the expected noise. In this way, a noise spike will corrupt only one sample. Choose the FILT_CNT value to reduce the probability that noisy samples cause an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of (FILT_CNT + 3).
- The values of FILT_PER and FILT_CNT must also be balanced against the desire for minimal latency in recognizing input transitions. Turning on the input filter (setting FILT_PER to a non-zero value) introduces a latency of (((FILT_CNT + 3) x FILT_PER) + 2) IP bus clock periods.

Diagram



Fields

Field	Function
15-11 —	RESERVED
10-8 FILT_CNT	Input Filter Sample Count Indicates the number of consecutive samples that must agree prior to the input filter accepting an input transition. A value of 0x0 indicates 3 samples. A value of 0x7 indicates 10 samples. The value of FILT_CNT affects the input latency.
7-0 FILT_PER	Input Filter Sample Period Indicates the sampling period (in IP bus clock cycles) of the TMR input signals. Each input is sampled multiple times at the rate specified by this field. If FILT_PER is 0x00 (default), then the input filter is bypassed. The value of FILT_PER affects the input latency. When changing values for FILT_PER from one non-zero value to another non-zero value, write a value of zero first to clear the filter.

72.5.1.14 Timer Channel DMA Enable Register (DMA0 - DMA3)

Offset

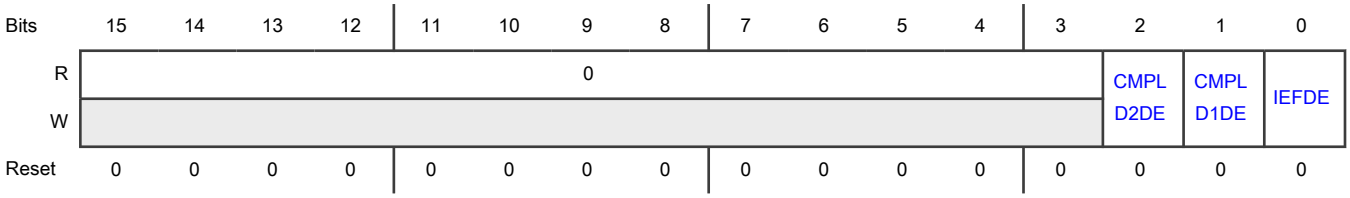
Register	Offset
DMA0	18h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
DMA1	38h
DMA2	58h
DMA3	78h

Diagram



Fields

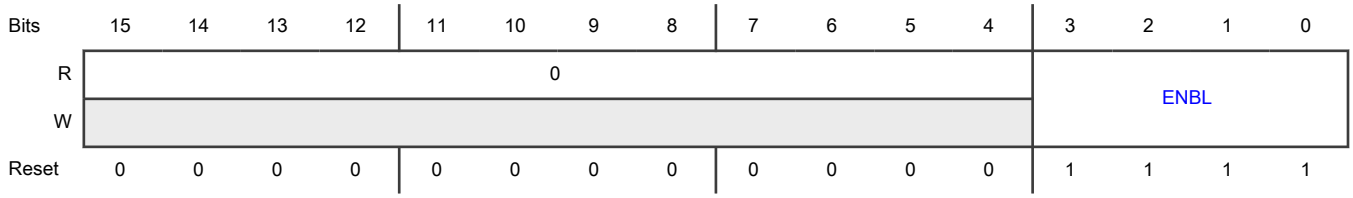
Field	Function
15-3 —	RESERVED
2 CMPLD2DE	Comparator Preload Register 2 DMA Enable Setting this bit enables DMA write requests for CMPLD2 whenever data is transferred out of the CMPLD2 register into the CNTR or COMP2 registers.
1 CMPLD1DE	Comparator Preload Register 1 DMA Enable Setting this bit enables DMA write requests for CMPLD1 whenever data is transferred out of the CMPLD1 register into the COMP1 register.
0 IEFDE	Input Edge Flag DMA Enable Setting this bit enables DMA read requests for CAPT when SCTRL[IEF] is set. **RESTRICTION** Do not set both this bit and SCTRL[IEFIE].

72.5.1.15 Timer Channel Enable Register (ENBL)

Offset

Register	Offset
ENBL	1Eh

Diagram



Fields

Field	Function
15-4 —	RESERVED
3-0 ENBL	Timer Channel Enable Specifies whether to enable or disable the prescaler (if it is being used) and counter in each channel. Multiple ENBL bits can be set at the same time to synchronize the start of separate counters. If an ENBL bit is set, then the corresponding channel starts its counter as soon as the CTRL[CM] field has a value other than 0. When an ENBL bit is clear, the corresponding counter maintains its current value. 0000b - Disables the timer channel. 0001b - Enables the timer channel. (default)

Chapter 73
Quadrature Decoder (eQDC)

73.1 Chip-specific eQDC Information

Table 1085. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

73.2 Overview

eQDC does the following:

- Interfaces to position or speed sensors that are used in industrial motor control applications.
- Decodes shaft position, revolution count, and speed.

eQDC receives the following 8 input signals from position or speed sensors:

- PHASEA
- PHASEB
- INDEX/PRESET
- TRIGGER
- HOME/ENABLE
- ICAP[3:1]

eQDC outputs the following 11 signals for system use:

- POS_MATCH[3:0]
- COMP_FLG[3:0]
- DIR
- CNT_DN

- CNT_UP

NOTE

For eQDC, "asserted" indicates an output of 1 and "de-asserted" indicates an output of 0.

73.2.1 Block diagram

This is the block diagram of eQDC.

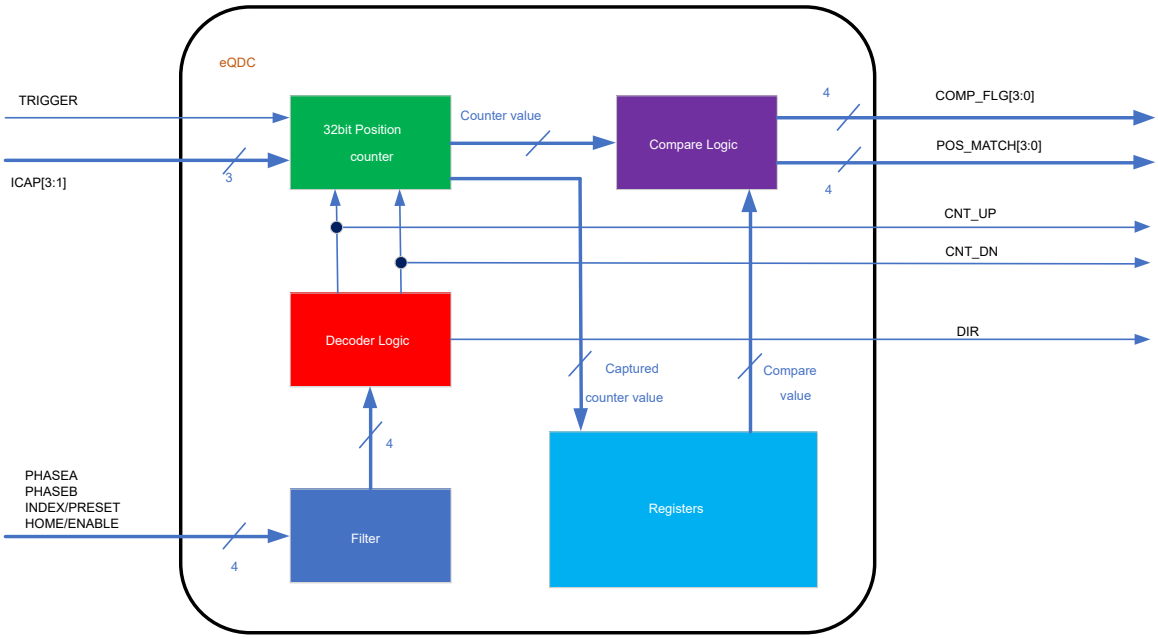


Figure 580. Block diagram

73.2.2 Features

- Logic to decode quadrature signals
- Configurable digital filters for inputs (these filters can be bypassed)
- 32-bit position counter capable of modulo counting
- Position counter that can be initialized by software or external events
- 16-bit position difference register
- Compare function that can indicate when shaft has reached a defined position
- A watchdog timer that can detect a non-rotating shaft condition
- Preloadable 16-bit revolution counter
- Maximum count frequency equals the peripheral clock rate
- Configurable interrupt when both PHASEA and PHASEB inputs change in the same cycle

73.3 Functional description

The following timing diagram shows the basic operation of an incremental position quadrature decoder.

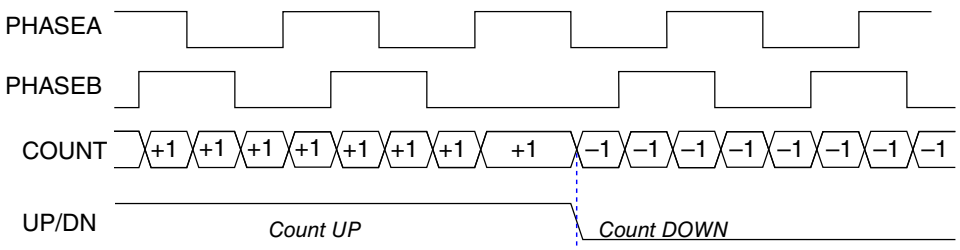


Figure 581. Quadrature decoder signals

73.3.1 Positive versus negative direction

A typical quadrature encoder has 3 outputs: PHASEA signal, PHASEB signal, INDEX pulse (not shown).

- If PHASEA leads PHASEB, then motion is in the positive direction.
- If PHASEA trails PHASEB, then motion is in the negative direction.

Transitions on these phases can be integrated to yield position or differentiated to yield velocity. The quadrature decoder is designed to perform these functions in hardware.

73.3.2 Speed measurement

For applicationswith a fast-moving shaft encoder, the speed can be measured using either of the following methods:

- calculating the change in the position counter per unit time
- reading the position difference counter register

For applications with low motor speeds and low-line count quadrature encoders, the timer module enables high-resolution speed measurement by calculating the time intervals between quadrature phases.

- The timer module uses a 16-bit free running counter operated from a prescaled version of the peripheral clock.
- The prescaler FILT[PRSC] divides the peripheral clock by values ranging from 1 to 32768. A 100 MHz peripheral clock frequency would yield a resolution of from 10 ns to 327.68 µs and a maximum count period of from 0.65535 ms to 21474.5 ms. For example, with a 1000-tooth decoder, speeds could be calculated down to 0.0007 rpm using a prescaler.

73.3.3 Glitch filter

Because the quadrature decoder logic must sense signal transitions, the signal inputs are first run through a glitch filter. This glitch filter has a digital delay line, which samples multiple time points on the signal and verifies a stable new signal state before outputting this new signal state to the internal quadrature decoder logic. To adapt to a variety of signal bandwidths, the sample rate of this delay line is programmable.

73.3.4 Edge detect state machine

TheEdge detect state machine looks for changes in the 4 possible states of the filtered PHASEA and PHASEB input signals. Direction of motion is calculated using these changes. Direction of motion is formatted as Count_Up and Count_Down signals. These signals are routed into up to 3 up/down counters:

- Position counter
- Revolution counter
- Position difference counter

73.3.5 Counter registers, and hold registers and how to initilize

When any of the counter registers is read, the contents of this counter register is written to the corresponding hold register. Taking a snapshot of the counters' values allows a consistent view of a system's position and the speed to be attained. If CTRL2[PMEN] is set, the position difference hold register is only updated when the position difference counter register is read. To capture a time stamp of when these registers are read, use the POS_MATCH output with a timer channel.

73.3.5.1 Position counter (POS)

POS counts up (upper position counter, UPOS) or down (lower position counter, LPOS) on every count pulse generated by the position difference of PHASEA and PHASEB input signals. POS acts as an integration_info, whose count value is proportional to position. De-assertion of ENABLE pin stops the operation of POS. POS is initialized to a pre-determined value (if CTRL[REV]=0,POS is initialized to the initialization register; if CTRL[REV]=1, POS is initialized to the modulus register) if any of the following conditions is met:

- If CTRL2[INITPOS] is set, a positive edge of TRIGGER input occurs.
- A write to CTRL[SWIP]
- If CTRL2[OPMODE] is cleared and CTRL[XIP] is set to 1, INDEX signal transition occurs.
- IfCTRL2[OPMODE] is cleared and CTRL[HIP] is set to 1, HOME signal transition occurs.
- If CTRL2[OPMODE] is set, a positiveedge of PRESET signal occurs.
- Position counter roll-over and roll-under

The INDEX and HOME signals can be programmed to interrupt the processor. When UPOS or LPOS is read, a snapshot of each of the following counter registers is placed into the corresponding hold register:

- position counter register
- If CTRL2[PMEN] is cleared, position difference counter register
- revolution counter register

73.3.5.2 Position counter hold (POSH)

POSH stores a copy of POS when POS is read.

When POS is initialized by any event, the contents of POS are also copied into POSH. Meanwhile, count direction flag IOMR[DIR] is copied into count direction flag hold IOMR[DIRH]. When CTRL2[UPDHLD] is set to1, the rising edge of TRIGGER input takes a snapshot of POS and stores it in UPOSH or LPOS.

Beside taking snapshot of POS counter by positive transient of TRIGGER input, 3 more signals are added to take snapshot of POS counter:

- Positive transition of ICAP[3] input can take snapshot of POS counter into UPOSH3/LPOS3
- Positive transition of ICAP[2] input can take snapshot of POS counter into UPOSH2/LPOS2
- Positive transition of ICAP[1] input can take snapshot of POS counter into UPOSH1/LPOS1

Note: This is used for an error checking mechanism to check of the position counter accumulated the correct number of counts between the same event. As example, the 1024-line incremental encoder must count 4096 counts when moving in the same direction between the index.

73.3.5.3 Position difference counter (POSD)

POSD contains the position difference value occurring between each read of the position register. The position register counts up or down on every count pulse. The position difference counter acts as a differentiator, whose count value is proportional to the change in position since the last time the position counter was read.

POSD is cleared when any of the following conditions is met:

- If CTRL2[PMEN] is cleared, when POS, POSD, or REV register is read, POSD is cleared and its contents are copied into POSDH.
- If CTRL2[PMEN] is cleared, positive edge of TRIGGER input occurs when CTRL2[UPDPOS] is set
- If CTRL2[PMEN] is set, when POSD is read.
- If CTRL2[PMEN] is set, positive edge of TRIGGER input occurs when CTRL2[UPDHLD] is set.

73.3.5.4 Position difference counter hold (POSDH)

POSDH takes and stores a snapshot of POSD. The snapshot is taken when any of the following conditions is met:

- If CTRL2[PMEN] is cleared, when POS, POSD, or REV register is read.
- When POSD is read (no matter CTRL2[PMEN] is clear or set).
- If CTRL2[UPDHLD] is set, a positive edge of the TRIGGER input occurs (no matter CTRL2[PMEN] is cleared or set).

73.3.5.5 Revolution counter (REV)

REV counts or integrates revolutions by counting index pulses. The direction of the count is determined by PHASEA and PHASEB input signals. A different count direction on the rising and falling edges of the index pulse indicates that eQDC changed direction on the index pulse. If CTRL2[OPMODE] is set, de-assertion of ENABLE input signal stops the operation of REV and PRESET signal transition initializes revolution counter to zero.

73.3.5.6 Revolution counter hold (REVH)

REVH takes a snapshot of REV when either of the following conditions is met:

- When POS, POSD, or REV register is read.
- If CTRL2[UPDHLD] is set, when a positive edge of the TRIGGER input signal occurs.

73.3.5.7 Watchdog timer (WDG)

WDG ensures that the algorithm is indicating motion in the shaft; 2 successive counts indicate proper operation and resets the timer. If CTRL2[OPMODE] is set, PRESET signal transition initializes WDG r to zero, and de-assertion of ENABLE input stops the operation of the WDG.

- The timeout value is programmable.
- When a timeout occurs, an interrupt to the processor can be generated.

73.3.5.8 Last edge time counter (LASTEDGE)

LASTEDGE contains the time since the last edge on PHASEA or PHASEB. The last edge time counter counts up on every prescaled clock pulse.

When POSD is read, contents in LASTEDGE are copied into LASTEDGEH.

73.3.5.9 Position difference period counter (POSDPER)

POSDPER contains the accumulated time from the last time the position difference counter was read. It counts up on every prescaled clock pulse and is loaded from the last edge time counter when POSD is read.

73.3.6 Double-set registers loading operation

The following registers are double-set (namely outer or inner-set), which enables them to be re-configured when eQDC is operating:

- Compare registers (UCOMP0/LCOMP0,UCOMP1/LCOMP1,UCOMP2/LCOMP2,UCOMP3/LCOMP3)

- Initial registers (UINIT/LINIT)
- Modulus registers (UMOD/LMOD)

After a value is written to one of these registers, the value is "buffered" into outer-set registers temporarily. Values will be loaded into inner-set registers and take effect using the following two methods:

- If CTRL2[LDMODE] is set to 1, "buffered" values are loaded into inner-set and take effect at the next roll-over or roll-under if CTRL[LDOK] is set.
- If CTRL2[LDMODE] is set to 0, "buffered" values are loaded into inner-set and take effect immediately when CTRL[LDOK] is set.

The double-set registers can be configured manually or automatically.

To configure the double-set registers manually:

1. Initialize the double-set registers:
    a. Set CTRL2[LDMODE] to 0.
    b. Clear CTRL[LDOK] if it is not 0.
    c. Write values into double-set registers, so the values are "buffered" into outer-set registers.
    d. Set CTRL[LDOK] to 1.
    e. Wait for CTRL[LDOK] to become 0, which means the values have been loaded into inner-set registers.
    f. Change CTRL2[LDMODE] value as desired.
2. Update the double-set registers:
    a. Clear CTRL[LDOK] if it is not 0.
    b. Write values into double-set registers, so the values are "buffered" into outer-set registers.
    c. Set CTRL[LDOK] to 1. When the values are loaded into inner-set registers, CTRL[LDOK] is cleared automatically.
    d. Go back to step a for iteration.

To configure the double-set registers automatically:

1. Initialize the double-setregisters:
    a. Set CTRL2[LDMODE] to 0.
    b. Clear CTRL[LDOK] if it is not 0.
    c. Write values into double-set registers, so the values are "buffered" into outer-set registers.
    d. Set CTRL[LDOK] to 1.
    e. Wait for CTRL[LDOK] to become 0, which means the values have been loaded into inner-set registers.
    f. Set CTRL2[LDMODE] to 1.
    g. Set CTRL[DMAEN] to 1 to enable DMA function.
    h. Set CTRL[LDOK] to 1. This step is to ensure the DMA automatic operation occurs properly. DMA is triggered by loading values from outer-set to inner-set.
2. After initialization, the DMA is triggered in the following manner:
    a. Values are loaded into inner-set registers when the counter rolls over or rolls under, then CTRL[LDOK] is cleared automatically and a DMA request is sent out for next outer-set registers value update.
    b. After DMA updates the outer-set registers, CTRL[LDOK] is set to 1 automatically. It then goes back to step 1 for iteration.

73.3.7 Modes of operation

eQDC operates in the following modes:

- Quadrature Decode Operation mode (CTRL[PH1] = 0,CTRL2[OPMODE] = 0)
- Quadrature Count Operation mode (CTRL[PH1] = 0,CTRL2[OPMODE] = 1)
- Single Phase Decode Operation mode (CTRL[PH1] = 1,CTRL2[OPMODE] = 0)
- Single Phase Count Operation mode (CTRL[PH1] = 1,CTRL2[OPMODE] = 1)

These operation modes support the following applications:

- Pulse accumulator
- Frequency Meter
- Period Meter
- PulseTrain Output with optional Quadrature and Direction

73.3.7.1 Quadrature decode (QDC) operation mode (CTRL[PH1] = 0, CTRL2[OPMODE] = 0)

In this mode, PHASEA and PHASEB input signals from speed or position sensors are decoded by the following signals in eQDC:

1. PHASEA/PHASEB: Two 90 degree out of phase pulse trains.
2. INDEX: Can be configured to cause a change of state on REV, initialize POS, and reset POSD.
3. HOME: Can be configured to initialize POS.
4. TRIGGER: Can be configured to take a snapshot of POS, POSD, and REV and to reinitialize POS.
5. ICAP[3:1]: Takes a snapshot of POS.

eQDC outputs the following signals for system use:

- POS_MATCH[3:0]: Position match:

  Records the time at which the position of the shaft matches a user-defined value of the compare regsietr or the time at which the position information is read:

  — If CTRL2[OUTCTL] is cleared, POS_MATCH[x](x range is 0-3) is asserted when the value of POS equals the value of the corresponding compare regsiter (UCOMPx/LCOMPx), and de-asserted when the value of POS does not equal the value of the corresponding compare register (UCOMPx/LCOMPx).

  — If CTRL2[OUTCTL] is set, POS_MATCH[x](x range is 0-3) is asserted a pulse with 1 peripheral clock cycle width when UPOS or LPOS, REV, or POSD is read.

- COMP_FLG[3:0]: Position Counter Compare Output:

  — COMP_FLG[x](x range is 0-3) is asserted when the value of Position Counter is equal to or greater than the value of the corresponding compare register (UCOMPx/LCOMPx).

  — COMP_FLG[x](x range is 0-3) is de-asserted when the value of Position Counter is smaller than the value of the corresponding compare register (UCOMPx/LCOMPx).

- DIR: Direction of position counting:

  — Direction is up when DIR is asserted

  — Direction is down when DIR is de-asserted.

- CNT_UP: Position counter count up: when eQDC decodes a count-up event, CNT_UP outputs a pulse which width is 1 peripheral clock cycle.
- CNT_DN: Position counter count down: when eQDC decodes a count-down event, CNT_DN outputs a pulse which width is 1 peripheral clock cycle.

There are three count modes(if CTRL[REV]=0,it is normal operation; if CTRL[REV]=1,it is reverse operation):

- Mode 0[CM0](CTRL2[CMODE]=00): Normal Quadrature X4 / Reverse Quadrature X4
- Mode 1[CM1](CTRL2[CMODE]=01): Normal Quadrature X2 / Reverse Quadrature X2
- Mode 2[CM2](CTRL2[CMODE]=10): Normal Quadrature X1 / Reverse Quadrature X1

73.3.7.2 Quadrature count (QCT) operation mode (CTRL[PH1] = 0, CTRL2[OPMODE] = 1)

In this mode, PHASEA and PHASEB input signals from speed or position sensors are counted by the following signals in eQDC:

- PHASEA/PHASEB: Two 90 degree out of phase pulse trains.
- PRESET: Initializes POSand resets POSD:
  - If CTRL[REV]=0, a positive edge of PRESET initializes POS to INIT value (UNIT/LINIT).
  - If CTRL[REV]=1, a positive edge of PRESET initializes POS to modulus value (UMOD/LMOD).
- ENABLE: All counters start counting when ENABLE is asserted; all counters stop counting when ENABLE is de-asserted.
- TRIGGER: Can be configured totake a snapshot of POS, POSD, and REV and to reinitialize POS.
- ICAP[3:1]: Takes a snapshot of POS.

eQDC outputs the following signals for system use:

- POS_MATCH[3:0]: position match:

  Records the time at which the position of the shaft matches a user-defined value of the compare regsietr or the time at which the position information is read:
  - If CTRL2[OUTCTL] is cleared, POS_MATCH[x](x range is 0-3) is asserted when the value of POS equals the value of the corresponding compare regsiter (UCOMPx/LCOMPx), and de-asserted when the value of POS does not equal the value of the corresponding compare register (UCOMPx/LCOMPx).
  - If CTRL2[OUTCTL] is set, POS_MATCH[x](x range is 0-3) is asserted a pulse with 1 peripheral clock cycle width when UPOS or LPOS, REV, or POSD is read.
- COMP_FLG[3:0]: Position Counter Compare Output:
  - COMP_FLG[x](x range is 0-3) is asserted when the value of POS is equal to or greater than the value of the corresponding compare register (UCOMPx/LCOMPx).
  - COMP_FLG[x](x range is 0-3) is de-asserted when the value of POS is smaller than the value of the corresponding compare register (UCOMPx/LCOMPx).
- DIR: Direction of position counting:
  - Direction is up when DIR is asserted
  - Direction is down when DIR is de-asserted.
- CNT_UP: Position counter count up: when eQDC decodes a count-up event, CNT_UP outputs a pulse which width is 1 peripheral clock cycle.
- CNT_DN: Position counter count down: when eQDC decodes a count-down event, CNT_DN outputs a pulse which width is 1 peripheral clock cycle.

There are three count modes(if CTRL[REV]=0,it is normal operation; if CTRL[REV]=1,it is reverse operation):

- Mode 0[CM0](CTRL2[CMODE]=00): Normal Quadrature X4 / Reverse Quadrature X4
- Mode 1[CM1](CTRL2[CMODE]=01): Normal Quadrature X2 / Reverse Quadrature X2
- Mode 2[CM2](CTRL2[CMODE]=10): Normal Quadrature X1 / Reverse Quadrature X1

73.3.7.3 Single phase decode (PH1DC) operation mode (CTRL[PH1] = 1, CTRL2[OPMODE] = 0)

In this mode, PHASEA and PAHSEB input signals from speed or position sensors are decoded by the following signals in eQDC:

- PHASEA: Pulse trains.
- PHASEB: Pulse trains or direction.
- INDEX: Can be configured to cause a change of state on REV and to initialize POS.
- HOME: Can be configured to initialize POS.
- TRIGGER: Can be configured to take a snapshot of POS, POSD, and REV and to reinitialize POS.
- ICAP[3:1]: Takes a snapshot of POS.

eQDC outputs the following signals for system use:

- POS_MATCH[3:0]: position match:

  Records the time at which the position of the shaft matches a user-defined value of the compare regsietr or the time at which the position information is read:

  — If CTRL2[OUTCTL] is cleared, POS_MATCH[x](x range is 0-3) is asserted when the value of POS equals the value of the corresponding compare regsiter (UCOMPx/LCOMPx), and de-asserted when the value of POS does not equal the value of the corresponding compare register (UCOMPx/LCOMPx).

  — If CTRL2[OUTCTL] is set, POS_MATCH[x](x range is 0-3) is asserted a pulse with 1 peripheral clock cycle width when UPOS or LPOS, REV, or POSD is read.

- COMP_FLG[3:0]: Position Counter Compare Output:

  — COMP_FLG[x](x range is 0-3) is asserted when the value of POS is equal to or greater than the value of the corresponding compare register (UCOMPx/LCOMPx).

  — COMP_FLG[x](x range is 0-3) is de-asserted when the value of POS is smaller than the value of the corresponding compare register (UCOMPx/LCOMPx).

- DIR: Direction of position counting:

  — Direction is up when DIR is asserted

  — Direction is down when DIR is de-asserted.

- CNT_UP: Position counter count up: when eQDC decodes a count-up event, CNT_UP outputs a pulse which width is 1 peripheral clock cycle.
- CNT_DN: Position counter count down: when eQDC decodes a count-down event, CNT_DN outputs a pulse which width is 1 peripheral clock cycle.

There are three count modes:

- Mode 0[CM0](CTRL2[CMODE]=00): UP/DOWN Pulse Count mode. Both position counter (POS) and position difference counter (POSD) count in the up direction when input PHASEA rising edge occurs. Both counters count in the down direction when input PHASEB rising edge occurs. If CTRL[REV] is 1, then the position counter will count in the opposite direction.
- Mode 1[CM1](CTRL2[CMODE]=01): Signed Count mode (Double Edge). Both position counter (POS) and position difference counter (POSD) count the input PHASEA on both rising edge and falling edge while the input PHASEB provides the selected position counter direction (up/down). If CTRL[REV] is 1, then the position counter will count in the opposite direction.
- Mode 2[CM2](CTRL2[CMODE]=10): Signed Count mode (Single Edge). Both position counter (POS) and position difference counter (POSD) count on the input PHASEA rising edge while the input PHASEB provides the selected position counter direction (up/down). If CTRL[REV] is 1, then the position counter will count in the opposite direction.

NOTE

If the positive edge of both PHASEA and PHASEB occurs simultaneously, an error interrupt is generated CTRL2[SABIRQ] and counter is unchanged.

73.3.7.4 Single phase count (PH1CT) operation mode (CTRL[PH1] =1, CTRL2[OPMODE] = 1)

In this mode, PHASEA and PHASEB input signals from speed or position sensors are counted by the following signals in eQDC:

- PHASEA: Pulse trains.
- PHASEB: Pulse trains or direction.
- PRESET: InitializesPOS and resets POSD:
  - IfCTRL[REV]=0, a positive edge of PRESET initializes POS to initial value (UNIT/LINIT).
  - IfCTRL[REV]=1, a positive edge of PRESET initializes POS to modulus value (UMOD/LMOD).
- ENABLE: All counters start counting when ENABLE is asserted; all counters stop counting when ENABLE is de-asserted.
- TRIGGER: Can be configured to takea snapshot of POS, POSD, and REV, and to reinitialize POS.
- ICAP[3:1]: Takes a snapshot of POS.

eQDC outputs the following signals for system use:

- POS_MATCH[3:0]: position match:

  Records the time at which the position of the shaft matches a user-defined value of the compare regsietr or the time at which the position information is read:

  - If CTRL2[OUTCTL] is cleared, POS_MATCH[x](x range is 0-3) is asserted when the value of POS equals the value of the corresponding compare regsiter (UCOMPx/LCOMPx), and de-asserted when the value of POS does not equal the value of the corresponding compare register (UCOMPx/LCOMPx).
  - If CTRL2[OUTCTL] is set, POS_MATCH[x](x range is 0-3) is asserted a pulse with 1 peripheral clock cycle width when UPOS or LPOS, REV, or POSD is read.
- COMP_FLG[3:0]: Position Counter Compare Output:
  - COMP_FLG[x](x range is 0-3) is asserted when the value of POS is equal to or greater than the value of the corresponding compare register (UCOMPx/LCOMPx).
  - COMP_FLG[x](x range is 0-3) is de-asserted when the value of POS is smaller than the value of the corresponding compare register (UCOMPx/LCOMPx).
- DIR: Direction of position counting:
  - Direction is up when DIR is asserted
  - Direction is down when DIR is de-asserted.
- CNT_UP: Position counter count up: when eQDC decodes a count-up event, CNT_UP outputs a pulse which width is 1 peripheral clock cycle.
- CNT_DN: Position counter count down: when eQDC decodes a count-down event, CNT_DN outputs a pulse which width is 1 peripheral clock cycle.

There are three count modes:

- Mode 0[CM0](CTRL2[CMODE]=00): UP/DOWN Pulse Count mode. Both position counter (POS) and position difference counter (POSD) count in the up direction when input PHASEA rising edge occurs, both counters count in the down direction when input PHASEB rising edge occurs.
- Mode 1[CM1](CTRL2[CMODE]=01): Signed Count mode (Double Edge). Both position counter (POS) and position difference counter (POSD) count the input PHASEA on both rising edge and falling edge while the input PHASEB provides the selected position counter direction (up/down) .
- Mode 2[CM2](CTRL2[CMODE]=10): Signed Count mode (Single Edge). Both position counter (POS) and position difference counter (POSD) count on the input PHASEA rising edge while the input PHASEB provides the selected position counter direction (up/down) .

NOTE

If the positive edge of both PHASEA and PHASEB occurs simultaneously, an error interrupt is generated CTRL2[SABIRQ] and counter is unchanged.

73.3.7.5 Normal quadrature X1

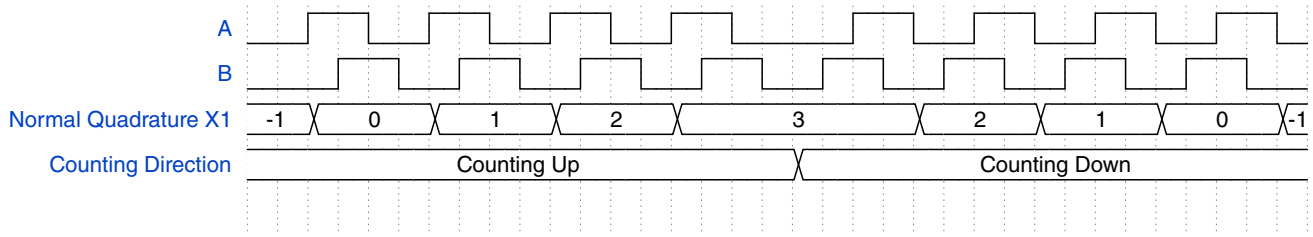


Figure 582. Normal quadrature X1

73.3.7.6 Normal quadrature X2

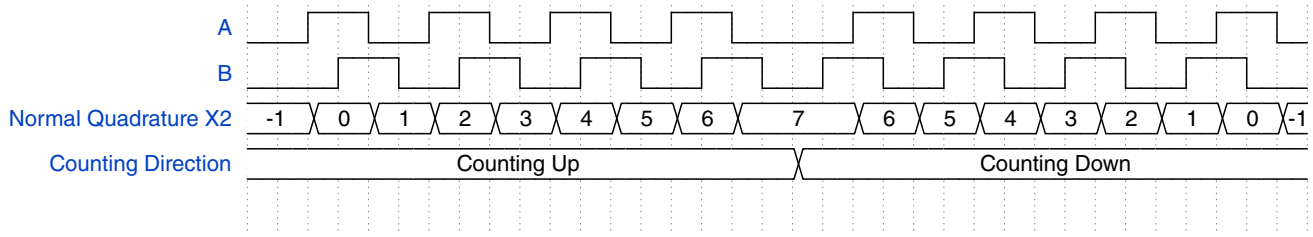


Figure 583. Normal quadrature X2

73.3.7.7 Normal quadrature X4

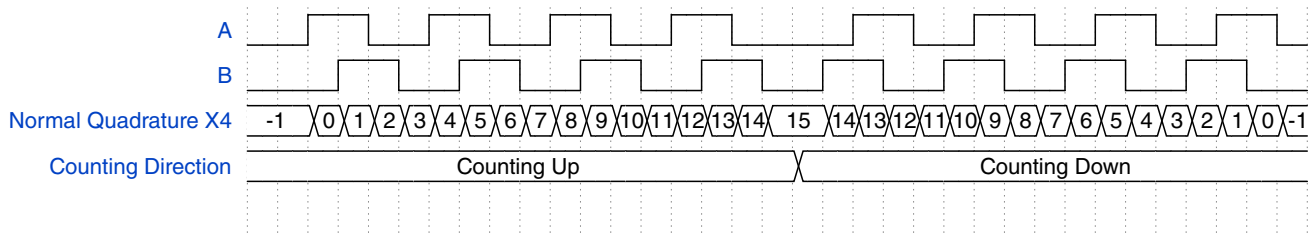


Figure 584. Normal quadrature X4

73.3.7.8 Reverse quadrature X1

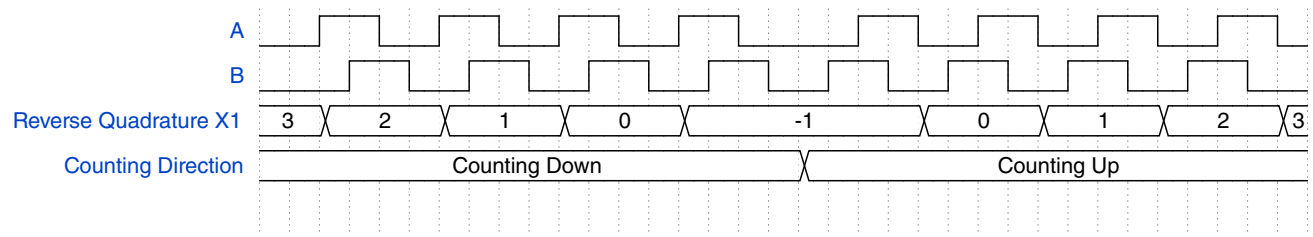


Figure 585. Reverse quadrature X1

NOTE

Set CTRL[REV] to 1 for the position counter to count in the opposite direction.

73.3.7.9 Reverse quadrature X2

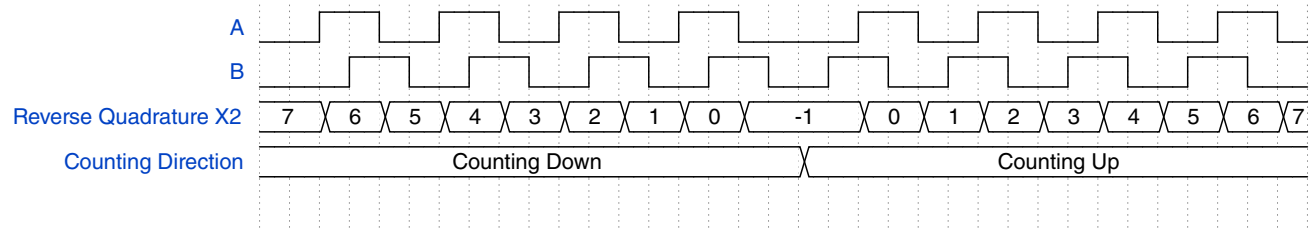


Figure 586. Reverse quadrature X2

NOTE

Set CTRL[REV] to 1 for the position counter to count in the opposite direction.

73.3.7.10 Reverse quadrature X4

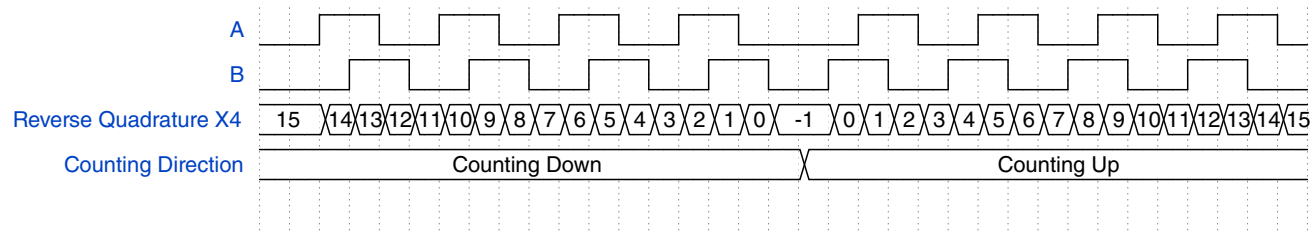


Figure 587. Reverse quadrature X4

NOTE

Set CTRL[REV] to 1 for the position counter to count in the opposite direction.

73.3.7.11 UP/DOWN pulse count mode

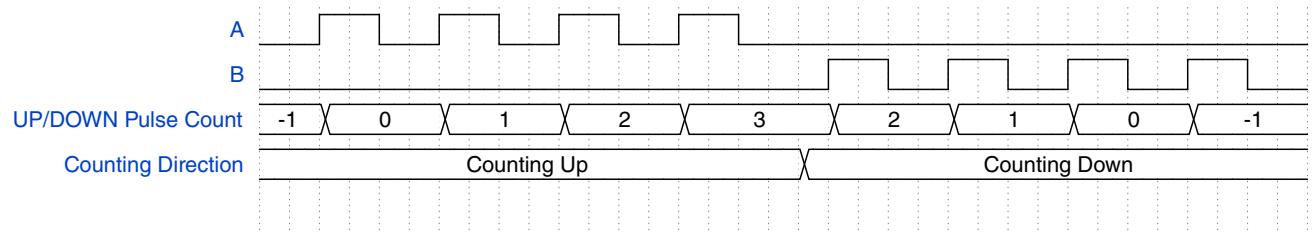


Figure 588. UP/DOWN pulse count mode

73.3.7.12 Signed count mode (double edge)

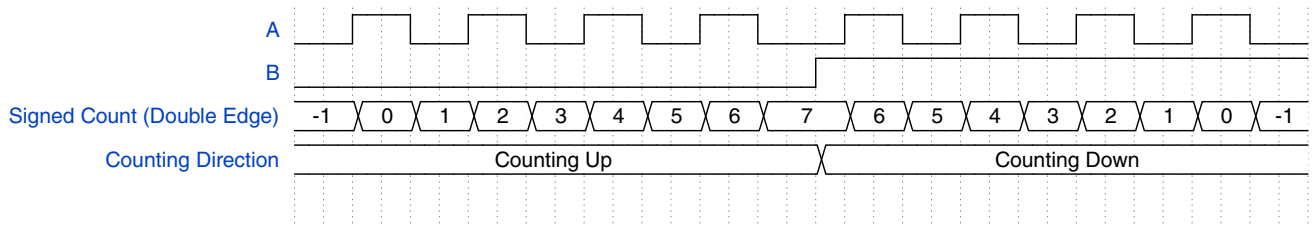


Figure 589. Signed count mode (double edge)

73.3.7.13 Signed count mode (single edge)

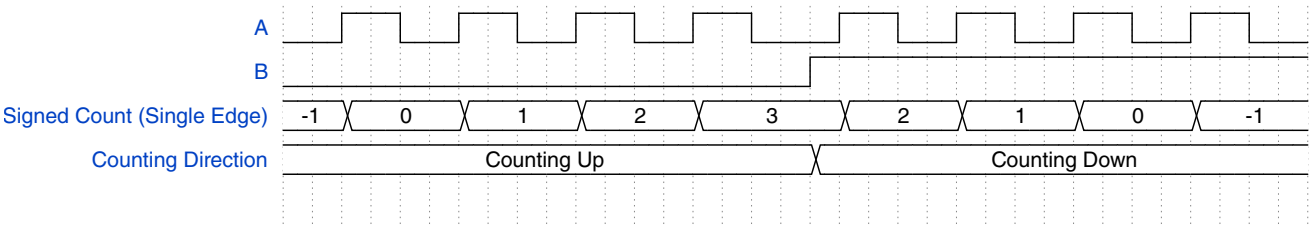


Figure 590. Signed count mode (single edge)

73.3.8 Speed measurement method

This section explains the speed measurement method, including the speed measurement algorithm.

The following figure shows the module working mechanism.

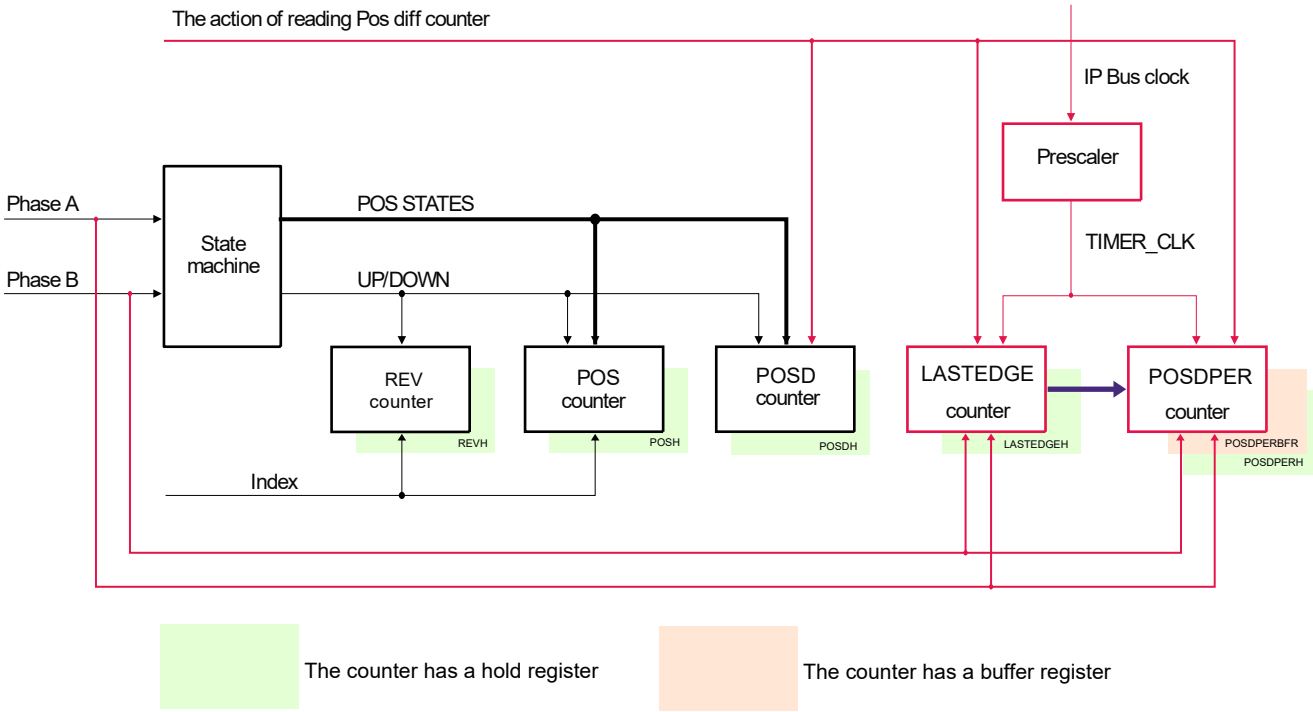


Figure 591. Quadrature decoder working mechanism

The following figure shows counters behavior and updating mechanism in the speed measurement method.

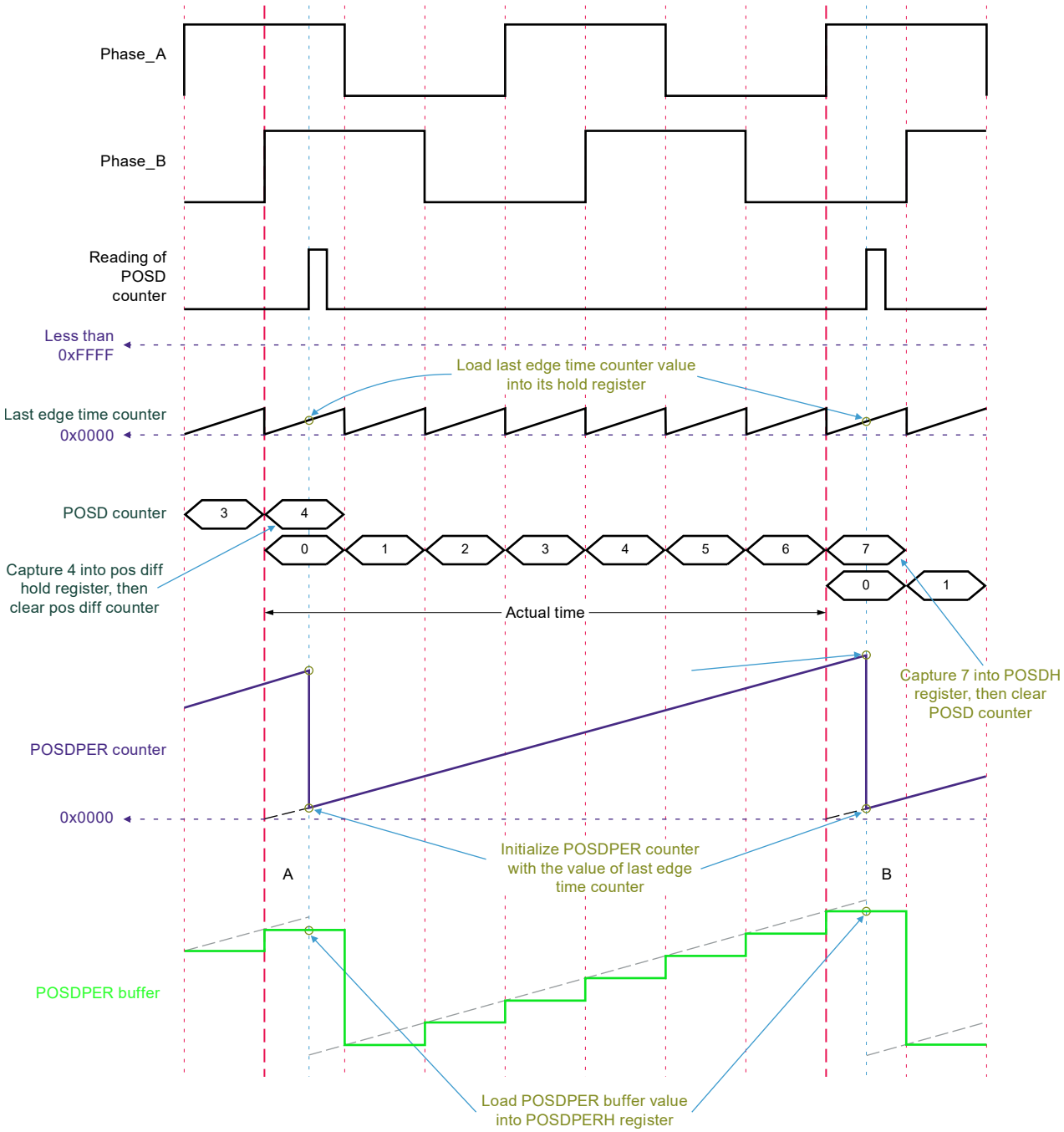


Figure 592. Counters behavior and updating mechanism

A reading of POSD occurs at time point A and it also occurs at time point B. “Actual time” represents the time duration between the first phase_a/b edges right before time points A and B. At time point B, after reading POSD, POSDPERH contains the time length of “Actual time” and POSDH contains the value of phase_a/b pulses within that “Actual time”. Speed can be calculated based on POSDPERH and POSDH. A visualized explanation of speed measurement with this method is shown in the figure below.

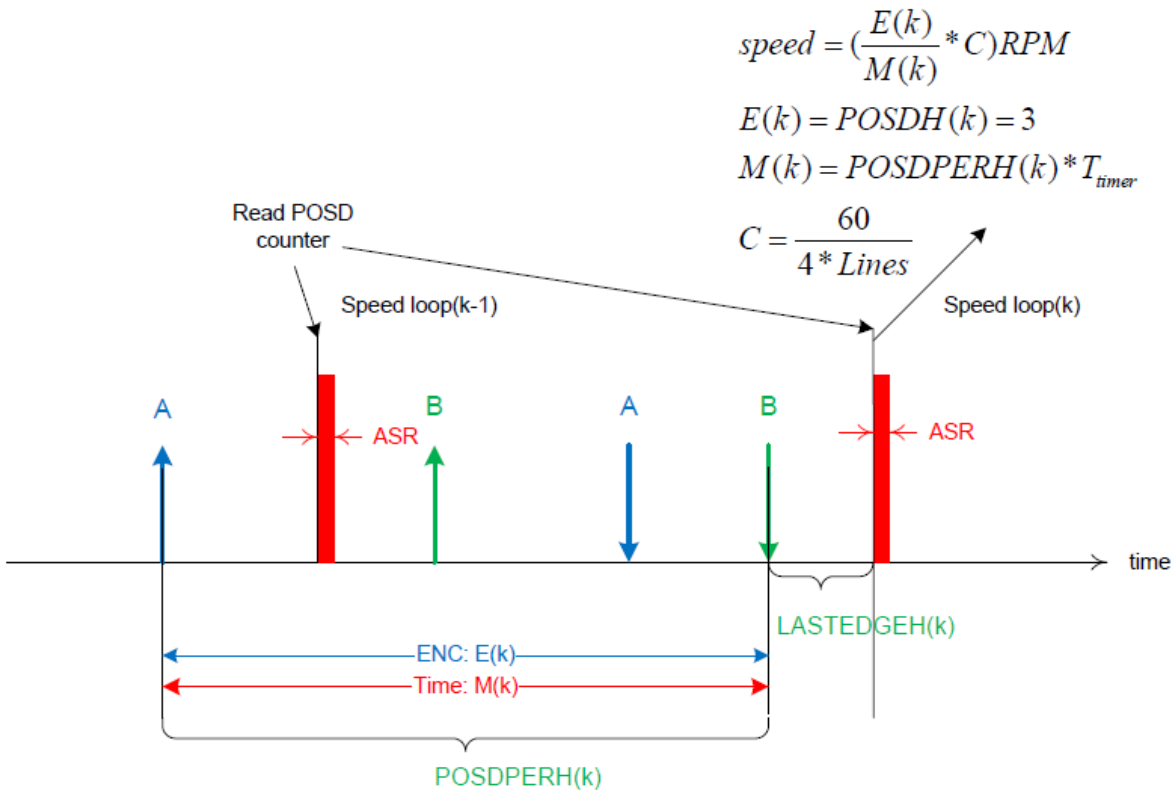


Figure 593. High speed measurement with method

Speed calculation algorithm

In order to cover all the cases in real applications, a simple speed measurement algorithm is necessary. See the algorithm shown below.

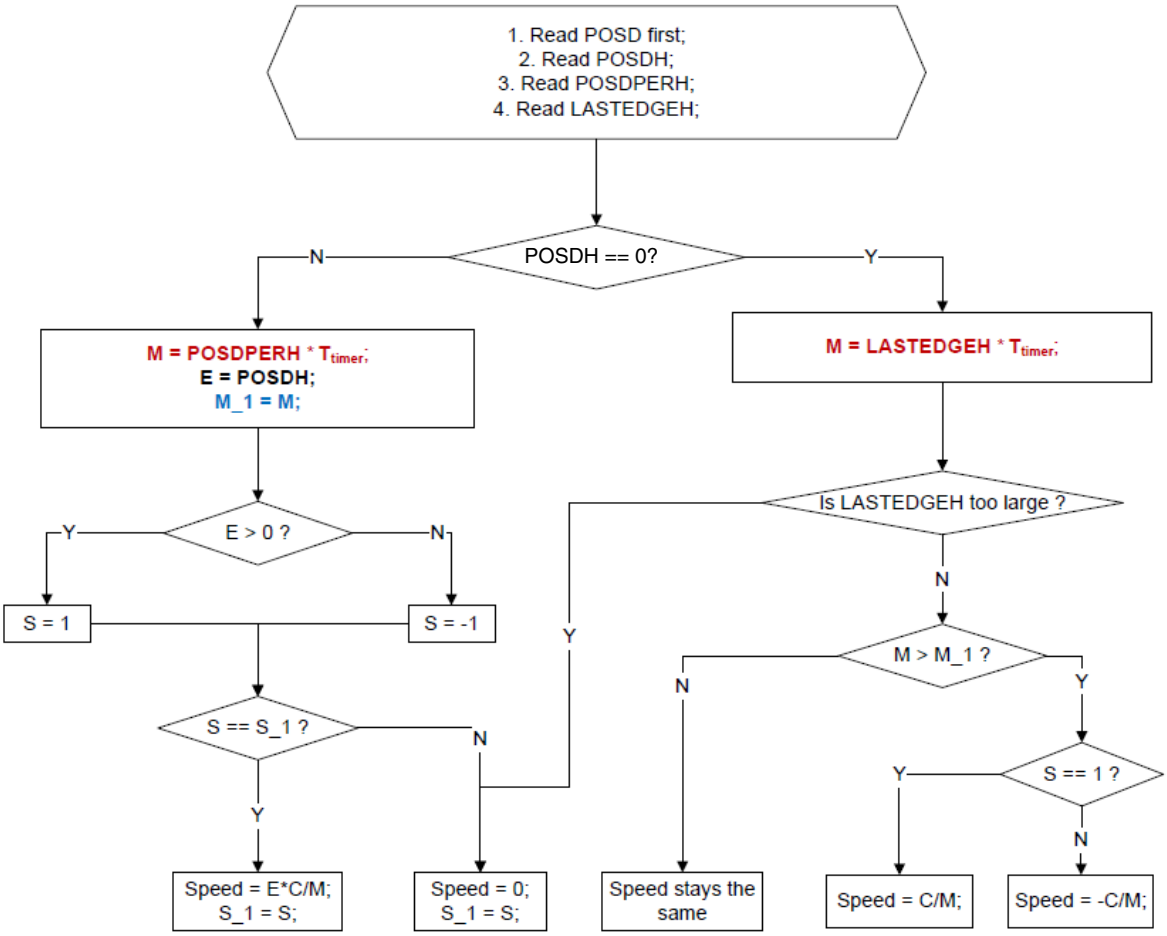


Figure 594. Flowchart of speed calculation with method

Speed calculation of motor's starting from standstill (chip is out of reset)

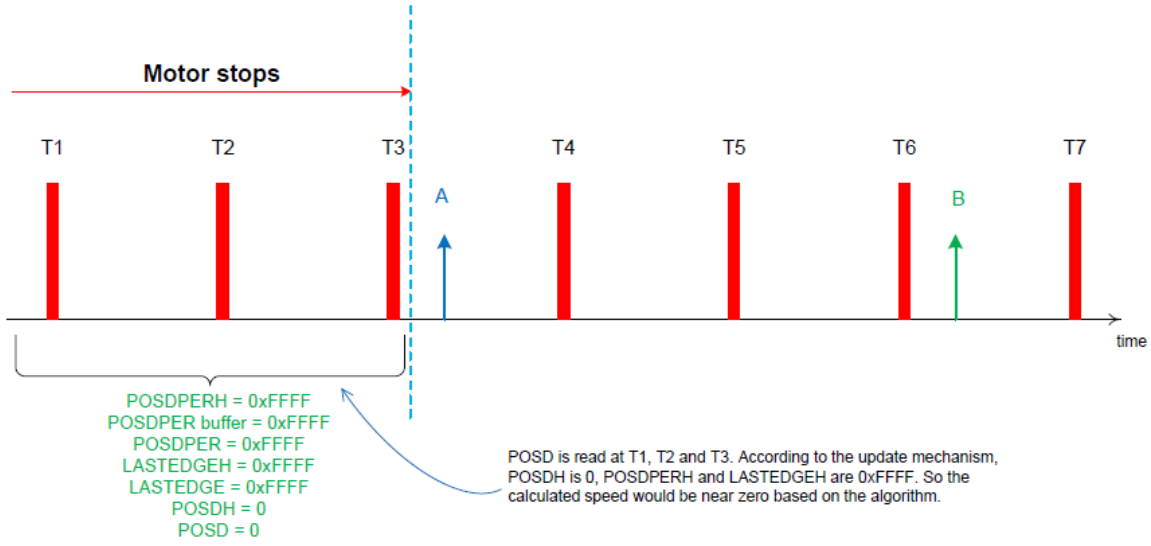


Figure 595. Speed measurement at time point T1~T3

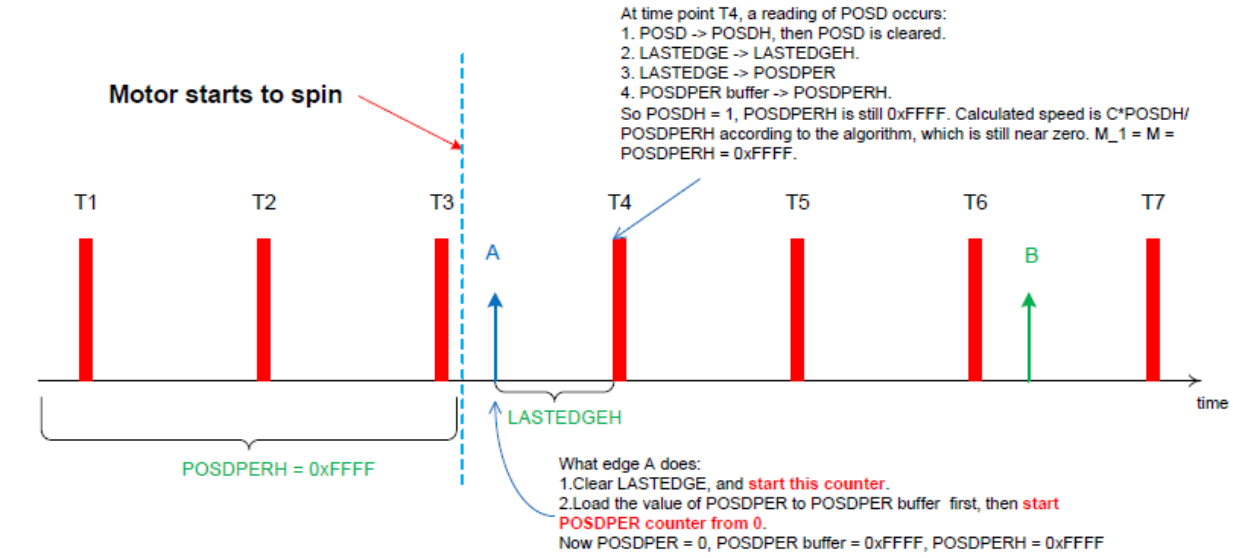


Figure 596. Speed measurement at time point T4

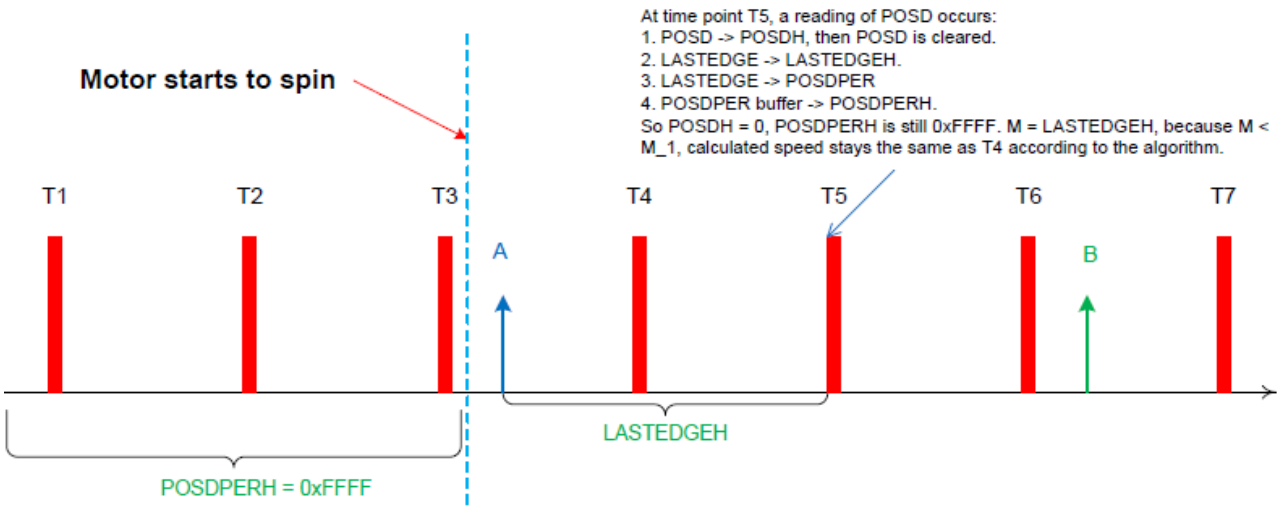


Figure 597. Speed measurement at time point T5

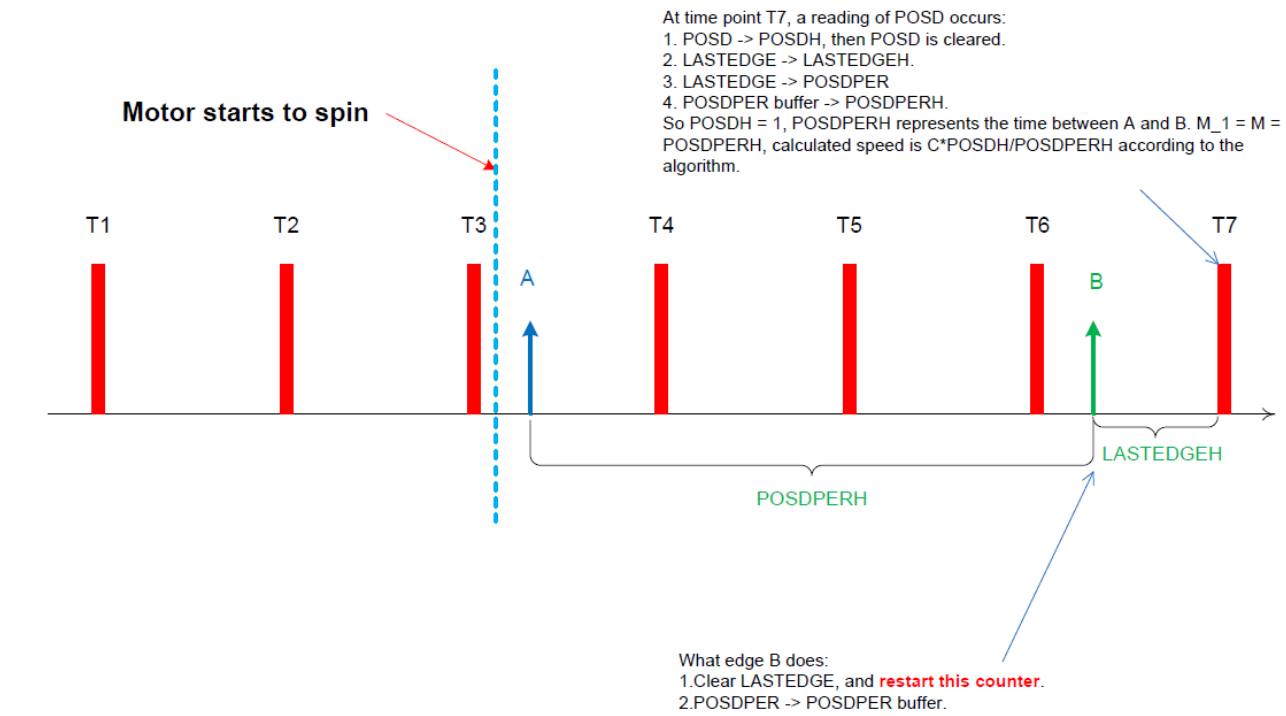


Figure 598. Speed measurement at time point T7

Speed calculation when motor stops

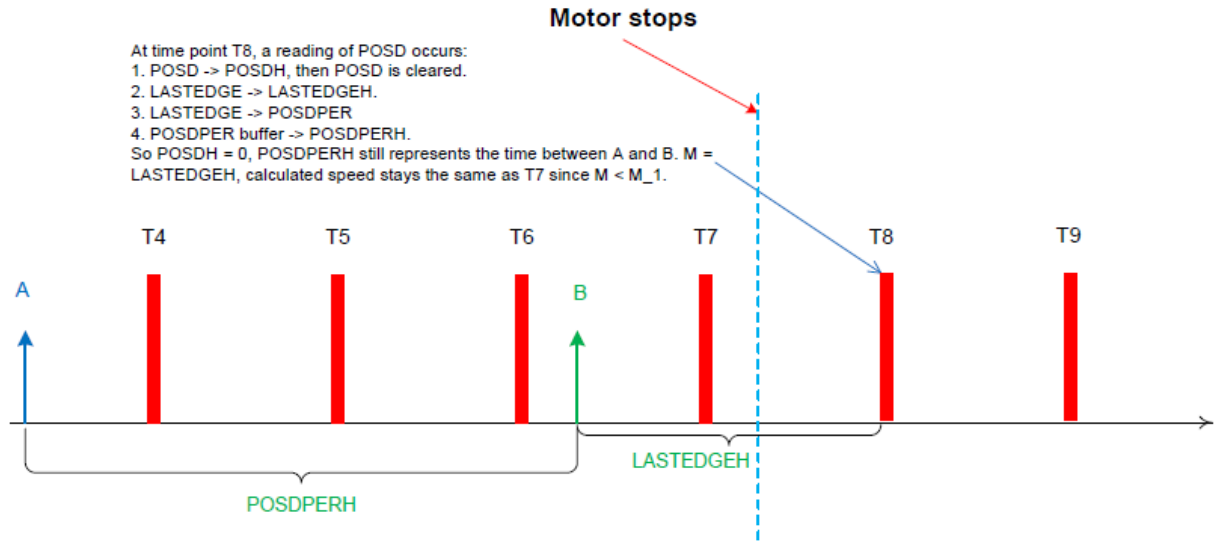


Figure 599. Speed measurement at time point T8

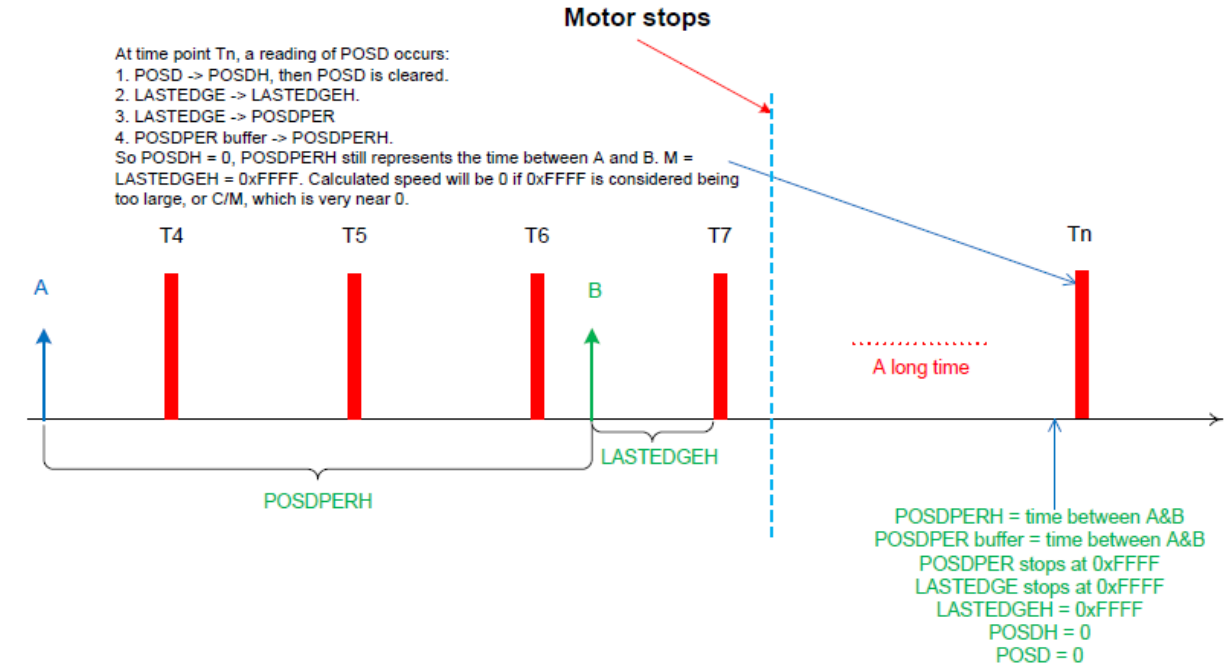


Figure 600. Speed measurement at time point Tn

Speed calculation when motor starts again

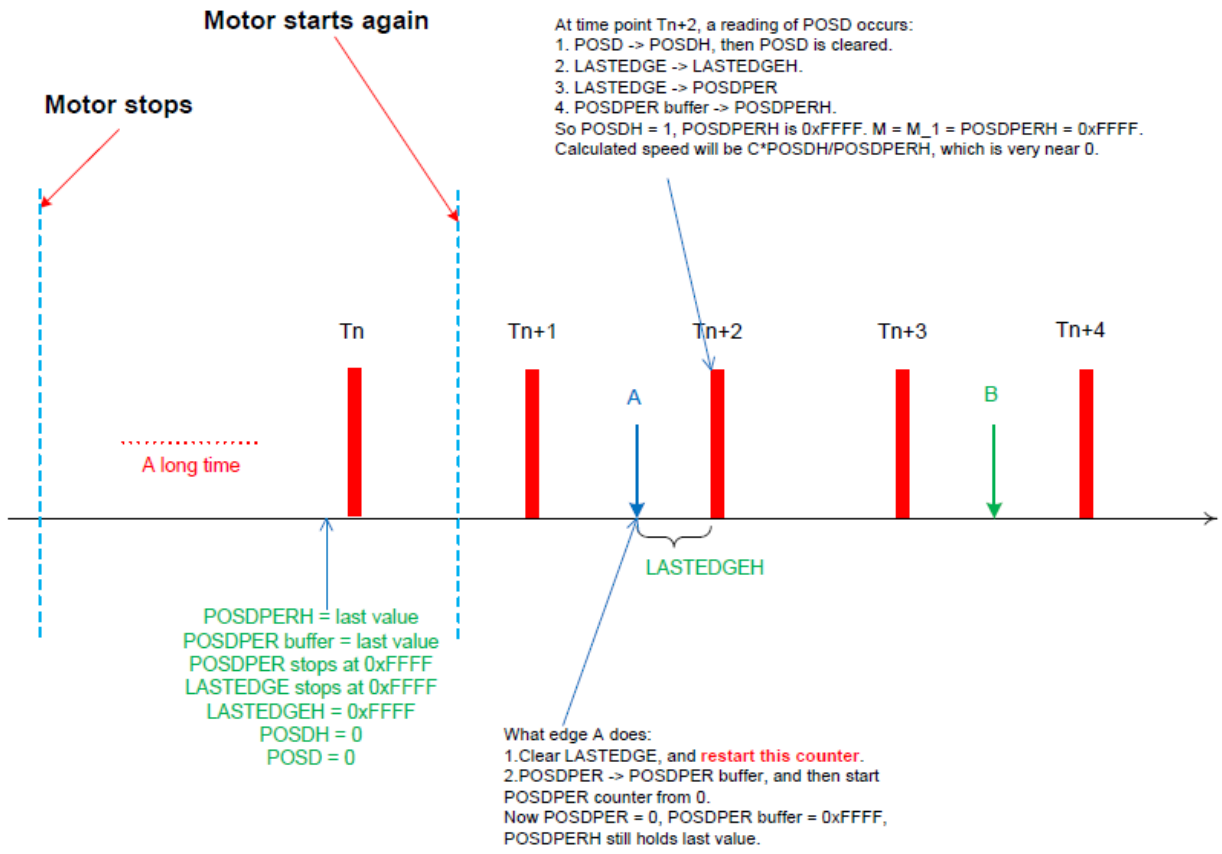


Figure 601. Speed measurement at time point Tn+2

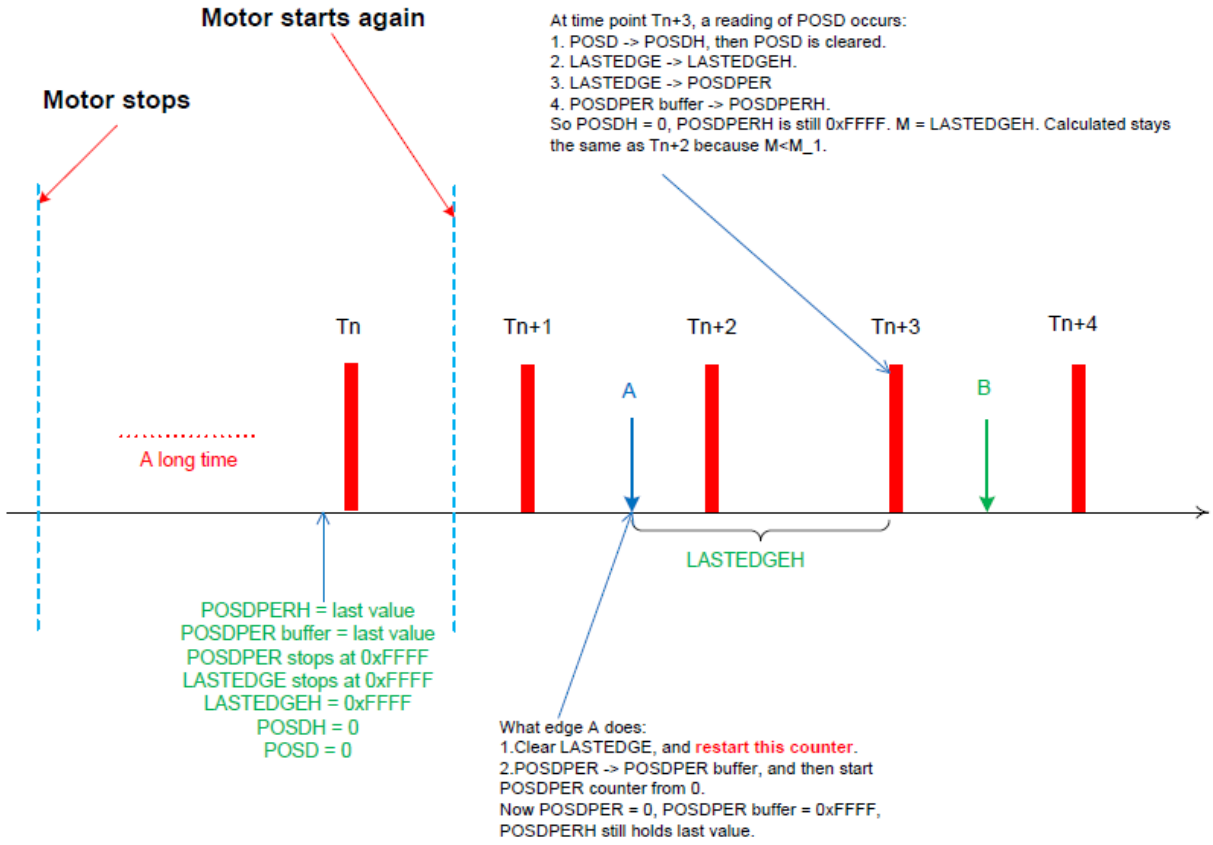


Figure 602. Speed measurement at time point Tn+3

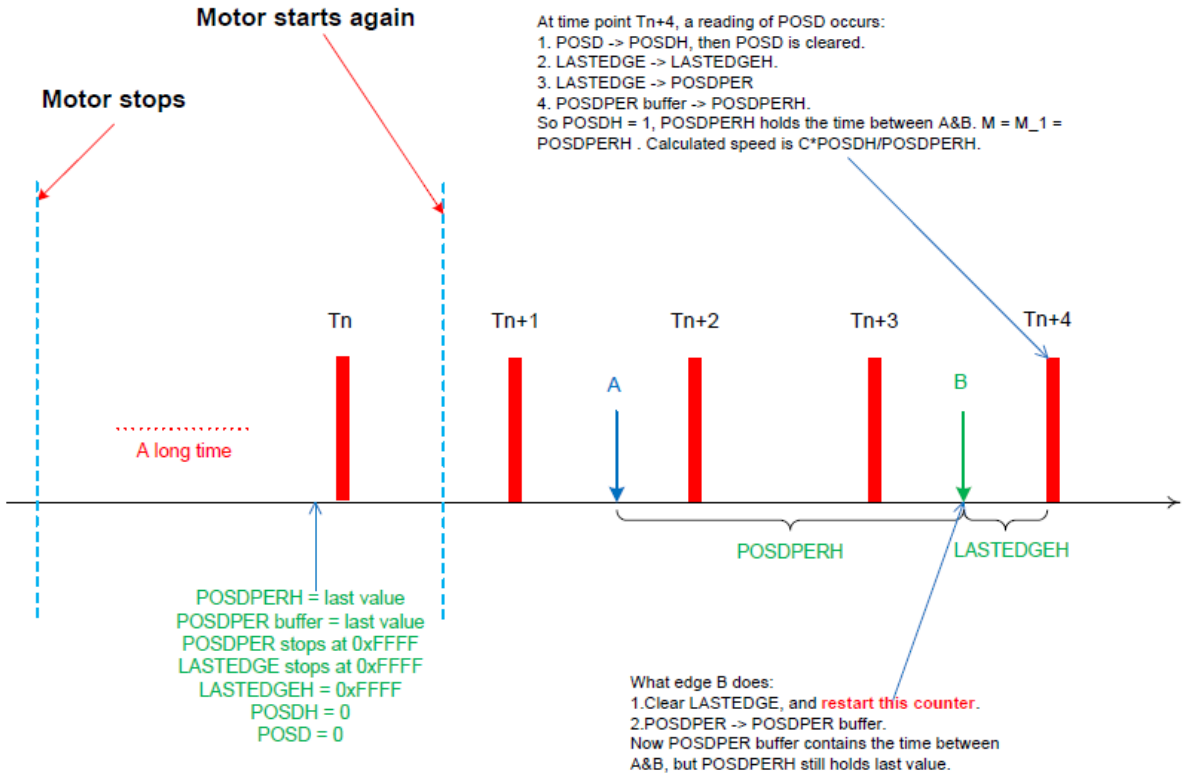


Figure 603. Speed measurement at time point Tn+4

73.3.9 Reset

eQDC is reset by any system reset. Make sure you clear all inputs (of PhaseA, PhaseB, and etc.) before resetting eQDC.

73.3.10 Clocking

The peripheral clock is the only clock required when eQDC operates.

73.3.11 Interrupts

The following table lists the module interrupts.

Table 1086. Interrupt Summary

Core Interrupt	Interrupt Flag	Interrupt Enable	Description
ipi_int_home	CTRL[HIRQ]	CTRL[HIE]	HOME/ENABLE signal transition interrupt
ipi_int_index	CTRL[XIRQ]	CTRL[XIE]	INDEX/PRESET signal transition interrupt
ipi_int_ro	INTCTRL[ROIRQ]	INTCTRL[ROIE]	roll-over interrupt
ipi_int_ru	INTCTRL[RUIRQ]	INTCTRL[RUIE]	roll-under interrupt

Table continues on the next page...

Table 1086. Interrupt Summary (continued)

Core Interrupt	Interrupt Flag	Interrupt Enable	Description
ipi_int_wdog	CTRL[WDIRQ]	CTRL[WDIE]	Watchdog timeout interrupt
ipi_int_dir	INTCTRL[DIRIRQ]	INTCTRL[DIRIE]	Count direction change interrupt
ipi_int_sab	INTCTRL[SABIRQ]	INTCTRL[SABIE]	Simultaneous PHASEA and PHASEB change interrupt

73.4 External signals

eQDC receives the following 8 input signals from position or speed sensors:

- PHASEA
- PHASEB
- INDEX/PRESET
- TRIGGER
- HOME/ENABLE
- ICAP[3:1]

eQDC outputs the following 11 signals for system use:

- POS_MATCH[3:0]
- COMP_FLG[3:0]
- DIR
- CNT_DN
- CNT_UP

Table 1087. Signals

Signal			Description
PHASEA	Phase A	Input	The PHASEA input can be connected to one of the phases from a two-phase shaft quadrature encoder output. PHASEA and PHASEB are used by eQDC to indicate a decoder increment has passed, and to calculate its direction. • When eQDC is used as a single phase decode/count, PHASEA can also be used as the single input. Direction for two-phase shaft quadrature encoder output when CTRL[REV] = 0: • PHASEA is the leading phase for a shaft rotating in the positive direction. • PHASEA is the trailing phase for a shaft rotating in the negative direction.
PHASEB	Phase B	Input	The PHASEB input can be connected to the other phase from a two-phase shaft quadrature encoder output.

Table continues on the next page...

Table 1087. Signals (continued)

Signal			Description
			• When eQDC is used as a single phase decode/count mode, PHASEB and CTRL[REV] control counter direction. Direction for two-phase shaft quadrature encoder output when CTRL[REV] = 0: • PHASEB is the trailing phase for a shaft rotating in the positive direction. • PHASEB is the leading phase for a shaft rotating in the negative direction.
INDEX/ PRESET	INDEX/ PRESET	Input	• This input can work as INDEX or PRESET, selection bit is CTRL2[OPMODE] • Normally connected to the index pulse output of a quadrature encoder, the INDEX pulse can optionally reset the position counter of eQDC. The INDEX pulse also causes a change of state on the revolution counter. The direction of this state change (increment or decrement) is calculated from the PHASEA and PHASEB inputs. In Quadrature Count mode or Single Phase Count mode, this signal can present the counter PRESET function. • The positive edge of PRESET input can initialize position/revolution/ watchdog counters.
HOME/ENABLE	HOME/ENABLE	Input	The HOME input can be used by eQDC and the timer module. • This input can work as HOME or ENABLE, selection bit is CTRL2[OPMODE] • The HOME input can be used to trigger the initialization of the position counter (UPOS/LPOS). Often the HOME signal is connected to a sensor on the motor or machine, sending notification that it has reached a defined home position. In Quadrature Count mode or Single Phase Count mode, this signal can present the counter ENABLE function. • All counters start counting when ENABLE is asserted,all counters stop counting when ENABLE is de-asserted.
TRIGGER	TRIGGER	Input	• Initializes UPOS or LPOS. • Takes a snapshot of POS, REV, and POSD.
ICAP[3:1]	Input capture	Input	Takes snapshots of POS and stores them into the corresponding UPOSH/LPOSH: • Positive edge of ICAP[3] input takes snapshots of POS and stores it into UPOSH3/LPOSH3. • Positive edge of ICAP[2] input takes snapshots of POS and stores it into UPOSH2/LPOSH2.

Table continues on the next page...

Table 1087. Signals (continued)

Signal			Description
			• Positive edge of ICAP[1] input takes snapshots of POS and stores it into UPOSH1/LPOSH1.
POS_MATCH[3:0]	Position Match	Output	• Records the time at which the position of the shaft matches a user-defined compare value (UCOMPx/LCOMPx), the POS_MATCH[x](x range is 0-3) output can be used to trigger a timer channel. • Alternatively, records the time at which the position information was read, (when the position counters (UPOS and LPOS), revolution counter (REV), or position difference counter (POSD) registers are read) the POS_MATCH[x](x range is 0-3) output can be used to trigger a timer channel. Note: • If CTRL2[OUTCTL] is cleared, POS_MATCH[x](x range is 0-3) is asserted when the value of POS equals the value of the corresponding compare register (UCOMPx/LCOMPx), and de-asserted when the value of POS does not equal the value of the corresponding compare register (UCOMPx/LCOMPx). • If CTRL2[OUTCTL] is set, POS_MATCH[x](x range is 0-3) is asserted a pulse with 1 peripheral clock cycle width when UPOS or LPOS, REV, or POSD is read.
COMP_FLG[3:0]	Position Counter Compare Output	Output	• COMP_FLG[x](x range is 0-3) is asserted when the value of POS is equal to or greater than the value of the corresponding compare register (UCOMPx/LCOMPx). • COMP_FLG[x](x range is 0-3) is de-asserted when the value of POS is smaller than the value of the corresponding compare register (UCOMPx/LCOMPx).
DIR	Direction of position counting	Output	Indicates the direction of position counting: • 1 indicates up. • 0 indicates down.
CNT_UP	Position counter count up	Output	WheneQDC decodes a count-up event, CNT_UP outputs a pulse with a width of 1 peripheral clock cycle.
CNT_DN	position counter count down	Output	When eQDC decodesa count-down event, CNT_DN outputs a pulse with a width of 1 peripheral clock cycle.

73.5 Initialization

This module does not require initialization.

73.6 Register descriptions

This section describes the memory map and registers of eQDC.

73.6.1 Quadrature_Decoder register descriptions

The address of a register is the sum of a base address and an address offset.

73.6.1.1 Quadrature_Decoder memory map

eQDC1 base address: 4271_0000h

eQDC2 base address: 4272_0000h

eQDC3 base address: 4273_0000h

eQDC4 base address: 4274_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Register (CTRL)	16	RW	0000h
2h	Control 2 Register (CTRL2)	16	RW	0000h
4h	Input Filter Register (FILT)	16	RW	0000h
6h	Last Edge Time Register (LASTEDGE)	16	R	FFFFh
8h	Position Difference Period Counter Register (POSDPER)	16	R	FFFFh
Ah	Position Difference Period Buffer Register (POSDPERBFR)	16	R	FFFFh
Ch	Upper Position Counter Register (UPOS)	16	RW	0000h
Eh	Lower Position Counter Register (LPOS)	16	RW	0000h
10h	Position Difference Counter Register (POSD)	16	RW	0000h
12h	Position Difference Hold Register (POSDH)	16	R	0000h
14h	Upper Position Hold Register (UPOSH)	16	R	0000h
16h	Lower Position Hold Register (LPOSH)	16	R	0000h
18h	Last Edge Time Hold Register (LASTEDGEH)	16	R	FFFFh
1Ah	Position Difference Period Hold Register (POSDPERH)	16	R	FFFFh
1Ch	Revolution Hold Register (REVH)	16	R	0000h
1Eh	Revolution Counter Register (REV)	16	RW	0000h
20h	Upper Initialization Register (UINIT)	16	RW	0000h
22h	Lower Initialization Register (LINIT)	16	RW	0000h
24h	Upper Modulus Register (UMOD)	16	RW	0000h
26h	Lower Modulus Register (LMOD)	16	RW	0000h
28h	Upper Position Compare Register 0 (UCOMP0)	16	RW	8000h
2Ah	Lower Position Compare Register 0 (LCOMP0)	16	RW	0000h
2Ch	Upper Position Compare 1 (UCOMP1)	16	W	8000h
2Ch	Upper Position Holder Register 1 (UPOSH1)	16	R	0000h
2Eh	Lower Position Compare 1 (LCOMP1)	16	W	0000h
2Eh	Lower Position Holder Register 1 (LPOSH1)	16	R	0000h
30h	Upper Position Compare 2 (UCOMP2)	16	W	8000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
30h	Upper Position Holder Register 3 (UPOSH2)	16	R	0000h
32h	Lower Position Compare 2 (LCOMP2)	16	W	0000h
32h	Lower Position Holder Register 2 (LPOSH2)	16	R	0000h
34h	Upper Position Compare 3 (UCOMP3)	16	W	8000h
34h	Upper Position Holder Register 3 (UPOSH3)	16	R	0000h
36h	Lower Position Compare 3 (LCOMP3)	16	W	0000h
36h	Lower Position Holder Register 3 (LPOSH3)	16	R	0000h
38h	Interrupt Control Register (INTCTRL)	16	RW	0000h
3Ah	Watchdog Timeout Register (WTR)	16	RW	0000h
3Ch	Input Monitor Register (IMR)	16	RW	0000h
3Eh	Test Register (TST)	16	RW	0000h
50h	Upper VERID (UVERID)	16	R	0001h
52h	Lower VERID (LVERID)	16	R	0001h

73.6.1.2 Control Register (CTRL)

Offset

Register	Offset
CTRL	0h

Function
Controls various features of eQDC.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HIRQ	HIE	HIP	HNE	0	REV	PH1	XIRQ	XIE	XIP	XNE	WDIRQ	WDIE	WDE	DMAEN	LDOK
W	W1C				SWIP			W1C				W1C				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 HIRQ	HOME/ENABLE Signal Transition Interrupt Request HOME/ENABLE Signal Transition Interrupt Request is set when a positive or negative transition on the HOME/ENABLE signal occurs, according to the Use Negative Edge of HOME/ENABLE Input bit (CTRL[HNE]). If HOME/ENABLE Signal Transition Interrupt Request bit is set and HOME/ENABLE Interrupt Enable bit (CTRL[HIE]) is set, then a HOME/ENABLE interrupt occurs. • HOME/ENABLE Signal Transition Interrupt Request bit remains set until it is cleared by software • Write a one to this bit (HIRQ) to clear it 0b - No transition on the HOME/ENABLE signal has occurred 1b - A transition on the HOME/ENABLE signal has occurred
14 HIE	HOME/ENABLE Interrupt Enable Enables/disables HOME/ENABLE signal interrupts. 0b - Disabled 1b - Enabled
13 HIP	Enable HOME to Initialize Position Counter UPOS/LPOS Enables the position counter to be initialized by the HOME signal. 0b - No action 1b - HOME signal initializes the position counter
12 HNE	Use Negative Edge of HOME/ENABLE Input Selects using the positive/negative edge of the HOME/ENABLE input. 0b - When CTRL[OPMODE] = 0,use HOME positive edge to trigger initialization of position counters. When CTRL[OPMODE] = 1,use ENABLE high level to enable POS/POSD/WDG/REV counters 1b - When CTRL[OPMODE] = 0,use HOME negative edge to trigger initialization of position counters. When CTRL[OPMODE] = 1,use ENABLE low level to enable POS/POSD/WDG/REV counters
11 SWIP	Software-Triggered Initialization of Position Counters UPOS and LPOS Writing 1 to Software-Triggered Initialization of Position Counters bit initializes the position counter. • Software-Triggered Initialization of Position Counters bit is always read as 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No action 1b - Initialize position counter
10 REV	Enable Reverse Direction Counting Selects the direction of the count and position counter initial value. 0b - Count normally and the position counter initialization uses upper/lower initialization register UINIT/LINIT 1b - Count in the reverse direction and the position counter initialization uses upper/lower modulus register UMOD/LMOD
9 PH1	Enable Single Phase Mode Bypass/use the quadrature decoder logic. 0b - Standard quadrature decoder, where PHASEA and PHASEB represent a two-phase quadrature signal. 1b - Single phase mode, bypass the quadrature decoder, refer to CTRL2[CMODE] description
8 XIRQ	INDEX/PRESET Pulse Interrupt Request INDEX/PRESET Pulse Interrupt Request is set when a transition on the INDEX/PRESET signal occurs, according to the Use Negative Edge of INDEX/PRESET Pulse bit (CTRL[XNE]). If INDEX/PRESET Pulse Interrupt Request bit is set and INDEX/PRESET Pulse Interrupt Enable bit (CTRL[XIE]) is set, then an INDEX/PRESET interrupt occurs. • INDEX/PRESET Pulse Interrupt Request remains set until cleared by software • Write a one to this bit (XIRQ) to clear it 0b - INDEX/PRESET pulse has not occurred 1b - INDEX/PRESET pulse has occurred
7 XIE	INDEX/PRESET Pulse Interrupt Enable Enables/disables INDEX/PRESET pulse interrupts. 0b - Disabled 1b - Enabled
6 XIP	INDEX Triggered Initialization of Position Counters UPOS and LPOS Enables/disables the position counter to be initialized by the INDEX pulse. 0b - INDEX pulse does not initialize the position counter 1b - INDEX pulse initializes the position counter
5 XNE	Select Positive/Negative Edge of INDEX/PRESET Pulse Selects the positive/negative edge of the INDEX/PRESET pulse. 0b - Use positive edge of INDEX/PRESET pulse

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Use negative edge of INDEX/PRESET pulse
4 WDIRQ	Watchdog Timeout Interrupt Request Watchdog Timeout Interrupt Request is set when a watchdog timeout interrupt occurs. • Watchdog Timeout Interrupt Request remains set until cleared by software • Write a one to this bit (WDIRQ) to clear it • Watchdog Timeout Interrupt Request bit is also cleared when Watchdog Enable (CTRL[WDE]) is disabled (=0) 0b - No Watchdog timeout interrupt has occurred 1b - Watchdog timeout interrupt has occurred
3 WDIE	Watchdog Timeout Interrupt Enable Enables/disables watchdog timeout interrupts. 0b - Disabled 1b - Enabled
2 WDE	Watchdog Enable Enables/disables the watchdog timer that is monitoring the PHASEA and PHASEB inputs for motor movement. 0b - Disabled 1b - Enabled
1 DMAEN	DMA Enable This bit enables DMA for new written buffer values of COMPx/INIT/MOD(x range is 0-3) Please note: When set to 1, this bit can't be cleared by writing 0 to it. 0b - DMA is disabled 1b - DMA is enabled. DMA request asserts automatically when the values in the outer-set of buffered compare registers (UCOMP0/LCOMP0;UCOMP1/LCOMP1;UCOMP2/LCOMP2;UCOMP3/LCOMP3), initial registers(UINIT/LINIT) and modulus registers (UMOD/LMOD) are loaded into the inner-set of buffer and then LDOK is cleared automatically. After the completion of this DMA transfer, LDOK is set automatically, it ensures outer-set values can be loaded into inner-set which in turn triggers DMA again.
0 LDOK	Load Okay This bit enables that the outer-set values of buffered compare registers (UCOMPx/LCOMPx, x=0~3), initial register(UINIT/LINIT) and modulus register(UMOD/LMOD) can be loaded into their inner-sets and take effect. When LDOK is set, this loading action occurs at the next position counter roll-over or roll-under if CTRL2[LDMOD] is set, or it occurs immediately if CTRL2[LDMOD] is cleared. LDOK is automatically cleared after the values in outer-set is loaded into the inner-set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Set LDOK bit by reading it, then write a logic 1 to it. LDOK can be manually cleared before a loading action occurs by writing a logic 0 to it. The outer-set of the registers cannot be written while LDOK is 1. 0b - No loading action taken. Users can write new values to buffered registers (writing into outer-set of these buffered registers) 1b - Outer-set values are ready to be loaded into inner-set and take effect. The loading time point depends on CTRL2[LDMOD].

73.6.1.3 Control 2 Register (CTRL2)

Offset

Register	Offset
CTRL2	2h

Function

Controls various features of eQDC.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMODE		ONCE	INITPOS	0	PMEN	OUTCTL	REVMOD	0				LDMOD	OPMODE	UPDPOS	UPDHLD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 CMODE	Counting Mode These bits control the basic counting and behavior of Position Counter and Position Difference Counter. Setting CTRL[REV] to 1 can reverse the counting direction. In quadrature Mode (CTRL[PH1] = 0): 00b - CM0: Normal/Reverse Quadrature X4 01b - CM1: Normal/Reverse Quadrature X2 10b - CM2: Normal/Reverse Quadrature X1 11b - CM3: Reserved In Single Phase Mode (CTRL[PH1] = 1): 00b - CM0: UP/DOWN Pulse Count Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - CM1: Signed Mode, count PHASEA rising/falling edge, position counter counts up when PHASEB is low and counts down when PHASEB is high 10b - CM2: Signed Count Mode,count PHASEA rising edge only, position counter counts up when PHASEB is low and counts down when PHASEB is high 11b - CM3: Reserved
13 ONCE	Count Once This bit selects modulo loop or one shot counting mode. **NOTE** Any of the following event initializes and restarts the position counter: • Write to CTRL[SWIP]. • INDEX signal transition if CTRL2[OPMODE] bit is clear and CTRL[XIP] is set. • HOME signal transition if CTRL2[OPMODE] bit is clear and CTRL[HIP] is set. • PRESET signal transition if CTRL2[OPMODE] bit is set and ENABLE signal is 1. • Positive edge transition of TRIGGER input if CTRL2[INITPOS] is set. 0b - Position counter counts repeatedly 1b - Position counter counts until roll-over or roll-under, then stop.
12 INITPOS	Initial Position Register • When Initial Position Register is set (=1), it allows the rising edge of TRIGGER input to initialize the position counter (UPOS and LPOS registers) to UINIT/LINIT values (if CTRL[REV]=0) or UMOD/LMOD values (if CTRL[REV]=1). • When Initial Position Register is clear (=0), the position counter (UPOS/LPOS) is not initialized on rising edge of TRIGGER input 0b - Don't initialize position counter on rising edge of TRIGGER 1b - Initialize position counter on rising edge of TRIGGER
11 —	Reserved
10 PMEN	Period measurement function enable This bit is used to indicate the period measurement functions of LASTEDGE, LASTEDGEH, POSDPER, POSDPERBFR, and POSDPERH are being used. 0b - Period measurement functions are not used. POSD is loaded to POSDH and then cleared whenever POSD, UPOS, LPOS or REV is read. 1b - Period measurement functions are used. POSD is loaded into POSDH and then cleared only when POSD is read.
9	Output Control

Table continues on the next page...

Table continued from the previous page...

Field	Function
OUTCTL	Output Control controls the pulsing of the POS_MATCH output signal. 0b - POS_MATCH[x](x range is 0-3) is asserted when the Position Counter is equal to according compare value (UCOMPx/LCOMPx)(x range is 0-3), and de-asserted when the Position Counter not equal to the compare value (UCOMPx/LCOMPx)(x range is 0-3) 1b - All POS_MATCH[x](x range is 0-3) are asserted a pulse, when the UPOS, LPOS, REV, or POSD registers are read
8 REVMOD	Revolution Counter Modulus Enable Revolution Counter Modulus Enable selects how the revolution counter (REV) is incremented/ decremented. By default, the revolution counter is controlled based on the count direction and the INDEX pulse. As an option, the revolution counter can be controlled using the roll-over/under detection during modulo counting. 0b - Use INDEX pulse to increment/decrement revolution counter (REV) 1b - Use modulus counting roll-over/under to increment/decrement revolution counter (REV)
7-4 —	Reserved
3 LDMOD	Buffered Register Load (Update) Mode Select This bit selects the loading time point of the buffered compare registers UCOMPx/LCOMPx, x=0~3, initial register (UINIT/LINIT), and modulus register (UMOD/LMOD) 0b - Buffered registers are loaded and take effect immediately upon CTRL[LDOK] is set. 1b - Buffered registers are loaded and take effect at the next roll-over or roll-under if CTRL[LDOK] is set.
2 OPMODE	Operation Mode Select 0b - Decode Mode: Input nodes INDEX/PRESET and HOME/ENABLE are assigned to function of INDEX and HOME. 1b - Count Mode: Input nodes INDEX/PRESET and HOME/ENABLE are assigned to functions of PRESET and ENABLE. In this mode: (1)only when ENABLE=1, all counters (position/ position difference/revolution/watchdog) can run, when ENABLE=0, all counters (position/position difference/revolution/watchdog) can't run. (2) the rising edge of PRESET input can initialize position/revolution/watchdog counters (position counter initialization also need referring to bit CTRL[REV]).
1 UPDPOS	Update Position Registers • When UPDPOS bit is set (=1), it allows the rising edge of TRIGGER input to clear the POSD, REV, UPOS and LPOS registers to zero. • When UPDPOS bit is clear (=0), the rising edge of TRIGGER input does not clear POSD, REV, UPOS and LPOS registers to zero. • If both UPDPOS bit and INITPOS bit are set, then when rising edge of TRIGGER input occurs, the UPOS/LPOS registers are not cleared to zero, but be initialized to value of UINIT/LINIT registers(if CTRL[REV]=0) or value of UMOD/LMOD registers(if CTRL[REV]=1).

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>• Do not set this bit when using the LASTEDGE and POSDPER registers to measure speed.</td></tr>
<tr><td>0
UPDHLD</td><td>Update Hold Registers
• When UPDHLD bit is set (=1), it allows the rising edge of TRIGGER input to cause an update of the POSDH, REVH, UPOSH, and LPOSH registers
• When UPDHLD bit is clear (=0), the hold registers (POSDH, REVH) are not updated by the rising edge of TRIGGER input, and hold registers UPOSH/LPOSH are not updated by the rising edge of TRIGGER input if INITPOS bit is also clear.
• Do not set this bit when using the LASTEDGE and POSDPER registers to measure speed.
UPOSH/LPOSH has little difference with POSDH and REVH on UPDHLD bit, both UPDHLD bit and INITPOS bit allow the rising edge of TRIGGER input to take effect on UPOSH/LPOSH, so make a list here:
<table>
<tr><th>INITPOS</th><th>UPDHLD</th><th>State of UPOSH/LPOSH</th></tr>
<tr><td>0</td><td>0</td><td>Rising edge of TRIGGER input does not take effect on UPOSH/LPOSH</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge of TRIGGER input updates UPOSH/LPOSH with the value of POS counter</td></tr>
<tr><td>1</td><td>0</td><td>Rising edge of TRIGGER input updates UPOSH/LPOSH with the value of POS counter, and then POS counter is initialized</td></tr>
<tr><td>1</td><td>1</td><td>Rising edge of TRIGGER input updates UPOSH/LPOSH with the value of POS counter, and then POS counter is initialized</td></tr>
</table>
Note:Updating the Position Difference Hold Register (POSDH) also causes the Position Difference Counter Register (POSD) to be cleared.</td></tr>
</table>

73.6.1.4 Input Filter Register (FILT)

Offset

Register	Offset
FILT	4h

Function

The Input Filter Register sets the values of the input filter sample period (FILT_PER) and the input filter sample count (FILT_CNT).

- The Input Filter Sample Period (FILT_PER) should be set so that the sampling period is larger than the period of the expected noise. Doing this means that a noise spike only corrupts one sample.
- The Input Filter Sample Count (FILT_CNT) should be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of FILT_CNT+3.

The values of the Input Filter Sample Period (FILT_PER) and the Input Filter Sample Count (FILT_CNT) must also be traded off against the desire for minimal latency in recognizing valid input transitions. Turning on the input filter (setting the Input Filter Sample Period (FILT_PER) to a non-zero value) introduces a latency of ((FILT_CNT+3)*FILT_PER+2) IPBus clock periods.

The filter latency can be measured:

- Drive the quadrature decoder inputs (PHASEA, PHASEB, INDEX, HOME)
- Monitor the filtered output in the Input Monitor Register (IMR)
- Determine how many IPBus clock cycles that it takes before the output shows up, by using the following equations, where f is FILT_PER and s is FILT_CNT.
  - DELAY (IPBus clock cycles) = f * (s+3) +1 (to read the filtered output)
  - DELAY (IPBus clock cycles) = f * (s+3) +2 (to monitor the output in the IMR)

One more additional IPBus clock cycle is needed to read the filtered output, and 2 more IPBus clock cycles are needed to monitor the filtered output in the IMR. The sample rate is set when it reaches the number f. The following examples use the preceding equations:

- Example: when f = 0, the filter is bypassed: DELAY = 1 or 2 clock cycles.
- Example: when f = 5 and s = 2: DELAY = 5 * (2+3) + (1 or 2) = 26 or 27 clock cycles.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PRSC				FILT_CS	FILT_CNT			FILT_PER							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-12 PRSC	Prescaler The Prescaler field is used to prescale the peripheral clock that is used by the LASTEDGE and POSDPER counters as well as watchdog timer and Input Filter. The clock is prescaled by a value of 2^PRSC which means that the prescaler logic can divide the clock by a minimum of 1 and a maximum of 32,768.
11 FILT_CS	Filter Clock Source selection 0b - Peripheral Clock 1b - Prescaled peripheral clock by PRSC
10-8 FILT_CNT	Input Filter Sample Count The Input Filter Sample Count represents the number of consecutive samples that must agree, before the input filter accepts an input transition.

Table continues on the next page...

Table continued from the previous page...

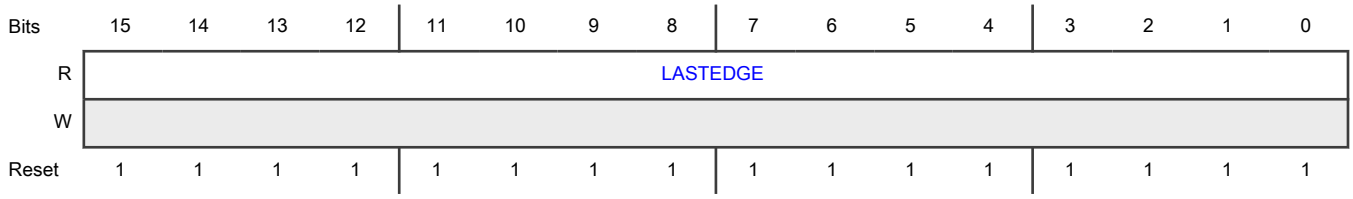
Field	Function
	• A value of 0x0 represents 3 samples • A value of 0x7 represents 10 samples The value of the Input Filter Sample Count (FILT_CNT) affects the input latency.
7-0 FILT_PER	Input Filter Sample Period The Input Filter Sample Period represents the sampling period (in IPBus clock cycles) of the decoder input signals. Each input is sampled multiple times at the rate specified by the Input Filter Sample Period. • If FILT_PER is 0x00 (default), then the input filter is bypassed. Bypassing the digital filter enables the position/position difference counters to operate with count rates up to the IPBus frequency. • The value of the Input Filter Sample Period (FILT_PER) affects the input latency. • When changing the Input Filter Sample Period (FILT_PER) from a non-zero value to another non-zero value, write a value of 0 first, to clear the filter.

73.6.1.5 Last Edge Time Register (LASTEDGE)

Offset

Register	Offset
LASTEDGE	6h

Diagram



Fields

Field	Function
15-0 LASTEDGE	Last Edge Time Counter The Last Edge Time counter represents the time since the last edge occurred on PHASEA or PHASEB. LASTEDGE counts up using the peripheral clock that is prescaled by FILT[PRSC]. Any edge on PHASEA or PHASEB resets this register to 0 and starts counting. If the LASTEDGE count reaches 0xffff, the counting stops in order to prevent an overflow. Counting continues when an edge occurs on PHASEA or PHASEB.

73.6.1.6 Position Difference Period Counter Register (POSDPER)

Offset

Register	Offset
POSDPER	8h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POSDPER															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
15-0 POSDPER	Position difference period The Position Difference Period counter counts up using the peripheral clock that is prescaled by FILT[PRSC]. Reading the POSD register also causes the value of the LASTEDGE register to be loaded into POSDPER. If the POSDPER count reaches 0xffff, the counting stops in order to prevent an overflow. Counting continues when an edge occurs on PHASEA or PHASEB.

73.6.1.7 Position Difference Period Buffer Register (POSDPERBFR)

Offset

Register	Offset
POSDPERBFR	Ah

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POSDPERBFR															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
15-0 POSDPERBFR	Position difference period buffer The Position Difference Period buffer register is a buffer register for the POSDPER value. POSDPERBFR is updated with the value of POSDPER when any edge occurs on PHASEA or PHASEB.

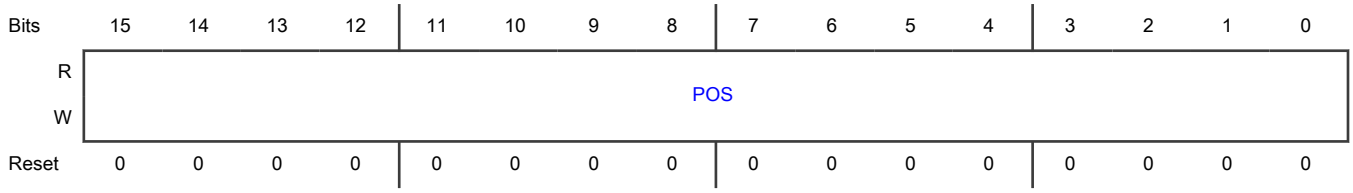
73.6.1.8 Upper Position Counter Register (UPOS)

Offset

Register	Offset
UPOS	Ch

Function
Contains the upper (most significant) half of the Position Counter. This is the binary count from the Position Counter.

Diagram



Fields

Field	Function
15-0 POS	POS The upper (most significant) half of the Position Counter

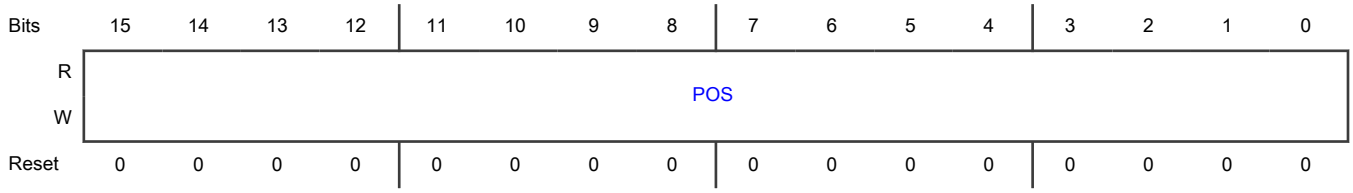
73.6.1.9 Lower Position Counter Register (LPOS)

Offset

Register	Offset
LPOS	Eh

Function
Contains the lower (least significant) half of the Position Counter. This is the binary count from the Position Counter.

Diagram



Fields

Field	Function
15-0 POS	POS The lower (least significant) half of the Position Counter

73.6.1.10 Position Difference Counter Register (POSD)

Offset

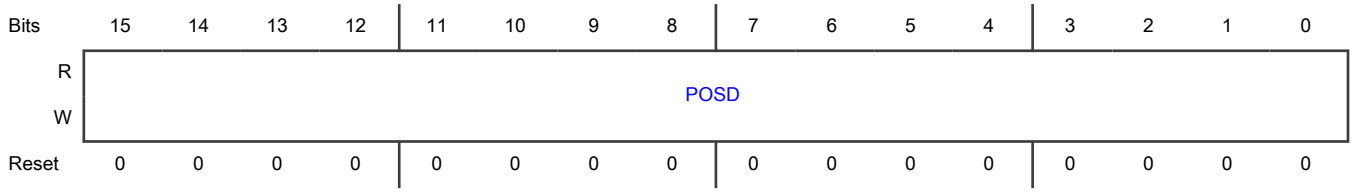
Register	Offset
POSD	10h

Function

The Position Difference Counter Register contains the position change in value occurring between each read of the Position Register. The value of the Position Difference Counter Register can be used to calculate velocity.

- The 16-bit Position Difference Counter computes up or down on every count pulse.
- This Position Difference Counter acts as a differentiator, whose count value is proportional to the change in position since the last time that the Position Counter was read.
- When the Position Difference Counter (POSD) is read, its contents are copied into the Position Difference Hold Register (POSDH) and the Position Difference Counter is cleared.

Diagram



Fields

Field	Function
15-0 POSD	POSD The position change in value between each read of the Position Register.

73.6.1.11 Position Difference Hold Register (POSDH)

Offset

Register	Offset
POSDH	12h

Function

The Position Difference Hold Register register contains a snapshot of the value of the Position Difference Counter Register (POSD). The value of the Position Difference Hold Register (POSDH) can be used to calculate velocity.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POSDH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 POSDH	POSDH The value of the POSD register

73.6.1.12 Upper Position Hold Register (UPOSH)

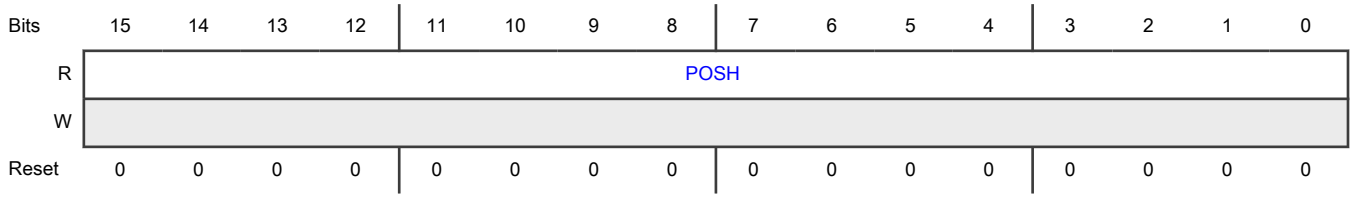
Offset

Register	Offset
UPOSH	14h

Function

Contains a snapshot of the Upper Position Counter Register (UPOS register).

Diagram



Fields

Field	Function
15-0 POSH	POSH The value of the Upper Position Counter Hold Register (UPOSH)

73.6.1.13 Lower Position Hold Register (LPOSH)

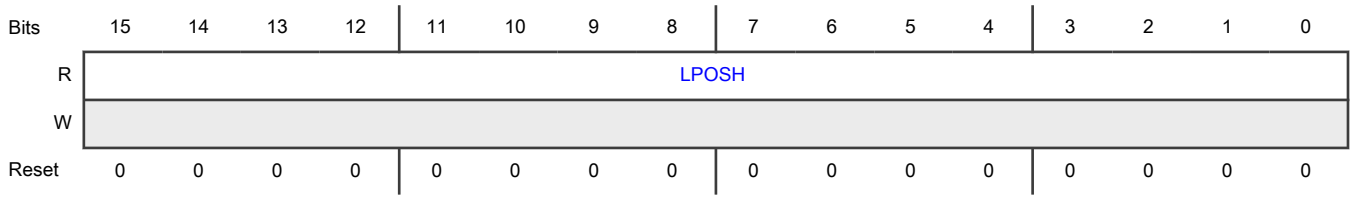
Offset

Register	Offset
LPOSH	16h

Function

Contains a snapshot of the Lower Position Counter Register (LPOS).

Diagram



Fields

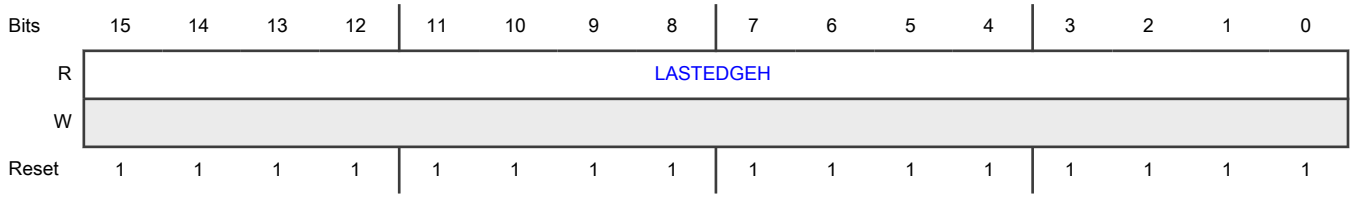
Field	Function
15-0 LPOSH	POSH The value of the Lower Position Counter Hold Register (LPOSH)

73.6.1.14 Last Edge Time Hold Register (LASTEDGEH)

Offset

Register	Offset
LASTEDGEH	18h

Diagram



Fields

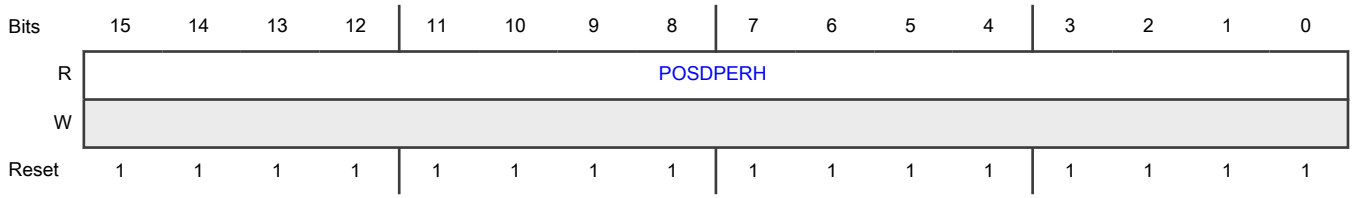
Field	Function
15-0 LASTEDGEH	Last Edge Time Hold The Last Edge Time Hold register is a hold register for the LASTEDGE value. LASTEDGEH is updated with the value of LASTEDGE when the POSD register is read.

73.6.1.15 Position Difference Period Hold Register (POSDPERH)

Offset

Register	Offset
POSDPERH	1Ah

Diagram



Fields

Field	Function
15-0 POSDPERH	Position difference period hold The Position Difference Period Hold register is a hold register for the POSDPER value. POSDPERH is updated with the value of POSDPERBFR when the POSD register is read.

73.6.1.16 Revolution Hold Register (REVH)

Offset

Register	Offset
REVH	1Ch

Function

Contains a snapshot of the value of the Revolution Counter Register (REVH).

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REVH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 REVH	REVH The value of the Revolution Counter Register (REV)

73.6.1.17 Revolution Counter Register (REV)

Offset

Register	Offset
REV	1Eh

Function

Contains the current value of the Revolution Counter.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REV															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 REV	REV The current value of the Revolution Counter

73.6.1.18 Upper Initialization Register (UINIT)

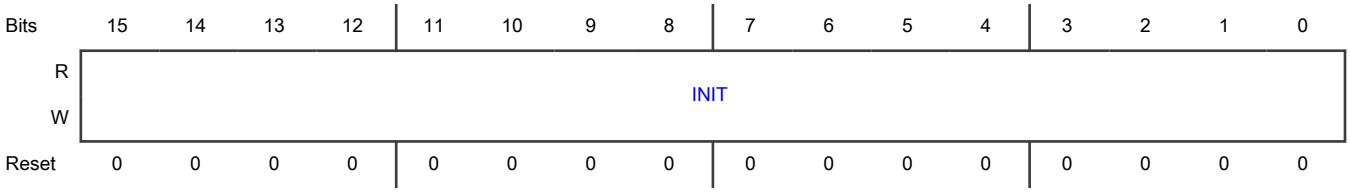
Offset

Register	Offset
UINIT	20h

Function

Contains the value to be used to initialize the upper half of the position counter (UPOS). It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram



Fields

Field	Function
15-0 INIT	INIT The value to be used to initialize the upper half of the position counter (UPOS).

73.6.1.19 Lower Initialization Register (LINIT)

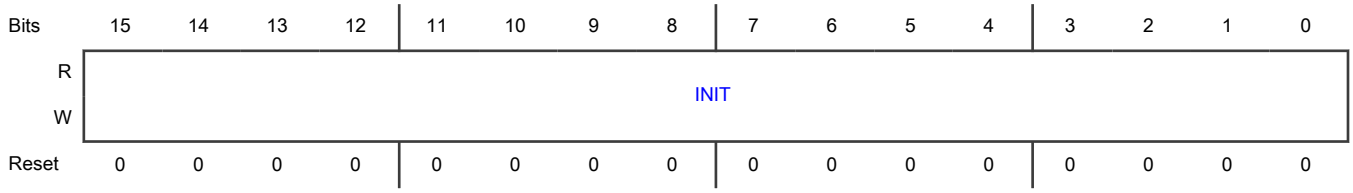
Offset

Register	Offset
LINIT	22h

Function

Contains the value to be used to initialize the lower half of the position counter (LPOS). It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram



Fields

Field	Function
15-0 INIT	INIT The value to be used to initialize the lower half of the position counter (LPOS)

73.6.1.20 Upper Modulus Register (UMOD)

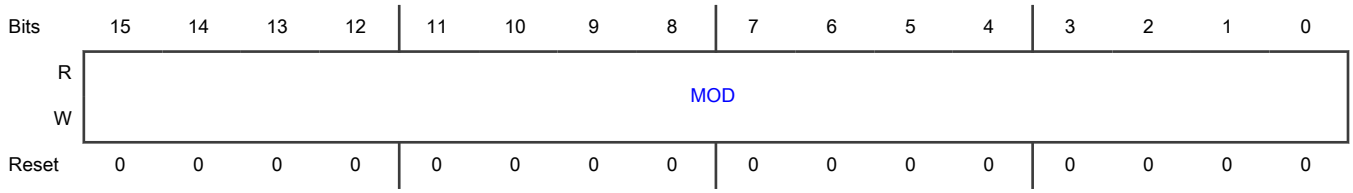
Offset

Register	Offset
UMOD	24h

Function

The Upper Modulus Register contains the upper (most significant) half of the Modulus register. MOD acts as the upper bound during modulo counting, and as the upper reload value when rolling over from the lower bound. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram



Fields

Field	Function
15-0 MOD	MOD Upper (most significant) half of the Modulus register

73.6.1.21 Lower Modulus Register (LMOD)

Offset

Register	Offset
LMOD	26h

Function

The Lower Modulus Register contains the lower (least significant) half of the Modulus register. MOD acts as the upper bound during modulo counting, and as the upper reload value when rolling over from the lower bound. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MOD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 MOD	MOD Lower (least significant) half of the Modulus register

73.6.1.22 Upper Position Compare Register 0 (UCOMP0)

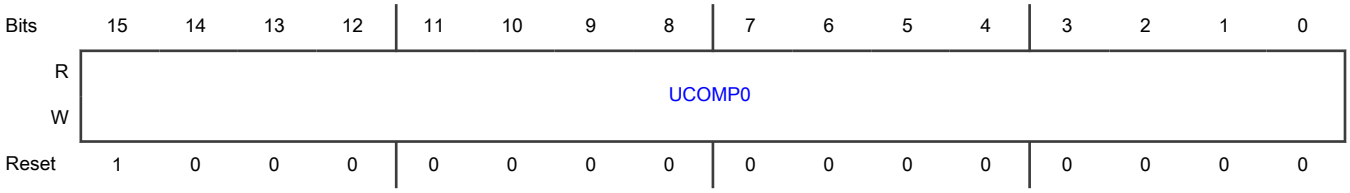
Offset

Register	Offset
UCOMP0	28h

Function

The Upper Position Compare Register contains the upper (most significant) half of the Position Compare register 0. When the value of the Position counter (POS) matches the value of the Position Compare register 0(COMP0), the Compare Interrupt Request flag (INTCTRL[CMP0IRQ]) is set and the POS_MATCH[0] output is asserted. COMP_FLG[0] is asserted when the Position Counter is equal or greater than COMP0. COMP_FLG[0] is de-asserted when the Position Counter is less than COMP0. Note: If COMP0 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[0] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram



Fields

Field	Function
15-0 UCOMP0	UCOMP0 Upper (most significant) half of the Position Compare register 0

73.6.1.23 Lower Position Compare Register 0 (LCOMP0)

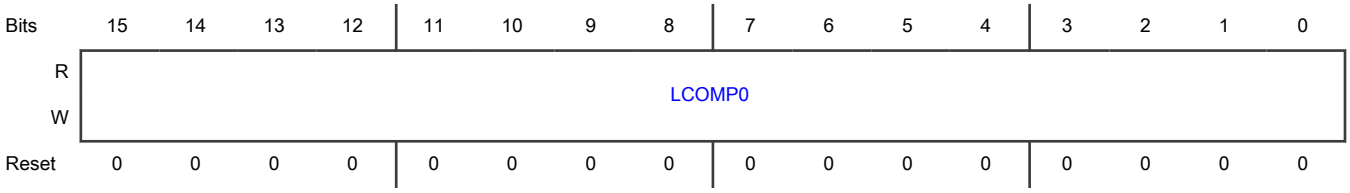
Offset

Register	Offset
LCOMP0	2Ah

Function

The Lower Position Compare Register contains the lower (least significant) half of the Position Compare register 0. When the value of the Position counter (POS) matches the value of the Position Compare register 0(COMP0), the Compare Interrupt Request flag (INTCTRL[CMP0IRQ]) is set and the POS_MATCH[0] output is asserted. COMP_FLG[0] is asserted when the Position Counter is equal or greater than COMP0. COMP_FLG[0] is de-asserted when the Position Counter is less than COMP0. Note: If COMP0 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[0] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram



Fields

Field	Function
15-0 LCOMP0	LCOMP0 Lower (least significant) half of the Position Compare register 0

73.6.1.24 Upper Position Compare 1 (UCOMP1)

Offset

Register	Offset
UCOMP1	2Ch

Function

The Upper Position Compare Register contains the upper (most significant) half of the Position Compare register 1. When the value of the Position counter (POS) matches the value of the Position Compare register 1 (COMP1), the Compare Interrupt Request flag (INTCTRL[CMP1IRQ]) is set and the POS_MATCH[1] output is asserted. COMP_FLG[1] is asserted when the Position Counter is equal or greater than COMP1. COMP_FLG[1] is de-asserted when the Position Counter is less than COMP1. Note: If COMP1 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[1] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	UCOMP1															
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 UCOMP1	UCOMP1 When write, it writes Upper (most significant) half of the Position Compare register 1.

73.6.1.25 Upper Position Holder Register 1 (UPOSH1)

Offset

Register	Offset
UPOSH1	2Ch

Function

UPOSH1 share the same address with UCOMP1.When read,this register means the value of the Upper Position Counter Hold Register 1(UPOSH1),which is the upper 16 bits of POSH1.Position counter is captured into POSH1 on the rising edge of ICAP[1].

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	UPOSH1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 UPOSH1	UPOSH1 When read, it means the value of the Upper Position Hold Counter Register 1(UPOSH1)

73.6.1.26 Lower Position Compare 1 (LCOMP1)

Offset

Register	Offset
LCOMP1	2Eh

Function

The Lower Position Compare Register contains the upper (most significant) half of the Position Compare register 1. When the value of the Position counter (POS) matches the value of the Position Compare register 1 (COMP1), the Compare Interrupt Request flag (INTCTRL[CMP1IRQ]) is set and the POS_MATCH[1] output is asserted. COMP_FLG[1] is asserted when the Position Counter is equal or greater than COMP1. COMP_FLG[1] is de-asserted when the Position Counter is less than COMP1. Note: If COMP1 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[1] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	LCOMP1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 LCOMP1	LCOMP1 When write, it writes Lower (most significant) half of the Position Compare register 1.

73.6.1.27 Lower Position Holder Register 1 (LPOSH1)

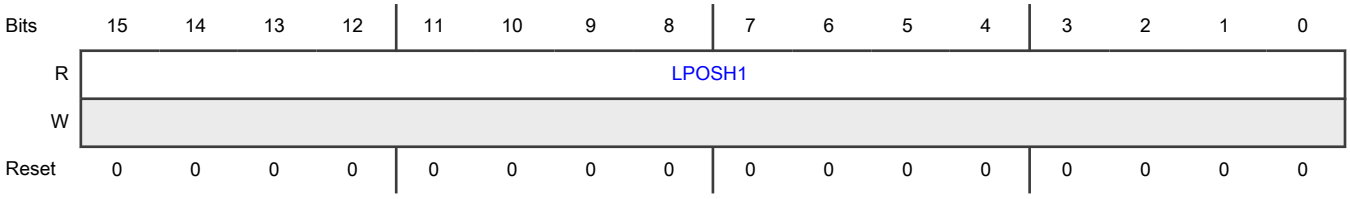
Offset

Register	Offset
LPOSH1	2Eh

Function

LPOSH1 share the same address with LCOMP1.When read, this register means the value of the Lower Position Counter Hold Register 1(LPOSH1),which is the lower 16 bits of POSH1.Position counter is captured into POSH1 on the rising edge of ICAP[1].

Diagram



Fields

Field	Function
15-0 LPOSH1	LPOSH1 When read, it means the value of the Lower Position Counter Hold Register 1(LPOSH1)

73.6.1.28 Upper Position Compare 2 (UCOMP2)

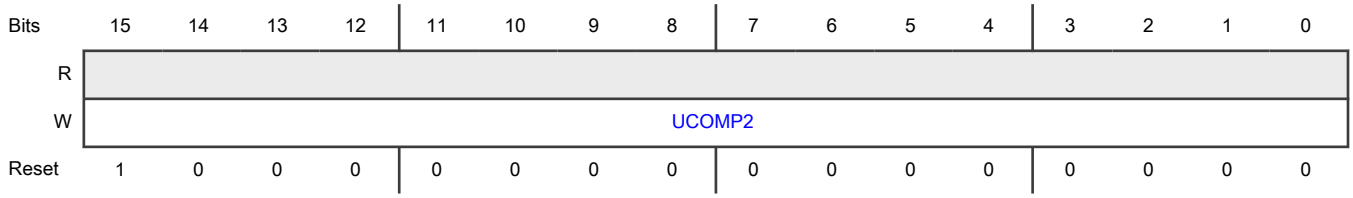
Offset

Register	Offset
UCOMP2	30h

Function

The Upper Position Compare Register contains the upper (most significant) half of the Position Compare register 2. When the value of the Position counter (POS) matches the value of the Position Compare register 2 (COMP2), the Compare Interrupt Request flag (INTCTRL[CMP2IRQ]) is set and the POS_MATCH[2] output is asserted. COMP_FLG[2] is asserted when the Position Counter is equal or greater than COMP2. COMP_FLG[2] is de-asserted when the Position Counter is less than COMP2. Note: If COMP2 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[2] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram



Fields

Field	Function
15-0 UCOMP2	UCOMP2 When write, it writes Upper (most significant) half of the Position Compare register 2.

73.6.1.29 Upper Position Holder Register 3 (UPOSH2)

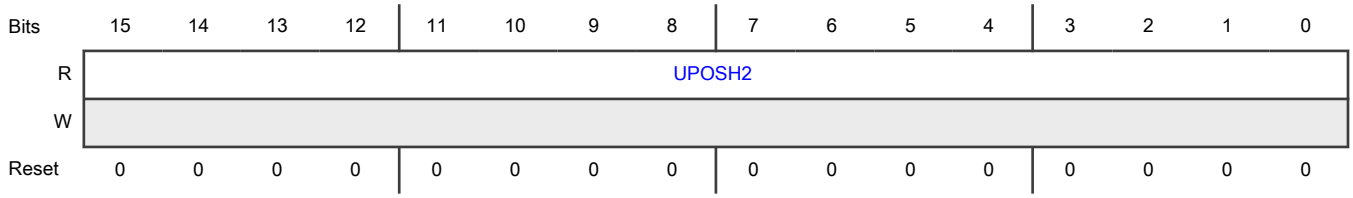
Offset

Register	Offset
UPOSH2	30h

Function

UPOSH2 share the same address with UCOMP2.When read,this register means the value of the Upper Position Counter Hold Register 2(UPOSH2),which is the upper 16 bits of POSH2.Position counter is captured into POSH2 on the rising edge of ICAP[2].

Diagram



Fields

Field	Function
15-0 UPOSH2	UPOSH2 When read, it means the value of the Upper Position Counter Hold Register 2(UPOSH2)

73.6.1.30 Lower Position Compare 2 (LCOMP2)

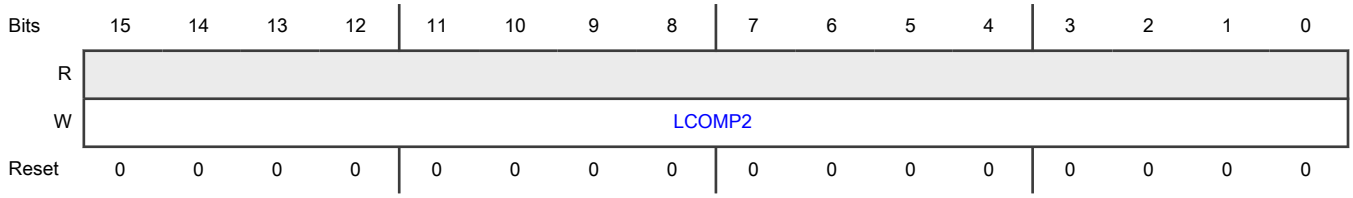
Offset

Register	Offset
LCOMP2	32h

Function

The Lower Position Compare Register contains the Lower (most significant) half of the Position Compare register 2. When the value of the Position counter (POS) matches the value of the Position Compare register 2 (COMP2), the Compare Interrupt Request flag (INTCTRL[CMP2IRQ]) is set and the POS_MATCH[2] output is asserted. COMP_FLG[2] is asserted when the Position Counter is equal or greater than COMP2. COMP_FLG[2] is de-asserted when the Position Counter is less than COMP2. Note: If COMP2 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[2] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram



Fields

Field	Function
15-0 LCOMP2	LCOMP2 When write, it writes Lower (most significant) half of the Position Compare register 2.

73.6.1.31 Lower Position Holder Register 2 (LPOSH2)

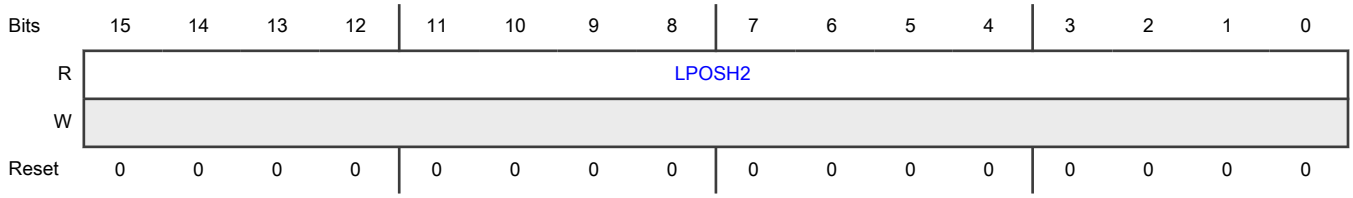
Offset

Register	Offset
LPOSH2	32h

Function

LPOSH2 share the same address with LCOMP2.When read,this register means the value of the Lower Position Counter Hold Register 2(LPOSH2),which is the lower 16 bits of POSH2.Position counter is captured into POSH2 on the rising edge of ICAP[2].

Diagram



Fields

Field	Function
15-0 LPOSH2	LPOSH2 When read, it means the value of the Lower Position Counter Hold Register 2(LPOSH2)

73.6.1.32 Upper Position Compare 3 (UCOMP3)

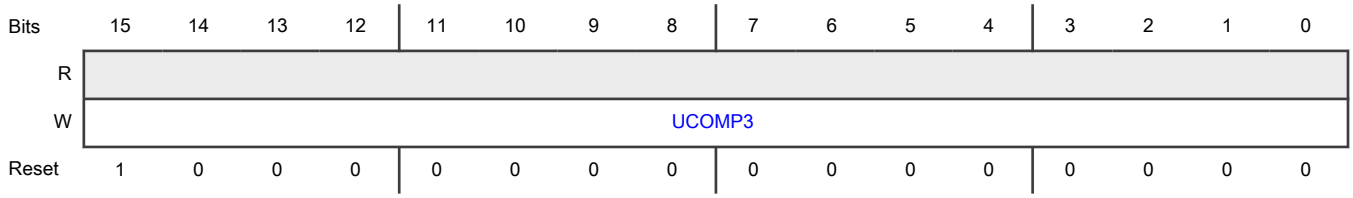
Offset

Register	Offset
UCOMP3	34h

Function

The Upper Position Compare Register contains the upper (most significant) half of the Position Compare register 3. When the value of the Position counter (POS) matches the value of the Position Compare register 3 (COMP3), the Compare Interrupt Request flag (INTCTRL[CMP3IRQ]) is set and the POS_MATCH[3] output is asserted. COMP_FLG[3] is asserted when the Position Counter is equal or greater than COMP3. COMP_FLG[3] is de-asserted when the Position Counter is less than COMP3. Note: If COMP3 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[3] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram



Fields

Field	Function
15-0 UCOMP3	UCOMP3 When write, it writes Upper (most significant) half of the Position Compare register 3.

73.6.1.33 Upper Position Holder Register 3 (UPOSH3)

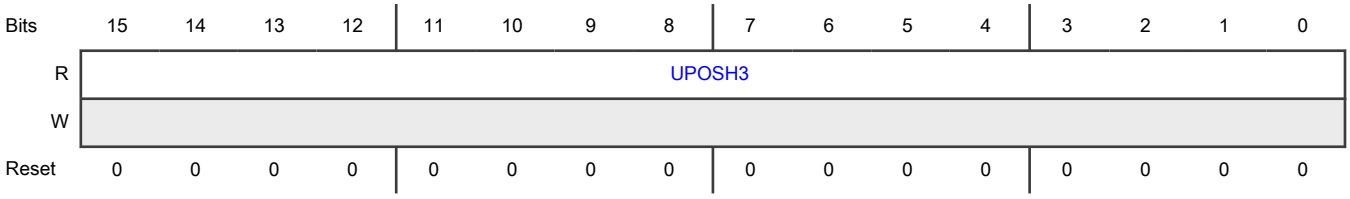
Offset

Register	Offset
UPOSH3	34h

Function

UPOSH3 share the same address with UCOMP3.When read,this register means the value of the Upper Position Counter Hold Register 3(UPOSH3),which is the upper 16 bits of POSH3.Position counter is captured into POSH3 on the rising edge of ICAP[3].

Diagram



Fields

Field	Function
15-0 UPOSH3	UPOSH3 When read, it means the value of the Upper Position Counter Hold Register 3(UPOSH3)

73.6.1.34 Lower Position Compare 3 (LCOMP3)

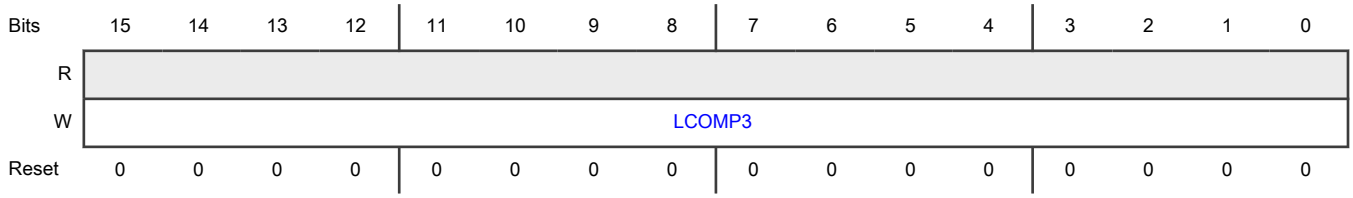
Offset

Register	Offset
LCOMP3	36h

Function

The Lower Position Compare Register contains the Lower (most significant) half of the Position Compare register 3. When the value of the Position counter (POS) matches the value of the Position Compare register 3 (COMP3), the Compare Interrupt Request flag (INTCTRL[CMP3IRQ]) is set and the POS_MATCH[3] output is asserted. COMP_FLG[3] is asserted when the Position Counter is equal or greater than COMP3. COMP_FLG[3] is de-asserted when the Position Counter is less than COMP3. Note: If COMP3 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[3] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram



Fields

Field	Function
15-0 LCOMP3	LCOMP3 When write, it writes Lower (most significant) half of the Position Compare register 3.

73.6.1.35 Lower Position Holder Register 3 (LPOSH3)

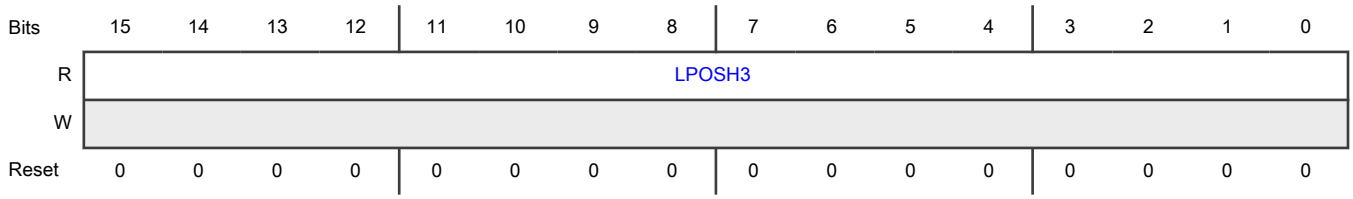
Offset

Register	Offset
LPOSH3	36h

Function

When read,this register means the value of the Lower Position Counter Register 3(LPOS)

Diagram



Fields

Field	Function
15-0 LPOSH3	LPOSH3 LPOSH3 share the same address with LCOMP3.When read,it means the value of the Lower Position Counter Register 3(LPOS3),which is the lower 16 bits of POSH3.Position counter is captured into POSH3 on the rising edge of ICAP[3].

73.6.1.36 Interrupt Control Register (INTCTRL)

Offset

Register	Offset
INTCTRL	38h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMP3IRQ	Reserved	CMP2IRQ	Reserved	CMP1IRQ	Reserved	CMP0IRQ	Reserved	ROIRQ	ROIE	RUIRQ	RUIE	DIRIRQ	DIRIE	SABIRQ	SABIE
W	W1C		W1C		W1C		W1C		W1C		W1C		W1C		W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 CMP3IRQ	Compare3 Interrupt Request Compare3 Interrupt Request is set when the position counter matches the COMP3 value. • Compare3 Interrupt Request remains set until cleared by software • Write 1 to this bit (CMP3IRQ) to clear it 0b - No match has occurred (the position counter does not match the COMP3 value) 1b - COMP3 match has occurred (the position counter matches the COMP3 value)
14 —	Reserved
13 CMP2IRQ	Compare2 Interrupt Request Compare2 Interrupt Request is set when the position counter matches the COMP2 value. • Compare2 Interrupt Request remains set until cleared by software • Write 1 to this bit (CMP2IRQ) to clear it 0b - No match has occurred (the position counter does not match the COMP2 value) 1b - COMP2 match has occurred (the position counter matches the COMP2 value)
12 —	Reserved
11 CMP1IRQ	Compare1 Interrupt Request Compare1 Interrupt Request is set when the position counter matches the COMP1 value. • Compare Interrupt Request remains set until cleared by software

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Write 1 to this bit (CMP1IRQ) to clear it 0b - No match has occurred (the position counter does not match the COMP1 value) 1b - COMP1 match has occurred (the position counter matches the COMP1 value)
10 —	Reserved
9 CMP0IRQ	Compare 0 Interrupt Request Compare 0 Interrupt Request is set when the position counter matches the COMP0 value. • Compare Interrupt Request remains set until cleared by software • Write 1 to this bit (CMP0IRQ) to clear it 0b - No match has occurred (the position counter does not match the COMP0 value) 1b - COMP match has occurred (the position counter matches the COMP0 value)
8 —	Reserved
7 ROIRQ	Roll-over Interrupt Request Roll-over Interrupt Request bit is set (=1) when the position counter (POS) value is greater than the modulus value, which means roll-over. • Roll-over Interrupt Request remains set until cleared by software • Write a one to this bit (ROIRQ) to clear it 0b - No roll-over has occurred 1b - Roll-over has occurred
6 ROIE	Roll-over Interrupt Enable Roll-over Interrupt Enable read/write bit enables roll-over interrupts, based on Roll-under Interrupt Request (INTCTRL[ROIRQ]) being set. 0b - Disabled 1b - Enabled
5 RUIRQ	Roll-under Interrupt Request Roll-under Interrupt Request bit is set (=1) when the position counter (POS) value less than initial value, which means roll-under • Roll-under Interrupt Request remains set until cleared by software • Write a one to this bit (RUIRQ) to clear it 0b - No roll-under has occurred 1b - Roll-under has occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 RUIE	Roll-under Interrupt Enable Roll-under Interrupt Enable read/write bit enables roll-under interrupts, based on the Roll-over Interrupt Request bit (INTCTRL[RUIRQ]) being set. 0b - Disabled 1b - Enabled
3 DIRIRQ	Count direction change interrupt Count direction change interrupt bit is set(=1) when POS/POSD/REV counter count direction changes, which means IMR[DIR] changes from 0 to 1 or from 1 to 0 0b - Count direction unchanged 1b - Count direction changed
2 DIRIE	Count direction change interrupt enable Count direction change interrupt enable interrupt 0b - Disabled 1b - Enabled
1 SABIRQ	Simultaneous PHASEA and PHASEB Change Interrupt Request Simultaneous PHASEA and PHASEB Change Interrupt Request indicates that the PHASEA and PHASEB inputs changed simultaneously (within a single clock period). This event typically indicates an error condition, because quadrature coding requires only one of these inputs to change at a time. • Simultaneous PHASEA and PHASEB Change Interrupt Request bit remains set until it is cleared by software or a reset • Write 1 to this bit (SABIRQ) to clear it 0b - No simultaneous change of PHASEA and PHASEB has occurred 1b - A simultaneous change of PHASEA and PHASEB has occurred
0 SABIE	Simultaneous PHASEA and PHASEB Change Interrupt Enable Simultaneous PHASEA and PHASEB Change Interrupt Enable bit enables simultaneous PHASEA and PHASEB change interrupts, based on the Simultaneous PHASEA and PHASEB Change Interrupt Request (SABIRQ) bit being set. 0b - Disabled 1b - Enabled

73.6.1.37 Watchdog Timeout Register (WTR)

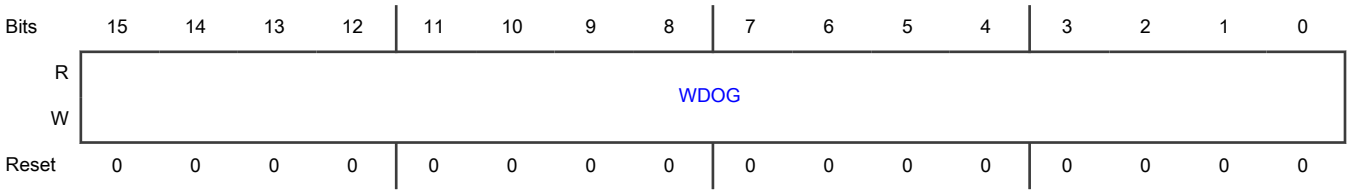
Offset

Register	Offset
WTR	3Ah

Function

The Watchdog Timeout Register stores the timeout count for the quadrature decoder module watchdog timer. This quadrature decoder module watchdog timer is separate from any other watchdog timer(s) that may also be in the device.

Diagram



Fields

Field	Function
15-0 WDOG	WDOG WDOG[15:0] is a binary representation of the number of clock cycles, using the peripheral clock prescaled by FILT[PRSC], 0xFFFF plus one count in this register determines the timing out period.

73.6.1.38 Input Monitor Register (IMR)

Offset

Register	Offset
IMR	3Ch

Function

The Input Monitor Register contains the values of the raw and filtered PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals. The reset value depends on the values of the raw and filtered values of PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals:

- The Input Monitor Register [3:0] contains the values of the raw value of PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals.
- The Input Monitor Register [7:4] contains the values of the filtered value of PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals.
- The Input Monitor Register [7:4] supply filter bypass function for PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals.
- The Input Monitor Register [11:8] contains the values of the 4 POS counter CMPs flag.
- The Input Monitor Register [15:14] contains the values of the count direction flag and count direction flag hold.
- If no pull-up or pull-down is connected to PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals, then the reset value of the 4 lower bits of the Input Monitor Register (IMR) are all unknown.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DIR	DIRH	0		CMP3F	CMP2F	CMP1F	CMPF0	FPHA	FPHB	FIND_PRE	FHOM_ENA	PHA	PHB	INDEX_P...	HOME_EN...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 DIR	Count Direction Flag Output The Count Direction Flag indicates the direction of the current count. 0b - Current count was in the down direction 1b - Current count was in the up direction
14 DIRH	Count Direction Flag Hold DIRH contains a snapshot of the DIR when initializing POS
13-12 —	Reserved
11 CMP3F	Position Compare3 Flag Output Flag of compare3 0b - When the position counter value is less than value of COMP3 register 1b - When the position counter is greater or equal than value of COMP3 register
10 CMP2F	Position Compare2 Flag Output Flag of compare2 0b - When the position counter is less than value of COMP2 register 1b - When the position counter is greater or equal than value of COMP2 register
9 CMP1F	Position Compare1 Flag Output Flag of compare1 0b - When the position counter is less than value of COMP1 register 1b - When the position counter is greater or equal than value of COMP1 register
8 CMPF0	Position Compare 0 Flag Output Flag of compare 0 0b - When the position counter is less than value of COMP0 register 1b - When the position counter is greater or equal than value of COMP0 register
7	filter operation on PHASEA input

Table continues on the next page...

Table continued from the previous page...

Field	Function
FPHA	When write 1, it means filter for PHASEA input is bypassed When read, it shows filtered version of PHASEA input
6 FPHB	filter operation on PHASEB input When write 1, it means filter for PHASEB input is bypassed When read, it shows filtered version of PHASEB input
5 FIND_PRE	filter operation on INDEX/PRESET input When write 1, it means filter for INDEX/PRESET input is bypassed When read, it shows filtered version of INDEX/PRESET input
4 FHOM_ENA	filter operation on HOME/ENABLE input When write 1, it means filter for HOME/ENABLE input is bypassed When read, it shows filtered version of HOME/ENABLE input
3 PHA	PHA Raw PHASEA input
2 PHB	PHB Raw PHASEB input
1 INDEX_PRESET	INDEX_PRESET Raw INDEX/PRESET input
0 HOME_ENABLE	HOME_ENABLE The raw HOME/ENABLE input

73.6.1.39 Test Register (TST)

Offset

Register	Offset
TST	3Eh

Function

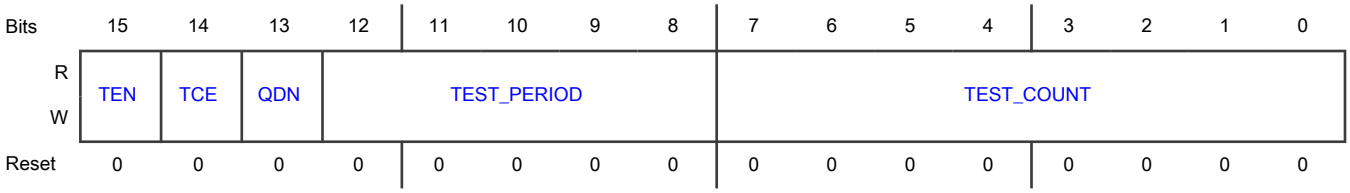
The Test Register controls and sets the frequency of a quadrature signal generator; it provides a quadrature test signal to the inputs of the quadrature decoder module.

- The TEST_COUNT value is counted down to zero when the test module is enabled (TEN = 1) and the counter is enabled (TCE = 1).

- Each count value of one represents a single quadrature cycle interpreted as a count of one by the position counter (UPOS and LPOS) if it is enabled (TEN = 1, TCE = 1).
- Repeated writing of new values to TEST_COUNT can cause an extra phase transition and therefore an extra count by the Position Counter.
- The period field determines the length (in IPBus clock cycles) of each quadrature cycle phase.

The Test Register is a factory test feature; however, it can also be useful in customer software development and testing.

Diagram



Fields

Field	Function
15 TEN	Test Mode Enable Connects the test module to inputs of the quadrature decoder module. 0b - Disabled 1b - Enabled
14 TCE	Test Counter Enable Connects the test counter to inputs of the quadrature decoder module. 0b - Disabled 1b - Enabled
13 QDN	Quadrature Decoder Negative Signal Selects whether a negative or positive Quadrature Decoder signal is generated. 0b - Generates a positive quadrature decoder signal 1b - Generates a negative quadrature decoder signal
12-8 TEST_PERIOD	TEST_PERIOD Period of quadrature phase in IPBus clock cycles
7-0 TEST_COUNT	TEST_COUNT The number of quadrature advances to generate

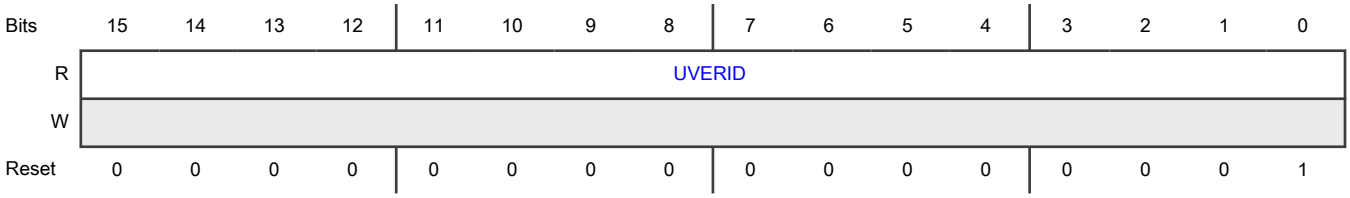
73.6.1.40 Upper VERID (UVERID)

Offset

Register	Offset
UVERID	50h

Function
Version ID Upper part

Diagram



Fields

Field	Function
15-0 UVERID	UVERID Upper (most significant) half of the VERID

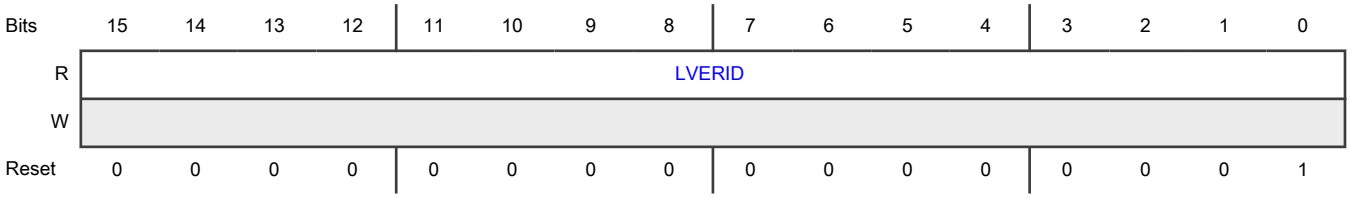
73.6.1.41 Lower VERID (LVERID)

Offset

Register	Offset
LVERID	52h

Function
Version ID Lower part

Diagram



Fields

Field	Function
15-0 LVERID	LVERID Lower (most significant) half of the VERID

Chapter 74
Watchdog timer (WDOG)

74.1 Chip-specific WDOG information

Table 1088. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

When RTWDOG interrupt is enabled by setting INT bit of CS register, RTWDOG backup reset could happen to trigger the system reset which means the counter value (CNT) could be twice of Timeout Value (TOVAL). However, it still requires SW refreshing before TOVAL value.

The following options are available for the CS[CLK] field:

Table 1089. CS[CLK] Options

CLK value	Description
00	Don't use/Reserved
01	32KHz clock
10	Don't use/Reserved
11	24MHz RC Oscillator

74.2 Overview

WDOG is an independent timer that is available for system use. It provides a safety feature to ensure that the software is executing as planned and that the CPU is not stuck in an infinite loop or executing unintended code. If WDOG is not serviced (refreshed) within a certain period, it resets the MCU.

74.2.1 Block diagram

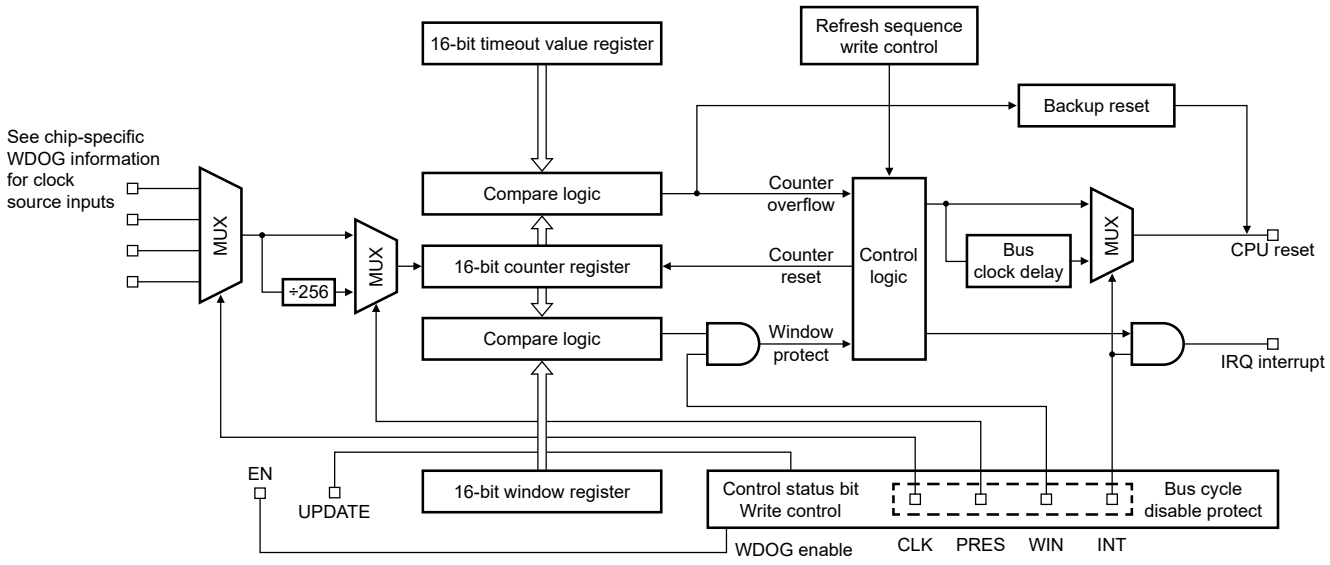


Figure 604. Block diagram

74.2.2 Features

- Configurable clock source inputs independent of the bus clock
- Programmable timeout period
  - Programmable 16-bit timeout value
  - Optional, fixed 256 clock prescaler when longer timeout periods are needed
- Robust write sequences for counter refresh: provision to refresh the sequence of writing to WDOG Counter (CNT)
- Window mode option for the refresh mechanism
  - Programmable 16-bit window value
  - Robust check to ensure that program flow is faster than expected
  - Early refresh attempts that trigger a reset
- Optional timeout interrupt to allow post-processing diagnostics
  - Interrupt request to CPU with an interrupt vector for an interrupt service routine (ISR)
  - Forced reset that occurs 128 bus clocks after the interrupt vector fetch
- Write-once-after-reset configuration fields to ensure that WDOG configuration is not altered mistakenly
- Robust write sequence for unlocking write-once configuration fields
  - Unlock sequence of writing to WDOG Counter (CNT), for allowing updates to write-once configuration fields
  - You must make updates within 1024 bus clocks after unlocking and before WDOG closing unlock window

74.3 Functional description

WDOG provides a fail-safe mechanism to ensure that you can reset the system to a known state of operation in case of system failure, such as the CPU clock stopping or there being a runaway condition in the software code. The WDOG counter runs continuously off a selectable clock source and expects to be serviced (refreshed) periodically. If it is not refreshed, it generates a reset triggering event.

74.3.1 Refresh mechanism

WDOG resets the MCU if the WDOG counter is not refreshed. A robust refresh mechanism makes it very unlikely for a runaway code to refresh WDOG.

To refresh the WDOG counter, you must execute a refresh write sequence before the timeout period expires. In addition, in case of Window mode, you must not start the refresh sequence until you set the time value in Watchdog Window (WIN). See the following figure for more.

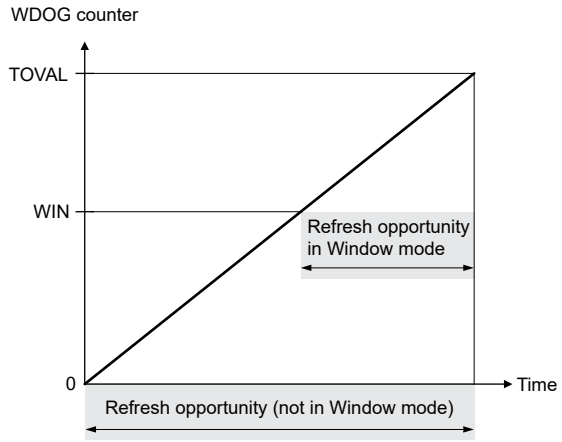


Figure 605. Refresh opportunity for the WDOG counter

74.3.1.1 Using Window mode

Software finishing its main control loop faster than expected could be an indication of a problem. Depending on the requirements of the application, you can program WDOG to force a reset when refresh attempts are early.

When Window mode is enabled, you must refresh WDOG after the counter has reached a minimum expected time value; otherwise, WDOG resets the MCU. The minimum expected time value is specified in Watchdog Window (WIN). Writing 1 to WIN enables Window mode.

74.3.1.2 Refreshing WDOG

The refresh write sequence is based on the following methods:

- Either two 16-bit writes (A602h, B480h) or four 8-bit writes (A6h, 02h, B4h, 80h) to WDOG Counter (CNT) if CMD32EN = 0
- One 32-bit write (B480_A602h) to WDOG Counter (CNT) if CMD32EN = 1

You must apply these methods before WDOG times out; otherwise, it resets the MCU.

Before starting the refresh sequence, disable the global interrupts. Otherwise, an interrupt could effectively invalidate the refresh sequence, if the interrupt occurs before the refresh writes finish. After the sequence finishes, restore the global interrupt control state.

See Application information for example code.

74.3.2 Configuring WDOG

74.3.2.1 Configuring WDOG once

All WDOG control fields, timeout value, and window value are write-once after reset. This means that after a write has occurred, they cannot be changed unless a reset occurs. You can ensure this by configuring the window and timeout values first, followed by the other control fields, when UPDATE = 0.

This provides a robust mechanism to configure WDOG and ensure that a runaway condition cannot mistakenly disable or modify the WDOG configuration, after the module is configured.

The new configuration takes effect only after you write to all registers except WDOG Counter (CNT) after reset. Otherwise, WDOG uses the reset values by default. If you do not use Window mode (WIN), you do not need to write to Watchdog Window (WIN) to bring the new configuration into effect.

74.3.2.2 Reconfiguring WDOG

In some cases (for example, when supporting a bootloader function), you may want to reconfigure or disable WDOG, without forcing a reset first:

- By writing 1 to UPDATE on the initial configuration of WDOG after a reset, you can reconfigure WDOG at any time by executing an unlock sequence.
- Conversely, if UPDATE remains 0, the only way to reconfigure WDOG is by initiating a reset.

The unlock sequence is similar to the refresh sequence but uses different values.

74.3.2.3 Unlocking WDOG

The unlock sequence is based on the following two methods:

- Either two 16-bit writes (C520h, D928h) or four 8-bit writes (C5h, 20h, D9h, 28h) to WDOG Counter (CNT), after WDOG is configured, if CMD32EN = 0
- One 32-bit write (D928_C520h) to WDOG Counter (CNT), after WDOG is configured, if CMD32EN = 1

An improper unlock sequence causes WDOG to reset. On completing the unlock sequence, you must reconfigure WDOG within 1024 bus clocks; otherwise, WDOG closes the unlock window.

NOTE

Because it requires 1024 bus clocks to reconfigure WDOG, you must insert some delays before executing stop or wait instructions after reconfiguring WDOG. This ensures that WDOG's new configuration takes effect before the MCU enters Low-Power mode. Otherwise, the MCU may not wake up from Low-Power mode.

74.3.3 Functionality in Debug and Low-Power modes

By default, WDOG is not functional in Debug, Wait, or Stop modes. However, it can remain functional in these modes as follows:

- For Debug mode, write 1 to DBG (this way WDOG is functional in Debug mode even when Debug mode holds the CPU).
- For Wait mode, write 1 to WAIT.
- For Stop mode, write 1 to STOP and WAIT, and ensure that the clock source is active in Stop mode.

NOTE

WDOG can generate an interrupt in Stop mode.

For Debug and Stop modes, in addition to the aforementioned configuration, you must use a clock source other than the bus clock as the reference clock for the counter; otherwise, WDOG cannot function.

74.3.4 Fast testing of WDOG

Before executing the application code in safety-critical applications, you must test that WDOG works as expected and resets the MCU. Testing every bit of a 16-bit counter by letting it run to the overflow value takes a relatively long time (64 k clocks).

To help minimize the startup delay for application code after reset, WDOG implements a feature that tests its functioning more quickly by splitting the counter into its constituent byte-wide stages. The low and high bytes are run independently and tested for a timeout against the corresponding byte of WDOG Timeout Value (TOVAL). For a complete coverage when testing the high byte of the counter, the test feature feeds the input clock via the 8th bit of the low byte, thus ensuring that the overflow connection from the low byte to the high byte is tested.

Using this test feature reduces the test time to 512 clocks (not including overhead, such as, user configuration and reset vector fetches). To further speed testing, use a faster clock (such as the bus clock) for the counter reference.

On a POR, the POR field in the system reset register becomes 1, indicating that you must perform the WDOG fast test.

74.3.4.1 Testing each byte of the counter

Perform this procedure to test each byte of the counter:

1. Program the preferred WDOG timeout value in WDOG Timeout Value (TOVAL) during the WDOG configuration period.
2. Select a byte of the counter to test by using the configuration TST = 10b for the low byte, and TST = 11b for the high byte.
3. Wait for WDOG to timeout. Optionally, in the idle loop, increment RAM locations as a parallel software counter for later comparison. Because RAM is not affected by a WDOG reset, the timeout period of the WDOG counter can be compared with the software counter to verify whether the timeout period occurred as expected.

   The WDOG counter times out and forces a reset.
4. Confirm that the WDOG flag in the system reset register is set, indicating that WDOG caused the reset (the POR flag remains clear).
5. Confirm that TST showing a test (10b or 11b) was performed.

If confirmed, the count and compare functions work for the selected byte. Repeat the procedure, selecting the next byte in step 2.

NOTE

Only a POR writes 0 to TST, which is not affected by other resets.

74.3.4.2 Entering User mode

After successfully testing the low and high bytes of the WDOG counter, you can configure TST to 01b to indicate that WDOG is ready for use in application User mode. Therefore, if a reset occurs again, you can recognize the reset trigger as a real WDOG reset caused by runaway or faulty application code.

As an ongoing test when using the default clock source, you can periodically read WDOG Counter (CNT) to ensure that the counter is being incremented.

74.3.5 Clocking

You can program CLK to select clock source options in the WDOG counter. See the chip-specific WDOG information for available clock inputs and the default option for this chip.

The option allows you to select a clock source that is independent of the bus clock for applications that need to meet more robust safety requirements. Using a clock source other than the bus clock ensures that the WDOG counter continues to run if the bus clock is somehow halted (see Backup reset).

For WDOG to function properly, you must enable the default WDOG clock source after its functional reset is deasserted.

An optional fixed prescaler for all clock sources allows longer timeout periods. When PRES = 1, the clock source is prescaled by 256 before clocking the WDOG counter.

The following table summarizes examples of the different available WDOG timeout periods. In the table, RCP means "reference clock period".

Table 1090. WDOG timeout availability

Reference clock	Prescaler	WDOG timeout availability
REF_CLK	Pass through	1 × RCP to 65535 × RCP
	Enable	256 × RCP to 16776960 × RCP

NOTE

When you switch clock sources during reconfiguration, WDOG holds the counter at zero for 2.5 periods of the previous clock source and 2.5 periods of the new clock source after the configuration period (1024 bus clocks) ends. This delay ensures a smooth transition before restarting the counter with the new configuration.

74.3.6 Backup reset

The backup reset function is a safeguard feature that independently generates a reset in case the main WDOG logic loses its clock (the bus clock) and can no longer monitor the counter. If the WDOG counter overflows twice in succession (without an intervening reset), the backup reset function takes effect and generates a reset.

The backup reset becomes valid when an interrupt is enabled and the WDOG clock is not from a bus clock. If an interrupt is enabled after the bus clock is cut off before exiting an interrupt routine, the normal WDOG reset is blocked. In this case, the second overflow causes a backup reset directly.

NOTE

You must use a clock source other than the bus clock as a reference clock for the counter; otherwise, the backup reset function becomes unavailable.

74.3.7 Interrupts

WDOG can generate an interrupt request to delay resets.

When interrupts are enabled (INT = 1), and after a reset-triggering event (such as a counter timeout or invalid refresh attempt), WDOG:

1. Generates an interrupt request.
2. Waits 128 bus clocks (from the interrupt vector fetch, not the reset-triggering event).
3. Forces a reset.

This process allows the ISR to perform tasks such as analyzing the stack to debug code.

When interrupts are disabled (INT = 0), WDOG does not wait before forcing a reset.

74.4 External signals

This module has no external signals.

74.5 Initialization

See Configuring WDOG once.

74.6 Application information

You must disable WDOG to reconfigure it before the first WDOG timeout. Disabling or reconfiguring WDOG must occur at the very beginning of the software code, for example, at the beginning of the startup or main function.

NOTE

- After you configure WDOG, it needs at least 2.5 periods of WDOG clock to take effect. This means you must have a gap of at least 2.5 clocks between two configurations.
- When the chip starts from boot ROM and then jumps to flash memory, you must disable WDOG at the beginning of the bootloader and enable it after the bootloader exits. To reconfigure WDOG using the flash memory program, it also needs the interval of at least 2.5 WDOG clocks after the bootloader exits.

To disable or reconfigure WDOG without forcing a reset, you must write 1 to UPDATE during the initial WDOG configuration. You can use the unlock sequence at any time, within the timeout limit, to reconfigure WDOG.

74.6.1 Disabling WDOG

To disable WDOG, you must first perform the unlock sequence. Then, write 0 to EN. The following code snippet shows an example of a 32-bit write.

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
WDOG_CS &= ~WDOG_CS_EN_MASK; //disable watchdog
EnableInterrupts; //enable global interrupt
```

74.6.2 Disabling WDOG after reset

All WDOG registers are unlocked by reset. Therefore, an unlock sequence is unnecessary but it needs to be written to all WDOG registers to make the new configuration take effect. The following code snippet shows an example of disabling WDOG after reset.

```
DisableInterrupts; // disable global interrupt
WDOG_CS &= ~WDOG_CS_EN_MASK; // disable watchdog
WDOG_TOVAL= 0xFFFF;
while(WDOG_CS[ULK]);  // waiting for lock
while(~WDOG_CS[RCS]); // waiting for new configuration to take effect
EnableInterrupts; // enable global interrupt
```

74.6.3 Configuring WDOG

You can write 0 to UPDATE to configure WDOG. After that, you cannot reconfigure WDOG until a reset. To reconfigure without forcing a reset, write 1 to UPDATE when configuring WDOG. The following example code shows how to configure WDOG without Window mode, clock source as LPO, interrupt enabled, and timeout value to 256 clocks. The following code snippet shows an example of a 32-bit write.

74.6.3.1 Configuring once

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
while(WDOG_CS[ULK]==0);  //wait until registers are unlocked
WDOG_TOVAL = 256; //set timeout value
WDOG_CS = WDOG_CS_EN(1) | WDOG_CS_CLK(1) | WDOG_CS_INT(1) |
          WDOG_CS_WIN(0) | WDOG_CS_UPDATE(0);
while(WDOG_CS[RCS]==0); //wait until new configuration takes effect
EnableInterrupts; //enable global interrupt
```

74.6.3.2 Configuring for Reconfigurable mode

```
DisableInterrupts; //disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
while(WDOG_CS[ULK]==0);  //wait until registers are unlocked
WDOG_TOVAL = 256; //set timeout value
WDOG_CS = WDOG_CS_EN(1) | WDOG_CS_CLK(1) | WDOG_CS_INT(1) |
          WDOG_CS_WIN(0) | WDOG_CS_UPDATE(1);
while(WDOG_CS[RCS]==0); //wait until new configuration takes effect
EnableInterrupts; //enable global interrupt
```

74.6.4 Refreshing WDOG

To refresh WDOG and reset the WDOG counter to zero, you require a refresh sequence. The following code snippet shows an example of a 32-bit write.

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xB480A602; // refresh watchdog
EnableInterrupts; // enable global interrupt
```

74.7 Memory map and register definition

74.7.1 WDOG register descriptions

74.7.1.1 WDOG memory map

RTWDOG1 base address: 442D_0000h

RTWDOG2 base address: 442E_0000h

RTWDOG3 base address: 4249_0000h

RTWDOG4 base address: 424A_0000h

RTWDOG5 base address: 424B_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	WDOG Control and Status (CS)	32	RW	0000_0900h
4h	WDOG Counter (CNT)	32	RW	0000_0000h
8h	WDOG Timeout Value (TOVAL)	32	RW	0000_0400h
Ch	Watchdog Window (WIN)	32	RW	0000_0000h

74.7.1.2 WDOG Control and Status (CS)

Offset

Register	Offset
CS	0h

Function
Describes watchdog control and status.

NOTE

TST is cleared (0:0) on POR only. Any other reset does not affect the value of this field.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WIN	FLG	CMD32EN	PRES	ULK	RCS	CLK		EN	INT	UPDATE	TST		DBG	WAIT	STOP
W		W1C														
Reset	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 WIN	WDOG Window Enables Window mode. See Using Window mode for more information. You can write to this field only once. 0b - Disable 1b - Enable
14 FLG	WDOG Interrupt Flag Acts as an interrupt indicator when INT = 1. 0b - No interrupt occurred 1b - An interrupt occurred
13 CMD32EN	Command 32 Enable Enables or disables WDOG support for 32-bit (otherwise 16-bit or 8-bit) refresh or unlock command write words. If this field = 0, it disables support for 32-bit refresh or unlock command write words. Only a 16-bit or 8-bit write is supported. If this field = 1, it enables support for 32-bit refresh or unlock command write words. A 16-bit or 8-bit write is not supported. This is a write-once field, and you must unlock WDOG after writing to this field for reconfiguration. 0b - Disable 1b - Enable
12 PRES	WDOG Prescaler Enables a fixed 256 prescaling of the WDOG counter reference clock. See Block diagram that shows the clock divider option.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This is a write-once field. 0b - Disable 1b - Enable
11 ULK	Unlock Status Indicates whether WDOG is unlocked. 0b - Locked 1b - Unlocked
10 RCS	Reconfiguration Success Indicates whether the reconfiguration is successful. This field becomes 1 when new configuration takes effect, and becomes 0 with a successful unlock command. 0b - Unsuccessful 1b - Successful
9-8 CLK	WDOG Clock Indicates the clock source that feeds the WDOG counter. You can write to this field only once.
7 EN	WDOG Enable Enables the WDOG counter to start counting. You can write to this field only once. 0b - Disable 1b - Enable
6 INT	WDOG Interrupt Configures WDOG to immediately generate an interrupt request upon a reset-triggering event (timeout or illegal write to WDOG), before forcing a reset. After the interrupt vector fetch (that takes place after the reset-triggering event), the reset occurs after a delay of 128 bus clocks. If this field = 0, it disables WDOG interrupts (WDOG resets are not delayed). If this field = 1, it enables WDOG interrupts (WDOG interrupts are delayed by 128 bus clocks from the interrupt vector fetch). You can write to this field only once. 0b - Disable 1b - Enable
5 UPDATE	Updates Allowed Allows you to reconfigure WDOG without a reset. If this field = 0, you cannot update WDOG after the initial configuration, without forcing a reset. If this field = 1, you can modify the WDOG configuration registers within 1024 bus clocks after performing the unlock write sequence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	You can write to this field only once. 0b - Updates not allowed 1b - Updates allowed
4-3 TST	WDOG Test Enables Fast Test mode, which allows you to exercise all bits of the counter to demonstrate that WDOG is functioning properly. See Fast testing of WDOG for more information. If this field = 0, it disables WDOG Test mode. If this field = 1, it enables WDOG User mode and disables WDOG Test mode. After testing WDOG, you must use this setting to indicate that WDOG is functioning normally in User mode. If this field = 10, it enables WDOG Test mode; only the low byte is used. CNTLOW is compared with TOVALLOW. If this field = 11, it enables WDOG Test mode; only the high byte is used. CNTHIGH is compared with TOVALHIGH. This write-once field is cleared (0:0) on POR only. Any other reset does not affect the value of this field. 00b - Disable WDOG Test mode 01b - Enable WDOG User mode 10b-11b - Enable WDOG Test mode
2 DBG	Debug Enable Enables WDOG to operate when the chip is in Debug mode. You can write to this field only once. 0b - Disable 1b - Enable
1 WAIT	Wait Enable Enables WDOG to operate when the chip is in Wait mode. You can write to this field only once. 0b - Disable 1b - Enable
0 STOP	Stop Enable Enables WDOG to operate when the chip is in Stop mode. You can write to this field only once. 0b - Disable 1b - Enable

74.7.1.3 WDOG Counter (CNT)

Offset

Register	Offset
CNT	4h

Function

Provides access to the value of the free-running WDOG counter. You can read the counter register at any time but cannot write directly to it. However, the following write sequences to this register have special functions:

1. The refresh sequence resets WDOG counter to 0000h. See Refreshing WDOG for more information.
2. The unlock sequence allows WDOG to be reconfigured without forcing a reset (when UPDATE = 1). See Configuring for Reconfigurable mode for more information.

NOTE

All other writes to this register are illegal and force a reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CNTHIGH								CNTLOW							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 CNTHIGH	Counter Low Byte Contains the WDOG low byte.
7-0 CNTLOW	Counter High Byte Contains the WDOG high byte.

74.7.1.4 WDOG Timeout Value (TOVAL)

Offset

Register	Offset
TOVAL	8h

Function

Contains the 16-bit value used to set the timeout period of WDOG.

The WDOG counter (CNT) is continuously compared with the timeout value (TOVAL). If the counter reaches the timeout value, WDOG forces a reset triggering event.

NOTE

Do not write 0 to this register (if TST = 11b, then TOVALHIGH cannot be written as 0; if TST = 10b, then TOVALLOW cannot be 0); otherwise, WDOG always generates a reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TOVALHIGH								TOVALLOW							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 TOVALHIGH	Timeout Value High Contains the high byte of the timeout value.
7-0 TOVALLOW	Timeout Value Low Contains the low byte of the timeout value.

74.7.1.5 Watchdog Window (WIN)

Offset

Register	Offset
WIN	Ch

Function

Determines the earliest time that a refresh sequence is considered valid, if WIN = 1. See Refresh mechanism for more information.

The WIN register value must be less than the TOVAL register value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WINHIGH								WINLOW							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 WINHIGH	High Byte Contains the high byte of the WDOG window.
7-0 WINLOW	Low Byte Contains the low byte of the WDOG window.

Chapter 75
External Watchdog Monitor (EWM)

75.1 Chip-specific EWM information

Table 1091. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

The following options are EWM low power clock sources for this chip:

Table 1092. EWM Low Power Clock Sources

lpo_clk_x	Description
lpo_clk_0	24MHz RC Oscillator
lpo_clk_1	1MHz RC Oscillator
lpo_clk_2	32KHz clock
lpo_clk_3	bus clock

75.2 Overview

For safety purposes, a redundant watchdog system, EWM, is designed to monitor external circuits and the MCU software flow. This provides a backup mechanism to the internal watchdog that resets the MCU's CPU and peripherals.

The internal watchdog is used to monitor the flow and execution of the embedded software within the MCU. It consists of a counter that, if allowed to overflow, forces an internal, asynchronous reset to all on-chip peripherals. The counter also optionally asserts the RESET_B pin to reset external devices and circuits. The watchdog counter must not overflow if the software code works well and services the watchdog to restart the actual counter.

The EWM does not reset the MCU's CPU and peripherals, making it different from internal watchdog. The EWM module provides an independent ewm_out_b signal that, when asserted, resets or places an external circuit into a safe mode. The ewm_out_b

signal asserts upon EWM counter timeout. An optional external input, ewm_in, allows additional control when asserting the ewm_out_b signal.

75.2.1 Block diagram

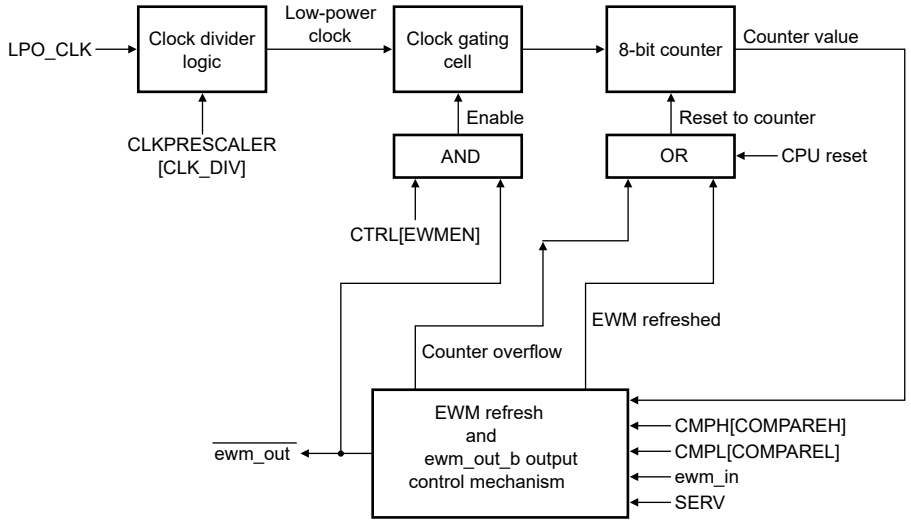


Figure 606. Block diagram

75.2.2 Features

- Independent LPO_CLK source
- Programmable timeout period, specified in terms of the number of EWM LPO_CLK cycles
- Windowed refresh option that provides:
  - A robust check to confirm that the program flow is faster than expected.
  - A programmable window.
  - Refresh outside the window, leading to assertion of the ewm_out_b signal.
- Robust refresh mechanism:
  - Write values of B4h and 2Ch to Service (SERV) within 15 peripheral bus clock cycles.
- One output port, ewm_out_b, which when asserted is used to reset or place the external circuit into Safe mode
- One input port, ewm_in, which allows an external circuit to control the assertion of the ewm_out_b signal

75.3 Functional description

The following sections discuss these aspects of EWM:

- Functional details
- Operating modes

NOTE

If the BUS_CLK is lost, EWM does not generate the ewm_out_b signal and no refresh operation is possible.

75.3.1 Modes of operation

75.3.1.1 Stop mode

When EWM is in Stop mode, the CPU cannot refresh EWM. After entering Stop mode, the EWM counter freezes.

Following are the possible ways to exit Stop mode:

- Through a reset: EWM remains disabled in this case.
- Through an interrupt: EWM is re-enabled and the counter continues to be clocked from the same value as prior to Stop mode entry.

NOTE

Consider the following if EWM enters Stop mode during the CPU refresh mechanism:

- While exiting Stop mode through an interrupt, the refresh mechanism starts from the previous state. That is, if you write the refresh command correctly and EWM enters Stop mode immediately, you must write the next command in *15* peripheral bus clocks after exiting Stop mode.
- You must mask all interrupts before executing the EWM refresh instructions.

75.3.1.2 Wait mode

The EWM module treats the Stop and Wait modes as the same. EWM functionality remains the same in both modes.

75.3.1.3 Debug mode

EWM remains unimpacted when entering Debug mode:

- If EWM is enabled before entering Debug mode, it remains enabled.
- If EWM is disabled before entering Debug mode, it remains disabled.

75.3.2 Using the EWM counter

EWM uses an 8-bit ripple counter that is fed by a clock source independent of the peripheral bus clock source. As the preferred timeout is between 1 ms and 100 ms, the actual clock source must be in the kHz range.

The counter is reset to 0 in these conditions:

- After CPU reset
- After the EWM refresh action completes
- At counter overflow

The CPU cannot access the counter value.

75.3.3 Using compare registers

You can write to Compare Low (CMPL) and Compare High (CMPH) only once after a CPU reset and you cannot modify them until another CPU reset occurs. These registers are used to create a refresh window for the EWM module.

You cannot program Compare Low (CMPL) and Compare High (CMPH) with the same value. In case of any attempt, the ewm_out_b signal asserts as soon as the counter reaches the value of Compare Low (CMPL) + 1.

NOTE

- You must update Compare Low (CMPL) and Compare High (CMPH) before enabling EWM. Therefore, you must provide a reasonable time after POR for the external monitoring circuit to stabilize. You must also ensure that the ewm_in pin is deasserted.
- Service should be requested after 1 clock period of slowest clock frequency while updating Compare Low (CMPL) register.

75.3.4 Using the refresh mechanism

Other than the initial configuration of EWM, the CPU can access EWM only through Service (SERV). The CPU must access this register by correctly writing unique data within the windowed time frame, as determined by Compare Low (CMPL) and Compare High (CMPH) for the correct EWM refresh operation. The following table describes conditions that exist and the refresh mechanisms that apply to those conditions.

Table 1093. Refresh mechanisms

Condition	Mechanism
The EWM refresh action completes when the value of Compare Low (CMPL) ≤ the counter value ≤ the value of Compare High (CMPH).	The software behaves as expected and the EWM counter resets to 0. The ewm_out_b output signal remains in Deasserted state if, during the EWM refresh action, the ewm_in input is in Deasserted state.
The EWM refresh action completes when the counter value < the value of Compare Low (CMPL).	The software refreshes EWM before the windowed time frame, the counter resets to 0, and the ewm_out_b output signal asserts no matter what the value of the ewm_in input signal is.
The counter value becomes greater than the value of Compare High (CMPH) prior to completion of the EWM refresh action.	The software does not refresh EWM. The EWM counter resets to 0 and the ewm_out_b output signal asserts no matter what the value of the ewm_in input signal is.

See Service (SERV) for more on the refresh mechanism.

75.3.5 Interrupt

When the ewm_out_b signal asserts, an interrupt request can be generated to indicate the assertion of the EWM reset out signal. The interrupt is enabled when CTRL[INTEN] = 1. Writing 0 to this field clears the interrupt request but does not affect the ewm_out_b signal, which can be deasserted only by forcing a system reset.

75.3.6 Clocking

The following table shows EWM clocks.

Table 1094. EWM Clocks

Clock	Description
IPG_CLK	This is the system clock and should be turned on for EWM to be able to work properly. During low power modes in which the core is powered down, this clock is disabled,
IPG_CLK_S	This is the IPS clock and is synchronous with IPG_CLK. It is disabled except during IPS write accesses. it is enabled with EWM's IPS_MODULE_EN
LPO_CLOCK[3:0]	EWM can have 4 different clock sources for running its EWM counter and one of them can be selected by EWM_CLKCTRL [CLK_SEL]. This clock is gated when EWM is disabled or when EWM_out is asserted.

75.3.7 Selecting the EWM counter clock

You can program CLKCTRL[CLKSEL] to select from the available low-power clock sources for the EWM counter.

75.3.8 Using the counter clock prescaler

You can program CLKPRESCALER[CLK_DIV] to divide the EWM counter clock source. This divided clock is used to run the EWM counter.

NOTE

The divided clock used to run the EWM counter must not exceed half the frequency of the bus clock.

75.4 External signals

EWM includes external signals and internal options for the counter clock sources, as shown in the following table.

NOTE

All active-low signals are represented with the suffix "_b" throughout the chapter.

Table 1095. Signal descriptions

Signal	Description	I/O
ewm_in	EWM's input for the safety status of external safety circuits. You can program the polarity of ewm_in by using CTRL[ASSIN]. The default polarity is active-low.	I
ewm_out_b	EWM's reset out signal	O
lpo_clk[3:0]	Low-power clock sources for the running counter	I

75.4.1 Using the ewm_out_b signal

The ewm_out_b signal is a digital output signal used to gate an external circuit (application-specific) that controls critical safety functions. For example, EWM_out must be connected to the high-voltage transistor circuits that control an AC motor in a large appliance.

The ewm_out_b signal remains deasserted when the CPU regularly refreshes EWM within the programmable refresh window, indicating that the application code is executing as expected.

The ewm_out_b signal asserts in any of the following conditions:

- An EWM refresh action occurs when the counter value is less than the value of Compare Low (CMPL).
- The EWM counter value becomes greater than the value of Compare High (CMPH) and no EWM refresh occurs.
- The functionality of the ewm_in pin is enabled and the ewm_in pin asserts when refreshing EWM.
- After any reset.

The ewm_out_b signal asserts after any reset by the virtue of the external pulldown mechanism on the ewm_out_b pin. To deassert the ewm_out_b signal, write 1 to CTRL[EWMEN] to enable EWM.

If the ewm_out_b signal shares its pad with a digital I/O pin, this actual pad defers to being an input signal on reset. The ewm_out_b signal controls the pad state only after CTRL[EWMEN] enables EWM.

NOTE

The ewm_out_b pad must be in Pulldown state when the EWM functionality is being used and EWM is under reset.

75.4.2 Using the ewm_out_b pin state in low-power modes

During Wait, Stop, and Power-Down modes, the ewm_out_b pin enters a high-impedance state. You have the option to control the logic state of the pin by using an external pull device or by configuring the internal pull device. When the CPU enters Run mode

from Wait or Stop mode recovery, the pin resumes its previous state before entering Wait or Stop mode. When the CPU enters Run mode after exiting Power-Down mode, the pin returns to its reset state.

75.4.3 Using the ewm_in signal

The ewm_in signal is a digital input signal for the safety status of external safety circuits. This signal allows an external circuit to control the assertion of the ewm_out_b signal. For example, in the application, an external circuit monitors a critical safety function, and if there is a fault with the safety function, the external circuit can actively initiate the ewm_out_b signal, which controls the gating circuit.

The ewm_in signal is ignored if EWM is disabled, or if CTRL[INEN] = 0 after any reset.

After you enable EWM (by writing 1 to CTRL[EWMEN]) and the ewm_in functionality (by writing 1 to CTRL[INEN]), the ewm_in signal must be in Deasserted state before the CPU starts refreshing EWM. This ensures that the ewm_out_b signal stays in Deasserted state; otherwise, the ewm_out_b output signal asserts.

75.5 Memory map and register definitions

This section contains the module memory map and registers.

NOTE

EWM supports only 8-bit register accesses; 16-bit and 32-bit accesses are not supported.

75.5.1 EWM register descriptions

75.5.1.1 EWM memory map

EWM base address: 427B_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CTRL)	8	RW	00h
1h	Service (SERV)	8	W	00h
2h	Compare Low (CMPL)	8	RW	00h
3h	Compare High (CMPH)	8	RW	FFh
4h	Clock Control (CLKCTRL)	8	RW	00h
5h	Clock Prescaler (CLKPRESCALER)	8	RW	00h

75.5.1.2 Control (CTRL)

Offset

Register	Offset
CTRL	0h

Function

Controls the functionality of EWM.

NOTE

You can write to CTRL[INEN], CTRL[ASSIN], and CTRL[EWMEN] only once after a CPU reset. Modifying these fields more than once generates a bus transfer error.

Diagram

Bits	7	6	5	4	3	2	1	0
R	0				INTEN	INEN	ASSIN	EWMEN
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-4 —	Reserved
3 INTEN	Interrupt Enable Enables interrupt request generation. If this field = 1 and the ewm_out_b signal is asserted, an interrupt request is generated. To deassert interrupt requests, write 0 to this field. 0b - Deasserts interrupt requests 1b - Generates interrupt requests
2 INEN	Input Enable Enables the ewm_in port. When this field = 1, it enables the ewm_in port. 0b - Disables 1b - Enables
1 ASSIN	Assertion State Select Specifies the asserted state of the ewm_in signal. By default, the asserted state of the ewm_in signal is logic 0 (active-low), which is when this field = 0. When this field = 1, the ewm_in asserted state is logic 1 (active-high). You can use this field to change the expected polarity of the ewm_in signal. 0b - Logic 0 1b - Logic 1
0 EWMEN	EWM Enable Enables the EWM module. If this field = 1, it enables the EWM module, and if the field = 0, it disables the EWM module. You cannot re-enable this field until the next reset because of its write-once nature.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disables 1b - Enables

75.5.1.3 Service (SERV)

Offset

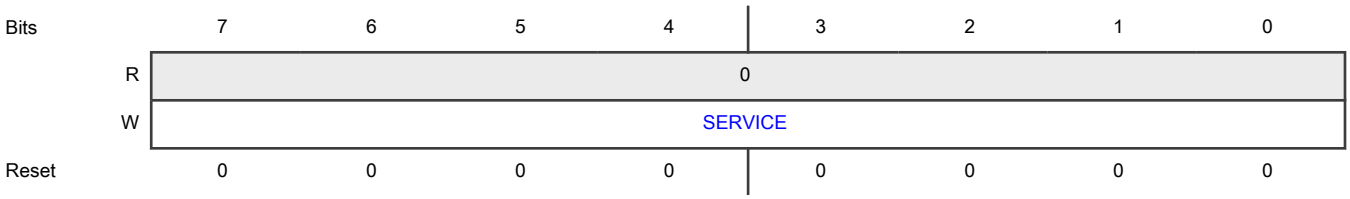
Register	Offset
SERV	1h

Function

Provides an interface from the CPU to the EWM module.

Attempted reads of this register return 0.

Diagram



Fields

Field	Function
7-0 SERVICE	Service Provides an interface from the CPU to the EWM module. The EWM refresh mechanism requires the CPU to write these values to this field: a first data byte of B4h, followed by a second data byte of 2Ch. The EWM refresh action is invalid if either of the following conditions is true: • The first or second data byte is not written correctly. • The second data byte is not written within a fixed number of peripheral bus cycles of the first data byte, known as EWM_refresh_time. The number of peripheral bus clock cycles required for EWM_refresh_time is 15.

75.5.1.4 Compare Low (CMPL)

Offset

Register	Offset
CMPL	2h

Function

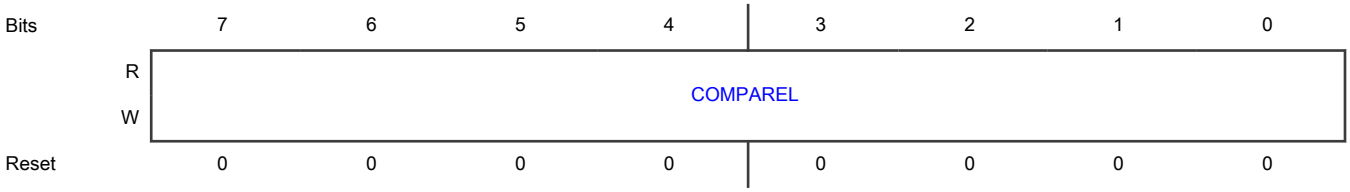
Determines the lower value of the windowed time frame for the correct EWM refresh operation.

This register is reset to 0 after a CPU reset. This provides no minimum time for the CPU to refresh the EWM counter.

NOTE

You can write to this register only once after a CPU reset. Writing to the register more than once generates a bus transfer error.

Diagram



Fields

Field	Function
7-0 COMPAREL	Compare Low Configures the minimum counter value when refreshes are allowed. If a refresh is attempted while the counter value is lower than COMPAREL, then the ewm_out_b signal is asserted. To prevent runaway code from changing the value of this field, you must write to this field after a CPU reset even if the (default) minimum refresh time is required.

75.5.1.5 Compare High (CMPH)

Offset

Register	Offset
CMPH	3h

Function

Determines the higher value of the windowed time frame for the correct EWM refresh operation.

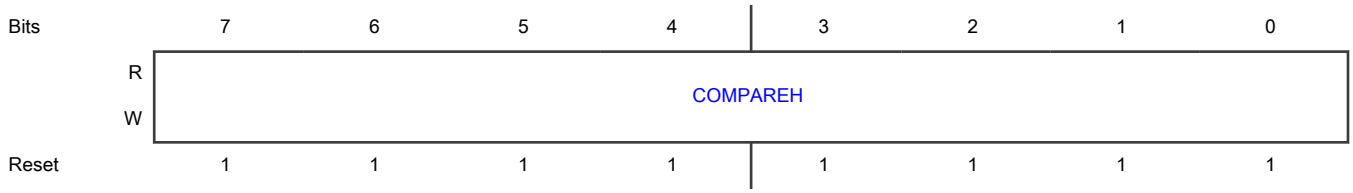
This register is reset to FFh after a CPU reset. This provides a maximum time of up to 256 clocks for the CPU to refresh the EWM counter.

NOTE

You can write to this register only once after a CPU reset. Writing to the register more than once generates a bus transfer error.

The valid values for this register are up to FEh because the EWM counter never expires when the value of COMPAREH = FFh. The expiration happens only if the EWM counter is greater than the value of COMPAREH.

Diagram



Fields

Field	Function
7-0 COMPAREH	Compare High Configures the maximum counter value till when refreshes are allowed. If a refresh is not attempted while the counter value is greater than COMPAREH, then the ewm_out_b signal is asserted. To prevent runaway code from changing the value of this field, you must write to this field after a CPU reset.

75.5.1.6 Clock Control (CLKCTRL)

Offset

Register	Offset
CLKCTRL	4h

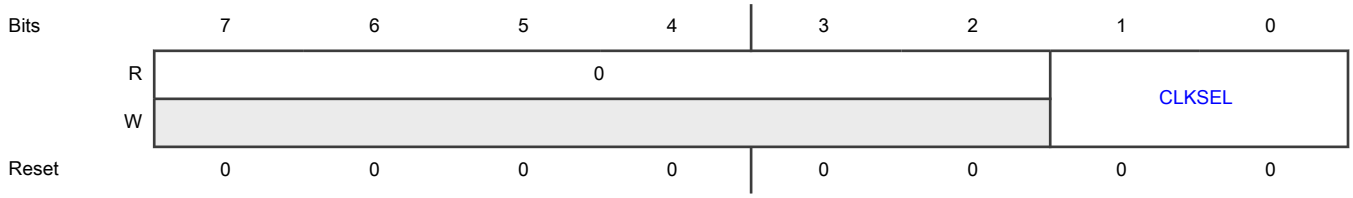
Function

Provides a selection mechanism for low-power clock sources to run the EWM counter.

NOTE

You can write to this register only once after a CPU reset. Writing to the register more than once generates a bus transfer error. You must select the required low-power clock before enabling EWM.

Diagram



Fields

Field	Function
7-2 —	Reserved
1-0 CLKSEL	Clock Select Provides a selection mechanism for low-power clock sources to run the EWM counter. EWM has the following available low-power clock sources for running the EWM counter. Write an appropriate value to this field to select one of the clock sources. • 00 - lpo_clk[0] • 01 - lpo_clk[1] • 10 - lpo_clk[2] • 11 - lpo_clk[3] See the chip-specific information for low power clock sources used in your device.

75.5.1.7 Clock Prescaler (CLKPRESCALER)

Offset

Register	Offset
CLKPRESCALER	5h

Function

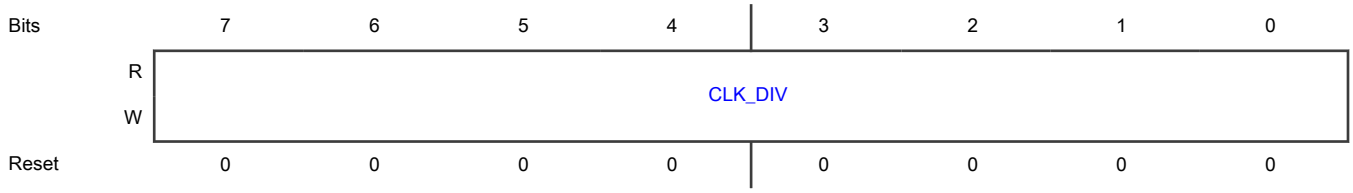
Prescales the EWM counter clock source by a clock divider.

This register is reset to 00h after a CPU reset.

NOTE

You can write to this register only once after a CPU reset. Writing to the register more than once generates a bus transfer error. You must write the required prescaler value before enabling EWM.

Diagram



Fields

Field	Function
7-0 CLK_DIV	Clock Divider Prescales the selected low-power clock source for running the EWM counter: Prescaled clock frequency = low-power clock source frequency ÷ (1 + the value of CLK_DIV) See chip-specific information for low-power clock source frequency used in your device.

Chapter 76
On Chip Cross-Triggers Overview

76.1 Overview

This chip integrates an on-chip cross trigger network.

See XBAR Resource Assignments for diagram of the cross trigger network of this device.

76.1.1 Cross BAR (XBAR)

Each crossbar switch is an array of MUXes with shared inputs. Each mux output provides one output of the crossbar. The number of inputs and the number of MUXes/outputs is user configurable and registers are provided to select which of the shared inputs is routed to each output. The crossbar switches are used to reconfigure data paths between peripherals (peripheral output to peripheral input) as well as between peripherals and GPIO.

76.1.2 And-Or-Inverter (AOI)

The AOI module provides an universal Boolean function generator using a four-term sum of products expression, with each product term containing true or complement values of the four selected inputs (A, B, C, D).

Chapter 77
And-Or-Inverter (AOI)

77.1 Chip-specific AOI Information

Table 1096. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

A typical application of the AOI module is to be integrated with one or more inter-peripheral crossbar switch modules as illustrated in the following figure. The 20 external inputs are shared by two crossbar switch modules. The crossbar switch on the top is used to select the inputs to four 4-input AOI functions in the AOI module. The outputs of these four AOI functions are output from the AOI module and are added to the original 20 external inputs to provide a total of 24 inputs to the bottom crossbar switch. As a result, the bottom crossbar can not only direct any of the original 20 external inputs to any of its outputs, it can also now direct any one of four 4-input AOI functions of those external inputs to any of its outputs.

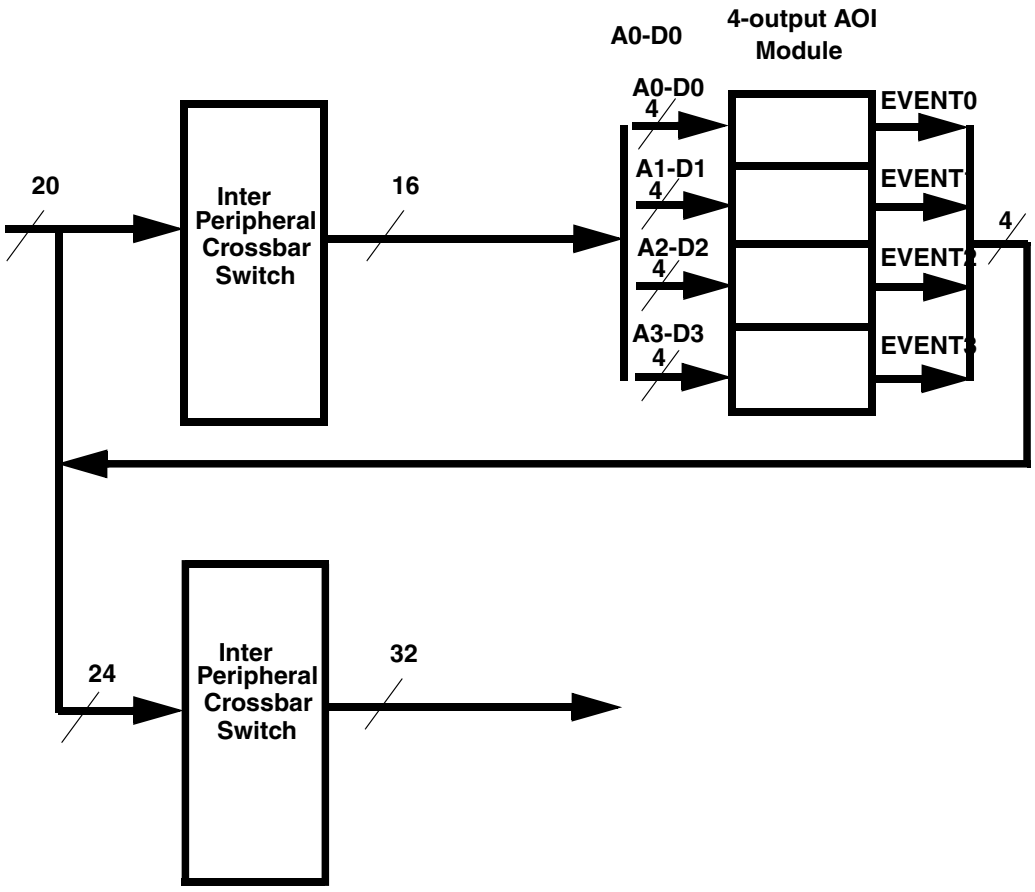


Figure 607. Integration Example of AOI with two Inter-Peripheral Crossbar Switches

77.2 Overview

AOI supports the generation of a configurable number of EVENT signals. Each output EVENT*n* is a configurable AOI function of four associated AOI inputs: A*n*, B*n*, C*n*, and D*n*, where *n* represents the number of channels.

The AOI controller is a target peripheral module that connects event input indicators from a variety of modules and generates event output signals routed to other peripherals. You can access its programming model through the standard IPS target interface.

You can configure AOI to implement different Boolean functions.

77.2.1 Block diagram

AOI supports 4 event outputs, each representing a programmable, combinational Boolean function.

NOTE

The connections from AOI outputs to other functions are chip-specific.

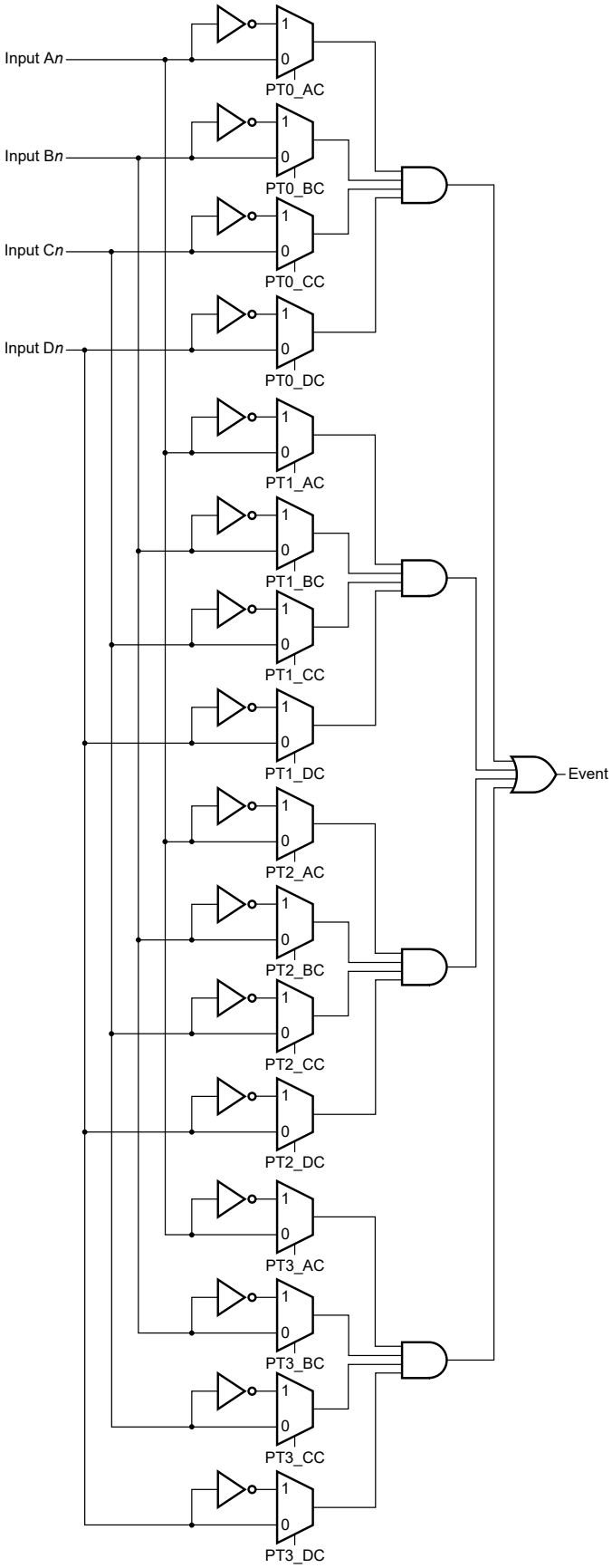


Figure 608. Block diagram

77.2.2 Features

- Enables you to create combinational Boolean events for use as hardware triggers:
  - Each channel includes four event inputs and one output
  - Evaluation of a combinational Boolean expression as the sum of four products, where each product term includes all four selected input sources available as true or complement values
  - Event output is formed as purely combinational logic and operates as a hardware trigger
- Includes a memory-mapped chip connected to the target peripheral (IPS) bus (programming model organized per channel for simplified software)

77.3 Functional description

AOI is highly programmable–you can use it for creating combinational Boolean outputs for use as hardware triggers. Each AOI output channel, as shown in Figure 608, has one logic function: Evaluation of combinational Boolean expression as a sum of four products, where each product term includes all four selected input sources available as true or complement values.

77.3.1 Modes of operation

AOI does not support any special modes of operation.

77.3.2 Clocks

The system uses the bus clock for register accesses, but AOI functions are asynchronous and do not use a clock.

77.3.3 Interrupts

This module has no interrupts.

77.4 External signals

This module has no external signals.

77.5 Initialization

This module does not require initialization.

77.6 Application information

The following sections describe configuration examples and Boolean expressions that you could use when programming AOI for your use.

77.6.1 Configuration examples for Boolean function evaluation

This section presents examples of the programming model configuration for simple Boolean expressions.

AOI provides a universal Boolean function generator using a four-term sum of product expression. Each product term contains true or complement values of the four selected event inputs (A*n*, B*n*, C*n*, and D*n*). See Code Listing 1 that presents a 4×4 Boolean expression defining the event output:

```
EVENTn
      = (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 0
      | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 1
```

```
      | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 2
      | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 3
```

Code Listing 1. Boolean expression defining event inputs (A*n*, B*n*, C*n*, and D*n*)

You can configure each selected input term in each product term to produce a logical 0 or 1, or pass the true or complement value of the selected event input. Each product term uses 8 bits of configuration information, 2 bits for each of the four selected event inputs. See Code Listing 2 that presents the actual Boolean expression implemented in each channel:

```
EVENTn
      = (PT0_AC[0] & An |  PT0_AC[1] & ~An)// product term 0
      & (PT0_BC[0] &  Bn | PTO_BC[1] & ~Bn)
      & (PT0_CC[0] &  Cn | PTO_CC[1] & ~Cn)
      & (PT0_DC[0] &  Dn | PTO_DC[1] & ~Dn)

| (PT1_AC[0] & An |  PT1_AC[1] & ~An)// product term 1
& (PT1_BC[0] &  Bn | PT1_BC[1] & ~Bn)
& (PT1_CC[0] &  Cn | PT1_CC[1] & ~Cn)
& (PT1_DC[0] &  Dn | PT1_DC[1] & ~Dn)

| (PT2_AC[0] & An |  PT2_AC[1] & ~An)// product term 2
& (PT2_BC[0] &  Bn | PT2_BC[1] & ~Bn)
& (PT2_CC[0] &  Cn | PT2_CC[1] & ~Cn)
& (PT2_DC[0] &  Dn | PT2_DC[1] & ~Dn)

| (PT3_AC[0] & An |  PT3_AC[1] & ~An)// product term 3
& (PT3_BC[0] &  Bn | PT3_BC[1] & ~Bn)
& (PT3_CC[0] &  Cn | PT3_CC[1] & ~Cn)
& (PT3_DC[0] &  Dn | PT3_DC[1] & ~Dn)
```

Code Listing 2. Boolean expressions for EVENT*n*

The combined fields of {Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013), Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233)} correspond to the `PT{0-3}_{A,B,C,D}C[1:0]` terms in the aforementioned example.

Consider the settings of the combined 32-bit registers {Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013), Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233)} for several simple Boolean expressions, as shown in Table 1097.

Table 1097. BFCRT*n* values for simple boolean expressions

Event output expression	PT0	PT1	PT2	PT3	{Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013), Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233)}
A & B	A & B	0	0	0	01011111_00000000_00000000_00000000
A & B & C	A & B & C	0	0	0	01010111_00000000_00000000_00000000
(A & B & C) + D	A & B & C	D	0	0	01010111_11111101_00000000_00000000
A + B + C + D	A	B	C	D	01111111_11011111_11110111_11111101
(A & ~B) + (~A & B)	A & ~B	~A & B	0	0	01101111_10011111_00000000_00000000

As explained through the aforementioned examples, the resulting logic provides a simple yet powerful Boolean function evaluation for defining an event output.

77.7 AOI register descriptions

You can access the AOI programming model via a 16-bit peripheral bus connection. AOI is designed to support 16-bit accesses only. Other accesses are undefined.

AOI supports a specific number of event outputs. Each output EVENT*n* outputs a four-term AOI function of four binary inputs: A*n*, B*n*, C*n*, and D*n*. A pair of 16-bit registers configures this four-term AOI function: Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013) and Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233) define the configuration for the evaluation of the Boolean function defining EVENT*n*, where *n* is the event output channel number. Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013) defines the configuration of product terms 0 and 1, and Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233) defines the configuration of product terms 2 and 3.

AOI provides a universal Boolean function generator using a four-term sum of product expression with each product term containing true or complement values of the four selected event inputs (A*n*, B*n*, C*n*, and D*n*). Specifically, the following "4 x 4" Boolean expression defines the EVENT*n* output:

```
EVENTn
                          = (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product
term 0
                          | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product
term 1
                          | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product
term 2
                          | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product
term 3
```

Code Listing 3. Boolean expression defining the EVENT*n* output

Using the aforementioned expression, you can configure each selected input in each product term to produce a logical 0 or 1 or pass the true or complement value of the selected event input. Each product term uses 8 bits of configuration information, 2 bits for each of the four selected event inputs. The resulting logic provides a simple yet powerful Boolean function evaluation for defining an event output.

These AOI functions are combinational in nature. You must sample and use them synchronously.

77.7.1 AOI memory map

AOI1 base address: 4278_0000h

AOI2 base address: 4279_0000h

AOI3 base address: 427E_0000h

AOI4 base address: 427F_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Boolean Function Term 0 and 1 Configuration for EVENT0 (BFCRT010)	16	RW	0000h
2h	Boolean Function Term 2 and 3 Configuration for EVENT0 (BFCRT230)	16	RW	0000h
4h	Boolean Function Term 0 and 1 Configuration for EVENT1 (BFCRT011)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
6h	Boolean Function Term 2 and 3 Configuration for EVENT1 (BFCRT231)	16	RW	0000h
8h	Boolean Function Term 0 and 1 Configuration for EVENT2 (BFCRT012)	16	RW	0000h
Ah	Boolean Function Term 2 and 3 Configuration for EVENT2 (BFCRT232)	16	RW	0000h
Ch	Boolean Function Term 0 and 1 Configuration for EVENT3 (BFCRT013)	16	RW	0000h
Eh	Boolean Function Term 2 and 3 Configuration for EVENT3 (BFCRT233)	16	RW	0000h

77.7.2 Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013)

Offset

Register	Offset
BFCRT010	0h
BFCRT011	4h
BFCRT012	8h
BFCRT013	Ch

Function

Configures the Boolean function for terms 0 and 1 of EVENT*n*.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PT0_AC		PT0_BC		PT0_CC		PT0_DC		PT1_AC		PT1_BC		PT1_CC		PT1_DC	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 PT0_AC	Product Term 0, Input A Configuration Defines the Boolean evaluation associated with the selected input A in product term 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Force input A to become 0 01b - Pass input A 10b - Complement input A 11b - Force input A to become 1
13-12 PT0_BC	Product Term 0, Input B Configuration Defines the Boolean evaluation associated with the selected input B in product term 0. 00b - Force input B to become 0 01b - Pass input B 10b - Complement input B 11b - Force input B to become 1
11-10 PT0_CC	Product Term 0, Input C Configuration Defines the Boolean evaluation associated with the selected input C in product term 0. 00b - Force input C to become 0 01b - Pass input C 10b - Complement input C 11b - Force input C to become 1
9-8 PT0_DC	Product Term 0, Input D Configuration Defines the Boolean evaluation associated with the selected input D in product term 0. 00b - Force input D to become 0 01b - Pass input D 10b - Complement input D 11b - Force input D to become 1
7-6 PT1_AC	Product Term 1, Input A Configuration Defines the Boolean evaluation associated with the selected input A in product term 1. 00b - Force input A to become 0 01b - Pass input A 10b - Complement input A 11b - Force input A to become 1
5-4 PT1_BC	Product Term 1, Input B Configuration Defines the Boolean evaluation associated with the selected input B in product term 1. 00b - Force input B to become 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Pass input B 10b - Complement input B 11b - Force input B to become 1
3-2 PT1_CC	Product Term 1, Input C Configuration Defines the Boolean evaluation associated with the selected input C in product term 1. 00b - Force input C to become 0 01b - Pass input C 10b - Complement input C 11b - Force input C to become 1
1-0 PT1_DC	Product Term 1, Input D Configuration Defines the Boolean evaluation associated with the selected input D in product term 1. 00b - Force input D to become 0 01b - Pass input D 10b - Complement input D 11b - Force input D to become 1

77.7.3 Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233)

Offset

Register	Offset
BFCRT230	2h
BFCRT231	6h
BFCRT232	Ah
BFCRT233	Eh

Function
Configures the Boolean function for terms 2 and 3 of EVENT*n*.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PT2_AC		PT2_BC		PT2_CC		PT2_DC		PT3_AC		PT3_BC		PT3_CC		PT3_DC	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 PT2_AC	Product Term 2, Input A Configuration Defines the Boolean evaluation associated with the selected input A in product term 2. 00b - Force input A to become 0 01b - Pass input A 10b - Complement input A 11b - Force input A to become 1
13-12 PT2_BC	Product Term 2, Input B Configuration Defines the Boolean evaluation associated with the selected input B in product term 2. 00b - Force input B to become 0 01b - Pass input B 10b - Complement input B 11b - Force input B to become 1
11-10 PT2_CC	Product Term 2, Input C Configuration Defines the Boolean evaluation associated with the selected input C in product term 2. 00b - Force input C to become 0 01b - Pass input C 10b - Complement input C 11b - Force input C to become 1
9-8 PT2_DC	Product Term 2, Input D Configuration Defines the Boolean evaluation associated with the selected input D in product term 2. 00b - Force input D to become 0 01b - Pass input D 10b - Complement input D 11b - Force input D to become 1
7-6 PT3_AC	Product Term 3, Input A Configuration Defines the Boolean evaluation associated with the selected input A in product term 3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Force input A to become 0 01b - Pass input A 10b - Complement input A 11b - Force input to become 1
5-4 PT3_BC	Product Term 3, Input B Configuration Defines the Boolean evaluation associated with the selected input B in product term 3. 00b - Force input B to become 0 01b - Pass input B 10b - Complement input B 11b - Force input B to become 1
3-2 PT3_CC	Product Term 3, Input C Configuration Defines the Boolean evaluation associated with the selected input C in product term 3. 00b - Force input C to become 0 01b - Pass input C 10b - Complement input C 11b - Force input C to become 1
1-0 PT3_DC	Product Term 3, Input D Configuration Defines the Boolean evaluation associated with the selected input D in product term 3. 00b - Force input D to become 0 01b - Pass input D 10b - Complement input D 11b - Force input D to become 1

Chapter 78
Inter-Peripheral Crossbar Switch (XBAR)

78.1 Chip-specific XBAR information

Table 1098. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

This device has three instances of the XBAR module: XBAR1, XBAR2, and XBAR3.

On this device, the instantation information of XBAR1 and XBAR2/3 is different, as shown in the following table.

Table 1099. XBAR instantiation

Instance	Number of inputs (N)	Number of outputs (M)	Number of outputs with control function (P)	Select registers	Control registers
XBAR1	216	221	4	SEL0-SEL110	CTRL0-CTRL1
XBAR2	178	32	0	SEL0-SEL15	NA
XBAR3	178	32	0	SEL0-SEL15	NA

NOTE

Only XBAR1 has control registers (CTRL*n*), which configure edge detection, interrupt, and DMA features for a subset of the crossbar outputs (XBAR_OUT[0] ~ XBAR_OUT[P-1]). XBAR2 and XBAR3 do not have control registers and corresponding functions.

NOTE

The XBAR_INn and XBAR_OUTn signals (n = 4 to 37) share the same IOs. The user needs to configure the corresponding XBAR_DIR_CTRL1[IOMUXC_XBAR_DIR_SEL_n] and XBAR_DIR_CTRL2[IOMUXC_XBAR_DIR_SEL_n] bit of BLK_CTRL_WAKEUP, to use either XBAR_IN or XBAR_OUT.

78.2 Overview

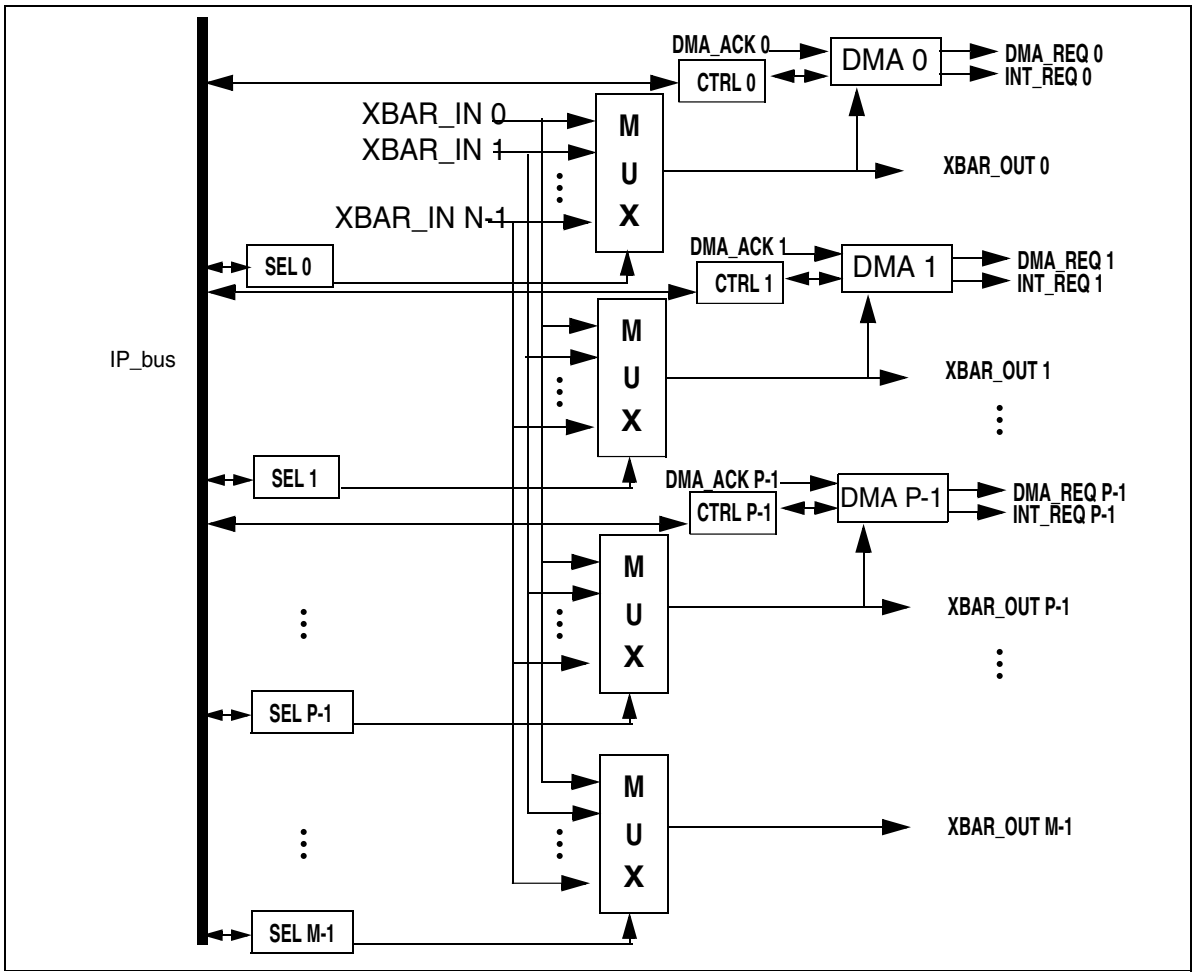
This module implements an array of M N-input combinational muxes. All muxes share the same N inputs in the same order, but each mux has its own independent select field.

The intended application of this module is to provide a flexible crossbar switch function that allows any input (typically from external GPIO or internal module outputs) to be connected to any output (typically to external GPIO or internal module inputs) under user control. This is used to allow user configuration of data paths between internal modules and between internal modules and GPIO.

A subset of the muxes (P in total) can be configured to support edge detection and either interrupt or DMA request generation based on detected signal edges on the mux output. This allows signal transitions on the signals feeding the crossbar to trigger interrupts or initiate data transfers via DMA into or out of other system modules.

78.2.1 Block Diagram

The block diagram for XBAR is shown in Figure 609.



Note: Control field support can be restricted to a subset of outputs. CTRLn represents the memory mapped control fields in the XBAR_CTRL registers for each corresponding XBAR_OUT output. Control fields provide edge status, edge selection, and interrupt/DMA generation controls.

Figure 609. XBAR Block Diagram

78.2.2 Features

The XBAR module design includes these distinctive features:

- M identical N-input muxes with individual select fields.
- Edge detection with associated interrupt or DMA request generation for a subset of mux outputs (XBAR_OUT[0] ~ XBAR_OUT[P-1]).
- Memory mapped registers with IPBus interface for select and control fields.

78.3 Functional Description

78.3.1 Functional Mode

XBAR has only one mode of operation: functional mode.

The value of each mux output is XBAR_OUT[n] = XBAR_IN[SELn]. The SELn select values are configured in the XBAR_SEL registers. All muxes share the same inputs in the same order.

The following is an example to show how to specify the specific input for the specific output:

To connect XBAR_IN03 with XBAR_OUT07, according to the index of XBAR_OUT07, select the SEL07 (same number with the selected XBAR_OUT index) fields in the MSBs of the Select Register 3 (XBAR_SEL3), and assign the index (0x03, Hexadecimal) of the selected XBAR_IN in it.

```
XBAR_SEL3 &= 0x00FF; /* Clear the SEL7 fields */
XBAR_SEL3 |= 0x0300; /* Assign the 0x03 to the SEL7 fields */
```

A subset of crossbar outputs (XBAR_OUT[0] ~ XBAR_OUT[P-1]) has dedicated control fields in a Crossbar Control (XBAR_CTRL) register. Control fields provide the ability to perform edge detection on the corresponding XBAR_OUT output. Edge detection in turn can optionally be used to trigger an interrupt or DMA request. The intention is that, by detecting specified edges on signals propagating through the Crossbar, interrupts or DMA requests can be triggered to perform data transfers to or from other system components.

Control fields include an edge status field (STS), an detected edge type field (EDGE), and interrupt and DMA enable fields (IEN and DEN). STSn is set to 1 when an edge consistent with EDGEn occurs on XBAR_OUT[n]. STSn is cleared by writing 1 to it. Writing 0 has no effect. See Interrupts and DMA Requests for details on the use of STSn for DMA and interrupt request generation.

78.3.2 Clocks

All sequential functionality is controlled by the Bus Clock.

78.3.3 Resets

The XBAR module can be reset by only a hard reset, which forces all registers to their reset state.

78.3.4 Interrupts and DMA Requests

For each XBAR_OUT[n] output with XBAR_CTRL register support, DMA or interrupt functionality can be enabled by setting the corresponding CTRL[DENn] or CTRL[IENn] to 1. DENn and IENn should not be set to 1 at the same time for the same output XBAR_OUT[n].

Setting DENn to 1 enables DMA functionality for XBAR_OUT[n]. When DMA functionality is enabled, the output DMA_REQ[n] reflects the value of STSn. Thus the DMA request asserts when the edge specified by EDGEn is detected on XBAR_OUT[n]. Also, a rising edge on DMA_ACK[n] sets STSn to zero and thus clears the DMA request when the data transfer is completed. When DEN is 0, DMA_REQ[n] is held low and DMA_ACK[n] is ignored.

Setting IENn to 1 enables interrupt functionality for XBAR_OUT[n]. When interrupt functionality is enabled, the output INT_REQ[n] reflects the value of STSn. Thus the interrupt request asserts when the edge specified by EDGEn is detected on XBAR_OUT[n]. The interrupt request is cleared by writing a 1 to STSn. When IENn is 0, INT_REQ[n] is held low.

78.4 External Signals

The following table summarizes the module's external signals.

Table 1100. Control Signal Properties

Name	I/O Type	Function	Reset State	Notes
XBAR_OUT[0:M-1]	O	Mux Outputs	XBAR_IN[0]	
XBAR_IN[0:N-1]	I	Mux Inputs	*	
DMA_REQ[0:P-1]	O	DMA request	0	
INT_REQ[0:P-1]	O	Interrupt request	0	
DMA_ACK[0:P-1]	I	DMA acknowledge	0	

At reset, each output XBAR_OUT[n] contains the reset value of the signal driving XBAR_IN[0].

78.4.1 XBAR_OUT[0:M-1] - MUX Outputs

This is a one-dimensional array of the mux outputs. The value on each output XBAR_OUT[n] is determined by the setting of the corresponding memory mapped register SELn such that XBAR_OUT[n] = XBAR_IN[SELn].

78.4.2 XBAR_IN[0:N-1] - MUX Inputs

This is a one-dimensional array consisting of the inputs shared by each mux. All muxes share the same inputs in the same order.

78.4.3 DMA_REQ[0:P-1] - DMA Request Output(s)

DMA_REQ[n] is a DMA request to the DMA controller.

78.4.4 DMA_ACK[0:P-1] - DMA Acknowledge Input(s)

DMA_ACK[n] is a DMA acknowledge input from the DMA controller.

78.4.5 INT_REQ[0:P-1] - Interrupt Request Output(s)

INT_REQ[n] is an interrupt request output to the interrupt controller.

78.5 Memory Map and Register Descriptions

78.5.1 XBAR register descriptions

The XBAR module has select registers and control registers.

In the XBAR select registers, the SELn fields select which of the shared inputs (XBAR_IN) is muxed to each mux output (XBAR_OUT). There is one SELn field per mux and therefore one per XBAR_OUT output. Crossbar output XBAR_OUT[n] presents the signal chosen by XBAR_IN[SELn]. In a select register (SELy for example), the lower 8 bits represent the input for XBAR_OUT[2y] and the higher 8 bits represent the input for XBAR_OUT[2y+1].

The actual signals connected to XBAR_IN and XBAR_OUT are application specific and are described in the XBAR Assignments tables.

The XBAR control registers configure edge detection, interrupt, and DMA features for a subset of the XBAR_OUT outputs.

78.5.1.1 XBAR memory map

XBAR1 base address: 4275_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Crossbar Select Register (SEL0)	16	RW	0000h
2h	Crossbar Select Register (SEL1)	16	RW	0000h
4h	Crossbar Select Register (SEL2)	16	RW	0000h
6h	Crossbar Select Register (SEL3)	16	RW	0000h
8h	Crossbar Select Register (SEL4)	16	RW	0000h
Ah	Crossbar Select Register (SEL5)	16	RW	0000h
Ch	Crossbar Select Register (SEL6)	16	RW	0000h
Eh	Crossbar Select Register (SEL7)	16	RW	0000h
10h	Crossbar Select Register (SEL8)	16	RW	0000h
12h	Crossbar Select Register (SEL9)	16	RW	0000h
14h	Crossbar Select Register (SEL10)	16	RW	0000h
16h	Crossbar Select Register (SEL11)	16	RW	0000h
18h	Crossbar Select Register (SEL12)	16	RW	0000h
1Ah	Crossbar Select Register (SEL13)	16	RW	0000h
1Ch	Crossbar Select Register (SEL14)	16	RW	0000h
1Eh	Crossbar Select Register (SEL15)	16	RW	0000h
20h	Crossbar Select Register (SEL16)	16	RW	0000h
22h	Crossbar Select Register (SEL17)	16	RW	0000h
24h	Crossbar Select Register (SEL18)	16	RW	0000h
26h	Crossbar Select Register (SEL19)	16	RW	0000h
28h	Crossbar Select Register (SEL20)	16	RW	0000h
2Ah	Crossbar Select Register (SEL21)	16	RW	0000h
2Ch	Crossbar Select Register (SEL22)	16	RW	0000h
2Eh	Crossbar Select Register (SEL23)	16	RW	0000h
30h	Crossbar Select Register (SEL24)	16	RW	0000h
32h	Crossbar Select Register (SEL25)	16	RW	0000h
34h	Crossbar Select Register (SEL26)	16	RW	0000h
36h	Crossbar Select Register (SEL27)	16	RW	0000h
38h	Crossbar Select Register (SEL28)	16	RW	0000h
3Ah	Crossbar Select Register (SEL29)	16	RW	0000h
3Ch	Crossbar Select Register (SEL30)	16	RW	0000h
3Eh	Crossbar Select Register (SEL31)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
40h	Crossbar Select Register (SEL32)	16	RW	0000h
42h	Crossbar Select Register (SEL33)	16	RW	0000h
44h	Crossbar Select Register (SEL34)	16	RW	0000h
46h	Crossbar Select Register (SEL35)	16	RW	0000h
48h	Crossbar Select Register (SEL36)	16	RW	0000h
4Ah	Crossbar Select Register (SEL37)	16	RW	0000h
4Ch	Crossbar Select Register (SEL38)	16	RW	0000h
4Eh	Crossbar Select Register (SEL39)	16	RW	0000h
50h	Crossbar Select Register (SEL40)	16	RW	0000h
52h	Crossbar Select Register (SEL41)	16	RW	0000h
54h	Crossbar Select Register (SEL42)	16	RW	0000h
56h	Crossbar Select Register (SEL43)	16	RW	0000h
58h	Crossbar Select Register (SEL44)	16	RW	0000h
5Ah	Crossbar Select Register (SEL45)	16	RW	0000h
5Ch	Crossbar Select Register (SEL46)	16	RW	0000h
5Eh	Crossbar Select Register (SEL47)	16	RW	0000h
60h	Crossbar Select Register (SEL48)	16	RW	0000h
62h	Crossbar Select Register (SEL49)	16	RW	0000h
64h	Crossbar Select Register (SEL50)	16	RW	0000h
66h	Crossbar Select Register (SEL51)	16	RW	0000h
68h	Crossbar Select Register (SEL52)	16	RW	0000h
6Ah	Crossbar Select Register (SEL53)	16	RW	0000h
6Ch	Crossbar Select Register (SEL54)	16	RW	0000h
6Eh	Crossbar Select Register (SEL55)	16	RW	0000h
70h	Crossbar Select Register (SEL56)	16	RW	0000h
72h	Crossbar Select Register (SEL57)	16	RW	0000h
74h	Crossbar Select Register (SEL58)	16	RW	0000h
76h	Crossbar Select Register (SEL59)	16	RW	0000h
78h	Crossbar Select Register (SEL60)	16	RW	0000h
7Ah	Crossbar Select Register (SEL61)	16	RW	0000h
7Ch	Crossbar Select Register (SEL62)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
7Eh	Crossbar Select Register (SEL63)	16	RW	0000h
80h	Crossbar Select Register (SEL64)	16	RW	0000h
82h	Crossbar Select Register (SEL65)	16	RW	0000h
84h	Crossbar Select Register (SEL66)	16	RW	0000h
86h	Crossbar Select Register (SEL67)	16	RW	0000h
88h	Crossbar Select Register (SEL68)	16	RW	0000h
8Ah	Crossbar Select Register (SEL69)	16	RW	0000h
8Ch	Crossbar Select Register (SEL70)	16	RW	0000h
8Eh	Crossbar Select Register (SEL71)	16	RW	0000h
90h	Crossbar Select Register (SEL72)	16	RW	0000h
92h	Crossbar Select Register (SEL73)	16	RW	0000h
94h	Crossbar Select Register (SEL74)	16	RW	0000h
96h	Crossbar Select Register (SEL75)	16	RW	0000h
98h	Crossbar Select Register (SEL76)	16	RW	0000h
9Ah	Crossbar Select Register (SEL77)	16	RW	0000h
9Ch	Crossbar Select Register (SEL78)	16	RW	0000h
9Eh	Crossbar Select Register (SEL79)	16	RW	0000h
A0h	Crossbar Select Register (SEL80)	16	RW	0000h
A2h	Crossbar Select Register (SEL81)	16	RW	0000h
A4h	Crossbar Select Register (SEL82)	16	RW	0000h
A6h	Crossbar Select Register (SEL83)	16	RW	0000h
A8h	Crossbar Select Register (SEL84)	16	RW	0000h
AAh	Crossbar Select Register (SEL85)	16	RW	0000h
ACh	Crossbar Select Register (SEL86)	16	RW	0000h
AEh	Crossbar Select Register (SEL87)	16	RW	0000h
B0h	Crossbar Select Register (SEL88)	16	RW	0000h
B2h	Crossbar Select Register (SEL89)	16	RW	0000h
B4h	Crossbar Select Register (SEL90)	16	RW	0000h
B6h	Crossbar Select Register (SEL91)	16	RW	0000h
B8h	Crossbar Select Register (SEL92)	16	RW	0000h
BAh	Crossbar Select Register (SEL93)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

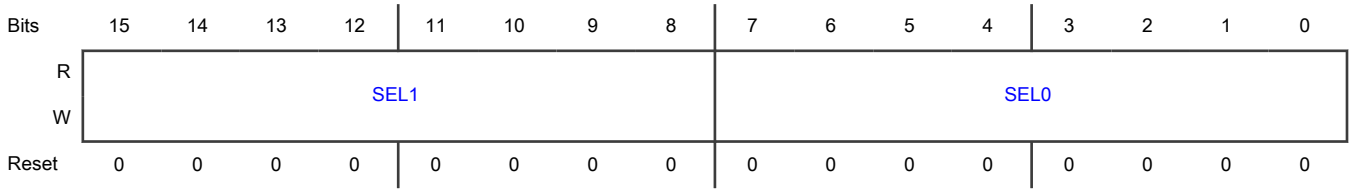
Offset	Register	Width (In bits)	Access	Reset value
BCh	Crossbar Select Register (SEL94)	16	RW	0000h
BEh	Crossbar Select Register (SEL95)	16	RW	0000h
C0h	Crossbar Select Register (SEL96)	16	RW	0000h
C2h	Crossbar Select Register (SEL97)	16	RW	0000h
C4h	Crossbar Select Register (SEL98)	16	RW	0000h
C6h	Crossbar Select Register (SEL99)	16	RW	0000h
C8h	Crossbar Select Register (SEL100)	16	RW	0000h
CAh	Crossbar Select Register (SEL101)	16	RW	0000h
CCh	Crossbar Select Register (SEL102)	16	RW	0000h
CEh	Crossbar Select Register (SEL103)	16	RW	0000h
D0h	Crossbar Select Register (SEL104)	16	RW	0000h
D2h	Crossbar Select Register (SEL105)	16	RW	0000h
D4h	Crossbar Select Register (SEL106)	16	RW	0000h
D6h	Crossbar Select Register (SEL107)	16	RW	0000h
D8h	Crossbar Select Register (SEL108)	16	RW	0000h
DAh	Crossbar Select Register (SEL109)	16	RW	0000h
DCh	Crossbar Select Register (SEL110)	16	RW	0000h
DEh	Crossbar Control Register (CTRL0)	16	RW	0000h
E0h	Crossbar Control Register (CTRL1)	16	RW	0000h

78.5.1.2 Crossbar Select Register (SEL0)

Offset

Register	Offset
SEL0	0h

Diagram



Fields

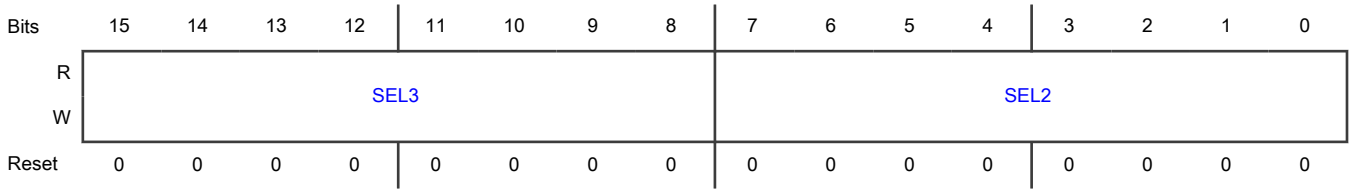
Field	Function
15-8: SEL1 7-0: SEL0	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.3 Crossbar Select Register (SEL1)

Offset

Register	Offset
SEL1	2h

Diagram



Fields

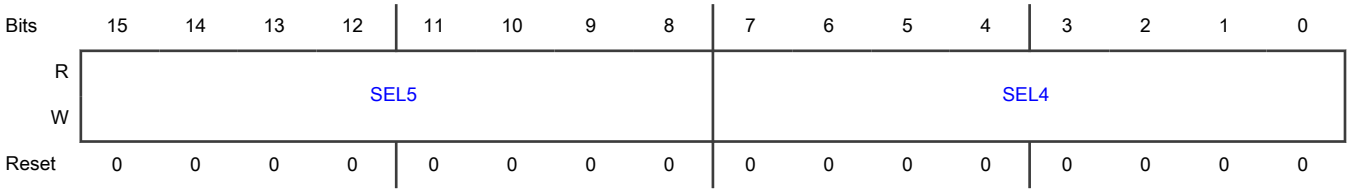
Field	Function
15-8: SEL3 7-0: SEL2	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.4 Crossbar Select Register (SEL2)

Offset

Register	Offset
SEL2	4h

Diagram



Fields

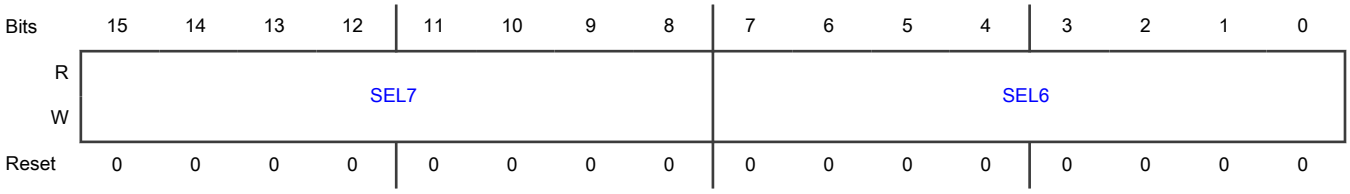
Field	Function
15-8: SEL5 7-0: SEL4	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.5 Crossbar Select Register (SEL3)

Offset

Register	Offset
SEL3	6h

Diagram



Fields

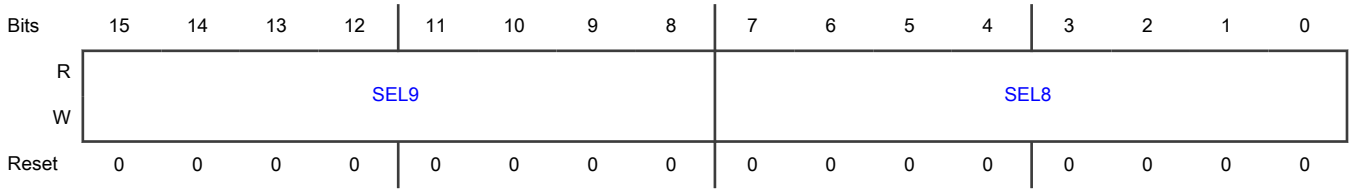
Field	Function
15-8: SEL7 7-0: SEL6	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.6 Crossbar Select Register (SEL4)

Offset

Register	Offset
SEL4	8h

Diagram



Fields

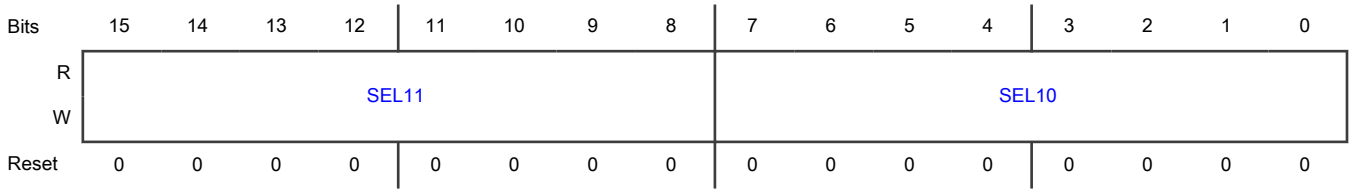
Field	Function
15-8: SEL9 7-0: SEL8	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.7 Crossbar Select Register (SEL5)

Offset

Register	Offset
SEL5	Ah

Diagram



Fields

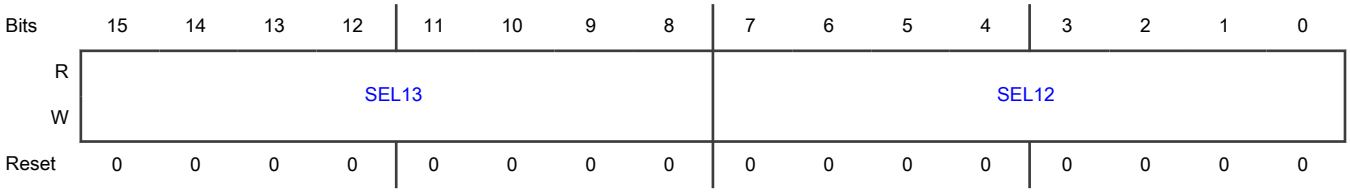
Field	Function
15-8: SEL11 7-0: SEL10	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.8 Crossbar Select Register (SEL6)

Offset

Register	Offset
SEL6	Ch

Diagram



Fields

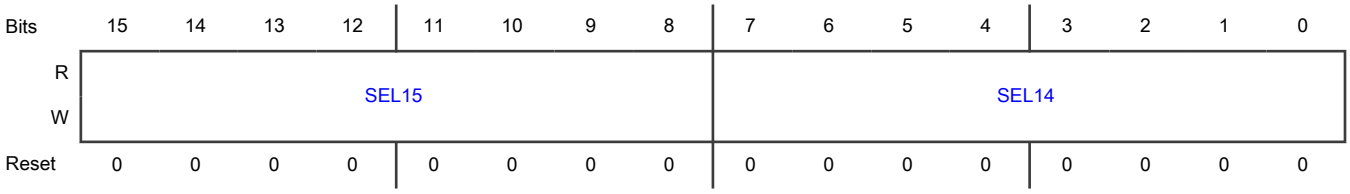
Field	Function
15-8: SEL13	SELn
7-0: SEL12	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.9 Crossbar Select Register (SEL7)

Offset

Register	Offset
SEL7	Eh

Diagram



Fields

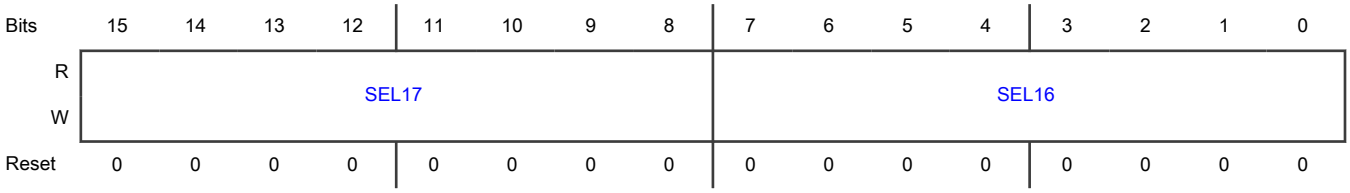
Field	Function
15-8: SEL15	SELn
7-0: SEL14	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.10 Crossbar Select Register (SEL8)

Offset

Register	Offset
SEL8	10h

Diagram



Fields

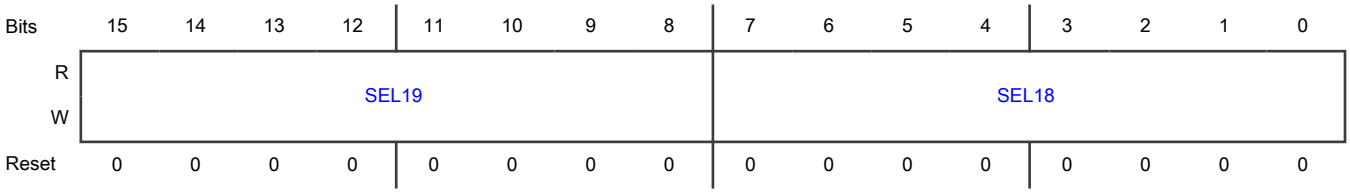
Field	Function
15-8: SEL17	SELn
7-0: SEL16	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.11 Crossbar Select Register (SEL9)

Offset

Register	Offset
SEL9	12h

Diagram



Fields

Field	Function
15-8: SEL19	SELn
7-0: SEL18	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.12 Crossbar Select Register (SEL10)

Offset

Register	Offset
SEL10	14h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SEL21								SEL20							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL21	SELn
7-0: SEL20	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.13 Crossbar Select Register (SEL11)

Offset

Register	Offset
SEL11	16h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SEL23								SEL22							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL23	SELn
7-0: SEL22	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.14 Crossbar Select Register (SEL12)

Offset

Register	Offset
SEL12	18h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL25								SEL24							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL25	SELn
7-0: SEL24	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.15 Crossbar Select Register (SEL13)

Offset

Register	Offset
SEL13	1Ah

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL27								SEL26							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

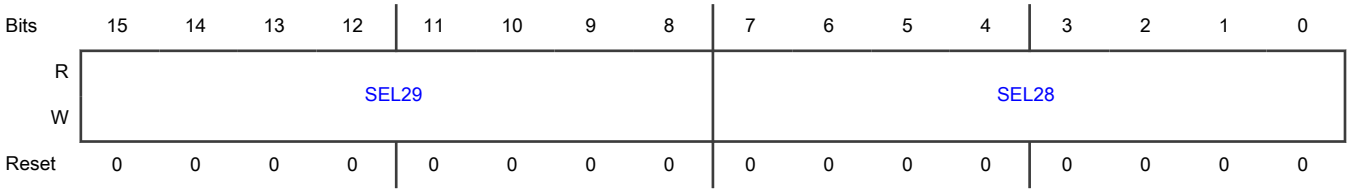
Field	Function
15-8: SEL27	SELn
7-0: SEL26	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.16 Crossbar Select Register (SEL14)

Offset

Register	Offset
SEL14	1Ch

Diagram



Fields

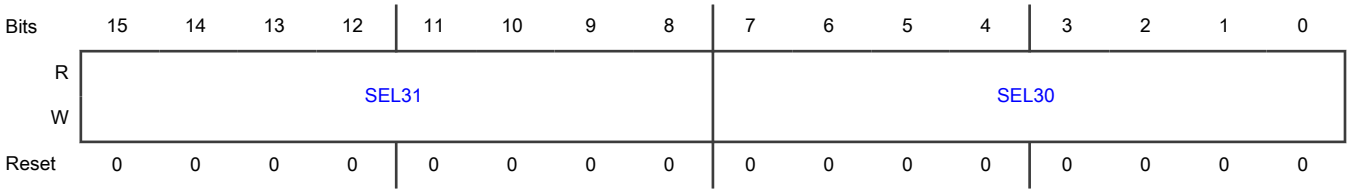
Field	Function
15-8: SEL29	SELn
7-0: SEL28	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.17 Crossbar Select Register (SEL15)

Offset

Register	Offset
SEL15	1Eh

Diagram



Fields

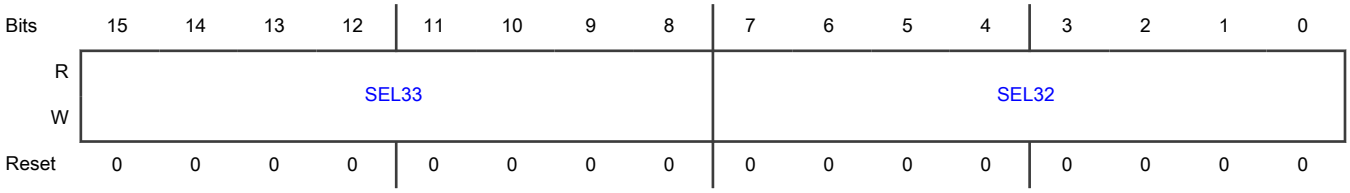
Field	Function
15-8: SEL31	SELn
7-0: SEL30	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.18 Crossbar Select Register (SEL16)

Offset

Register	Offset
SEL16	20h

Diagram



Fields

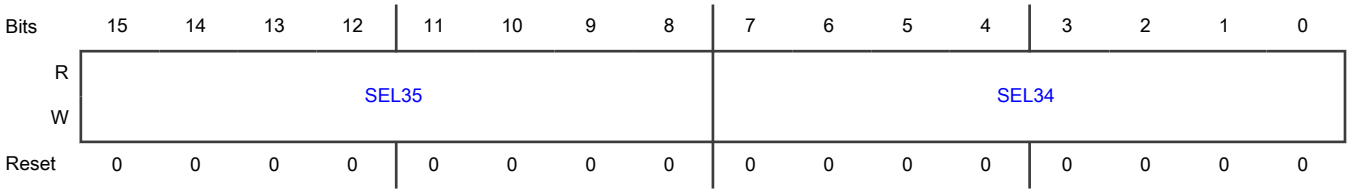
Field	Function
15-8: SEL33	SELn
7-0: SEL32	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.19 Crossbar Select Register (SEL17)

Offset

Register	Offset
SEL17	22h

Diagram



Fields

Field	Function
15-8: SEL35	SELn
7-0: SEL34	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.20 Crossbar Select Register (SEL18)

Offset

Register	Offset
SEL18	24h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL37								SEL36							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL37	SELn
7-0: SEL36	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.21 Crossbar Select Register (SEL19)

Offset

Register	Offset
SEL19	26h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL39								SEL38							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

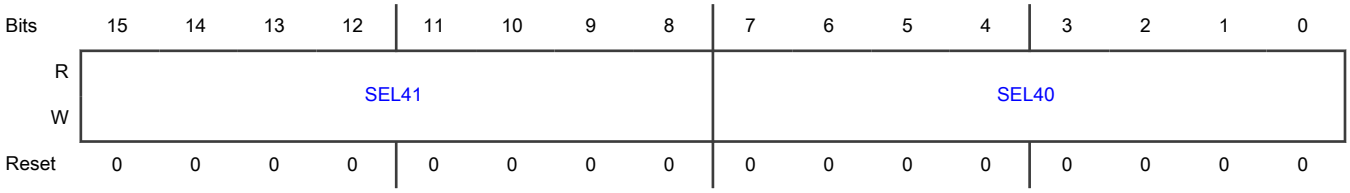
Field	Function
15-8: SEL39	SELn
7-0: SEL38	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.22 Crossbar Select Register (SEL20)

Offset

Register	Offset
SEL20	28h

Diagram



Fields

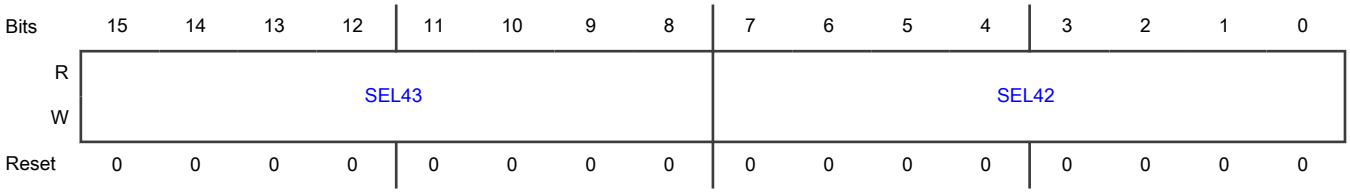
Field	Function
15-8: SEL41	SELn
7-0: SEL40	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.23 Crossbar Select Register (SEL21)

Offset

Register	Offset
SEL21	2Ah

Diagram



Fields

Field	Function
15-8: SEL43	SELn
7-0: SEL42	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.24 Crossbar Select Register (SEL22)

Offset

Register	Offset
SEL22	2Ch

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL45								SEL44							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL45	SELn
7-0: SEL44	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.25 Crossbar Select Register (SEL23)

Offset

Register	Offset
SEL23	2Eh

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL47								SEL46							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

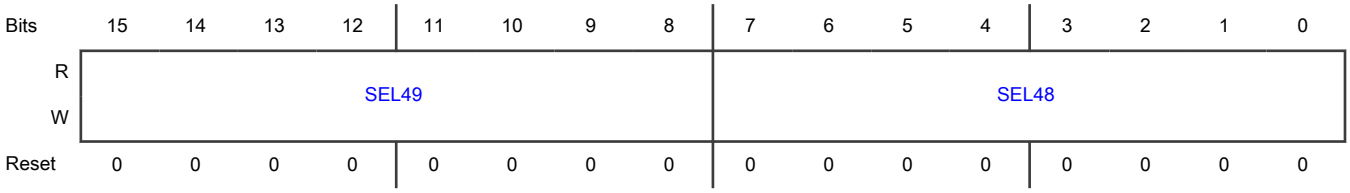
Field	Function
15-8: SEL47	SELn
7-0: SEL46	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.26 Crossbar Select Register (SEL24)

Offset

Register	Offset
SEL24	30h

Diagram



Fields

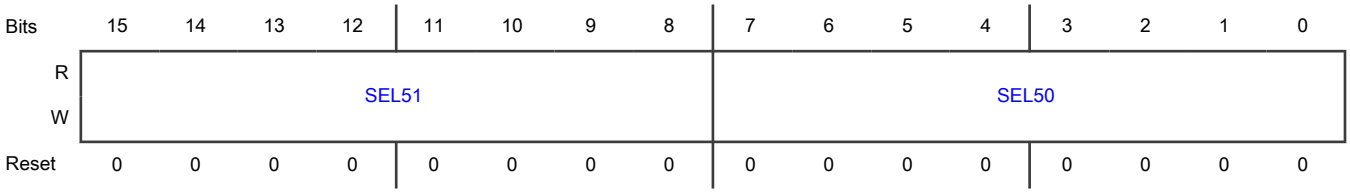
Field	Function
15-8: SEL49	SELn
7-0: SEL48	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.27 Crossbar Select Register (SEL25)

Offset

Register	Offset
SEL25	32h

Diagram



Fields

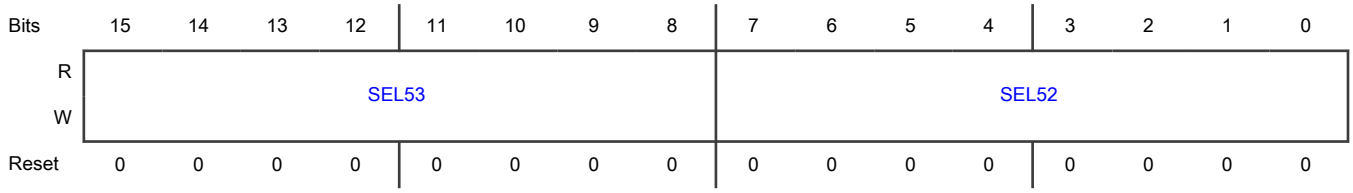
Field	Function
15-8: SEL51	SELn
7-0: SEL50	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.28 Crossbar Select Register (SEL26)

Offset

Register	Offset
SEL26	34h

Diagram



Fields

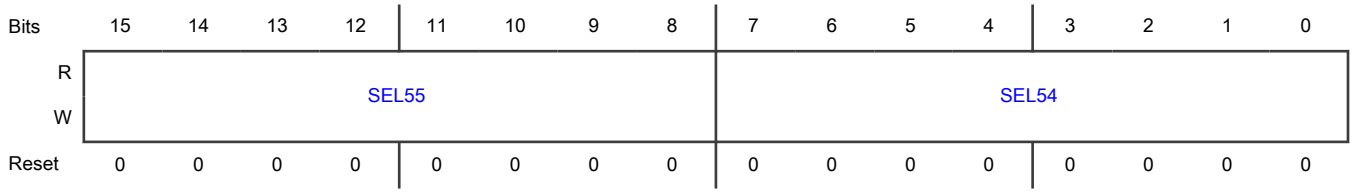
Field	Function
15-8: SEL53	SELn
7-0: SEL52	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.29 Crossbar Select Register (SEL27)

Offset

Register	Offset
SEL27	36h

Diagram



Fields

Field	Function
15-8: SEL55	SELn
7-0: SEL54	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.30 Crossbar Select Register (SEL28)

Offset

Register	Offset
SEL28	38h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL57								SEL56							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL57	SELn
7-0: SEL56	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.31 Crossbar Select Register (SEL29)

Offset

Register	Offset
SEL29	3Ah

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL59								SEL58							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

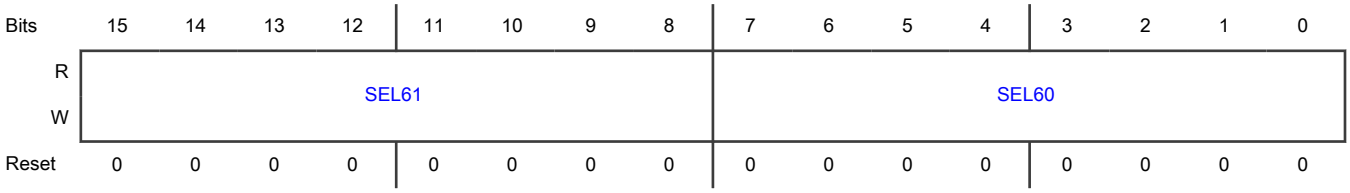
Field	Function
15-8: SEL59	SELn
7-0: SEL58	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.32 Crossbar Select Register (SEL30)

Offset

Register	Offset
SEL30	3Ch

Diagram



Fields

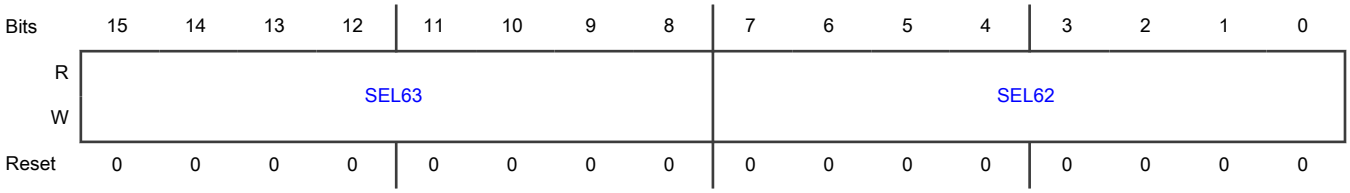
Field	Function
15-8: SEL61	SELn
7-0: SEL60	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.33 Crossbar Select Register (SEL31)

Offset

Register	Offset
SEL31	3Eh

Diagram



Fields

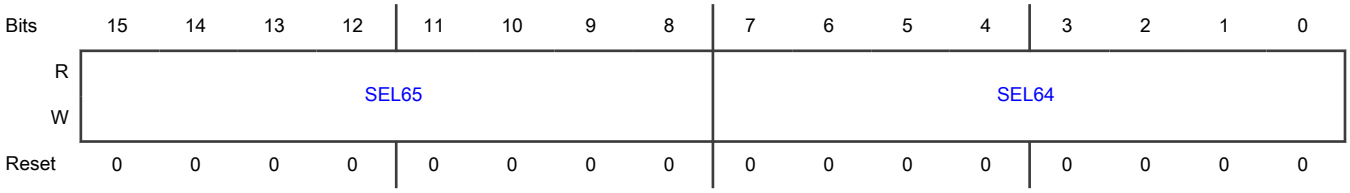
Field	Function
15-8: SEL63	SELn
7-0: SEL62	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.34 Crossbar Select Register (SEL32)

Offset

Register	Offset
SEL32	40h

Diagram



Fields

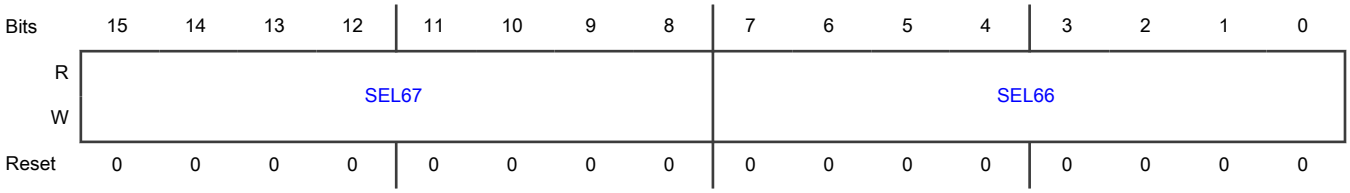
Field	Function
15-8: SEL65	SELn
7-0: SEL64	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.35 Crossbar Select Register (SEL33)

Offset

Register	Offset
SEL33	42h

Diagram



Fields

Field	Function
15-8: SEL67	SELn
7-0: SEL66	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.36 Crossbar Select Register (SEL34)

Offset

Register	Offset
SEL34	44h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL69								SEL68							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL69	SELn
7-0: SEL68	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.37 Crossbar Select Register (SEL35)

Offset

Register	Offset
SEL35	46h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL71								SEL70							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

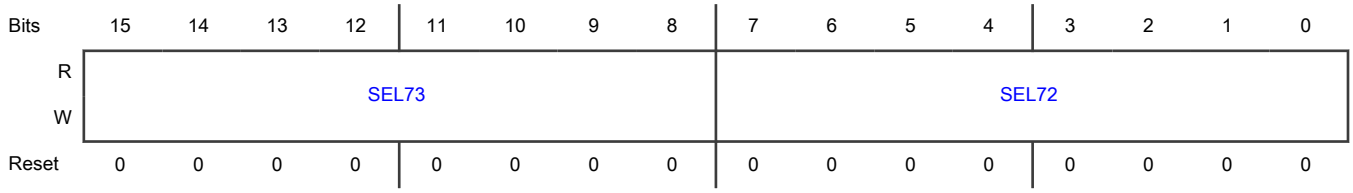
Field	Function
15-8: SEL71	SELn
7-0: SEL70	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.38 Crossbar Select Register (SEL36)

Offset

Register	Offset
SEL36	48h

Diagram



Fields

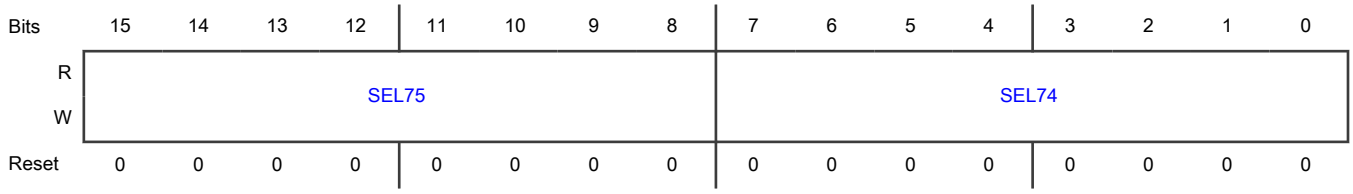
Field	Function
15-8: SEL73	SELn
7-0: SEL72	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.39 Crossbar Select Register (SEL37)

Offset

Register	Offset
SEL37	4Ah

Diagram



Fields

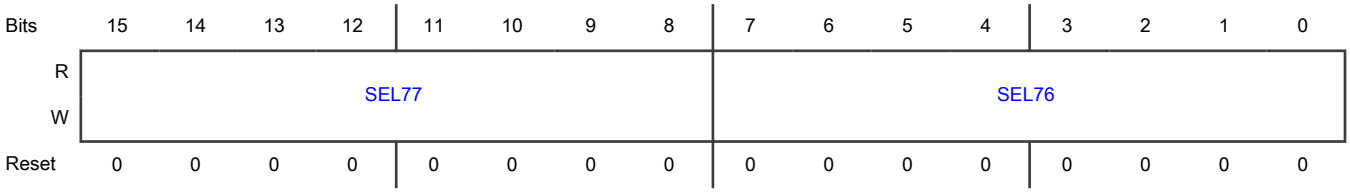
Field	Function
15-8: SEL75	SELn
7-0: SEL74	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.40 Crossbar Select Register (SEL38)

Offset

Register	Offset
SEL38	4Ch

Diagram



Fields

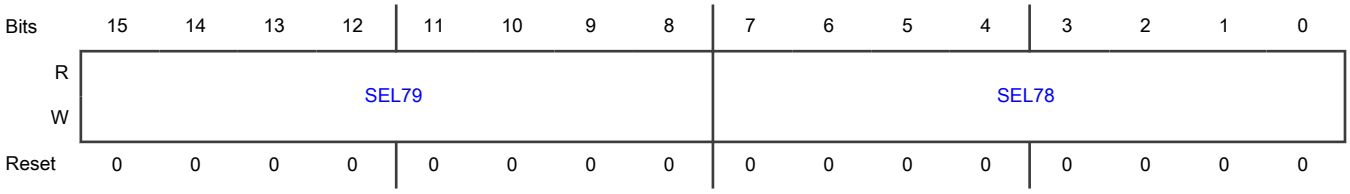
Field	Function
15-8: SEL77	SELn
7-0: SEL76	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.41 Crossbar Select Register (SEL39)

Offset

Register	Offset
SEL39	4Eh

Diagram



Fields

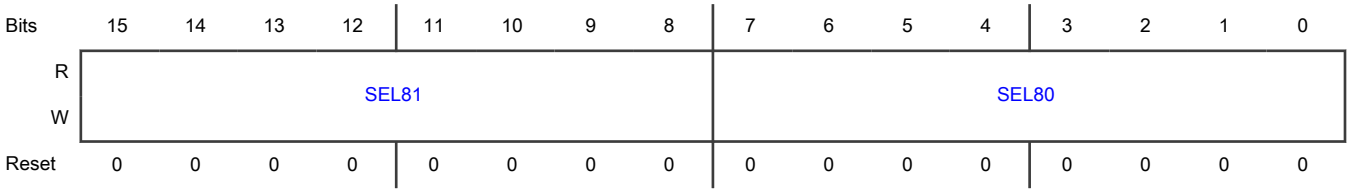
Field	Function
15-8: SEL79	SELn
7-0: SEL78	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.42 Crossbar Select Register (SEL40)

Offset

Register	Offset
SEL40	50h

Diagram



Fields

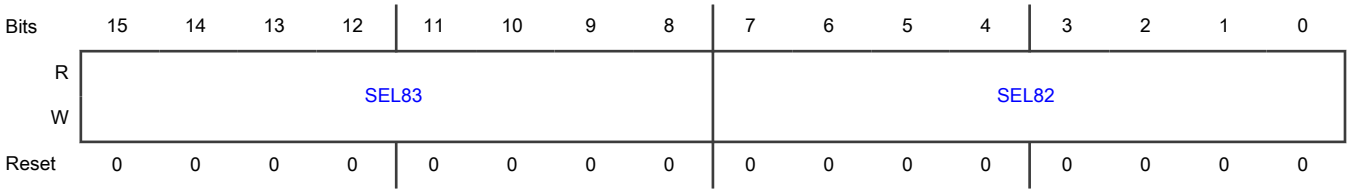
Field	Function
15-8: SEL81	SELn
7-0: SEL80	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.43 Crossbar Select Register (SEL41)

Offset

Register	Offset
SEL41	52h

Diagram



Fields

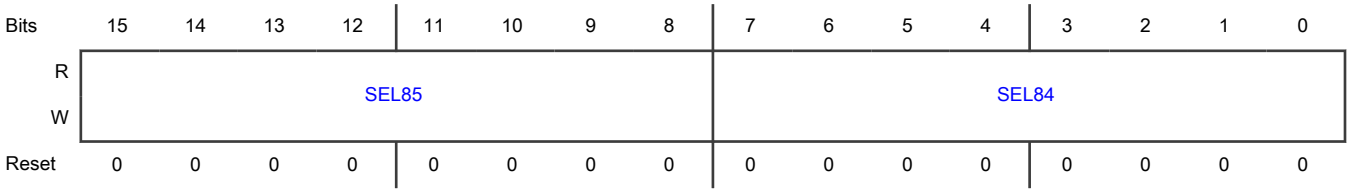
Field	Function
15-8: SEL83	SELn
7-0: SEL82	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.44 Crossbar Select Register (SEL42)

Offset

Register	Offset
SEL42	54h

Diagram



Fields

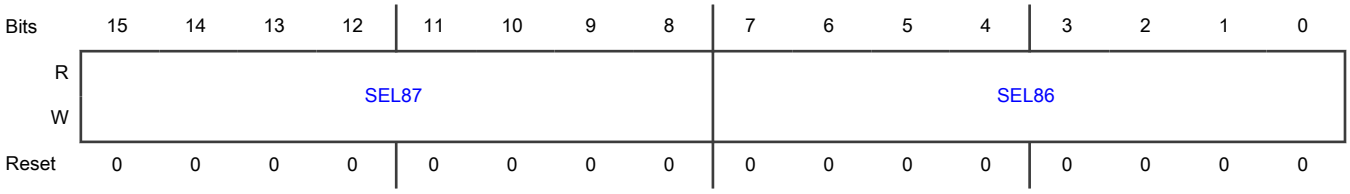
Field	Function
15-8: SEL85	SELn
7-0: SEL84	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.45 Crossbar Select Register (SEL43)

Offset

Register	Offset
SEL43	56h

Diagram



Fields

Field	Function
15-8: SEL87	SELn
7-0: SEL86	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.46 Crossbar Select Register (SEL44)

Offset

Register	Offset
SEL44	58h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL89								SEL88							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL89	SELn
7-0: SEL88	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.47 Crossbar Select Register (SEL45)

Offset

Register	Offset
SEL45	5Ah

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL91								SEL90							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

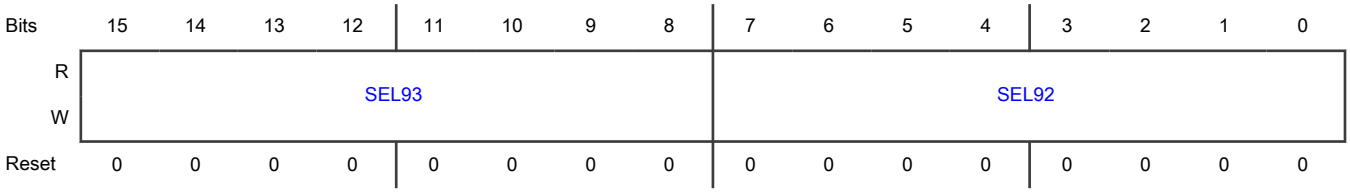
Field	Function
15-8: SEL91	SELn
7-0: SEL90	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.48 Crossbar Select Register (SEL46)

Offset

Register	Offset
SEL46	5Ch

Diagram



Fields

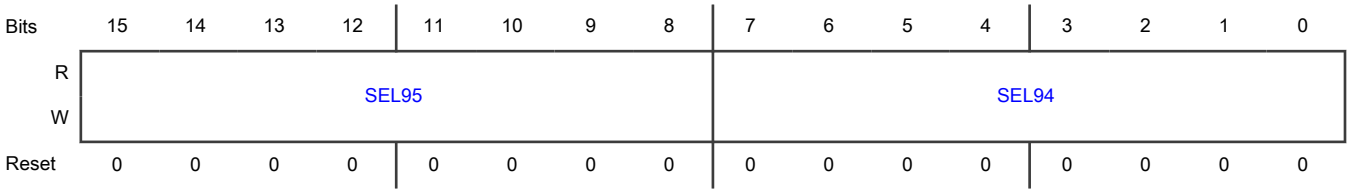
Field	Function
15-8: SEL93	SELn
7-0: SEL92	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.49 Crossbar Select Register (SEL47)

Offset

Register	Offset
SEL47	5Eh

Diagram



Fields

Field	Function
15-8: SEL95	SELn
7-0: SEL94	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.50 Crossbar Select Register (SEL48)

Offset

Register	Offset
SEL48	60h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL97								SEL96							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL97	SELn
7-0: SEL96	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.51 Crossbar Select Register (SEL49)

Offset

Register	Offset
SEL49	62h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL99								SEL98							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL99	SELn
7-0: SEL98	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.52 Crossbar Select Register (SEL50)

Offset

Register	Offset
SEL50	64h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL101								SEL100							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL101	SELn
7-0: SEL100	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.53 Crossbar Select Register (SEL51)

Offset

Register	Offset
SEL51	66h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL103								SEL102							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL103	SELn
7-0: SEL102	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.54 Crossbar Select Register (SEL52)

Offset

Register	Offset
SEL52	68h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL105								SEL104							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL105	SELn
7-0: SEL104	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.55 Crossbar Select Register (SEL53)

Offset

Register	Offset
SEL53	6Ah

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL107								SEL106							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL107	SELn
7-0: SEL106	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.56 Crossbar Select Register (SEL54)

Offset

Register	Offset
SEL54	6Ch

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL109								SEL108							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL109 7-0: SEL108	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.57 Crossbar Select Register (SEL55)

Offset

Register	Offset
SEL55	6Eh

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL111								SEL110							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

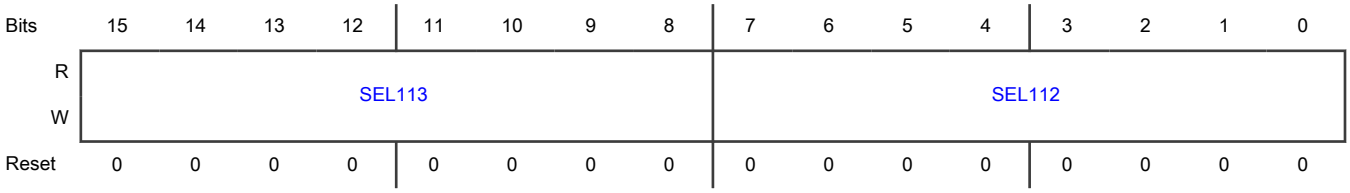
Field	Function
15-8: SEL111 7-0: SEL110	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.58 Crossbar Select Register (SEL56)

Offset

Register	Offset
SEL56	70h

Diagram



Fields

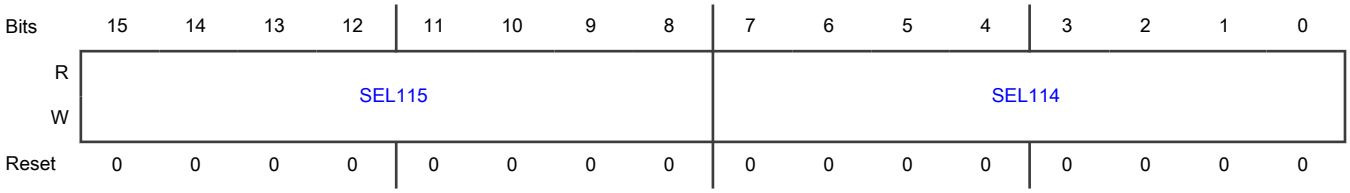
Field	Function
15-8: SEL113 7-0: SEL112	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.59 Crossbar Select Register (SEL57)

Offset

Register	Offset
SEL57	72h

Diagram



Fields

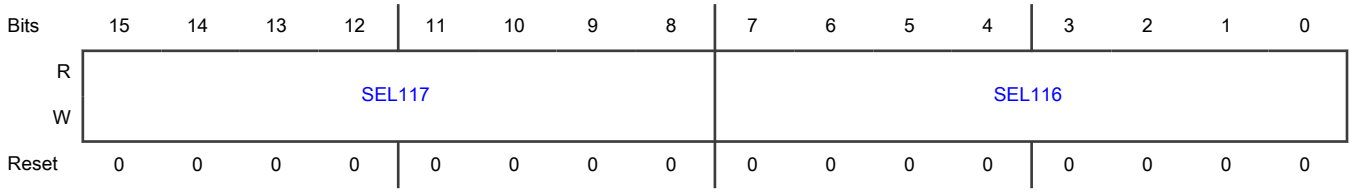
Field	Function
15-8: SEL115 7-0: SEL114	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.60 Crossbar Select Register (SEL58)

Offset

Register	Offset
SEL58	74h

Diagram



Fields

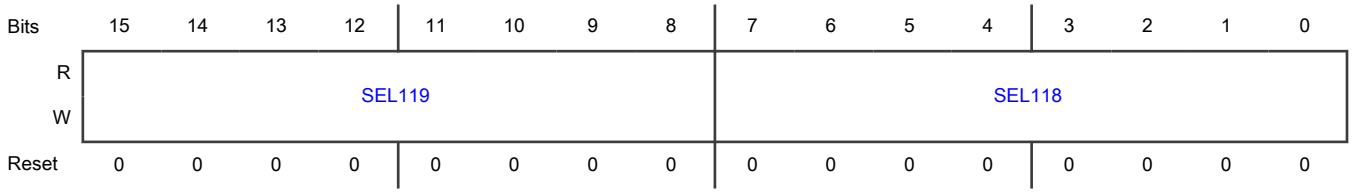
Field	Function
15-8: SEL117 7-0: SEL116	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.61 Crossbar Select Register (SEL59)

Offset

Register	Offset
SEL59	76h

Diagram



Fields

Field	Function
15-8: SEL119 7-0: SEL118	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.62 Crossbar Select Register (SEL60)

Offset

Register	Offset
SEL60	78h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL121								SEL120							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL121	SELn
7-0: SEL120	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.63 Crossbar Select Register (SEL61)

Offset

Register	Offset
SEL61	7Ah

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL123								SEL122							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

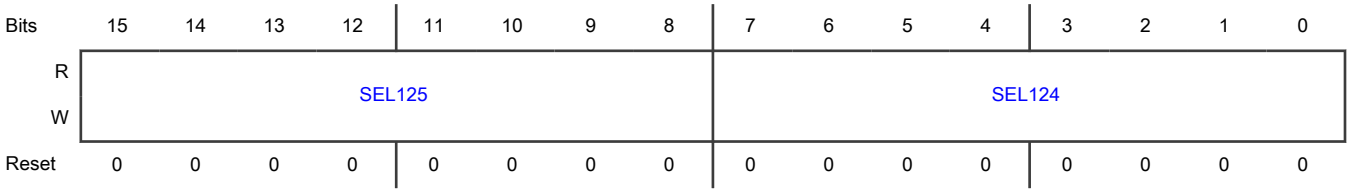
Field	Function
15-8: SEL123	SELn
7-0: SEL122	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.64 Crossbar Select Register (SEL62)

Offset

Register	Offset
SEL62	7Ch

Diagram



Fields

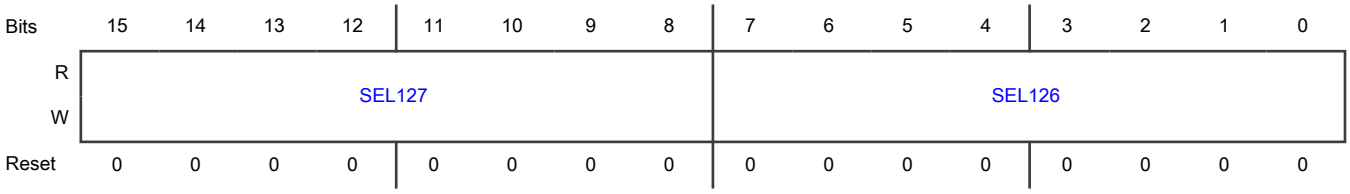
Field	Function
15-8: SEL125 7-0: SEL124	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.65 Crossbar Select Register (SEL63)

Offset

Register	Offset
SEL63	7Eh

Diagram



Fields

Field	Function
15-8: SEL127 7-0: SEL126	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.66 Crossbar Select Register (SEL64)

Offset

Register	Offset
SEL64	80h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL129								SEL128							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL129	SELn
7-0: SEL128	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.67 Crossbar Select Register (SEL65)

Offset

Register	Offset
SEL65	82h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL131								SEL130							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

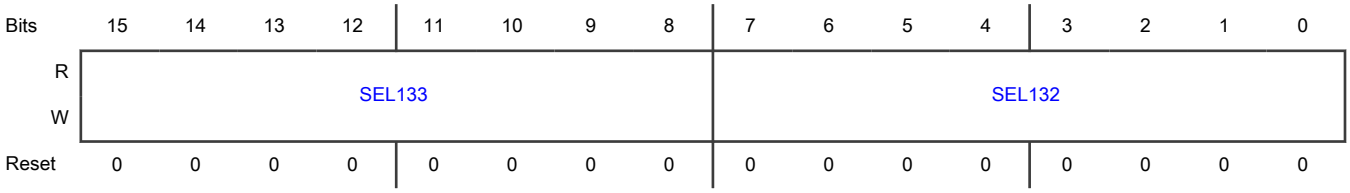
Field	Function
15-8: SEL131	SELn
7-0: SEL130	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.68 Crossbar Select Register (SEL66)

Offset

Register	Offset
SEL66	84h

Diagram



Fields

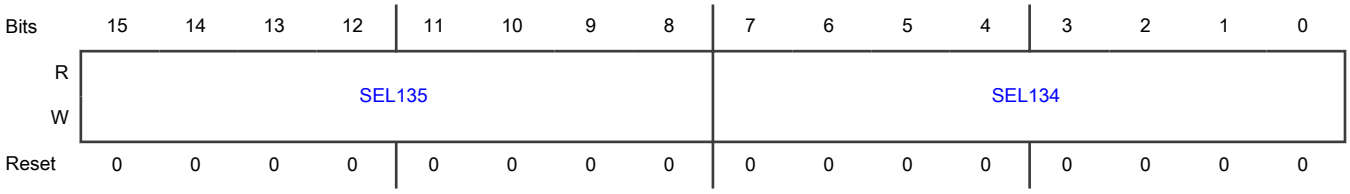
Field	Function
15-8: SEL133	SELn
7-0: SEL132	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.69 Crossbar Select Register (SEL67)

Offset

Register	Offset
SEL67	86h

Diagram



Fields

Field	Function
15-8: SEL135	SELn
7-0: SEL134	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.70 Crossbar Select Register (SEL68)

Offset

Register	Offset
SEL68	88h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL137								SEL136							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL137 7-0: SEL136	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.71 Crossbar Select Register (SEL69)

Offset

Register	Offset
SEL69	8Ah

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL139								SEL138							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

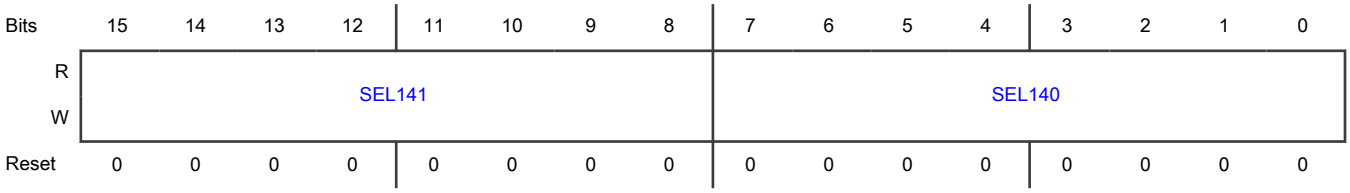
Field	Function
15-8: SEL139 7-0: SEL138	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.72 Crossbar Select Register (SEL70)

Offset

Register	Offset
SEL70	8Ch

Diagram



Fields

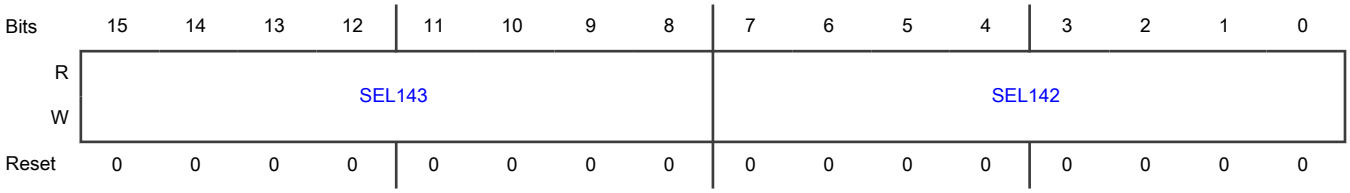
Field	Function
15-8: SEL141 7-0: SEL140	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.73 Crossbar Select Register (SEL71)

Offset

Register	Offset
SEL71	8Eh

Diagram



Fields

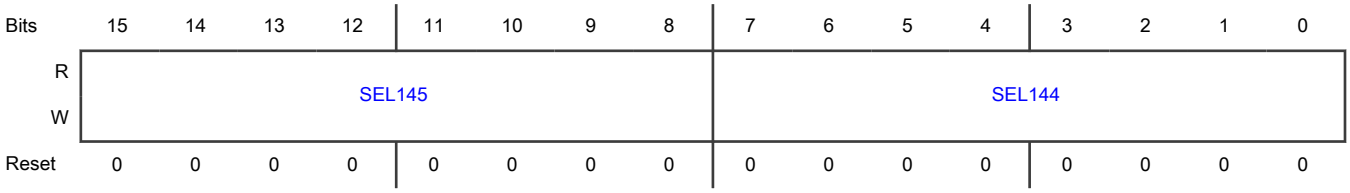
Field	Function
15-8: SEL143 7-0: SEL142	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.74 Crossbar Select Register (SEL72)

Offset

Register	Offset
SEL72	90h

Diagram



Fields

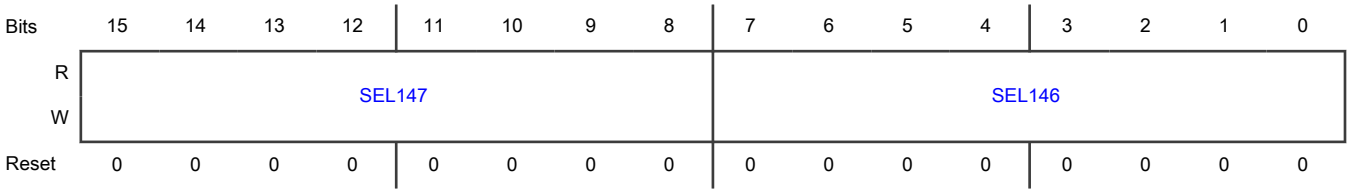
Field	Function
15-8: SEL145 7-0: SEL144	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.75 Crossbar Select Register (SEL73)

Offset

Register	Offset
SEL73	92h

Diagram



Fields

Field	Function
15-8: SEL147 7-0: SEL146	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.76 Crossbar Select Register (SEL74)

Offset

Register	Offset
SEL74	94h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL149								SEL148							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL149	SELn
7-0: SEL148	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.77 Crossbar Select Register (SEL75)

Offset

Register	Offset
SEL75	96h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL151								SEL150							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

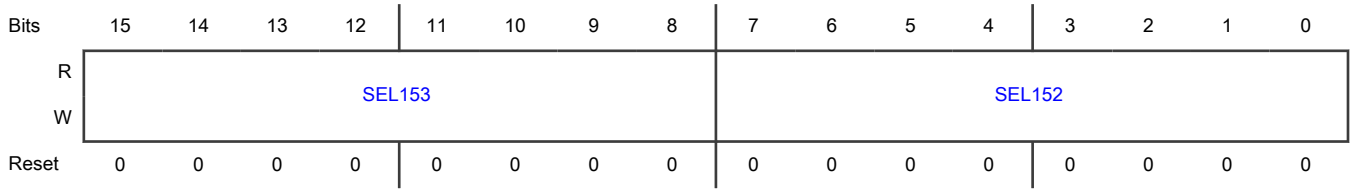
Field	Function
15-8: SEL151	SELn
7-0: SEL150	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.78 Crossbar Select Register (SEL76)

Offset

Register	Offset
SEL76	98h

Diagram



Fields

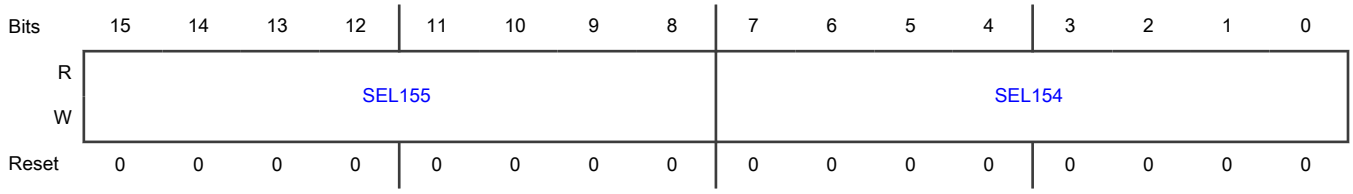
Field	Function
15-8: SEL153 7-0: SEL152	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.79 Crossbar Select Register (SEL77)

Offset

Register	Offset
SEL77	9Ah

Diagram



Fields

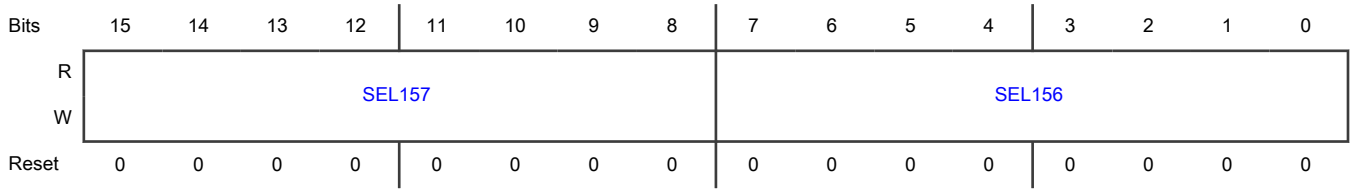
Field	Function
15-8: SEL155 7-0: SEL154	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.80 Crossbar Select Register (SEL78)

Offset

Register	Offset
SEL78	9Ch

Diagram



Fields

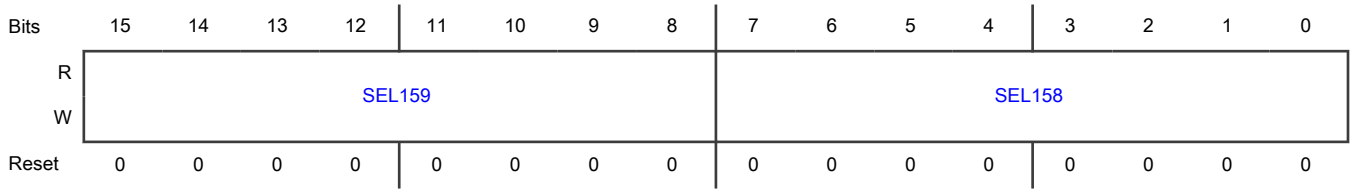
Field	Function
15-8: SEL157 7-0: SEL156	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.81 Crossbar Select Register (SEL79)

Offset

Register	Offset
SEL79	9Eh

Diagram



Fields

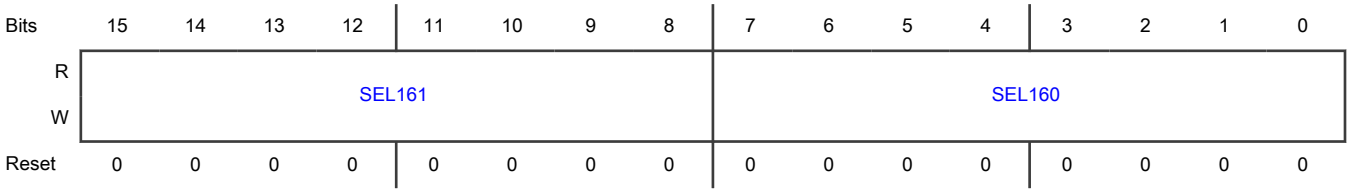
Field	Function
15-8: SEL159 7-0: SEL158	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.82 Crossbar Select Register (SEL80)

Offset

Register	Offset
SEL80	A0h

Diagram



Fields

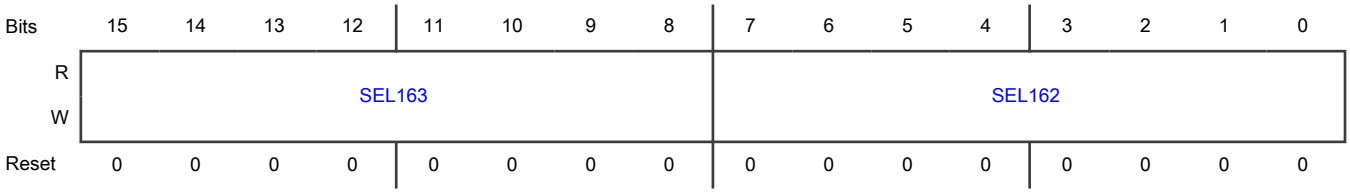
Field	Function
15-8: SEL161 7-0: SEL160	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.83 Crossbar Select Register (SEL81)

Offset

Register	Offset
SEL81	A2h

Diagram



Fields

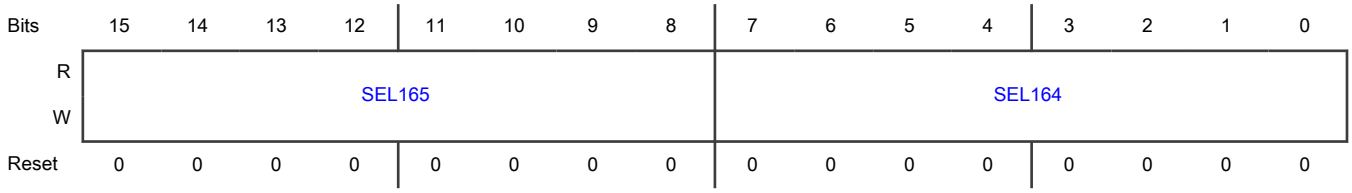
Field	Function
15-8: SEL163 7-0: SEL162	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.84 Crossbar Select Register (SEL82)

Offset

Register	Offset
SEL82	A4h

Diagram



Fields

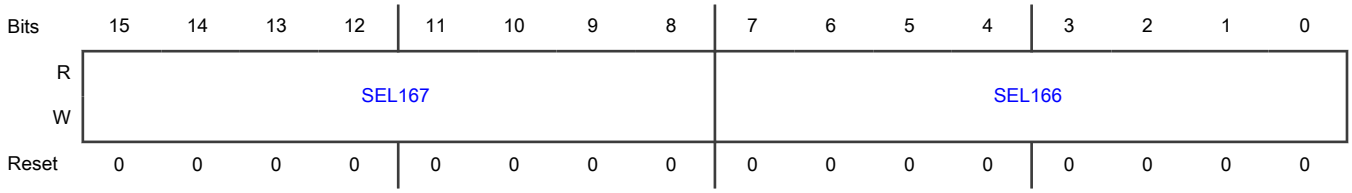
Field	Function
15-8: SEL165 7-0: SEL164	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.85 Crossbar Select Register (SEL83)

Offset

Register	Offset
SEL83	A6h

Diagram



Fields

Field	Function
15-8: SEL167 7-0: SEL166	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.86 Crossbar Select Register (SEL84)

Offset

Register	Offset
SEL84	A8h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL169								SEL168							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL169 7-0: SEL168	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.87 Crossbar Select Register (SEL85)

Offset

Register	Offset
SEL85	AAh

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL171								SEL170							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

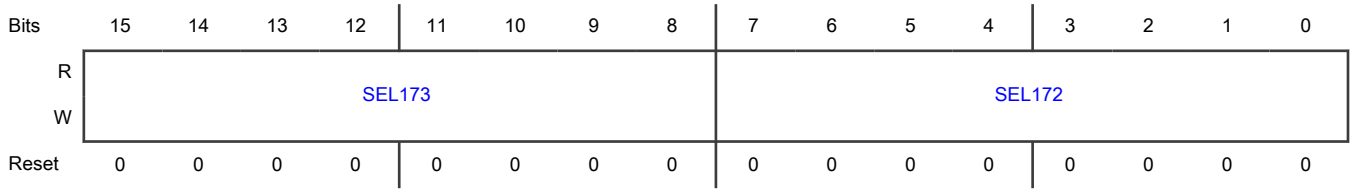
Field	Function
15-8: SEL171 7-0: SEL170	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.88 Crossbar Select Register (SEL86)

Offset

Register	Offset
SEL86	ACh

Diagram



Fields

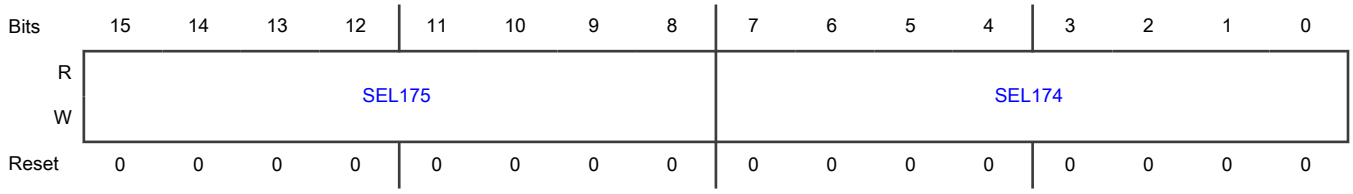
Field	Function
15-8: SEL173 7-0: SEL172	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.89 Crossbar Select Register (SEL87)

Offset

Register	Offset
SEL87	AEh

Diagram



Fields

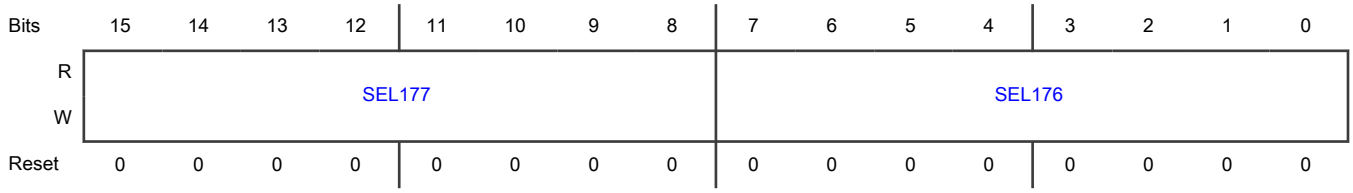
Field	Function
15-8: SEL175 7-0: SEL174	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.90 Crossbar Select Register (SEL88)

Offset

Register	Offset
SEL88	B0h

Diagram



Fields

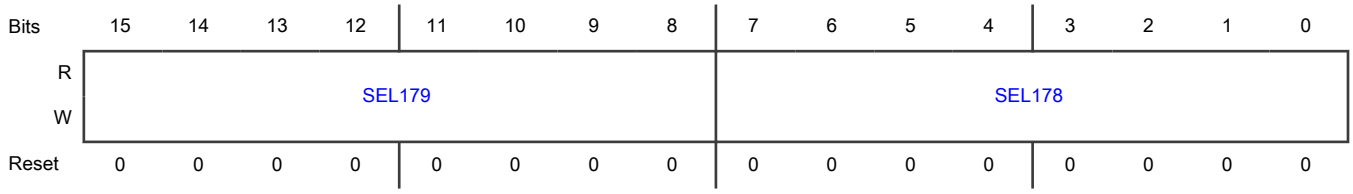
Field	Function
15-8: SEL177 7-0: SEL176	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.91 Crossbar Select Register (SEL89)

Offset

Register	Offset
SEL89	B2h

Diagram



Fields

Field	Function
15-8: SEL179 7-0: SEL178	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.92 Crossbar Select Register (SEL90)

Offset

Register	Offset
SEL90	B4h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL181								SEL180							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL181	SELn
7-0: SEL180	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.93 Crossbar Select Register (SEL91)

Offset

Register	Offset
SEL91	B6h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL183								SEL182							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL183	SELn
7-0: SEL182	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.94 Crossbar Select Register (SEL92)

Offset

Register	Offset
SEL92	B8h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL185								SEL184							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL185	SELn
7-0: SEL184	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.95 Crossbar Select Register (SEL93)

Offset

Register	Offset
SEL93	BAh

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL187								SEL186							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL187	SELn
7-0: SEL186	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.96 Crossbar Select Register (SEL94)

Offset

Register	Offset
SEL94	BCh

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL189								SEL188							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL189 7-0: SEL188	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.97 Crossbar Select Register (SEL95)

Offset

Register	Offset
SEL95	BEh

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL191								SEL190							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL191 7-0: SEL190	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.98 Crossbar Select Register (SEL96)

Offset

Register	Offset
SEL96	C0h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL193								SEL192							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL193	SELn
7-0: SEL192	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.99 Crossbar Select Register (SEL97)

Offset

Register	Offset
SEL97	C2h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL195								SEL194							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL195	SELn
7-0: SEL194	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.100 Crossbar Select Register (SEL98)

Offset

Register	Offset
SEL98	C4h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL197								SEL196							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL197 7-0: SEL196	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.101 Crossbar Select Register (SEL99)

Offset

Register	Offset
SEL99	C6h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL199								SEL198							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL199 7-0: SEL198	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.102 Crossbar Select Register (SEL100)

Offset

Register	Offset
SEL100	C8h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL201								SEL200							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL201	SELn
7-0: SEL200	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.103 Crossbar Select Register (SEL101)

Offset

Register	Offset
SEL101	CAh

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL203								SEL202							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

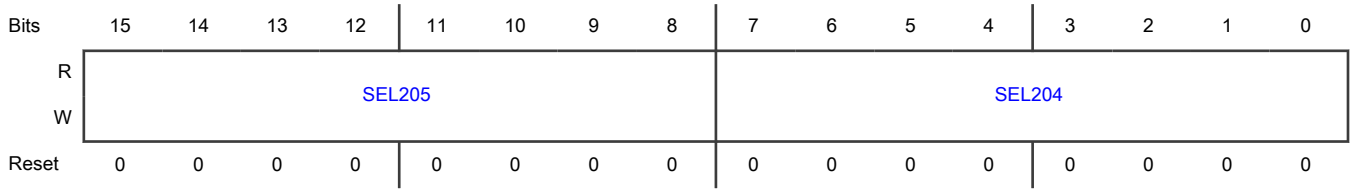
Field	Function
15-8: SEL203	SELn
7-0: SEL202	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.104 Crossbar Select Register (SEL102)

Offset

Register	Offset
SEL102	CCh

Diagram



Fields

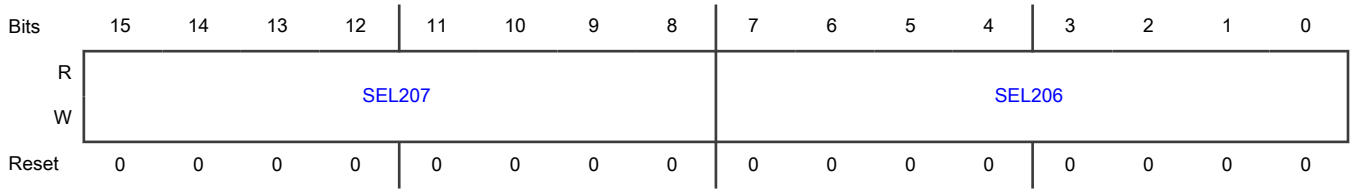
Field	Function
15-8: SEL205	SELn
7-0: SEL204	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.105 Crossbar Select Register (SEL103)

Offset

Register	Offset
SEL103	CEh

Diagram



Fields

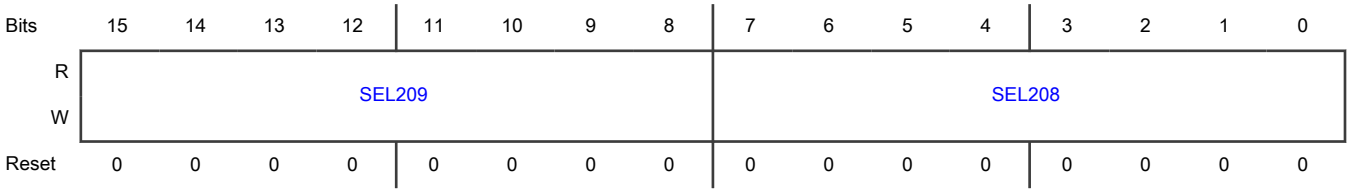
Field	Function
15-8: SEL207	SELn
7-0: SEL206	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.106 Crossbar Select Register (SEL104)

Offset

Register	Offset
SEL104	D0h

Diagram



Fields

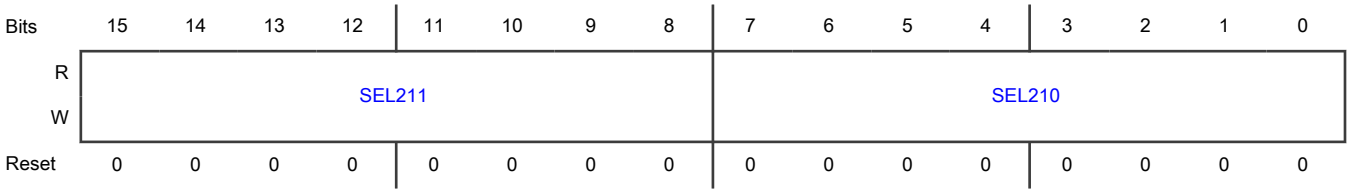
Field	Function
15-8: SEL209 7-0: SEL208	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.107 Crossbar Select Register (SEL105)

Offset

Register	Offset
SEL105	D2h

Diagram



Fields

Field	Function
15-8: SEL211 7-0: SEL210	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.108 Crossbar Select Register (SEL106)

Offset

Register	Offset
SEL106	D4h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL213								SEL212							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL213	SELn
7-0: SEL212	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.109 Crossbar Select Register (SEL107)

Offset

Register	Offset
SEL107	D6h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL215								SEL214							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

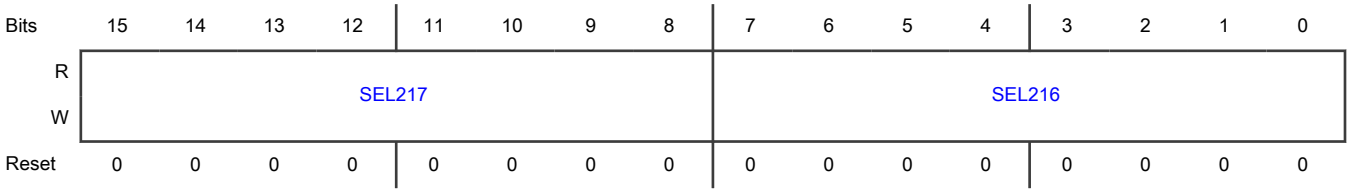
Field	Function
15-8: SEL215	SELn
7-0: SEL214	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.110 Crossbar Select Register (SEL108)

Offset

Register	Offset
SEL108	D8h

Diagram



Fields

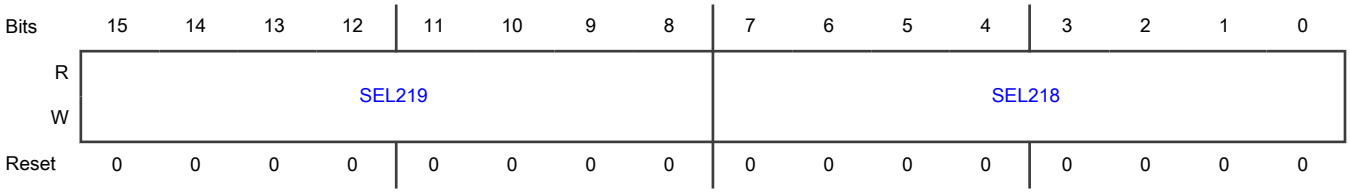
Field	Function
15-8: SEL217 7-0: SEL216	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.111 Crossbar Select Register (SEL109)

Offset

Register	Offset
SEL109	DAh

Diagram



Fields

Field	Function
15-8: SEL219 7-0: SEL218	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.112 Crossbar Select Register (SEL110)

Offset

Register	Offset
SEL110	DCh

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								SEL220							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8 —	Reserved
7-0 SEL220	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.1.113 Crossbar Control Register (CTRL0)

Offset

Register	Offset
CTRL0	DEh

Function

Use this register to configure edge detection, interrupt, and DMA features for the XBAR_OUT0 and XBAR_OUT1 outputs.

The XBAR_CTRL registers are organized similarly to the XBAR_SEL registers, with control fields for two XBAR_OUT outputs in each register. In control register 0, the LSBs contain the control fields for XBAR_OUT0, and the MSBs contain the control fields for XBAR_OUT1.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			STS1	EDGE1		IEN1	DEN1	0			STS0	EDGE0		IEN0	DEN0
W				W1C								W1C				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 —	RESERVED

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 STS1	Edge detection status for XBAR_OUT1 This bit reflects the results of edge detection for XBAR_OUT1. This field is set to 1 when an edge consistent with the current setting of EDGE1 is detected on XBAR_OUT1. This field is cleared by writing 1 to it or by a DMA_ACK1 reception when DEN1 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT1, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0b - Active edge not yet detected on XBAR_OUT1 1b - Active edge detected on XBAR_OUT1
11-10 EDGE1	Active edge for edge detection on XBAR_OUT1 This field selects which edges on XBAR_OUT1 cause STS1 to assert. 00b - STS1 never asserts 01b - STS1 asserts on rising edges of XBAR_OUT1 10b - STS1 asserts on falling edges of XBAR_OUT1 11b - STS1 asserts on rising and falling edges of XBAR_OUT1
9 IEN1	Interrupt Enable for XBAR_OUT1 This bit enables the interrupt function on the corresponding XBAR_OUT1 output. When the interrupt is enabled, the output INT_REQ1 reflects the value STS1. When the interrupt is disabled, INT_REQ1 remains low. The interrupt request is cleared by writing a 1 to STS1. **RESTRICTION** IEN1 and DEN1 should not both be set to 1. 0b - Interrupt disabled 1b - Interrupt enabled
8 DEN1	DMA Enable for XBAR_OUT1 This bit enables the DMA function on the corresponding XBAR_OUT1 output. When enabled, DMA_REQ1 presents the value STS1. When disabled, the DMA_REQ"1" output remains low. **RESTRICTION** IEN1 and DEN1 should not both be set to 1. 0b - DMA disabled 1b - DMA enabled
7-5 —	RESERVED
4	Edge detection status for XBAR_OUT0

Table continues on the next page...

Table continued from the previous page...

Field	Function
STS0	This bit reflects the results of edge detection for XBAR_OUT0. This field is set to 1 when an edge consistent with the current setting of EDGE0 is detected on XBAR_OUT0. This field is cleared by writing 1 to it or by a DMA_ACK0 reception when DEN0 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT0, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0b - Active edge not yet detected on XBAR_OUT0 1b - Active edge detected on XBAR_OUT0
3-2 EDGE0	Active edge for edge detection on XBAR_OUT0 This field selects which edges on XBAR_OUT0 cause STS0 to assert. 00b - STS0 never asserts 01b - STS0 asserts on rising edges of XBAR_OUT0 10b - STS0 asserts on falling edges of XBAR_OUT0 11b - STS0 asserts on rising and falling edges of XBAR_OUT0
1 IEN0	Interrupt Enable for XBAR_OUT0 This bit enables the interrupt function on the corresponding XBAR_OUT0 output. When the interrupt is enabled, the output INT_REQ0 reflects the value STS0. When the interrupt is disabled, INT_REQ0 remains low. The interrupt request is cleared by writing a 1 to STS0. **RESTRICTION** IEN0 and DEN0 should not both be set to 1. 0b - Interrupt disabled 1b - Interrupt enabled
0 DEN0	DMA Enable for XBAR_OUT0 This bit enables the DMA function on the corresponding XBAR_OUT0 output. When enabled, DMA_REQ0 presents the value STS0. When disabled, the DMA_REQ"0" output remains low. **RESTRICTION** IEN0 and DEN0 should not both be set to 1. 0b - DMA disabled 1b - DMA enabled

78.5.1.114 Crossbar Control Register (CTRL1)

Offset

Register	Offset
CTRL1	E0h

Function

Use this register to configure edge detection, interrupt, and DMA features for the XBAR_OUT2 and XBAR_OUT3 outputs.

The XBAR_CTRL registers are organized similarly to the XBAR_SEL registers, with control fields for two XBAR_OUT outputs in each register. In control register 0, the LSBs contain the control fields for XBAR_OUT2, and the MSBs contain the control fields for XBAR_OUT3.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			STS3	EDGE3		IEN3	DEN3	0			STS2	EDGE2		IEN2	DEN2
W				W1C								W1C				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 —	RESERVED
12 STS3	Edge detection status for XBAR_OUT3 This bit reflects the results of edge detection for XBAR_OUT3. This field is set to 1 when an edge consistent with the current setting of EDGE3 is detected on XBAR_OUT3. This field is cleared by writing 1 to it or by a DMA_ACK3 reception when DEN3 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT3, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0b - Active edge not yet detected on XBAR_OUT3 1b - Active edge detected on XBAR_OUT3
11-10 EDGE3	Active edge for edge detection on XBAR_OUT3 This field selects which edges on XBAR_OUT3 cause STS3 to assert. 00b - STS3 never asserts 01b - STS3 asserts on rising edges of XBAR_OUT3 10b - STS3 asserts on falling edges of XBAR_OUT3 11b - STS3 asserts on rising and falling edges of XBAR_OUT3
9 IEN3	Interrupt Enable for XBAR_OUT3 This bit enables the interrupt function on the corresponding XBAR_OUT3 output. When the interrupt is enabled, the output INT_REQ3 reflects the value STS3. When the interrupt is disabled, INT_REQ3 remains low. The interrupt request is cleared by writing a 1 to STS3. **RESTRICTION** IEN3 and DEN3 should not both be set to 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt disabled 1b - Interrupt enabled
8 DEN3	DMA Enable for XBAR_OUT3 This bit enables the DMA function on the corresponding XBAR_OUT3 output. When enabled, DMA_REQ3 presents the value STS3. When disabled, the DMA_REQ"3" output remains low. **RESTRICTION** IEN3 and DEN3 should not both be set to 1. 0b - DMA disabled 1b - DMA enabled
7-5 —	RESERVED
4 STS2	Edge detection status for XBAR_OUT2 This bit reflects the results of edge detection for XBAR_OUT2. This field is set to 1 when an edge consistent with the current setting of EDGE2 is detected on XBAR_OUT2. This field is cleared by writing 1 to it or by a DMA_ACK2 reception when DEN2 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT2, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0b - Active edge not yet detected on XBAR_OUT2 1b - Active edge detected on XBAR_OUT2
3-2 EDGE2	Active edge for edge detection on XBAR_OUT2 This field selects which edges on XBAR_OUT2 cause STS2 to assert. 00b - STS2 never asserts 01b - STS2 asserts on rising edges of XBAR_OUT2 10b - STS2 asserts on falling edges of XBAR_OUT2 11b - STS2 asserts on rising and falling edges of XBAR_OUT2
1 IEN2	Interrupt Enable for XBAR_OUT2 This bit enables the interrupt function on the corresponding XBAR_OUT2 output. When the interrupt is enabled, the output INT_REQ2 reflects the value STS2. When the interrupt is disabled, INT_REQ2 remains low. The interrupt request is cleared by writing a 1 to STS2. **RESTRICTION** IEN2 and DEN2 should not both be set to 1. 0b - Interrupt disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt enabled
0 DEN2	DMA Enable for XBAR_OUT2 This bit enables the DMA function on the corresponding XBAR_OUT2 output. When enabled, DMA_REQ2 presents the value STS2. When disabled, the DMA_REQ"2" output remains low. **RESTRICTION** IEN2 and DEN2 should not both be set to 1. 0b - DMA disabled 1b - DMA enabled

78.5.2 XBAR register descriptions

The XBAR module has select registers and control registers.

In the XBAR select registers, the SELn fields select which of the shared inputs (XBAR_IN) is muxed to each mux output (XBAR_OUT). There is one SELn field per mux and therefore one per XBAR_OUT output. Crossbar output XBAR_OUT[n] presents the signal chosen by XBAR_IN[SELn]. In a select register (SELy for example), the lower 8 bits represent the input for XBAR_OUT[2y] and the higher 8 bits represent the input for XBAR_OUT[2y+1].

The actual signals connected to XBAR_IN and XBAR_OUT are application specific and are described in the XBAR Assignments tables.

The XBAR control registers configure edge detection, interrupt, and DMA features for a subset of the XBAR_OUT outputs.

78.5.2.1 XBAR memory map

XBAR2 base address: 4276_0000h

XBAR3 base address: 4277_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Crossbar Select Register (SEL0)	16	RW	0000h
2h	Crossbar Select Register (SEL1)	16	RW	0000h
4h	Crossbar Select Register (SEL2)	16	RW	0000h
6h	Crossbar Select Register (SEL3)	16	RW	0000h
8h	Crossbar Select Register (SEL4)	16	RW	0000h
Ah	Crossbar Select Register (SEL5)	16	RW	0000h
Ch	Crossbar Select Register (SEL6)	16	RW	0000h
Eh	Crossbar Select Register (SEL7)	16	RW	0000h
10h	Crossbar Select Register (SEL8)	16	RW	0000h
12h	Crossbar Select Register (SEL9)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

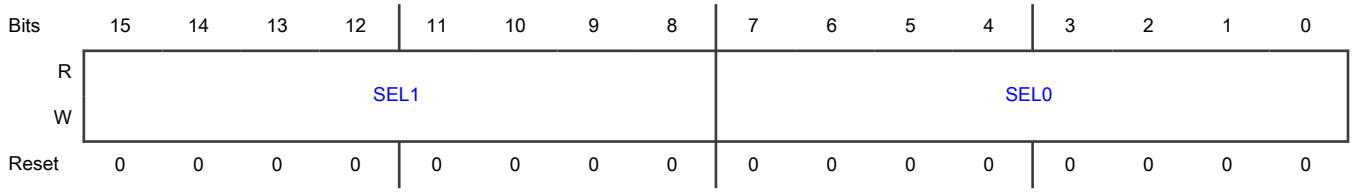
Offset	Register	Width (In bits)	Access	Reset value
14h	Crossbar Select Register (SEL10)	16	RW	0000h
16h	Crossbar Select Register (SEL11)	16	RW	0000h
18h	Crossbar Select Register (SEL12)	16	RW	0000h
1Ah	Crossbar Select Register (SEL13)	16	RW	0000h
1Ch	Crossbar Select Register (SEL14)	16	RW	0000h
1Eh	Crossbar Select Register (SEL15)	16	RW	0000h

78.5.2.2 Crossbar Select Register (SEL0)

Offset

Register	Offset
SEL0	0h

Diagram



Fields

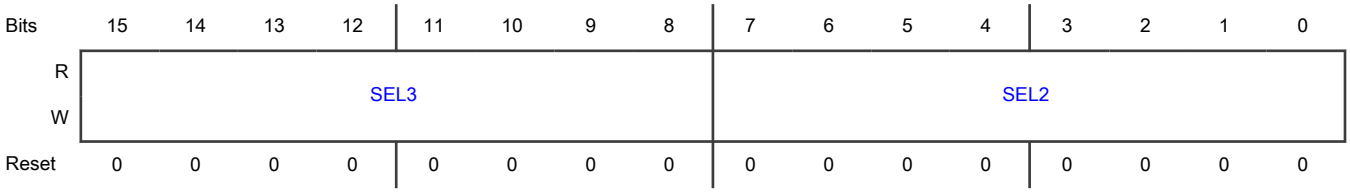
Field	Function
15-8: SEL1 7-0: SEL0	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.3 Crossbar Select Register (SEL1)

Offset

Register	Offset
SEL1	2h

Diagram



Fields

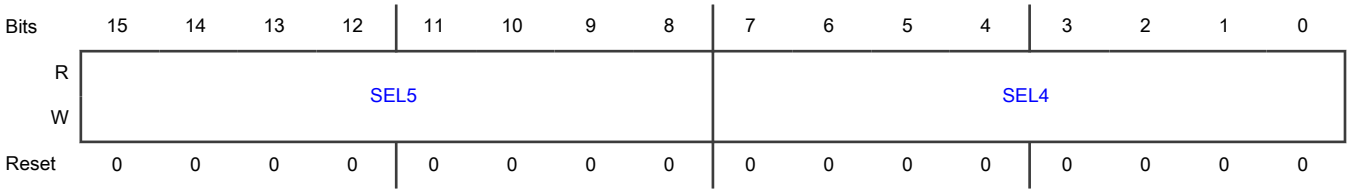
Field	Function
15-8: SEL3 7-0: SEL2	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.4 Crossbar Select Register (SEL2)

Offset

Register	Offset
SEL2	4h

Diagram



Fields

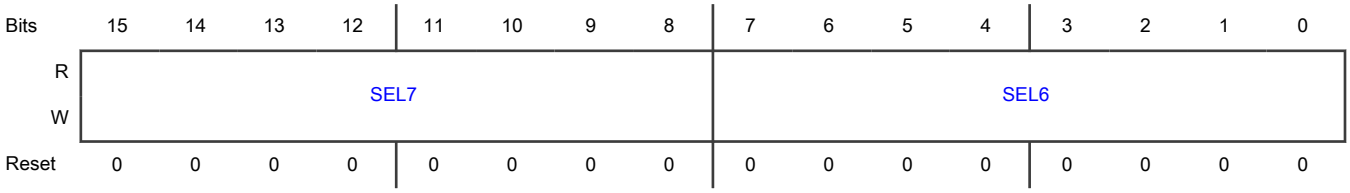
Field	Function
15-8: SEL5 7-0: SEL4	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.5 Crossbar Select Register (SEL3)

Offset

Register	Offset
SEL3	6h

Diagram



Fields

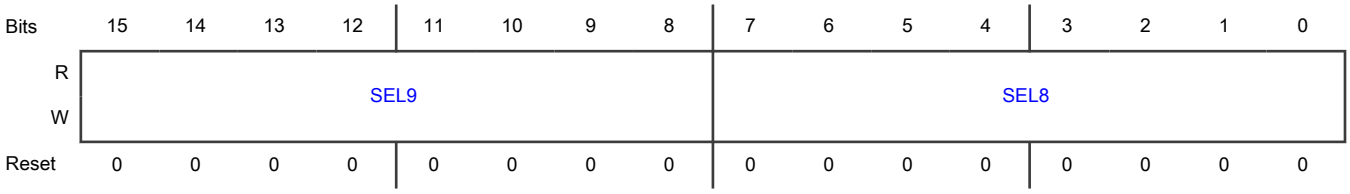
Field	Function
15-8: SEL7 7-0: SEL6	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.6 Crossbar Select Register (SEL4)

Offset

Register	Offset
SEL4	8h

Diagram



Fields

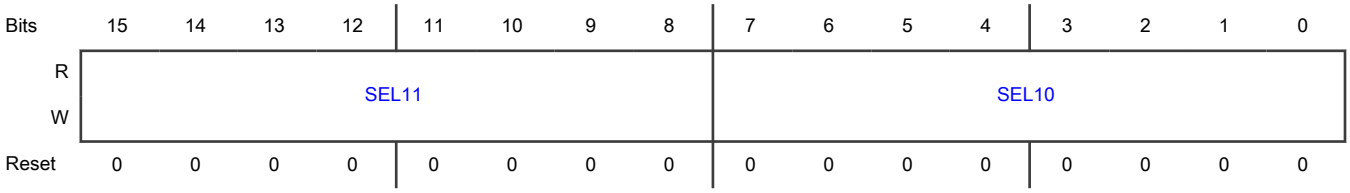
Field	Function
15-8: SEL9 7-0: SEL8	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.7 Crossbar Select Register (SEL5)

Offset

Register	Offset
SEL5	Ah

Diagram



Fields

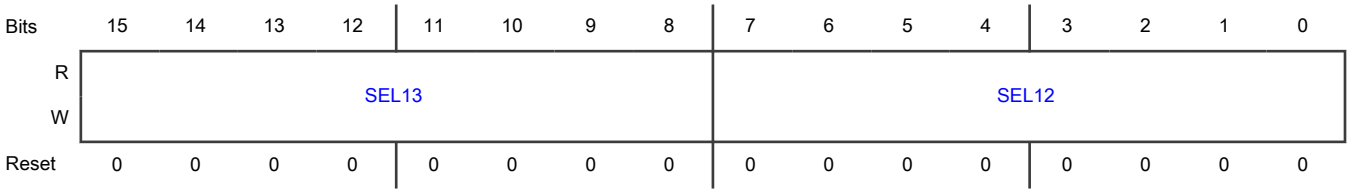
Field	Function
15-8: SEL11	SELn
7-0: SEL10	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.8 Crossbar Select Register (SEL6)

Offset

Register	Offset
SEL6	Ch

Diagram



Fields

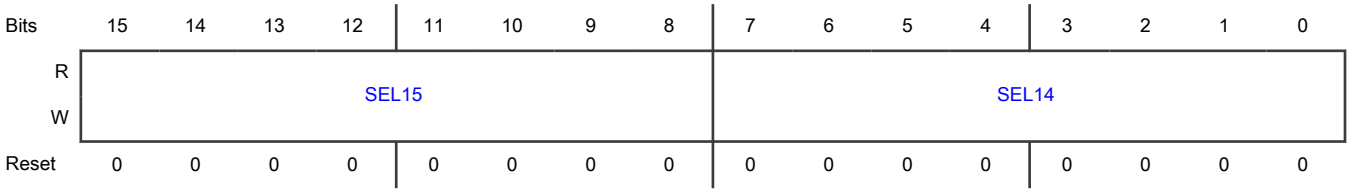
Field	Function
15-8: SEL13	SELn
7-0: SEL12	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.9 Crossbar Select Register (SEL7)

Offset

Register	Offset
SEL7	Eh

Diagram



Fields

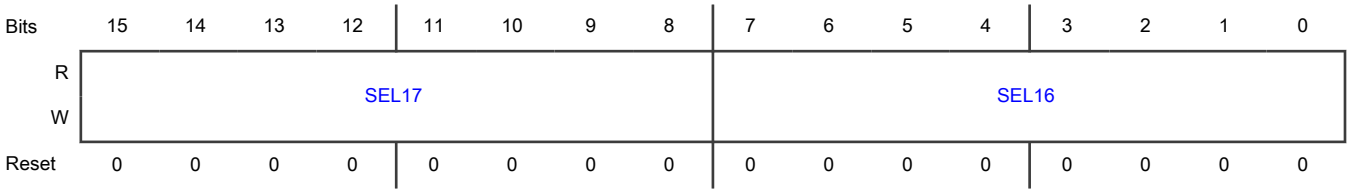
Field	Function
15-8: SEL15 7-0: SEL14	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.10 Crossbar Select Register (SEL8)

Offset

Register	Offset
SEL8	10h

Diagram



Fields

Field	Function
15-8: SEL17 7-0: SEL16	SELn Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.11 Crossbar Select Register (SEL9)

Offset

Register	Offset
SEL9	12h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL19								SEL18							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL19	SELn
7-0: SEL18	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.12 Crossbar Select Register (SEL10)

Offset

Register	Offset
SEL10	14h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL21								SEL20							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL21	SELn
7-0: SEL20	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.13 Crossbar Select Register (SEL11)

Offset

Register	Offset
SEL11	16h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL23								SEL22							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8: SEL23	SELn
7-0: SEL22	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.14 Crossbar Select Register (SEL12)

Offset

Register	Offset
SEL12	18h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEL25								SEL24							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

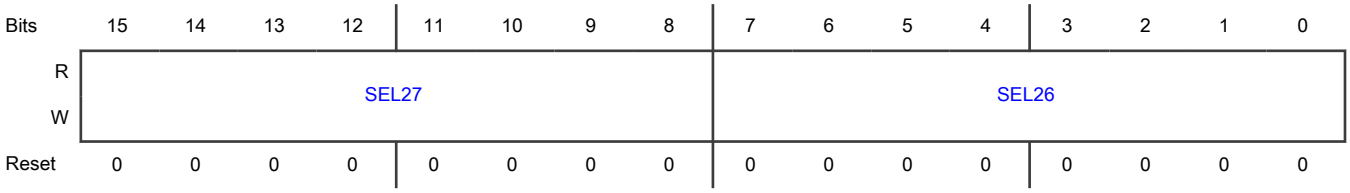
Field	Function
15-8: SEL25	SELn
7-0: SEL24	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.15 Crossbar Select Register (SEL13)

Offset

Register	Offset
SEL13	1Ah

Diagram



Fields

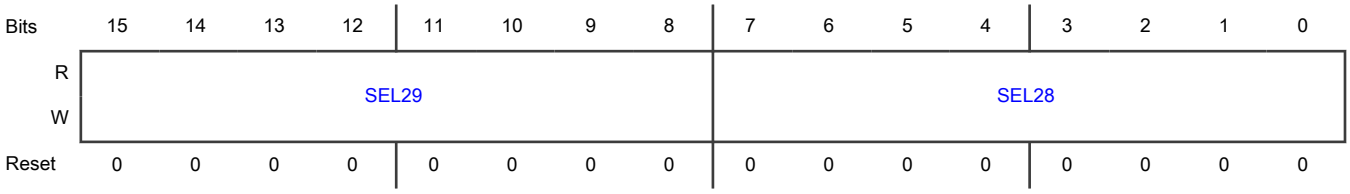
Field	Function
15-8: SEL27	SELn
7-0: SEL26	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.16 Crossbar Select Register (SEL14)

Offset

Register	Offset
SEL14	1Ch

Diagram



Fields

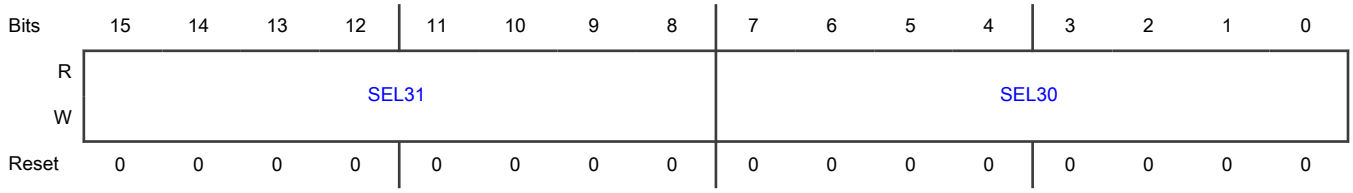
Field	Function
15-8: SEL29	SELn
7-0: SEL28	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

78.5.2.17 Crossbar Select Register (SEL15)

Offset

Register	Offset
SEL15	1Eh

Diagram



Fields

Field	Function
15-8: SEL31	SELn
7-0: SEL30	Input (XBAR_IN) to be muxed to XBAR_OUTn (see the tables in XBAR Assignments section for input/output assignment).

Chapter 79
Analog Overview

79.1 Overview

The following analog modules are integrated in this chip:

- Analog-Digital-Converter (ADC) - Supports up to 3.6MS/s sampling rate and up to 30 single-ended external analog inputs
- Comparator (CMP) - Operational over the entire supply range and features a integrated 8-bit DAC and a integrated 6 to 1 channel mux
- Digital-Analog-Converter (DAC) - 12-bit resolution with on-chip programmable reference generator output
- SINC Filter (SINC) - Converts external ADC sigma-delta modulator bit stream to data stream
- Voltage Reference (VREF) - Provides a reference voltage to external devices or used internally in the chip as a reference to analog peripherals

79.2 Analog-to-Digital Converter (ADC)

i.MX RT1180 integrates 2 ADCs.

The main features are:

- Linear successive approximation algorithm with up to 16-bit resolution
- Up to 3.6MS/s sampling rate
- Up to 30 single-ended external analog inputs
- Selectable asynchronous hardware conversion trigger with hardware channel select
- Supports both differential and single-end inputs

79.3 Analog Comparator (CMP)

i.MX RT1180 integrates 4 CMPs.

The analog comparator (CMP) allows for comparing two analog input voltages while the Analog MUX allows for selecting an analog input signal from six channels. The CMP is operational across the full range of the supply voltage. The integrated 8-bit DAC provides a selectable voltage reference for applications where voltage reference is needed.

79.4 Digital-Analog Converter (DAC)

i.MX RT1180 integrates 1 DAC.

The DAC module is 12-bit resolution with on-chip programmable reference generator output. The voltage output range is from 1/4096 Vin to Vin, and the step is 1/4096 Vin, where Vin is the input voltage from ADC_VREFH.

79.5 SINC Filter (SINC)

i.MX RT1180 integrates 3 SINCs.

The SINC Filter is an integrated module to convert external ADC sigma-delta modulator bit stream to data stream. The converters are based on up to 5-order SINC digital decimation filters with selectable oversampling ratio up to 512. Additional filtering can be done in software.

79.6 Reference Voltage (VREF)

The VREF can be used in applications to provide a reference voltage to external devices, or used internally in the device as a reference to analog peripherals (such as the ADC, DAC, or CMP). The Voltage Reference (VREF) can supply an accurate voltage output that can be trimmed in 0.5*(4/3) mV steps. The voltage reference has 3 operating modes that provide different levels of supply rejection and power consumption.

Chapter 80
Analog-to-Digital Converter (ADC)

80.1 Chip-specific ADC information

Table 1101. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments
XBAR		XBAR Chapter

The ADC supports the following voltage reference options CFG[REFSEL]:

- 00 - VREFH - connected as the primary reference option. An external reference voltage can be used. If an external reference voltage is not available, the on chip VREF_OUT can be used instead
- 01/10 - VDDA_ADC_1P8 - connected as the alternative reference option
- 11 - Reserved

NOTE

V_{SSA} and V_{REFL} is tied to common ground V_{SS} in package.

The only hardware triggers for ADC are from XBAR.

Table 1102. ADC Channel Mapping

ADC1 Channel ID	ADC2 Channel ID	GPIO
A7		GPIO_AD_02
B7		GPIO_AD_03
A6		GPIO_AD_04
B6		GPIO_AD_05

Table continues on the next page...

Table 1102. ADC Channel Mapping (continued)

ADC1 Channel ID	ADC2 Channel ID	GPIO
A5		GPIO_AD_06
B5		GPIO_AD_07
A4		GPIO_AD_08
B4		GPIO_AD_09
A3		GPIO_AD_10
B3		GPIO_AD_11
A2		GPIO_AD_12
B2		GPIO_AD_13
A1		GPIO_AD_14
B1		GPIO_AD_15
A0		GPIO_AD_16
B0		GPIO_AD_17
	A0	GPIO_AD_18
	B0	GPIO_AD_19
	A1	GPIO_AD_20
	B1	GPIO_AD_21
	A2	GPIO_AD_22
	B2	GPIO_AD_23
	A3	GPIO_AD_24
	B3	GPIO_AD_25
	A4	GPIO_AD_26
	B4	GPIO_AD_27
	A5	GPIO_AD_28
	B5	GPIO_AD_29
	A6	GPIO_AD_30
	B6	GPIO_AD_31

In the table above, each of the numeric pairs (for example, A0/B0, A1/B1, and so on) can be converted differentially by setting CMDLn[CTYPE]=10. There are eight differential pairs available on ADC1 and seven differential pairs available on ADC2. It is not recommended to convert differentially on non-designated channels (for example, A0 and B1).

NOTE

Each ADC includes an internal temperature sensor connected as an input channel to the ADC. The ADC channel assignment for temp sensor is channel A26/B26 on this chip. It is recommended to set CMDL[ALTBEN]=0. See Temperature sensor for additional information.

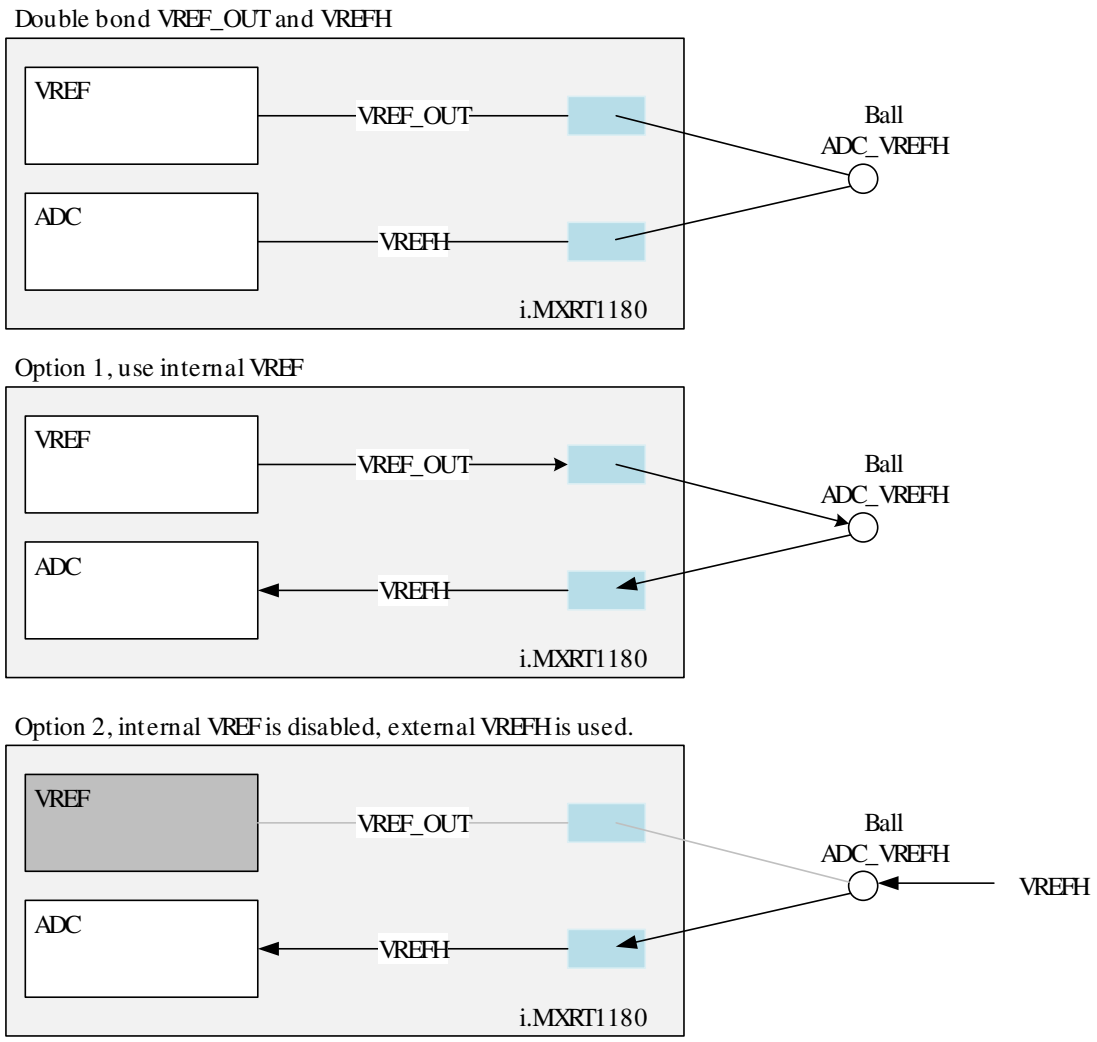


Figure 610. VREF connections and VREF_OUT and ADC_VREFH double bonding

80.2 Overview

The 16-bit analog-to-digital converter (ADC) is a dual successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

NOTE

For the chip specific modes of operation, see the power management information of the device.

80.2.1 Block diagram

The following figure is the ADC module block diagram.

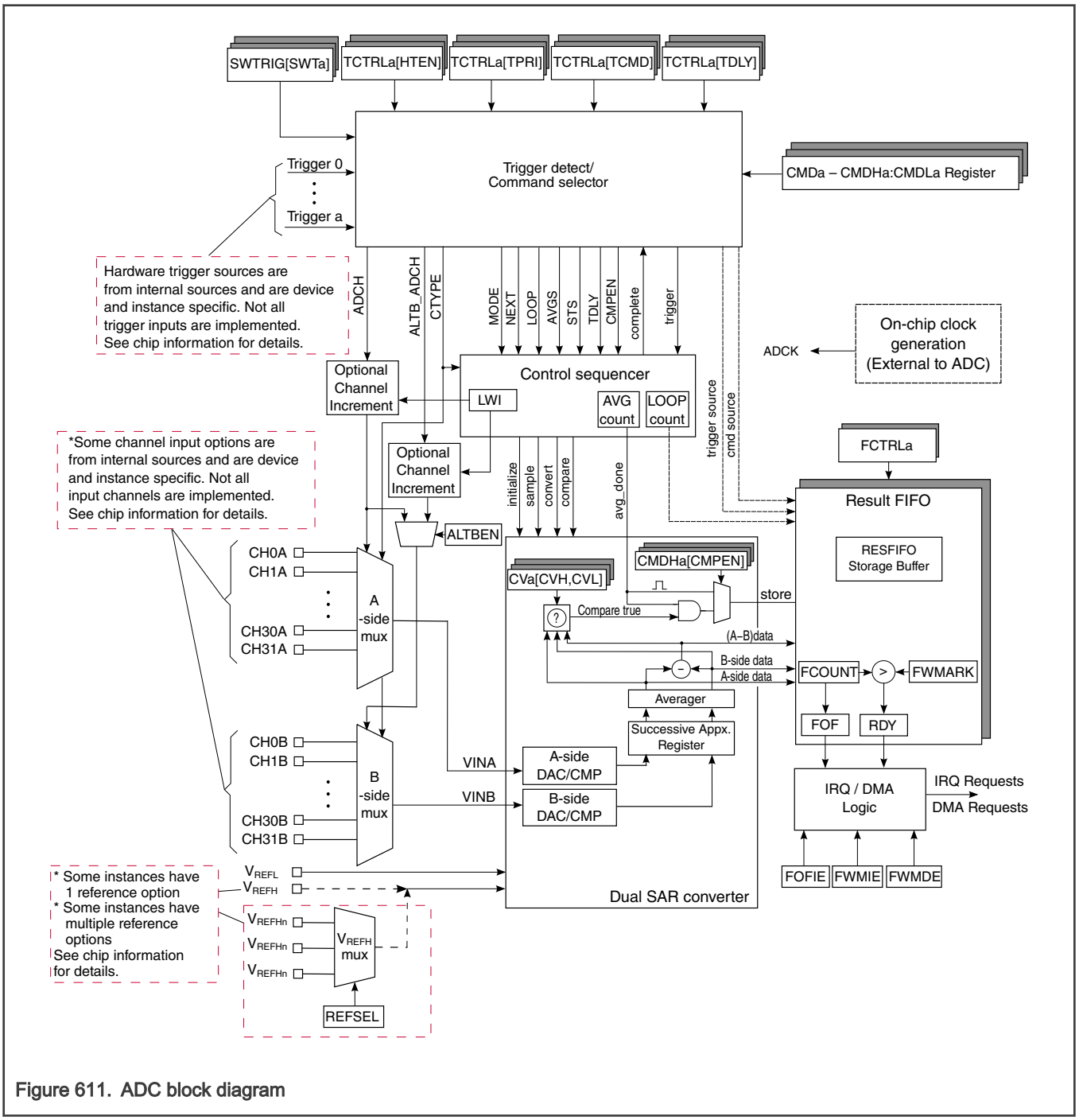


Figure 611. ADC block diagram

80.2.2 Features

Following are the features of the ADC module.

- Linear successive approximation algorithm
  - differential operation with 16-bit or 13-bit resolution
  - single-ended operation with 16-bit or 12-bit resolution
  - One ADC supports for two simultaneous single ended conversions.
- Channel scaling allows input voltage levels higher than the ADC reference voltage
- Configurable analog input sample time
- Configurable speed options to accommodate operation in low power modes of SoC
- Trigger detect with up to 8 trigger sources with priority level configuration. Software or hardware trigger option for each.
- 15 command buffers allow independent options selection and channel sequence scanning.
- Automatic compare for less-than, greater-than, within range, or out-of-range with "store on true" and "repeat until true" options
- 2 independent result FIFOs each contains 16 entries. Each FIFO has configurable watermark and overflow detection
- Interrupt, DMA or polled operation
- Linearity and gain adjustment calibration logic

80.3 Functional description

The ADC module performs analog-to-digital conversions on any of the software selectable analog input channels by a successive approximation algorithm. The module initializes to its lowest power state during reset. The ADC analog circuits can optionally be pre-enabled for faster starts to conversions at the expense of higher idle currents. Conversions are initiated by selectable trigger events from software or hardware sources. The trigger detect logic includes a configurable enable and priority scheme for the available trigger sources. The module includes multiple command buffers that provide configurable flexibility for channel scanning and independent channel selections for different trigger sources. Multiple command buffers also allow variable option selection such as input scaling factor, differential vs. single-ended, sample time and averaging on a per-channel basis.

The ADC module optionally averages the result of multiple conversions on a channel before storing the calculated result. The hardware average function is enabled by setting CMDH*a*[AVGS] bitfield to a non-zero value and operates in any of the conversion modes and configurations.

When the conversion and averaging loops are completed, the resulting data is placed in one of 2 available FIFO data buffers along with other tag information associated with the result. A configurable watermark level supports interrupt or DMA requests when the number of stored datawords exceeds the setting. Interrupts can also be enabled to indicate when FIFO overflow errors occur.

The ADC module optionally compares the result of a conversion with the contents of two value registers for less-than, greater-than, inside-range or outside-range detection. The compare function operates in any of the conversion modes and configurations.

The ADC module includes hardware calibration logic to correct for offset, gain and linearity errors. Calibration steps should be executed after any reset or power up. Each SAR conversion utilizes calibration data calculated during the calibration routine.

The sequencing of a ADC command is summarized in the flow diagram shown below.

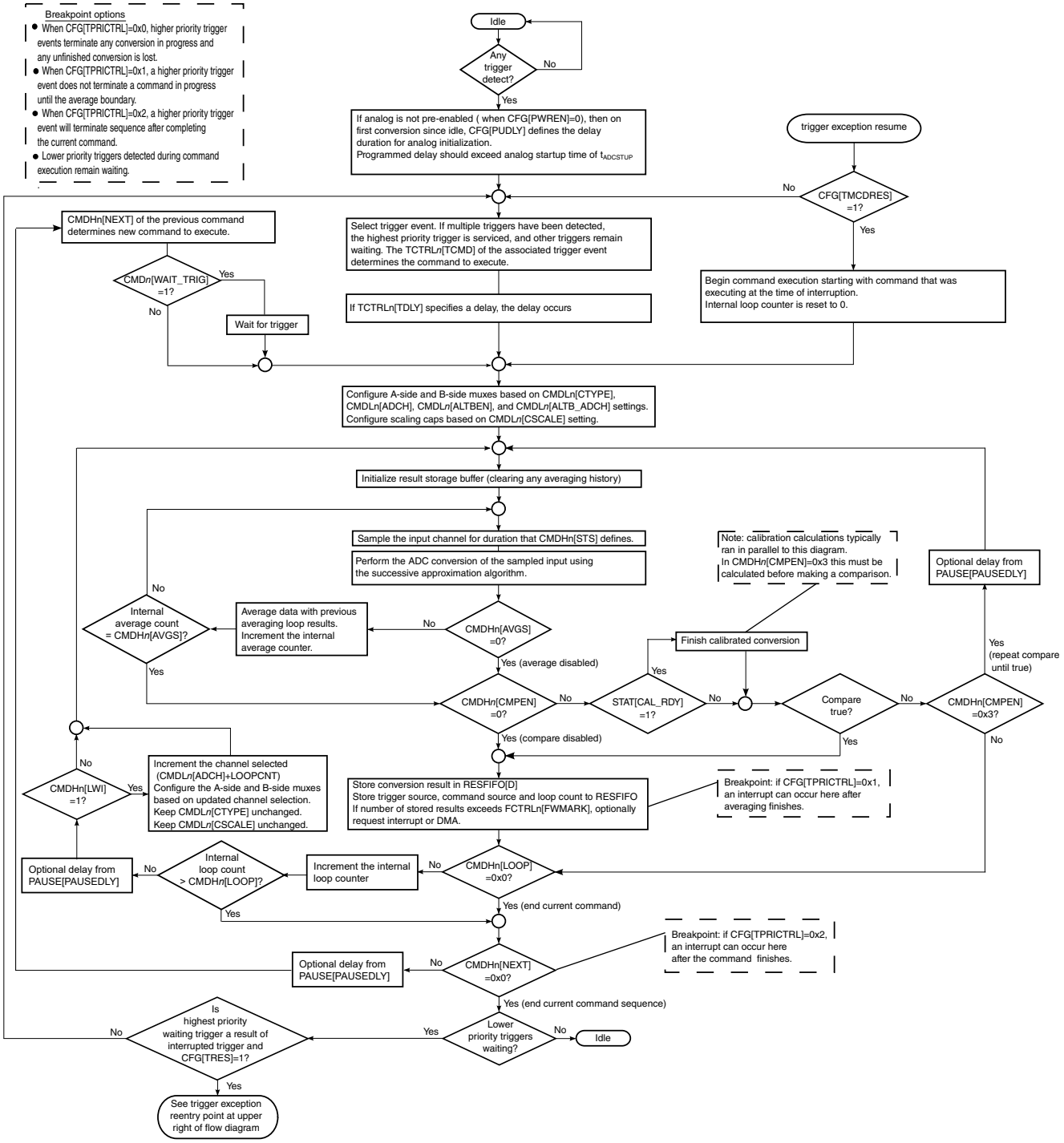


Figure 612. ADC command execution flow diagram

80.3.1 Power control mode

The default setting for the ADC analog circuits is disabled while the ADC is in its Idle state. When a trigger is detected and ADC command processing is initiated, the analog circuits are enabled and require a period of initialization before the first conversion cycle. The CFG[PUDLY] should be programmed such that a delay duration longer than $t_{ADCSTUP}$ is incurred. Accuracy of the initial conversion(s) after activation is degraded if CFG[PUDLY] is set to too small a value.

Faster conversion startup times can be achieved by optionally setting the CFG[PWREN] field to pre-enable the analog circuits of the ADC at the expense of power consumption even while the module is in an idle state. When CFG[PWREN] is set, the Power Enable timer is activated and enforces the minimum startup (controlled by the CFG[PUDLY] register field) before detected triggers are allowed to initiate ADC conversions.

80.3.2 ADC modes of operation

The ADC module supports the chip low power modes described in the following table. See section Clocking for more information.

Table 1103. Chip modes supported by the ADC module

Chip mode	ADC Operation
Run	Normal operation
Deepsleep/Sleep	When the Doze Enable bit (CTRL[DOZEN]) is 0, the ADC can continue to operate and the module is using an external or internal clock source which remains operating during Deepsleep/Sleep modes. ADC is allowed to continue operation while the system is transitioned to the low power state. Any conversion in progress is not disrupted. External hardware trigger detect and active conversions are operational. When the DOZEN bit is 1, the module waits for the current averaging iteration/FIFO storage to complete before acknowledging Deepsleep or Sleep mode entry. There is no associated ack/handshake and the system transitions to low power state without ADC interaction. ADC operation should terminate after completion of any conversion in progress.
Deep Powerdown	The Doze Enable (CTRL[DOZEN]) bit is ignored and the module waits for the current transfer to complete any pending operation before acknowledging Deep Powerdown mode entry.

80.3.3 Voltage reference

The voltage reference high (V_{REFH}) used by the ADC is supplied from either an on-chip voltage reference source or from an off-chip source supplied through external pins. V_{REFL} is always from an external pin and must be at the same voltage potential as V_{SSA}. See the chip configuration information on the voltage reference options specific to this packaged device.

This block supports a programmable selection of the Voltage Reference used for ADC conversions (via the CFG[REFSEL] field). See the chip configuration information on the voltage reference options specific to this packaged device.

80.3.4 Trigger detect and command execution

See Figure 612 for a flow diagram of command execution sequencing.

ADC command execution is initiated from up to 8 trigger sources. Each trigger can be software generated by writing 0b1 to the corresponding SWTRIG[SWT*n*] bitfield. Alternatively, hardware triggers can be generated from asynchronous input sources at the periphery of the module. The number and sources of hardware triggers implemented is device specific. See the chip-specific ADC information for description of available hardware trigger sources for this device.

Each hardware trigger source is enabled by setting the associated enable bit (TCTRLa[HTEN]). Each trigger source is assigned a priority via the associated priority control field (TCTRLa[TPRI]). Each of the trigger sources is associated with a command buffer via the associated command select field (TCTRLa[TCMD]).

When a hardware trigger input is enabled, hardware trigger events are detected on the rising-edge of the associated hardware trigger source.

Each trigger source has an associated priority field TCTRLa[TPRI] which allows for arbitration between trigger sources. Arbitration is in control of two things: selecting which trigger sequence to execute next, and selecting how to handle a trigger exception. Trigger exceptions are defined as allowing a higher priority trigger sequence to interrupt operation of a lower priority sequence. When a trigger exception occurs, programmable arbitration allows the configurable stop and resume points for low priority sequences. The fields affecting arbitration are CFG[HPT_EXDI],CFG[TCMDRES],CFG[TRES], and CFG[TPRICTRL].

1. If CFG[HPT_EXDI] is set to 1b, then trigger exceptions are disabled and any higher priority triggers are left pending until the current sequence completes. Note that new triggers are accepted based on priority.
2. If CFG[HPT_EXDI] is set to 0b (default), then exceptions are enabled and the higher priority sequence begins executing at a user specified breakpoint.

Breakpoint locations are determined by the register CFG[TPRICTRL]. CFG[TPRICTRL] has an affect on latency for accepting a trigger exception.

1. When CFG[TPRICTRL]=0x0, a higher priority trigger causes an immediate command abort and the new command specified by the trigger is immediately started.
2. When CFG[TPRICTRL]=0x1, the current conversion is allowed to complete (including averaging) before the higher priority exception is initiated. In this mode, if the command is running through a series of averages, this series completes. However, there is no requirement to finish the entire command before being interrupted. For example, if the command consists of 4 loop iterations, there is no requirement to complete all 4 iterations before the interrupt occurs.
3. When CFG[TPRICTRL]=0x2, a higher priority trigger begins once the current command is completed. If a command consists of 5 loop iterations each containing 8 averages, then all 5x8 conversions must be completed before accepting the trigger exception.

CFG[TCMDRES] and CFG[TRES] determine what the ADC does after accepting a trigger exception. The module can be programmed to resume commands after returning from a trigger exception.

1. If CFG[TRES] = 0x0 then commands are not automatically resumed after being stopped by an exception. However, an interrupt is set to indicate this case has occurred. The flag TSTAT[TEXC_NUM] can be used to resolve which trigger was stopped by the exception.
2. If CFG[TRES] = 0x1 the ADC automatically resumes commands after they were stopped by an exception.

By utilizing CFG[TRES] in conjunction with CFG[TCMDRES], the module can be programmed to resume commands at one of two possible locations.

1. If CFG[TCMDRES] = 0x0 then the trigger which was stopped by an exception is resumed from the beginning of its associated command sequence. Note, triggers which are waiting to be resumed take the same priority programmed to TCTRLa[TPRI].
2. If CFG[TCMDRES] = 0x1 then the trigger is resumed from the command that it was executing before being interrupted by an exception.

If a lower priority trigger occurs (i.e., a trigger event occurs that is configured for a lower priority than the trigger source associated with the currently executing command), the trigger detect is left pending until completion of the current command sequence. Lower priority trigger events cannot be serviced until a higher priority triggered command (or command sequence) completes.

When a conversion is completed (including hardware averaging when CMDHa[AVGS] is non-zero), the result is placed in a RESFIFO buffer. When an ADC command selects looping (when CMDHa[LOOP] is non-zero) a command stores multiple conversion results to the FIFO during execution of that command.

At the end of command execution, CMDHa[NEXT] selects the next command to be executed. Multiple commands can be executed sequentially by configuration of each commands CMDHa[NEXT] field. Setting the next command to 0x0 causes conversions to terminate at the completion of the current command. Unending circular command execution is allowed by setting the CMDHa[NEXT] field in the last command in a sequence to the first command in the sequence.

By default, command sequences executes automatically in the order that CMDHa[NEXT] fields are programmed. However, by utilizing the CMDHa[WAIT_TRIG], command execution can be stalled and launched based on trigger inputs. For example, if TRIGGER2 is programmed to start the command sequence CMD1, CMD2, CMD3, then receiving TRIGGER 2 one time unconditionally runs this sequence to completion. If CMDH2[WAIT_TRIG] is set to 0x1, however, then the sequence pauses after CMD1 until TRIGGER2 is received again. Therefore, sequences can be stalled until receiving a trigger assertion.

Disabling the ADC by writing 0b0 to the CTRL[ADCEN] bitfield terminates any active ADC command processing. Writing 0b0 to the CTRL[ADCEN] bitfield causes the current command (or command sequence) to terminate, clears any pending triggers and sends the ADC module to an IDLE state.

80.3.5 Pause option

When the maximum conversion rate is not required by an application the effective conversion rate can be reduced by implementing periodic trigger events to initiate ADC conversions or by selecting a reduced frequency clock as the ADACK source. Both of these options are chip specific and are dependent on ADC triggering and clocking options external to the ADC module. The latency associated with ADC analog power up delays results in a limit on the maximum conversion rate when using periodic triggering.

Another means of reducing conversion rates is by inserting a pause of a programmable duration between LOOP iterations, between commands in a sequence, and between conversions when command is executing in the "Compare Until True" configuration. When PAUSE[PAUSEEN] is set, the PAUSE[PAUSEDLY] field controls the duration of pausing during command execution sequencing. The pause delay is a count of (PAUSE[PAUSEDLY]*4) ADCK cycles. Note, the PAUSE register should not be changed while the CTRL[ADCEN] bit is set. Writes to the PAUSE register while CTRL[ADCEN] is set can lead to metastable operation.

See Figure 612 for the places during command execution sequencing where the pause is optionally inserted.

80.3.6 Resync functionality

Any trigger source (SW or HW) can be configured to act as a resync trigger. Trigger based resync functionality is used to interrupt a running trigger (resync target) and clear the FIFO it's writing to. This can either be used to abort a running sequence, or restart a running sequence depending on the configuration of CFG[TRES]. If CFG[TRES] = 0b1 then the target sequence is aborted, the FIFO cleared, and the sequence restarted after the resync occurs. If CFG[TRES] = 0b0 then the target sequence is aborted and the FIFO is cleared after the RESYNC occurs. Note, the FIFO(s) cleared are based on the resync target TCTRLm[FIFO_SEL_A] and TCTRLm[FIFO_SEL_B]. To only clear one FIFO, TCTRLm[FIFO_SEL_A] == TCTRLm[FIFO_SEL_B]. Any results not associated with the resync target are lost if they are stored in either TCTRLm[FIFO_SEL_A] or TCTRLm[FIFO_SEL_B] at the time of the resync.

A resync trigger needs to have a specific target. The resync only occurs if the resync target is running at the time of the trigger. For the following description, let n be the resync trigger number, let m be the resync target number. According to these variables, trigger n should resync trigger m. To enable a trigger source to act as a resync trigger, the following conditions must be satisfied:

1. The resync trigger TCTRLn[RSYNC] must be set to 0b1.
2. The resync trigger must have higher priority than the resync target (TCTRLn[TPRI] must be less than TCTRLm[TPRI]).
3. The resync target is specified using TCTRLn[TCMD]. In this case the resync target, m, must be equal to TCTRLn[TCMD].
4. The resync target, m, must be executing commands when the resync trigger, n, is asserted.
5. Trigger m must have at least one conversion left to begin when trigger n is received.

If a trigger source n has TCTRLn[RSYNC] set to 0b1, but some of the above conditions are not met then the trigger source n is ignored.

The following figure illustrates a resync trigger sequence executing. Note, in this example, trigger source 1 is configured to resync trigger source 0.

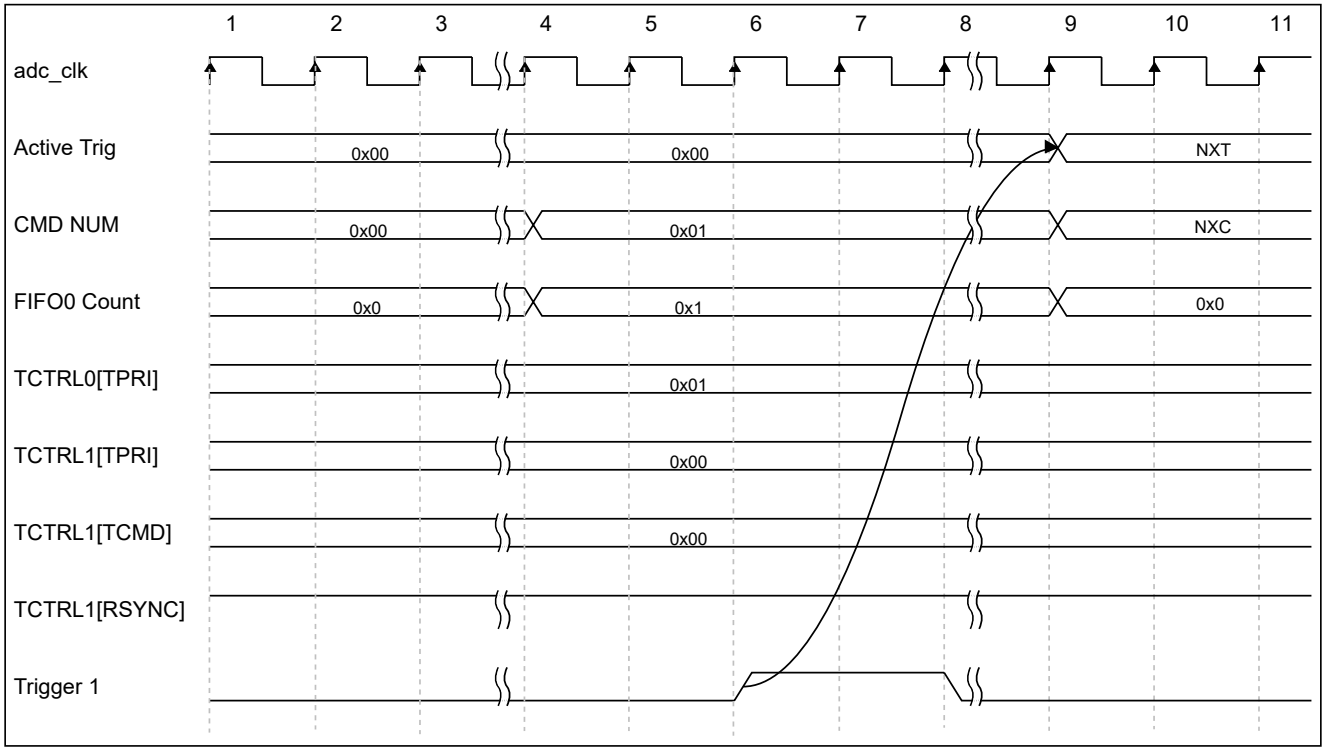


Figure 613. ADC Resync Example

In this figure, trigger source 0 was executing a sequence of commands when trigger source 1 was asserted. Notice that trigger source 0 is stopped when trigger source 1 is asserted (after some synchronization delay). In addition, the FIFO being written to by trigger source 0 is cleared (FIFO0 in this example). After the trigger 0 sequence is stopped, the ADC runs the next trigger pending with the highest priority. This is marked as NXT, for next trigger. If resume functionality is enabled, and trigger source 0 had the highest priority pending, then NXT = 0x00.

80.3.7 Temperature sensor

The ADC module has a dedicated input channel for an on-chip temperature sensor. Refer to the chip specific channel definition to determine which channel is connected to the on-chip temperature sensor.

To calculate the temperature, application software must first execute a conversion of the temperature sensor channel with some configuration requirements. This sequence for converting the temperature sensor channel is outlined below:

1. Refer to chip specific channel definition to determine which channel corresponds to the temperature sensor.
2. Configure a command buffer register to convert the temperature sensor channel. CMDLa[ADCH] = Temperature Sensor Channel.
3. The command buffer must also be programmed with the following parameters:
   - 16 bit mode (CMDLa[MODE] = 0x1)
   - Differential (CMDLa[CTYPE] = 0x2)
   - Max averaging (CMDHa[AVGS] = 0xA)
   - Max sample time (CMDHa[STS] = 0x7)
   - LOOP set to 1 (CMDHa[LOOP] = 0x1)
   - Loop with increment disabled (CMDHa[LWI] = 0x0)
   - Compare function disabled (CMDHa[CMPEN] = 0x0)

4. Configure a trigger control register TCTRLa with the following parameters:
   - TCTRLa[TCMD] = command buffer used in steps 2 and 3.
   - TCTRLa[FIFO_SEL_A] will direct conversion results to FIFO0 or FIFO1
5. Trigger a conversion of the temperature sensor channel. Software triggers the conversion by writing "1" to the associated bit in SWTRIG register (i.e., the trigger configured in step 4).

After completion of the conversion of the temperature sensor channel, two results are stored in the FIFO selected with TCTRLa[FIFO_SEL_A]. Each result corresponds to a component of the overall temperature value. Application software reads these values from the FIFO and uses the equation below to calculate the ambient temperature.

$$Temp = A * [\frac{\alpha * (Vbe8 - Vbe1)}{Vbe8 + (\alpha * (Vbe8 - Vbe1))}] - B$$

Figure 614. Temperature sensor function

Where:

- Vbe1 is the first value stored to the FIFO as a result of the temperature sensor channel conversion
- Vbe8 is the second value stored to the FIFO as a result of the temperature sensor channel conversion
- A is the slope factor
- B is the offset factor
- α is the bandgap coefficient

A, B, and α are specified constant values from the ADC Electrical information in the device datasheet.

80.3.8 Result FIFO operation

The ADC includes 2 16 entry FIFOs in which the result of ADC conversions are stored. In addition, a valid indicator bit, the trigger source, the source command and the loop count are also stored along with the data. FCTRLa[FCOUNT] indicates how many valid datawords are stored in each RESFIFO.

A programmable watermark threshold supports configurable notification of data availability. When FCTRLa[FCOUNT] is greater than FCTRLa[FWMARK], the associated RDY flag is asserted. When IE[FWMIEx] is set, a watermark interrupt request is issued. When DE[FWMDEx] is set, a DMA request is issued. Reading RESFIFO provides the oldest unread dataword entry in the FIFO and decrements FCTRLa[FCOUNT]. When FCTRLa[FCOUNT] falls equal to or below FCTRLa[FWMARK], the RDY flag is cleared.

Each FIFO can be emptied by successive reads of RESFIFOa. When the RESFIFOa[VALID] bit is 1 the associated FIFO entry is valid. Reading RESFIFOa when the FIFO is empty (when RESFIFOa[VALID] is clear and FCTRLa[FCOUNT]=0x0) provides an undefined dataword. All FIFOs are reset by writing 0b1 to the CTRL[RSTFIFOx] bit.

If the ADC attempts to store a dataword to the FIFO when the FIFO is full the FIFO overflow flag (FCTRLa[FOF]) is set. When IE[FOFIEx] is set, a overflow interrupt request is issued. The FOF flag is cleared by writing 1 to STAT[FOFx]. On overflow events no new data is stored and the data associated with the store that triggered the overflow is lost.

Conversion results can be steered to any FIFO in the design. TCTRLa[FIFO_SEL_A] and TCTRLa[FIFO_SEL_B] is utilized to determine which FIFO to write the final result. Therefore, depending on which trigger is executing the results can be steered to different locations. Depending on the type of conversion selected the FIFO destination register fields are interpreted differently. During either differential or single-ended mode (CMDLa[CTYPE] != 0x3) only one result is produced. The destination during these modes is determined from TCTRLa[FIFO_SEL_A]. In dual-single-ended mode, both TCTRLa[FIFO_SEL_A] and TCTRLa[FIFO_SEL_B] are used to determine the Channel A and Channel B destinations respectively.

80.3.9 Sampling modes

The ADC module supports three different sampling modes: differential, single-ended, and dual single-ended. The sampling mode is determined by the currently executing command using the register field CMDLa[CTYPE]. When executing a command in

dual single-ended mode, two independent conversion results are calculated and stored in selectable FIFO destinations. Note, however, that command processing is not individually controlled for each independent channel being sampled. When operating in dual single-ended mode both channels are sampled and processed simultaneously. The selection of Channel B is controlled by the CMDLa[ALTBEN] control bit. When CMDLa[ALTBEN] bit is set, the channel for the B-side is selected independently (via CMDLa[ALTB_ADCH] setting). When CMDLa[ALTBEN] bit is clear, the Channel B selection is controlled by CMDLa[ADCH] and Channel A and Channel B are paired (ex. CH0A/CH0B, CH1A/CH1B, ...). If comparisons are enabled in dual single-ended mode, then only the A-side channels (CH0A, CH1A, CH2A, ...) are used for the comparison. The A-side comparison is used to determine if both the A-side and B-side results are written to the FIFOs:

- If the A-side comparison passes, both the A-side and B-side results are stored.
- If the A-side comparison fails, neither the A-side nor B-side results are written to the FIFOs.

Single-ended mode is configurable to allow either A-side or B-side channels to be sampled. In single-ended mode, the results from each conversion can be written to a selectable FIFO using TCTRLa[FIFO_SEL_A]. In differential mode, the final SAR calculation is equivalent to V(CHA) - V(CHB). If the result is negative, then the value is stored in sign-extended two's complement format. TCTRLa[FIFO_SEL_A] also determines where differential conversion results are stored. In dual single-ended mode, however, two independent results are produced. Individual control is provided by using TCTRLa[FIFO_SEL_A] and TCTRLa[FIFO_SEL_B] to select a FIFO destination for both results during this mode. Note that both single-ended results may be written to the same destination by programming TCTRLa[FIFO_SEL_A] = TCTRLa[FIFO_SEL_B]. In this case, the CH_A result is always stored before the CH_B result.

80.3.10 Compare function

After the input is sampled and converted and any averaging iterations are performed, the CMDH*a*[CMPEN] field guides operation of the automatic compare function to optionally only store when the compare operation is true. There are multiple options on command sequencing related to the compare function as summarized in the table below.

NOTE

Latency is added to the end of a compare until true conversion to resolve the next command or loop in a sequence. This latency is necessary to calibrate the SAR data before resolving the result of a comparison. Delay for this feature is only added when resolving the result of a conversion. This means that intermediate samples during averaging do not include extra latency. Only loop and command boundaries experience this delay. The latency is always less than or equal to 5 ADC clock cycles.

NOTE

Not all Command Buffers have an associated Compare Value register. The compare function is only available on Command Buffers that have a corresponding Compare Value register.

Table 1104. Compare modes

CMDH*a*[CMPEN]	Compare Function	Description
0b00	Compare disabled	Do not perform compare operation. Always store the conversion result to the FIFO.
0b01	Reserved	
0b10	Store on true	Perform compare operation. Store conversion result to FIFO at end of averaging only if compare is true. If compare is false do not store the result to the FIFO. In either the true or false condition, the LOOP setting is considered and increments the LOOP counter before deciding whether the current command has completed or additional LOOP iterations are required.
0b11	Repeat compare until true	Perform compare operation. Store conversion result to FIFO at end of averaging only if compare is true. Once the true condition is found the

Table continues on the next page...

Table 1104. Compare modes (continued)

CMDH*a*[CMPEN]	Compare Function	Description
		LOOP setting is considered and increments the LOOP counter before deciding whether the current command has completed or additional LOOP iterations are required. If the compare is false do not store the result to the FIFO. The conversion is repeated without consideration of LOOP setting and does not increment the LOOP counter.

Depending on CV*a*[CVH] and CV*a*[CVL] values programmed, the compare operation checks whether the result is less than, greater than, or if the result falls within or outside a range determined by two compare values. The compare values are used as described in the following table.

Table 1105. Compare operations

CV*a*[CVL] vs. CV*a*[CVH]	Operation	Description
set CVa[CVL] < CVa[CVH]	Outside range (General form)	Compare true if the result is less than CVa[CVL] value **OR** greater than CVa[CVH] value.
set CVa[CVH] to max value set CVa[CVL] to compare point	Less than	Compare true if the result is less than CVa[CVL] value.
set CVa[CVL] to min value set CVa[CVH] to compare point	Greater than	Compare true if the result is greater than CVa[CVH] value.
set CVa[CVL] > CVa[CVH]	Inside range	Compare true if the result is less than CVa[CVL] value **AND** greater than CVa[CVH] value.

NOTE

In low power modes where the ADC continues to operate, the compare function can monitor the voltage and only wake the device when the compare condition is met.

80.3.11 Clocking

The ADC operates from the ADCK clock input provided from an on-chip clock select block and is used by the SAR conversion control sequencing logic and the FIFO storage buffer. The ADCK frequency must be within the specified frequency range for ADCK. Refer to the device datasheet for the supported frequency range.

The ADC continues operating in Deepsleep and Sleep modes provided the Doze Enable bit (CTRL[DOZEN]) is clear and the on-chip clock select block continues to supply an ADCK clock source. Note, in Deepsleep mode with CTRL[DOZEN] == 0b0, the bus clock can be shut off, and asynchronous interrupts and DMA requests can be configured. In addition, the ADC continues processing commands and writing data to the internal FIFO. The ADC has four sources for async interrupts during Deepsleep mode: watermark, FIFO overflow, TCOMP, and TEXC. To enable them, properly configure the bits IE[FWMIEx], IE[FOFIEx], IE[TCOMP_IE], and IE[TEXC_IE] before entering Deepsleep mode.

When the CTRL[DOZEN] bit is set in Deepsleep and Sleep modes, the ADC waits for the current averaging iteration/FIFO storage to complete before acknowledging Deepsleep or Sleep mode entry. Any pending triggers are dropped when a Deepsleep/Sleep mode request is made with CTRL[DOZEN] set. The ADC is forced into its lowest power setting after acknowledging the CTRL[DOZEN] Deepsleep/Sleep mode request. The same behavior is observed when entering a Deep Powerdown Mode.

80.3.12 Resets

Table 1106. ADC Resets

Signal	Description
Chip reset	The logic and registers for the ADC are reset to their default state on a chip reset.
Software reset	The ADC implements a software reset bit in its Control Register. The CTRL[RST] bit resets all logic and registers to their default state, except for the CTRL register itself.
FIFO reset	The ADC implements write-only control that reset the FIFO0 (CTRL[RSTFIFO0]) and FIFO1 (CTRL[RSTFIFO1]). After a FIFO is reset, that FIFO is empty.

80.3.13 Interrupts and DMA requests

The ADC includes several sources for interrupts and/or DMA requests. The table below summarizes these sources.

A programmable watermark threshold supports configurable notification of data availability and can generate either an interrupt exception or a DMA request. When FCTRLn[FCOUNT] is greater than FCTRLn[FWMARK], the associated RDYn flag is asserted. Masking for this exception is controlled by the IE[FWMIEn] and DE[FWMDEn] control bits. When RDYn is asserted and mask control bit IE[FWMIEn] is set, a watermark interrupt request is issued. When RDYn is asserted and mask control bit DE[FWMDEn] is set, a DMA request is issued.

The other exception sources can only generate interrupts and do not have a DMA request option. Each of these sources has a mask control bit in the IE register and no corresponding bit in the DE register.

Table 1107. ADC Interrupts and DMA Requests

Status Register (STAT)		Description	Can generate		
Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
RDYn	Result FIFO n Ready Flag	Conversion result data is written to Result FIFO and has a watermark configurable trigger level to generate an exception request as controlled by FCTRLn[FWMARK].	Y	Y	Y
FOFn	Result FIFO n Overflow Flag	Attempts to store data to the FIFO when the FIFO is full is an error condition.	Y	N	Y
TCOMP_INT	Trigger Completion Flag	A trigger sequence has been completed (all associated commands have been run).	Y	N	Y
TEXC_INT	High Priority Trigger Flag	A high priority trigger exception has occurred.	Y	N	Y

80.4 External signals

The ADC module supports analog channel inputs with differential and single-ended conversion options for all channels. The module also requires supply and ground connections.

Table 1108. ADC signal descriptions

Signal	Description	I/O
V_{DDA}	Analog Power Supply	I
V_{SSA}	Analog Ground	I

Table continues on the next page...

Table 1108. ADC signal descriptions (continued)

Signal	Description	I/O
CHnA–CH0A[1]	A-side Analog Channel Inputs	I
CHnB–CH0B[1]	B-side Analog Channel Inputs	I

1. where n is the maximum channel number supported in the chip. See the chip-specific information to know the number of channels supported on your device.

80.4.1 Analog Channel Inputs (CH*n*A and CH*n*B)

The CMDLa[ADCH] and CMDLa[CTYPE] bitfields control selection of paired or individual input channels. Each ADC command independently makes a channel and conversion type selection. Each CMDLa[ADCH] channel selection has an associated A side and an associated B side input. Each CMDLa[ADCH] pair can optionally be converted in a differential mode but only limited pairs are intended to be converted as differential channels (i.e., adjacent pins that have been designed with matched impedance). For the pin pairings available for differential conversions for your device, see the Chip Configuration details.

NOTE

Some of the input channels are from on-chip sources such as temperature sensors and reference voltage sources and may only be connected to individual instances of the ADC module. Some of the input channel options in the bitfield-setting descriptions may not be available for your device. Refer to chip-specific information for the channels supported on this device.

80.5 Initialization

80.5.1 Calibration

The ADC module has multiple calibration functions that must be executed as part of ADC setup to achieve the specified accuracy. Calibration must be run after any reset and before a conversion is initiated. Prior to offset calibration or calibration, the user must configure the ADC's clock source and frequency according to the application's clock source availability and needs. The ADC must be enabled (CTRL[ADCEN] = 0x1) before a calibration function will run. If calibration is requested while the ADC is actively converting, that sequence completes before calibration function is initiated.

Improved accuracy can be achieved during the calibration routines by averaging multiple conversions. It is recommended to set CTRL[CAL_AVGS] for minimum of 256 averaging during calibration steps. If the application uses the ADC in a wide variety of configurations, the configuration for which the highest accuracy is required should be selected, or multiple calibrations can be done for the different configurations.

16 bit Offset Calibration function:

The OFSTRIM16 register is used to trim for ADC comparator offset voltage for 16 bit ADC conversions. The ADC supports an offset calibration function in which the OFSTRIM16 register is automatically updated. Below are the steps for executing the function.

1. Configure for the desired averaging via CTRL[CAL_AVGS] bitfield. The minimum recommended setting is for 256 averaging (CTRL[CAL_AVGS] set to 0x8 or greater).
2. Initiate 16 bit Offset Calibration by setting CTRL[CALOFS] with CTRL[CALOFSMODE] set.
3. Poll the STAT[CAL_RDY] flag. When STAT[CAL_RDY] is asserted, the offset calibration function has completed and the OFSTRIM16 register has updated.

12 bit Offset Calibration function:

The OFSTRIM12 register is used to trim for ADC comparator offset voltage for 12 bit ADC conversions. The ADC supports an offset calibration function in which the OFSTRIM12 register is automatically updated. Below are the steps for executing the function.

1. Configure for the desired averaging via CTRL[CAL_AVGS] bitfield. The minimum recommended setting is for 256 averaging (CTRL[CAL_AVGS] set to 0x8 or greater).

2. Initiate 12 bit Offset Calibration by setting CTRL[CALOFS] with CTRL[CALOFSMODE] clear.
3. Poll the STAT[CAL_RDY] flag. When STAT[CAL_RDY] is asserted, the offset calibration function has completed and the OFSTRIM12 register has updated.

Calibration function:

The ADC includes hardware calibration logic in which the A-side and B-side converters are calibrated for gain error and linearity error correction of the raw conversion result. The GCR0[GCALR] and CAL_GAR registers control calibration for the A-side converter. The GCR1[GCALR] and CAL_GBR registers control calibration for the B-side converter. The ADC supports a calibration function in which the CAL_GAR and CAL_GBR registers are automatically updated. The calibration function also updates GCC0[GAIN_CAL] (associated with A-side calibration) and GCC1[GAIN_CAL] (associated with B-side calibration). For A-side calibration, a software calculation is required to derive GCR0[GCALR] from GCC0[GAIN_CAL]. For B-side calibration, a corresponding calculation is needed to derive GCR1[GCALR] from GCC1[GAIN_CAL]. Below are the steps for completing calibration setup.

1. Execute the 16 bit Offset Calibration function described above. The OFSTRIM16 register is used during calibration to trim for comparator offset voltage.
2. Configure for the desired averaging via CTRL[CAL_AVGS] bitfield. The minimum recommended setting is for 256 averaging (CTRL[CAL_AVGS] set to 0x8 or greater).
3. Initiate the calibration routine by setting CTRL[CAL_REQ]. Once set, the CTRL[CAL_REQ] bit remains set until the CAL routine has been accepted by the ADC. After acceptance, CTRL[CAL_REQ] is automatically cleared.
4. The A-side calibration data is updated first. Poll the GCC0[RDY] flag. When it is asserted, the hardware controlled A-side calibration operation is complete and CAL_GAR and GCC0[GAIN_CAL] registers are updated. The updated value in GCC0[GAIN_CAL] is needed for further software processing described in the following steps.
5. Read the GCC0[GAIN_CAL] register and store for use in the gain_adjustment calculation.
6. Calculate the A-side gain_adjustment = (131072)/(131072-GCC0[GAIN_CAL]). GCC0[GAIN_CAL] is 2's complement value. This results in a floating point value between 0 and 2.
7. Convert the floating point value to its integer component (0 or 1) and its fractional component rounded to 16-bits. The integer value is stored to GCR0[GCALR[16]] and the fractional component is stored to GCR0[GCALR[15:0]]. Write this value to the GCR0[GCALR] register.
8. Next, execute the same calculation for the B-side converter. Poll the GCC1[RDY] flag. When it is asserted, the hardware controlled B-side calibration operation is complete and CAL_GBR and GCC1[GAIN_CAL] registers are updated. The updated value in GCC1[GAIN_CAL] is needed for further software processing.
9. Read the GCC1[GAIN_CAL] register and store for use in the gain_adjustment calculation.
10. Calculate the B-side gain_adjustment = (131072)/(131072-GCC1[GAIN_CAL]). Formatting for the value is the same as the A-side.
11. Write this value to the GCR1[GCALR] register.
12. Once GCR0[GCALR] and GCR1[GCALR] contain the results from the gain_adjustment calculations, set the GCR0[RDY] and GCR1[RDY] flags to indicate they are valid. It is acceptable for the GCRa[GCALR] and corresponding GCRa[RDY] bit to be updated on the same write cycle.

After completing the steps above, the calibration sequence is complete and the STAT[CAL_RDY] flag is set. The STAT[CAL_RDY] flag remains set until the user resets the system or requests a new calibration sequence.

When STAT[CAL_RDY] is set, the ADC is configured to run in calibrated mode. Each conversion uses a combination of linearity and gain calibration results to correct SAR data. Calibration conversion latency is required to process each sample. However, due to the pipelined nature of data and control sequences, each conversion can still be initiated without experiencing this calibration delay.

The diagram below shows how calibration data is both calculated and utilized. The left portion of this image represents the calibration sequencing, and the right portion represents how calibration data is used.

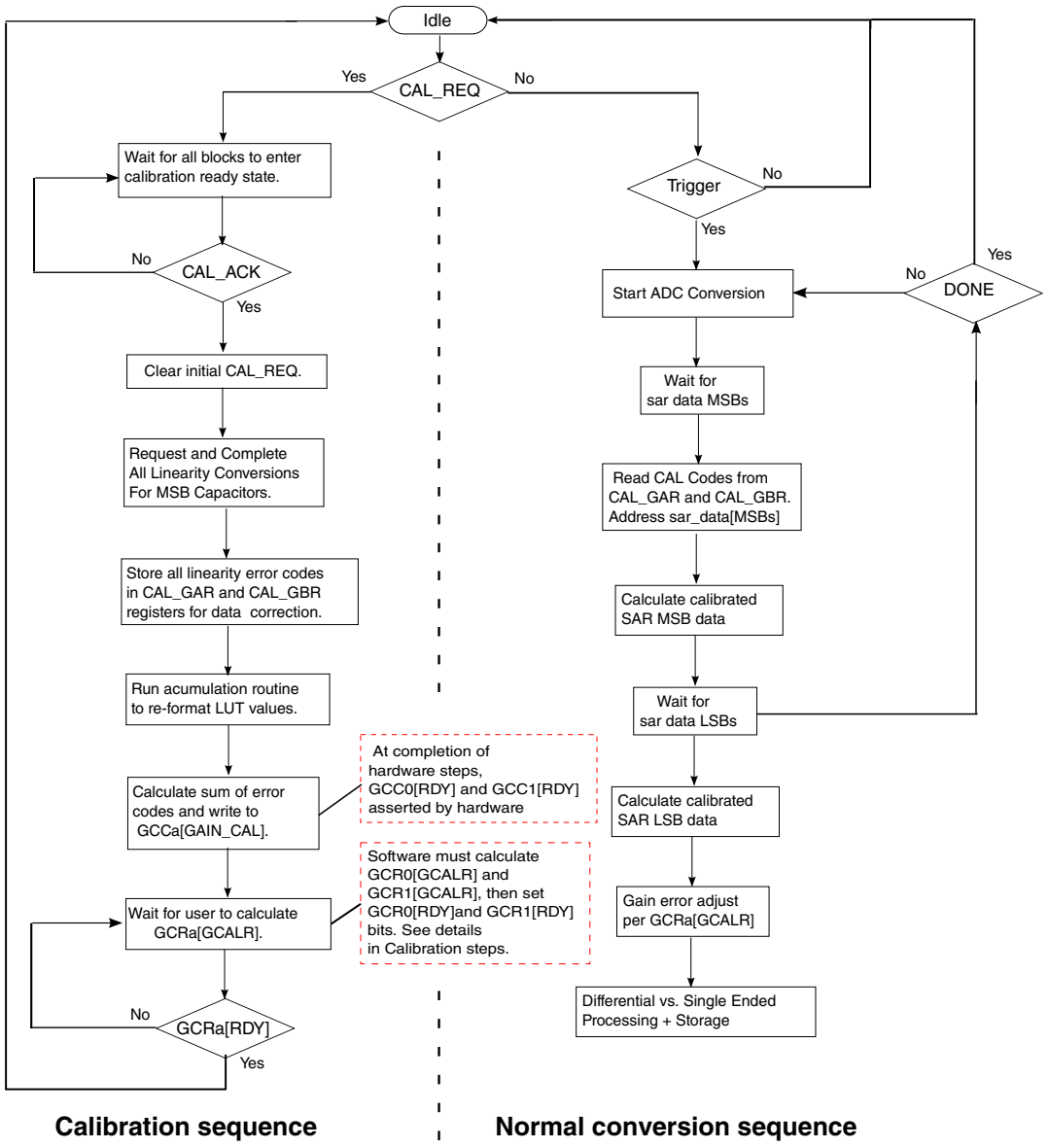


Figure 615. Simplified flow diagram for ADC calibration and conversion

80.5.1.1 Calibration General A-Side and B-Side Register valid ranges

The general calibration value registers CAL_GARa and CAL_GBRa have 33 registers in each array. The data in each register is a signed 16-bit value but has a variable range of possible values. The following table defines the valid range each register supports.

Table 1109. Calibration General Registers Valid Values

Element CAL_GxR[N]	Valid Range
N = 0x00, 0x20	–1024 to +1023
N = 0x01	–2048 to +2047

Table continues on the next page...

Table 1109. Calibration General Registers Valid Values (continued)

Element CAL_GxR[N]	Valid Range
N = 0x02–0x03	–4096 to +4095
N = 0x04–0x07	–8192 to +8191
N = 0x08–0x0F	–16384 to +16383
N = 0x10–0x1F	–32768 to +32767

These registers are typically updated automatically during the self calibration sequence. To reduce the latency associated with ADC setup, the CAL_GxR values from a calibration sequence can be stored in non-volatile memory after an initial calibration and written to the CAL_GxR registers via software prior to the first ADC conversion. If these registers are set to values not generated by the calibration function, the linearity error specifications may not be met. Note, these values can only be written in a single access, byte accesses are not supported.

80.6 ADC register descriptions

This section describes the ADC registers.

80.6.1 ADC memory map

ADC1 base address: 4260_0000h

ADC2 base address: 42E0_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	R	0200_2C1Bh
4h	Parameter Register (PARAM)	32	R	0F04_1008h
10h	Control Register (CTRL)	32	RW	0000_0020h
14h	Status Register (STAT)	32	RW	0000_0000h
18h	Interrupt Enable Register (IE)	32	RW	0000_0000h
1Ch	DMA Enable Register (DE)	32	RW	0000_0000h
20h	Configuration Register (CFG)	32	RW	0080_0000h
24h	Pause Register (PAUSE)	32	RW	0000_0000h
34h	Software Trigger Register (SWTRIG)	32	RW	0000_0000h
38h	Trigger Status Register (TSTAT)	32	RW	0000_0000h
40h	Offset Trim 16 bit Register (OFSTRIM16)	32	RW	0000_0000h
44h	Offset Trim 12 bit Register (OFSTRIM12)	32	RW	0000_0000h
A0h - BCh	Trigger Control Register (TCTRL0 - TCTRL7)	32	RW	0000_0000h
E0h - E4h	FIFO Control Register (FCTRL0 - FCTRL1)	32	RW	0000_0000h
F0h - F4h	Gain Calibration Control (GCC0 - GCC1)	32	R	0000_0000h
F8h - FCh	Gain Calculation Result (GCR0 - GCR1)	32	RW	0001_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
100h	Command Low Buffer Register (CMDL1)	32	RW	0080_2000h
104h	Command High Buffer Register (CMDH1)	32	RW	0000_0000h
108h	Command Low Buffer Register (CMDL2)	32	RW	0080_2000h
10Ch	Command High Buffer Register (CMDH2)	32	RW	0000_0000h
110h	Command Low Buffer Register (CMDL3)	32	RW	0080_2000h
114h	Command High Buffer Register (CMDH3)	32	RW	0000_0000h
118h	Command Low Buffer Register (CMDL4)	32	RW	0080_2000h
11Ch	Command High Buffer Register (CMDH4)	32	RW	0000_0000h
120h	Command Low Buffer Register (CMDL5)	32	RW	0080_2000h
124h	Command High Buffer Register (CMDH5)	32	RW	0000_0000h
128h	Command Low Buffer Register (CMDL6)	32	RW	0080_2000h
12Ch	Command High Buffer Register (CMDH6)	32	RW	0000_0000h
130h	Command Low Buffer Register (CMDL7)	32	RW	0080_2000h
134h	Command High Buffer Register (CMDH7)	32	RW	0000_0000h
138h	Command Low Buffer Register (CMDL8)	32	RW	0080_2000h
13Ch	Command High Buffer Register (CMDH8)	32	RW	0000_0000h
140h	Command Low Buffer Register (CMDL9)	32	RW	0080_2000h
144h	Command High Buffer Register (CMDH9)	32	RW	0000_0000h
148h	Command Low Buffer Register (CMDL10)	32	RW	0080_2000h
14Ch	Command High Buffer Register (CMDH10)	32	RW	0000_0000h
150h	Command Low Buffer Register (CMDL11)	32	RW	0080_2000h
154h	Command High Buffer Register (CMDH11)	32	RW	0000_0000h
158h	Command Low Buffer Register (CMDL12)	32	RW	0080_2000h
15Ch	Command High Buffer Register (CMDH12)	32	RW	0000_0000h
160h	Command Low Buffer Register (CMDL13)	32	RW	0080_2000h
164h	Command High Buffer Register (CMDH13)	32	RW	0000_0000h
168h	Command Low Buffer Register (CMDL14)	32	RW	0080_2000h
16Ch	Command High Buffer Register (CMDH14)	32	RW	0000_0000h
170h	Command Low Buffer Register (CMDL15)	32	RW	0080_2000h
174h	Command High Buffer Register (CMDH15)	32	RW	0000_0000h
200h - 20Ch	Compare Value Register (CV1 - CV4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
300h - 304h	Data Result FIFO Register (RESFIFO0 - RESFIFO1)	32	R	0000_0000h
400h - 480h	Calibration General A-Side Registers (CAL_GAR0 - CAL_GAR32)	32	RW	0000_0000h
500h - 580h	Calibration General B-Side Registers (CAL_GBR0 - CAL_GBR32)	32	RW	0000_0000h
FF8h	Configuration 2 Register (CFG2)	32	RW	0000_0000h

80.6.2 Version ID Register (VERID)

Offset

Register	Offset
VERID	0h

Function

The Version ID register indicates the version integrated for this instance on the device and also indicates inclusion/exclusion of several optional features.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	NUM_FIFO			NUM_SEC	CALOFSI	IADCKI	VR1RNGI	0	CSW			MVI	0	DIFFEN	RES
W																
Reset	0	0	1	0	1	1	0	0	0	0	0	1	1	0	1	1

Fields

Field	Function
31-24 MAJOR	Major Version Number This read-only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read-only field returns the minor version number for the module specification.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 —	Reserved This read-only field is reserved and always has the value 0.
14-12 NUM_FIFO	Number of FIFOs This read-only field indicates the number of result FIFOs implemented in the design. 000b - N/A 001b - This design supports one result FIFO. 010b - This design supports two result FIFOs. 011b - This design supports three result FIFOs. 100b - This design supports four result FIFOs.
11 NUM_SEC	Number of Single Ended Outputs Supported This read-only field indicates the number of single ended channels which can be processed simultaneously. 0b - This design supports one single ended conversion at a time. 1b - This design supports two simultaneous single ended conversions.
10 CALOFSI	Calibration Function Implemented This read-only field indicates if the ADC contains hardware calibration functions. When supported, CTRL[CALOFS] and CTRL[CAL_REQ] can be used to request the calibration routines to run. 0b - Calibration Not Implemented. 1b - Calibration Implemented.
9 IADCKI	Internal ADC Clock Implemented This read-only field indicates if this implementation of the ADC block includes an internal clock source. When supported, the CFG[ADCKEN] field is available and documented for enabling the clock source. When available, this clock source is used in clock selection logic external to the module for use within the system. 0b - Internal clock source not implemented. 1b - Internal clock source (and CFG[ADCKEN]) implemented.
8 VR1RNGI	Voltage Reference 1 Range Control Bit Implemented This read-only field indicates if a control bit is implemented for selecting the input voltage range on Voltage Reference Option 1.When range control is required, the CFG[VREF1RNG] field is available and documented for controlling the Voltage Reference Option 1. 0b - Range control not required. CFG[VREF1RNG] is not implemented. 1b - Range control required. CFG[VREF1RNG] is implemented.
7 —	Reserved This read-only field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-4 CSW	Channel Scale Width This read-only field indicates if channel scaling is supported by this implementation. When supported, each command buffer has a control field (CMDLa[CSCALE]) for setting input scaling. 000b - Channel scaling not supported. CSCALE control field not implemented. 001b - Channel scaling supported. 1-bit CSCALE control field. 110b - Channel scaling supported. 6-bit CSCALE control field.
3 MVI	Multi Vref Implemented This read-only field indicates if multiple Voltage Reference High inputs are supported by this implementation. When multiple voltage references are supported, the CFG[REFSEL] field is available and documented for selecting the Voltage Reference High options. 0b - Single voltage reference high (VREFH) input supported. 1b - Multiple voltage reference high (VREFH) inputs supported.
2 —	Reserved This read-only field is reserved and always has the value 0.
1 DIFFEN	Differential Supported This read-only field indicates if differential operation is supported by this implementation. When supported, each command buffer has control fields (CMDLa[CTYPE]) for configuring for differential operation and expanding the number of supported channels from 32 to 64. 0b - Differential operation not supported. 1b - Differential operation supported. CMDLa[CTYPE] controls fields implemented.
0 RES	Resolution This read-only field indicates the maximum accuracy supported by this implementation. When RES=1, each command buffer has a control field (CMDLa[MODE]) for selecting resolution of conversions for the associated command. 0b - Up to 13-bit differential/12-bit single ended resolution supported. 1b - Up to 16-bit differential/16-bit single ended resolution supported.

80.6.3 Parameter Register (PARAM)

Offset

Register	Offset
PARAM	4h

Function

The Parameter register indicates the size of several variable integration options for this instance on the device.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMD_NUM								CV_NUM							
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIFOSIZE								TRIG_NUM							
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-24 CMD_NUM	Command Buffer Number Number of command buffers implemented.
23-16 CV_NUM	Compare Value Number Number of compare value registers implemented.
15-8 FIFOSIZE	Result FIFO Depth The maximum number of conversion datawords that can be stored in the result FIFO before an overflow occurs. This field is read-only. 0000_0001b - Result FIFO depth = 2 dataword. 0000_0100b - Result FIFO depth = 4 datawords. 0000_1000b - Result FIFO depth = 8 datawords. 0001_0000b - Result FIFO depth = 16 datawords. 0010_0000b - Result FIFO depth = 32 datawords. 0100_0000b - Result FIFO depth = 64 datawords.
7-0 TRIG_NUM	Trigger Number Number of Triggers implemented.

80.6.4 Control Register (CTRL)

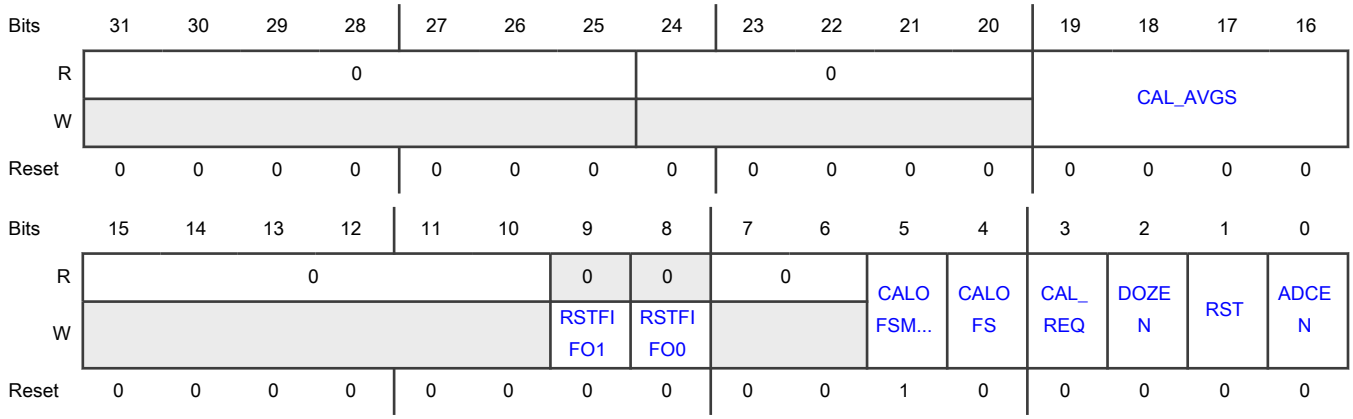
Offset

Register	Offset
CTRL	10h

Function

Control Register includes primary control bits.

Diagram



Fields

Field	Function
31-25 —	Reserved This read-only field is reserved and always has the value 0.
24-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 CAL_AVGS	Auto-Calibration Averages Selects how many ADC conversions are averaged to calculate each calibration value. Selecting a higher number of averages will lead to more accurate conversions after completing calibration. The recommended minimum setting is for 256 averaging (CAL_AVGS set to 0x8 or greater). The CAL_AVGS bit field applies to both CALOFS and CAL_REQ calibration types. This value should be fixed when requesting and running calibration with CTRL[CAL_REQ] or CTRL[CALOFS]. 0000b - Single conversion. 0001b - 2 conversions averaged. 0010b - 4 conversions averaged. 0011b - 8 conversions averaged. 0100b - 16 conversions averaged. 0101b - 32 conversions averaged. 0110b - 64 conversions averaged. 0111b - 128 conversions averaged. 1000b - 256 conversions averaged. 1001b - 512 conversions averaged. 1010b - 1024 conversions averaged.
15-10 —	Reserved This read-only field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 RSTFIFO1	Reset FIFO 1 0b - No effect. 1b - FIFO 1 is reset.
8 RSTFIFO0	Reset FIFO 0 0b - No effect. 1b - FIFO 0 is reset.
7-6 —	Reserved This read-only field is reserved and always has the value 0.
5 CALOFSMODE	Configure Mode for Offset Calibration Function When CALOFSMODE=1, Executing the offset calibration function configures the analog block for 16-bit mode and results are updated to the OFSTRIM16 register. When CALOFSMODE=0, Executing the offset calibration function configures the analog block for 12-bit mode and results are updated to the OFSTRIM12 register. 0b - Configure offset calibration for 12-bit mode. 1b - Configure offset calibration for 16-bit mode.
4 CALOFS	Offset Calibration Request Setting CALOFS to 1 initiates a hardware offset calibration calculation. The CTRL[CALOFSMODE] setting controls whether a 16 bit Offset Calibration or a 12 bit Offset Calibration is executed. Any conversions in progress are completed before launching the offset calibration function. After being accepted, the ADC calculates the offset value and depending on the CTRL[CALOFSMODE] setting updates either the OFSTRIM16 or the OFSTRIM12 register automatically. The CALOFS bit is cleared by hardware upon completion of the offset calibration calculation. The STAT[CAL_RDY] bit indicates when the offset calibration routine has completed. 0b - Calibration function disabled 1b - Request for offset calibration function
3 CAL_REQ	Auto-Calibration Request Request the hardware calibration routine to run. This bit is automatically cleared when the system accepts the hardware calibration request. Note, the status flag STAT[CAL_RDY] will indicate when this calibration routine has been completed. 0b - No request for hardware calibration has been made. 1b - A request for hardware calibration has been made
2 DOZEN	Doze Enable Controls system transition to Deepsleep/Sleep power modes while ADC is converting. When DOZEN is clear, immediate entries to Deepsleep or Sleep modes are allowed and ADC conversion functions remain enabled. Any conversion in progress is not disrupted. Note that the selected clock source provided from the on-chip clock source must be able to continue operating. When the DOZEN bit is set, If the ADC is actively converting, when the system is entering in low power mode, the ADC waits for

Table continues on the next page...

Table continued from the previous page...

Field	Function
	the current averaging iteration/FIFO storage to complete before acknowledging low power entry request. After system entry to the low power state, the ADC is kept in inactive state until the system exits the low power state. When the system is entering Deep Powerdown mode, the DOZEN bit is ignored and the ADC waits for the current transfer to complete any pending operation. 0b - ADC is enabled in low power mode. 1b - ADC is disabled in low power mode.
1 RST	Software Reset Resets all internal logic and registers, except the Control Register. Remains set until cleared by software. 0b - ADC logic is not reset. 1b - ADC logic is reset.
0 ADCEN	ADC Enable 0b - ADC is disabled. 1b - ADC is enabled.

80.6.5 Status Register (STAT)

Offset

Register	Offset
STAT	14h

Function

The Status Register provides the current status of the ADC module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				CMDACT				0				TRGACT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				ADC_ACT...	CAL_RDY	TCOMP_I...	TEXC_INT	0				FOF1	RDY1	FOF0	RDY0
W							W1C	W1C					W1C		W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved This read-only field is reserved and always has the value 0.
27-24 CMDACT	Command Active CMDACT is a read-only status field indicating the command that is actively being processed. Note, commands are only shown here when they are actively being processed by the SAR routine. Use CMDACT in conjunction with ADC_ACTIVE to determine which conversion is running. 0000b - No command is currently in progress. 0001b - Command 1 currently being executed. 0010b - Command 2 currently being executed. 0011b-1111b - Associated command number is currently being executed.
23-19 —	Reserved This read-only field is reserved and always has the value 0.
18-16 TRGACT	Trigger Active TRGACT is a read-only status field indicating the trigger actively being processed. This can be used to determine which trigger is running through a trigger delay or being converted by the SAR routine. The command associated with TRGACT will be running through a conversion when CMDACT is non-zero. 000b - Command (sequence) associated with Trigger 0 currently being executed. 001b - Command (sequence) associated with Trigger 1 currently being executed. 010b - Command (sequence) associated with Trigger 2 currently being executed. 011b-111b - Command (sequence) from the associated Trigger number is currently being executed.
15-12 —	Reserved This read-only field is reserved and always has the value 0.
11 ADC_ACTIVE	ADC Active This flag shows if the module is currently processing a conversion, or has pending triggers to service. When the ADC is IDLE, this bit is set to 0b0. 0b - The ADC is IDLE. There are no pending triggers to service and no active commands are being processed. 1b - The ADC is processing a conversion, running through the power up delay, or servicing a trigger.
10 CAL_RDY	Calibration Ready This flag shows if the adc CTRL[CAL_REQ] or CTRL[CALOFS] request for calibration has been completed and the results are ready. This flag is automatically cleared when a new hardware calibration or offset calibration request is made.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Calibration is incomplete or hasn't been ran. 1b - The ADC is calibrated.
9 TCOMP_INT	Interrupt Flag For Trigger Completion When a trigger sequence has been completed (all associated commands have been run) the TCOMP_INT flag is asserted. Note that IE[TCOMP_IE] must be set for the specific trigger source to flag an interrupt upon completion. 0b - Either IE[TCOMP_IE] is set to 0, or no trigger sequences have run to completion. 1b - Trigger sequence has been completed and all data is stored in the associated FIFO.
8 TEXC_INT	Interrupt Flag For High Priority Trigger Exception When a high priority trigger exception has occurred and CFG[TRES] = 0, this flag is asserted. This flag only asserts if trigger exception interrupts are enabled (IE[TEXC_IE] = 0x1). This flag can be used in conjunction with TSTAT[TEXC_NUM] to resolve which trigger source was interrupted. 0b - No trigger exceptions have occurred. 1b - A trigger exception has occurred and is pending acknowledgement.
7-4 —	Reserved This read-only field is reserved and always has the value 0.
3 FOF1	Result FIFO1 Overflow Flag Indicates that more data has been written to the Result FIFO1 than it can hold. The newer data is not stored and the FIFO remains holding the original contents. This flag asserts regardless of the value of IE[FOFIE1]. However, an interrupt request is issued only if IE[FOFIE1] is set. This flag is cleared by writing a 1. 0b - No result FIFO1 overflow has occurred since the last time the flag was cleared. 1b - At least one result FIFO1 overflow has occurred since the last time the flag was cleared.
2 RDY1	Result FIFO1 Ready Flag Indicates when the number of valid datawords in the result FIFO1 is greater than the watermark level set in the FCTRL[FWMARK] bitfield. This flag asserts regardless of the value of IE[FWMIE1]. However, an interrupt request or DMA request occurs only when the associated control bit (IE[FWMIE1] and DE[FWMDE1]) is set. This flag is cleared when the FCTRLa[FCOUNT] (which decrements on each read of the RESFIFO register) is less than or equal to the watermark level set in the FCTRL[FWMARK] bitfield. 0b - Result FIFO1 data level not above watermark level. 1b - Result FIFO1 holding data above watermark level.
1 FOF0	Result FIFO 0 Overflow Flag Indicates that more data has been written to the Result FIFO 0 than it can hold. The newer data is not stored and the FIFO remains holding the original contents. This flag asserts regardless of the value of

Table continues on the next page...

Table continued from the previous page...

Field	Function
	IE[FOFIE0]. However, an interrupt request is issued only if IE[FOFIE0] is set. This flag is cleared by writing a 1. 0b - No result FIFO 0 overflow has occurred since the last time the flag was cleared. 1b - At least one result FIFO 0 overflow has occurred since the last time the flag was cleared.
0 RDY0	Result FIFO 0 Ready Flag Indicates when the number of valid datawords in the result FIFO 0 is greater than the watermark level set in the FCTRL[FWMARK] bitfield. This flag asserts regardless of the value of IE[FWMIE0]. However, an interrupt request or DMA request occurs only when the associated control bit (IE[FWMIE0] and DE[FWMDE0]) is set. This flag is cleared when the FCTRLa[FCOUNT] (which decrements on each read of the RESFIFO register) is less than or equal to the watermark level set in the FCTRL[FWMARK] bitfield. 0b - Result FIFO 0 data level not above watermark level. 1b - Result FIFO 0 holding data above watermark level.

80.6.6 Interrupt Enable Register (IE)

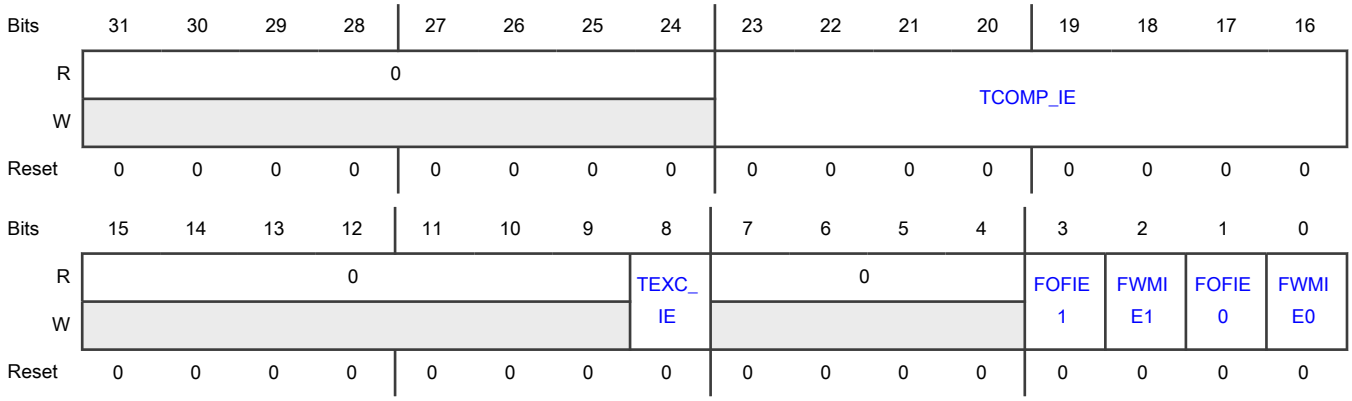
Offset

Register	Offset
IE	18h

Function

Interrupt Enable Register includes system interrupt masking control bits.

Diagram



Fields

Field	Function
31-24 —	Reserved This read-only field is reserved and always has the value 0.
23-16 TCOMP_IE	Trigger Completion Interrupt Enable Each bit in TCOMP_IE corresponds to a trigger source. (TCOMP_IE[0] corresponds to trigger 0, TCOMP_IE[1] corresponds to trigger 1, ...) A trigger completion interrupt is used to indicate when a complete trigger command sequence has been executed. All results will be stored in the FIFO when a sequence is considered complete. 0000_0000b - Trigger completion interrupts are disabled. 0000_0001b - Trigger completion interrupts are enabled for trigger source 0 only. 0000_0010b - Trigger completion interrupts are enabled for trigger source 1 only. 0000_0011b-1111_1110b - Associated trigger completion interrupts are enabled. 1111_1111b - Trigger completion interrupts are enabled for every trigger source.
15-9 —	Reserved This read-only field is reserved and always has the value 0.
8 TEXC_IE	Trigger Exception Interrupt Enable This register field enables the ADC to assert an interrupt request when a high priority trigger exception occurs. TSTAT[TEXC_NUM] will contain the value of the corresponding trigger which was affected by the exception. 0b - Trigger exception interrupts are disabled. 1b - Trigger exception interrupts are enabled.
7-4 —	Reserved This read-only field is reserved and always has the value 0.
3 FOFIE1	Result FIFO1 Overflow Interrupt Enable Configures the module to generate overflow interrupt requests when FOF1 flag is asserted. 0b - No result FIFO1 overflow has occurred since the last time the flag was cleared. 1b - At least one result FIFO1 overflow has occurred since the last time the flag was cleared.
2 FWMIE1	FIFO1 Watermark Interrupt Enable Configures the module to generate watermark interrupt requests when STAT[RDY1] flag is asserted. 0b - FIFO1 watermark interrupts are not enabled. 1b - FIFO1 watermark interrupts are enabled.
1 FOFIE0	Result FIFO 0 Overflow Interrupt Enable Configures the module to generate overflow interrupt requests when FOF flag is asserted.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - FIFO 0 overflow interrupts are not enabled. 1b - FIFO 0 overflow interrupts are enabled.
0 FWMIE0	FIFO 0 Watermark Interrupt Enable Configures the module to generate watermark interrupt requests when STAT[RDY0] flag is asserted. 0b - FIFO 0 watermark interrupts are not enabled. 1b - FIFO 0 watermark interrupts are enabled.

80.6.7 DMA Enable Register (DE)

Offset

Register	Offset
DE	1Ch

Function

DMA Enable Register includes DMA request masking control bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														FWMDE1	FWMDE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved This read-only field is reserved and always has the value 0.
1 FWMDE1	FIFO1 Watermark DMA Enable Configures the module to generate DMA requests when STAT[RDY1] flag is asserted.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DMA request disabled. 1b - DMA request enabled.
0 FWMDE0	FIFO 0 Watermark DMA Enable Configures the module to generate DMA requests when STAT[RDY0] flag is asserted. 0b - DMA request disabled. 1b - DMA request enabled.

80.6.8 Configuration Register (CFG)

Offset

Register	Offset
CFG	20h

Function

The Configuration Register controls ADC functions that are common to all commands. The CFG cannot be changed while the CTRL[ADCEN] bit is set. Writes to CFG while CTRL[ADCEN] is set are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	PWREN	0				PUDLY							
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0			HPT_EXDI	TCMDRES	TRES	REFSEL			0		TPRICTRL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved This read-only field is reserved and always has the value 0.
30	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This read-only field is reserved and always has the value 0.
29 —	Reserved This read-only field is reserved and always has the value 0.
28 PWREN	ADC Analog Pre-Enable Enables the ADC analog circuits. When setting CFG[PWREN], user code should delay for a period exceeding the analog startup time of $t_{ADCSTUP}$ before enabling the ADC for operation. The module is still operational even when CFG[PWREN] is left clear but command execution start is delayed for a period defined by CFG[PUDLY]. Refer to the device datasheet for ADC idle and ADC active power consumption parameters. 0b - ADC analog circuits are only enabled while conversions are active. Performance is affected due to analog startup delays. 1b - ADC analog circuits are pre-enabled and ready to execute conversions without startup delays (at the cost of higher DC current consumption). Note that a single power up delay (CFG[PUDLY]) is executed immediately once PWREN is set, and any detected trigger does not begin ADC operation until the power up delay time has passed. After this initial delay expires the analog remains pre-enabled and no additional delays are executed.
27-24 —	Reserved This read-only field is reserved and always has the value 0.
23-16 PUDLY	Power Up Delay The PUDLY must be programmed to a non-zero value to enable the ADC. Note, the PUDLY is executed in one of two modes depending on the value configured in PWREN: When CFG[PWREN]=0b0, the ADC analog circuits are only powered on while the module is active. The delay defined by CFG[PUDLY] is executed after an initial trigger transitions the ADC from its Idle state. This delay allows time for the analog circuits to stabilize after being powered on. The startup delay count of (PUDLY*4) ADCK cycles must result in a longer delay than the analog startup time of $t_{ADCSTUP}$. Accuracy of the initial conversion(s) after activation is degraded if CFG[PUDLY] is set to too small a value. At the end of active conversions, if there are no subsequent pending conversions, the ADC analog is automatically reverted back to its low power idle state and the PUDLY runs again after the ADC is awakened with a later trigger. When CFG[PWREN]=0b1 prior to ADC activation, the analog circuits are pre-enabled and the activation delay defined by CFG[PUDLY] is executed immediately. After the delay has been executed, the analog remains enabled regardless of ADC activity. This configuration has the benefit of beginning conversions immediately after trigger event detection at the cost of increased DC power consumption of the analog circuits. Note, the ADC does not begin an initial conversion until the PUDLY has been executed regardless of the PWREN setting.
15 —	Reserved This read-only field is reserved and always has the value 0.
14-11	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This read-only field is reserved and always has the value 0.
10 HPT_EXDI	High Priority Trigger Exception Disable Controls how higher priority trigger events are handled. See section Trigger detect and command execution for a detailed description on trigger exception handling. 0b - High priority trigger exceptions are enabled. 1b - High priority trigger exceptions are disabled.
9 TCMDRES	Trigger Command Resume Determines where to resume from a high priority trigger exception. CFG[TRES] must be asserted for TCMDRES to be used. If asserted, the interrupted trigger sequence is resumed from the command where it was interrupted. If de-asserted the trigger sequence is restarted from the beginning. **NOTE** The trigger interrupted when TCMDRES = 1b, will be resumed at the command it was interrupted and if in LOOP mode, the CMDHa[LOOP] count will always re-start back at 1st loop count iteration. 0b - Trigger sequences interrupted by a high priority trigger exception is automatically restarted. 1b - Trigger sequences interrupted by a high priority trigger exception is resumed from the command executing before the exception.
8 TRES	Trigger Resume Enable Determines what happens after completing a high priority trigger exception. If CFG[TRES] is asserted, the interrupted trigger sequence is resumed or restarted automatically. The sequence can be resumed along command boundaries or restarted from the beginning of a sequence. This is controlled by CFG[TCMDRES]. If CFG[TRES] is de-asserted an interrupted trigger sequence does not resume automatically. In this case, interrupted triggers can be resumed via software by monitoring STAT[TEXC_INT] bit (or ISR handling for a trigger exception interrupt when IE[TEXC_IE] is set). Software determines which trigger was interrupted by reading TSTAT[TEXC_NUM] and re-starts the trigger with a write of 1 to the corresponding SWTRIG[SWTx] bit. 0b - Trigger sequences interrupted by a high priority trigger exception are not automatically resumed or restarted. 1b - Trigger sequences interrupted by a high priority trigger exception are automatically resumed or restarted.
7-6 REFSEL	Voltage Reference Selection Selects the voltage reference high used for conversions. **NOTE** See the chip-specific ADC information on the voltage reference options specific to this packaged device. 00b - (Default) Option 1 setting.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Option 2 setting. 10b - Option 3 setting. 11b - Reserved
5-2 —	Reserved This read-only field is reserved and always has the value 0.
1-0 TPRICTRL	ADC Trigger Priority Control Controls how higher priority trigger exceptions are handled. See Trigger detect and command execution for a detailed explanation trigger event handling. 00b - If a higher priority trigger is detected during command processing, the current conversion is aborted and the new command specified by the trigger is started. 01b - If a higher priority trigger is received during command processing, the current command is stopped after completing the current conversion. If averaging is enabled, the averaging loop will be completed. However, CMDHa[LOOP] will be ignored and the higher priority trigger will be serviced. 10b - If a higher priority trigger is received during command processing, the current command will be completed (averaging, looping, compare) before servicing the higher priority trigger. 11b - RESERVED

80.6.9 Pause Register (PAUSE)

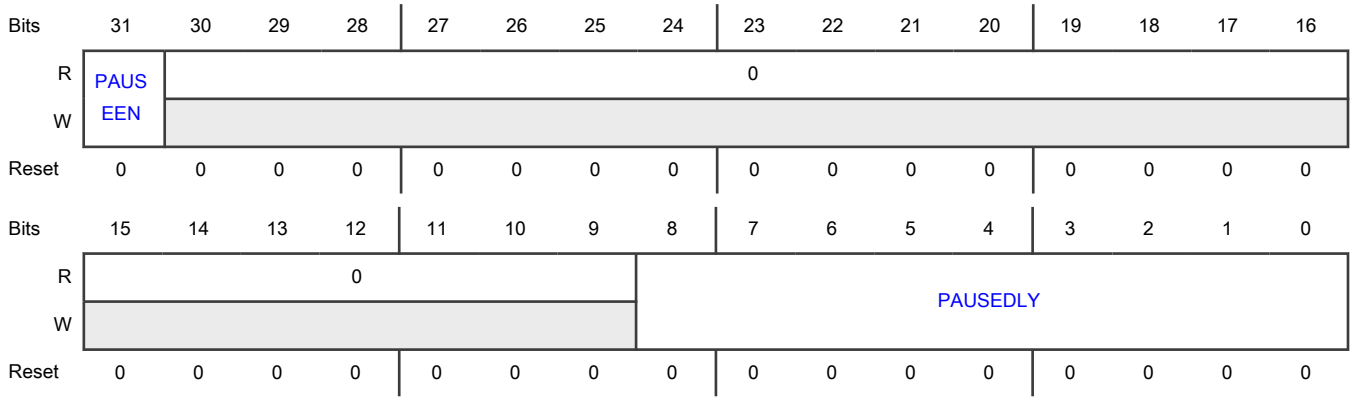
Offset

Register	Offset
PAUSE	24h

Function

The Pause Register controls an optional inserted delay between conversions. Note, the PAUSE register should not be modified while the CTRL[ADCEN] bit is set.

Diagram



Fields

Field	Function
31 PAUSEEN	PAUSE Option Enable Enables the ADC pausing function. When enabled, a programmable delay is inserted during command execution sequencing between LOOP iterations, between commands in a sequence, and between conversions when command is executing in for "Compare Until True" configuration. Note, CMDHa[WAIT_TRIG] and PAUSE are mutually exclusive. CMDHa[WAIT_TRIG] will take priority over PAUSE between commands when both are enabled. 0b - Pause operation disabled 1b - Pause operation enabled
30-9 —	Reserved This read-only field is reserved and always has the value 0.
8-0 PAUSEDLY	Pause Delay When PAUSEEN is set, the PAUSEDLY field controls the duration of pausing during command execution sequencing. The pause delay is a count of (PAUSEDLY*4) ADCK cycles.

80.6.10 Software Trigger Register (SWTRIG)

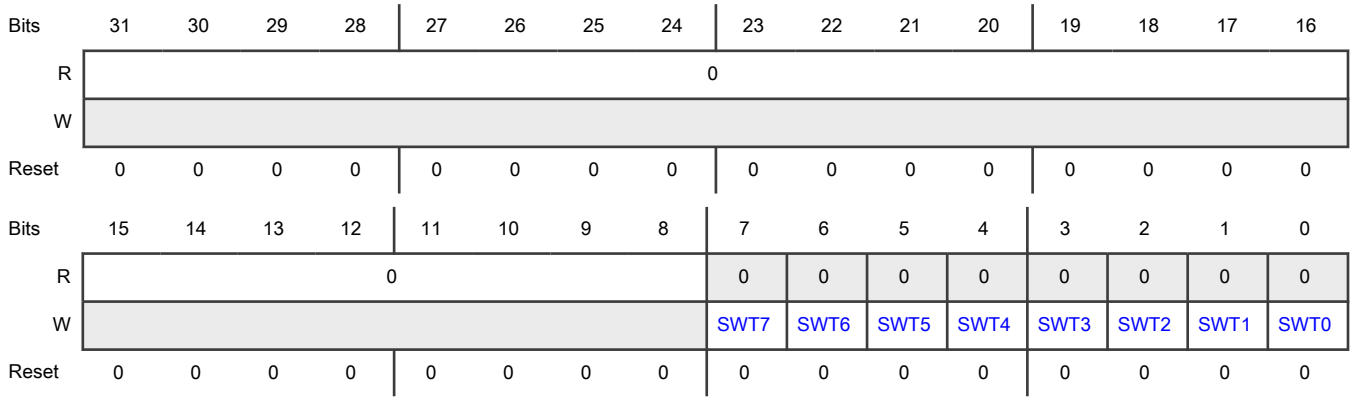
Offset

Register	Offset
SWTRIG	34h

Function

The Software Trigger Register (SWTRIG) is written to initiate software triggered conversions. Writes to SWTRIG register are ignored while CTRL[ADCEN] is clear. Note that there is an approximately 3 ADC Clock cycle synchronization delay between asserting CTRL[ADCEN] until SWTRIG can be accepted.

Diagram



Fields

Field	Function
31-8 —	Reserved This read-only field is reserved and always has the value 0.
7 SWT7	Software trigger 7 event Write 1 to SWT7 generates a trigger 7 event. Writing 1 to SWT7 is ignored while the trigger 7 event is being serviced or is pending. 0b - No trigger 7 event generated. 1b - Trigger 7 event generated.
6 SWT6	Software trigger 6 event Write 1 to SWT6 generates a trigger 6 event. Writing 1 to SWT6 is ignored while the trigger 6 event is being serviced or is pending. 0b - No trigger 6 event generated. 1b - Trigger 6 event generated.
5 SWT5	Software trigger 5 event Write 1 to SWT5 generates a trigger 5 event. Writing 1 to SWT5 is ignored while the trigger 5 event is being serviced or is pending. 0b - No trigger 5 event generated. 1b - Trigger 5 event generated.
4 SWT4	Software trigger 4 event Write 1 to SWT4 generates a trigger 4 event. Writing 1 to SWT4 is ignored while the trigger 4 event is being serviced or is pending. 0b - No trigger 4 event generated. 1b - Trigger 4 event generated.
3	Software Trigger 3 Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
SWT3	Write 1 to SWT3 generates a trigger 3 event. Writing 1 to SWT3 is ignored while the trigger 3 event is being serviced or is pending. 0b - No trigger 3 event generated. 1b - Trigger 3 event generated.
2 SWT2	Software Trigger 2 Event Write 1 to SWT2 generates a trigger 2 event. Writing 1 to SWT2 is ignored while the trigger 2 event is being serviced or is pending. 0b - No trigger 2 event generated. 1b - Trigger 2 event generated.
1 SWT1	Software Trigger 1 Event Write 1 to SWT1 generates a trigger 1 event. Writing 1 to SWT1 is ignored while the trigger 1 event is being serviced or is pending. 0b - No trigger 1 event generated. 1b - Trigger 1 event generated.
0 SWT0	Software Trigger 0 Event Write 1 to SWT0 generates a trigger 0 event. Writing 1 to SWT0 is ignored while the trigger 0 event is being serviced or is pending. 0b - No trigger 0 event generated. 1b - Trigger 0 event generated.

80.6.11 Trigger Status Register (TSTAT)

Offset

Register	Offset
TSTAT	38h

Function

This register contains status flags to indicate when trigger sequences have been completed or interrupted by a high priority trigger exception. Each bit in this register is set by hardware and cleared by software. To clear a bit in this register, write a 0b1 to the corresponding bit position.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								TCOMP_FLAG							
W									W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TEXC_NUM							
W									W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved This read-only field is reserved and always has the value 0.
23-16 TCOMP_FLAG	Trigger Completion Flag Each bit corresponds to a trigger. When the corresponding trigger sequence has been completed a bit in TCOMP_FLAG will be set. For example, if trigger 1 has been completed, then bit 17 will be set in TCOMP_FLAG. This register will only be active if the corresponding bit in IE[TCOMP_IE] = 1. Note, synchronization delay may be added to the end of the final conversion in a sequence prior to this flag being set. This delay can range from 2-4 ADC CLK cycles. 0000_0000b - No triggers have been completed. Trigger completion interrupts are disabled. 0000_0001b - Trigger 0 has been completed and trigger 0 has enabled completion interrupts. 0000_0010b - Trigger 1 has been completed and trigger 1 has enabled completion interrupts. 0000_0011b-1111_1110b - Associated trigger sequence has completed and has enabled completion interrupts. 1111_1111b - Every trigger sequence has been completed and every trigger has enabled completion interrupts.
15-8 —	Reserved This read-only field is reserved and always has the value 0.
7-0 TEXC_NUM	Trigger Exception Number Each bit corresponds to a trigger. When the corresponding trigger sequence has been interrupted by a high priority trigger exception a 1 will be set in its bit position. For example, if trigger 1 has been interrupted, then bit 1 will be set in TEXC_NUM. This register will be active regardless of the value in IE[TEXC_IE]. 0000_0000b - No triggers have been interrupted by a high priority exception. Or CFG[TRES] = 1. 0000_0001b - Trigger 0 has been interrupted by a high priority exception.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0000_0010b - Trigger 1 has been interrupted by a high priority exception. 0000_0011b-1111_1110b - Associated trigger sequence has interrupted by a high priority exception. 1111_1111b - Every trigger sequence has been interrupted by a high priority exception.

80.6.12 Offset Trim 16 bit Register (OFSTRIM16)

Offset

Register	Offset
OFSTRIM16	40h

Function

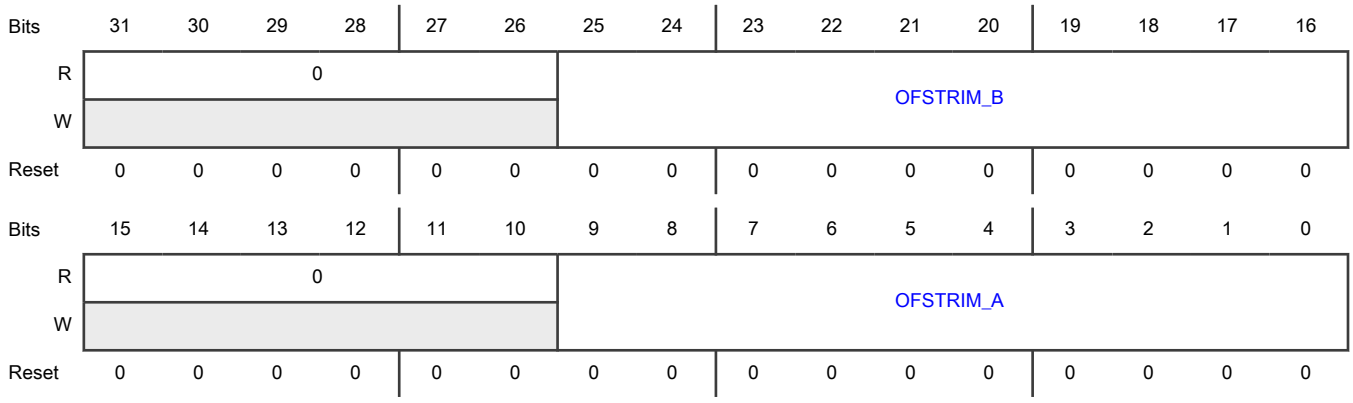
The Offset Trim 16 bit register is used to trim for comparator offset for 16 bit ADC conversions (CMDLa[MODE] == 0b1). The ADC supports an offset calibration function in which the OFSTRIM16 register is automatically updated.

To trigger the automatic update of OFSTRIM16, set the CTRL[CALOFS] bit with CTRL[CALOFSMODE] set to 0b1. Upon completion of the offset calibration sequence, the STAT[CAL_RDY] bit is set to 0b1 and the OFSTRIM16[OFSTRIM_A] and OFSTRIM16[OFSTRIM_B] bitfields are updated with 10-bit signed values.

The OFSTRIM16[OFSTRIM_A] and OFSTRIM16[OFSTRIM_B] values are used to minimize offset for normal 16 bit conversions (when CMDLa[MODE] is set). The OFSTRIM registers support a range of -512 to 511. Each increment is 1/2 LSB resulting in a programmable offset range of -256 to +255.5 LSB for 16-bit conversions.

The OFSTRIM16 trim values are also used in the calibration function.

Diagram



Fields

Field	Function
31-26 —	Reserved This read-only field is reserved and always has the value 0.
25-16 OFSTRIM_B	Trim for Offset in B-side Converter for 16-bit Conversions OFSTRIM_B is a 10-bit signed value between -512 and 511. This value is subtracted from the raw ADC B-side conversion result.
15-10 —	Reserved This read-only field is reserved and always has the value 0.
9-0 OFSTRIM_A	Trim for Offset in A-side Converter for 16-bit Conversions OFSTRIM_A is a 10-bit signed value between -512 and 511. This value is subtracted from the raw ADC A-side conversion result.

80.6.13 Offset Trim 12 bit Register (OFSTRIM12)

Offset

Register	Offset
OFSTRIM12	44h

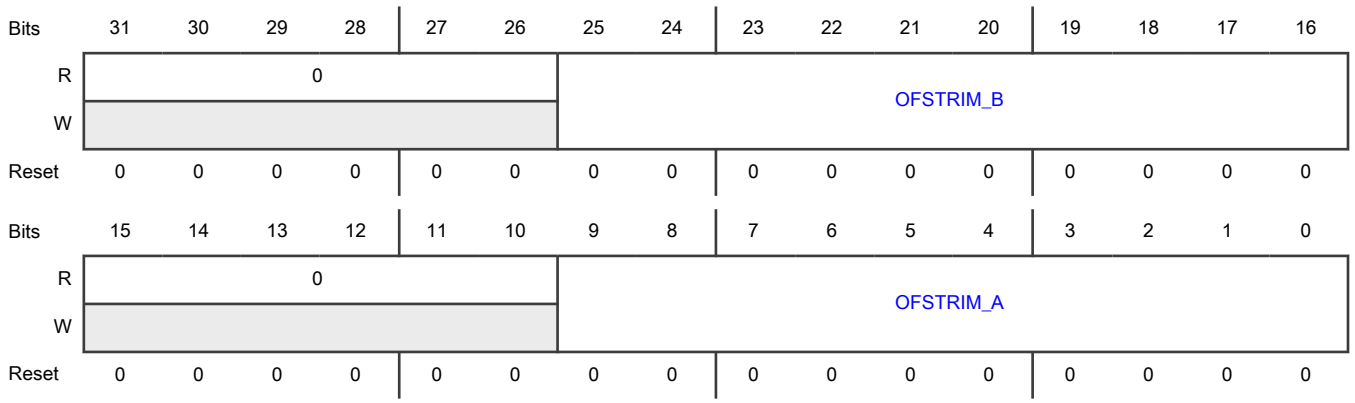
Function

The Offset Trim 12 bit register is used to trim for comparator offset for 12 bit ADC conversions (CMDLa[MODE] == 0b0). The ADC supports an offset calibration function in which the OFSTRIM12 register is automatically updated.

To trigger the automatic update of OFSTRIM12, set the CTRL[CALOFS] bit with CTRL[CALOFSMODE] set to 0b0. Upon completion of the offset calibration sequence, the STAT[CAL_RDY] bit is set to 0b1 and the OFSTRIM12[OFSTRIM_A] and OFSTRIM12[OFSTRIM_B] bitfields are updated with 10-bit signed values.

The OFSTRIM12[OFSTRIM_A] and OFSTRIM12[OFSTRIM_B] values are used to minimize offset for normal 12 bit conversions (when CMDLa[MODE] is clear). The OFSTRIM registers support a range of -512 to 511. Each increment is 1/32 LSB resulting in a programmable offset range of -16 to +15.96875 LSB for 12-bit conversions.

Diagram



Fields

Field	Function
31-26 —	Reserved This read-only field is reserved and always has the value 0.
25-16 OFSTRIM_B	Trim for Offset in B-side Converter for 12-bit Conversions OFSTRIM_B is a 10-bit signed value between -512 and 511. This value is subtracted from the raw ADC B-side conversion result.
15-10 —	Reserved This read-only field is reserved and always has the value 0.
9-0 OFSTRIM_A	Trim for Offset in A-side Converter for 12-bit Conversions OFSTRIM_A is a 10-bit signed value between -512 and 511. This value is subtracted from the raw ADC A-side conversion result.

80.6.14 Trigger Control Register (TCTRL0 - TCTRL7)

Offset

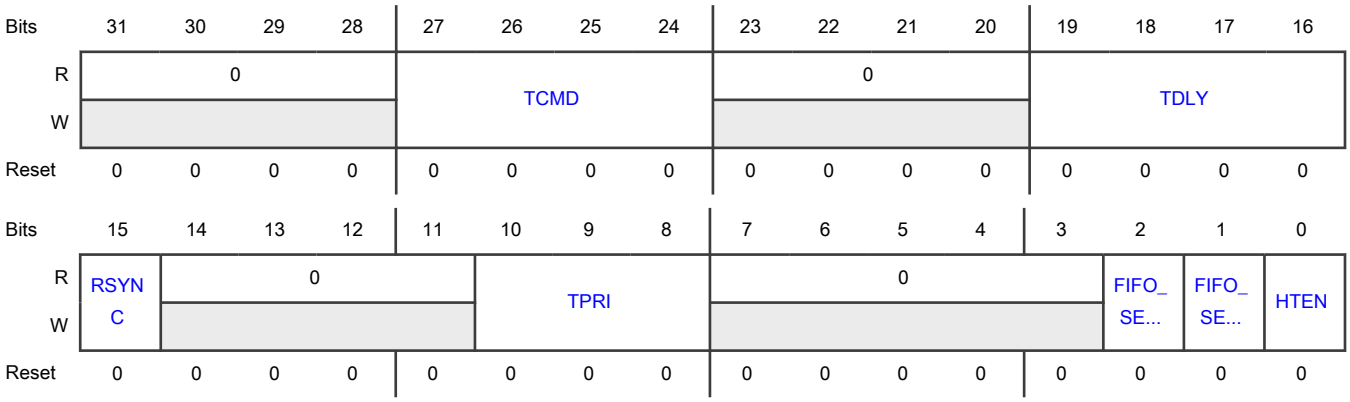
For a = 0 to 7:

Register	Offset
TCTRLa	A0h + (a × 4h)

Function

The Trigger Control (TCTRLa) register implements control fields associated with each implemented trigger source. When the ADC is actively executing commands, only one of the TCTRLa registers is actively controlling ADC conversions. The actively controlling TCTRLa register should not be updated while the ADC is active. A write to a TCTRLa registers while that trigger control register is controlling the ADC operation may cause unpredictable behavior.

Diagram



Fields

Field	Function
31-28 —	Reserved This read-only field is reserved and always has the value 0.
27-24 TCMD	Trigger Command Select Selects the command from command buffer to execute upon detection of the associated trigger event. When a trigger is received while the ADC is busy converting and the new trigger has higher priority than the trigger associated with the current command being executed, the command in this register, associated with the new trigger, is executed under control of CFG[TPRICTRL]. See section Trigger detect and command execution for a more detailed description on the relationship between CFG[TPRICTRL] and TCMD. 0000b - Not a valid selection from the command buffer. Trigger event is ignored. 0001b - CMD1 is executed 0010b-1110b - Corresponding CMD is executed 1111b - CMD15 is executed
23-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 TDLY	Trigger Delay Select Selects the trigger delay duration to wait at the start of servicing a trigger event. Each trigger source has an associated programmable delay prior to beginning an initial conversion. When TDLY field is clear, then no delay is incurred. When TDLY is set to a non-zero value, the duration for the delay is 2^{TDLY} ADCK cycles.
15 RSYNC	Trigger Resync If RSYNC is set to 1, this trigger source will be used as a resync trigger. A resync trigger can be configured to restart or abort a running trigger sequence (the target). Each resync trigger is configured to have a resync target. When a resync trigger is asserted, the target sequence will be aborted, the target FIFO destination(s) will be cleared, and the target sequence can optionally be restarted depending on CFG[TRES]. Multiple conditions must be met for a resync trigger to execute properly. 1. The resync trigger must have higher priority than the resync target. 2. The resync trigger TCTRLa[RSYNC] must be set to 0b1. 3. The resync target, specified by TCTRLa[TCMD], must be executing when the resync trigger is asserted. Please see Resync functionality for more information.
14-11 —	Reserved This read-only field is reserved and always has the value 0.
10-8 TPRI	Trigger Priority Setting This bitfield sets the priority of the associated trigger source. If two or more triggers have the same priority level setting, the lower order trigger event has the higher priority (e. g., if Trigger 0 and Trigger

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1 are both pending triggers and TCTRL0[TPRI] is configured the same as TCTRL1[TPRI], then the command associated with Trigger 0 is serviced first). 000b - Set to highest priority, Level 1 001b-110b - Set to corresponding priority level 111b - Set to lowest priority, Level 8
7-3 —	Reserved This read-only field is reserved and always has the value 0.
2 FIFO_SEL_B	SAR Result Destination for Channel B This register field is only used during dual single-ended mode CMDLa[CTYPE]=0b11. In this mode, the SAR result from Channel B will be written to the FIFO number specified by TCTRLa[FIFO_SEL_B]. See Sampling modes for more information. 0b - Result written to FIFO 0 1b - Result written to FIFO 1
1 FIFO_SEL_A	SAR Result Destination for Channel A Conversion results will be stored in the FIFO number specified by TCTRLa[FIFO_SEL_A] under the following conditions: Single ended mode: CMDLa[CTYPE]=0b01 or CMDLa[CTYPE]=0b00. Differential mode: CMDLa[CTYPE]=0b10. Dual single ended mode: CMDLa[CTYPE]=0b11. TCTRLa[FIFO_SEL_A] will provide the destination for all single ended conversions, differential conversions, and A side conversions in dual single-ended mode. 0b - Result written to FIFO 0 1b - Result written to FIFO 1
0 HTEN	Trigger Enable Enables hardware trigger source to initiate conversion on the rising-edge of the input trigger source. **NOTE** Enabling hardware trigger will not disable software triggers. 0b - Hardware trigger source disabled 1b - Hardware trigger source enabled

80.6.15 FIFO Control Register (FCTRL0 - FCTRL1)

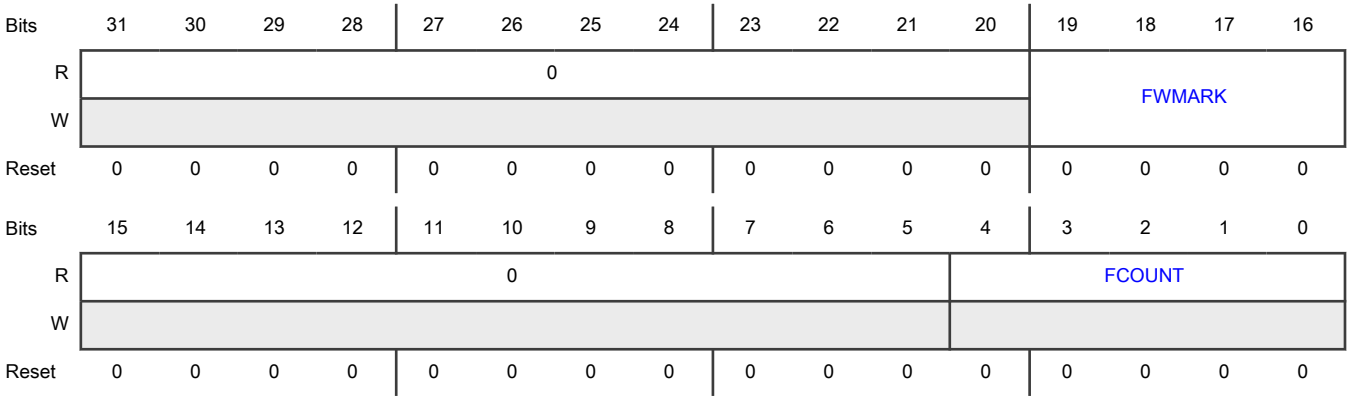
Offset

Register	Offset
FCTRL0	E0h
FCTRL1	E4h

Function

The FIFO Control (FCTRL*a*) registers contain control and status fields for each FIFO in the design. A programmable watermark can be set for each FIFO which can be used to trigger an interrupt. In addition, the number of entries stored in each FIFO can be monitored by reading FCTRLa[FCOUNT].

Diagram



Fields

Field	Function
31-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 FWMARK	Watermark Level Selection FWMARK is a programmable threshold setting. When the number of datawords stored in the ADC Result FIFO is greater than the value in FWMARK, the STAT[RDY] flag is asserted to indicate stored data has reached the programmable threshold. When IE[FWMIEx] is set, an interrupt request is generated. When DE[FWMDEx] is set, a DMA request is generated.
15-5 —	Reserved This read-only field is reserved and always has the value 0.
4-0 FCOUNT	Result FIFO Counter This read-only field indicates the number of datawords that are stored in the result FIFO. This value may be used in conjunction with PARAM[FIFOSIZE] to calculate how much room is left in the result FIFO. FCOUNT is incremented with each store of new data to the result FIFO and decrements with

Table continues on the next page...

Table continued from the previous page...

Field	Function
	each read of the result FIFO. The FIFO is reset by writing to the CTRL[RSTFIFOx] bit, resulting in FCTRLa[FCOUNT] initialized to 0x0.

80.6.16 Gain Calibration Control (GCC0 - GCC1)

Offset

Register	Offset
GCC0	F0h
GCC1	F4h

Function

The Gain Calibration Control register (GCC0 and GCC1) hold intermediate values that are utilized as part of the calibration steps. GCC0 is associated with the A-side converter and GCC1 is associated with the B-side converter. The GCCa[GAIN_CAL] field is calculated with hardware and auto updated during the calibration steps. Once the hardware calibration sequence has updated GCCa[GAIN_CAL] the GCCa[RDY] status flag is asserted automatically. Note that requesting a calibration routine automatically clears the GCCa[RDY] bit until the new GCCa[GAIN_CAL] value is calculated. For the calibration steps to be completed, further software processing is needed where the GCCa[GAIN_CAL] value is used to calculate the gain adjustment and the result is stored to GCRa[GCALR]. For more information see Calibration in the functional description.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							RDY	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GAIN_CAL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved This read-only field is reserved and always has the value 0.
24	Hardware Calculated GAIN_CAL Value Ready This status flag indicates when the data stored in GCCa[GAIN_CAL] is valid.

Table continues on the next page...

Table continued from the previous page...

Field	Function
RDY	0b - The GAIN_CAL value is invalid. Run the hardware calibration routine for this value to be set. 1b - The GAIN_CAL value is valid. GAIN_CAL should be used by software to derive GCRa[GCALR].
23-16 —	Reserved This read-only field is reserved and always has the value 0.
15-0 GAIN_CAL	Gain Calibration Value GAIN_CAL is read-only. As part of the hardware calibration steps, GAIN_CAL is automatically updated with a 16-bit signed number and is used in the gain adjustment calculations to derive the value written to the GCRa[GCALR] register. For more information see Calibration in the functional description.

80.6.17 Gain Calculation Result (GCR0 - GCR1)

Offset

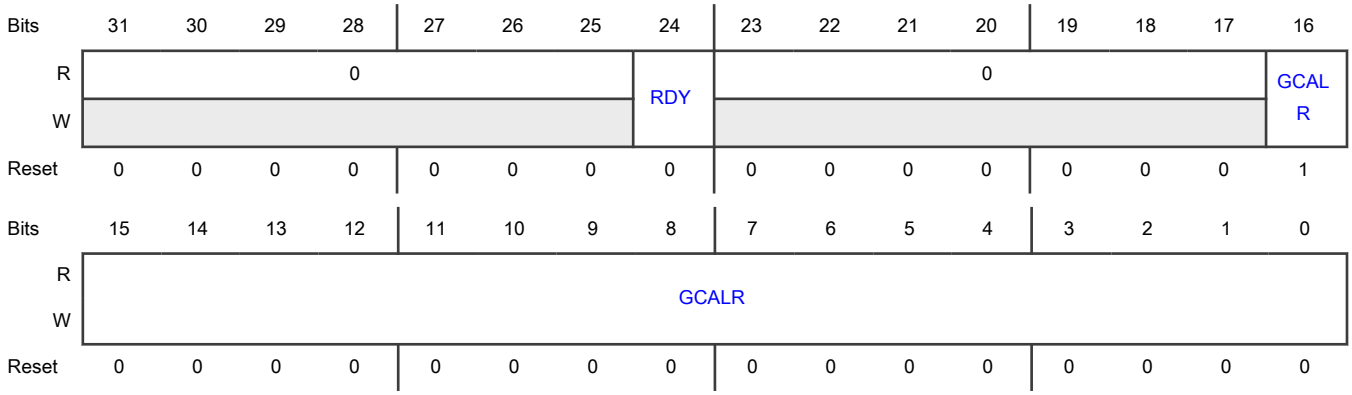
Register	Offset
GCR0	F8h
GCR1	FCh

Function

The Gain Calculation Result register (GCR0 and GCR1) is used during ADC conversions for automated gain error adjustment. GCR0 is associated with the A-side converter and GCR1 is associated with the B-side converter. Determining the values to store to GCC0[GCALR] and GCC1[GCALR] is the result of software calculations described in setup steps in Calibration. There is a RDY status flag in the GCR register which indicates whether the value GCRa[GCALR] is valid. Upon writing the GCRa[GCALR] value, the user should also assert GCRa[RDY] to indicate that the gain adjustment result is valid.

After beginning a calibration sequence by asserting CTRL[CAL_REQ], the calibration sequence is not complete until GCRa[GCALR] is calculated and GCRa[RDY] is asserted. The gain adjustment calculation results in a floating point value between 0 and 2. GCRa[GCALR][16] holds the integer value (0 or 1) and GCRa[GCALR][15:0] holds the 16-bit fractional component of the gain adjustment calculation. To convert the GCRa[GCALR] value back into the gain adjustment value in decimal format, this would be calculated as: 1*GCRa[16]+0.5*GCRa[15]+0.25*GCRa[14]+0.125*GCRa[13]+...

Diagram



Fields

Field	Function
31-25 —	Reserved This read-only field is reserved and always has the value 0.
24 RDY	Gain Calculation Ready The RDY flag indicates to the ADC when the data stored in GCRa[GCALR] is valid and is used for gain adjustment. The GCRa[GCALR] must be written from memory or calculated each time the calibration routine is run. The RDY flag must be asserted by the user after writing valid data into the GCALR field. The RDY flag is cleared automatically when requesting a new calibration sequence. 0b - The GCALR value is invalid. 1b - The GCALR value is valid.
23-17 —	Reserved This read-only field is reserved and always has the value 0.
16-0 GCALR	Gain Calculation Result This value holds the 17-bit value generated from the gain adjustment calculation during the calibration steps.

80.6.18 Command Low Buffer Register (CMDL1 - CMDL15)

Offset

For a = 1 to 15:

Register	Offset
CMDLa	F8h + (a × 8h)

Function

There are 15 command buffers (CMD*a*), each constructed from two 32-bit registers (CMDH*a*:CMDL*a*) that can be configured for different channel select and varying conversion options. Any of the command buffers is selected and used as the controlling

command by association to a trigger event via configuration of the TCTRLa[TCMD] bitfield. When the ADC is actively executing commands, only one of the CMD buffers is actively controlling ADC conversions. The actively controlling CMD buffer should not be updated while the ADC is active. A write to a CMD buffer while that CMD buffer is controlling the ADC operation may cause unpredictable behavior.

NOTE

The 15 command buffers are numbered [CMDH1:CMDL1] through [CMDH15:CMDL15]. In NXP-supplied header files, these are likely to be defined as two 15-element arrays that are indexed from 0. I.e., the type declaration would be:

```
unsigned int CMDH[15];

unsigned int CMDL[15];
```

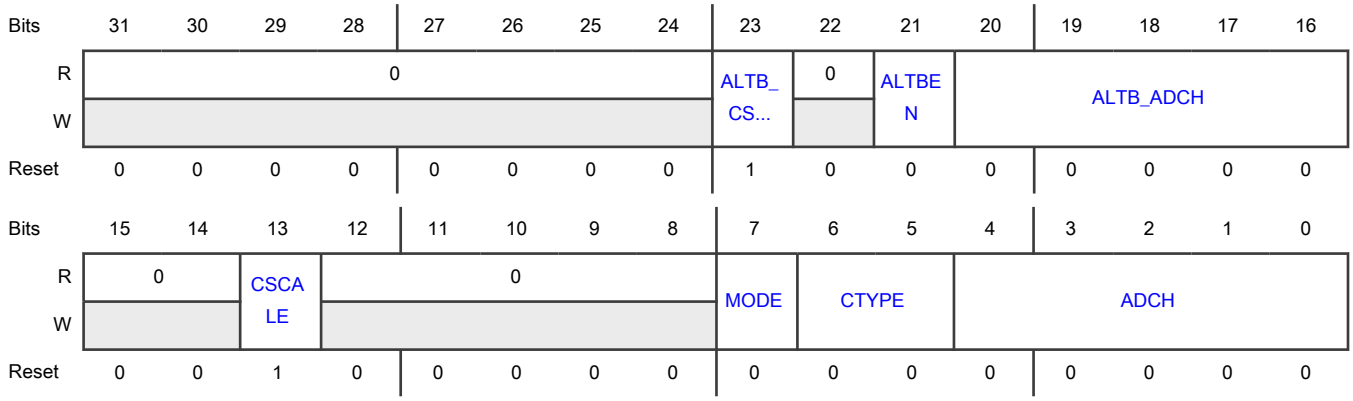
and software would access ADC0 CMDH1 as `ADC0->CMDH[0]` and ADC0 CMDL15 as `ADC0->CMDL[14]`.

The CMDLa[ADCH] and CMDLa[CTYPE] bitfields control selection of paired or individual input channels. Each ADC command independently makes a channel and conversion type selection. Each ADCH channel selection has an associated A side and an associated B side input. Each ADCH pair can optionally be converted in a differential mode but only limited pairs are intended to be converted as differential channels (i.e., adjacent pins that have been designed with matched impedance). For the pin pairings available for differential conversions for your device, see the Chip Configuration details.

NOTE

Some of the input channels are from on-chip sources such as temperature sensors and reference voltage sources and may only be connected to individual instances of the ADC module. Some of the input channel options in the bitfield-setting descriptions may not be available for your device. Refer to chip-specific information for the channels supported on this device.

Diagram



Fields

Field	Function
31-24 —	Reserved This read-only field is reserved and always has the value 0.
23 ALTB_CSCALE	Alt Channel B Scale When ALTBEN is set, reduces the selected ADC analog channel B input voltage level by a factor. The maximum possible voltage on the ADC channel input should be considered when selecting a CSCALE value to ensure that the reducing factor always results voltage level at or below the selected voltage

Table continues on the next page...

Table continued from the previous page...

Field	Function
	reference. This reducing capability allows conversion of analog inputs higher than selected voltage reference. When ALTBEN is disabled, B-side channel inputs are both scaled using the CSCALE field. 0b - Scale selected analog channel (Factor of 1/2) 1b - (Default) Full scale (Factor of 1)
22 —	Reserved This read-only field is reserved and always has the value 0.
21 ALTBEN	Alternate Channel B Select Enable Enables the ALTB_ADCH to select the input for Channel B independent of ADCH register settings when CTYPE is configured to 0b01 or 0b11. **NOTE** Setting CTYPE to differential (0b10) will still work but not recommended 0b - ALTBEN_ADCH disabled. Channel A and Channel B inputs are selected based on ADCH settings. 1b - ALTBEN_ADCH enabled. Channel A inputs selected by ADCH setting and Channel B inputs selected by ALTB_ADCH setting.
20-16 ALTB_ADCH	Alternate Channel B Input Channel Select When ALTBEN is set, ALTB_ADCH selects the Channel B input when CTYPE is configured to 0b01 or 0b11. **NOTE** Some of the input channels are from internal resources such as temperature sensors and bandgap voltage sources and may only be connected to individual instances of the ADC module. Some of the input channel options in the bitfield-setting descriptions might not be available for your device. For the ADC channel assignments for your device, see the chip-specific information. 0_0000b - Select CH0B 0_0001b - Select CH1B 0_0010b - Select CH2B 0_0011b - Select CH3B 0_0100b-1_1101b - Select corresponding channel CHnB 1_1110b - Select CH30B 1_1111b - Select CH31B
15-14 —	Reserved This read-only field is reserved and always has the value 0.
13 CSCALE	Channel Scale Reduces the selected ADC analog channel input voltage level by a factor. The maximum possible voltage on the ADC channel input should be considered when selecting a CSCALE value to ensure that

Table continues on the next page...

Table continued from the previous page...

Field	Function
	the reducing factor always results voltage level at or below the selected voltage reference. This reducing capability allows conversion of analog inputs higher than selected voltage reference. A-side and B-side channel inputs are both scaled using the CSCALE field. 0b - Scale selected analog channel (Factor of 1/2) 1b - (Default) Full scale (Factor of 1)
12-8 —	Reserved This read-only field is reserved and always has the value 0.
7 MODE	Select Resolution of Conversions The MODE field, selects the ADC resolution. 0b - Standard resolution. Single-ended 12-bit conversion; Differential 13-bit conversion with 2's complement output. 1b - High resolution. Single-ended 16-bit conversion; Differential 16-bit conversion with 2's complement output.
6-5 CTYPE	Conversion Type Chooses how a pair of A and B channels are converted. 00b - Single-Ended Mode. Only A side channel is converted. 01b - Single-Ended Mode. Only B side channel is converted. 10b - Differential Mode. A-B. 11b - Dual-Single-Ended Mode. Both A side and B side channels are converted independently.
4-0 ADCH	Input Channel Select Each ADCH channel selection has an associated A side and an associated B side input. The ADCH setting in conjunction with the CTYPE bitfield settings selects the input from one of the channel inputs or one of the input pairs. See the CTYPE bitfield descriptions. **NOTE** Not all pairs are intended to be converted as differential channels. For the pin pairings available for differential conversions for your device, see the chip-specific information. **NOTE** Some of the input channels are from internal resources such as temperature sensors and bandgap voltage sources and may only be connected to individual instances of the ADC module. Some of the input channel options in the bitfield-setting descriptions might not be available for your device. For the ADC channel assignments for your device, see the chip-specific information. 0_0000b - Select CH0A or CH0B or CH0A/CH0B pair. 0_0001b - Select CH1A or CH1B or CH1A/CH1B pair. 0_0010b - Select CH2A or CH2B or CH2A/CH2B pair.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0_0011b - Select CH3A or CH3B or CH3A/CH3B pair. 0_0100b-1_1101b - Select corresponding channel CHnA or CHnB or CHnA/CHnB pair. 1_1110b - Select CH30A or CH30B or CH30A/CH30B pair. 1_1111b - Select CH31A or CH31B or CH31A/CH31B pair.

80.6.19 Command High Buffer Register (CMDH1 - CMDH4)

Offset

Register	Offset
CMDH1	104h
CMDH2	10Ch
CMDH3	114h
CMDH4	11Ch

Function

There are 15 command buffers (CMD*a*), each constructed from two 32-bit registers (CMDH*a*:CMDL*a*) that can be configured for different channel select and varying conversion options. Any of the command buffers is selected and used as the controlling command by association to a trigger event via configuration of the TCTRLa[TCMD] bitfield. When the ADC is actively executing commands, only one of the CMD buffers is actively controlling ADC conversions. The actively controlling CMD buffer should not be updated while the ADC is active. A write to a CMD buffer while that CMD buffer that is controlling ADC operation may result in unpredictable behavior.

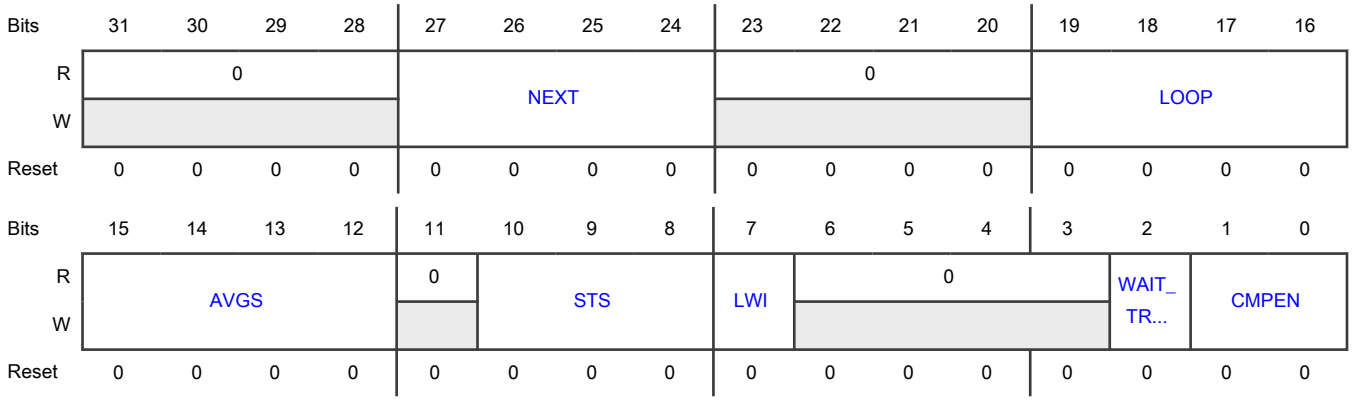
NOTE

The 15 command buffers are numbered [CMDH1:CMDL1] through [CMDH15:CMDL15]. In NXP-supplied header files, these are likely to be defined as two 15-element arrays that are indexed from 0. I.e., the type declaration would be:

```
unsigned int CMDH[15];
unsigned int CMDL[15];
```

and software would access ADC0 CMDH1 as `ADC0->CMDH[0]` and ADC0 CMDL15 as `ADC0->CMDL[14]`.

Diagram



Fields

Field	Function
31-28 —	Reserved This read-only field is reserved and always has the value 0.
27-24 NEXT	Next Command Select Selects the next command to be executed after this command completes. Multiple commands can be configured in a scan configuration by linking the next command in a daisy-chain sequence. The command buffer number is not indicative of any particular order and the order of execution is strictly controlled by the NEXT field (for example, a sequence of commands could be CMD2 to CMD1 to CMD3). Unending circular command execution is allowed by setting the NEXT field in the last command in a sequence to the first command in the sequence. It is also allowed for a command to set the next command to itself, resulting in a continuous conversion configuration. Setting the next command to 0x0 causes conversions to terminate at the completion of the command. Lower priority trigger events cannot be serviced until a higher priority triggered command (or sequence of commands) completes. 0000b - No next command defined. Terminate conversions at completion of current command. If lower priority trigger pending, begin command associated with lower priority trigger. 0001b - Select CMD1 command buffer register as next command. 0010b-1110b - Select corresponding CMD command buffer register as next command 1111b - Select CMD15 command buffer register as next command.
23-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 LOOP	Loop Count Select Selects how many times this command executes (and stores conversion result to RESFIFO) before finish and transition to the next command or Idle state. LWI field controls whether a single channel is converted on each iteration or auto channel increment results in channel scanning functionality. 0000b - Looping not enabled. Command executes 1 time. 0001b - Loop 1 time. Command executes 2 times.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0010b - Loop 2 times. Command executes 3 times. 0011b-1110b - Loop corresponding number of times. Command executes LOOP+1 times. 1111b - Loop 15 times. Command executes 16 times.
15-12 AVGS	Hardware Average Select Selects how many ADC conversions are averaged to create the ADC result (2^{AVGS}). An internal storage buffer is used to capture temporary results while the averaging iterations are executed. Hardware averaging is a nested loop control and does not extend across LOOP boundaries. See Functional description for more detailed description on usage of AVGS, LOOP, and NEXT bitfields in command execution sequencing. 0000b - Single conversion. 0001b - 2 conversions averaged. 0010b - 4 conversions averaged. 0011b - 8 conversions averaged. 0100b - 16 conversions averaged. 0101b - 32 conversions averaged. 0110b - 64 conversions averaged. 0111b - 128 conversions averaged. 1000b - 256 conversions averaged. 1001b - 512 conversions averaged. 1010b - 1024 conversions averaged.
11 —	Reserved This read-only field is reserved and always has the value 0.
10-8 STS	Sample Time Select When programmed to 0b000 the minimum sample time of 3.5 ADCK cycles is selected. When STS is programmed to a non-zero value the sample time is ($3.5 + 2^{STS}$) ADCK cycles. The shortest sample time maximizes conversion speed for lower impedance inputs. Extending sample time allows higher impedance inputs to be accurately sampled. Longer sample times can also be used to lower overall power consumption when command looping and sequencing is configured and high conversion rates are not required. 000b - Minimum sample time of 3.5 ADCK cycles. 001b - $3.5 + 2^1$ ADCK cycles; 5.5 ADCK cycles total sample time. 010b - $3.5 + 2^2$ ADCK cycles; 7.5 ADCK cycles total sample time. 011b - $3.5 + 2^3$ ADCK cycles; 11.5 ADCK cycles total sample time. 100b - $3.5 + 2^4$ ADCK cycles; 19.5 ADCK cycles total sample time. 101b - $3.5 + 2^5$ ADCK cycles; 35.5 ADCK cycles total sample time.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - 3.5 + 2^6 ADCK cycles; 67.5 ADCK cycles total sample time. 111b - 3.5 + 2^7 ADCK cycles; 131.5 ADCK cycles total sample time.
7 LWI	Loop with Increment When LWI is clear, the LOOP field selects the number of times the selected channel is converted consecutively. When LWI is set, auto channel incrementing is enabled and the LOOP field defines how many consecutive channels are converted as part of the command execution. Example 1: LOOP = 0x8, LWI = 0b0, CMDLa[CTYPE] = 0x0, CMDLa[ADCH] = 0xD. Convert on channel 13A 9 times. Example 2: LOOP = 0x8, LWI= 0b1, CMDLa[CTYPE] = 0x0, CMDLa[ADCH] = 0xD. Run channels 13A, 14A, 15A, ... 21A each one time. Maximum channel scanning using a single command buffer then is defined by the maximum value of the LOOP field (16). 0b - Auto channel increment disabled 1b - Auto channel increment enabled
6-3 —	Reserved This read-only field is reserved and always has the value 0.
2 WAIT_TRIG	Wait for Trigger Assertion before Execution. Controls if commands are automatically executed or if a trigger must be received before execution. If WAIT_TRIG is asserted, wait states will be added before the command until the active trigger is asserted again. When WAIT_TRIG is disabled, each command will be automatically executed when called. 0b - This command will be automatically executed. 1b - The active trigger must be asserted again before executing this command.
1-0 CMPEN	Compare Function Enable After an ADC channel input is sampled and converted and any averaging iterations are performed, the CMDH*a*[CMPEN] field guides operation of the automatic compare function to optionally only store when the compare operation is true. When compare is enabled, the conversion result is compared to the compare value registers (CV*a*[CVH] and CV*a*[CVL]). There are multiple options on command sequencing related to the compare function. See Compare function for a detailed explanation of the compare options. **NOTE** Not all Command Buffers have implemented the CMPEN field. The CMPEN field is only available in Command Buffers that have a corresponding Compare Value register. 00b - Compare disabled. 01b - Reserved 10b - Compare enabled. Store on true. 11b - Compare enabled. Repeat channel acquisition (sample/convert/compare) until true.

80.6.20 Command High Buffer Register (CMDH5 - CMDH15)

Offset

For a = 5 to 15:

Register	Offset
CMDHa	FCh + (a × 8h)

Function

There are 15 command buffers (CMD*a*), each constructed from two 32-bit registers (CMDH*a*:CMDL*a*) that can be configured for different channel select and varying conversion options. Any of the command buffers is selected and used as the controlling command by association to a trigger event via configuration of the TCTRLa[TCMD] bitfield. When the ADC is actively executing commands, only one of the CMD buffers is actively controlling ADC conversions. The actively controlling CMD buffer should not be updated while the ADC is active. A write to a CMD buffer while that CMD buffer that is controlling ADC operation may result in unpredictable behavior.

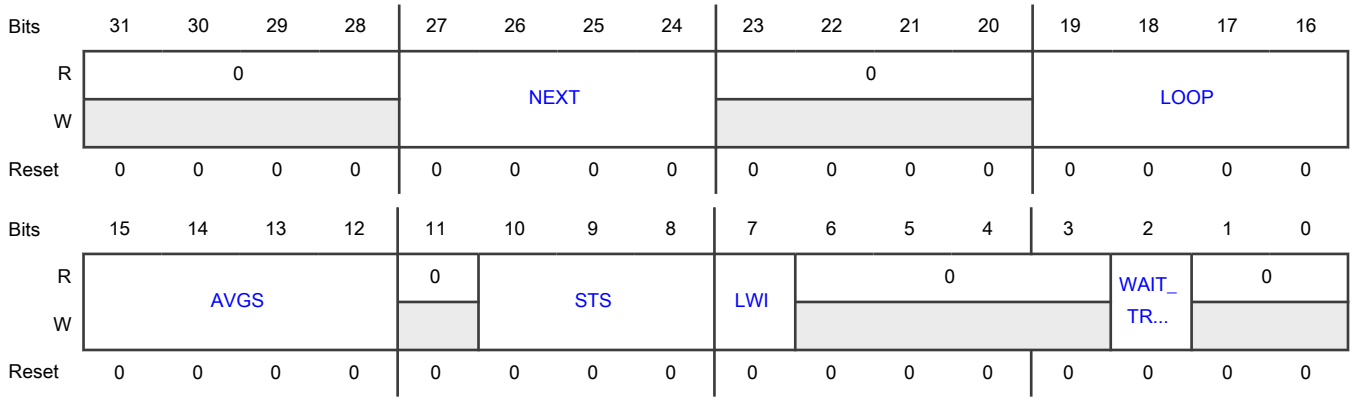
NOTE

The 15 command buffers are numbered [CMDH1:CMDL1] through [CMDH15:CMDL15]. In NXP-supplied header files, these are likely to be defined as two 15-element arrays that are indexed from 0. I.e., the type declaration would be:

```
unsigned int CMDH[15];
unsigned int CMDL[15];
```

and software would access ADC0 CMDH1 as `ADC0->CMDH[0]` and ADC0 CMDL15 as `ADC0->CMDL[14]`.

Diagram



Fields

Field	Function
31-28 —	Reserved This read-only field is reserved and always has the value 0.
27-24	Next Command Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
NEXT	Selects the next command to be executed after this command completes. Multiple commands can be configured in a scan configuration by linking the next command in a daisy-chain sequence. The command buffer number is not indicative of any particular order and the order of execution is strictly controlled by the NEXT field (for example, a sequence of commands could be CMD2 to CMD1 to CMD3). Unending circular command execution is allowed by setting the NEXT field in the last command in a sequence to the first command in the sequence. It is also allowed for a command to set the next command to itself, resulting in a continuous conversion configuration. Setting the next command to 0x0 causes conversions to terminate at the completion of the command. Lower priority trigger events cannot be serviced until a higher priority triggered command (or sequence of commands) completes. 0000b - No next command defined. Terminate conversions at completion of current command. If lower priority trigger pending, begin command associated with lower priority trigger. 0001b - Select CMD1 command buffer register as next command. 0010b-1110b - Select corresponding CMD command buffer register as next command 1111b - Select CMD15 command buffer register as next command.
23-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 LOOP	Loop Count Select Selects how many times this command executes (and stores conversion result to RESFIFO) before finish and transition to the next command or Idle state. LWI field controls whether a single channel is converted on each iteration or auto channel increment results in channel scanning functionality. 0000b - Looping not enabled. Command executes 1 time. 0001b - Loop 1 time. Command executes 2 times. 0010b - Loop 2 times. Command executes 3 times. 0011b-1110b - Loop corresponding number of times. Command executes LOOP+1 times. 1111b - Loop 15 times. Command executes 16 times.
15-12 AVGS	Hardware Average Select Selects how many ADC conversions are averaged to create the ADC result (2^{AVGS}). An internal storage buffer is used to capture temporary results while the averaging iterations are executed. Hardware averaging is a nested loop control and does not extend across LOOP boundaries. See Functional description for more detailed description on usage of AVGS, LOOP, and NEXT bitfields in command execution sequencing. 0000b - Single conversion. 0001b - 2 conversions averaged. 0010b - 4 conversions averaged. 0011b - 8 conversions averaged. 0100b - 16 conversions averaged. 0101b - 32 conversions averaged.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0110b - 64 conversions averaged. 0111b - 128 conversions averaged. 1000b - 256 conversions averaged. 1001b - 512 conversions averaged. 1010b - 1024 conversions averaged.
11 —	Reserved This read-only field is reserved and always has the value 0.
10-8 STS	Sample Time Select When programmed to 0b000 the minimum sample time of 3.5 ADCK cycles is selected. When STS is programmed to a non-zero value the sample time is ($3.5 + 2^{STS}$) ADCK cycles. The shortest sample time maximizes conversion speed for lower impedance inputs. Extending sample time allows higher impedance inputs to be accurately sampled. Longer sample times can also be used to lower overall power consumption when command looping and sequencing is configured and high conversion rates are not required. 000b - Minimum sample time of 3.5 ADCK cycles. 001b - $3.5 + 2^1$ ADCK cycles; 5.5 ADCK cycles total sample time. 010b - $3.5 + 2^2$ ADCK cycles; 7.5 ADCK cycles total sample time. 011b - $3.5 + 2^3$ ADCK cycles; 11.5 ADCK cycles total sample time. 100b - $3.5 + 2^4$ ADCK cycles; 19.5 ADCK cycles total sample time. 101b - $3.5 + 2^5$ ADCK cycles; 35.5 ADCK cycles total sample time. 110b - $3.5 + 2^6$ ADCK cycles; 67.5 ADCK cycles total sample time. 111b - $3.5 + 2^7$ ADCK cycles; 131.5 ADCK cycles total sample time.
7 LWI	Loop with Increment When LWI is clear, the LOOP field selects the number of times the selected channel is converted consecutively. When LWI is set, auto channel incrementing is enabled and the LOOP field defines how many consecutive channels are converted as part of the command execution. Example 1: LOOP = 0x8, LWI = 0b0, CMDLa[CTYPE] = 0x0, CMDLa[ADCH] = 0xD. Convert on channel 13A 9 times. Example 2: LOOP = 0x8, LWI= 0b1, CMDLa[CTYPE] = 0x0, CMDLa[ADCH] = 0xD. Run channels 13A, 14A, 15A, ... 21A each one time. Maximum channel scanning using a single command buffer then is defined by the maximum value of the LOOP field (16). 0b - Auto channel increment disabled 1b - Auto channel increment enabled
6-3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This read-only field is reserved and always has the value 0.
2 WAIT_TRIG	Wait for Trigger Assertion before Execution. Controls if commands are automatically executed or if a trigger must be received before execution. If WAIT_TRIG is asserted, wait states will be added before the command until the active trigger is asserted again. When WAIT_TRIG is disabled, each command will be automatically executed when called. 0b - This command will be automatically executed. 1b - The active trigger must be asserted again before executing this command.
1-0 —	Reserved This read-only field is reserved and always has the value 0.

80.6.21 Compare Value Register (CV1 - CV4)

Offset

Register	Offset
CV1	200h
CV2	204h
CV3	208h
CV4	20Ch

Function

The compare value registers (CV*a*) contain values used to compare the conversion result when the compare function is enabled. This register is formatted in the same way as the D field in the RESFIFO registers in different modes of operation for both bit position definition and value format using unsigned or signed 2's complement. There is a direct association of each compare value register to a specific command buffer register (i.e., CV1 is only used during execution of CMD1 command).

NOTE

Not all Command Buffers have an associated Compare Value register. The compare function is only available on Command Buffers that have a corresponding Compare Value register.

When the ADC is actively executing commands, the CV*a* register associated with the active command (CMD*a*) should not be updated. Writes to associated CV*a* register during this time may result in unpredictable behavior.

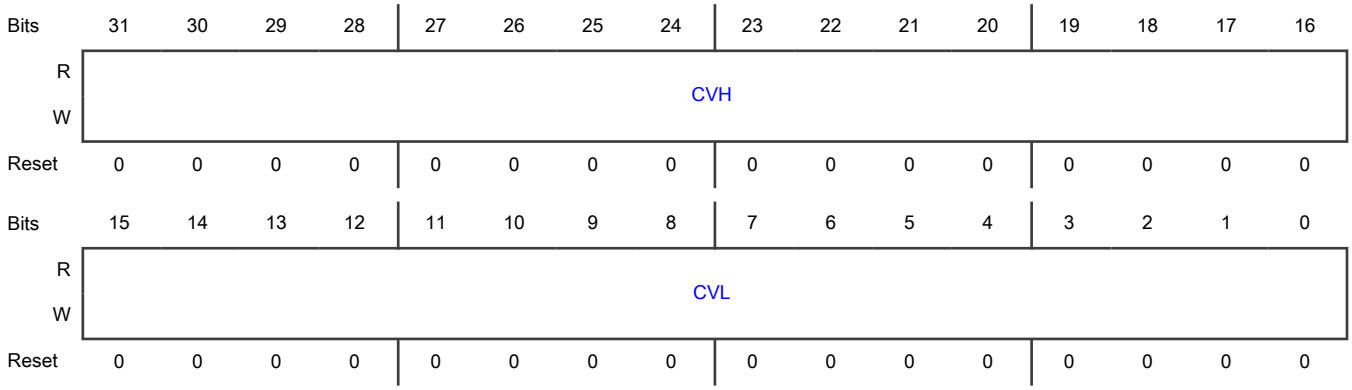
NOTE

The 4 compare value registers are numbered [CV1] through [CV4]. In NXP-supplied header files, this is likely to be defined as a 4-element array that is indexed from 0. I.e., the type declaration would be:

```
unsigned int CV[4];
```

and software would access ADC0 CV1 as `ADC0->CV[0]` and ADC0 CV4 as `ADC0->CV[3]`.

Diagram



Fields

Field	Function
31-16 CVH	Compare Value High The compare function can be configured to check whether the result is less than, greater than, or if the result falls within or outside a range determined by two compare values. After the input is sampled and converted and any averaging iterations are performed, the compare values in CVL and CVH are optionally used in a compare operation on the result. See Compare function for a description on how CVH is used.
15-0 CVL	Compare Value Low The compare function can be configured to check whether the result is less than, greater than, or if the result falls within or outside a range determined by two compare values. After the input is sampled and converted and any averaging iterations are performed, the compare values in CVL and CVH are optionally used in a compare operation on the result. See Compare function for a description on how CVL is used.

80.6.22 Data Result FIFO Register (RESFIFO0 - RESFIFO1)

Offset

Register	Offset
RESFIFO0	300h
RESFIFO1	304h

Function

The data result FIFO register (RESFIFO) is a 16 entry FIFO that stores the data result of ADC conversions. In addition, several tag fields of source command and trigger information are stored along with the data. FCTRLa[FCOUNT] indicates how many valid datawords are stored in the RESFIFO. Reading RESFIFO provides the oldest unread dataword entry in the FIFO and decrements FCTRLa[FCOUNT]. The FIFO can be emptied by successive reads of RESFIFO. The FIFO is reset by writing 0b1 to the CTRL[RSTFIFOx] bit.

The following table describes the format of data in the result FIFO in the different modes of operation. The sign bit is the (MSB) in signed 2's complement modes. For example, when configured for 12-bit single-ended mode, D[15] and D[2:0] are cleared. When configured for 13-bit differential mode, D[15] is the sign bit and D[2:0] are cleared.

Table 1110. Data result register format description

Conversion mode	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Format
16-bit differential	S	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Signed 2's complement, JLEFT=X
16-bit single-ended	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Unsigned, 16-bit magnitude, JLEFT=X
13-bit differential	S	D	D	D	D	D	D	D	D	D	D	D	D	0	0	0	Signed 2's complement, left-justified, zero extended, JLEFT=X
12-bit single-ended	0	D	D	D	D	D	D	D	D	D	D	D	D	0	0	0	Unsigned, JLEFT=0, zero in D[15] and D[2:0]
12-bit single-ended	D	D	D	D	D	D	D	D	D	D	D	D	0	0	0	0	Unsigned, JLEFT=1, zero in D[3:0]

NOTE

S: Sign bit;

D: Data, which is 2's complement data if indicated

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VALID	0			CMDSRC				LOOPCNT				0	TSRC		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	D															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VALID	FIFO Entry is Valid Indicate the FIFO entry is valid. When the FIFO is holding data the VALID bit is set. When the FIFO is empty the VALID bit is clear. 0b - FIFO is empty. Discard any read from RESFIFO. 1b - FIFO record read from RESFIFO is valid.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-28 —	Reserved This read-only field is reserved and always has the value 0.
27-24 CMDSRC	Command Buffer Source Indicate the command buffer being executed that generated this result. 0000b - Not a valid value CMDSRC value for a dataword in RESFIFO. 0x0 is only found in initial FIFO state prior to an ADC conversion result dataword being stored to a RESFIFO buffer. 0001b - CMD1 buffer used as control settings for this conversion. 0010b-1110b - Corresponding command buffer used as control settings for this conversion. 1111b - CMD15 buffer used as control settings for this conversion.
23-20 LOOPCNT	Loop Count Value Indicate the loop count value during command execution that generated this result. When CMDHa[LOOP] is non-zero, the command execution stores off a result multiple times during command execution (at the loop boundary). 0000b - Result is from initial conversion in command. 0001b - Result is from second conversion in command. 0010b-1110b - Result is from LOOPCNT+1 conversion in command. 1111b - Result is from 16^{th} conversion in command.
19 —	Reserved This read-only field is reserved and always has the value 0.
18-16 TSRC	Trigger Source Indicate the trigger source that initiated a conversion and generated this result. When multiple commands are chained together using the CMDHa[NEXT] field, the TSRC field for all datawords stored to the RESFIFO indicate the trigger source that started the sequence of commands. 000b - Trigger source 0 initiated this conversion. 001b - Trigger source 1 initiated this conversion. 010b-110b - Corresponding trigger source initiated this conversion. 111b - Trigger source 7 initiated this conversion.
15-0 D	Data Result The formatting for the data in D is summarized in Table 1110.

80.6.23 Calibration General A-Side Registers (CAL_GAR0 - CAL_GAR32)

Offset

For a = 0 to 32:

Register	Offset
CAL_GARa	400h + (a × 4h)

Function

The A-side general calibration value registers (CAL_GAR0 - CAL_GAR32) are used to correct for linearity errors of the A-side converter. All 33 CAL_GAR registers contain calibration information that is automatically updated during the self calibration sequence. See Calibration for more information on completing ADC calibration steps.

The calibration values in the CAL_GAR registers affect the end conversion result by conditionally being subtracted from the conversion before the end result is transferred into the FIFOs. Calibration must be run each time the ADC is powered down or a hard reset is issued. To reduce the latency required to run calibration, the CAL_GAR values can be stored in non-volatile memory after an initial calibration and recovered prior to the first ADC conversion. If these registers are set to values not generated by the calibration function, the linearity error specifications may not be met.

The CAL_GAR registers are only read write accessible when the ADC is disabled with CTRL[ADCEN] = 0b0. Note that access time when writing to these registers is larger than 3 ADC clock cycles. Wait states are inserted on the bus to meet synchronization timing to the associated CAL_GAR register.

Note that the values in each register is a 16-bit signed value but has non-uniform supported range of values. The valid range of each register is summarized in Calibration General A-Side and B-Side Register valid ranges.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CAL_GAR_VAL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved This read-only field is reserved and always has the value 0.
15-0 CAL_GAR_VAL	Calibration General A Side Register Element The CAL_GAR_VAL is a 16-bit signed value. The valid range for each register in this array is non-uniform and summarized in Calibration General A-Side and B-Side Register valid ranges.

80.6.24 Calibration General B-Side Registers (CAL_GBR0 - CAL_GBR32)

Offset

For a = 0 to 32:

Register	Offset
CAL_GBRa	500h + (a × 4h)

Function

The B-side general calibration value registers (CAL_GBR0 - CAL_GBR32) are used to correct for linearity errors of the B-side converter. All 33 CAL_GBR registers contain calibration information that is automatically updated during the self calibration sequence. See Calibration for more information on completing ADC calibration steps.

The calibration values in the CAL_GBR registers affect the end conversion result by conditionally being subtracted from the conversion before the end result is transferred into the FIFOs. Calibration must be run each time the ADC is powered down or a hard reset is issued. To reduce the latency required to run calibration, the CAL_GBR values can be stored in non-volatile memory after an initial calibration and recovered prior to the first ADC conversion. If these registers are set to values not generated by the calibration function, the linearity error specifications may not be met.

The CAL_GBR registers are only read write accessible when the ADC is disabled with CTRL[ADCEN] = 0b0. Note that access time when writing to these registers is larger than 3 ADC clock cycles. Wait states are inserted on the bus to meet synchronization timing to the associated CAL_GBR register.

Note that the values in each register is a 16-bit signed value but has non-uniform supported range of values. The valid range of each register is summarized in Calibration General A-Side and B-Side Register valid ranges.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CAL_GBR_VAL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved This read-only field is reserved and always has the value 0.
15-0 CAL_GBR_VAL	Calibration General B Side Register Element The CAL_GBR_VAL is a 16-bit signed value. The valid range for each register in this array is non-uniform and summarized in Calibration General A-Side and B-Side Register valid ranges.

80.6.25 Configuration 2 Register (CFG2)

Offset

Register	Offset
CFG2	FF8h

Function

The Configuration 2 Register controls ADC functions that are common to all commands.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							JLEFT	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved This read-only field is reserved and always has the value 0.
8 JLEFT	Justified Left Enable register In 12-bit single-ended mode, enables the data to be stored in RESFIFO register in left-justify format. The formatting for the data in RESFIFOa[D] is summarized in Table 1110. The JLEFT bit cannot be changed while the CTRL[ADCEN] bit is set. Writes to JLEFT while CTRL[ADCEN] is set are ignored. **NOTE** In JLEFT mode, CVa[CVL] and CVa[CVH] must be programmed in left-justify format to generate correct compare results. 0b - For 12-bit single-ended conversions, RESFIFO data format is in standard format with data presented in bits RESFIFOa[D][14:3]. 1b - For 12-bit single-ended conversions, RESFIFO data format is left-justified.
7-0 —	Reserved This read-only field is reserved and always has the value 0.

Chapter 81
Analog Comparator (CMP)

81.1 Chip-specific CMP information

Table 1111. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

- The fast clock is acmpx_clk_root
- The bus clock is bus_wakeup_clk_root. The DAC module clock input is the bus clock
- The slow clock is the OSC_32K clock
- Round robin clock is not used on this chip

NOTE

The 1.8V input channels are not used in this device. Vin1 is not used and tied to ground on this chip. Vin2 is from VDDA_1P8_IN.

Table 1112. Input signal source selections

		CMP1	CMP2	CMP3	CMP4
CMP Inputs	IN1	GPIO_AD_00	GPIO_AD_04	GPIO_AD_28	GPIO_AD_32
	IN2	GPIO_AD_01	GPIO_AD_05	GPIO_AD_29	GPIO_AD_33
	IN3	GPIO_AD_02	GPIO_AD_26	GPIO_AD_30	GPIO_AD_34
	IN4	GPIO_AD_03	GPIO_AD_27	GPIO_AD_31	GPIO_AD_35
	IN5	Reserved	Reserved	Reserved	Reserved
	IN6	Reserved	Reserved	Reserved	Reserved

Table continues on the next page...

Table 1112. Input signal source selections (continued)

		CMP1	CMP2	CMP3	CMP4
	IN7	CMP internal 8b DAC output	CMP internal 8b DAC output	CMP internal 8b DAC output	CMP internal 8b DAC output
CMP Outputs		GPIO_AD_17	GPIO_AD_18	GPIO_AD_19	GPIO_AD_20

81.2 Overview

ACMP includes the following submodules:

- High-speed comparator (CMP) (see ACMP block diagram)
- Digital-to-analog converter (DAC) (see CMP block diagram)
- Analog mux (ANMUX) (see ANMUX key features)

CMP provides a circuit to compare two analog input voltages. The comparator circuit operates across the full range of the supply voltage, known as rail-to-rail operation.

ANMUX provides a circuit to select an analog input signal from eight channels. DAC provides one signal. The mux circuit operates across the full range of the supply voltage.

DAC is a 256-tap resistor ladder network that provides a selectable voltage reference for applications requiring a voltage reference. It divides the supply reference, V_{in}, into 256 voltage levels. An 8-bit digital register input selects the output voltage level, which varies from V_{in} to $V_{in}/256$. You can select V_{in} from two voltage sources—V_{in1} or V_{in2}.

81.2.1 ACMP block diagram

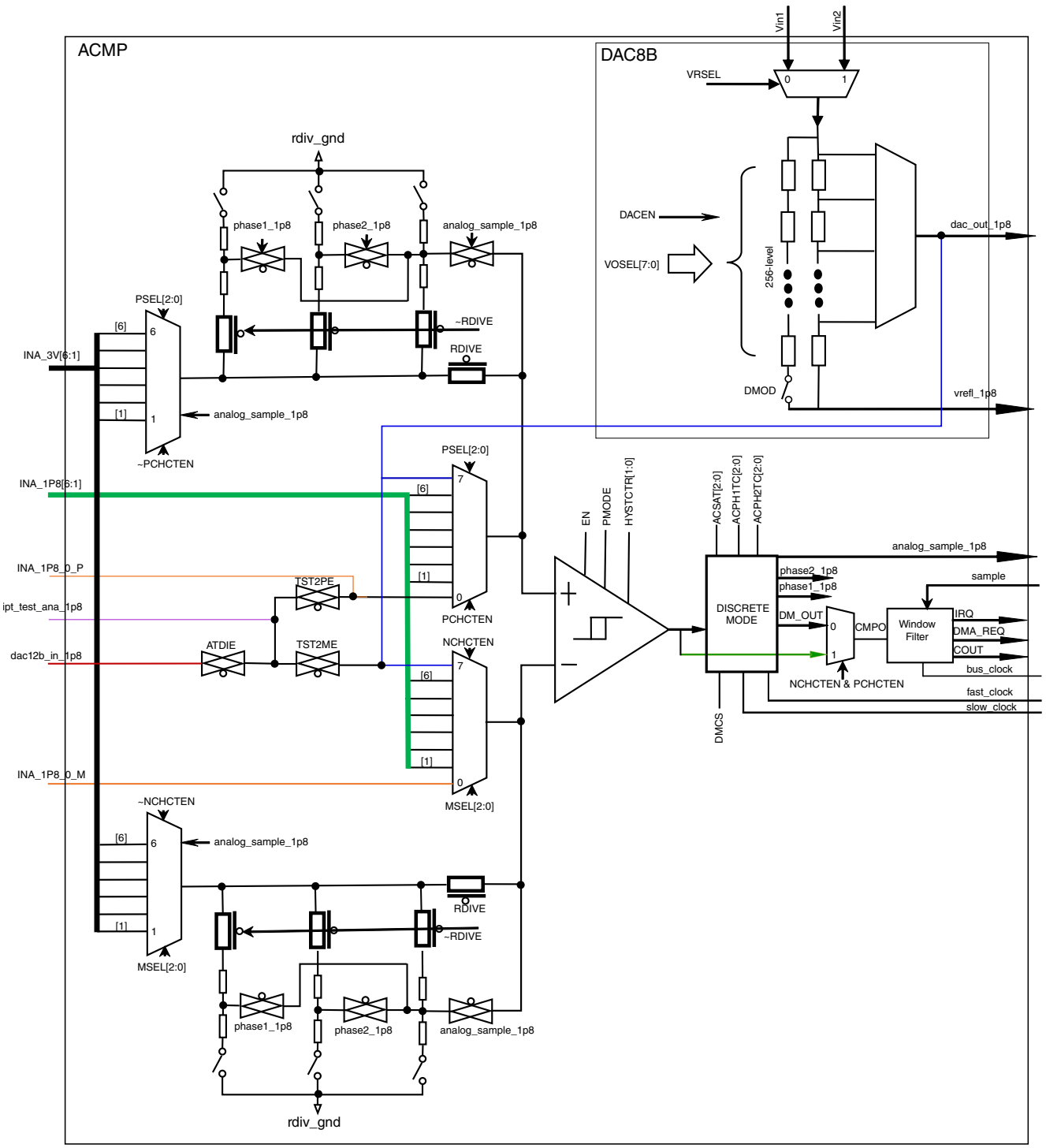


Figure 616. ACMP block diagram

81.2.2 CMP block diagram

In Figure 617 CMP:

- Bypasses the window control block when C0[WE] = 0.

- Samples the comparator output on every bus clock when the Window signal is 1 to generate COUTA, if C0[WE] = 1. Sampling does not occur when the Window signal is 0.
- Bypasses the filter block when it is not in use.

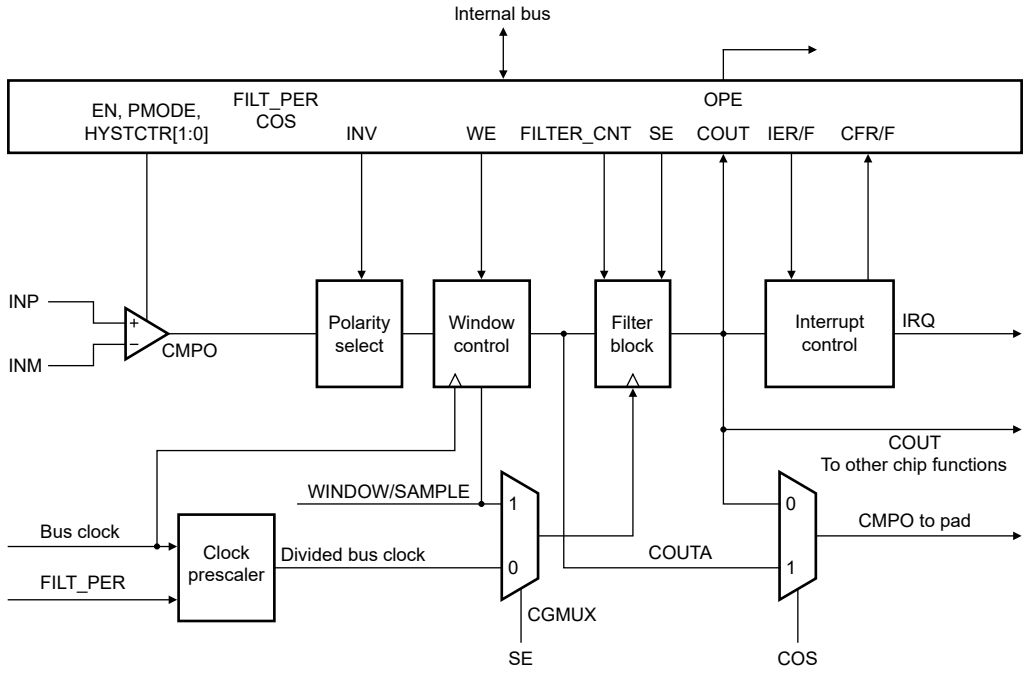


Figure 617. CMP block diagram

The filter block bypass operates as follows:

- The filter block acts as a simple sampler if you bypass the filter and C0[FILTER_CNT] becomes 01h.
- The filter block filters based on multiple samples when you bypass the filter and C0[FILTER_CNT] becomes greater than 01h.
  - If C0[SE] = 1, use the external Sample input as the sampling clock.
  - If C0[SE] = 0, use the divided bus clock as the sampling clock.
- If enabled, the filter block incurs an additional latency penalty of up to one bus clock on COUT. This is because COUT, which crosses clock domain boundaries, must resynchronize with the bus clock.
- C0[WE] and C0[SE] are mutually exclusive.
- If enabled, the filter clock and sample period must be at least four times slower than the system clock to the comparator.
- If C0[SE] = 0, ACMP bypasses the filter block when you write 00h to C0[FILTER_CNT] or when you write 00h to C0[FPR].

81.2.3 Features

The following subsections list the features of high-speed CMP, 8-bit DAC, and ANMUX.

81.2.3.1 CMP features

- Operates over the entire supply range
- Supports rail-to-rail inputs
- Supports programmable hysteresis control
- Provides a selectable interrupt on rising-edge, falling-edge, or both comparator outputs

- Provides a selectable inversion on comparator output
- Includes the capability to produce a wide range of outputs, such as:
  - Sampled
  - Windowed, which is ideal for certain PWM zero-crossing-detection applications
  - Digitally filtered:
    - Bypass filter
    - Clocks via an external Sample signal or a scaled bus clock
- Uses external hysteresis at the same time that the output filter is used for internal functions
- Supports two software selectable performance levels:
  - Shorter propagation delay at the expense of higher power
  - Low power, with a longer propagation delay
- Supports DMA transfer (CMP can select a comparison event to trigger a DMA transfer)
- Functional in all MCU power modes

CMP's window and filter functions are unavailable in Stop modes.

Other peripherals trigger the comparator.

81.2.3.2 8-bit DAC key features

- 8-bit resolution
- Selectable supply reference source
- Low-Power mode or High-Speed mode that you can configure
- Power-Down mode to conserve power when not in use
- Routing of the output of the 8-bit DAC to the internal comparator input, where you can select it as one of the CMP inputs as a reference for comparison

81.2.3.3 ANMUX key features

- Two 8-to-1 channel MUXes that can be selected as the source for comparator input
- Operation over the entire supply range

81.3 CMP functional description

CMP can compare two analog input voltages applied to INP and INM. CMPO is high when the non inverting input is greater than the inverting input, and it is low when the non inverting input is less than the inverting input. Write 1 to C0[INVT] to selectively invert this signal.

C0[IER] and C0[IEF] select the condition that causes CMP to assert an interrupt to the processor. C0[CFF] sets on a falling edge, and C0[CFR] sets on a rising edge of the comparator output. You can read the optionally filtered CMPO directly through C0[COUT].

81.3.1 CMP functional modes

CMP includes the following main sub-blocks:

- Comparator
- Window function
- Filter function

You can clock C0[FILTER_CNT] from an internal or external clock source. The filter is programmable with respect to the number of samples that must agree before a change in the output is registered. In the simplest case, only one sample must agree. In this case, the filter acts as a simple sampler.

Enable the external sample input using C0[SE]. When C0[SE] is 1, sample the output of the comparator only on the rising edges of the sample input.

Write 1 to C0[WE] to enable the window function. When C0[WE] is 1, sample the comparator output only when the Window signal is 1. You can use this feature to ignore the comparator output during time periods in which the input voltages are not valid. This is useful when you implement zero-crossing detection for certain PWM applications.

Combine the comparator filter and sampling features as shown in the following table. Individual modes are also discussed.

Table 1113. Comparator sample or filter controls

Mode		C0[EN]	C0[WE]	C0[SE]	C0[FILTER_CNT]	C0[FPR]	Operation
Name	#						
Disabled	1	0	X	X	X	X	See Disabled mode.
Continuous	2A	1	0	0	00h	X	See Continuous mode.
	2B	1	0	0	X	00h	
Sampled, Nonfiltered	3A	1	0	1	01h	X	See Sampled, Nonfiltered mode.
	3B	1	0	0	01h	> 00h	
Sampled, Filtered	4A	1	0	1	> 01h	X	See Sampled, Filtered mode.
	4B	1	0	0	> 01h	> 00h	
Windowed	5A	1	1	0	00h	X	Sample the comparator output on every rising bus clock edge when the Sample signal is 1 to generate COUTA. See Windowed mode.
	5B	1	1	0	X	00h	
Windowed/ Resampled	6	1	1	0	01h	01h–FFh	Sample the comparator output on every rising bus clock edge when the Sample signal is 1 to generate COUTA, which is then resampled on an interval that C0[FPR] determines to generate COUT. See Windowed/ Resampled mode.
Windowed/ Filtered	7	1	1	0	> 01h	01h–FFh	Sample the comparator output on every rising bus clock edge when the Sample signal is 1 to generate COUTA,

Table continues on the next page...

Table 1113. Comparator sample or filter controls (continued)

Mode		C0[EN]	C0[WE]	C0[SE]	C0[FILTER_CNT]	C0[FPR]	Operation
Name	#						
							which resamples and filters to generate COUT. See Windowed/Filtered mode.
Trigger	8	1	1	1	X	X	In Run mode, comparator is configured to the modes same with Windowed mode, Windowed/Resampled mode, and Windowed/Filtered mode as described in the aforementioned rows, depending on the settings of C0[FILTER_CNT] and C0[FPR]. In Stop mode, trigger the comparator, and possibly the 8-bit DAC works periodically. The active channels follow the round-robin cycling scheme. See Trigger mode.
All other combinations of C0[EN], C0[WE], C0[SE], C0[FILTER_CNT], and C0[FPR] are illegal.							

For cases where a comparator drives a fault input, for example, for a motor-control module such as FTM or FlexPWM, it must configure to operate in Continuous mode so that an external fault can immediately pass through the comparator to the target fault circuitry.

NOTE

Filtering and sampling settings must change only after writing 0 to C0[SE] and C0[FPR], and 00h to C0[FILTER_CNT]. This resets the filter to a known state.

81.3.1.1 Disabled mode

In this mode:

- The analog comparator is nonfunctional and consumes no power.
- CMPO is 0.

81.3.1.2 Continuous mode

In this mode, the analog comparator block is powered and active. CMPO may be optionally inverted but is not subject to external sampling or filtering. CMP completely bypasses both window control and filter blocks and update C0[COUT] continuously. The path from comparator input pins to output pins operates in a combinational unclocked mode. COUT and COUTA are identical.

See CMP block diagram section for control configurations that result in disabling the filter block.

NOTE

See the chip-specific ACMP information for the source of sample or window input.

Figure 618 describes the comparator operation in Continuous mode.

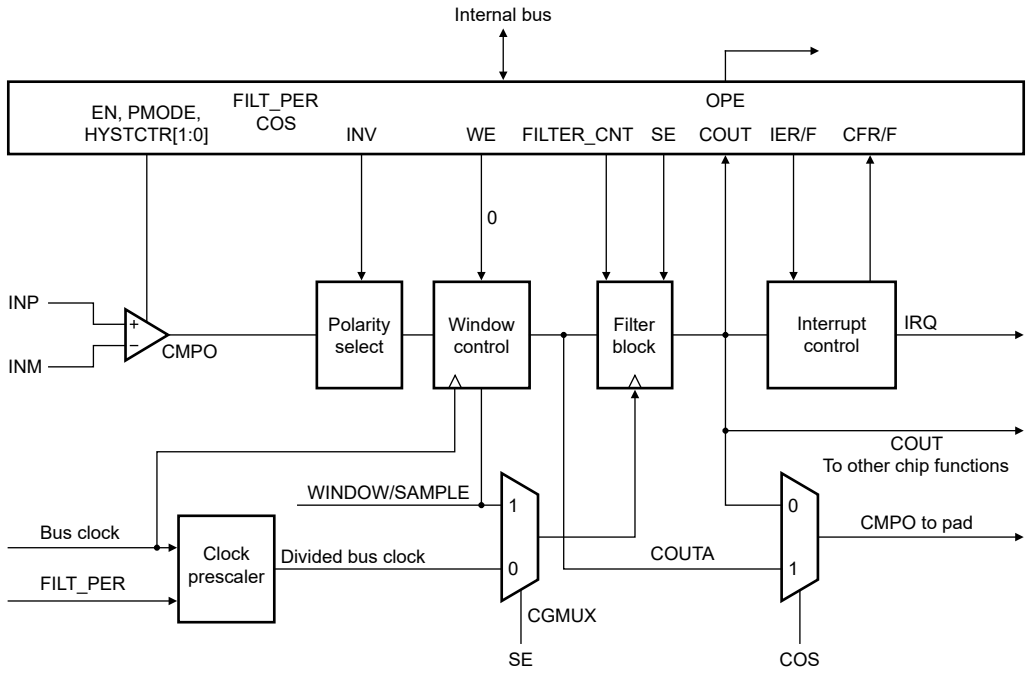


Figure 618. Comparator operation in Continuous mode

81.3.1.3 Sampled, Nonfiltered mode

In this mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. CMP completely bypasses the windowing control. Sample COUTA whenever you detect a rising edge on the filter block clock input.

The only operational difference between Sampled, Nonfiltered and Sampled, Nonfiltered modes (3A and 3B in Table 1113) is how the clock to the filter block is derived. In Sampled, Nonfiltered mode (3A in Table 1113) , the clock-to-filter block is externally derived (see Figure 619) and in Sampled, Nonfiltered mode (3B in Table 1113), the clock-to-filter block is internally derived (see Figure 620).

The comparator filter has no other function than sampling or holding the comparator output in this mode (3B in Table 1113).

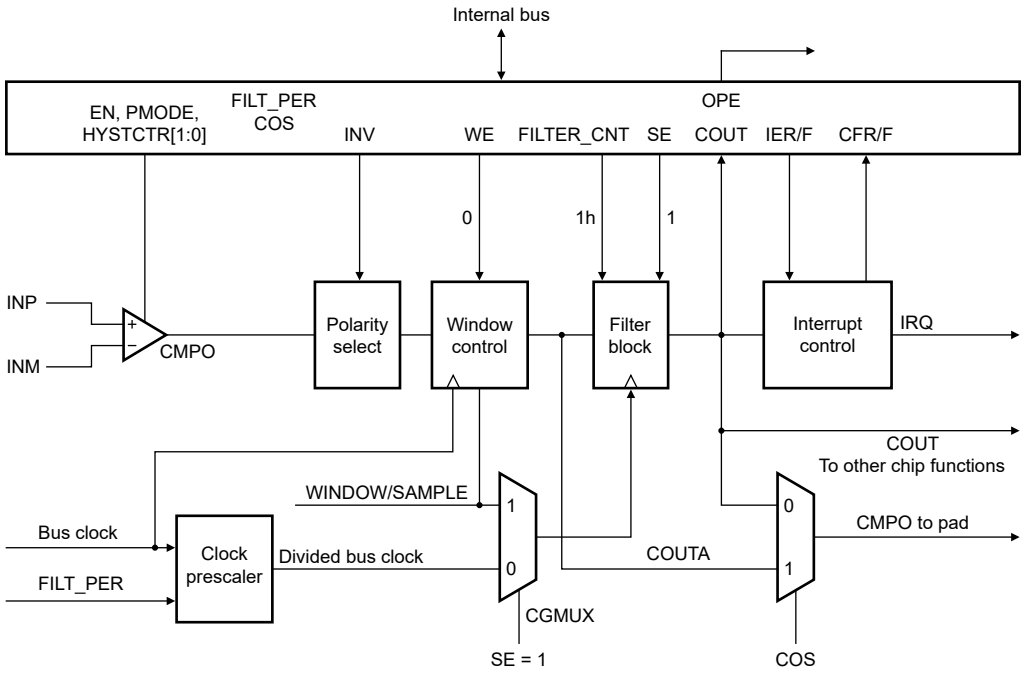


Figure 619. Sampled, Nonfiltered (# 3A): sampling point externally driven

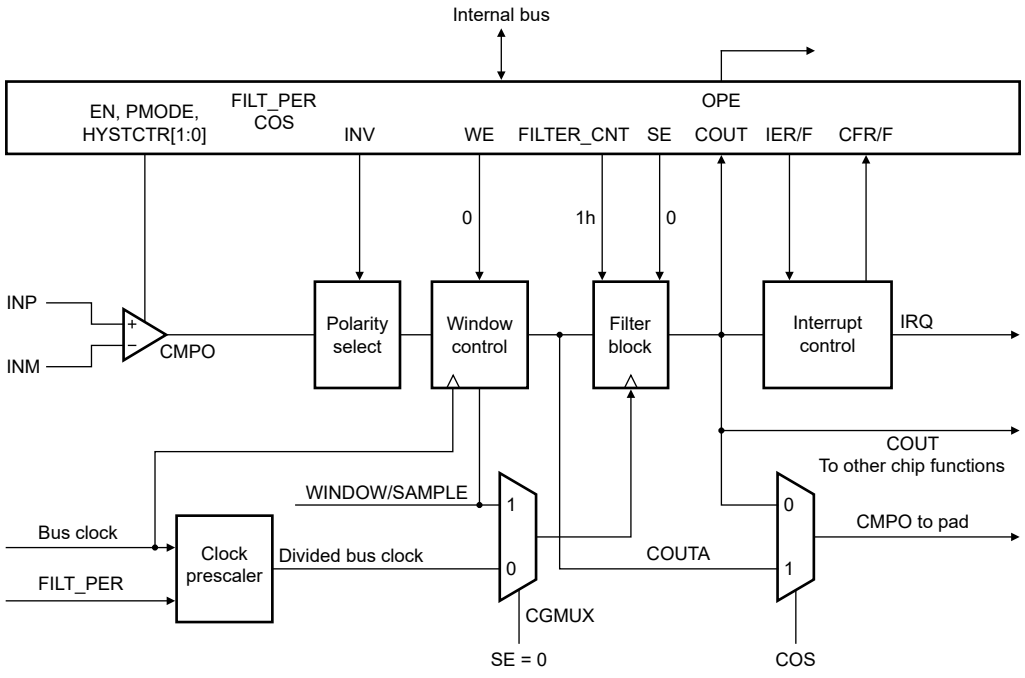


Figure 620. Sampled, Nonfiltered (# 3B): sampling interval internally derived

81.3.1.4 Sampled, Filtered mode

In this mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Completely bypass the windowing control and sample COUTA whenever you detect a rising edge on the filter block clock input.

The only operational difference between Sampled, Nonfiltered and Sampled, Filtered modes (3A and 4A in Table 1113) is C0[FILTER_CNT] > 1, which activates the filter operation (see Figure 621).

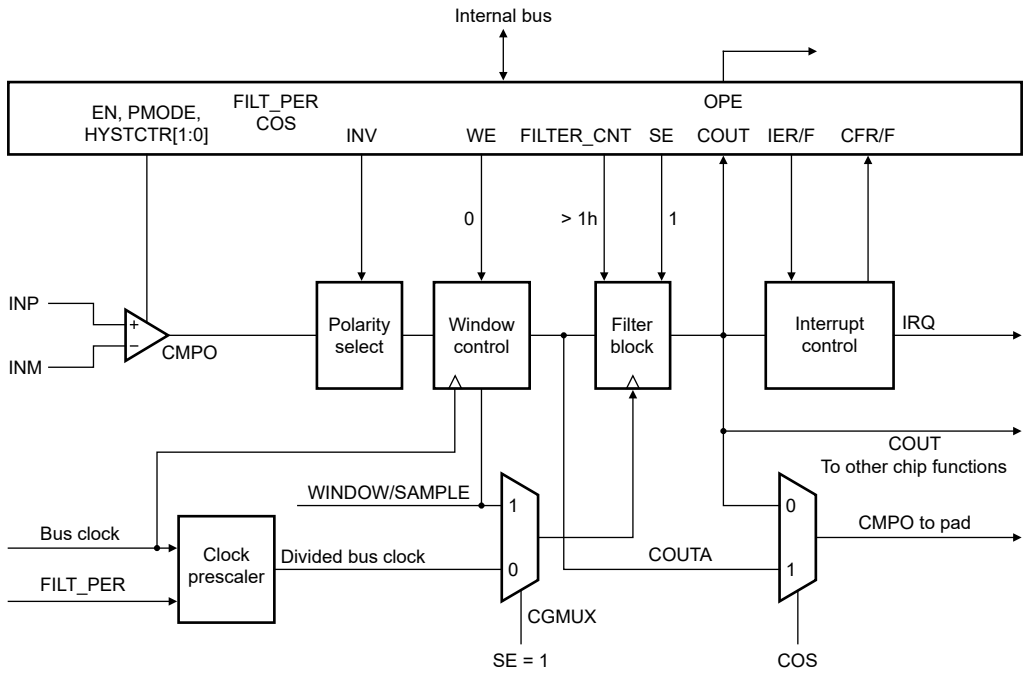


Figure 621. Sampled, Filtered (# 4A): sampling point externally driven

The only operational difference between Sampled, Nonfiltered, and Sampled, Filtered modes (3B and 4B in Table 1113) is C0[FILTER_CNT] > 1, which activates the filter operation (see Figure 622).

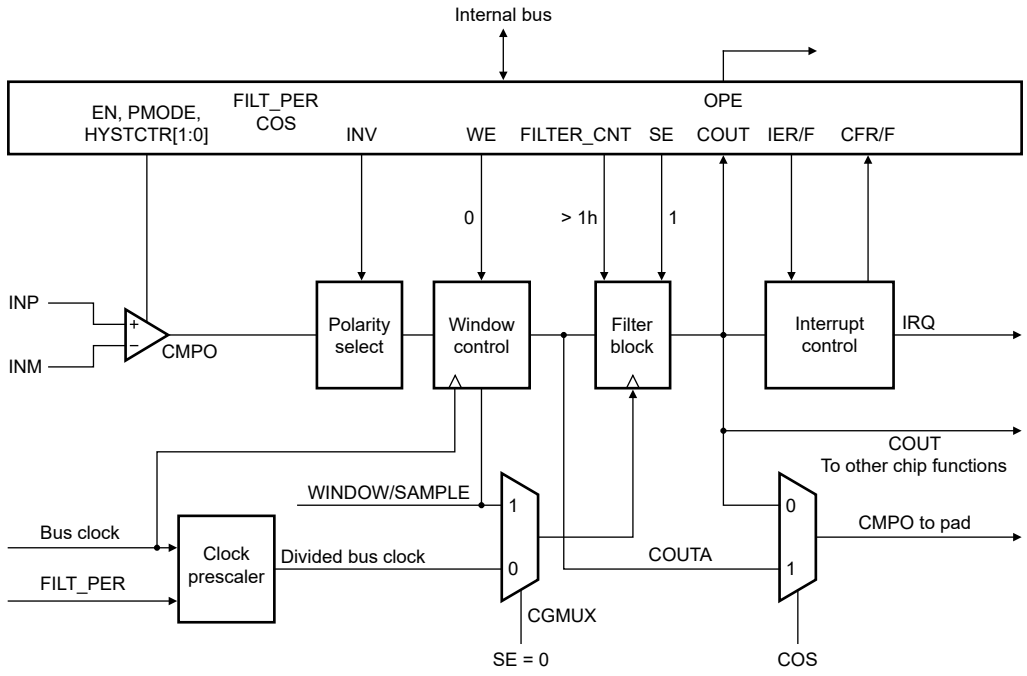


Figure 622. Sampled, Filtered (# 4B): sampling point internally derived

81.3.1.5 Windowed mode

Figure 623 shows the following:

- Comparator operation in Windowed mode
- Ignoring the latency of the analog comparator (see Latency issues)
- Polarity select
- Window control block

It also assumes that the polarity select sets to a noninverting state.

NOTE

The analog comparator output passes to COUTA only after the Window signal becomes high.

In actual operation, COUTA may lag the analog inputs by up to one bus clock cycle plus the combinational path delay through the comparator and polarity select logic.

See CMP block diagram section for control configurations that result in disabling the filter block.

When a windowed mode is active, the bus clock clocks COUTA whenever the Window signal is 1. The bus clock holds the last latched value when the Window signal is 0.

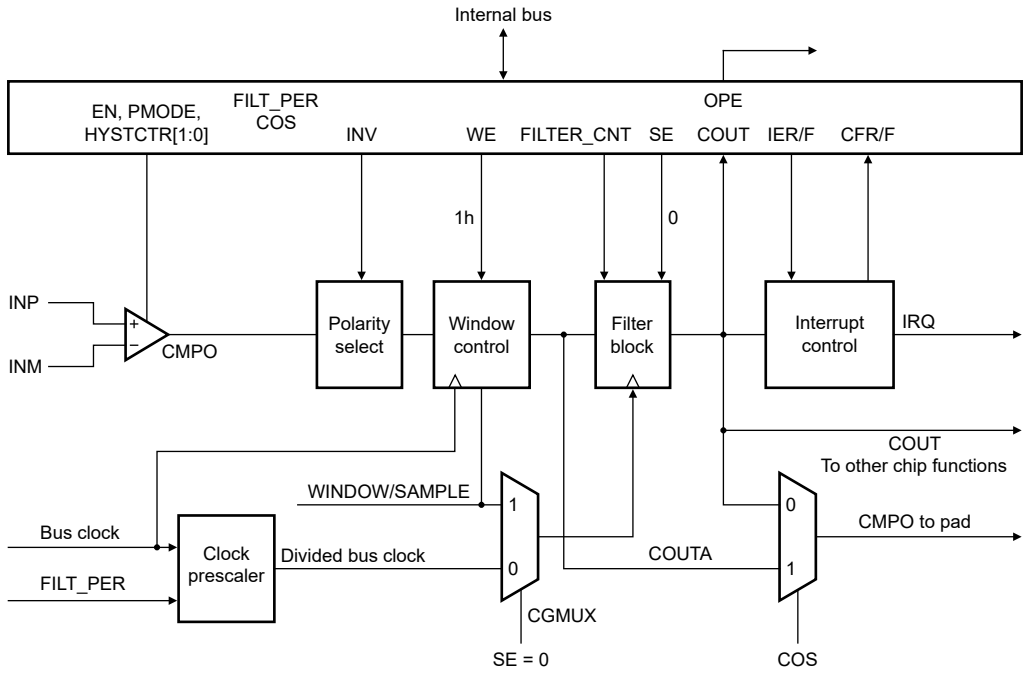


Figure 623. Windowed mode

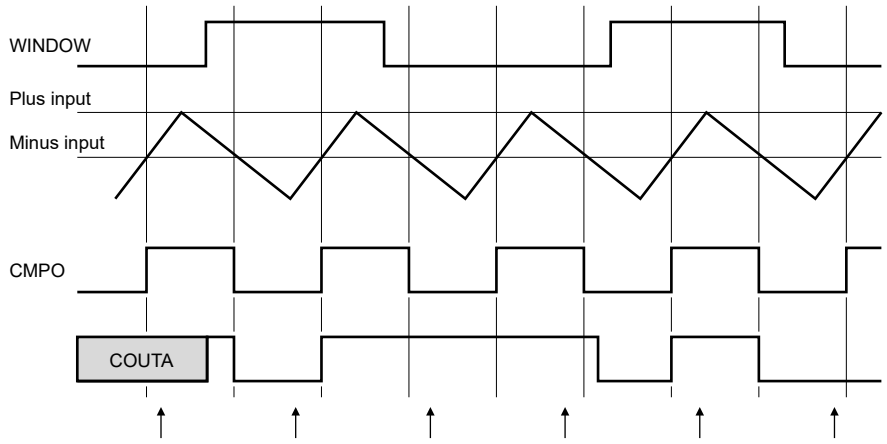


Figure 624. Windowed mode timing diagram

81.3.1.6 Windowed/Resampled mode

Figure 625 uses the same input stimulus as shown in Figure 624, and adds resampling of COUTA to generate COUT. The arrows in the figure indicate the time points at which the samples are taken. You can ignore prop delays and latency for clarity.

This example demonstrates operation of the comparator in Windowed/Resampled mode, and does not reflect any specific application. Depending upon the sampling rate and window placement, COUT may not see zero-crossing events, which the analog comparator detects. You must carefully consider the sampling period and window placement for a given application.

This mode of operation results in an unfiltered string of comparator samples where C0[FPR] and the bus clock rate determine the interval between the samples. The next section describes that the configuration for this mode is identical to that of Windowed/Filtered mode. The only difference is that the value of C0[FILTER_CNT] must be 1.

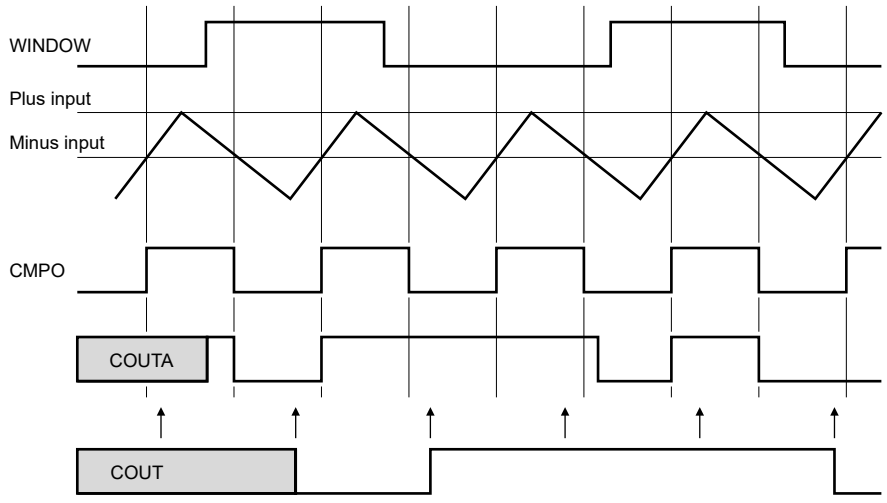


Figure 625. Windowed/Resampled mode operation

81.3.1.7 Windowed/Filtered mode

This is the most complex mode of operation for the comparator block, because this mode uses both windowing and filtering features. It also has the highest latency of any of the modes. This is approximately: up to 1 bus clock synchronization in the window function + ((C0[FILTER_CNT] × C0[FPR]) + 1) × bus clock for the filter function.

When any windowed mode is active, the bus clock clocks the COUTA whenever the Window signal is 1. The bus clock holds the last latched value when the Window signal is 0.

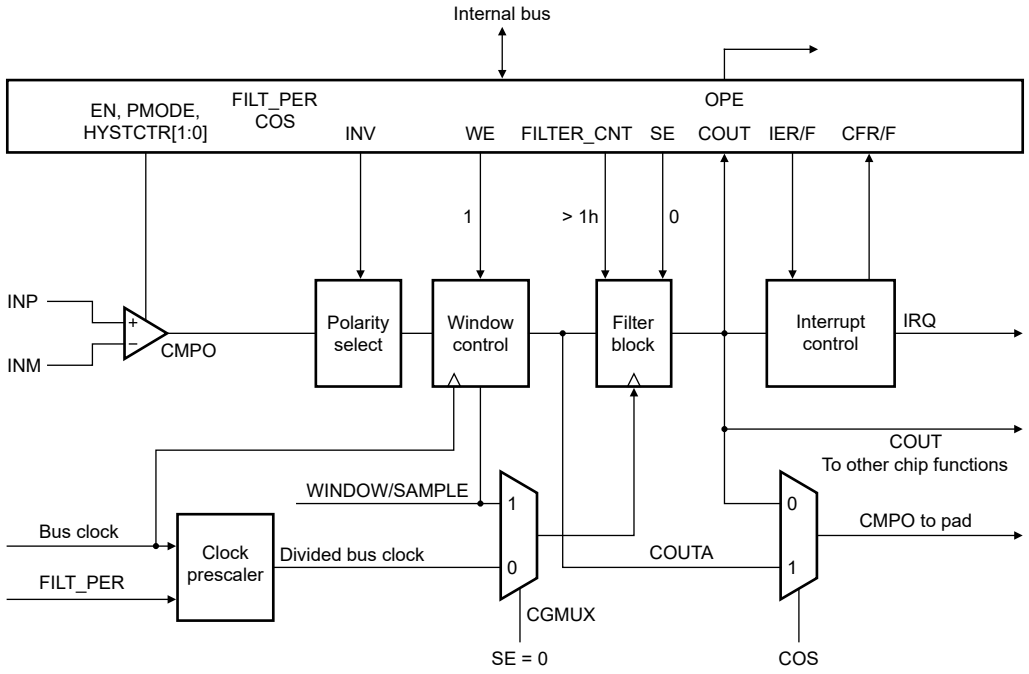


Figure 626. Windowed/Filtered mode

81.3.2 Low-pass filter

The low-pass filter operates on the unfiltered, unsynchronized, and optionally inverted comparator output, COUTA, and generates the filtered and synchronized output, COUT. You can configure both COUTA and COUT as module outputs, and use them for different purposes within the system.

Synchronization and edge detection always determine the status register bit values. They also apply to COUT for all sampling and windowed modes. You can perform filtering using an internal timebase defined by C0[FPR], or use an external sample input to determine sample time.

The need for digital filtering and the amount of filtering depends on your requirements. Filtering can become more useful in the absence of an external hysteresis circuit, without which you can generate high-frequency oscillations at COUTA when the selected INM and INP input voltages differ by less than the offset voltage of the differential comparator.

81.3.2.1 Enabling filter modes

To enable filter modes:

1. Write a value greater than 1h to C0[FILTER_CNT].
2. Write a nonzero value to C0[FPR] or write 1 to C0[SE].

If you use the divided bus clock to drive the filter, it samples COUTA at every bus clock cycle that C0[FPR] defines.

The filter output is at logic 0 when first initialized and later changes when all the consecutive samples that C0[FILTER_CNT] specifies agree that the output value has changed. In other words, C0[COUT] is 0 for some initial period, even when COUTA is at logic 1.

Writing 0 to C0[SE], C0[FPR] and C0[FILTER_CNT] disables the filter and eliminates the switching current associated with the filtering process.

NOTE

Always switch to this setting before you make any changes in filter parameters. This resets the filter to a known state. Switching C0[FILTER_CNT] on the fly without this intermediate step can result in unexpected behavior.

If C0[SE] = 1, the filter samples COUTA on each positive transition of the sample input. The output state of the filter changes when all the consecutive samples that C0[FILTER_CNT] specifies agree that the output value has changed.

81.3.2.2 Latency issues

Program the value of C0[FPR] or the sample period in a way that the sampling period is longer than the period of the expected noise. This way, a noise spike corrupts only one sample. You must choose the value of C0[FILTER_CNT] to reduce the probability of noisy samples causing an incorrect transition to recognize. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of the value in C0[FILTER_CNT].

You must trade off the values of C0[FPR] or the sample period and C0[FILTER_CNT] against the desire for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the power of the value in C0[FILTER_CNT].

The following table summarizes the maximum latency values for the various modes of operation in the absence of noise. Filtering latency restarts each time the noise masks an actual output transition.

Table 1114. Comparator sample or filter maximum latencies

Mode #	C0[EN]	C0[WE]	C0[SE]	C0[FILTER_CNT]	C0[FPR]	Operation	Maximum latency[1]
1	0	X	X	X	X	Disabled	N/A
2A	1	0	0	00h	X	Continuous mode	T_{PD}
2B	1	0	0	X	00h		
3A	1	0	1	01h	X	Sampled, Nonfiltered mode	$T_{PD} + T_{SAMPLE} + T_{per}$
3B	1	0	0	01h	> 00h		$T_{PD} + (C0[FPR] \times T_{per}) + T_{per}$
4A	1	0	1	> 01h	X	Sampled, Filtered mode	$T_{PD} + (C0[FILTER\_CNT] \times T_{SAMPLE}) + T_{per}$
4B	1	0	0	> 01h	> 00h		$T_{PD} + (C0[FILTER\_CNT] \times C0[FPR] \times T_{per}) + T_{per}$
5A	1	1	0	00h	X	Windowed mode	$T_{PD} + T_{per}$
5B	1	1	0	X	00h		$T_{PD} + T_{per}$
6	1	1	0	01h	01h – FFh	Windowed/ Resampled mode	$T_{PD} + (C0[FPR] \times T_{per}) + 2T_{per}$
7	1	1	0	> 01h	01h – FFh	Windowed/ Filtered mode	$T_{PD} + (C0[FILTER\_CNT] \times C0[FPR] \times T_{per}) + 2T_{per}$

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{SAMPLE} is the clock period of the external sample clock. T_{per} is the period of the bus clock.

NOTE

The delay in this table does not include the delay caused by the Discrete Mode.

81.3.3 CMP Discrete mode timing sequence

In order to support the 3 V domain input the analog comparator should be configured to work in Discrete Mode since the internal transistors are not 3 V tolerance. This section summarizes the possible configurations depending on the input range.

Note that the sample signal below means that the analog comparator samples the input analog input. It is different from the sample mode that occurs on the digital process to the final comparison result. Also, note that the cross domain round robin cycling (see Trigger mode) is not allowed.

1. When CMP_C3[PCHCTEN] and CMP_C3[NCHCTEN] are set to 1, the CMP is in continuous time operation. No special timing is required.
2. When either CMP_C3[PCHCTEN] or CMP_C3[NCHCTEN] is 0, it means at least one input comes from the 3V PAD.
   a. If CMP_C3[RDIVE] is zero, it means the input signal comes from 3V PAD, but its range still is in the range of 0 to 1.8V, no timing requirement for the generation of Phase1 and Phase2.

      In this condition, there are two modes for this comparator, high speed mode and low speed mode. Clock source may come from fast clock(16-20M)

      In Figure 627, the Analog ready indicates the analog settling time is reached and is ready for comparison. Tc is set by CMP_C3[ACSAT], in high speed mode, it is normally used as 1*T, 2*T, 4*T and 8*T where T is CLOCK period, and in low speed mode (CMP_C0[PMODE] = 0), CMP_C3[ACSAT] is normally used as 16*T, 32*T, 64*T and 256*T.

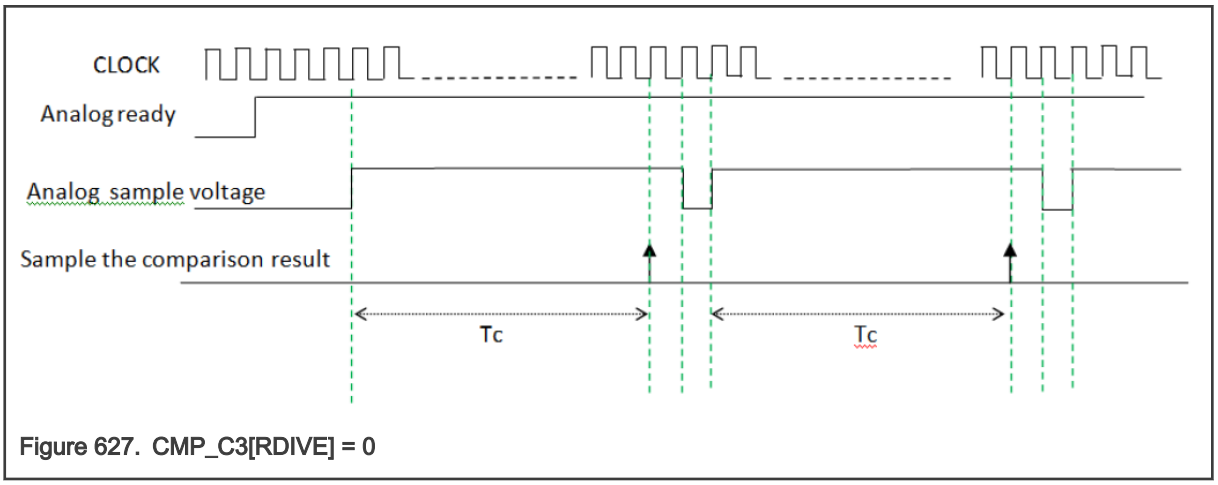


Figure 627. CMP_C3[RDIVE] = 0

   b. If CMP_C3[RDIVE] == 1, it means the signals come from the 3V PAD and it could over 1.8 V, Phase1, Phase2 will be generated.

      If CMP is in high speed mode (CMP_C0[PMODE] = 1) , Tc, Phase1, and Phase2 can be set based on the possible combinations in Table 1115.

      If CMP works in low speed mode, Phase1 is suggested to set to 1*T, Phase2 is suggested to set to 8*T, and Tc is normally set to 16*T, 32*T, 64*T and 256*T.

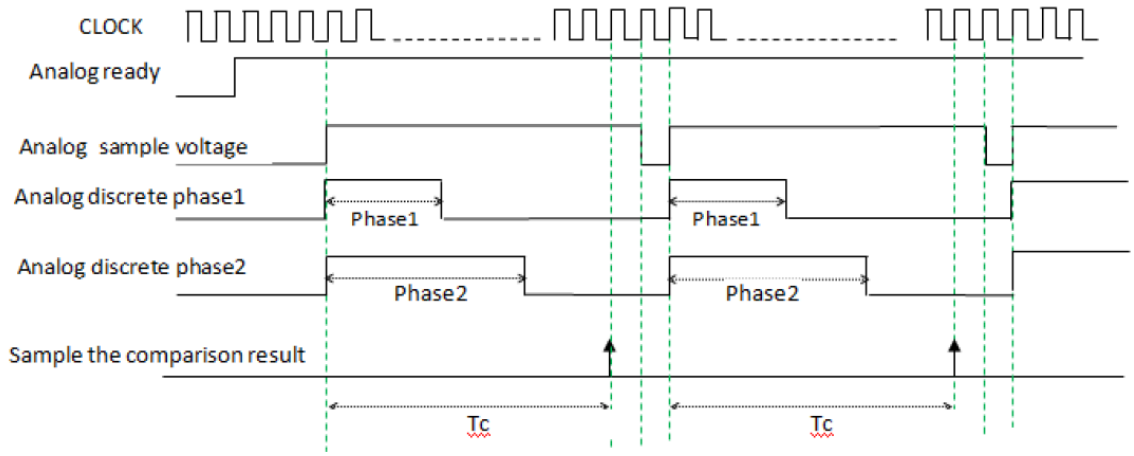


Figure 628. CMP_C3[RDIVE] = 1

Table 1115. Possible Combinations of CMP_C3[ACSAT], CMP_C3[ACPH1TC] and CMP_C3[ACPH2TC]

Tc	Phase1	Phase2
1*T	1*T	1*T
2*T	2*T	2*T
4*T	4*T	4*T
8*T	8*T	8*T
16*T	1*T	16*T
32*T	1*T	32*T
64*T	1*T	64*T
256*T	0	16*T

If the clock comes from 32KHz slow clock(CMP_C3[DMCLKSEL] = 0), both analog phase 1 and phase 2 will be driven to 0 and Tc is limited to T as shown in the figure below.

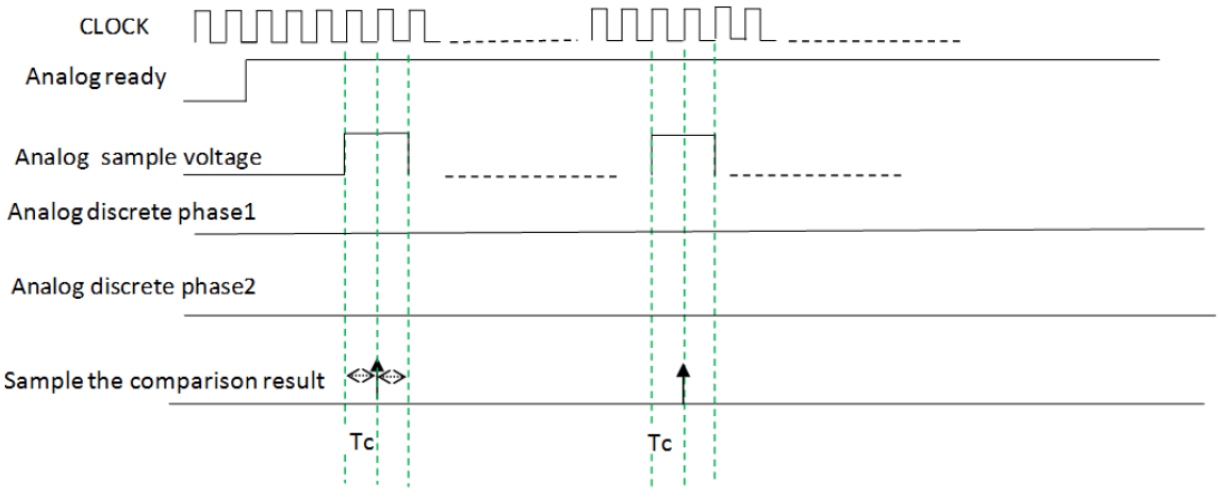


Figure 629. Slow (32 kHz) clock timing

81.3.4 CMP clocks

This module contains the following clocks:

- Bus clock for register access and window or filter operation
- Fast clock for Discrete mode
- Slow clock for Discrete mode
- Round-robin clock for Trigger mode

81.3.5 Interrupts

CMP generates an interrupt on either the rising- or falling-edge of the comparator output, or both. Assuming that C0[DMAEN] is not 1, the following table presents the conditions in which the interrupt request is asserted and deasserted.

Table 1116. CMP interrupt generations

Condition	Result
C0[IER] and C0[CFR] are 1.	The interrupt request is asserted.
C0[IEF] and C0[CFF] are 1.	The interrupt request is asserted.
C0[IER] and C0[CFR] are 0 for a rising-edge interrupt.	The interrupt request is deasserted.
C0[IEF] and C0[CFF] are 0 for a falling-edge interrupt.	The interrupt request is deasserted.

81.3.6 DMA support

CMP generates a CPU interrupt if there is a change on the COUT. The corresponding change on COUT forces a DMA transfer request rather than a CPU interrupt after you write 1 to C0[DMAEN] to enable the DMA support, and write 1 to C0[IER], C0[IEF], or both to enable the interrupt. After the DMA completes the transfer, it sends a transfer-completing indicator signal that deasserts the DMA transfer request and clears the flags (C0[CFR] and C0[CFF]) to allow a subsequent change on the comparator output to occur and force another DMA request.

The comparator can remain functional in stop modes. The corresponding change on COUT forces a DMA transfer request to wake up the system from stop modes after you write 1 to C0[DMAEN] to enable the DMA support and write 1 to C0[IER], C0[IEF], or both, to enable the interrupt. After the data transfer has finished, the system goes back to stop modes. See the DMA chapters in this document for more information on the asynchronous DMA function.

81.4 DAC functional description

81.4.1 DAC block diagram

The following figure shows a 256-tap resistor ladder network and a 256-to-1 multiplexer. They select an output voltage from one of the 256 distinct levels that outputs from DACO.

CMP Control 1 (C1) controls DAC. You can select its supply reference source from two sources, V_{in1} and V_{in2}. Power down or disable the module when not in use. In Disabled mode, DACO is connected to the analog ground.

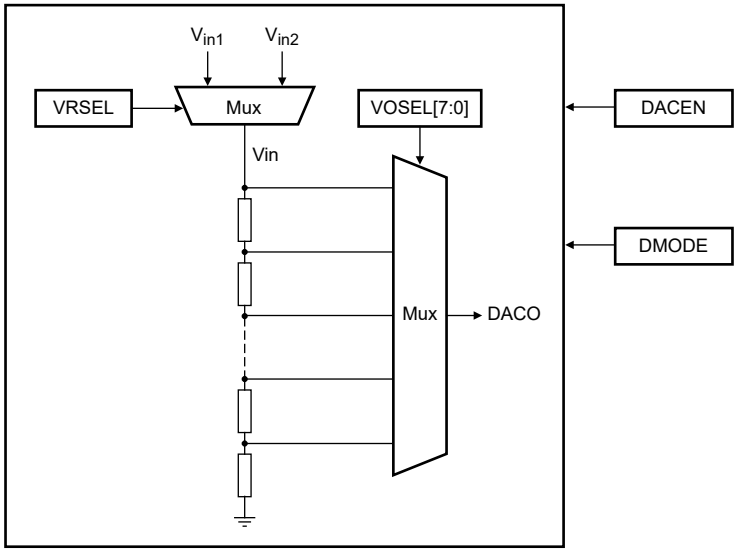


Figure 630. 8-bit DAC block diagram

81.4.2 Voltage reference source select

You must use:

- V_{in1} to connect to the primary voltage source as a supply reference of the 256-tap resistor ladder.
- V_{in2} to connect to an alternate voltage source, or primary source, if an alternate voltage source is not available.

81.4.3 DAC resets

This module contains a single reset input corresponding to the chip-wide peripheral reset.

81.4.4 DAC clocks

This module contains a single clock input—the bus clock.

81.4.5 DAC interrupts

This module contains no interrupts.

81.4.6 Trigger mode

CMP and the 8-bit DAC support the trigger mode operation, which is enabled when the MCU enters Stop mode with C0[WE], C0[SE], and C0[EN] = 1.

When this mode is enabled, the trigger events that include the operation clock and a trigger start signal initiate a compare sequence that must first enable CMP and DAC before performing a CMP operation and capturing the output. You can select a fixed channel for either the plus side mux or the minus side mux with C2[FXMP] and C2[FXMXCH]. You must set the round-robin cycling period for a longer duration than the time that all active channels take to complete the comparison cycles specified by C2[NSAM].

The active channels that C1[CHN*n*] selects route to the non fixed channel mux and compare with the reference input in a round-robin manner. To meet the comparator stabilization time, after the configurable number of operation clocks that C2[NSAM] defines, sample the comparison result for the selected channel. Write to C2[ACOn] to configure a software pre-programmed state for each channel. After you sample all the active channels, if the comparison result changes from its pre-programmed state, the corresponding flag in C2[CH*n*F] sets. An asynchronous reset asserts to bring the MCU out of Stop mode, if C2[RRIE] becomes 1.

Note that these flags do not support generating a DMA transfer event.

This mode is active when the MCU is in Stop mode, so no window or filter functions are available. A basic assumption of this mode is that the selected inputs change at a much slower rate than the operation clock. NXP recommends configuring the comparator in Low-Power Comparison mode as well. To program C2[INITMOD], you must ensure that the INITMOD × round-robin clock period must be longer than the initialization delay (see the chip data sheet for more information).

Figure 631 shows the basic flow of this mode. In the diagram, C1[CHN1], C1[CHN3], and C1[CHN4] are 1, so channels #1, #3, and #4 are selected for round-robin based on their polarity setting. When you write 1 to C2[NSAM], CMP samples the clock after one round-robin clock cycle. After you compare channel #3, the result is sampled, and round-robin ends. If any of the comparison results from channel #1, #3, or #4 deviates from its programmed value (written to C2[ACO1], C2[ACO3], and C2[ACO4]), an interrupt is generated to wake up the MCU from Stop mode. During Stop mode, you can poll C2[CH*n*F] to see the values of which channel input(s) are changed.

NOTE

In Round-Robin mode, you must ensure that the RTC_CLK period is greater than the comparison time corresponding to the value of C0[PMODE].

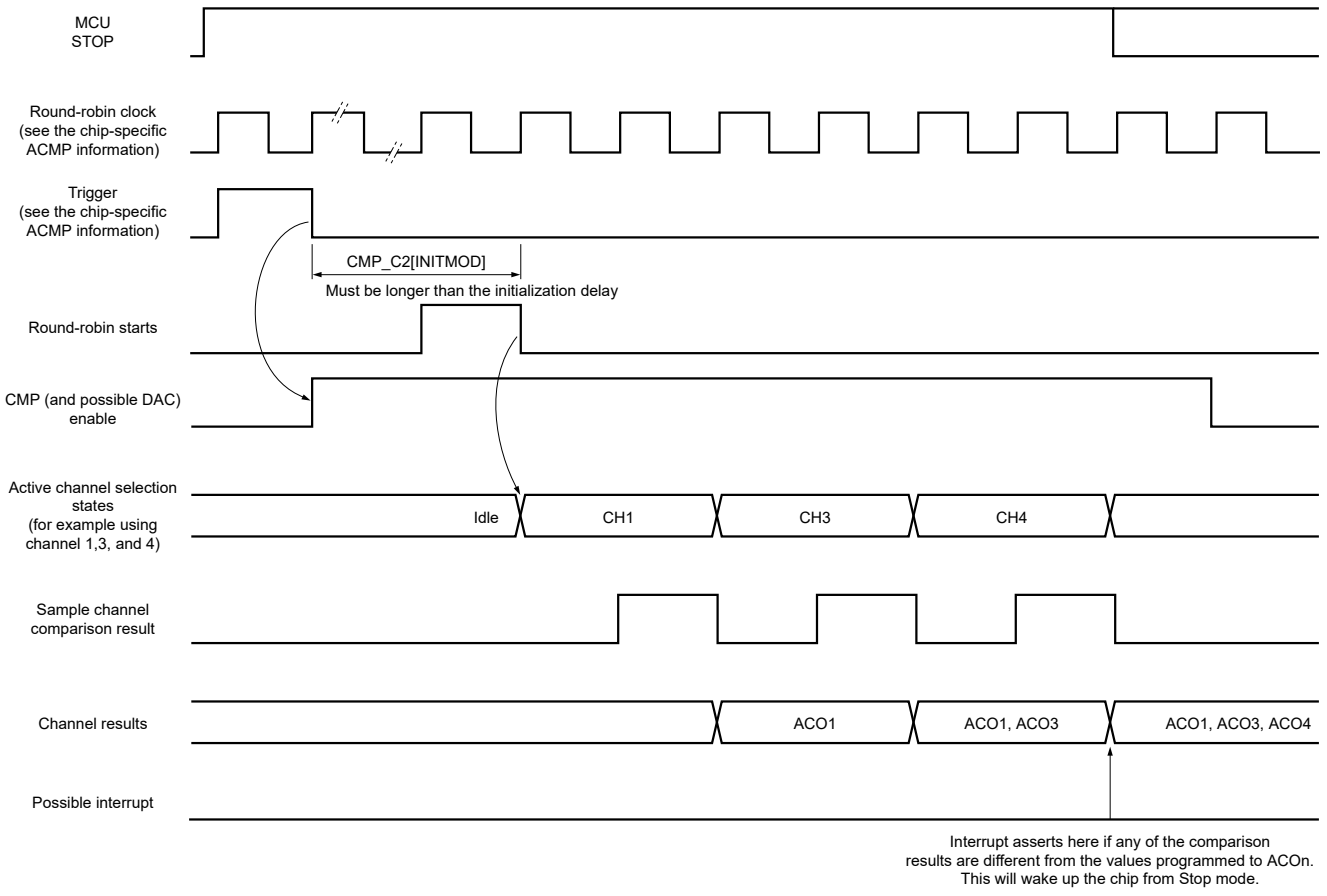


Figure 631. Trigger mode

81.5 CMP external signals

C1[PSEL] and C1[MSEL] mux the external 3V domain inputs, INA_3V[6:1] and 1.8 V domain INA_1P8 [4:1], INA_1P8_0_P, and INA_1P8_0_M, before going to the positive and negative ports of the comparator, respectively.

Table 1117. CMP external signals

Signal	Description	I/O
INA_3V[6:1]	The 3 V domain analog inputs [6:1]	I
INA_1P8 [4:1]	Analog 1.8 V input channels [4:1]	I
INA_1P8_0_P	Analog 1.8 V input channel 0 positive	I
INA_1P8_0_m	Analog 1.8 V input channel 0 minus	I

NOTE

It is not recommended to compare 3V domain channel with 1.8V domain channel. If the 3V domain channel's voltage is smaller than 1.8V, in this case, it can be compared with 1.8V domain channel by setting C3[RDIVE] to 0.

81.5.1 External pins

CMP contains two analog inputs:

- INP
- INM

Each pin can accept an input voltage that varies across the full operating range of the MCU. If the module is not enabled, you can use each pin as a digital input or output. See the specific MCU documentation for more information on what functions are shared with these analog inputs.

You can select either filtered or unfiltered comparator outputs for use on an external I/O pad.

81.6 Initialization

The time required to stabilize COUT is the power on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function, and filter. The windowing function has a maximum of one bus clock period delay. Low-pass filter specifies the filter delay.

During operation, you must always consider the propagation delay of the selected data paths. It may take many bus clock cycles for COUT and C0[CFR] or C0[CFF] to reflect an input change or a configuration change to one of the components involved in the data path.

When programmed for filtering modes, COUT initially equals 0 until sufficient clock cycles have elapsed to fill all stages of the filter. This occurs even if COUTA is at a logic 1.

81.7 ACMP register descriptions

81.7.1 ACMP memory map

CMP1 base address: 42DC_0000h

CMP2 base address: 42DD_0000h

CMP3 base address: 42DE_0000h

CMP4 base address: 42DF_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	R	0100_0000h
4h	Parameter Register (PARAM)	32	R	0000_0000h
8h	CMP Control 0 (C0)	32	RW	0000_0000h
Ch	CMP Control 1 (C1)	32	RW	0000_0000h
10h	CMP Control 2 (C2)	32	RW	0000_0000h
14h	CMP Control 3 (C3)	32	RW	1100_0000h

81.7.2 Version ID Register (VERID)

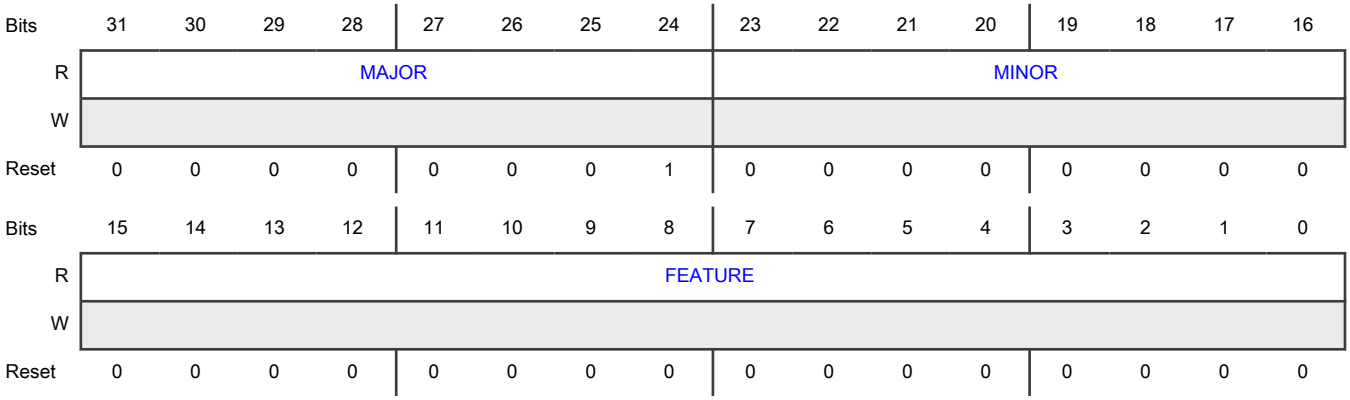
Offset

Register	Offset
VERID	0h

Function

Indicates the version integrated for this instance on the chip.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the module specification.
23-16 MINOR	Minor Version Number Indicates the minor version number for the module specification.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 FEATURE	Feature Specification Number Indicates the feature set number.

81.7.3 Parameter Register (PARAM)

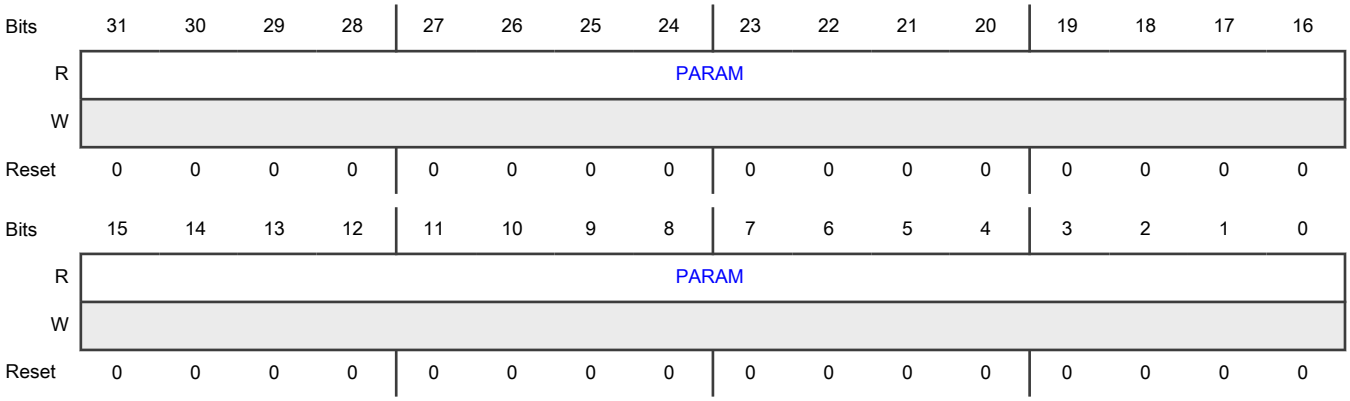
Offset

Register	Offset
PARAM	4h

Function

Indicates the feature parameters for this instance on the chip.

Diagram



Fields

Field	Function
31-0 PARAM	Parameters Indicates the feature parameters implemented along with Version ID Register (VERID).

81.7.4 CMP Control 0 (C0)

Offset

Register	Offset
C0	8h

Function

Contains configuration options for the CMP function, such as DMA, interrupt control, and filter function.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LINKEN	DMAEN	0	IER	IEF	CFR	CFF	COUT	FPR							
W						W1C	W1C									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SE	WE	0	PMODE	INVT	COS	OPE	EN	0	FILTER_CNT			0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LINKEN	CMP to DAC Link Enable Enables the link from CMP to DAC. If this field is 1, C0[EN] instead of C1[DACEN] controls the DAC enable and disable function. 0b - Disable 1b - Enable
30 DMAEN	DMA Enable Enables the DMA transfer that CMP triggers. If this field and the corresponding interrupt enable field are 1, a DMA request is asserted when C0[CFR] or C0[CFF] is set. 0b - Disable 1b - Enable
29 —	Reserved
28 IER	Comparator Interrupt Enable Rising Enables the CFR interrupt from CMP. If this field is 1, an interrupt asserts when C0[CFR] is set. 0b - Disable 1b - Enable
27 IEF	Comparator Interrupt Enable Falling Enables the CFF interrupt from CMP. If this field is 1, an interrupt asserts when C0[CFF] is set. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 CFR	Analog Comparator Flag Rising Detects a rising edge on COUT, when set, during normal operation. In stop mode, this field is level sensitive. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
25 CFF	Analog Comparator Flag Falling Detects a falling edge on COUT, when set, during normal operation. In Stop mode, this field is level sensitive . **NOTE** This field behaves differently for read and write operations. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
24 COUT	Analog Comparator Output Returns the current value of the analog comparator output, when read. The field resets to 0 and read as C0[INVT] when ACMP is disabled, that is, when C0[EN] = 0. Writes to this field are ignored.
23-16 FPR	Filter Sample Period Specifies the sampling period, in bus clock cycles, of the comparator output filter, when C0[SE] = 0. Writing 0h to this field disables the filter. See CMP functional description for more information on filter programming and latency. This field has no effect when C0[SE] = 1. In that case, the external sample signal determines the sampling period.
15 SE	Sample Enable Enables Sampling mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If a write to this register writes 1 both this field and C0[WE], then only Sampling mode is effective in MCU Run mode, and the round-robin cycling (Trigger mode) is enabled in MCU Stop mode. 0b - Disable 1b - Enable
14 WE	Windowing Enable Enables Windowing mode. If a write to this register writes 1 to both C0[SE] and this field, then only Sampling mode is effective in MCU Run mode, and the round-robin cycling (Trigger mode) is enabled in MCU Stop mode. 0b - Disable 1b - Enable
13 —	Reserved
12 PMODE	Power Mode Select Selects the power mode. 0b - Low-speed (LS) 1b - High-speed (HS)
11 INVT	Comparator Invert Selects the polarity of the analog comparator function. It is also driven to the COUT output (on the chip pin and as C0[COUT]) when C0[OPE] = 0. If the value of this field is 1, it inverts the comparator output. If the value of this field is 0, it does not invert the comparator output. 0b - Do not invert 1b - Invert
10 COS	Comparator Output Select Specifies whether CMPO must be set to equal COUT (filtered comparator output) or COUTA (unfiltered comparator output). 0b - COUT 1b - COUTA
9 OPE	Comparator Output Pin Enable Enables the path from the comparator output to a selected pin. If the value of this field is 0, the comparator output (after window or filter settings depending on software configuration) is not available to a packaged pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If the value of this field is 1 and the software configures the comparator to own a packaged pin, the comparator is available in a packaged pin. 0b - Disable 1b - Enable
8 EN	Analog Comparator Module Enable Enables ACMP. When the module is not enabled, the analog part remains in the off state, and consumes no power. When you select the same input from the analog mux to the positive and negative ports, the comparator is disabled automatically. 0b - Disable 1b - Enable
7 —	Reserved
6-4 FILTER_CNT	Filter Sample Count Specifies the number of consecutive samples that must agree prior to the comparator output filter accepting a new output state. 000b - Filter is disabled (if C0[SE] = 1, then COUT is a logic zero (this is not a legal state, and is not recommended); if C0[SE] = 0, COUT = COUTA) 001b - One consecutive sample (comparator output is simply sampled) 010b - Two consecutive samples 011b - Three consecutive samples 100b - Four consecutive samples 101b - Five consecutive samples 110b - Six consecutive samples 111b - Seven consecutive samples
3-2 —	Reserved
1-0 —	Reserved

81.7.5 CMP Control 1 (C1)

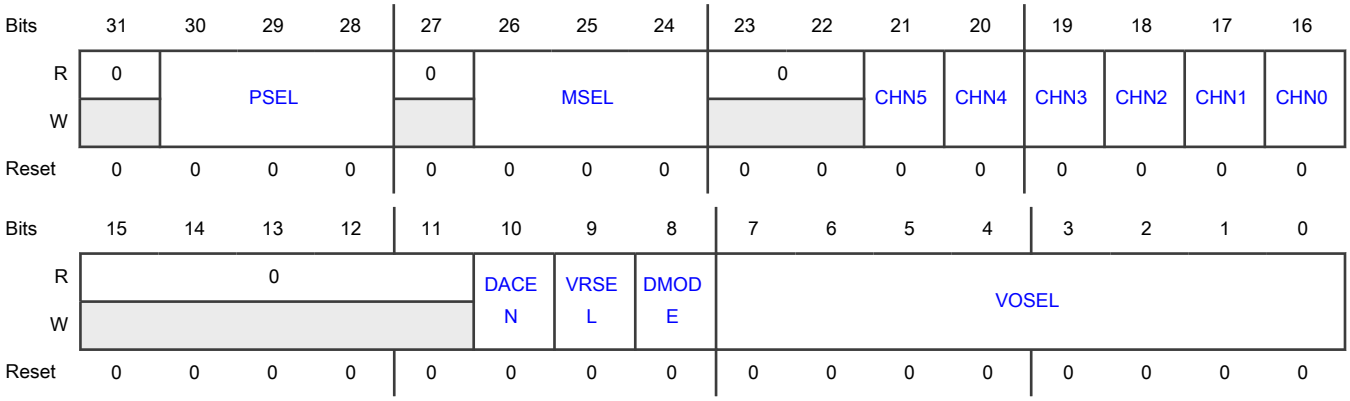
Offset

Register	Offset
C1	Ch

Function

Contains configuration options for CMP channel selection as well as the control for the analog DAC.

Diagram



Fields

Field	Function
31 —	Reserved
30-28 PSEL	Plus Input MUX Control Determines which input is selected for the positive mux. See the chip configuration chapters for more information on connections about reference inputs {0,1,2,3,4,5} and internal positive input 0. The 8-bit DAC output connects internally. Note: In Round-Robin mode of operation, you must exclude internal positive input 0 from the active and fixed channel ports. C1[MSEL] and this field must have different values. 000b - Internal positive input 0 for plus channel (internal plus input) 001b - External input 1 for plus Channel (reference input 0) 010b - External input 2 for plus channel (reference input 1) 011b - External input 3 for plus channel (reference input 2) 100b - External input 4 for plus channel (reference input 3) 101b - Reserved 110b - Reserved 111b - Internal 8-bit DAC output
27 —	Reserved
26-24 MSEL	Minus Input MUX Control Determines which input is selected for the negative mux. See the chip configuration chapters for more information on connections about reference inputs {0,1,2,3,4,5} and internal negative input 0. The 8-bit DAC output connects internally.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Note:In Round-Robin mode of operation, you must exclude internal negative input 0 from the active and fixed channel ports. This field and C1[PSEL] must have different values. 000b - Internal negative input 0 for minus channel (internal minus input) 001b - External input 1 for minus channel (reference input 0) 010b - External input 2 for minus channel (reference input 1) 011b - External input 3 for minus channel (reference input 2) 100b - External input 4 for minus channel (reference input 3) 101b - Reserved 110b - Reserved 111b - Internal 8-bit DAC output
23-22 —	Reserved
21 CHN5	Channel 5 Input Enable Enables channel 5 of the input for the round-robin checker. Writing 1 to this field enables the corresponding channel to the non fixed mux port to check its voltage value in Round-Robin mode. If you select the same channel as the reference voltage, this field has no effect. 0b - Disable 1b - Enable
20 CHN4	Channel 4 Input Enable Enables channel 4 of the input for the round-robin checker. Writing 1 to this field enables the corresponding channel to the non fixed mux port to check its voltage value in Round-Robin mode. If you select the same channel as the reference voltage, this field has no effect. 0b - Disable 1b - Enable
19 CHN3	Channel 3 Input Enable Enables channel 3 of the input for the round-robin checker. Writing 1 to this field enables the corresponding channel to the non fixed mux port to check its voltage value in Round-Robin mode. If you select the same channel as the reference voltage, this field has no effect. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 CHN2	Channel 2 Input Enable Enables channel 2 of the input for the round-robin checker. Writing 1 to this field enables the corresponding channel to the non fixed mux port to check its voltage value in Round-Robin mode. If you select the same channel as the reference voltage, this field has no effect. 0b - Disable 1b - Enable
17 CHN1	Channel 1 Input Enable Enables channel 1 of the input for the round-robin checker. Writing 1 to this field enables the corresponding channel to the non fixed mux port to check its voltage value in Round-Robin mode. If you select the same channel as the reference voltage, this field has no effect. 0b - Disable 1b - Enable
16 CHN0	Channel 0 Input Enable Enables channel 0 of the input for the round-robin checker. Writing 1 to this field enables the corresponding channel to the non fixed mux port to check its voltage value in Round-Robin mode. If you select the same channel as the reference voltage, this field has no effect. 0b - Disable 1b - Enable
15-11 —	Reserved
10 DACEN	DAC Enable Enables the DAC. After the DAC is disabled, it is powered down to conserve power. 0b - Disable 1b - Enable
9 VRSEL	Supply Voltage Reference Source Select Selects a supply voltage reference source. If this field is 0, Vin1 is selected as resistor ladder network supply reference Vin. Vin1 is from internal PMC. If this field is 1, Vin2 is selected as resistor ladder network supply reference Vin. Vin2 is from PAD. 0b - Vin1 1b - Vin2

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 DMODE	DAC Mode Select Selects a DAC mode. 0b - Low-Speed and Low-Power mode 1b - High-Speed and High-Power mode
7-0 VOSEL	DAC Output Voltage Select Selects an output voltage from one of 256 distinct levels. DACO = (Vin/256) * (VOSEL[7:0] + 1), so the DACO range is from Vin/256 to Vin.

81.7.6 CMP Control 2 (C2)

Offset

Register	Offset
C2	10h

Function
Contains configuration options for Trigger mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	RRIE	FXMP	0	FXMXCH				Reserved							
											CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
W											W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NSAM		INITMOD						0		ACOn					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30	Round-Robin Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
RRIE	Enables the interrupt or wake-up when the comparison result changes for a given channel from the last sample. 0b - Disable 1b - Enable
29 FXMP	Fixed MUX Port Fixes the analog mux port for Round-Robin mode. If this field is 0, it fixes the plus port. Sweeps only the minus port in each round. If this field is 1, it fixes the minus port. Sweeps only the plus port in each round. 0b - Fix plus port 1b - Fix minus port
28 —	Reserved
27-25 FXMXCH	Fixed Channel Select Indicates which channel is selected as the fixed reference input for the fixed mux port in a given round-robin mode. 000b - External reference input 0 001b - External reference input 1 010b - External reference input 2 011b - External reference input 3 100b - External reference input 4 101b - External reference input 5 110b - Reserved 111b - 8-bit DAC
24-22 —	Reserved
21 CH5F	External Channel 5 Input Changed Flag Sets if the channel 5 input is changed from the last comparison with the fixed mux port.
20 CH4F	External Channel 4 Input Changed Flag Sets if the channel 4 input is changed from the last comparison with the fixed mux port.
19 CH3F	External Channel 3 Input Changed Flag Sets if the channel 3 input is changed from the last comparison with the fixed mux port.
18	External Channel 2 Input Changed Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
CH2F	Sets if the channel 2 input is changed from the last comparison with the fixed mux port.
17 CH1F	External Channel 1 Input Changed Flag Sets if the channel 1 input is changed from the last comparison with the fixed mux port.
16 CH0F	External Channel 0 Input Changed Flag Sets if the channel 0 input is changed from the last comparison with the fixed mux port.
15-14 NSAM	Number of Sample Clocks Specifies the number of round-robin clock cycles after which the sampling takes place. 00b - As soon as the active channel is scanned in one round-robin clock 01b - After one round-robin clock cycle 10b - After two round-robin clock cycles 11b - After three round-robin clock cycles
13-8 INITMOD	Comparator and DAC Initialization Delay Modulus Specifie the number of round robin clock cycles that determines the comparator and DAC initialization delays specified in the chip data sheet. For example, if the initialization delay is 80 µs and the round robin clock is 100 kHz, then program this field to 80 µs/10 µs = 8. 000000 - 64 (same with 111111) Other values - Initialization delay sets to INITMOD × round robin clock period
7-6 —	Reserved
5-0 ACOn	ACOn Specifies the result of the input comparison for channel *n*. This field stores the latest comparison result of the input channel *n* with the fixed mux port. Reading this field returns the latest comparison result. Writing to this field defines the pre-set state of channel *n*.

81.7.7 CMP Control 3 (C3)

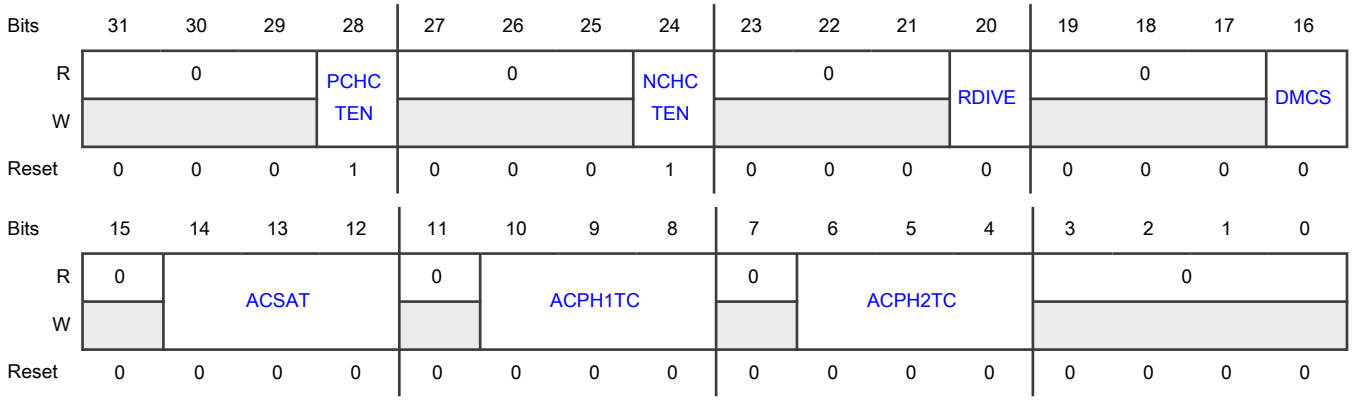
Offset

Register	Offset
C3	14h

Function

Contains configuration options for the discrete mode.

Diagram



Fields

Field	Function
31-29 —	Reserved
28 PCHCTEN	Positive Channel Continuous Mode Enable Enables the plus channel working in continuous mode. When this bit is set to 1, it means all the inputs to plus channels are coming from 1.8v domain and no discrete timing is required for the plus channel mux. Otherwise, when it is cleared to 0, plus channel will work in discrete mode, means the positive inputs are coming from 3v domain. 0b - in Discrete Mode and special timing needs to be configured 1b - in Continuous Mode and no special timing is required
27-25 —	Reserved
24 NCHCTEN	Negative Channel Continuous Mode Enable Enables the minus channel working in continuous mode. When this bit is set to 1, it means all the inputs to minus channels are coming from 1.8v domain and no discrete timing is required for the plus channel mux. Otherwise, when it is cleared to 0, minus channel will work in discrete mode, means the negative inputs are coming from 3v domain. 0b - in Discrete Mode and special timing needs to be configured. 1b - in Continuous Mode and no special timing is required.
23-21 —	Reserved
20 RDIVE	Resistor Divider Enable Enables the resistor divider for the inputs when they come from 3v domain and their values are above 1.8v.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable even when either NCHEN or PCHEN is set to1 but the actual input is in the range of 0 to 1.8v. 1b - Enable because the inputs are above 1.8v.
19-17 —	Reserved
16 DMCS	Discrete Mode Clock Select Selects the clock source in order to generate the required timing for comparator to work in discrete mode. There are generally two kinds of clocks coming from SoC, one is a fast clock(16 to 20 MHz) to generate fast prop delay, and another one is normally 32 kHz to work in low power mode. 0b - Slow clock is selected for the timing generation. 1b - Fast clock is selected for the timing generation.
15 —	Reserved
14-12 ACSAT	Analog Comparator Sampling Time Control Configures the analog comparator sampling timing (specified by the discrete mode clock period T which is selected by DMCS) in discrete mode. 000b - The sampling time equals to T 001b - The sampling time equals to 2 * T 010b - The sampling time equals to 4 * T 011b - The sampling time equals to 8 * T 100b - The sampling time equals to 16 * T 101b - The sampling time equals to 32 * T 110b - The sampling time equals to 64 * T 111b - The sampling time equals to 256 * T
11 —	Reserved
10-8 ACPH1TC	Analog Comparator Phase 1 Timing Control Configures the analog comparator phase 1 timing when RDIV is set to 1, which means the input voltage level is above 1.8v. T means the discrete mode clock period in the table. 000b - Phase 1 active time in one sampling period equals to T 001b - Phase 1 active time in one sampling period equals to 2 * T 010b - Phase 1 active time in one sampling period equals to 4 * T 011b - Phase 1 active time in one sampling period equals to 8 * T

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - Phase 1 active time in one sampling period equals to T 101b - Phase 1 active time in one sampling period equals to T 110b - Phase 1 active time in one sampling period equals to T 111b - Phase 1 active time in one sampling period equals to 0
7 —	Reserved
6-4 ACPH2TC	Analog Comparator Phase 2 Timing Control Configures the analog comparator phase 2 timing when RDIV is set to 1, which means the input voltage level is above 1.8v. T means the discrete mode clock period in the table. 000b - Phase 2 active time in one sampling period equals to T 001b - Phase 2 active time in one sampling period equals to 2 * T 010b - Phase 2 active time in one sampling period equals to 4 * T 011b - Phase 2 active time in one sampling period equals to 8 * T 100b - Phase 2 active time in one sampling period equals to 16 * T 101b - Phase 2 active time in one sampling period equals to 32 * T 110b - Phase 2 active time in one sampling period equals to 64 * T 111b - Phase 2 active time in one sampling period equals to 16 * T
3-0 —	Reserved

Chapter 82
12-bit Digital-to-Analog Converter (DAC)

82.1 Chip-specific DAC information

Table 1118. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

82.2 Overview

The 12-bit digital-to-analog converter (DAC) is a low-power, general-purpose DAC. The output of the DAC can be placed on an external pin or set as one of the inputs to the analog comparator, op-amps, or ADC.

82.2.1 Block diagram

The block diagram of the DAC module is as follows:

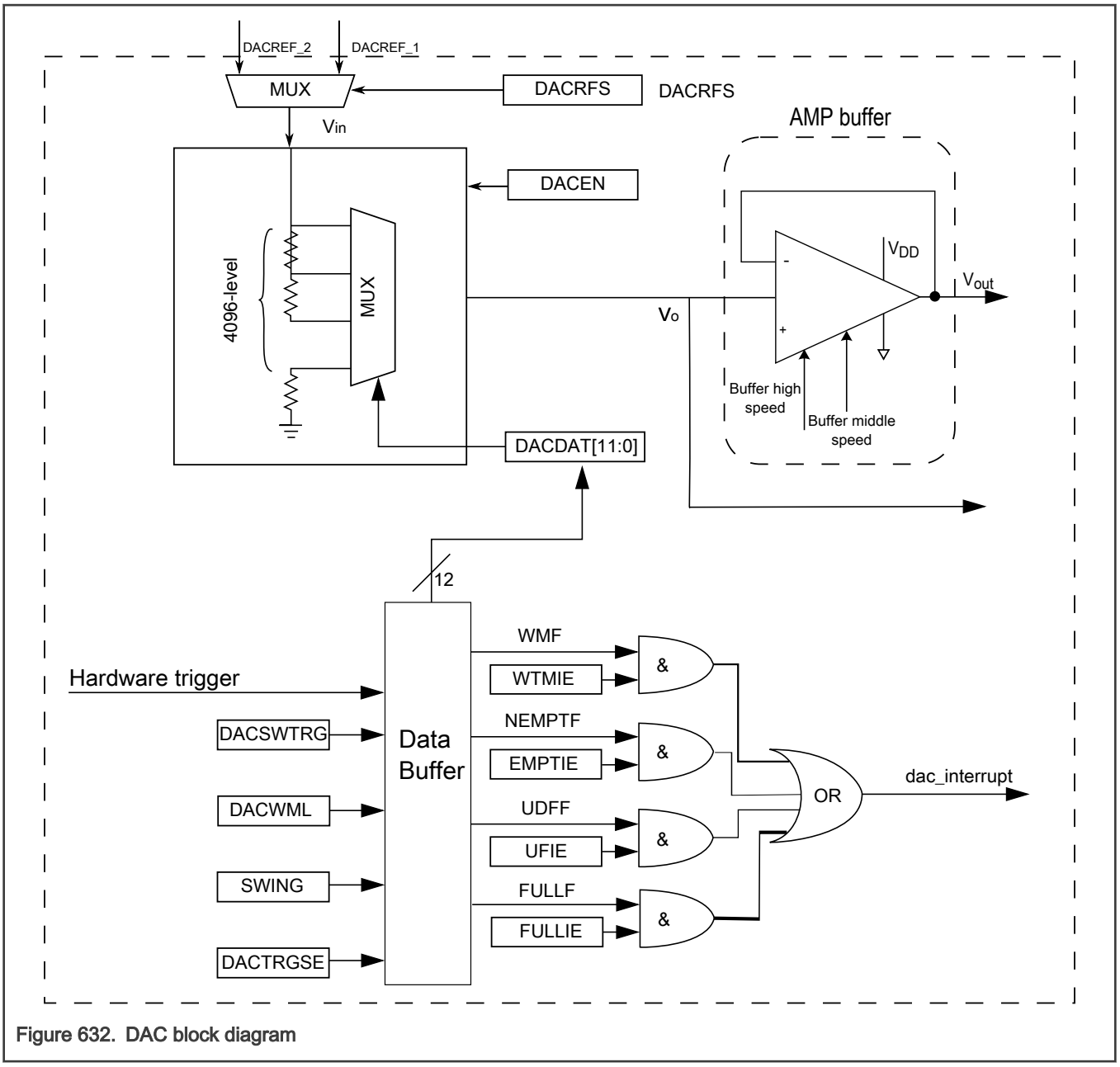


Figure 632. DAC block diagram

82.2.2 Features

The features of the DAC module include:

- On-chip programmable reference generator output. The voltage output range is from 1⁄4096 V_{in} to V_{in}, and the step is 1⁄4096 V_{in}, where V_{in} is the input voltage.
- V_{in} can be selected from two reference sources. See Chip-Specific Information section.
- 16 -word data buffer supported with configurable watermark and multiple operation modes
- DMA support

82.3 Functional description

The 12-bit DAC module can select one of the two reference inputs - DACREF_1 and DACREF_2 as the DAC reference voltage, V_{in} by C0[DACRFS]. See the chip-specific DAC information to determine the source options for DACREF_1 and DACREF_2.

When the DAC is enabled, it converts the data in DATA[11:0] or the data from the DAC data buffer to a stepped analog output voltage. The output voltage range is from V_{in} to $V_{in}/4096$, and the step is $V_{in}/4096$.

82.3.1 FIFO mode

In FIFO mode, the buffer is organized as a FIFO. For a valid write to the DATA register, the data is put into the FIFO, and the write pointer is automatically incremented. The module is connected internally to a 32-bit interface. For any 16-bit or 8-bit FIFO access, address bit[1] needs to be 0; otherwise, the write is ignored. A successful 32-bit FIFO write increases the write pointer by one.

For an 8-bit write, the LSB determines which byte lane is written to. Only if both the byte lanes are written to, will the write pointer increase. The user needs to ensure that an 8-bit access is done in pair, and both the upper and lower bytes are written. There is no priority for which byte is written to first.

FIFO mode needs to be enabled with DACCR[FIFOEN].

82.3.2 DAC data buffer operation

When the DAC FIFO is disabled, and the one entry buffer is enabled, the DAC converts the data in the buffer to analog output voltage. Any write to the DATA register will replace the data in the buffer and push data to analog conversion without trigger support.

When the IP interface bus access (IPS) is in write state, the write speed of DATA[DATA0] is high. However, the analog part of the DAC cannot convert data at the same speed. Therefore, the multi write speed limit needs to be defined in the firmware.

82.3.3 DAC interrupts

There are several interrupts and associated flags that can be configured for the DAC buffer.

FIFO request flag

The FIFO request flag is set based on the number of entries in the FIFO and the FIFO watermark configuration. This flag is set when the number of entries in any of the enabled FIFOs is less than or equal to the FIFO watermark configuration and is automatically cleared when the number of entries in FIFO is greater than the FIFO watermark configuration. The FIFO request flag can generate an interrupt or a DMA request.

FIFO warning flag

The FIFO warning flag is set based on the number of entries in the FIFO. The empty flag and full flags are both warning flags. The FIFO warning flag can generate an interrupt, or a DMA request (for empty flag only).

FIFO error flag

The FIFO error flag is set when overflow and underflow happens.

82.3.4 DMA operation

When DMA is enabled, DMA requests are generated instead of interrupt requests. The DMA Done signal clears the DMA request.

The status register flags are still set and are cleared automatically when the DMA completes.

82.3.5 Resets

During reset, the DAC is configured in the default mode and is disabled.

82.3.6 Low-Power mode operation

The following table shows the wait mode and the stop mode operation of the DAC module.

Table 1119. Modes of operation

Modes of operation	Description
Wait mode	The DAC will operate normally, if enabled.
Stop mode	If enabled, the DAC module continues to operate normally, with the hardware trigger initiating the conversion.

NOTE

The assignment of module modes to core modes is chip-specific. For module-to-core mode assignments, see the chapter that describes how modules are configured.

82.3.7 Background Debug Mode Operation

When the microcontroller is in active background debug mode, the DAC continues to operate normally.

82.4 Memory map/register definition

The DAC has registers to control analog comparator and programmable voltage divider to perform the digital-to-analog functions.

82.4.1 DAC register descriptions

82.4.1.1 DAC memory map

DAC base address: 42E2_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version Identifier Register (VERID)	32	R	0100_0000h
4h	Parameter Register (PARAM)	32	R	0000_0003h
8h	DAC Data Register (DATA)	32	RW	0000_0000h
Ch	DAC Status and Control Register (CR)	32	RW	0000_0002h
10h	DAC FIFO Pointer Register (PTR)	32	R	0000_0000h
14h	DAC Status and Control Register 2 (CR2)	32	RW	0000_0000h

82.4.1.2 Version Identifier Register (VERID)

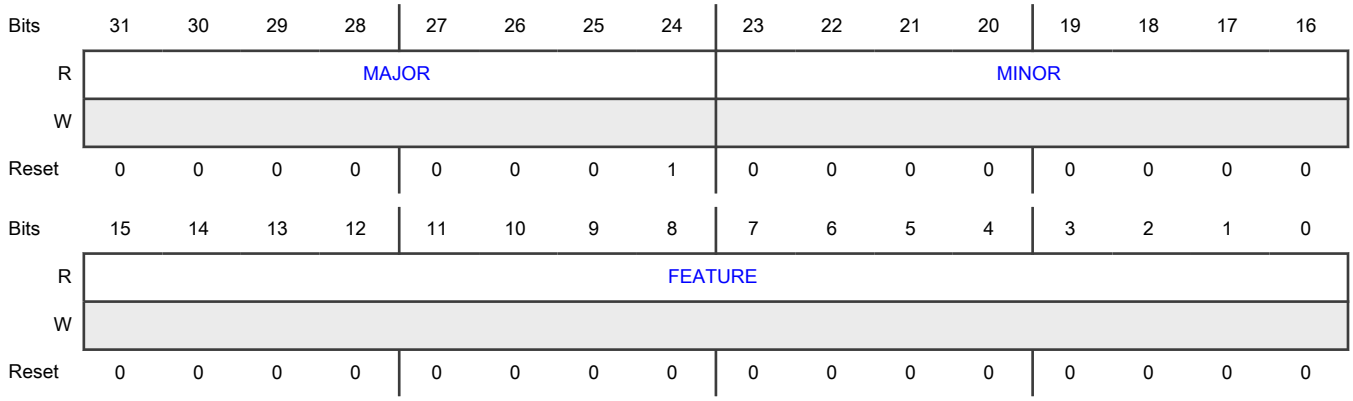
Offset

Register	Offset
VERID	0h

Function

The Version ID register indicates the version integrated for this instance on the device.

Diagram



Fields

Field	Function
31-24 MAJOR	Major version number This read-only field returns the major version number for the module specification.
23-16 MINOR	Minor version number This read-only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number This read-only field returns the feature set number. 0000_0000_0000_0000b - Standard feature set 0000_0000_0000_0001b - C40 feature set 0000_0000_0000_0010b - 5V DAC feature set 0000_0000_0000_0100b - ADC BIST feature set

82.4.1.3 Parameter Register (PARAM)

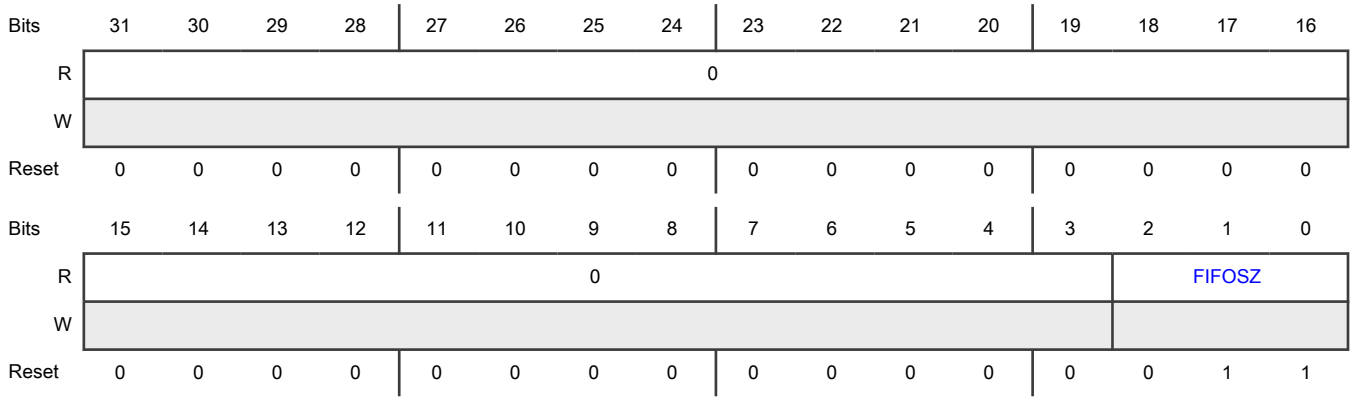
Offset

Register	Offset
PARAM	4h

Function

The Parameter register indicates the feature parameters for this instance on the device.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 FIFOSZ	FIFO size The number of words in the DAC FIFO is 2^(FIFOSZ+1). 000b - FIFO depth is 2 001b - FIFO depth is 4 010b - FIFO depth is 8 011b - FIFO depth is 16 100b - FIFO depth is 32 101b - FIFO depth is 64 110b - FIFO depth is 128 111b - FIFO depth is 256

82.4.1.4 DAC Data Register (DATA)

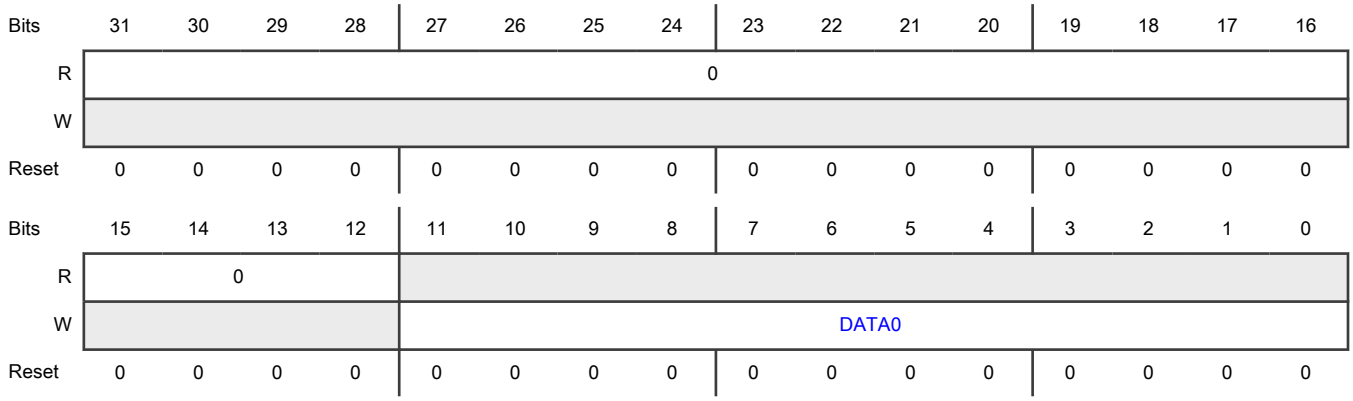
Offset

Register	Offset
DATA	8h

Function

The DAC Data Register holds the entry of FIFO in FIFO mode. It also works as the data buffer for buffer mode.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 DATA0	FIFO DATA0

82.4.1.5 DAC Status and Control Register (CR)

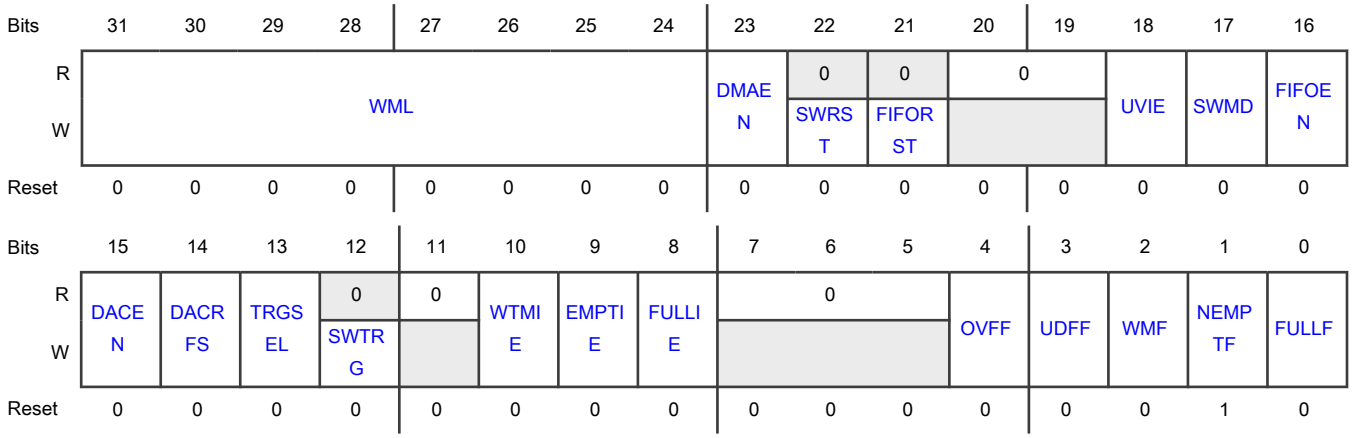
Offset

Register	Offset
CR	Ch

Function

The DAC Status and Control Register contains configuration options for the DAC operation, such as DMA and interrupt, trigger select and FIFO mode select. It also includes the FIFO status flags.

Diagram



Fields

Field	Function
31-24 WML	Watermark Level Select MAX is the FIFO size. **NOTE** If the FIFO depth is 2, then all settings are treated the same. That is, if the FIFO is not full, the watermark status bit and NEMPTF will be set. One of the status bits maybe needed for DMA or IRQ request.
23 DMAEN	DMA Enable Select 0b - DMA is disabled. 1b - DMA is enabled. When DMA is enabled, the DMA request will be generated by original interrupts. The interrupts will not be presented on this module at the same time.
22 SWRST	Software reset Resets all internal logic and registers, in case of any failure during the Low-power mode. This field is written to only to trigger reset, self clear, or read 0.
21 FIFORST	FIFO Reset Resets the FIFO pointers. Reading this field will always return zero. FIFO pointers should only be reset when the DAC is disabled. This field can be used to configure both pointers to the same address to reset the FIFO as empty. This field is written to only to trigger reset, self clear, or read zero. 0b - No effect 1b - FIFO reset
20-19 —	Reserved
18 UVIE	Underflow and overflow interrupt enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Underflow and overflow interrupt is disabled. 1b - Underflow and overflow interrupt is enabled.
17 SWMD	DAC FIFO Mode Select In Swing back mode, the FIFO must be set up such that the FIFO is full. In Swing back mode, a trigger changes the read pointer to make it swing between the FIFO Full and Nearly Empty state. That is, the trigger increases the read pointer till FIFO is nearly empty and decreases the read pointer till the FIFO is full, and so on. 0b - Normal mode 1b - Swing back mode
16 FIFOEN	FIFO Enable 0b - FIFO is disabled and only one level buffer is enabled. Any data written from this buffer goes to conversion. 1b - FIFO is enabled. Data will first read from FIFO to buffer then go to conversion.
15 DACEN	DAC Enable Starts the Programmable Reference Generator operation. 0b - The DAC system is disabled. 1b - The DAC system is enabled.
14 DACRFS	DAC Reference Select 0b - The DAC selects DACREF_1 as the reference voltage. 1b - The DAC selects DACREF_2 as the reference voltage.
13 TRGSEL	DAC Trigger Select 0b - The DAC hardware trigger is selected. 1b - The DAC software trigger is selected.
12 SWTRG	DAC Software Trigger Active high. This is a write-only field, which always reads 0. If DAC software trigger is selected and FIFO is enabled, writing 1 to this field will advance the FIFO read pointer once. **NOTE** It is not suggested to adopt hardware trigger and software trigger in mixed use, during continuous conversions. 0b - The DAC soft trigger is not valid. 1b - The DAC soft trigger is valid.
11 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 WTMIE	Watermark Interrupt Enable 0b - Watermark interrupt is disabled. 1b - Watermark interrupt is enabled.
9 EMPTIE	Nearly Empty Interrupt Enable 0b - FIFO Nearly Empty interrupt is disabled. 1b - FIFO Nearly Empty interrupt is enabled.
8 FULLIE	Full Interrupt Enable 0b - FIFO Full interrupt is disabled. 1b - FIFO Full interrupt is enabled.
7-5 —	Reserved
4 OVFF	Overflow Flag This is the FIFO overflow status flag. This flag indicates that more data has been written into FIFO than it can hold. This field asserts regardless of the value of UVIE. However, an interrupt is issued to the host only if UVIE is set. This flag is cleared by writing a 1 to it. 0b - No overflow has occurred since the last time the flag was cleared. 1b - At least one FIFO overflow has occurred since the last time the flag was cleared.
3 UDFF	Underflow Flag This is the FIFO underflow status flag. If this field is set, it means that there is a new trigger after NEMPTF is set. This field asserts regardless of the value of UVIE. However, an interrupt is issued to the host only if UVIE is set. This flag is cleared by writing a 1 to it. 0b - No underflow has occurred since the last time the flag was cleared. 1b - At least one trigger underflow has occurred since the last time the flag was cleared.
2 WMF	FIFO Watermark Status Flag This field is set if the remaining FIFO data is less than or equal to the watermark setting. It is cleared automatically by writing data into FIFO by DMA or CPU. Write to this bit is ignored in FIFO mode. 0b - The DAC buffer read pointer has not reached the watermark level. 1b - The DAC buffer read pointer has reached the watermark level.
1 NEMPTF	Nearly Empty Flag This is the FIFO nearly empty flag. It is set when only one data remains in FIFO. 0b - More than one data is available in the FIFO. 1b - One data is available in the FIFO.
0	Full Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
FULLF	This is the FIFO Full status bit. This means that the FIFO read pointer equals the write pointer, because of the write pointer increase. This field is cleared if any DAC trigger is making the DAC read pointer increase. Any write to this field is ignored in FIFO mode. 0b - FIFO is not full. 1b - FIFO is full.

82.4.1.6 DAC FIFO Pointer Register (PTR)

Offset

Register	Offset
PTR	10h

Function

The DAC FIFO Pointer Register contains the FIFO read pointer and write pointer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								DACRFP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								DACWFP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 DACRFP	DACRFP This is the FIFO read pointer.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The read pointer is later than the trigger for one cycle. After the conversion finishes and the next trigger comes, then the read pointer updates in the hardware trigger mode.
15-8 —	Reserved
7-0 DACWFP	DACWFP This is the FIFO write pointer. It's value is initially set to equal the read pointer value automatically, and the FIFO status is empty. Both write pointer and read pointer in FIFO mode do not change by IPS read access.

82.4.1.7 DAC Status and Control Register 2 (CR2)

Offset

Register	Offset
CR2	14h

Function

The DAC Status and Control Register 2 contains configuration options for the analog DAC.

NOTE

For CR2[6:4] (IREF, IREF1 and IREF2), it is suggested to enable one of the bits to reduce power. If more than one bit is selected, the power consumption will increase but the conversion rate will also speed up.

NOTE

IREF works in all modes, but the operation mode of IREF1 or IREF2 depends on PMC'.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								IREF	IREF1	IREF2	BFHS	BFMS	OEN	BFEN	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved
6 IREF	Internal Current Reference Select 0b - Internal Current Reference not selected 1b - Internal Current Reference selected
5 IREF1	Internal ZTC (Zero Temperature Coefficient) Current Reference Select 0b - Internal ZTC Current Reference not selected 1b - Internal ZTC Current Reference selected
4 IREF2	Internal PTAT (Proportional To Absolute Temperature) Current Reference Select 0b - Internal PTAT Current Reference not selected 1b - Internal PTAT Current Reference selected
3 BFHS	Buffer High Speed Select 0b - Buffer high speed not selected 1b - Buffer high speed selected
2 BFMS	Buffer Middle Speed Select 0b - Buffer middle speed not selected 1b - Buffer middle speed selected
1 OEN	Optional Enable 0b - Output buffer is not bypassed 1b - Output buffer is bypassed
0 BFEN	Buffer Enable 0b - Opamp is not used as buffer 1b - Opamp is used as buffer

Chapter 83
SINC Filter (SINC)

83.1 Chip-specific SINC information

Table 1120. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

NOTE

Alternative DMA is not supported, access register C0ACFR[ADMASEL] has unpredictable results.

NOTE

Input signals INP[3:0] are not connected for RT1180, program CnCFR[IBSEL]=01b are unpredictable.

83.2 Overview

SINC converts an external ADC sigma-delta modulator bitstream to a data stream. The converters are based on:

- A maximum of 3-order sinc digital decimation filters with FastSinc support.
- A maximum selectable oversampling ratio (OSR) of 2048.

You can combine SINC functionality with additional software-based filtering.

83.2.1 Block diagram

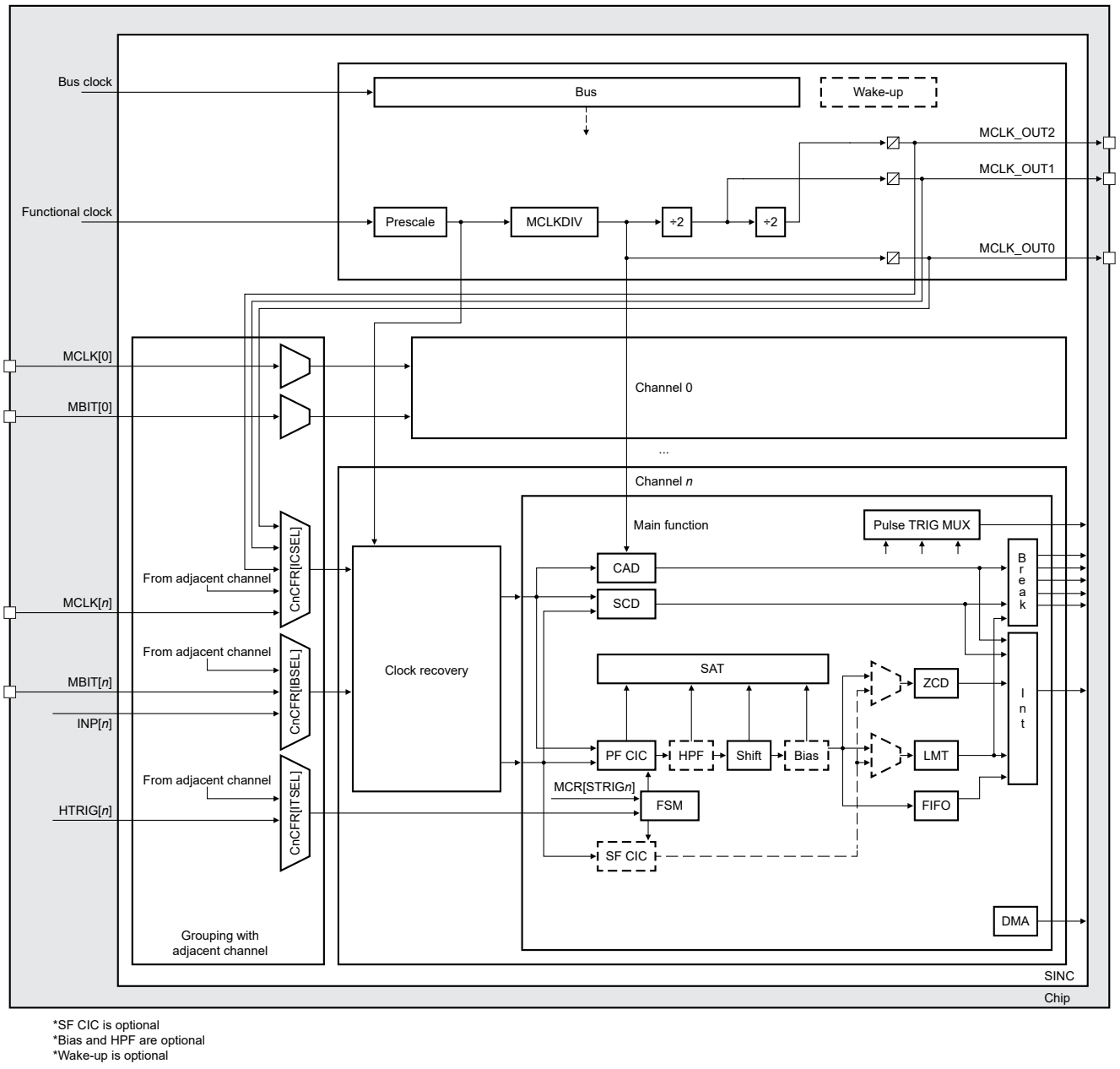


Figure 633. Block diagram

83.2.2 Features

- Filter channels with dedicated modulator clocks, bitstreams, and triggers
- Modulator clock outputs
- One software and one hardware trigger per channel
- One primary filter (PF) for each channel, with a shift and cascaded integrator comb (CIC) filter, for 24-bit precision
- Break signal outputs to turn off the external peripheral when a protection event occurs
- A 16-entry FIFO with watermark support for each channel

- Two DMA requests per channel with various event sources
- Various interrupt sources available for each channel (see Interrupts and breaks)
- Programmable high and low detection limits
- Support for short-circuit detector (SCD), clock-absence detector (CAD), and zero-cross detector (ZCD)
- Programmable shift operations
- Programmable bias (offset) operations
- high-pass filters (HPF) for DC removal
- Support for a Manchester-encoded bitstream
- Pulse-triggered output for additional control and monitoring

83.3 Functional description

83.3.1 Components

SINC includes the following two main components.

Table 1121. Components

Component	Supported functionality
Decimation filter	• A PF per channel • A high-precision 24-bit result with gain scaling • Facility for DC removal using an HPF • Offset bias compensation logic
Digital protection logic	• SCD • Hi-low limit detection (LMT) • CAD • ZCD

83.3.2 Operations

SINC performs operations based on:

- Modulator bitstream sources. See Bitstream source selection for more information.
- Conversion modes. See Modes and behaviors for more information.
- The selected PF. See PF for more information on SINC PF components, and see Calculating ORD and Equation for OSR that define the calculation for ORD and OSR, respectively, of the PF.

83.3.2.1 Calculating ORD

Use the following table to calculate ORD for the PF.

Table 1122. Calculating ORD

When C*n*DR[PFORD] is:	ORD is:
0	4
> 0	C*n*DR[PFORD]

83.3.2.2 Equation for OSR

$$OSR = C\textit{n}DR[PFOSR] + 1$$

Equation 35. Equation for OSR

83.3.2.3 Bitstream source selection

These are SINC's modulator bitstream sources:

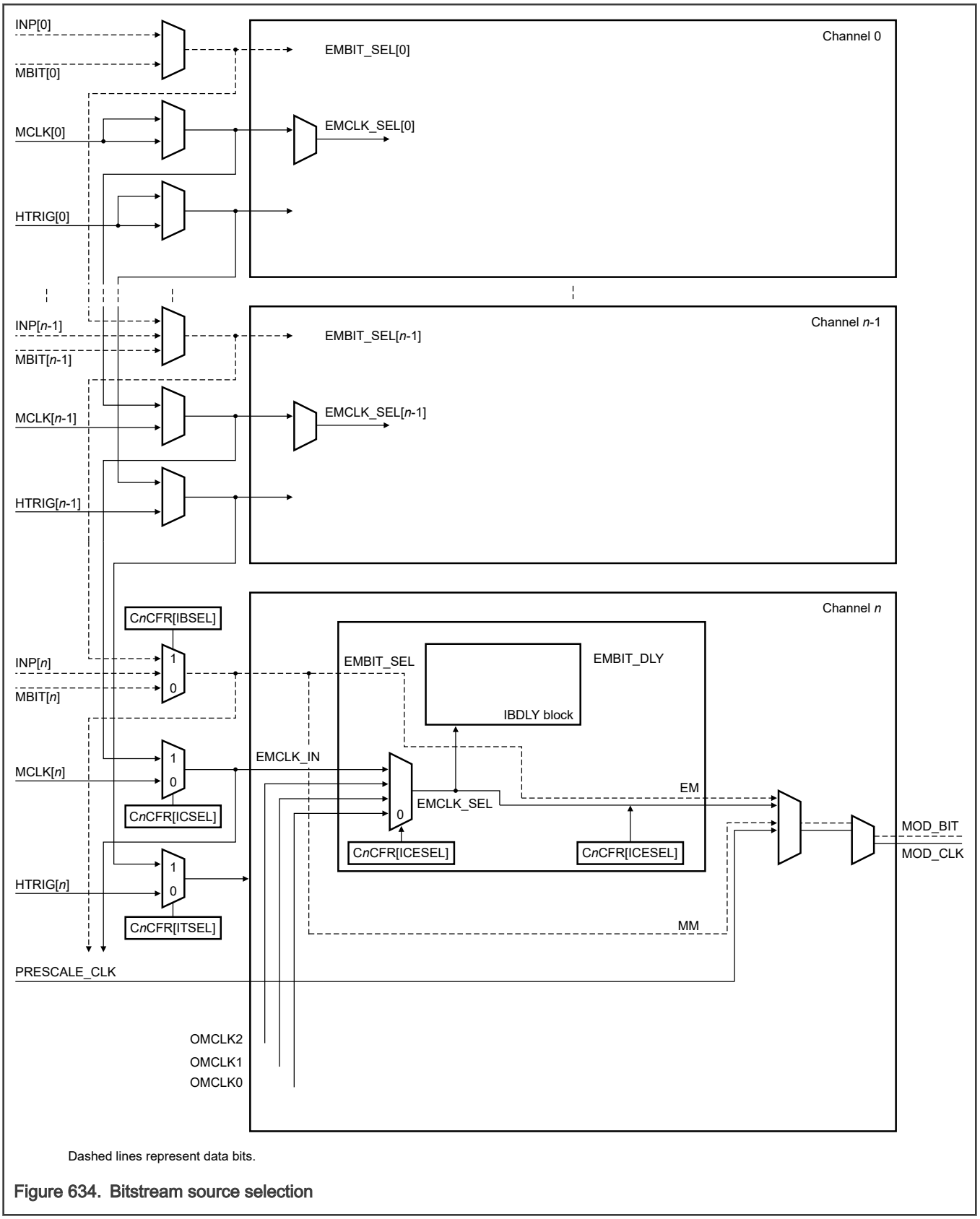
- The MBIT[3:0] signals (see External signals)
- The INP[3:0] signals (see External signals)
- Internal data from registers in Parallel or Serial mode

The following diagram shows MCLK[*n*] and MBIT[*n*] decoder circuits along with these key components:

- External modulator (EM) bitstream (see EM clock input bitstream sampling for EM clock input bitstream sampling)
- Manchester modulator (MM) bitstream (see MM clock input bitstream sampling for MM clock input bitstream sampling)
- Parallel modulator (PM) bitstream (see PM clock input bitstream sampling for PM clock input bitstream sampling)
- Serial modulator (SM) bitstream (see SM clock input bitstream sampling for SM clock input bitstream sampling)

The diagram also illustrates an example of bitstream source selection using:

- An external MCLK[*n*]
- An internal route back clock (OMCLK)
- An internal auto-generated clock for Parallel or Serial mode
- An internal auto-generated clock for Manchester format



Dashed lines represent data bits.

Figure 634. Bitstream source selection

83.3.2.3.1 Modulator bitstream and channel clock selection

Following is the modulator bitstream and clock mux configuration you could perform for each channel.

Table 1123. Modulator bitstream and channel clock selection

Use C*n*CFR[IBSEL] to	Use C*n*CFR[ICSEL] to	Use C*n*CFR[ITSEL] to
Select a modulator input bitstream (EMBIT_SEL) for channel *n* that can be either of the following: • A dedicated external modulator bitstream (MBIT[*n*]) if C*n*CFR[IBSEL] = 00b • A dedicated internal modulator bitstream (INP[*n*]) if C*n*CFR[IBSEL] = 01b • A shared group bitstream from an adjacent channel (*n*-1) if C*n*CFR[IBSEL] = 11b	Select a modulator input clock (EMCLK_SEL) for channel *n* that can be either of the following: • A dedicated modulator clock (MCLK[*n*]) if C*n*CFR[ICSEL] = 011b • A shared group clock (MCLK[*n*-1]) from an adjacent channel if C*n*CFR[ICSEL] = 111b • One of the three internal clocks routed to the chip modulator output clock if C*n*CFR[ICSEL] = 000b, 001b, or 010b	Select a modulator input trigger (HTRIG_SEL) for channel *n* that can be either of the following: • A dedicated software trigger (STRIG[*n*]) if C*n*CFR[ITSEL] = 00b • A dedicated hardware trigger (HTRIG[*n*]) if C*n*CFR[ITSEL] = 01b • A shared group hardware trigger (HTRIG[*n*-1]) from an adjacent channel if C*n*CFR[ITSEL] = 11b

83.3.2.3.2 Cross channel modulator bitstreams and clock groups

Following is the modulator bitstream and clock mux configuration you could perform with an adjacent channel.

Table 1124. Cross channel modulator bitstreams and clock groups

Write 111b to C*n*CFR[IBSEL] for	Write 111b to C*n*CFR[ICSEL] for	Write 11b to C*n*CFR[ITSEL] for
Combining the adjacent channel modulator bitstreams as groups, which: • Share the same modulator bitstreams and clocks if C*n*CFR[ICSEL] = 111b. • Can be configured to use one channel sample by rising edge and another channel sample by falling edge as a microphone use case. • Can be cascaded.	Combining adjacent channels as groups for the same clock to do either of these: • Reduce the I/O pin number. • Retain channels with the same time sequence. • Cascade, possibly.	Combining adjacent channels as groups for the same hardware trigger to: • Retain channels with the same time sequence. • Share the hardware trigger. • Cascade, possibly.

83.3.2.3.3 Modulator bitstream clock edge selection

Following are the modulator bitstream clock and trigger options you could use for each channel.

Table 1125. Modulator bitstream clock edge selection

Use C*n*CFR[ICESEL] to	Use C*n*CFR[ITLVL] to
Select a clock edge and use the selected modulator input clock (EMCLK_SEL) as either of the following:	Select a modulator input trigger event type. The selected modulator input trigger (HTRIG_SEL) can be used as either of the following:

Table continues on the next page...

Table 1125. Modulator bitstream clock edge selection

Use CnCFR[ICESEL] to	Use CnCFR[ITLVL] to
• Positive edge (by writing 001b to CnCFR[ICESEL]) • Negative edge (by writing 010b to CnCFR[ICESEL]) • Dual edge (by writing 011b to CnCFR[ICESEL]) • Every other even positive edge (by writing 101b to CnCFR[ICESEL]) • Every other odd negative edge (by writing 110b to CnCFR[ICESEL]) • Every other odd positive edge (by writing 100b to CnCFR[ICESEL]) • Every other even negative edge (by writing 111b to CnCFR[ICESEL])	• A positive edge trigger (by writing 0b to CnCFR[ITLVL]) • A high-level trigger (by writing 1b to CnCFR[ITLVL])

83.3.2.3.4 Modulator bitstream programmed delay

You can write an appropriate value to CnACFR[IBDLY] for the selected modulator bitstream to be delay sampled by the number of prescaled clock cycles. The delay number can be 0 (bypass), and extend to a maximum of 15 clock cycles.

See the following figure for a pictorial representation of the modulator bitstream programmed delay and see Modulator bitstream delay case 0, Modulator bitstream delay case 1, Modulator bitstream delay case 2, Modulator bitstream delay case 3, and Modulator bitstream delay case 4 for delay cases.

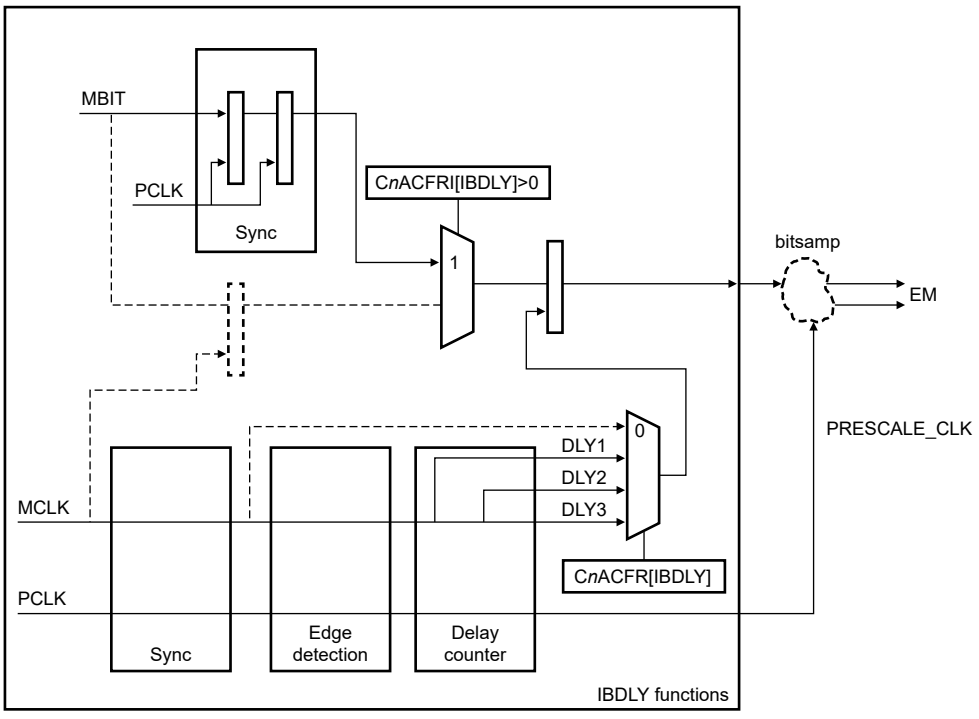


Figure 635. Modulator bitstream programmed delay

83.3.2.3.4.1 Modulator bitstream delay case 0

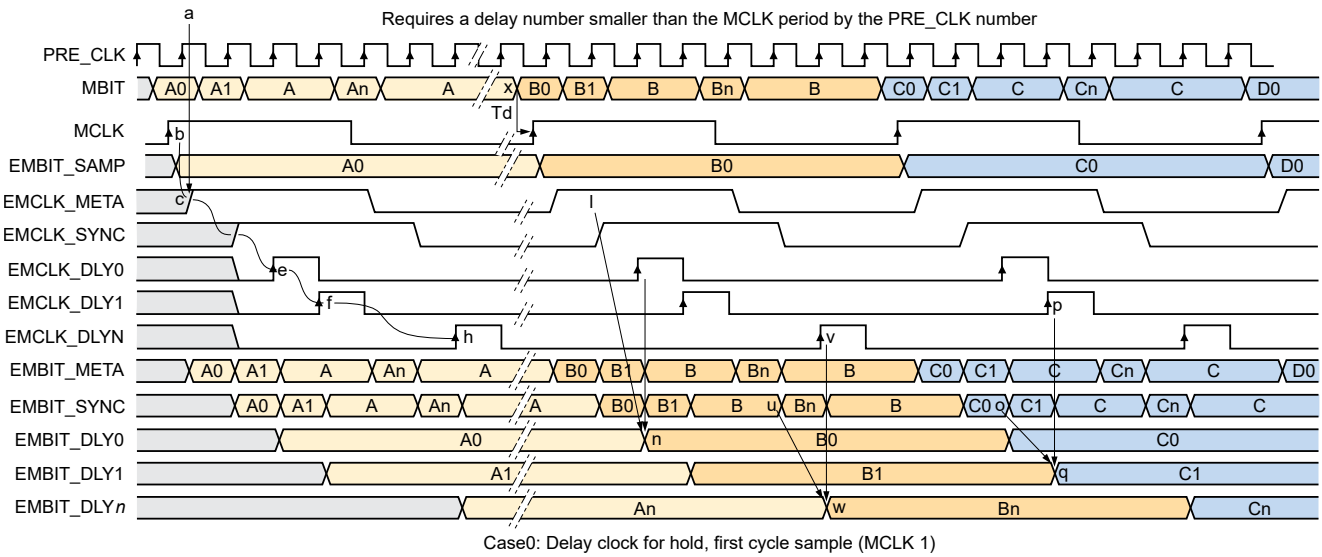


Figure 636. Modulator bitstream delay case 0

83.3.2.3.4.2 Modulator bitstream delay case 1

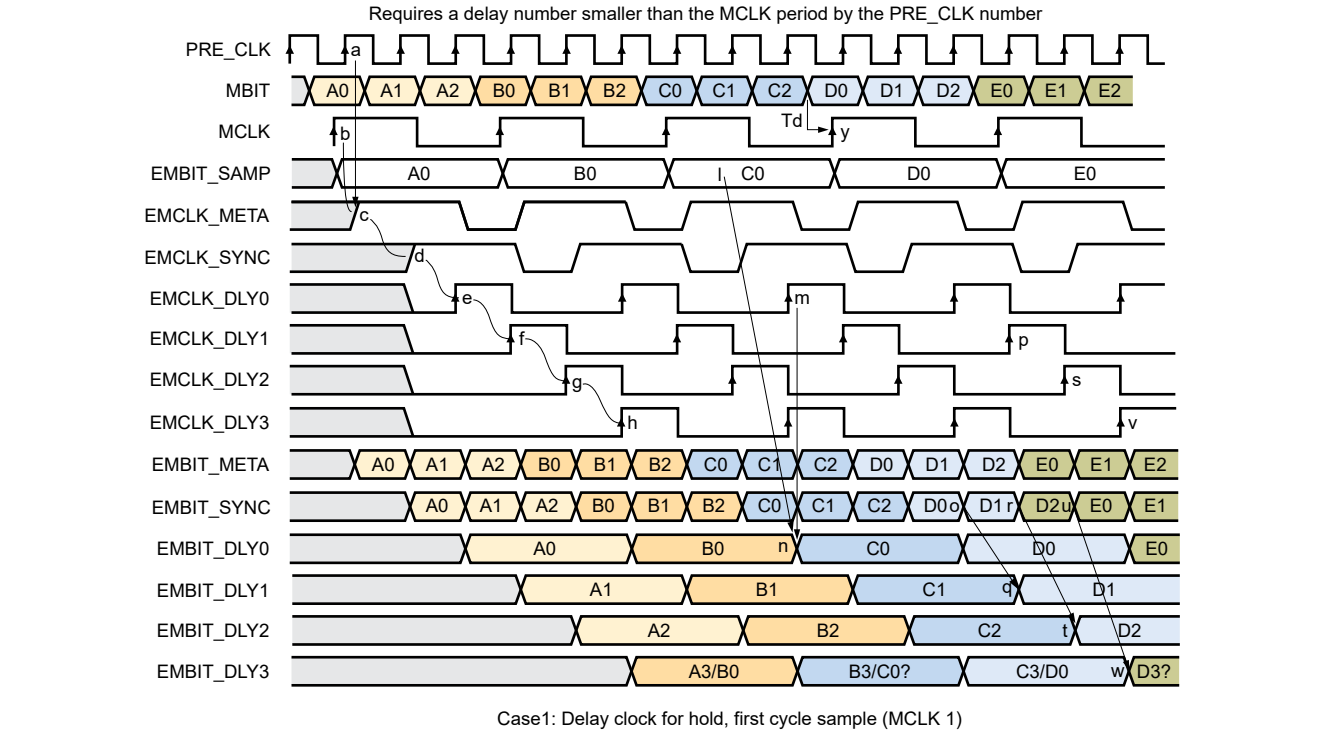


Figure 637. Modulator bitstream delay case 1

83.3.2.3.4.3 Modulator bitstream delay case 2

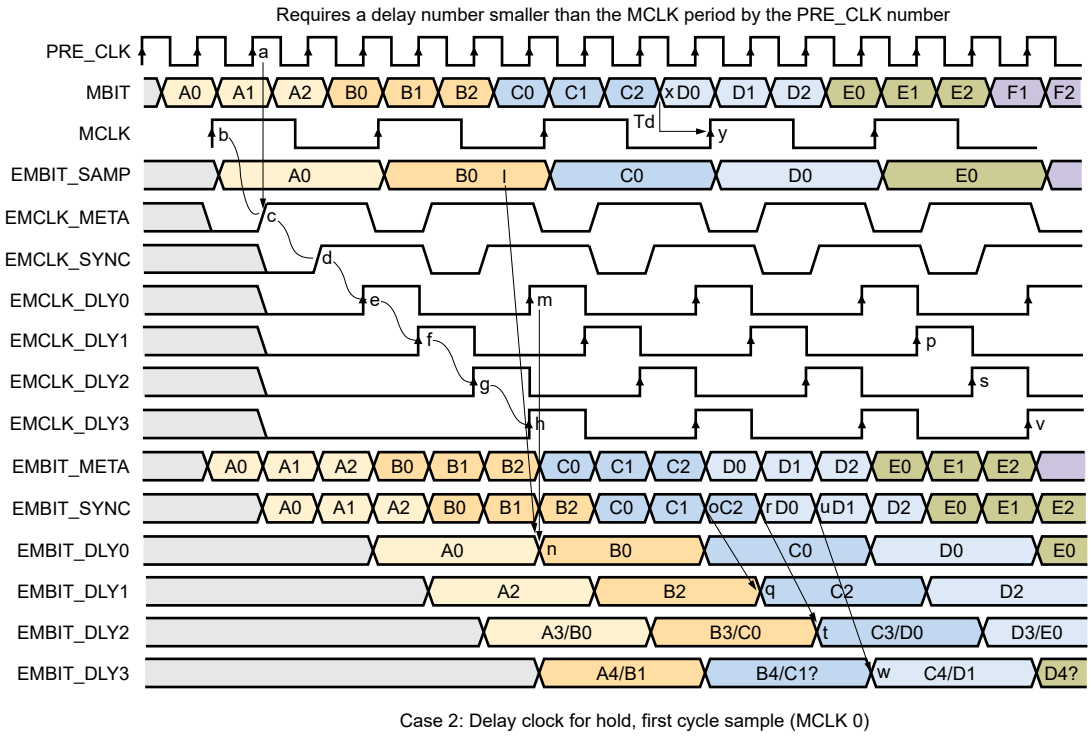


Figure 638. Modulator bitstream delay case 2

83.3.2.3.4.4 Modulator bitstream delay case 3

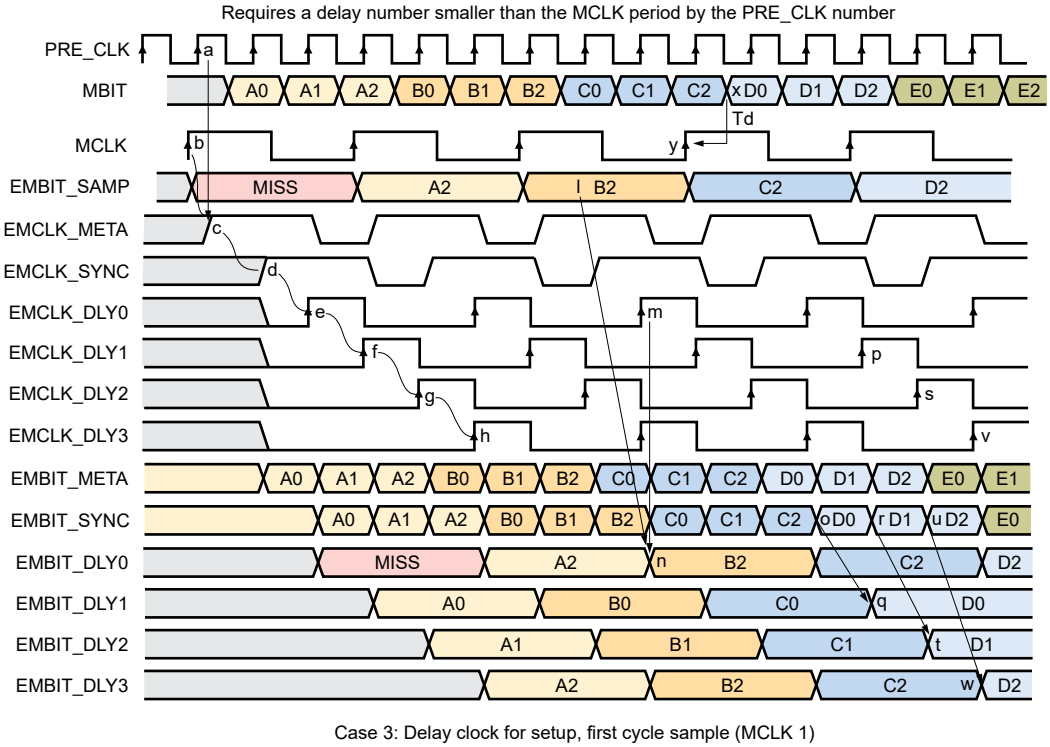


Figure 639. Modulator bitstream delay case 3

83.3.2.3.4.5 Modulator bitstream delay case 4

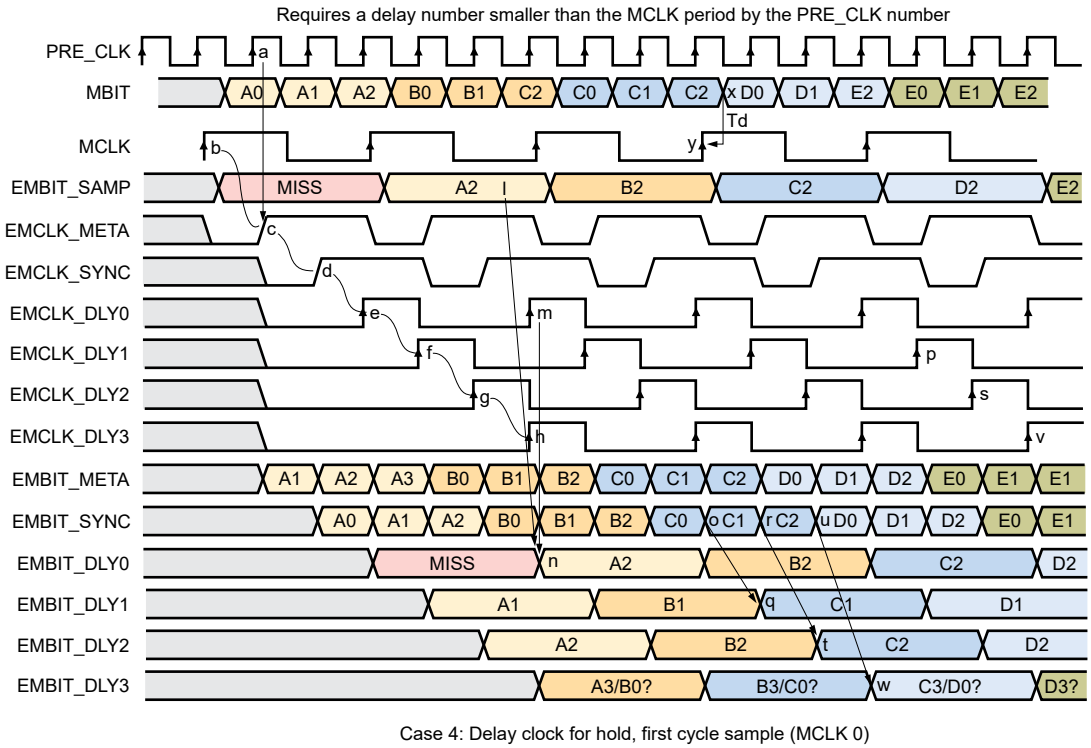


Figure 640. Modulator bitstream delay case 4

83.3.2.3.5 EM clock input bitstream sampling

See EM clock input bitstream sampling: settings for the values you need to write to SINC register fields for EM clock input bitstream sampling and see EM clock input bitstream sampling: diagram for the related bitstream sampling diagram.

83.3.2.3.5.1 EM clock input bitstream sampling: settings

Table 1126. EM clock input bitstream sampling: settings

Action	Result
Writing 00b to C*n*CFR[IBFMT]	Selects the CIC filter input from the selected modulator bitstream input (EMBIT_SEL[*n*]), which the EMCLK_SEL[*n*] clock samples to meet the I/O timing requirement and deliver for further decode.
If you write 011b to C*n*CFR[ICSEL]	The selected external clock (EMCLK_SEL) becomes a dedicated modulator clock, MCLK[*n*], for this channel.
If you write 111b to C*n*CFR[ICSEL]	The selected external clock (EMCLK_SEL) becomes a shared group clock from an adjacent channel.
If you write 000b, 001b, or 010b to C*n*CFR[ICSEL]	The selected external clock (EMCLK_SEL) becomes one of the three internal clocks that are routed as modulator clock outputs.

Table continues on the next page...

Table 1126. EM clock input bitstream sampling: settings (continued)

Action	Result
If you write 00b to CnCFR[IBSEL]	The selected external modulator bitstream (EMBIT_SEL) becomes a dedicated external modulator bitstream (MBIT[n]) for this channel.
If you write 01b to CnCFR[IBSEL]	The selected external modulator bitstream (EMBIT_SEL) becomes a dedicated internal modulator bitstream (inp[n]) for this channel.
If you write 11b to CnCFR[IBSEL]	The selected external modulator bitstream (EMBIT_SEL) becomes a shared group bitstream from an adjacent channel, n-1.
Writing an appropriate value to CnCFR[ICESEL]	Helps you select the type of edge to sample EMBIT_SEL.
Writing an appropriate value to CnACFR[IBDLY]	Helps you select the modulator bitstream delay function.

NOTE

You must configure the prescaler correctly to sample EMCLK_SEL according to CnCFR[ICESEL]. If you configure CnCFR[ICESEL] as a single-edge sample, you must program the prescaler to be 3x faster than EMCLK_SEL; if you configure CnCFR[ICESEL] as a dual-edge sample, you must program the prescaler to be 6x faster than EMCLK_SEL.

See the following for more information:

- Modulator bitstream and channel clock selection
- Modulator bitstream programmed delay
- Bitstream source selection

83.3.2.3.5.2 EM clock input bitstream sampling: diagram

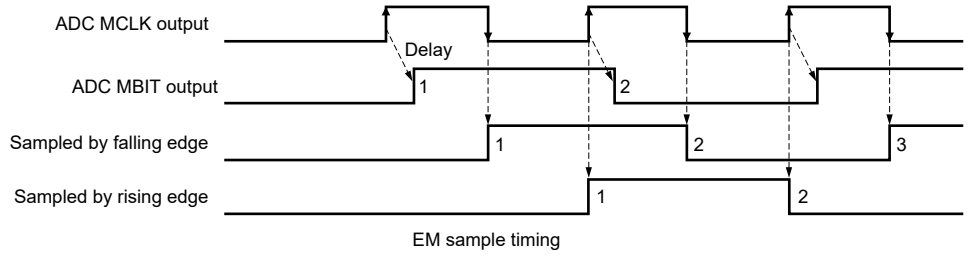


Figure 641. EM clock input bitstream sampling: diagram

83.3.2.3.6 MM clock input bitstream sampling

See MM clock input bitstream sampling: settings for the values you need to write to SINC register fields for MM clock input bitstream sampling and see MM clock input bitstream sampling: diagram for the related bitstream sampling diagram.

83.3.2.3.6.1 MM clock input bitstream sampling: settings

Table 1127. MM clock input bitstream sampling: settings

Action	Result
Writing 01b to C*n*CFR[IBFMT]	Helps you select the input bitstream format as external Manchester code. The modulator clock is recovered from the bitstream.
Writing an appropriate value to C*n*CFR[ICESEL]	Helps you select the input clock edge as rising or falling, with only 001b and 010b are valid values (see MM clock input bitstream sampling: diagram).
Writing 00b to C*n*CFR[IBSEL]	Helps you define the selected external modulator bitstream (EMBIT_SEL) as a dedicated external modulator bitstream (MBIT[*n*]) for this channel.
Writing 01b to C*n*CFR[IBSEL]	Helps you define the selected external modulator bitstream (EMBIT_SEL) as a dedicated internal modulator bitstream (INP[*n*]) for this channel.
Writing 11b to C*n*CFR[IBSEL]	Helps you define the selected external modulator bitstream (EMBIT_SEL) as a shared group bitstream from an adjacent channel, *n*-1.
Writing an appropriate value to C*n*MPDATA	Helps you define the correct threshold for the Manchester decoder. The threshold value must lie between the long pulse width and the short pulse width, and the prescaler clock counts this MBIT[*n*] pulse width. As shown in MM clock input bitstream sampling: diagram, if the previous short pulse is not recovered correctly, the first long pulse is considered as the first valid bit used for whole packet synchronization.

For more on the aforementioned settings, see the following:

- Modulator bitstream and channel clock selection
- Bitstream source selection

83.3.2.3.6.2 MM clock input bitstream sampling: diagram

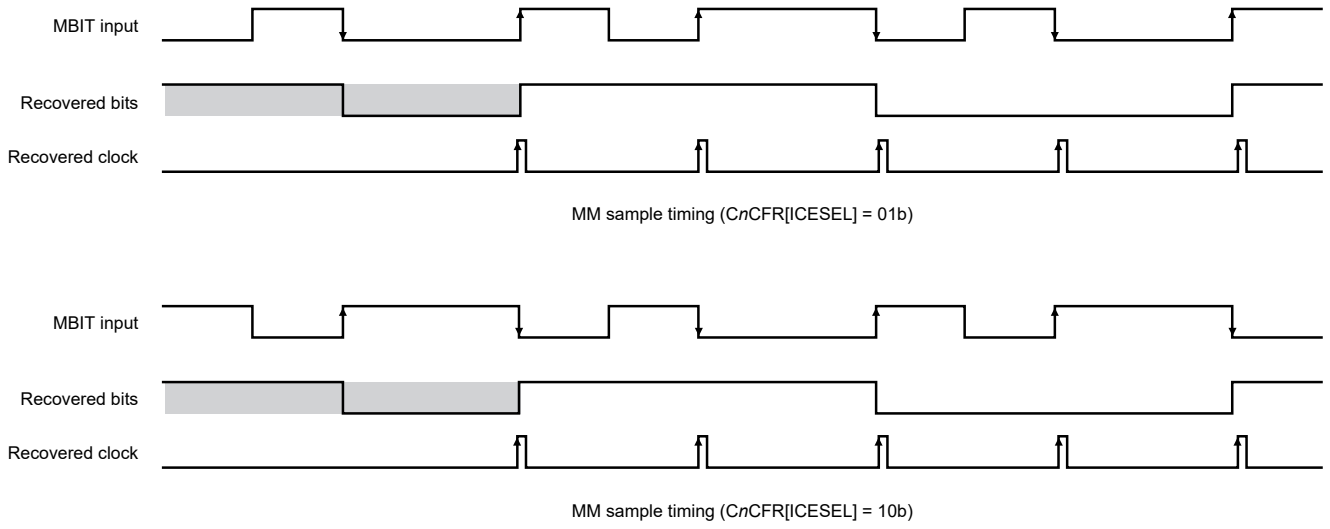


Figure 642. MM clock input bitstream sampling: diagram

83.3.2.3.7 PM clock input bitstream sampling

Table 1128. PM clock input bitstream sampling

Action	Result
Writing 10b to CnCFR[IBFMT]	Selects the CIC filter input from Channel n Multipurpose Data (C0MPDATA - C3MPDATA).
Checking the value of CnSR[PSRDY]	Helps you determine if the next write is allowed. CnSR[PSRDY] = 1 indicates that the previously written data is processed and the field is waiting for the next PM write.
Writing to CnCFR[RDFMT]	Selects PM data as signed or unsigned.
Writing to Channel n Multipurpose Data (C0MPDATA - C3MPDATA)	Sends the lower 16 bits to the CIC filter. The PM clock generates automatically after a write event; IMCLK0 decides the clock rate. If a PM write is not allowed (CnSR[PSRDY] = 0 when CnCFR[IBFMT] = 10), writing to Channel n Multipurpose Data (C0MPDATA - C3MPDATA) returns a bus error.

NOTE

Modulator clocks are automatically generated when you write data to Channel n Multipurpose Data (C0MPDATA - C3MPDATA). Because of conversion latency, you must write an extra two dummy data samples to provide extra clocks after the conversion is complete.

83.3.2.3.8 SM clock input bitstream sampling

See SM clock input bitstream sampling: settings for the values you need to write to SINC register fields for SM clock input bitstream sampling and see SM clock input bitstream sampling: diagram for the related bitstream sampling diagram.

83.3.2.3.8.1 SM clock input bitstream sampling: settings

Table 1129. SM clock input bitstream sampling: settings

Action	Result
Writing 11b to C*n*CFR[IBFMT]	Selects the CIC filter input from Channel n Multipurpose Data (C0MPDATA - C3MPDATA). The entire 32-bit serial data is shifted to the CIC filter. SM clocks are auto generated after a write event, and IMCLK0 decides the clock rate.
Checking the value of C*n*SR[PSRDY]	Determines if the next SM write is allowed. C*n*SR[PSRDY] = 1 indicates that the previously written data is processed and the field is waiting for the next SM write.
Writing to Channel n Multipurpose Data (C0MPDATA - C3MPDATA) for SM	Returns a bus error if an SM write is not allowed (C*n*SR[PSRDY] = 0), and no bitstream remains to be shifted after conversion finishes (C*n*CFR[IBFMT] = 11b).

NOTE

After conversion, the remaining bits are retained in Channel n Multipurpose Data (C0MPDATA - C3MPDATA). To reset them, you could toggle C*n*CR[CHEN] or MCR[RST].

83.3.2.3.8.2 SM clock input bitstream sampling: diagram

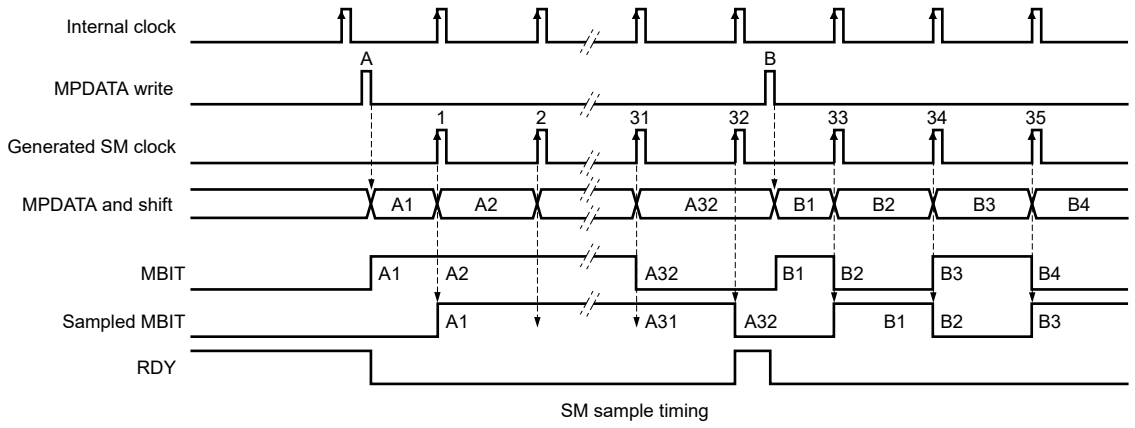


Figure 643. SM clock input bitstream sampling: diagram

83.3.2.4 Conversion process

83.3.2.4.1 Modes and behaviors

After a conversion process triggers:

- The decimation filters are activated.
- NIS[COC*n*] becomes 1 after the conversion completes.

You could use C*n*DR[PFCM] to select one of the conversion modes shown in the following table.

Table 1130. Modes and behaviors

Mode	Behavior of the conversion process in this mode
Single (C*n*DR[PFCM] = 00b)	• Each conversion requires a conversion trigger. • A single trigger starts only one conversion.
Continuous (C*n*DR[PFCM] = 01b)	• After a trigger is detected, the conversion process continues until the next trigger event. • The conversion process also continues after the first sample. You can perform a series of conversions at the rate of one sample per decimation clock cycle, which equals the value of C*n*DR[PFOSR] + 1. For details on subsequent samples, see Total conversion time. • The next trigger cancels the current conversion and restarts it.
Always (C*n*DR[PFCM] = 10b)	• After a trigger is detected, the conversion process continues until it is disabled. • Conversions also continue after the first sample. You can perform a series of conversions at the rate of one sample per decimation clock cycle, which equals the value of C*n*DR[PFOSR] + 1. For details on subsequent samples, see Total conversion time. • The next trigger event is ignored.
Fixed-Number (C*n*DR[PFCM] = 11b)	• After a trigger is detected, the conversion process continues until the required conversion number is achieved or until the next trigger event. • The conversion process stops when NIS[CHF*n*] becomes 1, so you can configure C*n*CFR[FIFOWMK] to determine the number of needed conversions. • The conversion process also continues after the first sample. You can perform a series of conversions at the rate of one sample per decimation clock cycle, which equals the value of C*n*DR[PFOSR] + 1. For details on subsequent samples, see Total conversion time. • The next trigger cancels the current conversion and restarts it.

83.3.2.4.2 Completion

Following are some of the events that take place after a conversion completes:

- The new sample value is transferred to the FIFO, and you can get the FIFO content by reading Channel n Result Data (C0RDATA - C3RDATA).
- NIS[COC*n*] becomes 1 and the FIFO data is available.
- NIS[CHF*n*] becomes 1 if the FIFO from the PF channel *n* has overflowed its watermark level and the FIFO data is available.
- A conversion complete and CHF interrupt are asserted if NIE[COCIE*n*] = 1 or NIE[CHFIE*n*] = 1.
- A DMA request and CHF are asserted if C*n*CR[DMAEN] = 1.

Additionally, you can read the current conversion data directly from Channel n Result Data (C0RDATA - C3RDATA) if FIFO is disabled.

83.3.2.4.3 Discontinuation

Conversion involves the following steps:

- CIC calculation
- Shift
- DC removal
- Bias
- Push FIFO and interrupt

You could abort a conversion by using any of these techniques:

- Disable
- Next Trigger
- Stop

This is what you could do to abort conversions, by using different register fields, modes, and triggers:

- To abort a conversion in progress, disable the PF by writing 0 to C*n*CR[PFEN]. However, in case you cancel the CIC filter by using C*n*CR[PFEN], the SCD and CAD blocks can still keep working.
- To abort a conversion along with its subfunctions (PF, CAD, and SCD), disable the channel by writing 0 to C*n*CR[CHEN].
- To abort or disable all the conversions globally, write 0 to MCR[MEN]. The FIFO retains the values of the previous conversion result that had not been read.
- In Single and Continuous conversion modes, if you select an edge trigger, a new trigger aborts the current conversion and restarts a new one.
- If you select a level trigger, a deasserted software or hardware trigger cancels the current conversion.
- In Stop mode, the current conversion is canceled.

If an abort occurs after the CIC filter has performed its function, you could:

- Use the Disable or Stop technique to cancel the current conversion immediately.
- Use the Next Trigger technique for the current conversion to continue the shift and bias operations and save the result to the FIFO. At the same time, the CIC can process the next conversion.

NOTE

For a motor use case, you must calculate the maximum speed (trigger frequency) to ensure that the required conversion number and the next trigger do not overlap. The margin must be at least two modulator clocks.

83.3.2.5 Conversion timing

83.3.2.5.1 Total conversion time

Table 1131. Total conversion time

In this mode:	The conversion time is:
Single	The value derived using this formula for a single conversion after the trigger event is over: ORD × OSR × MOD_CLK period
Continuous Always Fixed-Number	The value derived using this formula for the first 24-bit sample after the trigger event is over: ORD × OSR × MOD_CLK period However, the subsequent 24-bit samples are available after the OSR × MOD_CLK period.

where MOD_CLK period = 1 ÷ the frequency of MOD_CLK.

During a conversion, if you opt to resynchronize the conversion by providing a trigger, the first conversion result is available after the ORD × OSR × MOD_CLK period.

However, if you opt for a run-time phase compensation for a channel, SINC does not support it. You must provide a hardware trigger delay (for example, a PDB block) at the chip level. SINC does not support a software trigger delay.

83.3.2.5.2 Trigger alignment

When a trigger asserts, the recovered modulator clock samples the trigger event (see MOD_CLK in Bitstream source selection). At the same time, the modulator clock samples the MBIT[*n*] signal (see External signals). Therefore, the first valid bitstream to CIC is the clock cycle during which the trigger is asserted.

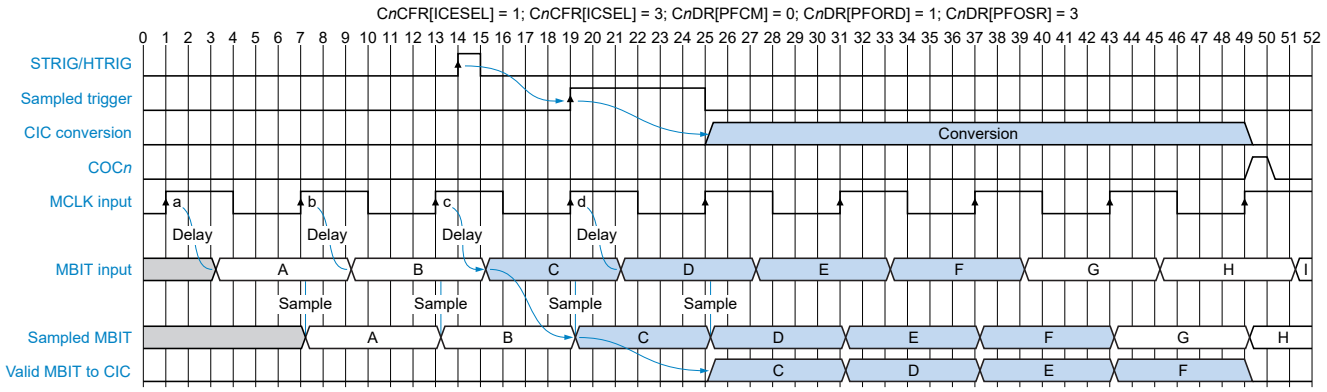


Figure 644. Trigger alignment

83.3.2.6 Trigger timing

83.3.2.6.1 Single mode software edge trigger

As illustrated in the following figure, when a software edge trigger event occurs in Single mode:

- Triggers a and b are ignored because SR[CHRDY*n*] = 0 (not ready for conversion).
- Trigger c starts conversion C1; trigger d starts conversion D1.
- Trigger e starts conversion e1 and trigger f cancels it before the conversion is complete. Simultaneously, trigger f starts (or resynchronizes) conversion F1.
- Trigger g starts conversion g1 but deassertion of the "Enable" signal cancels it.

See Total conversion time that defines the formula for Single mode conversion time.

NOTE

- The "Enable" signal is a combined AND logic of C*n*CR[PFEN] and SR[CHRDY*n*].
- You write 1 to MCR[STRIG*n*] and SINC writes 0 to this field for a software edge trigger case.

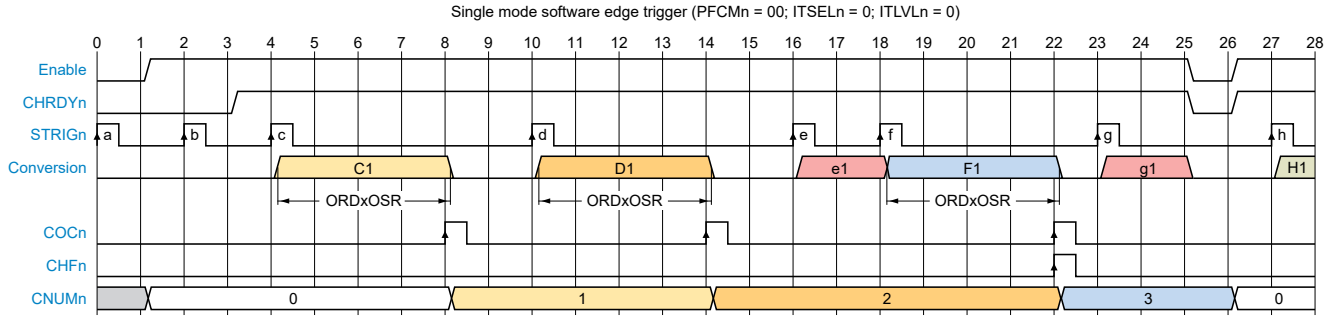


Figure 645. Single mode software edge trigger

83.3.2.6.2 Single mode hardware edge trigger

As illustrated in the following figure, when a hardware edge trigger event occurs in Single conversion mode:

- Triggers a and b are ignored because SR[CHRDY*n*] = 0 (not ready for conversion).
- Trigger c starts conversion C1; trigger d starts conversion D1.
- Trigger e starts conversion e1, but trigger f cancels it before completion. Simultaneously, trigger f starts (or resynchronizes) conversion F1.
- Trigger g starts conversion g1 but deassertion of the "Enable" signal cancels it.

The external hardware both asserts and deasserts the HTRIG[*n*] signal (see External signals); however, only the rising edge of the signal is accepted as the startup event for conversion. For example, trigger d exists several times, but the fall of trigger g does not cancel conversion D1.

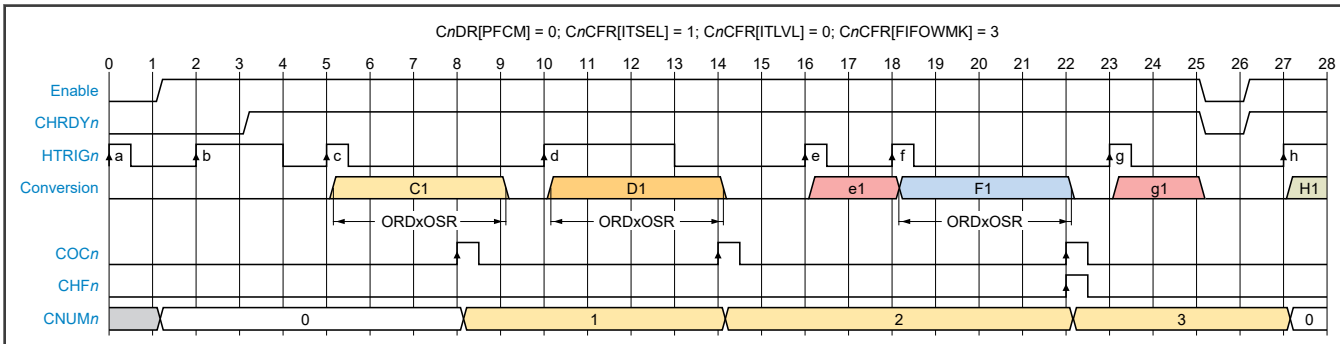


Figure 646. Single mode hardware edge trigger

83.3.2.6.3 Single mode hardware level trigger

As illustrated in the following figure, when a hardware level trigger event occurs in Single mode:

- Trigger a is ignored because SR[CHRDY*n*] = 0 (not ready for conversion).
- Trigger b starts conversion B1; trigger c starts conversion C1.
- Trigger d starts conversion d1 but the conversion cancels because trigger d deasserts before conversion completion.
- Trigger f starts conversion f1 but the conversion cancels because the "Enable" signal deasserts.

An external hardware asserts and deasserts HTRIG[*n*]. The rising of HTRIG[*n*] starts the conversion and the falling of HTRIG[*n*] cancels the conversion before its completion.

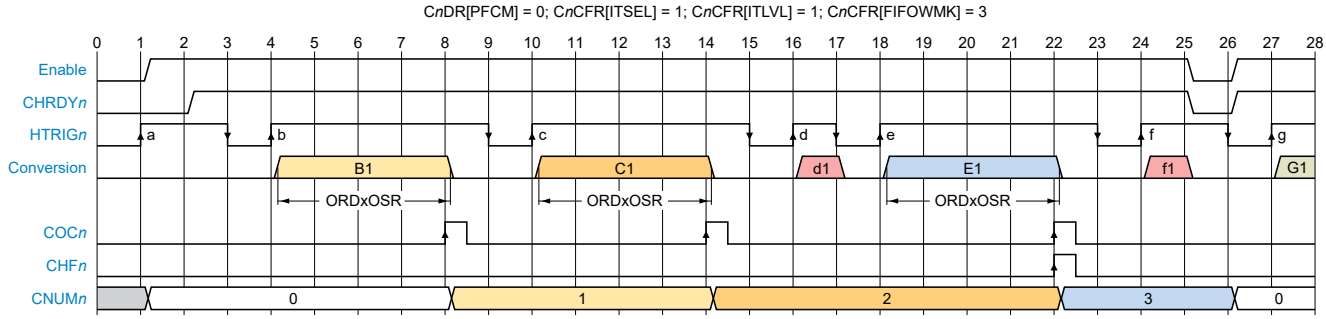


Figure 647. Single mode hardware level trigger

83.3.2.6.4 Continuous mode software edge trigger

In Continuous mode, conversions comprise a set of subconversions and continue until aborted by another event. For example, as shown in the following figure, conversion C comprises subconversions C1, C2, C3, c4, and so on; and conversion D comprises subconversions D1, D2, d3, and so on. When a software edge trigger event occurs in Continuous conversion mode:

- Triggers a and b are ignored because SR[CHRDY*n*] = 0 (not ready for conversion).
- Trigger c starts conversion C1 and trigger d starts conversion D1.
- The output delay time for C1 is defined as OSR×ORD. When conversion C1 comes to an end, conversion C2 starts automatically.
- The output delay time for C2 is defined as OSR, and so is the output delay time for conversion C3.
- Trigger d cancels conversion c4 and simultaneously resynchronizes conversion D1.
- Writing 0 to C*n*CR[PFEN] cancels conversion d3.

You write 1 to MCR[STRIG*n*] and SINC writes 0 to this field for a software edge trigger event in Continuous mode.

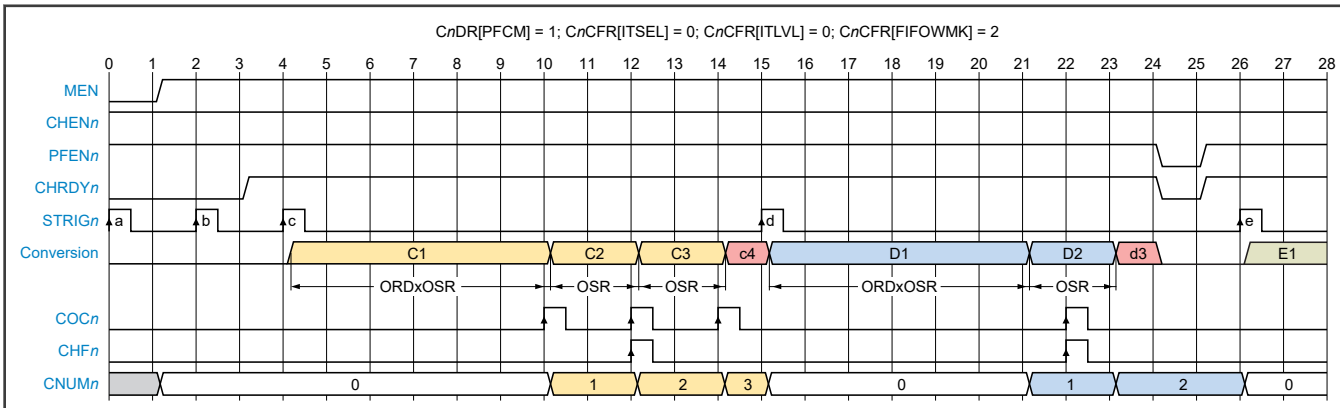


Figure 648. Continuous mode software edge trigger

83.3.2.6.5 Continuous mode hardware edge trigger

As illustrated in the following figure, when a hardware edge trigger event occurs in Continuous mode:

- Triggers a and b are ignored because SR[CHRDY*n*] = 0 (not ready for conversion).
- Trigger c starts conversion C1, and hardware trigger d starts conversion D1.
- The output delay time for C1 is defined by OSR×ORD. After conversion C1 comes to an end, conversion C2 starts automatically.

- The output delay time for C2 is defined as OSR, and so is the output delay time for C3.
- Trigger d cancels conversion c4 and simultaneously resynchronizes conversion D1.
- Writing 0 to C*n*CR[PFEN] cancels conversion d4.

The external hardware asserts and deasserts HTRIG[*n*]. However, only the rising of HTRIG[*n*] starts the conversion. For example, the hardware trigger d exists several times, but the fall of trigger d does not cancel conversion D1. Hardware trigger b asserts before SR[CHRDY*n*] becomes 1 and remains asserted after SR[CHRDY*n*] becomes 1, but no conversion starts.

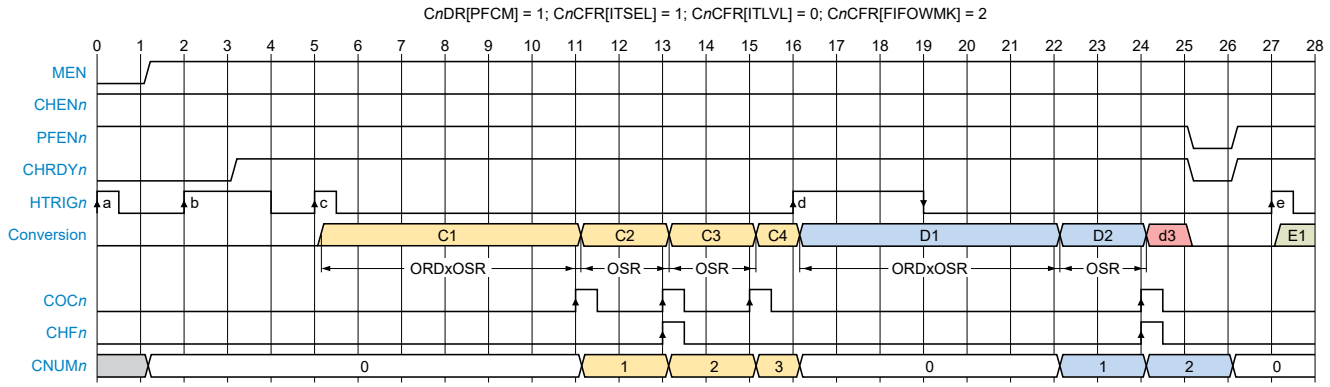


Figure 649. Continuous mode hardware edge trigger

83.3.2.6.6 Continuous mode software level trigger

As illustrated in the following figure, when a software level trigger event occurs in Continuous mode:

- Triggers a and b are ignored because SR[CHRDY*n*] = 0 (not ready for conversion).
- Trigger c starts conversion C1; trigger d starts conversion D1.
- The output delay time for C1 is defined as OSR×ORD. After conversion C1 comes to an end, conversion C2 starts automatically.
- The output delay time for C2 is defined as OSR, and so is the output delay time for C3.
- Trigger c deassertion cancels conversion c4.
- Signal "Enable" deassertion cancels conversion d3.

For a software level trigger event in Continuous mode, you can write both 1 (to start the conversion process) and 0 (to stop the conversion process) to MCR[STRIG*n*] if SR[CHRDY*n*] is 1. SINC can also write 0 to MCR[STRIG*n*] to cancel the conversion. The following example shows writing 0 to MCR[STRIG*n*] cancels conversion c4.

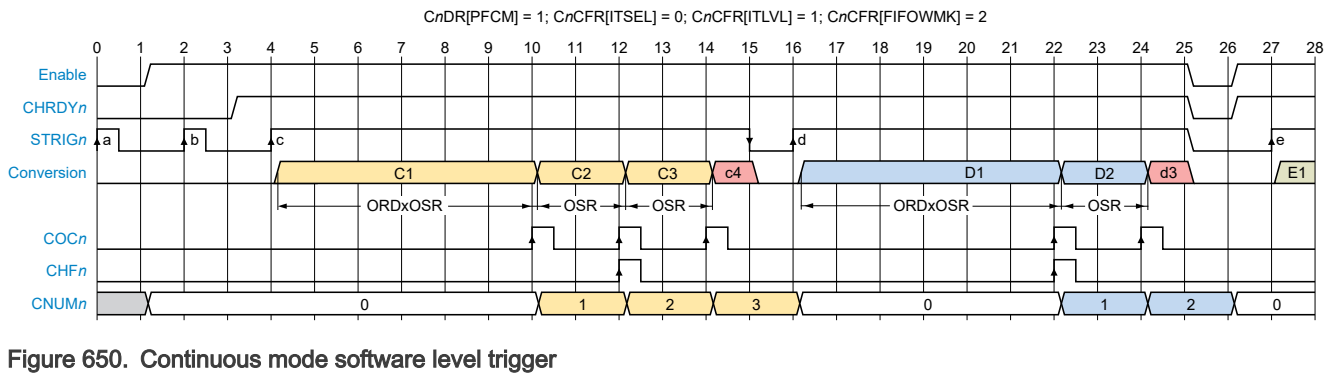


Figure 650. Continuous mode software level trigger

83.3.2.6.7 Continuous mode hardware level trigger

As illustrated in the following figure, when a hardware level trigger event occurs in Continuous mode:

- Triggers a and b are ignored because SR[CHRDY*n*] = 0 (not ready for conversion).
- Trigger c starts conversion C1; and trigger d starts conversion D1.
- The output delay time for C1 is defined as OSR×ORD. When conversion C1 comes to an end, conversion C2 starts automatically.
- The output delay time for C2 is defined as OSR, and so is the output delay time for conversion C3.
- Trigger c deassertion cancels conversion c4.
- Signal "Enable" deassertion cancels conversion d3.

An external hardware asserts and deasserts HTRIG[*n*]. It asserts HTRIG[*n*] to start the conversion process and deasserts it to stop the conversion process. The following example shows that HTRIG[*n*] deassertion cancels conversion c4.

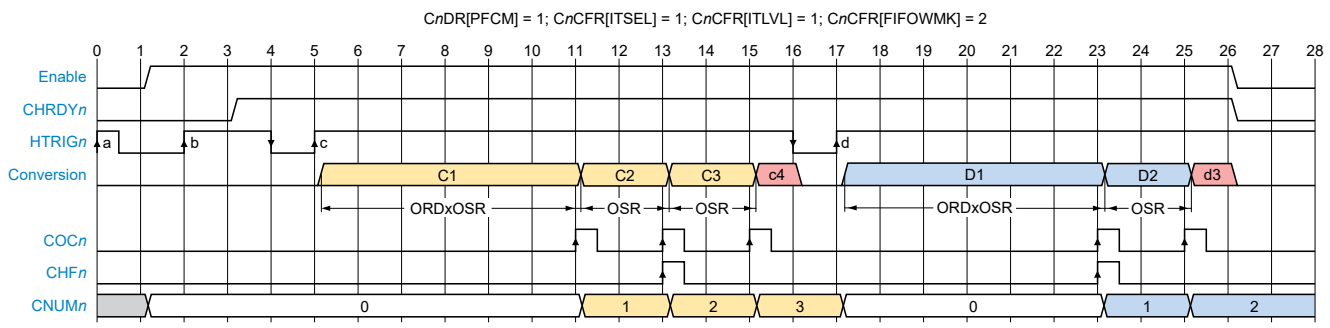


Figure 651. Continuous mode hardware level trigger

83.3.2.6.8 Always mode hardware edge trigger

As illustrated in the following figure, when a hardware edge trigger event occurs in Always mode:

- Triggers a and b are ignored because SR[CHRDY*n*] = 0 (not ready for conversion).
- Trigger c starts conversion C1; trigger e starts conversion D1.
- The output delay time for C1 is defined as OSR×ORD. After conversion C1 comes to an end, conversion C2 starts automatically.
- The output delay time for C2 is defined as OSR, and so is the output delay time for conversion C3.
- Trigger d is ignored.

After the conversion process starts in this mode, the only way to stop it is by writing 0 to MCR[MEN], C*n*CR[CHEN], or C*n*CR[PFEN].

An external hardware both asserts and deasserts HTRIG[*n*].

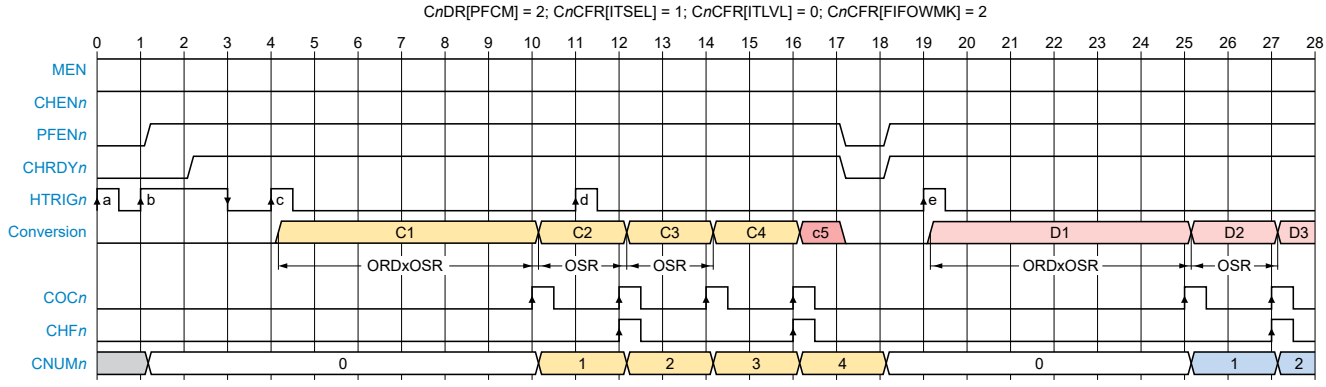


Figure 652. Always mode hardware edge trigger

83.3.2.7 PF

SINC's PF comprises the following:

- A CIC filter
- An optional HPF for DC removal operations
- A shift
- An optional bias

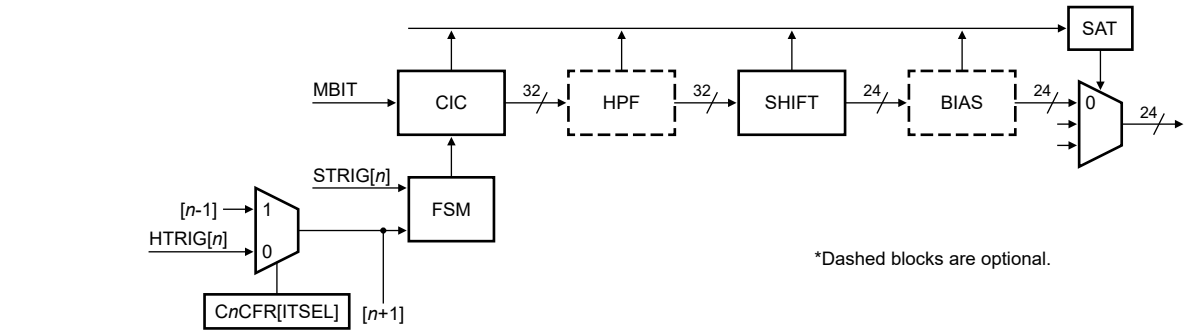


Figure 653. PF

83.3.2.7.1 CIC filter

CIC filters:

- Consist of one or more integrator and comb filter pairs.
- Represent an optimized implementation of low-pass filters (LPFs) with programmable, multiple stages.
- Are widely used because they can be implemented using just adders and subtractors.
- Output raw 32-bit data and deliver it to the HPF (or to the PF shift if HPF is not present).
- Select a CIC filter input, decode modulator bitstream, and decode modulator clock from the recovery block.

The following figure shows the architecture of a three-stage CIC filter, comprising integrator sections followed by comb sections downsampled.

See CnDR[PFORD] for information on PF order and CnDR[PFOSR] for information on PF OSR.

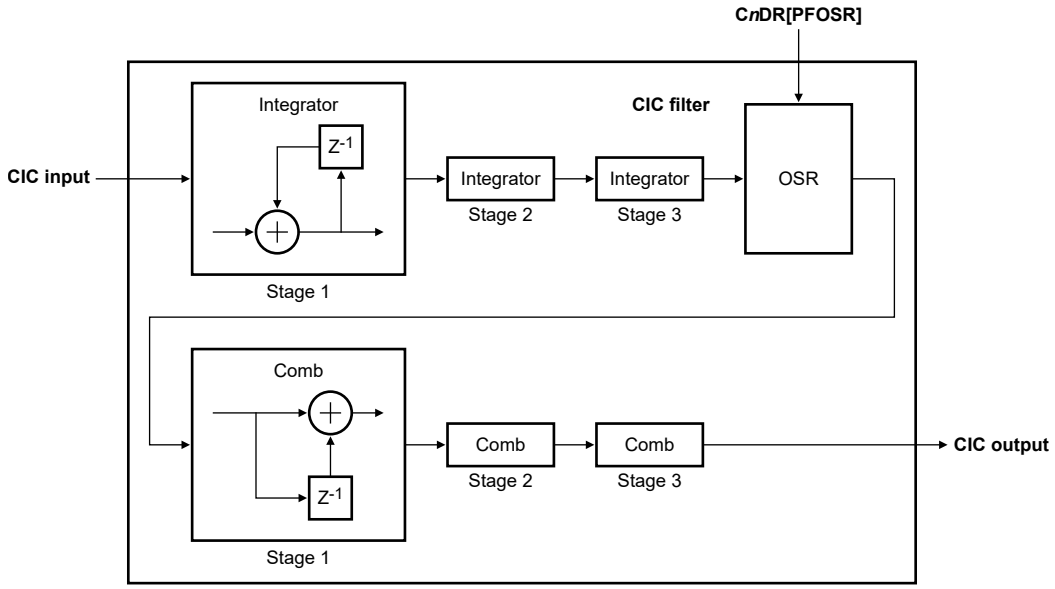


Figure 654. CIC filter

83.3.2.7.1.1 CIC filter transfer function

You can calculate the transfer function of the CIC filter, when CnDR[PFORD] is any value except 0, based on the following equation.

$$H(Z) = \frac{Y(Z)}{X(Z)} = \left[\frac{1-Z^{-OSR}}{1-Z^{-1}}\right]^{ORD}$$

Equation 36. CIC filter transfer function

You can see an example of Frequency Response in Frequency response

You can get CIC output effective data width by below equation. Where OSR are described at Equation for OSR.

when CnCFR[RDFMT]=0b(signed):

WDITH = log2(OSR^^CnDR[PFORD]+1)+1

when CnCFR[RDFMT]=1b(unsigned):

WDITH = log2(OSR^^CnDR[PFORD]+1)

Equation 37. Equation for CIC effective data width

83.3.2.7.1.2 FastSinc filter transfer function

You can calculate the transfer function of the FastSinc filter, when CnDR[PFORD] = 0, based on the following equation.

$$H(Z) = \frac{Y(Z)}{X(Z)} = \left[\frac{1-Z^{-OSR}}{1-Z^{-1}}\right]^{2} * \left[1+Z^{-2*OSR}\right]$$

Equation 38. FastSinc filter transfer function

You can get CIC output effective data width by below equation. Where OSR are described at Equation for OSR

when CnCFR[RDFMT]=0b(signed):

WDITH = log2(OSR^^2*2+1)+1

when C*n*CFR[RDFMT]=1b(unsigned):

WDITH = log2(OSR^^2*2+1)

Equation 39. Equation for CIC effective data width

83.3.2.7.1.3 CIC effective data width

You can see an example of Frequency Response in Frequency response

You can get example CIC output effective data width by Table 1132. Where OSR are described at Equation for OSR

Table 1132. data width vs OSR and ORD

OSR	C*n*CFR[RDFMT]=0b(signed)				C*n*CFR[RDFMT]=1b(unsigned)			
	PFORD=1	PFORD=2	PFORD=0 (fastsinc)	PFORD=3	PFORD=1	PFORD=2	PFORD=0 (fastsinc)	PFORD=3
16	6	10	11	14	5	9	10	13
32	7	12	13	17	6	11	12	16
64	8	14	15	20	7	13	14	19
128	9	16	17	23	8	15	16	22
256	10	18	19	26	9	17	18	25
1291	12	22	23	33(not support)	11	21	22	32
1626	12	23	24	34(not support)	11	22	23	33(not support)

83.3.2.7.2 Trigger events

The following table shows trigger cases based on the configuration of PF conversion modes (C*n*DR[PFCM]), input trigger selection (C*n*CFR[ITSEL]), and input trigger level type (C*n*CFR[ITLVL]).

NOTE

C*n*CFR[ITSEL] = 11b indicates that a grouped trigger shared with an adjacent channel behaves in the same way as a hardware trigger (01b).

Table 1133. Trigger events

C*n*DR[PFCM]	C*n*CFR[ITSEL]	C*n*CFR[ITLVL]	Case	Functioning
00	00	0	Single mode software edge trigger	• Writing 1 to MCR[STRIG*n*] starts the conversion process. In case of an ongoing conversion, writing 1 to this field cancels it and starts a new conversion. • MCR[STRIG*n*] automatically becomes 0 after the conversion starts.

Table continues on the next page...

Table 1133. Trigger events (continued)

CnDR[PFCM]	CnCFR[ITSEL]	CnCFR[ITLVL]	Case	Functioning
	00	1	Single mode software level trigger	Reserved
	01	0	Single mode hardware edge trigger	• An external hardware trigger rising edge starts the conversion process. • The next trigger during conversion cancels the current conversion and restarts a new one.
	01	1	Single mode hardware level trigger	• An external hardware trigger rising edge starts the conversion process. • The hardware trigger deasserts during conversion and cancels the current conversion.
01	00	0	Continuous mode software edge trigger	• Writing 1 to MCR[STRIGn] starts the conversion process, which continues unless the filter is disabled. In case of an ongoing conversion, writing 1 to this field cancels it and starts a new conversion. • MCR[STRIGn] automatically becomes 0 after the conversion starts.
	00	1	Continuous mode software level trigger	• Writing from 0 to 1 to MCR[STRIGn] starts the conversion process, which continues unless the channel filter is disabled. • Writing 0 to MCR[STRIGn] cancels the current conversion.
	01	0	Continuous mode hardware edge trigger	• An external hardware trigger rising edge starts the conversion process, which continues unless the channel filter is disabled. • The next trigger cancels the current conversion and restarts a new one.
	01	1	Continuous mode hardware level trigger	• An external hardware trigger rising edge starts the conversion process, which continues unless the channel filter is disabled. • The hardware trigger deasserts and cancels the current conversion.

Table continues on the next page...

Table 1133. Trigger events (continued)

CnDR[PFCM]	CnCFR[ITSEL]	CnCFR[ITLVL]	Case	Functioning
10	00	0	Always mode software edge trigger	Reserved
	00	1	Always mode software level trigger	Reserved
	01	0	Always mode hardware edge trigger	• An external hardware trigger rising edge starts the conversion process, which continues unless the channel filter is disabled. • The next trigger is ignored.
	01	1	Always mode hardware level trigger	Reserved
11	Any value	1	Always mode hardware level trigger	• Trigger behavior is the same as in Continuous mode for CnCFR[ITSEL] and CnCFR[ITLVL]. • The conversion process automatically stops when the FIFO watermark is asserted.

83.3.2.7.3 HPF transfer function for DC removal

An HPF performs the DC removal function. The CIC filter provides an input to the HPF and outputs 32-bit data, which is sent to the PF shift.

CnACFR[HPFA] specifies the HPF alpha coefficient or disables the HPF when it is 0.

The HPF restarts if CnCR[CHEN] changes from 0 to 1 (in case of a rise event) and the following transfer function specifies the HPF frequency response.

$$H(Z) = \frac{Y(Z)}{X(Z)} = \frac{1\text{-}Z^{-1}}{1\text{-}aZ^{-1}}$$

Equation 40. HPF transfer function for DC removal

NOTE

For DC removal, you can perform HPF gain compensation and use only the signed data format.

83.3.2.7.4 Shift

The CIC filter outputs raw 32-bit data. The higher part of that data is most likely to consist of zeros or sign bits. You must shift this data to attain 24-bit alignment.

If the raw CIC data contains more than 24 bits, then you must shift it right by CnCFR[PFSFT] to fill bits 23–0, and copy the result to data bits 31–8.

If the raw CIC data contains fewer than 24 bits, then you must shift it left by CnCFR[PFSFT] to fill bits 23–0, and copy the result to data bits 31–8.

For right shift, SINC rounds the result up (as if adding 0.5×LSB) before truncation.

Rounding is not necessary for a left shift. If the shift and rounding result is more than 24 bits, the result is saturated and SINC changes the corresponding saturation status field to 1.

83.3.2.7.4.1 Shift right block diagram

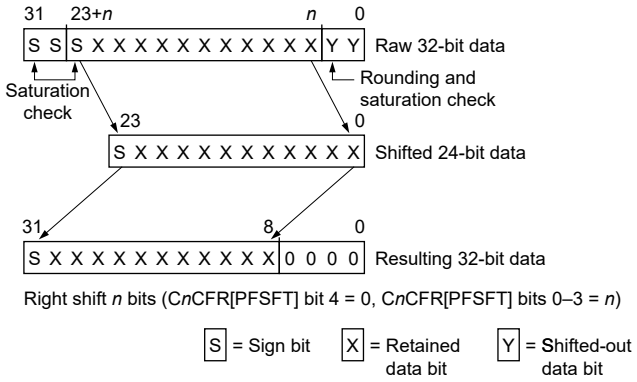


Figure 655. Shift right block diagram

83.3.2.7.4.2 Shift left block diagram

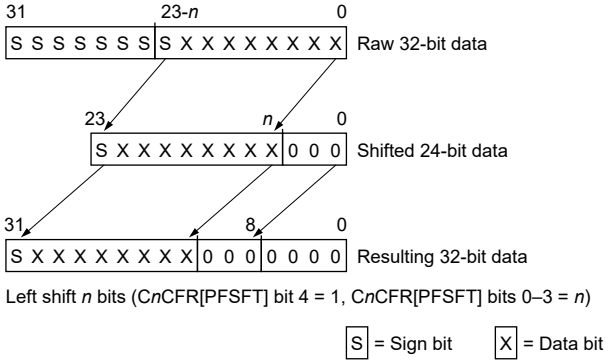


Figure 656. Shift left block diagram

83.3.2.7.5 Bias setting

Channel n Bias (C0BIAS - C3BIAS) specifies a bias offset for PF output, derives input from PF shift, and outputs the final result or value to FIFO and other compare blocks. The valid value is represented in the two's-complement format and the PF subtracts this bias value from the data.

83.3.2.8 SCD operations

An SCD is applied to each channel. Its purpose is to quickly respond if an analog signal is close to its maximum threshold value. The SCD can detect both short-circuit (overcurrent) and open-circuit (overvoltage) errors.

NOTE

An SCD does not support parallel modulator inputs.

Table 1134. SCD operations

Using	You can
C*n*CR[SCDEN]	Write 0 to disable the SCD.
EIS[SCD*n*]	Check whether SINC detected a short circuit on filter channel *n*.
Channel n Debug (C0DBGR - C3DBGR) with an appropriate value of C*n*CR[DBGSEL]	Check SCD status anytime. When an SCD event occurs, the SCD continues to count for the same field until a bit toggle occurs and holds the counter for you to read.
EIE[SCDIEn]	Enable and disable the Short Circuit Detected interrupt for filter channel *n*.
C*n*PROT[SCDBK]	Enable and disable the automatic assertion of the BREAK_SCD signal (see External signals).
C*n*PROT[SCDLMT]	Specify the threshold value of the SCD counter. There exists an upcounting counter on each input channel that counts consecutive 0s or 1s on the modulator bitstream. The counter restarts in case of a bit toggle. If this counter reaches a threshold value (C*n*PROT[SCDLMT]), a short-circuit event is invoked.
C*n*PROT[SCDOP]	Program conversion options and control detection of the bit value (0, 1, or both) that increments the SCD counter.
C*n*PROT[SCDCM]	Program conversion modes and control when the SCD operates—when C*n*CR[CHEN] = 1 or when the PF is performing a conversion.

83.3.2.9 CAD operations

A CAD is applied to each channel. You must check the modulator clock input for absence to ensure that the correct conversion process and error detection are in progress.

NOTE

When a clock absence event occurs, the data conversion process and detectors provide incorrect data. You must manage this event and discard the given data when a clock absence is reported.

Table 1135. CAD operations

Using field	You can
C*n*CR[CADEN]	Write 0 to disable CAD.
FIFOIS[CAD*n*]	Check whether SINC detected the absence of a clock on filter channel *n*. When a CAD event occurs, CAD continues to count until an overflow occurs and then latches the value until you clear the flag.
FIFOIE[CADIE*n*]	Enable and disable the Clock Absence interrupt for filter channel *n*.

Table continues on the next page...

Table 1135. CAD operations (continued)

Using field	You can
C*n*PROT[CADBK]	Enable and disable the automatic assertion of the BREAK_CAD signal (see External signals).
C*n*PROT[CADLMT]	Specify the threshold value of the CAD counter. The internal counters use IMCLK0 (see Function clock); therefore, you must program MCR[PRESCALE] and MCR[MCLKDIV] correctly to ensure that MCLK0_OUT is faster than the expected modulator clock input. However, MCLK0_OUT must not be more than 15 times faster.

83.3.2.10 ZCD operations

A ZCD asserts when a filter channel changes sign. In case of the absence of a threshold register and ripples, the ZCD asserts multiple times.

Table 1136. ZCD operations

Using field	You can
C*n*CFR[RDFMT]	Configure the field as signed for the ZCD to work correctly.
C*n*CR[ZCDEN]	Write 0 to disable ZCD.
NIS[ZCD*n*]	Check the status of ZCD.
NIE[ZCDIE*n*]	Enable and disable the Zero Cross Detected interrupt for filter channel *n*.
C*n*CFR[ZCOP]	Select the type of zero crossing—rise, fall, or both.

Various zero cross compare outputs can be sent to the pulse trigger (see Pulse trigger).

83.3.2.11 Limit detection operations

Each filter sample is compared to values in the limit registers. This comparison is based on the final conversion value with offset correction applied. C*n*HILMT[HILMT] and C*n*LOLMT[LOLMT] registers contain the corresponding values that are used to compare with the result. C*n*HILMT[HILMT] is used for result comparison with the high limit, and C*n*LOLMT[LOLMT] is used for result comparison with the low limit.

Table 1137. Limit detection operations

Using field	You can
C*n*CR[LMTEN]	Write 0 to disable limit checking.
EIS[HLMT*n*], EIS[LLMT*n*], and EIS[WLMT*n*]	Check the status of high limit flag, low limit flag, and window limit flag, respectively.
EIE[HLMTIE*n*], EIE[LLMTIE*n*], and EIE[WLMTIE*n*]	Enable or disable the High Limit interrupt, Low Limit interrupt, and Window Limit Interrupt, respectively.
C*n*ACFR[ADMASEL]	Enable an alternative DMA and select the limit flag that triggers the DMA.
C*n*PROT[HLMTBK], C*n*PROT[LLMTBK], and C*n*PROT[WLMTBK]	Program to assert break signals BREAK_HIGH, BREAK_LOW, and BREAK_WIN, respectively (see External signals).
C*n*PROT[LMTOP]	Program a limit detection option.

Various limit compare outputs can be sent to the pulse trigger block (see Pulse trigger).

83.3.2.12 Pulse trigger

A pulse trigger is used to generate various sets of additional signals for each channel. You can use C*n*ACFR[PTMUX] to select one of these signals and send them to the chip for additional control. A dedicated conversion complete signal (FILT_CC) is also sent to the chip for each channel. For more on FILT_CC, see External signals.

83.3.2.12.1 High-low limit signals to pulse trigger

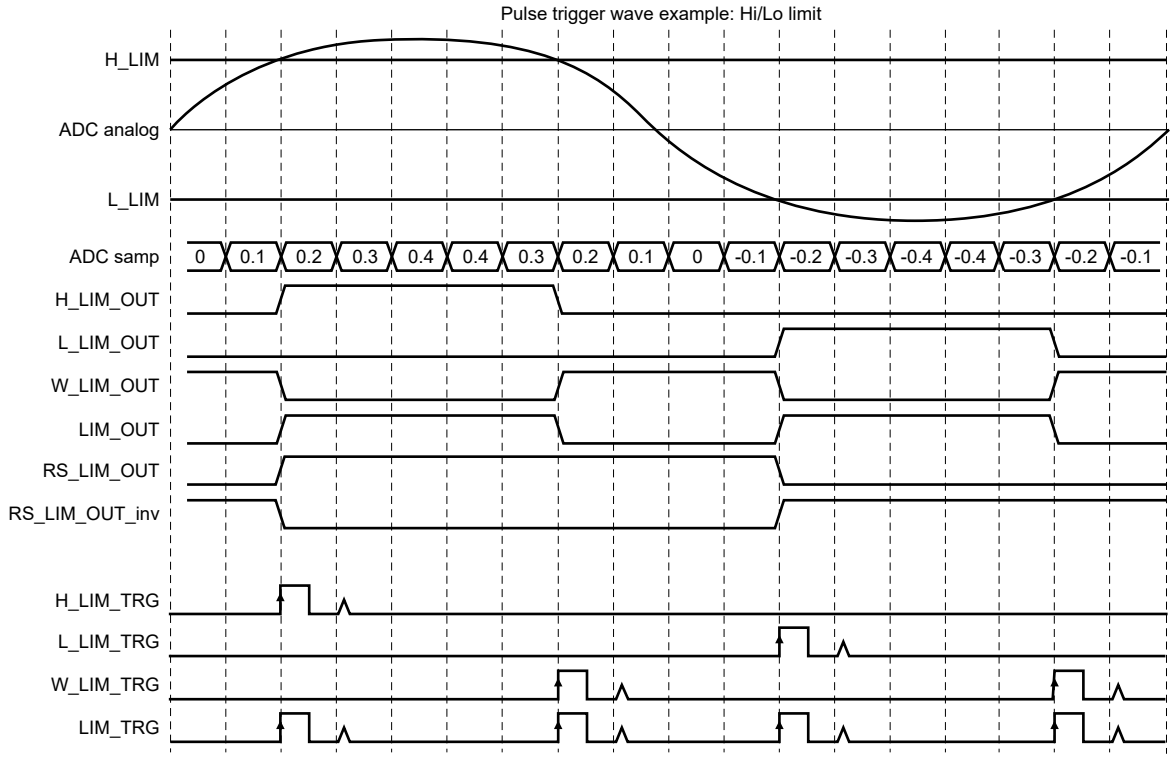


Figure 657. High-low limit signals to pulse trigger

83.3.2.12.2 ZCD output signals to pulse trigger

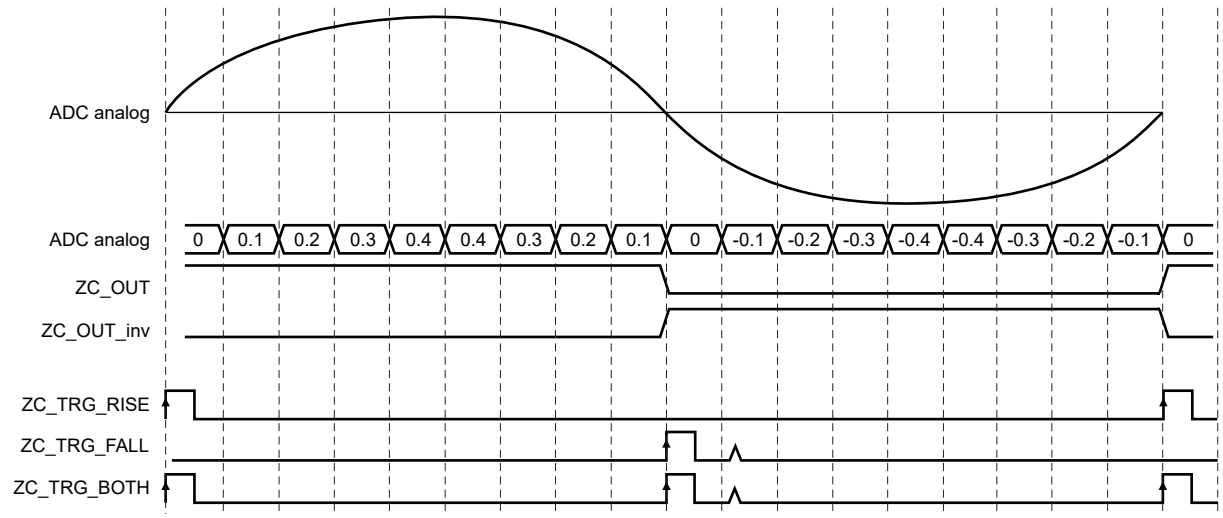


Figure 658. ZCD output signals to pulse trigger

83.3.3 FIFO

The FIFO is a first-in, first-out buffer that saves the final filtering results. Each FIFO has a depth of 16 entries, and each entry stores a 24-bit result. Each channel has one FIFO. SINC writes (pushes) data into the FIFO, and you read (pop) data from the FIFO by using Channel n Result Data (C0RDATA - C3RDATA).

The FIFO has a configurable watermark (C*n*CFR[FIFOWMK]). When the amount of data in the FIFO is greater than C*n*CFR[FIFOWMK], SINC requests a DMA, IRQ, or both, depending on NIE[CHFIE*n*] and C*n*CR[DMAEN].

When the FIFO is full, SINC discards a new filtered result and sets the overflow flag (FIFOIS[FOVF*n*]). Similarly, if you attempt a pop operation and the FIFO is empty, SINC does not return any valid data and sets the underflow flag (FIFOIS[FUNF*n*]). When an overflow or underflow occurs, SINC sets the aforementioned flags and can generate an error interrupt you write 1 to the corresponding field in FIFO And CAD Error Interrupt Enable (FIFOIE).

When FIFO is enabled (C*n*CR[FIFOEN] = 1), MCR.MEN = 1, but C*n*CR[CHEN] = 0, you can still read the FIFO to flush it.

When you reset the FIFO by writing 1 to MCR[RST], SINC fills all FIFO entries with zeros and resets the push and pop pointers to their initial position.

Only a PF can serve as the input to a FIFO.

You can use either of the following methods to determine how much data remains in a FIFO:

- Read C*n*SR[FIFOAVIL].
- Write 9d to C*n*CR[DBGSEL], then read Channel n Debug (C0DBGR - C3DBGR).

83.3.4 Wake-up

SINC does not support an asynchronous interrupt to wake up the chip.

83.3.5 Details of debug data

Table 1138. Details of debug data

When CnCR[DBGSEL] is:	CnDBGR[DBGDATA] contains:
0b	Final data from the PF (24 bits)
1b	Offset data (24 bits)
10b	Shifted data from the PF (24 bits)
11b	DC remover (HPF) data (32 bits)
100b	Raw data from the PF's CIC filter
110b	Historical data from short-circuit detection: • Bit 0 indicates the current count bit value. • Bits 1–8 indicate the current count. • Bits 9–16 indicate the previous count. • Bits 17–24 indicate the second-previous count. • Bits 25–31 are reserved.
111b	Data from the Manchester decoder: • Bits 0–7 indicate the number of decode errors (resync) that occurred after you enabled the channel. When this number exceeds 256, SINC resets it to 128. • Bits 8–18 indicate how long (in clock cycles) the current bit value has existed. • Bit 19 is the current bit value. • Bits 20–31 are reserved.
1000b	Data from CAD: • Bits 0–3 indicate the number of cycles during which SINC did not detect a modulator clock. SINC resets this number when SINC detects the modulator clock again. If it exceed the CnPROT[CADLMT], it will hold the value until software clear it. The maximum reportable number of clockless cycles is 15d (1111b). If more clockless cycles occur, the value in bits 0–3 remains 15d (1111b). • Bits 4–31 are zero.
1001b	Number of available entries in the FIFO: • Bits 0–7 indicate the number of available entries. • Bits 8–31 are zero.
1010b	Status of the parallel or serial data converter: • Bits 0–5 indicate the number of bits that SINC must still shift out as mod bits, for serial data converter. • Bits 6–7 are reserved. • Bits 8–31 are zero.

83.3.6 Modes of operation

Table 1139. Modes of operation

Mode	Description
Normal	Default operational mode in which channels perform filtering operations. An external bitstream from the MBIT[*n*] signal (see External signals) supplies the input data and the MCLK_OUT[*n*] signal (see External signals) delivers the modulator clock to the external ADC.
Low-Power	Mode in which the chip gates the CPU clock, and if MCR[DOZEN] = 1, SINC functions are canceled. If the CPU is in Wait mode, SINC filters and protects the modulator bitstream data and can generate a wake-up event through a synchronous DMA or interrupt, depending on the values of MCR[MEN], C*n*CR[CHEN], C*n*CR[PFEN], and other protection block enable fields (C*n*CR[CADEN], C*n*CR[SCDEN], or C*n*CR[ZCDEN]). If the CPU clock enters very low power stop (VLPS) mode, the CPU clock is gated and a peripheral-bus stop acknowledge signal is raised if: • MCR[MEN] = C*n*CR[CHEN] = 1. • Either C*n*CR[PFEN] or the other related protection enable fields (C*n*CR[CADEN], C*n*CR[SCDEN], or C*n*CR[ZCDEN]) are 1. You cannot generate DMAs and interrupts asynchronously to wake up the CPU clock.
Single	Mode in which you can control a typical motor using coils. For more on this mode, see Use case: Motor control with Single mode.
Continuous	Mode in which you can control audio applications and power meters, and create a sinusoidal decoder. For more on this mode, see Use case: Audio system with Continuous mode, Use case: Sinusoidal decoder with Continuous mode, and Use case: Power meter with Continuous mode.
Fixed-Number	Mode in which you can control a typical motor and enhance this control by using techniques as such: • Adding a compensation filter to achieve better precision. • Flattening the passband. • Improving roll-off. For more on this mode, see Use case: Motor control with Fixed-Number mode.

83.3.7 Clocks

83.3.7.1 Overview

SINC has these types of clocks:

- Bus (see Bus clock)
- Function (see Function clock)
- Modulator (see Modulator clock)

The following sections provide:

- A block diagram of SINC clocks.

- An overview of the functioning and special clocking requirements of SINC clock types.

83.3.7.1.1 Clocks: block diagram

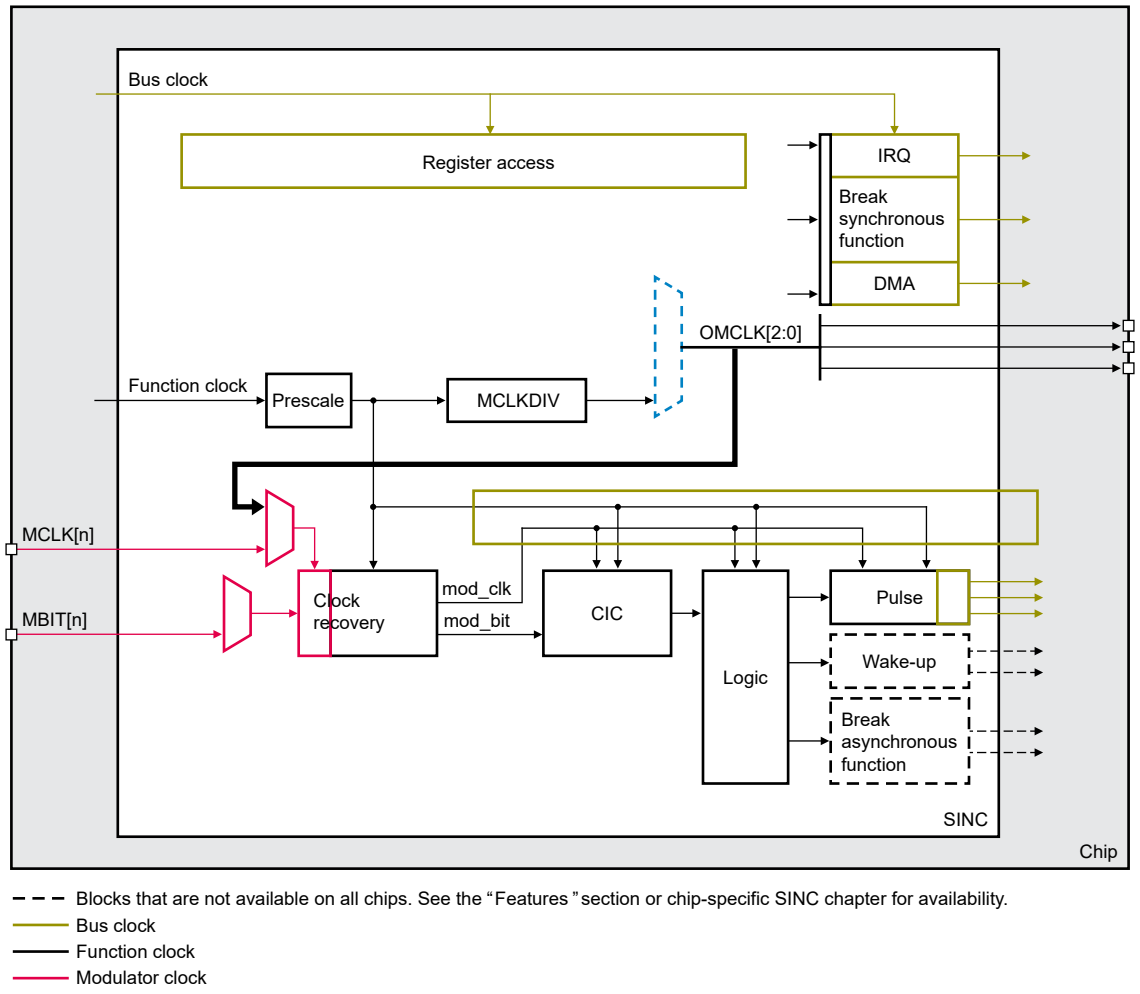


Figure 659. Clocks: block diagram

83.3.7.2 Bus clock

- Provides register access
- Allows control over low-power request signals, DMA, and interrupts

83.3.7.3 Function clock

The chip-provided function clock drives major functions, including:

- The prescaler
- The modulator clock divider (MCLKDIV)

The function clock is prescaled and divided to the correct operating range for SINC.

83.3.7.3.1 Main clock logic

The main clock logic, which is an integral part of the function clock:

- Is shared among all channels.

- Comprises a prescaler, MCLKDIV (divided by 2), and a clock output enable to external clocks.

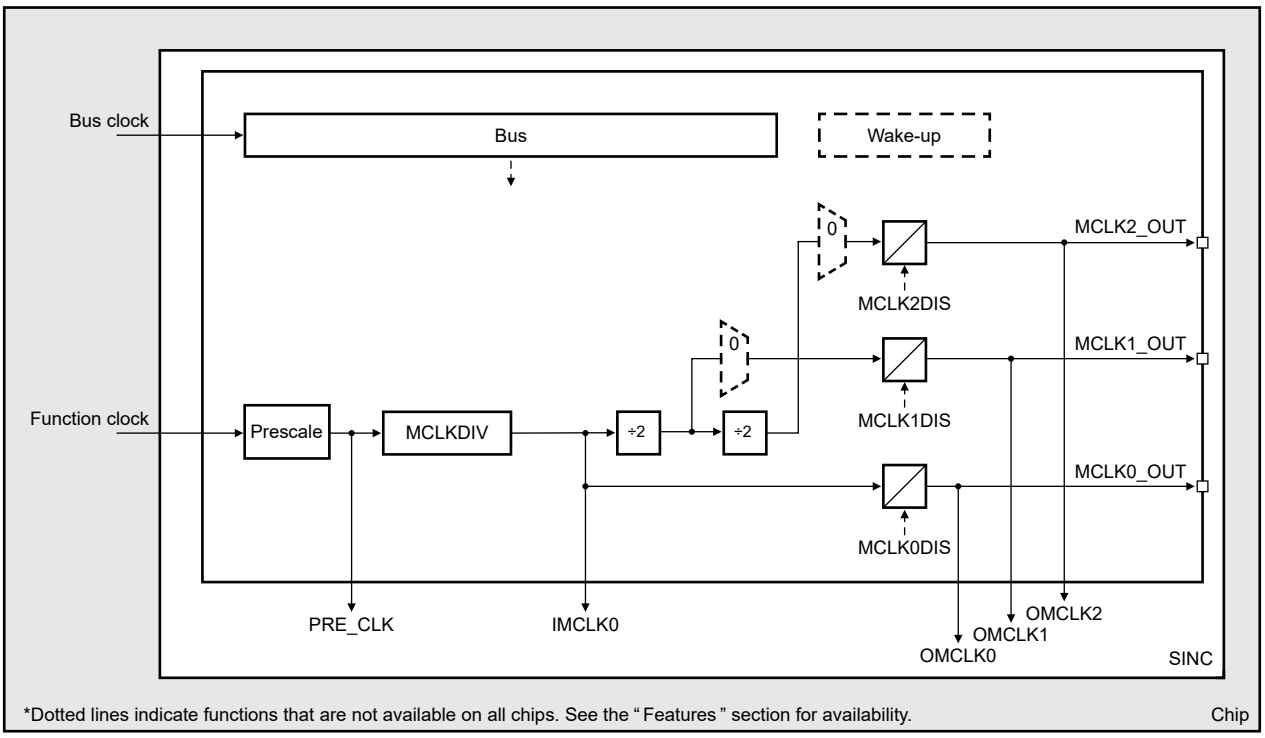


Figure 660. Main clock logic

83.3.7.3.2 Prescaler and output clocks

The prescaler supports power saving. It divides a high-frequency function clock (a major clock for other blocks) with an associated low-frequency clock. The prescaler divides the function clock by using MCR[PRESCALE], and then sends the clock to the divider and other blocks. See Main clock logic.

MCLKDIV sends the following three clocks to the chip for driving an external modulator ADC:

- MCLK_OUT0: Clock arrived at by adding 1 to the value of MCR[MCLKDIV].
- MCLK_OUT1: Clock arrived at by further dividing MCLK_OUT0 by 2.
- MCLK_OUT2: Clock arrived at by further dividing MCLK_OUT1 by 2.

These three aforementioned divided output clocks can be enabled or disabled by using MCR[MCLK0DIS], MCR[MCLK1DIS], and MCR[MCLK2DIS], respectively.

You must write an appropriate value to MCR[MCLKDIV]:

- For the target modulator clock to function properly. In some use cases, the modulator clock frequency must be certain times of the PWM frequency to use SINC's CIC filter.
- To obtain a 50% duty cycle, depending on the requirements of the external modulator ADC.

83.3.7.4 Modulator clock

A modulator clock:

- Is dedicated to each channel.
- Is provided by an external or system source.
- Can generate a digital output bitstream at a high data rate.

The SINC decimation filters reduce the sampling rate of the high-rate modulator output. Because of the latency of the decimation filters, each time a decimation filter is enabled or resynchronized, it takes one to four output samples (depending on the filter order) to reach a steady state producing valid output values.

83.3.7.4.1 Channel clock logic

The channel clock logic, which is a part of the modulator clock:

- Is dedicated to each channel individually.
- Comprises the following:
  - A normal decoder
  - A Manchester decoder
  - A clock source selection for these decoders
  - An MPDATA clock generator
- Includes the following:
  - An input asynchronous MCLK[*n*] selection, sample, or resynchronization
  - An MBIT[*n*] selection or sample
  - Manchester clock recovery
  - A clock select

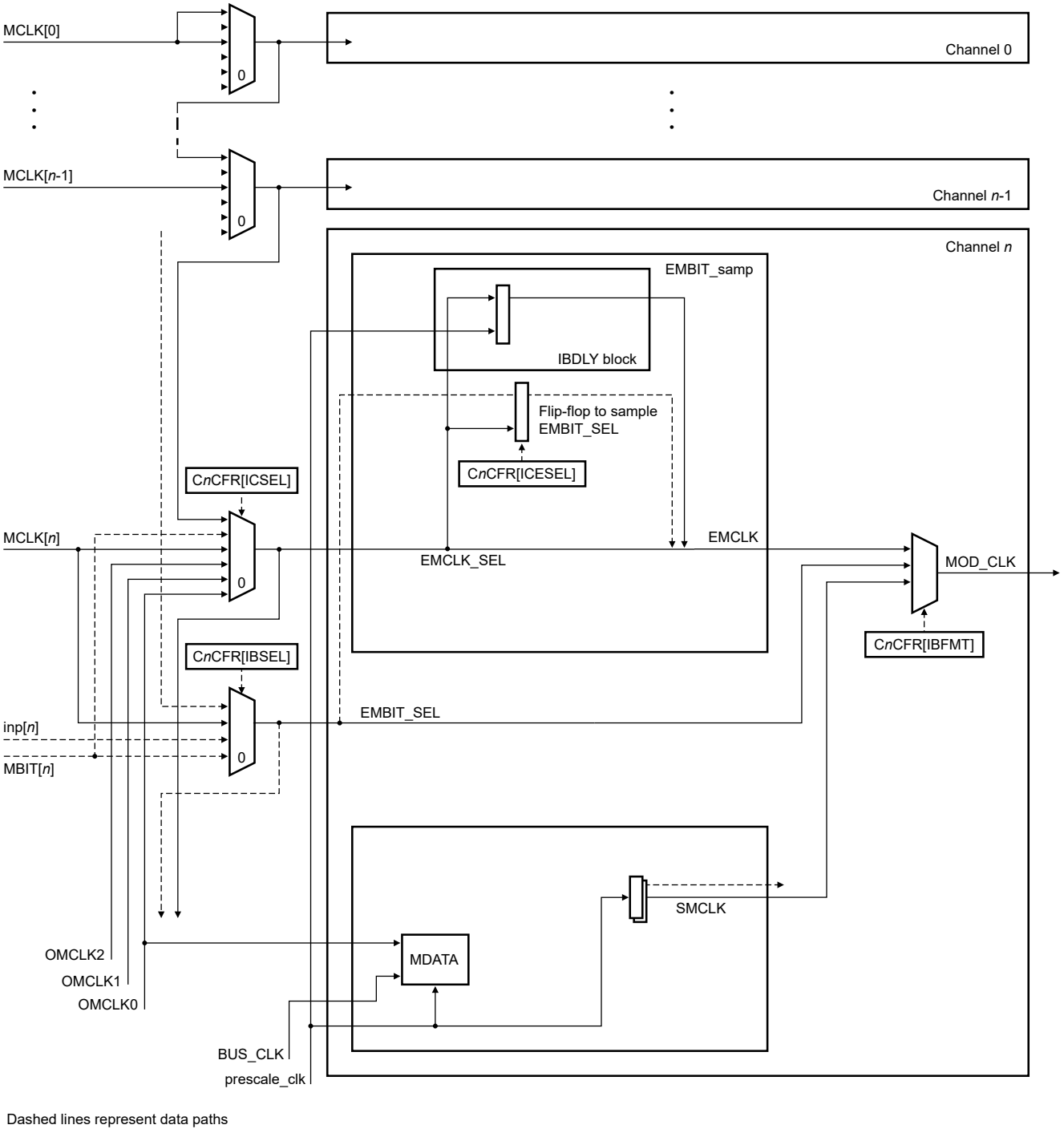


Figure 661. Channel clock logic

83.3.7.4.2 Prescaler considerations

You must program the prescaler to ensure that the predivided clock is:

- At least 3x faster than the MCLK[*n*] input for a single-edge sample.
- 6x faster than the MCLK[*n*] input for a dual-edge sample.
- 6x faster for Manchester coding, else the clock recovery block may function inappropriately.

83.3.7.4.3 Input clock configuration

To configure input clock edges and bitstream formats, you can:

- Write to C*n*CFR[ICSEL] to perform clock source selection and configure C*n*CFR[ICESEL] for the channel modulator clock to be rising edge, falling edge, or dual edge. These clock edges are used to sample the modulator bitstream.
- Write to C*n*CFR[IBFMT] for Manchester coding, when clocks are auto generated.

83.3.7.5 Clock source selection

The selected channel modulator clock input (EMCLK_SEL[*n*]) can be a dedicated external input (MCLK[*n*]), or it could be shared with an adjacent channel (MCLK[*n*-1]), if you write 111b to C*n*CFR[ICSEL]. Additionally, the shared clock can be cascaded between channels (for example, all channels share the same clock, MCLK[0]).

This is what you can do by writing appropriate values to the fields of Channel n Configuration (C0CFR - C3CFR):

- Route the output clocks OMCLK0, OMCLK1, and OMCLK2 as internal clock inputs by writing 000b, 001b, and 010b, respectively, to C*n*CFR[ICSEL]. Thus, SINC can provide output clocks to drive both the external modulator ADC and internal demodulator.
- Recover the manchester clock (MMCLK) by writing 01b to C*n*CFR[IBFMT]. You can use external modulator bitstream selection (EMBIT_SEL), which can be dedicated to each channel (external MBIT[*n*] and internal INP[*n*]) or shared with the adjacent channel (MBIT[*n*-1]), if you write 11b to C*n*CFR[IBSEL].
- Program the Manchester format by writing an appropriate value to C*n*CFR[ICESEL].

See Clock source selection settings for the values you must write to C*n*CFR[ICSEL], C*n*CFR[IBFMT], and C*n*CFR[ICESEL] for clock source selection.

83.3.7.5.1 Clock source selection settings

Table 1140. Clock source selection settings

C*n*CFR[ICSEL]	C*n*CFR[IBFMT]	C*n*CFR[ICESEL]	Source	Description
000	00	Any value	OMCLK0	Output clock 0 selected
001	00	Any value	OMCLK1	Output clock 1 selected
010	00	Any value	OMCLK2	Output clock 2 selected
011	00	Any value	MCLK	External modulator clock selected
100	00	Any value	—	—
101	00	Any value	—	Reserved
110	00	Any value		
111	00	Any value	GCLK	Group clock (a previous channel clock selected as clock source)
Any value	01	01	MMCLK_RISE	A manchester clock selected per channel in the Rise format
		10	MMCLK_FALL	A manchester clock selected per channel in the Fall format
		00	Reserved	—
		11		

Table continues on the next page...

Table 1140. Clock source selection settings (continued)

CnCFR[ICSEL]	CnCFR[IBFMT]	CnCFR[ICESEL]	Source	Description
Any value	10	Any value	PMCLK	An internal (IMCLK0), auto-generated parallel or serial modulator clock selected for clock source generation when you write a bitstream to Channel n Multipurpose Data (C0MPDATA - C3MPDATA)
	11		SMCLK	

83.3.7.6 Clock edge selection

For clock edge selection, you must program CnCFR[ICESEL] to select when to sample a modulator bitstream (EMBIT_SEL), if CnCFR[IBFMT] = 0.

See Clock edge selection settings for information about which modulator input clock (EMCLK_SEL) and clock edge you must select to sample this modulator bit (EMBIT_SEL), and see Clock edge selection diagram for a pictorial representation of these edge settings.

83.3.7.6.1 Clock edge selection settings

Table 1141. Clock edge selection settings

CnCFR[IBFMT]	CnCFR[ICESEL]	Name	Clock edge used to sample the modulator bit (EMBIT_SEL)
00	001	MCLK positive edge	EMCLK_SEL as positive edge sample—SINC uses even-numbered MCLK edges (0, 2, 4, 6, 8, and so on) to sample the selected input data.
	010	MCLK negative edge	EMCLK_SEL as negative edge sample—SINC uses odd-numbered MCLK edges (1, 3, 5, 7, 9, and so on) to sample the selected input data.
	011	MCLK dual edge	EMCLK_SEL as dual edge sample—SINC uses all MCLK edges to sample the selected input data.
	101	MCLK positive edge even	EMCLK_SEL as every other even positive edge sample—SINC uses even-numbered MCLK edges (0, 4, 8, 12, 16, and so on) to sample the selected input data.
	110	MCLK negative edge odd	EMCLK_SEL as every other odd negative edge sample—SINC uses odd-numbered MCLK edges (1, 5, 9, 13, 17, and so on) to sample the selected input data.
	000—111	Reserved	—
01	00		
	01	MMCLK_RISE	A manchester clock per channel in the Rise format—no edge sample options.
	10	MMCLK_FALL	A manchester clock per channel in the Fall format—no edge sample options.
	11	Reserved	—

Table continues on the next page...

Table 1141. Clock edge selection settings (continued)

CnCFR[IBFMT]	CnCFR[ICESEL]	Name	Clock edge used to sample the modulator bit (EMBIT_SEL)
10	Any value	PMCLK	An auto-generated internal clock when you write a bitstream to Channel n Multipurpose Data (C0MPDATA - C3MPDATA)—no edge sample options. IMCLK0 is selected for clock source generation.
11		SMCLK	

83.3.7.6.2 Clock edge selection diagram

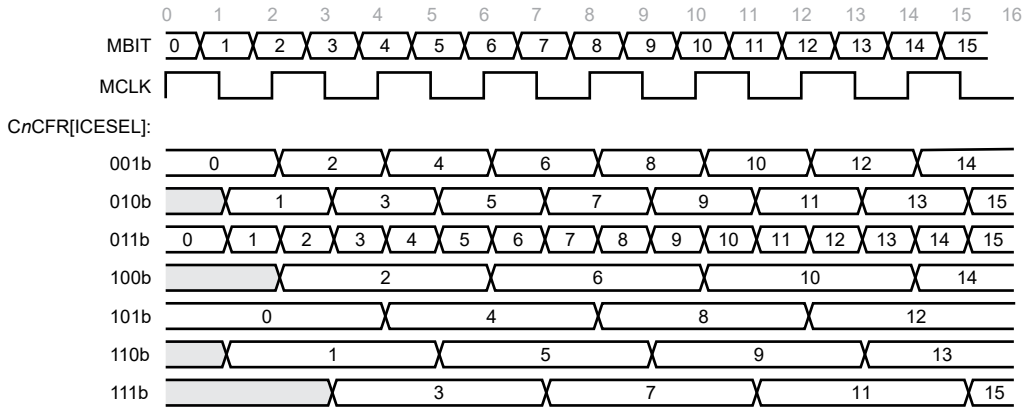


Figure 662. Clock edge selection diagram

83.3.7.7 Modulator clock frequencies and bit rates

The frequency of the function clock provided by the chip determines the minimum and maximum modulator clock frequencies.

83.3.7.7.1 Minimum clock output example

Assume the frequency of the function clock is 133 MHz. At this frequency, the minimum clock output frequency for MCLK_OUT2 is 16 kHz, and an internal or external clock can route it to SINC. Alternatively, an external ADC can divide the clock by 2. Therefore, in this example, the bit rate is 16 kbit/s or 8 bit/s in ideal scenarios.

See Minimum clock output settings for minimum clock output frequency settings of signals at different data rates and the resultant conditions.

83.3.7.7.1.1 Minimum clock output settings

Table 1142. Minimum clock output settings

Signal	Minimum frequency	Condition	CnCFR[ICESEL]	Data rate (kbit/s)	Comments
MCLK_OUT0	65 kHz (133 MHz ÷ 8 ÷ 256 = 65 kHz)	Clock routed (CnCFR[ICSEL] = 000b)	001 = positive edge 010 = negative edge	65	—
			011	130	

Table continues on the next page...

Table 1142. Minimum clock output settings (continued)

Signal	Minimum frequency	Condition	CnCFR[ICESEL]	Data rate (kbit/s)	Comments
			1xx	32	If ADC divides the clock by 2 but does not connect it
		ADC routes the clock after dividing it by 2 (CnCFR[ICSEL] = 011b)	001 = positive edge 010 = negative edge	32	—
			011	65	
			1xx	16	
MCLK_OUT1	32 kHz (133 MHz ÷ 8 ÷ 256 ÷ 2 = 32 kHz)	Clock routed (CnCFR[ICSEL] = 001b)	001 = positive edge 010 = negative edge	32	
			011	65	
			1xx	16	
		ADC routes the clock after dividing it by 2 (CnCFR[ICSEL] = 011b)	001 = positive edge 010 = negative edge	16	
			011	32	
			1xx	8	
MCLK_OUT2	16 kHz (133 MHz ÷ 8 ÷ 256 ÷ 2 ÷ 2 = 16 kHz)	Clock routed (CnCFR[ICSEL] = 010b)	001 = positive edge 010 = negative edge	16	
			011	32	
			1xx	8	
		ADC routes the clock after dividing it by 2 (CnCFR[ICSEL] = 011b)	001 = positive edge 010 = negative edge	8	
			011	16	
			1xx	4	

83.3.7.7.2 Maximum clock output example

Assume the frequency of the function clock is 133 MHz. At this frequency:

- 66 MHz is the maximum clock output frequency for MCLK_OUT0. The actual frequency depends on the chip I/O timing and the specific use case.

- 44 Mbit/s is the maximum data rate that SINC can support.

Therefore, an external ADC must divide a 66 MHz clock by 2 and route it as 33 MHz. The chip's I/O timing may also limit this high-speed clock.

With an output of 44 MHz, the duty cycle cannot be 50% when the clock is divided by 3. Most external ADCs require a 50% duty cycle. Therefore, 33 MHz is a better choice for the maximum frequency.

See Maximum clock output settings for maximum clock output frequency settings of signals at different data rates and the resultant conditions.

83.3.7.7.2.1 Maximum clock output settings

Table 1143. Maximum clock output settings

Signal	Condition	C*n*CFR[ICESEL]	Maximum frequency (MHz)	Data rate (Mbit/s)	Comments
MCLK_OUT0	Clock routed (C*n*CFR[ICSEL] = 000b)	001 or 010	33	33	133 MHz ÷ 4 = 33 MHz (50% duty cycle)
			44	44	133 MHz ÷ 3 = 44 MHz (66% duty cycle, depending on the maximum data rate)
		011	22	44	133 MHz ÷ 6 = 22 MHz (depending on the maximum data rate)
		1xx	66	33	133 MHz ÷ 2 = 66 MHz
	ADC routes the clock after dividing it by 2 (C*n*CFR[ICSEL] = 011b)	001 or 010	66	33	133 MHz ÷ 2 = 66 MHz
		011	33	33	133 MHz ÷ 4 = 33 MHz (50% duty cycle)
			44	44	133 MHz ÷ 3 = 44 MHz (66% duty cycle, depending on the maximum data rate)
		1xx	66	17	133 MHz ÷ 2 = 66 MHz (SINC does not support clock division by 1)
MCLK_OUT1	Clock routed (C*n*CFR[ICSEL] = 001b)	001 or 010	17	17	133 MHz ÷ 4 ÷ 2 = 17 MHz (assuming you are also using MCLK_OUT0)
			33	33	133 MHz ÷ 2 ÷ 2 = 33 MHz (depending on the maximum output)
		011	17	33	133 MHz ÷ 4 ÷ 2 = 17 MHz (assuming you are also using MCLK_OUT0)
			22	44	133 MHz ÷ 3 ÷ 2 = 22 MHz (depending on the maximum output)
		1xx	17	8	133 MHz ÷ 4 ÷ 2 = 17 MHz (assuming you are also using MCLK_OUT0)
			33	17	133 MHz ÷ 2 ÷ 2 = 33 MHz (depending on the maximum output)
	ADC routes the clock after dividing it by 2 (C*n*CFR[ICSEL] = 011b)	001 or 010	17	8	133 MHz ÷ 4 ÷ 2 = 16 MHz (assuming you are also using MCLK_OUT0)
			33	17	133 MHz ÷ 2 ÷ 2 = 33 MHz (depending on the maximum output)

Table continues on the next page...

Table 1143. Maximum clock output settings (continued)

Signal	Condition	CnCFR[ICESEL]	Maximum frequency (MHz)	Data rate (Mbit/s)	Comments
MCLK_OUT2	Clock routed (CnCFR[ICSEL] = 010b)	011	17	17	133 MHz ÷ 4 ÷ 2 = 16 MHz (assuming you are also using MCLK_OUT0)
			33	33	133 MHz ÷ 2 ÷ 2 = 33 MHz (depending on the maximum output)
		1xx	17	4	133 MHz ÷ 4 ÷ 2 = 17 MHz (assuming you are also using MCLK_OUT0)
			33	8	133 MHz ÷ 2 ÷ 2 = 33 MHz (depending on the maximum output)
		001 or 010	8	8	133 MHz ÷ 4 ÷ 2 ÷ 2 = 8 MHz (assuming you are also using MCLK_OUT0)
			17	17	133 MHz ÷ 2 ÷ 2 ÷ 2 = 17 MHz (depending on the maximum output)
		011	8	17	133 MHz ÷ 4 ÷ 2 ÷ 2 = 8 MHz (assuming you are also using MCLK_OUT0)
			17	33	133 MHz ÷ 2 ÷ 2 ÷ 2 = 17 MHz (depending on the maximum output)
		1xx	8	4	133 MHz ÷ 4 ÷ 2 ÷ 2 = 8 MHz (assuming you are also using MCLK_OUT0)
			17	8	133 MHz ÷ 2 ÷ 2 ÷ 2 = 17 MHz (depending on the maximum output)
	ADC routes the clock after dividing it by 2 (CnCFR[ICSEL] = 011b)	001 or 010	8	4	133 MHz ÷ 4 ÷ 2 ÷ 2 = 8 MHz (assuming you are also using MCLK_OUT0)
			17	8	133 MHz ÷ 2 ÷ 2 ÷ 2 = 17 MHz (depending on the maximum output)
		011	8	8	133 MHz ÷ 4 ÷ 2 ÷ 2 = 8 MHz (assuming you are also using MCLK_OUT0)
			17	17	133 MHz ÷ 2 ÷ 2 ÷ 2 = 17 MHz (depending on the maximum output)
		1xx	8	2	133 MHz ÷ 4 ÷ 2 ÷ 2 = 8 MHz (assuming you are also using MCLK_OUT0)
			17	4	133 MHz ÷ 2 ÷ 2 ÷ 2 = 17 MHz (depending on the maximum output)

83.3.7.7.3 Maximum clock input example

The modulator input clock can be either routed from a SINC output clock or provided by an external ADC (see Block diagram for details).

The maximum input frequency allowed depends on:

- The maximum input data rate and use case.
- The chip's I/O timing.

Assuming that the frequency of the Function clock is 133 MHz, the maximum data rate is 44 Mbit/s (133MHz ÷ 3 = 44 Mbit/s) for a typical use case. For Manchester formats, the maximum allowed clock frequency (recovery from bit) is 22 MHz (133 MHz ÷ 6 = 22 Mbit/s).

The minimum allowed input frequency is not limited if the clock input comes from an external ADC. However, if you select the Manchester format, the limit is 133 MHz ÷ 1024 ÷ 8 = 16 kHz.

NOTE

To calculate the output sample rate, ORD = 3 in Continuous mode.

See Maximum clock input settings for maximum clock input frequency settings of signals at different data rates and the resultant conditions.

83.3.7.7.3.1 Maximum clock input settings

Table 1144. Maximum clock input settings

Case	C*n*CFR[ICESEL]	Maximum frequency (MHz)	Input data rate (Mbit/s)	OSR	Output sample rate	Comments
C*n*CFR[IBFMT] = 00	001 010	44	44	16	2.7 Mbit/s	44 MHz ÷ 16 = 2.7 Mbit/s; 14-bit CIC
				256	172 kbit/s	44 MHz ÷ 256 = 172 kbit/s; 26-bit CIC
				1290	34 kbit/s	44 MHz ÷ 1290 = 34 kbit/s; 32-bit CIC
	011	22	44	16	2.7 Mbit/s	—
				256	172 kbit/s	
				1290	34 kbit/s	
	1xx	44	22	16	1.4 Mbit/s	
				256	86 kbit/s	
				1290	17 kbit/s	
C*n*CFR[IBFMT] = 01	Any value	22	22	16	1.4 Mbit/s	For the Manchester format, recovered clocks must be a value smaller than 22 Mbit/s.
				256	86 kbit/s	—
				1290	17 kbit/s	

83.3.7.8 Modulator clock I/O timing

83.3.7.8.1 I/O timing diagram

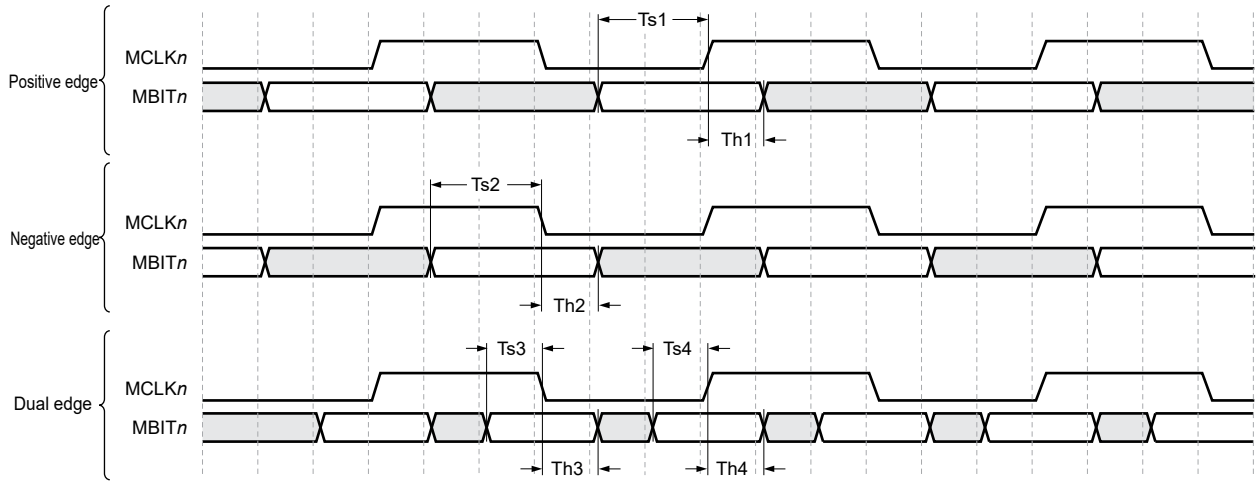


Figure 663. I/O timing diagram

See I/O timing settings for the symbols used in this diagram.

83.3.7.8.2 I/O timing settings

Table 1145. I/O timing settings

Symbol	Description	Minimum time (in ns)	Maximum time (in ns)
T_{s1}	Setup time from data valid to clock high	5	—
T_{h1}	Hold time from clock high to data valid	5	—
T_{s2}	Setup time from data valid to clock low	5	—
T_{h2}	Hold time from clock low to data valid	5	—
T_{s3}	Setup time from data valid to clock high	5	—
T_{h3}	Hold time from clock high to data valid	5	—
T_{s4}	Setup time from data valid to clock low	5	—
T_{h4}	Hold time from clock low to data valid	5	—

83.3.8 Reset

This section describes how to reset the SINC module and explains special requirements related to reset.

Writing 1 to MCR[RST] resets the following function blocks and registers:

- All function blocks (such as filters, short-circuit detectors, and so on) except for the clock
- Normal Interrupt Status (NIS)
- Error Interrupt Status (EIS)
- FIFO And CAD Error Interrupt Status (FIFOIS)
- Status (SR)

- MCR[STRIG*n*], together with related functions

83.3.9 Interrupts and breaks

This section describes all the interrupts that SINC generates.

Each channel has several interrupt sources that you can control and check with dedicated register fields.

If NIE[COCIE*n*] = 1, then an interrupt request is used to indicate that one filtering result is available in the corresponding channel's FIFO. You can then read the result in Channel n Result Data (C0RDATA - C3RDATA).

If NIE[CHFIE*n*] = 1, then an interrupt request is used to indicate that the FIFO surpassed its watermark level. In this case, SINC sets the corresponding NIS[CHF*n*] flag.

SINC can also generate an error interrupt request when an unexpected condition such as a FIFO overflow or underflow, a loss of clock, saturation, or exceeded value threshold occurs.

When an error occurs, SINC can assert break signals to deactivate an external peripheral. You control this feature by using Channel n Protection (C0PROT - C3PROT).

Table 1146. Interrupts and breaks

Status register	Status field	Enable register	Enable field	Break enable field in C*n*PROT	Description
NIS	COC*n*	NIE	COCIE*n*	—	Conversion complete
	CHF*n*		CHIE*n*	—	Data ready
	ZCS*n*		ZCIE*n*	—	Zero cross detected
EIS	SCD*n*	EIE	SCDIE*n*	SCDBK	Short circuit detected
	WLMT*n*		WLMTIE*n*	WLMTBK	Window limit reached
	LLMT*n*		LLMTIE*n*	LLMTBK	Low limit reached
	HLMT*n*		HLMTIE*n*	HLMKBK	High limit reached
FIFOIS	FUNF*n*	FIFOIE	FUNFIE*n*	—	FIFO underflow
	FOVF*n*		FOVFIE*n*	—	FIFO overflow
	CADF*n*		CADIE*n*	CADBK	Clock absence detected
	SAT*n*		SATIE*n*	—	Saturation

83.3.10 DMA

SINC has two DMAs for each channel:

- Primary DMA exclusively for the FIFO
- Alternate DMA for various sources (see C*n*ACFR[ADMASEL])

If C*n*CR[DMAEN] = 1, you read the FIFO-stored samples through DMA transactions. Each channel has dedicated DMA requests. If the FIFO of each channel reach their watermark the output interface makes a request for DMA transaction.

SINC clears the "data output ready" flags (NIS.CHFn) when a DMA transaction finishes.

SINC supports all the alternate DMA sources mentioned in C*n*ACFR[ADMASEL]. You can select only one source at a time. After an alternate DMA transaction finishes, SINC clears the related flags in Normal Interrupt Status (NIS), Error Interrupt Status (EIS), and FIFO And CAD Error Interrupt Status (FIFOIS).

83.4 External signals

Table 1147. External signals

Signal	Used for	Description
BREAK_CAD[3:0]	Output to the chip	CAD-related break signals sent to the chip for turning off other functions
BREAK_HIGH[3:0]	Output to the chip	High-limit break signals sent to the chip for turning off other functions
BREAK_LOW[3:0]	Output to the chip	Low-limit break signal sent to the chip for turning off other functions
BREAK_SCD[3:0]	Output to the chip	SCD-related break signal sent to the chip for turning off other functions
BREAK_WIN[3:0]	Output to the chip	Window-limit-related break signal sent to the chip for turning off other functions
FILT_CC[3:0]	Output to the chip	Signal acting as an output to be used by downstream modules in the signal processing chain that SINC is a component of
HTRIG[3:0]	Input from the chip	Hardware external trigger
INP[3:0]	Input from the chip	Internal modulator bitstream input
MBIT[3:0]	Input from the chip	External modulator bitstream input
MCLK_OUT0	Output to the chip	Modulator clock output 0
MCLK_OUT1	Output to the chip	Modulator clock output 1, which is MCLK_OUT0 divided by 2
MCLK_OUT2	Output to the chip	Modulator clock output 2, which is MCLK_OUT1 divided by 2
MCLK[3:0]	Input from the chip	External modulator clock input
PULSE_TRG[3:0]	Output to the chip	Pulse-triggered output sent to the chip
ipi_coc_int[3:0]	Output to the chip	Conversion complete interrupt
ipi_chf_int[3:0]	Output to the chip	Channel (FIFO) output interrupt
ipi_zcs_int[3:0]	Output to the chip	Zero cross interrupt
ipi_scd_int[3:0]	Output to the chip	Short circuit detect interrupt
ipi_low_int[3:0]	Output to the chip	Low limit interrupt
ipi_high_int[3:0]	Output to the chip	High limit interrupt
ipi_win_int[3:0]	Output to the chip	Window limit interrupt
ipi_fund_int[3:0]	Output to the chip	FIFO underflow interrupt
ipi_fover_int[3:0]	Output to the chip	FIFO overflow interrupt
ipi_sat_int[3:0]	Output to the chip	Saturation interrupt
ipi_cad_int[3:0]	Output to the chip	Clock monitor interrupt

Table continues on the next page...

Table 1147. External signals (continued)

Signal	Used for	Description
ipi_chn_ored_int[3:0]	Output to the chip	ORed together for above interrupt per channel
ipd_req_sinc[3:0]	Output to the chip	Conversion complete DMA request for SINC
ipd_done_sinc[3:0]	Input from the chip	Handshake signal from chip, indicates that DMA request is processed
ipd_req_alt[3:0]	Output to the chip	Alternative DMA request for SINC
ipd_done_alt[3:0]	Input from the chip	Handshake signal from chip, indicates that DMA request is processed

83.5 Internal signals

Table 1148. Internal signals

Signal	Used for	Description
H_LIM_OUT	Internal SINC functionality	Level signal that indicates a high limit
H_LIM_TRG	Internal SINC functionality	Pulse signal that indicates a high limit
HL_LIM_TRG	Internal SINC functionality	Pulse signal that indicates a high or low limit
L_LIM_OUT	Internal SINC functionality	Level signal that indicates a low limit
L_LIM_TRG	Internal SINC functionality	Pulse signal that indicates a low limit
LIM_OUT	Internal SINC functionality	Level signal that indicates a low or high limit
LIM_TRG	Internal SINC functionality	Pulse signal that indicates a high, low, or window limit
RS_LIM_OUT	Internal SINC functionality	Level signal that indicates a high level from an RS flip-flop or a Schmitt trigger
RS_LIM_OUT_inv	Internal SINC functionality	Level signal that indicates a low level from an RS flip-flop or a Schmitt trigger
W_LIM_OUT	Internal SINC functionality	Level signal that indicates a window limit
W_LIM_TRG	Internal SINC functionality	Pulse signal that indicates a window limit
ZC_OUT	Internal SINC functionality	Level signal that indicates a rising zero crossing
ZC_OUT_inv	Internal SINC functionality	Level signal that indicates a falling zero crossing

83.6 Initialization

83.6.1 Starting SINC

By default, SINC is disabled (MCR[MEN] = 0). To start or configure SINC, do this:

1. Configure the shared clock generation logic as described in Configure clocking-related fields before enabling SINC.

2. Configure OSR, ORD, and all other necessary options.
3. Enable the modulator clock output.
4. Enable all the required decimation filters and protection functions.
5. Write 1 to MCR[MEN] to start SINC.

The corresponding clock is generated for both the internal (IMCLK0) and external (MCLK_OUT*n*) signals (see External signals). If MCLK_OUT*n* is enabled, SINC waits for seven clock cycles before the external ADC is stable. SR[MCLKRDY*n*] fields become 1 after that.

This is how SINC behaves during and after starting up:

- For an internal clock source (if C*n*CFR[ICSEL] = 000b, 010b, or 001b), MCLK_OUT*n* checks SR[MCLKRDY*n*]. The field changes into a valid state when it becomes 1.
- If the clock source is external, Manchester, or grouped, all the three SR[MCLKRDY*n*] fields are selected and wait for a certain number of clock cycles before becoming 1.
- If MCLK_OUT*n* is disabled, SINC waits for one clock cycle for the internal logic to be stable.
- After startup, SINC waits for the conversion trigger to start conversion. The enabled channel conversions begin after SINC receives a software or hardware trigger. During startup, SINC ignores this signal.

83.6.2 Configure clocking-related fields before enabling SINC

Before you enable SINC (write 1 to MCR[MEN]), you must configure the following fields:

Table 1149. Configure clocking-related fields before enabling SINC

Register	Fields to configure
Main Control (MCR)	MCLK0DIS MCLK1DIS MCLK2DIS MCLKDIV PRESCALE

83.6.3 Configure fields before enabling channels

Before you enable a channel (write 1 to C*n*CR[CHEN]), you must configure the following fields:

Table 1150. Configure fields before enabling channels

Register	Fields to configure
Channel n Data Rate (C0DR - C3DR)	All
Channel n Configuration (C0CFR - C3CFR)	All except FIFOWMK
Channel n Protection (C0PROT - C3PROT)	All
Channel n Bias (C0BIAS - C3BIAS)	All
Channel n Low Limit (C0LOLMT - C3LOLMT)	All
Channel n High Limit (C0HILMT - C3HILMT)	All
Channel n Advanced Configuration (C0ACFR - C3ACFR)	All

83.7 Application information

This section describes several ways to use SINC.

83.7.1 Reading asynchronous data safely

1. Write 1 to C*n*SR[SRDS].
2. Wait until SINC changes C*n*SR[SRDS] to 0.
3. Read the data in Channel n Debug (C0DBGR - C3DBGR).

83.7.2 Transferring data between domains safely

To transfer data from the function-clock domain to the bus-clock domain safely, perform this procedure:

1. Choose the type of data that you want (see Details of debug data).
2. Write the corresponding value to C*n*CR[DBGSEL].
3. Write 1 to C*n*SR[SRDS].
4. Wait until C*n*SR[SRDS] becomes 0.
5. Read the data from C*n*DBGR.
6. Ensure that C*n*SR[DBGRS] = 0.

If you skip steps 1 and 2, C*n*DBGR contains the data from the last time you performed the full procedure.

Due to SINC's internal microarchitecture, the returned instantaneous data may be incomplete and invalid. Step 6 ensures that the data you read is valid.

83.7.3 Use case: Motor control with Single mode

83.7.3.1 Overview

This example uses Single mode to control a typical motor with three coils. Six on-chip PWM channels drive those three coils. Three external ADCs measure the coil current, and additional ADCs monitor total current or voltage to trigger critical events (such as overcurrent or short circuit) to shut down the motor.

The ADCs produce a high-speed bitstream and send it to SINC through MBIT signals (see External signals). Then, SINC:

1. Demodulates the bitstream.
2. Decimates the bitstream.
3. Converts the bitstream to a 24-bit result at a lower frequency.
4. Saves the result in a FIFO.

You can then:

1. Read the coil current from the FIFO.
2. Read other information (such as speed and position) from other modules on the chip.
3. Process the data to determine the parameters (on or off, PWM ratio drive strength) for the next coil-drive step.

Schematic shows the motor-control schematic.

Frequency response shows the resulting frequency response of the PF's CIC. In that diagram, pay particular attention to the following:

- Without a compensation filter, there is a drop in the passband, and roll-off (aliasing) occurs in the stop band.
- F_D (346 kHz) is near the 17th harmonic wave of the PWM switch frequency (20 kHz). This example filters out this noise more efficiently.

- A dropoff of –3 dB occurs at F_C (91 kHz). Therefore, the input signal must have a much-lower frequency—1 kHz or lower. If this is not possible, your software must compensate accordingly.

83.7.3.2 Schematic

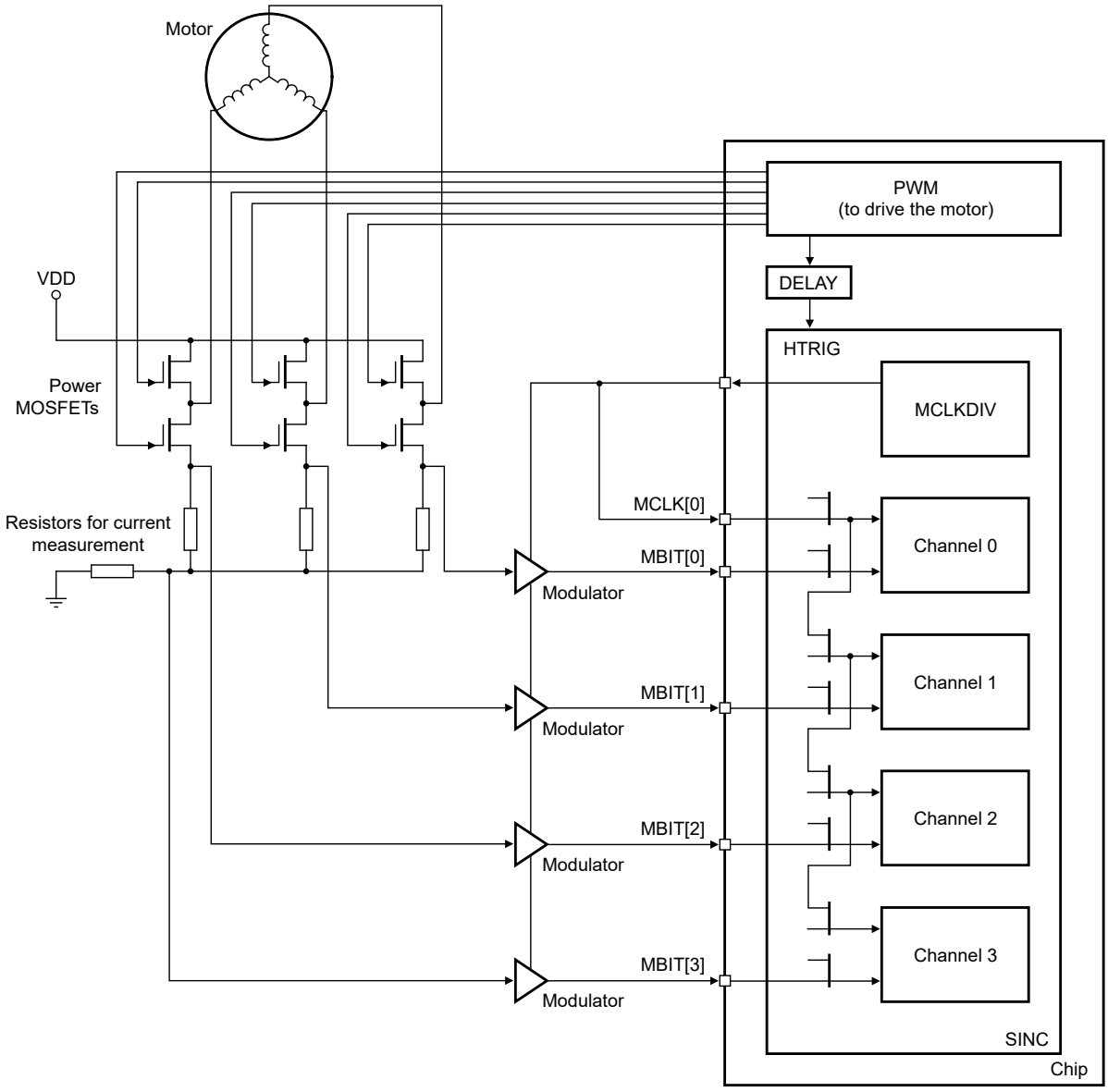


Figure 664. Schematic

83.7.3.3 Assumptions

Table 1151. Assumptions

Parameter	Value
SINC function clock	133 MHz

Table continues on the next page...

Table 1151. Assumptions (continued)

Parameter	Value
Target external modulator ADC clock (F_S)	22.2 MHz
Motor speed	6000 RPM (100 Hz)
PWM switch frequency (F_P)	20 kHz (corresponding to a 50 µs period, or 200 PWM steps per revolution)
OSR	64
Decimation frequency (F_D)	346 kHz (calculated from target external modulator ADC clock and OSR)
Cutoff frequency (F_C)	91 kHz
Signal-to-noise ratio (SNR)	79 dB
Effective number of bits (ENOB)	Approximately 12.9 (calculated using SNR)
Data width	20 bits (calculated using OSR and ORD)

83.7.3.4 Frequency response

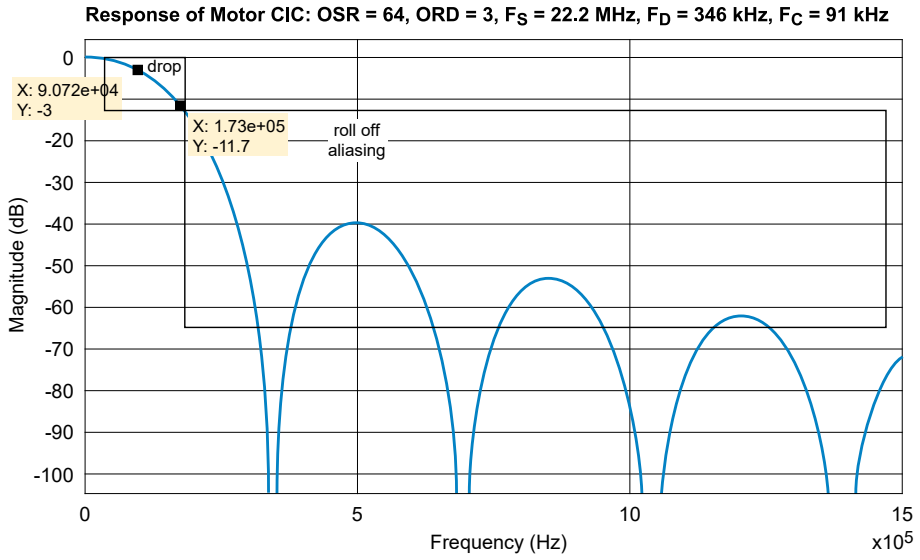


Figure 665. Frequency response

83.7.3.5 Procedure

1. Configure the related PWM and other chip modules to establish the correct pin connection with SINC. In particular, ensure that the PWM event signal connects to SINC as a hardware trigger, so that when PWM starts driving the coil, SINC can start conversion at a specific time.
2. Program the PWM event delay such that when SINC starts conversion, the coil currents are stable, as shown in Measurement time.
3. If the external ADC does not provide a clock, configure the modulator clock output. In this case, MCLK_OUT0 drives all external ADCs. To achieve MCLK_OUT0 = 22.2 MHz:
   a. Write 0 to MCR[PRESCALE].
   b. Write 5d to MCR[MCLKDIV].

4. Configure the data rate for channels 0–2 (*n* = 0–2) as follows:
    a. Write 3d to C*n*DR[PFORD].
    b. Write 63d to C*n*DR[PFOSR] to select OSR = 64.
    c. Write 0 to C*n*DR[PFCM] to select Single mode.
5. Configure the data rate for channel 3 (*n* = 3) as follows:
    a. Write 2d to C*n*DR[PFORD] to select the optional fast response.
    b. Write 63d to C*n*DR[PFOSR] to select OSR = 64.
    c. Write 10b to C*n*DR[PFCM] to select Always mode for monitoring total current.
6. Configure fields in Channel n Configuration (C0CFR - C3CFR) for data operation as follows:

Channels (value of *n*)	C*n*CFR field to write	Value to write	Purpose
0–3	ITLVL	0	Select edge triggering
0–2	ITSEL	1	Select hardware triggering
3	ITSEL	0	Select software triggering
0–3	IBSEL IBFMT	0	Select a bitstream from the MBIT signals (see External signals)
0	ICSEL	11b	Select the external modulator clock
1–3	ICSEL	111b	Group the clocks with channel 0
0–3	ICESEL	1b or 10b	Configure the clock edge that you want
0–3	FIFOWMK	0	Set the FIFO watermark
0–3	RDFMT	0	Select the signed data format
0–3	PFSFT	1_0100b	Select a left shift of 4

NOTE

For channels 0–2, the conversion time (calculated using ORD×OSR÷F_S) is 8.6 µs. This is smaller than the PWM period (50 µs) and allows enough margin to adjust sample time that Measurement time requires.

7. Enable SINC features as follows:

Channels (value of *n*)	Register and field to write	Value to write	Purpose
0–3	C*n*CR[PFEN]	1	Enable the PFs
0–3	C*n*CR[CHEN]	1	Enable the channels and their subfunctions
0–3	C*n*CR[FIFOEN]	1	Enable each PF's FIFO transfers
3	C*n*CR[SCDEN]	1	Enable SCD
3	C*n*PROT[SCDOP]	0	Specify that both repeating bit values increment the SCD counter
3	C*n*PROT[SCDBK]	1	Enable the SCD break signal
3	C*n*PROT[SCDLMT]	192d	Specify the SCD limit threshold

8. Optionally enable other subfunctions if you need them.
9. To enable interrupts for all channels (*n* = 0–3), write 1 to NIE[COCIEn].
10. To enable conversion, write 1 to MCR[MEN].
11. Wait until SR[CHRDYn] becomes 1 for all channels (*n* = 0–3).
12. Start PWM.
13. To start conversion for channel 3, write 1 to MCR[STRIG3].
14. Wait until NIS[COC*n*] = 1. This indicates conversion completion.
15. Read the ADC results from the FIFO using Channel n Result Data (C0RDATA - C3RDATA).
16. Process the motor-control data.
17. Reconfigure PWM.
18. Go to step 14 for the next loop.

83.7.3.6 Measurement time

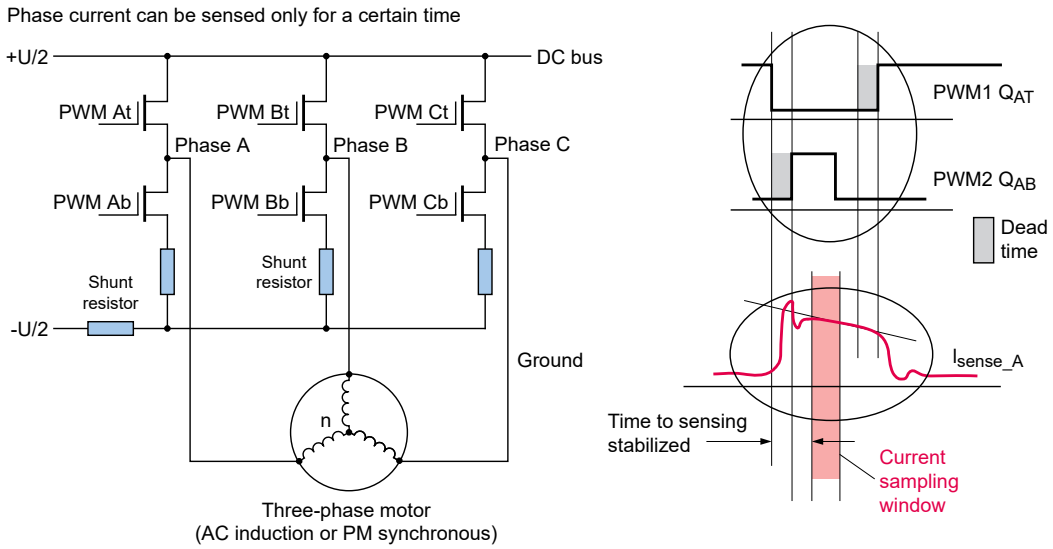


Figure 666. Measurement time

83.7.4 Use case: Motor control with Fixed-Number mode

83.7.4.1 Overview

This example improves on Use case: Motor control with Single mode by adding a compensation filter to achieve better precision, flatten the passband, improve roll-off, and so on. This method requires several samples for each PWM turn to perform filter compensation and sharpening with additional decimation. After the FIFO reaches its watermark, the conversion and filtering processes automatically stop.

NOTE

You can also achieve the same result with Continuous mode. When the FIFO reaches its watermark, you can toggle C*n*CR[PFEN] to abort conversion and wait for another trigger. The problem with this approach is that you must toggle C*n*CR[PFEN] as quickly as possible when SINC sets the corresponding NIS[CHF*n*] flag. If you disable conversion too late, it is hard to identify whether the 13th conversion had completed before you terminated conversion.

83.7.4.2 Assumptions

Table 1152. Assumptions

Parameter	Value
SINC function clock	133 MHz
Target external modulator ADC clock (F_S)	22.2 MHz
Motor speed	6000 RPM (100 Hz)
PWM switch frequency (F_P)	20 kHz (corresponding to a 50 µs period, or 200 PWM steps per revolution)
Number of samples per PWM period	12
OSR	64
Decimation frequency (F_D)	346 kHz (calculated using the target external modulator ADC clock and OSR)
Cutoff frequency (F_C)	91 kHz
Additional decimation factor	2, in software
SNR	Approximately 94 dB (see the ADC specifications in the chip data sheet)
ENOB	Approximately 15.4 (calculated using SNR)
Data width	20 bits (calculated using OSR and ORD)

83.7.4.3 Frequency response

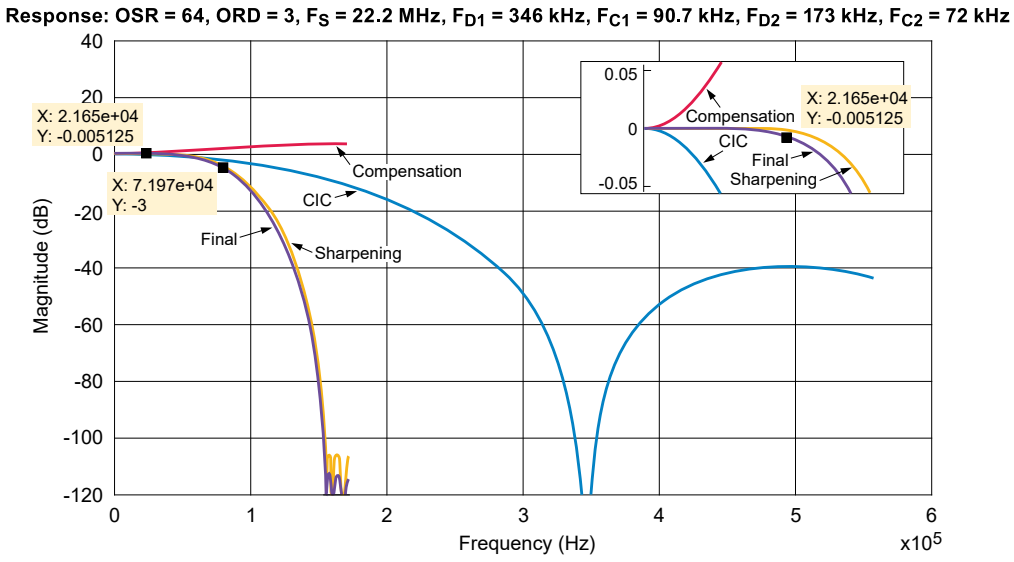


Figure 667. Frequency response

83.7.4.4 Procedure

1. Configure the related PWM and other chip modules to establish the correct pin connection with SINC. In particular, ensure that the PWM event signal connects to SINC as a hardware trigger, so that when PWM starts driving the coil, SINC can start conversion at a specific time.

2. Program the PWM event delay such that when SINC starts conversion, the coil currents are stable, as shown in Measurement time.
3. If the external ADC does not provide a clock, configure the modulator clock output. For this example, MCLK_OUT0 drives all external ADCs. To achieve MCLK_OUT0 = 22.2 MHz:
    a. Write 0 to MCR[PRESCALE].
    b. Write 5d to MCR[MCLKDIV].
4. Configure the data rate for channels 0–2 (*n* = 0–2) as follows:
    a. Write 3d to C*n*DR[PFORD].
    b. Write 63d to C*n*DR[PFOSR] to select OSR = 64.
    c. Write 11b to C*n*DR[PFCM] to select Fixed-Number mode.
5. Configure the data rate for channel 3 (*n* = 3) as follows:
    a. Write 2d to C*n*DR[PFORD] to select the optional fast response.
    b. Write 63d to C*n*DR[PFOSR] to select OSR = 64.
    c. Write 10b to C*n*DR[PFCM] to select Always On mode for monitoring total current.
6. Configure fields in Channel n Configuration (C0CFR - C3CFR) for data operation as follows:

Channels (value of *n*)	C*n*CFR field to write	Value to write	Purpose
0–3	ITLVL	0	Select edge triggering
0–2	ITSEL	1	Select hardware triggering
3	ITSEL	0	Select software triggering
0–3	IBSEL IBFMT	0	Select a bitstream from the MBIT signals (see External signals)
0	ICSEL	11b	Select the external modulator clock
1–3	ICSEL	111b	Group the clocks with channel 0
0–3	ICESEL	1b or 10b	Configure the clock edge that you want
0–3	FIFOWMK	11b	Set the FIFO watermark
0–3	RDFMT	0	Select the signed data format
0–3	PFSFT	1_0100b	Select a left shift of 4

NOTE

For channels 0–2, the conversion time for 12 samples (calculated using {[ORD+11]×OSR]}÷F_S) is 40.3 µs. This is smaller than the PWM period (50 µs) and allows enough margin to adjust sample time that Measurement time requires.

7. Enable SINC features as follows:

Channels (value of *n*)	Register and field to write	Value to write	Purpose
0–3	C*n*CR[PFEN]	1	Enable the PFs

Table continues on the next page...

Table continued from the previous page...

Channels (value of *n*)	Register and field to write	Value to write	Purpose
0–3	C*n*CR[CHEN]	1	Enable the channels and their subfunctions
0–3	C*n*CR[FIFOEN]	1	Enable each PF's FIFO transfers
3	C*n*CR[SCDEN]	1	Enable SCD
3	C*n*PROT[SCDOP]	0	Specify that both repeating bit values increment the SCD counter
3	C*n*PROT[SCDBK]	1	Enable the SCD break signal
3	C*n*PROT[SCDLMT]	192d	Specify the SCD limit threshold

8. Optionally enable other subfunctions if you need them.
9. To enable interrupts for all channels (*n* = 0–3), write 1 to NIE[COCIEn].
10. To enable conversion, write 1 to MCR[MEN].
11. Wait until SR[CHRDYn] becomes 1 for all channels (*n* = 0–3).
12. Start PWM.
13. To start conversion for channel 3, write 1 to MCR[STRIG3].
14. Wait until NIS[CHF*n*] = 1. This indicates conversion completion.
15. Read the ADC results from the FIFO using Channel n Result Data (C0RDATA - C3RDATA).
16. Optionally perform compensation:
    a. CIC gain adjust (divide 40_0000h if necessary) to get a fixed-point decimal between -1.0 and 1.0, than take part in the ADC reference voltage to get the final value in the correct format.
    b. Post-process the ADC result at a low speed, perform filter compensation and sharpening, and decimate the frequency by 2 to get the final ADC result as shown in Frequency response.
    c. If you have enough samples and don't have a hardware-delay feature, optionally perform delay compensation (for example, with a Lagrange polynomial).
17. Process the motor-control data.
18. Reconfigure PWM.
19. Go to step 14 for the next loop.

83.7.5 Use case: Audio system with Continuous mode

83.7.5.1 Overview

You can use SINC for two typical audio applications:

- Stereo recorder, using two SINC channels to sample the left and right audio channels
- Microphone array, using two or more SINC channels to sample a person's voice (based on distance from the person) and a beamforming software compensation to filter out environment noise

Schematic shows the audio schematic.

Waveforms: Audio system illustrates why you need different edges and why one MBIT input goes to two channels.

Frequency response after CIC filter shows the resulting frequency response of the PF's CIC.

Frequency response after HPF shows that HPF response is not linear and does not approach 0 dB.

83.7.5.2 Schematic

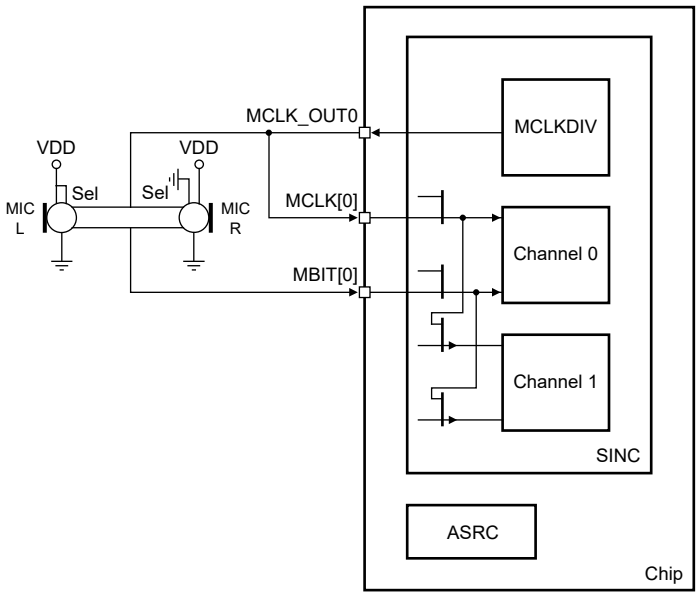


Figure 668. Schematic

83.7.5.3 Assumptions

Table 1153. Assumptions

Parameter	Value
SINC function clock	133 MHz
Target external modulator ADC clock	16.6 MHz
Target audio sample rate	44.0 kHz
Data precision	24 bits
OSR	94
Additional decimation factor	4
C*n*ACFR[HPFA]	1
Signal-to-noise ratio	Approximately 118 dB
Equivalent number of bits	Approximately 19.2
Final output frequency	44.22 kHz (calculated using the ADC clock, OSR, and additional decimation)

83.7.5.4 Waveforms: Audio system

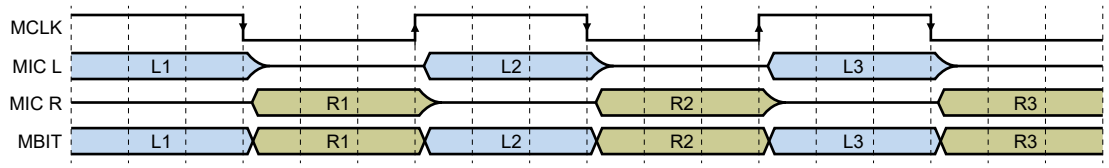


Figure 669. Waveforms: Audio system

83.7.5.5 Frequency response after CIC filter

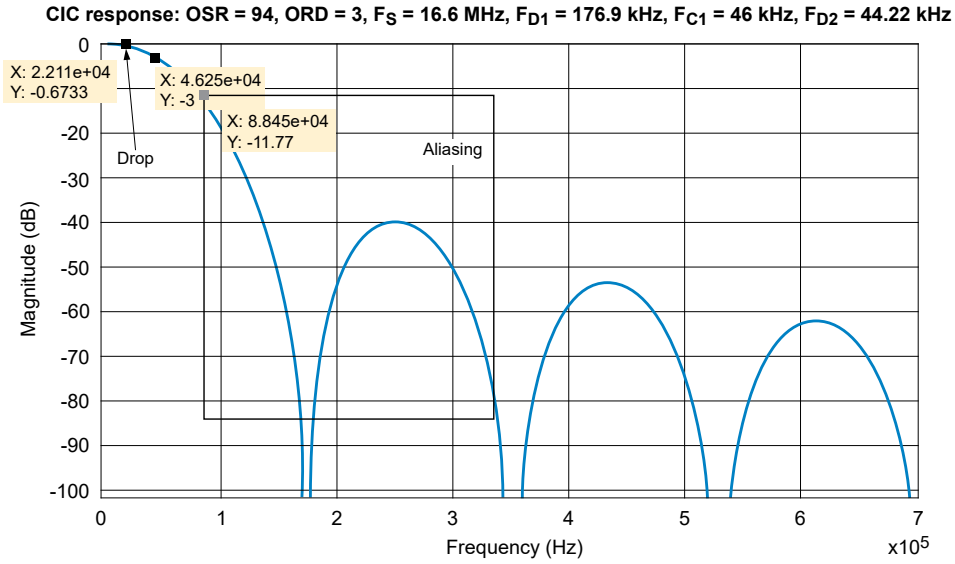


Figure 670. Frequency response after CIC filter

83.7.5.6 Frequency response after LPF and compensation

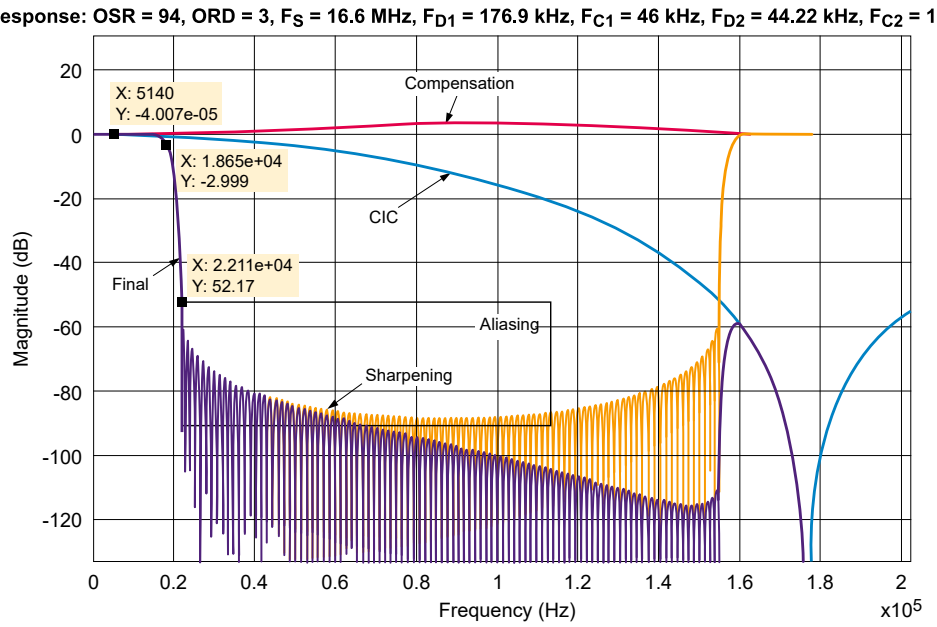


Figure 671. Frequency response after LPF and compensation

83.7.5.7 Frequency response after HPF

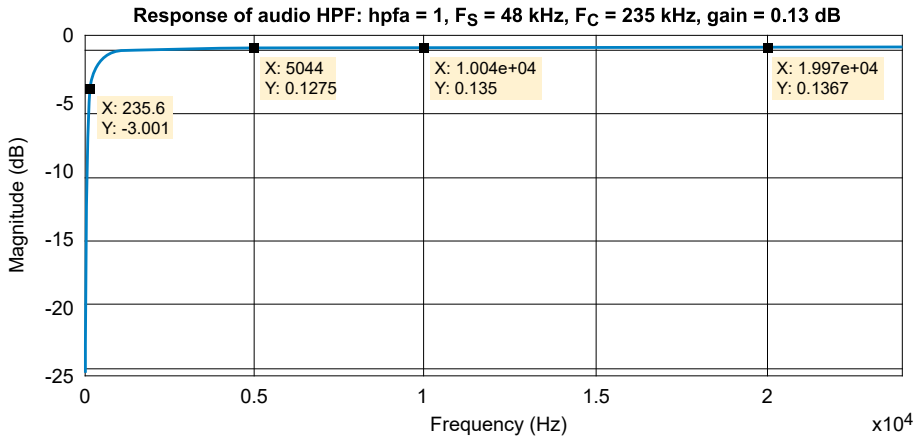


Figure 672. Frequency response after HPF

83.7.5.8 Procedure

This procedure involves only channels 0 and 1 (*n* = 0–1).

1. Configure the chip pin connections so that one MBIT goes to both SINC channels 0 and 1 (containing the left and right bitstreams, respectively).
2. If the external ADC does not provide a clock, configure the modulator clock output. For this use case, MCLK_OUT0 drives all external ADCs. To achieve MCLK_OUT0 = 16.6 MHz:
    a. Write 0 to MCR[PRESCALE].
    b. Write 7d to MCR[MCLKDIV].

3. Configure the data rate as follows:
    a. Write 3d to C*n*DR[PFORD].
    b. Write 93d to C*n*DR[PFOSR] to select OSR = 94.
    c. Write 01b to C*n*DR[PFCM] to select Continuous mode.
4. Configure fields in Channel n Configuration (C0CFR - C3CFR) for data operation as follows:

Channels (value of *n*)	C*n*CFR field to write	Value to write	Purpose
0–1	ITLVL	0	Select edge triggering
0–1	ITSEL	0	Select software triggering
0–1	IBSEL IBFMT	0	Select a bitstream from the MBIT signals (see External signals)
0	ICSEL	11b	Select the external modulator clock
1	ICSEL	111b	Group the clocks with channel 0
0	ICESEL	1	Select the left clock edge
1	ICESEL	10b	Select the right clock edge
0–1	FIFOWMK	7d	Set the FIFO watermark
0–1	RDFMT	0	Select the signed data format
0–3	PFSFT	1_0011b	Select a left shift of 3

5. Enable SINC features:
    - Optionally enable other subfunctions if you need them.

Channels (value of *n*)	Register and field to write	Value to write	Purpose
0–1	C*n*CR[PFEN]	1	Enable the PFs
0–1	C*n*CR[CHEN]	1	Enable the channels and their subfunctions
0–1	C*n*CR[FIFOEN]	1	Enable each PF's FIFO transfers
0–1	C*n*ACFR[HPFA]	1	Select the alpha coefficient

6. Optionally enable other subfunctions if you need them.
7. To enable interrupts for channels 0 and 1, write 1 to CHFIE0 and CHFIE1 in Normal Interrupt Enable (NIE).
8. To enable conversion, write 1 to MCR[MEN].
9. Wait until SR[CHRDYn] becomes 1.
10. To start conversion, write 1 to MCR[STRIGn].
11. Wait until NIS[CHF*n*] = 1. This indicates that the FIFOs are nearly full.
12. Process the audio data. This can include:
    - Adjusting the CIC gain. For example, divide the 24-bit result by OSR^{ORD} (in this case, C_AC78h) to get a fixed-point decimal value between –1.0 and 1.0. Then, account for the ADC reference voltage to get the real value.

- Using your software to perform additional compensation, low-pass filtering (sharpening), and decimation by 4. See Frequency response after LPF and compensation for results.
- Using your software to perform HPF gain compensation. Frequency response after HPF shows that you must compensate by adding 0.1367 dB.
- Using your software or another module (such as a sample-rate converter) to perform asynchronous data-rate conversion. In this use case, the final data rate is 44.22 kHz, but the audio system requires 44.000 kHz.
- Optionally performing delay compensation or beamforming. You can use a Lagrange polynomial for this.

13. Go to step 11 for the next loop.

83.7.6 Use case: Sinusoidal decoder with Continuous mode

83.7.6.1 Overview

You can use Continuous mode to create a sinusoidal decoder.

Consider a traditional quadrature encoder—an incremental encoder with two out-of-phase output channels that is used in many general automation applications for sensing the direction of rotary movement. (See Waveforms: Crude-position quadrature decoder.) Compared with such an encoder, a sinusoidal encoder provides much higher position and speed resolution. This improvement comes from a pair of quadrature sine and cosine signals as the shaft rotates. The signal typically comes from optical or magnetic sensors and typically generates approximately 1000 steps or cycles per revolution.

This example uses two external modulator ADCs to measure the sine and cosine signals. The modulator ADC results are high-speed bitstreams. The chip sends these results to SINC through MBIT signals (see External signals). Then, SINC:

1. Demodulates the bitstream.
2. Decimates the bitstream.
3. Converts the bitstream to a 24-bit lower-speed result.
4. Saves the result in a FIFO.

The internal pulse trigger generates a pulse to the quadrature decoder elsewhere on the chip. This generates the crude position. You can then read the sine and cosine wave from the FIFO and process the data to get the fine position. The crude and fine positions, combined together, can achieve an approximately 24-bit precision of position.

Schematic shows the sinusoidal-decoder schematic.

83.7.6.2 Schematic

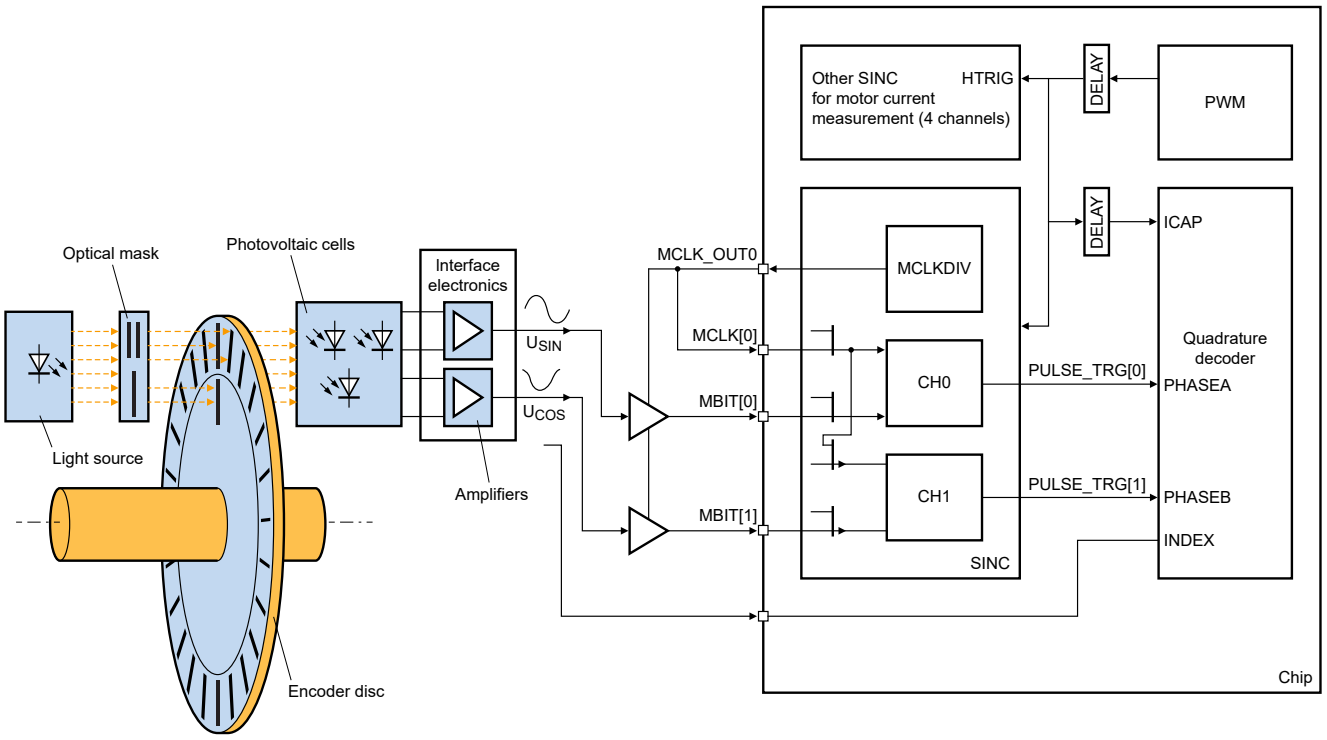


Figure 673. Schematic

83.7.6.3 Assumptions

For channels measuring motor current:

- Ensure one of the following. This helps you get the position when you measure the current.
  - OSR is the same for all channels.
  - OSR for one channel is an integer multiple of all other OSRs.
- Use the hardware trigger for both sinusoidal and current-measurement channels with different delay settings.

Table 1154. Assumptions

Parameter	Value
SINC function clock	133 MHz
Target external modulator ADC clock	22.2 MHz
Motor speed	600 RPM (10 Hz)
Encoder disk pulses per revolution	512 sine and cosine
Data precision	25 bits
ORD	3
OSR	64
Additional decimation factor	2

Table continues on the next page...

Table 1154. Assumptions (continued)

Parameter	Value
Signal-to-noise ratio	Approximately 94 dB
Equivalent number of bits	Approximately 15.4
Final output frequency	173 kHz
Final output precision	25 bits

83.7.6.4 Waveforms: Crude-position quadrature decoder

This diagram shows signals that are typical in a generic quadrature decoder.

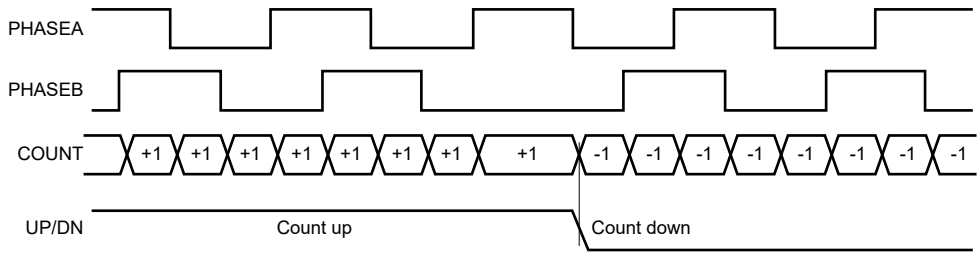


Figure 674. Waveforms: Crude-position quadrature decoder

83.7.6.5 Procedure

This procedure involves only channels 0 and 1 (*n* = 0–1).

1. Configure the chip pin connections and related modules so they match Schematic.
2. Configure the additional SINC instance shown in Schematic for motor-current measurement according to Use case: Motor control with Single mode.
3. If the external ADC does not provide a clock, configure the modulator clock output. For this example, MCLK_OUT0 drives drive all external ADCs. To achieve MCLK_OUT0 = 22.2 MHz:
    a. Write 0 to MCR[PRESCALE].
    b. Write 5d to MCR[MCLKDIV].
4. Configure the data rate as follows:
    a. Write 3d to C*n*DR[PFORD].
    b. Write 63d to C*n*DR[PFOSR] to select OSR = 64.
    c. Write 01b to C*n*DR[PFCM] to select Continuous mode.
5. Configure fields in Channel n Configuration (C0CFR - C3CFR) for data operation as follows:

Channels (value of *n*)	C*n*CFR field to write	Value to write	Purpose
0–1	ITLVL	0	Select edge triggering
0–1	ITSEL	0	Select software triggering
0–1	IBSEL	0	Select a bitstream from the MBIT signals (see External signals)

Table continues on the next page...

Table continued from the previous page...

Channels (value of *n*)	C*n*CFR field to write	Value to write	Purpose
	IBFMT		
0	ICSEL	11b	Select the external modulator clock
1	ICSEL	111b	Group the clocks with channel 0
0–1	ICESEL	1b or 10b	Configure the clock edge that you want
0–1	FIFOWMK	7d	Set the FIFO watermark
0–1	RDFMT	0	Select the signed data format
0–1	PFSFT	1_0100b	Select a left shift of 4

6. Configure the pulse trigger:
    a. Write 111b to C*n*ACFR[PTMUX].
    b. To set the high-detect margin, write 1000d to C*n*HILMT[HILMT].
    c. To set the low-detect margin, write FF_FC18h to C*n*LOLMT[LOLMT].
7. Enable SINC features as follows:

Channels (value of *n*)	Register and field to write	Value to write	Purpose
0–1	C*n*CR[PFEN]	1	Enable the PFs
0–1	C*n*CR[CHEN]	1	Enable the channels and their subfunctions
0–1	C*n*CR[FIFOEN]	1	Enable each PF's FIFO transfers

8. Optionally enable other subfunctions if you need them.
9. To enable interrupts for channels 0 and 1, write 1 to CHFIE0 and CHFIE1 in Normal Interrupt Enable (NIE).
10. To enable conversion, write 1 to MCR[MEN].
11. For channels 0–1 (*n* = 0–1), wait until SR[CHRDY*n*] becomes 1.
12. Start quadrature for crude decoder and PWM for motor drive.
13. Wait until NIS[CHF*n*] becomes 1. This indicates that the FIFOs are nearly full.
14. Process the resulting data:
    a. Read ADC results from the FIFO.
    b. Apply compensation, low-pass filtering, and additional decimation to the ADC results.
    c. Calculate position.
    d. Save results to memory.
    e. Save the timestamp or number of conversions after the hardware trigger occurred. You can use C*n*SR[CNUM] to determine how many samples SINC gathered after receiving a hardware trigger in Continuous mode.
15. Go to step 13 for the next loop.

NOTE

In your software motor control main loop (separate from the procedure above), select the memory-stored position that corresponds to the time when you measured the coil current.

83.7.7 Use case: Power meter with Continuous mode

83.7.7.1 Overview

You can use SINC for a power meter.

A typical low-cost power meter can achieve a wide dynamic range (approximately 6 mA to 60 A) using:

- A successive approximation register ADC (SAR ADC), with 12- to 16-bit resolution, to measure typical voltages (for example, 220 V or 380 V) that have a stable wave.
- A sigma-delta ADC (SD ADC), with 20- to 24-bit resolution, to measure current.

For a single-phase power meter, use one SINC channel to measure current and one SAR ADC to measure voltage.

For a three-phase power meter, use three or four SINC channels to measure current and three SAR ADCs to measure voltage. If your SAR ADC supports multiplexed channels, you can use one SAR ADC to sample three voltages at different times.

Ideally, voltage and current measurements must occur at the same time. However, in this use case, simultaneous measurement is hard to achieve. You can use SINC's FILT_CC output to synchronize the measurement time between SINC and SAR ADC.

Schematic shows the power-meter schematic.

83.7.7.2 Schematic

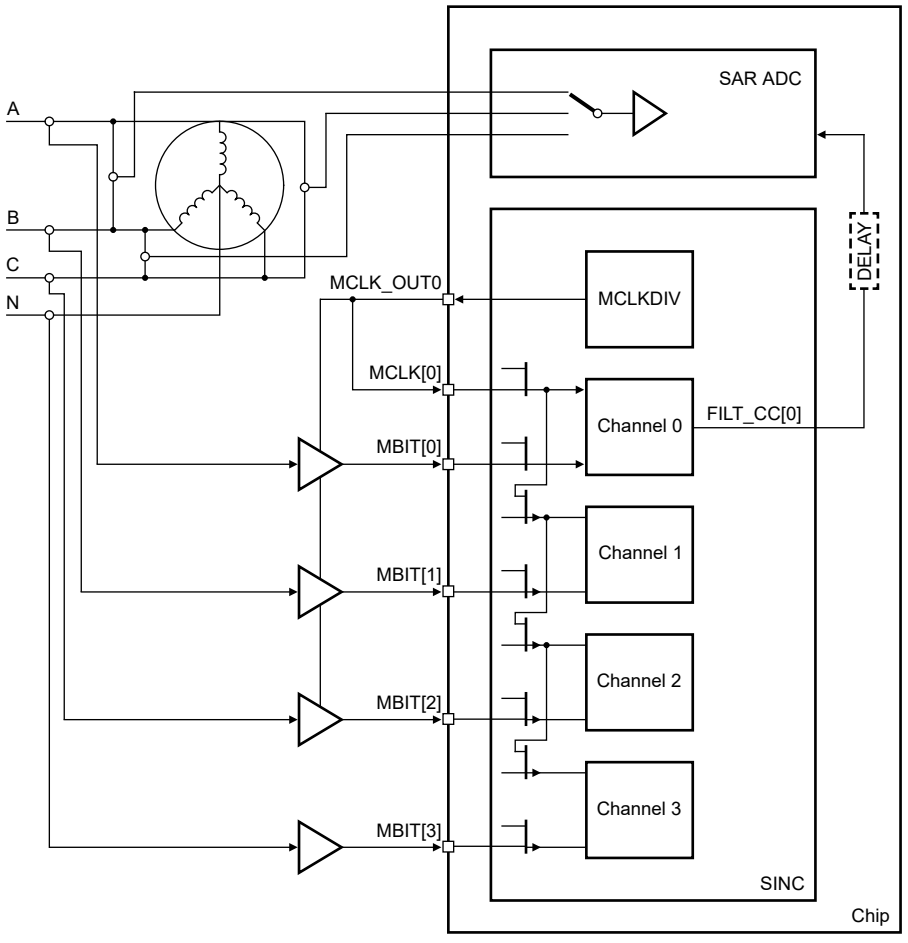


Figure 675. Schematic

83.7.7.3 Considerations

SAR ADC must support switching and must sample three voltage inputs when it receives a trigger.

This use case includes the following delays:

- From the SINC sample time to FILT_CC
- From FILT_CC to the delay block
- From the delay block to the start of SAR ADC's conversion of A in Schematic
- From the end of SAR ADC's conversion of A to the start of conversion B

You can calculate the delay, and you must compensate for that delay.

You can also manage delays by configuring the delay value to be near the SINC sample rate (0.5 × OSR), and ignore SINC's first output. In this case, when SINC completes its second conversion, SAR ADC has also just finished its first sample. Therefore, you can reduce the delay-compensation effort and improve precision by carefully configuring the delay value.

For the exact delay setting, consider SINC's CIC phase response and align it to SAR ADC's analog sample-and-hold time (not including conversion time). CIC phase response is linear, so you can convert it to a delay time, which is about half of OSR in Continuous mode.

83.8 SINC register descriptions

SINC supports a programming model for:

- Configuring the module.
- Storing the FIFO buffer.
- Enabling clock generation.
- Generating triggers.
- Protecting SINC operation.
- Using DMA.

Before you enable SINC or its channels, you must configure certain fields as described in Configure clocking-related fields before enabling SINC and Configure fields before enabling channels.

In the memory map, the following addresses are reserved and do not generate a transfer error if you access them:

- 28h
- 2Ch
- 30h
- 34h

83.8.1 SINC memory map

SINC1 base address: 42BF_0000h

SINC2 base address: 42C0_0000h

SINC3 base address: 42C1_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0200_006Dh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4h	Parameters (PARAMETER)	32	RW	0070_0410h
8h	Main Control (MCR)	32	RW	0000_0000h
Ch	Normal Interrupt Enable (NIE)	32	RW	0000_0000h
10h	Error Interrupt Enable (EIE)	32	RW	0000_0000h
14h	FIFO And CAD Error Interrupt Enable (FIFOIE)	32	RW	0000_0000h
18h	Normal Interrupt Status (NIS)	32	RW	0000_0000h
1Ch	Error Interrupt Status (EIS)	32	RW	0000_0000h
20h	FIFO And CAD Error Interrupt Status (FIFOIS)	32	RW	0000_0000h
24h	Status (SR)	32	R	000F_0000h
38h	Channel 0 Control (C0CR)	32	RW	0000_0000h
3Ch	Channel 0 Data Rate (C0DR)	32	RW	0000_1800h
40h	Channel 0 Configuration (C0CFR)	32	RW	0020_0000h
44h	Channel 0 Protection (C0PROT)	32	RW	0000_0000h
48h	Channel 0 Bias (C0BIAS)	32	RW	0000_0000h
4Ch	Channel 0 Low Limit (C0LOLMT)	32	RW	0000_0000h
50h	Channel 0 High Limit (C0HILMT)	32	RW	0000_0000h
54h	Channel 0 Result Data (C0RDATA)	32	R	0000_0000h
58h	Channel 0 Multipurpose Data (C0MPDATA)	32	RW	0000_0000h
5Ch	Channel 0 Advanced Configuration (C0ACFR)	32	RW	0000_0000h
60h	Channel 0 Status (C0SR)	32	RW	0000_0000h
64h	Channel 0 Debug (C0DBGR)	32	R	0000_0000h
68h	Channel 1 Control (C1CR)	32	RW	0000_0000h
6Ch	Channel 1 Data Rate (C1DR)	32	RW	0000_1800h
70h	Channel 1 Configuration (C1CFR)	32	RW	0020_0000h
74h	Channel 1 Protection (C1PROT)	32	RW	0000_0000h
78h	Channel 1 Bias (C1BIAS)	32	RW	0000_0000h
7Ch	Channel 1 Low Limit (C1LOLMT)	32	RW	0000_0000h
80h	Channel 1 High Limit (C1HILMT)	32	RW	0000_0000h
84h	Channel 1 Result Data (C1RDATA)	32	R	0000_0000h
88h	Channel 1 Multipurpose Data (C1MPDATA)	32	RW	0000_0000h
8Ch	Channel 1 Advanced Configuration (C1ACFR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
90h	Channel 1 Status (C1SR)	32	RW	0000_0000h
94h	Channel 1 Debug (C1DBGR)	32	R	0000_0000h
98h	Channel 2 Control (C2CR)	32	RW	0000_0000h
9Ch	Channel 2 Data Rate (C2DR)	32	RW	0000_1800h
A0h	Channel 2 Configuration (C2CFR)	32	RW	0020_0000h
A4h	Channel 2 Protection (C2PROT)	32	RW	0000_0000h
A8h	Channel 2 Bias (C2BIAS)	32	RW	0000_0000h
ACh	Channel 2 Low Limit (C2LOLMT)	32	RW	0000_0000h
B0h	Channel 2 High Limit (C2HILMT)	32	RW	0000_0000h
B4h	Channel 2 Result Data (C2RDATA)	32	R	0000_0000h
B8h	Channel 2 Multipurpose Data (C2MPDATA)	32	RW	0000_0000h
BCh	Channel 2 Advanced Configuration (C2ACFR)	32	RW	0000_0000h
C0h	Channel 2 Status (C2SR)	32	RW	0000_0000h
C4h	Channel 2 Debug (C2DBGR)	32	R	0000_0000h
C8h	Channel 3 Control (C3CR)	32	RW	0000_0000h
CCh	Channel 3 Data Rate (C3DR)	32	RW	0000_1800h
D0h	Channel 3 Configuration (C3CFR)	32	RW	0020_0000h
D4h	Channel 3 Protection (C3PROT)	32	RW	0000_0000h
D8h	Channel 3 Bias (C3BIAS)	32	RW	0000_0000h
DCh	Channel 3 Low Limit (C3LOLMT)	32	RW	0000_0000h
E0h	Channel 3 High Limit (C3HILMT)	32	RW	0000_0000h
E4h	Channel 3 Result Data (C3RDATA)	32	R	0000_0000h
E8h	Channel 3 Multipurpose Data (C3MPDATA)	32	RW	0000_0000h
ECh	Channel 3 Advanced Configuration (C3ACFR)	32	RW	0000_0000h
F0h	Channel 3 Status (C3SR)	32	RW	0000_0000h
F4h	Channel 3 Debug (C3DBGR)	32	R	0000_0000h

83.8.2 Version ID (VERID)

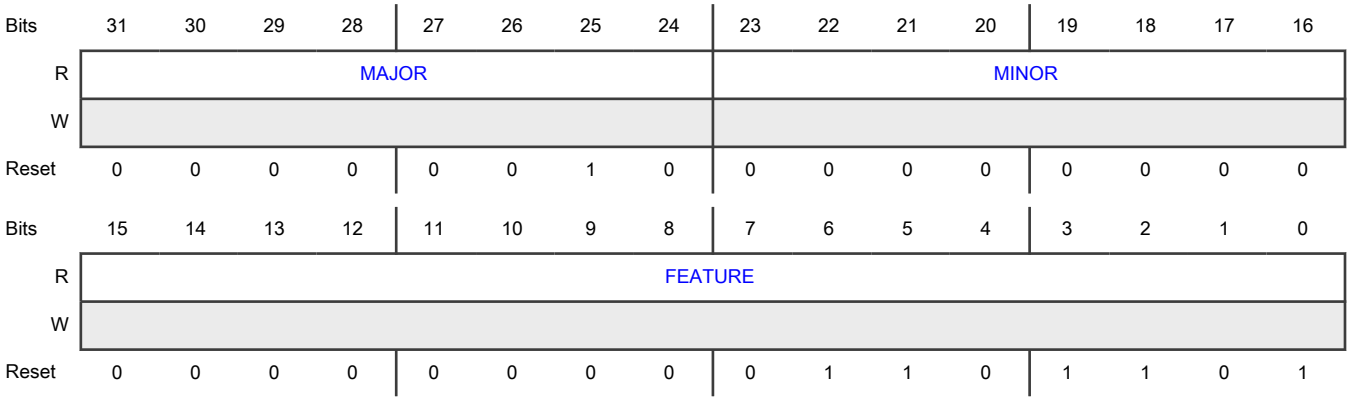
Offset

Register	Offset
VERID	0h

Function

Records the specific SINC version in the chip.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number of the SINC design. 0000_0001b - 1.x 0000_0010b - 2.x All other values are reserved.
23-16 MINOR	Minor Version Number Indicates the minor version number of the SINC design. 0000_0000b - x.0 All other values are reserved.
15-0 FEATURE	Feature Specification Code Contains a coded value in which each bit represents certain features available in this SINC instance.

Bit position	Feature
0	Feature set
1	Alternate clock selection and divider
2	Asynchronous function clock
4	Reserved
5	Availability of HPF
6	Availability of bias

Field	Function
	<table><tr><th>Bit position</th><th>Feature</th></tr><tr><td>15–7</td><td>Reserved</td></tr></table>

83.8.3 Parameters (PARAMETER)

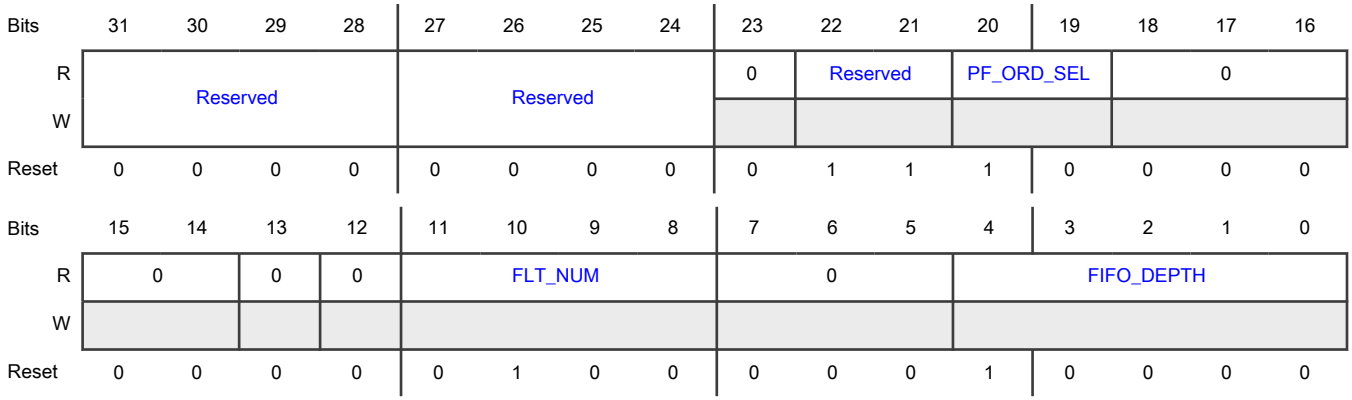
Offset

Register	Offset
PARAMETER	4h

Function

Specifies and indicates some of SINC's operating parameters.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 —	Reserved
23 —	Reserved
22-21	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
20-19 PF_ORD_SEL	PF Order Select Indicates the maximum order of the PF. 10b - 3 11b - 2 All other values are reserved.
18-16 —	Reserved
15-14 —	Reserved
13 —	Reserved
12 —	Reserved
11-8 FLT_NUM	Filter Channel Number Each filter channel has dedicate input and CIC filters.
7-5 —	Reserved
4-0 FIFO_DEPTH	FIFO Depth Entry numbers for each channel's FIFO.

83.8.4 Main Control (MCR)

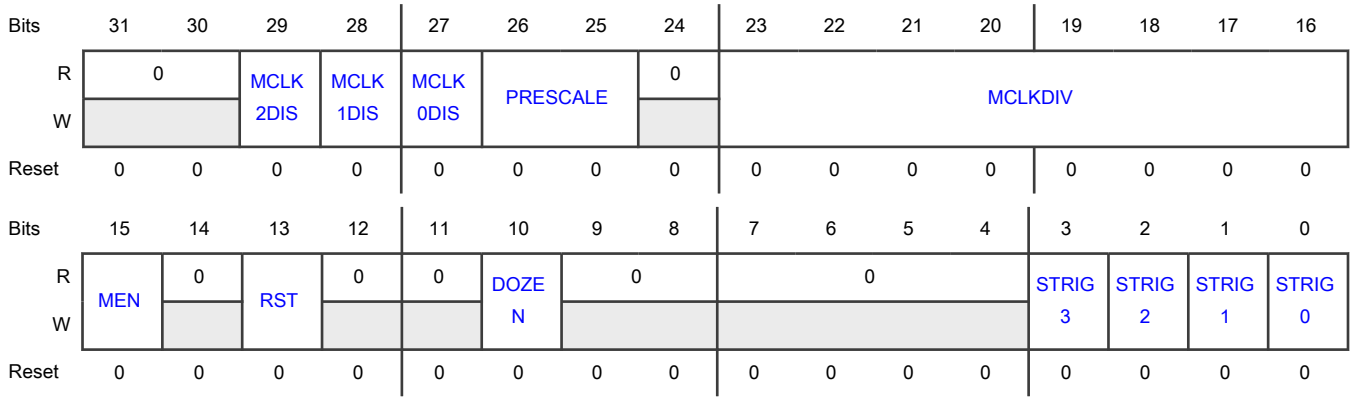
Offset

Register	Offset
MCR	8h

Function

Controls the global features of all channels.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 MCLK2DIS	Disable Modulator Clock 2 Output Works with MEN to disable the MCLK2 output. 0b - Enabled when MEN = 1 1b - Disabled regardless of MEN value
28 MCLK1DIS	Disable Modulator Clock 1 Output Works with MEN to disable the MCLK1 output. 0b - Enabled when MEN = 1 1b - Disabled regardless of MEN value
27 MCLK0DIS	Disable Modulator Clock 0 Output Works with MEN to disable the MCLK0 output. 0b - Enabled when MEN = 1 1b - Disabled regardless of MEN value
26-25 PRESCALE	Prescale Before Clock Divider Specifies the clock divider ratio for the modulator clock. 00b - No prescale 01b - 2 10b - 4 11b - 8
24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 MCLKDIV	Modulator Clock Divider Contributes to the modulator-clock divider ratio according to the following expression: Clock divider ratio = MCLKDIV + 1 The minimum clock divider ratio is 2, and therefore the minimum valid value of MCLKDIV is 1. MCLKDIV = 0 is prohibited. To obtain a 50% duty cycle in the MCLK output, write an odd value to MCLKDIV. 0000_0000b - Prohibited All other values - Added to 1 to specify the clock divider
15 MEN	Master Enable Simultaneously enables all function blocks enabled in their respective registers. Before you write 1 to this field, you must configure other fields as described in Configure clocking-related fields before enabling SINC. 0b - Disables 1b - Enables
14 —	Reserved
13 RST	Software Reset Triggers a reset for functional blocks and some registers as described in Reset. 0b - Do not reset 1b - Reset
12 —	Reserved
11 —	Reserved
10 DOZEN	Doze Or Stop Enable Disables SINC when the chip enters Doze or Stop mode. 0b - Enables 1b - Disables
9-8 —	Reserved
7-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3-0 STRIGn	Software Trigger For Channel n 0b - No effect 1b - Trigger

83.8.5 Normal Interrupt Enable (NIE)

Offset

Register	Offset
NIE	Ch

Function

Enables normal interrupt requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0				ZCDIE 3	ZCDIE 2	ZCDIE 1	ZCDIE 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				CHFIE 3	CHFIE 2	CHFIE 1	CHFIE 0	0				COCIE 3	COCIE 2	COCIE 1	COCIE 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-20 —	Reserved
19-16 ZCDIEn	Zero Cross Detected Interrupt Enable Enables the Zero Cross Detected interrupt for filter channel *n*.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disables 1b - Enables
15-12 —	Reserved
11-8 CHFIEn	Data Output Ready Interrupt Enable Enables the Data Output Ready interrupt for PF channel *n*. 0b - Disables 1b - Enables
7-4 —	Reserved
3-0 COCIEn	Conversion Complete Interrupt Enable Enables the Conversion Complete interrupt for PF channel *n*. 0b - Disables 1b - Enables

83.8.6 Error Interrupt Enable (EIE)

Offset

Register	Offset
EIE	10h

Function

Enables error-related interrupt requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				HLMTIE3	HLMTIE2	HLMTIE1	HLMTIE0	0				LLMTIE3	LLMTIE2	LLMTIE1	LLMTIE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				WLMTIE3	WLMTIE2	WLMTIE1	WLMTIE0	0				SCDIE3	SCDIE2	SCDIE1	SCDIE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 HLMTIEn	High Limit Interrupt Enable Enables the High Limit interrupt for filter channel *n*. 0b - Disables 1b - Enables
23-20 —	Reserved
19-16 LLMTIEn	Low Limit Interrupt Enable Enables the Low Limit interrupt for filter channel *n*. 0b - Disables 1b - Enables
15-12 —	Reserved
11-8 WLMTIEn	Window Limit Interrupt Enable Enables the Window Limit interrupt for filter channel *n*. 0b - Disables 1b - Enables
7-4 —	Reserved
3-0 SCDIEn	Short Circuit Detected Interrupt Enable Enables the Short Circuit Detected interrupt for filter channel *n*. 0b - Disables 1b - Enables

83.8.7 FIFO And CAD Error Interrupt Enable (FIFOIE)

Offset

Register	Offset
FIFOIE	14h

Function
Enables FIFO- and CAD-related interrupt requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SATIE3	SATIE2	SATIE1	SATIE0	0				CADIE3	CADIE2	CADIE1	CADIE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FOVFIE3	FOVFIE2	FOVFIE1	FOVFIE0	0				FUNFIE3	FUNFIE2	FUNFIE1	FUNFIE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 SATIEn	Saturation Interrupt Enable Enables the Saturation interrupt for PF channel *n*. 0b - Disables 1b - Enables
23-20 —	Reserved
19-16 CADIEn	Clock Absence Interrupt Enable Enables the Clock Absence interrupt for filter channel *n*. 0b - Disables 1b - Enables
15-12 —	Reserved
11-8 FOVFIEn	FIFO Overflow Interrupt Enable Enables the FIFO Overflow interrupt for PF channel *n*. 0b - Disables 1b - Enables
7-4 —	Reserved
3-0	FIFO Underflow Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
FUNFIEn	Enables the FIFO Underflow interrupt for PF channel *n*. 0b - Disables 1b - Enables

83.8.8 Normal Interrupt Status (NIS)

Offset

Register	Offset
NIS	18h

Function

Contains flags for normal interrupts on each SINC channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0				ZCD3	ZCD2	ZCD1	ZCD0
W													W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				CHF3	CHF2	CHF1	CHF0	0				COC3	COC2	COC1	COC0
W					W1C	W1C	W1C	W1C					W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-20 —	Reserved
19-16 ZCDn	Zero Cross Detected Flag Indicates whether the resulting data on filter channel *n* crossed zero and therefore changed sign. Use C*n*CFR[ZCOP] to configure the type of zero crossing that sets the Zero Cross Detected flag.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
15-12 —	Reserved
11-8 CHFn	Data Output Ready Flag Indicates whether the FIFO from PF channel *n* has overflowed its watermark level and the data is available in Channel n Result Data (C0RDATA - C3RDATA). This flag contributes to generating an interrupt or DMA request if you enabled these by using NIE[CHFIE*n*] or C*n*CR[DMAEN], respectively. SINC sets this flag according to the watermark level. If you enabled DMA requests, SINC clears the flag after the DMA controller finishes the DMA transaction. In other cases—for example, if you enabled interrupts—you clear the flag by writing 1 to it. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overflow; data not available 1b - Overflow; data available When writing 0b - No effect 1b - Clear the flag
7-4 —	Reserved
3-0 COCn	Conversion Complete Flag Indicates whether one conversion has finished and the data is available in Channel n Result Data (C0RDATA - C3RDATA). **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - Not finished; data not available 1b - Finished; data available When writing 0b - No effect 1b - Clear the flag

83.8.9 Error Interrupt Status (EIS)

Offset

Register	Offset
EIS	1Ch

Function

Contains flags for error-related interrupts on each SINC channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				HLMT3	HLMT2	HLMT1	HLMT0	0				LLMT3	LLMT2	LLMT1	LLMT0
W					W1C	W1C	W1C	W1C					W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				WLMT3	WLMT2	WLMT1	WLMT0	0				SCD3	SCD2	SCD1	SCD0
W					W1C	W1C	W1C	W1C					W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 HLMTn	High Limit Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates whether the conversion result for filter channel *n* exceeded the threshold value specified in C*n*HILMT[HILMT]. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not exceeded 1b - Exceeded When writing 0b - No effect 1b - Clear the flag
23-20 —	Reserved
19-16 LLMTn	Low Limit Flag Indicates whether the conversion result for filter channel *n* exceeded the threshold value specified in C*n*LOLMT[LOLMT]. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not exceeded 1b - Exceeded When writing 0b - No effect 1b - Clear the flag
15-12 —	Reserved
11-8 WLMTn	Window Limit Flag Indicates whether filter channel *n* exceeded its window limit. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not exceeded 1b - Exceeded

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When writing 0b - No effect 1b - Clear the flag
7-4 —	Reserved
3-0 SCDn	Short Circuit Detected Flag Indicates whether SINC detected a short circuit on filter channel *n*. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag

83.8.10 FIFO And CAD Error Interrupt Status (FIFOIS)

Offset

Register	Offset
FIFOIS	20h

Function

Contains flags for interrupts pertaining to FIFO and CAD errors on each SINC channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SAT3	SAT2	SAT1	SAT0	0				CAD3	CAD2	CAD1	CAD0
W					W1C	W1C	W1C	W1C					W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FOVF3	FOVF2	FOVF1	FOVF0	0				FUNF3	FUNF2	FUNF1	FUNF0
W					W1C	W1C	W1C	W1C					W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 SATn	Saturation Flag Indicates whether PF channel *n* is saturated. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not saturated 1b - Saturated When writing 0b - No effect 1b - Clear the flag
23-20 —	Reserved
19-16 CADn	Clock Absence Flag Indicates whether SINC detected the absence of a clock on filter channel *n*. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Clock present 1b - Clock absent When writing 0b - No effect 1b - Clear the flag
15-12 —	Reserved
11-8 FOVFn	FIFO Overflow Flag Indicates whether a FIFO overflow occurred on PF channel *n*. **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Did not occur 1b - Occurred When writing 0b - No effect 1b - Clear the flag
7-4 —	Reserved
3-0 FUNFn	FIFO Underflow Flag Indicates whether a FIFO underflow occurred on PF channel *n*. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not occur 1b - Occurred When writing 0b - No effect 1b - Clear the flag

83.8.11 Status (SR)

Offset

Register	Offset
SR	24h

Function

Presents the status of each SINC channel and modulator clock.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				Reserved	MCLKRDY2	MCLKRDY1	MCLKRDY0	0				FIFOEMP...	FIFOEMP...	FIFOEMP...	FIFOEMP...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				CHRDY3	CHRDY2	CHRDY1	CHRDY0	0				CIP3	CIP2	CIP1	CIP0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 MCLKRDY2	Modulator Clock 2 Ready Indicates the readiness of modulator clock 2. SINC changes this field to 1 when both of the following conditions are true: • MCR[MEN] = 1. • Seven output clock cycles have passed. 0b - Not ready 1b - Ready
25 MCLKRDY1	Modulator Clock 1 Ready Indicates the readiness of modulator clock 1. SINC changes this field to 1 when both of the following conditions are true: • MCR[MEN] = 1. • Seven output clock cycles have passed. 0b - Not ready 1b - Ready
24 MCLKRDY0	Modulator Clock 0 Ready Indicates the readiness of modulator clock 0. SINC changes this field to 1 when both of the following conditions are true:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• MCR[MEN] = 1. • Seven output clock cycles have passed. 0b - Not ready 1b - Ready
23-20 —	Reserved
19-16 FIFOEMPTYn	FIFO Empty Indicates whether the FIFO for filter channel *n* is empty. 0b - Not empty 1b - Empty
15-12 —	Reserved
11-8 CHRDYn	Channel Ready For Conversion Indicates whether filter channel *n* is ready for conversion. SINC changes this field to 1 when all of these conditions are true: • You have enabled a filter. • MCR[MEN] = 1. • C*n*CR[CHEN] = 1. • The FSM of the associated filter accepts the enable signals. • If you are using Manchester decoding, the valid clock edges are synchronized. **NOTE** If you write 0 to C*n*CR[PFEN] to break the ongoing conversion when selecting parallel or serial data, SINC preserves the value of the corresponding CHRDY*n* field for two more cycles. 0b - Not ready 1b - Ready
7-4 —	Reserved
3-0 CIPn	Conversion In Progress Indicates whether a conversion is in progress on PF channel *n*. 0b - Not in progress 1b - In progress

83.8.12 Channel n Control (C0CR - C3CR)

Offset

Register	Offset
C0CR	38h
C1CR	68h
C2CR	98h
C3CR	C8h

Function

Configures various channel features.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0		0		DBGSEL				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	FIFOEN	LMTEN	ZCDEN	0		CADEN	SCDEN	0				DMAEN	0	PFEN	CHEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27-26 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23-20 DBGSEL	Debug Output Selection Selects the instantaneous data that you want to read from Channel n Debug (C0DBGR - C3DBGR). 0000b - Final data from the PF (24 bits) 0001b - Offset data (24 bits) 0010b - Shifted data from the PF (24 bits) 0011b - DC remover (HPF) data (32 bits) 0100b - Raw data from the PF's CIC filter 0110b - Historical data from SCD 0111b - Data from the Manchester decoder 1000b - Data from CAD 1001b - Number of available entries in the FIFO 1010b - Status of the parallel or serial data converter All other values are reserved.
19-16 —	Reserved
15 —	Reserved
14 FIFOEN	FIFO Enable Enables FIFO transfers for the PF. 0b - Disables 1b - Enables
13 LMTEN	Limit Enable Enables a limit check for the PF. 0b - Disables 1b - Enables
12 ZCDEN	Zero Cross Detect Enable Enables zero-cross detection for the PF. 0b - Disables 1b - Enables

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-10 —	Reserved
9 CADEN	Clock Absence Detect Enable Enables the channel's clock-absence detection. 0b - Disables 1b - Enables
8 SCDEN	Short Circuit Detect Enable Enables the channel's short-circuit detection. 0b - Disables 1b - Enables
7-4 —	Reserved
3 DMAEN	DMA Enable Enables primary DMA transfers when the channel's FIFO exceeds its watermark (C*n*CFR[FIFOWMK]). 0b - Disables 1b - Enables
2 —	Reserved
1 PFEN	PF Enable Enables the channel's PF. 0b - Disables 1b - Enables
0 CHEN	Channel Enable Enables the channel and its subfunctions (PF and SCD). SINC enables the channel first and the channel's subfunctions second. Therefore, before you write 1 to CHEN, you must configure other fields as described in Configure fields before enabling channels. 0b - Disables 1b - Enables

83.8.13 Channel n Data Rate (C0DR - C3DR)

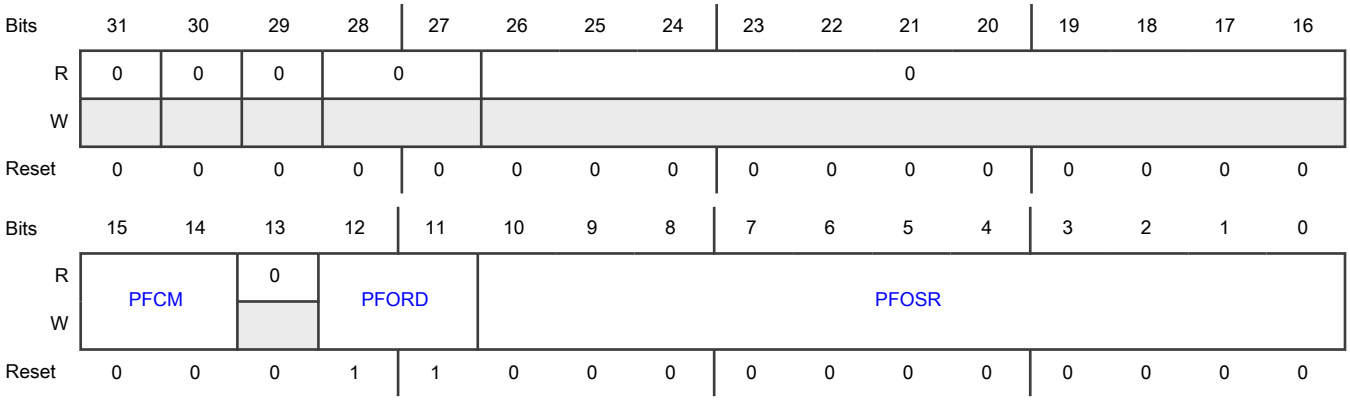
Offset

Register	Offset
C0DR	3Ch
C1DR	6Ch
C2DR	9Ch
C3DR	CCh

Function

Controls functions related to data rate.

Diagram



Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28-27 —	Reserved
26-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>15-14
PFCM</td><td>PF Conversion Mode
Specifies the conversion mode as described in the following table.
<table>
<tr><th>Mode</th><th>Description</th></tr>
<tr><td>Single</td><td>One conversion that follows an edge- or level-triggering event.</td></tr>
<tr><td>Continuous</td><td>Multiple conversions that follow a triggering event. A new triggering event cancels and restarts conversion.</td></tr>
<tr><td>Always</td><td>Multiple conversions that follow the first triggering event. SINC ignores the next triggering event.</td></tr>
<tr><td>Fixed number</td><td>Fixed number conversions that follow the first triggering event. A new triggering event cancels and restarts conversion.</td></tr>
</table>
00b - Single
01b - Continuous
10b - Always
11b - Fixed number</td></tr>
<tr><td>13
—</td><td>Reserved</td></tr>
<tr><td>12-11
PFORD</td><td>PF Order
Defines the primary CIC filter order.
00b - FastSinc
01b - First order
10b - Second order
11b - Third order</td></tr>
<tr><td>10-0
PFOSR</td><td>PF OSR
Controls the PF OSR as described in Equation for OSR.
The minimum permissible value is 3. Lower values produce unpredictable results.
The maximum permissible value depends on PFORD and the desired data format (CnCFR[RDFMT]), as shown in the following table. Higher values produce incorrect results. It is essential to provide proper shifter value to ensure final results are within 24 bits.
<table>
<tr><th>PFORD</th><th>Data format</th><th>Maximum PFOSR</th><th>Minimum right shift bits at maximum PFOSR case</th></tr>
<tr><td rowspan="2">3d</td><td>Signed</td><td>1289</td><td>8</td></tr>
<tr><td>Unsigned</td><td>1624</td><td>8</td></tr>
</table></td></tr>
</table>

Table continued from the previous page...

Field	Function

PFORD	Data format	Maximum PFOSR	Minimum right shift bits at maximum PFOSR case
Any other value	Signed or unsigned	2047	0

83.8.14 Channel n Configuration (C0CFR - C3CFR)

Offset

Register	Offset
C0CFR	40h
C1CFR	70h
C2CFR	A0h
C3CFR	D0h

Function

Configures additional channel features.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ZCOP		0	ITLVL	IBSEL		ITSEL		ICESEL			ICSEL			IBFMT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	FIFOWMK				0		0	RDFMT	0	PFSFT				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 ZCOP	Zero Cross Option Specifies the type of zero crossing that SINC detects. 00b - Both rise and fall

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>01b - Fall
10b - Rise
All other values are reserved.</td></tr>
<tr><td>29
—</td><td>Reserved</td></tr>
<tr><td>28
ITLVL</td><td>Input Trigger Level Type
Specifies the input trigger level type as described in the following table.
<table><tr><th>ITLVL</th><th>What cancels the current conversion</th><th>What happens in Single or Continuous mode</th></tr><tr><td>0</td><td>Next edge event</td><td>Restart</td></tr><tr><td>1</td><td>Deassertion</td><td>Stop filter function</td></tr></table>0b - Edge
1b - Level</td></tr>
<tr><td>27-26
IBSEL</td><td>Input Bit Select
Defines the modulate bit source selection.
00b - External bitstream from the MBIT[n] signal
01b - Alternate internal bitstream from the INP[n] signal
11b - Grouped bitstream shared with an adjacent channel; the adjacent channel's IBSEL field determines the input
All other values are reserved.</td></tr>
<tr><td>25-24
ITSEL</td><td>Input Trigger Select
Defines the filter trigger source selection.
00b - Software
01b - Hardware trigger dedicated to the channel
11b - Grouped trigger shared with an adjacent channel; the adjacent channel's ITSEL field determines the trigger
All other values are reserved.</td></tr>
<tr><td>23-21
ICESEL</td><td>Input Clock Edge Select
Defines the modulate clock edge selection. When selected edge occurs, the modulate bit are sampled for further process.
001b - Positive edge
010b - Negative edge</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - Both edges 100b - Every other odd positive edge 101b - Every other even positive edge 110b - Every other odd negative edge 111b - Every other even negative edge All other values are reserved.
20-18 ICSEL	Input Clock Select Defines the modulate clock source selection. 000b - MCLK_OUT0 with internal routeback 001b - MCLK_OUT1 with internal routeback 010b - MCLK_OUT2 with internal routeback 011b - External modulator clock dedicated to this channel 111b - Grouped clock shared with an adjacent channel; the adjacent channel's ICSEL field determines the input clock All other values are reserved.
17-16 IBFMT	Input Bit Format Defines the input modulate bit format. 00b - External bitstream from the MBIT[n] signal 01b - External Manchester code; ICESEL selects the rise or fall decoder 10b - Internal 16-bit parallel data from MPDATA 11b - Internal 32-bit serial data from MPDATA
15 —	Reserved
14 —	Reserved
13-10 FIFOWMK	FIFO Watermark Specifies the FIFO watermark. When the number of entries in the FIFO exceeds this value, SINC sets the Data Output Ready flag (see NIS[CHF*n*]). If you enable interrupt requests (write 1 to NIE[CHFIE*n*]), SINC also generates an interrupt request. If you enable DMA (write 1 to C*n*CR[DMAEN]), SINC also generates a DMA request.
9-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7 —	Reserved
6 RDFMT	Result Data Format Result data are 24bit width, located at CnRDATA[31:8]. This field controls the signed or unsigned format. If signed format, CnRDATA[31] indicates the negative or positive value. 0b - Left justified, signed 1b - Left justified, unsigned
5 —	Reserved
4-0 PFSFT	PF Shift Shifts the PF data for the correct 24-bit precision. See Shift for details. Bits 0–3 specify the number of bits to shift the data. Bit 4 specifies the shift direction: 0b - Right 1b - Left

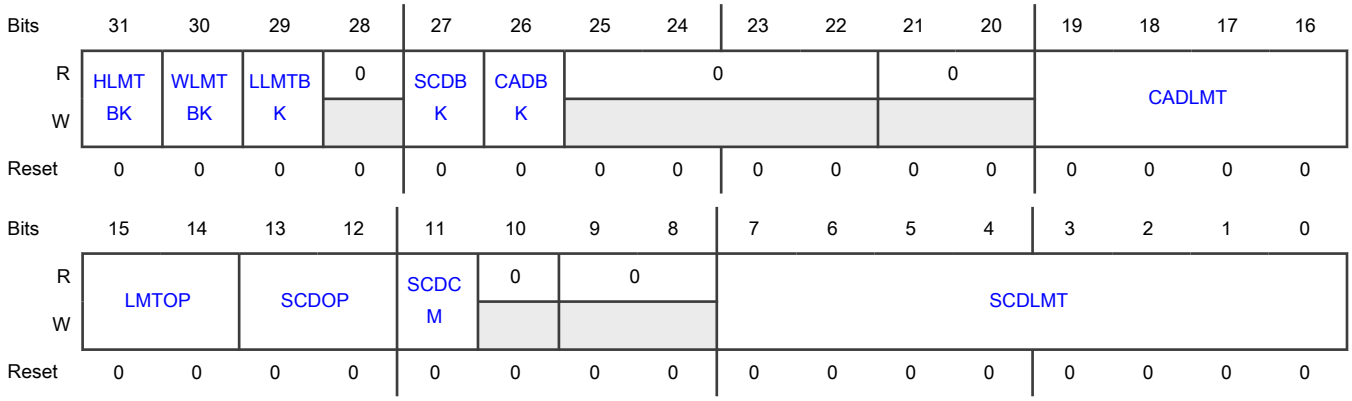
83.8.15 Channel n Protection (C0PROT - C3PROT)

Offset

Register	Offset
C0PROT	44h
C1PROT	74h
C2PROT	A4h
C3PROT	D4h

Function
Configures the SINC functions related to protection (SCD, CAD, and limits).

Diagram



Fields

Field	Function
31 HLMTBK	High Limit Break Signal Enables the automatic assertion of the BREAK_HIGH signal (see External signals) when SINC detects a high-limit event on the associated channel. 0b - Disables 1b - Enables
30 WLMTBK	Window Limit Break Signal Enables the automatic assertion of the BREAK_WIN signal (see External signals) when SINC detects a window-limit event on the associated channel. 0b - Disables 1b - Enables
29 LLMTBK	Low Limit Break Signal Enables the automatic assertion of the BREAK_LOW signal (see External signals) when SINC detects a low-limit event on the associated channel. 0b - Disables 1b - Enables
28 —	Reserved
27 SCDBK	SCD Break Signal Enables the automatic assertion of the BREAK_SCD signal (see External signals) when SINC detects an SCD event on the associated channel. 0b - Disables 1b - Enables
26	CAD Break Signal

Table continues on the next page...

Table continued from the previous page...

Field	Function
CADBK	Enables the automatic assertion of the BREAK_CAD signal (see External signals) when SINC detects a CAD event on the associated channel. 0b - Disables 1b - Enables
25-22 —	Reserved
21-20 —	Reserved
19-16 CADLMT	CAD Limit Threshold Specifies the threshold value for the CAD counter. The CAD counter tracks the number of clock cycles during which SINC does not detect a clock. If that number exceeds CADLMT, a CAD event occurs on the associated channel. You disable CAD by writing 0 to CADLMT. 0000b - Disables CAD All other values - Threshold value
15-14 LMTOP	Limit Detection Option Specifies whether SINC compares the filter sample value to: • The low limit (C*n*LOLMT[LOLMT]) • The high limit (C*n*HILMT[HILMT]) • Both the high and low limits • A window of values between the low and high limits See Limit detection operations for more information. 00b - Both high and low limits 01b - High limit 10b - Low limit 11b - Windowed value
13-12 SCDOP	SCD Option Specifies which repeating bit value increments the SCD counter. 00b - Both 0 and 1 01b - Only 1 10b - Only 0 11b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 SCDCM	SCD Conversion Mode Specifies when SCD operates. 0b - Constantly when CnCR[CHEN] = MCR[MEN] = 1 1b - Only when the PF is performing a conversion
10 —	Reserved
9-8 —	Reserved
7-0 SCDLMT	SCD Limit Threshold Specifies the threshold value for the SCD counter. The SCD counter tracks the number of received bits with the same repeating value (always 0 or always 1). If that number exceeds SCDLMT, an SCD event occurs on the associated channel. 0000_0000b-0000_0001b - Disables SCD All other values - Threshold value

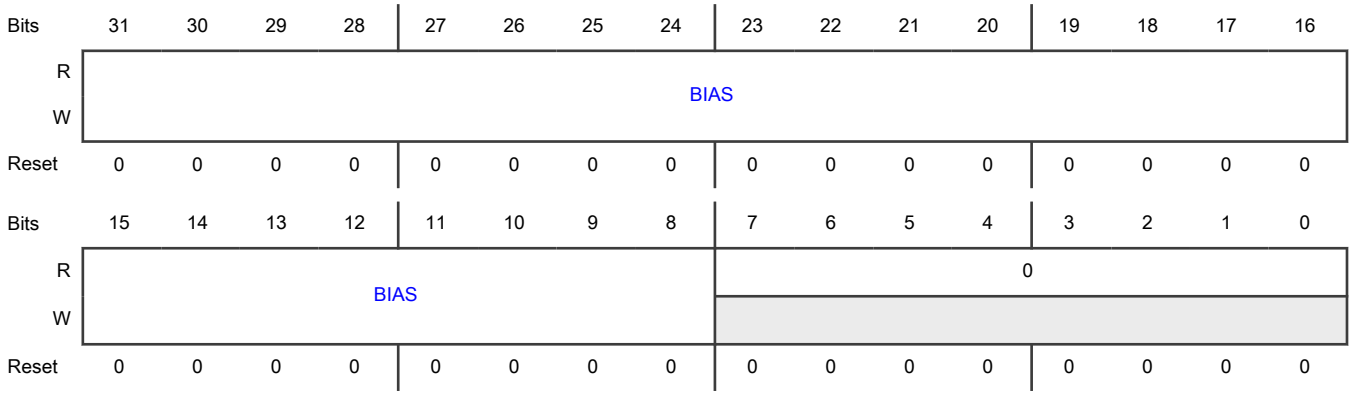
83.8.16 Channel n Bias (C0BIAS - C3BIAS)

Offset

Register	Offset
C0BIAS	48h
C1BIAS	78h
C2BIAS	A8h
C3BIAS	D8h

Function
Configures the filter bias.

Diagram



Fields

Field	Function
31-8 BIAS	Bias Value Specifies the bias offset for this channel's PF. After the PF shifts its data, the filter subtracts this bias value from the data. Bit 23 of this field (bit 31 within the register) specifies the bias sign: 0b - Positive 1b - Negative
7-0 —	Reserved

83.8.17 Channel n Low Limit (C0LOLMT - C3LOLMT)

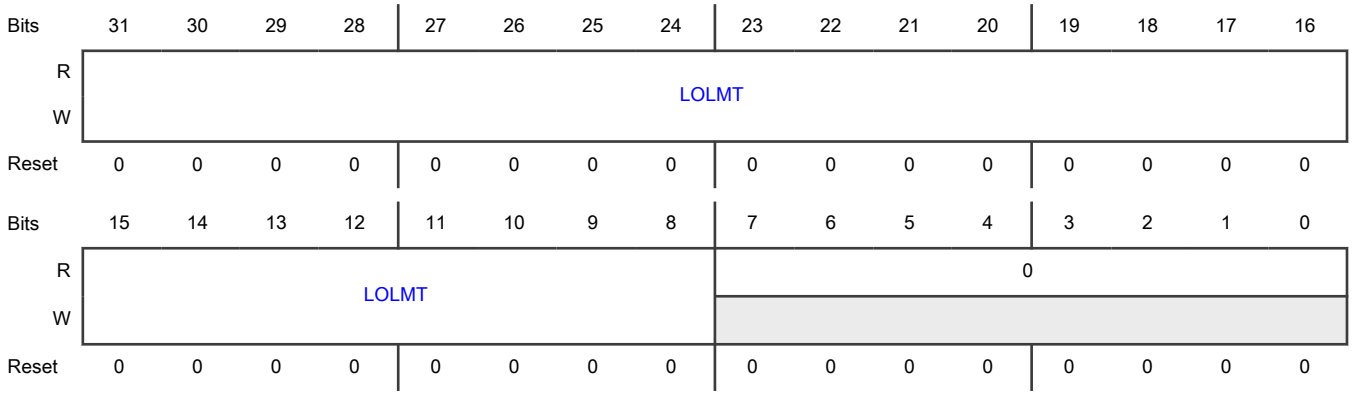
Offset

Register	Offset
C0LOLMT	4Ch
C1LOLMT	7Ch
C2LOLMT	ACh
C3LOLMT	DCh

Function

Specifies the low-limit threshold value.

Diagram



Fields

Field	Function
31-8 LOLMT	Low Limit Threshold Specifies the low-limit threshold value. When the data exceeds this threshold value, a low-limit event occurs. C*n*CFR[RDFMT] determines the limit threshold format (signed or unsigned). **NOTE** Ensure that LOLMT is lower than C*n*HILMT[HILMT]. Otherwise, the low-limit threshold does not work.
7-0 —	Reserved

83.8.18 Channel n High Limit (C0HILMT - C3HILMT)

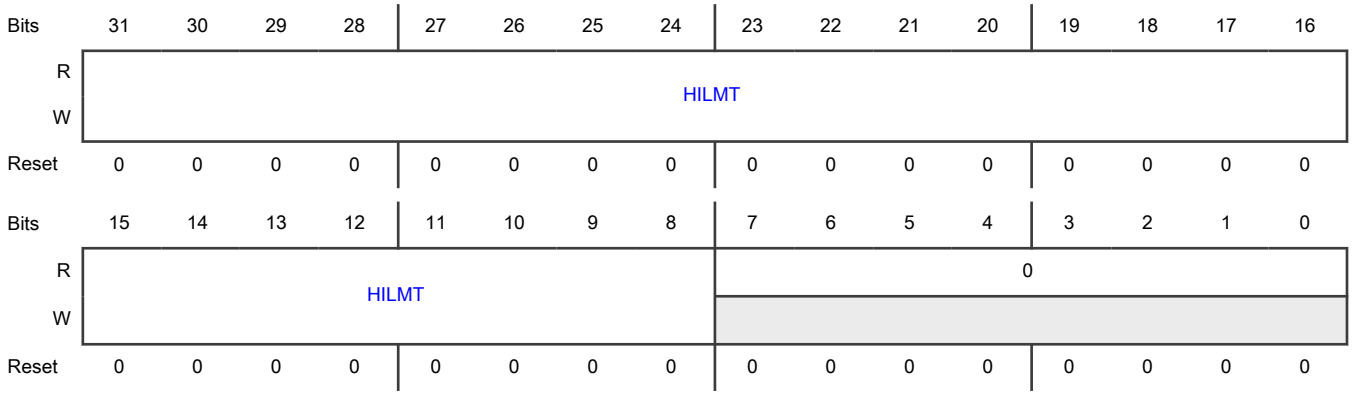
Offset

Register	Offset
C0HILMT	50h
C1HILMT	80h
C2HILMT	B0h
C3HILMT	E0h

Function

Specifies the high-limit threshold value.

Diagram



Fields

Field	Function
31-8 HILMT	High Limit Threshold Specifies the high-limit threshold value. When the data exceeds this threshold value, a high-limit event occurs. C*n*CFR[RDFMT] determines the limit threshold format (signed or unsigned). **NOTE** Ensure that C*n*LOLMT[LOLMT] is lower than HILMT. Otherwise, the low-limit threshold does not work.
7-0 —	Reserved

83.8.19 Channel n Result Data (C0RDATA - C3RDATA)

Offset

Register	Offset
C0RDATA	54h
C1RDATA	84h
C2RDATA	B4h
C3RDATA	E4h

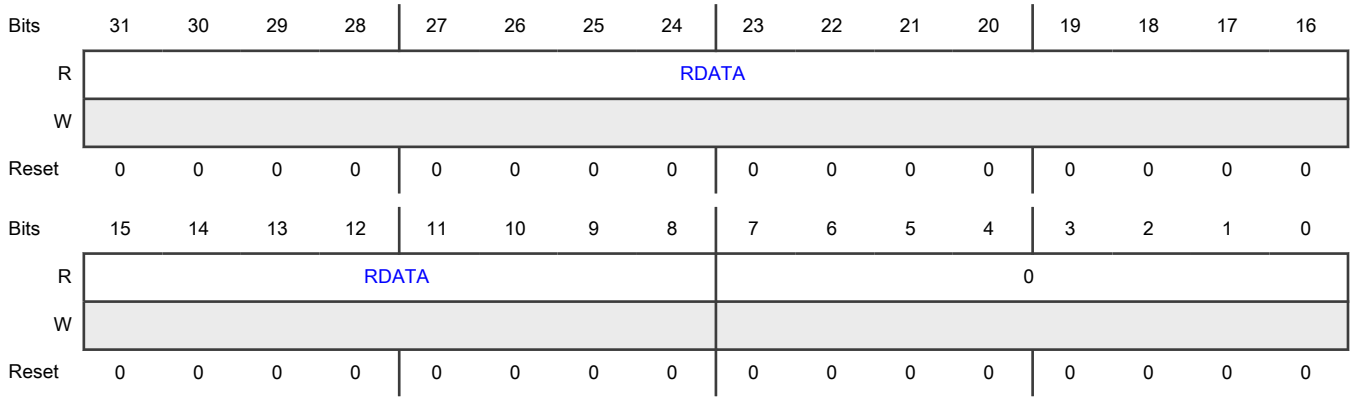
Function

Returns data from the FIFO.

NOTE

NXP recommends that you use a 32-bit access to read this register. If you use a byte access to read this register, you must read the highest byte last, and this increases the FIFO level.

Diagram



Fields

Field	Function
31-8 RDATA	Result Data Contains data from the FIFO. Bit 23 of this field (bit 31 within the register) serves as the sign bit if you specify the signed-data format (C*n*CFR[RDFMT] = 0). When you read this field, be aware of the following circumstances:
7-0 —	Reserved

Conditions when you read RDATA	Result
SINC has not yet completed the conversion	One of the following occurs: • RDATA contains all zeros. • SINC triggers a FIFO underflow error.
FIFO is disabled (C*n*CR[FIFOEN] = 0)	RDATA contains the last conversion result. You must poll the C*n*SR[SRDS] flag, and if that flag is set after you read RDATA, you must read RDATA again.

83.8.20 Channel n Multipurpose Data (C0MPDATA - C3MPDATA)

Offset

Register	Offset
C0MPDATA	58h
C1MPDATA	88h
C2MPDATA	B8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
C3MPDATA	E8h

Function

Contains multipurpose data as specified in C*n*CFR[IBFMT].

You can use this register for a parallel bitstream, a serial bitstream, or a Manchester decoder threshold as described in the following table.

Table 1155. MPDATA content options

Data type	C*n*CFR[IBFMT]	Description
Manchester	01b	Bits 10–0 contain the Manchester decoder threshold value.
Parallel	10b	Bits 15–0 contain the parallel 16-bit data. You can write to MPDATA only when parallel or serial data is ready (C*n*SR[PSRDY] = 1). Otherwise, a transfer error occurs.
Serial	11b	Bits 31–0 contain the serial data. You can write to MPDATA only when parallel or serial data is ready (C*n*SR[PSRDY] = 1) or MPDATA is not completely full. Otherwise, a transfer error occurs. **NOTE** Allow write MPDATA when C*n*SR[PSRDY] = 0, give you the chance if keep the remain bit for next conversion or overwrite with new data. In this case, the write are only allowed one time. **NOTE** MPDATA are shift out,you can read back for debug. But in corner case, last bit does not shift out while remain bit are zero.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MPDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MPDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MPDATA	Multipurpose Data Contains multipurpose data as described in Table 1155. **NOTE** if used as Serial Data, MCLKDIV at least divider by 4. And make sure MCLKOUT[0] period bigger than 2x bus clock period plus 2x func clock period

83.8.21 Channel n Advanced Configuration (C0ACFR - C3ACFR)

Offset

Register	Offset
C0ACFR	5Ch
C1ACFR	8Ch
C2ACFR	BCh
C3ACFR	ECh

Function
Configures the SINC functional subblocks.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				PTMUX				IBDLY				HPFA			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADMASEL				0	0										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-24 PTMUX	Pulse Trigger Mux Select Selects the signal for pulse-trigger output. For more information, see External signals, High-low limit signals to pulse trigger, and ZCD output signals to pulse trigger . 0_0000b - Disabled; outputs 0 0_0001b - Asserts H_LIM_OUT 0_0010b - Asserts L_LIM_OUT 0_0011b - Asserts LIM_OUT 0_0100b - Asserts W_LIM_OUT 0_0101b - Asserts ZC_OUT 0_0110b - Asserts ZC_OUT_inv 0_0111b - Asserts RS_LIM_OUT 0_1000b - Asserts RS_LIM_OUT_inv 0_1001b - Channel raw input modulator bitstream 0_1010b - Channel raw input modulator clock 0_1011b - Channel output recovered modulator bitstream 0_1100b - Channel output recovered modulator clock 0_1101b - Asserts H_LIM_TRG 0_1110b - Asserts L_LIM_TRG 0_1111b - Asserts LIM_TRG 1_0000b - Asserts W_LIM_TRG 1_0001b - Asserts HL_LIM_TRG 1_0010b - Zero cross rise pulse signal 1_0011b - Zero cross fall pulse signal 1_0100b - Zero cross rise and fall pulse signal 1_0101b - FIFO watermark ok pulse signal 1_0110b - FIFO overflow pulse signal 1_0111b - FIFO underflow pulse signal 1_1000b - FIFO empty pulse signal 1_1001b - Clock monitor assert pulse signal 1_1010b - Short circuit assert pulse signal 1_1011b - Saturation pulse signal 1_1100b - Conversion complete pulse signal All other values are reserved.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>23-20
IBDLY</td><td>Input Modulator Bitstream Delay
Specifies the delay, in prescale clock cycles, for sampling modulator bitstreams after an EMCLK edge.
A value of 0 disables the delay and SINC uses EMCLK to sample EMBIT.
NOTE
When you choose the delay, ensure that it does not exceed the next EMCLK edge. For example, if the EMCLK width is 13 cycles, the delay value cannot exceed 12d. Values greater than the next EMCLK edge produce unpredictable results.
0000b - Disabled
All other values - Delay in clock cycles</td></tr>
<tr><td>19-16
HPFA</td><td>HPF DC Remover Alpha Coefficient
Specifies the HPF alpha coefficient or disables HPF as described in the following table.
<table><tr><th>HPFA</th><th>Result</th></tr><tr><td>0</td><td>Disable HPF</td></tr><tr><td>1</td><td>Coefficient = $1 - 2^{-5}$</td></tr><tr><td>2</td><td>Coefficient = $1 - 2^{-6}$</td></tr><tr><td>3</td><td>Coefficient = $1 - 2^{-7}$</td></tr><tr><td>4</td><td>Coefficient = $1 - 2^{-8}$</td></tr><tr><td>...</td><td></td></tr><tr><td>m</td><td>Coefficient = $1 - 2^{-4-m}$</td></tr></table></td></tr>
<tr><td>15-12
ADMASEL</td><td>Alternate DMA Source Selection
Selects the trigger source for alternate DMA.
0000b - Alternate DMA disabled
0001b - PF conversion complete
0010b - PF data output ready
0011b - Zero crossing detected
0100b - Short circuit detected
0101b - Window limit detected
0110b - Low limit detected
0111b - High limit
1000b - FIFO underflow
1001b - FIFO overflow</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1010b - Clock absence 1011b - Saturation
11 —	Reserved
10-0 —	Reserved

83.8.22 Channel n Status (C0SR - C3SR)

Offset

Register	Offset
C0SR	60h
C1SR	90h
C2SR	C0h
C3SR	F0h

Function
Provides channel-status information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0		0		CNUM_OV	CNUM						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DBGRS		SRDS	RDRS	BIASSAT	SFTSAT	HPFSAT	PFSAT	PSRDY	0			FIFOAVIL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31-28
—</td><td>Reserved</td></tr>
<tr><td>27-26
—</td><td>Reserved</td></tr>
<tr><td>25-24
—</td><td>Reserved</td></tr>
<tr><td>23
CNUM_OV</td><td>Overflow In Number Of Conversions
Indicates when the number of conversions exceeds 127. When this occurs, SINC changes CNUM_OV to 1. After resynchronization, SINC changes CNUM_OV to 0.
0b - No overflow
1b - Overflow</td></tr>
<tr><td>22-16
CNUM</td><td>Number Of Conversions
In coordination with CNUM_OV, specifies the number of conversions:
<table><tr><th>Mode</th><th>Event</th></tr><tr><td>Continuous</td><td>After the start of a trigger</td></tr><tr><td>Single</td><td>After the enablement of conversion</td></tr></table>
SINC counts the number of conversions internally and modifies CNUM and CNUM_OV as follows:
<table><tr><th>When the number of conversions is:</th><th>SINC does this:</th></tr><tr><td>127 or fewer</td><td>Stores the number of conversions in CNUM</td></tr><tr><td>128 or more</td><td>• Stores [(number of conversions) mod 128] in CNUM
• Changes CNUM_OV to 1</td></tr></table></td></tr>
<tr><td>15-14
DBGRS</td><td>Debug Data Read Status
Indicates whether the debug data is valid.
When this value is 0, the CIC, HPF, shift, and offset outputs are all correct.
00b - Valid
01b-11b - Invalid</td></tr>
<tr><td>13
SRDS</td><td>Start Read Debug Data Sync
Starts the debug data latch procedure and reports the procedure status.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When you write 1 to this field, SINC starts the procedure and changes this field to 0. When the field becomes 0 again, you can safely read the debug data (Channel n Debug (C0DBGR - C3DBGR)). **NOTE** This field behaves differently for register reads and writes. When reading 0b - Data valid 1b - Procedure in progress When writing 0b - No effect 1b - Starts the procedure
12 RDRS	Result Data Direct Read Status When you disable the FIFO to read channel data from it, indicates that the data in Channel n Result Data (C0RDATA - C3RDATA) is stable and valid. Before this value is 1, the data may not be stable. Therefore, to obtain stable data, do one of the following: • Wait until the conversion completes, then read C*n*RDATA. • When you read data, ensure that RDRS = 0. 0b - Valid 1b - Invalid
11 BIASSAT	Bias Saturation Flag Indicates whether bias saturation occurred. 0b - Did not occur 1b - Occurred
10 SFTSAT	Shift Saturation Flag Indicates whether shift saturation occurred. 0b - Did not occur 1b - Occurred
9 HPFSAT	HPF Saturation Flag Indicates whether HPF saturation occurred. 0b - Did not occur 1b - Occurred
8 PFSAT	Primary CIC Saturation Flag Indicates whether primary CIC filter saturation occurred.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Did not occur 1b - Occurred
7 PSRDY	Parallel or Serial Data Ready Indicates whether Channel n Multipurpose Data (C0MPDATA - C3MPDATA) is ready for you to write parallel or serial data. 0b - Not ready 1b - Ready
6-5 —	Reserved
4-0 FIFOAVIL	FIFO Available Data Indicates the number of remaining data entries in the FIFO.

83.8.23 Channel n Debug (C0DBGR - C3DBGR)

Offset

Register	Offset
C0DBGR	64h
C1DBGR	94h
C2DBGR	C4h
C3DBGR	F4h

Function

Contains the instantaneous debug data and status for debugging each channel's subblocks.

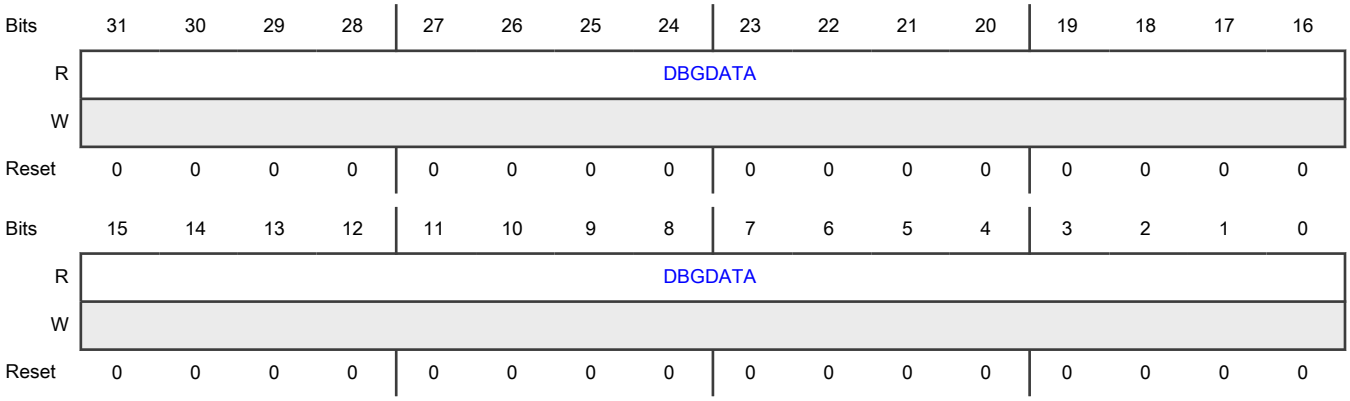
You select the data by writing to C*n*CR[DBGSEL], as shown in Details of debug data. For example:

- If you write 3d to C*n*CR[DBGSEL], SINC offers you 32 bits of data from the DC remover (HPF).
- If you write 4d to C*n*CR[DBGSEL], SINC offers you raw data from the PF's CIC filter.

The instantaneous data is read-only, and the contents are for each subblock.

See Reading asynchronous data safely and Transferring data between domains safely for the procedures to manage data safely.

Diagram



Fields

Field	Function
31-0 DBGDATA	Debug Data Returns the debug data requested in C*n*CR[DBGSEL]. Details of debug data shows the structure of the debug data.

Chapter 84
Voltage Reference (VREF)

84.1 Chip-specific VREF information

Table 1156. Reference links to related information

Topic	Related module	Reference
System memory map	-	System memory map
Clocking	CCM ANADIG - XTALOSC, PLL	Clock Generation Overview Clock Control Module (CCM) ANADIG - Crystal Oscillator (XTALOSC) ANADIG - Phase Locked Loop (PLL)
Power management	ANADIG - PMU	Power management ANADIG - Power Management Unit (PMU)
Signal multiplexing	IOMUXC BLK_CTRL_AON BLK_CTRL_WAKEUP	External signals and pin multiplexing IOMUX Controller (IOMUXC) Block Control, AON Domain (BLK_CTRL_AON) Block Control, Wakeup Domain (BLK_CTRL_WAKEUP)
Interrupts, DMA Events and XBAR Assignments	-	Interrupts, DMA Events and XBAR Assignments

NOTE

VREF_OUT shares same pin ADC_VREFH. In case no external VREFH is available, VREF_OUT can be used instead to save BOM cost.

84.2 Overview

The Voltage Reference provides a buffered reference voltage for use as an external reference, which can be set to 1.2 V. In addition, the buffered reference is available internally for use with on-chip peripherals (such as ADCs and DACs) . When the VREF is enabled, the reference voltage is placed on a dedicated output pin.

- The Voltage Reference output can be fine-tuning, trimmed with 0.5×(4/3) mV resolution, using the UTRIM register VREFTRIM[5:0] bitfield.

84.2.1 Block diagram

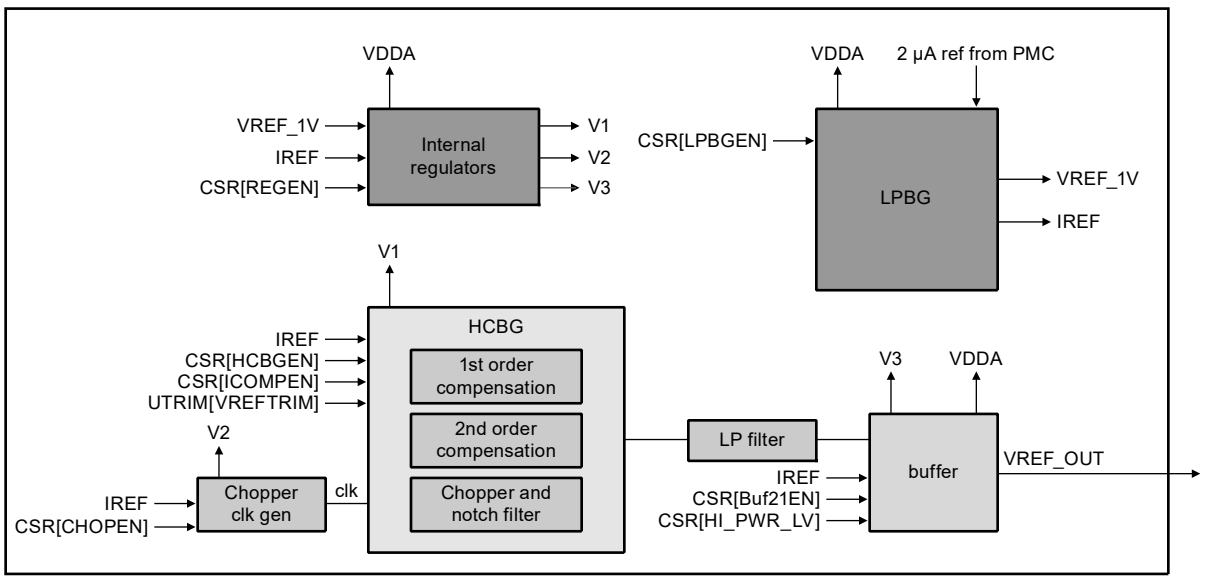


Figure 676. Voltage reference block diagram

84.2.2 Features

- Programmable trim register with 0.5×(4/3) mV steps in fine-tuning; Upon reset, the trim register is automatically loaded with a factory-trimmed value.
- Programmable buffer mode selection:
  - Off
  - Bandgap enabled/standby (output buffer disabled)
  - Low power buffer mode (output buffer enabled)
  - High power buffer mode (output buffer enabled)
- Low-Power bandgap and High-Accurate bandgap selection
- Dedicated output pin, VREF_OUT

84.3 Functional description

The Voltage Reference is a bandgap buffer system. Unity gain amplifiers are used.

The VREF_OUT signal can be used by both internal and external peripherals in low and high power buffer mode. A 220 nF capacitor must always be connected between VREF_OUT and VSS if the VREF is being used.

When CSR[LPBGEN] = 1 and CSR[HCBGEN] = 1, the Voltage Reference is enabled and different modes should be set by the bits CSR[Buf21EN] and CSR[HI_PWR_LV].

The Voltage Reference is an accurate buffered voltage 1.2 V.

84.3.1 Voltage Reference Disabled

When (CSR[LPBGEN] = 0) and (CSR[HCBGEN] = 0) and (CSR[Buf21EN] = 0), the Voltage Reference is disabled, the VREF bandgap and the output buffers are disabled. The Voltage Reference is in off mode.

84.3.2 Voltage Reference Enabled

When Voltage Reference Enabled, we have following cases:

1. Enable the CSR[LPBGEN]
2. Enable (CSR[LPBGEN]=1), (CSR[REGEN]=1), (CSR[HCBGEN]=1) and (CSR[Buf21EN]=1), this configuration enable VREF supply voltage on VREF_OUT

84.3.2.1 CSR[Buf21EN]=0,CSR[HI_PWR_LV]=X

The internal VREF bandgap is enabled to generate an accurate 0.9 V output that can be trimmed with the UTRM register's UTRIM[13:8] bitfield. The bandgap requires some time for startup and stabilization. CSR[VREFST] can be monitored to determine if the stabilization and startup is complete when the chop oscillator is not enabled.

If the chop oscillator is being used, the internal bandgap reference voltage settles within 400µs (see T_{STUP} in the device data sheet).

The output buffer is disabled in this mode, and there is no buffered voltage output. The Voltage Reference is in standby mode. If this mode is first selected and the low power or high power buffer mode is subsequently enabled, there will be a delay of 400µs before the buffer output is settled at the final value.

84.3.2.2 CSR[Buf21EN]=1,CSR[HI_PWR_LV]=1

The internal VREF bandgap is on. The high power buffer is enabled to generate a buffered voltage to VREF_OUT which is 1.2 V. The high power buffer has higher bandwidth than low power buffer. It can also be used as a reference to internal analog peripherals such as an ADC, VREF output pad and ADC VREFH are bonding to same ball in RT1180.

If this mode is entered from the standby mode (CSR[Buf21EN]=0) there will be a delay of 400µs before the buffer output is settled at the final value. If this mode is entered when the VREF module is enabled then you must wait for 400µs or until CSR[VREFST] = 1 when the chop oscillator is not enabled. If the chop oscillator is being used, you must wait for 400µs to ensure the VREF output has stabilized.

In this mode, a 220 nF capacitor is required to connect between the VREF_OUT pin and VSS.

84.3.2.3 CSR[Buf21EN]=1,CSR[HI_PWR_LV]=0

The internal VREF bandgap is on. The low power buffer is enabled to generate a buffered voltage to VREF_OUT which is 1.2 V. It can also be used as a reference to internal analog peripherals such as an ADC: VREF output pad and ADC VREFH pad are bonding to the same pin in the chip.

If this mode is entered from the standby mode (CSR[Buf21EN]=0), there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (T_{STUP}) and the value is specified in the appropriate device data sheet. If this mode is entered when the VREF module is enabled, then you must wait for 400µs or until CSR[VREFST] = 1 when the chop oscillator is not enabled. If the chop oscillator is being used, you must wait for 400µs to ensure the VREF output has stabilized.

In this mode, a 220 nF capacitor is required to connect between the VREF_OUT pin and VSS.

84.3.3 Internal voltage regulator

The VREF module contains an internal voltage regulator that can be enabled to provide additional supply noise rejection and internal oscillator which provide the chop clock. It is recommended that when possible, this regulator and oscillator can be enabled to provide the optimum VREF performance.

If the chop function is being used, and internal voltage regulator must also be enabled. A specific sequence must be followed when enabling the internal regulator as followings:

1. Enable (CSR[LPBGEN]=1)
2. Enable the internal regulator by setting (CSR[REGEN]=1)
3. Enable the chop oscillator(CSR[CHOPEN]=1), (CSR[ICOMPEN]=1) and (CSR[HCBGEN]=1)

84.3.4 Clocking

This module has no clocking considerations.

84.3.5 Interrupts

This module has no interrupts.

84.4 External signals

The following table shows the Voltage Reference signals properties.

Table 1157. VREF Signal Descriptions

Signal	Description	I/O
VREF_OUT	Internally-generated Voltage Reference output	O

NOTE

When the VREF output buffer is disabled, the status of the VREF_OUT signal is high-impedance.

84.5 Initialization

The Voltage Reference requires some time for startup and stabilization. After CSR[LPBGEN] = 1 and CSR[HCBGEN] = 1, CSR[VREFST] can be monitored to determine if the stabilization and startup is completed when the chop oscillator is not enabled. When the chop oscillator is enabled, the settling time of the internal bandgap reference is defined by Tstup (400µs). You must wait for 400µs after the internal bandgap has been enabled to ensure the VREF internal reference voltage has stabilized.

When the Voltage Reference is already enabled and stabilized, changing CSR[HI_PWR_LV] will not clear CSR[VREFST] but there will be some startup time before the output voltage at the VREF_OUT pin has settled. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet. Also, there will be some settling time when a step change of the load current is applied to the VREF_OUT pin. When the 1.75 V VREF regulator is disabled, the VREF_OUT voltage will be more sensitive to supply voltage variation. It is recommended to use this regulator to achieve optimum VREF_OUT performance.

The CSR[CHOPEN], CSR[REGEN] and CSR[ICOMPEN] bits must be written to 1 to achieve the performance stated in the device data sheet.

NOTE

See section "Internal voltage regulator" for details on the required sequence to enable the internal regulator.

84.6 Memory Map and Register Definition

84.6.1 VREF register descriptions

84.6.1.1 VREF memory map

VREF base address: 42E3_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0000h
8h	Control and Status Register (CSR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10h	User Trim (UTRIM)	32	RW	0000_0000h
1Ch	Test Unlock (TEST_UNLOCK)	32	RW	0000_0000h
24h	Test Trim 0 (TRIM0)	32	RW	0000_0000h

84.6.1.2 Version ID (VERID)

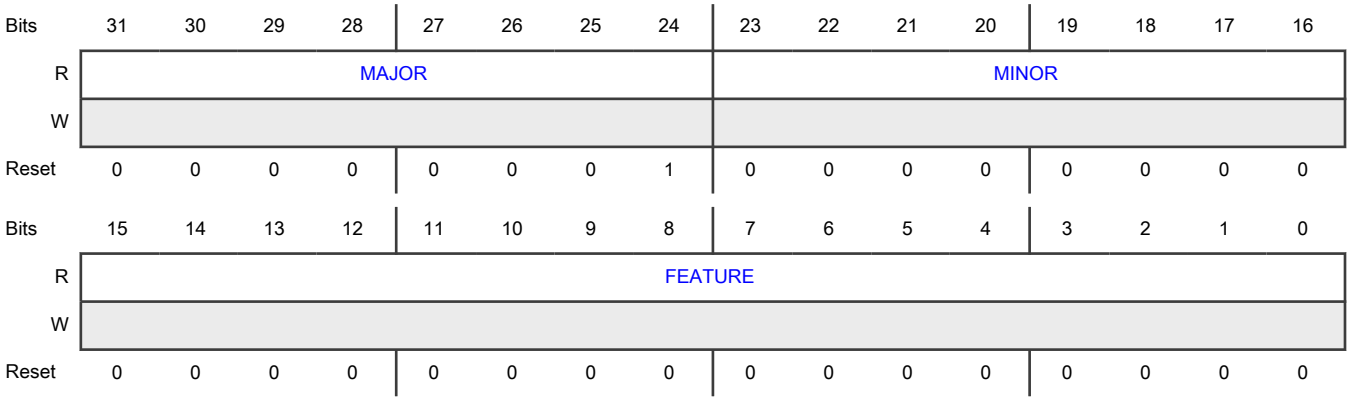
Offset

Register	Offset
VERID	0h

Function

This register contains version/feature number for VREF is read only.

Diagram



Fields

Field	Function
31-24 MAJOR	MAJOR Major Version Number: This read only field returns the major version number for the module specification.
23-16 MINOR	MINOR Minor Version Number. This read only field returns the minor version number for the module specification.
15-0 FEATURE	FEATURE Feature Specification Number. This read only field returns the feature set number.

84.6.1.3 Control and Status Register (CSR)

Offset

Register	Offset
CSR	8h

Function

This register contains the control and status bits of VREF

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VREFST	0														Buf21EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				HI_PWR_...	0	0		0	0	REGEN	ICOMPEN	CHOPEN	0	LPBGEN	HCBGEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VREFST	Internal High Accuracy Voltage Reference stable Indicates that the bandgap reference within the Voltage Reference module has finished its startup and stabilization. **NOTE** VREFST bit is valid only when the chop oscillator is not being used. 0b - The module is disabled or not stable. 1b - The module is stable.
30-17 —	Reserved
16 Buf21EN	Internal buffer enable Output voltage is 1.2 V 0b - buffer is disabled 1b - buffer is enabled
15-12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11 HI_PWR_LV	Buffer mode control 0b - buffer is in low power mode 1b - buffer is in high power mode
10 —	Reserved
9-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 REGEN	Regulator enable Enables the internal 1.75 V regulator to produce a constant internal voltage supply in order to reduce the sensitivity to external supply noise and variation. REGEN bit should be written to 1 to achieve the performance stated in the data sheet. **NOTE** See section "Internal voltage regulator" for details on the required sequence to enable the internal regulator. 0b - Internal 1.75 V regulator is disabled. 1b - Internal 1.75 V regulator is enabled.
4 ICOMPEN	Second order curvature compensation enable ICOMPEN bit should be written to 1 to achieve the performance stated in the data sheet. 0b - Disabled 1b - Enabled
3 CHOPEN	Chop oscillator enable. When set, the internal chopping operation is enabled and the internal analog offset will be minimized. If the internal voltage regulator is being used (REGEN bit is set to 1), then the chop oscillator must also be enabled. If the chop oscillator will be used in very low power modes, then the High Accuracy bandgap voltage reference must also be enabled. See the chip-specific VREF information (also known as "chip configuration" details) to see how this can be achieved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Chop oscillator is disabled. 1b - Chop oscillator is enabled.
2 —	Reserved
1 LPBGEN	Low Power Bandgap enable This bit is used to enable the Low Power Bandgap. 0b - LP Bandgap is disabled 1b - LP Bandgap is enabled
0 HCBGEN	High Accuracy Bandgap enabled This bit is used to enable the High Accuracy Bandgap. **NOTE** This bit can only be enabled when LPBGEN =1. 0b - HC Bandgap is disabled 1b - HC Bandgap is enabled

84.6.1.4 User Trim (UTRIM)

Offset

Register	Offset
UTRIM	10h

Function

This register contains the trim data can be read/write in user mode

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		VREFTRIM						0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved
13-8 VREFTRIM	VREF Trim bits These bits change the resulting VREF output by 0.5×(4/3) mV for each step. The trim value is stored in eFuse . 00_0000b - default–32×0.5×(4/3) mV 10_0000b - default 11_1111b - default+31×0.5×(4/3) mV All other values - see the value options above as example for calculation
7-4 —	Reserved
3-0 —	Reserved

84.6.1.5 Test Unlock (TEST_UNLOCK)

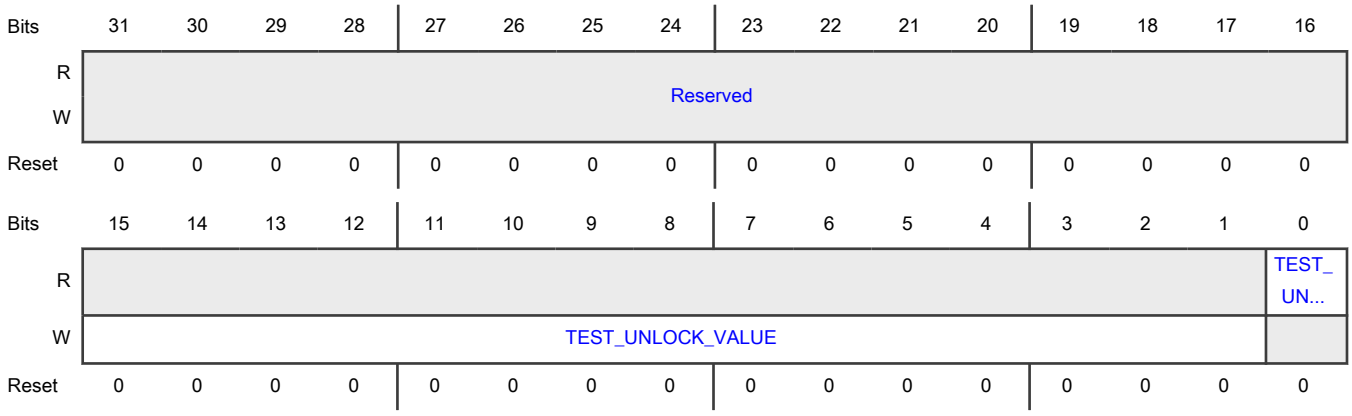
Offset

Register	Offset
TEST_UNLOCK	1Ch

Function

Unlocks read/write into test register. After reset, software cannot read/write into test register. After software writes into bit[15:1] value 0x5AA5, the read/write is unlocked.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-1 TEST_UNLOCK_VALUE	Test unlock value After reset, software cannot read/write into test register. If software writes value 0x5AA5 into this field, the read/write will be unlocked.
0 TEST_UNLOCK	Test_unlock status bit This bit will be set "1", if software writes 0x5AA5 into TEST_UNLOCK_VALUE field. 0b - Lock read/write into test register 1b - Unlock read/write into test register

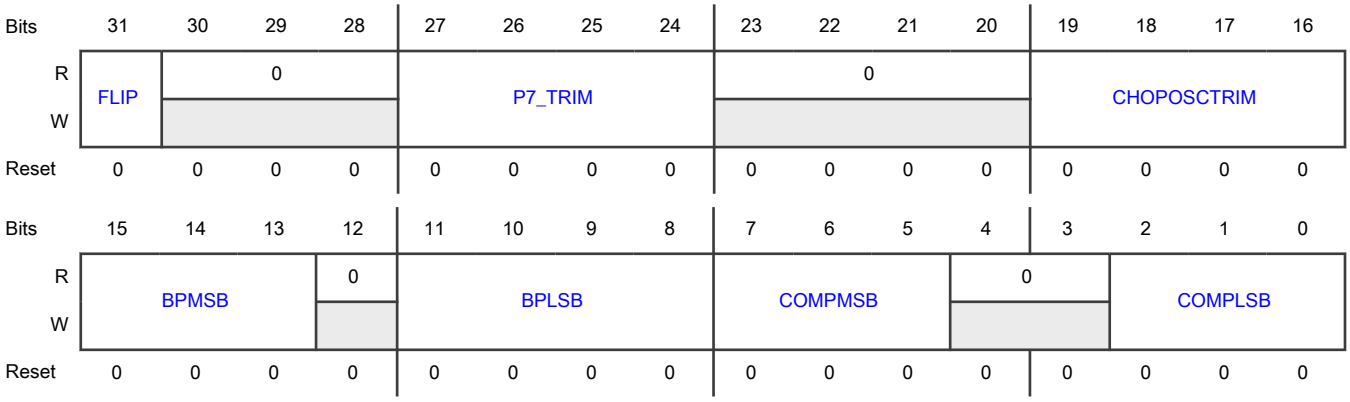
84.6.1.6 Test Trim 0 (TRIM0)

Offset

Register	Offset
TRIM0	24h

Function
This register contains the trim values read/write in functional test mode.

Diagram



Fields

Field	Function
31	Amplifier Polarity

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLIP	Reverses the amplifier polarity. FLIP can be used to test bandgap amplifier offset. The trim value is stored in IFR .
30-28 —	RESERVED
27-24 P7_TRIM	P7_TRIM P7_TRIM bits are used for trimming the 0.7V for V to I at 3 mV/bit. The trim value is stored in IFR .
23-20 —	RESERVED
19-16 CHOPOSCTRIM	CHOPOSCTRIM CHOPOSCTRM bits trim the chop oscillator clock. The center frequency of the chop oscillator is 250 kHz at the trim value of 0b1000. The trim step size is 5% time domain when chop oscillator is enabled.
15-13 BPMSB	BPMSB BPMSB bits are used for thin trimming HC bandgap temperature curvature. BPMSB bits are active low and will shift the bandgap inflection point ~15 °C/bit. The trim value is stored in IFR .
12 —	RESERVED
11-8 BPLSB	BPLSB BPLSB bits are used for thin trimming HC bandgap temperature curvature. BPLSB bits are active low and will shift the bandgap inflection point ~2 °C/bit. The trim value is stored in IFR .
7-5 COMPMSB	COMPMSB COMPMSB bits are used for trimming the curvature used for the second-order temperature compensation. The trim value is stored in IFR .
4-3 —	RESERVED
2-0 COMPLSB	COMPLSB COMPLSB bits are used for trimming the curvature used for the second-order temperature compensation. The trim value is stored in IFR .

Appendix A
General Changes

A.1 Release Notes for Rev. 6

- Initial public release

Appendix B
Release Notes

B.1 Revision history

Because this is the initial release of the document, there are no changes.

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile — are trademarks and/or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

EdgeLock — is a trademark of NXP B.V.

I2C-bus — logo is a trademark of NXP B.V.

For more information, please visit: https://www.nxp.com

Date of release: 09/2024

Document identifier: IMXRT1180RM